

UNIVERSIDAD TÉCNICA DEL NORTE



Facultad de Ingeniería en Ciencias Aplicadas

Carrera de Ingeniería en Sistemas Computacionales

**“ESTUDIO DE LA INTEGRACIÓN DEL LENGUAJE DE PROGRAMACIÓN PHP
CON LA BASE DE DATOS NOSQL APACHE COUCHDB.**

**SISTEMA DE GESTIÓN DE EXPEDIENTES DE LOS ESTUDIANTES DE LA
CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES”**

Trabajo de Grado Previo a la Obtención del Título de Ingeniera en Sistemas
Computacionales.

Autora:

Nataly Estefanía Zuleta Sevilla

Director:

Ing. Xavier Mauricio Rea Peñafiel, MSC.

Ibarra – Ecuador

2018



UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN

A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1.- IDENTIFICACIÓN DE LA OBRA:

La UNIVERSIDAD TÉCNICA DEL NORTE dentro del proyecto Repositorio Digital Institucional determina de disponer textos completos en formato digital con la finalidad de apoyar los procesos de investigación, docencia y extensión de la Universidad. Por medio del presente documento dejo sentada mi voluntad de participar en este proyecto, para lo cual pongo a disposición la siguiente información:

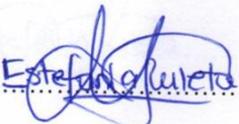
DATOS DEL AUTOR	
CÉDULA DE IDENTIDAD	1003554795
APELLIDOS Y NOMBRES	ZULETA SEVILLA NATALY ESTEFANÍA
DIRECCIÓN	PIMAMPIRO, BOLÍVAR Y GARCÍA MORENO
E-MAIL	nezuletas@utn.edu.ec
TELÉFONO MÓVIL	0988803995
DATOS DE LA OBRA	
TÍTULO	“ESTUDIO DE LA INTEGRACIÓN DEL LENGUAJE DE PROGRAMACIÓN PHP CON LA BASE DE DATOS NOSQL APACHE COUCHDB. SISTEMA DE GESTIÓN DE EXPEDIENTES DE LOS ESTUDIANTES DE LA CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES”
AUTORA	ZULETA SEVILLA NATALY ESTEFANÍA
FECHA	ABRIL DEL 2018
PROGRAMA	PREGRADO
TÍTULO POR EL QUE OPTA	INGENIERÍA EN SISTEMAS COMPUTACIONALES
DIRECTOR	ING. XAVIER MAURICIO REA PEÑAFIEL, MSC.

2.- AUTORIZACIÓN DE USO A FAVOR DE LA UNIVERSIDAD

Yo, NATALY ESTEFANÍA ZULETA SEVILLA, con cédula de identidad Nro. 100355479-5, en calidad de autora y titular de los derechos patrimoniales de la obra o trabajo de grado descrito anteriormente, hago entrega del ejemplar respectivo en forma digital y autorizo a la Universidad Técnica del Norte, la publicación de la obra en el Repositorio Digital Institucional y uso del archivo digital en la Biblioteca de la Universidad con fines académicos, para ampliar la disponibilidad del material y como apoyo a la educación, investigación y extensión; en concordancia con la Ley de Educación Superior Artículo 144.

3.- CONSTANCIAS

La autora manifiesta que la obra de la presente autorización es original y se desarrolló, sin violar derechos 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.



Firma

Nombre: Nataly Estefanía Zuleta Sevilla

Cédula: 100355479-5

Ibarra, Abril del 2018



UNIVERSIDAD TÉCNICA DEL NORTE
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

**CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE GRADO A FAVOR DE
LA UNIVERSIDAD TÉCNICA DEL NORTE**

Yo, Nataly Estefanía Zuleta Sevilla, con cédula de identificación Nro. 100355479-5, manifiesto mi voluntad de ceder a la Universidad Técnica del Norte los derechos patrimoniales consagrados en la ley de propiedad intelectual del Ecuador, artículo 4, 5 y 6, en calidad de autor del trabajo denominado: **“ESTUDIO DE LA INTEGRACIÓN DEL LENGUAJE DE PROGRAMACIÓN PHP CON LA BASE DE DATOS NOSQL APACHE COUCHDB, SISTEMA DE GESTIÓN DE EXPEDIENTES DE LOS ESTUDIANTES DE LA CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES.”**, que ha sido desarrollado para obtener el título de Ingeniera en Sistemas Computacionales, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En mi condición de autora me reservo los derechos morales de la obra antes citada. En concordancia suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital a la biblioteca de la Universidad Técnica del Norte.



Firma
Nombre: Nataly Estefanía Zuleta Sevilla
Cédula: 100355479-5
Ibarra, Abril del 2018



UNIVERSIDAD TÉCNICA DEL NORTE
FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICACIÓN DIRECTOR DE TESIS

Por medio del presente yo Msc. Xavier Rea, certifico que la Srta. Nataly Estefanía Zuleta Sevilla, portadora de la cédula de identidad Nro. 100355479-5. Ha trabajado en el desarrollo del proyecto de tesis **“ESTUDIO DE LA INTEGRACIÓN DEL LENGUAJE DE PROGRAMACIÓN PHP CON LA BASE DE DATOS NOSQL APACHE COUCHDB, SISTEMA DE GESTIÓN DE EXPEDIENTES DE LOS ESTUDIANTES DE LA CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES”**, previo a la obtención del título de ingeniería en sistemas computacionales, lo cual ha realizado en su totalidad con responsabilidad.

Es todo cuanto puedo certificar en honor a la verdad.

Ing. Xavier Mauricio Rea Peñafiel, Msc

DIRECTOR DE TRABAJO DE GRADO

DEDICATORIA

A DIOS

Por haberme permitido llegar hasta este punto dándome fortaleza, salud para continuar cuando a punto de caer he estado y lograr así con mis objetivos, con toda la humildad que mi corazón puede emanar, dedico primeramente a él mi trabajo.

A la Institución

A la Universidad Técnica del Norte por haberme permitido crecer día a día formándome como persona y profesional para poder servir a la sociedad.

A mi madre Wilma Sevilla

Por su ejemplo diario de lucha, sus consejos, sus valores, por la motivación constante, y el haberme guiado siempre por el camino correcto y finalmente por el haberme permitido estudiar y poder ser una profesional.

A mi amor

Marcelo Mejía, siendo mi mayor motivación en mi vida encaminada al éxito, fue el ingrediente perfecto para lograr alcanzar esta dicha y muy merecida victoria en la vida, el haber culminado esta tesis con éxito, y el disfrutar del privilegio de ser agradecido, ser grato con esa persona que se preocupó en cada momento y siempre quiso lo mejor para mi porvenir.

A mis abuelitos y familia

A mis abuelitos por el apoyo moral, económico y sentimental que me han brindado en el desarrollo de este proyecto. Y mi familia por estar siempre a mi lado.

A mis docentes

A todos mis Docentes por haberme guiado en el desarrollo de este proyecto, especialmente al Ing. Mauricio Rea por su tiempo compartido y por impulsar el desarrollo de mi formación profesional, además por haberme compartido sus conocimientos en las aulas a lo largo de mi vida estudiantil.

Nataly Estefanía Zuleta Sevilla

AGRADECIMIENTO

Agradezco a Dios por haberme acompañado y guiado a lo largo de mi carrera, por ser mi fortaleza en los momentos de debilidad y por brindarme una vida llena de aprendizaje, experiencias y sobre todo felicidad.

