UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERIA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

DISEÑO DE UN PROTOTIPO QUE UNIFIQUE, UN SISTEMA DE PAGO BASADO EN RECARGAS, UN SISTEMA DE CONTROL DE INVENTARIO Y UN SISTEMA DE CONTROL DE ACCESO, MEDIANTE LA TECNOLOGÍA NEAR FIELD CONNECTION (NFC) PARA LA ASOCIACIÓN DE ESTUDIANTES CITEL-CIERCOM.

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TITULO DE INGENIERO EN TELECOMUINICACIONES

AUTOR: DARWIN STEVEN FLORES RUANO

DIRECTOR: MSC. JAIME ROBERTO MICHILENA CALDERÓN

Ibarra-Ecuador

2023

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	0450088166		
APELLIDOS Y NOMBRES:	Flores Ruano Darwin Steven		
DIRECCIÓN:	La Paz- Antonino Narváez y Simón Bolívar		
EMAIL:	dsfloresr@utn.edu.ec		
TELÉFONO FIJO:		**TELÉFONO MÓVIL:**	0999173023
DATOS DE LA OBRA			
TÍTULO:	Diseño de un prototipo que unifique, un sistema de pago basado en recargas, un sistema de control de inventario y un sistema de control de acceso, mediante la tecnología near field connection (nfc) para la asociación de estudiantes citel-ciercom.		
AUTOR (ES):	Flores Ruano Darwin Steven		
FECHA: DD/MM/AAAA	31/05/2023		
SOLO PARA TRABAJOS DE GRADO			
PROGRAMA:	X **PREGRADO** ☐ **POSGRADO**		
TITULO POR EL QUE OPTA:	Ingeniero en Telecomunicaciones		
ASESOR /DIRECTOR:	**MSC. Jaime Michilena**		

2. CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 31 días del mes de mayo de 2023.

EL AUTOR:

Flores Ruano Darwin Steven

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCAS APLICADAS

CERTIFICACIÓN:

MAGISTER JAIME MICHILENA, DIRECTOR DEL PRESENTE TRABAJO DE TITULACIÓN CERTIFICA:

Que, el presente trabajo de titulación "DISEÑO DE UN PROTOTIPO QUE UNIFIQUE, UN SISTEMA DE PAGO BASADO EN RECARGAS, UN SISTEMA DE CONTROL DE INVENTARIO Y UN SISTEMA DE CONTROL DE ACCESO, MEDIANTE LA TECNOLOGÍA NEAR FIELD CONNECTION (NFC) PARA LA ASOCIACIÓN DE ESTUDIANTES CITEL-CIERCOM" ha sido desarrollado por el señor Darwin Steven Flores Ruano bajo mi supervisión.

Es todo en cuanto puedo certificar en honor de la verdad.

Ing. Jaime Roberto Michilena Calderón. MsC.

DIRECTOR

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

DEDICATORIA

Dedico este trabajo a mi familia y mis amigos, quienes fueron participes de cada uno de los momentos durante el desarrollo de este proyecto

Flores Ruano Darwin Steven

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

AGRADECIMIENTO

En primer lugar, quiero agradecer a mi familia por su apoyo incondicional durante todo este proceso. Sin su amor, paciencia y comprensión, no habría sido posible llegar hasta aquí.

También quiero expresar mi agradecimiento a mi director de tesis, Msc. Jaime Michilena por su guía, consejos y experiencia. Su dedicación y compromiso con mi proyecto fueron fundamentales para el éxito de mi trabajo de titulación.

Además, quiero agradecer al docente PhD. Marcelo Zambrano y Msc. Mauricio Domínguez por su enseñanza y conocimiento impartido, lo que me permitió adquirir las habilidades necesarias para llevar a cabo este proyecto.

También quiero agradecer a mis amigos, quienes me brindaron su apoyo y ánimo durante todo este proceso.

Flores Ruano Darwin Steven

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

RESUMEN

El presente trabajo de titulación detalla el desarrollo de un sistema que busca unificar diferentes procesos que se realizan dentro de la Asociación de estudiantes CITEL-CIERCOM con el fin de agilizar los mismos mediante la tecnología (Near Field Connection) NFC.

El sistema considera una parte de desarrollo de hardware y una parte de desarrollo de software, de esta manera, es posible contemplar estos dos bloques y analizarlos de manera individual. El primer bloque contempla el desarrollo de un dispositivo lector NFC, para ello se utiliza un microcontrolador y un módulo lector de tarjetas. Dicho lector cumple la función de enviar los datos obtenidos de cualquier tarjeta NFC hacía el PC en el que se encuentra alojada la aplicación desarrollada, además el apartado del hardware contempla el desarrollo de un control de acceso, para cumplir con ello, se parte de la base del lector desarrollado y se agregan elementos para que esté cumpla con la función indicada, de este modo la parte de desarrollo de hardware se ve completada. El siguiente bloque por desarrollar contempla la aplicación que permite la gestión de procesos y también unifica el funcionamiento con el hardware desarrollado, para cumplir con la etapa de desarrollo de software se diseñan de manera individual las aplicaciones y posteriormente se unifican, para el almacenamiento de todos los datos que se procesan, se hace uso de una base de datos alojada en la nube completando el desarrollo del sistema propuesto.

Finalmente, se realizan las respectivas pruebas de funcionamiento del sistema tanto en ambientes controlados como en ambientes reales donde se verifica que todas las funciones planteadas al inicio del proyecto sean resueltas con la culminación de este. Para esto se realiza pruebas a los componentes de hardware y software, las mismas permiten comprobar que el sistema cumple con los diferentes requerimientos propuestos al inicio logrando unificar los diferentes procesos realizados en la asociación y que los mismos sean agilizados mediante el uso de la tecnología NFC.

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

ABSTRACT

This degree work details the development of a system that seeks to unify different processes that are performed within the student association CITEL-CIERCOM in order to streamline them through Near Field Connection (NFC) technology.

The system considers a hardware development part and a software development part, in this way, it is possible to contemplate these two blocks and analyze them individually. The first block includes the development of an NFC reader device, using a microcontroller and a card reader module. This reader fulfills the function of sending the data obtained from any NFC card to the PC where the developed application is hosted, also the hardware section contemplates the development of an access control, to accomplish this, we start from the basis of the developed reader and add elements so that it fulfills the indicated function. In this way, the hardware development part is completed.

The next block to be developed contemplates the application that allows the management of processes and also unifies the operation with the developed hardware, to comply with the software development stage the applications are designed individually and then unified, for the storage of all data that are processed, use is made of a database hosted in the cloud completing the development of the proposed system.

Finally, the respective system performance tests are performed both in controlled environments and in real environments where it is verified that all the functions proposed

at the beginning of the project are resolved with the completion of this. For this purpose, the hardware and software components are tested and the results obtained are analyzed.

INDICE DE CONTENIDOS

Capítulo I Antecedentes 1

1.1 Problema Investigado 1

1.2 Objetivos 2

1.2.1 Objetivo General 2

1.2.2 Objetivos Específicos 2

1.3 Justificación 3

1.4 Alcance 4

Capítulo II Marco Teórico 7

2.1 Comunicaciones Inalámbricas 7

2.1.1 RFID 8

2.1.1.1 NFC 10

2.1.1.1.1 Historia 11

2.1.1.1.2 Estándares NFC 12

2.1.1.1.3 Modos de Operación 12

2.1.1.1.4 Elementos básicos de la comunicación NFC. 13

2.1.1.1.5 Comunicación NFC 14

2.1.1.1.6 Tecnología MIFARE 15

2.1.1.1.7 Formato NDEF 15

2.1.1.1.8 NFC Shield 16

2.1.1.1.9 Ventajas NFC 17

2.1.1.1.10 Limitaciones NFC 17

2.1.1.1.11 Aplicaciones NFC 18

2.2 Microcontroladores 19

2.3 Lenguaje de programación 23

2.3.1 Lenguaje de bajo nivel 24

2.3.2 Lenguaje de alto nivel 24

2.4 IDE 25

2.5 Software de programación de alto nivel 25

2.5.1 C++ 26

2.5.2 Arduino IDE 26

2.5.3 Java 26

2.5.4 Netbeans IDE 27

2.6 Bases de datos 27

2.6.1 Base de datos en la Nube 28

2.6.1.1 Características de Bases de datos en la Nube 28

2.6.2 Base de datos relacionales 29

2.6.2.1 SQL 29

2.6.3 Base de datos no relacionales 29

2.6.3.1 MongoDB 31

2.6.3.2 JSON 31

2.6.3.3 BSON 32

2.7 Metodología en Espiral 33

2.7.1 Análisis 34

2.7.2 Evaluación de Riesgos 34

2.7.3 Desarrollo y prueba 34

2.7.4 Planificación 35

Capítulo III. Diseño e implementación 36

3.1 Situación Actual 36

3.2 Modelo de Referencia IoT aplicado al sistema a desarrollar 41

3.3 Diagrama de Bloques 42

3.4 Elección de Hardware y Software 44

3.4.1 Tarjeta Lectora NFC. 45

3.4.2 Microcontrolador 47

3.4.3 Lenguaje de programación del microcontrolador. 49

3.4.4 Lenguaje de programación para el desarrollo de la aplicación Windows. 50

3.4.5 Base de Datos usada en el sistema 52

3.5 Análisis de Riesgos 54

3.6 Desarrollo del prototipo de control de acceso 56

3.6.1 Arquitectura del prototipo del sistema de control de acceso 56

3.6.2 Diagrama de conexión Eléctrica. 59

3.6.3 Programación Hardware. 63

3.7 Desarrollo prototipo de lector NFC 69

3.7.1 Arquitectura de prototipo de lector NFC. 70

3.7.2 Diagrama de conexión, del prototipo lector NFC. 71

3.7.3 Programación del Hardware del lector NFC. 72

3.8 Desarrollo Aplicaciones Windows 76

3.8.1 Desarrollo Aplicación Usuarios 77

3.8.2 Desarrollo Aplicación Control Acceso 81

3.8.3 Desarrollo Aplicación Micro Transacciones 85

3.8.4 Desarrollo Aplicación Aportes Voluntarios 90

3.8.5 Desarrollo Aplicación Gestión de Inventario 93

3.8.6 Unificación Aplicaciones Desarrolladas 98

3.9 Desarrollo de base de datos 99

3.9.1 Creación de Base de Datos. 100

3.9.2 Conexión Base de Datos 104

3.10 Implementación Prototipos 107

3.11 Pruebas Iniciales del sistema 110

3.11.1 Pruebas Iniciales de Hardware. 111

3.11.2 Pruebas Iniciales de Software. 113

3.12 Implementación del sistema 117

CAPITULO IV. PRUEBAS Y RESULTADOS 122

4.1 Cronograma de Pruebas 122

4.2 Prueba 1 Funcionamiento Lector NFC 124

4.2.1 Visualización funcionamiento 124

4.2.2 Conclusión 127

4.3 Prueba 2 Funcionamiento Aplicación Registro Usuarios 127

4.3.1 Visualización funcionamiento 128

4.3.2 Conclusiones 133

4.4 Prueba 3 Funcionamiento Aplicación Control de Acceso 134

4.4.1 Visualización funcionamiento 134

4.4.2 Conclusiones 139

4.5 Prueba 4 Funcionamiento Control Acceso 140

4.5.1 Visualización funcionamiento 140

4.5.2 Conclusiones 145

4.6 Prueba 5 Funcionamiento Aplicación Inventario 145

4.6.1 Visualización funcionamiento 146

4.6.2 Conclusiones 152

4.7 Funcionamiento Aplicación Micro Transacciones 153

4.7.1 Visualización funcionamiento 153

4.7.2 Conclusiones 159

4.8 Prueba 7 Funcionamiento Aplicación Matriculas 160

4.8.1 Verificación funcionamiento 160

4.8.2 Conclusiones 166

4.9 Resultados de las pruebas 167

Referencias 175

INDICE DE FIGURAS

Figura 1 Arquitectura propuesta para el desarrollo del proyecto 6

Figura 2 Formato NDEF..... 16

Figura 3 Controlador ESP8266 22

Figura 4 ESP32-C3..... 23

Figura 5 Formato de documento en base de datos no relacional..... 30

Figura 6 Ejemplo de formato JSON 32

Figura 7 Comparativa JSON y BSON..... 33

Figura 8 Metodología en Espiral..... 34

Figura 9 Modelo de referencia IoT aplicado al sistema 42

Figura 10 Diagrama de bloques propuesto..... 43

Figura 11 Diagrama de arquitectura del control acceso 57

Figura 12 Diagrama de conexión control de acceso..... 60

Figura 13 Librerías utilizadas para la programación..... 63

Figura 14 Código Arduino..... 65

Figura 15 Diagrama de flujo de configuración ESP32..... 68

Figura 16 Diagrama de Secuencia del módulo control de acceso 68

Figura 17 Arquitectura aplicación Windows control acceso..... 70

Figura 18 Diagrama de conexión aplicación Windows control acceso. 72

Figura 19 Configuración ESP32 comunicación serial..... 73

Figura 20 Diagrama de flujo programación ESP32 75

Figura 21 Diagrama Secuencia Modulo Lector NFC..... 76

Figura 22 Interfaz de Aplicación NetBeans..... 77

Figura 23 Interfaz gráfica de aplicación usuarios..... 78

Figura 24 Diagrama de secuencia para caso de uso de aplicación usuarios..... 80

Figura 25 Interfaz gráfica de aplicación control acceso 1/2 82

Figura 26 Interfaz gráfica de aplicación control acceso 2/2 83

Figura 27 Diagrama de secuencia para caso de uso de aplicación control de acceso .. 84

Figura 28 Interfaz gráfica aplicación micro transacciones 1/3 86

Figura 29 Interfaz gráfica aplicación micro transacciones 2/3 87

Figura 30 Interfaz gráfica aplicación micro transacciones 3/3 88

Figura 31 Diagrama de secuencia caso de uso aplicación micro transacciones 89

Figura 32 Interfaz gráfica aplicación pago aportes 91

Figura 33 Diagrama de secuencia caso de uso aplicación aportes voluntarios 92

Figura 34 Interfaz gráfica aplicación gestión inventario 1/3 94

Figura 35 Interfaz gráfica aplicación gestión inventario 2/3 95

Figura 36 Interfaz gráfica aplicación gestión inventario 3/3 96

Figura 37 Diagrama de secuencia caso de uso aplicación gestión inventario 97

Figura 38 Interfaz aplicación principal 99

Figura 39 Diagrama de base de datos 100

Figura 40 Registro en MongoDB 101

Figura 41 Creación cluster MongoDB Atlas 102

Figura 42 Selección de región en MongoDB Atlas 102

Figura 43 Cluster Creado 103

Figura 44 Creación base de datos y colección MongoDb Atlas 104

Figura 45 Conexión base de datos con aplicaciones 105

Figura 46 Conectar aplicación con MongoDb Atlas 106

Figura 47 Conexión aplicaciones a MongoDB Atlas 107

Figura 48 Diseño carcasa prototipo NFC 108

Figura 49 Prototipo Lector NFC 109

Figura 50 Prototipo Control Acceso 110

Figura 51 Lectura datos NFC. 111

Figura 52 Control Acceso Cerrado 112

Figura 53 Control Acceso Abierto 113

Figura 54 Registro Usuario 114

Figura 55 Micro Transacción Realizada 114

Figura 56 Registro Aportes Voluntarios 115

Figura 57 Registro Inventario NFC 116

Figura 58 Registro Control de Acceso 116

Figura 59 Ubicación instalaciones asociación CITEL-CIERCOM 117

Figura 60 Dispositivo control acceso implementado 118

Figura 61 Inventario agregado con etiquetas NFC 119

Figura 62 Funcionamiento Lector NFC 120

Figura 63 Funcionamiento Aplicación Windows 121

Figura 64 Prueba 1 Lectura de datos NFC 125

Figura 65 Prueba 1 Datos Obtenidos-Lector NFC 125

Figura 66 Prueba 1 Serial Port Monitor 126

Figura 67 Prueba 2 - Registro Usuario 128

Figura 68 Prueba 2 - Actualizar Cambios 129

Figura 69 Prueba 2 - Eliminar Usuarios 130

Figura 70 Matriz de Pruebas Aplicación Usuarios 131

Figura 71 Prueba 2 Base de datos aplicación usuarios 133

Figura 72 Prueba 3-Agregar Usuario 134

Figura 73 Prueba 3- Eliminar Usuario 135

Figura 74 Prueba 3 Función Registro Manual 136

Figura 75 Matriz Pruebas Control Acceso 137

Figura 76 Prueba 3 - Almacenamiento Datos 139

Figura 77 Prueba 4 Usuarios con Acceso 140

Figura 78 Prueba 4 Control Acceso Abierto 141

Figura 79 Prueba 4 registro ingreso control de acceso 142

Figura 80 Conexión a internet del dispositivo 142

Figura 81 Datos obtenidos de la base de datos 143

Figura 82 Generación Registro a Base de Datos 144

Figura 83 Registro Completado 144

Figura 84 Prueba 4 comparación registro automático y registro Manual 145

Figura 85 Prueba 5 registro inventario 146

Figura 86 Prueba 5 registro artículo inventario 147

Figura 87 Prueba 5 Proceso Préstamo Inventario 148

Figura 88 Prueba 5 Registro de Inventario 149

Figura 89 Matriz Pruebas Aplicación Inventario 150

Figura 90 Prueba 5 Datos alojados en la nube. 152

Figura 91 Prueba 6 acreditación sistema micro transacciones 154

Figura 92 Prueba 6 aplicación Micro transacciones desacreditación 155

Figura 93 Prueba 6 registro operaciones realizadas 156

Figura 94 Matriz Pruebas Aplicación Micro Transacciones 157

Figura 95 Prueba 6 almacenamiento datos en nube 159

Figura 96 Prueba 7 Uso Tarjeta NFC 161

Figura 97 Prueba 7 Aplicación Registro Aportes 162

Figura 98 Prueba 7 Aplicación actualización aportes 163

Figura 99 Prueba 7 aplicación aportes - generación registro 164

Figura 100 Matriz de Pruebas Aplicación Matriculas 165

Figura 101 Prueba 7 Base de datos aplicación aportes..... 166

Figura 102 Análisis de Resultados 171

INDICE DE TABLAS

Tabla 1 Abreviatura Requerimientos Hardware y Software 44

Tabla 2 Requerimientos Hardware Tarjeta Lectora 45

Tabla 3 Análisis comparativo de especificaciones técnicas lectores NFC..... 46

Tabla 4 Elección Hardware Modulo Lector..... 46

Tabla 5 Requerimientos de Hardware del Microcontrolador..... 47

Tabla 6 Comparativa de especificaciones de microcontroladores 48

Tabla 7 Elección Hardware Microcontrolador..... 48

Tabla 8 Requerimientos Lenguaje Programación..... 49

Tabla 9 Comparativa entre software para microcontroladores 50

Tabla 10 Requerimientos del lenguaje de programación de aplicación..... 51

Tabla 11 Análisis comparativo de lenguajes de programación para aplicaciones de escritorio 51

Tabla 12 Requerimientos de base de datos 52

Tabla 13 Comparativa Bases de Datos 53

Tabla 14 Análisis de Riesgos de Hardware Seleccionado 54

Tabla 15 Análisis de Riesgos Software..... 55

Tabla 16 Consumo de dispositivos electrónicos 62

Tabla 17 Cronograma de Pruebas 122

INDICE DE ECUACIONES

Ecuación (1) Ley de Ohm..... 60

Ecuación (2) Cálculo para resistencia de led..... 61

Ecuación (3) Cálculo para intensidad de corriente..... 62

Capítulo I Antecedentes

1.1 Problema Investigado

Los avances tecnológicos realizados en estas últimas décadas han permitido la automatización y optimización de una gran cantidad de tareas de nuestra vida cotidiana Actualmente, actividades como las de comunicación, intercambio de bienes y servicios, control de accesos, supervisión, entre muchas otras, pueden ser gestionadas y automatizadas a través de la utilización de hardware y software desarrollado con este propósito (smartphones, sensores y actuadores digitales, cámaras, *tag*s, etc.)

El uso de dinero electrónico se ha ido implementando en diferentes áreas, especialmente en entornos más comerciales. Dicho dinero electrónico incluye a las tarjetas prepago, tarjetas de crédito o monederos electrónicos. Sin embargo, en nuestro entorno el uso de dicha tecnología no ha sido del todo aprovechad.(Rodríguez Andrea, 2019).

Actualmente, nos encontramos en una situación de emergencia debido a la pandemia provocada por el covid-19 y uno de los principales problemas radica en el contacto físico requerido para el intercambio de dinero, pues se ha determinado que el coronavirus puede sobrevivir de cierta manera en los billetes o monedas.(elEconomista, 2021).

Además de que el usar dinero físico es de cierta manera mucho más lento debido a la propia naturaleza de los intercambios físicos de bienes y servicios, sin embargo, es la única forma de pago en establecimientos pequeños como la copiadora de la Facultad de Ingeniería en Ciencias Aplicadas. En está, gran parte de los estudiantes y docentes realizan compras de artículos de papelería y similares, de tal modo que al ser la única copiadora dentro de la facultad es inevitable la aglomeración, lo que supone una mayor posibilidad de contagio.

La asociación de CITEL-CIERCOM, ha buscado la forma de beneficiar a los estudiantes en la obtención de copias en la copiadora de la facultad, sin embargo, esto nunca ha llegado a utilizarse de manera eficiente. Por lo que el uso de dinero digital podría ayudar a solventar dicho inconveniente.

Otro claro problema es que en la asociación de estudiantes CITEL-CIERCOM no se tiene ningún tipo de control sobre quien ingresa a las oficinas, de tal modo que tanto estudiantes miembros y no miembros de dicha asociación, pueden ingresar sin ningún tipo de restricción, esto puede suponer diferentes problemas debido a que cualquier estudiante tendría acceso a información que se encuentre dentro, esto sumado a que no se tiene un control concreto de lo que se posee dentro de la asociación puede derivarse en la perdida de diferentes artículos.

1.2 Objetivos

1.2.1 Objetivo General

Diseñar un prototipo que unifique, un sistema de pago basado en recargas, un sistema de control de inventario y un sistema de control de acceso, mediante la tecnología *Near Field Connection* (NFC) para la asociación de estudiantes CITEL-CIERCOM.

1.2.2 Objetivos Específicos

- Realizar una investigación acerca de los diferentes procesos de la asociación de estudiantes CITEL-CIERCOM para establecer necesidades que serán resueltas en el sistema multifunción NFC.
- Diseñar los 3 sistemas propuestos de manera individual y desarrollar un programa para Windows que permita la gestión y unificación de dichos sistemas utilizando una base de datos alojada en la nube.
- Evaluar el funcionamiento del sistema diseñado realizando las pruebas necesarias en la asociación de estudiantes CITEL-CIERCOM.

- Analizar los beneficios obtenidos en cuanto a la agilización de procesos al utilizar el sistema propuesto comparado con alternativas tradicionales no tecnológicas.

1.3 Justificación

Este proyecto tiene como finalidad, aplicar los diferentes conocimientos adquiridos en el transcurso de la carrera de Ingeniería en Telecomunicaciones. Además de contribuir al desarrollo social e incentivar el uso de nuevas tecnologías en distintas actividades.

De este modo el sistema que se pretende desarrollar tendrá como objetivo facilitar y agilizar tareas cotidianas como realizar pagos, acceder a información personal y control de acceso, mediante la automatización de las mismos. Además, se podrá integrar todos estos servicios en una sola plataforma, logrando una mejor comodidad para los usuarios del sistema. Puesto que la comunicación entre dispositivos se realiza rápida y fácilmente simplemente acercando los dispositivos NFC, sin necesidad de realizar configuraciones (Anaya-Cantellán & López-Martínez, 2014).

Con el sistema también se pretende tener un aporte de mayor seguridad, pues al implementar los pagos mediante un sistema de tarjetas NFC, el portar dinero físico, no será una prioridad por lo que siempre será más seguro.

Además, al implementar el sistema propuesto, se pretende solucionar algunos de los problemas que se han visualizado en la asociación de estudiantes CITEL-CIERCOM mencionados anteriormente. Esto además genera un incentivo para implementar nuevas tecnologías en beneficio de la carrera de Ingeniería en Telecomunicaciones de la Universidad “Técnica del Norte”.

1.4 Alcance

El presente proyecto tiene como finalidad la unificación de diferentes servicios en un solo sistema, para ello, se pretende unificar las siguientes aplicaciones NFC.

- Sistema de pagos basado en recargas mediante el uso de una caja común para la copiadora de la Facultad de Ingeniería en Ciencias Aplicadas.
- Sistema de control de inventario para préstamo de material a estudiantes, que dispone la aso-escuela CITEL-CIERCOM.
- Sistema de acceso mediante una cerradura electrónica para el ingreso a la oficina de la asociación de estudiantes CITEL-CIERCOM.

De esta manera, se busca la unificación de todas las aplicaciones mencionadas en un sistema NFC.

En primera instancia, se pretende desarrollar un sistema de recaudo, que servirá para agilizar procesos dentro de la copiadora de la Facultad de Ingeniería en Ciencias Aplicadas utilizando un lector electrónico que permitirá realizar recargas y cobros de dinero en una tarjeta NFC.

Haciendo uso del convenio que existe entre la asociación de estudiantes CITEL-CIERCOM y la copiadora de la Facultad de Ingeniería en Ciencias Aplicadas, las recargas se realizarán únicamente en las oficinas de la asociación de estudiantes CITEL-CIERCOM aplicando métodos de seguridad para que las recargas sean veraces. Por otro lado, la encargada de la copiadora podrá realizar el cobro de dinero de la tarjeta utilizando un lector NFC.

En este apartado se deberán establecer normas para el correcto uso de una caja común para que la solución a implementar sea eficiente y funcional.

En el segundo caso, se desarrollará un sistema de inventario para artículos que posee la asociación con el fin de realizar los préstamos a estudiantes de manera eficiente, para ello,

se deberá etiquetar cada elemento que posee la aso-escuela CITEL-CIERCOM como pueden ser placas Arduino, cables UTP y *Protoboards*.

Con todos los elementos disponibles para préstamo correctamente etiquetados, se puede llevar un mejor registro de cada préstamo realizado, evitando así la perdida de dicho material, para realizar un préstamo o devolución, cada material deberá pasar por el lector NFC para que este sea verificado.

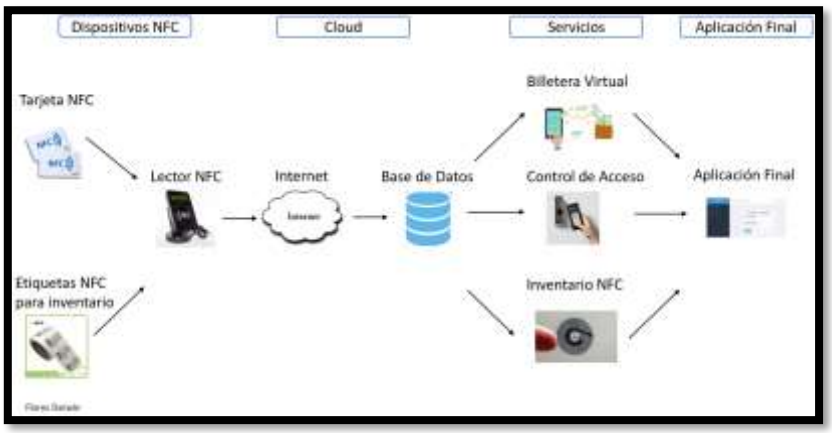
En tercera instancia, se desarrollará un sistema de control de acceso utilizando una ranura electrónica NFC para que los estudiantes miembros de la Aso-Escuela CITEL-CIERCOM puedan acceder a sus instalaciones, para ello cada miembro deberá tener un código único que será registrado en su tarjeta NFC que permitirá que la cerradura se abra, logrando de esta manera tener un mejor control de quienes ingresan a las oficinas de la asociación.

Una vez se hayan diseñado los diferentes servicios de manera individual, se procederá al desarrollo de una aplicación que permita la unificación y gestión de dichos servicios. Está aplicación será desarrollada en Java, además dicha aplicación hará uso de una base de datos que será alojada en la nube.

Con todo ello mencionado, la arquitectura en la que se basará el proyecto se presenta en la **Figura 1**

Figura *1*

Arquitectura propuesta para el desarrollo del proyecto



Fuente: Elaborado por el autor

Para finalizar, se realizarán las pruebas correspondientes entregando diferentes tarjetas NFC a estudiantes de la carrera de Ingeniería en Telecomunicaciones, con ello se busca recopilar datos de uso, así como también recopilar posibles errores que puedan aparecer. Posteriormente se analizarán los datos obtenidos para verificar cuales son los puntos fuertes y puntos débiles de la implementación del sistema NFC propuesto.

Capítulo II Marco Teórico

En el presente capítulo, se brinda la información acerca de las diferentes tecnologías que serán utilizadas dentro de la realización del presente proyecto, de esta forma se pretende tener un acercamiento inicial que permitirá comprender de mejor manera el desarrollo que se tendrá posteriormente.

Con ello, en el capítulo se habla inicialmente de las comunicaciones inalámbricas, ya que está es la base del presente proyecto, en esta sección se tendrán definiciones, elementos, aplicaciones, una vez se haya completado lo referente a comunicaciones inalámbricas se abordan los microcontroladores, otra parte fundamental para el proyecto, en este apartado se busca identificar las funciones que pueden realizar y su uso en los sistemas embebidos, para finalmente tener algunos ejemplos de los microcontroladores.

Posteriormente se tiene un apartado dedicado al lenguaje de programación y los respectivos IDE que fundamental para el desarrollo de aplicación y la propia programación del microcontrolador, el tema siguiente será aquel que comprende las bases de datos, buscando comprender como funcionan y cuál es el uso dentro de los sistemas embebidos.

Finalmente se presenta la metodología, que es aquella que nos permitirá tener avances regulares y completar de manera satisfactoria el proyecto presentado.

2.1 Comunicaciones Inalámbricas

La comunicación inalámbrica es una de las tecnologías más impactantes de la historia que afecta drásticamente la forma en que vivimos, trabajamos, jugamos e interactuamos con las personas y el mundo. Hay miles de millones de usuarios de teléfonos móviles en todo el mundo, y una amplia gama de dispositivos además de los

teléfonos utilizan diferentes tecnologías inalámbricas para interconectarse (Goldsmith, 2020).

Las nuevas tecnologías inalámbricas han evolucionado logrando brindar aplicaciones de transporte de datos, voz y video en todo tipo de dispositivos como celulares, computadores, televisiones, automóviles, electrodomésticos, relojes e incluso la ropa., esto ha permitido avances en diferentes áreas, pues al tener como medio de transmisión el aire, la implementación con tecnologías inalámbricas es sencilla, rápida y en algunos casos más económico que las variantes que utilizan medios físicos.

Entre las tecnologías inalámbricas más utilizadas para la transmisión de datos, se encuentran Wi-Fi y Bluetooth, sin embargo, no son las únicas, tal es el caso de la tecnología de identificación por radio frecuencia, una tecnología que es capaz de transmitir una pequeña cantidad datos, suficientes para lograr la identificación de un usuario, de manera rápida y eficaz. Un ejemplo de uso local de dicha tecnología se encuentra en el uso de telepass, en la empresa de Panavial.

2.1.1 RFID

RFID (identificación por radiofrecuencia) es una forma de comunicación inalámbrica que, mediante el uso de campos magnéticos y ciertas frecuencias específicas, se logra intercambiar información, esto es especialmente útil para identificar de manera única un objeto, animal o persona.

Entre las características que presenta RFID están:

- Trabaja en un rango de frecuencias que van desde bandas de baja frecuencia (KHz) hasta bandas de alta frecuencia (GHz).
- Existen tres tipos de tags (etiquetas): activos, pasivos y semi pasivos.

- “Para los tags activos, su fuente de alimentación es propia mediante baterías de larga duración, generalmente compuestas de Litio o Dióxido de Manganeso. La duración de estas depende del modelo de tag y de la actividad que tenga, pero suele ser de varios años” (Israel, 2017).
- Tiene distintas distancias para la lectura y escritura de sus tags (etiquetas) y pueden llegar generalmente hasta los 100m.
- La memoria interna generalmente es de 4 y 32 kbytes. (Israel, 2017)

El uso de sistemas basados en RFID no se ha desarrollado completamente, sin embargo, cada vez se utiliza en más ámbitos, debido a que la implementación de esta no representa costos elevados.

La tecnología RFID considera dos elementos principales para la comunicación: etiquetas y lectores, si alguno de ellos falla, dicha comunicación no será posible.

Etiquetas RFID: “Son el componente principal de cualquier sistema de radiofrecuencia y son las encargadas de emitir una señal de respuesta a otra señal enviada por un lector. Estas etiquetas RFID también reciben el nombre de transponder (*Transmitter* + *Responder*)” (Broseta, 2012).

Lectores RFID: El lector RFID es el dispositivo encargado de leer y escribir en las etiquetas RFID, todo esto lo realizan sin contacto alguno puesto que la transferencia de datos se realiza por proximidad utilizando comunicación inalámbrica.

Estos lectores pueden escribir uno o varios tags y su funcionamiento es sencillo. La antena del lector crea un campo magnético y cuando el tag entra en contacto con el campo magnético creado por el lector, reacciona de forma inmediata y envía al lector la información almacenada. El lector decodifica esos datos que por medio de una

infraestructura de red son procesados y tratados para efectuar acciones necesarias (Broseta, 2012).

2.1.1.1 NFC

NFC (*Near Field Communication)* es una tecnología de conectividad inalámbrica de corto alcance (distancias menores a 5cm) basada en el estándar ISO 1444, que permite el intercambio de datos y la interconexión entre dispositivos electrónicos de consumo, dispositivos móviles, electrodomésticos y etiquetas NFC compatibles, de manera rápida y eficiente.

> El hecho de que la comunicación no pueda producirse a más de 5 centímetros "garantiza" la seguridad de la comunicación al ser prácticamente imposible interceptar la señal sin que una persona se percate de ello. Además, si dos dispositivos establecen una comunicación NFC podemos asegurar que se encuentran sumamente cercanos (Broseta, 2012).

Dicha conexión puede ser de forma unidireccional o bidireccional. Una de las principales características de NFC es que utiliza una radiocomunicación de alta frecuencia. Esta tecnología opera en la banda de los 13,56Mhz y es capaz de transmitir datos en diferentes velocidades.

Es posible determinar que la tecnología NFC tiene un gran potencial en la sociedad actual, debido a las características que está ofrece, puede ser utilizada en una gran variedad de ámbitos.

Según (Broseta, 2012) , la tecnología NFC puede estar y estará presente en tareas tan variadas como identificación de personas, etiquetado de objetos, pagos por móvil, canjeo de *tickets*, transporte, *loggeo* seguro, accesos, intercambio de archivos, intercambio de tarjetas de visita, agregar personas a redes sociales, etc.

2.1.1.1.1 Historia

La comunicación de campo cercano (NFC) tiene sus orígenes en la identificación por radiofrecuencia (RFID). De hecho, NFC es en realidad un subconjunto de RFID con un rango de comunicación más corto por motivos de seguridad. En 2004, Nokia, Sony y Philips se unieron para formar el NFC *Forum*. Este grupo está dedicado a promover la seguridad, la facilidad de uso y la popularidad de la comunicación de campo cercano. Su objetivo es educar a las empresas sobre la tecnología y mantiene los estándares que permiten que NFC funcione entre diferentes dispositivos. Aquellos que deseen crear dispositivos compatibles con NFC deben cumplir con estos estándares establecidos por NFC *Forum*. Esto garantiza que cualquier usuario con cualquier dispositivo NFC pueda usarlo con cualquier otro dispositivo NFC o etiqueta NFC (NearFieldCommunication.org, 2017).

Aunque el NFC *Forum* se formó en 2004, no fue hasta 2006 que el grupo produjo el primer conjunto de especificaciones para las etiquetas NFC. Las etiquetas NFC son objetos pequeños, como una calcomanía, que contienen información que un dispositivo compatible con NFC, como un teléfono inteligente, puede interceptar cuando se pasa sobre la etiqueta NFC. La información en la etiqueta generalmente es de solo lectura, pero ciertas etiquetas permiten que el dispositivo que la lee escriba información nueva o también altere la información anterior en la etiqueta (NearFieldCommunication.org, 2017).

Actualmente dicha tecnología se ha convertido en una de las principales formas de pago, ya que la mayoría de los dispositivos móviles inteligentes (Smartphones) cuentan con dicha tecnología.

2.1.1.1.2 Estándares NFC

Al desarrollar dispositivos de comunicación de campo cercano y nueva tecnología, se deben cumplir los estándares NFC. Existen estándares para garantizar que todas las formas de tecnología de comunicación de campo cercano puedan interactuar con otros dispositivos compatibles con NFC y funcionen con dispositivos más nuevos en el futuro.

"Existen dos especificaciones principales para la tecnología NFC: ISO/IEC 14443 e ISO/IEC 18000-3. El primero define las tarjetas de identificación utilizadas para almacenar información, como la que se encuentra en las etiquetas NFC. Este último especifica la comunicación RFID utilizada por los dispositivos NFC" (NearFieldCommunication.org, 2017).

> ISO/IEC 18000-3 es un estándar internacional para todos los dispositivos que se comunican de forma inalámbrica en la frecuencia de 13,56 MHz mediante tarjetas de tipo A o tipo B, como lo hace la comunicación de campo cercano. Los dispositivos deben estar dentro de los 4 cm uno del otro antes de que puedan transmitir información. Los estándares explican cómo un dispositivo y la etiqueta NFC que está leyendo deben comunicarse entre sí. El dispositivo se conoce como el dispositivo de interrogación, mientras que la etiqueta NFC se conoce simplemente como la etiqueta (NearFieldCommunication.org, 2017).

2.1.1.1.3 Modos de Operación

Al igual que en otras tecnologías de comunicación inalámbrica tales como: Bluetooth, SigFox, Zigbee o Wi-Fi, existen diferentes modos de operación mediante los cuales trabaja la tecnología NFC.

Según (Broseta, 2012), existen dos modos de funcionamiento y todos los dispositivos del estándar NFCIP1 deben soportar ambos modos:

- ***Modo Activo***

Cada uno de los dispositivos, es decir, emisor y receptor son capaces de generar su propio campo electromagnético, de tal modo que ambos dispositivos requieren de energía para funcionar.

- ***Modo Pasivo***

En este caso, solamente un dispositivo es capaz de generar un campo electromagnético y el otro, hace uso de la modulación de carga para poder transmitir los datos. El iniciador es el encargado de generar el campo electromagnético.

2.1.1.1.4 Elementos básicos de la comunicación NFC.

Al igual que la tecnología MIFARE, NFC requiere de dos elementos fundamentales para que se establezca la comunicación, estos son: Dispositivos NFC y etiquetas NFC.

- ***Dispositivos NFC***

Existe una gran variedad de dispositivos NFC, sin embargo, entre los dispositivos más habituales que integran la tecnología se encuentran teléfonos móviles, *readers* o lectores, algunas PCs, relojes inteligentes, pulseras inteligentes, electrodomésticos entre otros.

- ***Etiquetas NFC***

De la misma manera, se tienen dispositivos electrónicos como los mencionados anteriormente, a esto se debe agregar diferentes etiquetas que no necesitan una fuente de energía constante para su funcionamiento. Es en estas etiquetas en las que se la información es almacenada.

2.1.1.1.5 Comunicación NFC

La comunicación NFC puede darse entre diferentes dispositivos de los nombrados anteriormente, para ello a continuación se detalla las tres formas más utilizadas de comunicación.

- ***Móvil NFC – Etiqueta NFC***

El dispositivo iniciador, en este caso el móvil y gracias a la energía emitida por el lector contenido en este y al circuito transpondedor integrado en la etiqueta RFID, es posible obtener la información contenida en la etiqueta. Del mismo modo se podrá escribir en la etiqueta RFID (Broseta, 2012).

- ***Móvil NFC – Lector NFC***

Los dos dispositivos generan un campo electromagnético y mediante la conexión inalámbrica se logran transmitir los datos desde el móvil NFC al lector, para ello el móvil emula los datos como si de una etiqueta se tratase, un claro ejemplo de este tipo de comunicación es aquella que se genera mediante aplicaciones de pago como Google Pay.

- ***Lector NFC- Etiqueta NFC***

También es posible la comunicación en la que una etiqueta RFID puede ser leída y/o escrita por un lector o *reader* fijo. Este *reader* podrá estar conectado a un PC (Broseta, 2012).

Toda comunicación NFC consta de 5 fases muy bien diferenciadas.

- Descubrimiento
- Autenticación
- Negociación
- Transferencia
- Reconocimiento

2.1.1.1.6 Tecnología MIFARE

MIFARE constituye un estándar tecnológico para comunicaciones de "no contacto" a 13,56MHz cuyo propietario es *Philips Electronics*. Esta compañía no fabrica etiquetas ni lectores RFID, si no que fabrica y vende al mercado los chips que luego se incluyen en etiquetas y lectores. Una de las características de dicha tecnología es que permite leer y escribir en etiquetas RFID. A su vez, MIFARE está descrita en el estándar ISO 14443 Tipo A. Las etiquetas MIFARE permiten incrustar un módulo opcional de chip inteligente de contacto. Pero también está diseñada para tener una banda magnética. En este caso, la etiqueta cumpliría con el estándar 7811 (Broseta, 2012).

Las etiquetas sin contacto MIFARE y los lectores de etiquetas MIFARE fueron desarrollados en un principio para transacciones de pago en sistemas de transporte público. Gracias a su corto alcance de lectura, la tecnología MIFARE es especialmente apropiada para realizar funciones de adicción/sustracción (Broseta, 2012).

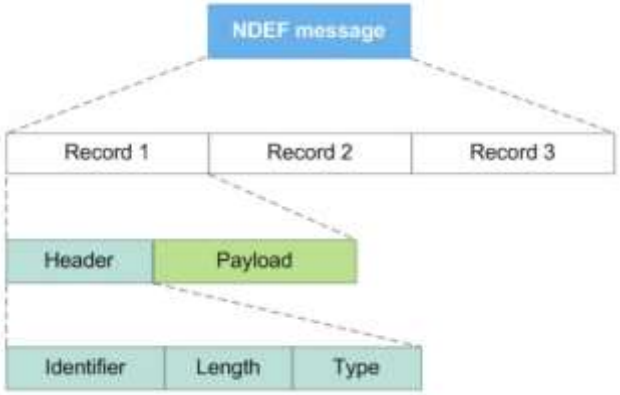
> Según (Broseta,2012), el alcance típico de lectura/escritura de un lector de etiquetas sin contacto MIFARE es de 2 a 10cm. Y la capacidad normal de memoria de las etiquetas MIFARE es de 1KB de memoria EEPROM, siendo la máxima de 4KB. Una etiqueta MIFARE tiene 16 sectores que son independientes uno del otro en los que es posible almacenar información. Sin embargo, el primer sector siempre se usa como directorio de la etiqueta, con lo que solo será posible almacenar datos en los 15 sectores restantes.

2.1.1.1.7 Formato NDEF

El formato NDEF o *NFC Data Exchange Format* es un formato común registrado por el NFC *Forum* para poder compartir datos entre los dispositivos NFC y/o entre los dispositivos NFC y las etiquetas. Por lo tanto, las etiquetas que son leídas y/o escritas utilizando este formato, se les conoce como etiquetas NDEF o etiquetas NFC (Broseta,

2012). Para comprender de mejor manera, la **Figura *2*** muestra el formato NDEF descrito anteriormente

Figura 2
Formato NDEF



Fuente*: (Mngomezulu, s.f.)*

Mediante el formato NDEF es posible organizar los datos almacenados en bytes dentro de la etiqueta. NDEF se caracteriza por poseer una cabecera de datos, denominada cabecera NDEF a partir de la cual se encuentran los bloques de información. Cada bloque de información se agrupa en registros que contienen a su vez los datos agrupados en mensajes NDEF y caracterizados por un tipo MIME definido (Broseta, 2012).

Sin embargo, NDEF tiene desventajas, entre ellas es que todos los bloques de información son accesibles inicialmente para cualquier dispositivo, debido a que utiliza una clave por defecto para acceder a dichos datos. Esto significa que cualquiera puede alterar los datos en cualquier momento (Broseta, 2012).

El formato NDEF es soportado por cualquier dispositivo NFC, puesto que forma parte de un estándar para el intercambio y el almacenamiento de información.

2.1.1.1.8 NFC Shield

Es un chip capaz de leer y escribir en las etiquetas NFC, dicha placa puede estar basada en el módulo NFC PN532 y el módulo RC522, de este modo, se constituye como

una plataforma ideal para el desarrollo de aplicaciones y trabaja con tarjetas NFC en la frecuencia 13,56Mhz.

Los módulos tienen un diseño compacto que incluye la antena en el mismo PCB. Posee 3 tipos de interfaces de comunicación: SPI, I2C y UART. El chip PN532 es muy conocido por lo que existen librerías para Arduino, Raspberry Pi y otras plataformas. Incluye soporte para comunicación *peer-to-peer* NFC, haciendo de este módulo superior al módulo RFID RC522 (NXPSemiconductors, 2019).

2.1.1.1.9 Ventajas NFC

El uso de la tecnología NFC cada vez está creciendo, esto dada la gran cantidad de aplicaciones que tienen en diferentes áreas, cada tecnología inalámbrica posee características que enfocan el uso en al cual se orientan más, de la misma manera NFC posee ciertas características que brindan ventajas respecto a otras tecnologías de comunicación de campo cercano, algunas de las principales ventajas son:

- Múltiples aplicaciones de la tecnología como control de acceso, pagos, identificación, entre otras.
- No requiere de configuraciones complicadas.
- Proporciona seguridad respecto a tecnologías basadas en bandas magnéticas.
- La comunicación se realiza de manera instantánea.
- Alta eficiencia en sistemas

2.1.1.1.10 Limitaciones NFC

NFC a pesar de ser una tecnología robusta, también tiene ciertos inconvenientes, los cuales hacen que el uso de esta tecnología se vea limitado, no pudiéndose utilizar en ciertos ámbitos. Con ello se tienen algunas de los inconvenientes que presenta la tecnología NFC.

- La distancia de operación de la tecnología es muy corta, alrededor de unos 5 a 10cm.
- Las velocidades de transferencia no son muy altas si las comparamos con otras tecnologías inalámbricas como wifi.

2.1.1.1.11 Aplicaciones NFC

Las aplicaciones de NFC pueden ser muy variadas, es por ello que a continuación se describen algunas de ellas para tener un acercamiento al desarrollo del proyecto que se va a tratar.

- ***Automatización de procesos***

NFC es la tecnología perfecta para modificar perfiles o preferencias desde el teléfono móvil a cualquier dispositivo. Por ejemplo, es posible controlar en una casa domótica la potencia de la luz, la temperatura, el encendido o apagado de un ordenador, etc. Un dispositivo que cada vez aparece con más fuerza en las tiendas y grandes almacenes son los altavoces NFC, que reproducen la música del Smartphone que esté cerca (Moreno, 2012).

- ***Control de acceso***

Ya sea en el hogar, hoteles o en garajes particulares, es una gran comodidad contar con un sistema NFC que permita simplemente acercando el teléfono móvil poder abrir una puerta. En un garaje particular es una fantástica opción, para evitar los tradicionales mandos y sus copias, de esta forma cada persona con su teléfono móvil siempre tiene la llave del garaje (Moreno, 2012).

- ***Identificación***

NFC está siendo ya utilizando como sistema de identificación en edificios de oficinas, donde los empleados acceden a la oficina con validación directa desde su

Smartphone. Es probable que en los próximos años también empiecen a aparecer pasaportes o permisos de conducir integrados con los teléfonos móviles para llevarlos siempre encima y aprovechar todo el potencial de NFC (Moreno, 2012).

- ***Transacciones***

Una de las aplicaciones que existe en algunos países es el hecho de poder realizar transacciones en cuestiones del día a día. Por ejemplo, al momento de realizar pagos referentes al transporte público. Con la utilización de una tarjeta y simplemente acercándola al dispositivo de validación se puede ejecutar la transacción, descontar el saldo que ha costado el viaje. "La tecnología NFC también se está utilizando en otros medios de transporte para comprobar que cada pasajero ha pagado su pasaje, esto ocurre actualmente en ciudades como Los Ángeles" (Moreno, 2012).

- ***Emparejamiento con dispositivos inteligentes***

Aunque tradicionalmente durante los últimos años se ha utilizado bluetooth, con NFC existe una posibilidad de utilizar la tecnología NFC para compartir datos sin limitaciones, siempre aprovechando la proximidad entre dos usuarios (Moreno, 2012). Además, es útil para el emparejamiento de modo que las tecnologías se pueden complementar entre ellas.

2.2 Microcontroladores

Los microcontroladores se han desarrollado para cubrir las más diversas aplicaciones. Se usan en automoción, en equipos de comunicaciones y de telefonía, en instrumentos electrónicos, en equipos médicos e industriales de todo tipo, en electrodomésticos, en juguetes, etc. (Valdés Pérez & Ramon, 2007).

Los microcontroladores están concebidos fundamentalmente para ser utilizados en aplicaciones puntuales, es decir, aplicaciones donde el microcontrolador debe realizar un pequeño número de tareas, al menor costo posible (Valdés Pérez & Ramon, 2007).

De este modo, los microcontroladores pueden definirse como pequeños computadores consolidados, ya que, en un mismo circuito electrónico son capaces de albergar la memoria RAM, memoria interna y el procesador. Además, la mayoría de estos microcontroladores poseen pines tanto de entrada como de salida que permiten acoplar diferentes equipos electrónicos como son los sensores.

El uso de microcontroladores es fundamental en el desarrollo del presente proyecto, puesto que, para la interconexión y el procesamiento de los datos, se necesita una unidad de procesamiento, de este modo un microcontrolador cumple con los requerimientos necesarios además de que su tamaño lo hace ideal para trabajar en elementos portables como puede ser el sistema.

Existen algunos tipos de microcontroladores que cada vez son más utilizados, a continuación, se brinda una lista de algunos de los microcontroladores que más repercusión han tenido en los últimos años

- **Arduino Uno**

Arduino Uno es un microcontrolador que ha sido diseñado para el desarrollo de sistemas embebidos, dichos sistemas pueden ser tan básicos como encender un led o tan complejos como el monitoreo, Arduino uno es una de las mejores opciones a la hora de realizar proyectos básicos, pues cuenta con conexión USB, pines de entrada y salida y un procesador ATmega328P, lo suficientemente potente para iniciar en el mundo de los sistemas embebidos.

- **Raspberry Pi Pico**

Raspberry Pi Pico es un microcontrolador de bajo costo que cuenta con algunas características de Raspberry Pi original, sin embargo, la principal diferencia radica en el tamaño, pues es más comparable al tamaño de un Arduino uno, entre algunas de las características se tiene que posee USB 1.1, 264Kb de RAM, 2Mb de memoria RAM, que son suficientes para desarrollar proyectos, además contiene una gran documentación por parte de *raspberry*.

- **Microcontrolador ESP8266**

  El SoC (*System On a Chip*) ESP8266 de *Espressif Systems* es un chip especialmente diseñado para las necesidades de un mundo conectado, integra un potente microcontrolador con arquitectura de 32 bits y conectividad Wi-Fi. El SoM(*System on Module*) ESP-01 fabricado por Ai-Thinker integra en un módulo el SoC ESP8266, memoria FLASH, cristal oscilador y antena WiFi en PCB. (Naylampmechatronics, 2021)

La plataforma ESP8266 permite el desarrollo de aplicaciones en diferentes lenguajes de programación como: Arduino, Lua, MicroPython, C/C++, Scratch. Al trabajar con el entorno Arduino es posible utilizar un lenguaje de programación conocido y utilizar un IDE sencillo, además de hacer uso de toda la información sobre proyectos y librerías disponibles en internet. La comunidad de usuarios de Arduino es muy activa y da soporte a plataformas como el ESP8266. Al trabajar con Lua podemos experimentar con un lenguaje interpretado. Dentro de las principales placas de desarrollo o módulos basados en el ESP8266 tenemos: ESP-01, ESP-12E, Wemos D1 mini y *NodeMCU* v2 (Naylampmechatronics, 2021). En la **Figura 3** es posible visualizar el kit de desarrollo basado en ESP8266.

Figura *3*
Controlador ESP8266

Fuente: *(Naylampmechatronics, 2021)*

- **Microcontrolador ESP32**

El ESP-WROOM-32 es un potente módulo que integra Wi-Fi y Bluetooth, ideal para desarrollar productos de IoT. La integración de Bluetooth, Bluetooth LE y Wi-Fi permite una amplia gama de aplicaciones, el uso de Wi-Fi permite una comunicación de mediano alcance y conectarse a una red LAN y a través de un Router conexión a Internet, mientras que el Bluetooth nos permite conectarse directamente a otro dispositivo como un celular (Naylampmechatronics, 2021).

En el núcleo de este módulo está el chip ESP32-D0WDQ6. El chip integrado está diseñado para ser escalable y adaptado, en la **Figura 4** es posible visualizar el kit de desarrollo el cual ha sido adaptado y diseñado para la creación de prototipos electrónicos. Hay dos núcleos de CPU que se pueden controlar individualmente, y la frecuencia del reloj es ajustable de 80 MHz a 240 MHz. El usuario también puede apagar el CPU y utilizar el coprocesador de baja potencia para supervisar constantemente los periféricos para detectar cambios de estado (Naylampmechatronics, 2021).

Figura *4*
ESP32-C3

Fuente: *(ESPRESSIF Systems, 2022)*

ESP32 puede funcionar como un sistema independiente completo o como un dispositivo esclavo, reduciendo la sobrecarga de la pila de comunicación en el procesador de la aplicación principal. ESP32 puede interactuar con otros sistemas para proporcionar funcionalidad Wi-Fi y Bluetooth a través de sus interfaces; SPI / SDIO o I2C / UART (Arduino, 2016).

Todos los microcontroladores necesitan de instrucciones de acuerdo con los proyectos que se pretende realizar, esto se lo realiza mediante programación. Existen diferentes lenguajes de programación que trabajan con diferentes microcontroladores, es por ello que en la siguiente sección se tratan los lenguajes de programación.

2.3 Lenguaje de programación

El lenguaje de programación es un lenguaje que permite a un desarrollador, establecer diferentes acciones mediante una serie de instrucciones ordenadas, de esta manera es posible ejecutar operaciones, obtención de datos, y diferentes tipos de algoritmos que posteriormente permiten crear aplicaciones con las que se controlan dispositivos electrónicos (Redator, 2019).

Los lenguajes de programación cumplen con la función de intercomunicar a el programador y la máquina, con esto es posible tener de forma precisa diferentes aspectos como:

- Especificar datos con los que se va a operar
- Acciones que se realizarán

Según (Redator, 2019), el lenguaje de programación es la toda la base que permite el desarrollo de todo tipo de aplicaciones digitales alrededor de todo el planeta

2.3.1 *Lenguaje de bajo nivel*

Son lenguajes totalmente orientados a la máquina.

Este lenguaje crea un vínculo prácticamente inquebrantable entre el hardware y el software. Además, ejerce un control directo sobre el equipo y su estructura física. Para aplicarlo adecuadamente es necesario que el programador conozca sólidamente el hardware. (Redator, 2019).

Dicho lenguaje está cerca de la representación de la máquina, es decir, es menos legible para los humanos y requiere un mayor conocimiento de los detalles de la arquitectura de la computadora. Estos lenguajes se utilizan para crear sistemas operativos, controladores de dispositivos y programas de sistemas específicos que requieren un rendimiento máximo o un control preciso del hardware.

2.3.2 Lenguaje de alto nivel

El principal objetivo del lenguaje de programación del alto nivel es facilitar al programador utilizar instrucciones más sencillas de entender. El lenguaje de alto nivel se enfoca más en la lógica utilizada para realizar las acciones, además es independiente de la arquitectura del hardware que se está utilizando.

De este modo el lenguaje de alto nivel requiere menos conocimiento de arquitectura dada su independencia de la misma. Algunos ejemplos de este tipo de lenguaje son JAVA, Python y C.

2.4 IDE

Un entorno de desarrollo integrado (IDE) es un sistema de software para el diseño de aplicaciones que combina herramientas comunes para desarrolladores en una sola interfaz de usuario gráfica (GUI) (RedHat, 2019).

