UNIVERSIDAD TÉCNICA DEL NORTE

Facultad de Ingeniería en Ciencias Aplicadas

Carrera de Ingeniería en Sistemas Computacionales

DESARROLLO DE UNA APLICACIÓN, PARA LA GESTIÓN DE DENUNCIAS DE LA JCPD DEL GAD MUNICIPAL DEL CANTÓN OTAVALO MEDIANTE EL FRAMEWORK "ELECTRON"

Trabajo de grado previo a la obtención del título de Ingeniero en Sistemas Computacionales

AUTOR:

Bryan Xavier Inuca Chicaiza

DIRECTOR:

Msc. Daisy Elizabeth Imbaquingo Esparza

Ibarra, 2020

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 DEL AUTOR

CÉDULA DE IDENTIDAD:	100311187-7		
APELLIDOS Y NOMBRES:	INUCA CHICAIZA BRYAN XAVIER		
DIRECCIÓN:	GRAN COLOMBIA Y JORGE JARRIN - GONZÁLES SUÁREZ		
EMAIL:	bryanxavier95@gmail.com, hxinucac@utn.edu.ec		
CELULAR:	0996322192	**TELEFONO FIJO:**	062918472

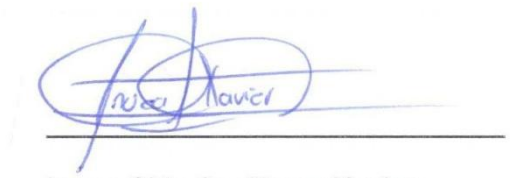
DATOS DE LA OBRA

TÍTULO:	DESARROLLO DE UNA APLICACIÓN, PARA LA GESTIÓN DE DENUNCIAS DE LA JCPD DEL GAD MUNICIPAL DEL CANTÓN OTAVALO MEDIANTE EL FRAMEWORK “ELECTRON”.
AUTOR:	INUCA CHICAIZA BRYAN XAVIER
FECHA:	06 DE FEBRERO DE 2020
PROGRAMA:	Pregrado
TITULO POR EL QUE OPTA:	Pregrado
DIRECTOR:	Msc. DAISY IMBAQUINGO

2. CONSTANCIA

El autor manifiesta 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 el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 6 días del mes de febrero de 2020.



Inuca Chicaiza Bryan Xavier

CI: 100311187-7

UNIVERSIDAD TÉCNICA DEL NORTE

Resolución No. 001-073 CEAACES-2013-13

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

Ibarra, 7 de febrero 2020

CERTIFICACIÓN DIRECTOR

Por medio del presente, yo MSc. Daisy Imbaquingo, certifico que el Sr. Bryan Xavier Inuca Chicaiza, portador de la cédula de identidad Nro.100311187-7. Ha trabajo en el desarrollo del proyecto de grado denominado "**DESARROLLO DE UNA APLICACIÓN, PARA LA GESTIÓN DE DENUNCIAS DE LA JCPD DEL GAD MUNICIPAL DEL CANTÓN OTAVALO MEDIANTE EL FRAMEWORK "ELECTRON"**", previo a la obtención del título de Ingeniería en sistemas computacionales, lo cual ha realizado en su totalidad con responsabilidad y esmero.

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

Atentamente,

MSc. Daisy Imbaquingo

DIRECTORA DE TESIS

DEDICATORIA

Dedico este proyecto de fin de carrera a mis familiares, amigos y compañeros que en algún momento durante estos años de estudio me acompañaron, guiaron y apoyaron. En especial a mis padres, que sin su esfuerzo e inalcanzable apoyo nunca tendría la dicha de realizar este uno de los tantos objetivos planeados. A mis pequeños hermanos que son el pilar fundamental de mi vida, quienes siguen mis pasos y quiero ser un ejemplo para ellos.

No hay obstáculos imposibles; hay voluntades mas fuertes y mas débiles, ¡Eso es todo!

Julio Verne

AGRADECIMIENTOS

A mis padres por todo el esfuerzo, atención y constante preocupación de toda mi vida; mis logros son gracias a su ayuda y apoyo incondicional. Ellos han representado siempre el motor que mueve mi vida y me han motivado a avanzar y cumplir mis objetivos trazados.

Un agradecimiento muy especial a la MSc. Daisy Imbaquingo quien aceptó ser mi Directora de Trabajo de Grado, por todo el apoyo brindado, consejos, por su tiempo, sobre todo por su amistad y sus valiosas sugerencias en momentos de duda para poder culminar con éxito este proyecto.

A la Universidad Técnica del Norte por haberme acogido durante mi formación académica por formarme como persona, profesional y encaminarme para poder servir a la sociedad.

A mi familia, en especial a mi abuelita Clorinda por sus palabras de aliento y apoyo, siempre me ha inspirado con su ejemplo de humildad, honradez, responsabilidad y perseverancia.

Bryan Xavier Inuca Chicaiza

Tabla de Contenido

Índice de Figuras

Índice de Cuadros

Resumen

El Gobierno Autónomo Descentralizado Municipal de Otavalo con su departamento “Junta Cantonal de protección de derechos de la niñez y adolescencia(JCPD)” viene llevando a cabo el registro y control de denuncias, mismo que no permite llevar un control adecuado del registro de las denuncias o casos, avocatorias, audiencias y resoluciones, el sistema a desarrollarse permitirá a cabalidad cumplir una gran parte del proceso de llevar un control y seguimiento de las denuncias hasta terminar su proceso cuando ya se tengan una resolución.

Electron es un potente framework de desarrollo de aplicaciones de escritorio multiplataforma, este nos permite desarrollar aplicaciones en corto tiempo ya que es posible realizar la compilación de estas en diferentes plataformas (Microsoft, MacOS, Linux) haciendo uso del mismo código fuente.

En la Introducción, se declara el objetivo general y específicos; se detallan los problemas que han generado la falta de gestión de denuncias de la JCPD-Otavalo y se plantea una solución.

En el capitulo 1: Se redacta la teoría de donde proviene la JCPD-Otavalo y la especificación en base a las leyes de su proceso. Además, un breve resumen de las herramientas a usarse para la validación del sistema.

En el capitulo 2: Se describe el estudio de las herramientas, sus características y sub-características a implementarse para la creación del aplicativo, realizando una breve comparación de ventajas y desventajas. Además, un breve manual técnico del uso de Electron framework.

En el capitulo 3: Se presenta todo el proceso de desarrollo del sistema utilizando la metodología XP. Además, se realiza la evaluación de seguridad del aplicativo en base a la ISO/IEC 25010.

Al final del documento se presentan las conclusiones y recomendaciones que se basan en el estudio del framework Electron y el desarrollo del aplicativo para la JCPD-Otavalo.

Al final del documento se presentan las conclusiones y recomendaciones que se basan en el estudio del framework Electron y el desarrollo del aplicativo para la JCPD-Otavalo.

PALABRAS CLAVES: Electron, XP, ISO/IEC 25010, aplicación de escritorio multiplataforma.

Abstract

The Municipal Autonomous Decentralized Government of Otavalo with its department “Cantonal Board for the Protection of the Rights of Children and Adolescents (JCPD)” has been carrying out the registration and control of complaints, which does not allow adequate control of the registration of complaints or cases, avocatories, hearings and resolutions, the system to be developed will fully comply with a large part of the process of keeping track and monitoring of the complaints until the end of their process when they have a resolution.

Electron is a powerful multiplatform desktop application development framework, this allows us to develop applications in a short time since it is possible to compile these on different platforms (Microsoft, MacOS, Linux) using the same source code.

In the introduction, the general and specific objectives are stated; the problems that have generated the lack of complaint management of the JCPD-Otavalo are detailed and a solution is proposed.

In Chapter 1: The theory from where the JCPD-Otavalo comes from and the specification based on the laws of its process. In addition, a brief summary of the tools to be used for system validation.

In Chapter 2: The study of the tools, their characteristics and sub-characteristics to be implemented for the creation of the application is described, making a brief comparison of advantages and disadvantages. In addition, a brief technical manual on the use of the Electron framework.

In Chapter 3: The entire system development process is presented using the XP methodology. In addition, the safety assessment of the application is carried out based on ISO / IEC 25010.

At the end of the document, the conclusions and recommendations that are based on the study of the Electron framework and the development of the application for the JCPD-Otavalo are presented.

KEY WORDS: Electron, XP, ISO / IEC 25010, cross-platform desktop application.

INTRODUCCIÓN

Antecedentes

Hoy en día es posible utilizar tecnologías web en distintos ámbitos. Existen múltiples frameworks[i] que nos facilitan desarrollar aplicaciones móviles o aplicaciones de escritorio, como veremos a continuación. Esto supone un ahorro de costes cuando el equipo tiene experiencia con estas tecnologías, ya que se utilizan los mismos lenguajes y el cambio de contexto es menor,(Morante Luis, 2017a).

Es así como el framework Electron desarrollado por GitHub y anteriormente conocido como atom-shell[ii] que se lanzó a la web el 17 de abril del 2015.(Morante Luis, 2017b) , posee la ventaja de incorporar las últimas tecnologías en lenguajes de programación para la web como: HTML, Css y JavaScript con Node.js pero centrándose en aplicaciones de escritorio en lugar de servidores web. Adicionalmente proporciona acceso a APIs nativas.

No ha sido el primer framework en aparecer (ya existían alternativas como Node-Webkit, AppJS) pero ha sido adoptado rápidamente por distintas compañías como Slack, Mapbox e incluso Microsoft, para realizar su editor de código gratuito Visual Studio Code.(Bethencourt Michael, 2015)

El JCPD (Junta Cantonal de Protección de Derechos de la Niñez y Adolescencia) del GAD Municipal del Cantón Otavalo creado el primero de junio del 2001.(OTAVALO, 2014), el Concejo Municipal de Otavalo expidió la ordenanza de constitución y funcionamiento del Concejo Cantonal de la Niñez y Adolescencia con la normativa legal vigente, la misma que ha cambiado y por lo tanto se debe sujetar a la legislación actual, esta tiene como misión llevar a cabo que se cumplan los derechos, debido a los diferentes casos que se tiene de abuso y maltrato infantil.

Situación Actual

Actualmente la JCPD del GAD Municipal de Otavalo viene trabajando en sus procesos de forma manual, siendo este quien lleva a cabo el registro y seguimiento de diferentes denuncias que se realizan diariamente, debido a esto su proceso es mucho más lento en el momento de gestionar cualquier tipo de denuncia, lo cual permite prevalecer los derechos de la niñez y adolescencia de la población Otavaleña.

En la actualidad gran parte de las instituciones públicas o privadas para su mejora, recurren al desarrollo de aplicaciones web o de escritorio. Un número elevado de estas aplicaciones son diseñadas para distintos sistemas operativos como: MacOs, Microsoft y GNU/Linux dependiendo del uso respectivo que se tenga en cada entidad, lo que la implementación de una aplicación multiplataforma para un programador implica crear

distintas aplicaciones nativas para cada sistema operativo con el fin de cumplir un mismo servicio.

Debido a esto, se pretende desarrollar un sistema utilizando el framework Electron para la JCPD del GAD Municipal de Otavalo que permitirá al departamento que lleva a cabo el proceso de denuncias tener todos los registros ordenados correctamente en una base de datos, con esta información las autoridades podrán tomar las medidas necesarias de acuerdo con las denuncias en seguimiento u almacenadas.

La calidad del producto software se puede interpretar como el grado en que dicho producto satisface los requisitos de sus usuarios aportando de esta manera un valor. Son precisamente estos requisitos (funcionalidad, rendimiento, seguridad y mantenibilidad) los que se encuentran representados en el modelo de calidad, el cual categoriza la calidad del producto en características y sub-características.(ISO 25000, 2015a)

Prospectiva

Una vez realizado el Aplicativo JCPD, para el GAD Municipal de Otavalo se encontrará en una mejor posición ya que contará con procesos y mejores servicios con el uso de tecnología informática de punta, al brindar a la población calidad de servicio y mejores prestaciones. La interfaz desarrollada será un baluarte debido a que los servidores públicos podrán navegar con mayor agilidad, registrar y manejar el seguimiento de cada caso de denuncia agregado.

Con el estudio a realizarse se generará la documentación necesaria para que estudiantes y profesionales interesados en el estudio del Framework Electron multiplataforma puedan incrementar sus conocimientos sobre esta herramienta de desarrollo. Por lo tanto, con el presente proyecto se busca proponer el uso de herramientas de desarrollo multiplataforma que garanticen dichas necesidades como productividad y accesibilidad en las instituciones.

Planteamiento del Problema

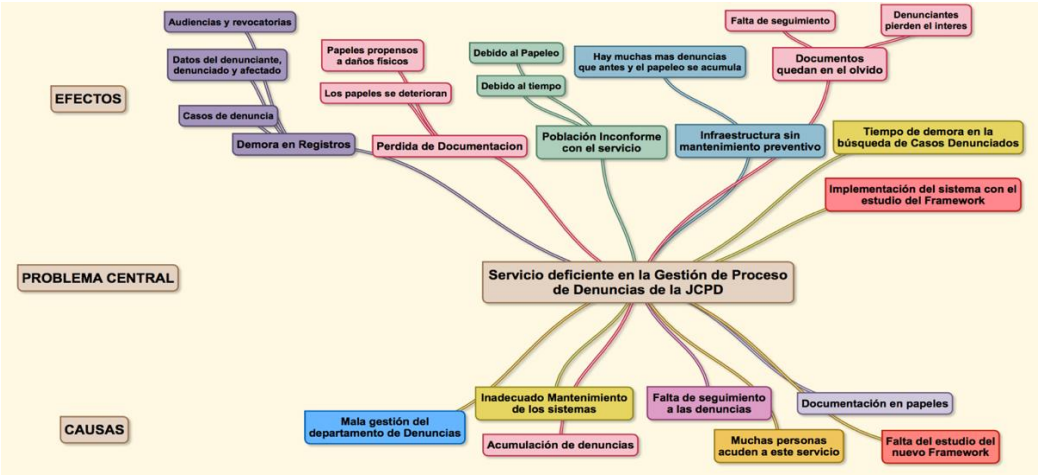


Fig. 1 Árbol de problemas
Fuente: propia

¿Cómo puede ayudar la integración de las herramientas que van a ser objeto de estudio a la automatización del proceso de gestión de denuncias de la población en el JCPD del GAD Municipal del cantón Otavalo?

Objetivos

Objetivo General

Desarrollar una Aplicación, para la Gestión de denuncias de la Junta Cantonal de Protección de Derechos (JCDP) del GAD municipal del Cantón Otavalo mediante el framework Electron.

Objetivos Específicos

- Estudiar el Framework multiplataforma Electron y realizar la documentación técnica.
- Automatizar los procesos que se llevan a cabo como: registro y seguimiento de denuncias en la Junta Cantonal de Protección de Derechos.
- Desarrollar un sistema multiplataforma para la Gestión de Denuncias de la Junta Cantonal de Protección de derechos de la niñez y adolescencia.
- Utilizar la metodología de desarrollo XP, y evaluar la seguridad del sistema mediante la norma de Calidad de Software ISO 25010.

Alcance

El presente estudio proporcionará a los programadores interesados en la creación de aplicaciones de escrito multiplataforma con el Framework Electron una guía que facilite el aprendizaje y desarrollo.

Además, permitirá a la Junta Cantonal de Protección de Derechos del GAD municipal Otavalo llevar el registro y seguimiento del proceso de denuncias que se llevan a cabo diariamente mediante la aplicación desarrollada en base al framework estudiado.

Una vez ya construido el Aplicativo se procederá a evaluar puntos estratégicos del sistema para poder demostrar que es un software de calidad mediante la ISO 25010.

Arquitectura:

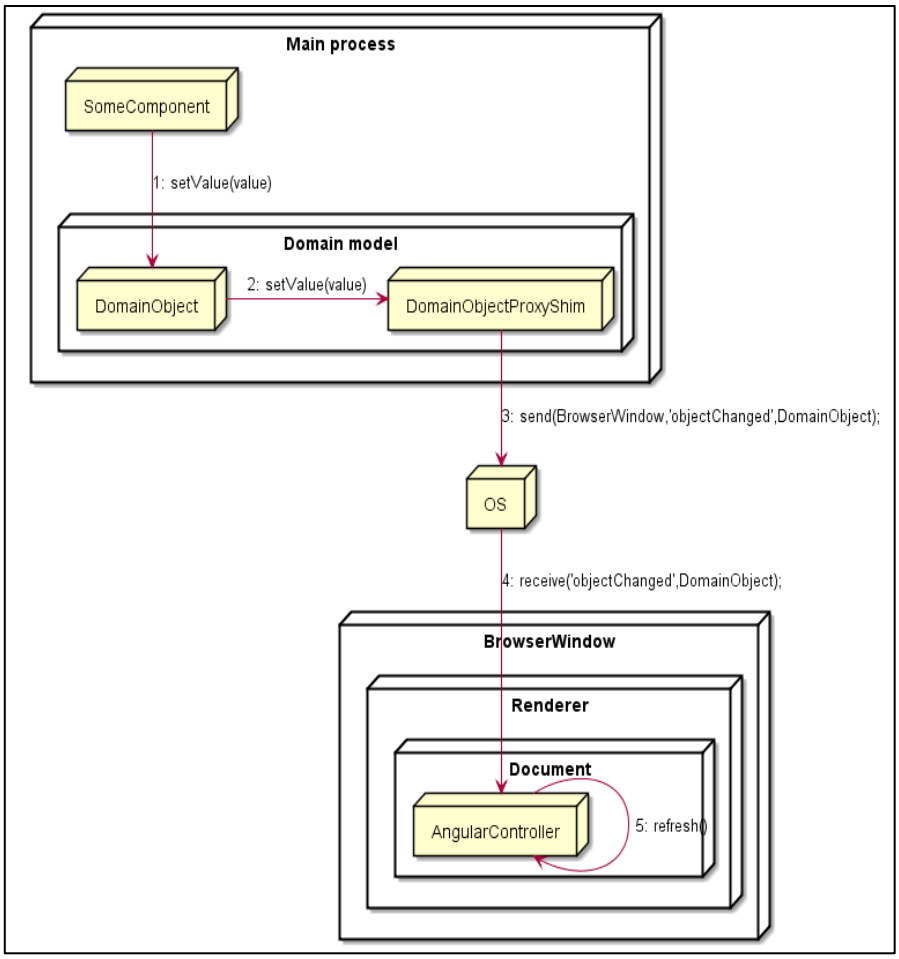


Fig. 2 Arquitectura
Fuente: (Imgur, 2015)

Se detalla cómo va a estar conformado el Sistema de Gestión de denuncias de la población del GAD Municipal De Otavalo, en donde se obtendrá información relevante.

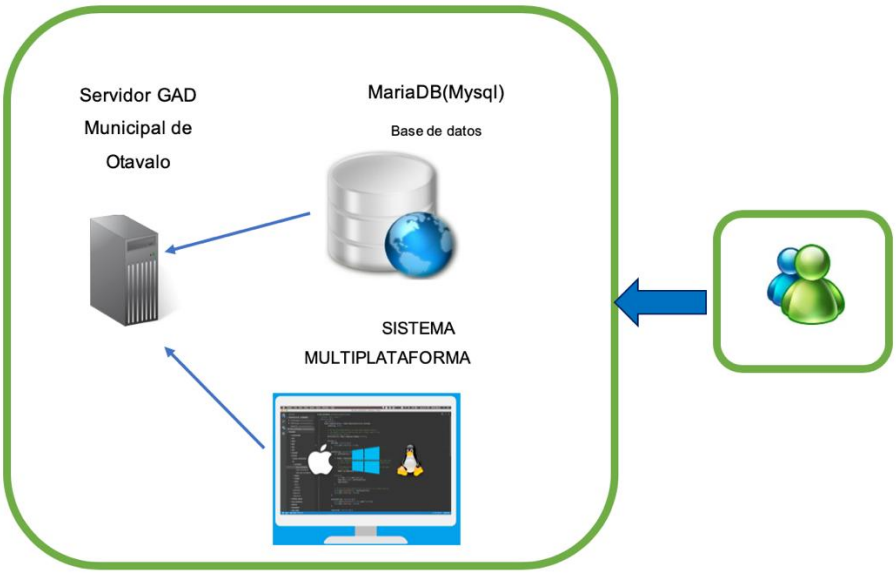


Fig. 3: Diagrama de Despliegue
Fuente: Propia

Metodología XP es un proceso de la metodología ágil que se usa para minimizar los riesgos durante la realización de un proyecto. (Fojtik, 2011), pero de manera colaborativa, también agregamos la Norma ISO 25010 de calidad del producto de software aplicando puntos estratégicos en el sistema para la seguridad de este.

Justificación

Impacto Tecnológico

Mediante este estudio permitirá a los desarrolladores escoger el framework Electron para crear aplicaciones multiplataforma, esta herramienta que está dentro de la nueva tecnología, la misma que posibilitará el ahorro en tiempo y recursos al disponer en forma inmediata, con precisión y garantía en la utilización de este medio tecnológico por parte de los usuarios.

En un futuro se busca que el desarrollo de aplicaciones de escritorio multiplataforma se convierta en una forma de programación rápida, accesible y a bajo costo, donde las empresas no tendrán que invertir grandes cantidades de recursos con el fin de reescribir sus aplicaciones para cada sistema operativo en la que se desea desplegar.

Impacto Social

Mediante el desarrollo de este aplicativo damos aporte social a la población Otavaleña, permitiendo que tengan un servicio digno y ágil al momento de realizar denuncias, cabe recalcar que también se ve enfocado en derechos de personas y es por ello por lo que se debe observar el correcto seguimiento de denuncias, sin perder ningún proceso y que no se dé tramite a cualquier tipo de denuncia.

El presente estudio brindará información para la comunidad de desarrolladores de habla hispana interesada en Electron Framework, ya que en la actualidad la documentación en español a nivel mundial abarca el 3% según el número de búsquedas obtenidas de Google Search con el patrón de búsqueda: “Electron Framework”.

Contexto

Después de haber revisado temas referentes en la biblioteca de la UTN el tema propuesto no se ha tratado.

Pero, se ha revisado en Internet y existe un proyecto realizado el año 2015 por los autores Vicente Alejandro Retamal, Aravena Gustavo Lisandro Saavedra Saavedra, para un optar al título profesional de Ingenieros Ejecución Informática, en la Pontificia Universidad Católica de VALPARAISO con el tema “Venta Alimenticia Personalizada”,(Vicente & Gustavo, 2016), Esta investigación utiliza para el desarrollo del aplicativo con el framework Electron, por lo tanto, no tiene una relación con el tema propuesto.

CAPÍTULO I
Marco Teórico

1.1. Junta Cantonal de protección de derechos Gobierno Autónomo Descentralizado Municipal de la Ciudad de Otavalo.

La provincia de Imbabura se encuentra ubicada en la Zona 1 del Ecuador, compuesta por seis cantones, uno de ellos el cantón Otavalo, donde se encuentra un organismo municipal, como la Junta Cantonal de Protección de los Derechos de la Niñez y la Adolescencia (JCPD),quien cuenta con autonomía administrativa y funcional, cuya misión es dar protección a la niñez y adolescencia del Cantón, en los casos de amenazas y/o violaciones de sus derechos, cumpliendo con el mandato constitucional que garantiza los Derechos de la Niñez y Adolescencia.

El Consejo Cantonal de Protección de Derechos, dando cumplimiento a la Normativa Legal establecida en la Constitución de la República y el marco legal vigente, presenta el Plan Estratégico Institucional 2015 – 2019, en base a los objetivos y lineamientos del GAD Municipal del Cantón Otavalo; este plan incorpora los criterios técnicos de la planificación estratégica y se fundamenta en la Normativa Legal vigente para los Consejos Cantonales de Protección de Derechos, especialmente en los establecido en la Constitución de la República, la Ley Orgánica de los Consejos de la Igualdad el Código Orgánico de Organización Territorial, Autonomías y Descentralización (COOTAD). (OTAVALO, 2019).

1.1.1. Misión

El Consejo Cantonal de protección de Derechos de Otavalo es el organismo encargado de ejercer las atribuciones de formulación, transversalización[iii], observancia, seguimiento y evaluación de políticas públicas para la igualdad en las temáticas intergeneracionales, de género, discapacidad, movilidad humana e intercultural garantizando los derechos de la ciudadanía del Cantón Otavalo. (OTAVALO, 2019)

1.1.2. Visión

El Consejo Cantonal de Protección de Derechos se consolida como un organismo que exige, protege y garantice el ejercicio pleno de los derechos de todas las personas y los grupos de atención prioritaria del Cantón Otavalo, contemplados en la Constitución de la República del Ecuador, los instrumentos internacionales de Derechos Humanos y la ley Orgánica de los Consejos Nacionales para la Igualdad. (OTAVALO, 2019)

La Junta Cantonal de protección de derechos lleva como registro casos sobre la vulneración de derechos de niños, niñas o adolescentes del cantón y a nivel de parroquias aledañas, en base a denuncias realizadas tenemos resultados muy altos como podemos observar en la Fig.4

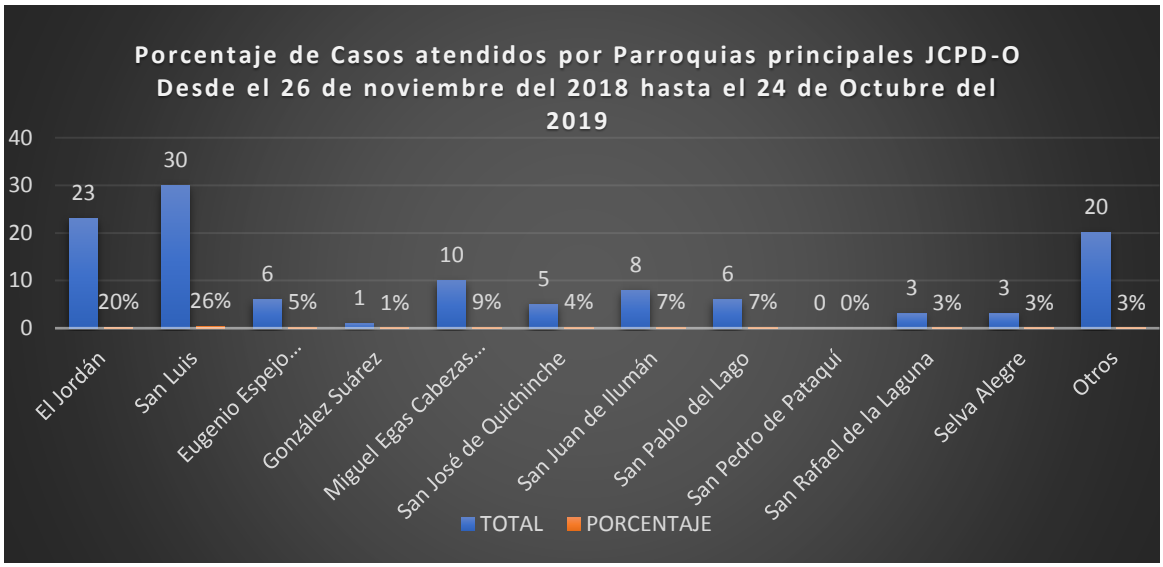


Fig. 4: Casos atendidos en la JCPD-OTAVALO
Fuente: Propia

De los datos obtenidos podemos evidenciar que a nivel cantonal la Junta de Protección de Derechos de la Niñez y Adolescencia en el transcurso de los meses de noviembre de 2018 a octubre de 2019, han atendido 138 casos de diferente vulneración a derechos de niñas, niños y adolescentes. De este total, la mayor parte de quejas con un reporte del 46% de casos registrados, han sido del sector urbano, perteneciente a la parroquia el Jordán y San Luis del cantón Otavalo.

La vulneración de derechos a nivel cantonal, son semejantes a nivel parroquial, incumplimiento con los derechos básicos de un niño, niña o adolescente como es; educación, alimentación, salud, negligencia en el cuidado por parte de sus padres o responsables.

El maltrato físico y psicológico son tipos de vulneraciones comunes que sufren niñas, niños o adolescentes. En la tabla 1.1, podemos observar los casos que se han registrado.

TABLA 1.1: Tipos y Porcentajes de vulneración de derechos

Siglas	Tipo Maltrato	Nro. Casos	Porcentaje
MF	Maltrato Físico	8	6%
MP	Maltrato Psicológico	11	8%
MF-MP	Maltrato Físico y Psicológico	12	8%
PDS	Presunto Delito Sexual	10	7%
OC	Otros Casos	12	8%
VD	Violación de Derechos	1	1%
PD	Presunto Delito	4	3%
AD	Amenaza de Derechos	80	55%
	TOTAL, DE CASOS	138	100%

Fuente: Propia

A nivel cantonal y parroquial, es necesario conformar instancias veedoras del cumplimiento de los derechos, no únicamente en estos grupos de edad, sino a nivel general.

Es importante recalcar que desde el año 2003 existe una oficina a nivel cantonal como una instancia municipal, la JCPD (Junta Cantonal de Protección de Derechos) del cantón Otavalo con el papel de exigir cumplimiento de derechos en base a las normas establecidas por la constitución de la asamblea nacional.

1.1.3. Constitución de la Asamblea Nacional del Ecuador

Según (Corte Constitucional, 2020), "La Corte Constitucional es el máximo órgano de control, interpretación constitucional y de administración de justicia en esta materia".

Según el **Art.424** de la Asamblea Constituyente del Ecuador, dice que la constitución es la norma suprema y prevalece sobre cualquier otra del ordenamiento jurídico. Las normas y actos del poder público deberán mantener conformidad con las disposiciones constitucionales; en caso contrario carecerán de eficacia jurídica. (Asamblea Nacional de la Republica del Ecuador, 2008)

La constitución y los tratados internacionales de derechos humanos ratificados por el estado que reconozcan derechos más favorables a los contenidos en la constitución, prevalecerán sobre cualquier otra norma jurídica o acto del poder publico.

1.1.4. Orden jerárquico de aplicación a las normas de la Asamblea constituyente.

Según el **Art.425** el orden establecido es: La constitución; los tratados y convenios internacionales; las leyes ordinarias; las normas regionales y las ordenanzas distritales; los decretos y reglamentos; las ordenanzas; los acuerdos y las resoluciones; y los demás actos y decisiones de los poderes públicos. (Corte Constitucional, 2020)

1.1.5. Sección quinta de la corte constitucional, niñas niños y adolescentes.

En base al **Art. 44**, El estado, la sociedad y la familia promoverán de forma prioritaria el desarrollo integral de las niñas niños y adolescentes, y aseguraran el ejercicio pleno de sus derechos; se entenderá al principio de su interés superior y sus derechos prevalecerán sobre los de las demás personas. (Asamblea Nacional de la Republica del Ecuador, 2008)

Las niñas niños y adolescentes tendrán derecho a su desarrollo integral, entendido como proceso de crecimiento, maduración y despliegue de su intelecto y de sus capacidades, potencialidades y aspiraciones, en un entorno familiar, escolar, social y comunitario de afectividad y seguridad. Este entorno permitirá la satisfacción de sus necesidades sociales, efectivo-emocionales y culturales, con el apoyo de políticas intersectoriales nacionales y locales.

El **Art**.**45,** Las niñas, niños y adolescentes gozarán de los derechos comunes del ser humano, además de los específicos de su edad. El estado reconocerá y garantizará la vida, incluido el cuidado y protección desde la concepción[iv].

1.2. Proceso de Gestión de Denuncias

La gestión de procesos se apoya en la documentación, sin que esto haya sido percibido necesariamente por todos, el eje central de procesos en la actualidad es la tecnología de la información. Garantizando que procesos empresariales cuenten con aplicaciones y datos que requieren para funcionar.

El modelo de negocio de diversos sectores comerciales puede ser cambiante, por ello es preciso modificar de manera constante los procesos existentes, mejorándolos o creando nuevos caso similar sucede en la documentación y en las aplicaciones de TI existentes.(D.Márquez Fernández, C.Foronda Robles, 2015)

1.2.1. Procesos de denuncias Junta cantonal de protección de derechos

Al ser los procesos necesarios para todas las actividades desarrolladas en nuestro entorno es importante que en la Junta Cantonal de Protección de Derechos conozca de manera oportuna las situaciones o casos respecto a los procedimientos administrativos de protección de los derechos, una escucha activa al menor y la disposición de medidas de protección que aseguren: la finalización de la situación de vulneración, respeto permanente a todos los derechos del menor.

Según el código orgánico de la niñez y adolescencia (CONA, 2017), Articulo 237. “El procedimiento administrativo de protección de derechos puede iniciarse de oficio o mediante una denuncia verbal o escrita en la que se señalará”.

1. El organismo ante el cual comparece. La junta cantonal o lugar judicial el cual asiste a la denuncia.
2. Los nombres, apellidos, edad y domicilio del denunciante y la calidad en la que comparece, obtener la mayor información necesaria.
3. La identificación más detallada posible del niño, niña o adolescente afectado.
4. La identificación más detallada posible de la persona o entidad denunciada.
5. Las circunstancias del hecho denunciado, con indicación del derecho afectado o de la irregularidad imputada.

Dentro de las 48 horas de tener conocimiento del hecho o haber recibido la denuncia, el organismo de la JCPD, procederá avocar conocimiento y señalará el día y la hora para la audiencia de contestación. La citación para la audiencia se practicará personalmente o una boleta dejada en el domicilio del citado o denunciado, en día y horas hábiles.

Según el articulo 238, (CONA, 2017) procedemos a la **Audiencia**. Aquí es donde se escucharán los alegatos verbales de las partes, comenzando por el denunciante, una vez concluido se procederá a escuchar reservadamente al adolescente, niña o niño que estén en condiciones de expresar su opinión.

A continuación, el organismo sustanciador procurará la conciliación de las partes, si la naturaleza del asunto lo permite, de conformidad con la ley. Así mismo, puede remitir el caso a un centro especializado de mediación.

Articulo 239. Audiencia de prueba. - Las partes rendirán todas sus pruebas en la misma audiencia, luego de lo cual podrán exponer verbalmente sus alegatos, comenzando por la parte denunciante. Si el organismo sustanciador lo estima necesario por la extensión de las pruebas, podrá establecer un receso de hasta tres días hábiles. (CONA, 2017)

Articulo 240. Resolución. - El organismo sustanciador pronunciará su resolución definitiva en la misma audiencia o, a más tardar, dentro de los dos días hábiles siguientes.

Los requerimientos de las acciones de protección si son urgentes, deberán cumplirse de inmediato o en su defecto dentro del plazo de cinco días contados desde la notificación de la resolución correspondiente, la misma que podrá hacerse en la misma audiencia.

En caso de incumplimiento del requerimiento, el denunciante o la Junta Cantonal de Protección recurrirán al Juez de la Niñez y Adolescencia para la aplicación de las sanciones por violación a los derechos. Para este efecto se observará el trámite correspondiente de la acción de amparo constitucional. (CONA, 2017).

Articulo 243.- Duración máxima del procedimiento administrativo**.** En ningún caso el procedimiento sustanciado ante el organismo administrativo podrá durar más de treinta días hábiles.