Le doy gracias a mi madre Wilma por apoyarme en todo momento, inculcarme valores, y haberme dado la oportunidad de tener una excelente educación en el transcurso de mi vida. Gracias mamá, por ser un excelente ejemplo a seguir.

Le agradezco la confianza, apoyo y dedicación de tiempo al Ing. Mauricio Rea. Por haber compartido conmigo sus conocimientos y sobre todo su amistad.

Gracias Ingeniero Mauricio Rea por creer en mí, y haberme brindado el apoyo y la oportunidad de crecer profesionalmente y aprender cosas nuevas.

A mis amigos especialmente a Lorena, por brindarme su amistad y confiar en mí, y a los demás por haber hecho de mi etapa universitaria un trayecto de vivencias que nunca olvidaré.

A mi abuelito Antonio que, aunque ya no se encuentre con nosotros físicamente, siempre estará presente en mi corazón, por el apoyo y cariño brindado hasta el último momento, gracias papá. ¡Ya soy Ingeniera!

Al resto de mi familia, que estuvieron apoyándome y aportaron de una u otra manera a culminar con éxito esta etapa de mi vida.

Son muchas las personas que han formado parte de mi vida profesional a las que me encantaría agradecerles su amistad, consejos, ánimo y compañía en los momentos más difíciles de mi vida. Algunas están aquí conmigo y otras en mis recuerdos y en mi corazón, sin importar en donde estén quiero darles las gracias por formar parte de mí, por todo lo que me han brindado y sus bendiciones.

ÍNDICE DE CONTENIDO

AUTORIZACIÓN DE USO Y PUBLICACIÓN	II
CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE GRADO A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE	IV
CERTIFICACIÓN DIRECTOR DE TESIS	V
DEDICATORIA	VI
AGRADECIMIENTO	VII
ÍNDICE DE TABLAS	XI
ÍNDICE DE FIGURAS	XII
RESUMEN	XIV
ABSTRACT	XV
GLOSARIO	XVI
INTRODUCCIÓN	1
Antecedentes.....	1
Situación actual de la Coordinación de la Carrera CISIC.....	1
Prospectiva.....	1
Planteamiento del problema	2
Objetivos	2
Objetivo general.....	2
Objetivos específicos	2
Alcance.....	3
Metodología de Desarrollo	4
Justificación.....	4
CAPÍTULO 1	5
Marco Teórico.....	5
1.1 Introducción.....	5
1.1.1 Historia de la carrera.....	5
1.2 Situación actual de la carrera.....	7
1.3 Proyección.....	7
1.3.1 Sistema de gestión de expedientes	7
1.3.2 Beneficios	7
1.4 Estudio de la integración del lenguaje PHP.	7
1.4.1 Programación PHP	7
1.5 Herramientas de programación PHP	8
1.5.1 Laravel PHP.	8

1.5.2 Arquitectura.....	8
1.5.3 Capas MVC en Laravel.....	8
1.5.4 Arquitectura MVC Laravel.....	9
1.5.5 Comparativa de Laravel con otros Framework de PHP	11
1.5.6 Análisis de la comparativa de las principales características de los Framework PHP.	15
1.5.7 Análisis	16
1.6 Base de datos NoSQL Apache CouchDB.....	16
1.6.1 Base de datos NoSQL	16
1.7 Base de datos Documental	18
1.8 CouchDB.....	19
1.8.1 Historia de CouchDB	19
1.8.2 Definición	19
1.8.3 Arquitectura:.....	19
1.8.4 Características CouchDB	21
1.8.5 Documentos CouchDB	21
1.8.6 Ventajas	21
1.8.7 Desventajas.....	22
1.9 JSON	22
1.10 APACHE	22
1.11 Servicios REST.....	23
1.12 HTML5	25
1.13 CSS3.....	26
1.14 Estudio de la Integración del lenguaje PHP y CouchDB	26
1.15 Metodología Scrum.....	46
1.15.1 Beneficios.....	47
1.15.2 Roles y responsabilidades	48
1.15.3 Elementos	49
1.15.4 Reuniones	50
1.15.5 Características.....	52
1.15.6 Ventajas	52
1.15.7 Desventajas.....	52
CAPÍTULO 2	53
2 Desarrollo del Sistema Gestión de Expedientes.	53
2.1 Desarrollo.....	53
2.2 ROLES DEL SISTEMA	53
2.3 HISTORIAS DE USUARIOS Y CRITERIOS DE ACEPTACIÓN	54

2.4 PILA DE PRODUCTOS.....	57
2.5 PILA DE TAREAS.....	58
2.6 PLANIFICACIÓN DEL PROYECTO	60
2.7 ITERACIONES	61
2.7.1 ITERACIÓN 1 ANÁLISIS Y ESTRUCTURACIÓN DEL PROYECTO	61
2.7.2 ITERACIÓN 2 REGISTRO Y CREACIÓN DEL ADMINISTRADOR.	69
2.7.3 ITERACIÓN 3 SEGURIDAD Y CONTROL.	73
2.7.4 ITERACIÓN 4 CREACIÓN DE USUARIOS.....	76
2.7.5 ITERACIÓN 5 CREACIÓN DE ALUMNOS.....	79
2.7.6 ITERACIÓN 6 IMPLEMENTACIÓN DEL SISTEMA.	83
CAPÍTULO 3	89
CONCLUSIONES Y RECOMENDACIONES.....	89
Referencias	91

ÍNDICE DE TABLAS

Tabla 1: Comparativa de Laravel con otros Framework de PHP.....	14
Tabla 2: Comparación frameworks PHP.	15
Tabla 3: Cuadro comparativo de las bases de datos NoSQL y SQL.	18
Tabla 4: Características de CouchDB.....	21
Tabla 5: Documentos CouchDB.....	21
Tabla 6: Arquitectura del Sistema.....	27
Tabla 8: Beneficios de Scrum.....	47
Tabla 9: Equipo de trabajo para el desarrollo del sistema.....	53
Tabla 10: Historias de usuario y criterios de aceptación.....	56
Tabla 11: Pila de productos.....	57
Tabla 12: Análisis y Estructuración del proyecto.....	58
Tabla 13: Registro y creación del administrador.....	58
Tabla 14: Seguridad y control.....	59
Tabla 15: Creación de Usuarios.....	59
Tabla 16: Creación de alumnos.....	59
Tabla 17: Implementación del sistema.....	60
Tabla 18: Planificación del Proyecto.....	60
Tabla 19: Sprint 1 – Hoja Electrónica.....	68
Tabla 20: Sprint 1 – Pizarrón.....	69
Tabla 21: Sprint 2 – Hoja electrónica.....	73
Tabla 22: Sprint 2 – Pizarrón.....	73
Tabla 23: Sprint 3 – Hoja electrónica.....	75
Tabla 24: Sprint 3 – Pizarrón.....	76
Tabla 25: Sprint 4 – Hoja electrónica.....	78
Tabla 26: Sprint 4 – Pizarrón.....	79
Tabla 27: Sprint 5 – Hoja electrónica.....	82
Tabla 28: Sprint 5 – Pizarrón.....	82
Tabla 29: Sprint 6 – Hoja electrónica.....	86
Tabla 30: Sprint 6 – Pizarrón.....	87