Generalmente, un IDE cuenta con las siguientes características:

- **Editor de código fuente**: editor de texto que ayuda a escribir el código de software con funciones como el resaltado de la sintaxis con indicaciones visuales, el relleno automático específico para el lenguaje y la comprobación de errores a medida que se escribe el código (RedHat, 2019).
- **Automatización de compilaciones locales**: herramientas que automatizan tareas sencillas y repetitivas como parte de la creación de una compilación local del software para su uso por parte del desarrollador, como la compilación del código fuente de la computadora en un código binario, el empaquetado de ese código y la ejecución de pruebas automatizadas (RedHat, 2019).
- **Depurador**: programa que sirve para probar otros programas y mostrar la ubicación de un error en el código original de forma gráfica (RedHat, 2019).

2.5 Software de programación de alto nivel

En esta sección se muestran algunos de los lenguajes de alto nivel utilizados actualmente, además de los correspondientes IDE que permiten el desarrollo de programas, esto se realiza con el fin de analizar algunas de las características.

2.5.1 C++

C++ es un lenguaje multiplataforma que se puede utilizar para crear aplicaciones de alto rendimiento. Fue desarrollado por Bjarne Stroustrup, como una extensión del lenguaje C. Además, brinda a los programadores un alto nivel de control sobre los recursos y la memoria del sistema (W3School, 2022).

C++ es un lenguaje que se puede utilizar en una gran cantidad de sistemas operativos actuales, además este puede utilizarse para desarrollar aplicaciones que se pueden adaptar a múltiples plataformas. Entre algunas de las aplicaciones de C++ se tiene:

- Navegadores WEB.
- Bases de datos.
- Sistemas Operativos.
- Videojuegos.
- Programación de dispositivos electrónicos.

2.5.2 Arduino IDE

Arduino IDE es el entorno de desarrollo integrado de Arduino que utiliza el lenguaje de programación C++, es una aplicación multiplataforma que facilita la escritura de código y la carga en diferentes placas que utilizan microcontroladores. Este software es posible utilizarlo con diferentes microcontroladores, entre ellas se incluyen las placas ESP32 y ESP8266

2.5.3 Java

Java es un lenguaje de programación y una plataforma informática lanzado por primera vez por Sun Microsystems en 1995. Ha evolucionado desde sus humildes comienzos hasta impulsar una gran parte del mundo digital actual, proporcionando la

plataforma confiable sobre la que se construyen muchos servicios y aplicaciones. Los productos nuevos e innovadores y los servicios digitales diseñados para el futuro también continúan confiando en Java (Java, 2022).

Existen muchas aplicaciones que se encuentran basadas en Java, al igual que C++ permite utilizarse para el desarrollo de aplicaciones.

2.5.4 Netbeans IDE

NetBeans IDE es un entorno de desarrollo integrado de código abierto y gratuito para el desarrollo de aplicaciones en los sistemas operativos Windows, Mac, Linux y Solaris. El IDE simplifica el desarrollo de aplicaciones web, empresariales, de escritorio y móviles que utilizan las plataformas Java y HTML5. Además, ofrece soporte para el desarrollo de aplicaciones PHP y C/C++ (Oracle, 2022).

NetBeans IDE ofrece herramientas de primer nivel para el desarrollo de aplicaciones móviles, de escritorio, empresariales y web Java. Es el primer IDE que admite las últimas versiones de JDK, Java EE y JavaFX. Brinda resúmenes inteligentes que le ayudan a comprender y administrar sus aplicaciones, incluida la compatibilidad inmediata con tecnologías populares, como Maven (Oracle, 2022).

Gracias a sus características de desarrollo de aplicaciones integrales, la mejora constante de Java Editor y las mejoras continuas de velocidad y rendimiento, NetBeans IDE marca el ritmo para el desarrollo de aplicaciones con tecnologías novedosas listas para usar (Oracle, 2022).

2.6 Bases de datos

Una base de datos es una recopilación organizada de información de cualquier índole, que es almacenada en servidores de información dedicados. Por lo general, las bases de datos utilizan un sistema de gestión de bases de datos (DBMS). En conjunto, la

información y el sistema de gestión de bases de datos, junto con las aplicaciones asociadas a ellos, reciben el nombre de sistema de bases de datos, abreviado normalmente a simplemente base de datos (Oracle Mexico, 2022).

Los datos de los tipos más comunes de bases de datos en funcionamiento actualmente se suelen utilizar como estructuras de filas y columnas en una serie de tablas para aumentar la eficacia del procesamiento y la consulta de datos. Así, se puede acceder, gestionar, modificar, actualizar, controlar y organizar fácilmente los datos. La mayoría de las bases de datos utilizan un lenguaje de consulta estructurada (SQL) para escribir y consultar datos (Oracle Mexico, 2022).

2.6.1 Base de datos en la Nube

Las bases de datos en la nube son servicios que de bases de datos que han sido creados basados en PaaS (Plataforma como servicios). Dicha base de datos, al estar alojada en la nube, permite el acceso desde cualquier lugar, siempre y cuando se tenga acceso a internet, es por ello que brinda una gran flexibilidad respecto a las bases de datos tradicionales (IBM, 2022).

2.6.1.1 Características de Bases de datos en la Nube

- Un servicio de base de datos al que se puede acceder a través de una plataforma en la nube
- Permite alojar bases de datos sin la necesidad de adquirir hardware dedicado
- Puede ser gestionado por el usuario o por un proveedor externo
- Admite bases de datos relacionales y bases de datos no relacionales

2.6.2 Base de datos relacionales

Una base de datos relacional es una recopilación de elementos de datos con relaciones establecidas entre dichos elementos. Estos elementos se organizan como un conjunto de tablas que utilizan filas y columnas (Amazon Web Services, 2022).

Las tablas, se utilizan para guardar información acerca de los elementos que se van a representar en la base de datos. Cada columna de la tabla guarda un tipo de datos y un campo almacena el valor real del atributo que se establece, por otro lado, las filas de la tabla creada, determina una recopilación de algunos valores relacionados de un objeto (Amazon Web Services, 2022).

2.6.2.1 SQL

SQL (Structured Query Language) es la interfaz principal utilizada para la comunicación con las bases de datos relacionales.

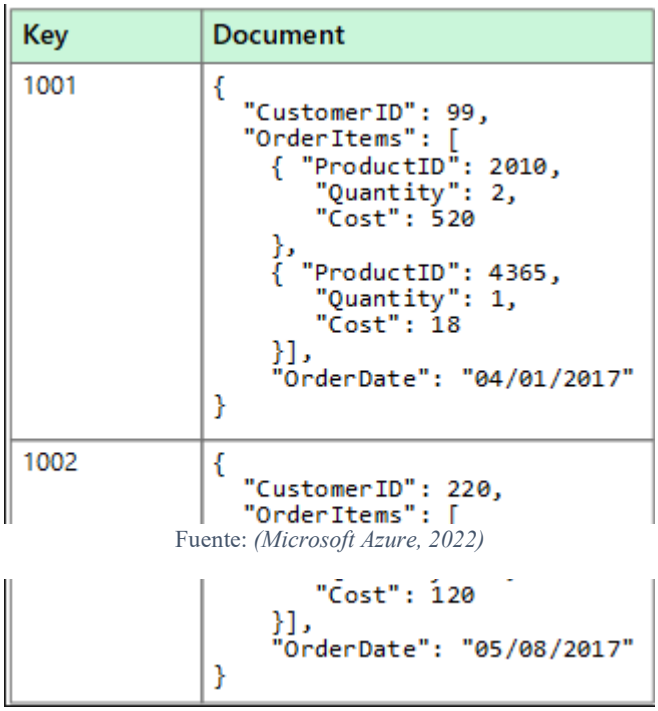
SQL se convirtió en un estándar de la ANSI (Instituto Nacional Estadounidense de Estándares) en el año de 1986. La mayoría de los motores de bases de datos admiten QL como estándar, ya que tiene funcionalidades como añadir, eliminar o actualizar filas de datos, por lo que es ideal cuando se requiere obtener conjuntos de datos para ciertas aplicaciones específicas (Amazon Web Services, 2022).

2.6.3 Base de datos no relacionales

Las bases de datos no relacionales son aquellas que no utilizan un sistema de almacenamiento mediante tablas, es decir filas y columnas. De este modo, los datos son almacenados de acuerdo con el tipo de dato que se está utilizando, un claro ejemplo de esto es que las bases de datos no relacionales son capaces de guardar archivos en formato JSON (Microsoft Azure, 2022).

Las bases de datos no relacionales también son conocidas como bases de datos NOSQL, pues dichas bases no utilizan SQL como lenguaje de consulta, en su lugar, se utilizan otros lenguajes de programación como son C++ o Python, aun así, existen bases de datos que son compatibles con la consulta SQL. En **Figura 5** es posible visualizar algunos ejemplos de cómo se almacena un documento en una base no relacional.

Figura *5*
Formato de documento en base de datos no relacional



Key	Document
1001	{ "CustomerID": 99, "OrderItems": [{ "ProductID": 2010, "Quantity": 2, "Cost": 520 }, { "ProductID": 4365, "Quantity": 1, "Cost": 18 }], "OrderDate": "04/01/2017" }
1002	{ "CustomerID": 220, "OrderItems": ["Cost": 120 }], "OrderDate": "05/08/2017" }

Fuente: *(Microsoft Azure, 2022)*

Las bases de datos no relacionales tienen una gran ventaja sobre las bases de datos relacionales, esta es la capacidad de manejo de grandes cantidades de datos que se generan rápidamente, un ejemplo claro de ello, son las redes sociales, pues es bien conocido que gran parte de la población hace uso de ellas, lo que a su vez genera datos de forma masiva.

2.6.3.1 MongoDB

MongoDB, es una base de datos de documentos fácilmente escalable, que utiliza un modelo de consultas avanzado. Mongo DB es una base de datos que puede utilizarse para cualquier propósito, sin embargo, las principales aplicaciones en las que se utiliza MongoDB, corresponden al desarrollo de aplicaciones WEB y aplicaciones de entornos móviles (MongoDB, Inc, 2022).

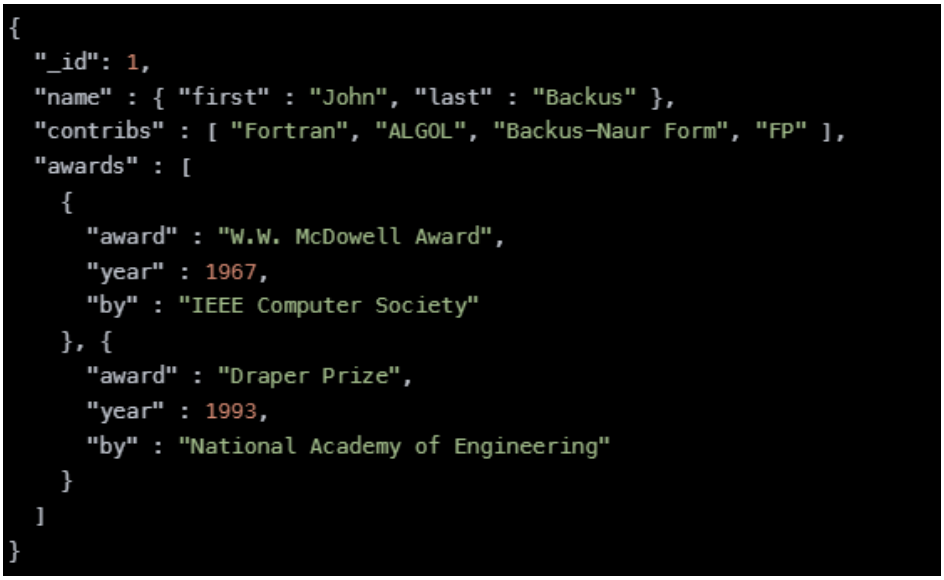
MongoDB permite una gran escalabilidad, esto proporciona una facilidad gigante a la hora de utilizar los recursos, puesto que, si se requiere un mejor procesamiento de los datos, no es necesario cambiar toda la infraestructura, bastará con interconectar la nueva infraestructura a la antigua y esto añadirá recursos a la base de datos, este arreglo de hardware se lo conoce como clúster. Esto garantiza actualizaciones sin necesidad de realizar grandes inversiones económicas.

El formato de documentos que utiliza MongoDB es el formato BSON, de este modo, MongoDB garantiza que exista una mejor velocidad a la hora de realizar consultas, solucionando uno de los inconvenientes que se tienen al utilizar el formato JSON (MongoDB, Inc, 2022).

2.6.3.2 JSON

JSON (*JavaScript Object Notation)* es un formato de intercambio de datos. JSON es capaz de almacenar datos mediante contenedores asociativos, en los que una clave de cadena es asignada a un valor, que puede ser un número, una cadena o incluso otro objeto. Esto permite que los objetos de JavaScript se representen de manera sencilla en el texto y se visualiza en la **Figura 6** (MongoDB,Inc, 2022).

Figura *6*
Ejemplo de formato JSON



Fuente: *(MongoDB, Inc, 2022)*

Al ser un formato fácilmente entendible tanto para máquinas y humanos y de sencilla implementación incluso con otros lenguajes, JSON se convirtió en uno de los formatos para almacenar objetos más usados a nivel global (MongoDB,Inc, 2022). Actualmente, JSON se utiliza en diferentes aplicaciones como:

- API
- Archivos de configuración
- Registrar mensajes
- Almacenamiento de bases de datos

2.6.3.3 BSON

BSON simplemente significa "JSON binario", y eso es exactamente lo que se inventó para ser. La estructura binaria de BSON codifica información de tipo y longitud, lo que permite analizarla mucho más rápidamente. Desde su formulación inicial, BSON se ha ampliado para agregar algunos tipos de datos no nativos de JSON opcionales, como

fechas y datos binarios, sin los cuales a MongoDB le habría faltado un soporte valioso. (MongoDB,Inc, 2022).

A continuación, en la **Figura** *7* se presenta una comparativa entre JSON y BSON proporcionada por (MongoDB,Inc, 2022).

Figura 7
Comparativa JSON y BSON

	JSON	BSON
Codificación	Cadena UTF-8	Binario
Soporte de datos	Cadena, booleano, número, matriz	Cadena, booleano, número (entero, flotante, largo, decimal 128..), matriz, fecha, binario sin formato
Legibilidad	Humano y Máquina	Solo máquina

Fuente: *(MongoDB, Inc, 2022)*.

Una forma particular en la que BSON se diferencia de JSON es en su compatibilidad con algunos tipos de datos más avanzados como se puede visualizar en la **Figura** *7* JavaScript no diferencia, por ejemplo, entre números enteros y números de coma flotante (MongoDB,Inc, 2022).

2.7 Metodología en Espiral

El modelo de desarrollo Espiral es una combinación entre dos modelos, concretamente el modelo de cascada y el modelo basado en iteraciones. El proceso básicamente consiste en definir 4 pasos para cada etapa que se va a cumplir, en este caso se tienen 4 etapas que se muestran en la **Figura 8**.

Figura *8*
Metodología en Espiral

Fuente: (Corvo, 2020)

2.7.1 Análisis

La primera etapa del modelo consiste en definir los objetivos que se pretenden alcanzar, en este caso se deberá establecer con mayor detalle, el hardware y software que se va a utilizar, así como también los requerimientos que tendrá dicho sistema (Corvo, 2020).

2.7.2 Evaluación de Riesgos

En esta etapa se evalúan algunas alternativas que pueden ayudar a cumplir con el objetivo que se ha planteado. “Además, se deberán identificar los riesgos que puede tener cada una de las alternativas contempladas” (Corvo, 2020).

En esta etapa también es posible utilizar prototipos para tener una mejor visión del sistema a desarrollar.

2.7.3 Desarrollo y prueba

En esta etapa se realiza todo el desarrollo de lo propuesto buscando cumplir con el objetivo, de esta manera se realizar pruebas para verificar que objetivo se haya cumplido alcanzando el resultado deseado (Corvo, 2020).

2.7.4 Planificación

Finalmente, una vez se haya completado el objetivo anteriormente planificado, se deberá realizar la planificación del siguiente objetivo que se pretende cumplir, esto siempre y cuando el objetivo anterior se haya cumplido, sin embargo, si la situación amerita realizar cambios en el objetivo no cumplido, se deberá iniciar nuevamente para alcanzar dicho objetivo (Corvo, 2020).

Capítulo III. Diseño e implementación

El capítulo 3 presenta el diseño y la implementación del prototipo a desarrollar, para cumplir con los objetivos propuestos, utilizando la metodología de espiral que se menciona en la sección 2.7 del capítulo 2, de esta manera, se contemplan algunos de los requerimientos del sistema que posteriormente serán evaluados mediante un análisis de riesgos para tener una visión de los componentes a utilizar, una vez elegido los componentes, el siguiente paso es desarrollar y realizar las pruebas correspondientes del sistema para corregir los posibles errores que se puedan presentar.

3.1 Situación Actual

Actualmente la Carrera de Ingeniería en Telecomunicaciones (CITEL), anteriormente Carrera de Ingeniería en Electrónica y Redes de Comunicación (CIERCOM) la cual se encuentra habilitada solo para la emisión de títulos, posee alrededor de 300 estudiantes de acuerdo con el número de matriculados en el periodo septiembre 2022 – febrero 2023, comprendidos entre las dos carreras.

La Carrera de Ingeniería en Telecomunicaciones, cuenta con una asociación de estudiantes, la cual está comprendida por: presidente, vicepresidente, secretario y tesorero.

La asociación de estudiantes tiene como principal objetivo apoyar a los estudiantes durante la estancia en la universidad, de esta manera promueve el desarrollo de actividades culturales, deportivas y académicas para los estudiantes.

Para cumplir sus cometidos, trabaja en conjunto con los representantes de cada nivel, dichos representantes son escogidos de manera democrática por parte de los estudiantes que correspondan a cada uno de los niveles.

La asociación de estudiantes cumple algunas funciones adicionales, como puede ser la recepción de pago voluntario para la realización de actividades culturales, préstamo de artículos que posee la asociación, realizar convenios con tiendas en beneficio de estudiantes. Así se ha visto la posibilidad de realizar mejoras en dichas funciones mediante el uso de la tecnología NFC. Para ello inicialmente se realiza una entrevista a la presidenta con el fin de establecer las necesidades actuales y suplir las mismas. En este sentido, a continuación, se presenta la entrevista realizada:

	UNIVERSIDAD TÉCNICA DEL NORTE **CARRERA DE INGENIERÍA EN TELECOMUNICACIONES**
PROYECTO: Diseño de un prototipo que unifique, un sistema de pago basado en recargas, un sistema de control de inventario y un sistema de control de acceso, mediante la tecnología Near Field Connection (NFC) para la asociación de estudiantes CITEL-CIERCOM.	
Fecha de Entrevista:	**16-11-2022**
Organización:	**ASOCIACIÓN ESTUDIANTES CARRERA INGENIERÍA EN TELECOMUNICACIONES**
Entrevistado/a:	**KATHERINE ALMEIDA**
Cargo:	**PRESIDENTA DE LA ASOCIACIÓN**
INTRODUCCIÓN	

La presente entrevista tiene como fin el levantamiento de información de algunos procesos que se dan dentro de la asociación de estudiantes de la Carrera de Ingeniería en Telecomunicaciones de la Universidad Técnica del Norte, con el fin de establecer necesidades que serán resueltas con el desarrollo del presente trabajo de titulación.

PREGUNTAS:

1. **¿Qué problemas se presentan en cuanto al inventario que posee la asociación?**

Actualmente en la asociación, el inventario se lo ha llevado sin ningún tipo de registro, de tal manera que existen varios artículos que se han extraviado.

La asociación busca realizar en un futuro préstamo de artículos que puedan utilizar los estudiantes de la carrera de ingeniería en telecomunicaciones, estos pueden ser arduinos, cables UTP, calculadoras, cargadores, cables micro USB, cables tipo C, protoboards. De esta manera aquellos estudiantes que requieran podrán hacer uso de dichos artículos.

2. **¿Qué problemas se presenta de acuerdo con el acceso a las instalaciones de la asociación CITEL-CIERCOM?**

Dentro de la asociación, es posible visualizar que no se tiene un control de quien ingresa a la asociación, de este modo, la seguridad de los artículos que se encuentran dentro se puede ver vulnerada.

Actualmente tienen acceso los miembros de la asociación y los dirigentes de cada nivel. Por lo que esto se debe considerar.

3. **¿Cuál es la situación actual respecto al valor de aporte voluntario que realizan los estudiantes?**

Cada vez que se realizan las matrículas, se realiza un pago de aportación para la organización de eventos durante el semestre, sin embargo, esto se lo hace de manera tradicional y son los dirigentes de la asociación quienes deben estar pendientes de que se realice el pago, de este modo llevar el registro de todos los que han pagado puede resultar tedioso de realizar.

4. **¿Cuál es la situación actual respecto a la tarjeta de copias que se asignado a cada uno de los estudiantes de la carrera?**

Existió un acuerdo hablado entre la copiadora de la FICA y la asociación, en la que cada estudiante tenía asignado un número de copias gratuitas, para ello se entregaba una tarjeta de cartulina con el número de copias asignadas y se iba marcando cada que se hacía uso de las mismas, sin embargo, no todos los estudiantes hacían uso de la cartulina por diferentes motivos.

La asociación habría pagado por cada uno de los estudiantes una cantidad de dinero previamente, por lo que, al no llevar un registro exacto, se habría pagado más de lo que se había consumido por parte de los estudiantes.

5. **¿Estaría interesada en utilizar un sistema basado en NFC, que permita gestionar algunos de los problemas mencionados de manera más eficiente?**
   Si

6. **¿Cuáles son algunos de los requerimientos que se consideran necesarios para el desarrollo de un sistema NFC?**

El sistema debería solventar los problemas que se han mencionado antes, de esta manera el sistema deberá contar con 4 apartados fundamentales.

- **Sistema de control de acceso**
  Requerimientos:

  El sistema de control de acceso mediante NFC, será capaz de generar un registro de cada persona que ingrese a la asociación.

  El sistema deberá permitir que se pueda agregar usuarios para que ingresen en la asociación de manera sencilla

  El sistema solo se abrirá con aquellos usuarios miembros seleccionados.

- **Sistema de inventario**
  Requerimientos:

  El sistema deberá ser capaz de llevar un registro de todos los artículos que se vayan a prestar a los estudiantes.

El sistema permitirá tener un registro con hora y fecha de cuando ha sido prestado el articulo

Aquellos que pueden prestar artículos dentro de la asociación, son los dirigentes de cada nivel y la directiva de la asociación.

El sistema deberá registrar el usuario quién ha prestado el articulo y el usuario a quien se le presta.

- **Sistema de micro transacciones.**
  Requerimientos:

  En el sistema de micro transacciones, se deberá tener un registro de todas las transacciones que se realicen.

  Los usuarios de este sistema serán todos los estudiantes de la carrera de ingeniería en telecomunicaciones

  El sistema NFC podrá reemplazar al sistema de tarjetas de cartulina que se utilizaba anteriormente.

- **Sistema de pago de aportación para eventos**
  Requerimientos:

  Los usuarios de este sistema serán todos los estudiantes de la carrera CITEL-CIERCOM

  El sistema permitirá gestionar un registro de todos aquellos estudiantes que haya realizado el pago y de aquellos que no.

Una vez finalizada la encuesta se determina los requerimientos de usuario, sin embargo, es indispensable contar también con requerimientos técnicos que tendrá el sistema, dichos requerimientos se mencionan a continuación.

- El sistema deberá funcionar las 24 horas del día.
- Uso mediante Microcontroladores
- El sistema necesita de una fuente de alimentación de 5v.

- Requiere internet para su correcto funcionamiento.
- Base de datos de gran capacidad.
- El tamaño del prototipo deberá ser pequeño.
- Los costos del hardware no deben ser elevados.

De este modo se definen los parámetros considerados para cumplir con el desarrollo del proyecto presentado.

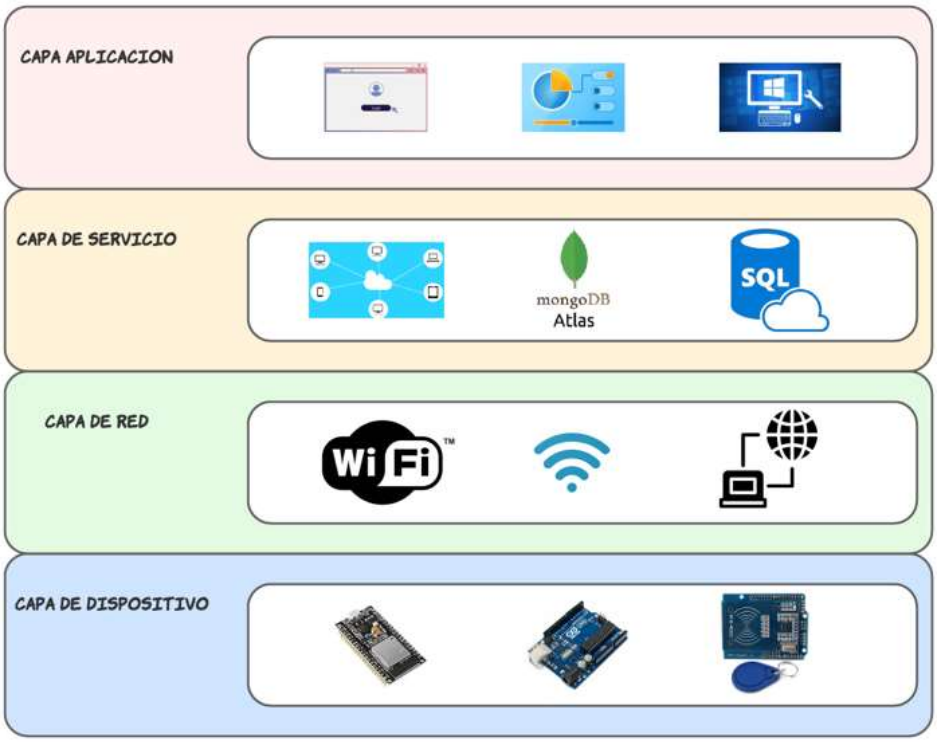
3.2 Modelo de Referencia IoT aplicado al sistema a desarrollar

El modelo de referencia IoT es aplicable a diferentes sistemas, en este apartado se muestra como el modelo de referencia IoT se aplica al presente proyecto, de esta manera se describen cada una de las capas y posteriormente se muestra la **Figura *9*** en la que se visualizan las mismas de manera detallada.

- **Capa de dispositivo:** En esta primera capa contempla los elementos físicos tales como microcontroladores, sensores, detectores, etc. De este modo, en el sistema propuesto, se contempla el microcontrolador, el *NFC Shield,* alimentación y demás elementos electrónicos.
- **Capa de red:** Es la infraestructura que consiste en la comunicación ya sea de forma inalámbrica o alámbrica. Para el caso del sistema a desarrollar, en esta capa se considera la transmisión de datos Wi-Fi, este será vital, ya que permite el acceso del microcontrolador a la base de datos.
- **Capa de servicio:** Esta capa es aquella que contempla la computación en la nube o el uso de servidores, para el caso del sistema desarrollado, esta capa hace referencia a la computación en la nube en la que estará almacenada la base de datos.

- **Capa de aplicación:** Es aquella capa en la que se presentan los datos y se realizan las diferentes acciones, en el sistema propuesto, esta capa contempla el desarrollo de una aplicación para sistemas operativos Windows utilizando diferentes lenguajes de programación.

Figura 9
Modelo de referencia IoT aplicado al sistema

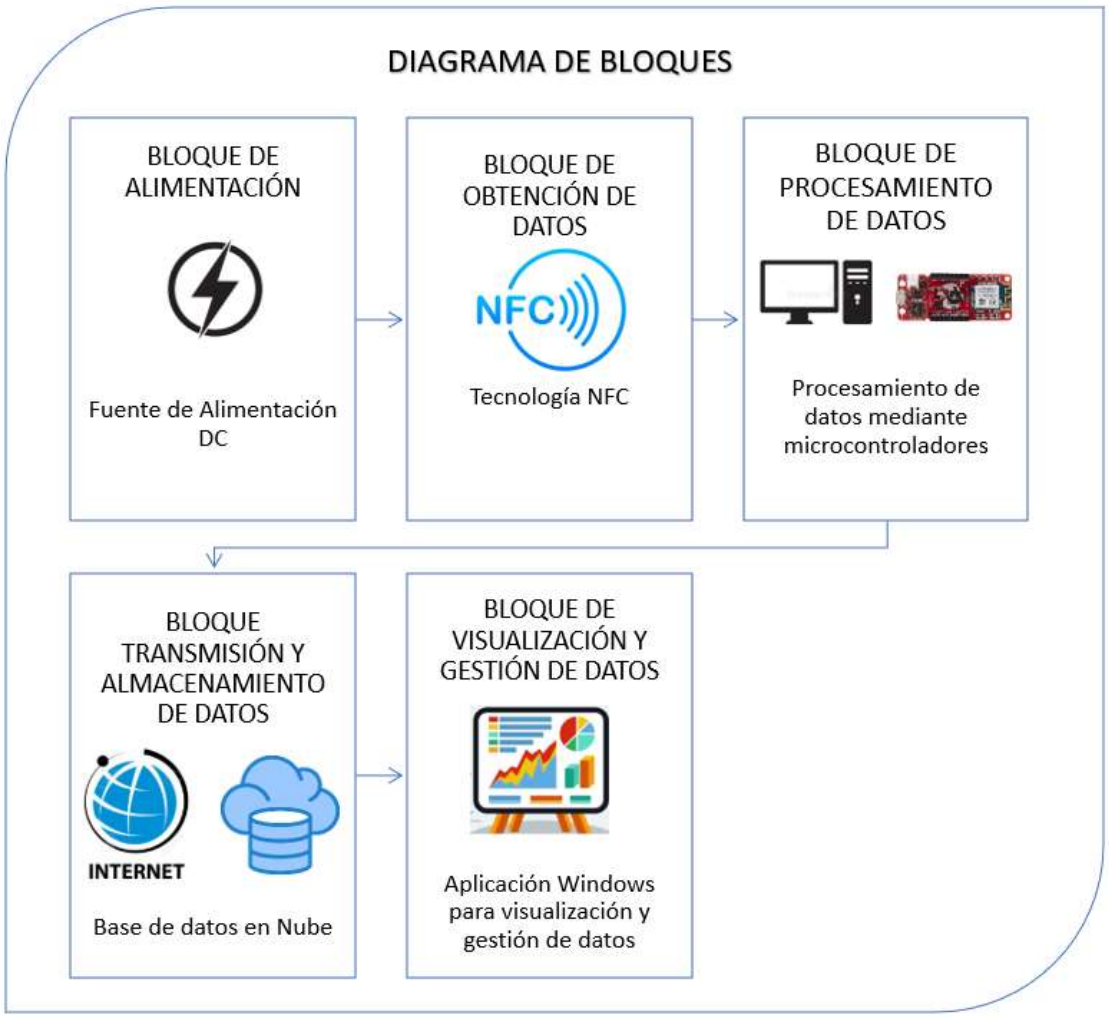


Nota: Cada capa del modelo permite gestionar de mejor manera las funciones del sistema

3.3 Diagrama de Bloques

La **Figura 10** muestra el diagrama de interacción propuesto entre los diferentes componentes, de esta manera, se tienen 5 bloques fundamentales que serán de vital importancia para desarrollar el proyecto y la selección tanto de hardware como de software.

Figura *10*
Diagrama de bloques propuesto



Una vez visualizado el diagrama de bloques en el que se basa el prototipo, se describe el funcionamiento de manera detallada de cada uno de los bloques de este.

- **Bloque de Alimentación. –** Este bloque es el encargado de suministrar energía eléctrica para el funcionamiento del prototipo a desarrollar. De esta manera, un elemento crucial de dicho bloque será la fuente de alimentación.
- **Bloque de Obtención de datos. –** Dicho bloque contempla la obtención de datos realizada mediante tecnología NFC que será enviada desde la tarjeta NFC hacía el lector NFC para su posterior procesamiento.

- **Bloque de Procesamiento de datos. –** En este bloque se tiene el dispositivo encargado de procesar los datos que hayan sido obtenidos mediante el lector NFC, de esta manera, se consideran dos maneras de procesar los datos obtenidos, para el desarrollo del sistema de control de acceso, los datos se procesarán dentro de un microcontrolador, sin embargo, para los otros sistemas planteados el procesamiento deberá realizarse en la PC que se ejecute el programa.
- **Bloque de transmisión y almacenamiento de datos. –** Dicho bloque permite el envío y almacenamiento de datos obtenidos y procesados hacia una base de datos que se encuentra alojada en la nube, con el fin de facilitar la visualización y gestión de estos.
- **Bloque de visualización y gestión de datos. –** Este bloque permitirá visualizar todos los datos que hayan sido procesados y enviados a la base de datos que se encuentra alojada en la nube, para ello es necesario el desarrollo de una aplicación de escritorio para sistemas operativos Windows.

3.4 Elección de Hardware y Software

En la elección de hardware y software se detallan las diferentes características tanto de hardware como software que serán evaluadas y con ello realizar una elección que se adecue a los requerimientos, de este modo, inicialmente se realiza la **Tabla 1** en la que se definen los diferentes requerimientos con las respectivas abreviaturas.

Tabla 1
Abreviatura Requerimientos Hardware y Software

Requerimientos	Abreviatura
Requerimientos de Tarjeta Lectora	**RHWTL**
Requerimientos de microcontrolador	**RHWMCU**
Requerimientos lenguaje de programación microcontrolador	**RSWMCU**
Requerimientos lenguaje de programación aplicación	**RSWAPP**

Requerimientos Base de Datos	**RSWDB**

3.4.1 Tarjeta Lectora NFC.

Existen diferentes alternativas que nos permiten dotar de comunicación NFC, entre las principales se tienen las tarjetas PN532 y la tarjeta RC522. De esta manera, los parámetros que serán considerados a la hora de realizar la elección se definen en la **Tabla *2***.

Tabla 2
Requerimientos Hardware Tarjeta Lectora

Nomenclatura	Requerimiento	Prioridad
RHWTL1	Frecuencia de Operación de conectividad Inalámbrica	Alta
RHWTL2	Voltaje de Operación del dispositivo	Media
RHWTL3	Tasa de transferencia de datos	Alta
RHWTL4	Estándares Soportados por el dispositivo	Alta
RHWTL5	Dimensiones de tamaño que tiene el dispositivo	Media
RHWTL6	Precio del dispositivo	Alta

Una vez se han definido los parámetros que se consideran para la elección de la tarjeta lectora NFC, se desarrolla la **Tabla 3** con las especificaciones de cada una de las tarjetas nombradas anteriormente.

Tabla 3
Análisis comparativo de especificaciones técnicas lectores NFC

COMPARATIVA ESPECIFICACIONES TÉCNICAS LECTORES NFC		
Módulo	**RC522**	**PN532**
DISEÑO		
RHWTL1	13,56Mhz	13,56Mhz
RHWTL2	3.3v DC	3.3 – 5v DC
RHWTL3	Max. 10Mbit/s	Max. 10Mbit/s
RHWTL4	Mifare1 S50, Mifare 1K, MIFARE Ultralight, Mifare Pro, Mifare DESFire.	Mifare1 S50, Mifare 1K, Mifare Ultralight, Mifare Pro, Mifare DESFire, Mifare Classic.
RHWTL5	40 mm x 60 mm	43 x 40 mm
RHWTL6	5,90 $- en MercadoLibre	14,00$ en MercadoLibre

Fuente: Adaptado de (Naylampmechatronics, 2021).

Una vez se tienen todos los parámetros necesarios visualizados en la Tabla 3, se realiza una tabla de puntuación en la que se considera como principales requerimientos a solventar aquellos que se encuentran calificados con prioridad alta, de este modo se obtiene la **Tabla 4**.

Tabla 4
Elección Hardware Modulo Lector

Hardware	**Requerimientos**				**Puntuación**
	RHWTL1	**RHWTL3**	**RHWTL4**	**RHWTL6**	
RC522	2	2	2	2	8
PN532	2	2	2	1	7
	2 excelente	**1 bueno**	**0 insuficiente**		

Finalmente se selecciona el módulo lector RC522 de acuerdo con la **Tabla 4** debido a que es quien obtiene la mejor calificación de acuerdo con los requerimientos planteados, a continuación, se muestra la elección del microcontrolador.

3.4.2 Microcontrolador

Para la unidad encargada del procesamiento de datos también se debe realizar una comparativa, en este caso se tiene en consideración algunos módulos tales como: Arduino Uno, Arduino Nano, ESP32. Los parámetros esenciales a la hora de realizar la elección del microcontrolador se muestran en la **Tabla *5***.

Tabla 5
Requerimientos de Hardware del Microcontrolador

Nomenclatura	Requerimiento	Prioridad
RHWMCU1	Frecuencia del procesador del microcontrolador	Alta
RHWMCU2	Voltaje de Operación del dispositivo	Media
RHWMCU3	Cantidad de Memoria RAM	Alta
RHWMCU4	Cantidad de Memoria Flash	Alta
RHWMCU5	Dimensiones de que posee el microcontrolador	Media
RHWMCU6	Tipo de Conectividad Soportada	Alta
RHWMCU7	Número de pines que posee	Alta
RHWMCU8	Precio en el mercado	Media

Una vez definidos los diferentes requerimientos establecidos para la selección del hardware, se procede a comparar las diferentes alternativas existentes en el mercado local. La **Tabla 6** muestra las especificaciones de las diferentes opciones de microcontrolador.

Tabla 6
Comparativa de especificaciones de microcontroladores

COMPARATIVA MICROCONTROLADORES			
Modulo	**Arduino UNO R3**	**ESP32**	**ARDUINO NANO**
DISEÑO			
RHWMCU1	16Mhz	160Mhz	16Mhz
RHWMCU2	2.7- 5.5v DC	3.3 – 5v DC	5v DC
RHWMCU3	2Kb	520Kb	1Kb
RHWMCU4	32Kb	16Mb	32Kb
RHWMCU5	80 x 55,1mm	51 x 23mm	44 x 18 mm
RHWMCU6	USB tipo B	Wifi, Bluetooth, Micro USB	Mini USB
RHWMCU7	70 pines	30 pines	22 pines
RHWMCU8	15,00 $- en MercadoLibre	14,00$ en MercadoLibre	12,50$ en MercadoLibre

Fuente: Adaptado de (Domínguez, 2017)

Al obtener las especificaciones, se realiza una tabla comparativa para realizar la mejor elección del hardware, de esta manera, se contemplan todos los requerimientos que tienen alta prioridad y se muestran en la **Tabla** *7*.

Tabla 7
Elección Hardware Microcontrolador

Hardware	**Requerimientos**					**Puntuación**
	RHWMCU1	**RHWMCU3**	**RHWMCU4**	**RHWMCU6**	**RHWMCU7**	

Arduino UNO	1	1	1	0	2	5
ESP32	2	2	2	2	1	9
Arduino Nano	1	0	1	0	1	3
		2 excelente	**1 bueno**	**0 insuficiente**		

3.4.3 Lenguaje de programación del microcontrolador.

Para programar todas las instrucciones que debe realizar el microcontrolador, se realiza mediante un lenguaje de programación, para ello, las opciones principales están dadas por Arduino y MicroPython, estos lenguajes de programación están orientados a microcontroladores, por lo que son viables para el presente proyecto. Los parámetros que serán contemplados para la selección del lenguaje de programación se definen en la **Tabla *8***.

Tabla 8
Requerimientos Lenguaje Programación

Nomenclatura	Requerimiento	Prioridad
RSWMCU1	Compatibilidad con microcontroladores ESP	Alta
RSWMC2	Tipo de Licencia de software que posee	Media
RSWMCU3	Posee Librerías compatibles con el módulo NFC	Alta
RSWMCU4	Documentación y Soporte	Media

Una vez definidos los parámetros correspondientes a la selección del lenguaje de programación, se realiza la comparativa de acuerdo con los mismos y se muestra en la**Tabla *9***.

Tabla 9
Comparativa entre software para microcontroladores

COMPARATIVA ARDUINO y MICROPYTHON		
Software	**Arduino**	**MicroPython**
Logo		
Compatibilidad Microcontroladores	Arduino Uno, Arduino Mega, Arduino Nano, ESP32, ESP8266	ESP32, ESP8266, Rasperry Pico, Adafruit's Circuit
Software Libre/Pagado	Software Libre	Software Libre
Librerías NFC	Si	Si
Documentación y Soporte	Foros, página oficial, videos, libros.	Foros, página oficial, videos.

Fuente: Adaptado de (Arduino, 2019)

De acuerdo con la **Tabla 8**, los principales requerimientos que deberán ser solventados son aquellos que se marcan con prioridad alta, sin embargo, como se puede visualizar en la **Tabla *11***, los parámetros son similares, por lo que se procede a escoger de acuerdo a la prioridad media, en este caso, se visualiza que como prioridad media se tiene el requerimiento de soporte, por lo que Arduino al tener un mejor soporte será la opción más adecuada.

3.4.4 Lenguaje de programación para el desarrollo de la aplicación Windows.

Con el fin de garantizar el funcionamiento óptimo del sistema y cumplir con el objetivo de automatizar diferentes procesos, es necesario el desarrollo de una aplicación de escritorio para Windows, para esto es necesario hacer uso de lenguaje de programación.

Entre los más usados se tienen C++, JAVA y recientemente HTML5, por lo que la selección de software estará contemplada entre dichos lenguajes. Los parámetros que se deberán contemplar para realizar la elección se muestran en la **Tabla *10***.

Tabla 10
Requerimientos del lenguaje de programación de aplicación

Nomenclatura	Requerimiento	Prioridad
RSWAPP1	Tipo de Licencia de software utilizado	Alta
RSWAPP2	Uso específico del software	Media
RSWAPP3	Documentación y Soporte	Media

Al definirse los parámetros de evaluación, se elabora la **Tabla *11*** , la cual muestra las diferentes características que poseen los lenguajes de programación que han sido considerados para realizar una elección de acuerdo con las necesidades del proyecto planteado.

Tabla 11
Análisis comparativo de lenguajes de programación para aplicaciones de escritorio

COMPARATIVA C++, JAVA, HTML			
Software	**C++**	**JAVA**	**HTML5**
RSWAPP1	Software Libre	Software Libre	Software Libre
RSWAPP2	Usado en la programación de sistemas operativos	Desarrollo de aplicaciones de escritorio y aplicaciones móviles.	Utilizado en el desarrollo de aplicaciones y páginas web

RSWAPP3	Foros, Página Oficial, Libros	Foros, Página Oficial, Libros	Foros, Página Oficial, Libros

Fuente: Adaptado de (Bhardwaj, s.f.)

De este modo, se visualiza que, dado que la prioridad alta para la elección de software es similar en los casos a analizar, se procede a realizar la elección basado en los requerimientos de prioridad media, en este caso la orientación de desarrollo del proyecto la cual está establecida en una aplicación de escritorio para Windows permite que sea más favorable la opción de JAVA como mejor alternativa.

3.4.5 Base de Datos usada en el sistema

La base de datos es especialmente útil, ya que permite como su nombre lo dice almacenar todo tipo de datos que serán necesarios para el correcto funcionamiento del sistema, para este apartado inicialmente se consideran opciones como MySQL, MongoDB y PostgreSQL. Para evaluar y realizar la elección de base de datos adecuada se definen los parámetros en la **Tabla *12***.

Tabla 12
Requerimientos de base de datos

Nomenclatura	Requerimiento	Prioridad
RSWDB1	Tipo de base de datos	Media
RSWDB2	Lenguaje utilizado para realizar consultas	Media
RSWDB3	Tipo de Escalabilidad	Media
RSWDB4	Posee servicio en la nube	Alta
RSWDB5	Beneficios adicionales que ofrece la base de datos	Media

Posteriormente se realiza la comparativa entre las diferentes bases tomando en consideración los requerimientos mencionados en la **Tabla *12*** y como resultado, se obtiene la **Tabla *13***.

Tabla 13
Comparativa Bases de Datos

COMPARATIVA MYSQL, POSTGRESQL y MONGODB			
RSWDB1	**MYSQL**	**POSTGRESQL**	**MONGODB**
RSWDB2	Relacional	Relacional	No Relacional
RSWDB3	SQL	SQL	JavaScript
RSWDB4	Vertical	Vertical	Horizontal
RSWDB5	Si (Aplicaciones Externas)	Si (Aplicaciones Externas)	Si (Aplicación Propietaria)
RSWDB6	Amplia Compatibilidad	Soporte para datos personalizados	Consultas eficientes

Fuente: Adaptado de (Kuzmenko, 2022)

Una vez presentadas las diferentes características, dado que la elección se realiza en base a la prioridad más alta, MongoDB es la opción más adecuada ya que posee un servicio en la nube propietario denominado MongoDB Atlas, de modo que la implementación de servicios en la nube es más sencilla, además de que uno de los principales beneficios es el rendimiento en las consultas, lo que lo hace ideal para aplicaciones en las que se busca obtener el mejor rendimiento posible. De este modo se culmina la elección de cada uno de los elementos en los que se basará el presente proyecto, con ello se continúa con el análisis de riesgos que puede representar cada una de las decisiones tomadas en la elección de hardware y software y también se considera

las posibles dificultades a la hora de realizar la implementación dentro de la Asociación de estudiantes CITEL-CIERCOM.

3.5 Análisis de Riesgos

El análisis de riesgos da una mejor visión de los problemas que se pueden presentar al momento de desarrollar el proyecto, de este modo, se realiza un análisis de riesgos para hardware que será detallado en la **Tabla 14**.

Tabla 14
Análisis de Riesgos de Hardware Seleccionado

ANÁLSIS DE RIESGOS HARDWARE

COMPONENTE	RIESGO	IMPACTO	MITIGACIÓN
ESP32	Precio elevado	MEDIO	Buscar en diferentes sitios web
	Disponibilidad Local	MEDIO	Adquirir en otros lugares (Quito, Guayaquil)
	Compatibilidad de librerías	BAJO	Existen librerías actuales
Módulo RC522	Soporte	MEDIO	Búsqueda de soporte en foros y paginas oficiales
	Disponibilidad Local	MEDIO	Es posible adquirirlo en ciudades como Quito o Guayaquil

Una vez se realiza el análisis de riesgos de hardware, se procede a desarrollar de igual manera, para el apartado del software, en él se analizan los riesgos referentes al software seleccionado y se muestra la **Tabla *15***.

Tabla 15
Análisis de Riesgos Software

ANÁLSIS DE RIESGOS SOFTWARE

COMPONENTE	RIESGO	IMPACTO	MITIGACIÓN
Arduino IDE	Incompatibilidad con MongoDB	MEDIO	Usar API dedicada de MongoDB
	Lenguaje de programación utilizado	BAJO	Lectura de fuentes oficiales
	Compatibilidad de serie con ESP32	BAJO	Instalar librerías para ESP32
Netbeans (JAVA)	Incompatibilidad con ESP32	MEDIO	Instalar librerías para comunicación mediante comunicación serial
	Lenguaje de programación utilizado	MEDIO	Búsqueda de información en diferentes foros
	Soporte	MEDIO	Uso de foros oficiales de JAVA
MongoDB Atlas	Espacio Gratuito Limitado	MEDIO	Adquirir más almacenamiento mediante suscripción
	Compatibilidad con el sistema	MEDIO	Información relacionada
	Se requiere de una cuenta para su uso	BAJO	Se crea una cuenta específica para el sistema

Al finalizar el análisis de riesgos, se determina que las opciones de hardware y software seleccionadas son viables a la hora de desarrollar el proyecto planteado, de esta

manera se inicia con el desarrollo de los sistemas de manera individual de acuerdo con la metodología planteada.

3.6 Desarrollo del prototipo de control de acceso

En esta sección se desarrolla el sistema correspondiente al control de acceso, de este modo, el control de acceso estará asociado principalmente con el ESP32 y el lector NFC, además se debe unificar hardware y software de tal manera que el sistema sea capaz de obtener y enviar datos desde la base de datos y con ello cumplir con los requerimientos que se mencionan anteriormente. Finalmente se complementa el sistema mediante una cerradura electrónica que será conectada al dispositivo ESP32. Con ello, inicialmente se muestra la arquitectura que va a utilizar el prototipo del sistema planteado con el que se da un primer vistazo al funcionamiento.

3.6.1 Arquitectura del prototipo del sistema de control de acceso

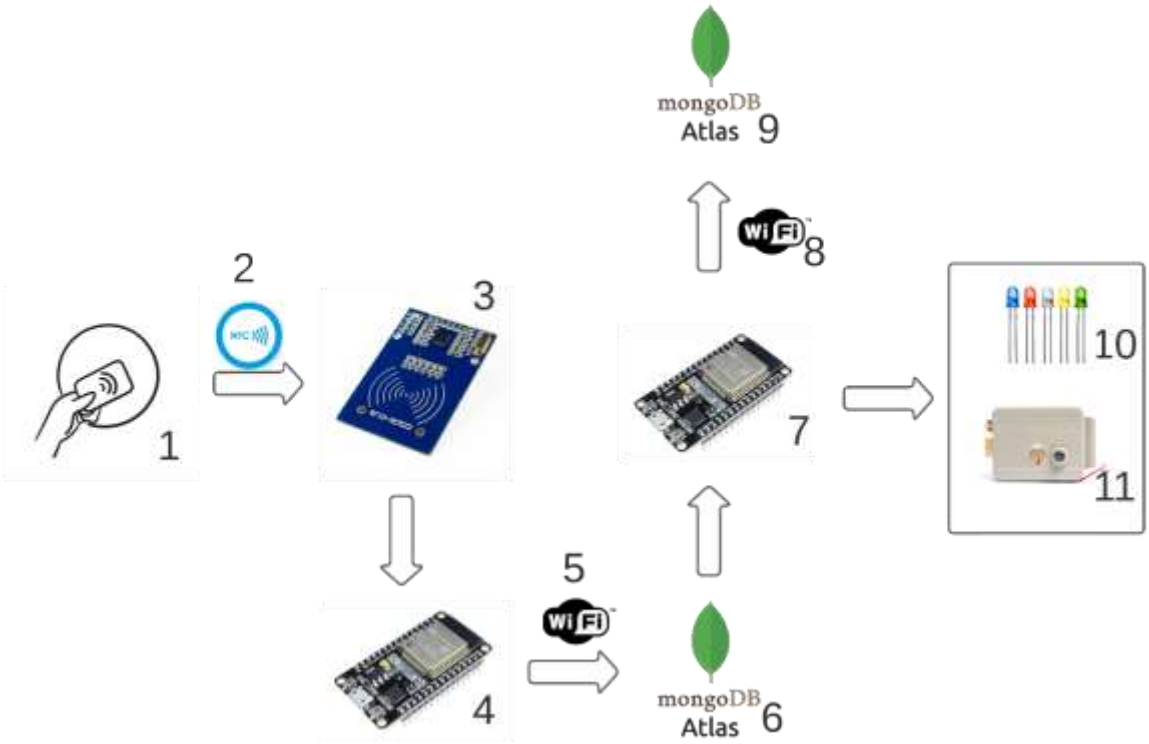
Los elementos que forman parte de la arquitectura del sistema propuesto marcan un orden a seguir de acuerdo con el funcionamiento que cumplen, de este modo se visualiza los 11 elementos que se consideran a continuación:

1. Tarjeta NFC 13,56Mhz
2. Comunicación Inalámbrica NFC
3. NFC Shield
4. Microcontrolador ESP32
5. Comunicación inalámbrica Wi-Fi
6. Base de datos en la nube MongoDB ATLAS
7. Microcontrolador ESP32
8. Comunicación inalámbrica Wi-Fi
9. Base de datos en la nube MongoDB ATLAS
10. Leds Indicadores

11. Cerradura Eléctrica

El esquema que se presenta en la **Figura 11**, es una representación en la que se especifican los componentes y como se interconectan entre ellos para conformar el sistema funcional.

Figura *11*
Diagrama de arquitectura del control acceso



Una vez se tiene la arquitectura, se describe los cada uno de los elementos y la función que cumplen dentro de la arquitectura, de esta manera se tiene un orden que se sigue tal como se visualiza en la **Figura 11**.

- **Tarjeta NFC. -** La tarjeta de identificación NFC, tendrá almacenada un valor único correspondiente a cada usuario.

- **Comunicación inalámbrica NFC. -** La comunicación inalámbrica NFC, es la tecnología que permite el envío de datos alojado en la tarjeta NFC hacía el *shield NFC*.
- **Shield NFC. -** Es el encargado de recibir los datos que se encuentran almacenados en la tarjeta y que posteriormente son enviados al microcontrolador para ser procesados.
- **Microcontrolador ESP32. –** El microcontrolador ESP32 deberá estar conectado a la alimentación mediante el puerto micro USB que posee, posteriormente recibe los datos del Shield y los procesa, de este modo envía una solicitud de búsqueda de datos a la base de datos alojada en la nube.
- **Comunicación inalámbrica Wi-Fi. –** Es la tecnología de comunicación inalámbrica que permite el acceso a internet para realizar las consultas y el intercambio de datos entre la base de datos en la nube y el microcontrolador.
- **Mongo DB Atlas. –** La base de datos recibe la petición realizada por el ESP32 y envía los datos que se haya solicitado el ESP32 para realizar la comparación.
- **Microcontrolador ESP32. –** Recibe los datos alojados en la nube y compara los datos de la base y los datos obtenidos mediante el NFC Shield, verifica si existen coincidencias y activa los leds y la cerradura electrónica, finalmente registra el usuario y la hora de ingreso en la base de datos.
- **Comunicación inalámbrica Wi-Fi. –** Permite el envío de datos del registro a la base de datos MongoDB Atlas.
- **Mongo DB Atlas. –** Almacena el registro del usuario que ha ingresado.
- **Led Indicador. –** El led, cumple la función de indicar si el acceso ha sido correcto o fallido.

- **Cerradura Eléctrica. –** La cerradura debe estar conectada a una fuente de 12v, posteriormente recibe una señal y se abre permitiendo el acceso.

Al culminar la etapa de funcionamiento del sistema, la siguiente sección detalla la forma en la que son interconectados los diferentes componentes de hardware.

3.6.2 Diagrama de conexión Eléctrica.

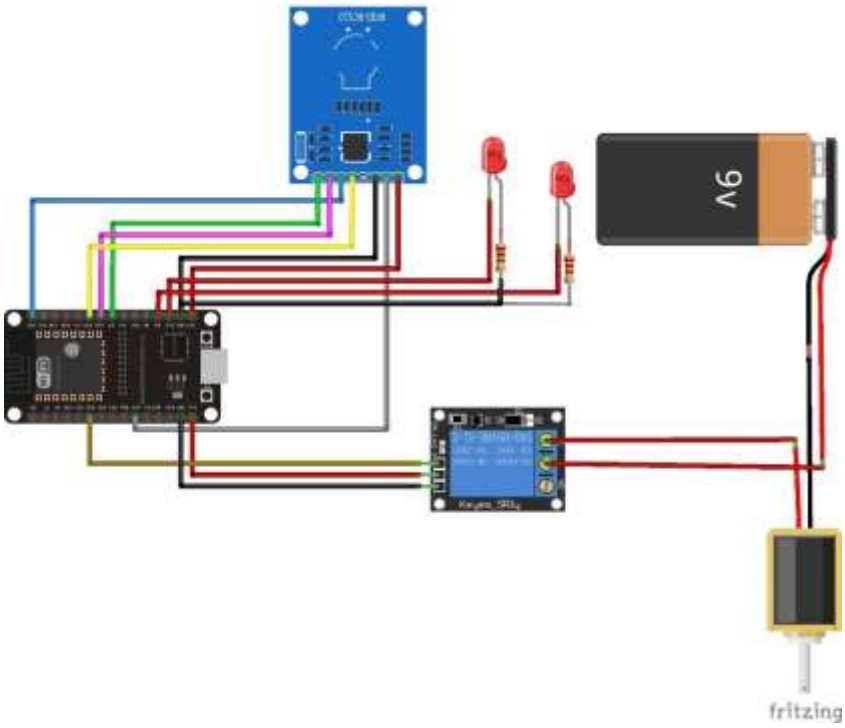
Para el sistema de control de acceso, se tienen los elementos que se enlistan a continuación: Microcontrolador ESP32, Shield NFC, Relé 5v, Leds, Resistencias, Alimentación y Solenoide que permitirá la activación de la cerradura eléctrica. Estos elementos funcionan en conjunto de acuerdo con la arquitectura planteada.

Posteriormente se realizan las conexiones tal como se visualiza en la **Figura 12** siguiendo los pasos detallados a continuación.