1.2.2. Diagrama de flujo del proceso de denuncias

En base a toda la información, recolectada y aclarando con actuales miembros de la JCPD-Otavalo tenemos el proceso, diseñado en un diagrama de flujo Fig.5

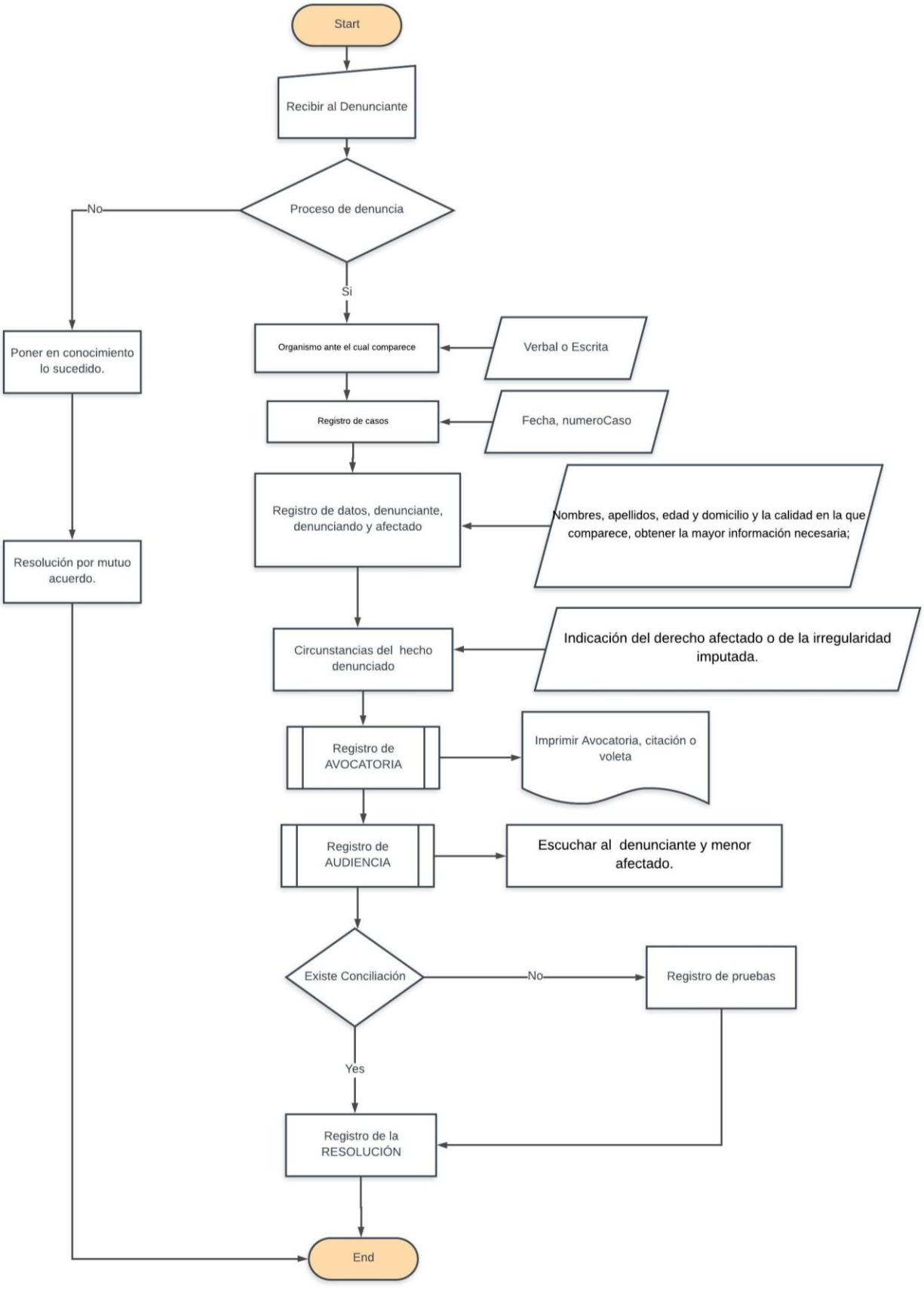


Fig. 5 : Diagrama de flujo del proceso de la JCPD
Fuente: Propia

1.3. Metodología de desarrollo XP

Según a (Fojtik, 2011) en un proceso de software existen numerosas propuestas metodológicas que inciden en distintas dimensiones del transcurso de desarrollo. Por una parte, tenemos aquellas propuestas más tradicionales que se centrandas especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, las herramientas y notaciones que se usarán.

Programación extrema se basa en la simplicidad, la comunicación y la reutilización continua de código. Esta tiene como objetivo principal satisfacer a los clientes. Trata de dar al cliente el Software o aplicación que necesita y cuando lo necesita. Como segundo objetivo, trata de potenciar el trabajo al máximo ya sea en grupo o individual. Tanto los jefes de proyecto, los clientes y desarrolladores, son involucrados en el desarrollo del software(Fojtik, 2011).

La metodología XP define cuatro variables para cualquier proyecto de software: costo, tiempo, calidad y alcance las cuales son enfatizadas en el proceso que se va a llevar según el proyecto.

1.3.1. Características Fundamentales

Las características fundamentales del método las podemos observar en la TABLA 1.2

TABLA 1.2: Características Fundamentales de la metodología XP

Características	Descripción
Desarrollo iterativo e incremental	Ciclos cortos que generan mejoras constantemente.
Pruebas unitarias continuas	Se realizan pruebas frecuentemente para comprobar la funcionalidad del código.
Programación en parejas	Es recomendable que el desarrollo se lo haga en parejas, así el código es revisado y discutido mientras se lo escribe.
Frecuente integración	El equipo de programación debe integrarse con el cliente o usuario. Se recomienda que un representante del cliente trabaje en conjunto con el equipo de desarrollo.
Corrección de errores	Todos los errores deben ser corregidos antes de añadir nueva funcionalidad. Hacer entregas frecuentes.
Refactorización del código	Se refactoriza el código reescribiendo ciertas partes para aumentar su legibilidad sin afectar su funcionalidad, para esto se usan las pruebas unitarias, confirmando así su correcto funcionamiento.
Propiedad del código compartida	Se promueve que todo el equipo de trabajo pueda corregir y extender cualquier parte del proyecto

Fuente: (Laínez, 2014)

1.3.2. Roles

En la Tabla 1.3, encontraremos los roles que tenemos en base a diferentes fuentes de información. Los roles que se pueden desempeñar con respecto a la metodología XP.

TABLA 1.3: Roles en la Metodología XP

Roles	Características
Programador	Es quien se dedica a escribir las diferentes pruebas unitarias y produce el código del sistema
Cliente	Escribe las historias de los usuarios y realiza las diferentes pruebas funcionales para así validar su implementación.
Encargado de pruebas (Tester)	Se encarga de ayudar al cliente a escribir las pruebas funcionales.
Encargado de seguimiento (Tracker)	Es quien realiza el correspondiente seguimiento y proporciona la realimentación al equipo en el proceso XP. Determina los cambios necesarios que se deben realizar ya sea en el instante o hacia el futuro.
Entrenador (Coach)	Responsable del proceso global, es quien tiene un conocimiento global sobre la metodología XP.
Consultor	Miembro externo que sirve para resolver algún problema específico, con un conocimiento diferente a los demás miembros del equipo.
Gestor (Big boss)	Es el vínculo entre clientes y programadores, en conjunto se ayudan a que el equipo trabaje efectivamente creando las condiciones adecuadas.

Fuente: Propia

1.3.3. Proceso XP.

Según (Penadés & Letelier Torres, 2006), un proyecto XP tiene éxito cuando el cliente selecciona el valor de negocio a implementar, basado en la habilidad del equipo para medir la funcionalidad que puede entregar a través del tiempo. El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

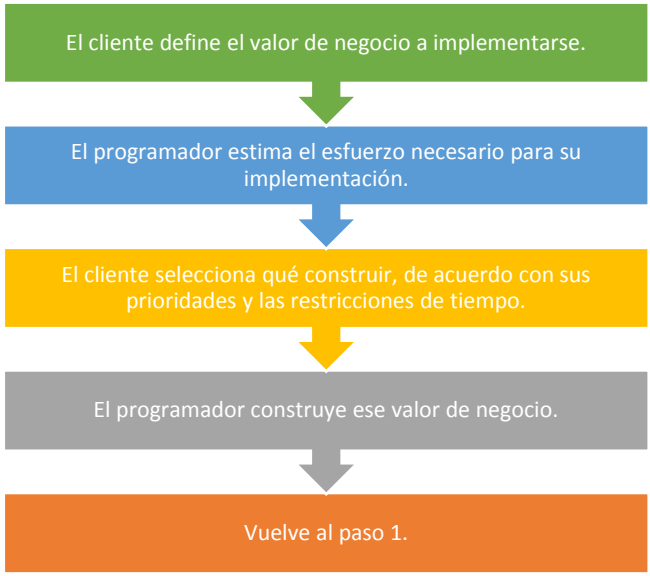


Fig. 6 : Ciclo de desarrollo
Fuente: Propia

En todas estas interacciones, el cliente y el programador aprenden. Es decir que el cliente aprende a que no debe presionar a realizar mas trabajo del estimado hacia el programador, ya que el software perderá calidad o no se cumplirán los plazos establecidos, y a la vez el cliente también tiene la obligación de manejar el ámbito de entrega del producto.

(Penadés & Letelier Torres, 2006) menciona que el ciclo de vida de un proyecto XP incluye, al igual que las otras metodologías, entender lo que el cliente necesita, estimar el esfuerzo, crear la solución y entregar el producto final al cliente. Sin embargo, la metodología Xp propone un ciclo de vida dinámico donde los clientes admiten en muchos casos que no son capaces de especificar sus requerimientos al principio del proyecto.

Si bien el ciclo de vida de un proyecto XP es muy dinámico, se puede separar en fases(Joskowicz, 2008).

- **Fase de exploración**

Esta es la fase en donde los clientes plasman a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. También el desarrollador o desarrolladores se familiarizan con las herramientas, tecnologías y practicas que se utilizarán en el proyecto(Letelier & Penadés, 2017).

- **Fase de planificación**

Se considera por ser una fase corta, en donde el cliente, gerente y desarrolladores acuerdan el orden en el cual se debe implementar las historias de usuario, y en referencia a estas serán las entregas. Básicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un plan de entregas(Letelier & Penadés, 2017).

Según (Penadés & Letelier Torres, 2006) la planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. Al planificar por tiempo, se multiplica el número de iteraciones por la velocidad del proyecto, determinándose cuántos puntos se pueden completar.

- **Iteraciones**

En esta fase se reflejan todas las iteraciones que se han realizado sobre el sistema para luego ser entregado. Hay varias iteraciones las cuales pueden ser permanentes como podría ser una arquitectura formal que sea establecida durante el resto del proyecto. Pero esto decide el cliente dando a conocer q historias se implementarán en cada iteración, siendo esto para maximizar el valor del negocio, en fin, cuando todas las iteraciones del sistema hayan concluido estará listo para entrar en producción.(Letelier & Penadés, 2017)

Podemos ver en la Fig.7 todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores (Penadés & Letelier Torres, 2006). Los principales elementos que se deben tomar en cuenta durante la elaboración del plan de la iteración son:

- Historias del usuario no abordadas

- Velocidad del proyecto
- Pruebas de iteración no superadas en la iteración anterior
- Tareas no terminadas en la iteración anterior

Todo el trabajo que expresa cada iteración esta basado en tareas de programación, cada una de las iteraciones es asignada a un programador como responsable, pero a la vez llevadas a cabo en parejas como programadores.

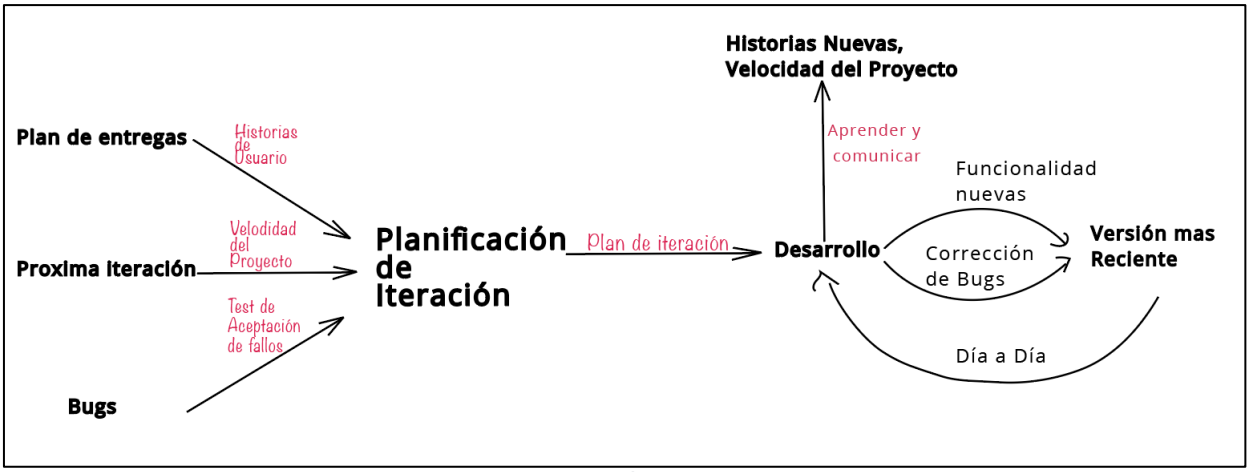


Fig. 7 Iteración
Fuente: Propia

➢ Fase de puesta en producción

Pues bien, al final de cada iteración para dar como finalizado el proyecto se entregan los módulos funcionales y sin ningún error, el cliente debe ser quien decida si el sistema debe ser puesto en producción hasta cuando no se tenga la funcionalidad completa. En esta fase ya no se realizan mas desarrollos funcionales, pero pueden ser necesarias tareas de ajuste(Letelier & Penadés, 2017).

Según (Letelier & Penadés, 2017), la fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase.

➢ Mantenimiento

Una vez que el sistema este en producción, el proyecto en base a Xp debe permanecer en funcionamiento y al mismo tiempo se van realizando nuevas iteraciones. Para llevar a cabo esto se necesitan tareas de soporte para el cliente. Debido a esto la velocidad de desarrollo del sistema puede bajar ya que el proyecto está puesto en producción (Letelier & Penadés, 2017).

➢ Muerte del proyecto

Es aquí donde finalizan las iteraciones, debido a que el cliente ya no tiene mas historias para ser incluidas en el sistema. Esto requiere la satisfacción hacia el cliente, pero por otros

aspectos como es el rendimiento y confiabilidad del sistema. Para finalizar se genera la documentación necesaria del sistema y no se realizan mas cambios en la arquitectura (Letelier & Penadés, 2017).

1.4. Aplicación

Una aplicación es un programa informático diseñado para facilitar a los usuarios la realización de un tipo de trabajo determinado y llega a resultar una solución informática para la automatización de ciertas tareas complicadas para resolver un problema específico. (Johnn Calvopiña, 2012)

Dentro del Gad Municipal Del Cantón Otavalo se utilizan diferentes tipos de aplicativos, como son de tipo web y de escritorio estos están desarrollados en diferentes tipos de lenguajes de programación, estas son las mas utilizadas y de gran ayuda para mejorar la gestión de información.

1.4.1. Aplicación Web

En la Ingeniería de software se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un Servidor Web a través de Internet o de una intranet mediante un navegador. En la Tabla 1.4 podemos identificar las ventajas y desventajas de las aplicaciones web.

TABLA 1.4: Ventajas y desventajas de las aplicaciones web

Ventajas	Desventajas
Son totalmente compatibles con todo tipo de navegadores web actuales que se encuentran en internet.	Requiere casi siempre de una conexión a internet para que este tipo de aplicativos funcione.
Un usuario puede ver su información desde cualquier dispositivo mientras este conectado a la web.	Muchas de las aplicaciones creadas para la web necesitan las extensiones apropiadas y actualizadas para operar.
Son aplicaciones multiplataforma, pero se debe tener una conexión a internet y también un navegador web para que así tenga funcionamiento el aplicativo.	Mayor vulnerabilidad al momento de estar en la Nube.

Fuente: Propia

1.4.2. Aplicaciones de Escritorio

Aplicaciones de escritorio son aquellas que están directamente instalada en un ordenador, con un sistema operativo diferente como es Mac-Os, Windows, Linux o Solaris. El rendimiento de este depende de las configuraciones de hardware como la memoria RAM, Disco duro, tarjeta de video, etc. Se ilustra en la TABLA 1.5 ventajas y desventajas de aplicaciones de escritorio.

TABLA 1.5: Ventajas y Desventajas de Aplicaciones de Escritorio

Ventajas	Desventajas
Rapidez y agilidad si este aplicativo esta instalado en un mismo ordenador donde se ejecuta la aplicación.	Acceso limitado, solo se puede acceder a la aplicación donde ha sido instalado, es decir en el mismo ordenador donde se ejecuta

El acceso a datos locales mucho mas fácil, y mayor aprovechamiento de la CPU hacen que la aplicación solo dependa del ordenador.	Desventaja al momento de hacer una actualización a la aplicación.
Pueden ser mas robustas	Dependencias del ordenador, debemos tener el ordenador en las mejores condiciones.
Se puede hacer cualquier cosa que permita el software (cuestión gráfica, control total de periféricos de entrada)	Problemas de virus

Fuente: (buyto, 2020)

1.5. ISO 25010

La norma (International Organization For Standardization 25010, 2015) define la calidad del producto software y los sistemas de computación de software, con lo cual se va a satisfacer las necesidades de los usuarios y cumplir los objetivos propuestos.

El estándar ISO / IEC-25010: 2015 está diseñado para un mejor ajuste del contexto y está respaldado por requisitos concretos y una metodología que cubre el ciclo de vida de los productos de software en proyectos tanto nuevos como nuevos. Además, la organización tuvo que examinar el estado actual de las capacidades de TI existentes para establecer una línea base de calidad para el desarrollo futuro, y desarrollar una visión compartida y hojas de ruta para la calidad del producto.(Mohagheghi & Aparicio, 2017)

1.5.1. Marco de referencia del modelo de calidad

- **Modelos de calidad**

La calidad de un sistema es el grado en el cual el sistema satisface las necesidades establecidas e implícitas de varias partes interesadas y así proveer un valor, las cuales en algunos casos son subdivididas en sub-características. (Algunas sub-características están divididas otras sub-características). Esta descomposición jerárquica provee un desglose conveniente de la calidad del producto. Sin embargo, el conjunto de sub-características asociadas con una característica han sido seleccionadas para ser representantes de inquietudes típicas sin que sean necesariamente exhaustivas (Sistemas & Software, 2015). En la Fig.8 se ilustra la estructura usada para los modelos de calidad.

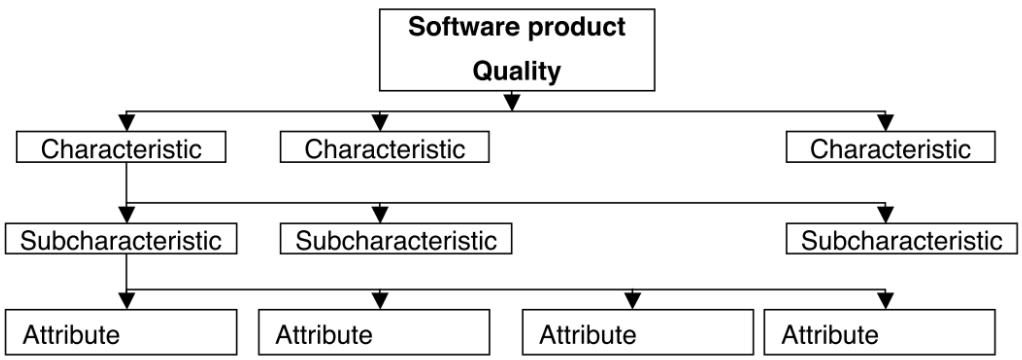


Fig. 8: Estructura usada para los modelos de calidad
Fuente: (International Organization For Standardization 25010, 2015)

- **Modelo de calidad del producto**

El modelo de calidad del producto puede ser aplicado solo a un producto de software, o a un sistema de computación que incluye software, ya que muchas de las sub-características son relevantes tanto para el software como para los sistemas. En la Fig.9 se ilustra el modelo de calidad de producto de software.

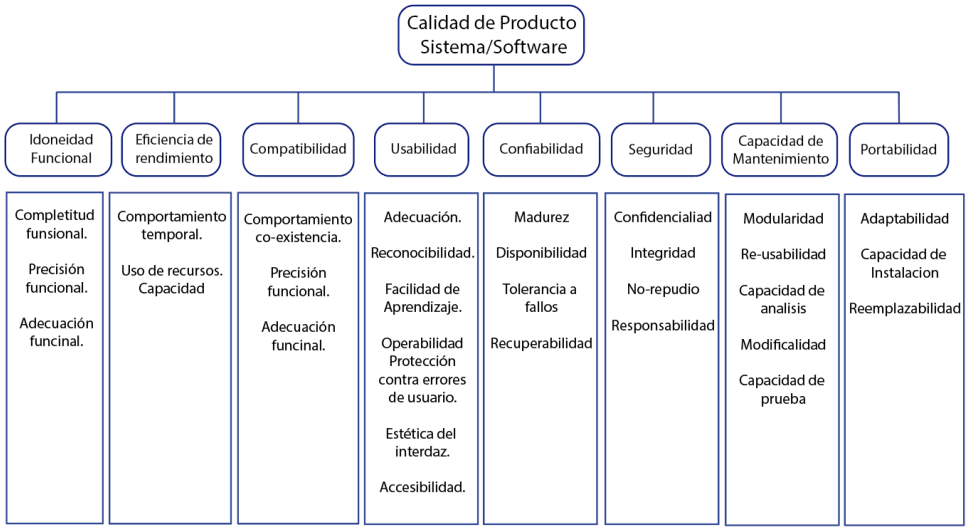


Fig. 9: Modelo de calidad del producto
Fuente: (International Organization For Standardization 25010, 2015)

Determina las diferentes características de la calidad del producto software sabiendo que es un concepto muy amplio compuesto por varias dimensiones, así se pueden evaluar en total son 8 las características que se identifican.

- **Uso de un modelo de calidad**

La calidad de producto y los modelos de calidad en uso son útiles para especificar los requisitos, establecer las medidas y llevar a cabo las evaluaciones de calidad.

Este modelo debería ser adaptado antes de su uso como parte de la descomposición de los requisitos para identificar aquellas características y sub-características que son más importantes y los recursos asignados entre los diferentes tipos de medida dependiendo de las metas de las partes interesadas y los objetivos para el producto.(International Organization For Standardization 25010, 2015)

- **La calidad desde las perspectivas de diferentes partes interesadas.**

Podemos observar en la Fig.10 los diferentes usuarios que interactúan en el sistema a desarrollarse, siendo quienes realicen las suficientes validaciones del sistema y lo hagan verdaderamente funcional.

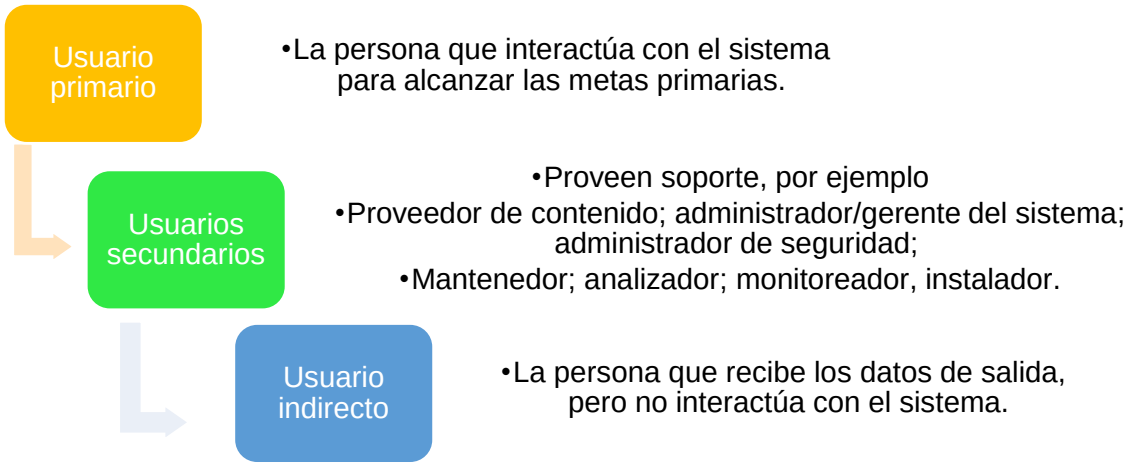


Fig. 10: La calidad desde las perspectivas de diferentes partes interesadas.
Fuente: Propia

En la TABLA 1.6 identificamos el cuestionario para saber la calidad de uso y la calidad de producto en base a las perspectivas de diferentes partes interesadas que mostramos en la Fig.10

TABLA 1.6: Ejemplos de necesidades de usuarios para la calidad de uso y la calidad de producto

Necesidades del usuario	**Usuario primario**	**Usuarios secundarios**		**Usuario indirecto**
		Proveedor de contenido	**Mantenedor**	
	Interacción	**Interacción**	**Mantenimiento o portabilidad**	**Uso de datos de salida**
Efectividad	¿Qué tan efectivo debe ser el usuario al usar el sistema para realizar las tareas?	¿Qué tan efectivo debe ser el proveedor de contenidos al actualizar el sistema?	¿Qué tan efectiva debe ser la persona que mantiene el sistema?	¿Qué tan efectiva debe ser la persona que usa los datos de salida?
Eficiencia	¿Qué tan eficiente debe ser el usuario al usar el sistema para realizar las tareas?	¿Qué tan eficiente debe ser el proveedor de contenidos al actualizar el sistema?	¿Qué tan eficiente debe ser la persona que mantiene el sistema?	¿Qué tan eficiente debe ser la persona que usa los datos de salida?
Satisfacción	¿Qué tan satisfecho debe estar el usuario al usar el sistema para realizar las tareas?	¿Qué tan satisfecho debe estar el proveedor de contenidos al actualizar el sistema?	¿Qué tan satisfecha debe estar la persona que mantiene el sistema?	¿Qué tan satisfecha debe estar la persona que usa los datos de salida?
Libertad de riesgo	¿Qué tan libre de riesgo debe ser para el usuario el usar el sistema para realizar las tareas?	¿Qué tan libre de riesgo debe ser la actualización de contenidos del sistema?	¿Qué tan libre de riesgos debe ser el hacer cambios en el mantenimiento o portabilidad del sistema?	¿Qué tan libre de riesgo ser el uso de datos de salida?
Confiabilidad	¿Qué tan confiable debe ser el sistema cuando el usuario lo usa para realizar su tarea?	¿Qué tan confiable debe ser la actualización del sistema con el nuevo contenido?	¿Qué tan confiable debe ser el mantener o portar el sistema?	¿Qué tan confiable deben ser los datos de salida del sistema?
Seguridad	¿Qué tan seguro debe ser el sistema cuando el usuario lo usa para realizar su tarea?	¿Qué tan seguro debe ser el sistema luego que el proveedor de contenidos lo actualiza?	¿Qué tan seguro debe ser el sistema luego que los cambios de mantenimiento son hechos o cuando es portable?	¿Qué tan seguros deben ser los datos de salida?

Cobertura de contexto	¿Hasta qué grado debe el sistema ser efectivo, eficiente, libre de riesgo y satisfacer todos los contextos previstos y potenciales de uso?	¿Hasta qué grado debe el contenido provisto ser efectivo, eficiente, libre de riesgo y satisfacer todos los contextos previstos y potenciales de uso?	¿Hasta qué grado debe el mantenimiento o portabilidad del sistema ser efectivo, eficiente, libre de riesgo y satisfacer todos los contextos previstos y potenciales de uso?	¿Hasta qué grado deben los datos de salida del sistema ser efectivos, eficientes, libres de riesgo y satisfacer en todos los contextos previstos y potenciales de uso?
Capacidad de aprendizaje	¿Hasta qué grado debe el aprendizaje de uso del sistema ser efectivo, eficiente, libre de riesgo y satisfactorio?	¿Hasta qué grado debe el aprendizaje de provisión de contenidos ser efectivo, eficiente, libre de riesgo y satisfactorio?	¿Hasta qué grado debe el aprendizaje de mantenimiento o portabilidad del sistema debe ser efectivo, eficiente, libre de riesgo y satisfactorio?	¿Hasta qué grado debe el aprendizaje de uso de los datos de salida del sistema ser efectivo, eficiente, libre de riesgo y satisfactorio?
Accesibilidad	¿Hasta qué grado debe el sistema ser efectivo, eficiente, libre de riesgo y satisfactorio para ser usado con personas con discapacidad?	¿Hasta qué grado debe el proveedor de contenidos para el sistema ser efectivo, eficiente, libre de riesgo y satisfactorio para personas con discapacidad?	¿Hasta qué grado debe el mantenimiento o portabilidad del sistema ser efectivo, eficiente, libre de riesgo y satisfactorio para personas con discapacidad?	¿Hasta qué grado debe el uso de datos de salida ser efectivo, eficiente, libre de riesgo y satisfactorio para personas con discapacidad?

Fuente: (International Organization For Standardization 25010, 2015)

1.5.2. Términos y definiciones

- **Modelo de calidad de producto**

El modelo de calidad de producto categoriza las propiedades de calidad de producto en ocho características (idoneidad funcional, confianza, eficiencia de rendimiento, usabilidad, seguridad, compatibilidad, capacidad de mantenimiento y portabilidad). Cada característica está compuesta de un conjunto de sub-características relacionadas.(International Organization For Standardization 25010, 2015). En la TABLA 1.7 observamos los términos y definiciones del modelo de calidad del producto.

TABLA 1.7: Términos y definiciones

1. Idoneidad funcional

- Grado en el cual un producto o sistema provee funciones que satisfacen las necesidades establecidas e implícitas cuando se las usa bajo condiciones específicas

1.1. Integridad funcional

- Grado en el cual el conjunto de funciones cubre todas las tareas y objetivos específicos del usuario

1.2. Exactitud funcional

- Grado de exactitud en el cual un producto o un sistema provee los resultados correctos con el grado necesario de precisión

1.3. Oportunidad funcional

- Grado en el cual las funciones facilitan el cumplimiento de las tareas y objetivos específicos

2. Eficiencia de rendimiento

- El rendimiento relativo a la cantidad de recursos usados bajo las condiciones establecidas

2.1. Comportamiento en el tiempo

- Grado en el cual la respuesta, los tiempos de procesamiento y las tasas de rendimiento de un producto o sistema satisfacen los requisitos al desempeñar sus funciones

2.2. Utilización de recursos

- Grado en el cual las cantidades y tipos de recursos usados por un producto o sistema satisfacen los requisitos al desempeñar sus funciones

2.3. Capacidad

- Grado en el cual los límites máximos de un parámetro de un producto o sistema satisface los requisitos

3. Compatibilidad

- Grado en el cual un producto, sistema o componente pueden intercambiar información con otros productos, sistemas o componentes, y realizar sus funciones requeridas, mientras que comparte el mismo ambiente de hardware o software

3.1. Coexistencia

- Grado en el cual un producto puede realizar sus funciones requeridas eficientemente mientras comparte un ambiente y recursos comunes con otros productos, sin un impacto perjudicial en ningún otro producto

3.2. Interoperabilidad

- Grado en el cual dos o más sistemas, productos o componentes pueden intercambiar información y usar la información que ha sido intercambiada

4. Usabilidad

- Grado en el cual un producto o un sistema puede ser usado por usuarios específicos para alcanzar metas específicas con efectividad, eficiencia y satisfacción en un contexto específico de uso

4.1 Reconocimiento de oportunidad

- Grado en el cual los usuarios pueden reconocer si un producto o sistema es apropiado para sus necesidades

4.2. Capacidad de aprendizaje

- Grado en el cual un producto o sistema puede ser usado por usuarios específicos para alcanzar las metas especificadas y así aprender a usar el producto o sistema con efectividad, eficiencia, libertad de riesgo y satisfacción en un contexto de uso específico

4.3. Operabilidad

- Grado en el cual un producto o sistema tiene atributos que lo hacen más fácil de operar y controlar

4.4. Protección contra error del usuario

- Grado en el cual un sistema protege a los usuarios de cometer errores

4.5. Estética de la interfaz del usuario

- Grado en el cual una interfaz del usuario permite una interacción agradable y satisfactoria para el usuario

4.6. Accesibilidad

- Grado en el cual un producto o sistema puede ser usado por las personas con el rango más amplio de características y capacidades para alcanzar una meta específica en un contexto de uso específico

5. Confiabilidad

- Grado en el cual un sistema, producto o componente realiza las funciones específicas bajo condiciones específicas por un período de tiempo específico

5.1. Madurez

- Grado en el cual un sistema, producto o componente satisface las necesidades de confiabilidad bajo operación normal

5.2. Disponibilidad

- Grado en el cual un sistema, producto o componente es operativo y accesible cuando se lo requiere para el uso

5.3. Tolerancia del error

- grado en el cual un sistema, producto o componente opera como se pretende a pesar de la presencia de errores del hardware o del software

5.4. Capacidad de recuperación

- Grado en el cual, en el caso de una interrupción o falla, un producto o sistema puede recuperar los datos directamente afectados y re-establecer el estado deseado del sistema

6. Seguridad

- Grado en el cual un producto o sistema protege la información y los datos de tal manera que las personas u otros productos o sistemas tienen el grado de acceso a los datos apropiado a sus tipos y niveles de autorización

6.1. Confidencialidad

- Grado en el cual un producto o sistema asegura que los datos son accesibles solo para aquellos autorizados a tener acceso

6.2. Integridad

- Grado en el cual el sistema, producto o componente previene acceso no autorizado a, o modificación de los programas de computación o de los datos

6.3. No repudio

- Grado en el cual se puede probar que las acciones o hechos sucedieron, de tal manera que los eventos o acciones que no pueden ser repudiadas más tarde

6.4. Responsabilidad

- Grado en el que las acciones de una entidad pueden ser rastreados de forma exclusiva a la entidad

6.5. Autenticidad

- Grado en el que la identidad de un objeto o recurso se puede probar en caso de reclamo

7. Capacidad de mantenimiento

- Grado de efectividad y eficiencia con el cual un producto o sistema puede ser modificado por los mantenedores destinados

7.1. Modularidad

- Grado en el cual un sistema o programa de computación está compuesto de componentes discretos de tal manera que un cambio a un componente tiene un impacto mínimo en otros componentes

7.2. Reusabilidad

- Grado en el cual un activo puede ser usado en más de un sistema, o ser armado en otros activos

7.3. capacidad de análisis

- Grado de efectividad y eficiencia con el cual es posible evaluar el impacto en un producto o sistema por un cambio previsto sobre uno o más de sus partes, o diagnosticar un producto por las deficiencias o causas de falla, o identificar las partes a ser modificadas

7.4. Capacidad de modificación

- Grado en el cual un producto o sistema puede ser efectivamente y eficientemente modificado sin introducir defectos o sin degradar la calidad de producto existente

7.5. Capacidad de prueba

- Grado de efectividad y eficiencia con el cual los criterios de prueba pueden ser establecidos para un sistema, producto o componente y las pruebas pueden ser realizadas para determinar si aquellos criterios han sido satisfechos

8. Portabilidad

- Grado de efectividad y eficiencia con el cual un sistema, producto o componente puede ser transferido de un hardware, software u otro ambiente operativo o de uso a otro

8.1. Adaptabilidad

- Grado en cual un producto o sistema puede efectivamente y eficientemente ser adaptado en un hardware, software, u otros ambientes operativos o de uso

8.2. Capacidad de instalación

- Grado de efectividad y eficiencia con el cual un producto o sistema puede ser instalado exitosamente y desinstalado en un ambiente específico

8.3. Capacidad de reemplazo

- Grado en el cual un producto puede reemplazar otro producto de software especificado para el mismo propósito en el mismo ambiente

Fuente: Propia

CAPÍTULO II
2. Estudio Investigativo

2.1. Arquitectura de la Aplicación

Para crear un software de calidad se ha implementado la arquitectura MVC (Modelo, Vista, Controlador), con el objetivo de separar la lógica de programación y tener un código mucho mas ordenado, a continuación, podremos apreciar brevemente cada capa del sistema.

Patrón de diseño en software que permite la separación de obligaciones de cada pieza de código, en el aplicativo web. Es decir, enfatiza la separación de la lógica de programación y lo que se muestra en pantalla. En la Fig. 11 ilustramos la arquitectura MVC.

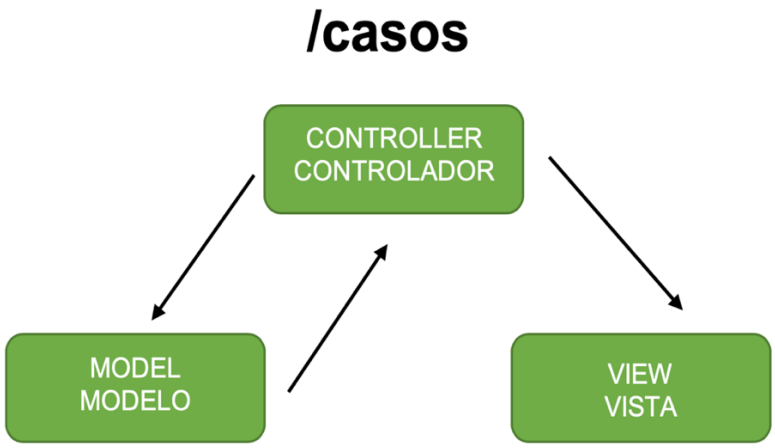


Fig. 11 Arquitectura MVC
Fuente: Propia

2.2. Modelo, vista y controlador

2.2.1. Modelo

Es la capa que interactúa con la base de datos. Encargado de los datos y de la lógica para traer los datos.