ÍNDICE DE FIGURAS

Fig. 1: Árbol de problemas	2
Fig. 2: Arquitectura del Sistema	3
Fig. 3: Diagrama de despliegue	4
Fig. 4: Historia Carrera de Ingeniería en Sistemas Computacionales	6
Fig. 5: Digitalización de documentos.....	7
Fig. 6: Logo framework Laravel.....	8
Fig. 7: Petición típica MVC en Laravel	10
Fig. 8: Estadística de comparación de Framework PHP.....	16
Fig. 9: Logo NoSQL.....	17
Fig. 10: Diferencia del almacenamiento de la base de datos NoSQL con la base de datos SQL	17
Fig. 11: Base de datos documental	19
Fig. 12: Arquitectura CouchDB.....	20
Fig. 13: Logo de Apache	23
Fig. 14: Servicios Rest – Cliente/Servidor	23
Fig. 15: Servicios Rest - Sin Estado	24
Fig. 16: Servicios Rest – Cache.....	24
Fig. 17: Servicios Rest – Servicios uniformes.....	24
Fig. 18: Servicios Rest – Arquitectura en capas.....	25
Fig. 19: Diagrama de integración de la base CouchDB con Laravel PHP.	28
Fig. 20: Clases PHP, del ejemplo Conexión	29
Fig. 21: Proyecto ejemplo, conexión	30
Fig. 22: Ingreso al proyecto http://localhost/conexion/index.php	40
Fig. 23: Ingreso a la base de datos NoSQL CouchDB	41
Fig. 24: Ingreso nuevo alumno.....	41
Fig. 25: Creación de la base de datos cont1 en CouchDB	42
Fig. 26: Editar Datos Alumno.....	42
Fig. 27: Edición del nombre	43
Fig. 28: Ingreso de un tercer alumno para su eliminación	43
Fig. 29: Eliminación del alumno ingresado.....	44
Fig. 30: Eliminación	44
Fig. 31: Alumno eliminado.....	44
Fig. 32: Base de datos cont1 mediante tabla.....	45
Fig. 33: Base de datos cont1 mediante Metadata.....	45
Fig. 34: Base de datos cont1 mediante JSON.....	45
Fig. 35: Opción Buscar	46
Fig. 36: Componentes Scrum.....	47
Fig. 37: Roles Scrum.....	49
Fig. 38: Elementos principales del Scrum.....	50
Fig. 39: Sprint.....	51
Fig. 40: Instalación de CouchDB.....	61
Fig. 41: Instalación de Apache	62
Fig. 42: Instalación de Composer	62
Fig. 43: Instalación de Laravel.....	63
Fig. 44: Diagrama de Clases, Controller.....	64
Fig. 45: Diagrama de clases, Model	65
Fig. 46: Comando ingreso al directorio XAMPP	66

Fig. 47: Comando para crear proyecto en Laravel.....	66
Fig. 48: Creación del proyecto.	66
Fig. 49: Instalación de librerías Laravel.	67
Fig. 50: Finalización de instalación librerías de Laravel.....	67
Fig. 51: Vista de inicio del proyecto en Laravel	68
Fig. 52: Interfaz de Registro para el ingreso al sistema	70
Fig. 53: Ingreso de datos personales.....	71
Fig. 54: Ingreso al sistema como administrador.....	71
Fig. 55: Perfil del administrador.....	72
Fig. 56: Ingreso al perfil del administrador.	72
Fig. 57: Login del sistema.	74
Fig. 58: Seguridad del sistema.....	74
Fig. 59: Creación de usuario.....	75
Fig. 60: Creación de usuario.....	76
Fig. 61: Ingreso al perfil de usuario creado para ser editado.	77
Fig. 62: Usuarios.....	78
Fig. 63: Creación de alumno.....	79
Fig. 64: Creación de módulos con sus diferentes ítems	80
Fig. 65: Ítems del módulo Documentos Matricula Primer Semestre.	80
Fig. 66: Creación de estadísticas – documentos que se suben una sola vez.	81
Fig. 67: Creación de estadísticas – documentos que se suben cada período.	81
Fig. 68: Botón para crear nuevo alumno.	83
Fig. 69: Creación de alumno.....	83
Fig. 70: Módulos e ítems para subir documentación.	84
Fig. 71: Estadísticas – documentos que se suben una sola vez.	85
Fig. 72: Estadísticas – documentos que se suben cada periodo.....	85

RESUMEN

El presente Trabajo de Titulación se basa en realizar un estudio utilizando la integración de una base de datos NoSQL Apache CouchDB con el lenguaje de programación PHP, con la finalidad de aplicarla en el diseño del sistema para la gestión de expedientes de los estudiantes de la carrera de Ingeniería en Sistemas Computacionales. La presente propuesta de tesis tiene como objetivo el estudio de la integración de estas herramientas para demostrar mediante el sistema de gestión de expedientes de los estudiantes de la carrera CISIC y al mismo tiempo incorporando este sistema como forma de ayuda a la coordinación de la carrera. Para cumplir con el estudio de estas herramientas está estructurado de la siguiente manera:

Introducción

En esta parte se define varios conceptos como los antecedentes, situación actual, prospectiva, planteamiento del problema, se detalla el objetivo general y objetivos específicos en las cuales se explica el estudio de las herramientas, de igual manera el alcance y la justificación que fueron aprobados para este proyecto.

Capítulo I

En este capítulo se detalla el marco teórico como conceptos básicos acerca de las herramientas a utilizar, se define la Metodología que se utilizará para el desarrollo del sistema en este caso se trabajará con la metodología Scrum en la cual se explicará definición, características, fases de desarrollo y a su vez implementando al sistema esta metodología.

Además contiene el estudio de la integración de la base de datos NoSQL Apache CouchDB con el lenguaje de programación PHP, mediante un ejemplo.

Capítulo II

En este capítulo se explica el desarrollo del sistema, el proceso de diseño los requisitos funcionales y una introducción de las herramientas que se utilizara para el desarrollo del sistema, diagramas de casos de uso, construcción, pruebas e implementación.

Capítulo III

Finalmente se añade conclusiones, recomendaciones y referencias bibliográficas.

ABSTRACT

The present work of Qualification is based on a study using the integration of a NoSQL database Apache CouchDB with PHP programming language, with the purpose of it is to apply to the design of the system for the management of records of students of the career of Computer Systems Engineering. The present thesis proposal has as its objective the study of the integration of these tools to demonstrate through the records management system of the CISIC students and at the same time incorporating this system as a way to assist in the coordination of the race. To comply with the study of these tools is structured in the following way:

Introduction

In this part defines a number of concepts such as the background, current situation, forward-looking approach to the problem, details the overall goal and specific objectives in which explains the study of the tools, in the same way the scope and rationale that were approved for this project.

Chapter I

In this chapter details the theoretical framework and basic concepts about the tools to use, defines the methodology that will be used for the development of the system in this case is to work with the Methodology Scrum in which will be explained definition, features, stages of development and at the same time implementing the system this methodology.

It also contains the study of the integration of the database NoSQL Apache CouchDB with PHP programming language, by using an example.

Chapter II

This chapter explains the development of the system, the process of designing the functional requirements and an introduction of the tools that will be used for the development of the system, use case diagrams, construction, testing, and implementation.

Chapter III

Finally is added conclusions, recommendations and references.

GLOSARIO

SGBD: Sistema de Gestión de Base de datos es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información de una base de datos, además proporciona herramientas para añadir, borrar, modificar y analizar los datos.