- Conectar el pin GND del NFC Shield a GND del ESP32
- Conectar el pin VCC del NFC Shield al pin VCC (3.3v) del ESP32
- Conectar el pin RST del NFC Shield al pin GIOP27 del ESP32
- Conectar el pin MISO del NFC Shield al pin GIOP19 del ESP32
- Conectar el pin MOSI del NFC Shield al pin GIOP23 del ESP32
- Conectar el pin SCK del NFC Shield al pin GIOP18 del ESP32
- Conectar el pin SS del NFC Shield al pin GIOP5 del ESP32
- Conectar el pin señal del relé al pin GIOP32 del ESP32
- Conectar el pin GND del relé al pin GND del ESP32
- Conectar el pin Vin del relé al pin Vin (5v) del ESP32
- Conectar las resistencias a los cátodos de los leds
- Conectar las resistencias al GND del ESP32
- Conectar el ánodo del led1 al pin 22

- Conectar el ánodo del led2 al pin 21
- Conectar el negativo de la alimentación al negativo del solenoide de la cerradura
- Conectar el positivo de la alimentación al común del relé
- Conectar el positivo del solenoide al comúnmente cerrado del relé

Figura *12*
Diagrama de conexión control de acceso



Posteriormente, se realiza el cálculo de las diferentes resistencias utilizadas y de la misma manera el cálculo total de consumo. Para realizar el cálculo de resistencias, se utiliza la ley de ohm en la cual se establece que voltaje es el producto de la intensidad y la resistencia.

$$V = I * R \quad (1)$$

Donde:

V = Voltaje

I = Intensidad

R = Resistencia

En este caso, para que la ley de Ohm de la ecuación (1) sea aplicable, se deberá despejar inicialmente la resistencia y posteriormente aplicar una diferencia entre el voltaje de la fuente y el voltaje con el que trabaja el led, resultando así la ecuación (2)

$$R = \frac{Vfuente - Vled}{Intensidad\ Led} \tag{2}$$

Una vez se tiene la ecuación correcta se reemplazan los diferentes valores, inicialmente se tiene que el microcontrolador es capaz de proporcionar 3.3v siendo este el valor de voltaje de la fuente, además, se tiene que, para un led rojo, el voltaje de led será de 1.9v y su intensidad de 15mA, por lo que utilizando la ecuación (2), se obtiene el siguiente valor de resistencia.

$$R = \frac{3.3v - 1.9v}{15mA}$$

$$R = 93.33\ ohms$$

De la misma manera, se calcula el valor de resistencia correspondiente para utilizar un led verde, en este caso, el voltaje que utiliza el led es de 2.2v y la intensidad de 10mA, por lo que al reemplazar los valores en la ecuación (2), se obtiene el valor de resistencia que se muestra a continuación.

$$R = \frac{3.3v - 2.2v}{10mA}$$

$$R = 110\ ohms$$

Con ello, se obtienen los valores de las respectivas resistencias, pudiendo utilizar en el circuito armado resistencias cercanas a dicho valor. Posteriormente se calcula el consumo total de todas las conexiones, para ello se considera la ecuación (3) la cuál menciona que la intensidad total es la suma de todas las intensidades del sistema.

$$I_t = I_1 + I_2 + I_3 + \cdots I_x \tag{3}$$

Donde:

I_t = Corriente total

I_x = Corriente de cada elemento

Mediante la ecuación (3) es posible calcular cuál es el consumo total del sistema, para ello, los valores correspondientes de cada componente electrónico se resumen en la **Tabla 16**.

Tabla 16
Consumo de dispositivos electrónicos

Dispositivo Electrónico	**Consumo**
Microcontrolador ESP32	180mA
Módulo MFRC522	40mA
Relé Arduino	70mA
Led Rojo	15mA
Led Verde	10mA

Una vez obtenido los valores de consumo de cada elemento, se aplica la ecuación (3), obteniendo el siguiente resultado.

$$I_t = 180mA + 40mA + 70mA + 15mA + 10mA$$

$$I_t = 315mA$$

De esta manera, es posible establecer el consumo total del sistema desarrollado por lo que se procede al siguiente apartado.

3.6.3 Programación Hardware.

En esta sección se da una descripción detallada del código desarrollado para el funcionamiento del sistema, para de este modo comprender de mejor manera el diagrama de flujo que se presenta posteriormente.

Para ello, inicialmente se presentan las librerías que son utilizadas en la programación, estás las podemos visualizar en la **Figura 13** y posteriormente se describe el funcionamiento y acciones que realiza cada una de ellas.

Figura *13*
Librerías utilizadas para la programación

```
#include <Arduino.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include <WiFiClientSecure.h>
#include <ArduinoJson.h>
#include <SPI.h>
#include <MFRC522.h>
#include <Separador.h>
#include <ESP32Time.h>
```

- **Librería Arduino.h:** Esta librería agrega los complementos necesarios que necesita el ESP32 para trabajar correctamente con el software de Arduino.
- **Librería WiFi.h:** Permite utilizar el Wi-Fi incluido del microcontrolador ESP32 para acceder a internet.
- **Librería HTTPClient.**h: Esta permite la interacción con servidores web, necesario para la conexión a la base de datos alojada en la nube.
- **Librería WiFiClientSecure.h:** Permite establecer conexiones seguras mediante HTTPS.

- **Librería ArduinoJson:** Establece el uso de documentos en formato JSON que son utilizados para realizar consultas en servidores de bases de datos no relacionales.
- **Librería SPI.h:** Permite la comunicación mediante de forma síncrona estableciendo un pin para comunicación serial.
- **Librería MFRC522.h:** Es la librería que permite el funcionamiento del módulo NFC, de esta manera es posible realizar lectura/escritura en tarjetas NFC compatibles con el módulo.
- **Librería Separador.h:** Permite separar texto obtenido, esto es especialmente útil al momento de realizar las diferentes consultas.
- **Librería ESP32Time.h:** Librería que controla el reloj interno del dispositivo ESP32.

Una vez establecidas las librerías a utilizar, se realiza la configuración del dispositivo para su funcionamiento tal como se muestra en la **Figura 14**.

Figura *14*
Código Arduino

```
WiFiClient client;

void setup() {
    pinMode(relay, OUTPUT);
    pinMode(pin_led, OUTPUT);
    pinMode(led2,OUTPUT);
    Serial.begin(9600);
    SPI.begin();
    mfrc522.PCD_Init();
    WiFi.mode(WIFI_STA);
     WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
    }
    Serial.print("Dirección IP");
    Serial.println(WiFi.localIP());
    configTime(gmtOffset_sec,daylightOffset_sec,ntpServer);
    struct tm timeinfo;
    if ( getLocalTime(&timeinfo)){
      rtc.setTimeStruct(timeinfo);
      Serial.println(rtc.getTime("%d/%m/%Y %H:%M"));
    } 0
}


void loop() {
  ///Variables para datos
  String id = "";
  String nombre = "";
  String apellido = "";
  String cedula = "";
  String celular = "";
  String fecha = "";
  String datos = "";
  byte buffer0[4];
  // prepara la clave que será utilizada para la encriptación de los datos
  MFRC522::MIFARE_Key key;
  for (byte i = 0; i < 6; i++) key.keyByte[i] = 0xFF;
  MFRC522::StatusCode status;       //Estado del lector
  if ( ! mfrc522.PICC_IsNewCardPresent()) {
    return;
  }
```

Una vez se visualiza el código que se está utilizando, se procede a describir detalladamente cada sección del mismo.

- Inicialmente se definen los pines correspondientes de acuerdo con el diagrama de conexión mostrado anteriormente para RST, SS, Señal relé, Led1 y Led2.
- Se inicializa el MFRC522 para la lectura de las tarjetas.
- Se crea una variable para el valor del separador, una variable para el reloj interno del ESP32 y una variable que permitirá el acceso.
- Se define una variable con la dirección del servidor que se va a conectar utilizando la API de MongoDB.
- Se define la clave de la API de MongoDB.
- Se define el ssid y clave del punto de acceso WiFi.
- Se establece el servidor para obtener la hora mediante NTP (*Network Time Protocol*).

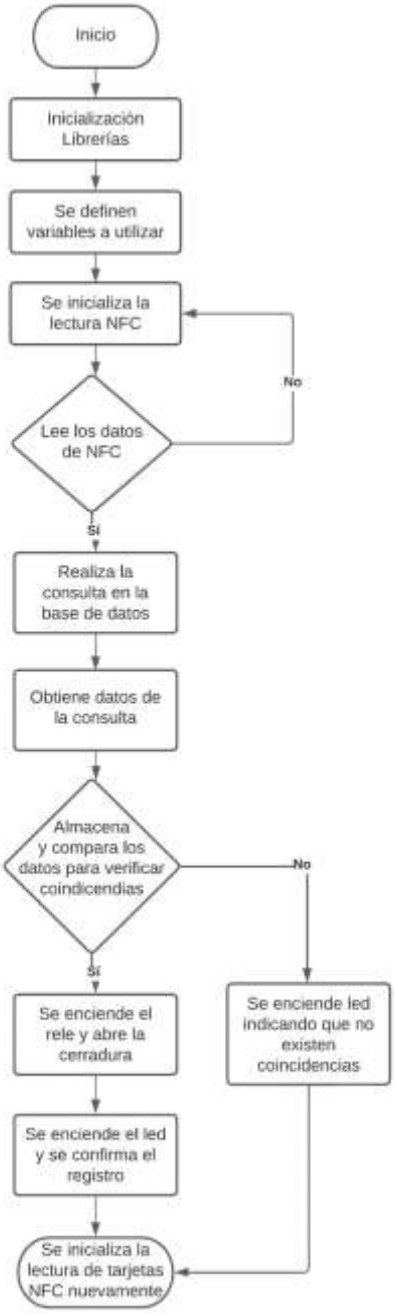
- Se establece la zona horaria en este caso Ecuador (GTM-5).
- Se configura el horario de verano, en caso de existir.
- Se define el certificado que permitirá realizar la conexión segura utilizando HTTPS.
- Una vez definidas las variables, se inicializa el servicio de Wi-Fi para conectarse al punto de acceso.
- En el *void setup*, se inicializa los pines correspondientes para enviar señales, la comunicación serial, la comunicación SPI, el módulo NFC, el Wi_Fi con los valores obtenidos SSID, su respectiva clave y finalmente se tiene la configuración del reloj interno mediante el servidor NTP.
- En el *void loop* se definen las variables que van a ser utilizadas para el registro de datos que serán enviados a la base de datos.
- Posteriormente se configura el método que permite inicializar la clave con la que se encuentra cifrada la información alojada en las tarjetas.
- Luego se realiza el método que permite la detección de tarjetas NFC y la lectura de las mismas.
- Una vez obtenido los datos de la tarjeta, se almacenan a una variable y se enciende un led que confirma la lectura.
- Posteriormente se genera la sesión HTTP con los parámetros ingresados en las variables, como son la dirección Web de la API, la clave de API y valores de *header.*
- Luego, se genera un documento de tipo JSON con parámetros de colección, base de datos y *cluster* que servirán para la búsqueda y comparación de los datos obtenidos de la tarjeta con los datos almacenados en la nube.

- Se realiza una solicitud de datos al servidor de mongoDB Atlas con el documento generado.
- Se recuperan los datos de la solicitud realizada y se almacenan en variables.
- Se compara si existe un valor de los datos obtenidos de la base de datos con los obtenidos de la tarjeta
- Si la comparación es verdadera, entonces se activa una señal que va al relé y que activa la cerradura.
- Si la cerradura ya ha sido activada, se enciende un led como indicador y se genera una nueva petición al servidor para registrar el ingreso con los datos que han sido almacenados previamente.
- En caso de que la comparación de datos sea falsa, se activa un led indicador y la cerradura no se abre.
- Finalmente se tiene que los datos de la lectura NFC se eliminan y se disponen a esperar la detección de una tarjeta NFC nueva para su lectura.

Una vez finalizada la descripción detallada del código, se procede a elaborar un diagrama de flujo que permita comprender de mejor manera el código planteado.

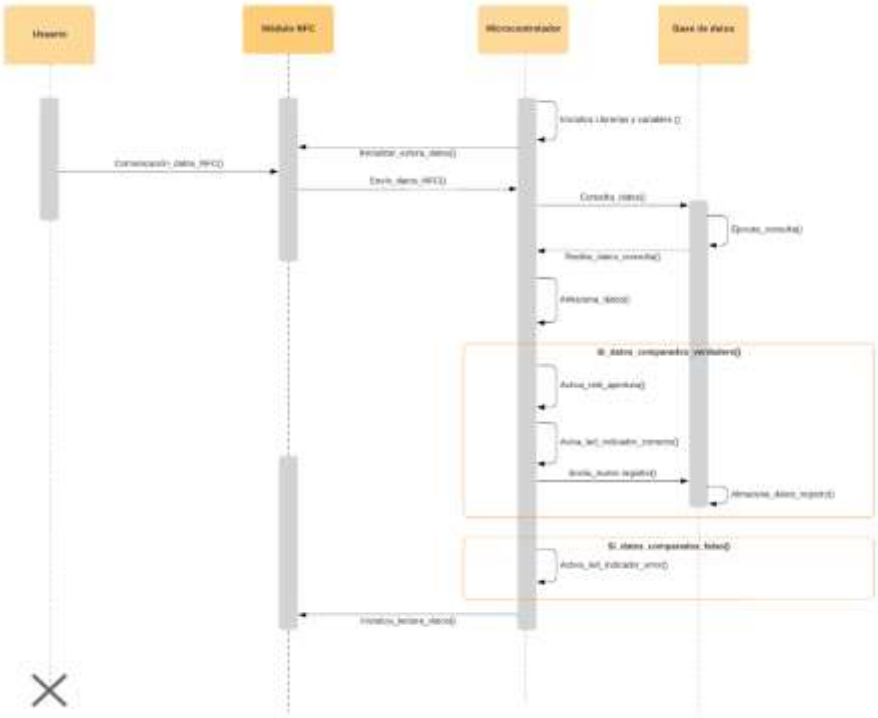
El diagrama de flujo muestra en un esquema los diferentes procesos mencionados anteriormente y los resume, de esta manera se puede visualizar el funcionamiento del sistema simplificado en la **Figura 15**.

Figura 15
Diagrama de flujo de configuración ESP32



Finalmente se presenta un diagrama de secuencia en el que es posible visualizar como se relacionan los diferentes elementos que conforman dicho sistema, la **Figura *16*** muestra el diagrama de secuencia.

Figura 16
Diagrama de Secuencia del módulo control de acceso



Nota: En el diagrama de secuencia, los recuadros de borde amarillo representan un condicional.

3.7 Desarrollo prototipo de lector NFC

Al finalizar el desarrollo del prototipo para control de acceso, lo siguiente será, desarrollar el prototipo que permita la lectura de datos mediante tecnología NFC y la transmita al computador para posteriormente ser utilizada mediante las aplicaciones de escritorio a desarrollar.

De este modo, el dispositivo a desarrollar cumple la principal función de leer aquellos datos cifrados que se encuentran dentro de la tarjeta NFC y posteriormente enviarlos al computador mediante el uso de comunicación serial establecido mediante un puerto USB.

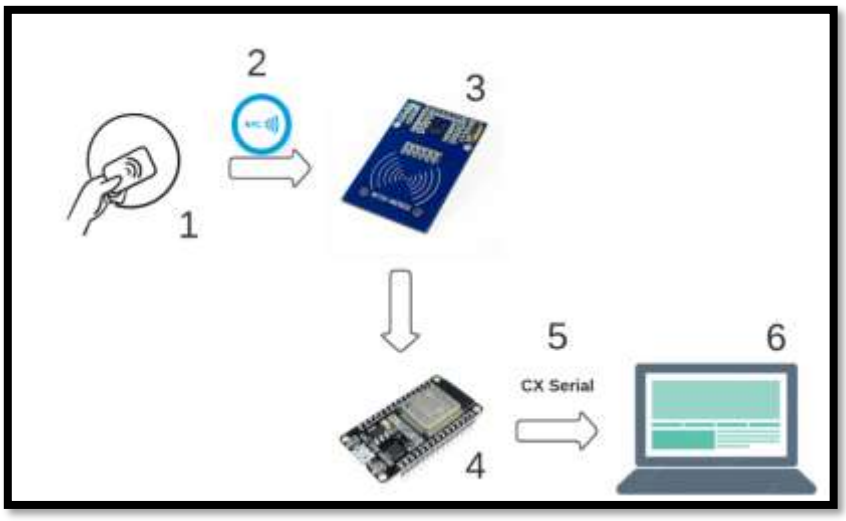
3.7.1 Arquitectura de prototipo de lector NFC.

Con el fin de desarrollar de mejor manera el sistema, se muestra la arquitectura del prototipo lector de NFC, dicha arquitectura cuenta con diferentes elementos que son visualizados en la **Figura *17***.

A continuación, se presentan los elementos considerados en la arquitectura del sistema:

1. Tarjeta NFC 13,56Mhz
2. Comunicación Inalámbrica NFC
3. NFC Shield
4. Microcontrolador ESP32
5. Comunicación Serial
6. Computador que recibe los datos del dispositivo NFC.

Figura 17
Arquitectura aplicación Windows control acceso



Una vez visualizado la arquitectura, se describe cada uno de los elementos a continuación:

- **Tarjeta NFC:** La tarjeta de identificación NFC, tendrá almacenada un valor único correspondiente a cada usuario.

- **Comunicación inalámbrica NFC:** La comunicación inalámbrica NFC, es la tecnología que permite el envío de datos alojado en la tarjeta NFC hacía el *shield NFC*.
- **Shield NFC:** Es el encargado de recibir los datos que se encuentran almacenados en la tarjeta y que posteriormente son enviados al microcontrolador para ser procesados.
- **Microcontrolador ESP32:** El Microcontrolador ESP32 debe estar conectado mediante el puerto Micro USB al computador que recibirá los datos.
- **Comunicación Serial:** La comunicación serial permite la transmisión de los datos desde el ESP32 hacia el computador.
- **Computador:** El computador será el encargado de recibir los datos procesados y posteriormente serán utilizados de acuerdo con la aplicación desarrollada.

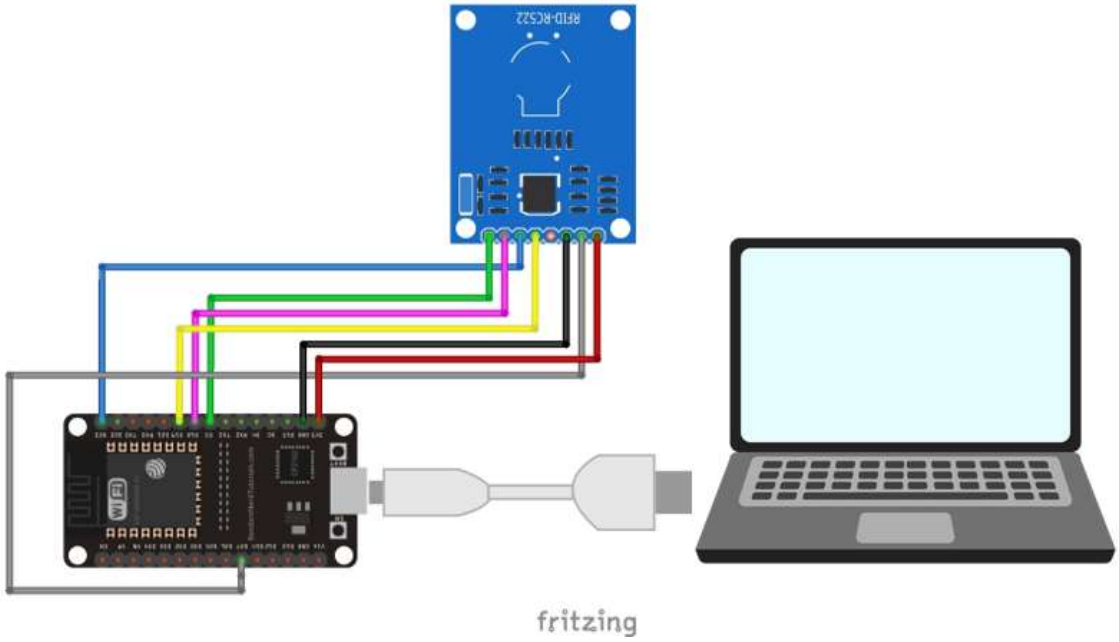
3.7.2 Diagrama de conexión, del prototipo lector NFC.

En la siguiente sección se muestra cómo se interconectan todos los elementos de la arquitectura mencionados anteriormente. A continuación, se describen los pasos a seguir para realizar la interconexión correctamente.

- Conectar el pin GND del NFC Shield a GND del ESP32
- Conectar el pin VCC del NFC Shield al pin VCC (3.3v) del ESP32
- Conectar el pin RST del NFC Shield al pin GIOP27 del ESP32
- Conectar el pin MISO del NFC Shield al pin GIOP19 del ESP32
- Conectar el pin MOSI del NFC Shield al pin GIOP23 del ESP32
- Conectar el pin SCK del NFC Shield al pin GIOP18 del ESP32
- Conectar el pin SS del NFC Shield al pin GIOP5 del ESP32
- Conectar la interfaz Micro USB del ESP32 a un USB del computador que tenga alojado la aplicación.

Una vez finalizado los pasos, se obtendrá el esquema mostrado en la **Figura *18***.

Figura 18
Diagrama de conexión aplicación Windows control acceso.



Con las conexiones establecidas correctamente, se realiza la programación correspondiente a utilizarse en el ESP32 para la transmisión de datos.

3.7.3 Programación del Hardware del lector NFC.

En esta sección se presenta el código realizado que permite la obtención de los datos de tarjetas NFC compatibles y posteriormente la transmisión de los mismos utilizando la comunicación serial mediante el puerto micro USB del ESP32 al puerto USB del computador a utilizar.

Para ello inicialmente se muestra el código desarrollado en la **Figura *19***

Figura 19

Configuración ESP32 comunicación serial

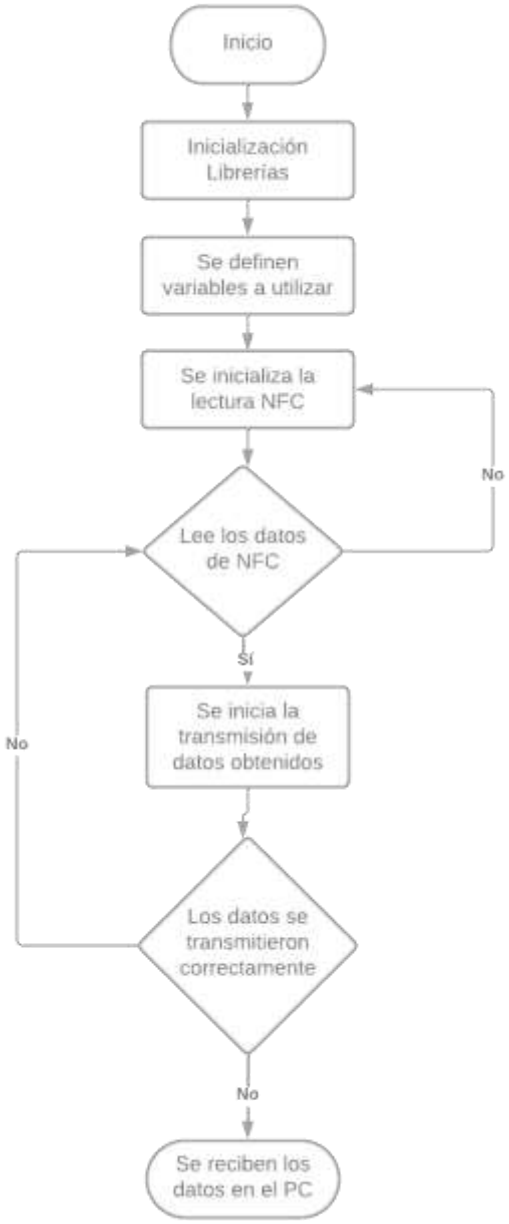
```
#include <SPI.h>
#include <MFRC522.h>
#define RST_PIN         27
#define SS_PIN          5
MFRC522 mfrc522(SS_PIN, RST_PIN);
void setup() {
  Serial.begin(9600);
  SPI.begin();
  mfrc522.PCD_Init();
}

void loop() {
  String str;
  byte buffer0[4];
  // prepara la clave que será utilizada para la encriptación de los datos
  MFRC522::MIFARE_Key key;
  for (byte i = 0; i < 6; i++) key.keyByte[i] = 0xFF;
  //some variables we need
  MFRC522::StatusCode status;
  if ( ! mfrc522.PICC_IsNewCardPresent()) {
    return;
  }
  if ( ! mfrc522.PICC_ReadCardSerial()) {
    return;
  }
  for (byte i = 0; i < 4; i++) {
    buffer0[i] = mfrc522.uid.uidByte[i];
  }
  for (byte i = 0; i < mfrc522.uid.size; i++) {
    Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? "0":"");
    Serial.print(mfrc522.uid.uidByte[i],HEX);
  }
  Serial.println("");
  delay(1000);
  mfrc522.PICC_HaltA();
  mfrc522.PCD_StopCrypto1();
}
```

- Inicialmente se obtienen las librerías SPI y MFRC522 que permiten el funcionamiento del NFC Shield utilizado.
- Posteriormente se definen los pines que serán utilizados para la transmisión de datos entre el NFC Shiel y el ESP32.
- Una vez definidos los pines, se inicializa la librería correspondiente del módulo NFC.
- Dentro del *void setup*, se establece la comunicación serial que tendrá lugar entre el ESP32 y la computadora, esta será de 9600 baudios.

- Se inicializa el SPI para la transmisión de datos entre el NFC Shield y el ESP32.
- Se inicializa el NFC Shield
- En el *void loop*, inicialmente se prepara la clave que será utilizada para desencriptar los datos almacenados en la tarjeta NFC.
- Posteriormente se genera el estado para detectar tarjetas NFC
- Luego se inicializa la condición que permite leer los datos de la tarjeta NFC.
- El ciclo *for* permite leer cada uno de los bytes almacenados dentro de la tarjeta.
- Posteriormente se envían los datos mediante la comunicación serial mediante la orden de *Serial.print*.
- Luego, se agrega un salto de línea para identificar que todos los datos han sido enviados.
- Se detiene la lectura de datos y se reinicia el sistema para la detección de una nueva tarjeta.

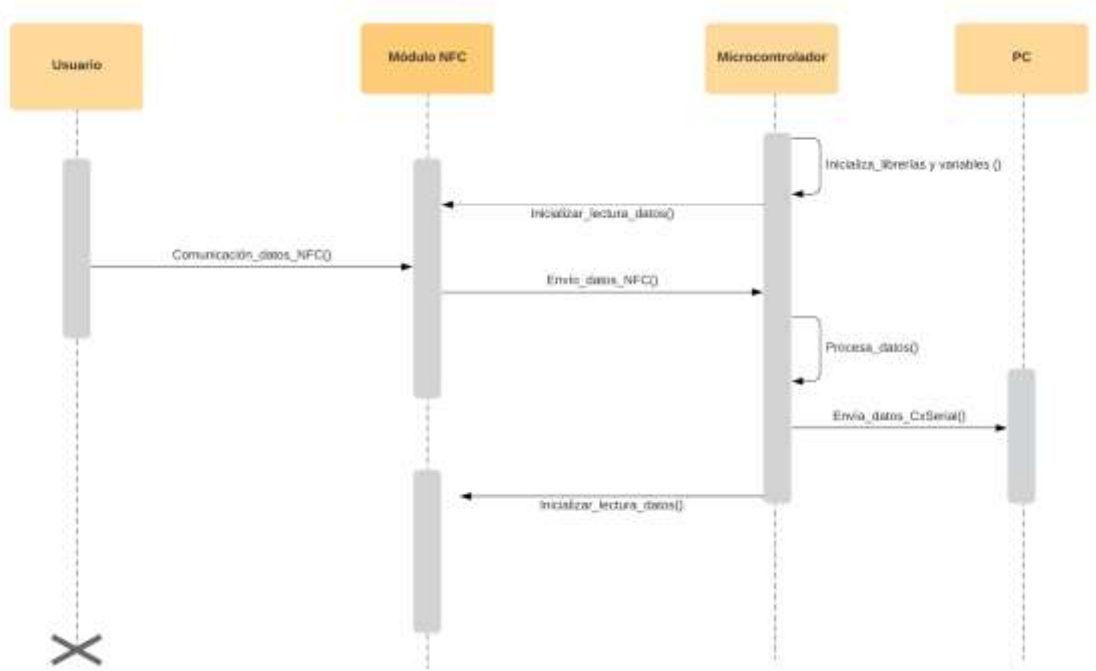
Al concluir la descripción del código desarrollado, se propone un diagrama de flujo muestra de manera simplificada el código de programación, esto da una mejor visión del funcionamiento del programa realizado, de esta manera, se tiene el diagrama en la **Figura *20***.

Figura 20
Diagrama de flujo programación ESP32



De la misma manera, se desarrolla un diagrama de secuencia que se visualiza en la **Figura 21**, el cual permite visualizar como interactúan los diferentes elementos del sistema desarrollado.

Figura 21
Diagrama Secuencia Modulo Lector NFC



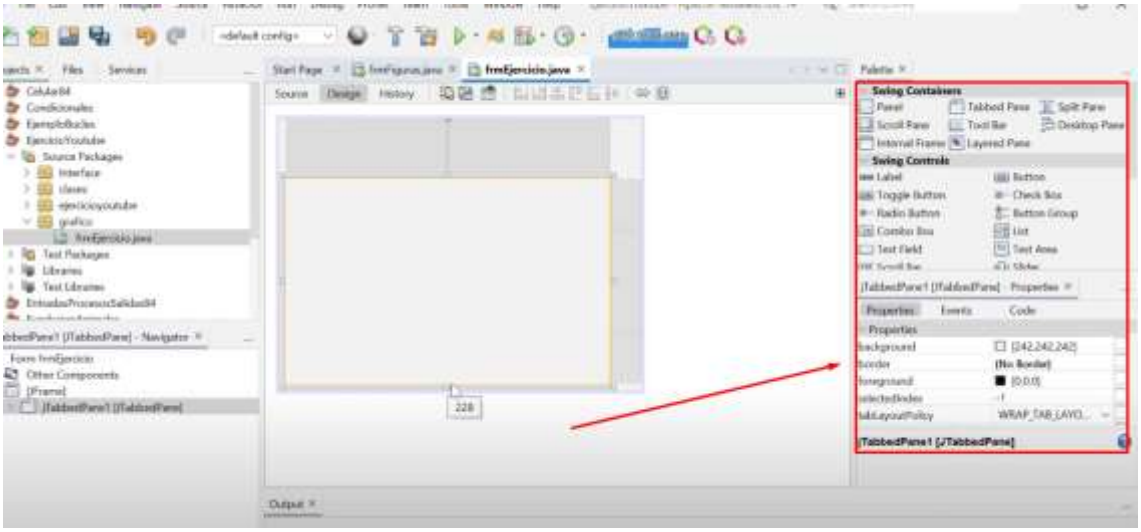
Nota: El diagrama de secuencia permite identificar como se relacionan los elementos que interactúan en el sistema en función del tiempo.

3.8 Desarrollo Aplicaciones Windows

En la siguiente sección, se muestra el proceso de desarrollo de las diferentes aplicaciones para Windows que debe considerar el sistema. Para ello se consideran el uso del del lenguaje de programación JAVA y el entorno de desarrollo NetBeans.

Dado que NetBeans permite el uso de interfaz gráfica de elementos, cada uno de ellos deberá ser agregado previamente de acuerdo con el panel para su posterior programación, la **Figura *22*** muestra los diferentes elementos que se utilizan para la creación de interfaces.

Figura 22
Interfaz de Aplicación NetBeans



Nota: Para crear diferentes interfaces se utilizan todos los elementos del *Palette*, en el, es posible encontrar elementos como tablas, botones, cuadros de texto, etc.

Una vez visualizado cuál es el modelo de desarrollo de aplicaciones, se procede a definir el desarrollo de cada aplicación de manera individual y posteriormente se unifica cada aplicación desarrollada logrando un sistema completo, finalmente es posible obtener el código de programación y el desarrollo de cada aplicación en el Anexo B. Con esto mencionado, se inicia con el desarrollo de la aplicación de control de acceso.

3.8.1 Desarrollo Aplicación Usuarios

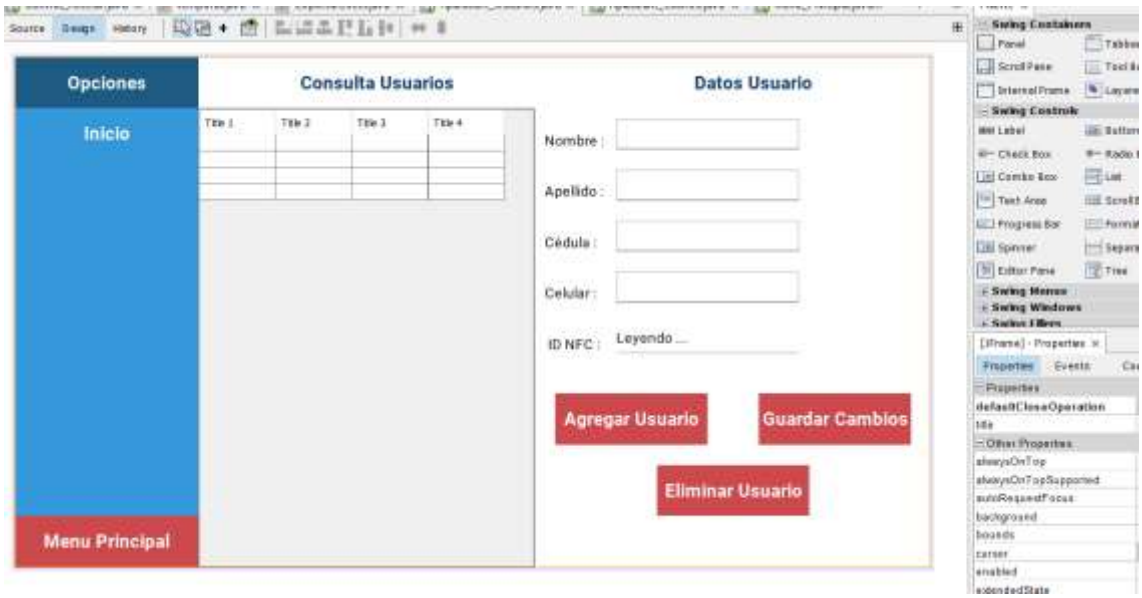
En este apartado se muestra el desarrollo de aplicación orientada al registro de usuarios. De esta manera, la aplicación será capaz de conectarse a la base de datos alojada en la nube mediante internet y realizar diversas acciones como consultas, agregar datos o visualizarlos en dicha base.

Como requisito principal para el desarrollo de dicha aplicación se tiene que la aplicación será capaz de registrar a todos los nuevos usuarios, editar cualquier cambio relacionado con el mismo y visualizarlo de manera eficiente mediante una tabla, esta

aplicación será fundamental, ya que mediante esta se registran todos los usuarios que sean participes de las demás aplicaciones.

Para ello, lo primero que se realiza es crear un nuevo proyecto en el entorno de desarrollo y agregar los diferentes campos que tendrá la interfaz. Esto es posible visualizarlo en la **Figura *23***.

Figura 23
Interfaz gráfica de aplicación usuarios



Nota: La interfaz de programación que se visualiza, se ha desarrollado utilizando el *palette* tal como se muestra en la **Figura *22***

De esta manera, se procede a describir el funcionamiento de la aplicación desarrollada. Al igual que el desarrollo de la aplicación anterior, es necesario tener conectado el dispositivo lector NFC que se ha desarrollado previamente, una vez realizado dicha conexión la aplicación será capaz de recibir el ID NFC mediante comunicación serial.

La aplicación se usuarios está orientada al registro y visualización de los usuarios que harán uso de dicho sistema, de este modo existen diferentes campos necesarios que

deben ser llenados para poder agregar un nuevo usuario, caso contrario, el usuario no se puede agregar.

Inicialmente se debe agregar los nombres y apellidos del usuario en los campos correspondientes, dichos campos además solo permiten el ingreso de letras, dado que todos los caracteres especiales, se encuentran bloqueados, esto con el fin de evitar problemas al momento del registro.

Posterior, se debe ingresar los campos de cédula y celular que son necesarios de la misma manera para completar el registro, en estos apartados, solo se permite el ingreso de números para completar un registro correcto, además el número de dígitos posibles a ingresar se encuentra limitado a 10 dado que es el número correspondiente a los dígitos de celular y de numero de cedula.

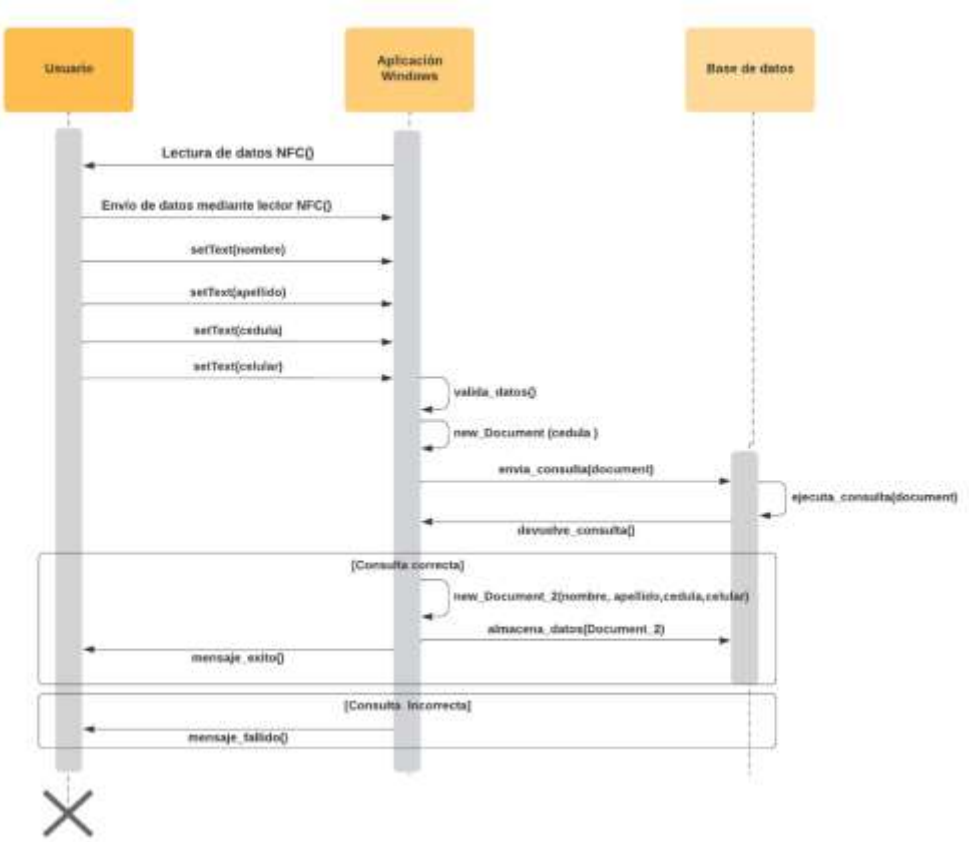
Posterior, el campo de NFC ID, será asignado de manera automática al momento de pasar una nueva tarjeta por el lector, dicha tarjeta será entrega al usuario que haga uso del sistema.

Finalmente, el sistema permite agregar el usuario con los campos mencionados antes, realizar cambios en el mismo o eliminarlo de la base de datos de manera rápida realizando clic en los botones correspondientes que se observan en la **Figura *23***.

Con el fin de comprender de mejor manera el funcionamiento de la aplicación, se desarrolla un diagrama de secuencia para un caso de uso específico, en la **Figura *24*** se tiene el diagrama de secuencia que representa la manera de registrar un nuevo usuario al sistema.

Figura 24

Diagrama de secuencia para caso de uso de aplicación usuarios



El Pseudocódigo 1 muestra las diferentes variables que se utilizan en la aplicación "Usuarios", así como también la lógica de programación que se aplica en la misma lo que brinda una mejor visión del desarrollo de código.

Pseudocódigo 1: Caso de Uso Aplicación Usuarios Registro de Nuevo Usuario

1. **Input:** ID NFC = Valor identificador obtenido mediante lector NFC
2. **Input:** Nombre = nombre del usuario a registrar
3. **Input:** Apellido = apellido del usuario a registrar
4. **Input:** Cedula = cedula del usuario a registrar
5. **Input:** Celular = celular del usuario a registrar
6. Validar los datos ingresados
7. Crear filtro de verificación de usuario existente
8. Realiza consulta de verificacion a la base de datos utilizando el filtro
9. **if** filtro = true
10. Crear documento con variables input
11. Insertar documento en la base de datos
12. Mostrar mensaje de confirmación
13. Inicializar valores input en cero

14. Mostrar la tabla actualizada de usuarios
15. **else**
16. Enviar mensaje de alerta de usuario existente
17. **Fin del proceso**

3.8.2 Desarrollo Aplicación Control Acceso

En esta sección se muestra el desarrollo de una aplicación que permite la gestión de información almacenada en la base de datos referente al control de acceso y los resultados obtenidos al culminar la misma.

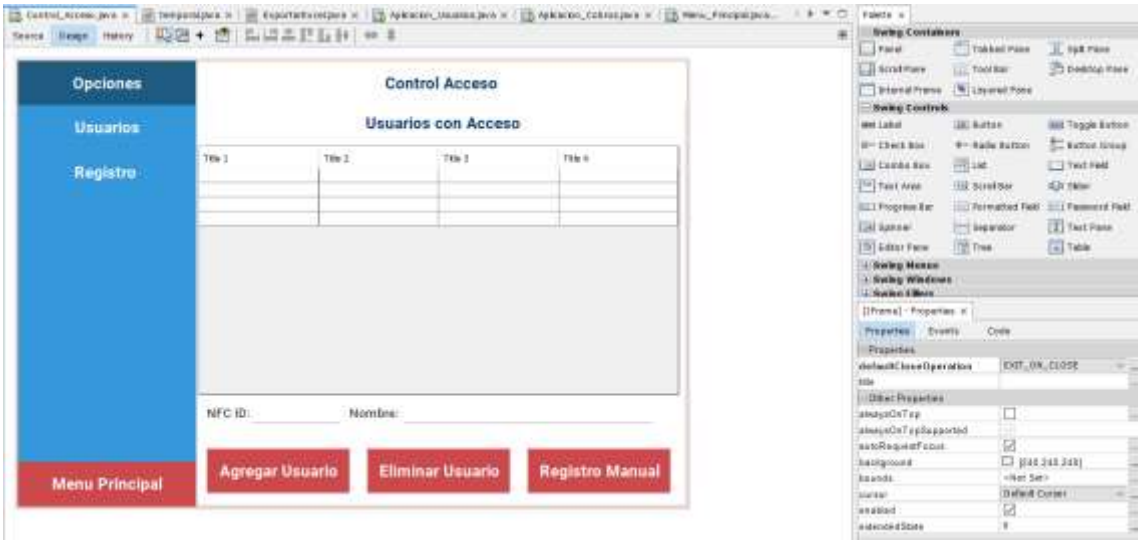
De esta manera se definen algunos de los parámetros que se consideran al desarrollar dicha aplicación:

- La aplicación permite llevar un registro de todos los usuarios que tienen acceso.
- Se permite agregar y eliminar usuarios de manera rápida y eficiente.
- En caso de fallas en el control de acceso, el sistema permite el registro de forma manual.
- Se tiene un resumen de todos aquellos que hayan ingresado.

De este modo, para cumplir con los requisitos antes mencionados, se hace uso de diferentes elementos que nos proporciona NetBeans, inicialmente se tiene el desarrollo de la interfaz gráfica que se visualiza en la **Figura *25***. Dicha interfaz posee elementos como tablas, botones y también etiquetas que permiten realizar las 3 primeras funciones que se han propuesto anteriormente.

Figura 25

Interfaz gráfica de aplicación control acceso 1/2



Al terminar el desarrollo de la primera interfaz, se describe de manera rápida el funcionamiento que la misma.

Inicialmente se debe tener conectado el lector NFC desarrollado al computador en el cual se encuentra instalada la aplicación, dicho lector utiliza el puerto serial para la comunicación de datos.

Al momento de inicializar la aplicación, la misma se conecta mediante internet a la base de datos alojada en la nube y obtiene los elementos correspondientes al registro de usuarios que tienen acceso y los muestra en la tabla visualizada.

Para agregar o eliminar un usuario al sistema de control de acceso, dicho usuario deberá utilizar el lector para que los datos sean transmitidos a la aplicación, al momento de recibir los datos, la aplicación genera una consulta a la base de datos en la que se muestra el nombre del usuario que se pretende agregar, posteriormente se deberá presionar el botón de agregar o eliminar usuario, cabe recalcar, que si el usuario ya ha sido agregado anteriormente, no se genera una nueva entrada y por el contrario, si se desea eliminar un usuario que no se encuentra agregado, el sistema no realiza ninguna acción.

De la misma manera, si se requiere agregar un usuario de forma manual, este inicialmente debe obtener los datos mediante el lector y adicionalmente, se registra la fecha y hora en la que el usuario accedió.

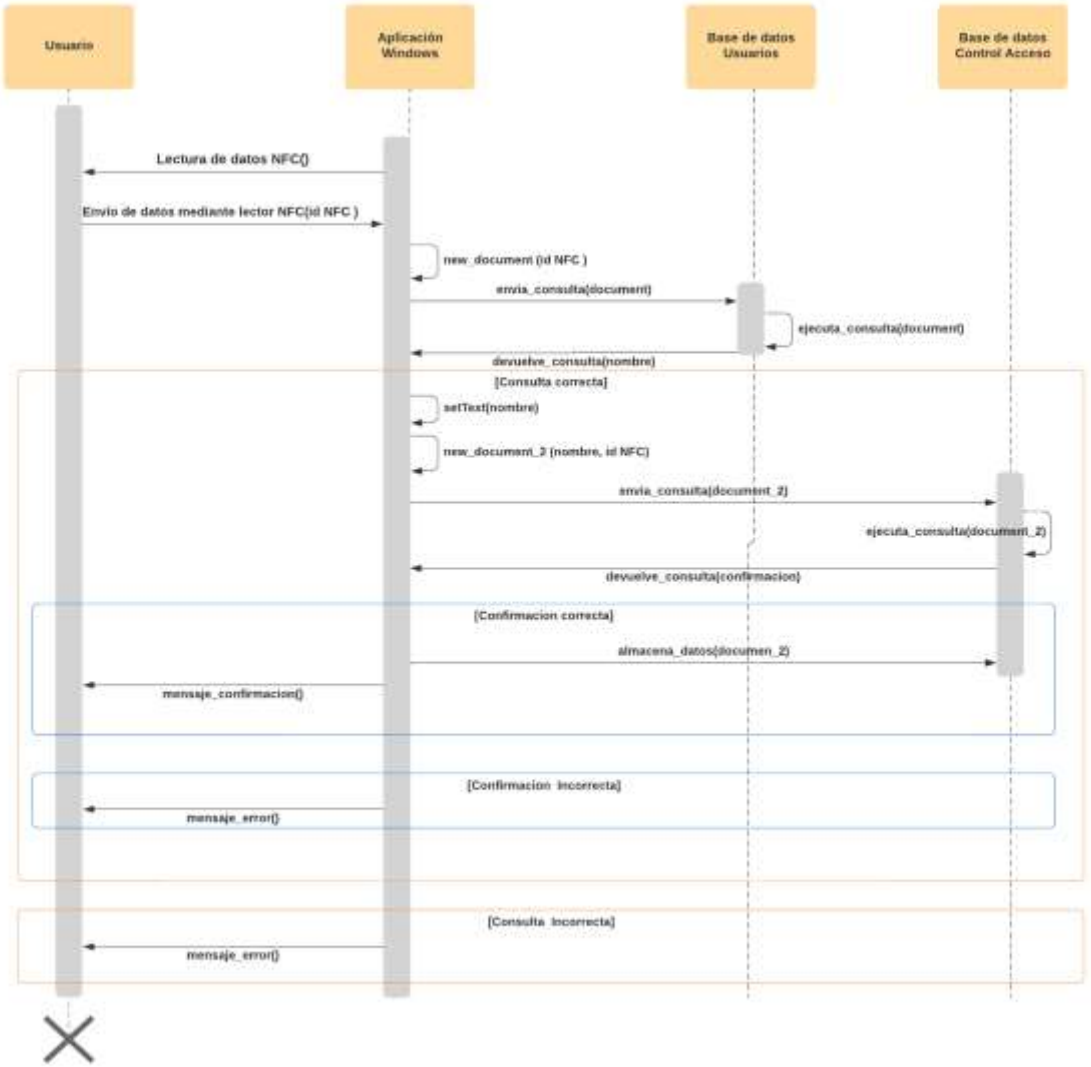
Para cumplir con la 4ta función mencionada anteriormente que el sistema debe realizar, se plantea una nueva interfaz en la cual se tiene una tabla que recupera todos los datos almacenados de la base de datos alojada en la nube y los muestra en la tabla creando un registro de todo aquel que haya ingresado. Dicha interfaz se puede visualizar en la **Figura *26*** mostrada a continuación.

Figura 26
Interfaz gráfica de aplicación control acceso 2/2

Posteriormente se muestra un diagrama de secuencia para un determinado caso de uso , este contempla la manera de proporcionar acceso a un usuario registrado, de este modo, la funcionalidad de las clases y como se relacionan entre ellas es fácilmente comprendida tal como se muestra en la **Figura *27***.

Figura 27

Diagrama de secuencia para caso de uso de aplicación control de acceso



La lógica de programación utilizada para desarrollar la aplicación correspondiente al control de acceso se detalla en el Pseudocódigo 2, en él, se visualizan las diferentes variables utilizadas así como también los condicionales que hacen referencia a las validaciones necesarias para el correcto funcionamiento de la aplicación.

Pseudocódigo 2: Caso de Uso Aplicación Control Acceso-Acceso a Usuario Registrado

1. **Input**: ID NFC = Valor identificador obtenido mediante lector NFC
2. Validar los datos input
3. Crear filtro utilizando ID NFC
4. Realizar consulta de verificacion a la base de datos usuarios
5. **if** filtro = true
6. filtro **return** nombre
7. Mostrar nombre del usuario
8. Mostrar ID NFC usuario
9. Generar documento2 utilizando nombre y usuario
10. Realizar consulta con el documento2 a la base de datos de acceso
11. **if** documento2 = false
12. Almacenar documento en base de datos acceso
13. **else**
14. Mostrar mensaje de alerta de usuario existe en base de datos acceso
15. **else**
16. Mostrar mensaje usuario no existe en base de datos usuario
17. **Fin del proceso**

3.8.3 Desarrollo Aplicación Micro Transacciones

A continuación, se propone la aplicación de micro transacciones, así como también el desarrollo de esta.

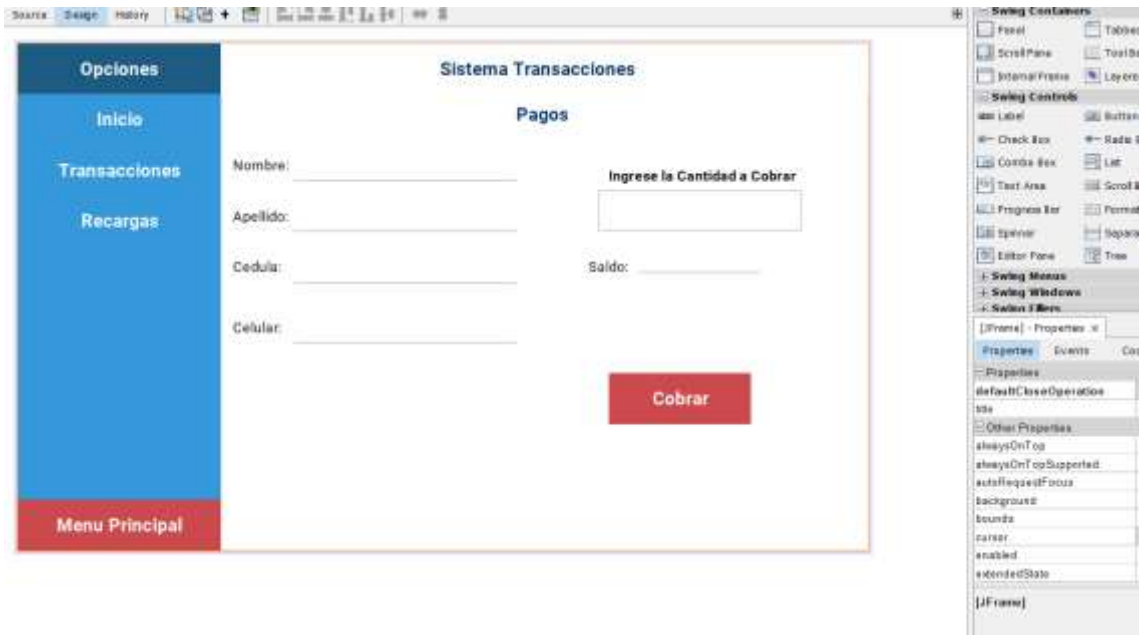
La Aplicación “micro transacciones” tiene como finalidad mejorar el acuerdo existente entre la asociación y la copiadora de la FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS, que dicta que cada estudiante de la carrera de CITEL-CIERCOM, tiene acceso a número determinado de impresiones que corre por parte de la asociación. Con dicha aplicación se plantea mejorar el sistema, para que exista un saldo disponible por cada estudiante y este sea el que decida como utilizarlo sin la limitación de estar orientado siempre al uso de impresiones o copias. Para ello inicialmente se definen algunos requerimientos que debe cumplir dicha aplicación.

- El sistema permite realizar un pago directamente, siempre y cuando se tenga el saldo disponible en la cuenta del usuario registrado
- El sistema permite agregar saldo a las cuentas de usuario registradas

- El sistema permite llevar un registro de todas aquellas micro transacciones que se realicen.

Para cumplir con los requerimientos mencionados, inicialmente se tiene el desarrollo de la interfaz sobre la cual se trabaja. En la **Figura *28***, es posible visualizar la interfaz sobre la que se trabaja para cumplir el primer requisito.

Figura 28
Interfaz gráfica aplicación micro transacciones 1/3

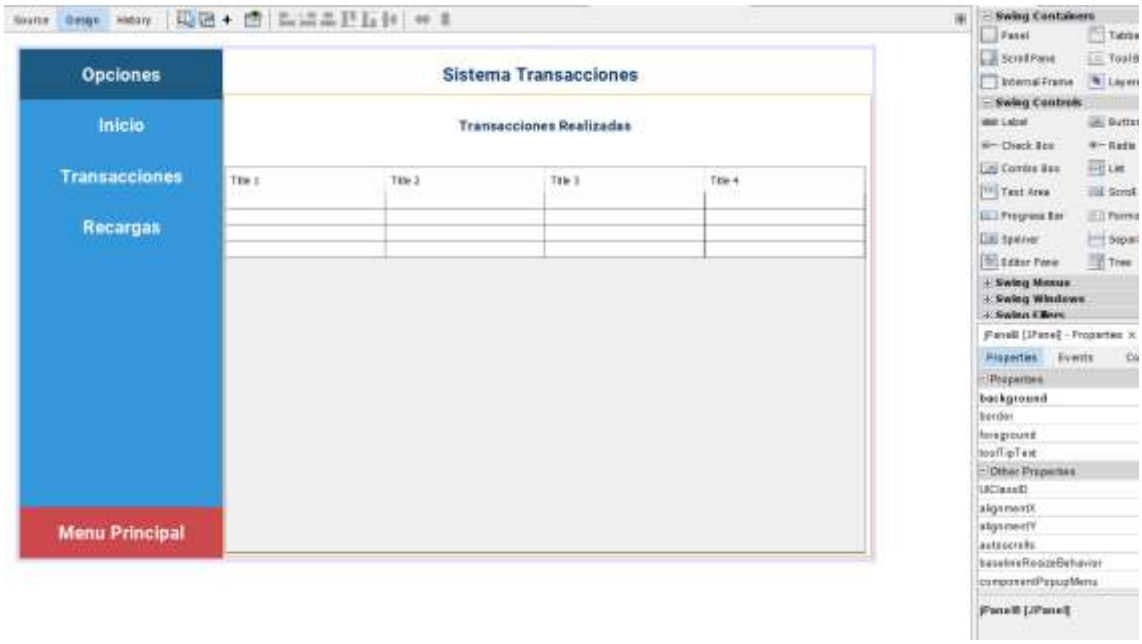


Para el funcionamiento correcto de la aplicación, es importante mencionar que el dispositivo lector NFC desarrollado previamente, debe estar conectado y enviando datos hacia el PC.