2.2.2. Vista

El paso en donde se muestra en pantalla es decir los datos que el modelo nos envía, se encarga de todo lo que se ve (HTML).

Permite mostrar la parte visual (HTML) en una aplicación express[v], debido a que el modelo retorna un objeto o arreglo de datos, un template engine[vi] permitirá acceder a los resultados de una consulta y mostrarlos en pantalla. Si nuestro conocimiento es un poco mas avanzado podemos reemplazar la vista en los templates y podemos usar los frameworks Electron, Angular, React, o Vue para así tener unas vistas mas robustas.

2.2.3. Controlador

Es el que se comunica entre el modelo y la vista, antes de que el modelo consulte a la base de datos es el encargado de mandarlo a llamar, y también una vez que el modelo tiene los resultados de la consulta es el que se encarga de pasarlos a la vista.

2.2.4. Rutas

Encargado de registrar todas las url's que la aplicación soporta. Mediante url, se puede acceder desde cualquier vista mientras se tenga acceso al servidor.

2.3. Git

Git es el sistema de control de versiones más utilizado en la actualidad, popularizado en gran medida gracias al servicio de GitHub, el más popular de los hosting para repositorios Git ("Manual de Git," n.d.).

Git es una de las herramientas fundamentales para cualquier equipo de desarrollo. "Ningún grupo de programadores debería desarrollar software sin usar un sistema de control de versiones". Existen muchos en el mercado, pero Git se destaca por ser un sistema de control de versiones distribuido que simplifica bastante el trabajo en cualquier tipo de entorno. No queremos decir que Git sea simple, sino que hace fáciles algunas tareas de control de versiones donde otros sistemas resultan bastante más complejos ("Manual de Git," n.d.).

Git nos facilita llevar un control de los cambios de cualquier pieza de software, mediante el control de todos los archivos que forman parte de un proyecto. De este modo puedes saber qué estados han tenido tus archivos a lo largo del tiempo y permite que los componentes del equipo de desarrollo sincronicen sus cambios los unos con los otros.

2.3.1. Bitbucket

El servicio también es muy similar a GitHub y refleja la mayoría de sus características con ligeras diferencias. Bitbucket está mejor orientado a los equipos de desarrollo profesional, ya que proporciona grandes beneficios para ellos, como repositorios privados gratuitos, integración con Jira, revisión de código avanzado y CI/CD (Integración Continua/Despliegue continuo) integrado. Al mismo tiempo, con el crecimiento del equipo, Bitbucket ofrece condiciones de precios más adecuadas comparadas con GitHub y GitLab. Bitbucket también proporciona una modelo de implementación flexible para equipos.("GitHub vs. GitLab vs. Bitbucket: ¿Que repositorio elegir?," n.d.)

- **Beneficios**

Se muestra en la Fig. 12 la descripción de los beneficios que tiene la herramienta bitbuket.

Fig. 12 Beneficios de Bitbucket
Fuente: Propia

Desventajas de Bitbuket:

- No es de código abierto, pero admite proyectos de código abierto.
- Bitbucket brinda un servicio igual a Github, por lo que nos permite alojar en la nube nuestros proyectos que utilizan Git como sistema de control de versiones.

2.4. Base de datos MariaDB

2.4.1. Introducción

Servidor MariaDB es un reemplazo mejorado y directo del popular MySQL, es uno de los servidores de base de datos más populares del mundo. Creada por los mismos desarrolladores de MySQL y garantiza que seguirá siendo de código abierto.

Este servidor convierte los datos de información estructurada en una amplia gama de aplicaciones; puede ser usada por equipos de desarrollo de software, testeo de aplicaciones, sitios web de empresas y blogs, servidores vps o locales y de hosting compartido, desde aplicaciones web hasta la banca.

MariaDB se utiliza porque es rápido, estable y robusto con un rico ecosistema de motores de almacenamiento, complementos y muchas otras herramientas que lo hacen mas versátil para una amplia variedad de casos de uso.(MariaDB, 2019).

Este servidor de base de datos esta desarrollado como software de código abierto y como una base de datos relacional que proporciona una interfaz SQL para acceder a los datos. Las ultimas versiones de MariaDB también incluyen funciones GIS Y JSON.

Fig. 13: Logo MariaDB
Fuente: https://mariadb.org/about/logos/

2.4.2. Origen

MariaDB fue desarrollado inicialmente por Michael "Monty" Widenius, uno de los fundadores de la compañía MySQL AB. A fines de febrero del 2008, MySQL AB fue adquirida por Sun Microsystems[vii]. Michael Widenius fue quién comenzó el desarrollo de MariaDB debido a sus preocupaciones en cuanto a la adquisición de Sun Microsystems por parte de Oracle Corporation, ya que no había certeza de que MySQL continuara siendo libre.(MariaDB, 2019)

2.4.3. Características y Comparación MariaDB y MySQL

Se han agregado muchas nuevas características a MariaDB manteniendo las de MySQL pero a la vez mejorando las nuevas que son agregadas. Se muestra la comparación y descripción de Mysql y MariaDB en la TABLA 2.1

TABLA 2.1: Comparativa entre Mysql y MariaDB

COMPARACIÓN		MySQL	MariaDB
Conectores de base de datos	C	X	X
	C++	X	X
	Lau	X	
	Perl	X	X
	Go	X	
	Delphi	X	
	Java	X	X
	Net	X	
	Python	X	X
	Erlang	X	
	ADO. NET		X
	Ruby		X
	Javascript		X
	ODBC		X
	PHP	X	X
	Visual Studio		X
Quién usa la Base de Datos	Facebook	X	
	Twitter	X	
	Youtube	X	
	Netflix	X	
	Wechat	X	
	Zendesk	X	
	NASA	X	
	Linux		X
	Wikipedia		X
	Fedora		X
	Google		X
Comunidades		Es administrada por Oracle corporation, con una gran variedad de foros para ejecutarlo.	Desarrollada por una comunidad de código abierto, con el fin de que todos puedan aportar con información apropiada que ayuden en la documentación y desarrollo
Estructura de Base de Datos		Es un sistema con base de datos relacionales con una estructura de código abierto RDBMS[viii], con tablas desencadenantes, roles, restricciones y componentes centrales.	Es la bifurcación de Mysql, cuenta con la misma estructura e índices. Lo que permite que pueda cambiarse sin tener modificaciones.

Documentación	Es mantenida por la corporación Oracle.	Mantenida libremente por personas que puedan colaborar.
Desarrollo continuo	Cuenta con un desarrollo continuo por parte de Corporación Oracle con decisiones que no están abiertas al público.	Desarrolla con una metodología diferente de MySQL y que esta abierto al público, en donde todas las decisiones de desarrollo se debaten y revisan mediante una lista de correo pública.
Replicación o Agrupamientos disponibles	El servidor actúa como maestro y los demás como esclavos, además pueden replicarse todas las bases de datos e incluso tablas seleccionadas.	Usa una galera clúster para multi-máster, lo que permite una agrupación simple, así como la activación de parámetros de configuración.
Soporte Técnico	Cuenta con un soporte técnico de por vida por parte de Corporación Oracle que aportan correcciones de errores, versiones de mantenimiento y parches.	Ofrece un soporte técnico mediante un grupo de ingenieros expertos 24/7 mediante una subscripción empresarial.

Fuente: Propia

2.5. Frameworks

Un framework está definido como un marco de trabajo que agrupa conceptos, prácticas y criterios de programación que serán utilizados en el desarrollo de una aplicación. Estos se basan en patrones de diseño generando la estructura de la aplicación. “Existen diferentes Framework que intentan unificar la mayoría de las aplicaciones partiendo de un lenguaje común y la tendencia debido a la tendencia de unificar el programador web con el programador de aplicaciones” (Mateu, 2004) El patrón de diseño MVC es uno de los más usados en la actualidad, siendo implementado por frameworks conocidos como: Ruby on Rails, Spring, Angular, Electron, Ionic, Symfony, etc.

Esto permite que los programadores se enfoquen en escribir código limpio, dejando de lado la estructura de la aplicación.

Ventajas de utilizar un framework:

- El programador no necesita plantear la estructura de una aplicación, ya que el framework proporciona un esqueleto que debe escribir el código.
- Existen varios foros y documentación 100% confiable y relevante para que el programador pueda guiarse para resolver problemas al momento de realizar una aplicación.
- Es mas fácil encontrar herramientas, librerías que han sido adaptadas a un framework en concreto para facilitar el desarrollo del programador, hay varias herramientas de mucha utilidad para no volver a escribir dicho código ya existente.

2.5.1. Framework Electron

- **Historia**

Electron Fue desarrollado en el 2013 por Cheng Zhao ingeniero de GitHub, proyecto el cual fue inicialmente desarrollado para un editor de textos que existe en la actualidad llamado Atom, Electron recibía el nombre de Atom Shell, desde entonces se ha empleado para crear infinidad de aplicaciones como GitHubDesktop, WhatsappDesktop, Visual Studio Code, y una infinidad de aplicaciones mas.

- **Introducción**

Si puedes crear un sitio web, puedes crear una aplicación de escritorio. Electron es un framework basado en Node.js (Entorno de ejecución para JavaScript[ix] construido con el motor JavaScript V8 de Chrome) que permite programar aplicaciones de ventanas nativas de escritorio basadas en tecnologías Web como Javascript, HTML y CSS.

Electron está alojado de forma publica en GitHub, es de código abierto y garantiza revisiones constantes por parte de sus creadores. Además, las aplicaciones pueden ser corridas en las tres plataformas de escritorio existentes Mac, Windows y Linux, siendo así un framework multiplataforma.(De Oliveira, Egidio, & Oliveira, 2017).

***ELECTRON**= NodeJS + Chromium*

Este framework utiliza paginas web como una interfaz gráfica de usuario por lo que se podría ver como un navegador Google Chrome (Que internamente posee el motor de Chromium) el cual es controlado por JavaScript. Los frameworks Angular, React, Vue, etc; son quienes impulsan a Electron a desarrollar aplicaciones de escritorio con mayor robustez. Ilustramos el logotipo de Electron en la Fig.14

Fig. 14: Logo Electron
Fuente: https://electronjs.org/

- **Arquitectura Electron**

Antes de entrar a fondo en Apis[x] de Electron, necesitamos analizar los dos procesos disponibles, proceso principal (Main Process) y proceso de renderizado (Renderer Process), cada uno de estos procesos juega un papel especifico. Son completamente diferentes e importantes de entender para comprender el funcionamiento de este framework.

- **Proceso Principal (Main Process)**

El proceso principal maneja varias actividades a nivel del sistema, como eventos de ciclo de vida de la aplicación, también es el proceso donde se crean los menús de la aplicación y otros cuadros de dialogo nativos, como abrir y salvar un archivo. El proceso principal es el fichero JavaScript referenciado por el fichero package.json[xi].

El script que corre en el proceso principal puede mostrar una GUI (Interfaz Gráfica de Usuario) creando páginas web. Una aplicación Electron siempre tiene un proceso principal, pero nunca más de uno. (Electron.org, 2019).

Debido a que Electron usa Chromium para mostrar páginas web, utiliza tambien la arquitectura multi-proceso de Chromium. Cada página web en Electron corre en su propio proceso, el cual es llamado el proceso renderizador.

En los navegadores normales, las páginas web generalmente se ejecutan en espacio aislado y no se les permite el acceso a recursos nativos. Los usuarios de Electron, sin embargo, tienen el poder de utilizar Node.js APIs en las páginas web permitiendo interacciones inferiores a nivel de sistema operativo(Griffith & Wells, 2017).

- **Módulos de Proceso Principal:**

El proceso principal controla el inicio de la aplicación Electron y crea el browserWindows que mostrará el contenido. Para lograr esto se tiene el control sobre una lista de módulos. La TABLA 2.2 muestra los módulos específicos del proceso principal.

TABLA 2.2: Tabla de módulos del proceso principal

Módulos	Definición
App	Módulo que es responsable de controlar el tiempo de vida de la aplicación.
Auto-updater	Se encarga de actualizar la aplicación a una nueva versión.
Browser-Window	Se utiliza para crear la ventana del navegador y la ventana sin marco.
contentTracing	Recopila datos de rastreo del modulo de contenido chromium, para encontrar cuellos de botella de rendimiento y operaciones lentas.
Dialog	Proporciona API para mostrar cuadros de diálogo del sistema nativo. Abrir y guardar archivos, alertas, etc.
Global-shortcut	Este módulo puede registrar / anular el registro de un método abreviado de teclado global en el sistema operativo
Main ipc	Maneja mensajes asíncronos y síncronos enviados desde un proceso de representación
Menu	Crea menús nativos que pueden usarse como menús de aplicaciones y menús contextuales.
Menu-item	Se utiliza para agregar un nuevo elemento a la lista del menú.
Power-monitor	Monitorea el cambio de estado de energía.
Protocol	Registra un nuevo protocolo o intercepta un protocolo existente.
Tray	Representa un icono en el área de notificación del sistema operativo.

Fuente: Propia

- **Renderer Process(Proceso de Renderizado)**

Según, (Morante Luis, 2017a) el proceso de renderizado a su vez tiene otra función, muestra el UI de nuestra aplicación. Cada uno de los procesos de renderizado carga su contenido y lo ejecuta en un hilo separado. Podemos desplegar tantas ventanas con queramos para nuestra aplicación. El proceso de renderizado esta aislado de cualquier iteración con los eventos a nivel del sistema, Electron incluye un sistema de comunicación entre el proceso principal y cualquier proceso de renderizado. Se puede apreciar en la Fig.15 el proceso de renderizado del framework Electron.

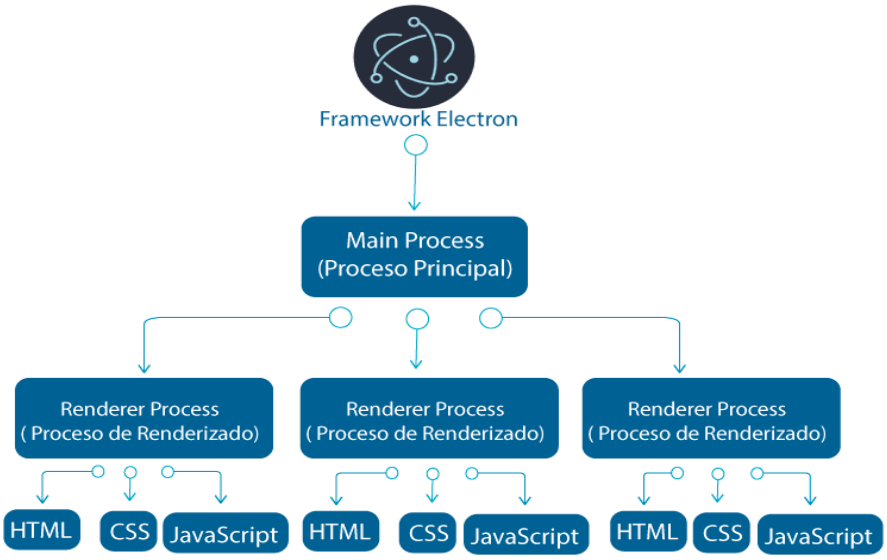


Fig. 15: Architecture Electron Main and Render process
***Fuente**: Propia*

Aunque en la actualidad predominan las tecnologías de desarrollo para entornos web y móviles, las aplicaciones de ventanas de escritorio clásicas aún son muy necesarias, por ejemplo, cuando tenemos la necesidad de interactuar con el sistema operativo local. Gracias al framework Electron un desarrollador web puede construir fácilmente aplicaciones de este tipo.

- **¿Por qué usar o no framework Electron?**

Electron es una de las muchas herramientas para poder diseñar aplicaciones de escritorio, pero esta vez una opción fue estudiarla debido a que los desarrolladores de paginas web también tienen la necesidad de usar recursos y periféricos del computador.

Electron es la versión reducida de chromium, su combinación con Nodejs es prácticamente usar dos aplicaciones para ejecutar una tercera aplicación, un “HOLA MUNDO” de Electron consume aproximadamente alrededor de 60Mb de RAM y hay que admitir que es un valor considerable en comparación con otras herramientas.

Usando Electron podemos tener excelentes resultados de calidad, como lo podemos apreciar una de las buenas aplicaciones que es VS Code muy popular en el mundo del desarrollo y usada por miles de usuarios. Esta aplicación es muy apreciada por el rendimiento y la cantidad de características y extensiones que ayudan al trabajo del desarrollo.

Como resultado se tiene que, si realizamos de una forma correcta el uso del framework Electron, se puede construir aplicaciones a gran escala y lo mas efectivo es que ya no serán de la forma tradicional, una de las mayores ventajas es que podemos usar una aplicación tanto en la web como en el escritorio a nivel multiplataforma.

Ahora bien, si después de considerar estos puntos, aún quieres conocer sobre Electron, veamos los fundamentos para empezar a desarrollar una aplicación con esta herramienta.

- **Instalación y configuración**

  - **Prerrequisitos**

Para hacer uso del framework Electron es necesario cumplir ciertos requisitos para su optimo funcionamiento.

1. Sobre el sistema operativo.

Electron es compatible con los sistemas operativos que indica en la tabla: requisitos del sistema operativo.

TABLA 2.3: Sistemas Operativos

SISTEMA OPERATIVO	VERSIÓN
Mac Os	IOS 7 o Superior
Linux	Ubuntu 12.0 o Superior
Windows	Windows Vista o Superior

Fuente: Propia

2. Sobre memoria y almacenamiento

Para el desarrollo de aplicaciones con Electron el almacenamiento no es un gran impedimento ya que tan solo se necesitaría 2 Gb adicionales de las que exige el sistema operativo en el que se va a instalar, en cuestión de memoria es necesario adicionar 512 Mb de memoria RAM que las requeridas por el sistema operativo, es recomendable que nuestro ordenador tenga un mínimo de 4 Gb de memora RAM para que el desarrollo no se vea afectado.

3. Software Adicional

El framework que estamos estudiando funciona mediante dependencias, por lo que es estrictamente necesario el uso de NODE JS. Lo cual para instalarlo podemos acceder a su pagina oficial y descargarlo desde el siguiente enlace: https://nodejs.org/es/download/. Dependiendo del sistema operativo en el cual deseamos ejecutarlo. Ilustramos en la Fig.16 la pagina donde podemos encontrar NodeJs.

Fig. 16: Pagina de instalación NODE JS.
Fuente: https://nodejs.org/es/download/

- **Instalación**

Una vez instalado y configurado los pre-requisitos, procedemos con la instalación para lo es necesario una línea de comandos en Windows usaremos CMD y en Linux o Mac OS usaremos Terminal y procedemos a escribir el texto indicado a continuación:

$ npm install electron --save-dev

También se puede instalar como una dependencia modo global y se puede usar desde el CLI que es la librería para la instalación del angular:

$ npm install electron -g

Una vez que tengamos instalado procedemos a verificar la instalación.

$ electron –v

- **Configuración**

La configuración no es mas que ubicarnos en el directorio que será en el que creamos nuestros aplicativos de Electron.

- **Creación del Proyecto**

La creación de un proyecto se lo hace por líneas de comando o también con el uso de un editor de texto como VS Code o Atom que son creador por el framework Electron.

Nos dirigimos a la carpeta que vamos a tener nuestra aplicación y creamor el package.json.

1. **Contenido del proyecto**

Nuestra aplicación va a contener 3 archivos principales como se puede observar en la TABLA 2.4

TABLA 2.4: Archivos principales de un proyecto Electron.

Archivo	Contexto
package.js	Archivo donde se va a quedar la configuración del aplicativo, este tiene que estar en la raíz del proyecto. El cual apunta al archivo principal main.js
main.js	Donde se ejecuta el proceso principal, script de inicio de la aplicación. Se ocupa de crear la ventana del aplicativo. Es quien apunta al archivo web index.html
index.html	Una web para renderizar con aspecto de ventana de escritorio.

Fuente: Propia

2. Descripción de la aplicación

- ***Package.json***

Para instalar este fichero cuando comenzamos desde cero lo hacemos con la instrucción:

$ npm init

Un archivo package.json puede tener algunos elementos típicos como en la Fig.17:

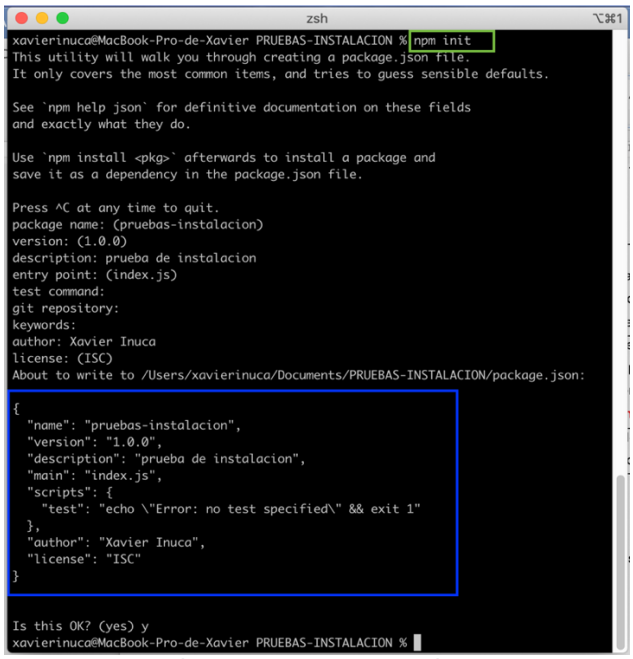


Fig. 17: Ejecución del npm init para la creación del package.json
Fuente: Propia

En la Fig.18 podemos apreciar el archivo con la información que se ha creado al momento de ejecutar npm init, es muy fácil crear nuestro package.json, y asi podemos continuar con la ejecución de nuestro aplicativo Electron.

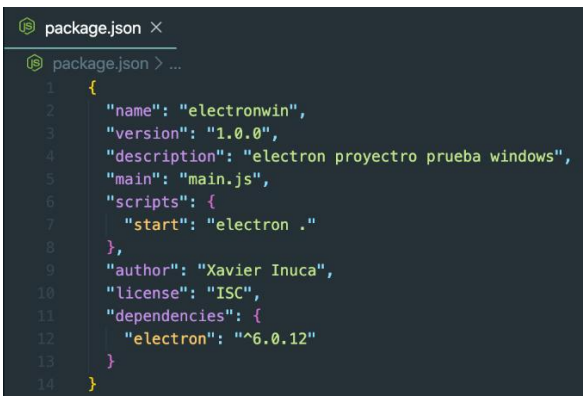


Fig. 18: Estructura package.json
Fuente: Propia

En la Tabla 2.5 tenemos la descripción de algunos parámetros a destacar dentro del package.json.

TABLA 2.5: Descripción de los parámetros del package.json

Nombre	Descripción
"name"	Se refiere evidentemente al nombre de nuestra aplicación, el formato debe ser en minúsculas y para separar letras con un guion.
"version"	Representa la versión del código fuente del proyecto.
"description"	Breve descripción del contenido de nuestra aplicación.
"main"	Es donde indicamos a Electron donde se encuentra el código del proceso principal (Main Process).
"scripts"	Tenemos que agregarlo, y designar los comandos específicos de NPM, en este vamos a invocar a Electron.

Fuente: Propia

- ***Main.js***

Todas las aplicaciones Electron comienzan con un script principal que es similar a una aplicación node.js. El fichero main.js crea las ventanas de nuestra aplicación, maneja los eventos y ejecuta el proceso principal. Para poder mostrar el interfaz de usuario, el proceso principal crea un proceso de renderizado de la ventana usando BrowserWindow.(Patel Sandeep Kumar, 2015).

Lo primero es agregar el módulo Electron dentro del archivo main.js.

```
const {app,BrowserWindow} = require('electron')
```

También se puede escribir como el siguiente ejemplo.

```
const electron = require("electron");
const app = electron.app;
const BrowserWindow = electron.BrowserWindow;
```

Definimos dos variables constantes que a destacar dentro del main.js como se puede ver en la TABLA 2.6

TABLA 2.6: Variables constantes de Electron.

Nombre	Descripción
electron.app	Controla el ciclo de vida de la app.
electron.browserwindow	Clase para crear la ventana de la app. Acceso al gestor de ventanas de Chromium.

Fuente: Propia

La siguiente función es la encargada de visualizar la ventana principal usando una instancia de la clase BrowserWindow que se asocia a la variable global mainWindow. Pasamos al constructor la anchura y altura de la ventana en pixeles.

Luego llamamos al método loadFile indicando la ruta al fichero HTML como parámetro.

La última instrucción maneja el evento de cierre de la ventana.

```
let win
function createWindow() {
  win = new BrowserWindow({
    width: 800,
    height: 600,
    webPreferences: {
      nodeIntegration: true
    }
  })
  win.loadFile('index.html')
  win.webContents.openDevTools()
  win.on('closed', () => {
    win = null
  })
}
```

El siguiente método espera a que Electron acabe su inicialización antes de llamar a la función que crea la ventana. El evento ready es emitido cuando la aplicación ha terminado su iniciación básica.

```
app.on('ready', createWindow)
```

El evento window-all-closed se emitirá cuando todas las ventanas asociadas a la app se cierren. Vale decir, cuando no existan instancias de la clase BrowserWindow.

```
app.on('window-all-closed', () => {
  if (process.platform !== 'darwin') {
    app.quit()
  }
})
```

activate es emitido cuando la aplicación está activada. Comprobamos que no este la ventana ya cargada para llamar a la función createWindow().

```
app.on('activate', () => {
  if (win === null) {
    createWindow()
  }
})
```

- ***Index.html***

El fichero HTML es cargado usando BrowserWindow. Cualquier HTML, CSS o Javascript que funcione en un navegador Web funciona aquí también(Patel Sandeep Kumar, 2015).

En la página index.html definimos el aspecto que le queremos dar. Podemos referenciar a ficheros CSS como Bootstrap para darle estilo o a librerias JS como jQuery.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Hello World!</title>
  </head>
  <body>
    <h1>Hello World!</h1>
    <!-- All of the Node.js APIs are available in this renderer process. -->
    We are using Node.js <script>document.write(process.versions.node)</script>,
    Chromium <script>document.write(process.versions.chrome)</script>,
    and Electron <script>document.write(process.versions.electron)</script>.