XML: (Extensible Markup Language) es un lenguaje de etiquetas, es decir, cada paquete de información está delimitado por dos etiquetas, que a diferencia del HTML separa el contenido de la presentación, y se está convirtiendo en un estándar de amplio uso para intercambio de datos.

Apache License: es una licencia de software libre permisiva creada por la Apache Software Foundation (ASF), requiere la conservación del aviso de derecho de autor y el descargo de responsabilidad.

CouchBase: originalmente conocido como Membase, es un proyecto de código abierto, un paquete de software distribuido (del inglés arquitecturas shared-nothing) multi-modelo orientado a base de datos documental y que está optimizado para aplicaciones interactivas.

HTTP: (Hypertext Transfer Protocol), protocolo de transferencia de hipertextos, que se utiliza en algunas direcciones de internet.

Erlang: es un lenguaje de programación concurrente esto quiere decir que es la simultaneidad en la ejecución de múltiples tareas interactivas y un sistema de ejecución que incluye una máquina virtual (BEAM) y bibliotecas (OTP).

Restful: transferencia de estado representacional es un estilo de arquitectura software para sistemas hipermedias distribuidos.

MapReduce: es un framework que proporciona un sistema de procesamiento de datos paralelo y distribuido. Su nombre se debe a las funciones principales que son Map y Reduce, las cuales explicaremos a continuación. MapReduce está pensado para la solución práctica de algunos problemas que pueden ser paralelizados, pero se ha de tener en cuenta que no todos los problemas pueden resolverse eficientemente con MapReduce.

API: Es un conjunto de funciones y procedimientos que cumplen una o muchas funciones con el fin de ser utilizadas por otro software. Las siglas API vienen del inglés Application Programming Interface. En español sería Interfaz de Programación de Aplicaciones. (andrearrrs, 2014)

ROI: (Return On Investment) es el valor económico generado como resultado de la realización de diferentes actividades de marketing. Con este dato, podemos medir el rendimiento que hemos obtenido en una inversión. (@AVILLAREJO)

Eloquent: el ORM Eloquent incluido con Laravel proporciona una simple y elegante implementación para trabajar con tu base de datos. (Drauta, 2016)

JSON: es un formato de texto para la serialización de datos estructurados. (Sanchez, 2014)

Scrum: es un proceso de gestión y control que reduce la complejidad en el desarrollo de productos para satisfacer las necesidades de los clientes. (Francia, 2017)

Open Source: llamado también “código abierto”, es un término que se utiliza para denominar a cierto tipo de software que le permite al usuario, utilizar el código fuente del programa para estudiarlo, modificarlo y realizar mejoras en el mismo. (fácil, 2015)

JavaScript: es una especie de lenguaje de programación ligera, interpretado por la mayoría de los navegadores y que les proporciona a las páginas web, efectos y funciones complementarias a las consideradas como estándar HTML. Este tipo de lenguaje de programación, con frecuencia son empleados en los sitios web, para realizar acciones en el lado del cliente, estando centrado en el código fuente de la página web. (J, 2017)

Web Dinámica: se conoce como web dinámica aquella, cuyo contenido se genera a partir de lo que un usuario introduce en un web o formulario. El contenido de la página no está incluido en un archivo HTML como en el caso de las páginas web estáticas. (Mera, 2013)

INTRODUCCIÓN

Antecedentes

CouchDB es una base de datos Open Source orientada a documentos, accesible mediante una API RESTFUL que hace uso extensivo de JavaScript Object Notation (JSON). Su misión es ser muy escalable, con alta disponibilidad y robustez, incluso cuando se ejecuta en hardware convencional. (Artaza, 2012). Hypertext Pre-Processor(PHP) es el heredero de un producto anterior, llamado PHP/FI.

Fue creado por Rasmus Lerdorf en 1995, inicialmente como un simple conjunto de scripts de Perl para controlar los accesos su trabajo online. Rasmus fue escribiendo una implementación C mucho mayor, que era capaz de comunicarse con la base de datos, y permitía a los usuarios desarrollar sencillas aplicaciones Web dinámica.

En PHP se puede tener distintos productos que utilizan esta tecnología, las más conocidas son de tipo documental y de tipo clave-valor. Bases de datos documentales tenemos por ejemplo CouchDB y MongoDB, en cuanto a bases de datos de tipo clave-valor quizás la más conocida sea Cassandra, creada por y para Facebook. Todos estos ejemplos son de código abierto.

Los sistemas documentales son sistemas informáticos creados para almacenar, administrar y controlar el flujo de documentos dentro de una organización. Se trata de una forma de organizar los documentos e imágenes digitales en una localización centralizada a la que los empleados puedan acceder de forma fácil y sencilla.

Beneficios de los sistemas documentales, reducción de costes gracias a información estructurada, mejor porcentaje en documentos y aumento de la velocidad de creación de contenidos. (tic.Portal, 2016)

Situación actual de la Coordinación de la Carrera CISIC

Actualmente en la Coordinación de la Carrera de Ingeniería en Sistemas Computacionales no existe un registro digital de los expedientes, lo que crea una demora en la verificación de documentos de forma física en los archivos de la carrera, existiendo inconformidad de los estudiantes en la obtención rápida y oportuna de la información.

Prospectiva

Con este sistema se logrará dar un soporte informático a través de un medio lógico que sea administrable, debido a que los medios físicos se pueden dañar, corroer, quemar, alterar, causando la pérdida parcial o total de la información, además con estas herramientas a utilizar

se realizará el sistema con lo que se quiere facilitar la gestión del personal administrativo de la Carrera de Ingenierías en Sistemas Computacionales, ofreciendo mejor opción para ahorrar espacio, tiempo, dinero y aportar a la conservación del medio ambiente.

Planteamiento del problema

Escasa información en lo referente a la investigación de herramientas tecnológicas. ¿Cómo puede ayudar la integración de las herramientas que van a ser objeto de estudio a la automatización del proceso de archivo de expedientes de los estudiantes en la Carrera de Ingeniería en Sistemas Computacionales de la Universidad Técnica del Norte?



Fig. 1: Árbol de problemas
Fuente: Propia

Objetivos

Objetivo general.

Investigar la integración de PHP con la base de datos NoSQL Apache CouchDB para el desarrollo del sistema gestión de expedientes para la “Coordinación de la carrera de Ingeniería en Sistemas Computacionales”.

Objetivos específicos

- Analizar la integración del lenguaje PHP con la base de datos NoSQL Apache CouchDB
- Desarrollar el Sistema de Gestión de expedientes de los estudiantes de la CISIC para automatizar la administración de archivos, utilizando PHP y Apache CouchDB.

- Aplicar la metodología Scrum y elaborar la documentación técnica.

Alcance

El proyecto tiene como finalidad realizar el estudio de la integración de PHP con la base de datos NoSQL CouchDB.

Se utilizará los módulos de:

- Módulo de autenticación

La funcionalidad de este módulo consiste en restringir el acceso al sistema a usuarios no autorizados, además de limitar las funciones del sistema de acuerdo al rol que tenga el usuario autenticado.

- Módulo de control por semestres

El módulo permitirá subir la información de los expedientes de los estudiantes.

- Módulo de seguimiento

Este módulo permitirá ingresar, eliminar y actualizar los datos al inicio de cada uno de los semestres.

- Módulo de resultados

Este módulo permitirá presentar las estadísticas de los expedientes estudiantiles.