En la primera interfaz, la aplicación será capaz de recibir los datos enviados mediante el lector y realizar una consulta inmediata sobre los diferentes parámetros visualizados, como nombre, apellido, cedula y celular. De la misma manera se realiza una consulta del saldo disponible para poder realizar dicha transacción. En esta interfaz, el usuario seleccionará el valor a cancelar y al momento de realizar el cobro, se descontará el mismo y se registrará dicha transacción.

La segunda interfaz desarrollada, es posible visualizarla en la **Figura *29***, dicha interfaz consta de una tabla en la cual la aplicación recupera todos los datos almacenados en la base de datos en la nube y muestra un registro de todas las micro transacciones que han sido realizadas.

Figura 29
Interfaz gráfica aplicación micro transacciones 2/3



Finalmente, para completar los requerimientos establecidos inicialmente, se desarrolla la interfaz 3, dicha interfaz es similar a la primera establecida en esta sección y permite el aumento de saldo disponible por parte de los estudiantes usuarios del sistema.

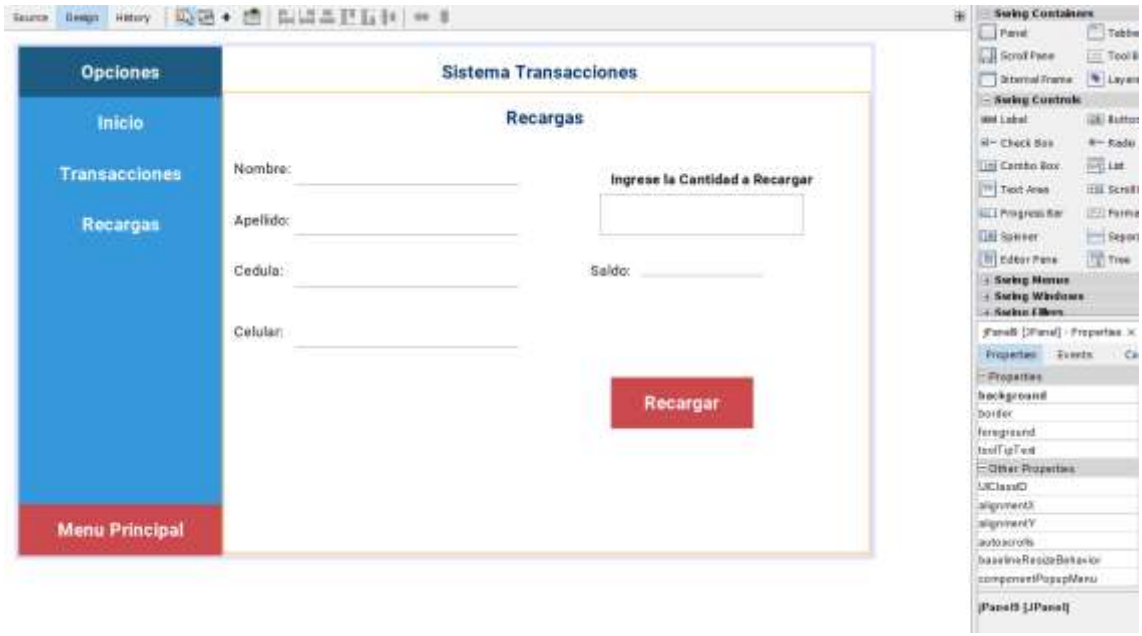
Dicha aplicación se plantea para realizar un aumento en el saldo disponible de cada usuario, por lo que, de ser el caso el estudiante podrá solicitar el aumento de saldo al sistema para ser utilizado.

El funcionamiento de dicha aplicación es similar a de la interfaz 1, inicialmente se requiere que el lector NFC desarrollado, sea conectado a la PC y desde ahí se realice la lectura de los datos almacenados.

Una vez se tengan los datos, se realiza una consulta y se completan todos los campos que se visualizan en la **Figura *30***. Posteriormente se debe establecer el valor de dinero que se requiere agregar a la cuenta y se realiza la acción correspondiente actualizando los valores que se encuentren establecidos en la base de datos, esto a la vez genera un mensaje de confirmación.

Figura 30

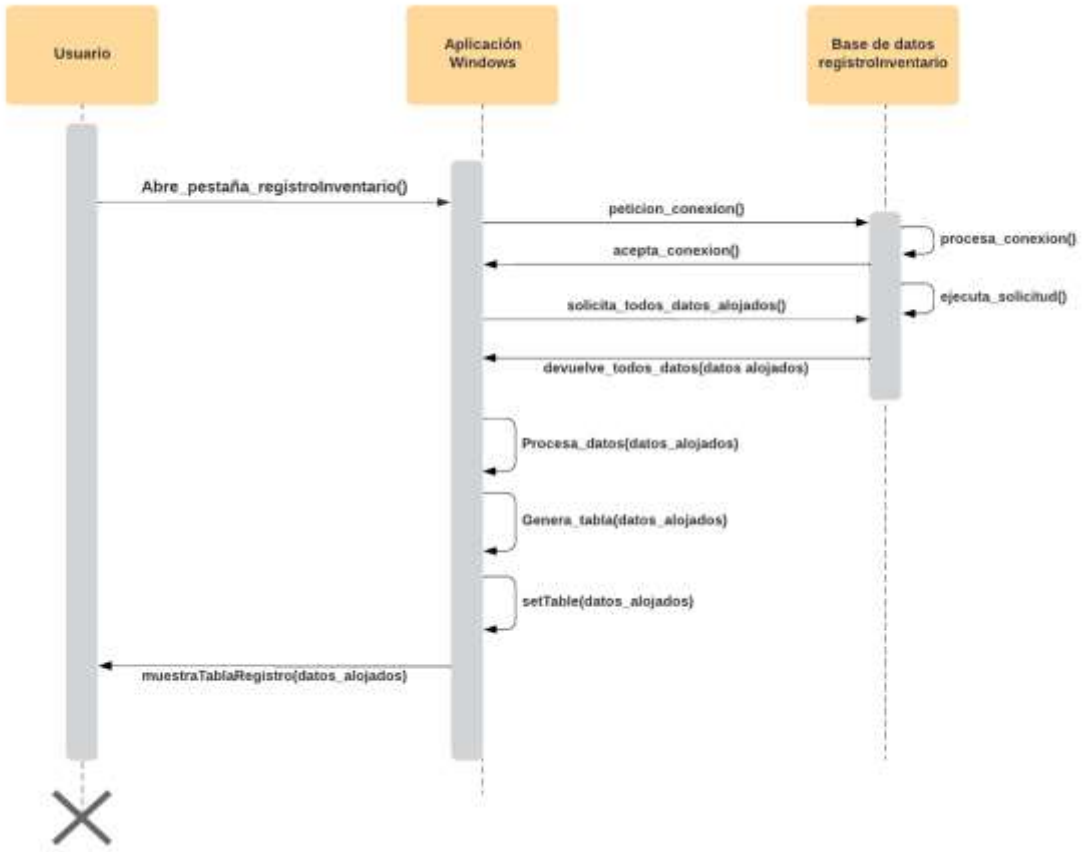
Interfaz gráfica aplicación micro transacciones 3/3



Con el fin de comprender de mejor manera el funcionamiento de la aplicación, se desarrolla un diagrama de secuencia simplificado en el que se observa un caso de uso

para la aplicación presentada, este diagrama contempla el caso de visualizar el registro de micro transacciones realizadas y se logra observar en la **Figura *31***.

Figura 31
Diagrama de secuencia caso de uso aplicación micro transacciones



Al finalizar el desarrollo de la aplicación “Micro Transacciones” se muestra el pseudocódigo 3 con la lógica de programación utilizada, así como también las diferentes variables y condicionales permiten la correcta funcionalidad de esta.

Pseudocódigo 3: Caso de Uso Aplicación Micro Transacciones Registro de micro transacciones

1. **Input:** Presionar pestaña registro
2. Realizar petición_conexión
3. **if** (petición_conexión = true
4. Solicitar datos_alojados en base de datos
5. **if** datos_alojados =! vacio
6. Procesar los datos

7. Generar tabla acorde al número de datos procesados
8. Mostrar los datos obtenidos de la base en cada columna y fila de la tabla
9. **else**
10. Mostrar tabla sin datos
11. **else**
12. Mostrar mensaje error conexión
13. **Fin del proceso**

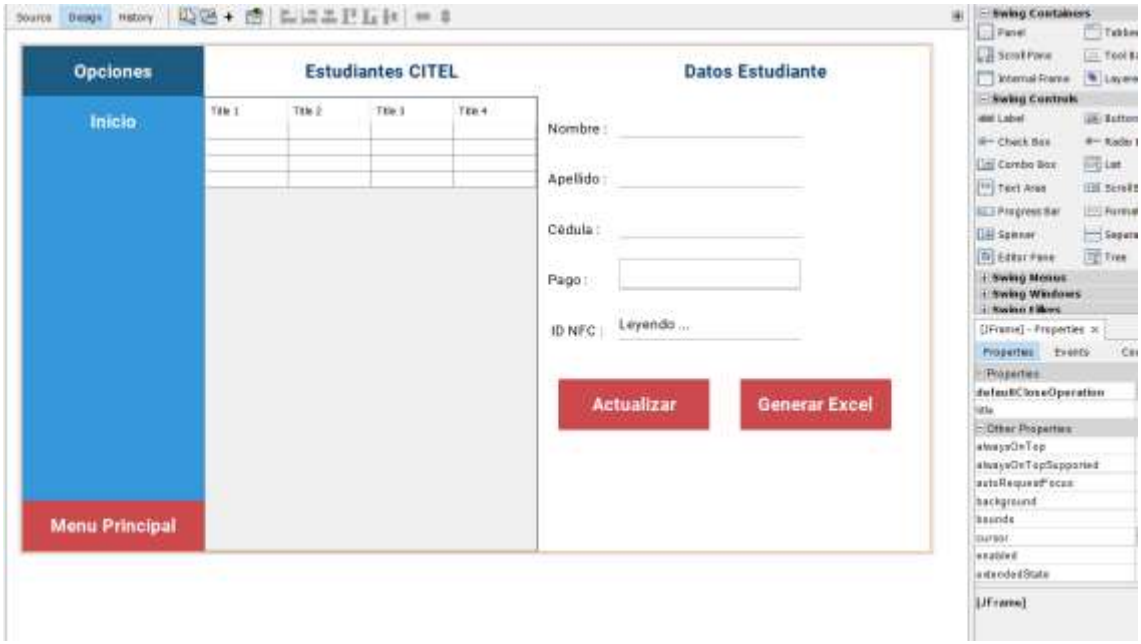
3.8.4 Desarrollo Aplicación Aportes Voluntarios

En esta sección se desarrolla un sistema orientado al registro y gestión de un aporte que se realiza de manera semestral, este con la finalidad de registrar una mejor organización de todos aquellos estudiantes que lo hacen.

De este modo, la aplicación debe ser capaz de llevar un registro de todos los estudiantes que hayan realizado la aportación, además el sistema será capaz de emitir un comprobante si así lo requiere el estudiante y también se debe imprimir el registro y almacenarlo en Excel en caso de ser necesario para desarrollar otras actividades.

En la **Figura *32***, se tiene la interfaz sobre la cual se trabaja para el desarrollo de la aplicación. En ella es posible visualizar la existencia de diferentes botones, una tabla y algunos campos que serán llenados para realizar el pago de aporte voluntario.

Figura 32
Interfaz gráfica aplicación pago aportes



Al visualizar la interfaz gráfica que se está utilizando, se continúa con la descripción del funcionamiento de dicha aplicación.

Inicialmente, se debe tener conectado el dispositivo Lector NFC desarrollado anteriormente, una vez realizado, el sistema agrega todos los campos necesarios referentes a dicho usuario y se completará la asignación de pago.

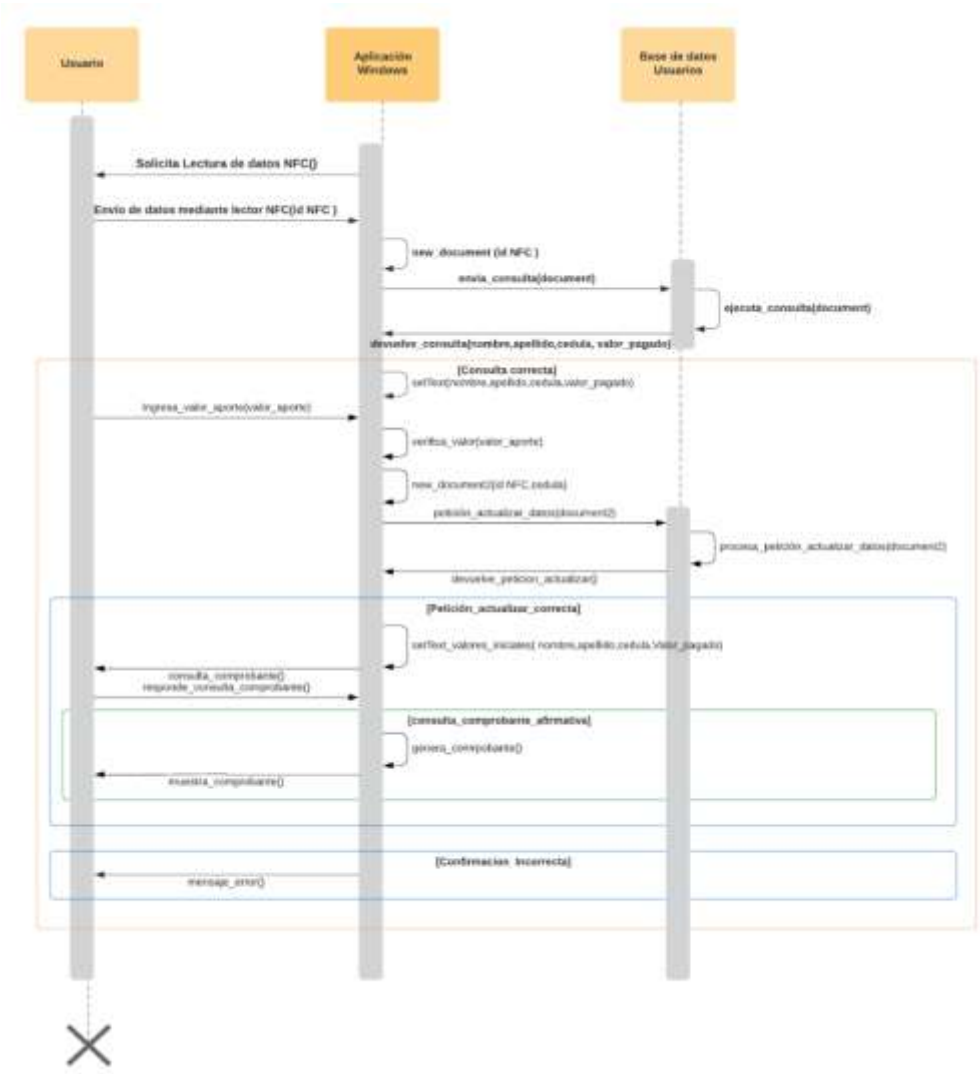
En este apartado, es indispensable la configuración del campo de pago, puesto que solo se permite el ingreso de números para evitar problemas en el sistema. Una vez registrado el pago, el estudiante tendrá la opción de escoger si requiere que se envíe un comprobante de pago o no y con ello se registra su aporte en la base de datos y se actualiza la tabla reflejándose en esta.

Finalmente, si se requiere, es posible generar un Excel con todos los aportes registrados en el mismo mediante el botón de "Generar Excel" visualizado en la **Figura *32***, cumpliendo los parámetros que se habían establecido previamente para dicha aplicación.

A continuación, se muestra un diagrama de secuencia de un determinado caso de uso con el fin de comprender de mejor manera el funcionamiento que tiene la aplicación, para ello, como caso de estudio se contempla la acción de registrar un nuevo aporte para un determinado usuario, esto se muestra en la **Figura *33***.

Figura 33

Diagrama de secuencia caso de uso aplicación aportes voluntarios



Al finalizar el desarrollo se da una breve descripción del funcionamiento del caso de uso presentado en la sección anterior mediante el uso de pseudocódigo, este tiene la finalidad de resumir el código sin utilizar demasiados tecnicismos, las diferentes variables y los respectivos condicionales utilizados se observan en el pseudocódigo 4.

Pseudocódigo 4: Aplicación Aportes Voluntarios, caso de uso correspondiente a registro nuevo aporte.

```
1. Input: ID_NFC = Valor identificador obtenido mediante lector NFC
2. Validar ID_NFC
3. Crear filtro utilizando ID NFC
4. Realizar Solicitud_datos a la base de datos usuarios utilizando filtro
5. if (Solicitud_datos = true)
6.     Mostrar datos de nombre
7.     Mostrar datos de apellido
8.     Mostrar datos de cedula
9.     Mostrar datos de valor_aporte
10.    Input: Valor_Aporte = Valor de aporte a cancelar
11.    Generar un filtro utilizando cedula
12.    Solicitud_actualizar utilizando filtro y Valor_Aporte
13.    if Solicitud_actualizar == correcta
14.      Eliminar variables usadas
15.      Realizar pregunta de imprimir Comprobante
16.      Input: Respuesta_Comprobante = Eleccion si o no pregunta realizada
17.      if Comprobante = si)
18.          Generar comrpobante
19.          Imprimir comprobante
20.    else
21.    Mostrar mensaje error
22. Fin del proceso
```

3.8.5 Desarrollo Aplicación Gestión de Inventario

El desarrollo de una aplicación de inventario es crucial, esto con la finalidad de poder organizar de mejor manera aquel inventario que posee la asociación CITEL-CIEROM.

Para el desarrollo de dicha aplicación es necesario considerar algunos aspectos, tales como:

- Se registre a quien se realiza el préstamo y quién es el responsable de realizar dicho préstamo.
- Llevar un registro de inventario de lo que posee la Asociación
- Agregar nuevos artículos a dicho inventario

Para cumplir con los requisitos presentados anteriormente, inicialmente se realiza la interfaz gráfica sobre la cuál va a trabajar el programa la cual se la visualiza en la **Figura *34***. Esta primera interfaz se encuentra enfocada a cumplir con el primer parámetro.

Figura 34
Interfaz gráfica aplicación gestión inventario 1/3

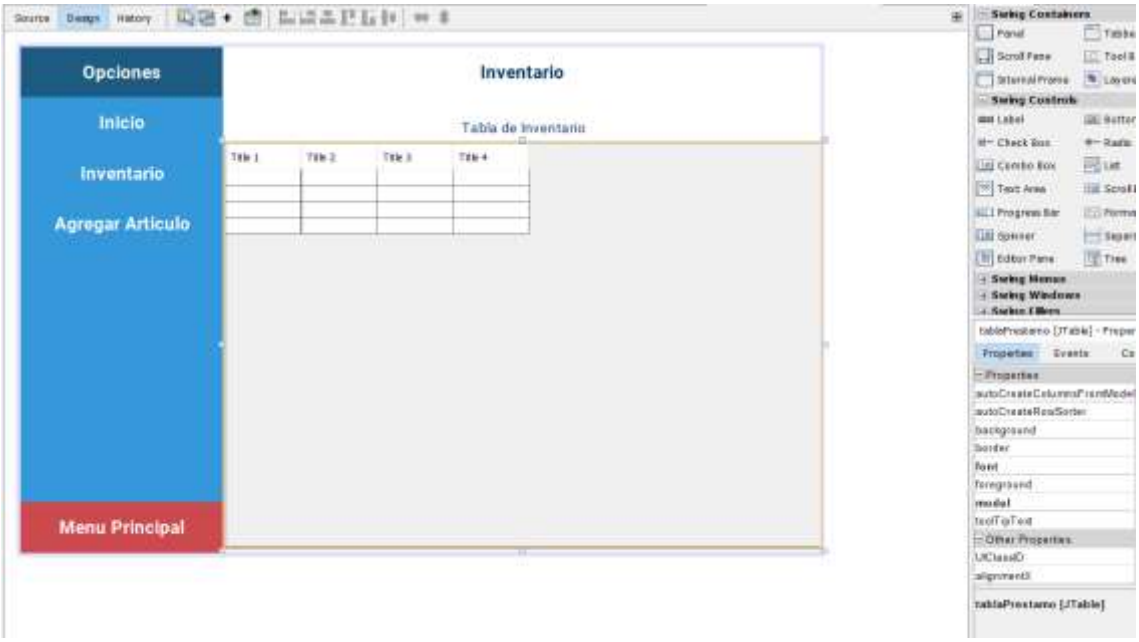
Al visualizar la figura en la que se muestra la interfaz, se describe el funcionamiento de la misma, para ello, inicialmente se debe conectar el Lector NFC desarrollado anteriormente.

Posteriormente la aplicación empieza a recibir cualquier valor que sea leído mediante el lector NFC, una vez obtenido el Id NFC, la aplicación realiza una consulta a la base de datos y de este modo clasifica los datos correspondientes, ya sean estos para el usuario a quién se va a prestar o el artículo. Posteriormente se realiza una consulta dentro de la base de datos para aquellos usuarios responsables y son mostrados dentro de la lista.

Una vez se completen los campos correspondientes a usuario, artículo y responsable, el sistema genera un nuevo registro con estos para constatar que dicho

artículo se encuentra prestado. Con ello es posible generar un registro preciso de todos los elementos disponibles. Para mostrar aquel registro de inventario, se tiene una nueva interfaz que se visualiza en la **Figura *35***.

Figura 35
Interfaz gráfica aplicación gestión inventario 2/3

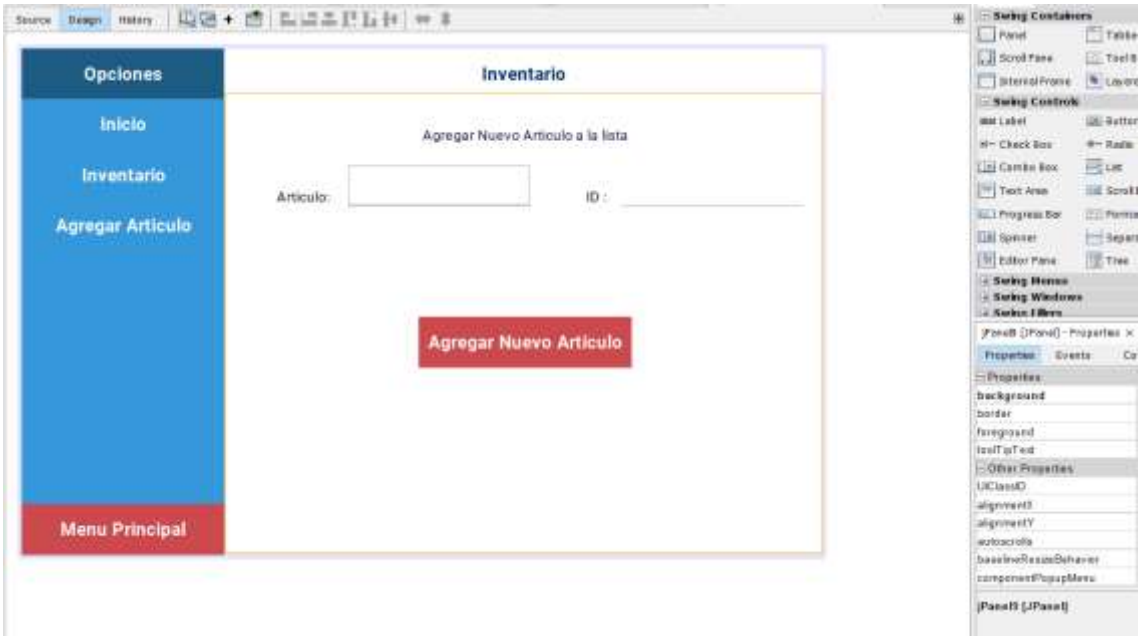


En esta interfaz se puede visualizar una tabla, la cual se llena con todos los elementos que se encuentren registrados en la base de datos de MongoDB Atlas.

Dicha tabla, se actualiza cada vez que se abre la interfaz mostrando los artículos que se encuentran disponibles y también aquellos que han sido prestados con los parámetros mencionados anteriormente.

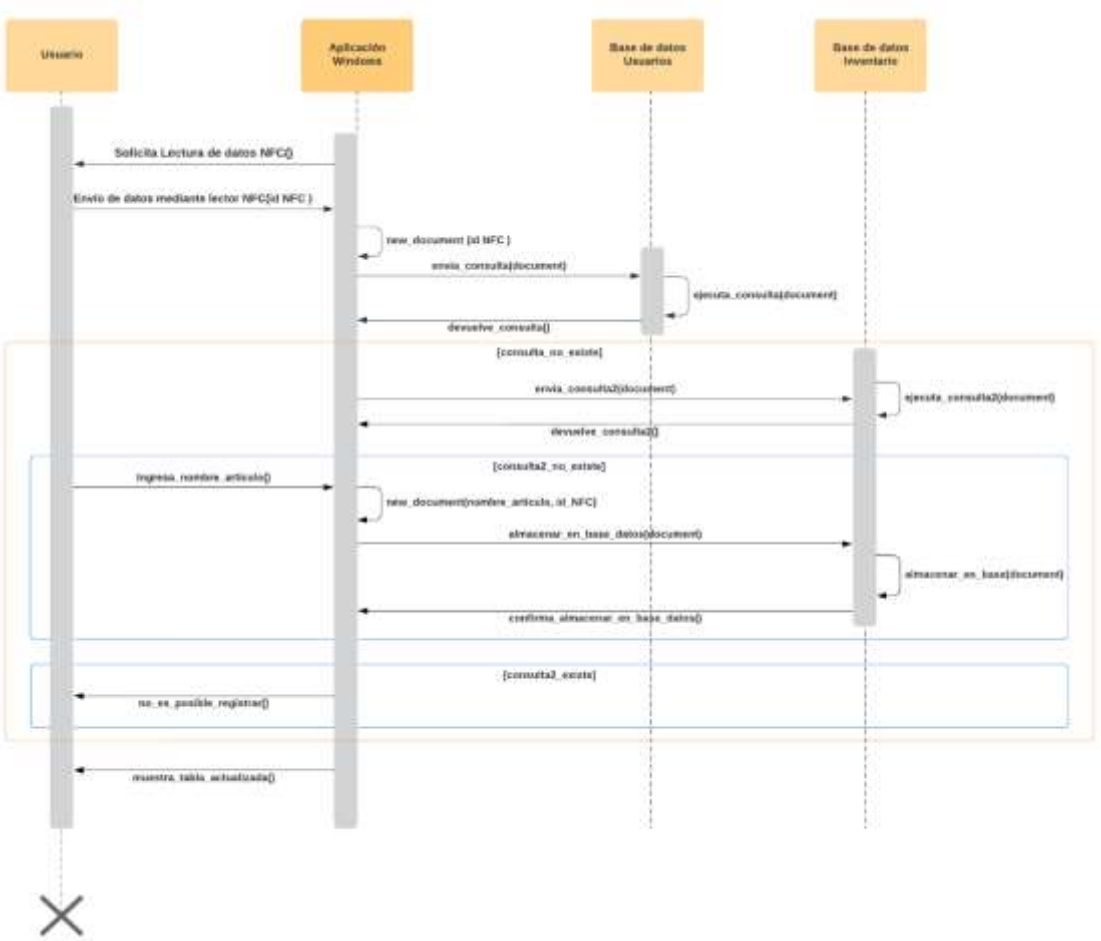
Finalmente se tiene una última interfaz que se visualiza en la **Figura *36*** que permite agregar nuevos artículos al sistema, dicha interfaz también hace uso del Lector NFC desarrollado y su funcionamiento es sencillo ya que simplemente se registra el nombre del artículo, el identificador NFC y se realiza una consulta de si existe o no dicho artículo, en caso de que ya exista uno, el sistema nos arroja un mensaje de advertencia, pudiendo solamente registrar artículos que no hayan sido ingresados anteriormente.

Figura 36
Interfaz gráfica aplicación gestión inventario 3/3



El diagrama de secuencia da una mejor visión del cómo se relacionan los diferentes elementos en función del tiempo, la **Figura *37*** muestra el diagrama de secuencia correspondiente al caso de uso con el que se realiza el registro de un nuevo artículo de inventario en la aplicación desarrollada.

Figura 37
Diagrama de secuencia caso de uso aplicación gestión inventario



Finalmente, se presenta el código simplificado, desarrollado para el caso de uso mostrado en la sección anterior. Dicho código pretende explicar de forma precisa el uso de las variables y condicionales, así como también entradas de datos de la aplicación con el respectivo caso de uso sin la necesidad de utilizar demasiados tecnicismos de un lenguaje de programación.

Pseudocódigo 5. Aplicación Gestión Inventario-Registro nuevo articulo inventario

1. **Input:** ID_NFC = Valor identificador obtenido mediante lector NFC
2. Crear filtro utilizando ID_NFC
3. Realizar Consulta_datos a la base de datos usuarios utilizando filtro
4. **if** (Consulta_datos = false)
5. Realizar Consulta_datos2 a la base de datos inventario utilizando filtro
6. **if** (Consulta_datos2= false)
7. **Input**: Ingresa nombre articulo de inventario

8. Crear nuevo documento utilizando ID_NFC y nombre
9. Almacenar documento en base de datos inventario
10. Mostrar mensaje de confirmacion
11. Mostrar tabla atualizada
12. **Fin del proceso**

3.8.6 Unificación Aplicaciones Desarrolladas

Para realizar la unificación de las diferentes aplicaciones que se han desarrollado, se crea una nueva aplicación y está debe ser capaz de redirigir a las anteriormente creadas.

Para ello utilizando el entorno de desarrollo de NetBeans, se genera una nueva clase JFrame y al igual que en anteriores secciones, se configura la interfaz que será utilizada.

En este caso se observa que la interfaz contiene un elemento de lista, dicho elemento permite escoger el puerto serial por el cuál será transmitida la información desde el lector NFC hacía las aplicaciones, esto es fundamental, ya que no se pueden ejecutar varías aplicaciones a la vez por el mismo puerto serial. Con ello, la **Figura *38*** muestra lo mencionado anteriormente.

Figura 38
Interfaz aplicación principal

Posteriormente se programa cada uno de los botones para que redirija a las aplicaciones que han sido desarrolladas anteriormente. De esta manera, se obtiene un sistema unificado utilizando software para que todos los sistemas pequeños conformen un nuevo sistema centralizado.

3.9 Desarrollo de base de datos.

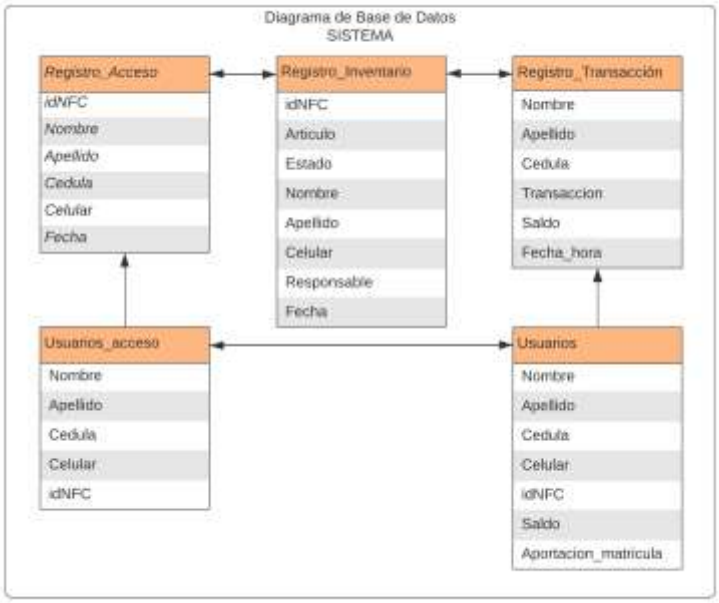
En esta sección se muestra todo lo relacionado con el desarrollo de la base de datos en la nube, para ello de manera inicial de debe crear una base de datos que contenga todos los elementos de cada una de las aplicaciones de manera centralizada.

De este modo, esta sección aborda de manera inicial la creación de la base de datos y posteriormente como se realiza la respectiva conexión de las aplicaciones para que estas funcionen de manera adecuada con la base de datos alojada en la nube.

Dado que todos los sistemas han sido desarrollados previamente de manera individual, cada uno de estos, tiene la necesidad de tener su propia colección dentro de la base de datos. Con ello es posible definir un esquema el cuál seguirá el desarrollado de la

base de datos del sistema propuesto. La **Figura *39*** muestra el esquema de diagrama de base de datos.

Figura 39
Diagrama de base de datos



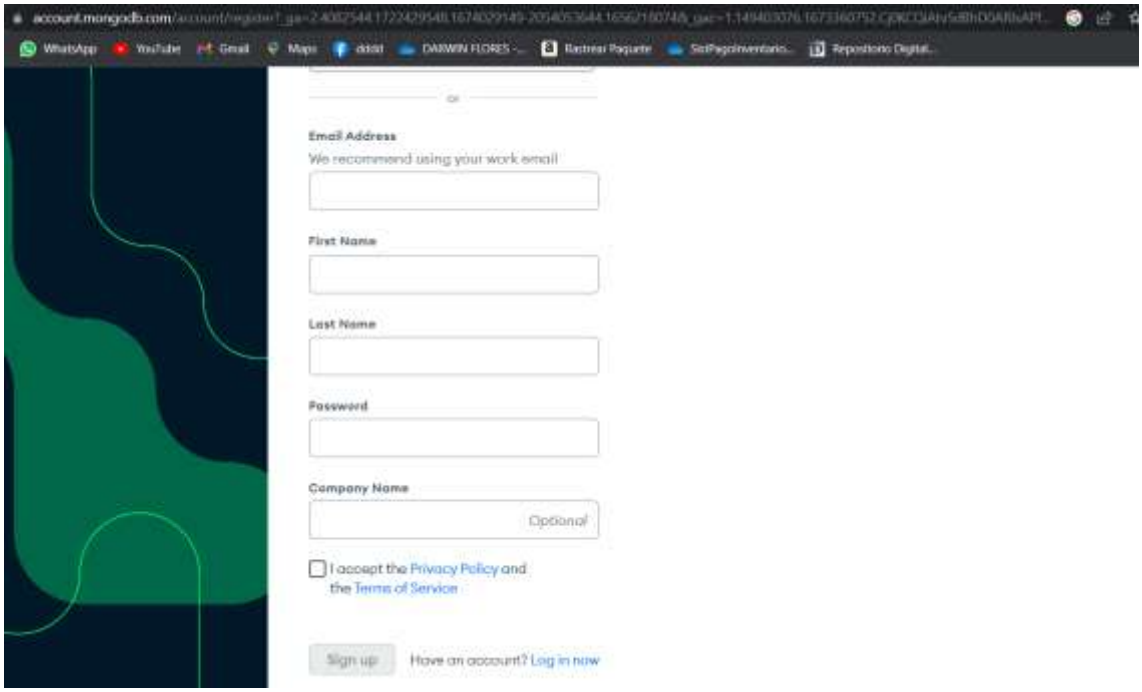
Nota: Cada elemento de la base de datos, forma parte de una colección de elementos y cada colección de elementos forma parte del sistema total de la base de datos.

3.9.1 Creación de Base de Datos.

Una vez se ha definido la base de datos a utilizar, en este caso, MongoDB, se realiza la creación de la misma. De este modo, se plantea utilizar el servicio MongoDB Atlas que proporciona una base de datos gratuita que es alojada en la nube.

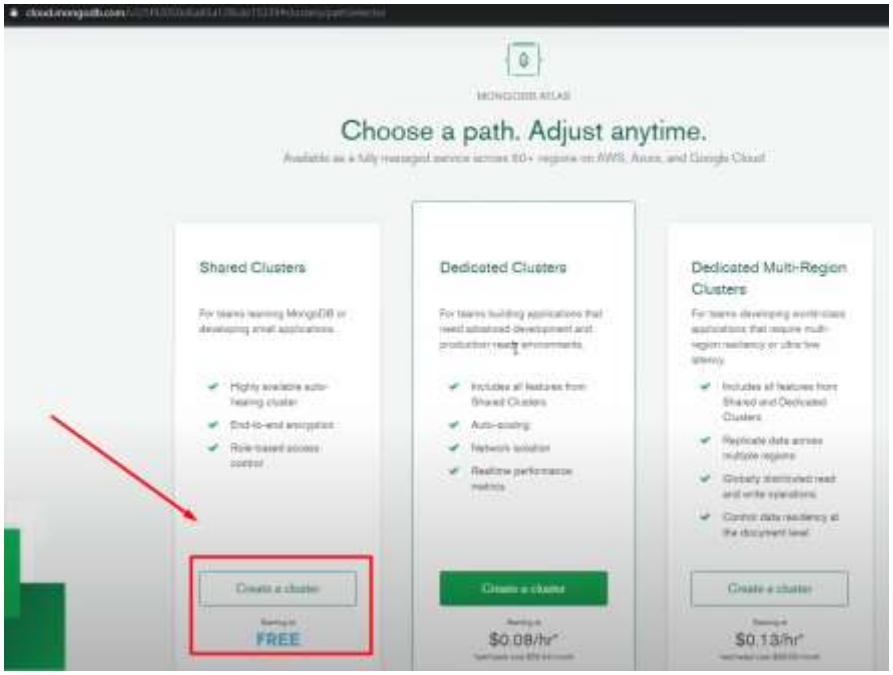
El primer paso para crear una base de datos alojada en la nube es registrarse en la página web de MongoDB tal como se visualiza en la **Figura *40***. Para registrarse se debe llenar los campos de email, nombre, apellido y una contraseña y aceptar los términos y condiciones de uso.

Figura 40
Registro en MongoDB



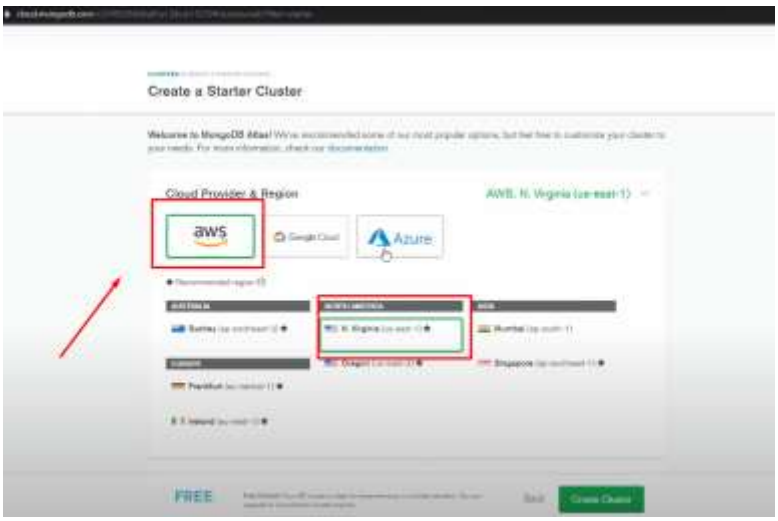
Una vez registrados, se debe seleccionar una opción de *Cluster*, en este apartado es posible visualizar algunas opciones pagadas, sin embargo, la opción gratuita nos brinda espacio suficiente para el desarrollo e implementación del presente proyecto, por lo que se selecciona dicha opción, tal como se muestra en la **Figura *41***.

Figura 41
Creación cluster MongoDB Atlas



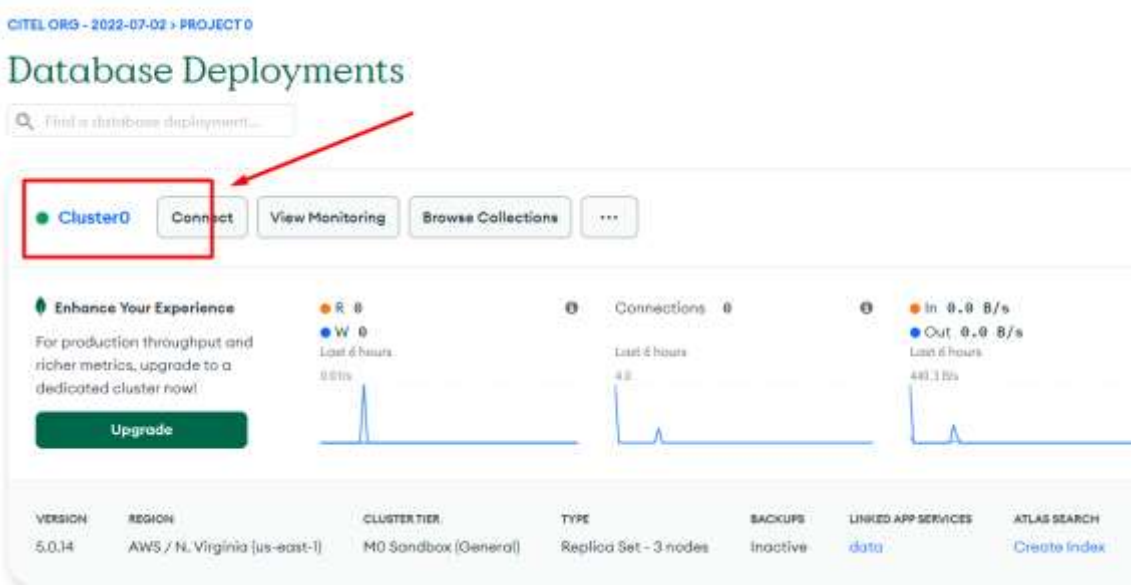
Posteriormente se escoge un proveedor y la zona en la cual estará ubicado el *cluster*, esto es importante ya que, si se escoge una zona lejana, el tiempo de conexión será mayor. Sin embargo, actualmente MongoDB no ofrece servidores en Latinoamérica, por lo que se escoge un servidor ubicado en Norte América, tal como se visualiza en la **Figura *42***.

Figura 42
Selección de región en MongoDB Atlas



Con ello ya se tendrá creado el *Cluster* que se va a utilizar y se logra visualizar en la **Figura *43***.

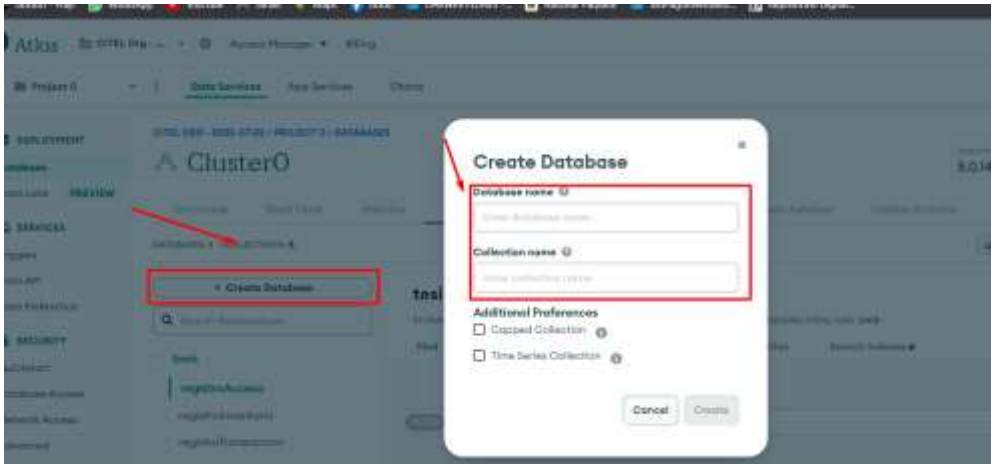
Figura 43
Cluster Creado



Finalmente, el último paso para tener una base de datos funcional es crear una base de datos dentro del *Cluster* y en ella crear una colección, esto se realiza ingresado en el botón “Create Database”, una vez ingresado nos solicitará un nombre de base de datos y un nombre de colección donde serán almacenados los documentos, tal como se muestra en la **Figura *44***.

Figura 44

Creación base de datos y colección MongoDb Atlas



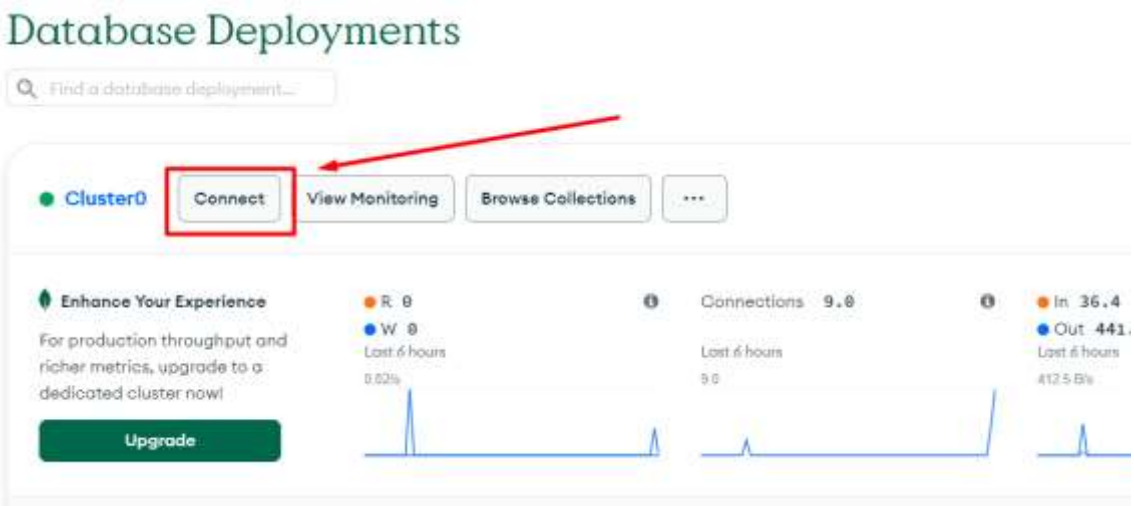
Con esto, ya se tiene creado la base de datos funcional para el proyecto, por lo que el siguiente paso es realizar la conexión entre la base de datos y las aplicaciones anteriormente desarrolladas.

3.9.2 Conexión Base de Datos

En esta sección se muestra el proceso a seguir para la conexión de aplicaciones con la base de datos alojada en la nube, para ello, MongoDB Atlas nos da diferentes opciones.

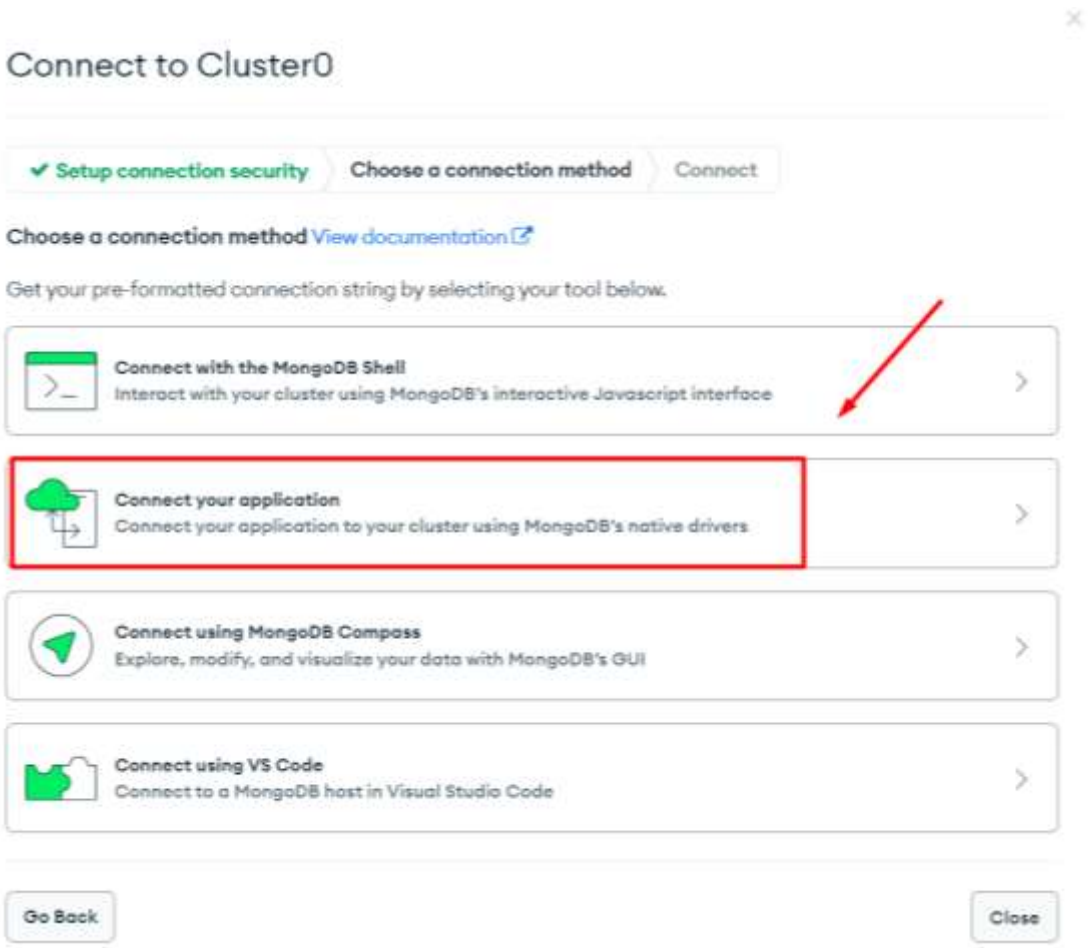
Para conectar la base de datos con aplicaciones, inicialmente se realiza clic en el botón "Connect" tal como se visualiza en la **Figura *45***.

Figura 45
Conexión base de datos con aplicaciones



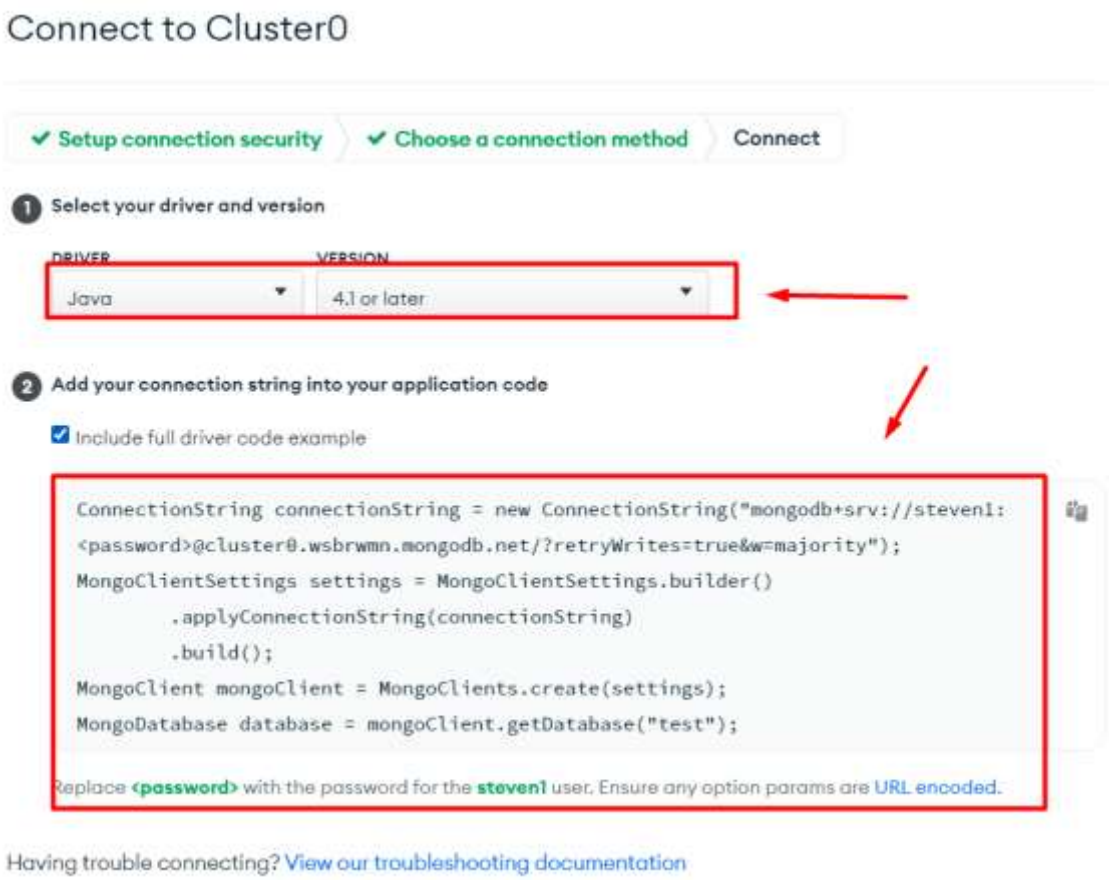
Posteriormente se debe escoger como queremos conectarnos a la base de datos, en este caso, se elige la opción de conectar aplicaciones tal como se visualiza en la **Figura *46***, ya que este método permitirá conectar las aplicaciones desarrolladas en los diferentes lenguajes de programación.

Figura 46
Conectar aplicación con MongoDb Atlas



Para finalizar se escoge el lenguaje de programación en el que se realiza las aplicaciones anteriores y la respectiva versión que se está utilizando como se muestra en la **Figura *47***, con ello, se genera un archivo de configuración que permite la conexión desde la aplicación desarrollada.

Figura 47
Conexión aplicaciones a MongoDB Atlas



De esta manera, se agrega el código proporcionado por MongoDb Atlas a las aplicaciones desarrolladas en Java y se tendrá conexión entre la aplicación y la base de datos alojada en la nube.

3.10 Implementación Prototipos

Al concluir la etapa de diseño, se realiza la implementación del sistema correspondiente a la integración de todos los elementos electrónicos y la posterior verificación del funcionamiento con el fin de corregir errores en caso de existir.

Para ello, inicialmente se desarrolla la carcasa del prototipo en el que serán ubicados los diferentes elementos electrónicos, para cumplir dicho objetivo se realiza el diseño utilizando la herramienta de modelado 3D fusión 360, en la **Figura *48*** se visualiza

el diseño de la carcasa del prototipo obteniendo diferentes vistas de el mismo, inicialmente una imagen del dispositivo cerrado, en la siguiente, se tiene el mecanismo de apertura que tiene el dispositivo y finalmente la ubicación de los elementos dentro del mismo, tanto en la parte inferior, como en la parte superior del prototipo.

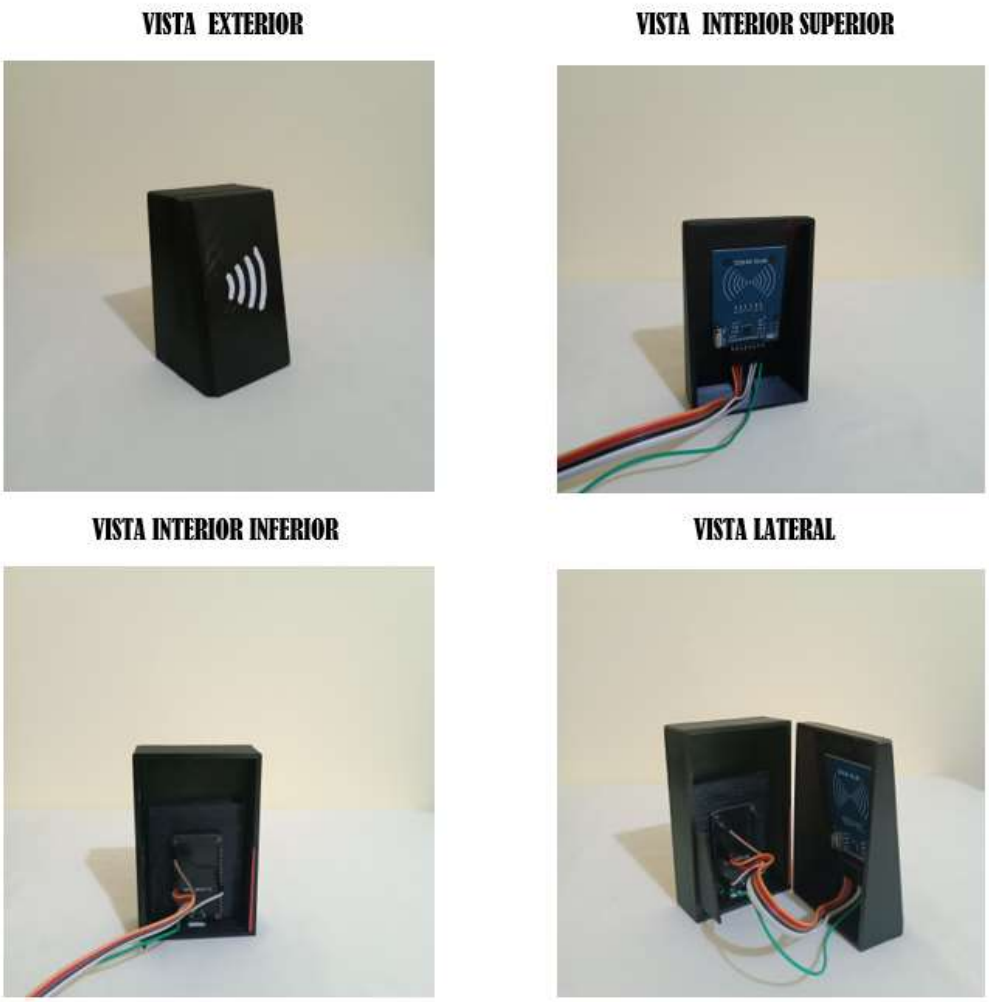
Figura 48
Diseño carcasa prototipo NFC

Nota: La imagen muestra diferentes perspectivas del modelado 3D que se ha desarrollado con el fin de dar una mejor perspectiva del prototipo.