    <script>
      // You can also require other files to run in this process
      require('./renderer.js')
    </script>
  </body>
</html>
```

- ***Estructura final***

Para obtener tener todos los demas recursos instalados procedemos a ejecutar el siguiente comando.

npm install

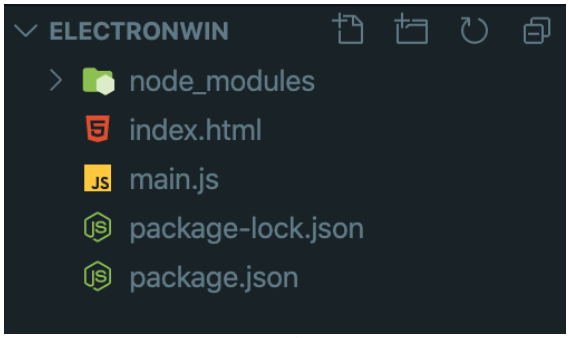


Fig. 19: Estructura final del aplicativo
Fuente: Propia

Una vez que tengamos la estructura de la Fig.18 tendremos todos los node_modules y dependencias necesarias para poder ejecutar el aplicativo.

Dentro del directorio de la carpeta del proyecto en un terminal o cmd procedemos a ejecutar el aplicativo Electron con el siguiente comando.

npm run start

Se ilustra en la Fig.20 el ejemplo de la aplicación ejecutada en el sistema operativo Windows.

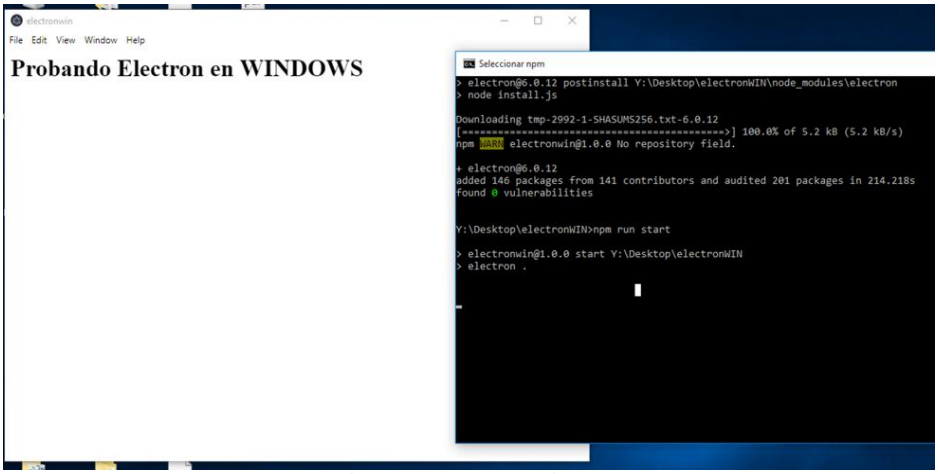


Fig. 20: Ejemplo de ejecución del aplicativo con el framework Electron en Windows.
Fuente: Propia

Se ilustra en la Fig.21 el ejemplo de la aplicación ejecutada en el sistema operativo MacOS.

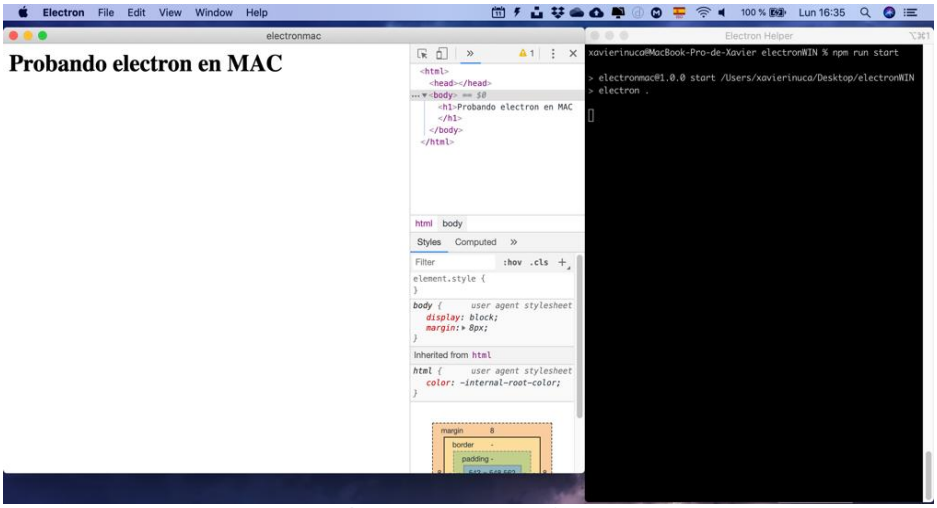


Fig. 21 Ejemplo de ejecución del aplicativo con el framework Electron en Mac-Os.
Fuente: Propia

2.5.2. NodeJs

Node.js se lanzó inicialmente en 2009 como un proyecto de código abierto, lo que permite a los desarrolladores crear aplicaciones del lado del servidor utilizando JavaScript. Lo que hizo interesante este proyecto fue que aprovechó el nuevo motor V8 de código abierto de Google para actuar como su tiempo de ejecución de JavaScript. (Vamshi Krishna A, 2019)

Además de ese tiempo de ejecución, el proyecto agregó APIS para acceder al sistema de archivos local, crear servidores, así como la capacidad de cargar módulos. Node ha disfrutado de un enorme aumento de popularidad en toda la comunidad de desarrollo. Como tal, hay

una gran colección de módulos que están disponibles para su uso dentro del aplicativo Electron.

Esta tecnología se usa como plataforma para ejecutar el servicio express, luego la aplicación backend sobre el mismo y se usa para tener la conexión a la base de datos, también se usa para la ejecución del aplicativo Electron y angular. (Vamshi Krishna A, 2019)

Según (Chandru, Dinesh Kumar, Karthikeyan, & Saranya, 2019) dice que NodeJS es la herramienta perfecta para manejar llamadas asincrónicas. Los datos se transferirán en forma de contrato JSON para facilitar la interpretación. La aplicación web estará compuesta por puntos de conexión REST api para realizar diversas operaciones. La aplicación se basará en la arquitectura MicroServices para admitir modularidad, la escalabilidad y la facilidad de uso.

- **Express**

El framework web express está construido sobre Connect, proporcionando herramientas y estructura que hacen que la escritura de aplicaciones web sea más fácil, más rápida y divertida. Esto incluye características como un sistema de vista unificada que le permite usar casi cualquier template engines que desee, utilidades simples para responder a diversos formatos de datos, negociación de contenido, transferencia de archivos, enrutamiento y más.(Gheorghe & Lee Hinman, 2013).

- **Sequelize ORM**

Sequelize es un ORM[xii] Node.js basado en promesas para Postgres, MySQL, MariaDB, SQLite y Microsoft SQL Server. Cuenta con soporte de transacciones sólidas, relaciones, carga ansiosa y perezosa, replicación de lectura y más.

Nos sirve para poder conectar nuestro aplicativo a una base de datos relacional, realizando la parte del backend y realizando las API Rest.

2.5.3. Framework Angular

Fig. 22: Angular
Fuente: (Genuitec, 2017)

Desde su creación este framework a sido uno de los mas populares y preferidos por desarrolladores de JavaScript. Tiene una gran popularidad que ha venido creciendo en el transcurso del tiempo con constantes actualizaciones y mejoras del framework.

Angular sirve para ayudar a organizar el código, creando aplicaciones más fáciles de mantener y con un buen rendimiento, y que además permite utilizar TypeScript como lenguaje, lo que permite optimizar y organizar el código con el manejo de estructuras de datos y metodología de orientación a objetos.

Angular es un potente framework de JavaScript creado para el desarrollo de aplicaciones web dinámicas. Nos permite extender la sintaxis de HTML por medio de atributos propios del framework, para expresar componentes de nuestra aplicación de manera dinámica (Arizmendi, 2018).

- **Características de Angular**

Angular es una plataforma de desarrollo para crear aplicaciones utilizando estándares web modernos. Incluye una gran cantidad de características esenciales (Vásquez, 2017).

Fig. 23: Características Angular
Fuente: Propia

- **Ventajas de Utilizar Angular**

La gran ventaja de utilizar este Framework es que es extremadamente modular, ligero y fácil de aprender. Al manejar todo en base a módulos y tener componentes separado en el entorno de desarrollo permite un mejor mantenimiento de las aplicaciones. Está demostrado que una aplicación Angular es superior en performance frente a Angular (Vásquez, 2017).

Otra de las ventajas que presenta este Framework y la gran mayoría de ellos es que son Open Source, permitiendo que el proyecto a desarrollar no piense en costos al momento de la etapa de desarrollo (Vásquez, 2017).

Ideal para aplicaciones empresariales con arquitecturas orientadas a servicios, lo que se conoce en inglés como SOA (Service Oriented Architecture) y aplicaciones conocidas como de una sola página SPA (Single Page Application). (Arizmendi, 2018).

No podríamos decir que es el mejor Framework para el desarrollo FrontEnd, porque existen varios Framework que presentan sus propias características y que tienen el objetivo común de ayudar a crear aplicaciones modernas de forma rápida, sencilla y de fácil mantenimiento (Vásquez, 2017).

- **Arquitectura angular**

De acuerdo a sus autores, adopta el patrón de diseño (MVW) Model-View-Whatever, lo más importante aquí, es que separa claramente las capas del Modelo-Vista-Controlador y las ventajas que esto Conlleva, además algo a destacar es que las aplicaciones desarrolladas en Angular hacen uso de Servicios, Directivas y Filtros (Arizmendi, 2018).

2.6. Autenticaciones

La autenticación es la capacidad de demostrar que un usuario es dicha persona que asegura ser, para tener control y acceso a los datos que están dentro del aplicativo. Teniendo en cuenta los roles de cada usuario para poder ingresar a los datos específicos. Uno de los servicios para autenticación tradicional es JTW (Json Web Token).(jwt.io, 2019)

2.6.1. JWT

Json Web Token es un estándar de código abierto que define una forma compacta y autónoma para transmitir información de forma segura entre las partes como un objeto JSON[xiii]. Esta información se puede verificar y confiar por que está firmada digitalmente.

Según (jwt.io, 2019), los JWT se pueden firmar usando un secreto (con el algoritmo HMAC[xiv]) o un par de claves pública/privada usando RSA[xv] o ECDSA[xvi].

- **¿Cómo Funcionan los tokens web JSON?**

En la autenticación, cuando el usuario inicia sesión con éxito e ingresa al sistema utilizando sus respectivas credenciales, se devolverá como resultado su token web JSON. Dado que los tokens son credenciales, se debe tener mucho cuidado para evitar problemas de seguridad. En realidad, los tokens no se deben mantener mas del tiempo requerido. Tampoco se debe almacenar datos confidenciales de sesión en el navegador web o plantilla de escritorio debido a la falta de seguridad.

El siguiente diagrama muestra como se obtiene un JWT y cómo se utiliza para acceder a API o recursos.

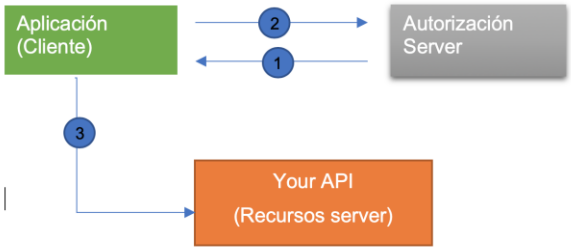


Fig. 24:Funcionalidad de la Autenticaciones con JWT
Fuente: Propia

- (1) El cliente que ingresa a la aplicación solicita autorización al servidor.
- (2) Una vez que el servidor reciba la petición y sean correctas las credenciales de acceso envía una aceptación al cliente para que pueda ingresar
- (3) La aplicación usa el token de acceso para así mostrar los datos o recursos que están protegidos (como una API).

➢ **¿Por qué se debe usar JSON WEB TOKEN?**

Los beneficios de JSON Web Tokens (JWT) en comparación con Simple Web Tokens (SWT[xvii]) y Security Assertion Markup Language Tokens (SAML[xviii]).

En cuanto a la seguridad, SWT solo puede ser firmado simétricamente por un secreto compartido utilizando el algoritmo HMAC. Sin embargo, los tokens JWT y SAML pueden usar un par de claves pública / privada en forma de un certificado X.509 para la firma. Firmar XML con firma digital XML sin introducir agujeros de seguridad oscuros es muy difícil en comparación con la simplicidad de firmar JSON. (jwt.io, 2019).

2.7. Validaciones

Para validar los diferentes campos y funciones que tendremos dentro del aplicativo, se procederá a usar diferentes métodos, para tener mayor facilidad se usara recursos de Nodejs.

2.7.1. Express-Validator

Es una biblioteca la cual brinda funciones de validación y desinfección. En palabras claras es quien valida la autenticación y demás funciones que estarán en uso dentro de la aplicación.

Validador es uno de los muchos paquetes de NPM que tiene como función validar una solicitud de una aplicación Express.

- **¿Por qué la validación del lado del servidor?**

Según (Copes, 2018) menciona que “Debe preocuparse por la validación del lado del servidor porque no se puede confiar en los datos procedentes de otros clientes (aplicaciones de una sola página, aplicaciones web normales, aplicaciones móviles, etc.)”. Por ejemplo, sus servidores no tienen medios para saber si un usuario malicioso (o virus) deshabilitó la validación de front-end (por ejemplo, JavaScript) para permitir que la aplicación envíe datos falsos a su servidor. Es decir, la validación del lado del cliente no es suficiente.

Para poder tener validaciones de calidad es recomendable realizar tanto del lado servidor como del cliente, sea para seguridad de datos como autenticación o también para la validación de los diferentes campos, como es el limite de datos, verificación de números, caracteres, letras, etc.

CAPÍTULO III
3. Desarrollo del Aplicativo

3.1. Planeación del proyecto

En la planeación de proyectos de software las tareas técnicas y de gerencia son fundamentalmente importantes para el éxito del proyecto.

Las actividades de la administración de un proyecto comprenden los métodos para organizar y seguir el curso del proyecto, estimación de costos, políticas de asignación de recursos, control de presupuesto, determinación de avances, ajustes al calendario de trabajo, procedimientos de control de calidad, comunicación con el cliente, etc.(Gálvez Zambrano, 2012).

3.1.1. Roles de usuario

Se definen los roles de los usuarios que ayudará a distinguir a las personas involucradas en el desarrollo del aplicativo en la TABLA 3.1

TABLA 3.1: Roles de usuarios

Nombre	Descripción	Rol
Msc. Daysi Imbaquingo	Encargada de revisiones de los avances del sistema.	Consultor
Miembros JCPD-Otavalo	Encargado de pruebas funcionales.	Cliente
Xavier Inuca	Encargado del desarrollo del sistema.	Programador

Fuente: Propia

3.1.2. Tipos de usuarios

- Usuario Administrador: Permite controlar el modulo administrativo del sistema y visualizar usuarios.
- Usuario: En este modulo todos los usuarios del JCPD registran los datos de los casos de los diferentes afectados.

3.1.3. Historias del Usuario

Durante la reunión realizada con los miembros que conforman la JCPD, se trato sobre el aplicativo a desarrollar, en base a cada uno de los detalles de los miembros de la JCPD, se procedió a elaborar la historia de usuario presentada a continuación. Se muestran las historias de usuarios desde la TABLA 3.2 hasta la TABLA 3.9

TABLA 3.2: Historia de usuario 001

	Historia de Usuario
Número: 001	**Usuario:** Administrador
Nombre Historia: Acceso al aplicativo de escritorio	
Prioridad: **Alta**	**Riesgo en desarrollo:** Alto
Programador: Sr.Xavier Inuca	**Iteraciones:** 2
Descripción: Para ingresar al sistema se necesita de un registro por parte de los usuarios que accedan al sistema mismo que debe estar conformado por usuario que será el correo electrónico y una contraseña.	

Fuente: Propia

TABLA 3.3: Historia de usuario 002

Historia de Usuario	
Número:002	**Usuario:** Administrador
Nombre Historia: Gestión de Usuarios	
Prioridad: **Alta**	**Riesgo en desarrollo:** Alto
Programador: Sr.Xavier Inuca	**Iteraciones:** 2
Descripción: Si el usuario deja de ser miembro de la JCPD-Otavalo, procede a ser eliminado del acceso al aplicativo. Para registrar un usuario, previamente debemos estar logueado como un administrador. También la asignación de roles, por defecto se crea rol de USER_ROL, pero si quiere ser un ADMIN_ROL debe ser creado por el administrador principal.	

Fuente: Propia

TABLA 3.4: Historia de usuario 003

Historias de usuario	
Numero: 003	**Usuario:** Personal de la JCPD
Historias: Gestión de Casos	
Prioridad: Alta	**Riesgo en desarrollo:** Alto
Programador: Sr.Xavier Inuca	**Iteraciones:** 2
Descripción: Se deben registrar las denuncias, donde se debe escoger si es de oficio, verbal o escrita, es importante que se registre una pequeña descripción del motivo de la denuncia y si hubiera alguna observación, se deben presentar una lista de la clasificación de los derechos del amenazado donde se pueda seleccionar varias medidas de los aspectos amenazados o violentados. Paralelamente se debe ingresar los datos del derecho o los derechos amenazados o vulnerados, si son derechos colectivos o individuales donde algunos parámetros no obligatorios es decir se llenen opcionalmente. Se necesita también obtener la mayor información del Afectado, Denunciado y Denunciante.	

Fuente: Propia

TABLA 3.5: Historia de usuario 004

Historias de usuario	
Numero: 004	**Usuario:** Personal de la JCPD
Historias: Revisión de Casos	
Prioridad: Alta	**Riesgo en desarrollo:** Alto
Programador: Sr.Xavier Inuca	**Iteraciones: 1**
Descripción: El sistema debe contar con un apartado que permita buscar un caso con los siguientes filtros: • Fecha desde • Fecha hasta • Numero de caso Y si existe el caso se deberá realizar un análisis de dicho caso donde se podrá revisar y analizar el caso, para calificar la denuncia, abstención de la junta, poner en conocimiento a otras autoridades o iniciar el PAPD. En el caso de que no se dé inicio al proceso de restitución de derechos será necesario de que se pueda escribir las justificaciones dependiendo del caso; caso contrario pasará al siguiente paso que es avocar el conocimiento.	

Fuente: Propia

TABLA 3.6: Historia de usuario 005

Historias de usuario	
Numero: 005	**Usuario:** Personal de la JCPD
Historias: Avocatorias	
Prioridad: Alta	**Riesgo en desarrollo:** Alta
Programador: Sr.Xavier Inuca	**Iteraciones:** 1
Descripción: En registro de avocatorias se debe ingresar la fecha de audiencia, de conciliación, la hora de la audiencia. CRUD de avocatorias.	

Fuente: Propia

TABLA 3.7: Historia de usuario 006

Historias de usuario	
Numero: 006	**Usuario:** Personal de la JCPD
Historias: Audiencias	
Prioridad: Alta	**Riesgo en desarrollo:** Alta
Programador: Sr.Xavier Inuca	**Iteraciones:** 2
Descripción: Una vez que se finalice el proceso de avocatorias se procederá a revisar las audiencias donde seleccionando el caso se registre la audiencia, en el formulario debe contener una observación, el resumen de la audiencia e indicar si hubo conciliación se debe proceder a la resolución, en el caso de que no haya resolución, pasara a una nueva audiencia.	

Fuente: Propia

TABLA 3.8: Historia de usuario 007

Historias de usuario

Numero: 007	**Usuario:** Personal de la JCPD
Historias: Resolución	
Prioridad: Alta	**Riesgo en desarrollo:** Alta
Programador: Sr.Xavier Inuca	**Iteraciones:** 2
Descripción: En el caso de que si exista una conciliación se procederá a registrar la resolución donde se redactará un corto resumen de la resolución y el registro de una medida administrativa con sus respectivas fechas de aplicación de medidas correspondientes.	

Fuente: Propia

TABLA 3.9: Historia de usuario 008

Historias de usuario	
Numero: 008	**Usuario:** Personal de la JCPD
Historias: Almacenamiento de archivos	
Prioridad: Alta	**Riesgo en desarrollo:** Alta
Programador: Sr.Xavier Inuca	**Iteraciones:** 2
Descripción: Todos los archivos firmados por las partes y que pertenecen a un caso serán almacenados en una base de datos, para así en un futuro tener acceso a estos archivos.	

Fuente: Propia

3.1.4. Plan de entregas.

En esta fase se prosigue a ordenar en orden cronológico las historias de usuarios, para poder realizar un plan de entregas que debe ser cumplido en las iteraciones propuestas.

Los tiempos de desarrollo se medirán de acuerdo con la TABLA 3.10:

TABLA 3.10: Equivalencias de tiempo de desarrollo

Tiempo	Equivalencia
Un día	Ocho horas.

Fuente: Propia

Se plantea los planes de entrega de cada historia de usuario con las fechas de cumplimiento en la TABLA 3.11

TABLA 3.11: Plan de Entregas

Módulos	Nro.	Historias de usuarios	Fechas estimadas	Esfuerzo		Iteraciones		Entregas	
				Días	Horas	1	2	1	2
Módulos de administración	1	Acceso al sistema de escritorio	14/10/2019 a 23/10/2019	10	80	X		X	
	2	Gestión de usuarios	28/10/2019 a 31/10/2019	4	32	X		X	
Módulos de cliente y administrador	3	Gestión de Casos	05/11/2019 a 13/11/2019	7	56	X		X	
	4	Revisión de Casos	14/11/2019 a 18/11/2019	3	24		X		X
	5	Gestión de Avocatorias	19/11/2019 a 20/11/2019	2	16		X		X
	6	Audiencias	26/11/2019 a 27/11/2019	2	16		X		X
	7	Resolución	28/11/2019 a 29/11/2019	2	16		X		X
	8	Almacenamiento de archivos.	02/12/2019 a 06/12/2019	5	40		X		X

Fuente: Propia

3.2. Definición de requisitos del sistema

El aplicativo estará enfocado en la gestión, seguimiento, ingreso de información para automatizar los procesos de un caso atendido por la Junta Cantonal de protección de derechos. A fin de que todo el resultado del desarrollo se base en el contenido de este documento.

La aplicación tendrá que cumplir con los siguientes requisitos funcionales. Es aquí donde se detalla el trabajo a realizarse en cada historia de usuario, viendo si cada uno de los requerimientos es funcional o no funcional. Tenemos los requerimientos desde la TABLA 3.12 hasta la TABLA 3.24

➢ Requerimientos Funcionales

TABLA 3.12: Tabla de Diseño de la Interfaz

Responsable	Xavier Inuca
Identificación del requerimiento:	RF01
Nombre del Requerimiento:	Diseño Interfaz
Características:	El sistema contará con una interfaz sencilla y fácil de manipular por parte de los usuarios
Descripción del requerimiento:	La interfaz debe poseer un diseño sencillo y con los colores principales del GAD de Otavalo (Azul, Verde, Rojo), la interfaz principal debe poseer menús desplegables y la cantidad de pestañas según requiera el usuario, principalmente se establecerán las siguientes: Casos • Registro de Casos • Búsqueda de Casos Procesos • Revisión de Casos • Revisión de Avocatorias • Revisión de Audiencias • Revisión de Resolución
Prioridad del requerimiento:	Alta

Fuente: Propia

TABLA 3.13: Autenticación de Usuarios

Responsable	Xavier Inuca
Identificación del requerimiento:	RF02
Nombre del Requerimiento:	Autentificación de Usuario.
Características:	Todos los usuarios deberán identificarse para acceder a cualquier parte del sistema con su debido nombre de usuario o correo electrónico y contraseña, este nos dará acceso depende del rol q este establecido.
Prioridad del requerimiento:	Alta

Fuente: Propia

TABLA 3.14: Registro de usuarios por parte del Administrador

Responsable	Xavier Inuca
Identificación del requerimiento:	RF03
Nombre del Requerimiento:	Registro de Usuarios.
Características:	El administrador general del sistema podrá registrar y dar de baja del sistema a los usuarios.
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.15: Registro de Casos

Responsable	Xavier Inuca
Identificación del requerimiento:	RF04
Nombre del Requerimiento:	Registro de Casos
Características:	La aplicación permitirá registrar nuevos casos.
Descripción del requerimiento:	El registro de casos debe contener los siguientes parámetros establecidos por los miembros de la JCPD: • Numero de Caso • Procesamiento con conocimiento De oficio A petición de parte • Tipo de denuncia Verbal Escrita • Motivo • Observaciones Conjuntamente se mostrará una lista desplegable con la clasificación de derechos en donde se podrá seleccionar uno o varios derechos.
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.16: Registro de Datos Afectado

Responsable	Xavier Inuca
Identificación del requerimiento:	RF05
Nombre del Requerimiento:	Registrar Datos del Afectado
Características:	Permite registrar información personal del afectado