A continuación, se detalla cómo va a estar conformado el Sistema de Gestión de expedientes de los estudiantes de la carrera de Ingeniería en Sistemas Computacionales, en donde se obtendrá información relevante.

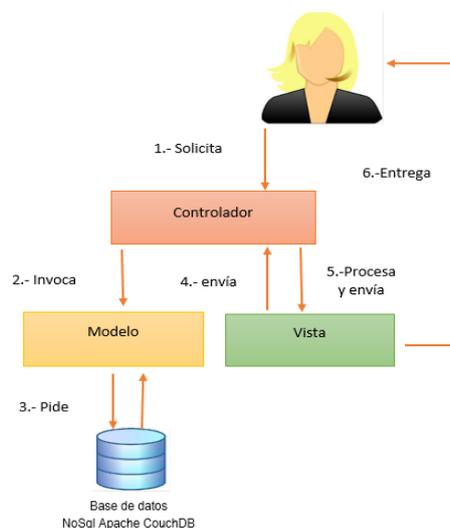


Fig. 2: Arquitectura del Sistema
Fuente: Propia

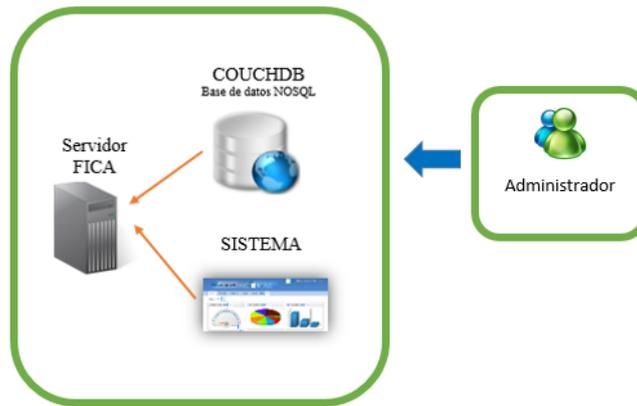


Fig. 3: Diagrama de despliegue
Fuente: Propia

Metodología de Desarrollo

Scrum es como un salvavidas para aquellas empresas que enfrentan dificultades al seguir la metodología de cascada o que ni siquiera están usando metodología alguna para desarrollar su software. Scrum es un marco de referencia para crear software complejo y entregarlo a tiempo de una forma mucho más sencilla. (Dimes, 2015)

Justificación

Mediante este estudio permitirá a los desarrolladores a escoger las herramientas adecuadas que se ajusten de manera eficiente en el momento de crear la aplicación, y ofrezca un buen soporte a las exigencias de la Coordinación de la Carrera de Ingeniería en Sistemas Computacionales por este motivo la integración de PHP y base de datos NoSQL CouchDB a utilizar son herramientas que están dentro de la nueva tecnología, las mismas que posibilitará el ahorro en tiempo y recursos al disponer en forma inmediata, con precisión, garantía y credibilidad de la base de datos en la utilización de este medio tecnológico por parte de los usuarios, además la parte ambiental se verá beneficiada al disminuir notablemente el uso del papel.

CAPÍTULO 1

Marco Teórico

1.1 Introducción

La carrera de Ingeniería en Sistemas Computacionales, continúa formando profesionales competentes, críticos, humanistas, líderes y emprendedores con responsabilidad social, en un entorno altamente propositivo y expectante de cambios sistemáticos y profundos en las áreas científicas y tecnológicas.

En sus 26 años, constantemente ha estado vinculada con el medio social y productivo de la provincia, región y del país, más del 70% de empresas pequeñas, medianas y grandes; públicas y privadas, de la provincia, han recibido a nuestros graduados como elementos de asesoría y apoyo a la gestión de las mismas.

Empresas de cobertura nacional e internacional, dedicadas tanto a hardware, software y comunicaciones, han puesto especial interés en la formación profesional y humanística con la que cuenta nuestro proyecto curricular, observaciones y recomendaciones que han indicado notablemente en su contenido, que cambia y se renueva periódicamente.

“A todos los compañeros estudiantes, administrativos y profesores que somos parte de esta carrera les invito a seguir trabajando por conseguir se haga realidad nuestra visión, a sentir el orgullo de estar en la mejor carrera de Ingeniería en Sistemas del País” (Ing. Pedro Granda – Coordinador CISIC). (UTN-INTRODUCCIÓN, 2010)

1.1.1 Historia de la carrera

El 18 de julio de 1986 se crea la Universidad Técnica del Norte mediante ley número 43 publicada en el registro oficial número 482 y como parte de la misma consta en el decreto de creación la Facultad de Ingeniería Textil.

Dado el avance de la tecnología y la importancia de contar en nuestro medio con profesionales acorde a los cambios tecnológicos, se creó la Escuela de Ingeniería en Sistemas Computacionales (EISIC), mediante resolución del H. Consejo Universitario, en sesión ordinaria del 31 de julio de 1990, como parte de la Facultad de Ingeniería en Ciencias Aplicadas, e inició sus actividades académicas en octubre de 1990.

La primera promoción de profesionales egreso a finales de 1995, en aquella época el sistema era por años, posteriormente a partir del periodo septiembre 2007 – marzo 2008 funcionó la carrera en sistemas semestral a partir del año 2000 (periodo octubre 2000 marzo

2001) está funcionando en sistemas de créditos, la misma que se mantiene hasta la actualidad. (UTN-HISTORIA, 2010)



Fig. 4: Historia Carrera de Ingeniería en Sistemas Computacionales
Fuente: (UTN-HISTORIA, 2010)

1.1.2 Creación de la CISIC

Durante el año 1989, a escasos tres años de la creación oficial de la Universidad Técnica del Norte, en el mundo comenzaban a cristalizar los albores de la globalización por incidencia directa del desarrollo tecnológico, su expansión cobijó también a nuestra región y aquel cuerpo colegiado de hombres visionarios guiados por el Dr. Antonio Posso Salgado, Rector de la Universidad Técnica del Norte, resolvieron realizar un estudio inmediato en el entorno geográfico con el fin de determinar la necesidad de implementar en esta casona Universitaria, una opción académica para todos aquellos jóvenes que aspiren a estudiar estas nuevas tendencias científicas y tecnológicas.

El estudio dio como resultado, la necesidad de crear la Escuela de Ingeniería en Sistemas Computacionales, la elaboración final del proyecto se encargó a la Facultad de Ingeniería Textil, y fue el Ing. René Brown, quien cristalizó el mismo para presentarlo para el análisis correspondiente en el H. Consejo Universitario.

El 31 de Julio de 1990, se aprueba en el H. Consejo Universitario y se emite la respectiva resolución mediante Oficio 202-HCU del 18 de agosto de 1990, comenzando así una nueva era para la Región Norte del País. A través de los medios de comunicación se da la buena nueva e inmediatamente se realiza la convocatoria a las inscripciones a la nueva Escuela de Ingeniería en Sistemas Computacionales que, por cierto, motivo el cambio para que la Facultad cambie su denominación y sea en CIENCIAS APLICADAS.

La expectativa de los aspirantes rebasó los límites y se tuvo una masiva concurrencia, sobrepasando los cuatrocientos ciudadanos, que deseaban estudiar en la EISIC. (UTN-CREACIÓN, 2010)

1.2 Situación actual de la carrera.

Actualmente en la Coordinación de la Carrera de Ingeniería en Sistemas Computacionales, no existe un registro digital de los expedientes de los estudiantes, lo que crea una demora en la verificación de documentos de forma física en los archivos de la carrera, existiendo inconformidad de los estudiantes en la obtención rápida y oportuna de la información.