De esta manera, se realiza la interconexión de todos los elementos electrónicos y ubicarlos tal como se muestra en la **Figura *48***. La **Figura *49*** muestra el prototipo de

lector NFC desarrollado visto desde diferentes perspectivas, entre ellas se tiene: exterior, interior superior, interior inferior y lateral.

Figura 49
Prototipo Lector NFC

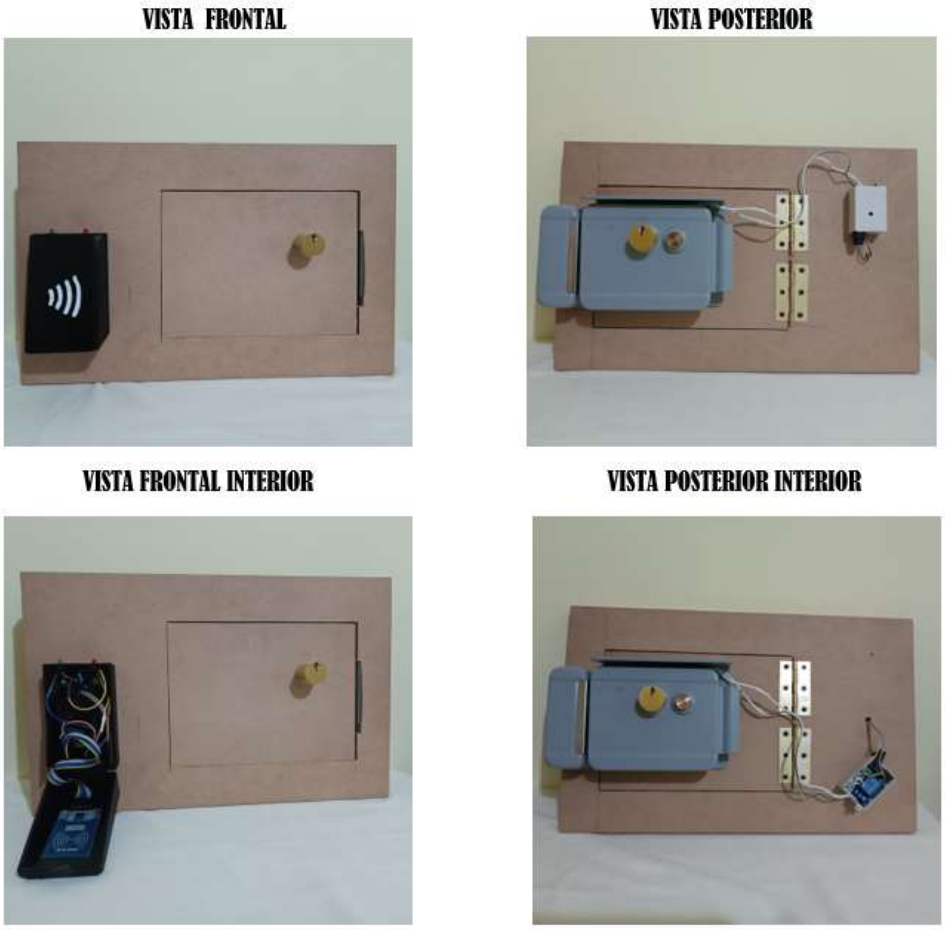


Nota: La figura muestra el desarrollo del prototipo impreso en 3D con los componentes debidamente interconectados.

Al finalizar el prototipo lector NFC, se realiza la interconexión de los elementos electrónicos del prototipo de control de acceso y se ubica los elementos tal como se muestra en la **Figura *49*** utilizando el mismo prototipo de carcasa desarrollada para el

lector NFC. La **Figura *50*** muestra diferentes perspectivas, entre ellas: frontal, posterior, interior frontal y el interior de la parte posterior.

Figura 50
Prototipo Control Acceso



Una vez finalizada la integración de hardware y software del prototipo desarrollado, se realizan pruebas iniciales para comprobar su funcionamiento.

3.11 Pruebas Iniciales del sistema

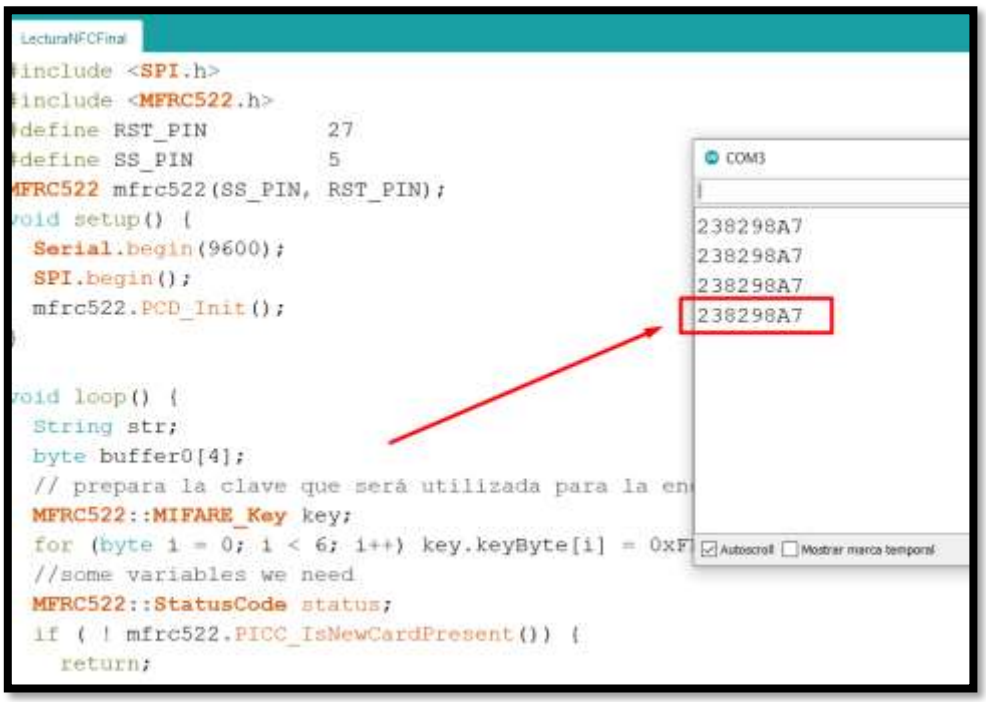
Esta sección comprende los resultados del sistema de manera individual, de este modo, las pruebas estarán divididas en pruebas de hardware y software, con ello se busca verificar el funcionamiento correcto de las funciones que han sido programadas.

3.11.1 Pruebas Iniciales de Hardware.

La finalidad de las pruebas iniciales es, verificar y validar el funcionamiento del lector NFC desarrollado y también el del control de acceso, para ello se utiliza el monitor serial del IDE de Arduino, ya que este permite visualizar los eventos que ocurren dentro del sistema.

Para verificar que el funcionamiento del lector sea correcto, se conecta el dispositivo desarrollado al PC, posteriormente se acerca una tarjeta NFC al prototipo y este deberá leer correctamente el identificador de la tarjeta, tal como se muestra en la **Figura *51***.

Figura 51
Lectura datos NFC.

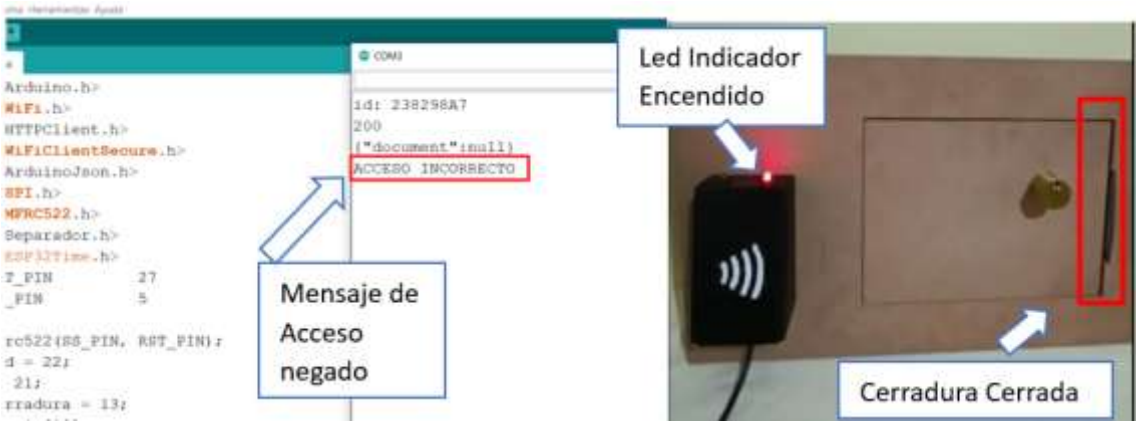


La siguiente prueba consiste en verificar el funcionamiento del control de acceso desarrollado, para ello, se tienen dos posibles casos, el primero deberá verificar que el control de acceso no funcione al utilizar una tarjeta que no se encuentre registrada y el segundo caso deberá verificar que el control de acceso se active con una tarjeta registrada.

Para visualizar de mejor manera las pruebas, también se hace uso del monitor serial que proporciona el IDE de Arduino, en la **Figura *52***, se visualiza como el control de acceso realiza la lectura de una tarjeta no registrada y posteriormente se muestra un mensaje en el monitor serial de acceso incorrecto, además, es posible visualizar que se enciende el led indicador rojo y que la cerradura se mantiene cerrada.

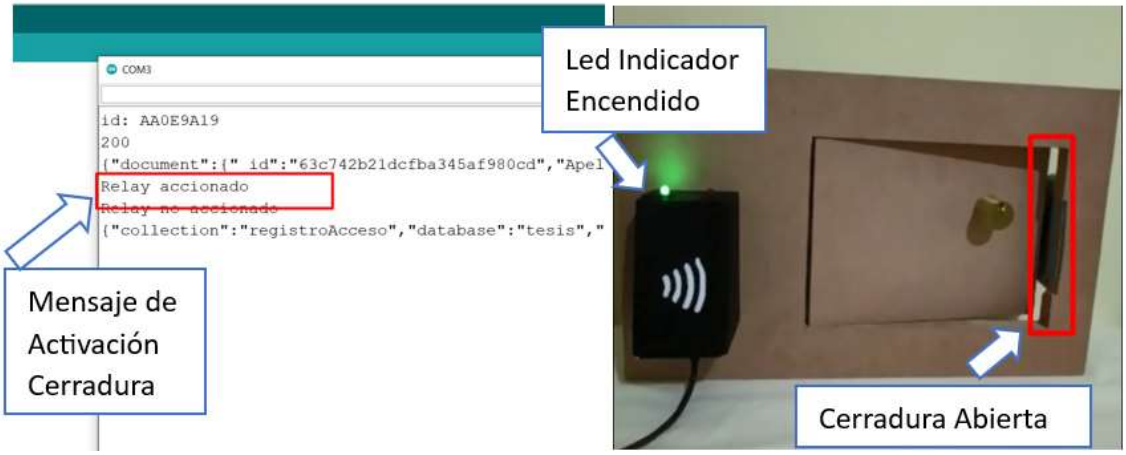
Figura 52

Control Acceso Cerrado



Para la prueba de control con tarjeta registrada, la **Figura *53*** muestra cómo se lee una tarjeta registrada y posteriormente se visualiza que genera un mensaje diferente, este indica que el relé ha sido activado y por tanto la cerradura se debe abrir, de la misma manera, el control de acceso enciende el led indicador y la cerradura se abre.

Figura 53
Control Acceso Abierto

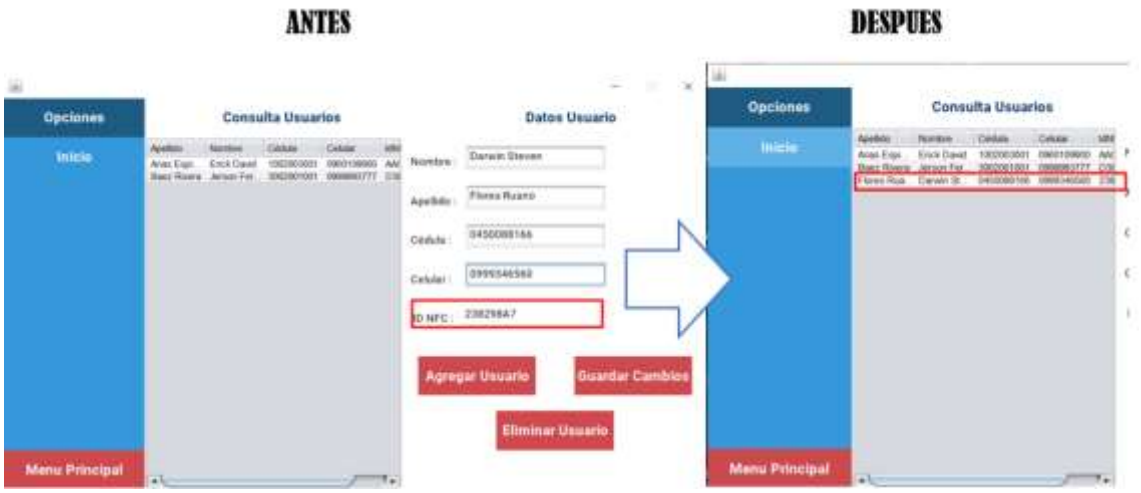


3.11.2 Pruebas Iniciales de Software.

Para verificar la funcionalidad del software de las aplicaciones desarrolladas para el entorno de Windows, se utiliza un PC con las siguientes especificaciones: Procesador Intel Core i5, 16Gb Ram, Sistema Operativo Windows 10.

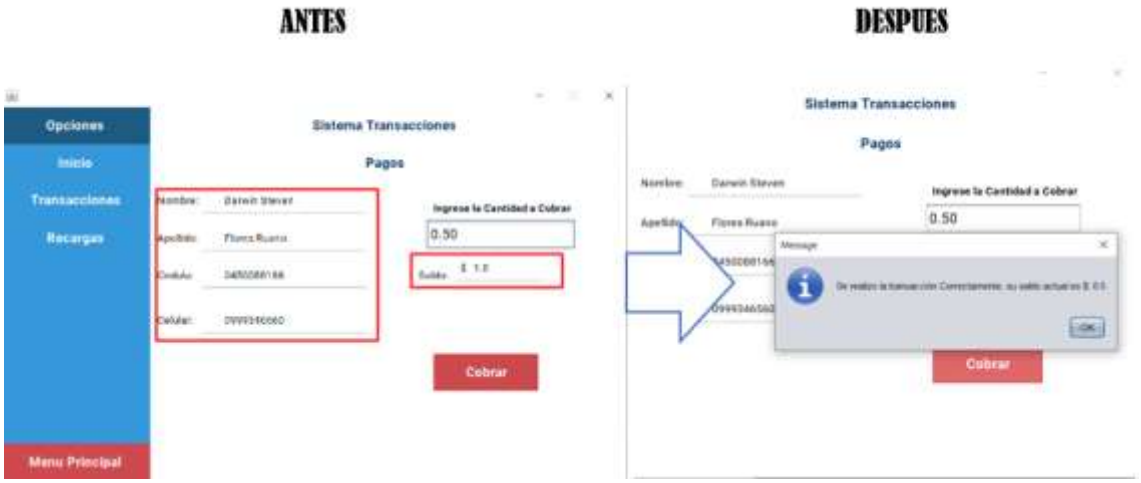
De esta manera, para realizar las pruebas de funcionamiento de las aplicaciones se debe conectar el lector NFC desarrollado y posteriormente abrir la aplicación. La primera prueba, consiste en verificar el registro de usuarios en su respectiva aplicación, para ello inicialmente se llenan los campos solicitados y se utiliza el lector NFC para leer una nueva tarjeta que será asociada, posteriormente se agrega y se obtiene que dicho usuario consta en la lista. En la **Figura *54*** se visualiza el antes y el después del registro.

Figura 54
Registro Usuario



La segunda prueba consiste en realizar una micro transacción de cobro, para ello inicialmente los datos deberán ser obtenidos utilizando la base de datos y el lector NFC y posteriormente se realiza el pago de acuerdo con el saldo disponible, la **Figura *55*** muestra una imagen del funcionamiento y el proceso de lo mencionado.

Figura 55
Micro Transacción Realizada

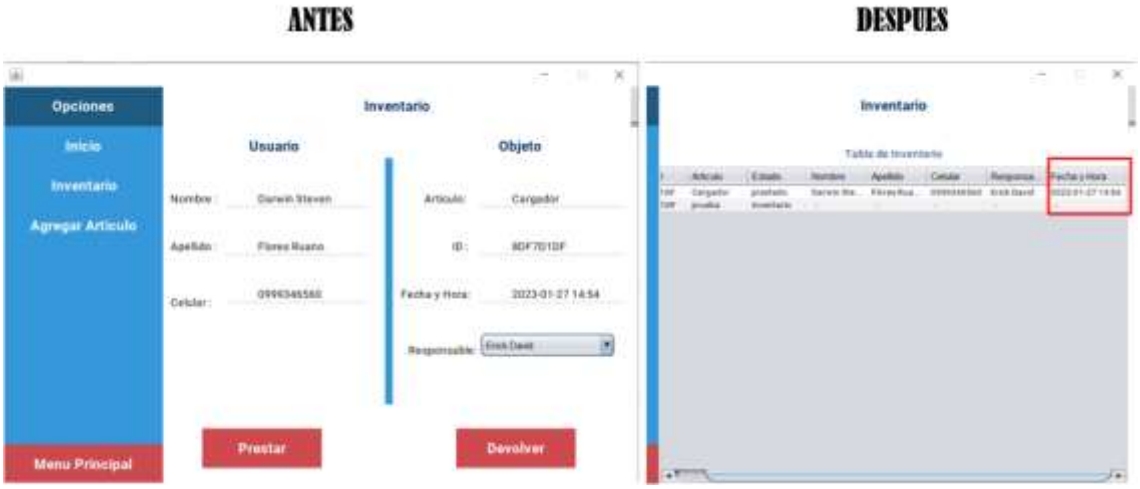


La prueba siguiente consiste en verificar la aplicación correspondiente a los aportes voluntarios, esta aplicación genera un registro de todos los usuarios y se utiliza la tarjeta NFC para agregar un nuevo registro, tal como se visualiza en la **Figura *56***.

Figura 56
Registro Aportes Voluntarios

La siguiente prueba verifica el funcionamiento de la aplicación de inventario, para ello, se utiliza una etiqueta NFC y una tarjeta NFC correspondientes al artículo y al usuario respectivamente, como se visualiza en la **Figura *57***, la aplicación obtiene los datos basándose en el identificador NFC y posteriormente se genera un registro de la solicitud generada.

Figura 57
Registro Inventario NFC



Finalmente, se verifica el funcionamiento de la aplicación de control de acceso visualizando el registro de ingreso que se haya obtenido mediante el control. La **Figura *58*** muestra el registro del acceso realizado.

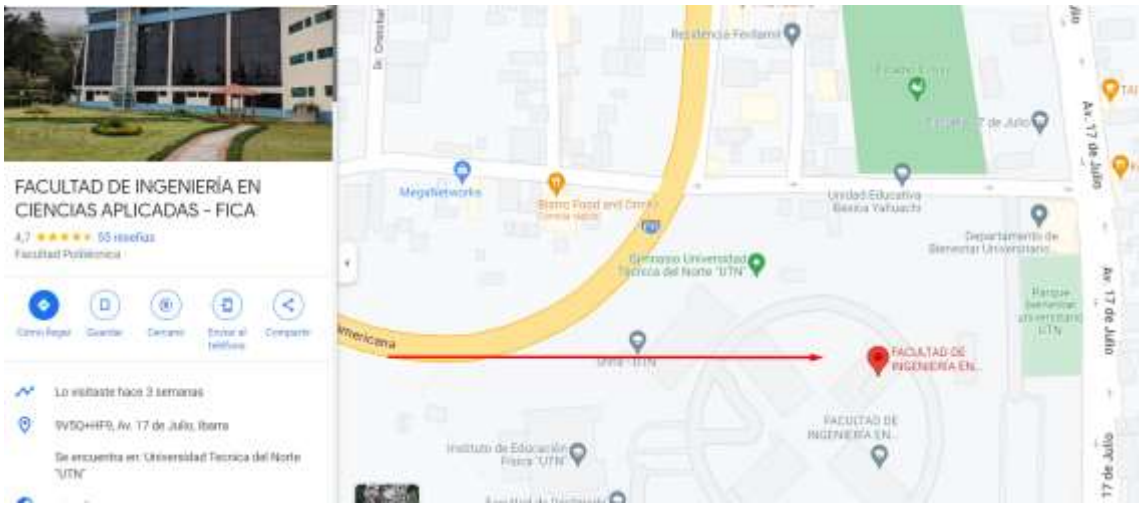
Figura 58
Registro Control de Acceso

Una vez finalizadas algunas pruebas iniciales, se realiza la implementación en el escenario real.

3.12 Implementación del sistema

El sistema desarrollado, se implementa en las instalaciones de la asociación de estudiantes CITEL-CIERCOM, ubicadas en la Universidad Técnica del Norte, en la Facultad de Ingeniería en Ciencias Aplicadas. En la **Figura *59***, se muestra la ubicación de la Facultad de Ingeniería en Ciencias Aplicadas.

Figura 59
Ubicación instalaciones asociación CITEL-CIERCOM



Nota: La figura muestra de manera específica el lugar donde se realiza la implementación y las pruebas correspondientes.

De esta manera inicialmente, se realiza la implementación del sistema de control de acceso, tal como se visualiza en la **Figura *60***, el dispositivo de control de acceso desarrollado es ubicado en la puerta de las instalaciones de la asociación de estudiantes CITEL-CIERCOM, de este modo, también se debe considerar que exista cobertura Wi-FI para el funcionamiento del sistema.

Figura 60
Dispositivo control acceso implementado

Por otra parte, se realiza la implementación del sistema de inventario, para ello se etiquetan los diferentes artículos que posee la asociación CITEL-CIERCOM para posteriormente ser registrados mediante el uso del lector y la aplicación desarrollada. En la **Figura *61*** se observan los artículos con las respectivas etiquetas NFC.

Figura 61
Inventario agregado con etiquetas NFC

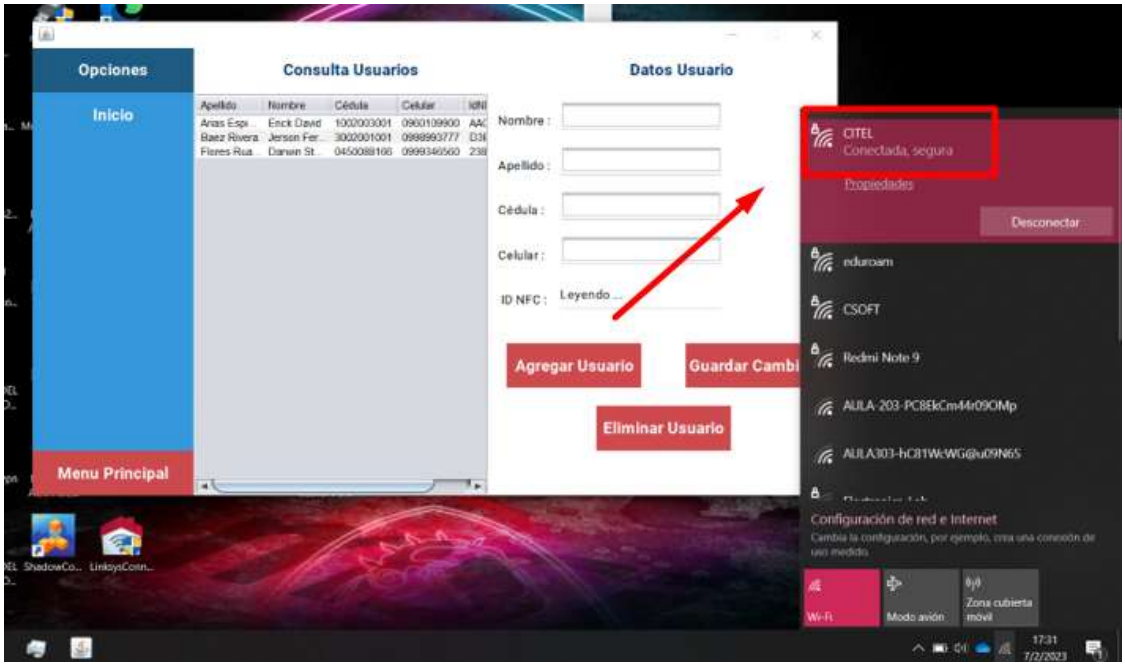
Una vez realizado la implementación de etiquetas para inventario, se realiza la implementación de tarjetas NFC entregando las mismas a la representante de la asociación CITEL-CIERCOM. Para culminar el proceso de implementación del sistema, se realiza la instalación de la aplicación desarrollada en Windows con el respectivo lector NFC para el funcionamiento correcto de la misma. Esto es posible visualizar en la **Figura *62***.

Figura 62
Funcionamiento Lector NFC

En la **Figura *63*** se observa la aplicación de Windows instalada y el funcionamiento mediante el uso de la red Wi-Fi correspondiente a la red que posee la Asociación CITEL-CIERCOM.

Figura 63

Funcionamiento Aplicación Windows



Nota: Con el fin de garantizar que las pruebas se realizan utilizando la red correspondiente es posible visualizar la red a la cual se encuentra conectado.

CAPITULO IV. PRUEBAS Y RESULTADOS

Este capítulo se centra en el funcionamiento del sistema desarrollado basándose en diversas pruebas. Para ello, el capítulo muestra inicialmente las pruebas a realizarse mediante un cronograma y posteriormente se evalúan los beneficios obtenidos que proporciona el sistema. Con esto, se cumplen todos los objetivos planteados de la presente investigación.

4.1 Cronograma de Pruebas

Con el fin de desarrollar de mejor manera las pruebas, se realiza un cronograma de pruebas que permite verificar el funcionamiento en un ambiente real, la **Tabla *17*** muestra algunas pruebas y la duración que tendrá cada una de las mismas.

Tabla 17
Cronograma de Pruebas

Cronograma de Pruebas			
Tipo de Prueba	**Ubicación donde se desarrolla**	**Resultados a obtener**	**Duración**
Prueba 1 Funcionamiento Lector NFC	Instalaciones Asociación CITEL-CIEROM	Esta prueba recopila el funcionamiento del lector NFC.	13 días del 1 al 13 de febrero del 2023
Prueba 2 Funcionamiento Aplicación Registro Usuarios	Instalaciones Asociación CITEL-CIEROM	Esta prueba demuestra el funcionamiento de la aplicación desarrollada con la respectiva base de datos	7 días del 1 al 7 de febrero del 2023
Prueba 3	Instalaciones Asociación CITEL-CIEROM	Esta prueba recopila los datos obtenidos al momento de	7 días del 1 al 7 de febrero del 2023

Funcionamiento Aplicación Control Acceso		evaluar el funcionamiento de la aplicación de control de acceso	
Prueba 4 Funcionamiento Control Acceso	Instalaciones Asociación CITEL-CIEROM	Esta prueba contempla el funcionamiento del dispositivo de control de acceso desarrollado	13 días del 1 al 13 de febrero del 2023
Prueba 5 Funcionamiento Aplicación Inventario	Instalaciones Asociación CITEL-CIEROM	Se evalúa el funcionamiento de la aplicación de inventario	13 días del 1 al 13 de febrero del 2023
Prueba 6 Funcionamiento Aplicación Micro Transacciones	Instalaciones Asociación CITEL-CIEROM	Se evalúa el funcionamiento de la aplicación de micro transacciones	7 días del 1 al 7 de febrero del 2023
Prueba 7 Funcionamiento Aplicación Matriculas	Instalaciones Asociación CITEL-CIEROM	Se evalúa el funcionamiento de la aplicación correspondiente a aporte de matriculas	7 días del 7 al 13 de febrero del 2023

De esta manera, para comprobar correctamente el hardware y software se realizan pruebas de caja negra y pruebas de caja blanca, con ello las pruebas de caja negra pretenden demostrar el funcionamiento de hardware y software de tal manera que solo se conoce el inicio y el resultado sin importar que es lo que sucede de manera técnica, por el contrario las pruebas de caja blanca consisten en analizar y comprobar de manera precisa el funcionamiento de cada elemento desarrollado, en este caso, para realizar las

diferentes pruebas de caja blanca se utilizan diferentes herramientas como ***Junit***, que permite comprobar el funcionamiento de código mediante diferentes métodos en JAVA.

4.2 Prueba 1 Funcionamiento Lector NFC

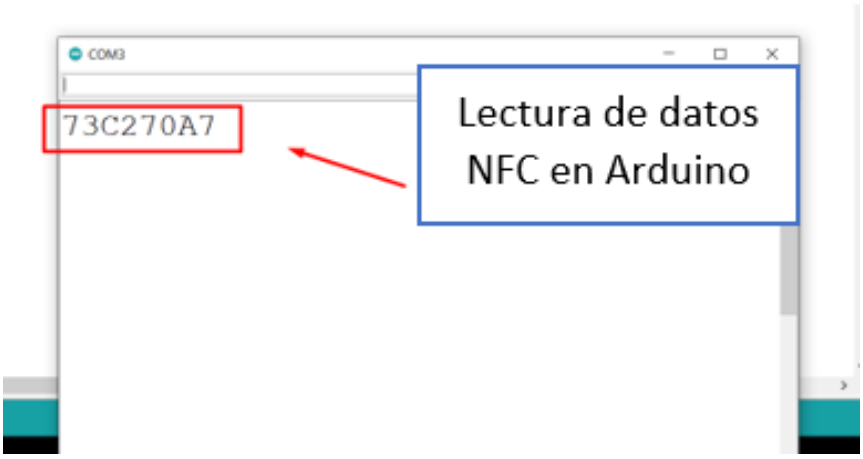
De acuerdo con el cronograma establecido, la primera prueba corresponde al funcionamiento del prototipo de lector NFC que se ha desarrollado, de este modo, se verifica la lectura de datos de las diferentes tarjetas lectoras durante 13 días tal como se describe anteriormente. Para ello la herramienta que se pretende utilizar es el monitor serie que brinda el IDE de Arduino, ya que los datos se envían desde el lector hacía el computador a través de comunicación serial.

Los resultados obtenidos de esta prueba permitirán visualizar la funcionalidad del lector y obtener posibles errores que se presenten en un entorno real, con el fin de realizar los respectivos cambios que mejoren la funcionalidad del lector NFC desarrollado.

4.2.1 Visualización funcionamiento

La prueba que se va a desarrollar consiste en acercar diferentes tarjetas NFC al lector varías veces durante los días de prueba y llevar un registro de forma manual de las lecturas correctas o incorrectas que realice el lector, de este modo, si la lectura se realiza correctamente, se visualizará un código conformado de 8 dígitos en formato Hexadecimal tal como se muestra en la **Figura *64***, de este modo, se realizan dos lecturas de datos diarias comprendidas entre las 10:00 y 15:00h durante los días previstos.

Figura 64
Prueba 1 Lectura de datos NFC



Con los datos obtenidos en los 13 días de pruebas, se obtiene un registro en el que se tiene todas las lecturas realizadas con la respectiva hora y fecha y si la lectura fue correcta o no. La **Figura *65*** muestra todos los datos obtenidos con la respectiva comparación de datos y el resultado final.

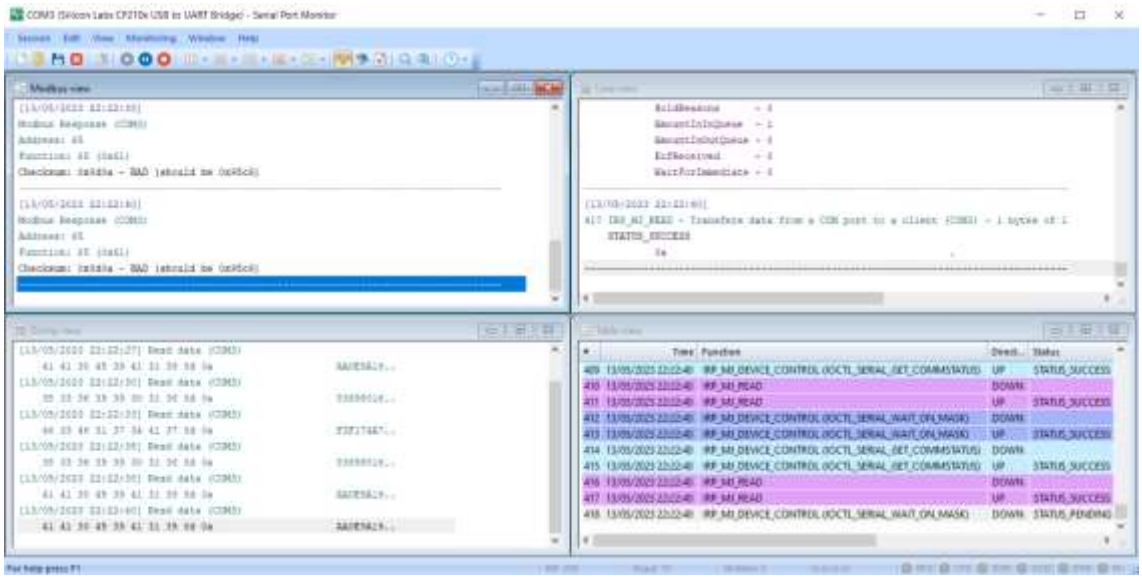
Figura 65
Prueba 1 Datos Obtenidos-Lector NFC

Fecha y Hora	Datos de NFC Reales	Datos NFC Lector	Resultado
1/2/2023 11:58	AA0E9A19	AA0E9A19	Correcto
1/2/2023 13:21	73C270A7	73C270A7	Correcto
2/2/2023 10:33	D3B19814	D3B19814	Correcto
2/2/2023 14:12	031560A7	031560A7	Correcto
3/2/2023 10:47	73C270A7	73C270A7	Correcto
3/2/2023 13:18	238298A7	238298A7	Correcto
7/2/2023 11:03	D3B19814	D3B19814	Correcto
7/2/2023 12:49	AA0E9A19	AA0E9A19	Correcto
8/2/2023 10:47	031560A7	031560A7	Correcto
8/2/2023 14:37	73C270A7	73C270A7	Correcto
9/2/2023 10:28	D3B19814	D3B19814	Correcto
9/2/2023 13:41	238298A7	238298A7	Correcto
10/2/2023 11:08	AA0E9A19	AA0E9A19	Correcto
10/2/2023 12:27	73C270A7	73C270A7	Correcto
13/2/2023 12:45	AA0E9A19	AA0E9A19	Correcto
13/2/2023 14:53	D3B19814	D3B19814	Correcto

Una vez culminado la primera etapa de pruebas, se realizan las denominadas pruebas de caja blanca, estas dan una mejor visión del funcionamiento tomando en

consideración los diferentes procesos que deben suceder para que el dispositivo funcione correctamente. Para ello se utiliza la herramienta ***"Serial Port Monitor"*** que permite verificar todas las comunicaciones que suceden mediante el puerto serial, de este modo se obtiene la **Figura 66**.

Figura 66
Prueba 1 Serial Port Monitor



Mediante la herramienta *"**Serial Port Monitor"*** se determina inicialmente el ***Checksum*** para cada uno de los elementos que han sido transmitidos mediante el lector NFC desarrollado, la herramienta de ***Line View*** permite verificar el estado de cada operación que se ha realizado e identificar el puerto por el que se transmiten los datos, la herramienta ***Dump View*** realiza un volcado de la información antes de ser procesada en este caso se realizan 6 transmisiones de datos en las cuales se verifica que todos los datos son correctos, además verifica que la información se transmite codificada, finalmente la

herramienta de ***Table View*** muestra los diferentes estados en los que se encuentra la comunicación.

4.2.2 Conclusión

Con todos los datos recuperados del registro mostrado en la sección anterior en la **Figura *65***, se visualiza que todos los datos leídos corresponden a los datos almacenados en las diferentes tarjetas NFC, de este modo se logra determinar que el funcionamiento del prototipo de lector NFC es correcto y tiene un porcentaje de funcionamiento del 100%, de la misma manera la herramienta de Serial Port Monitor de la **Figura 66**, muestra información más técnica sobre el funcionamiento del lector NFC desarrollado demostrando su funcionalidad.

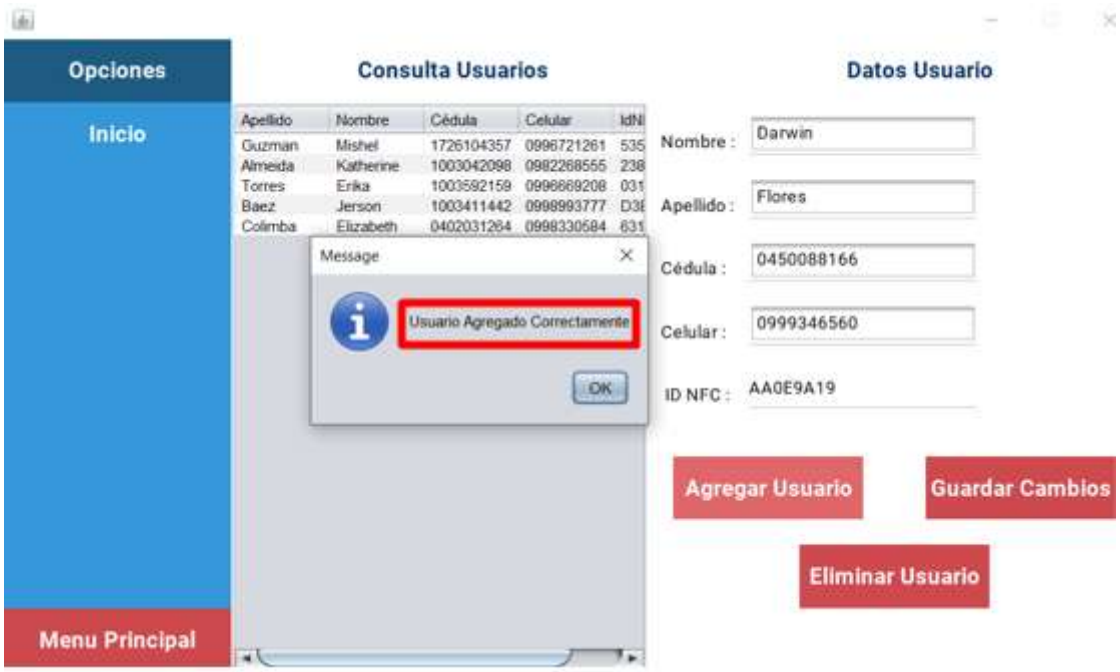
4.3 Prueba 2 Funcionamiento Aplicación Registro Usuarios

En esta sección se verifica el funcionamiento correspondiente a la aplicación de registro de usuarios, dicha prueba tiene una duración de 7 días. Para ello, la prueba debe verificar las diferentes funciones que efectúa la aplicación desarrollada. De esta manera se realizan pruebas correspondientes a las funciones de “Agregar Usuarios”, “Realizar Cambios” y “Eliminar Usuarios” y se muestran los resultados obtenidos.

4.3.1 Visualización funcionamiento

Para la comprobación de funcionamiento de la aplicación, inicialmente se registra un nuevo usuario, para ello se llenan los campos obtenidos y posteriormente se agrega el mismo, esto se visualiza en la **Figura *67***.

Figura 67
Prueba 2 - Registro Usuario

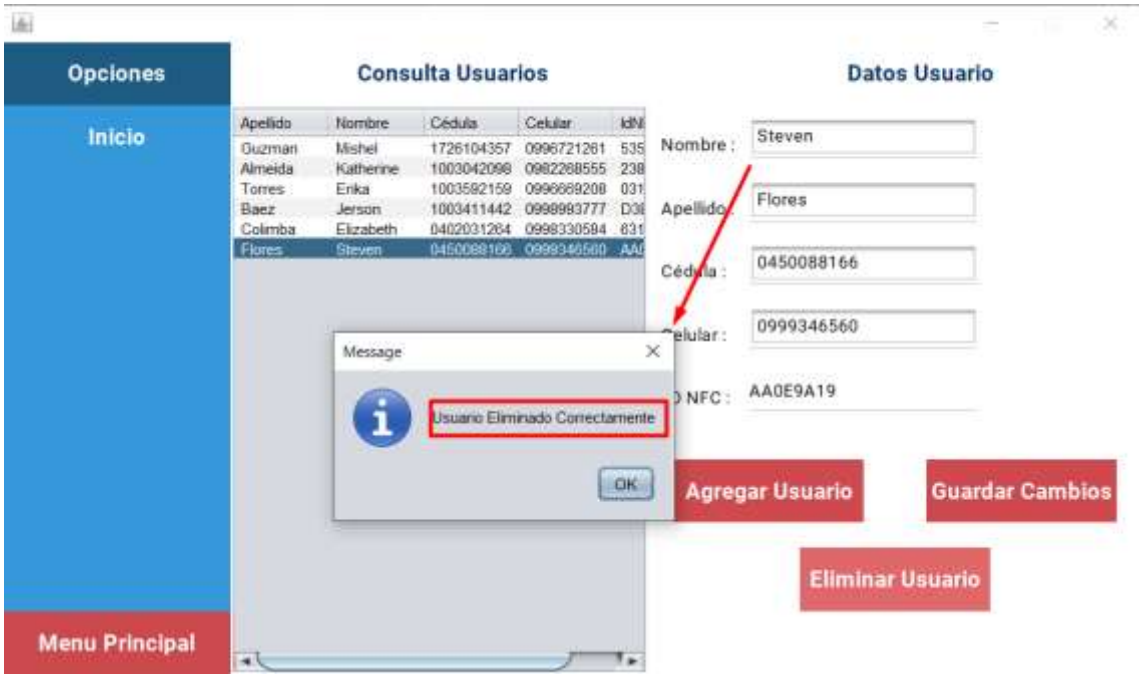


De igual manera se realiza la prueba de funcionamiento de realizar cambios, para ello se establece el cambio necesario a realizar y posteriormente se presiona en el botón correspondiente, la **Figura *68*** muestra el funcionamiento correspondiente de la aplicación con la función respectiva.

Figura 68
Prueba 2 - Actualizar Cambios

También se realizan las pruebas correspondientes a eliminar usuarios, para ello, inicialmente se debe escoger aquel usuario que se desea eliminar y posteriormente escoger el botón correspondiente para eliminar, este realiza toda la programación relacionada eliminando los datos desde la base de datos, es posible visualizar la prueba correspondiente en la **Figura *69***.

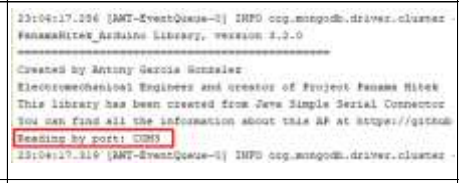
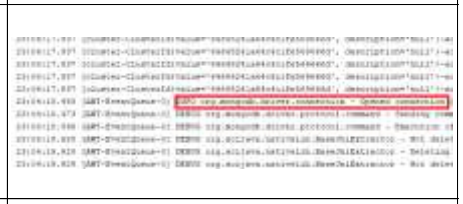
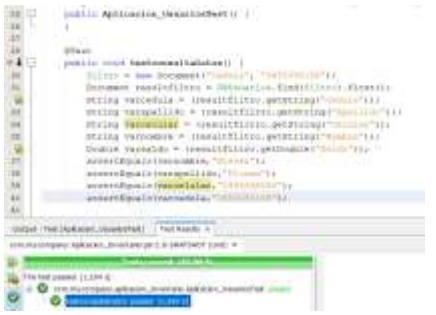
Figura 69
Prueba 2 - Eliminar Usuarios



Las pruebas de interfaz gráfica también son denominadas de caja negra, esto ya que solo muestran la funcionalidad desde un punto de vista de usuario sin conocer todos los procesos que suceden detrás que hacen posible la funcionalidad de la misma. Sin embargo, también existen pruebas que permiten verificar el funcionamiento desde un punto de vista más técnico, dichas pruebas son consideradas como pruebas de caja blanca.

Con la finalidad de comprobar el funcionamiento correcto, se realiza una matriz de pruebas de caja blanca en la que se detallan los casos que serán comprobados mediante la herramienta ***Junit***. La misma se muestra en la **Figura 70**.

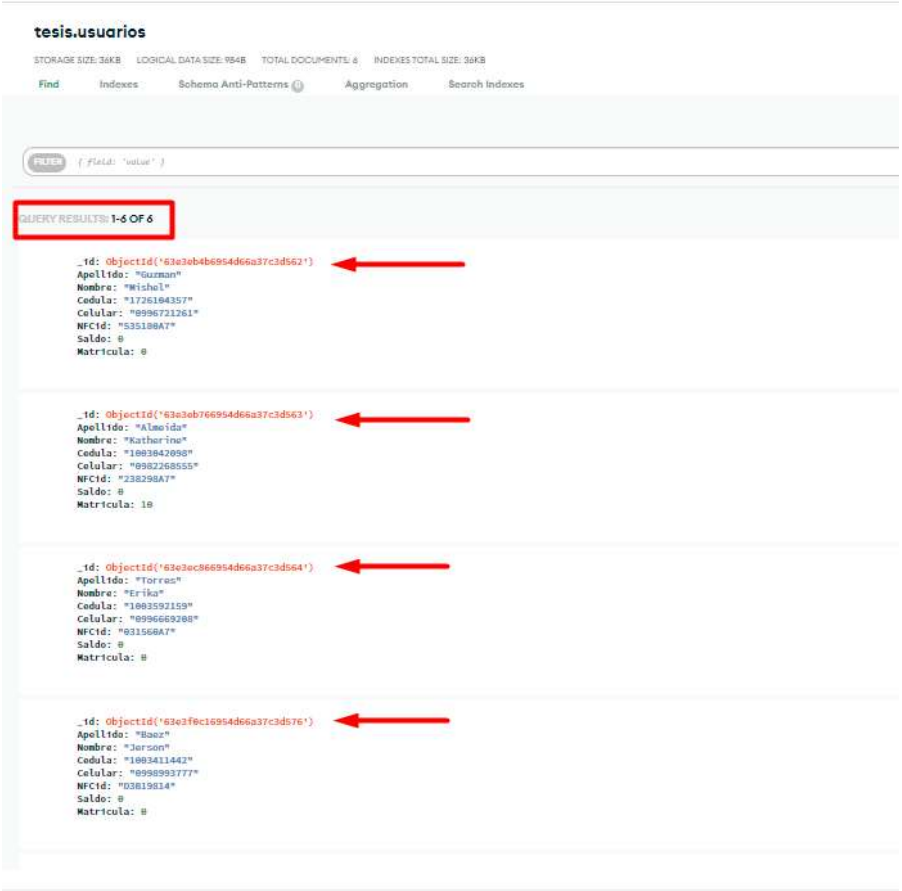
Figura 70
Matriz de Pruebas Aplicación Usuarios

Matriz Pruebas Aplicación Usuarios						
Id	**Caso de Prueba**	**Descripción**	**Resultado Esperado**	**Cumple**	**No Cumple**	**Metodo Verificación**
01	Conexión con Lector NFC	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión de la aplicación y el lector desarrollado sea correcta, para ello, se utiliza un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que el puerto abierto es correcto y se encuentra preparado para recibir los datos que serán transmitidos.			
02	Conexión Base de Datos	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión entre aplicación y la base de datos sea correcta utilizando un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que la conexión con los servidores en los que se aloja la base de datos se realiza correctamente			
03	Registrar Usuario	Mediante la herramienta Junit se verifica el código correspondiente para el registro de usuario, para ello se ejecuta el código en la herramienta y se compara con los resultados que son ingresados manualmente	El resultado esperado será que un nuevo usuario sea agregado en la base de datos, para ello se comparan los datos recuperados de la base de datos con los datos ingresados manualmente, dichos datos contemplan nombre, apellido, cedula y celular, para ello se utiliza el metodo ***assertEquals*** que proporciona Junit			

04	Actualizar Usuario	Mediante la herramienta Junit se verifica el código correspondiente para la actualización de datos, para ello se cambia de nombre un usuario y se verifica posteriormente que se actualice en la base de datos.	El resultado esperado será la actualización de únicamente el nombre de un usuario, para ello, se compara los datos anteriores de la base de datos con los nuevos datos y solo el dato de nombre deberá ser diferente, para esto se utiliza el metodo ***assertEquals*** y ***assertFalse*** que proporciona JUnit			
05	Eliminar Usuario	Mediante la herramienta Junit se verifica el código correspondiente para la eliminación de usuarios, para esto, el código deberá eliminar al usuario que se haya seleccionado previamente	El resultado esperado será que no exista ninguna coincidencia de usuarios en la base de datos, ya que dicho usuario debió ser eliminado mediante el código desarrollado, para ello se utiliza el metodo **assertTrue** que proporciona Junit			

Finalmente, para completar el funcionamiento se realiza la respectiva verificación de los datos que se han generado y almacenado en la base de datos, esto es posible visualizarlo en la plataforma de la base de datos que se encuentra alojada en la nube, esto se muestra en la **Figura *71***.

Figura 71
Prueba 2 Base de datos aplicación usuarios



4.3.2 Conclusiones

Es posible determinar que la aplicación de usuarios cumple con el objetivo principal, el cuál es registrar de manera rápida y eficaz a todos los usuarios que serán participes del sistema, de este modo, se continúa con las pruebas de funcionamiento de las otras aplicaciones.

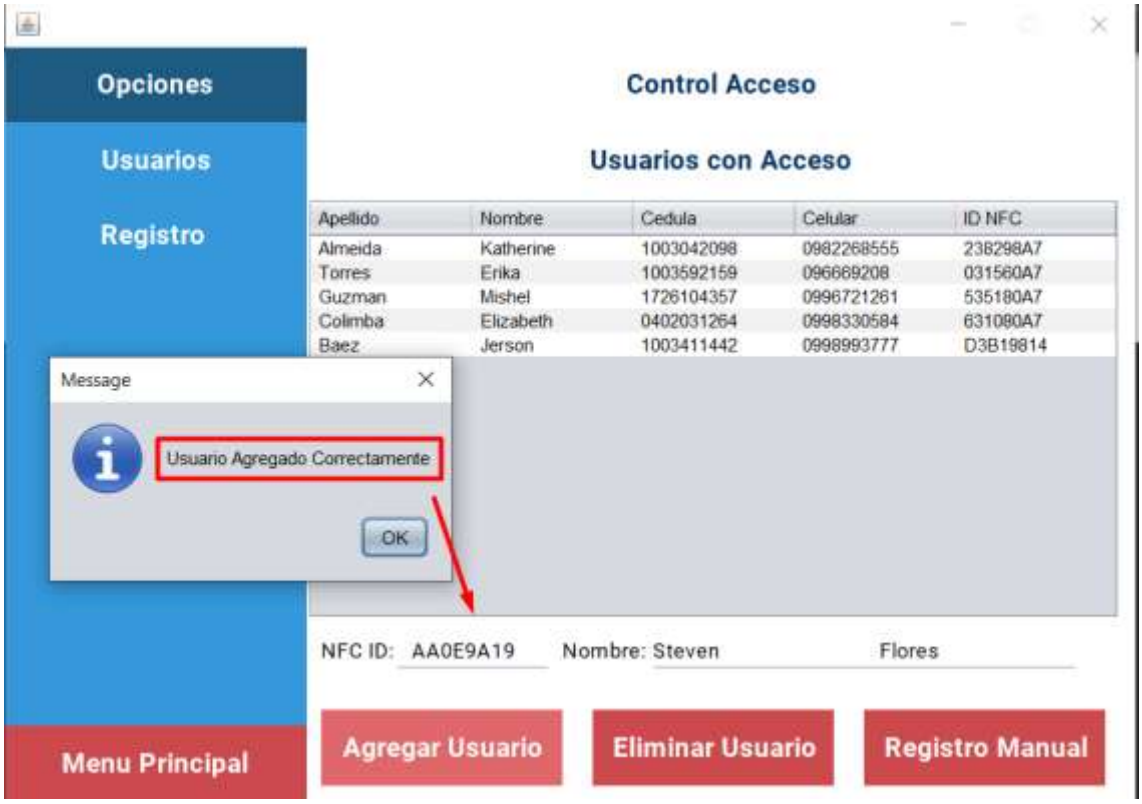
4.4 Prueba 3 Funcionamiento Aplicación Control de Acceso

En esta sección se evalúa el funcionamiento de la aplicación de control de acceso, la cual permite brindar o eliminar el acceso a cualquier usuario, para esto, se realiza una prueba de registro y posteriormente una prueba de quitar el acceso para posteriormente analizar y obtener conclusiones respecto al funcionamiento de la aplicación desarrollada.

4.4.1 Visualización funcionamiento

Inicialmente para comprobar el funcionamiento, se realiza el registro, de esta manera es posible visualizar en la **Figura** *72* como la aplicación registra de manera correcta un nuevo usuario.

Figura 72
Prueba 3-Agregar Usuario

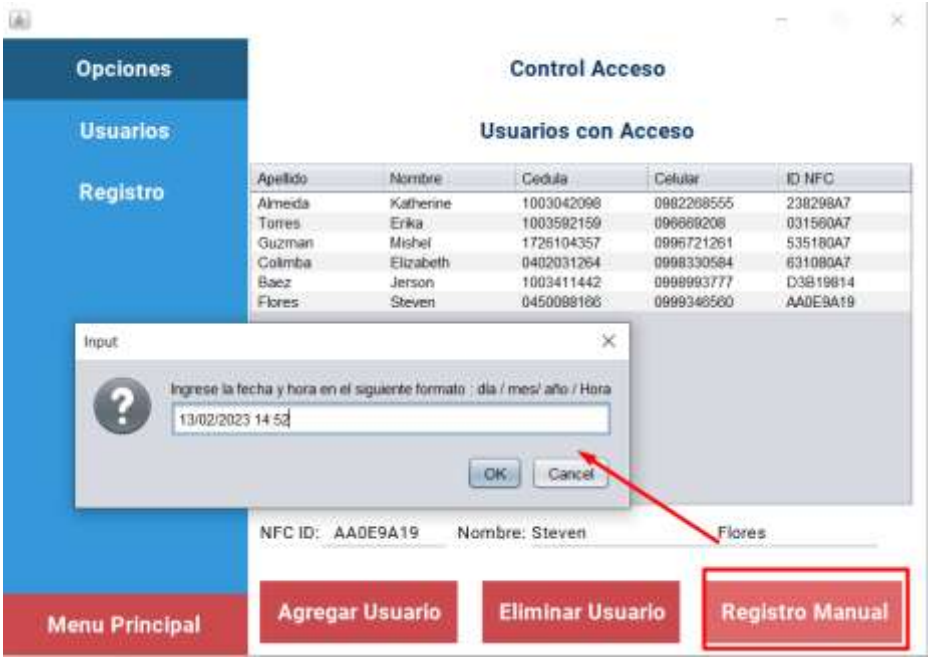


De igual manera, se realiza la prueba correspondiente a la eliminación de un usuario para remover el acceso que haya tenido previamente. La **Figura** *73* muestra de manera exitosa el funcionamiento adecuado de la aplicación

Figura 73
Prueba 3- Eliminar Usuario

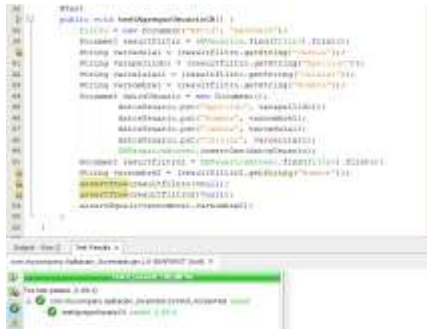
Posteriormente se realiza la prueba correspondiente de registro manual, esta función del sistema permite agregar un nuevo registro, para el caso de que el sistema principal de control de acceso no funcione y se requiera el ingreso de manera tradicional, el funcionamiento del mismo permite cumplir con un requisito que se establece en la sección 3.1 del presente documento. La **Figura** ***74*** muestra el funcionamiento de la función mencionada.