Descripción del requerimiento:	El registro de datos del afectado debe contener los siguientes parámetros establecidos por los miembros de la JCPD: • Nombre • Apellido • Número de cedula (opcional) • Fecha de nacimiento (opcional) • Edad(opcional) • Genero • Auto identidad Mestizo Indígena Afrodescendiente • Discapacidad Visual Auditivo Física Intelectual • Dirección • Sector • Teléfono (opcional) • Email (opcional) • Observación
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.17: Registro de Datos Denunciante

Responsable	Xavier Inuca
Identificación del requerimiento:	RF06
Nombre del Requerimiento:	Registro Datos del Denunciante
Características:	Permite ingresar información personal del denunciante
Descripción del requerimiento:	El registro de datos del afectado debe contener los siguientes parámetros establecidos por los miembros de la JCPD: • Persona Natural o Jurídica • Nombre • Apellido • Número de cedula (opcional) • Edad • Genero • Relación con el afectado • Auto identidad Mestizo Indígena Afrodescendiente • Discapacidad Visual Auditivo Física Intelectual • Dirección • Sector • Teléfono (opcional) • Email (opcional) • Observación
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.18: Registro de datos Denunciado

Responsable	Xavier Inuca
Identificación del requerimiento:	RF07
Nombre del Requerimiento:	Registro Datos del Denunciado
Características:	Permite registrar la información personal del denunciante
Descripción del requerimiento:	El registro de datos del denunciado debe contener los siguientes parámetros establecidos por los miembros de la JCPD: • Persona Natural o Jurídica • Nombre • Apellido • Número de cedula (opcional) • Genero • Relación con el afectado • Auto identidad Mestizo Indígena Afrodescendiente • Dirección • Discapacidad Visual Auditivo Física Intelectual • Sector • Teléfono (opcional) • Email (opcional) • Observación
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.19: Búsqueda de casos

Responsable	Xavier Inuca
Identificación del requerimiento:	RF08
Nombre del Requerimiento:	Búsqueda de Casos
Características:	Permite buscar un caso existente
Descripción del requerimiento:	El usuario puede buscar un caso específico con los siguientes parámetros • Fecha • Numero de Caso Si el usuario conoce el número de caso en específico puede buscar ingresando el mismo. Si existe el caso se debe presentar en una tabla indicando el número de caso, fecha de ingreso y la fecha de registro; mismo que se debe seleccionar para ver la información del caso.
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.20: Registro de Avocatorias

Responsable	Xavier Inuca
Identificación del requerimiento:	RF9
Nombre del Requerimiento:	Registro de Avocatorias
Características:	Permite a los miembros de la junta registrar una avocatoria
Descripción del requerimiento:	Al seleccionar la opción de "revisión de avocatorias" en el menú del sistema, se debe mostrar el encabezado con el número de caso, fecha de registro y la información del caso, a continuación, los parámetros requeridos para dicho registro, los cuales son: • Fecha audiencia de conciliación • Hora de la Audiencia • Observación Finalmente, el botón que permita guardar. Pasamos al requisito funcional n10 (RF10) de este documento.
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.21: Registro de Audiencia.

Responsable	Xavier Inuca
Identificación del requerimiento:	RF10
Nombre del Requerimiento:	Registro de Audiencia
Características:	Permite a los miembros de la junta registrar la información de la audiencia, además si hubo o no conciliación en la audiencia y la respectiva resolución.
Descripción del requerimiento:	Este formulario debe contener como encabezado el número de caso, fecha de registro, fecha de avocatoria, información del caso, fecha de la audiencia de conciliación y hora de la audiencia. Además, debe poseer los siguientes parámetros para el registro: • Observación • Resumen de la audiencia • Conciliación que consiste con la opción de SI y NO. En el caso de que sí exista conciliación se procederá con el registro de la resolución, mismo que se detallara en el requerimiento funcional n11 (RF11) en este documento.
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.22: Registro de resolución.

Responsable	Xavier Inuca
Identificación del requerimiento:	RF11
Nombre del Requerimiento:	Registro de Resolución
Características:	Permite a los miembros de la junta registrar la información de la resolución de la audiencia.
Descripción del requerimiento:	Este formulario debe contener como encabezado el número de caso y la información del caso Además, debe poseer los siguientes parámetros para el registro: • Resume de la resolución • Registro medida administrativa misma que contiene los parámetros detallados a continuación: • Fecha de inicio de medida • Fecha de conclusión de la medida • Responsable • También se debe seleccionar la medida de protección dependiendo del artículo 79, 94, 217, según sea el caso. Finalmente, el botón para guardar la información registrada.
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.23: Gestión administrativa

Responsable	Xavier Inuca
Identificación del requerimiento:	RF12
Nombre del Requerimiento:	Modulo Administrativo
Características:	Permite al usuario administrador poseer un control general del sistema
Descripción del requerimiento:	Al ingresar como administrador al sistema podrá realizar actividades como: • Registro de nuevos usuarios • Eliminar a un usuario • Asignación de roles
Prioridad del requerimiento: Alta	

Fuente: Propia

TABLA 3.24: Gestión de Recursos

Responsable	Xavier Inuca
Identificación del requerimiento:	RF13
Nombre del Requerimiento:	Modulo Recursos
Características:	Permite al usuario agregar recursos como un PDF o doc.
Descripción del requerimiento:	Al finalizar el proceso de gestión de denuncias se tiene varios documentos que han sido recopilados con firmas de las partes y se procederá almacenar para una futura intervención en el caso.
Prioridad del requerimiento: Alta	

Fuente: Propia

3.3. Codificación de la aplicación

Después de haber obtenido las historias de usuario, es necesario codificar y plasmar ideas de usuarios y programadores en conjunto a través de código. En programación, el código expresa la interpretación del problema; así podemos utilizar el código para comunicar, hacer común las ideas y en base a la codificación aprender y mejorar.

3.3.1. Módulos del sistema

- **Modulo Administrador**

El administrador podrá acceder al modulo desde cualquier otro computador con sus respectivas credenciales. Para gestionar los usuarios del sistema.

- **Modulo Usuario**

El aplicativo estará disponible en computadoras del GAD-JCPD. Además, el usuario podrá acceder a un menú donde tendrá la opción de poder ver casos, análisis de casos, avocatorias, audiencias y resoluciones. De la misma forma podrá seguir el proceso de denuncia.

3.3.2. Planificación: Iteración I

En la primera iteración en base a las historias de usuario se realiza el detalle del modulo administrador con todos sus requerimientos y especiaciones. Ilustradas desde la TABLA 3.25 hasta la TABLA 3.33

a) Cronograma

La TABLA 3.25 especifica el tiempo de duración para resolver el requerimiento o historia del usuario, asi mostrando la fecha de inicio y fin para poder lograr en un perido corto de tiempo.

TABLA 3.25: Cronograma de iteración 1

Nro.	Historias de usuarios	Fechas estimadas	Duración	
			Días	Horas
1	Acceso al sistema de escritorio	14/10/2019 a 23/10/2019	10	80
2	Gestión de usuarios	28/10/2019 a 31/10/2019	4	32
		TOTAL	**14**	**112**

Fuente: Propia

b) Tareas de Historia 1: Acceso al sistema de escritorio

TABLA 3.26: Tareas de historia 1

Nro.	Nombre	Tiempo Estimado
1	Configuración de entorno de programación	10 horas
2	Configuración del Servidor, creación de Apis Rest.	20 horas
3	Diseño de la interfaz gráfica, aplicación de plantilla de trabajo	20 horas
4	Implementación del modulo Login	30 horas
	TOTAL	**80**

Fuente: Propia

- **Configuración del entorno de Programación**

TABLA 3.27: Tarea de historia 1. Tarea 1 Configuración de entorno de programación

Tarea	
Numero de Tarea: 1	**Numero Historia:** 001
Nombre Tarea: Configuración del entorno de programación	
Tipo de Tarea: General	**Tiempo Estimado:** 10 horas
Fecha de Inicio: 14/10/2019	**Fecha Fin:** 14/10/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Se procede a la configuración del ambiente de trabajo para el desarrollo de el aplicativo de escritorio.	
Componentes Para Implementar:	
Gestor de Texto: Visual Code	
Gestor de Base de Datos: MariaDb, Sequelize ORM	
Programas: NodeJs, Postman, Navicat	

Fuente: Propia

- **Configuración del servidor creación de Apis Rest**

TABLA 3.28: Tarea de historia 1. Tarea 2 configuración del servidor creación de Apis Rest

Tarea	
Numero de Tarea: 2	**Numero Historia:** 001
Nombre Tarea: Configuración del servidor creación de Apis Rest	
Tipo de Tarea: General	**Tiempo Estimado:** 20 horas
Fecha de Inicio: 15/10/2019	**Fecha Fin:** 17/10/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Se procede a configurar el servidor de la aplicación de escritorio con Nodejs, sequelize ORM y Express.	

Fuente: Propia

- **Diseño de interfaz gráfica, aplicación de escritorio**

TABLA 3.29: Tarea de historia 1. Tarea 3 diseño de interfaz gráfica, aplicación de plantilla

Tarea	
Numero de Tarea: 3	**Numero Historia:** 001
Nombre Tarea: Diseño de interfaz gráfica, aplicación de plantilla	
Tipo de Tarea: Diseño	**Tiempo Estimado:** 20 horas
Fecha de Inicio: 18/10/2019	**Fecha Fin:** 20/10/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Se crea una interfaz simple para la ventana de login del sistema de escritorio administrador utilizando la plantilla de ADMINLTE, y modificándola para que pueda ser ejecutada en Angular y Electron.	

Fuente: Propia

- **Implementación de módulo de login**

TABLA 3.30: Tarea de historia 1. Tarea 4, implementación del módulo login

Tarea	
Numero de Tarea: 4	**Numero Historia:** 001
Nombre Tarea: Implementación del modulo login backend y frontend	
Tipo de Tarea: Programación	**Tiempo Estimado:** 30 horas
Fecha de Inicio: 21/10/2019	**Fecha Fin:** 23/10/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Se implementan las funciones del módulo de login para que el usuario pueda acceder al portal con su email y contraseña, según el rol que se establezca.	

Fuente: Propia

c) Tareas de Historia 2: Gestión del Usuarios

TABLA 3.31: Tareas de historia 2

Nro.	Nombre	Horas Estimadas
1	Configuración del entorno de programación	2 horas
2	Diseño de interfaz para cada opción, graficas	5 horas
3	Implementación de CRUD para Usuario	25 horas
	TOTAL	**32** horas

Fuente: Propia

- **Diseño de interfaz gráfica del aplicativo de escritorio**

TABLA 3.32: Tarea de historia 2. Tarea 2 diseño de la interfaz gráfica

Tarea	
Numero de Tarea: 1	**Numero Historia:** 002
Nombre Tarea: Diseño de interfaz gráfica del aplicativo	
Tipo de Tarea: Diseño	**Tiempo Estimado:** 7 horas
Fecha de Inicio: 28/10/2019	**Fecha Fin:** 28/10/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Se realizará un diseño que sea simple y atractivo para el usuario.	

Fuente: Propia

- **Implementación de CRUD para usuarios.**

TABLA 3.33: Tarea de historia 2. Tarea 3, implementación de un CRUD para los Usuarios

Tarea	
Numero de Tarea: 2	**Numero Historia:** 002
Nombre Tarea: Implementación de CRUD para usuarios.	
Tipo de Tarea: Programación	**Tiempo Estimado:** 25 horas
Fecha de Inicio: 29/10/2019	**Fecha Fin:** 31/10/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: La tarea describe implementar las funciones del administrador como la gestión de Usuarios.	

Fuente: Propia

3.3.3. Planificación: Iteración II

En la segunda iteración en base a las historias de usuario se realiza el detalle del modulo usuario con todos sus requerimientos y especiaciones. Ilustradas desde la TABLA 3.34 hasta la TABLA 3.58

a) Cronograma

TABLA 3.34: Cronograma de iteración II

Nro.	Historias de usuarios	Fechas estimadas	Duración	
			Días	Horas
1	Gestión de casos	5/11/2019 a 13/11/2019	7	56
2	Análisis de Casos	14/11/2019 a 18/11/2019	3	24
3	Avocatorias	19/11/2019 a 20/11/2019	2	16
4	Audiencias	26/11/2019 a 27/11/2019	2	16
5	Resolución	28/11/2019 a 29/11/2019	2	16
6	Repositorio de archivos	02/12/2019 a 06/12/2019	5	40
		TOTAL	**21**	**168**

Fuente: Propia

b) Tareas de Historia 3: Gestión de Casos

TABLA 3.35: Tareas de historia 3

Nro.	Nombre	Tiempo Estimado
1	Implementación de CRUD para Casos.	5 horas
2	Implementación de CRUD para Afectados	5 horas
3	Implementación de CRUD para Denunciantes	5 horas
4	Implementación de CRUD para Denunciados	5 horas
5	Mostrar toda la información necesaria del caso.	10 horas
6	Diseño de interfaz gráfica del caso	10 horas
7	Búsqueda del caso por el nombre.	6 horas
8	Activar o Desactivar un Caso	10 horas
	TOTAL	**56 horas**

Fuente: Propia

- **Implementación de un CRUD para Casos**

TABLA 3.36: Tarea de historia 3. Tarea 1 Implementación de un CRUD para Casos

Tarea	
Numero de Tarea: 1	**Numero Historia:** 003
Nombre Tarea: Implementación de CRUD para Casos.	
Tipo de Tarea: Programación	**Tiempo Estimado:** 5 horas
Fecha de Inicio: 5/11/2019	**Fecha Fin:** 5/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: La tarea describe que se debe realizar un CRUD para el registro de casos y se almacenamiento en la base de datos.	

Fuente: Propia

- **Implementación de CRUD para Afectados.**

TABLA 3.37: Tarea de historia 3. Tarea 2 Implementación de CRUD para Afectados

Tarea	
Numero de Tarea: 2	**Numero Historia:** 003
Nombre Tarea: Implementación de CRUD para Afectados.	
Tipo de Tarea: Programación	**Tiempo Estimado:** 5 horas
Fecha de Inicio: 6/11/2019	**Fecha Fin:** 6/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: La tarea describe que luego de añadir un Caso procedemos a añadir un Afectado el cual pertenece a un caso.	

Fuente: Propia

- **Implementación de CRUD para Denunciantes.**

TABLA 3.38: Tarea de historia 3. Tarea 3 implementación de CRUD para Denunciantes.

Tarea	
Numero de Tarea: 3	**Numero Historia:** 003
Nombre Tarea: Implementación de CRUD para Denunciantes.	
Tipo de Tarea: Programación	**Tiempo Estimado:** 5 horas
Fecha de Inicio: 7/11/2019	**Fecha Fin:** 7/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: La tarea describe que luego de añadir datos de un Afectado procedemos a añadir los datos de un Denunciante el cual pertenece a un caso.	

Fuente: Propia

- **Implementación de CRUD para Denunciados.**

TABLA 3.39: Tarea de historia 3. Tarea 4 implementación del CRUD para Denunciados

Tarea	
Numero de Tarea: 4	**Numero Historia:** 003
Nombre Tarea: Implementación de CRUD para Denunciados.	
Tipo de Tarea: Programación	**Tiempo Estimado:** 5 horas
Fecha de Inicio: 7/11/2019	**Fecha Fin:** 7/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: La tarea describe que luego de añadir datos de un Denunciante procedemos a añadir los datos de un Denunciado el cual pertenece a un caso.	

Fuente: Propia

- **Mostrar toda la información necesaria del caso.**

TABLA 3.40: Tarea de historia 3. Tarea 5 mostrar toda la información necesaria del caso

Tarea	
Numero de Tarea: 5	**Numero Historia:** 003
Nombre Tarea: Mostrar toda la información necesaria del caso.	
Tipo de Tarea: Diseño	**Tiempo Estimado:** 10 horas
Fecha de Inicio: 8/11/2019	**Fecha Fin:** 8/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Se crea un archivo JSON con las tablas relacionadas para poder obtener toda la información que pertenece a cada caso, y de la misma manera mostrar toda la información necesaria.	

Fuente: Propia

- **Diseño de interfaz gráfica del caso.**

TABLA 3.41: Tarea de historia 3. Tarea 6 diseño de interfaz grafica del caso

Tarea	
Numero de Tarea: 6	**Numero Historia:** 003
Nombre Tarea: Diseño de interfaz gráfica del caso.	
Tipo de Tarea: Diseño	**Tiempo Estimado:** 12 horas
Fecha de Inicio: 8/11/2019	**Fecha Fin:** 9/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Se diseña una plantilla de fácil manejo para los usuarios, donde se puede observar toda la información respectiva al proceso.	

Fuente: Propia

- **Búsqueda del caso por el nombre**

TABLA 3.42: Tarea de historia 3. Tarea 7 diseño de la interfaz grafica del caso

Tarea	
Numero de Tarea: 7	**Numero Historia:** 003
Nombre Tarea: Búsqueda del caso por el nombre**.**	
Tipo de Tarea: Programación	**Tiempo Estimado:** 6 horas
Fecha de Inicio: 10/11/2019	**Fecha Fin:** 11/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Se realizará un motor de búsqueda por el nombre del caso, o la fecha.	

Fuente: Propia

- **Activar o Desactivar un Caso**

TABLA 3.43: Tarea de historia 3. Tarea 8 activar o desactivar un caso

Tarea	
Numero de Tarea: 8	**Numero Historia:** 003
Nombre Tarea: Activar o Desactivar un Caso	
Tipo de Tarea: Programación	**Tiempo Estimado:** 15 horas
Fecha de Inicio: 11/11/2019	**Fecha Fin:** 13/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Una vez finalizado el caso después de la resolución, se procederá a desactivarlo o de la misma forma si en un futuro se lo requiere, en caso de que sea necesario el caso se lo podrá activar y se reabrirá el caso para seguir su proceso o revisión.	

Fuente: Propia

c) Tareas de Historia 4: Análisis de Casos

TABLA 3.44: Tareas de Historia 4

Nro.	Nombre	Tiempo Estimado
1	Diseño de la interfaz gestión de análisis de caso.	14 horas
2	Mantenimiento de gestión de análisis de casos.	10 horas
	TOTAL	**24 horas**

Fuente: Propia

- **Diseño de la interfaz gestión de análisis de caso.**

TABLA 3.45: Tarea de historia 4. Tarea 1 diseño de la interfaz gestión de análisis de caso

Tarea	
Numero de Tarea: 1	**Numero Historia:** 004
Nombre Tarea: Diseño de la interfaz Gestión de análisis de caso	
Tipo de Tarea: Diseño	**Tiempo Estimado:** 20 horas
Fecha de Inicio: 14/11/2019	**Fecha Fin:** 16/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Diseñamos una interfaz donde se puede ver casos, y cada caso nos muestre un botón que al presionarlo nos redirija a una ventana, donde podemos analizar el caso.	

Fuente: Propia

- **Mantenimiento gestión análisis de casos**

TABLA 3.46: Tarea de historia 4. Tarea 2 mantenimiento gestión análisis de casos

Tarea	
Numero de Tarea: 2	**Numero Historia:** 004
Nombre Tarea: Mantenimiento gestión análisis de casos	
Tipo de Tarea: Programación	**Tiempo Estimado:** 20 horas
Fecha de Inicio: 17/11/2019	**Fecha Fin:** 18/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Cada caso al acceder a analizarlo, podremos agregar la información respectiva luego del respectivo análisis.	

Fuente: Propia

d) Tareas de Historia 5: Avocatorias

TABLA 3.47: Tareas de Historia 5

Nro.	Nombre	Tiempo Estimado
1	Diseño de la interfaz gestión de avocatorias	8 horas
2	Mantenimiento gestión de avocatorias	8 horas
	TOTAL	**16 horas**

Fuente: Propia

- **Diseño de la interfaz gestión de avocatorias.**

TABLA 3.48: Tarea de historia 5. Tarea 1 diseño de la interfaz gestión de Avocatorias

Tarea	
Numero de Tarea: 1	**Numero Historia:** 005
Nombre Tarea: Diseño de la interfaz gestión de avocatorias	
Tipo de Tarea: Diseño	**Tiempo Estimado:** 8 horas
Fecha de Inicio: 19/11/2019	**Fecha Fin:** 19/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Diseñaremos una interfaz donde se pueda gestionar el registro de avocatorias.	

Fuente: Propia

- **Mantenimiento gestión de avocatorias**

TABLA 3.49: Tarea de historia 5. Tarea 2 mantenimiento gestión de avocatorias

Tarea	
Numero de Tarea: 2	**Numero Historia:** 005
Nombre Tarea: Mantenimiento gestión de avocatorias	
Tipo de Tarea: Programación	**Tiempo Estimado:** 8 horas
Fecha de Inicio: 20/11/2019	**Fecha Fin:** 20/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Podremos agregar avocatorias a un caso, a la vez establecer hora y fecha para proceder a una audiencia en el caso de que se requiera y si no se establece el por que y se abandona el caso.	

Fuente: Propia

e) Tareas de Historia 6: Audiencias

TABLA 3.50: Tareas de Historia 6

Nro.	Nombre	Tiempo Estimado
1	Diseño de la interfaz gestión de audiencias	8 horas
2	Mantenimiento gestión de audiencias	8 horas
	TOTAL	**16 horas**

Fuente: Propia

- **Diseño de la interfaz gestión de audiencias.**

TABLA 3.51: Tarea de historia 6. Tarea 1 diseño de la interfaz gestión de audiencias

Tarea	
Numero de Tarea: 1	**Numero Historia:** 006
Nombre Tarea: Diseño de la interfaz gestión de audiencias	
Tipo de Tarea: Diseño	**Tiempo Estimado:** 8 horas
Fecha de Inicio: 26/11/2019	**Fecha Fin:** 26/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Diseñaremos una interfaz donde se pueda gestionar el registro de audiencias.	

Fuente: Propia

- **Mantenimiento gestión de avocatorias**