1.3 Proyección

1.3.1 Sistema de gestión de expedientes

Permite almacenar una serie de documentos relacionados en expedientes del estudiante, de esta manera se puede mejorar la organización y facilitar un mejor resultado en la búsqueda de archivos solicitados por el estudiante obteniendo una mejor atención y satisfacción por el usuario.



Fig. 5: Digitalización de documentos
Fuente: (Cividas, 2015)

1.3.2 Beneficios

- Rapidez en la búsqueda de expedientes de los estudiantes.
- Reducir el riesgo de pérdida de la documentación.
- Reducir el uso de papel.
- Solo el administrador tiene acceso a los expedientes del sistema.

1.4 Estudio de la integración del lenguaje PHP.

1.4.1 Programación PHP

Es un lenguaje de programación de código abierto de uso libre y gratuito, adecuado para desarrollo web utilizado para generar páginas web dinámicas y que puede ser incrustado en HTML.

El lenguaje PHP se procesa en servidores, que son potentes ordenadores con un software y hardware especial. Se envían los datos de la solicitud al servidor que los procesa y este los devuelve en una página HTML como si fuera estática, es compatible con todos los navegadores. (Gutiérrez González)

1.5 Herramientas de programación PHP

1.5.1 Laravel PHP.

Laravel es gratuito, de código abierto y uno de los framework web PHP más populares basados en el patrón arquitectónico model-view-controller (MVC). Fue creado por Taylor Otwell, con la intención de reducir el costo del desarrollo inicial y mejorar la calidad de su código al definir las prácticas de diseño estándar de la industria. Usando Laravel, puede ahorrar horas de tiempo de desarrollo y reducir miles de líneas de código comparando con PHP sin formato. Desde su primer lanzamiento en junio de 2011, Laravel ha recorrido un largo camino ya que el lenguaje PHP está evolucionando. (Nilanchala, 2017)



Fig. 6: Logo framework Laravel
Fuente: (TAYLOR, 2016)

1.5.2 Arquitectura

Laravel maneja un patrón de arquitectura MVC tradicional, lo que separa la lógica de negocio, la de entrada y presentación asociada a una interfaz gráfica donde muestra la terminación del proyecto.

1.5.3 Capas MVC en Laravel

Capa del Modelo

El modelo en cualquier otro MVC framework, es el componente responsable de la comunicación de la base de datos. Es aquí donde interviene Eloquent que es una clase que se hereda a los modelos en Laravel, para trabajar de manera simple con los objetos de la base de datos. Obviamente el uso de Eloquent es opcional, pues también ese framework dispone de otros recursos que facilita interactuar con los datos, o específicamente la creación de modelos. (Cogneau, 2012)

Capa de Vista.

Laravel, usa su propio motor de plantillas llamado Blade, con el objetivo de dar un código limpio y fácil de comprender en las Vistas. Utiliza código PHP plano y además incluye un sistema de Caché que lo hace mucho más rápido. Para la creación de estas plantillas se debe crear el archivo dentro de la carpeta / view con la extensión nombrevista.blade.php (Cogneau, 2012).

Capa del Controlador

El controlador es donde la lógica de negocio está ubicada y gracias a esto se tienen funcionalidades como: recuperar todas las entradas de la base de datos para enumerarlas, actualizar, eliminar o realizar búsquedas de la información en las tablas de las bases de datos dado un identificador, añadir nuevos registros a la base de datos y crear mensajes de conformación de las funciones básicas. Esta capa permite organizar el código en clases sin tener que escribirlo todo en las rutas. Todos los controladores deben extenderse de la clase BaseController (Cogneau, 2012).

Trabaja como intermediario entre la vista y el modelo brindando un buen trabajo con el manejo de todos los datos.

1.5.4 Arquitectura MVC Laravel

Denominada también tres capas, es un patrón de arquitectura de software encargado de separar la lógica de negocio de la interfaz del usuario en tres componentes distintos y es el más utilizado en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página, el modelo es la abstracción de la Base de Datos y la lógica de negocios y el controlador es el responsable de recibir los eventos de entrada desde la vista. (Devacademy, 2015)

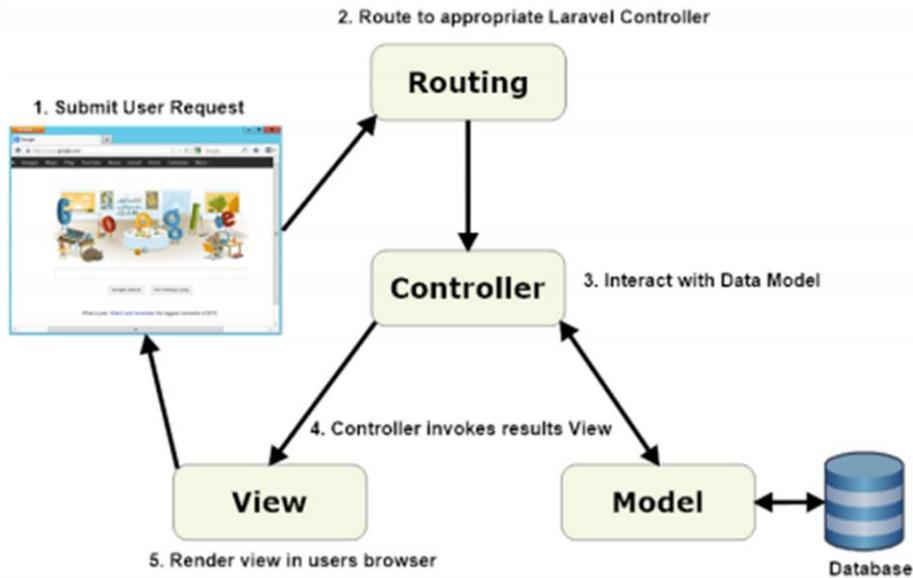


Fig. 7: Petición típica MVC en Laravel
Fuente: (LaravelBook, 2014)

Características de Laravel

- Sistemas de envasado modular con un gestor de dependencia dedicada.
- Diferentes formas de acceso a bases de datos relacionados, aunque no exista un enrutamiento.
- Utilidades que ayudan en la implementación de las aplicaciones y al mantenimiento de las mismas.
- Facilidad de autenticación, proporcionando una interfaz sencilla y fácil de usar. (Xnechijli01, 2016)

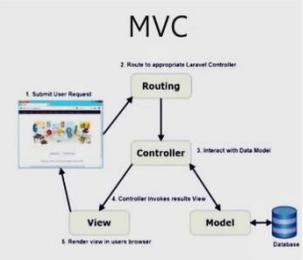
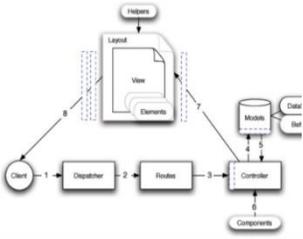
Ventajas

- Reducción de costos y tiempos en el desarrollo y mantenimiento.
- Flexible y adaptable no solo al MVC (Modelo Vista Controlador) sino que para reducir código propone usar "Routes with closures".
- Buena y abundante documentación sobre todo en el sitio oficial.
- Posee una amplia comunidad y foros.
- Es modular y con un amplio sistema de paquetes y drivers con el que se puede extender la funcionalidad de forma fácil, robusta y segura.
- Hace que el manejo de los datos en Laravel no sea complejo: mediante Eloquent (que es un ORM basado en el patrón active record) la interacción con las bases de datos.