Figura 74
Prueba 3 Función Registro Manual



Una vez verificado el funcionamiento correcto mediante la interfaz gráfica, con la ayuda de la herramienta ***JUnit***, se realiza una matriz de pruebas, en la que se detallan los procesos que suceden de manera más técnica, esto se conoce como pruebas de caja blanca, ya que es posible visualizar cual es el código que se ejecuta para que dicha operación funcione. Dicha matriz se observa en la **Figura 75**.

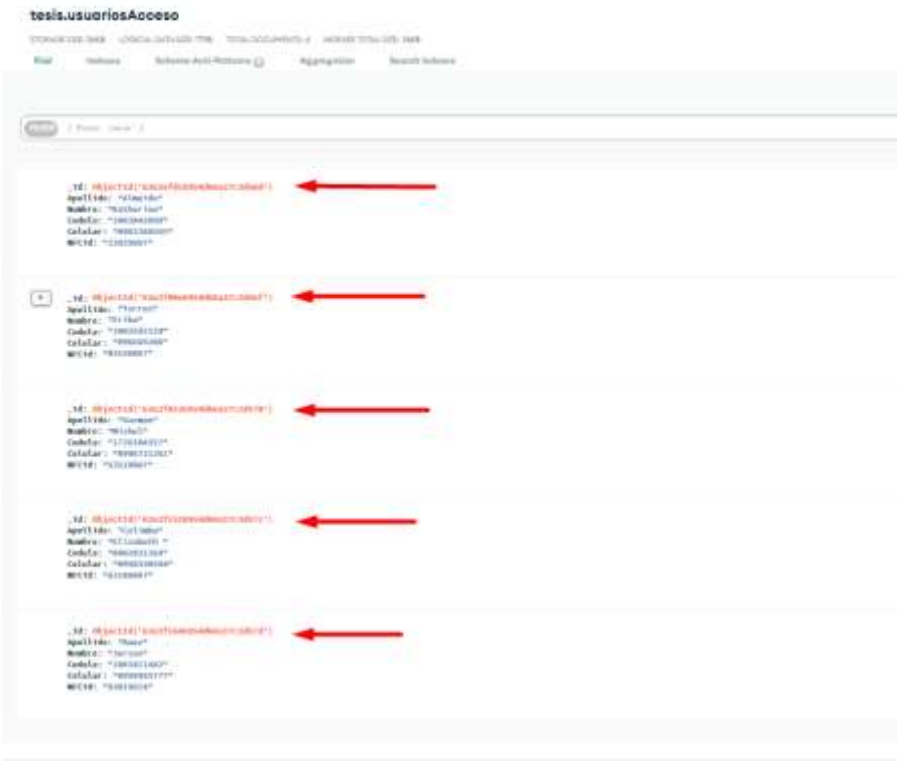
Figura 75
Matriz Pruebas Control Acceso

Matriz Pruebas Aplicación Control Acceso						
Id	**Caso de Prueba**	**Descripción**	**Resultado Esperado**	**Cumple**	**No Cumple**	**Metodo Verificación**
01	Conexión con Lector NFC	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión entre la aplicación y el lector desarrollado sea correcta utilizando un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que el puerto abierto es correcto y se encuentra preparado para escuchar los datos que serán recibidos.			
02	Conexión Base de Datos	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión de la aplicación y la base de datos sea correcta utilizando un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que la conexión con los servidores en los que se aloja la base de datos se realiza correctamente			
03	Agregar Usuario al Control Acceso	Mediante la herramienta Junit se verifica el código correspondiente para el registro de usuario, para ello se ejecuta el código en la herramienta y esta deberá verificar que se agregue un nuevo usuario	El resultado esperado será que un nuevo usuario sea agregado en la base de datos, para ello inicialmente se verifica que el usuario exista en la base de datos general y posteriormente se agrega el usuario a la base de datos de control de acceso y se verifica que se agregue correctamente, esto se realiza utilizando los metodos ***assertEqualsy assertTrue*** que proporciona JUnit			

04	Eliminar Usuario	Mediante la herramienta Junit se verifica el código correspondiente para la eliminación de usuarios de la base de datos de control de acceso, además se comprueba posteriormente que no existan dichos datos.	El resultado esperado será que, la base de datos general siga almacenando el respectivo usuario, sin embargo, la colección dedicada para control de acceso no tendrá dicho usuario, para ello se utiliza el metodo ***assertEquals*** y ***assertTrue*** que proporciona JUnit		
05	Registro Manual	Mediante la herramienta Junit se verifica el código correspondiente para el registro manual de usuarios, este deberá agregar una nueva entrada a la base de datos de registro.	El resultado esperado será que dicho registro sea almacenado en la base de datos correspondiente según se han ingresado los datos, para ello se utiliza **assertTrue** que proporciona Junit		

Con ello, se comprueba que la aplicación desarrollada funciona correctamente mediante el uso de pruebas de caja negra y pruebas de caja blanca, la **Figura *76*** muestra los datos correspondientes de las pruebas realizadas y como se almacenan en la base de datos de mongoDb.

Figura 76
Prueba 3 - Almacenamiento Datos



4.4.2 Conclusiones

Una vez culminada la fase de pruebas correspondiente a la aplicación de control de acceso, se realizan las respectivas conclusiones sobre el funcionamiento. Se determina que la aplicación cumple correctamente con las funciones propuestas en la sección 3.1. De la misma manera, dicha aplicación cumple con el objetivo de funcionar con la base de datos almacenada en la nube realizando consultas, actualización y publicación de datos tal como se mostró en la **Figura *76***.

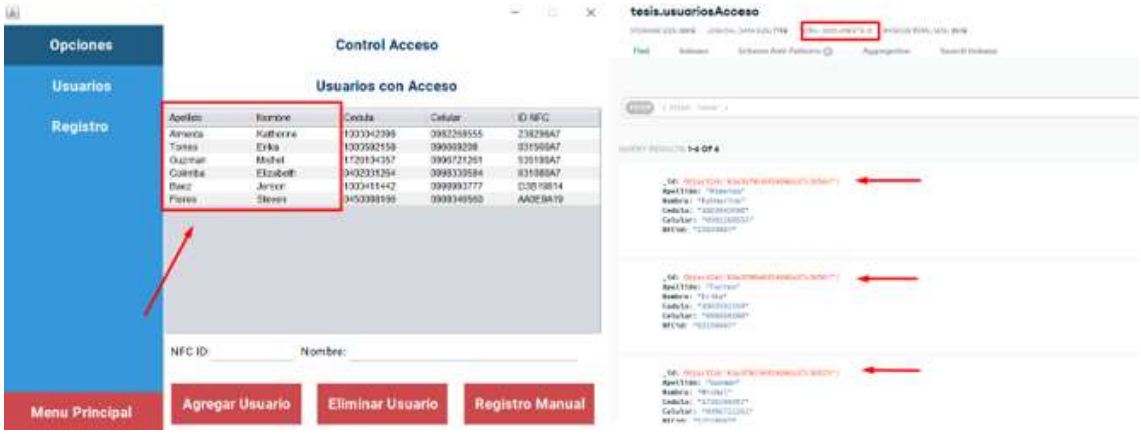
4.5 Prueba 4 Funcionamiento Control Acceso

En este apartado se demuestra el funcionamiento del prototipo de control de acceso desarrollado, para esta etapa inicialmente se verifica que aquellos usuarios que tengan acceso se encuentran previamente registrado y posteriormente se realiza la prueba correspondiente a la apertura y se muestra cómo se genera el registro en la respectiva base de datos alojada en la nube.

4.5.1 Visualización funcionamiento

En primera instancia se verifica que los usuarios que tienen acceso se encuentran debidamente registrados en la base de datos, esto se comprueba de manera sencilla en la propia base de datos y también en la aplicación. La **Figura** *77* muestra tanto en la aplicación como en la base de datos los diferentes usuarios que tienen acceso y han sido participes del desarrollo de la prueba.

Figura 77
Prueba 4 Usuarios con Acceso

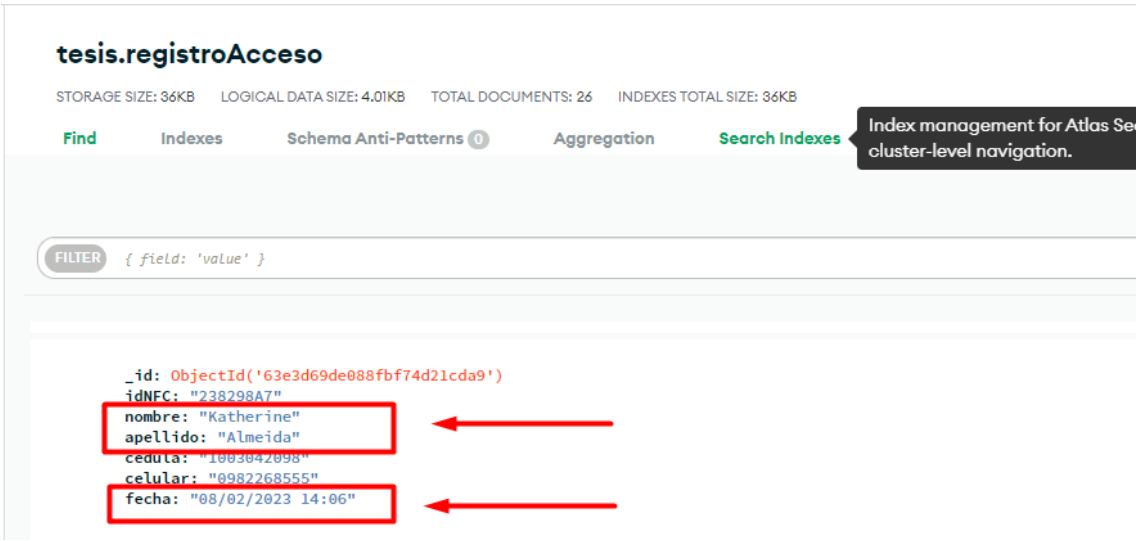


Con ello, se realiza la respectiva prueba utilizando el control de acceso y se obtiene el resultado de que el funcionamiento es el adecuado. La **Figura** *78* muestra como el dispositivo de control de acceso permite el ingreso mediante la utilización de tarjetas NFC que han sido registradas previamente tal como se observa en la **Figura** *77*.

Figura 78
Prueba 4 Control Acceso Abierto

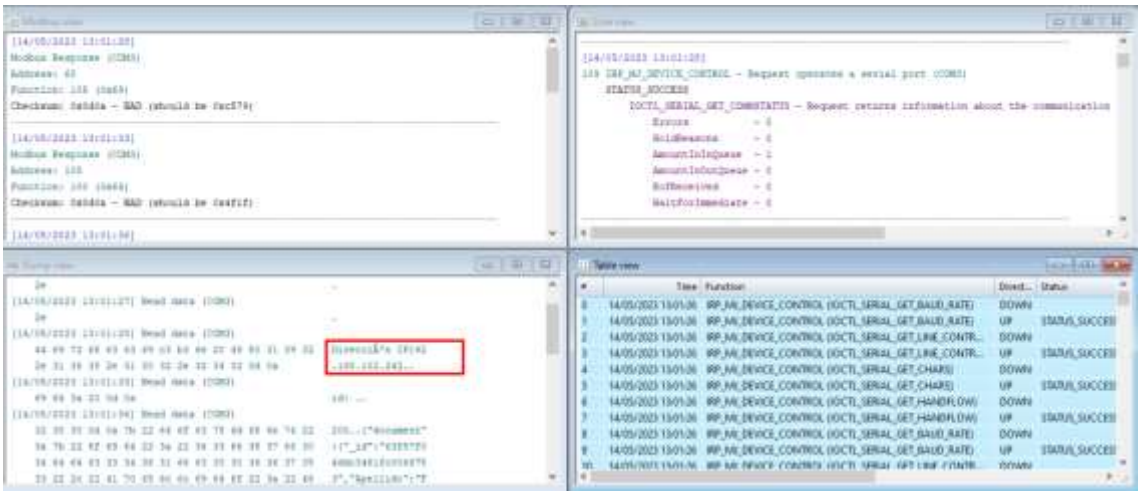
Posteriormente se verifica que el sistema se encuentre almacenando de manera correcta los diferentes datos, en este se obtiene información del usuario que ingresa, además de la fecha y hora en la que sucede dicho evento, con ello la **Figura *79*** muestra como los datos son almacenados en la nube.

Figura 79
Prueba 4 registro ingreso control de acceso



Con ello se verifica el funcionamiento del dispositivo cumpliendo así las denominadas pruebas de caja negra, esto principalmente ya que no se conoce el funcionamiento interno del dispositivo, sin embargo, también es posible realizar pruebas de caja negra mediante el uso de diversas herramientas, de este modo, la herramienta ***serial port monitor*** permite verificar como se realizan los procesos dentro del lector desarrollado. La **Figura 80** muestra cómo se realiza inicialmente la conexión a la red Wi-Fi para con ello tener acceso al servidor que se encuentra alojado en la nube.

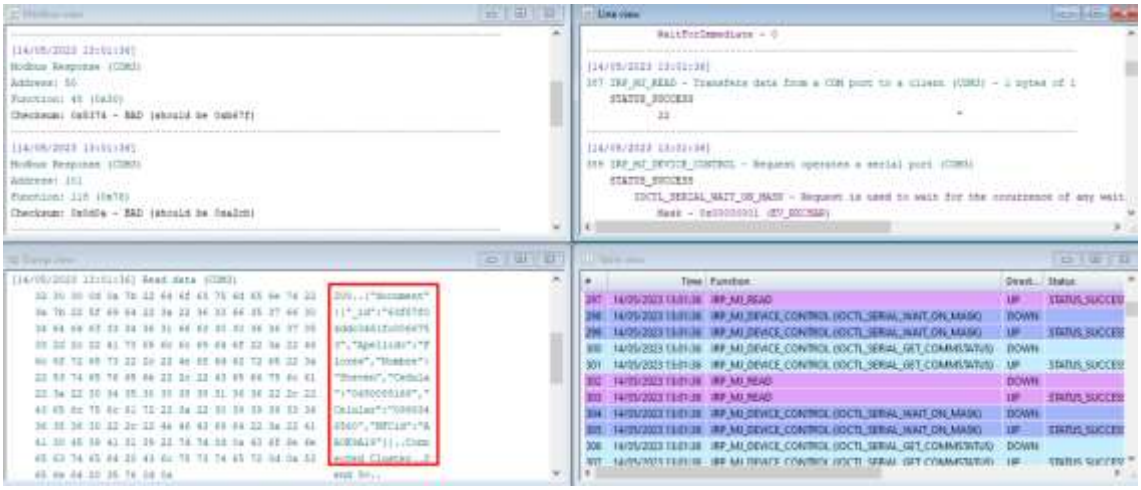
Figura 80
Conexión a internet del dispositivo



Una vez se conecta al servidor correspondiente con las credenciales obtenidas mediante NFC, el dispositivo realiza una consulta al servidor buscando coincidencias para los datos obtenidos previamente, en la **Figura 81** se pueden visualizar los diferentes datos que se han capturado, entre ellos, se observa el nombre, cedula, apellido del usuario quien accede mediante el control de acceso. Al obtener una coincidencia de datos, es posible visualizar que se envía una señal de 5v que permite que la cerradura actúe y esta se abra correctamente.

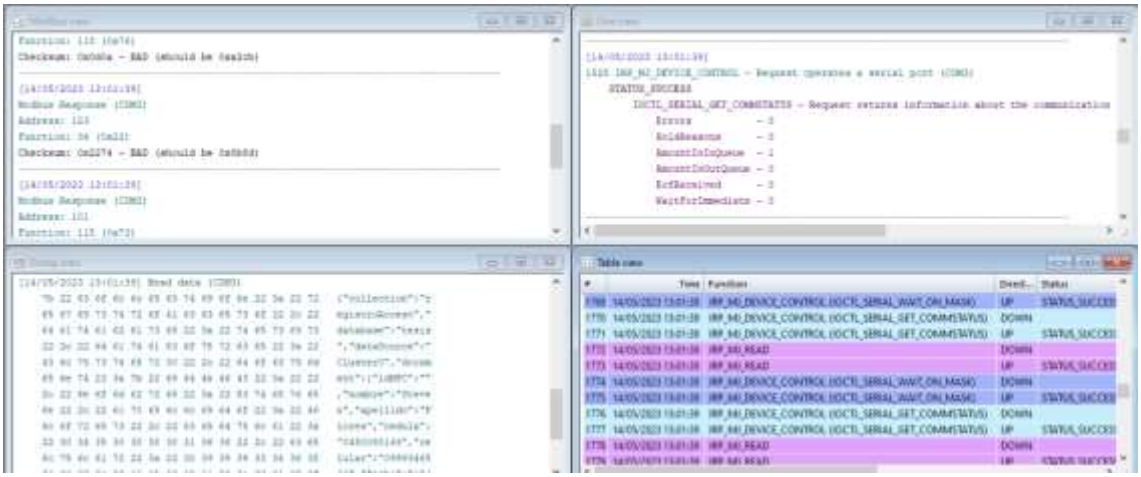
Figura 81

Datos obtenidos de la base de datos



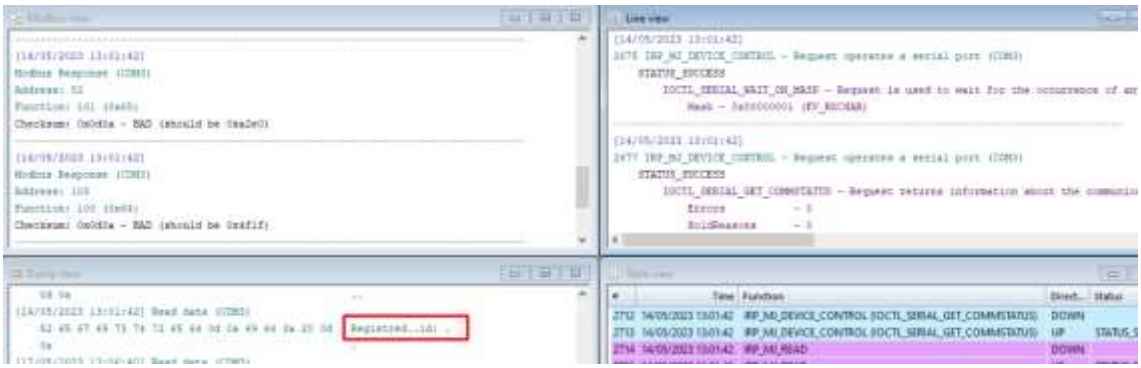
Una vez se abre la cerradura, la **Figura 82** muestra cómo se genera un nuevo registro que posteriormente será enviado y almacenado en la base de datos que se encuentra alojada en la nube, los datos que se envían corresponden al usuario que ha generado la petición de apertura.

Figura 82
Generación Registro a Base de Datos



Finalmente se visualiza como se recibe un mensaje de confirmación de que los datos se han enviado correctamente a la base de datos y además han sido registrados en la colección correspondiente, esto se muestra en la **Figura 83**.

Figura 83
Registro Completado



Con el fin de verificar el funcionamiento a lo largo de los días que se realizó las correspondientes pruebas, se generó un registro manual por parte de los diferentes usuarios que fueron parte de las pruebas, este registro será comparado con el registro automático para obtener los resultados correspondientes y de la misma manera un registro que muestra si la apertura del sistema fue correcta o no. La **Figura *84*** muestra el registro que se llevó de manera manual y se lo compara con el registro automático generado por el sistema desarrollado.

Figura 84

Prueba 4 comparación registro automático y registro Manual

Control Acceso

Registro de Usuarios que Ingresan

ID NFC	Nombre	Apellido	Cedula	Celular	Fecha y Hora
AADE9A19	Steven	Flores	0450088166	0999146560	01/02/2023 14:48
DB19814	Jerson	Baez	1003411442	0998993777	02/02/2023 12:03
DB19814	Jerson	Baez	1003411442	0998993777	02/02/2023 14:48
535180A7	Mishel	Guzman	1726104357	0996721261	03/02/2023 10:23
031560A7	Erika	Torres	1003592159	099699208	07/02/2023 09:57
631080A7	Elizabeth	Colimba	0402031264	0998330584	07/02/2023 15:36
AADE9A19	Steven	Flores	0450088166	0999346560	07/02/2023 17:37
AADE9A19	Steven	Flores	0450088166	0999346560	07/02/2023 17:39
DB19814	Jerson	Baez	1003411442	0998993777	08/02/2023 12:04
238298A7	Katherine	Almeida	1003042088	0982268555	08/02/2023 14:06
031560A7	Erika	Torres	1003592159	0999699208	08/02/2023 13:51
535180A7	Mishell	Guzman	1726104357	0996721261	09/02/2023 11:17
631080A7	Elizabeth	Colimba	0402031264	0998330584	09/02/2023 12:38
AADE9A19	Steven	Flores	0450088166	0999346560	09/02/2023 16:22
238298A7	Katherine	Almeida	1003042088	0982268555	10/02/2023 10:58
AADE9A19	Steven	Flores	0450088166	0999346560	13/02/2023 10:20
AADE9A19	Steven	Flores	0450088166	0999346560	13/02/2023 10:27
D3B198	Jerson	Baez	1003411442	0998993777	13/02/2023 13:23
631080A7	Elizabeth	Colimba	0402031264	0998330584	13/02/2023 13:55
238298A7	Katherine	Almeida	1003042088	0982268555	13/02/2023 14:10
238298A7	Katherine	Almeida	1003042088	0982268555	14/02/2023 08:53

Registro Manual de Ingreso				
Nombre	Apellido	Dia	Apertura	Resultado Obtenido
Steven	Flores	1 de febrero	Correcta	Registro Correcto
Jerson	Baez	2 de febrero	Correcta	Registro Correcto
Jerson	Baez	2 de febrero	Correcta	Registro Correcto
Mishell	Guzmán	3 de febrero	Correcta	Registro Correcto
Erika	Torres	7 de febrero	Correcta	Registro Correcto
Elizabeth	Colimba	7 de febrero	Correcta	Registro Correcto
Steven	Flores	7 de febrero	Correcta	Registro Correcto
Steven	Flores	7 de febrero	Correcta	Registro Correcto
Jerson	Baez	8 de febrero	Correcta	Registro Correcto
Katherine	Almeida	8 de febrero	Correcta	Registro Correcto
Erika	Torres	8 de febrero	Correcta	Registro Correcto
Mishell	Guzmán	9 de febrero	Correcta	Registro Correcto
Elizabeth	Colimba	9 de febrero	Correcta	Registro Correcto
Steven	Flores	9 de febrero	Correcta	Registro Correcto
Katherine	Almeida	10 de febrero	Correcta	Registro Correcto
Steven	Flores	13 de febrero	Correcta	Registro Correcto
Steven	Flores	13 de febrero	Correcta	Registro Correcto
Jerson	Baez	13 de febrero	Correcta	Registro Correcto
Elizabeth	Colimba	13 de febrero	Correcta	Registro Correcto
Katherine	Almeida	13 de febrero	Correcta	Registro Correcto

4.5.2 Conclusiones

Al culminar las diferentes pruebas que verifican el funcionamiento del prototipo desarrollado de control de acceso, es posible garantizar que el sistema funciona de acuerdo con lo establecido en un inicio como objetivo. De igual manera la **Figura *84*** muestra los diferentes registros y la comparación de los mismos en los que se obtiene que el sistema ha registrado durante 13 días todos los ingresos que se han realizado de manera exitosa, logrando así una efectividad del 100%, por lo que dicho sistema no tiene errores por corregir.

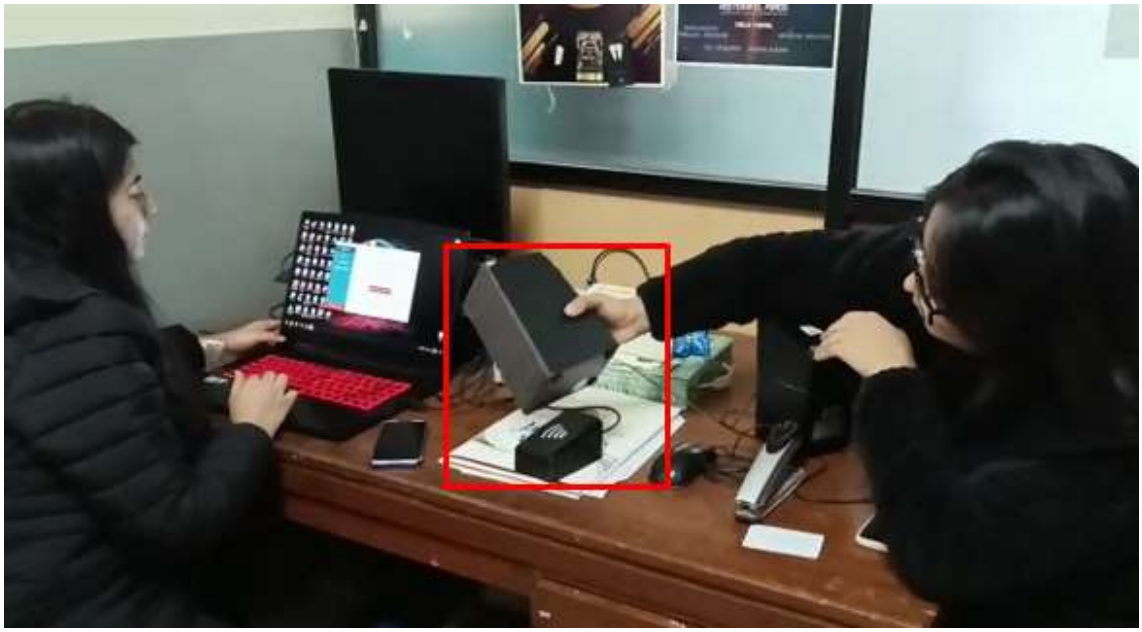
4.6 Prueba 5 Funcionamiento Aplicación Inventario

En esta sección se comprueba la funcionalidad de la aplicación desarrollada correspondiente a la gestión de inventario, para ello las pruebas necesarias a realizarse corresponden inicialmente a un registro de un nuevo artículo de inventario y posteriormente se visualiza como dicho inventario puede ser prestado mediante el uso de la misma aplicación. Finalmente, también se verifica el funcionamiento de la base de datos correspondiente que se encuentra alojada en la nube.

4.6.1 Visualización funcionamiento

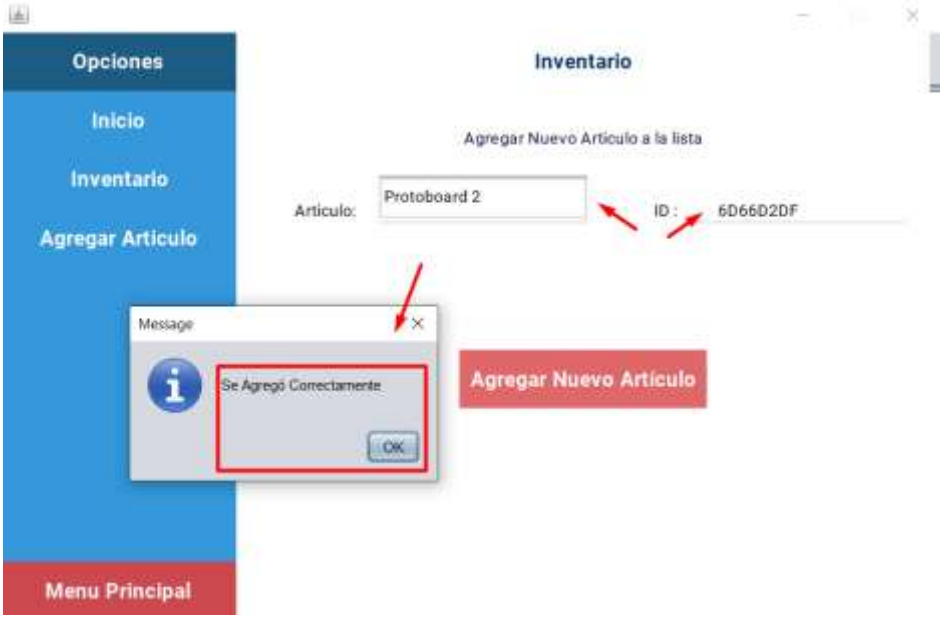
Inicialmente se debe etiquetar los diferentes artículos y posteriormente utilizarlos mediante el lector y la aplicación tal como se muestra en la **Figura *85***.

Figura 85
Prueba 5 registro inventario



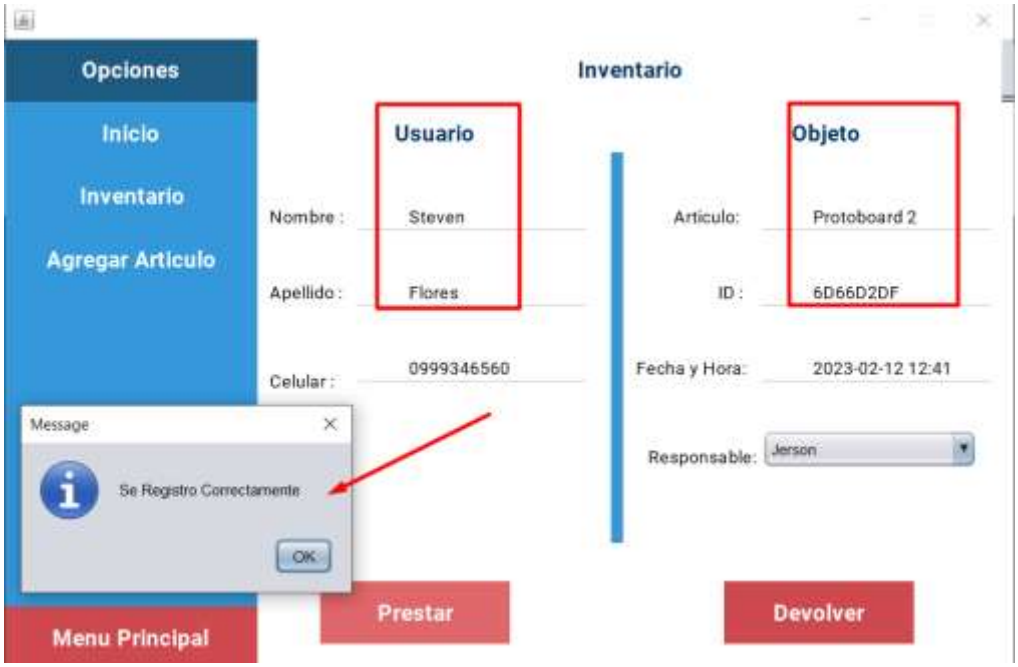
La **Figura *86*** muestra la prueba correspondiente al registro de un nuevo artículo, para ello inicialmente el artículo deberá contener la etiqueta NFC. De esta manera se obtiene el siguiente resultado.

Figura 86
Prueba 5 registro artículo inventario



Posteriormente para comprobar el funcionamiento de la aplicación se verifica que el sistema realice correctamente el préstamo de inventario a algún usuario de pruebas que se ha registrado. Para ello, la **Figura *87*** muestra el artículo que será prestado y también los datos de usuario de prueba que recibe el artículo.

Figura 87
Prueba 5 Proceso Préstamo Inventario



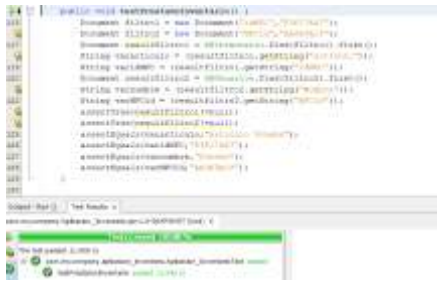
Una vez culminado el proceso de préstamo, la siguiente prueba consiste en verificar que la aplicación genere un registro adecuado del artículo con los respectivos datos obtenidos. La **Figura *88*** muestra el registro correspondiente con la prueba que se realiza anteriormente, de este modo se verifica que la aplicación cumple con el funcionamiento para la cual fue diseñada.

Figura 88
Prueba 5 Registro de Inventario

Una vez culminadas las pruebas realizadas mediante interfaz gráfica, se realizan las pruebas denominadas “Pruebas de Caja Blanca”, dichas pruebas pretenden demostrar de manera más técnica la funcionalidad del código, para ello se realiza una matriz de pruebas que se muestra en la **Figura 89**.

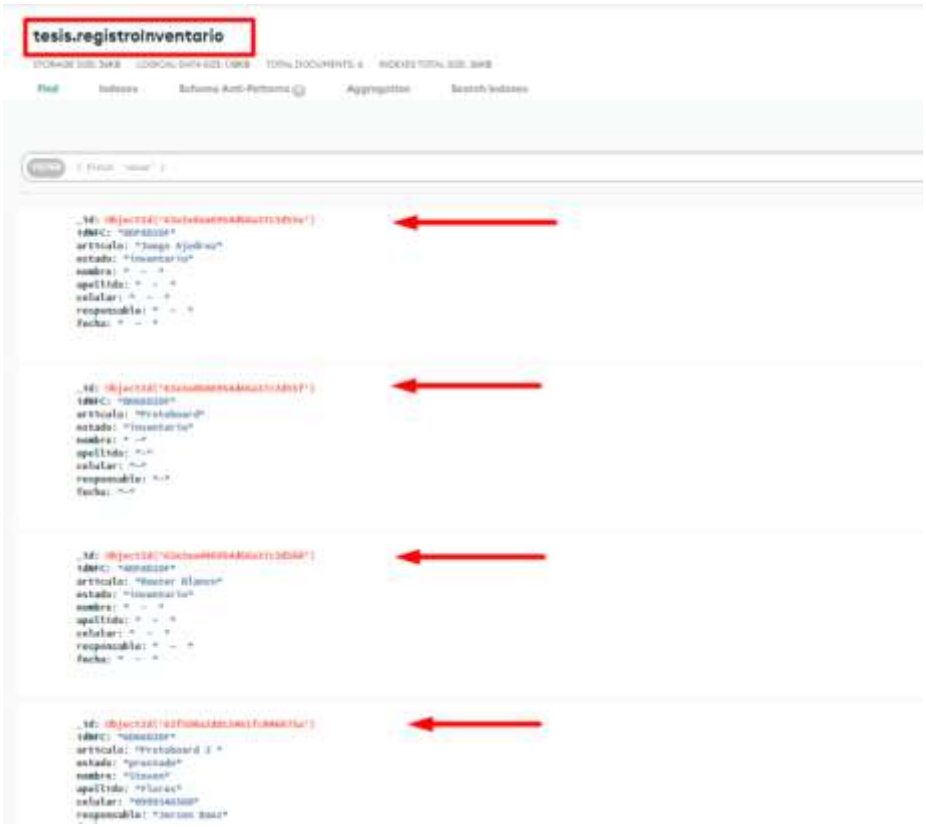
Figura 89
Matriz Pruebas Aplicación Inventario

Matriz Pruebas Aplicación Inventario						
Id	**Caso de Prueba**	**Descripción**	**Resultado Esperado**	**Cumple**	**No Cumple**	**Metodo Verificación**
01	Conexión con Lector NFC	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión de la aplicación y el lector desarrollado sea correcta utilizando un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que el puerto abierto es correcto y se encuentra preparado para escuchar los datos que serán recibidos.			
02	Conexión Base de Datos	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión de la aplicación y la base de datos sea correcta utilizando un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que la conexión con los servidores en los que se aloja la base de datos se realiza correctamente			
03	Registro Nuevo Articulo	Mediante la herramienta Junit se verifica el código correspondiente para el registro de un nuevo artículo, para ello se realiza el ingreso de un Id NFC y el nombre asociado al Id que corresponde al artículo y se verifica que se agregue a la base de datos correspondiente.	El resultado esperado será que exista una nueva entrada de un artículo que anteriormente no existía, de este modo también se verifica que el IdNFC se asocie correctamente con el nombre propuesto, esto se realiza utilizando los metodos ***assertEquals assertTrue*** que proporciona JUnit			

04	Prestamo Inventario	Mediante la herramienta Junit se verifica el código correspondiente para prestamo de inventario, este debe realizar una consulta a dos colecciones de la base de datos diferentes, una de usuarios y la otra correspondiente a articulos	El resultado esperado será que, se realicen las consultas a diferentes bases de datos, se obtengan los datos correspondientes esto se realiza utilizando los metodos ***assertEquals*** y ***assertTrue*** que proporciona Junit			
05	Registro Inventario	Mediante la herramienta Junit se verifica el código correspondiente para el registro de inventario, este deberá actualizar una entrada de inventario en la que se agrueguen los datos de usuario a quien se realizó el prestamo	El resultado esperado será que dicho registro sea almacenado en la base de datos muestre los valores de usuario y del articulo relacionados en una misma base de datos, para ello se utiliza **assertEquals** que proporciona Junit			

De esta manera se visualiza como la aplicación funciona correctamente tanto desde un punto de vista de usuario, como de forma técnica, con ello finalmente se muestra cómo se actualiza la base de datos desarrollada para dicho propósito. La **Figura *90*** muestra los diferentes datos almacenados en la base de datos alojada en la nube.

Figura 90
Prueba 5 Datos alojados en la nube.



4.6.2 Conclusiones

Al culminar los días de pruebas a los que se ha sometido la aplicación desarrollada, es posible determinar que dicha aplicación funciona de la manera esperada y para la cual fue diseñada. De la misma manera, la aplicación cumple con todos los parámetros de funcionamiento que se han sido establecidos

4.7 Funcionamiento Aplicación Micro Transacciones

A continuación, se describe las pruebas realizadas para comprobar la efectividad de la aplicación que se ha desarrollado, basándose en los requerimientos de la misma mencionados en la sección 3.1. Para ello la fase de pruebas de dicha aplicación se aborda durante 7 días, en este apartado se busca demostrar la funcionalidad de la aplicación verificando que la aplicación es capaz de acreditar o desacreditar valores exactos y además obtener un registro de dichas operaciones para finalmente visualizar como todos estos datos se están actualizando en la base de datos correspondiente.

4.7.1 Visualización funcionamiento

Inicialmente la prueba consiste en acreditar saldo a un usuario con la finalidad de visualizar el funcionamiento de la aplicación y también verificar que se realice el registro correspondiente de la operación realizada. Para ello el usuario deberá hacer uso de su tarjeta NFC. La **Figura *91*** muestra el funcionamiento de la aplicación haciendo uso de una acreditación de saldo.

Figura 91

Prueba 6 acreditación sistema micro transacciones

Una vez verificada que la operación cumple correctamente con los parámetros establecidos, se prueba la funcionalidad de realizar una desacreditación para verificar el funcionamiento de la misma. La **Figura *92*** muestra cómo se realiza la operación correspondiente.

Figura 92

Prueba 6 aplicación Micro transacciones desacreditación

Una vez visualizado que la aplicación funciona correctamente, se verifica que se estén generando los registros correspondientes a operaciones realizadas. Para ello, la **Figura *93*** muestra el registro de las operaciones que se llevan a cabo y se han registrado con éxito.

Figura 93
Prueba 6 registro operaciones realizadas

Una vez se definen las pruebas de interfaz también denominadas pruebas de caja negra, se procede a realizar pruebas de caja blanca, en ellas el principal objetivo radica en verificar el funcionamiento de código que hace posible el uso de interfaces, para ello mediante el uso de la herramienta de ***JUnit*** y el depurador proporcionado por NetBeans se realiza una matriz de pruebas que se observa en la **Figura *94***.

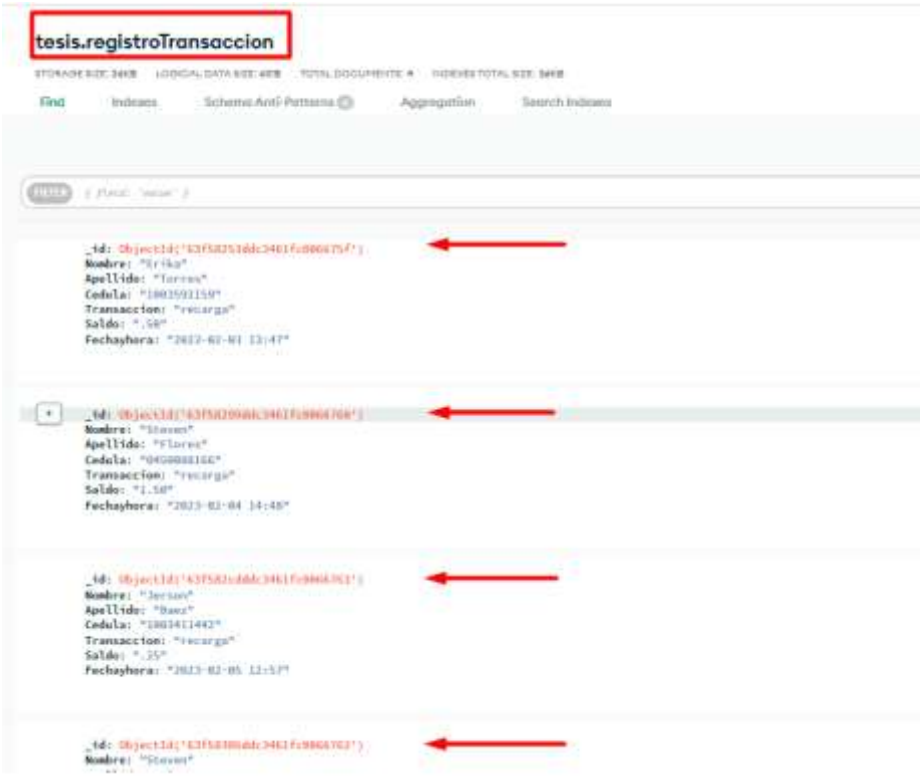
Figura 94
Matriz Pruebas Aplicación Micro Transacciones

Matriz Pruebas Aplicación Transacciones						
Id	**Caso de Prueba**	**Descripción**	**Resultado Esperado**	**Cumple**	**No Cumple**	**Metodo Verificación**
01	Conexión con Lector NFC	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión de la aplicación y el lector desarrollado sea correcta utilizando un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que el puerto abierto es correcto y se encuentra preparado para escuchar los datos que serán recibidos.			
02	Conexión Base de Datos	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión de la aplicación y la base de datos sea correcta utilizando un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que la conexión con los servidores en los que se aloja la base de datos se realiza correctamente			
03	Agregar Saldo de MicroTransacción	Mediante la herramienta Junit se verifica el código correspondiente que permite agregar saldo a los usuarios mediante microtransacciones.	El resultado esperado será obtener los datos iniciales del usuario, posteriormente realizar la operación de agregar saldo y se obtiene el resultado de dicha operación, con ello se verifica que los datos hayan sido actualizados, para ello se utilizan los métodos ***assertTrue y assertEquals*** que proporciona JUnit			

04	Eliminar Saldo Micro Transacción	Mediante la herramienta Junit se verifica el código correspondiente que permite eliminar saldo a los usuarios mediante microtransacciones.	Al igual que el proceso anterior, se inicia obteniendo los datos, posteriormente se realiza la operación correspondiente y finalmente se comparan los datos obtenidos mediante los métodos ***assertTrue y assertEquals*** que proporciona Junit			
05	Verificar Registro Transacciones	Mediante la herramienta Junit se verifica el código correspondiente que verifica que los datos de las microtransacciones se hayan agregado a la nueva colección de la base de datos	El resultado esperado será visualizar las diferentes microtransacciones, para ello, se espera que la colección no se encuentre vacía y posteriorme que exista el registro de las micro transacciones, para ello se utiliza el metodo ***assertTrue*** que proporciona JUnit			

Una vez culminado las pruebas de caja negra y caja blanca, se busca verificar que la aplicación se encuentra actualizando los datos correspondientes en la base de datos alojada en la nube. La **Figura *95*** muestra los datos actualizados en la base de datos.

Figura 95
Prueba 6 almacenamiento datos en nube



4.7.2 Conclusiones

Finalmente se obtienen las respectivas conclusiones de la aplicación desarrollada luego de una fase de pruebas comprendida durante 7 días. Para ello, es importante mencionar que el objetivo de la prueba es cumplir con la necesidad de un requerimiento mencionado en la sección 3.1. De este modo la aplicación desarrollada cumple con los requerimientos mostrados en dicha sección, por lo que resulta especialmente útil si se desea realizar la implementación de la misma. Con esto es posible determinar que la aplicación cumple su cometido para el cual fue diseñada.

4.8 Prueba 7 Funcionamiento Aplicación Matriculas

El desarrollo de esta prueba tiene como finalidad verificar el cumplimiento de objetivos y requerimientos establecidos previamente. Para ello la aplicación desarrollada correspondiente buscar realizar un registro de los aportes voluntarios que se realizan cada semestre, de este modo, se realiza un simulacro durante los 7 días de duración de esta prueba y se obtienen diferentes resultados que deberán cumplir con los requerimientos de la sección 3.1.

4.8.1 Verificación funcionamiento

Con el fin de comprobar el funcionamiento de la aplicación orientada al registro de aportes voluntarios, inicialmente se muestra un registro el cuál será la referencia que permite la visualización de los cambios correspondientes. Para hacer uso de la aplicación es necesario utilizar la tarjeta NFC y el lector tal como se visualiza en la **Figura *96***.

Figura 96
Prueba 7 Uso Tarjeta NFC

La **Figura *97*** muestra a todos los usuarios del sistema que son participes de la prueba por recomendación de la presidenta de la Asociación CITEL-CIERCOM, en dicha figura, se visualiza los valores iniciales que tendrían los aportes de los estudiantes.

Figura 97
Prueba 7 Aplicación Registro Aportes

Una vez determinado que la prueba inicializa con valores nulos referente a las aportaciones que tenga cada estudiante, se procede a realizar un simulacro actualizando los valores correspondientes del valor de aportación. La **Figura *98*** muestra cómo se realiza la actualización de este valor simulando que se realiza una aportación.

Figura 98

Prueba 7 Aplicación actualización aportes

De este modo, se visualiza que el apartado de actualizar los diferentes valores se realiza de manera correcta. El siguiente requerimiento mencionado con el que se debe cumplir de acuerdo con la sección 3.1 en la que se establecen los requerimientos para dicha aplicación es la generación automática de un documento de Excel en la que se obtenga el registro de todos los valores de manera ordenada para su posterior uso según crea conveniente la asociación. La **Figura *99*** muestra cómo se realiza la prueba correspondiente.

Figura 99

Prueba 7 aplicación aportes - generación registro

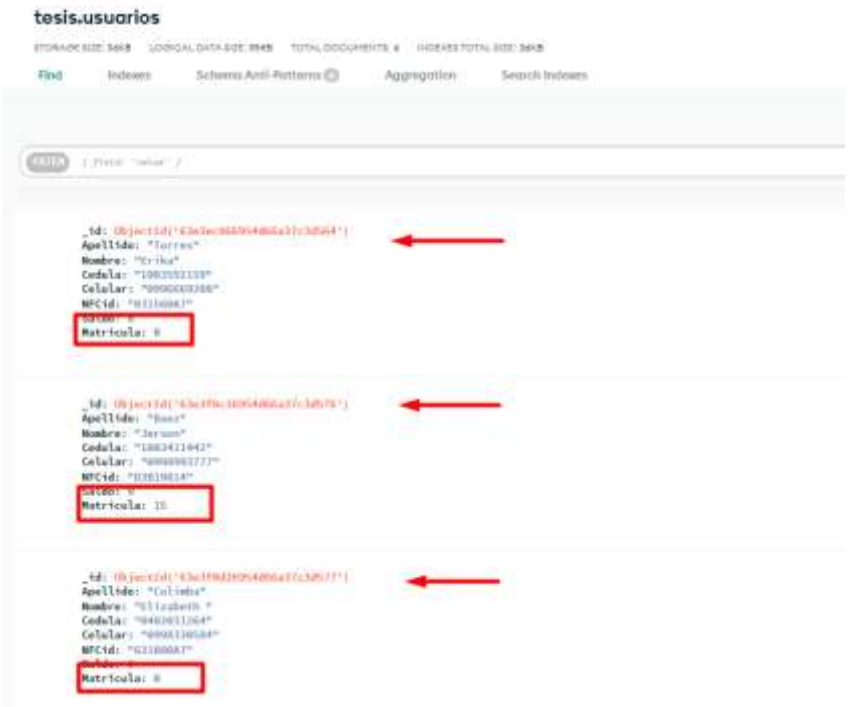
Una vez culminadas las pruebas de interfaz, se procede a realizar pruebas de caja blanca, estas consisten en verificar la funcionalidad del código desarrollado de manera más técnica, para ello se hace uso de la herramienta de Junit especializada para JAVA y también del depurador que brinda el propio NetBeans, la **Figura 100** muestra la matriz de pruebas realizadas.

Figura 100
Matriz de Pruebas Aplicación Matriculas

Matriz Pruebas Aplicación Matriculas						
Id	**Caso de Prueba**	**Descripción**	**Resultado Esperado**	**Cumple**	**No Cumple**	**Metodo Verificación**
01	Conexión con Lector NFC	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión de la aplicación y el lector desarrollado sea correcta utilizando un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que el puerto abierto es correcto y se encuentra preparado para escuchar los datos que serán recibidos.			
02	Conexión Base de Datos	Mediante la salida de depuración de la aplicación de NetBeans se debe comprobar que la conexión de la aplicación y la base de datos sea correcta utilizando un mensaje de confirmación.	El resultado esperado será un mensaje que confirme que la conexión con los servidores en los que se aloja la base de datos se realiza correctamente			
03	Verificar Usuarios con Saldo Inicial	Mediante la herramienta Junit se verifica el código correspondiente que permite realizar las consultas de aporte de matricula de los usuarios registrados.	El resultado esperado será obtener los datos principales del usuario, posteriormente realizar una consulta de saldo que deberá ser igual a cero, con ello se verifica que el sistema funciona, para ello se utilizan los metodos ***assertEquals*** y ***assertTrue*** que proporciona JUnit			
04	Actualizar Saldo de los Usuarios	Mediante la herramienta Junit se verifica el código que permite realizar actualización en los aportes de matriculas de los usuarios del sistema	El resultado esperado será obtener los datos principales del usuario, posteriormente realizar una consulta de saldo donde el saldo deberá actualizarse con el pago registrado y ser el mismo, para ello se utilizan los metodo ***assertEqualsy assertTrue*** que proporciona JUnit			

Finalmente, se debe comprobar la funcionalidad del sistema garantizando la utilización correcta de la base de datos que se encuentra alojada en la nube. Para ello, la **Figura *101*** muestra los diferentes valores que se encuentran almacenados en la nube y que han sido utilizados previamente por la aplicación.

Figura 101
Prueba 7 Base de datos aplicación aportes



4.8.2 Conclusiones

Al culminar las pruebas elaboradas durante los diferentes días establecidos acorde al cronograma presentado en la sección 4.1, se obtiene como resultado de las mismas que la funcionalidad de la aplicación desarrollada cumple con los requerimientos establecidos en la sección 3.1 que fueron dados por la Asociación CITEL-CIERCOM, además también se tiene que la fiabilidad del sistema ha sido del 100% durante la etapa de pruebas, por lo que no se deben realizar cambios en la estructura de la aplicación.

4.9 Resultados de las pruebas

Al culminar las pruebas realizadas en la asociación de estudiantes CITEL-CIERCOM, se verifica el funcionamiento del sistema planteado, de este modo se tiene que se cumplen con los objetivos propuestos inicialmente que ayudan a mejorar diferentes procesos que se realizan dentro de la asociación. También se comprueba que la tecnología NFC se puede aplicar para mejorar diferentes ámbitos. De igual manera las pruebas de caja blanca nos brindan un entorno en el cual se comprueba que el código desarrollado funciona correctamente mediante el uso de diferentes herramientas como ***JUnit.***

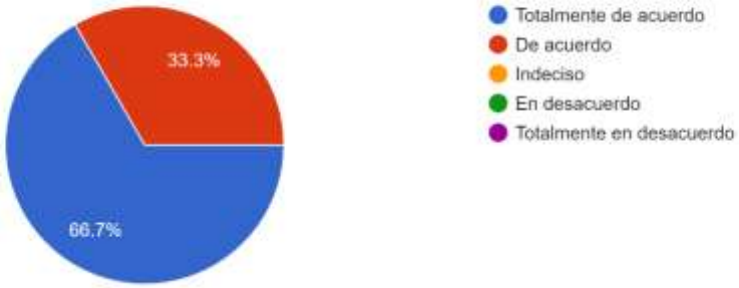
Con el fin de obtener un mejor análisis de resultados, se realiza una encuesta a los usuarios que fueron participes de las diferentes pruebas desarrolladas del 1 al 13 de febrero. La encuesta consta de 10 preguntas en las que se obtiene opiniones acerca del funcionamiento, uso y beneficios que brinda el sistema.

Encuesta Final - Trabajo de Titulación

Diseño de un prototipo que unifique, un sistema de pago basado en recargas, un sistema de control de inventario y un sistema de control de acceso, mediante la tecnología *Near Field Connection* (NFC) para la asociación de estudiantes CITEL-CIERCOM.

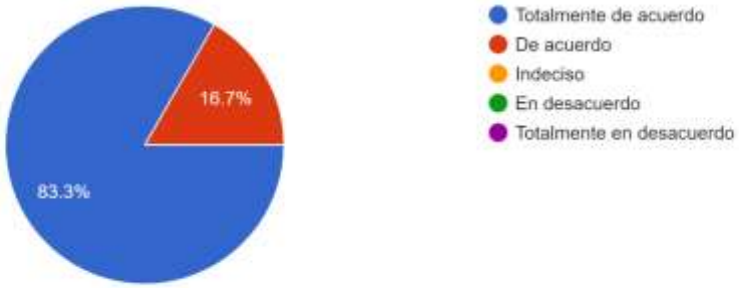
¿Cree qué el sistema agiliza los diferentes procesos que se realizan dentro de la Asociación CITEL-CIERCOM?

6 respuestas



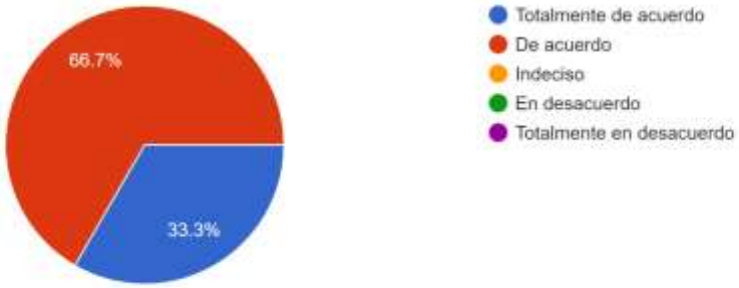
¿El sistema es fácil de usar y entender?

6 respuestas



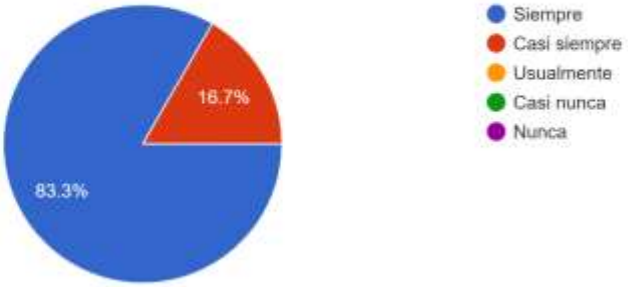
¿En la fase de pruebas realizadas, el sistema obtuvo los resultados esperados?

6 respuestas



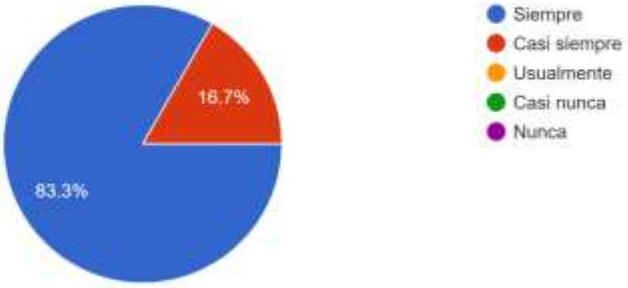
¿El sistema de control de acceso funcionó correctamente en los días de pruebas. ?

6 respuestas



¿El prototipo lector funcionó correctamente durante las pruebas realizadas. ?