TABLA 3.52: Tarea de historia 6. Tarea 2 mantenimiento gestión de avocatorias

Tarea	
Numero de Tarea: 2	**Numero Historia:** 006
Nombre Tarea: Mantenimiento gestión de audiencias	
Tipo de Tarea: Programación	**Tiempo Estimado:** 8 horas
Fecha de Inicio: 27/11/2019	**Fecha Fin:** 27/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Podremos agregar audiencias a un caso, después de haber concluido con la avocatoria. Si se llega a una conciliación se pasa al registro de la resolución o caso contrario se vuelve a realizar una nueva audiencia.	

Fuente: Propia

f) Tareas de Historia 7: Resolución

TABLA 3.53: Tareas de Historia 7

Nro.	Nombre	Tiempo Estimado
1	Diseño de la interfaz gestión de resolución	8 horas
2	Mantenimiento gestión de resolución	8 horas
	TOTAL	**16 horas**

Fuente: Propia

- **Diseño de la interfaz gestión de resolución.**

TABLA 3.54: Tarea de historia 7. Tarea 1 diseño de la interfaz gestión de resolución

Tarea	
Numero de Tarea: 1	**Numero Historia:** 007
Nombre Tarea: Diseño de la interfaz gestión de resolución	
Tipo de Tarea: Diseño	**Tiempo Estimado:** 8 horas
Fecha de Inicio: 28/11/2019	**Fecha Fin:** 28/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Diseñaremos una interfaz donde se pueda gestionar el registro de resolución.	

Fuente: Propia

- **Mantenimiento gestión de resolución**

TABLA 3.55: Tarea de historia 7. Tarea 2 mantenimiento gestión de resolución

Tarea	
Numero de Tarea: 2	**Numero Historia:** 007
Nombre Tarea: Mantenimiento gestión de resolución	
Tipo de Tarea: Programación	**Tiempo Estimado:** 8 horas
Fecha de Inicio: 29/11/2019	**Fecha Fin:** 29/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Podremos agregar resolución después de haber existido una conciliación en las audiencias que se han establecido anteriormente, una vez que registremos la resolución, procederemos a dar de baja el caso, desactivarlo y si en algún momento se lo necesita revisar se lo puede volver activar o para otra vez retomar el caso.	

Fuente: Propia

g) Tareas de Historia 8: Repositorio de archivos

TABLA 3.56: Tareas de Historia 8

Nro.	Nombre	Tiempo Estimado
1	Diseño de la interfaz repositorio de archivos para cada caso.	20 horas
2	Mantenimiento de repositorio archivos.	20 horas
	TOTAL	**40 horas**

Fuente: Propia

- **Diseño de la interfaz repositorio de archivos para cada caso**

TABLA 3.57: Tarea de historia 8. Tarea 1 Diseño de la interfaz repositorio de archivos para cada caso

Tarea	
Numero de Tarea: 1	**Numero Historia:** 008
Nombre Tarea: Diseño de la interfaz repositorio de archivos para cada caso	
Tipo de Tarea: Diseño	**Tiempo Estimado:** 8 horas
Fecha de Inicio: 02/11/2019	**Fecha Fin:** 04/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Diseñaremos una interfaz donde podamos ver los archivos añadidos, de forma fácil y ordenada.	

Fuente: Propia

- **Mantenimiento de repositorio archivos**

TABLA 3.58: Tarea de historia 8. Tarea 2 mantenimiento de repositorio archivos

Tarea	
Numero de Tarea: 2	**Numero Historia:** 008
Nombre Tarea: Mantenimiento de repositorio archivos	
Tipo de Tarea: Programación	**Tiempo Estimado:** 8 horas
Fecha de Inicio: 04/11/2019	**Fecha Fin:** 06/11/2019
Programador Responsable: Bryan Xavier Inuca Chicaiza	
Descripción: Podremos agregar diferentes archivos en formato .doc y .pdf, estos archivos serán de las diferentes partes que se han obtenido documentos.	

Fuente: Propia

3.4. Diseño de la Aplicación

En esta fase se diseña la aplicación a partir de los requerimientos agrupados de las historias de usuarios anteriormente mencionados.

3.4.1. Arquitectura que tendrá la aplicación.

El aplicativo de escritorio denominado JCPD-Otavalo tendrá como arquitectura el siguiente modelo, estará basado en estructura Backend y Frontend permitiendo gestionar el contenido de las interfaces, lógica de negocio y acceso a datos para segmentar el correcto envió de información.

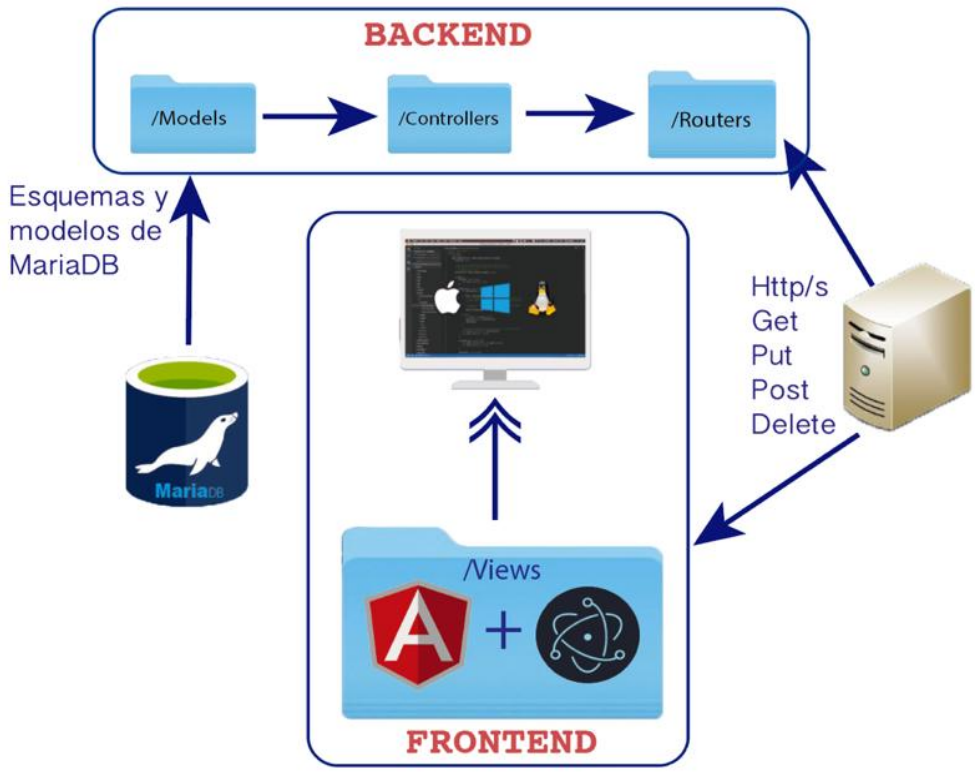


Fig. 25 Arquitectura FrontEnd/BackEnd
Fuente: Propia Autor 2019

La arquitectura antes mencionada permitirá gestionar los recursos de la aplicación tanto del cliente como del servidor, se indica en la siguiente tabla.

TABLA 3.59: Componentes de la estructura.

Objeto	Recurso
FrontEnd	Interfaces de usuario (HTML, CSS, JS). Lógica de gestión de denuncia (Usuario).
BackEnd	Lógica de servidor (Servidor). API Base de datos

Fuente: Propia

3.4.2. Diagrama de Base de datos

- **Diseño (Modelo Relacional)**

Para el desarrollo del aplicativo, en primer lugar, se tuvo que hacer el respectivo análisis de la información a manejar y luego podemos a diseñar el modelo de base de datos MariaDB, este trabajo se lo realizó con la herramienta sequelize ORM con NodeJs y express. Podemos observar la base de datos en la Fig. 26

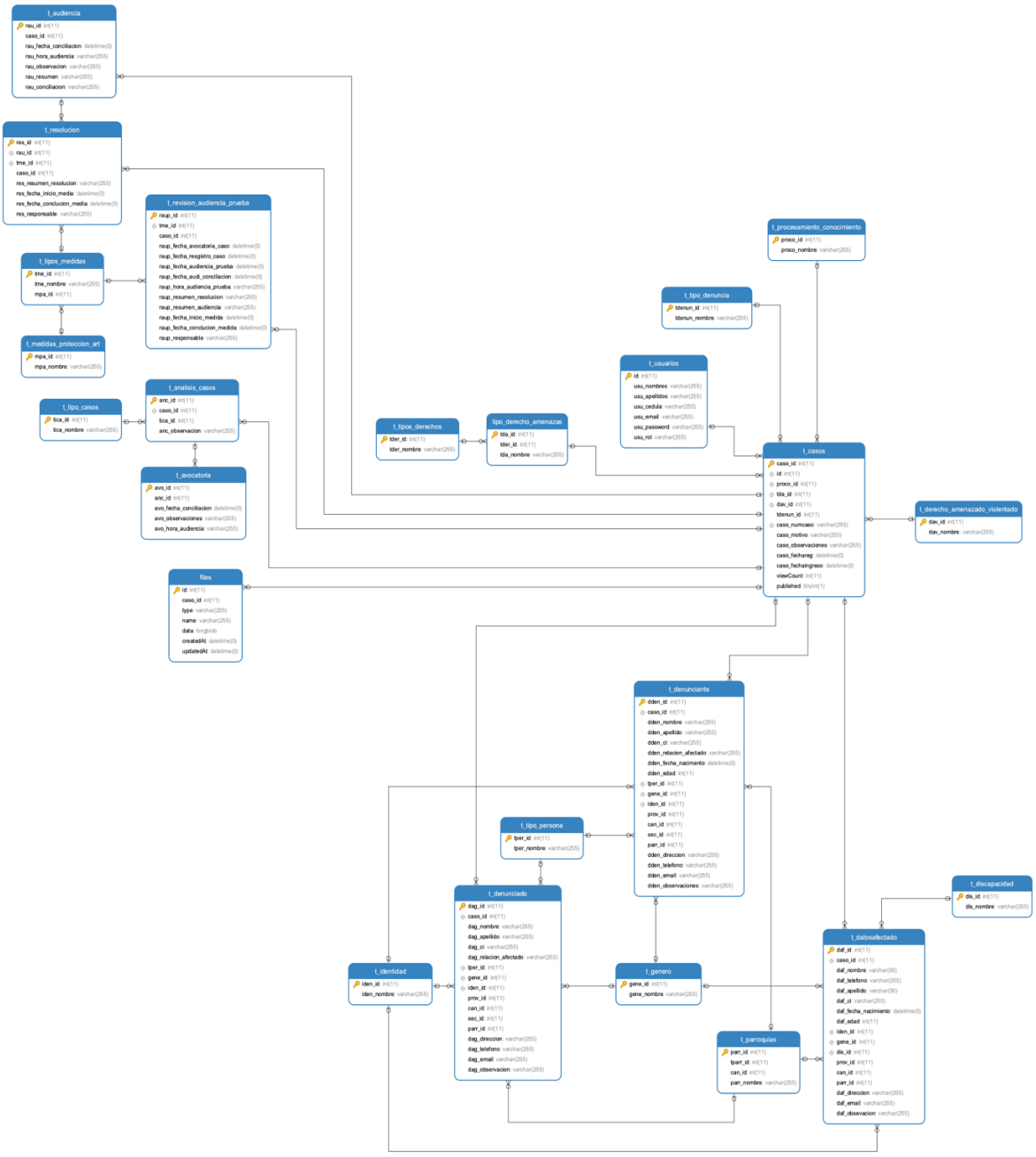


Fig. 26: Base de datos JCPD-OTAVALO
Fuente: Propia

3.4.3. Funcionamiento de la Aplicación

El aplicativo de escritorio será multiplataforma, constará de dos módulos destinados para diferentes tipos de usuarios como Administradores y Usuarios. El desarrollo será con herramientas Open Source (herramientas libres, que no tienen ningún costo).

3.4.4. Diseño de la Aplicación

Los módulos constaran de un diseño atractivo e intuitivo que sea de fácil manejo para los miembros de la JCPD, la aplicación constara de lo siguiente:

a) Módulo Administrador

El administrador tiene el acceso por medio de su email y contraseña, una vez dentro del sistema puede agregar usuarios, asignar roles y eliminar o deshabilitar un usuario. Fig.27

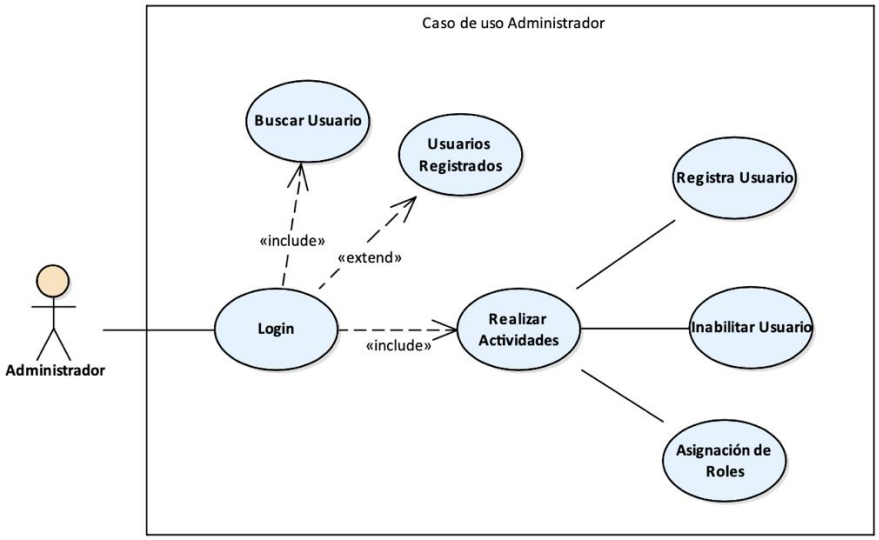


Fig. 27: Caso de uso, Módulo Administrador
Fuente: Propia

b) Modulo Usuario

El usuario podrá acceder al aplicativo de escritorio mediante el login con su email y contraseña, una vez dentro podrá realizar acciones como ingresar un nuevo caso, deshabilitar o habilitar un caso, búsqueda de casos por su nombre o fecha. El usuario podrá realizar el respectivo proceso de denuncia una vez registrado el caso también tiene disponible las funciones de agregar avocatoria, agregar audiencia, agregar resolución y también agregar archivos que pertenecen a cada caso. Una vez finalizado el proceso vamos a poder ver el caso con toda su información almacenada. Fig.28

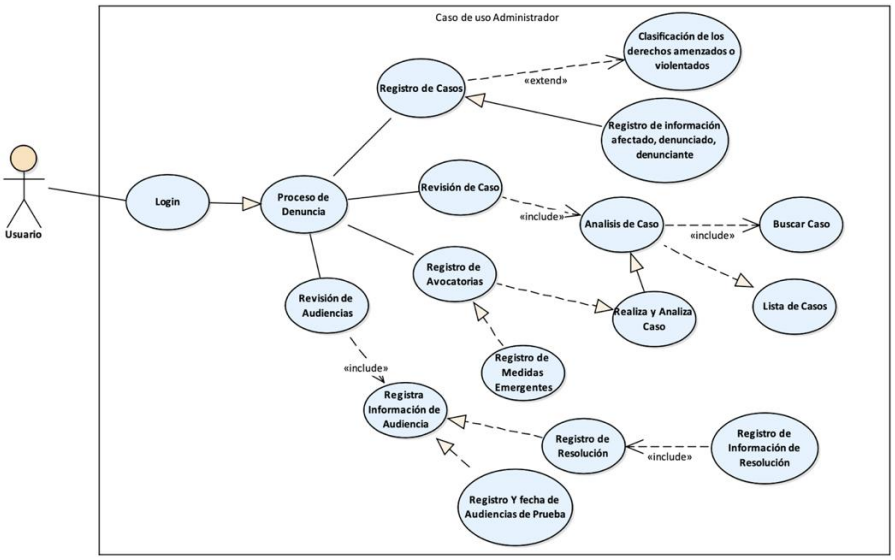


Fig. 28: Caso de uso Módulo Usuario
Fuente: Propia

3.5. Realización de pruebas

Este trabajo analiza la integración de las pruebas del sistema, o pruebas funcionales, dentro de un desarrollo basado en eXtreme Programming. Este trabajo también estudia los problemas que presenta el desarrollo de este tipo de pruebas tomando como base los artefactos generados por XP y ofrecemos una solución a estos problemas mediante un proceso de generación de pruebas del sistema aplicable a los usuarios. Se muestra las pruebas de iteracion desde la TABLA 3.60 hasta la TABLA 3.75

3.5.1. Prueba de Iteración I

- **Prueba de acceso al sistema de escritorio**

TABLA 3.60: Prueba de ingreso al aplicativo de escritorio

Prueba de Aceptación	
Caso de Prueba: Ingresar al aplicativo de escritorio	**Opción de Prueba:** Información general
Nro. De caso de Prueba: 01	**Nro. De historia de usuario:** 001
Nombre del caso de prueba: Acceso a la aplicación de escritorio.	
Descripción: Login, ingresar a la aplicación de escritorio	
Condiciones de ejecución: Acceso al servidor de la junta cantonal Otavalo. Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo	
Datos de entrada: Ingresar el correo electrónico y la contraseña.	
Pasos de ejecución: Ingresar a la aplicación de escritorio. Ingresar los datos solicitados.	
Resultados esperados: El aplicativo de escritorio se ejecuta correctamente.	
Evaluación: Correcto despliegue del aplicativo de escritorio.	

Fuente: Propia

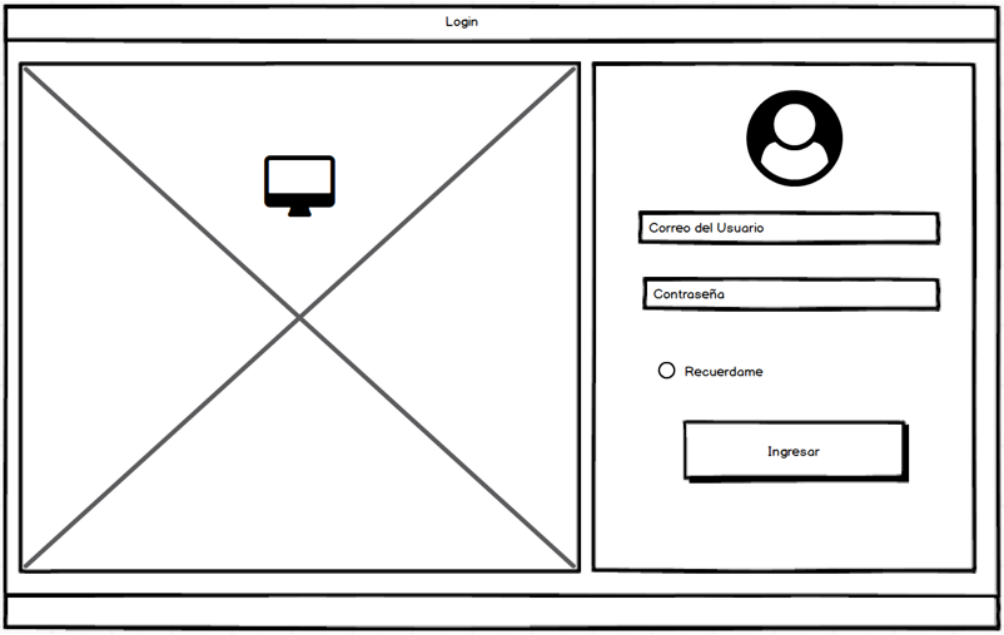


Fig. 29: Login de acceso al aplicativo.
Fuente: Propia

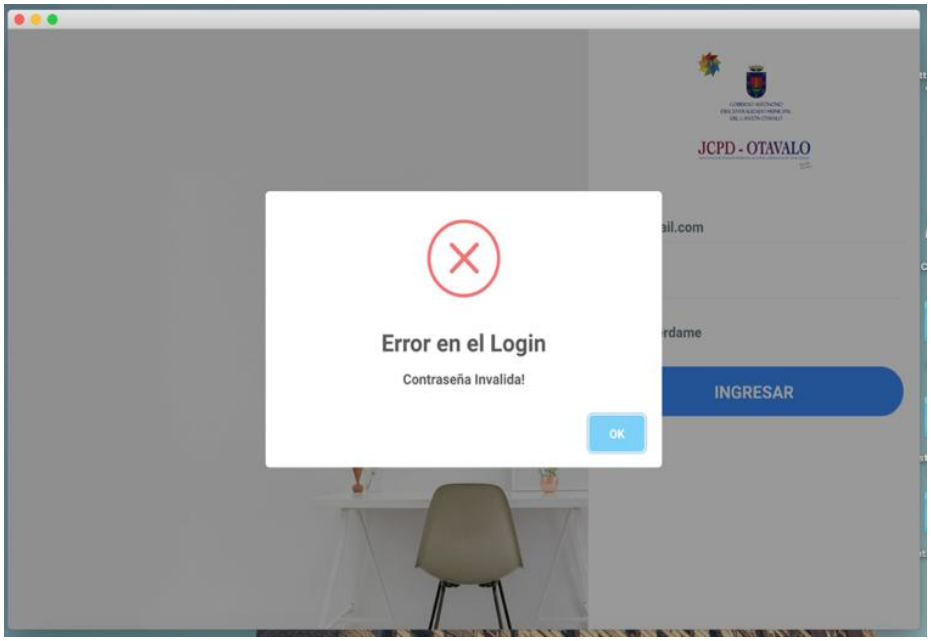


Fig. 30 Captura login aplicativo de escritorio
Fuente: Propia

- **Diseño de interfaz gráfica del aplicativo de escritorio**

TABLA 3.61: Prueba de interfaz del sistema

Prueba de Aceptación	
Caso de Prueba: Ver la interfaz grafica que tendrá el aplicativo de escritorio	**Opción de Prueba:** Diseño
Nro. De caso de Prueba: 02	**Nro. De historia de usuario:** 001
Nombre del caso de prueba: Interfaz del aplicativo.	
Descripción: Menú de opciones que tiene cada usuario y administrador dependiendo del rol.	
Condiciones de ejecución: Acceso al servidor de la junta cantonal Otavalo. Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña.	
Datos de entrada: No existen datos de entrada.	
Pasos de ejecución: Ingresar a la aplicación de escritorio.	
Resultados esperados: El aplicativo de escritorio se ejecuta correctamente y tiene una interfaz aceptable.	
Evaluación: Correcto despliegue del aplicativo de escritorio.	

Fuente: Propia

Interfaz de administrador:

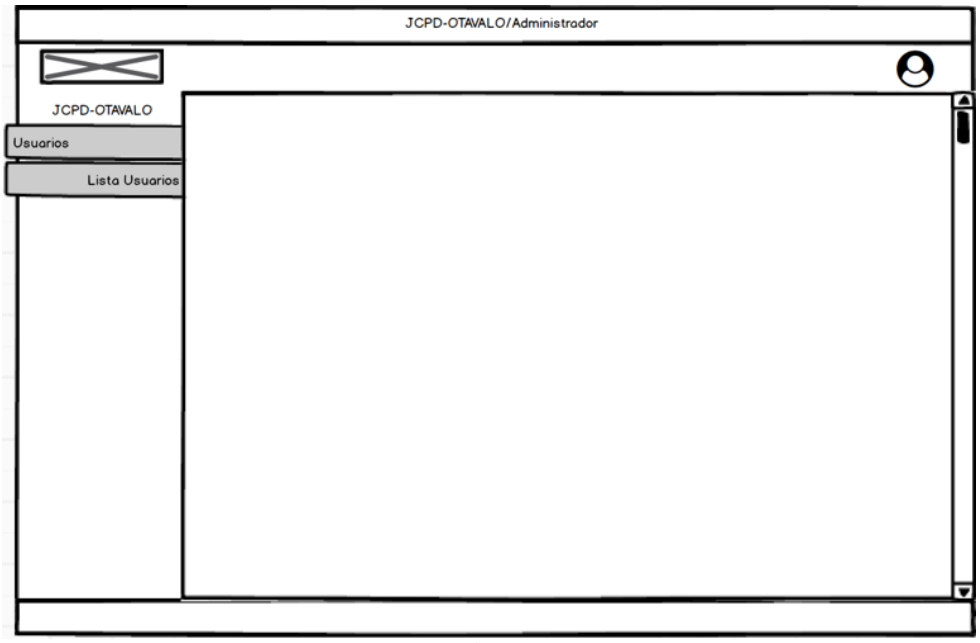


Fig. 31: Interfaz Administrador
Fuente: Propia

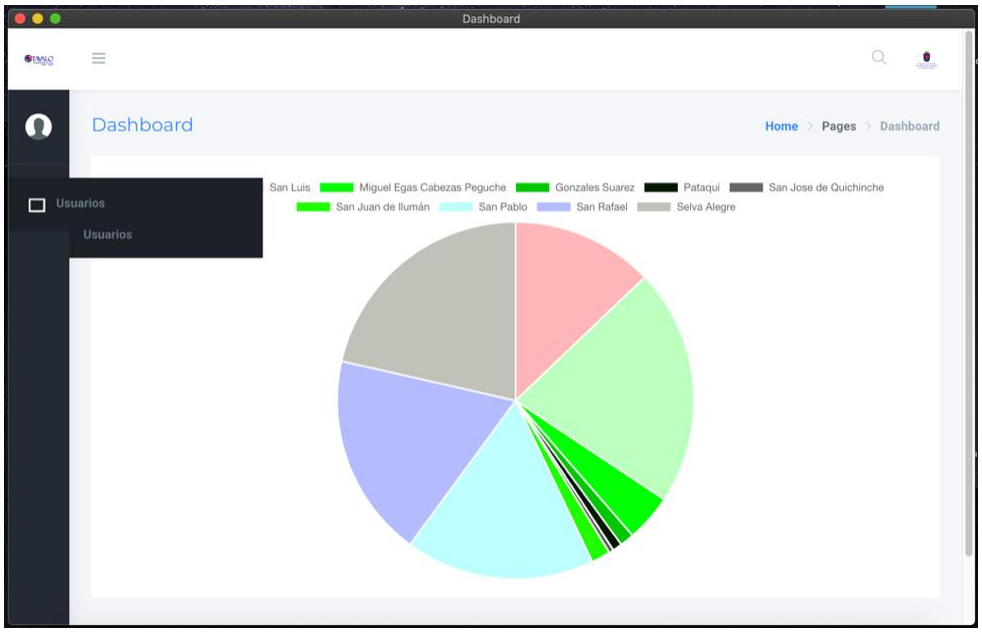


Fig. 32 Captura de la Intertfaz del Administrador.
Fuente: Propia

Interfaz de usuarios:

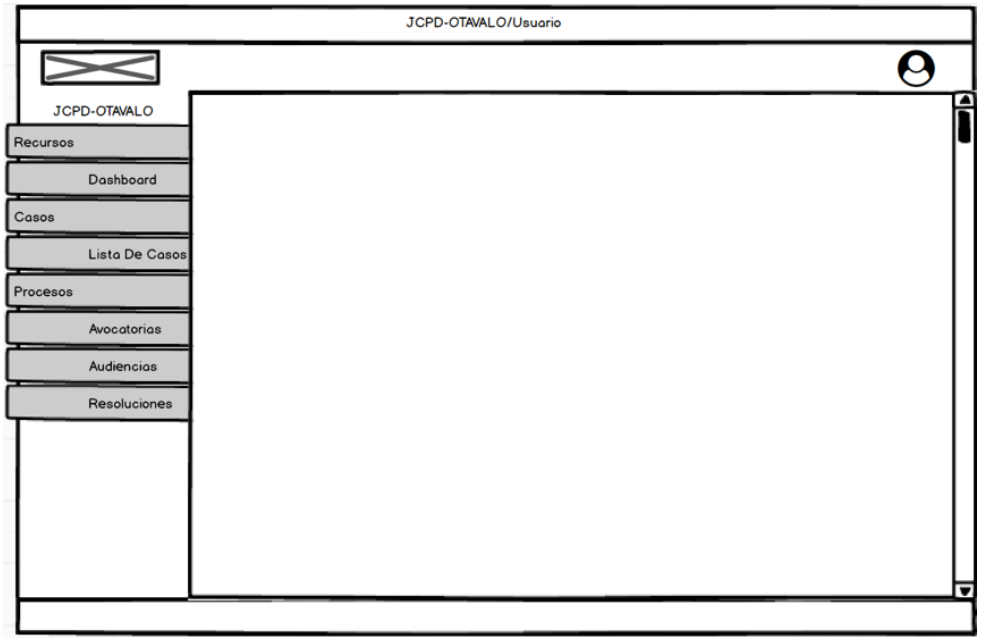


Fig. 33: Interfaz de usuarios
Fuente: Propia

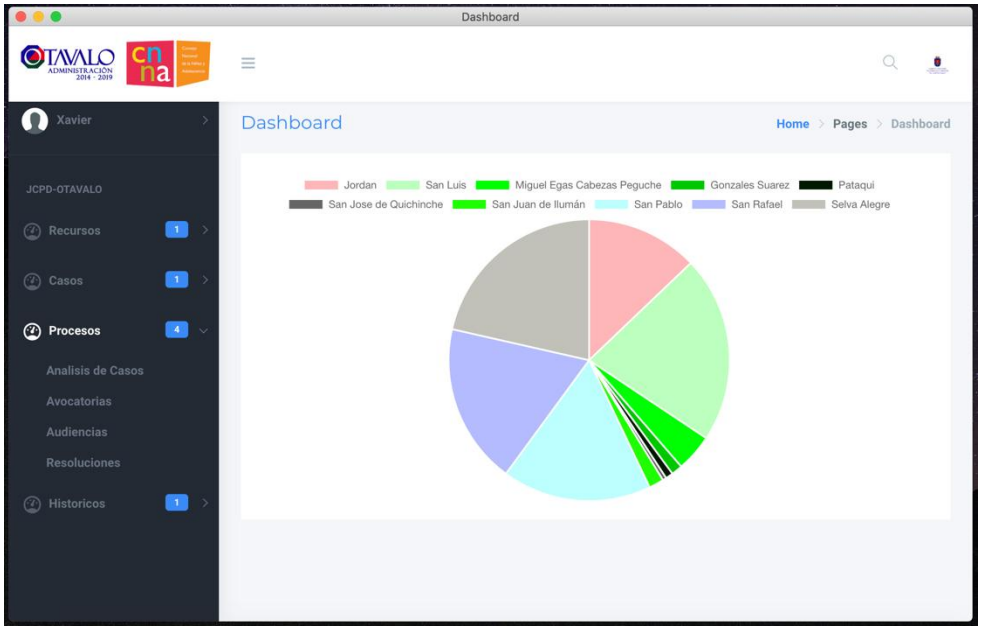


Fig. 34: Captura de interfaz de usuario
Fuente: Propia

- **Prueba de visualización de datos de los usuarios**

TABLA 3.62: Prueba de visualización datos de usuarios.

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 03	**Nro. De historia de usuario:** 002
Nombre del caso de prueba: Gestión de usuarios.	
Descripción: Visualización de datos de los usuarios.	
Condiciones de ejecución: Acceso al servidor de la junta cantonal Otavalo. Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener el rol de administrador.	
Datos de entrada: No existen datos de entrada.	
Pasos de ejecución:	
Ingresar a la aplicación de escritorio, y luego ir al menú de usuarios.	
Resultados esperados: La aplicación permite visualizar los datos de cada usuario y su rol.	
Evaluación: Correcto despliegue del aplicativo de escritorio.	

Fuente: Propia

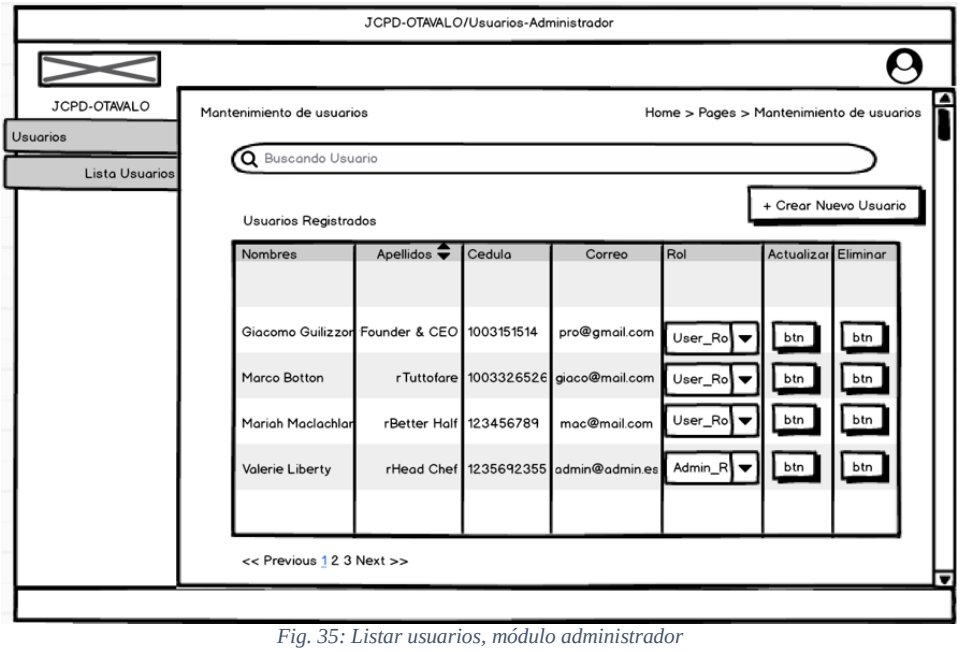


Fig. 35: Listar usuarios, módulo administrador

Fuente: Propia

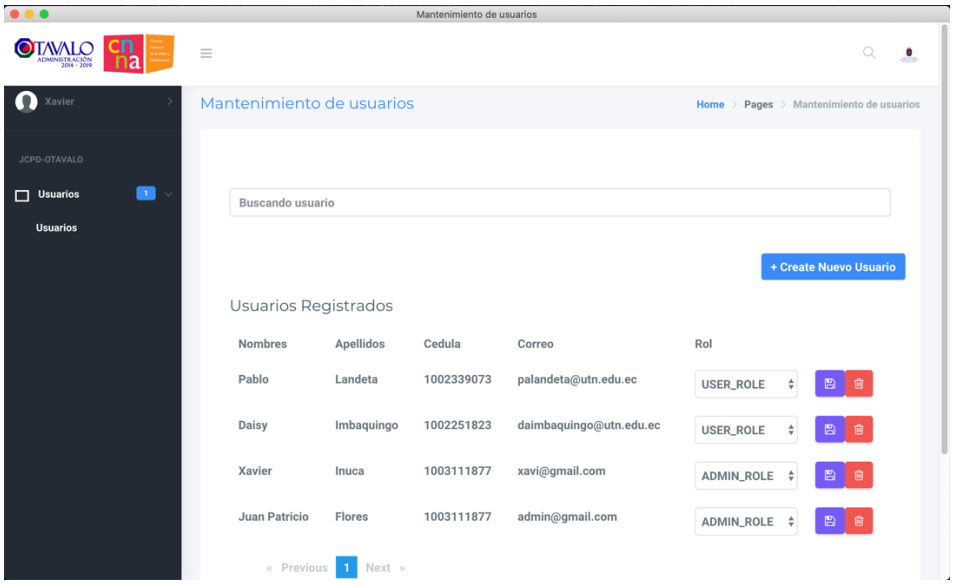


Fig. 36: Captura del modulo administrador con los usuarios y su CRUD

Fuente: Propia

- **Prueba de gestión de datos de usuarios**

TABLA 3.63: Prueba de gestión de datos de usuarios

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 04	**Nro. De historia de usuario:** 002
Nombre del caso de prueba: Gestión de usuarios.	
Descripción: Ingreso, edición de roles, eliminar un usuario.	
Condiciones de ejecución: Acceso al servidor de la junta cantonal Otavalo. Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener el rol de administrador.	
Datos de entrada: Ingresar datos pedidos en cada item.	
Pasos de ejecución: Ingresar a la aplicación de escritorio, luego ir al menú de usuarios, pulsar el botón agregar usuario.	
Resultados esperados: La aplicación permite ingresar, editar roles y eliminar un usuario.	
Evaluación: Correcto funcionamiento de la gestión de ítems.	

Fuente: Propia

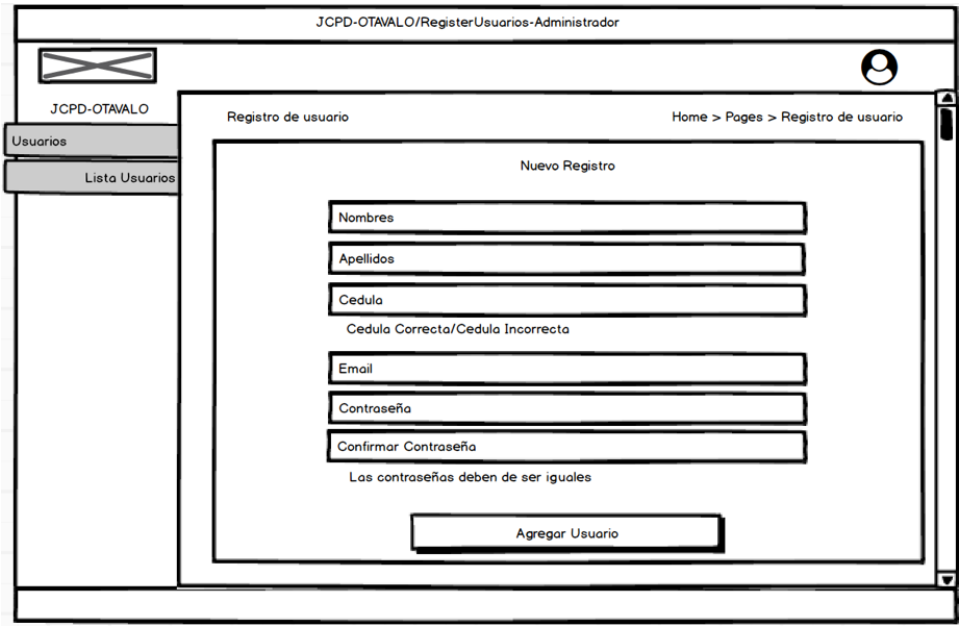


Fig. 37: Gestión de usuarios, módulo administrador

Fuente: Propia

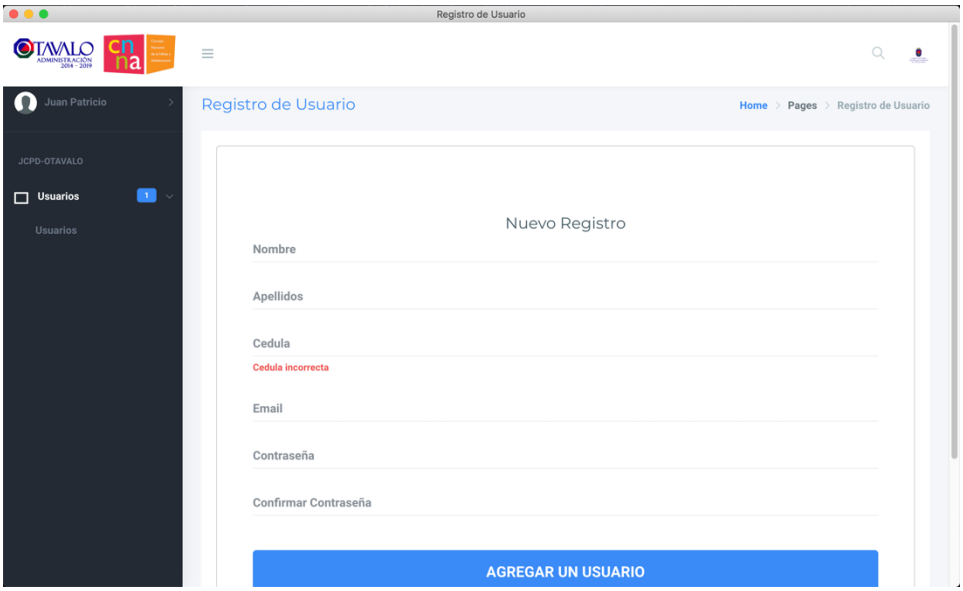


Fig. 38 : Captura de Gestión de usuarios, módulo administrador

Fuente: Propia

3.5.2. Pruebas de iteración II

- **Prueba de visualización de casos en el aplicativo.**

TABLA 3.64: Prueba de visualización de casos en el aplicativo

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 05	**Nro. De historia de usuario:** 003
Nombre del caso de prueba: Gestión de casos.	
Descripción: Visualización de casos y varias opciones para cada caso, búsqueda de casos, activar o desactivar un caso.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener el rol de usuario.	
Datos de entrada: No existen datos de entrada.	
Pasos de ejecución: Ingresar a la aplicación de escritorio, luego ir al menú de casos	
Resultados esperados: La aplicación permite visualizar los datos de la base de datos.	
Evaluación: Correcto funcionamiento del aplicativo de escritorio.	

Fuente: Propia

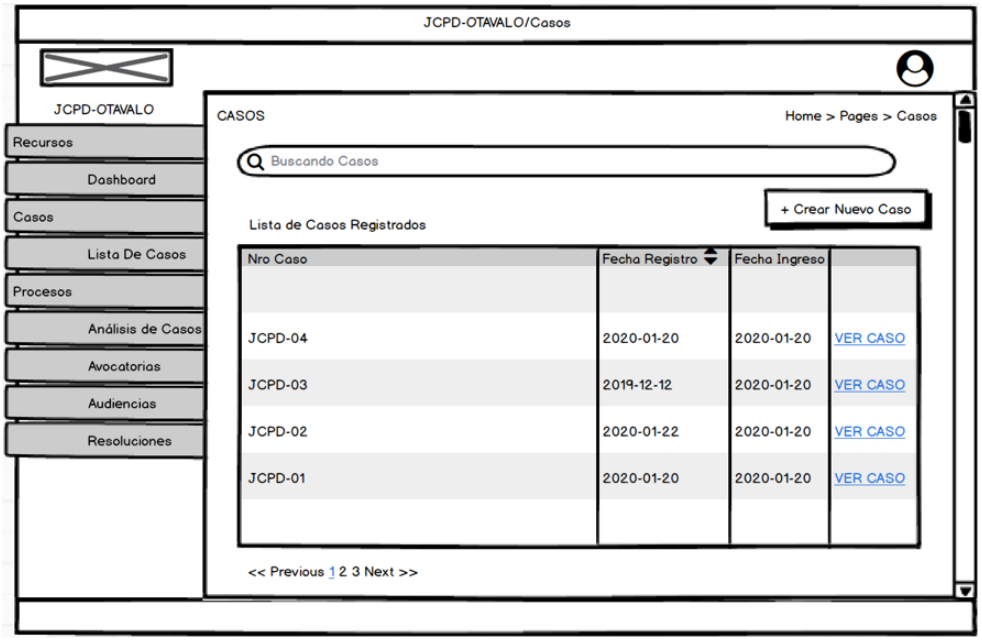


Fig. 39: Visualización de casos, modulo usuario

Fuente: Propia

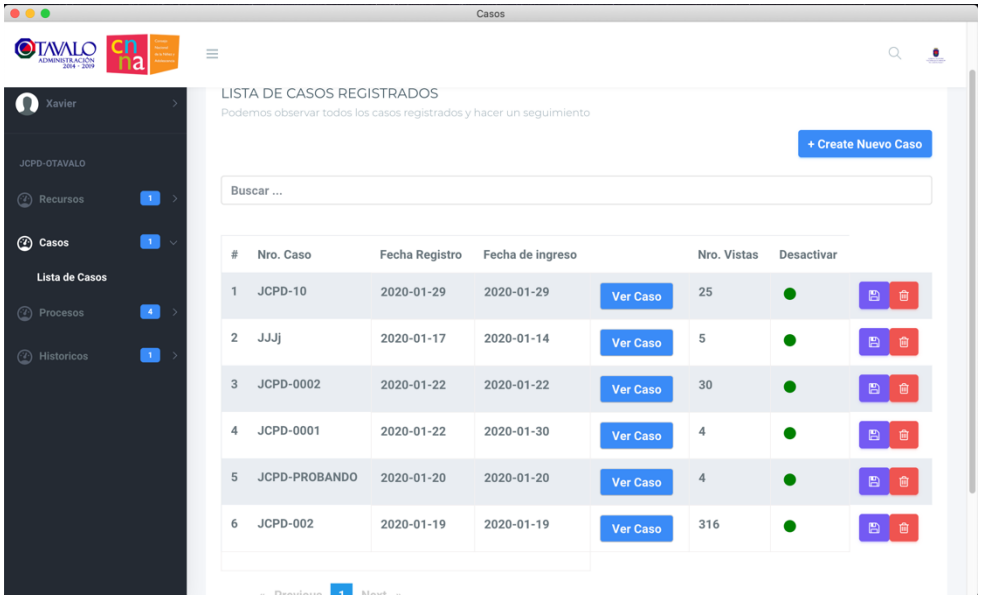


Fig. 40 : Captura de lista de casos activos, módulo usuario
Fuente: Propia

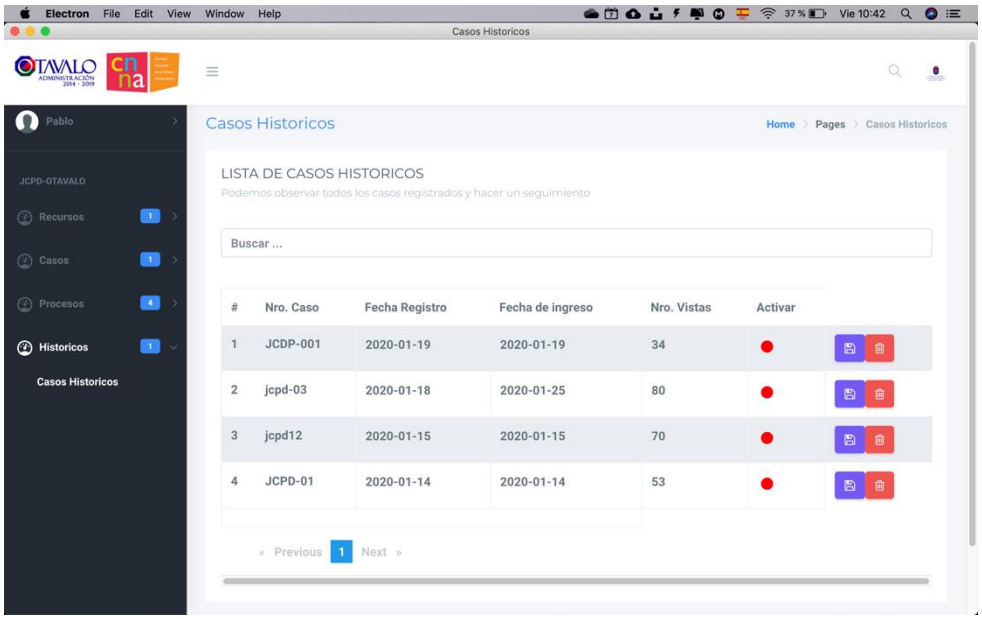


Fig. 41: Captura de lista de casos históricos, módulo usuario
Fuente: Propia

- **Prueba de gestión de datos para casos**

TABLA 3.65: Prueba de gestión de datos para casos

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 06	**Nro. De historia de usuario:** 003
Nombre del caso de prueba: Gestión de casos.	
Descripción: Gestión de casos, registro de un nuevo caso.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario.	
Datos de entrada: Ingresar datos pedidos en cada item.	
Pasos de ejecución: Ingresar a la aplicación de escritorio, luego ir al menú de casos, pulsar el botón agregar caso.	
Resultados esperados: La aplicación permite ingresar, editar y eliminar un caso.	
Evaluación: Correcto funcionamiento de la gestión de ítems.	

Fuente: Propia

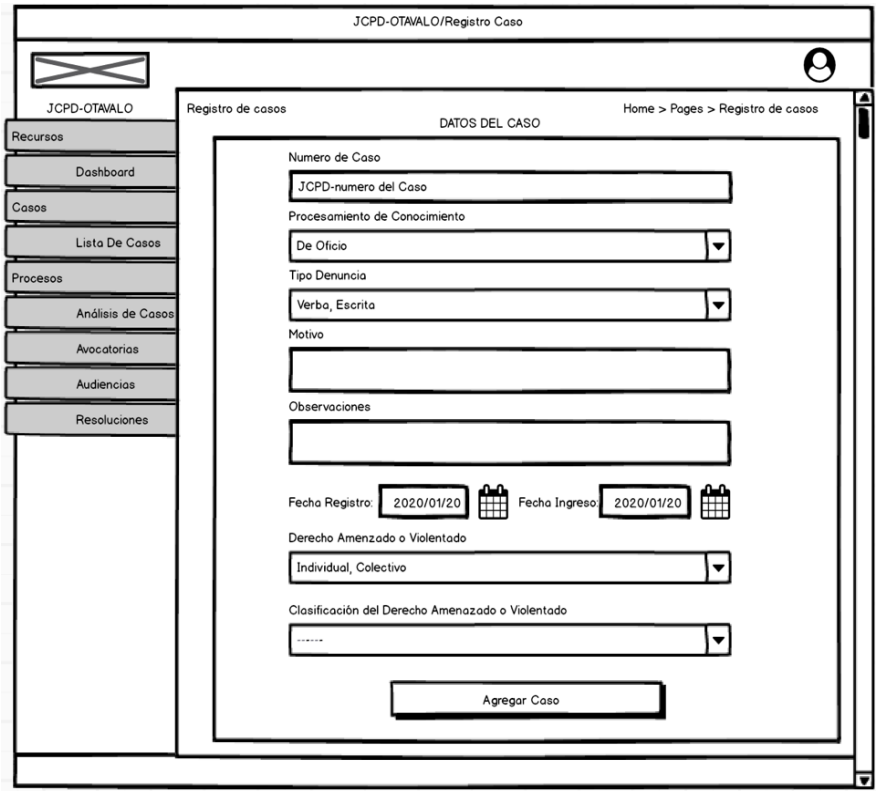


Fig. 42: Gestión de casos, modulo usuario
Fuente: Propia

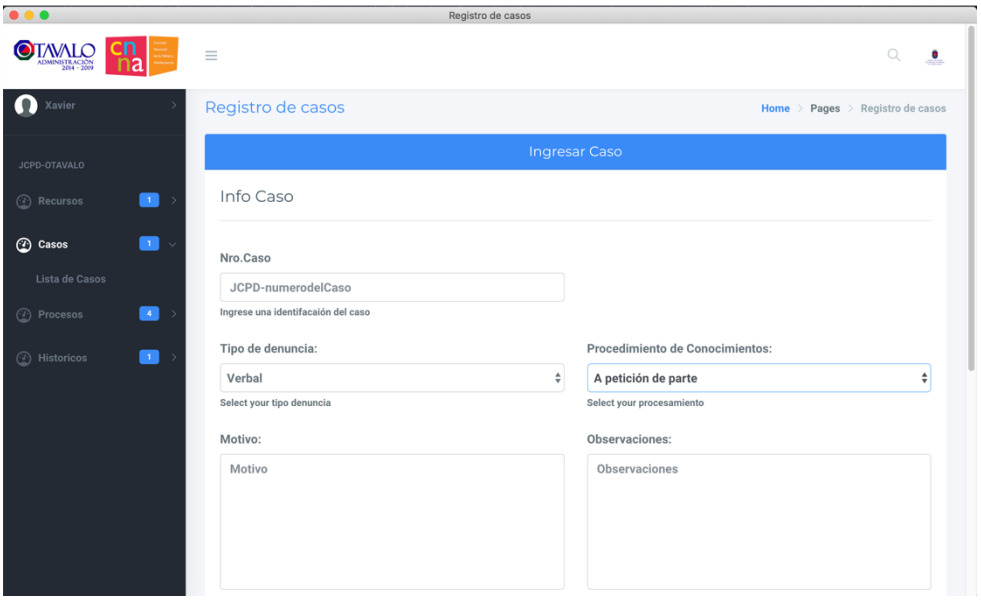


Fig. 43: Captura de registrar casos, módulo usuario
Fuente: Propia

- **Prueba de gestión de datos para afectados.**

TABLA 3.66: Prueba de gestión de datos para afectados

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 07	**Nro. De historia de usuario:** 003
Nombre del caso de prueba: Gestión de afectados.	
Descripción: Gestión de afectados, registro de un nuevo afectado.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado un caso.	
Datos de entrada: Ingresar datos pedidos en cada item.	
Pasos de ejecución: Ingresar a la aplicación de escritorio, luego ir al menú de casos, luego de agregar un caso nos pide agregar un afectado.	
Resultados esperados: La aplicación permite ingresar datos de un afectado.	
Evaluación: Correcto funcionamiento de la gestión de ítems.	

Fuente: Propia

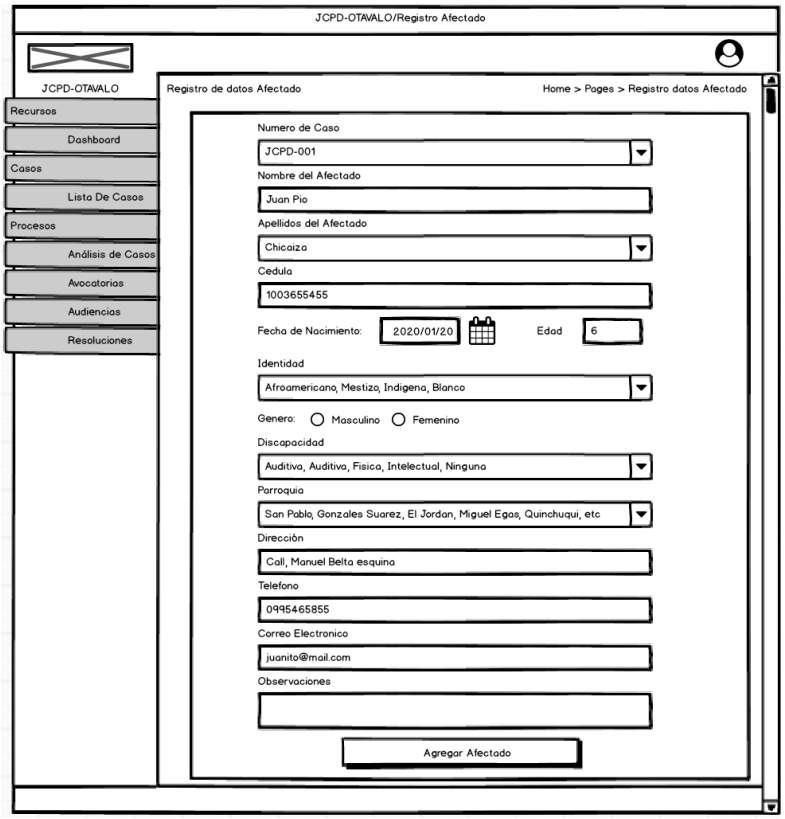


Fig. 44: Gestión de afectado, módulo usuario

Fuente: Propia

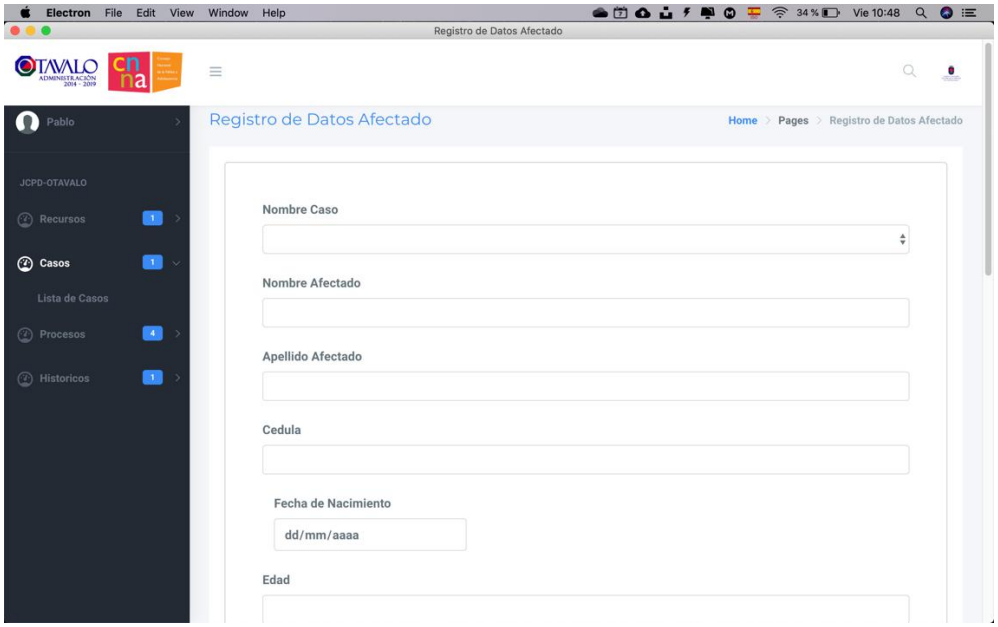


Fig. 45: Captura gestión de afectado, módulo usuario.
Fuente: Propia

- **Prueba de gestión de datos para denunciantes**

TABLA 3.67: Prueba de gestión de datos para denunciantes

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 08	**Nro. De historia de usuario:** 003
Nombre del caso de prueba: Gestión de denunciantes.	
Descripción: Gestión de denunciantes, registro de un nuevo denunciante.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado un caso.	
Datos de entrada: Ingresar datos pedidos en cada item.	
Pasos de ejecución: Ingresar a la aplicación de escritorio, luego ir al menú de casos, luego de agregar un caso nos pide agregar un denunciante.	
Resultados esperados: La aplicación permite ingresar datos de un afectado.	
Evaluación: Correcto funcionamiento de la gestión de ítems.	

Fuente: Propia

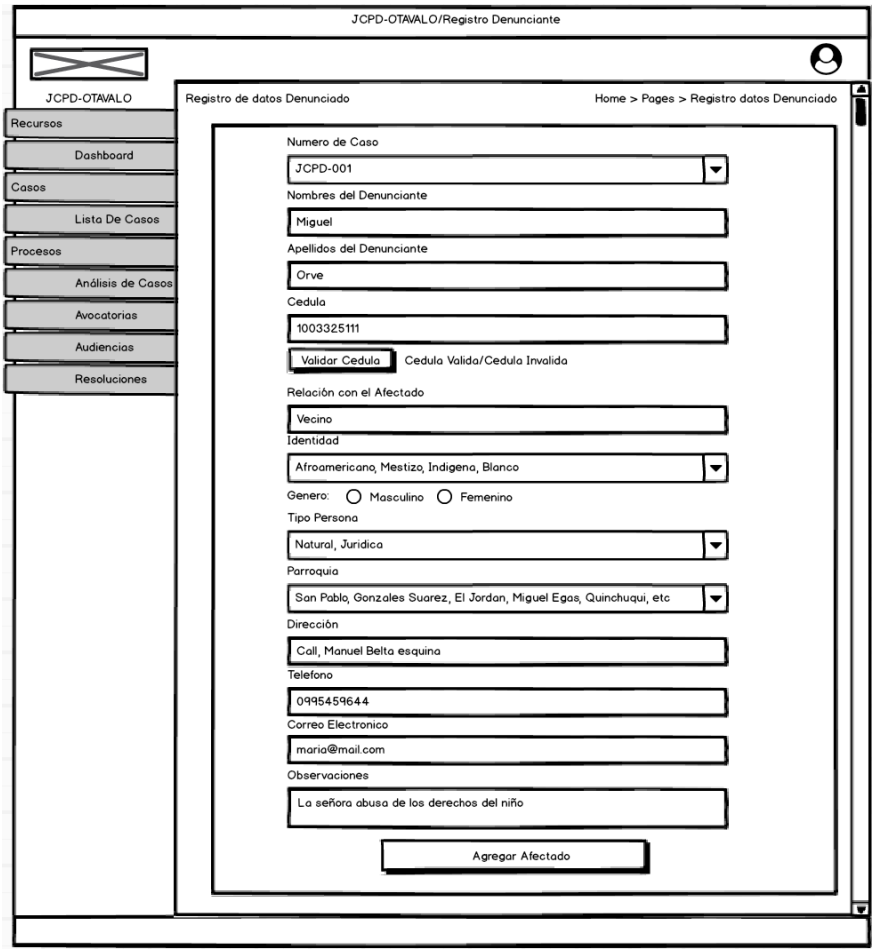


Fig. 46: Gestión de denunciante, módulo usuario.
Fuente: Propia

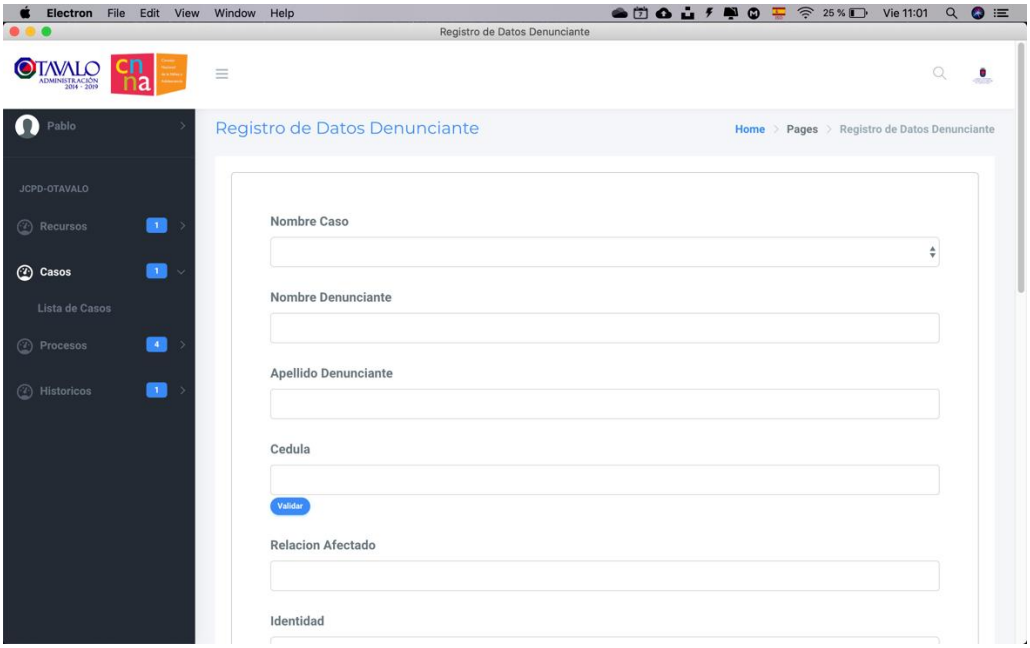


Fig. 47: Captura de registro de un denunciante, modulo usuario
Fuente: Propia

- **Prueba de gestión de datos para denunciados**

TABLA 3.68: Prueba de gestión de datos para denunciados

Prueba de Aceptación

Caso de Prueba: Gestionar datos en el aplicativo. **Opción de Prueba:** Datos
Nro. De caso de Prueba: 09 **Nro. De historia de usuario:** 003
Nombre del caso de prueba: Gestión de denunciados.
Descripción: Gestión de denunciados, registro de un nuevo denunciado.
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado un caso.
Datos de entrada: Ingresar datos pedidos en cada item.
Pasos de ejecución:
Ingresar a la aplicación de escritorio,
luego ir al menú de casos,
luego de agregar un caso nos pide agregar un denunciado.
Resultados esperados: La aplicación permite ingresar datos de un denunciado.
Evaluación: Correcto funcionamiento de la gestión de ítems.

Fuente: Propia

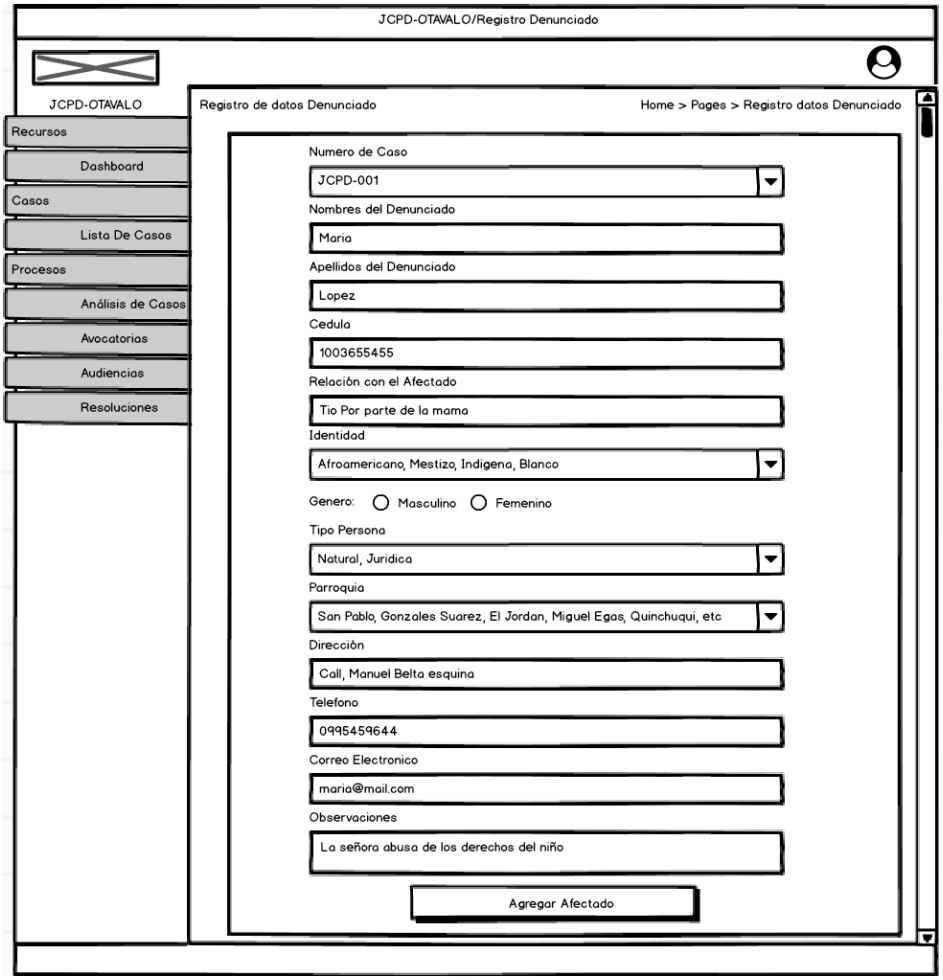


Fig. 48: Gestión de denunciado
Fuente: Propia

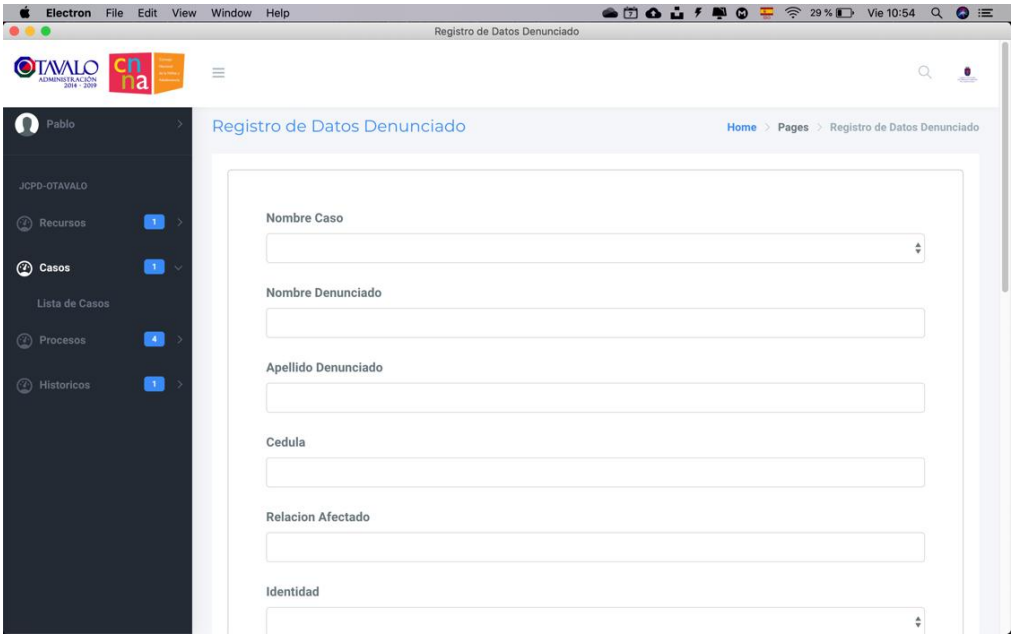


Fig. 49: Captura de gestión de denunciado, módulo usuario.
Fuente: Propia

- **Prueba de visualización de toda la información del caso.**

TABLA 3.69: Prueba de visualización de toda la información del caso

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 10	**Nro. De historia de usuario:** 003
Nombre del caso de prueba: Gestión de denunciados.	
Descripción: Visualización de datos registrados y que pertenecen a un caso.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado un caso.	
Datos de entrada: No existen datos de entrada.	
Pasos de ejecución: Ingresar a la aplicación de escritorio y en el menú casos, Pulsar el botón ver caso.	
Resultados esperados: La aplicación permite ver toda la información necesaria de un caso.	
Evaluación: Correcto despliegue del aplicativo de escritorio.	

Fuente: Propia

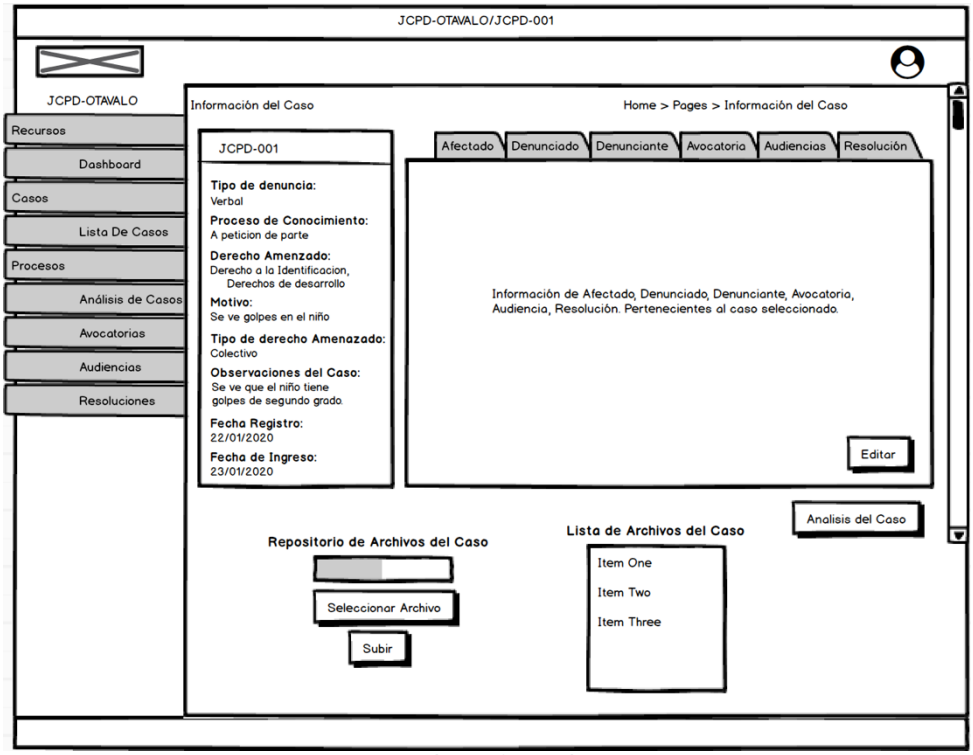


Fig. 50: Prueba de visualización de información del caso, modulo usuario
Fuente: Propia

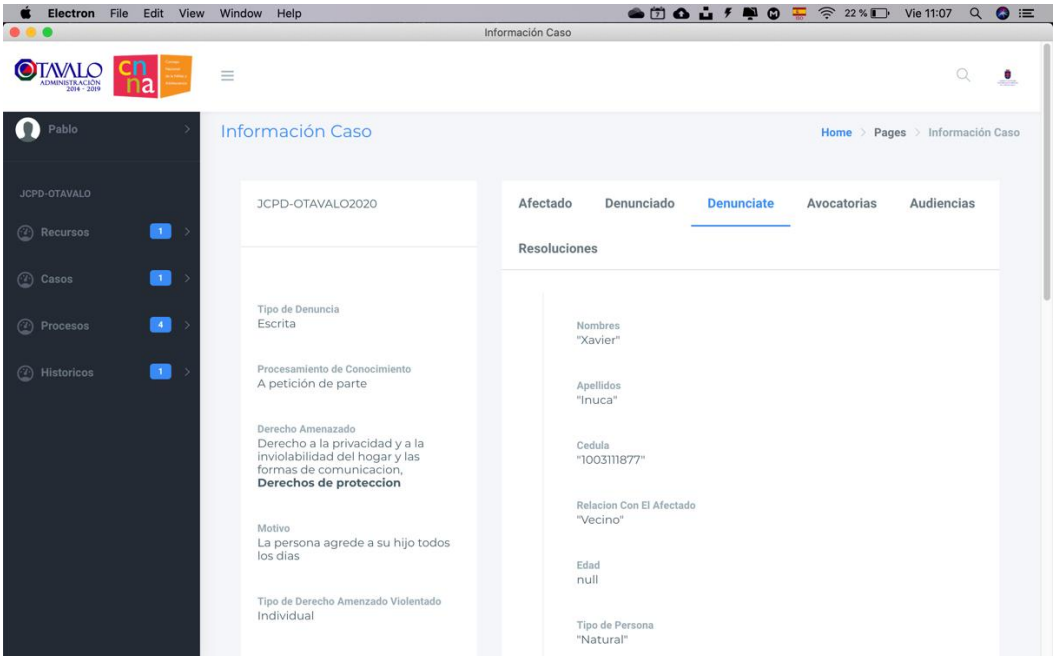


Fig. 51: Captura de visualización de toda la información que tiene cada caso, módulo usuario
Fuente: Propia

- **Prueba de visualización análisis de casos.**

TABLA 3.70: Prueba de visualización de análisis de casos

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 11	**Nro. De historia de usuario:** 004
Nombre del caso de prueba: Visualización de análisis de casos.	
Descripción: Visualización de análisis de casos registrados y que pertenecen a un caso.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado un caso.	
Datos de entrada: No existen datos de entrada.	
Pasos de ejecución:	
Ingresar a la aplicación de escritorio y e ir al menú análisis de casos	
Resultados esperados: La aplicación permite ver los casos registrados para poder analizarlos.	
Evaluación: Correcto despliegue del aplicativo de escritorio.	

Fuente: Propia

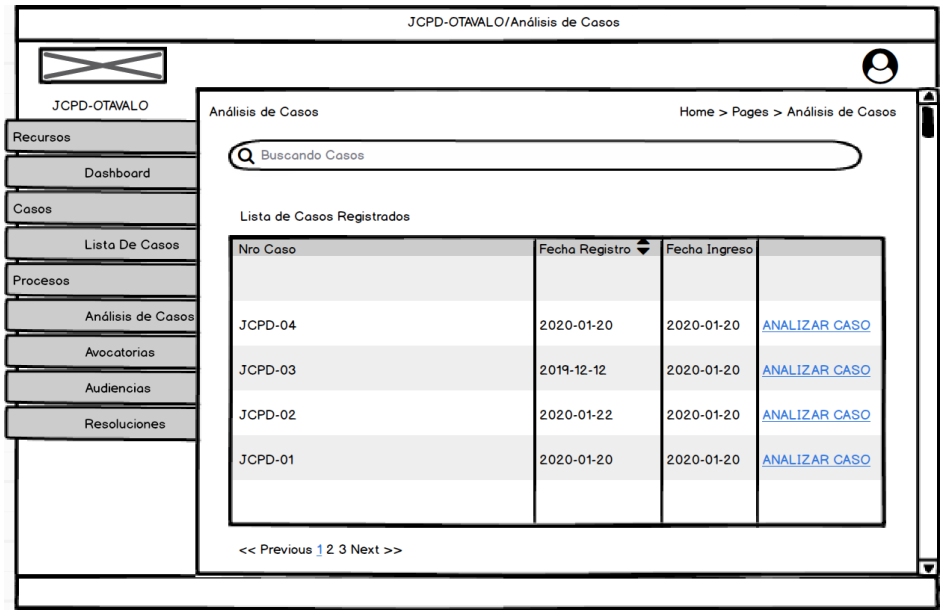


Fig. 52: Prueba de visualización de análisis casos

Fuente: Propia

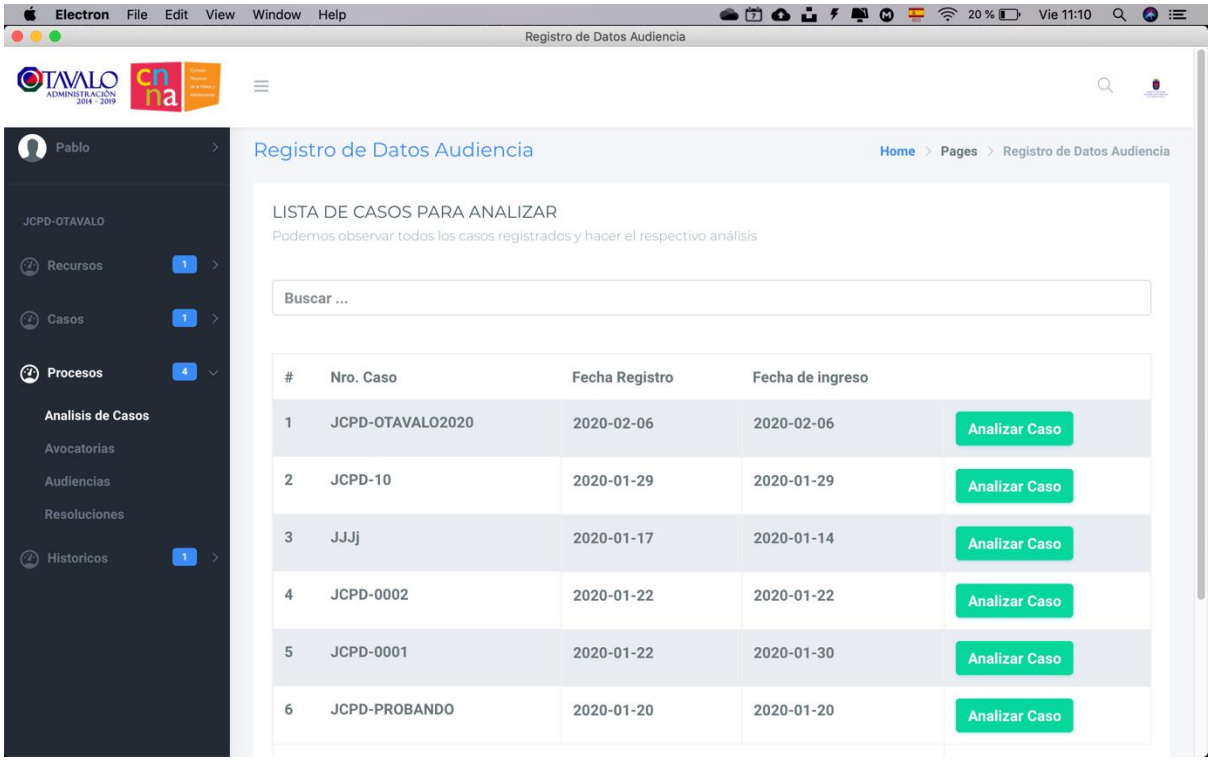


Fig. 53: Captura de visualización de casos para analizar, módulo usuario
Fuente: Propia

- **Prueba de gestión de datos para análisis de casos**

TABLA 3.71: Prueba de gestión de análisis de casos

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 12	**Nro. De historia de usuario:** 004
Nombre del caso de prueba: Gestión de análisis de casos.	
Descripción: Gestión de análisis de casos, registro de un nuevo análisis de caso.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado un caso el cual va a ser analizado.	
Datos de entrada: Ingresar datos pedidos en cada item.	
Pasos de ejecución: Ingresar a la aplicación de escritorio, luego ir al menú de análisis de casos, presionar botón analizar caso.	
Resultados esperados: La aplicación permite ingresar datos de un análisis de caso.	
Evaluación: Correcto funcionamiento de la gestión de ítems.	

Fuente: Propia

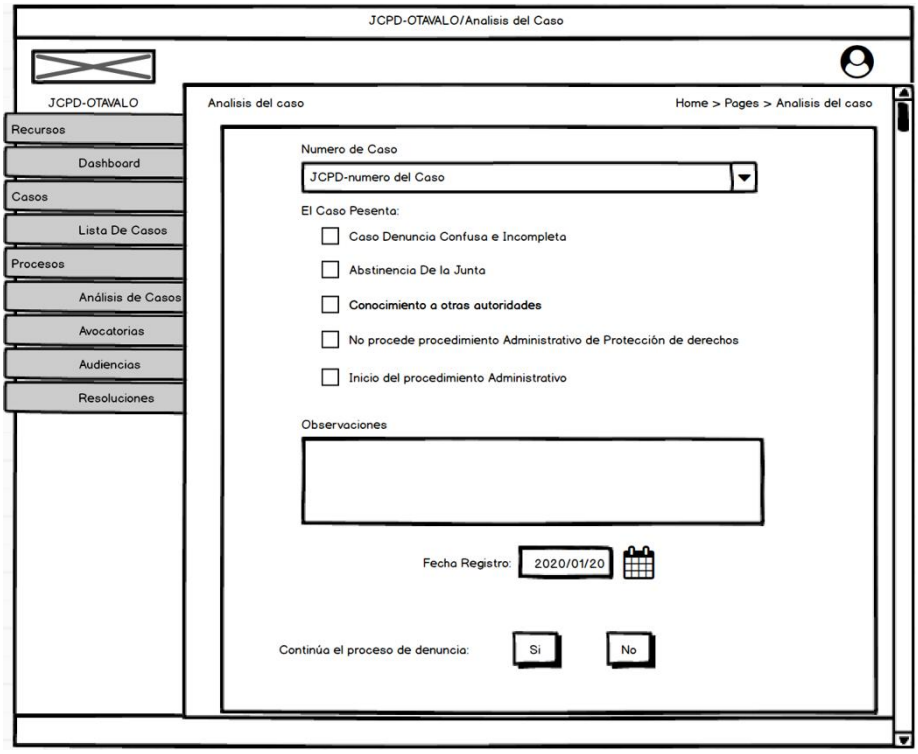


Fig. 54: Prueba de gestión de análisis de casos
Fuente: Propia

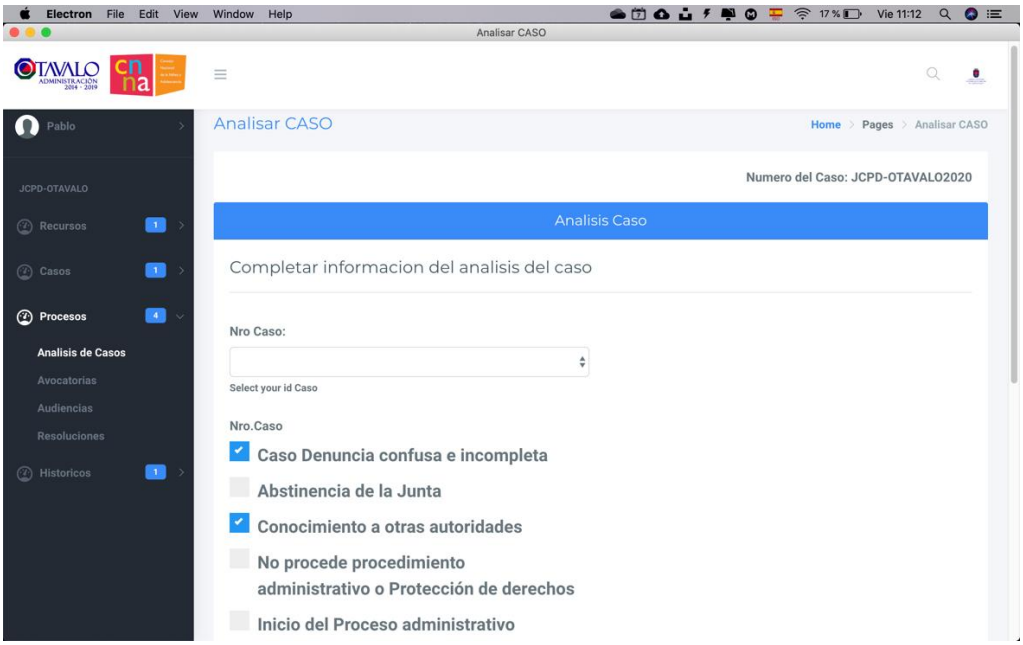


Fig. 55: Captura de gestión de análisis de casos, módulo usuario
Fuente: Propia

- **Prueba de gestión de avocatoria**

TABLA 3.72: Prueba de gestión de avocatoria

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 13	**Nro. De historia de usuario:** 005
Nombre del caso de prueba: Gestión de avocatoria.	
Descripción: Gestión de avocatoria, registro de una nueva avocatoria que pertenece a un análisis de caso luego de haberlo ingresado.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado un análisis de caso el cual va a tener la avocatoria.	
Datos de entrada: Ingresar datos pedidos en cada item.	
Pasos de ejecución: Ingresar a la aplicación de escritorio, luego ir al menú de avocatoria.	
Resultados esperados: La aplicación permite ingresar datos de una avocatoria.	
Evaluación: Correcto funcionamiento de la gestión de ítems.	

Fuente: Propia

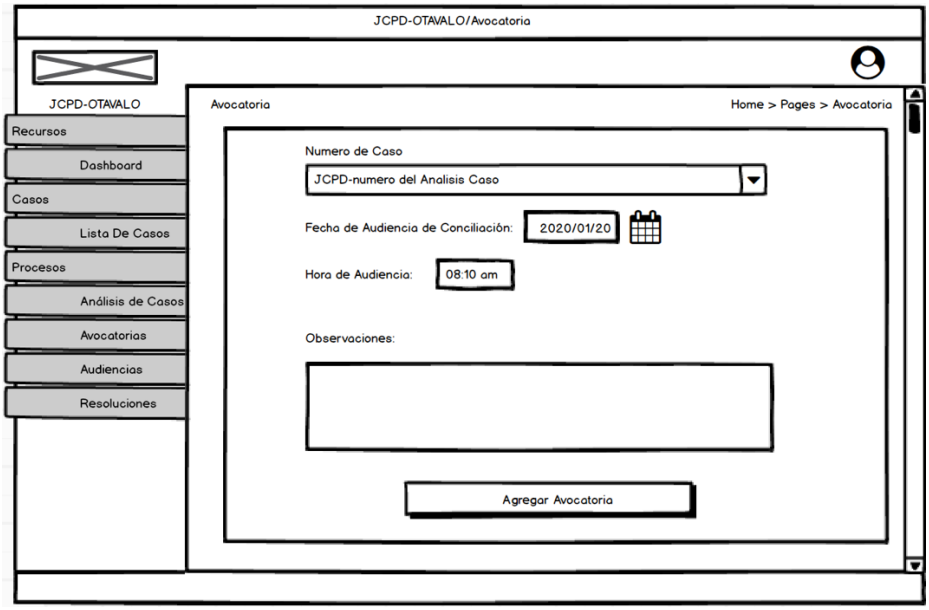


Fig. 56: Prueba de gestión de avocatoria

Fuente: Propia

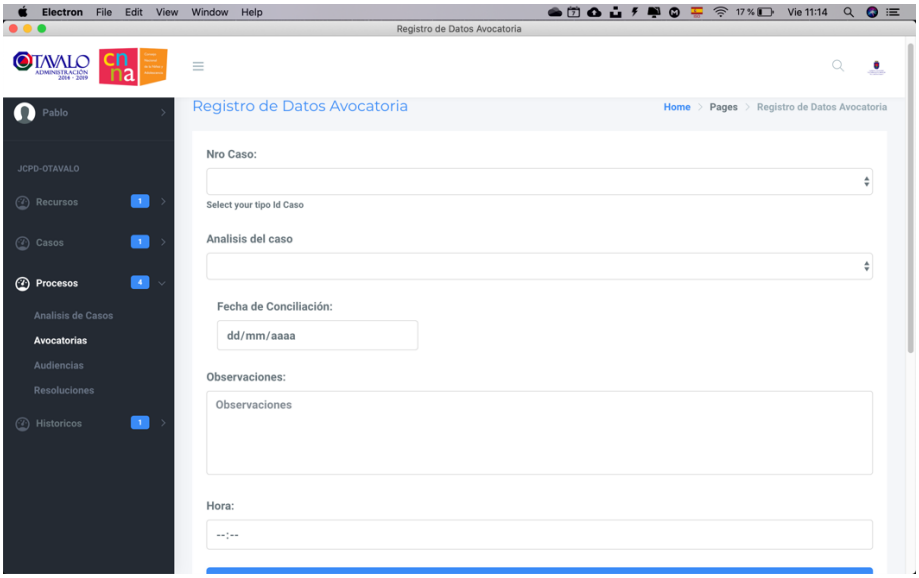


Fig. 57: Captura de gestión de avocatorias, módulo usuario
Fuente: Propia

- **Prueba de gestión de audiencia**

TABLA 3.73: Prueba de gestión de audiencia

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 14	**Nro. De historia de usuario:** 006
Nombre del caso de prueba: Gestión de audiencia.	
Descripción: Gestión audiencia, registro de una nueva audiencia que pertenece a un caso, luego de haber pasado el proceso de avocatoria.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado una avocatoria y un caso el cual va a tener la audiencia.	