- Facilita el manejo de ruteo de nuestra aplicación como así también la generación de url amigables y control de enlaces auto-actualizables lo que hace más fácil el mantenimiento de un sitio web.
- También cuenta con una herramienta de interfaces de líneas de comando llamada Artisan que permite programar tareas programadas como por ejemplo migraciones, pruebas programadas, etcétera. (freelancer, 2016)

1.5.5 Comparativa de Laravel con otros Framework de PHP

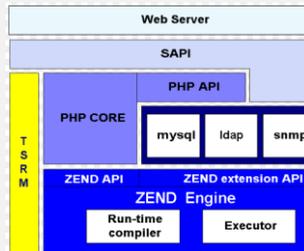
Existen muchos framework PHP, de los cuales se puede decir que el más apto para la realización de proyectos es Laravel seguido de otros frameworks PHP por la cual se realiza un cuadro comparativo con una descripción de cada uno de los frameworks como se muestra a continuación en la siguiente tabla.

FRAMEWORK	DESARROLLO	ARQUITECTURA	CARACTERÍSTICAS	VENTAJAS	DESVENTAJAS	VISIÓN
 <p>LARAVEL PHP</p>	<p>Taylor Otwell 2011</p>	 <p>The diagram illustrates the MVC (Model-View-Controller) pattern. It shows a flow starting from a 'Submit User Request' (represented by a browser icon) to 'Routing', then to 'Controller', then to 'View', and finally back to the browser as 'Render view in users browser'. The 'Controller' also interacts with a 'Model' (represented by a database icon), and the 'Model' interacts with a 'Database'.</p>	<ul style="list-style-type: none"> - Posee un conjunto poderoso de librerías. - Autenticación simple. - Basado en Composer. - Soporte para el caché. - Soporte para MVC. - Usa componentes de Symfony. 	<ul style="list-style-type: none"> - Es de código abierto u Open Source. - Facilita en manejo de ruteo de la aplicación. - Comunidad extremadamente activa y en continuo crecimiento. - Reducción de costos y tiempos en el desarrollo y mantenimiento. - Documentación que existe es sencilla y completa. 	<ul style="list-style-type: none"> - Se utiliza una consola para realizar ciertas funciones. - La instalación se hace mediante Composer. - El aprendizaje puede ser complejo si no has utilizado un framework. 	<p>Laravel es un framework que permite el uso de una sintaxis elegante y expresiva. Crea código de forma sencilla y permite multitud de funcionalidades.</p>
 <p>CAKE PHP</p>	<p>Michal Tatarynowicz 2005</p>	 <p>The diagram shows the CakePHP architecture. It starts with 'Users' interacting with a 'Layout' containing 'View' and 'Elements'. The flow goes from 'Users' to 'Dispatcher', then to 'Router', then to 'Controller', which interacts with 'Model' and 'Database'. The 'Controller' also interacts with 'Components'.</p>	<ul style="list-style-type: none"> - Compatibilidad con las versiones 4 y 4 de PHP. - Integra CRUD para la interacción con bases de datos. 	<ul style="list-style-type: none"> - Fácil de aprender. - Implementación rápida gracias a su estructura y los 	<ul style="list-style-type: none"> - Exceso de consultas en proyectos en los que el número de tablas es muy 	<p>El principal objetivo de CakePHP, es que puedas trabajar de forma estructurada y rápida, sin pérdida de flexibilidad, con</p>

- Arquitectura MVC. métodos propios del elevado y hay CakePHP el desarrollo
- Ayudas para AJAX, framework. muchas web ya no es monótono
- JavaScript, HTML, forms y más. - Migración entre relaciones, el porque se ofrece las
- Funciona en cualquier bien configurado el ORM de herramientas para que
- subdirectorio de un sitio servidor. CakePHP puede empieces a escribir el
- web, con poca o ninguna - Se puede encontrar excesivas necesita; la lógica
- configuración de por internet plugins consultas. específica de tu aplicación
- Apache. que realizan una - Tamaño de los
- Componentes de tarea específica sin ficheros del
- seguridad y sesión. tener que programar framework son
- Menos configuración. nada. demasiado
- pesados.
- Es una de las
- plataformas más
- lentas en cuanto
- a rendimiento y
- la carga de los
- códigos.



Andi
Gutmans y
Zeev
Suraski,
1999



- Basado en PHP. - Está totalmente - Difícil de Zend tiene administración
- Soporte para PHP 4 y orientada a objetos. instalar. de PHP, está basado en
- PHP 5. - Permite el uso de - Complejidad de 100% en SPL (Standard
- Manual de PHP cualquiera de sus uso y PHP
- integrado. librerías en vez de programación. library) y sencillo aplicar
- librerías JQuery, Bootstrap

	<ul style="list-style-type: none"> - Está orientada a objetos. - Soporte para gestión de grandes proyectos de desarrollo. - Ofrece una capa de abstracción a bases de datos fáciles de usar. <ul style="list-style-type: none"> - Contiene muchos componentes. - Comunidad y documentación. - Garantiza continuidad del Framework. <ul style="list-style-type: none"> - Aprendizaje y uso de layouts para darle diseño de manera fácil.
--	--

Tabla 1: Comparativa de Laravel con otros Framework de PHP

Fuente: Propia

1.5.6 Análisis de la comparativa de las principales características de los Framework PHP.

Existen muchos framework PHP, de los cuales se puede decir que el más apto para la realización de proyectos es Laravel debido a las características que posee.

Mediante un estudio comparativo de los principales framework PHP, se hizo la comparativa de tres de ellos y se muestran los siguientes resultados.

Características	Laravel PHP	CakePHP	ZendPHP
Incorpora el patrón MVC orientado a objetos.	X	X	X
Operaciones CRUD asociadas al patrón Active Record.	X	X	
Mapeado de objetos a bases de datos relacionales (ORM)	X	X	X
Independencia del manejador de base de datos.	X	X	X
Uso de plantillas PHP.	X	X	X
Manejo propio de sesiones de usuario.	X	X	
Almacenamiento de caché en las vistas.	X	X	
Almacenamiento en caché de configuraciones de las aplicaciones.	X		
Integración con otras herramientas a través de plugins.	X	X	
Licencias libres.	X11	X11	BSD

Tabla 2: Comparación frameworks PHP.
Fuente: (Colimba, 2016)

1.5.7 Análisis

De acuerdo con la tabla comparativa, el framework PHP con mayores características es Laravel PHP, seguido de Cake PHP y finalmente Zend PHP, lo que se puede mostrar en el siguiente cuadro estadístico.