6 respuestas

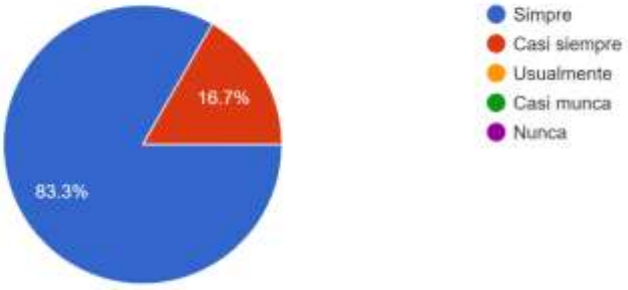


¿La aplicación de inventario funcionó correctamente durante las pruebas realizadas?

6 respuestas

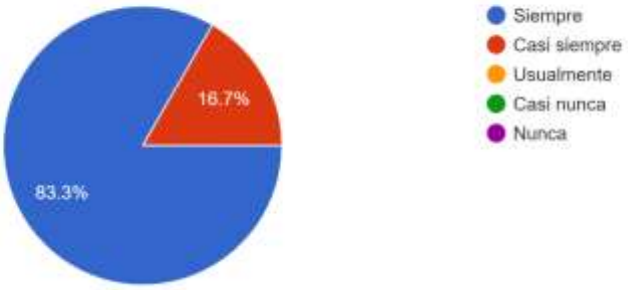
¿La aplicación de usuarios funcionó correctamente durante los días de prueba?

6 respuestas



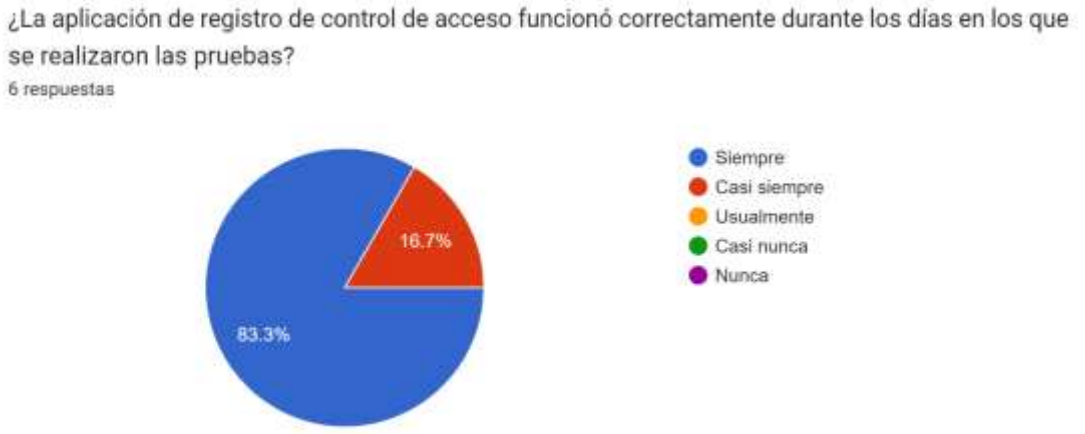
¿La aplicación de micro transacciones funcionó correctamente durante la etapa de pruebas?

6 respuestas



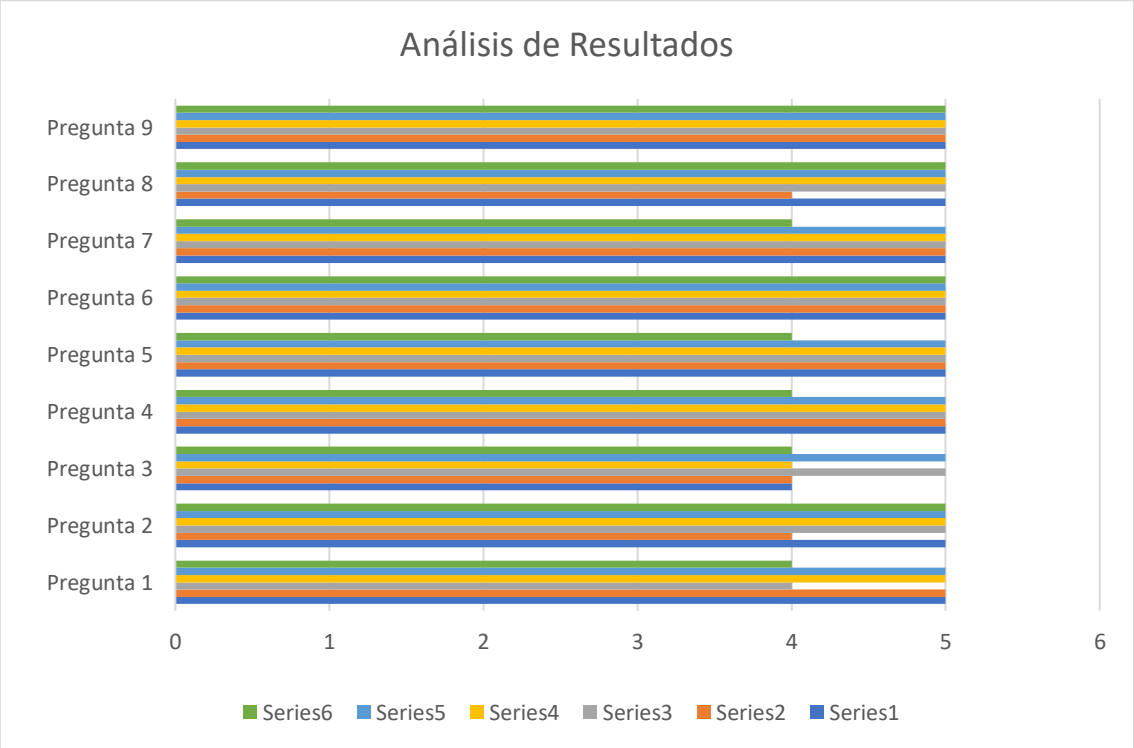
¿La aplicación de registro de aportes voluntarios funcionó correctamente durante las pruebas realizadas?

6 respuestas



Con ello, se realiza un análisis general en el que se obtienen las respuestas y se le asigna una calificación entre 1 y 5, considerando que 5 sería la mejor opción y 1 la peor. La **Figura *102*** muestra una recopilación de todas las preguntas realizadas.

Figura 102
Análisis de Resultados



De esta manera es posible obtener una nota promedio que da una mejor visión de manera cualitativa sobre el funcionamiento global del sistema desarrollado. Con ello el promedio

obtenido es de 4.8 sobre 5, lo que nos da un indicio de que el sistema cumple con los objetivos que se habían propuesto de manera inicial.

Conclusiones

- El sistema desarrollado desempeña de manera satisfactoria las diferentes funciones planteadas al iniciar el proyecto, utilizando la tecnología NFC (*Near Field Connection)* facilitando las funciones desempeñadas en la Asociación CITEL-CIERCOM como son el cobro de matrículas, control de acceso y control de inventario.
- El sistema demuestra la versatilidad que se puede obtener al utilizar la tecnología NFC (*Near Field Connection)*, al utilizarse en el desarrollo de diferentes aplicaciones.
- El sistema permite ser utilizado en cualquier lugar sin necesidad de un servidor dedicado, dado que hace uso de una base de datos alojada en la nube, esto representa un menor costo y una mayor facilidad de implementación.
- El desarrollo de una aplicación enfocada a plataformas Windows permite el fácil acceso a los datos a través de una interfaz de usuario amigable y funcional lo que ayuda a la visualización de los datos.
- Las pruebas de caja negra y caja blanca permiten obtener resultados precisos sobre el funcionamiento del sistema tanto en el apartado de hardware como de software.
- El sistema permite agilizar los diferentes procesos desarrollados en la Asociación CITEL-CIERCOM, por lo que la implementación del mismo es factible.

Recomendaciones

- El sistema desarrollado sirve como base para el desarrollo de futuros proyectos, de este modo se recomienda promover la implementación de sistemas similares en un mayor número de organizaciones dado que existe ahorro de tiempo, recursos y mejora la eficiencia de la organización.
- El uso de electrónica es fundamental a la hora de realizar proyectos relacionados con automatización y el internet de las cosas, de esta manera, se recomienda analizar los requerimientos electrónicos del sistema para obtener resultados adecuados.
- Es recomendable siempre establecer correctamente una etapa de planificación, diseño e implementación para asegurarse de que la funcionalidad del sistema sea la adecuada.
- Siempre se deben mantener medidas de seguridad al momento de realizar manipulación de dispositivos electrónicos como pulseras antiestáticas, ya que con ello se asegura la integridad del usuario y también se garantiza que se produzcan daños en los equipos electrónicos.
- Se recomienda siempre investigar y explorar nuevas tecnologías y tendencias en el campo de la automatización de tareas y comunicaciones inalámbricas para identificar oportunidades de mejora y mantener el sistema actualizado y competitivo.

Referencias

Amazon Web Services. (2022). *Amazon Web Services*. Obtenido de What is a relational database: https://aws.amazon.com/relational-database/#:~:text=Database%20Blog,be%20represented%20in%20the%20database.

Arduino. (2016). *Aprendiendo Arduino*. Obtenido de https://aprendiendoarduino.wordpress.com/2016/page/6/

Arduino. (2019). *FORUM ARDUINO*. Obtenido de ESP32 - Arduino or micropython? + Arduino PlatformIO problems: https://forum.arduino.cc/t/esp32-arduino-or-micropython-arduino-platformio-problems/614674

Bhardwaj, R. (s.f.). *IPWITHEASE*. Obtenido de C++ vs JAVA: https://ipwithease.com/c-vs-java/

Broseta, G. R. (2012). *Dispositivos móviles y NFC aplicados al canjeo de tickets [Tesis de Master, Universitat Politècnica de València].* Repositorio Institucional.

Corvo, H. S. (12 de febrero de 2020). *Lifeder*. Obtenido de Modelo espiral: historia, características, etapas, ejemplo.: https://www.lifeder.com/modelo-espiral/.

Domínguez, M. (1 de 3 de 2017). *SoloArduino.com*. Obtenido de Arduino y solo arduino: https://soloarduino.blogspot.com/2017/03/que-es-un-esp32.html

ESPRESSIF Systems. (2022). *ESPRESSIF*. Obtenido de https://www.espressif.com/

Goldsmith, A. (2020). *Wireless Communications.* Standford: Cambridge University Press.

IBM. (2022). *IBM*. Obtenido de What is a cloud database?: https://www.ibm.com/cloud/learn/what-is-cloud-database

Israel, V. (2017). *Diseño e implementación de un sistema electrónico para el registro de acceso y envío de información mediante tecnología NFC al personal administrativo y de soporte técnico de la empresa Wisp Airmaxtelecom soluciones tecnológicas [Tesis de pregrado].* Repositorio Institucional.

Java. (2022). *Java.com*. Obtenido de https://www.java.com/en/download/help/whatis_java.html

Kuzmenko, E. (12 de 3 de 2022). *KITRUM*. Obtenido de Difference Between MongoDB, MySQL, and PostgreSQL: https://kitrum.com/blog/difference-between-mongodb-mysql-and-postgresql/

Microsoft Azure. (2022). *Microsoft Azure*. Obtenido de Datos no relacionales y NOSQL: https://docs.microsoft.com/es-es/azure/architecture/data-guide/big-data/non-relational-data

Mngomezulu, Z. (s.f.). *Researchgate*. Obtenido de https://www.researchgate.net/publication/315697734_A_review_of_Bluetooth_and_NFC_for_financial_applications

MongoDB, Inc. (2022). *MongoDB*. Obtenido de ¿what is mongoDB?: https://www.mongodb.com/es/what-is-mongodb

MongoDB,Inc. (2022). *MongoDB*. Obtenido de JSON and BSON: https://www.mongodb.com/json-and-bson

Moreno, J. S. (2012). *PRACTICAL APPLICATIONS OF NFC [Revista de investigación].* 3ciencias.

Naylampmechatronics. (2021). *Naylamp Mechatronic.* Obtenido de https://naylampmechatronics.com/espressif-esp/382-modulo-esp32-

Naylampmechatronics. (2021). *Naylampmechatronics.com*. Obtenido de https://naylampmechatronics.com/espressif-esp/48-modulo-esp-01-esp8266-wifi-serial.html

NearFieldCommunication.org. (2017). *NearFieldCommunication.org*. Obtenido de History of Near Field Communication: http://nearfieldcommunication.org/history-nfc.html#:~:text=Near%20field%20communication%20(NFC)%20traces,to%20form%20the%20NFC%20Forum.

NXPSemiconductors. (2019). *naylampmechatronics.com*. Obtenido de NXP.com: https://naylampmechatronics.com/rfid-nfc/182-modulo-lector-rfid-nfc-1356mhz-pn532.html

Oracle. (2022). *NetBeans IDE | Oracle España*. Obtenido de https://www.oracle.com/es/tools/technologies/netbeans-ide.html

Oracle Mexico. (2022). *Oracle Mexico*. Obtenido de Base de datos definida: https://www.oracle.com/mx/database/what-is-database/

Redator, R. (2019). *Rockcontent*. Obtenido de https://rockcontent.com/es/blog/que-es-un-lenguaje-de-programacion/

RedHat. (2019). *RedHat*. Obtenido de https://www.redhat.com/es/topics/middleware/what-is-ide

Valdés Pérez, F., & Ramon, :. P. (2007). *Microcontroladores: Fundamentos y aplicaciones con pic.* Carles Parcerisas Civit.

W3School. (2022). *W3School.com*. Obtenido de https://www.w3schools.com/cpp/cpp_intro.asp

ANEXOS

ANEXO A. ENTREVISTA

	UNIVERSIDAD TÉCNICA DEL NORTE **CARRERA DE INGENIERÍA EN TELECOMUNICACIONES**
PROYECTO: Diseño de un prototipo que unifique, un sistema de pago basado en recargas, un sistema de control de inventario y un sistema de control de acceso, mediante la tecnología Near Field Connection (NFC) para la asociación de estudiantes CITEL-CIERCOM.	
Fecha de Entrevista:	**16-11-2022**
Organización:	**ASOCIACIÓN ESTUDIANTES CARRERA INGENIERÍA EN TELECOMUNICACIONES**
Entrevistado/a:	**KATHERINE ALMEIDA**
Cargo:	**PRESIDENTA DE LA ASOCIACIÓN**

INTRODUCCIÓN

La presente entrevista tiene como fin el levantamiento de información de algunos procesos que se dan dentro de la asociación de estudiantes de la Carrera de Ingeniería en Telecomunicaciones de la Universidad Técnica del Norte, con el fin de establecer necesidades que serán resueltas con el desarrollo del presente trabajo de titulación.

PREGUNTAS:

7. ¿Qué problemas se presentan en cuanto al inventario que posee la asociación?

Actualmente en la asociación, el inventario se lo ha llevado sin ningún tipo de registro, de tal manera que existen varios artículos que se han extraviado.

La asociación busca realizar en un futuro préstamo de artículos que puedan utilizar los estudiantes de la carrera de ingeniería en telecomunicaciones, estos pueden ser arduinos, cables UTP, calculadoras, cargadores, cables micro USB, cables tipo C, protoboards. De esta manera aquellos estudiantes que requieran podrán hacer uso de dichos artículos.

8. ¿Qué problemas se presenta de acuerdo con el acceso a las instalaciones de la asociación CITEL-CIERCOM?

Dentro de la asociación, es posible visualizar que no se tiene un control de quien ingresa a la asociación, de este modo, la seguridad de los artículos que se encuentran dentro se puede ver vulnerada.

Actualmente tienen acceso los miembros de la asociación y los dirigentes de cada nivel. Por lo que esto se debe considerar.

9. ¿Cuál es la situación actual respecto al valor de aporte voluntario que realizan los estudiantes?

Cada vez que se realizan las matrículas, se realiza un pago de aportación para la organización de eventos durante el semestre, sin embargo, esto se lo hace de manera tradicional y son los dirigentes de la asociación quienes deben estar pendientes de que se realice el pago, de este modo llevar el registro de todos los que han pagado puede resultar tedioso de realizar.

10. ¿Cuál es la situación actual respecto a la tarjeta de copias que se asignado a cada uno de los estudiantes de la carrera?

Existió un acuerdo hablado entre la copiadora de la FICA y la asociación, en la que cada estudiante tenía asignado un número de copias gratuitas, para ello se entregaba una tarjeta de cartulina con el número de copias asignadas y se iba marcando cada que se hacía uso de las mismas, sin embargo, no todos los estudiantes hacían uso de la cartulina por diferentes motivos.

La asociación habría pagado por cada uno de los estudiantes una cantidad de dinero previamente, por lo que, al no llevar un registro exacto, se habría pagado más de lo que se había consumido por parte de los estudiantes.

11. ¿Estaría interesada en utilizar un sistema basado en NFC, que permita gestionar algunos de los problemas mencionados de manera más eficiente?
Si

12. ¿Cuáles son algunos de los requerimientos que se consideran necesarios para el desarrollo de un sistema NFC?

El sistema debería solventar los problemas que se han mencionado antes, de esta manera el sistema deberá contar con 4 apartados fundamentales.

- **Sistema de control de acceso**
  Requerimientos:

  El sistema de control de acceso mediante NFC, será capaz de generar un registro de cada persona que ingrese a la asociación.

  El sistema deberá permitir que se pueda agregar usuarios para que ingresen en la asociación de manera sencilla

  El sistema solo se abrirá con aquellos usuarios miembros seleccionados.

- **Sistema de inventario**
  Requerimientos:

  El sistema deberá ser capaz de llevar un registro de todos los artículos que se vayan a prestar a los estudiantes.

  El sistema permitirá tener un registro con hora y fecha de cuando ha sido prestado el articulo

  Aquellos que pueden prestar artículos dentro de la asociación, son los dirigentes de cada nivel y la directiva de la asociación.

  El sistema deberá registrar el usuario quién ha prestado el articulo y el usuario a quien se le presta.

- **Sistema de micro transacciones.**
  Requerimientos:

  En el sistema de micro transacciones, se deberá tener un registro de todas las transacciones que se realicen.

  Los usuarios de este sistema serán todos los estudiantes de la carrera de ingeniería en telecomunicaciones

  El sistema NFC podrá reemplazar al sistema de tarjetas de cartulina que se utilizaba anteriormente.

- **Sistema de pago de aportación para eventos**
  Requerimientos:

  Los usuarios de este sistema serán todos los estudiantes de la carrera CITEL-CIERCOM

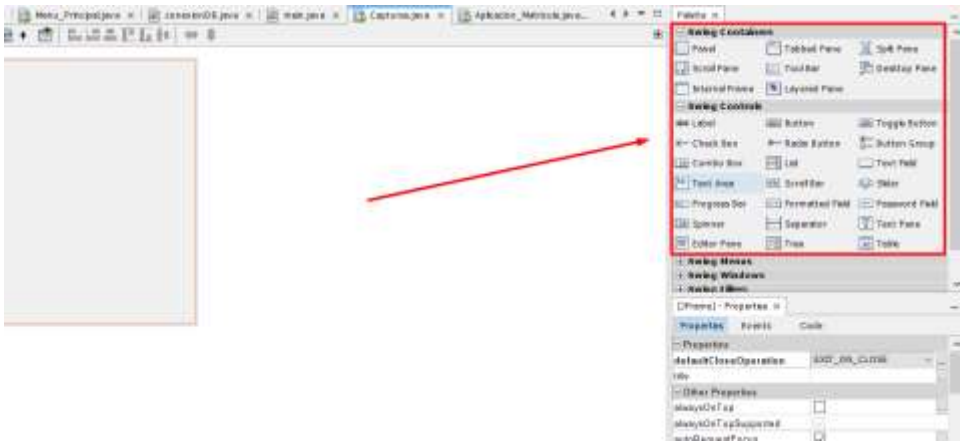
  El sistema permitirá gestionar un registro de todos aquellos estudiantes que haya realizado el pago y de aquellos que no.

ANEXO B - DESARROLLO DE APLICACIONES.

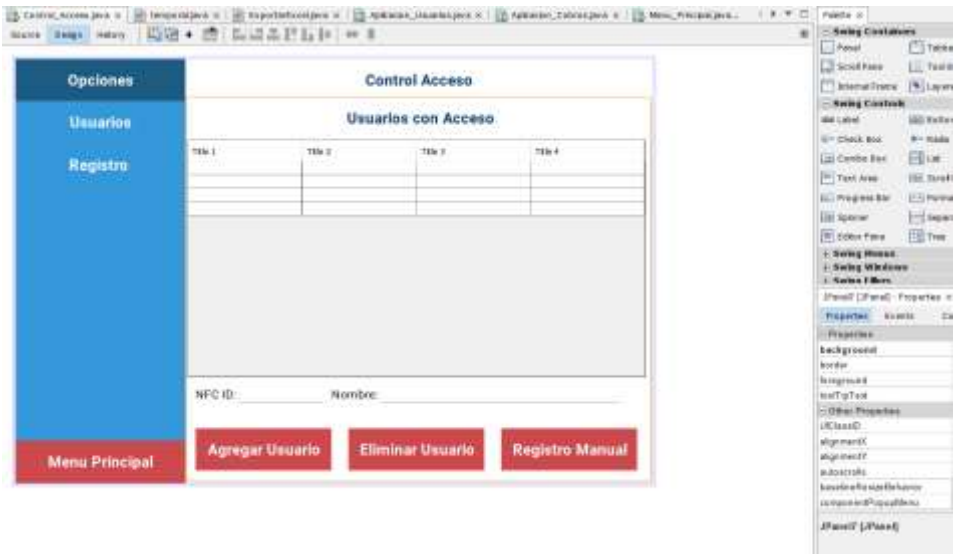
Desarrollo de Aplicación de Control Acceso

En esta sección se tiene el desarrollo de la aplicación de control de acceso de manera detalla, para ello se utilizan 3 interfaces cada una con sus respectivos elementos que serán mostrados a continuación.

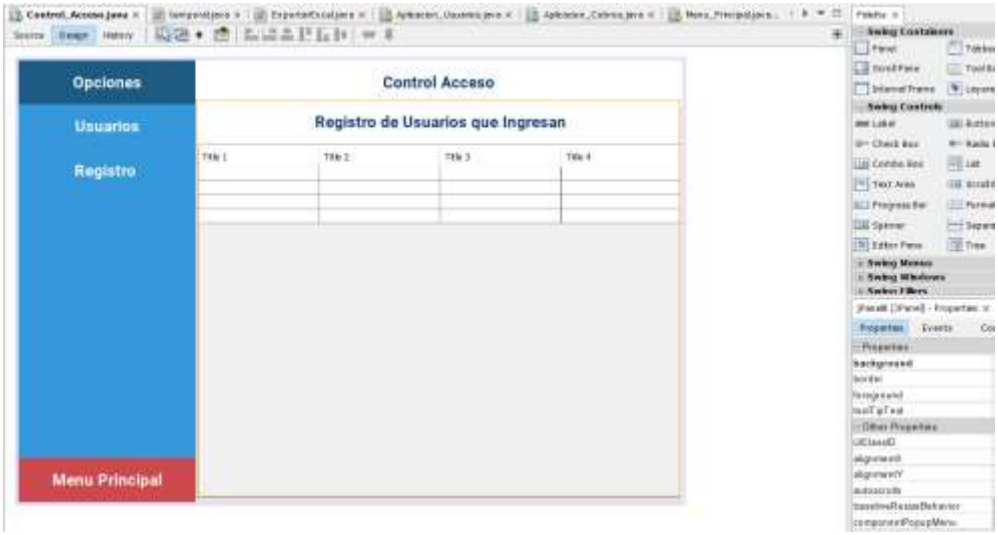
1. Inicialmente se desarrolla la aplicación de control de acceso, para ello se genera un nuevo JFRAME y se agregan los elementos desde el *palette* tal como se visualiza a continuación.



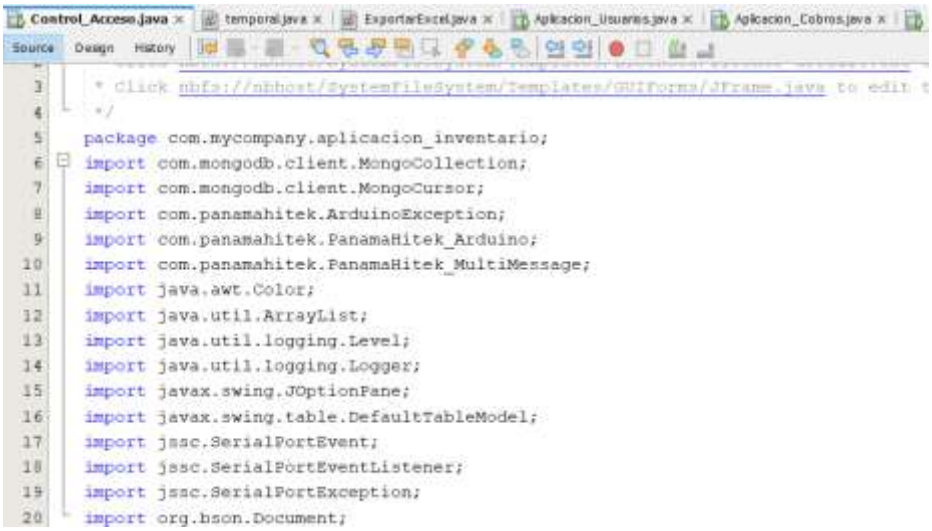
2. Al terminar de agregar los elementos, se les aplica opciones de color y de apariencia para obtener un aspecto mejor trabajado, dando como resultado la siguiente interfaz.



3. De la misma manera se genera la otra interfaz sobre la que se va a desarrollar la programación.



4. Una vez terminado la disposición de cada elemento se procede a visualizar la programación de los mismos. Inicialmente se tiene que importar las diferentes librerías necesarias para el funcionamiento correcto de la aplicación, aquí es posible visualizar librerías correspondientes a la conexión con la base de datos, la obtención de datos por comunicación serial y librerías relacionadas para el puerto serial y la generación de elementos dentro de la interfaz.



5. Posteriormente se procede a definir algunas de las variables que van a ser usadas, además se generan las conexiones tanto de la base de datos como con los puertos seriales para la comunicación serial.

```
public class Control_Acceso extends javax.swing.JFrame {
    Document filtro;
    Document filtro2;
    String puerto;
    String varID;
    String varapellido;
    String varnombre;
    String varcedula;
    String varcelular;
    MongoCollection<Document> DBControlAcceso = new conexionDB().obtenerDB().getCollection("registroAcceso");
    MongoCollection<Document> DBUsuarios = new conexionDB().obtenerDB().getCollection("usuarios");
    MongoCollection<Document> DBUsuariosAcceso = new conexionDB().obtenerDB().getCollection("usuariosAcceso");
    PanamaHitek_Arduino ino = new PanamaHitek_Arduino();
    PanamaHitek_MultiMessage multi = new PanamaHitek_MultiMessage(1, ino);
    SerialPortEventListener listener = new SerialPortEventListener() {
```

6. Una vez generadas las variables, se procede a generar algunos métodos que nos permiten realizar diferentes acciones, para ello inicialmente se establece el método que inicializa la recepción de datos por el puerto serial y almacena dichos datos en la variable varID, una vez almacenado los datos, se elimina el buffer para recibir nuevos datos mediante el lector NFC desarrollado.

```
    public void serialEvent(SerialPortEvent serialPortEvent){
      try {
            if(multi.dataReceptionCompleted()){
                varID = multi.getMessage(0);
                multi.flushBuffer();
            }
        } catch (ArduinoException ex) {
            Logger.getLogger(Control_Acceso.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SerialPortException ex) {
            Logger.getLogger(Control_Acceso.class.getName()).log(Level.SEVERE, null, ex);
        }
      consultaNombre();
      }
    );
```

7. En el siguiente paso es posible determinar cómo se generan las diferentes tablas que son utilizadas para el registro. En la siguiente imagen es posible visualizar como se agregan las diferentes columnas con su respectiva etiqueta y el ancho de cada una de las columnas, para las dos tablas consideradas en esta aplicación. Esto lo realizamos en el método principal ya que se debe ejecutar cuando se abra la aplicación. También se visualiza como se realiza el llamado a otros métodos que serán explicados posteriormente.

```
public Control_Acceso() {
    initComponents();
    obtenerPuerto();
    dataArduino();
    this.setLocationRelativeTo(null);
    tablaControl.addColumn("");tablaControl.addColumn("ID RFC");
    tablaControl.addColumn("Nombre");
    tablaControl.addColumn("Apellido");tablaControl.addColumn("Cedula");
    tablaControl.addColumn("Celular");tablaControl.addColumn("Fecha y Hora");
    Tabla_CAcceso.setModel(tablaControl);
    Tabla_CAcceso.getColumnModel().getColumn(0).setMaxWidth(0);
    Tabla_CAcceso.getColumnModel().getColumn(0).setMinWidth(0);
    Tabla_CAcceso.getColumnModel().getColumn(1).setPreferredWidth(85);
    Tabla_CAcceso.getColumnModel().getColumn(4).setPreferredWidth(55);
    Tabla_CAcceso.getColumnModel().getColumn(5).setPreferredWidth(50);
    Tabla_CAcceso.getTableHeader().getColumnModel().getColumn(0).setMaxWidth(0);
    Tabla_CAcceso.getTableHeader().getColumnModel().getColumn(0).setMinWidth(0);

    tablaUsuarios.addColumn("");tablaUsuarios.addColumn("Apellido");
    tablaUsuarios.addColumn("Nombre");
    tablaUsuarios.addColumn("Cedula");tablaUsuarios.addColumn("Celular");
    tablaUsuarios.addColumn("ID RFC");
    Tabla_UsuariosC.setModel(tablaUsuarios);
    Tabla_UsuariosC.getColumnModel().getColumn(0).setMaxWidth(0);
    Tabla_UsuariosC.getColumnModel().getColumn(0).setMinWidth(0);
    Tabla_UsuariosC.getTableHeader().getColumnModel().getColumn(0).setMaxWidth(0);
    Tabla_UsuariosC.getTableHeader().getColumnModel().getColumn(0).setMinWidth(0);


    mostrarDBUsuariosAcceso();
    mostrarDBregistroUsuarios();
}
```

8. Posteriormente se crea un método que permite obtener el puerto por el cuál se van a recibir los datos y también el método que es usado para empezar a recibir los datos acorde al puerto. En este apartado es indispensable marcar el puerto y la velocidad a la que se están recibiendo los datos.

```
private void obtenerPuerto() {
    temporal tm = new temporal();
    puerto = tm.getPuerto();
}

public void dataArduino() {
    try {
        JOptionPane.showMessageDialog(this, "Leyendo por el puerto " + puerto);
        ino.arduinoRX(puerto, 9600, listener);

    } catch (ArduinoException ex) {
        Logger.getLogger(Control_Acceso.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SerialPortException ex) {
        Logger.getLogger(Control_Acceso.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

9. El siguiente método muestra como se realizan las consultas de datos y estos datos obtenidos se disponen en las diferentes tablas generadas. Para ello, se accede a la colección en la que se encuentran almacenados los datos dentro de la base de datos en la nube y se recupera cada elemento de la colección.

```
public void mostrarDBregistroUsuarios() {
    MongoCursor<Document> consultaAcceso = DBControlAcceso.find().iterator();
    int totaltabla = tablaControl.getRowCount();
    for (int i = 0; i < totaltabla; i++) {
        tablaControl.removeRow(0);
    }
    while (consultaAcceso.hasNext()) {
        ArrayList<Object> docInv = new ArrayList<Object>(consultaAcceso.next().values());
        tablaControl.addRow(docInv.toArray());
    }
}

public void mostrarDBUsuariosAcceso() {
    MongoCursor<Document> consultaUsuariosAcceso = DBUsuariosAcceso.find().iterator();
    int totaltabla2 = tablaUsuarios.getRowCount();
    for (int i = 0; i < totaltabla2; i++) {
        tablaUsuarios.removeRow(0);
    }
    while (consultaUsuariosAcceso.hasNext()) {
        ArrayList<Object> docInv = new ArrayList<Object>(consultaUsuariosAcceso.next().values());
        tablaUsuarios.addRow(docInv.toArray());
    }
}
```

10. El siguiente método por usar, es aquel que permite realizar la consulta de cada elemento de manera individual, para ello, se genera un documento con las variables obtenidas y este será aquel que permita realizar la búsqueda, posteriormente se obtiene como resultado un documento de tipo JSON con toda la información, de tal manera que para obtener cada dato necesario se debe realizarlo mediante el valor getString y aplicar el filtro correspondiente.

```
public void consultaNombre() {
    try {
        filtro = new Document("NFCid", varID);
        Document resultfiltro = DBUsuarios.find(filtro).first();
        varapellido = (resultfiltro.getString("Apellido"));
        varnombre = (resultfiltro.getString("Nombre"));
        varcelular = (resultfiltro.getString("Celular"));
        varcedula = (resultfiltro.getString("Cedula"));
        txtNombre.setText(varnombre);
        txtApellido.setText(varapellido);
        txtId.setText(varID);
    } catch (Exception e) {
    }
}
```

11. Una vez finalizado los métodos, se procede a desarrollar la programación de los diferentes botones, en este caso se inicia con la programación del botón correspondiente a regresar al menú principal, en primera instancia es posible visualizar la configuración del botón para que este cambie cuando sea seleccionado y posteriormente se visualiza la acción que realiza el botón al momento de dar clic. En este apartado es posible visualizar que el botón cierra la ventana actual permitiendo regresar a la anterior y además deberá cerrar la conexión de comunicación serial establecida.

```
private void btnRegresarMouseEntered(java.awt.event.MouseEvent evt) {
    btnRegresarPanel.setBackground(new Color(224, 103, 106));
}

private void btnRegresarMouseClicked(java.awt.event.MouseEvent evt) {
    this.dispose();
    try {
        ino.killArduinoConnection();
    } catch (ArduinoException ex) {
        Logger.getLogger(Control_Acceso.class.getName()).log(Level.SEVERE, null, ex);
    }
}

private void btnRegresarMouseExited(java.awt.event.MouseEvent evt) {
    btnRegresarPanel.setBackground(new Color(204, 74, 78));
}
```

12. El siguiente botón programado es aquel que permite mostrar una interfaz diferente, para ello el código visualizado realiza la acción de cambiar el panel principal por el seleccionado y además se llama al método que llena la tabla, de este modo la tabla permanece actualizada cada que se ingrese a la interfaz, de la misma manera se realiza con el botón que llama a la interfaz principal.

```
private void btnRegistroTxtMouseClicked(java.awt.event.MouseEvent evt) {
    JPanelCentral.setSelectedIndex(1);
    mostrarDBregistroUsuarios();
}

private void btnRegistroTxtMouseEntered(java.awt.event.MouseEvent evt) {
    btnRegistro.setBackground(new Color(96, 177, 232));
}

private void btnRegistroTxtMouseExited(java.awt.event.MouseEvent evt) {
    btnRegistro.setBackground(new Color(52, 152, 219));
}
```

13. El siguiente botón es aquel que permite eliminar los usuarios de la lista de control de acceso, para ello inicialmente se realiza una búsqueda con el filtro basado en el ID obtenido mediante el lector NFC desarrollado, posteriormente se verifica si existe o no, en caso de no existir, se muestra un mensaje de advertencia, y en caso de si existir, se elimina dicho usuario y se muestra un mensaje de confirmación. Luego de ello se llama al método correspondiente para actualizar la tabla de usuarios registrados y se inicializan con valores neutrales las variables usadas.

```
private void btnEliminarUserTxtMouseClicked(java.awt.event.MouseEvent evt) {

    try {
        filtro2 = new Document("NFCid", varID);
        Document resultfiltro2 = DBUsuariosAcceso.find(filtro).first();
        if (resultfiltro2 == null) {
            JOptionPane.showMessageDialog(this, "El Usuario no tiene acceso ");
        } else {

            DBUsuariosAcceso.deleteOne(filtro2);
            JOptionPane.showMessageDialog(this, "Usuario Eliminado Correctamente");
        }

        mostrarDBUsuariosAcceso();
    } catch (Exception e) {
    }
    txtNombre.setText("");
    txtApellido.setText("");
    txtId.setText("");
    varID = "";
}
```

14. El botón “Agregar” es similar al botón eliminar, las acciones son prácticamente las mismas, de este modo, inicialmente se realiza una consulta y en caso de no existir ninguna coincidencia se agrega el usuario creando inicialmente un documento con los valores consultados de la base de datos y se actualiza la tabla con el método correspondiente, si por el contrario el usuario se encuentra en la lista, este no se agregará mostrando un mensaje de advertencia y posteriormente se vacían los datos de las variables utilizadas.

```
private void btnAgregarUserTxtMouseClicked(java.awt.event.MouseEvent evt) {
    try {
        filtro2 = new Document("NFCid", varID);
        Document resultfiltro2 = DBUsuariosAcceso.find(filtro).first();
        if (resultfiltro2 == null) {
            Document datosUsuario = new Document();
            datosUsuario.put("Apellido", varapellido);
            datosUsuario.put("Nombre", varnombre);
            datosUsuario.put("Cedula", varcedula);
            datosUsuario.put("Celular", varcelular);
            datosUsuario.put("NFCid", varID);
            DBUsuariosAcceso.insertOne(datosUsuario);
            JOptionPane.showMessageDialog(this, "Usuario Agregado Correctamente");

            mostrarDBUsuariosAcceso();
        } else {
            JOptionPane.showMessageDialog(this, "No es posible Agregar al Usuario");
        }

    } catch (Exception e) {
    }
    txtNombre.setText("");
    txtApellido.setText("");
    txtId.setText("");
    varID = "";
}
```

15. Luego se realiza la programación del botón que permite el registro manual en caso de que existan anomalías en el funcionamiento del sistema de control de acceso, para ello dicho botón inicialmente realiza una acción de modo que solicita la fecha y hora que se va a registrar el ingreso manual y posteriormente genera un

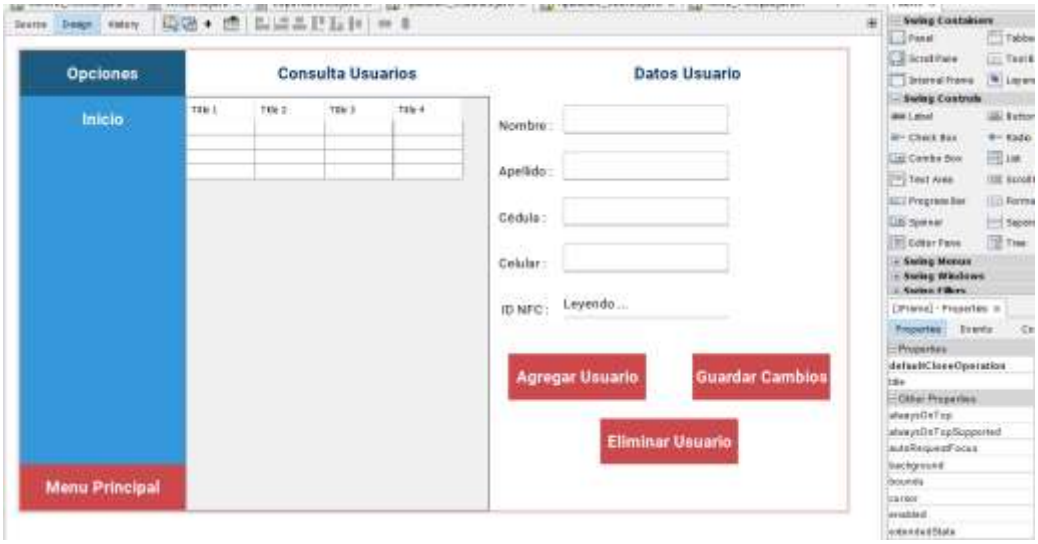
documento con los valores obtenidos de la base de datos tal como se visualiza para finalmente agregar dicho documento con el registro en la base de datos y actualiza el registro obtenido.

Con ello es posible visualizar el funcionamiento de cada uno de los elementos que conforman la aplicación.

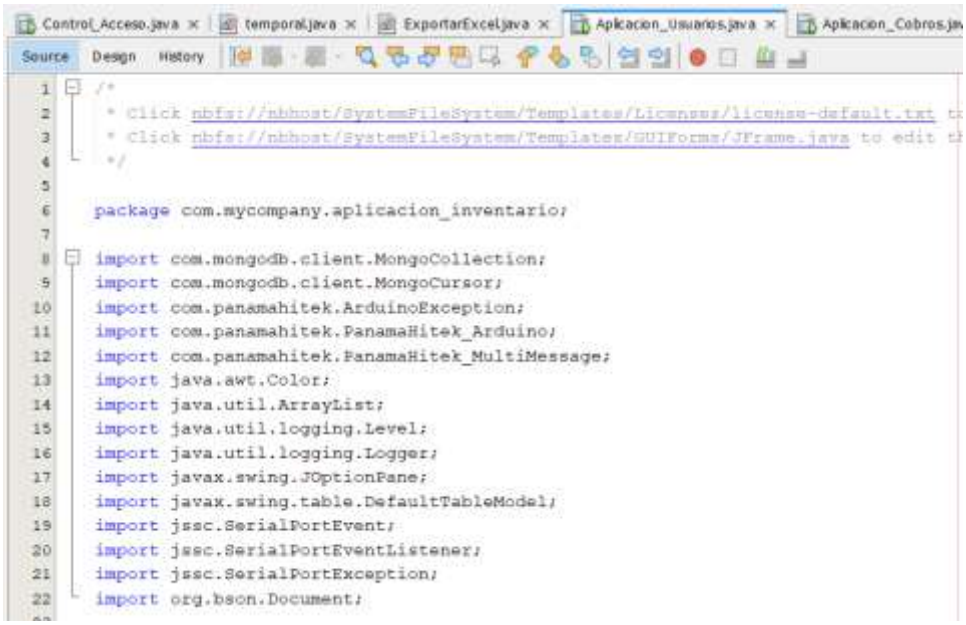
Desarrollo de Aplicación de Usuarios.

En esta sección se desarrolla la aplicación de gestión de usuarios, para ello dicha aplicación consta de una interfaz en la cual se tienen diferentes elementos, el desarrollo se muestra a continuación.

1. Inicialmente se utilizan los diferentes elementos que contiene el *Palette* y se construye la interfaz gráfica en la que se va a trabajar. De este modo se obtiene una interfaz como la siguiente.



2. Posteriormente se procede a realizar la programación de cada uno de los elementos, en este apartado inicialmente se deben importar todas las librerías que se van a utilizar para el desarrollo de la aplicación. En este apartado se visualiza librerías de mongo, comunicación serial y las librerías correspondientes a los elementos de Java.



3. Posteriormente se definen las variables que se van a utilizar y se establece la conexión con la base de datos y los puertos seriales. De la misma manera se define el puerto serial y se inicializa para que este pueda recibir los datos.

```
public class Aplicacion_Usuarios extends javax.swing.JFrame {
    Document filtro;
    String cad;
    String puerto;
    String varID;
    MongoCollection<Document> DBusuarios = new conexionDB().obtenerDB().getCollection("usuarios");
    DefaultTableModel tablaUsers = new DefaultTableModel(){
        @Override
        public boolean isCellEditable(int row, int colum){
            return false;
        }
    };
    PanamaHitek_Arduino ino = new PanamaHitek_Arduino();
    PanamaHitek_MultiMessage multi = new PanamaHitek_MultiMessage(1, ino);
    SerialPortEventListener listener = new SerialPortEventListener() {
      public void serialEvent(SerialPortEvent serialPortEvent){
        try {
            if(multi.dataReceptionCompleted()){
                varID = multi.getMessage(0);
                txtNFC.setText(varID);
                multi.flushBuffer();
            }
        } catch (ArduinoException ex) {
            Logger.getLogger(Aplicacion_Usuarios.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SerialPortException ex) {
            Logger.getLogger(Aplicacion_Usuarios.class.getName()).log(Level.SEVERE, null, ex);
        }
      }
    };
```

4. En el siguiente apartado muestra como se obtiene el puerto serial y además se genera la tabla correspondiente que muestra los usuarios, en el se observa las

etiquetas de cada columna y el ancho de las mismas para finalmente llamar al método correspondiente para llenar los datos de la tabla y recibir los elementos de Arduino.

```
public Aplicacion_Usuarios() {

    initComponents();
    obtenerPuerto();
    this.setLocationRelativeTo(null);

    tablaUsers.addColumn("");tablaUsers.addColumn("Apellido");
    tablaUsers.addColumn("Nombre");tablaUsers.addColumn("Cédula");
    tablaUsers.addColumn("Celular");tablaUsers.addColumn("IdNFC");

    tablaUsuarios.setModel(tablaUsers);
    tablaUsuarios.getColumnModel().getColumn(0).setMaxWidth(0);
    tablaUsuarios.getColumnModel().getColumn(0).setMinWidth(0);
    tablaUsuarios.getTableHeader().getColumnModel().getColumn(0).setMaxWidth(0);
    tablaUsuarios.getTableHeader().getColumnModel().getColumn(0).setMinWidth(0);
    mostrarUsuarios();
    dataArduino();
}
```

5. En la siguiente sección del código se muestra el método correspondiente para obtener el puerto que va a ser utilizado para la recepción de datos y el método que permite que se lean los datos, en dicho método, se tiene que establecer el puerto y la velocidad de transmisión.

```
private void obtenerPuerto(){
    temporal tm = new temporal();
    puerto = tm.getPuerto();
}
public void dataArduino (){
     try {
        JOptionPane.showMessageDialog (this,"Leyendo por el puerto"+puerto);
        ino.arduinoRX(puerto, 9600, listener);

    } catch (ArduinoException ex) {
        Logger.getLogger(Aplicacion_Usuarios.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SerialPortException ex) {
        Logger.getLogger(Aplicacion_Usuarios.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

6. Finalmente se tiene el método que permite agregar los valores de la base de datos a la tabla, para ello inicialmente se realiza la consulta y se van almacenando los datos dentro de cada tabla.

```
public void mostrarUsuarios (){
    MongoCursor<Document> consultaInventario = DBusuarios.find().iterator();
    //Eliminar datos de consulta anterior
    int totaltabla = tablaUsers.getRowCount();
    for(int i = 0; i<totaltabla; i++){
        tablaUsers.removeRow(0);
    }
    while (consultaInventario.hasNext()){
        ArrayList<Object> docInv = new ArrayList<Object>(consultaInventario.next().values());
        tablaUsers.addRow(docInv.toArray());
    }
}
```

7. El siguiente código muestra las funciones que realiza el botón “Guardar Cambios”, este botón inicialmente filtra mediante el número de cedula todos los

campos relacionados y los guarda en un documento. Una vez generado los cambios los almacena los datos obtenidos de los campos de texto, para posteriormente reemplazar el documento. Finalmente encera las variables utilizadas.

```
private void btnGuardarMouseClicked(java.awt.event.MouseEvent evt) {
    Document filtro = new Document("Cedula",ced);
    Document datosUsuario1 = new Document();
                    datosUsuario1.put("Apellido",txtApellido.getText() );
                    datosUsuario1.put("Nombre",txtNombre.getText() );
                    datosUsuario1.put("Cedula", txtCedula.getText());
                    datosUsuario1.put("Celular", txtCelular.getText());
                    datosUsuario1.put("NFCid", txtNFC.getText());
                    DBusuarios.updateOne(filtro, new Document("$set",datosUsuario1));
        JOptionPane.showMessageDialog (this,"Cambios Realizados Correctamente");
        txtNombre.setText("");
        txtApellido.setText("");
        txtCedula.setText("");
        txtCelular.setText("");
        txtNFC.setText("");
        mostrarUsuarios();
}
```

8. Posteriormente se desarrolla el código que permite agregar nuevos usuarios, para ello inicialmente se recuperan todos los textos de las cajas de texto y se almacenan, con el campo de cedula se genera un filtro y se procede a realizar una búsqueda, si el resultado de la búsqueda resulta positivo, entonces se genera un mensaje de que el usuario ya ha sido agregado, en caso de que no exista, se procede a generar un documento con los diferentes datos obtenidos mediante las cajas de texto y este es almacenado en la base de datos alojada en la nube para finalmente inicializar las variables utilizadas nuevamente.

```
private void btnAgregarMouseClicked(java.awt.event.MouseEvent evt) {
    String nombre = txtNombre.getText();
    String apellido = txtApellido.getText();
    String cedula = txtCedula.getText();
    String celular = txtCelular.getText();
    String idNFC = txtNFC.getText();
    Document filtro = new Document("Cedula",cedula);
    Document resultfiltro =DBusuarios.find(filtro).first();
    Document comp = new Document ("Cedula",cedula);
   if (resultfiltro == null){
    double saldo = 0.0;
    double matricula = 0.0;
    Document datosUsuario = new Document();
            datosUsuario.put("Apellido", apellido);
            datosUsuario.put("Nombre", nombre);
            datosUsuario.put("Cedula", cedula);
            datosUsuario.put("Celular", celular);
            datosUsuario.put("NFCid", idNFC);
            datosUsuario.put("Saldo", saldo);
            datosUsuario.put("Matricula", matricula);
    DBusuarios.insertOne(datosUsuario);
    JOptionPane.showMessageDialog (this,"Usuario Agregado Correctamente");
    txtNombre.setText("");
    txtApellido.setText("");
    txtCedula.setText("");
    txtCelular.setText("");
    txtNFC.setText("");
    mostrarUsuarios();
   }else {
   JOptionPane.showMessageDialog (this,"El Usuario ya existe");
   }
```

9. El siguiente método permite almacenar los datos de la tabla en los campos de texto, para ello inicialmente se establece la columna y posteriormente se extrae cada valor de la fila y se lo agrega en los campos de texto destinados para aquello.

```
private void tablaUsuariosMouseClicked(java.awt.event.MouseEvent evt) {
        int fila = tablaUsuarios.getSelectedRow();
        if(fila == -1){
            JOptionPane.showMessageDialog(this," No se seleccionó una fila");
        }
            else{
            String apel = (String) tablaUsuarios.getValueAt(fila,1);
            String nom = (String) tablaUsuarios.getValueAt(fila,2);
            ced = (String) tablaUsuarios.getValueAt(fila,3);
            String cel = (String) tablaUsuarios.getValueAt(fila,4);
            String nfc = (String) tablaUsuarios.getValueAt(fila,5);
            txtNombre.setText(nom);
            txtApellido.setText(apel);
            txtCedula.setText(ced);
            txtCelular.setText(cel);
            txtNFC.setText(nfc);
            )

}
```

10. El siguiente botón elimina los usuarios generados, para ello, inicialmente se realiza una búsqueda utilizando el número de cedula y posteriormente se procede a eliminar la entrada en la base de datos, una vez eliminado los datos, se procede

a generar un mensaje de confirmación y posteriormente se procede a encerar las variables y actualizar la tabla correspondiente.

```
private void btnEliminarMouseClicked(java.awt.event.MouseEvent evt) {
    Document filtro = new Document("Cedula",ced);
            DBusuarios.deleteOne(filtro);
    JOptionPane.showMessageDialog (this,"Usuario Eliminado Correctamente");
    txtNombre.setText("");
    txtApellido.setText("");
    txtCedula.setText("");
    txtCelular.setText("");
    txtNFC.setText("");
    mostrarUsuarios();


}
```

11. En este apartado se procede a establecer las restricciones para el ingreso de letras tanto en el campo de nombre como el de apellido, para ello el código inicialmente se genera el método que permite identificar la tecla presionada, posteriormente se establecen las condiciones en código ASCII, para ello, se definen las mayúsculas, minúsculas y los espacios. Una vez definidos los caracteres permitidos se realiza una comparación en la cual dice que si los caracteres presionados son diferentes de la condición, no serán tomados en cuenta, con ello, se permite siempre el ingreso de solo letras en el campo de nombre y apellido.

```
private void txtNombreKeyTyped(java.awt.event.KeyEvent evt) {

    int key = evt.getKeyChar();
    boolean mayusculas = key >= 65 && key <= 90;
    boolean minusculas = key >= 97 && key <= 122;
    boolean espacio = key == 32;
     if (!(minusculas || mayusculas || espacio))
    {
        evt.consume();
    }
}

private void txtApellidoKeyTyped(java.awt.event.KeyEvent evt) {
    int key = evt.getKeyChar();
    boolean mayusculas = key >= 65 && key <= 90;
    boolean minusculas = key >= 97 && key <= 122;
    boolean espacio = key == 32;
     if (!(minusculas || mayusculas || espacio))
    {
        evt.consume();
    }
}
```

12. De manera similar se procede a realizar la programación de restricciones correspondiente a números, para ello inicialmente se definen los caracteres permitidos, en este caso, de acuerdo a los valores ASCII, los números se encuentran comprendidos entre el 48 y el 57, posteriormente se realiza una comparación que permite el ingreso solo de números, finalmente se establece una

condición para que solo se permitan 10 caracteres como máximo, dado que es el número de dígitos que debe tener los números de cedula y el número de celular.

```
private void txtCedulaKeyTyped(java.awt.event.KeyEvent evt) {

    int key = evt.getKeyChar();
    boolean numeros = key >= 48 && key <= 57;
    if (!numeros)
    {
        evt.consume();
    }
    if (txtCedula.getText().trim().length() == 10) {
        evt.consume();
    }
}

private void txtCelularKeyTyped(java.awt.event.KeyEvent evt) {
    int key = evt.getKeyChar();
    boolean numeros = key >= 48 && key <= 57;
    if (!numeros)
    {
        evt.consume();
    }
    if (txtCelular.getText().trim().length() == 10) {
        evt.consume();
    }
}
```

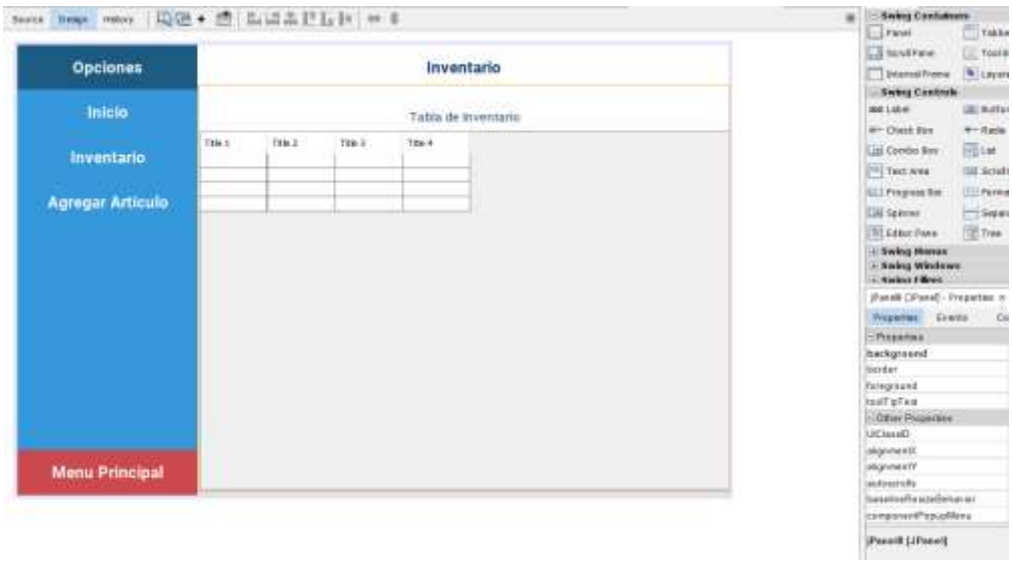
Con ello se termina la programación de la aplicación de gestión de usuarios.

Desarrollo de Aplicación de Inventario

Para desarrollar la aplicación de Inventario, se considera el entorno de programación en Java, de esta manera, se describe el proceso seguido para la programación de la aplicación.

1. De manera inicial, se crea un proyecto de tipo JFrame y posteriormente se procede a agregar todos los elementos que queremos en la interfaz a utilizar. Para cumplir con dicha aplicación se consideran 3 ventanas. Inicialmente se tiene la siguiente interfaz, en esta se puede visualizar etiquetas de texto, listas y botones, los cuales son añadidos arrastrándolos desde el panel lateral hacía la ventana principal

2. Posteriormente se desarrolla la siguiente interfaz, esta solamente considera una tabla, puesto que es la que permite visualizar el registro de los artículos inventarios registrados.



3. En la interfaz 3, se puede visualizar que se tiene un cuadro de texto, una etiqueta de texto y finalmente un botón que es el cual se debe presionar para registrar un nuevo elemento dentro del inventario.