Datos de entrada: Ingresar datos pedidos en cada item.	
Pasos de ejecución: Ingresar a la aplicación de escritorio, luego ir al menú de audiencia.	
Resultados esperados: La aplicación permite ingresar datos de una audiencia.	
Evaluación: Correcto funcionamiento de la gestión de ítems.	

Fuente: Propia

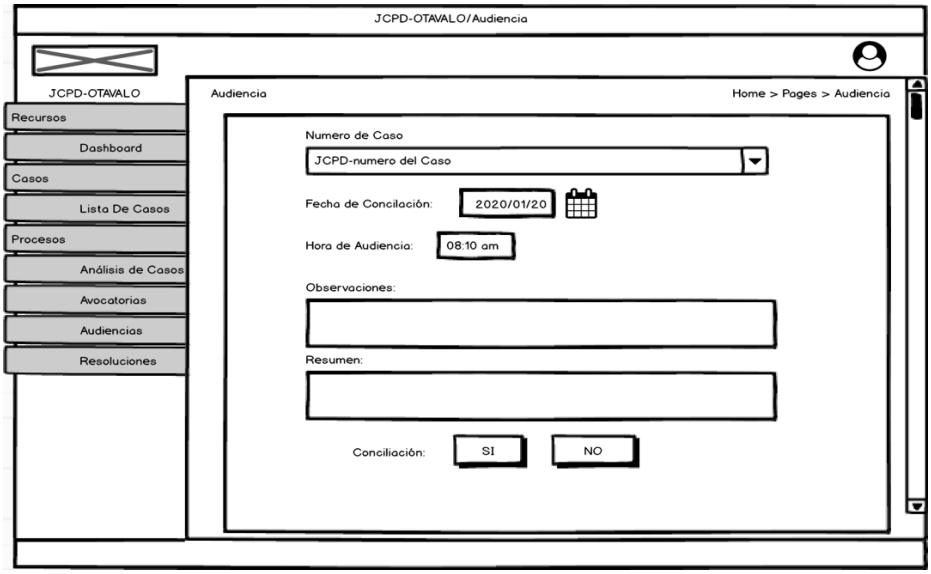


Fig. 58: Prueba de gestión de audiencia
Fuente: Propia

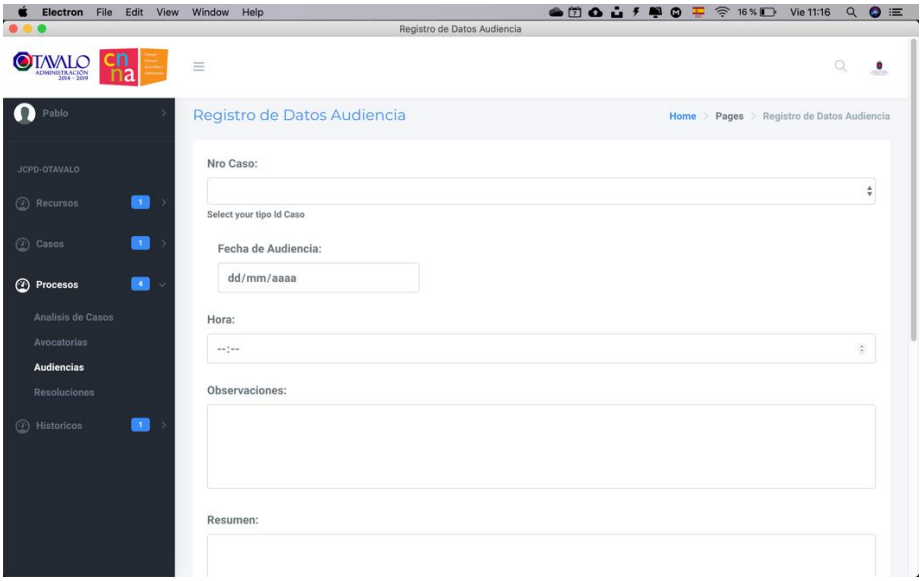


Fig. 59: Captura de gestión de audiencias, módulo usuario
Fuente: Propia

- **Prueba de gestión de resolución**

TABLA 3.74: Prueba de gestión de resolución

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 15	**Nro. De historia de usuario:** 0007
Nombre del caso de prueba: Gestión de resolución.	
Descripción: Gestión resolución, registro de una nueva resolución que pertenece a un caso, luego de haber pasado el proceso de audiencia, y teniendo como conclusión una conciliación de las partes. Damos por finalizado el proceso de denuncia y cerramos el caso.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado una audiencia y un caso el cual va a tener la resolución.	
Datos de entrada: Ingresar datos pedidos en cada item.	
Pasos de ejecución: Ingresar a la aplicación de escritorio, luego ir al menú de resolución.	
Resultados esperados: La aplicación permite ingresar datos de una resolución.	
Evaluación: Correcto funcionamiento de la gestión de ítems.	

Fuente: Propia

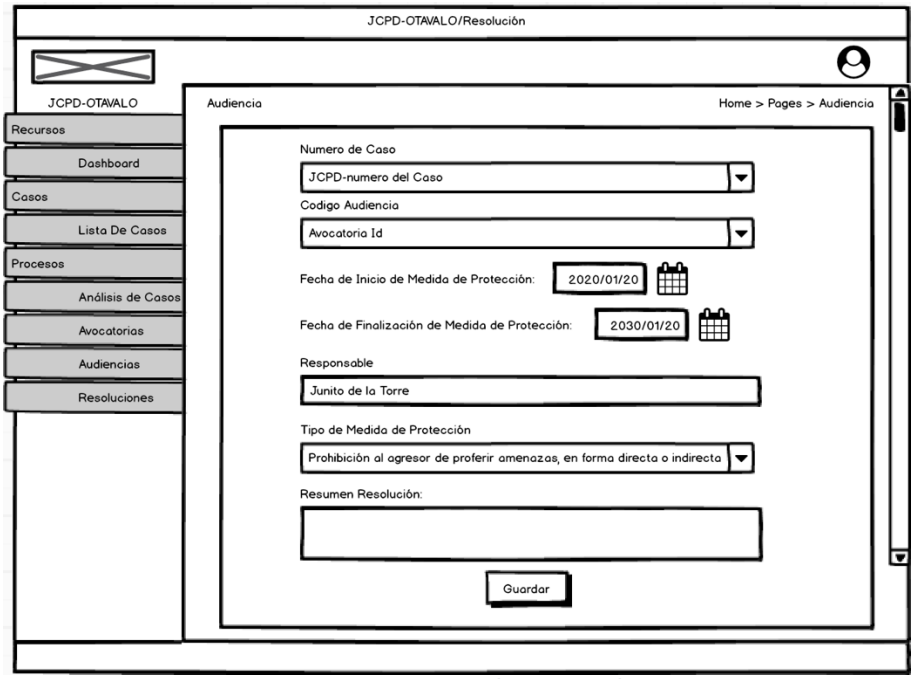


Fig. 60: Prueba de gestión de resolución
Fuente: Propia

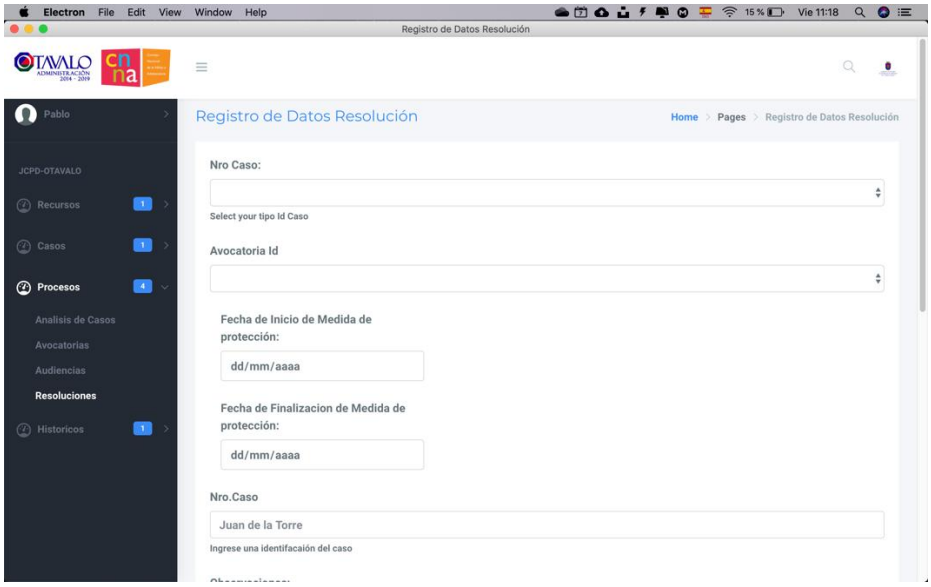


Fig. 61: Captura de gestión de resolución, modulo usuario
Fuente: Propia

- **Prueba de gestión de archivos.**

TABLA 3.75: Prueba de gestión de archivos

Prueba de Aceptación	
Caso de Prueba: Gestionar datos en el aplicativo.	**Opción de Prueba:** Datos
Nro. De caso de Prueba: 16	**Nro. De historia de usuario:** 008
Nombre del caso de prueba: Gestión de archivos.	
Descripción: Visualización de archivos que pertenecen a un caso agregados a la base de datos, opción de agregar nuevos archivos.	
Condiciones de ejecución: Tener instalado la aplicación de escritorio en un computador del GAD Municipal de Otavalo, haber accedido al sistema con su correo y contraseña, tener rol de usuario, tener agregado un caso.	
Datos de entrada: Agregar archivos a la base de datos.	
Pasos de ejecución:	
Ingresar a la aplicación de escritorio y e ir al caso.	
Resultados esperados: La aplicación permite ver los archivos agregados pertenecientes a cada caso y agregar nuevos archivos.	
Evaluación: Correcto despliegue del aplicativo de escritorio.	

Fuente: Propia

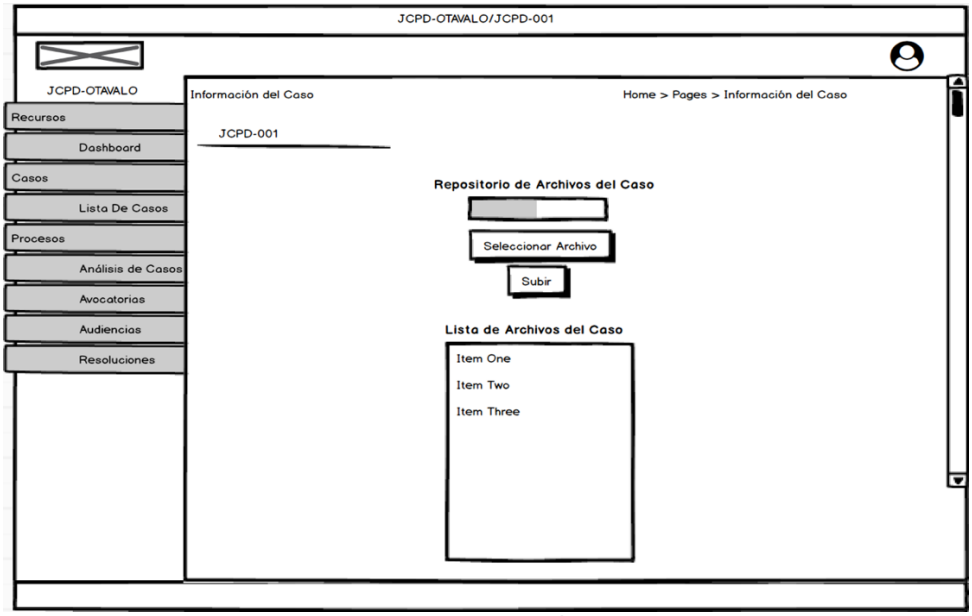


Fig. 62: Prueba de visualización de archivos

Fuente: Propia

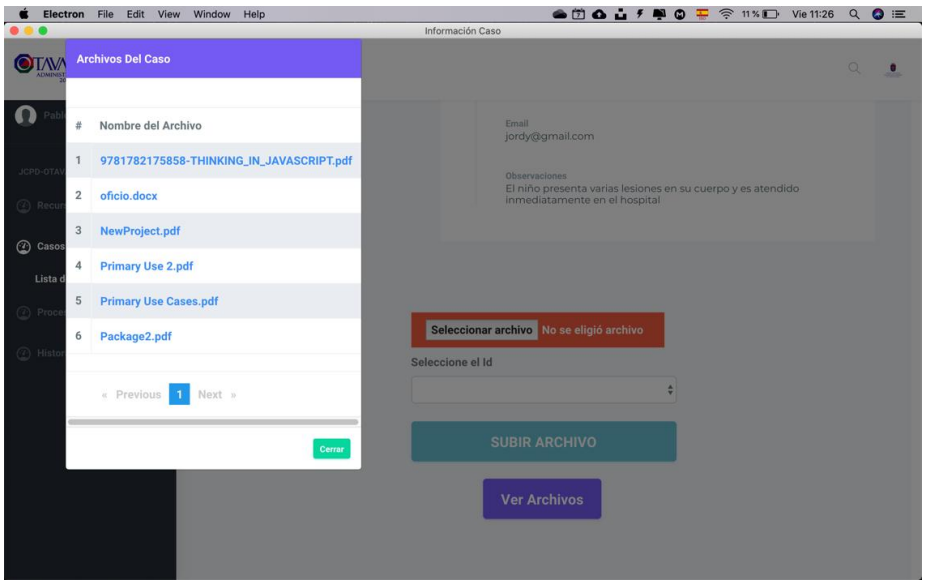


Fig. 63: Captura de prueba de gestión de archivos, módulo usuario

Fuente: Propia

3.6. Evaluar la característica de seguridad basado en la ISO/IEC 25010

Determinado el modelo de calidad de software, se propone las siguientes actividades a realizar la evaluación para validar el producto, en función del modelo de calidad.

1. Definir los análisis y definición de la ISO/IEC 25010.
2. Definir el tipo de producto software
3. Definir características, nivel de importancia y ponderación
4. Definir sub-características, nivel de importancia y ponderación
5. Definir las métricas a evaluar
6. Modelo de evaluación Característica 5 (C5P): Seguridad

3.6.1. Análisis y definición de la ISO/IEC 25010

La (ISO 25000, 2015b) define dos modelos, uno que es para la calidad de producto de software y otro que es para la calidad de uso, cada uno de estos modelos define ciertas características que al ser aplicadas en un producto de software garantizan la calidad. En el presente estudio, las características antes mencionadas se las toma como métricas para evaluar y validar cualquier aplicación, en este caso la aplicación de escritorio para el registro y seguimiento de denuncias en la Junta Cantonal de Protección de Derechos del cantón Otavalo.

3.6.2. Tipos de productos de software

El reporte de Clasificación Central de Productos (CPC), presenta un catálogo de productos relacionados al sector de software, que se muestra a continuación:

TABLA 3.76: Tipos de productos software

Productos	Tipos de productos
Página Web	Estática Animada Dinámica Portal Web Tienda Virtual o Comercio Electrónico Página Web con Gestor de Contenido Página Web 2.0
Base de Datos	Base de datos jerárquica Base de red Base de datos transaccional Base de datos relacional Base de datos multidimensional Base de datos orientado a objetos Base de datos documental Base de datos deductiva
Software de Aplicación	Software de Aplicación de productividad (editores de texto) Software de Aplicación de entretenimiento (videojuegos) Software de Aplicación de negocios (ERP) Software de Aplicación de educación (programas interactivos de aprendizaje) Software de Aplicación de tecnología (aplicaciones de control de sistemas, aplicaciones médicas, etc.)

Fuente: (INEC, 2017)

3.6.3. Nivel de Importancia

Presenta la definición del nivel de importancia que se aplicará a: características y sub-características de la aplicación de escritorio a evaluarse.

TABLA 3.77: Niveles de importancia

Nivel de Importancia	Nomenclatura	Descripción
Alta	A	El nivel de importancia de la característica y sub-característica obliga a realizar las mediciones.
Media	M	El nivel de importancia de la característica y sub-característica indica que se sujeta a criterio del evaluador
Baja	B	El nivel de importancia de la característica y sub-característica indica que no es necesaria la medición.
No Aplica	N/A	Significa que no se puede medir o aplicar.

Fuente: (Chavez, 2011)

3.6.4. Definición de característica de calidad

Las características de calidad que se presentan en la norma son aplicables para todos los sistemas de software, por lo tanto, se debe tomar en cuenta el tipo de sistema a validar, según los tipos de productos de software, así como el nivel de importancia, todo esto debido a la variación del grado de aplicabilidad e importancia de las características, sub-características y atributos (ISO 25000, 2015a).

3.6.5. Definición de sub-características de calidad

Las sub-características de calidad interna/externa y en uso más importantes para el producto software, se debe determinar de acuerdo el nivel de importancia apoyándose en la definición de características de calidad.

3.6.6. ISO/IEC 25040 – Evaluación Process (Proceso de evaluación)

Reemplaza a la ISO/IEC 14598-1:1999. Según (ISO/IEC 25040, 2019) la nueva versión define 13 procesos en cinco etapas:

1. Establecer los requisitos de la evaluación: a. Establecer el propósito de la evaluación. b. Obtener los requisitos de calidad del producto. c. Identificar las partes del producto que se deben evaluar. d. Definir el rigor de la evaluación.
2. Especificar la evaluación: a. Seleccionar los módulos de evaluación. b. Definir los criterios de decisión para las métricas. c. Definir los criterios de decisión de la evaluación.
3. Diseñar la evaluación: a. Planificar las actividades de la evaluación.
4. Ejecutar la evaluación: a. Realizar las mediciones. b. Aplicar los criterios de decisión para las métricas. c. Aplicar los criterios de decisión de la evaluación.

5. Finalizar la evaluación: a. Revisar los resultados de la evaluación. b. Crear el informe de evaluación. c. Revisar la calidad de la evaluación y obtener feedback. d. Tratar los datos de la evaluación.

5.1.1. Definición de métricas a evaluar

En la siguiente sección se va a detallar las métricas que se van a evaluar para que la aplicación de escritorio este validada, la importancia que se le dio a cada una, y el criterio personal por el cual se evaluará cada métrica.

- **GQM (Goal, Question, Metric)**

GQM (Goal, Question, Metric) es un método orientado a lograr una métrica que mida cierto objetivo de una manera determinada. El modelo de medición tiene tres niveles:

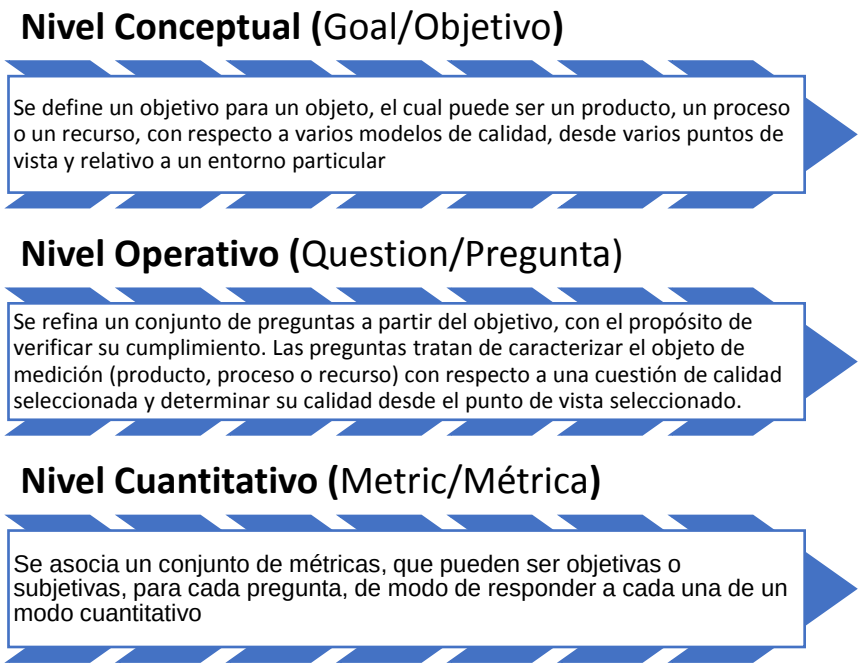


Fig. 64: GQM (Goal, Question, Metric)
Fuente: (Basili, Caldiera, & Rombach, 1994)

Un modelo GQM se desarrolla identificando un conjunto de objetivos de calidad y/o productividad, a nivel corporativo, de división o de proyecto. A partir de esos objetivos y en base a modelos del objeto de medición, se elaboran preguntas que definen esos objetivos de la manera más completa posible. El siguiente paso consiste en especificar las medidas que deben ser tomadas para responder a esas preguntas y para realizar un seguimiento de la conformidad de los productos y procesos con los objetivos. Una vez especificadas las medidas, es necesario desarrollar los mecanismos de recopilación de información, incluidos los mecanismos de validación y análisis. (Basili et al., 1994)

5.1.2. Modelo de evaluación Característica 5 (C5P): Seguridad

Nivel de importancia C5P: Alta

Criterio para evaluar C5P: Es importante que el aplicativo tenga la seguridad correcta para resguardar y administrar la información.

El modelo de desarrollo consiste en definir un conjunto de preguntas basadas en el enfoque GQM que luego, mediante conectivos lógicos, indicaran la satisfacción de los objetivos propuestos.

Como caso de estudio se tomó la característica de Seguridad que contienen las sub-características: *Confidencialidad, Integridad, No-Repudio, Responsabilidad y Autenticidad.* Podemos observar el nivel de importancia en la TABLA 3.78 correspondiente a cada una de las sub-característica y el criterio para evaluar.

TABLA 3.78: Sub características de la Característica 5 del Modelo de Calidad Externa, especificando importancia y criterio de evaluación

Sub características C5P	Nivel Importancia	Criterio para evaluar
Confidencialidad	Alta	Evalúa que los datos deben ser totalmente confidenciales, legibles solo para personas autorizadas; la información no debe ser divulgada a personas, entidades o procesos no autorizados
Integridad	Alta	Evalúa que la información no puede ser modificada por quien no este autorizado, la información debe mantenerse tal y cual fue generada con exactitud sin ser alterada por personas o procesos informáticos no permitidos.
No repudio	Alta	Evitar que la persona emisora o la persona receptora puedan negar la trasmisión de un mensaje.
Responsabilidad	Alta	Evalúa que las acciones de una entidad pueden ser rastreados de forma exclusiva.
Autenticidad	Alta	Evalúa la capacidad de demostrar la identidad de un sujeto de forma inequívoca.

Fuente: Propia

- **Cuestionario**

En función de las sub-características descriptas anteriormente, se definieron 33 preguntas para responder con verdadero/falso (V/F). Tabla 3.79

*TABLA 3.79: Cuestionario para la característica de **seguridad***

ID	PREGUNTA
P1	¿Se requiere que la contraseña posea al menos 8 caracteres?
P2	¿Se requiere que la contraseña posea letras mayúsculas y minúsculas?
P3	¿Se requiere que la contraseña posea números y letras?
P4	¿Se requiere que la contraseña posea caracteres especiales?
P5	¿El sistema utiliza conexión segura mediante HTTPS?
P6	¿La base de datos posee los datos encriptados?
P7	¿El sistema permite acceder a funcionalidades en las cuales no se tiene permiso?
P8	¿El sistema permite que cualquier persona tenga acceso a la base de datos?
P9	¿El sistema permite que cualquier persona tenga acceso al codigo del servidor de la aplicación?
P10	¿Cualquier persona tiene acceso al servidor físico?
P11	¿Cualquier persona tiene acceso al servidor remoto?
P12	¿El sistema posee redireccionamientos hacia sitios no seguros?
P13	¿El sistema solicita una confirmación de registro mediante un mail a la hora de registrarse?
P14	¿El sistema permite que cualquier persona pueda modificar la base de datos?
P15	¿El sistema permite que cualquier persona pueda modificar el codigo del servidor de la aplicación
P16	¿El sistema permite inyecciones SQL?
P17	¿El sistema posee un historial de acciones realizadas?
P18	¿El sistema posee algoritmos de cifrado de datos?
P19	¿El sistema posee un mecanismo criptográfico, como firma digital?
P20	¿El sistema solicita confirmación a la hora de realizar una acción?
P21	¿El sistema posee una protección con certificados SSL?
P22	¿El sistema da aviso cuando se accede desde una ubicación desconocida?
P23	¿El sistema informa vía mail las operaciones realizadas?
P24	¿El sistema guarda un registro de fecha y hora de ingreso al mismo?
P25	¿El sistema registra el tipo de navegador y sistema de operación utilizado para ingresar al sitio?
P26	¿El sistema registra la dirección IP desde la cual se ingresa al sitio?
P27	¿El sistema realiza una comprobación de identidad mediante un certificado digital?
P28	¿El sistema posee un sistema de verificación en dos pasos?
P29	¿Es requerida una clave de segundo nivel para el ingreso al sistema?
P30	¿El sistema realiza una comprobación de identidad mediante datos biométricos?
P31	¿El sistema realiza una comprobación de identidad mediante tarjeta de coordenadas?
P32	¿El sistema realiza una comprobación de identidad mediante credenciales?
P33	¿El sistema realiza una comprobación de identidad mediante una firma electronica?

Fuente: Propia

- **Descripción de criterios de evaluación (CE)**

Con el fin de lograr el objetivo, las respuestas de las preguntas fueron combinadas de forma lógica estableciendo un puntaje a cada uno de los CE. Tabla 3.80

TABLA 3.80: Descripción de criterios de evaluación (CE)

ID	Nombre	Descripción	Fórmula	Ptos
C-1	Conexiones seguras	Una conexión se considera segura si se utiliza HTTPS y si no se tienen redireccionamientos hacia sitios no seguros	P5 & ~P12 = V	1
C-2	Control de acceso	Se debe controlar que no se permita acceder a funcionalidades sin autorización, tampoco a la base de datos, al código de la aplicación ni a los servidores, físico ni remoto	si P7 | P8 | P9 | P10 | P11 = F	1
C-3	Encriptación de datos	Los datos de la base de datos deben estar encriptados	P6 = V	1
C-4	Contraseña de bajo nivel	La contraseña se considera de bajo nivel si posee menos de 8 caracteres, no posee letras mayúsculas y minúsculas, no posee letras y números y no posee caracteres especiales	P1 | P2 | P3 | P4 = F	0
	Contraseña de medio nivel	La contraseña se considera de medio nivel si posee al menos 8 caracteres o letras mayúsculas y minúsculas o letras y números	P1 | P2 | P3 | P4 = V	0,5
	Contraseña de alto nivel	La contraseña se considera de alto nivel si posee al menos 8 caracteres, letras mayúsculas y minúsculas, letras y números y caracteres especiales	P1 & P2 & P3 & P4 = V	1
I-5	Prevención de accesos	Se debe prevenir que no se permita acceder a funcionalidades sin autorización, tampoco a la base de datos ni al código de la aplicación, y que no se permitan inyecciones SQL	P7 | P8 | P9 | P16 =F	1
I-6	Prevención de modificaciones	Se debe prevenir que no se permita modificar datos de la base de datos ni modificar el código de la aplicación sin autorización	P14 | P15 = F	1
I-7	Confirmación de datos	Se debe realizar una confirmación de registro por mail	P13 = V	1
NR-8	Operaciones realizadas	Se debe poseer un historial de acciones realizadas o las mismas deben ser enviadas por mail	P17 | P23 = V	1
NR-9	Mecanismos de cifrado	Se debe poseer un algoritmo de cifrado de datos o un mecanismo criptográfico, como firma digital, o una protección con certificados SSL	P18 | P19 | P21 = V	1
NR-10	Confirmación de acciones	Se debe solicitar una confirmación al realizar una determinada acción	P20 = V	1
NR-11	Registro de ubicación	Se debe informar si se accedió al sistema desde una ubicación desconocida	P22 = V	1
R-12	Registro de acciones y datos	Se debe poseer un historial de acciones realizadas, o un registro de fecha y hora de ingreso al sistema o de la dirección IP desde la cual se ingresa o del tipo de navegador y sistema de operación utilizado	P17 | P24 | P25 | P26 = V	1
R-13	Control de ubicación	Se debe dar aviso cuando se accede al sistema desde una ubicación desconocida	p22 = V	1
A-14	Comprobación de identidad	El sistema debe realizar una comprobación de identidad mediante alguno de los siguientes métodos: datos biométricos, tarjeta de coordenadas, credenciales, firma electrónica o certificado digital	P27 | P30 | P31 | P32 | P33 = V	1
A-15	Comprobaciones adicionales	Se debe poseer un sistema de verificación en dos pasos, o se debe requerir una clave de segundo nivel para el ingreso al sistema o una confirmación de registro mediante un mail	P28 | P29 | P13 = V	1

Fuente: Propia

- **Métricas para cada sub-característica.**

Se combinaron los CE(Criterios de Evaluación) para definir las métricas que satisfacen los objetivos de las sub-características. Para cada una se definió un nombre, un propósito, un método de aplicación, valores de entradas y formula aplicada.(Calabrese et al., 2017) Podemos observar las sub-características con su propósito desde la TABALA 3.81 hasta la TABLA 3.85

*TABLA 3.81: Métrica de sub-característica **confidencialidad***

Confidencialidad	
Métrica:	*Confidencialidad*
Propósito:	*¿Cuán eficiente es el sistema a la hora de proteger el acceso de datos e información no autorizados, ya sea accidental o deliberadamente?*
Método de aplicación:	*Contestar las preguntas de los CE correspondientes a la subcaracterística "Confidencialidad" y calcular la puntuación obtenida, sumando los puntajes de los CE referenciados que cumplan con la meta esperada. "Puntaje total" hace referencia al máximo puntaje que se puede obtener.*
Entradas:	*A = Puntaje obtenido. B = Puntaje total.*
Fórmula:	*X = A/B*
Observaciones:	*Los CE a utilizar son: C-1, C-2, C-3 y C-4.*

Fuente: Propia

*TABLA 3.82: Métrica de sub-característica **integridad***

Integridad	
Métrica:	*Integridad*
Propósito:	*¿Cuán capaz es el sistema a la hora de prevenir accesos o modificaciones no autorizados a datos o programas de ordenador?*
Método de aplicación:	*Contestar las preguntas de los CE correspondientes a la sub- característica "Integridad" y calcular la puntuación obtenida, sumando los puntajes de los CE referenciados que cumplan con la meta esperada. "Puntaje total" hace referencia al máximo puntaje que se puede obtener.*
Entradas:	*A = Puntaje obtenido. B = Puntaje total.*
Fórmula:	*X = A/B*
Observaciones:	*Los CE a utilizar son: I-5, I-6 e I-7.*

Fuente: Propia

*TABLA 3.83: Métrica de sub-característica **No-Repudio***

No-Repudio	
Métrica:	*No-Repudio*
Propósito:	*¿Cuán capaz es el sistema de demostrar las acciones o eventos que han tenido lugar, de manera que dichas acciones o eventos no puedan ser repudiados posteriormente?*
Método de aplicación:	*Contestar las preguntas de los CE correspondientes a la sub- característica "No-Repudio" y calcular la puntuación obtenida, sumando los puntajes de los CE referenciados que cumplan con la meta esperada. "Puntaje total" hace referencia al máximo puntaje que se puede obtener.*
Entradas:	*A = Puntaje obtenido. B = Puntaje total.*
Fórmula:	*X = A/B*
Observaciones:	*Los CE a utilizar son: NR-8, NR-9, NR-10 y NR-11.*

Fuente: Propia

*TABLA 3.84: Métrica de sub-característica **responsabilidad***

Responsabilidad	
Métrica:	*Responsabilidad*
Propósito:	*¿Cuán capaz es el sistema de rastrear de forma inequívoca las acciones de una entidad?*
Método de aplicación:	*Contestar las preguntas de los CE correspondientes a la subcaracterística "Responsabilidad" y calcular la puntuación obtenida, sumando los puntajes de los CE referenciados que cumplan con la meta esperada. "Puntaje total" hace referencia al máximo puntaje que se puede obtener.*
Entradas:	*A = Puntaje obtenido. B = Puntaje total.*
Fórmula:	*X = A/B*
Observaciones:	*Los CE a utilizar son: R-12 y R-13.*

Fuente: Propia

*TABLA 3.85: Métrica de sub-característica **autenticidad***

Autenticidad	
Métrica:	*Autenticidad*
Propósito:	*¿Cuán capaz es el sistema de demostrar la identidad de un sujeto o un recurso?*
Método de aplicación:	*Contestar las preguntas de los CE correspondientes a la subcaracterística "Autenticidad" y calcular la puntuación obtenida, sumando los puntajes de los CE referenciados que cumplan con la meta esperada. "Puntaje total" hace referencia al máximo puntaje que se puede obtener.*
Entradas:	*A = Puntaje obtenido. B = Puntaje total.*
Fórmula:	*X = A/B*
Observaciones:	*Los CE a utilizar son: A14 y A15.*

Fuente: Propia

Establecemos la recopilación de las formula para cada sub-característica en la Tabla 3.86

TABLA 3.86: Formula para cada sub-característica

MÉTRICA	**FÓRMULA**
CONFIDENCIALIDAD	(C1+C2+C3+C4)/4
INTEGRIDAD	(I5+I6+I7)/3
NO REPUDIO	(NR8+NR9+NR10+NR11)/4
RESPONSABILIDAD	(R12+R13)/2
AUTENTICIDAD	(A14+A15)/2

Fuente: Propia

- **Casos de Estudio**

Se realizó el proceso de evaluación según la estructura de la ISO/IEC 25040 en la aplicación de escritorio para la JCPD del cantón Otavalo, con el propósito de evaluar la característica de ***Seguridad***.

El aplicativo evaluado se encuentra en etapa de testing, posee 6 usuarios con una frecuencia mínima por los usuarios que realizan el testing de la aplicación. En este caso tenemos preguntas técnicas para personas tester, que puedan validar la seguridad de un software, es por ello que solicitamos la intervención del club Ethical Hacking de la Universidad Técnica del Norte quienes aportaran con la validación del sistema.

- **Establecer los requisitos de la evaluación**

El propósito de la evaluación es medir, analizando diferentes aspectos, la seguridad de un sistema de escritorio multiplataforma. En base al propósito se selecciona la característica "***Seguridad***" definida en la norma ISO/IEC 25010.

El sistema se encuentra en una versión de prueba, y está bajo el uso de diferentes personas encargadas del testing.

- **Especificar la evaluación**

Según (Calabrese et al., 2017) se establece las métricas para las sub-características, estas son las definidas en el apartado de los criterios de aceptación para las sub-características:

- Inaceptable: 0 <= X < 40%
- Mínimamente aceptable: 40% <= X < 60%
- Rango objetivo: 60% <= X < 90%
- Excede los requerimientos: 90% <= X <=100%

El propósito se considerará aceptado si todas las sub-características se encuentran entre los rangos mínimamente aceptables y excede los requerimientos

➢ **Diseñar la evaluación**

Para realizar la evaluación se les solicitó a tres personas del Club Ethical Hacking de la UTN que respondieran con V/F (VERDADERO/FALSO) las diferentes preguntas planteadas en el apartado Tabla 3.79. Además, se les brindó una planilla de Excel preparada para el ingreso de las respuestas, y en la misma se obtuvieron automáticamente los valores A y B de las métricas.

➢ **Ejecutar la evaluación**

Se ejecutó la evaluación según lo planificado y se obtuvieron los siguientes resultados:

El caso tiene: Confidencialidad 62.5%, Integridad 66.66%, No-Repudio 50%, Responsabilidad 50% y Autenticidad 50%

➢ **Finalización de la evaluación**

El caso evaluado posee las sub-características *Confidencialidad* e *Integridad* en el rango aceptable, *No-Repudi*o y *Autenticidad* mínimamente aceptable y *Responsabilidad* mínimamente aceptable. Esto es correspondiente luego de haber realizado la correcta evaluación por parte de los miembros del club Ethical Hacking.

En la Fig.65 se presenta la comparación de las sub-características evaluadas del caso.

- **Análisis de la característica – *Seguridad***

El aplicativo de gestión de denuncias para la JCPD-Otavalo se considera que cumple con el propósito de evaluación debido a que todas sus sub-características se encuentran en un rango de aceptación. A pesar de que es un aplicativo que esta en versión de prueba, el rango de aceptación es mínimo, pero es valido en este caso. En un futuro se prevalecerá todos los puntos evaluados que estén en un rango que exceda los requerimientos, es decir, se desea tener como un porcentaje de mas del 90%.

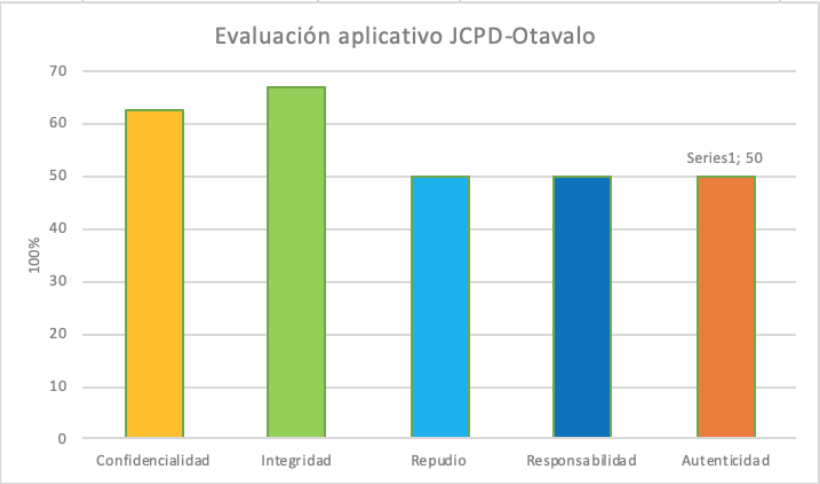


Fig. 65: Características del caso de estudio
Fuente: Propia

CAPÍTULO IV

6. Conclusiones y Recomendaciones

6.1. Conclusiones

Una vez analizado los objetivos y el alcance planteados en este proyecto de titulación se a llegado a las siguientes conclusiones:

Las nuevas tecnologías, nuevos lenguajes de programación y las tendencias de desarrollo como lo es Electron ayudan a los programadores y clientes, dueños del producto a ahorrar costos y tiempo de manera significante. En este caso generar aplicaciones de escritorio para diferentes plataformas partiendo de un solo proyecto de desarrollo que se basa en un único lenguaje de programación.

El estudio de una herramienta debe ser lo mas minucioso posible debido a que nos brinda información y ayuda aprovechar al máximo las bondades de un framework o lenguaje de programación.

La implantación de cada uno de los módulos logra optimizar los procesos que maneja la Junta Cantonal de Protección de derechos del GAD Otavalo.

La creación de un sistema multiplataforma adquiere muchos beneficios, debido a que las Juntas cantonales de protección de derechos manejan su proceso dentro de la institución no requiere que su sistema que este en la web, sino que sea un sistema normal de escritorio. Su proceso de denuncia es personal, así como el registro de casos y manejo del proceso en la JCPD, de esta manera mejora su gestión de datos debido a que sus registros pasaron de ser físicos a digitales.

Las metodologías de desarrollo de software tienen como objetivo principal agilizar el proceso de desarrollo. Permite que los desarrolladores puedan desarrollar software de calidad en un periodo de corto tiempo debido a que el cliente esta involucrado al proyecto a crearse obteniendo una retroalimentación en tiempo real mejorando la comunicación entre programador y cliente.

La norma ISO/IEC 25010 brinda un modelo de calidad para la evaluación de un conjunto de características aplicables a un producto de software, se presento un modelo de evaluación para las características y sub-características basado en el enfoque GQM. Se generaron preguntas para la sub-característica de la característica Seguridad y un conjunto de reglas de evaluación para las respuestas a las preguntas, que combinadas generaron las métricas a cada sub-característica y, en consecuencia, las métricas de la característica.

Se realizó la evaluación del sistema JCPD-Otavalo y paso la evaluación de forma positiva.

6.2. Recomendaciones

Estudiar constantemente nuevas tecnologías ayuda a un programador no quedar en el pasado y estar al día en tendencias tecnológicas a nivel del mercado actual.

Para realizar un proyecto de desarrollo se recomienda primero investigar las posibles herramientas a usar, evaluando sus características y funciones que puedan favorecer en el desarrollo.

Profundizar el aprendizaje de Electron framework para sacar mayor ventaja de todas las capacidades que este tiene para la creación de aplicaciones de escritorio multiplataforma.

Se debe seguir un proceso para poder cumplir con el objetivo del cliente, en este caso se ha automatizado el proceso existente en la JCPD, en base a ello se a logrado obtener resultados en el menor tiempo posible.

Fomentar la aplicación de metodologías de programación que permiten optimizar el costo y el tiempo en el desarrollo de una aplicación, siguiendo el ciclo de vida de software.

Se debe aplicar normas internacionales que den ciertos parámetros a seguir o cumplir con métricas para obtener un producto de calidad, que cumpla con las expectativas de los clientes.

REFERENCIAS

Ancheta, W. (15 de Julio de 2016). *Envatotuts+*. Obtenido de https://code.tutsplus.com/es/articles/an-introduction-to-nativescript--cms-26771

Pastor, J. (12 de Marzo de 2014). *Desarrollo de aplicaciones móviles (I): así está el mercado*. Obtenido de Xataca móvil: https://www.xatakamovil.com/mercado/desarrollo-de-aplicaciones-moviles-i-asi-esta-el-mercado

Pastor, J. (18 de Marzo de 2014). *Desarrollo de aplicaciones móviles (II): ¿Para qué plataformas móviles desarrollar y para cuál primero?* Obtenido de Xataka móvil: https://www.xatakamovil.com/mercado/desarrollo-de-aplicaciones-moviles-ii-para-que-plataformas-moviles-desarrollar-y-para-cual-primero

Carrasco Usano, S. (Julio de 2015). Análisis de la aplicación de la tecnología móvil en las empresas. Valencia, Valencia, España.

Wolf, G., Ruiz, E., Bergero, F., & Meza, E. (2015). *Fundamentos de sistemas operativos.* Coyoacan, México. Obtenido de https://sistop.org/pdf/sistemas_operativos.pdf

Branstein, M., & Branstein, N. (2018). *The NativeScript Book.* The Brosteins.

Domínguez Mateos, F., Paredes Velasco, M., & Santacruz Valencia, L. P. (2014). *Programación multimedia y dispositivos móviles.* Madrid: RA-MA Editorial.

Sanz, R. L., & López, R. G. (2014). *Introducción a la movilidad: 4G/LTE y el desarrollo de aplicaciones Android.* Madrid: Dextra Editorial.

Vallvé, A. M. (2017). *El mobile marketing y las apps.* Barcelona: Editorial UOC.

Gad Municipal del Canton Otavalo. (27 de 09 de 2019). Obtenido de www.otavalo.gob.ec: https://www.otavalo.gob.ec/direcciones/consejo-cantonal-de-proteccion-de-derechos.html

Arizmendi, P. (2018). *AngularJS.* Paimí Arizmendi.

Genuitec. (9 de Junio de 2017). *www.genuitec.com.* Obtenido de https://www.genuitec.com/products/angular-ide/

Vásquez, G. (28 de Junio de 2017). *Código OnClick*. Obtenido de https://codigoonclick.com/que-es-angular/

INEC. (2017). *Clasificación Central de Productos v2.* Quito: Insttituto nacionla de Estadisticas y Censos.

Chavez, M. M. (11 de marzo de 2011). *01.2. variables medicion*. Obtenido de 01.2. variables medicion: https://es.slideshare.net/SCSF2011/012-variables-medicion

Vaca Sierra, T. (2017). *Modelo de Calidad de software.* Ibarra.

buyto. (2020). Obtenido de diferencias-entre-aplicaciones-web-y-aplicaciones-desktop: buyto.es/general-diseno-web/diferencias-entre-aplicaciones-web-y-aplicaciones-desktop.html

ISO/IEC 25040. (2019). *Normas ISO 25000*. Obtenido de Normas ISO 25000: https://iso25000.com/index.php/normas-iso-25000/iso-25040

Asamblea Nacional de la Republica del Ecuador. (2008). Constitución del ecuador. *Registro Oficial*, *449*, 1–132. Retrieved from http://www.asambleanacional.gov.ec/documentos/constitucion_de_bolsillo.pdf

Basili, V. R., Caldiera, G., & Rombach, H. D. (1994). The goal question metric approach. *Encyclopedia of Software Engineering*, *2*, 528–532. https://doi.org/10.1.1.104.8626

Bethencourt Michael. (2015). NW.js y Electron.js: tecnología web en el escritorio | Tivix. Retrieved May 30, 2018, from 13 De Mayo website: https://www.tivix.com/blog/nwjs-and-electronjs-web-technology-desktop

Calabrese, J., Muñoz, R., Pasini, A., Esponda, S., Boracchia, M., & Pesado, P. (2017). Asistente para la evaluación de características de calidad de producto de software propuestas por ISO/IEC 25010 basado en métricas definidas usando el enfoque GQM. *XXIII Congreso Argentino de Ciencias de La Computación*, 660–671. Retrieved from http://hdl.handle.net/10915/63778

Chandru, T. K., Dinesh Kumar, M., Karthikeyan, S., & Saranya, K. (2019). Interactive coding platform for students. *International Journal of Recent Technology and Engineering*, *7*(4), 295–299.

CONA. (2017). Codigo de la Niñez y Adolescencia. In *Codigo de la Niñez y Adolescencia*. https://doi.org/10.1111/j.1095-8312.1989.tb01569.x

Copes, F. (2018). The Express Handbook. *Flaviocopes.Com*, 61. Retrieved from https://flaviocopes.com/express-handbook/

Corte Constitucional, E. (2020). *CONSTITUCIONAL*. (001).

D.Márquez Fernández, C.Foronda Robles, L. G. P. de A. y A. G. L. (2015). Eficacia y eficiencia de leader en Andalucia: Aproximacion a un índice-resultado en materia de turismo social. *Universidad & Empresa*, *17*(29), 131–156. Retrieved from https://dialnet.unirioja.es/servlet/articulo?codigo=5467300

De Oliveira, L. V., Egidio, M. A. R., & Oliveira, F. G. (2017). Software de suporte á decisão de gestão de custos e determinação de tarifas em perímetros de irrigação. *Irriga*, *1*(1

SpecialIssueIRRIGAandINOVAGRI), 48–56. https://doi.org/10.15809/irriga.2017v1n1p48-56

Electron.org. (2019). Arquitectura de Aplicación Electron | Electron. Retrieved October 22, 2019, from https://electronjs.org/docs/tutorial/application-architecture

Fojtik, R. (2011). Extreme programming in development of specific software. *Procedia Computer Science*, *3*, 1464–1468. https://doi.org/10.1016/j.procs.2011.01.032

Gálvez Zambrano, F. M. (2012). *Analisis, Diseño e implementación de una aplicación web que permita gestionar el cumplimiento de los requisitos y controles de una auditoria ISO 27001 basada en la norma técnica Ecuatoriana INEN-ISO/IEC27001:2011*. 198. Retrieved from http://dspace.ups.edu.ec/bitstream/123456789/3633/1/UPS-GT000348.pdf

Gheorghe, R., & Lee Hinman, M. (2013). *MEAP Edition Manning Early Access Program Elasticsearch in Action Version 3*. Retrieved from www.manning.comhttp://www.manning-sandbox.com/forum.jspa?forumID=871

GitHub vs. GitLab vs. Bitbucket: ¿Que repositorio elegir? (n.d.).

Griffith, C., & Wells, L. (2017). Electron: From Beginner to Pro. In *Electron: From Beginner to Pro*. https://doi.org/10.1007/978-1-4842-2826-5

Imgur. (2015). Imgur: la magia de internet. Retrieved April 30, 2018, from 27 De Octubre website: https://imgur.com/a/78nGo

International Organization For Standardization 25010. (2015). Software engineering - Software product Quality Requirements and Evaluation (SQuaRE) – System and software quality models. *Iso/Iec 25010:2011*, *2*(Resolution 937), 1–25. Retrieved from http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35733

ISO 25000. (2015a). Iso 25010. Retrieved January 30, 2019, from Iso 25000 website: https://iso25000.com/index.php/normas-iso-25000/iso-25010?limit=3&start=6

ISO 25000. (2015b). Iso 25010. *Iso 25000*, 3. Retrieved from https://iso25000.com/index.php/en/iso-25000-standards/iso-25010?limit=3&limitstart=0%0Ahttps://iso25000.com/index.php/en/iso-25000-standards/iso-25010?limit=3&start=3%0Ahttps://iso25000.com/index.php/en/iso-25000-standards/iso-25010?limit=3&start=6

Johnn Calvopiña. (2012). Lcdo. Johnn Calvopiña Ponce: ¿QUÉ ES UNA APLICACIÓN INFORMÁTICA?.. (material para 1ro. Bach. Aplinfor). Retrieved September 24, 2019, from http://johnnjc.blogspot.com/2012/04/que-es-una-aplicacion-informatica.html

Joskowicz, J. (2008). Reglas y prácticas en eXtreme Programming. *Universidad de Vigo.*

España, 1–22. Retrieved from http://iie.fing.edu.uy/~josej/docs/XP - Jose Joskowicz.pdf

jwt.io. (2019). Introducción al token web de JSON. Retrieved October 18, 2019, from Jwt.Io website: https://jwt.io/introduction/

Letelier, P., & Penadés, C. (2017). Masyxp.Pdf. *Métodologías Ágiles Para El Desarrollo de Software: EXtreme Programming (XP)*.

Manual de Git. (n.d.).

MariaDB. (2019). Sobre MariaDB - MariaDB.org. Retrieved October 16, 2019, from https://mariadb.org/about/

Mateu, C. (2004). *Desarrollo de aplicaciones web. http://daw-fiec.pbworks.com/w/page/16963465/Arquitectura%20aplicaci%C3%B3n%20Web*. Retrieved from http://daw-fiec.pbworks.com/w/page/16963465/Arquitectura aplicación Web

Mohagheghi, P., & Aparicio, M. E. (2017). An industry experience report on managing product quality requirements in a large organization. *Information and Software Technology*, *88*, 96–109. https://doi.org/10.1016/j.infsof.2017.04.002

Morante Luis. (2017a). Desarrollo de aplicaciones de escritorio con Electron JS. Retrieved May 10, 2018, from 8 Agosto website: https://www.tecnowired.com/programacion/desarrollo-de-aplicaciones-de-escritorio-con-electron-js/

Morante Luis. (2017b). Desarrollo de aplicaciones de escritorio con Electron JS. Retrieved January 30, 2019, from 8 Agosto website: https://www.tecnowired.com/programacion/desarrollo-de-aplicaciones-de-escritorio-con-electron-js/

OTAVALO, G. M. DE. (2014). Ordenanza-CCNyA. *30 de Agosto*, 3.

OTAVALO, G. M. (2019). Consejo Cantonal de Protección de Derechos. Retrieved September 23, 2019, from Consejo Cantonal de Protección de Derechos website: https://www.otavalo.gob.ec/direcciones/consejo-cantonal-de-proteccion-de-derechos.html#presentacion

Patel Sandeep Kumar. (2015). *Quick Desktop Application Development Using Electron*.

Penadés, M., & Letelier Torres, P. (2006). Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). *Técnica Administrativa*, *5*(26), 1.

Sistemas, E. D. E., & Software, Y. C. D. E. (2015). *Ecuatoriana Nte Inen-Iso / Iec 25010*.

Vamshi Krishna A, K. G. (2019). Mobile Application for Uploading Marks and Student Attendance Management System. *International Journal of Engineering and Advanced*

Technology, *8*(6S), 379–383. https://doi.org/10.35940/ijeat.F1080.0886S19

Vicente, R. A., & Gustavo, S. (2016). Sistema de Venta Alimenticia Personalizada.

GLOSARIO

[i] **Framework:** se podría traducir aproximadamente como marco de trabajo, es el esquema o estructura que se establece y que se aprovecha para desarrollar y organizar un software determinado. Fuente: https://neoattack.com/neowiki/framework/.

[ii] **atom-shell:** La cáscara atómica puede referirse a lo que se llama apropiadamente una electron shell o un atomic orbital que constituye una subcapa del electron. Fuente: https://en.wikipedia.org/wiki/Atom_shell

[iii] **Transversalización:** es un concepto de política pública que consiste en evaluar las diferentes implicaciones de cualquier acción política sobre las mujeres y hombres, lo que incluye la legislación y programas de cualquier área o nivel. Fuente: https://es.wikipedia.org/wiki/Transversalizaci%C3%B3n_de_g%C3%A9nero

[iv] **Concepción:** Acto de concebir o quedar fecundada una mujer. Fuente: https://www.wordreference.com/definicion/concepci%C3%B3n

[v] **Express** es un framework web transigente, escrito en JavaScript y alojado dentro del entorno de ejecución NodeJS. El modulo explica algunos de los beneficios clave de este framework, como configurar tu entorno de desarrollo y como realizar tareas comunes en desarrollo y publicación web. Fuente: https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs

[vi] **Template Engine** son los motores de plantillas toman cadenas con tokens y producen cadenas renderizadas con valores en lugar de los tokens como salida. Las plantillas se usan típicamente como un formato intermedio escrito por desarrolladores para producir mediante programación uno o más formatos de salida deseados, comúnmente HTML, XML o PDF. Fuente: https://www.fullstackpython.com/template-engines.html

[vii] **Sun Microsystems** es una empresa informática adquirida por Oracle Corporation, anteriormente parte de Silicon Valley, fabricante de semiconductores y software. Sun Microsystems es la creadora de la popular suite ofimática OpenOffice.org y el lenguaje de programación Java. Fuente: https://www.ecured.cu/Sun_Microsystems

[viii] **RDBMS** Un sistema de gestión de bases de datos relacionales (RDBMS, por sus siglas en inglés) es aquel que sigue el modelo relacional. Fuente: https://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos_relacionales

[ix] **JavaScript** es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Fuente: https://es.wikipedia.org/wiki/JavaScript

[x] **API** (Application Programming Interface) es un conjunto de funciones y procedimientos que cumplen una o muchas funciones con el fin de ser utilizadas por otro software. Fuente: https://hipertextual.com/archivo/2014/05/que-es-api/

[xi] **Package.json** este archivo es un elemento clave en muchas bases de código de aplicaciones basadas en el ecosistema Node.js. Fuente: https://flaviocopes.com/package-json/

[xii] **ORM (Object Relational Mapping:** es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y la utilización de una base de datos relacional como motor de persistencia. Fuente: https://es.wikipedia.org/wiki/Mapeo_objeto-relacional

[xiii] **(JavaScript Object Notation)** es un formato ligero de intercambio de datos. Es fácil para los humanos leer y escribir. Es fácil para las máquinas analizar y generar. Se basa en un subconjunto del Estándar de lenguaje de programación JavaScript ECMA-262, 3.a edición, diciembre de 1999. Fuente: https://www.json.org/

[xiv] **HMAC**: En la criptografía, un código de autentificación de mensajes en clave-hash (HMAC) es una construcción específica para calcular un código de autentificación de mensaje (MAC) que implica una función hash criptográfica en combinación con una llave criptográfica secreta. Fuente: https://es.wikipedia.org/wiki/HMAC

[xv] **RSA** (Rivest, Shamir y Adleman): En criptografía es un sistema criptográfico de clave pública desarrollado en 1979. Es el primer y más utilizado algoritmo de este tipo y es válido tanto para cifrar como para firmar digitalmente. Fuente: https://es.wikipedia.org/wiki/RSA

[xvi] **ECDSA** (Elliptic Curve Digital Secure Algorithm) La criptografía de curva elíptica es ya una de las más populares, es extremadamente segura desde el punto de vista matemático y comparado con RSA, ECDSA utiliza claves más pequeñas y en general es una implementación más eficiente. ECDSA es por el momento el esquema utilizado por Bitcoin, Ethereum y la mayor parte de los protocolos blockchain. Fuente: https://libroblockchain.com/ecdsa/

[xvii] **SWT:** El nombre lo dice todo. Es un token, es para web (lea HTTP) y es simple.

[xviii] **SAML:** Es el enlace entre la autenticación de la identidad de un usuario y la autorización para usar un servicio.