4. Una vez se tienen las interfaces configuradas de la manera visualizada, se procede a la programación del código para que dichos elementos puedan realizar actividades. De manera inicial se importan las librerías correspondientes, en ellas se tienen librerías relacionadas con la conexión de la base de datos de la nube, librerías para comunicación serial y las librerías correspondientes a los elementos utilizados en la interfaz y aquellas librerías utilizadas para inicializar el puerto serial y los documentos de tipo Json.

```
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoCursor;
import com.panamahitek.ArduinoException;
import com.panamahitek.PanamaHitek_Arduino;
import com.panamahitek.PanamaHitek_MultiMessage;
import java.awt.Color;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import jssc.SerialPortEvent;
import jssc.SerialPortEventListener;
import jssc.SerialPortException;
import org.bson.Document;
```

5. Posteriormente se inicializan las diferentes variables utilizadas para la aplicación, en este apartado se tienen variables de tipo documento que servirán para la búsqueda y variables que permiten el almacenamiento de datos, con ello se generan las variables correspondientes a la conexión de base de datos y conexión mediante el puerto serial y recepción de datos.

```
Document filtro;
Document filtro2;
String listaUsuarios[] = new String[12];
String puerto;
String varID;
String varIdArticulo;
String varArticulo;
String varNombre;
String varApellido;
String varCelular;
String varNombreU;
String varApellidoU;
String varCelularU;
String varEstado;
String varFecha;
MongoCollection<Document> DBInventario = new conexionDB().obtenerDB().getCollection("registroInventario");
MongoCollection<Document> DBUsuarios = new conexionDB().obtenerDB().getCollection("usuarios");
MongoCollection<Document> DBUsuariosAcceso = new conexionDB().obtenerDB().getCollection("usuariosAcceso");
PanamaHitek_Arduino ino = new PanamaHitek_Arduino();
PanamaHitek_MultiMessage multi = new PanamaHitek_MultiMessage(1, ino);
SerialPortEventListener listener = new SerialPortEventListener() {
```

6. Con ello, el siguiente paso es definir los métodos, de manera inicial se tiene el método que inicia el servicio para recibir los datos mediante la comunicación serial. En este apartado se visualiza como se obtienen los datos, se almacenan y posteriormente se borran los datos almacenados en el buffer.

```
    public void serialEvent(SerialPortEvent serialPortEvent) {
        try {
            if (multi.dataReceptionCompleted()) {
                varID = multi.getMessage(0);
                multi.flushBuffer();
            }
        } catch (ArduinoException ex) {
            Logger.getLogger(Aplicacion_Inventario.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SerialPortException ex) {
            Logger.getLogger(Aplicacion_Inventario.class.getName()).log(Level.SEVERE, null, ex);
        }
        consulta();
    }
};
```

7. Lo siguiente es definir la tabla, para ello, inicialmente se definen las etiquetas de la tabla y posteriormente se generan las dimensiones de las mismas. En este apartado se visualiza como se establece el tamaño de cada una de las columnas.

```
        tablaRes.addColumn("");
        tablaRes.addColumn("RFC ID");
        tablaRes.addColumn("Articulo");
        tablaRes.addColumn("Estado");
        tablaRes.addColumn("Nombre");
        tablaRes.addColumn("Apellido");
        tablaRes.addColumn("Celular");
        tablaRes.addColumn("Responsable");
        tablaRes.addColumn("Fecha y Hora");

        //Opciones Tabla 1
        tablaPrestamo.setModel(tablaRes);
        tablaPrestamo.getColumnModel().getColumn(0).setMaxWidth(0);
        tablaPrestamo.getColumnModel().getColumn(0).setMinWidth(0);
        tablaPrestamo.getTableHeader().getColumnModel().getColumn(0).setMaxWidth(0);
        tablaPrestamo.getTableHeader().getColumnModel().getColumn(0).setMinWidth(0);
        tablaPrestamo.getColumnModel().getColumn(0).setPreferredWidth(100);
    }
```

8. Los siguientes métodos cumplen la función de establecer el puerto COM que se va a utilizar y posteriormente el método que permite se inicie la comunicación

mediante dicho puerto, en este se define el puerto y la velocidad de transmisión que se está utilizando.

```
private void obtenerPuerto() {
    temporal tm = new temporal();
    puerto = tm.getPuerto();
}

public void dataArduino() {
    try {
        JOptionPane.showMessageDialog(this, "Leyendo por el puerto " + puerto);
        ino.arduinoRX(puerto, 9600, listener);

    } catch (ArduinoException ex) {
        Logger.getLogger(Aplicacion_Inventario.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SerialPortException ex) {
        Logger.getLogger(Aplicacion_Inventario.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

9. Posteriormente se define el método que permite obtener datos de la base de mongoDB y los agrega a la lista, dicha lista será aquella de usuarios responsables al momento de realizar el préstamo de material hacia los estudiantes.

```
public void lista() {
    try {
        MongoCursor<Document> resultDocument = DBUsuariosAcceso.find().iterator();
        int i = 0;
        while (resultDocument.hasNext()) {
            listaUsuarios[i] = resultDocument.next().getString("Nombre");
            i++;
        }
        listaEncargados.setModel(new DefaultComboBoxModel<>(listaUsuarios));
    } catch (Exception e) {
    }
}
```

10. El siguiente método es la consulta que se debe realizar al momento de tener una lectura mediante el dispositivo NFC. En este apartado inicialmente se realiza una consulta a una base de datos para verificar si el elemento que se ha detectado es parte de los usuarios o parte del inventario registrado. Una vez realizada la consulta correspondiente a si es usuario o inventario, se procede a obtener los datos almacenados en la nube de acuerdo al filtro de Id NFC obtenido mediante la comunicación serial. Una vez filtrada la información se procede a almacenarla en las variables y posteriormente es mostrada en las etiquetas de texto mostradas en la interfaz. Luego se mediante la función se obtiene la hora y fecha del computador que será utilizado para registrar en que momento se realiza el préstamo de inventario y se muestra de igual manera en las etiquetas. Lo mismo sucede para el caso de inventario, se obtienen los datos, se almacenan y se muestran en las etiquetas de texto correspondiente.

```
public void consulta() {
    try {
        filtro = new Document("NFCid", varID);
        Document resultfiltro = DBUsuarios.find(filtro).first();
        if (resultfiltro == null) {
            filtro2 = new Document("idNFC", varID);
            Document resultfiltro2 = DBInventario.find(filtro2).first();
            if (resultfiltro2 != null) {
                varIdArticulo = (resultfiltro2.getString("idNFC"));
                varArticulo = (resultfiltro2.getString("articulo"));
                varEstado = (resultfiltro2.getString("estado"));
                varNombre = (resultfiltro2.getString("nombre"));
                varApellido = (resultfiltro2.getString("apellido"));
                varCelular = (resultfiltro2.getString("celular"));
                txtID.setText(varIdArticulo);
                txtArticulo.setText(varArticulo);
                LocalDateTime fechaHora = LocalDateTime.now();
                DateTimeFormatter formato = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm");
                varFecha = fechaHora.format(formato);
                txtFecha.setText(varFecha);
                if (varEstado == "prestado") {
                    txtNombre.setText(varNombre);
                    txtApellido.setText(varApellido);
                    txtCelular.setText(varCelular);
                }
            } else {
                txtIdRegistro.setText(varID);
            }

        } else {
            varApellidoU = (resultfiltro.getString("Apellido"));
            varNombreU = (resultfiltro.getString("Nombre"));
            varCelularU = (resultfiltro.getString("Celular"));
```

11. El código que se puede visualizar en la siguiente figura es el encargado de realizar la consulta de elementos a la base de datos y mostrarlos dentro de la tabla. Para ello inicialmente se realiza una consulta y se van agregando todos los valores consultados en dicha tabla.

```
public void mostrarDBInventario() {
    MongoCursor<Document> consultaRegInv = DBInventario.find().iterator();
    //Eliminar datos de consulta anterior
    int totaltablaRes = tablaRes.getRowCount();
    for (int i = 0; i < totaltablaRes; i++) {
        tablaRes.removeRow(0);
    }
    while (consultaRegInv.hasNext()) {
        ArrayList<Object> docRegInv = new ArrayList<Object>(consultaRegInv.next().values());
        tablaRes.addRow(docRegInv.toArray());
    }
}
```

12. El método de limpiar parámetros permite encerar todas las variables que han sido utilizadas para la obtención de datos, de este modo dicho método será llamado únicamente cuando se realice un registro.

```
public void limpiarParametros() {
    varIdArticulo = "";
    varArticulo = "";
    varEstado = "";
    varNombreU = "";
    varApellidoU = "";
    varCelularU = "";
    varFecha = "";
    txtNombre.setText("");
    txtApellido.setText("");
    txtCelular.setText("");
    txtID.setText("");
    txtArticulo.setText("");
    txtFecha.setText("");
    //"responsable", "Steven Flores");
}
```

13. Una vez finalizado los métodos, se procede a realizar la programación correspondiente a cada uno de los botones. El botón devolver obtiene el filtro mediante comunicación serial y posteriormente crea un nuevo documento con algunos valores establecidos, en este apartado es posible visualizar que algunos parámetros se quedan en blanco, dado que al momento de devolver dichos artículos, los elementos se encuentran en la asociación por lo que no existen datos de préstamo, finalmente al tener el documento, se procede a actualizar los datos y limpiar todas las variables utilizadas.

```
private void btnDevolverTxtMouseClicked(java.awt.event.MouseEvent evt) {
    Document datosInventario = new Document();
    datosInventario.put("idNFC", varIdArticulo);
    datosInventario.put("articulo", varArticulo);
    datosInventario.put("estado", "inventario");
    datosInventario.put("nombre", "  -  ");
    datosInventario.put("apellido", "  -  ");
    datosInventario.put("celular", "  -  ");
    datosInventario.put("responsable", "  -  ");
    datosInventario.put("fecha", "  -  ");
    filtro2 = new Document("idNFC", varIdArticulo);
    DBInventario.updateOne(filtro2, new Document("$set", datosInventario));
    JOptionPane.showMessageDialog(this, "Se Devolvió Correctamente");
    limpiarParametros();
}
```

14. La siguiente imagen muestra el código desarrollado que permite realizar el préstamo, para ello se crea un documento con todas las variables de métodos mostrados anteriormente, correspondientes al usuario y al articulo de inventario, además utiliza variables como la fecha que se obtiene de otros métodos y con ello actualiza los datos, pasando de estar el articulo en inventario a estado de prestado. Finalmente se llama al método que inicializa las variables eliminando todos los datos almacenados.

```
private void btnPrestarTxtMouseClicked(java.awt.event.MouseEvent evt) {
    Document datosInventario = new Document();
    datosInventario.put("idNFC", varIdArticulo);
    datosInventario.put("articulo", varArticulo);
    datosInventario.put("estado", "prestado");
    datosInventario.put("nombre", varNombreU);
    datosInventario.put("apellido", varApellidoU);
    datosInventario.put("celular", varCelularU);
    datosInventario.put("responsable", listaEncargados.getSelectedItem().toString());
    datosInventario.put("fecha", varFecha);
    filtro2 = new Document("idNFC", varIdArticulo);
    DBInventario.updateOne(filtro2, new Document("$set", datosInventario));
    JOptionPane.showMessageDialog(this, "Se Registro Correctamente");
    limpiarParametros();
}
```

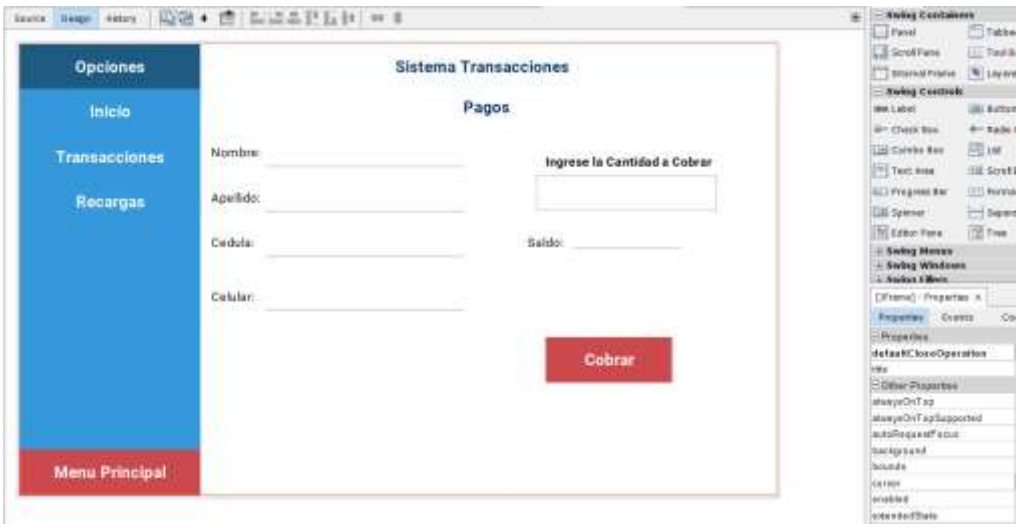
15. Para concluir la programación de esta aplicación se genera el código que permite agregar un nuevo artículo de inventario, para ello inicialmente se obtienen los campos que serán ingresados en el cuadro de texto de manera manual y posteriormente se genera un documento con el estado inicial del mismo. Una vez generado, se verifica que el articulo no haya sido registrado anteriormente para poder registrarse, además de que deberá tener un Id valido, en caso de que esto se cumpla, se registra correctamente y de no cumplirse las condiciones se envían mensajes de advertencia.

```
private void btnAgregarNuevoTxtMouseClicked(java.awt.event.MouseEvent evt) {
    String nuevoId = txtIdRegistro.getText();
    String nuevoArticulo = txtArticuloR.getText();
    Document datosNuevoArt = new Document();
    datosNuevoArt.put("idNFC", nuevoId);
    datosNuevoArt.put("articulo", nuevoArticulo);
    datosNuevoArt.put("estado", "inventario");
    datosNuevoArt.put("nombre", "  -  ");
    datosNuevoArt.put("apellido", "  -  ");
    datosNuevoArt.put("celular", "  -  ");
    datosNuevoArt.put("responsable", "  -  ");
    datosNuevoArt.put("fecha", "  -  ");
    filtro2 = new Document("NFCid", varIdArticulo);
    if (nuevoId != "") {
        if (nuevoArticulo.isEmpty()) {
            JOptionPane.showMessageDialog(this, "Parametro Articulo Vacio");
        } else {
            DBInventario.insertOne(datosNuevoArt);
            JOptionPane.showMessageDialog(this, "Se Agregó Correctamente");

        }

    } else {
        JOptionPane.showMessageDialog(this, "Parametro ID NFC Incorrecto");
    }
    txtIdRegistro.setText("");
    txtArticuloR.setText("");
}
```

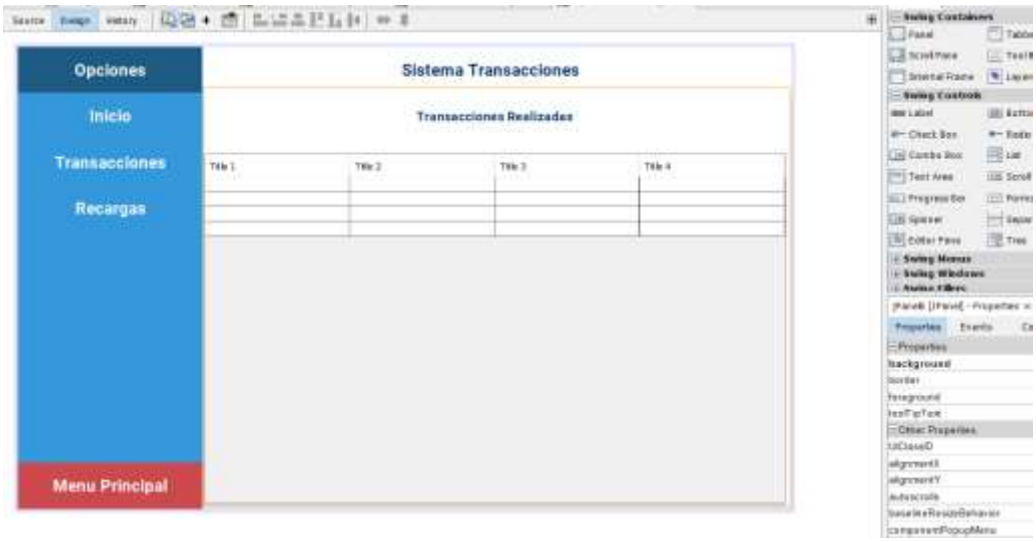
Desarrollo Aplicación Micro Transacciones

La aplicación a desarrollar será realizada en NetBeans utilizando el lenguaje de programación de JAVA para ello, a continuación, se muestra el código de programación utilizado.

1. La aplicación a desarrollar correspondiente a micro transacciones consta de 3 interfaces diferentes. La primera interfaz contiene elementos como etiquetas de texto un cuadro de texto y un botón, tal como se visualiza a continuación.

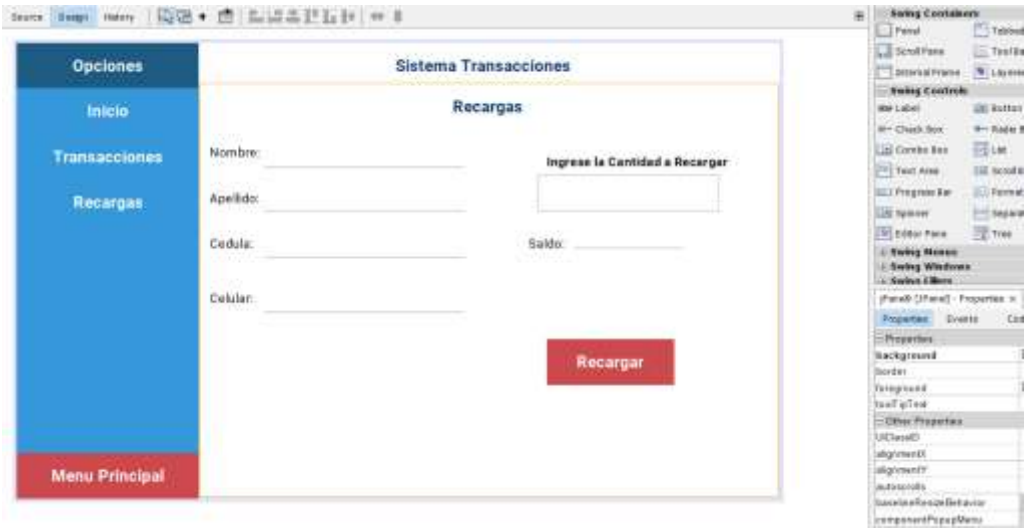


2. Posteriormente se diseña la segunda interfaz en esta se utiliza una tabla ya que dicha interfaz esta enfocada a mostrar el registro de micro transacciones realizadas.



3. Finalmente se tiene la interfaz número 3, la cuál es similar a la primera y está enfocada en las recargas de saldo que pueden darse. De este modo, para crear

dicha interfaz se utiliza etiquetas, un cuadro de texto y un botón que será el que realice la acción.



4. Una vez visualizado como se desarrollan las interfaces mediante el uso del panel lateral, se procede a realizar la programación de elementos. Inicialmente se procede a importar todas librerías necesarias para el correcto funcionamiento.

```
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoCursor;
import com.panamahitek.ArduinoException;
import com.panamahitek.PanamaHitek_Arduino;
import com.panamahitek.PanamaHitek_MultiMessage;
import java.awt.Color;
import java.awt.event.KeyEvent;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import jssc.SerialPortEvent;
import jssc.SerialPortEventListener;
import jssc.SerialPortException;
import org.bson.Document;
```

5. Posteriormente se definen las variables necesarias que serán utilizadas para el filtrado de documentos y también almacenar datos, posteriormente se definen las colecciones en las que se van a basar las búsquedas de la base de datos. Y se definen las variables usadas para la obtención de datos mediante el puerto COM y la comunicación serial. Luego se desarrolla el método para inicializar la comunicación serial la cual obtiene los datos y los almacena y posteriormente elimina el buffer para tener una nueva lectura. En este apartado también se incluye líneas de código que permiten que el código funcione a pesar de que se produzcan errores.

```
Document filtro;
Document filtro2;
String varID;
String puerto;
MongoCollection<Document> DBinventario = new conexionDB().obtenerDB().getCollection("inventario");
MongoCollection<Document> DBTransaccion = new conexionDB().obtenerDB().getCollection("registroTransaccion");
MongoCollection<Document> DBUsuarios = new conexionDB().obtenerDB().getCollection("usuarios");
MongoCollection<Document> DBReticos = new conexionDB().obtenerDB().getCollection("saldos");
PanamaHitek_Arduino ino = new PanamaHitek_Arduino();
PanamaHitek_MultiMessage multi = new PanamaHitek_MultiMessage(1, ino);
SerialPortEventListener listener = new SerialPortEventListener() {
  @Override
  public void serialEvent(SerialPortEvent serialPortEvent){
    try {
          if(multi.dataReceptionCompleted()){
              varID = multi.getMessage(0);
              multi.flushBuffer();
          }
      } catch (ArduinoException ex) {
          Logger.getLogger(Aplicacion_Cobros.class.getName()).log(Level.SEVERE, null, ex);
      } catch (SerialPortException ex) {
          Logger.getLogger(Aplicacion_Cobros.class.getName()).log(Level.SEVERE, null, ex);
      }
  consultaDatos();
  }
```

6. Lo siguiente a desarrollar es la programación correspondiente a la creación de tablas en las que se almacenan los datos de registro. De esta manera, se crean las columnas con las respectivas etiquetas y también se define el tamaño de cada una de ellas. Además, se llama algunos métodos que deben ser inicializados al momento de abrir la aplicación, dado que se encuentran dentro del método principal.

```
public Aplicacion_Cobros() {

    initComponents();
    obtenerPuerto();
    dataArduino();
    this.setLocationRelativeTo(null);
    tablaRes.addColumn("");
    tablaRes.addColumn("Nombre");tablaRes.addColumn("Apellido");
    tablaRes.addColumn("Cedula");tablaRes.addColumn("Transacción");
    tablaRes.addColumn("Saldo");tablaRes.addColumn("Fecha y Hora");
    tablaTransaccion.setModel(tablaRes);
    tablaTransaccion.getColumnModel().getColumn(0).setMaxWidth(0);
    tablaTransaccion.getColumnModel().getColumn(0).setMinWidth(0);
    tablaTransaccion.getTableHeader().getColumnModel().getColumn(0).setMaxWidth(0);
    tablaTransaccion.getTableHeader().getColumnModel().getColumn(0).setMinWidth(0);
}
```

7. Los siguientes métodos cumplen la función de establecer el puerto COM que se va a utilizar y posteriormente el método que permite se inicie la comunicación mediante dicho puerto, en este se define el puerto y la velocidad de transmisión que se está utilizando.

```
private void obtenerPuerto() {
    temporal tm = new temporal();
    puerto = tm.getPuerto();
}

public void dataArduino() {
    try {
        JOptionPane.showMessageDialog(this, "Leyendo por el puerto " + puerto);
        ino.arduinoRX(puerto, 9600, listener);

    } catch (ArduinoException ex) {
        Logger.getLogger(Aplicacion_Inventario.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SerialPortException ex) {
        Logger.getLogger(Aplicacion_Inventario.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

8. El siguiente método permite realizar las consultas al momento de realizar una lectura mediante el lector NFC desarrollado. De este modo inicialmente se obtiene los datos del ID NFC y se crea un nuevo documento, en base a dicho documento se recuperan valores de la base de datos como el nombre, cedula, apellido y celular y se almacenan en una variable, los mismos datos almacenados en las variables se muestran en las etiquetas de texto agregadas en la interfaz. Es importante mencionar que el saldo se encuentra en formato *double* por lo que es necesaria la conversión previamente.

```
public void consultaDatos(){
    filtro = new Document("NFCid",varID);
            Document resultfiltro =DBUsuarios.find(filtro).first();
            String varcedula = (resultfiltro.getString("Cedula"));
            String varapellido = (resultfiltro.getString("Apellido"));
            String varcelular = (resultfiltro.getString("Celular"));
            String varnombre = (resultfiltro.getString("Nombre"));
            Double varsaldo = (resultfiltro.getDouble("Saldo"));
            txtNombre.setText(varnombre);
            txtApellido.setText(varapellido);
            txtCedula.setText(varcedula);
            txtCelular.setText(varcelular);
            txtSaldo.setText("  $   "+String.valueOf(varsaldo));
            txtNombre2.setText(varnombre);
            txtApellido2.setText(varapellido);
            txtCedula2.setText(varcedula);
            txtCelular2.setText(varcelular);
            txtSaldo2.setText("  $   "+String.valueOf(varsaldo));
}
```

9. El siguiente método muestra el código necesario utilizado para llenar la tabla de transacciones con la información almacenada en la base de datos alojada en la nube.

```
public void mostrarTransaccionesDB (){
    MongoCursor<Document> consultaRegInv = DBTransaccion.find().iterator();
    //Eliminar datos de consulta anterior
    int totaltablaRes = tablaRes.getRowCount();
    for(int i = 0; i<totaltablaRes; i++){
        tablaRes.removeRow(0);
    }
    while (consultaRegInv.hasNext()){
        ArrayList<Object> docRegInv = new ArrayList<Object>(consultaRegInv.next().values());
        tablaRes.addRow(docRegInv.toArray());
    }
}
```

10. Una vez finalizado los métodos principales, se procede a llamarlos mediante el uso de botones, en la siguiente figura se muestra el código de los botones: “Inicio”, “Resumen” y “Inventario”, dichos botones realizan el llamado a la respectiva interfaz y en el caso del botón de resumen hace el llamado al método que llena los datos de la tabla.

```
private void btnInicioTxtMouseClicked(java.awt.event.MouseEvent evt) {
    JPanelCentral.setSelectedIndex(0);
}

private void btnResumenTxtMouseClicked(java.awt.event.MouseEvent evt) {
    JPanelCentral.setSelectedIndex(1);
    mostrarTransaccionesDB ();


}

private void btnInventarioTxtMouseClicked(java.awt.event.MouseEvent evt) {
    JPanelCentral.setSelectedIndex(2);
}
```

11. El botón que permite la transacción de cobro se define a continuación, para ello inicialmente se realiza una verificación para comprobar que efectivamente se requiere realizar la transacción, posteriormente se genera una comparación entre el saldo de usuario alojado en la base de datos y el saldo que se desea cobrar, si dicha condición se cumple, se procede a realizar la operación de resta, esto se lo debe realizar en datos de tipo *double*, por lo que se requiere una conversión previamente. Una vez se termine de realizar la transacción se almacena el nuevo saldo en una variable y con ella se actualiza dicho valor, además se genera un nuevo documento con todos los datos obtenidos de usuario y la fecha y hora para registrar dicha transacción en la base de datos. En el caso de que la primera opción no se cumpla, se tiene un mensaje de alerta especificando que existe un error.

```
private void btnCobrarMouseClicked(java.awt.event.MouseEvent evt) {
    int cobrar = JOptionPane.showConfirmDialog(this, "Quiere realizar el cobro?");
    if (cobrar == JOptionPane.OK_OPTION) {
        JOptionPane.showMessageDialog(this, "Realizando Accion");
        filtro = new Document("_id", varID);
        Document resultFiltro = DBUsuarios.find(filtro).first();
        Double valor1 = (resultFiltro.getDouble("Saldo"));
        String valor2str = txtValor.getText();
        Double valor2 = Double.parseDouble(valor2str);
        if (valor1 >= valor2) {
            try {
                Double valorRes = valor1 - valor2;
                double valorDec = Math.round(valorRes * 100.0) / 100.0;
                Document datosInventario = new Document();
                datosInventario.put("Saldo", valorDec);
                DBUsuarios.updateOne(filtro, new Document("$set", datosInventario));
                JOptionPane.showMessageDialog(this, "Se realizo la transaccion Correctamente, su saldo actual es $: " + valorDec);
                LocalDateTime fechaHora = LocalDateTime.now();
                DateTimeFormatter formato = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm");
                String fechayhora = fechaHora.format(formato);
                String nombre = txtNombre.getText();
                String apellido = txtApellido.getText();
                String cedula = txtCedula.getText();
                String transaccion = "pago";
                String saldoTran = valor2str;
                Document datosTransaccion = new Document();
                datosTransaccion.put("Nombre", nombre);
                datosTransaccion.put("Apellido", apellido);
                datosTransaccion.put("Cedula", cedula);
                datosTransaccion.put("Transaccion", transaccion);
                datosTransaccion.put("Saldo", saldoTran);
                datosTransaccion.put("Fechayhora", fechayhora);
                DBTransaccion.insertOne(datosTransaccion);
                JOptionPane.showMessageDialog(this, "Transaccion registrada");
```

12. Al igual que el botón anterior, el código siguiente muestra el proceso de aumento de saldo de la cuenta de usuario. Para ello inicialmente se tiene un dialogo de configuración y posteriormente se realiza el procedimiento, dicho procedimiento

consiste en adquirir los datos del lector NFC, realizar una búsqueda del usuario y obtener el saldo disponible, luego el valor es convertido a *double* para poder realizar operaciones y a este se le suma el valor ingresado mediante el cuadro de texto visualizado en la interfaz. Posteriormente se genera un documento y se actualizan los valores con los valores resultantes, además, también se genera un documento con el registro de la acción realizada y este es publicado en la base de datos alojada en la nube.

13. Lo siguiente es establecer el método que permita el ingreso de valores numéricos al cuadro de texto, en este apartado también se debe considerar valores de tipo decimal, para ello, inicialmente se definen los caracteres permitidos utilizando el valor del carácter ASCII y posteriormente se realiza una comparación en el que solo se permite el ingreso de valores numéricos y al momento de ingresar un valor diferente, este es desechado, para utilizar números decimales, inicialmente se debe detectar el uso de un punto, con ello, el sistema analiza el texto y mediante un ciclo for verifica que este solo tenga un punto, de esta manera, si se desea ingresar un segundo punto, el sistema no recibirá dicho carácter.

```
private void txtValorKeyTyped(java.awt.event.KeyEvent evt) {

    int key = evt.getKeyChar();
    boolean numeros = key >= 46 && key <= 57;
    if (!numeros) {
        evt.consume();
    }
    if (key == 46) {
        int tam = txtValor.getText().length();
        for (int i = 0; i <= tam; i++) {
            if (txtValor.getText().contains(".")) {
                evt.setKeyChar((char) KeyEvent.VK_CLEAR);
            }
        }
    }
}
```

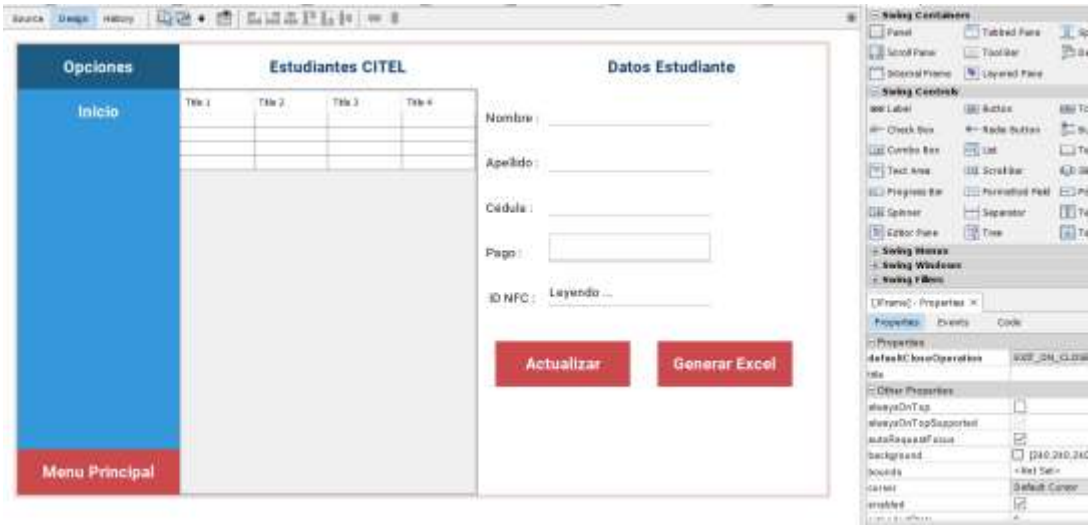
De esta manera se concluye el desarrollo de la programación para la aplicación de Micro Transacciones.

Desarrollo Aplicación Gestión Aportes Voluntarios

Para el desarrollo de dicha aplicación, se tiene el uso de la misma plataforma que las aplicaciones anteriores, en esta se tiene le uso de NetBeans con el lenguaje de programación JAVA.

Para desarrollar la programación correspondiente a la aplicación se consideran los siguientes pasos.

1. Inicialmente se desarrolla la interfaz con los diferentes elementos requeridos en la misma, para agregar elementos se debe utilizar el panel frontal denominado *Palette*. En esta aplicación se tiene el uso de una tabla para el registro de información, el uso de etiquetas de texto, un cuadro de texto y finalmente dos botones. De este modo al finalizar la interfaz se obtiene algo similar a lo que se observa en la siguiente figura.



2. Una vez definida toda la interfaz, se procede a la programación del código necesario para su correcto funcionamiento. Inicialmente se importan las librerías en las cuales se va a apoyar el código para el funcionamiento correcto, de esta manera se tienen librerías de conexión a la base de datos, librerías de conexión para comunicación serial y las librerías correspondientes a la interfaz gráfica.

```
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoCursor;
import com.panamahitek.ArduinoException;
import com.panamahitek.PanamaHitek_Arduino;
import com.panamahitek.PanamaHitek_MultiMessage;
import java.awt.Color;
import java.awt.Desktop;
import java.awt.event.KeyEvent;
import java.io.IOException;
import java.net.URI;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import jssc.SerialPortEvent;
import jssc.SerialPortEventListener;
import jssc.SerialPortException;
import org.bson.Document;
```

3. El siguiente a visualizar consiste en la creación de variables a utilizar de tipo documento y *String,* que permiten almacenar documentos en formato *Json* y cadenas de datos respectivamente. Posteriormente se generan las variables que permiten acceder a la colección correspondiente de la base de datos y las variables que se tienen para el uso de la comunicación serial.

   Además, también se visualiza el método que permite la obtención de datos de manera serial utilizando el puerto com. En dicho método se visualiza como los datos son almacenados en una variable, posteriormente se muestran en la interfaz

gráfica y finalmente se elimina el buffer para poder recibir una nueva entrada desde el Lector desarrollado.

```
Document filtro;
String ced;
String varID;
String puerto;
String celular;
MongoCollection<Document> DBusuarios = new conexionDB().obtenerDB().getCollection("usuarios");
DefaultTableModel tablaUsers = new DefaultTableModel(){
    @Override
    public boolean isCellEditable(int row, int colum){
        return false;
    }
};
PanamaHitek_Arduino ino = new PanamaHitek_Arduino();
PanamaHitek_MultiMessage multi = new PanamaHitek_MultiMessage(1, ino);
SerialPortEventListener listener = new SerialPortEventListener() {
  public void serialEvent(SerialPortEvent serialPortEvent){
    try {
          if(multi.dataReceptionCompleted()){
              varID = multi.getMessage(0);
              txtNFC.setText(varID);
              multi.flushBuffer();
          }
      } catch (ArduinoException ex) {
          Logger.getLogger(Aplicacion_Matricula.class.getName()).log(Level.SEVERE, null, ex);
      } catch (SerialPortException ex) {
          Logger.getLogger(Aplicacion_Matricula.class.getName()).log(Level.SEVERE, null, ex);
      }
```

4. Una vez terminado el método inicial, se definen los valores que tendrá la tabla utilizada para visualizar el registro de aquellos usuarios que tengan los valores de aportación cancelados. En esta sección del código, es posible visualizar como se agregan columnas con sus respectivas etiquetas y posteriormente se asignan valores de tamaño a cada una de las columnas que contiene la tabla, en este apartado se puede visualizar que existen columnas con valor de 0, esto permite gestionar y organizar de mejor manera los datos obtenidos de la base de datos.

```
tablaUsers.addColumn("");tablaUsers.addColumn("Apellido");
tablaUsers.addColumn("Nombre");tablaUsers.addColumn("Cédula");
tablaUsers.addColumn("");tablaUsers.addColumn("");tablaUsers.addColumn("");tablaUsers.addColumn("Matrícula");
tablaUsuarios.setModel(tablaUsers);
//
tablaUsuarios.getColumnModel().getColumn(0).setMaxWidth(0);
tablaUsuarios.getColumnModel().getColumn(0).setMinWidth(0);
tablaUsuarios.getTableHeader().getColumnModel().getColumn(0).setMaxWidth(0);
tablaUsuarios.getTableHeader().getColumnModel().getColumn(0).setMinWidth(0);
//
tablaUsuarios.getColumnModel().getColumn(4).setMaxWidth(0);
tablaUsuarios.getColumnModel().getColumn(4).setMinWidth(0);
tablaUsuarios.getTableHeader().getColumnModel().getColumn(4).setMaxWidth(0);
tablaUsuarios.getTableHeader().getColumnModel().getColumn(4).setMinWidth(0);
//
tablaUsuarios.getColumnModel().getColumn(5).setMaxWidth(0);
tablaUsuarios.getColumnModel().getColumn(5).setMinWidth(0);
tablaUsuarios.getTableHeader().getColumnModel().getColumn(5).setMaxWidth(0);
tablaUsuarios.getTableHeader().getColumnModel().getColumn(5).setMinWidth(0);
//
tablaUsuarios.getColumnModel().getColumn(6).setMaxWidth(0);
tablaUsuarios.getColumnModel().getColumn(6).setMinWidth(0);
tablaUsuarios.getTableHeader().getColumnModel().getColumn(6).setMaxWidth(0);
tablaUsuarios.getTableHeader().getColumnModel().getColumn(6).setMinWidth(0);
```

5. Luego se tiene la programación para la obtención del puerto que se va a utilizar y también el método que permite la lectura de los datos por dicho puerto, en este

método es esencial marcar el puerto COM utilizado y la velocidad con la que se transmiten los datos.

```
private void obtenerPuerto(){
    temporal tm = new temporal();
    puerto = tm.getPuerto();
}
public void dataArduino (){
     try {
        JOptionPane.showMessageDialog (this,"Leyendo por el puerto: " + puerto);
        ino.arduinoRX(puerto, 9600, listener);

    } catch (ArduinoException ex) {
        Logger.getLogger(Aplicacion_Matricula.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SerialPortException ex) {
        Logger.getLogger(Aplicacion_Matricula.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

6. La siguiente sección de código muestra el método utilizado para que la tabla de registro sea llenada con datos obtenidos de la base de datos, en este código podemos visualizar como se genera la tabla y se van agregando los datos obtenidos de la colección de la base de datos designada. Dicho proceso se realiza mientras exista un elemento para almacenar.

```
public void mostrarUsuarios (){
    MongoCursor<Document> consultaInventario = DBusuarios.find().iterator();
    //Eliminar datos de consulta anterior
    int totaltabla = tablaUsers.getRowCount();
    for(int i = 0; i<totaltabla; i++){
        tablaUsers.removeRow(0);
    }
    while (consultaInventario.hasNext()){
        ArrayList<Object> docInv = new ArrayList<Object>(consultaInventario.next().values());
        tablaUsers.addRow(docInv.toArray());
    }
}
```

7. Para realizar las consultas a la base de datos que permitan recuperar los valores, se debe realizar mediante el siguiente código, este código inicialmente recupera los datos mediante el lector NFC y posteriormente crea un filtro que es utilizado para realizar la búsqueda, una vez realizada, se recuperan los datos de usuario asociados a dicha búsqueda. Con esto se obtienen los datos y son asignados a cuadros de texto que se muestran en la interfaz principal visualizada anteriormente.

```
public void consulta(){
try {
        filtro = new Document("NFCid",varID);
        Document resultfiltro =DBusuarios.find(filtro).first();
                String varApellidoU = (resultfiltro.getString("Apellido"));
                String varNombreU = (resultfiltro.getString("Nombre"));
                String varCedulaU = (resultfiltro.getString("Cedula"));
                celular = (resultfiltro.getString("Celular"));
                ced = varCedulaU;
                Double varMatriculaU = (resultfiltro.getDouble("Matricula"));
                txtNombre.setText(varNombreU);
                txtApellido.setText(varApellidoU);
                txtCedula.setText(varCedulaU);
                txtPago.setText(String.valueOf(varMatriculaU));
    } catch (Exception e) {
    }
}
```

8. A continuación se muestra la programación del botón que permite guardar un nuevo registro del pago de aportación que se está realizando. Inicialmente se realiza una consulta del usuario, posteriormente se utiliza el nuevo valor del aporte para generar un documento y se actualizan los valores con este, posteriormente se muestra un mensaje de confirmación para verificar si se requiere algún tipo de comprobante o no, en el caso de que la respuesta sea afirmativa, el sistema procede a generar una cadena de texto que se convierte en URL y esta a su vez abre la ventana de navegador permitiendo enviar un mensaje de confirmación al número de teléfono registrado, si la respuesta es negativa solo se obtiene un mensaje que muestra que el proceso ha sido correcto, finalmente se limpian los valores presentados en las etiquetas de texto de la interfaz.

```
private void btnMatriculatxtMouseClicked(java.awt.event.MouseEvent evt) {
    Document filtro = new Document("Cedula",ced);
    String valor2str = txtPago.getText();
    Double valor = Double.parseDouble(valor2str);
    Document datosUsuario1 = new Document();
                datosUsuario1.put("Matricula",valor);
                DBusuarios.updateOne(filtro, new Document("$set",datosUsuario1));
        mostrarUsuarios();
        int response = JOptionPane.showConfirmDialog(this, "Desea recibir un comprobante","Confirmacion",JOptionPane.YES_NO
        if(response == JOptionPane.YES_OPTION){
            try {
            String sms_comp = "https://api.whatsapp.com/send?phone=593"+celular+"&text=ASOCIACION%20CITEL-CIENCIAS%20Pago%20
            Desktop d = Desktop.getDesktop();
            d.browse(new URI(sms_comp));
            } catch(Exception e){
            }
        }
        JOptionPane.showMessageDialog (this,"Cambios Realizados Correctamente");
        txtNombre.setText("");
        txtApellido.setText("");
        txtCedula.setText("");
        txtPago.setText("");
        txtNFC.setText("");

    }
```

9. Una de las consideraciones que se debe tener en cuenta, es que el cuadro de texto solo permita el ingreso de valores numéricos, para ello se utiliza el siguiente código. Este código permite solamente el ingreso de caracteres numéricos de acuerdo al sistema ASCII, de tal modo que, si no es parte de los caracteres permitidos, el sistema no ingresa dicho carácter al cuadro de texto, además también se tiene una segunda comprobación de tal manera que el valor numérico

ingresado solo tenga un punto, indicando que es decimal, sin esta comprobación el sistema permitiría el ingreso de una cadena de puntos, lo que desbordaría en una falla del mismo.

```
private void txtPagoKeyTyped(java.awt.event.KeyEvent evt) {
    int key = evt.getKeyChar();
boolean numeros = key >= 46 && key <= 57;
    if (!numeros)
    {
        evt.consume();
    }
    if(key == 46){
        int tam = txtPago.getText().length();
        for ( int i=0; i<=tam;i++){
            if(txtPago.getText().contains(".")){
            evt.setKeyChar((char)KeyEvent.VK_CLEAR);
            }
        }
    }
}
```

10. Finalmente se tiene la programación del botón que permite generar una hoja de Excel a partir de la tabla mostrada. El botón cumple la función de llamar una nueva clase que se encarga de generar el Excel, para ello se programa el nombre de la tabla y está tabla se almacena en un objeto que es exportado a otra clase.

```
  private void btnGenerarExcelMouseClicked(java.awt.event.MouseEvent evt) {
     ExportarExcel obj;
      try {
          obj = new ExportarExcel();
          obj.exportarExcel(tablaUsuarios);
      } catch (IOException ex) {
          System.out.println("Error: " + ex);
      }
  }
```

11. Aquella clase que permite generar el archivo de Excel se visualiza a continuación. Inicialmente se importan las librerías necesarias para que se pueda exportar dicha tabla en forma de documento Excel.

```
import java.awt.Desktop;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import javax.swing.JFileChooser;
import javax.swing.JTable;
import javax.swing.filechooser.FileNameExtensionFilter;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
```

12. Posteriormente se programa el código que permite la generación del archivo .xls

En este apartado es posible visualizar como se genera un cuadro de elección inicialmente, en el que es posible agregar el nombre del archivo que se va a generar además de escoger la ubicación en la que se va a guardar, posteriormente se compara si tiene un nombre correcto y se genera un archivo de formato xls.

Una vez escogido la extensión del archivo, este empieza a ser llenado mediante las librerías anteriormente importadas y el uso de la tabla que se había seleccionado.

De esta manera se van obteniendo de forma periódica todas las filas y columnas, se crea un libro de Excel y se almacenan todos los datos que contiene la tabla anteriormente seleccionada y con ello se genera un archivo xls.

```
public void exportarExcel(JTable t) throws IOException {
    JFileChooser chooser = new JFileChooser();
    FileNameExtensionFilter filter = new FileNameExtensionFilter("Archivos de excel", "xls");
    chooser.setFileFilter(filter);
    chooser.setDialogTitle("Guardar archivo");
    chooser.setAcceptAllFileFilterUsed(false);
    if (chooser.showSaveDialog(null) == JFileChooser.APPROVE_OPTION) {
        String ruta = chooser.getSelectedFile().toString().concat(".xls");
        try {
            File archivoXLS = new File(ruta);
            if (archivoXLS.exists()) {
                archivoXLS.delete();
            }
            archivoXLS.createNewFile();
            Workbook libro = new HSSFWorkbook();
            FileOutputStream archivo = new FileOutputStream(archivoXLS);
            Sheet hoja = libro.createSheet("Mi hoja de trabajo 1");
            hoja.setDisplayGridlines(false);
            for (int f = 0; f < t.getRowCount(); f++) {
                Row fila = hoja.createRow(f);
                for (int c = 0; c < t.getColumnCount(); c++) {
                    Cell celda = fila.createCell(c);
                    if (f == 0) {
                        celda.setCellValue(t.getColumnName(c));
                    }
                }
            }
            int filaInicio = 1;
            for (int f = 0; f < t.getRowCount(); f++) {
                Row fila = hoja.createRow(filaInicio);
                filaInicio++;
                for (int c = 0; c < t.getColumnCount(); c++) {
                    Cell celda = fila.createCell(c);
```

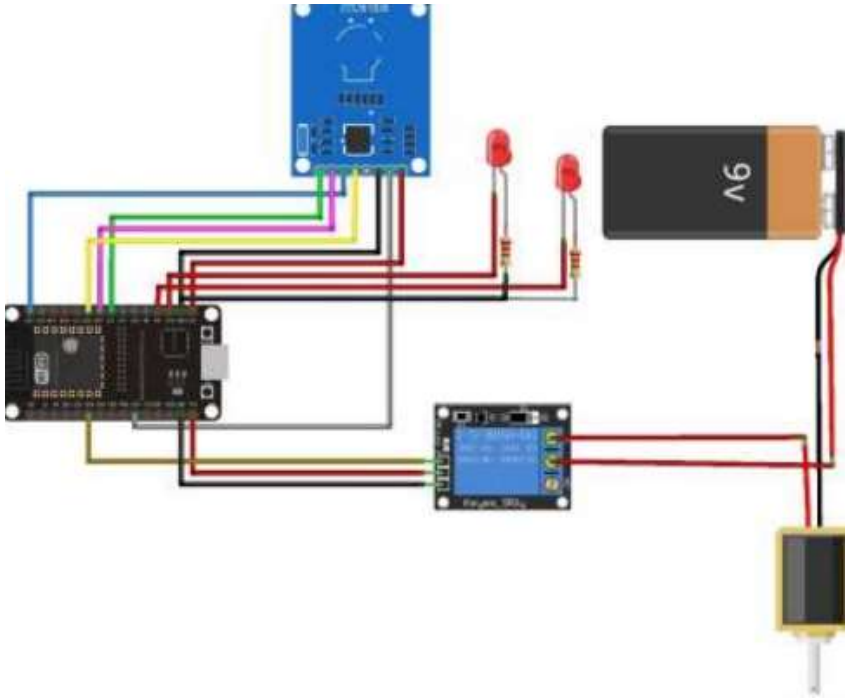
De esta manera, finaliza la documentación de los diferentes códigos desarrollados para las aplicaciones presentadas.

ANEXO C - MANUAL DE ADMINISTRADOR

Para la correcta implementación de los sistemas, se deben realizar algunas configuraciones iniciales, de esta manera, se describen los pasos que se deben seguir para la configuración correcta del sistema.

1. **Diagrama de Conexión de Control de Acceso.**

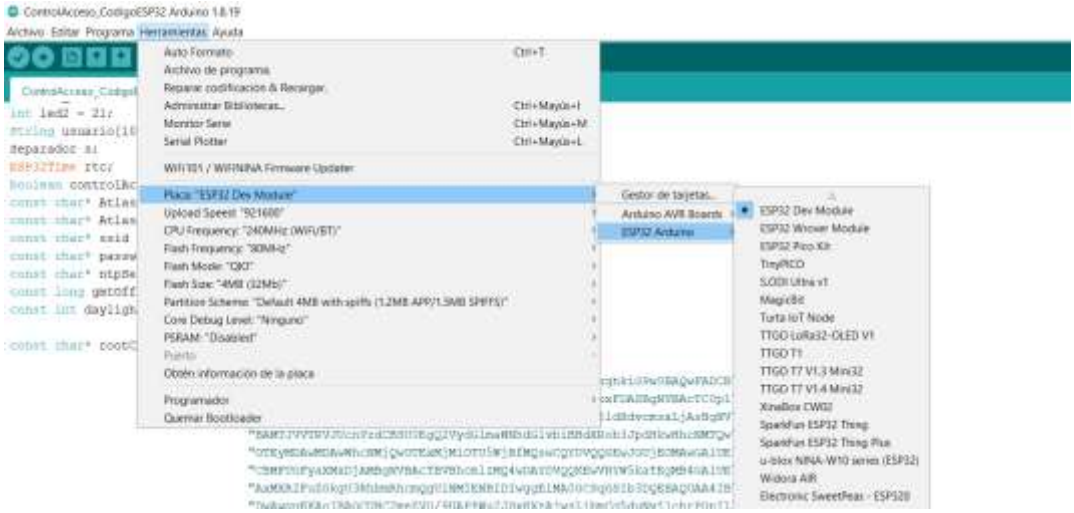
Lo primero a realizar es interconectar los elementos conforme se muestra en el siguiente esquema y verificar que los dispositivos enciendan.



2. **Conexión a internet del prototipo de control de acceso**

Para realizar la respectiva conexión correspondiente, se deberá actualizar el código con los nuevos valores de SSID y Clave de la red inalámbrica que se va a utilizar.

Para ello inicialmente se procede a configurar la placa en el entorno de Arduino tal como se observa en la siguiente imagen, en ella se debe especificar la placa que se está utilizando y el modelo del micro controlador.



Una vez verificado la placa, se realiza los respectivos cambios en el código en el cual se especifica el SSID y clave de acceso de la red tal como se muestra en la siguiente imagen.

Una vez asignada el SSID y clave de la red respectiva, se procederá a cargar el nuevo código en el dispositivo ESP32 que deberá encontrarse conectado al PC.

Para ello se realiza clic en la ventana de cargar y se debe esperar a que el programa sea cargado adecuadamente.

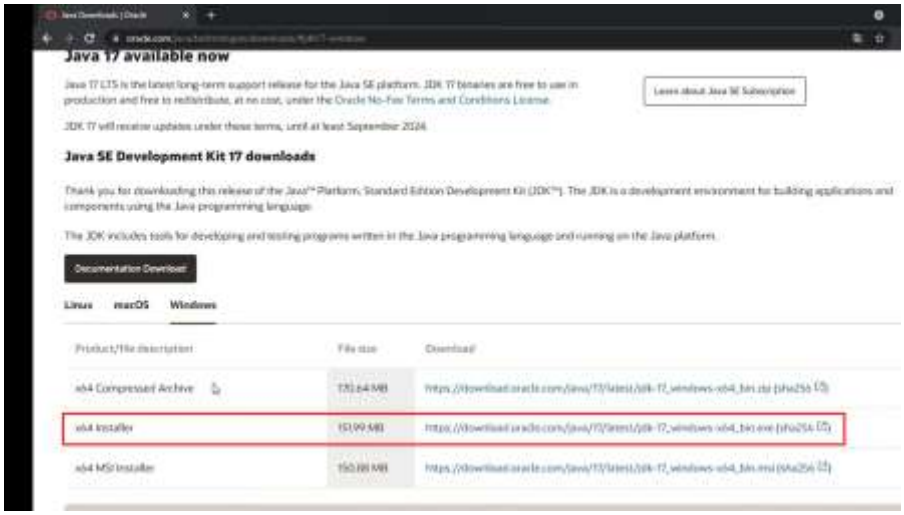
Con estos pasos y la interconexión mencionada anteriormente el dispositivo de control de acceso funcionará adecuadamente.

3. Configuración Modulo Lector Desarrollado

Para realizar las respectivas configuraciones del modulo lector, solo será necesario conectar el dispositivo a la PC mediante un puerto USB y posteriormente cargar el código correspondiente de manera similar a la anterior, el proceso se visualiza en la siguiente figura.

4. Instalación y configuración de la Aplicación Desarrollada.

Para el correcto funcionamiento de la aplicación, inicialmente se debe descargar los elementos necesarios para el funcionamiento de la misma. Inicialmente se debe realizar la descarga del paquete de Java, para ello se ingresa a la pagina oficial de Oracle y posteriormente se descarga el paquete de instalación de Java como se visualiza a continuación.



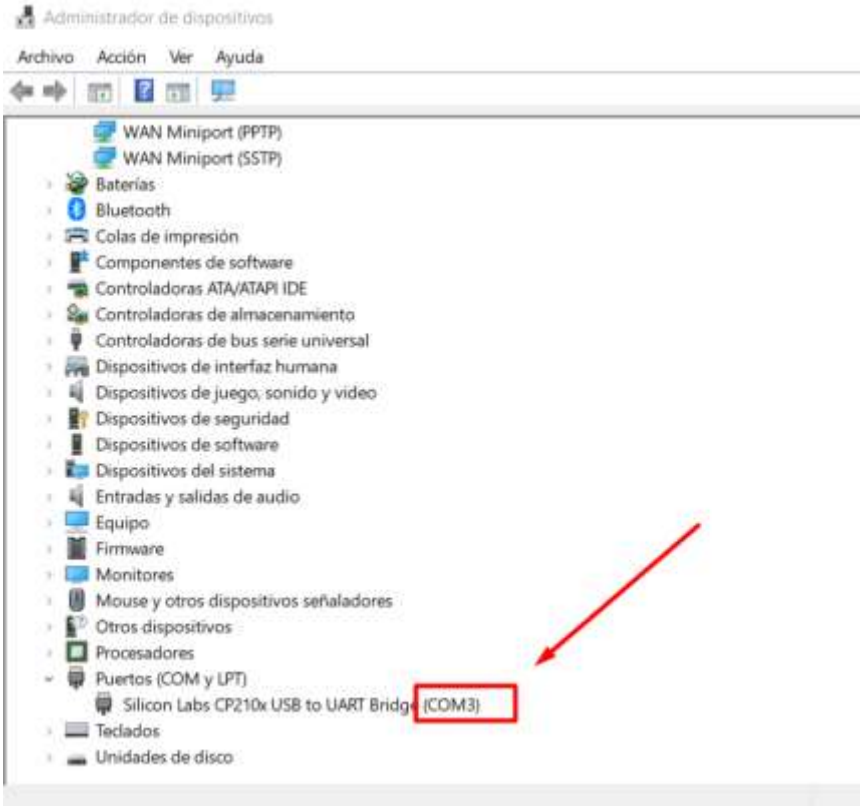
Una vez descargado el paquete de instalación se ejecuta y se sigue los pasos tal como se muestra a continuación.

Se procede a escoger la carpeta en la que se desea instalar el paquete de JAVA.

Una vez finalizada la instalación se obtiene un mensaje como el siguiente, este será el indicio de que la aplicación desarrollada puede funcionar correctamente.

Una vez realizada la aplicación de Java se procede a abrir la aplicación obteniendo el siguiente resultado.

En este apartado se deberá realizar la conexión entre la PC y el módulo de lector desarrollado utilizando un puerto USB. Una vez realizada la conexión, se dirige al administrador de dispositivos y se identifica el puerto mediante el cual se ha conectado el modulo lector, en este caso es el puerto 3 tal como se muestra en la siguiente figura.



Una vez identificado el puerto, se procede a seleccionar el mismo puerto en la aplicación y con ello, está funcionará correctamente.