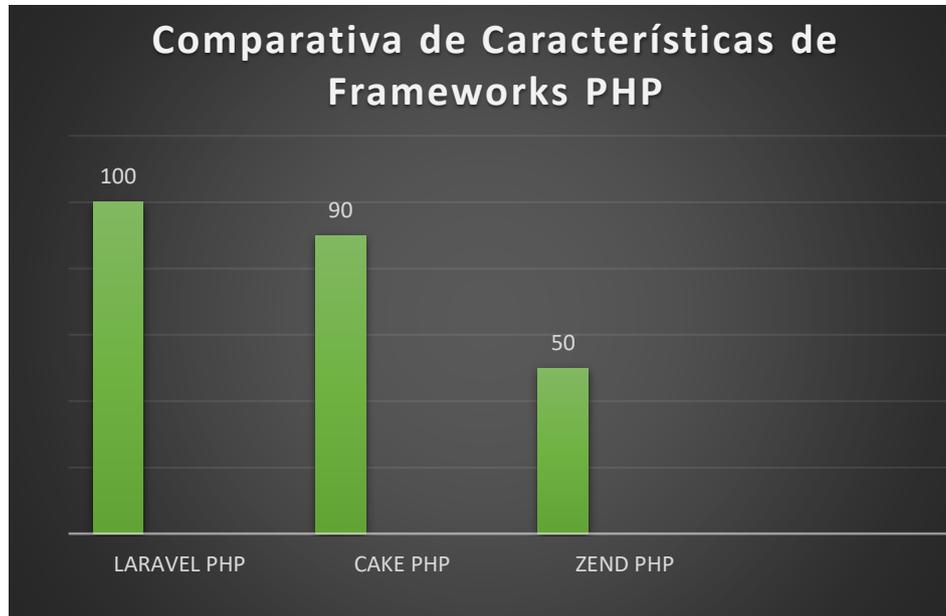


Fig. 8: Estadística de comparación de Framework PHP.
Fuente: Propia

1.6 Base de datos NoSQL Apache CouchDB.

1.6.1 Base de datos NoSQL

NoSQL tuvo inicio con la llegada de la web 2.0 ya que hasta ese momento sólo subían contenidos a la red aquellas empresas que tenían un portal, pero con la llegada de aplicaciones como Facebook, YouTube o twitter, cualquier usuario podía subir contenido, provocando así un crecimiento exponencial de los datos. (Guest, 2016)

La base de datos NoSQL es hablar de estructuras que nos permitan almacenar información en aquellas situaciones en las que las bases de datos relacionales generan ciertos problemas debido principalmente a problemas de escalabilidad y rendimiento de las bases de datos relacionales donde se dan citas miles de usuarios y millones de consultas diarias.

Estos son una amplia gama de SGBD que difieren del modelo clásico de gestión de bases relacionales en aspectos importantes. El aspecto más destacado es que “no sólo SQL”, ya que también pueden soportar lenguajes de consulta de tipo SQL. (Caballero & Montoya, 2016)

NoSQL son sistemas de almacenamiento de información que no cumplen con el esquema entidad-relación, no utiliza una estructura de datos en forma de tabla donde se van almacenando los datos, sino que para el almacenamiento hacen uso de otros formatos como clave-valor, mapeo de columnas o grafos.



Fig. 9: Logo NoSQL
Fuente: (Editor, Devellar, 2014)

¿Qué hace diferente a NoSQL?

La clave entre la base de datos NoSQL con las bases de datos relacionales tradicionales es que NoSQL tiene una forma de almacenamiento no estructurado que significa que no tiene una estructura de tabla fija, como lo indica en la siguiente imagen.

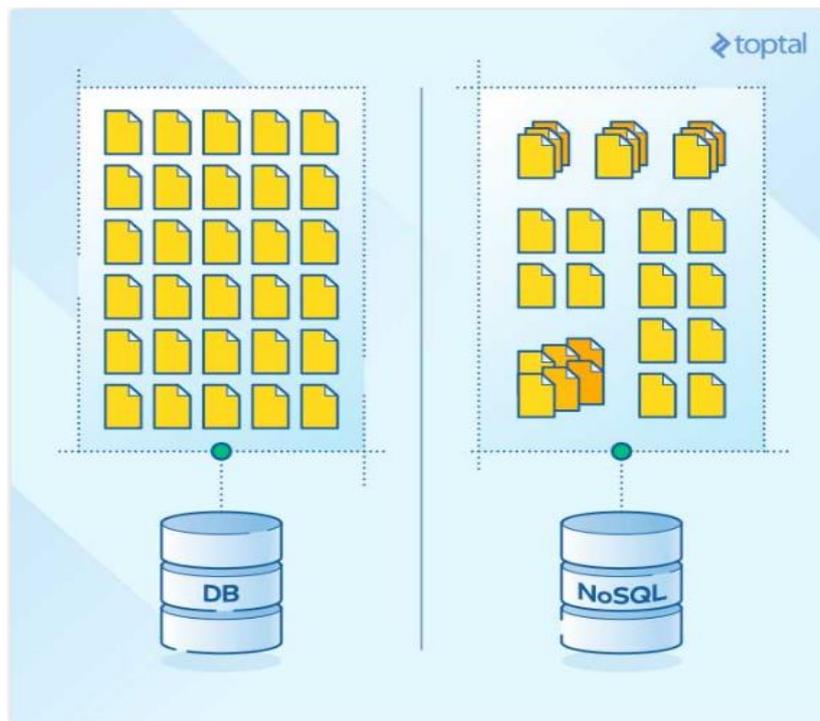


Fig. 10: Diferencia del almacenamiento de la base de datos NoSQL con la base de datos SQL
Fuente: (Martín, 2017)

Comparación entre la base de datos NoSQL y SQL.

NoSQL	SQL
Almacena datos relacionados en JSON.	Almacena datos relacionados en las tablas.
Pueden almacenar datos sin especificar un esquema.	Requiere un esquema que defina las tablas antes de su uso.
Por lo general se deben “desnormalizar” por lo que la información sobre un artículo está contenido en un solo documento.	Fomenta la normalización que defina las tablas antes de su uso.
No debe requerir JOINS (suponiendo que utiliza los documentos desnormalizados).	Tabla de soporte JOIN para recuperar datos relacionados de varias tablas en un solo comando.
Permite guardar cualquier dato, en cualquier lugar, en cualquier momento y sin verificación.	Aplica las reglas de integridad de datos.
Proporciona un excelente rendimiento y escalabilidad.	Posibilidad de escalar.
Utiliza objetos JSON para consultas de datos.	Utiliza un lenguaje declarativo de gran alcance para las consultas.
Es una tecnología más reciente.	Ofrece apoyo, experiencia y herramientas.

Tabla 3: Cuadro comparativo de las bases de datos NoSQL y SQL.
Fuente: (Imaginanet, 2015)

1.7 Base de datos Documental

A diferencia de las filas y las columnas, las bases de datos NoSQL mantienen los datos en documento. Estos documentos siguen un mínimo de reglas estándares para que la base de datos pueda entenderlo durante el post-procesamiento.

Su formato podría ser JSON, XML, pero por buenas razones el formato elegido para la base de datos documental NoSQL es JSON. JSON es un documento más compacto y más legible.

La ventaja principal de JSON reside en su facilidad de uso, tanto para las personas como por los ordenadores. El formato JSON tiene tres fundamentos básicos:

- Pares de valores clave o atributos: JSON es almacenado en valores clave, a estos se les llama a veces atributos, las claves son cadenas simples y los valores pueden ser de cualquier tipo.
- Incrustación de objetos JSON: los valores clave también pueden ser objetos, lo que permite crear una jerarquía de objetos, colocar un objeto JSON dentro de otro se denomina modelo de datos incrustados, en base de datos documentales.

- Matrices: son un lenguaje de programación natural en todos los lenguajes de programación y estructura de datos. El formato JSON también admite almacenamiento de matrices como valores contra una clave. (KYOCERA, 2017)

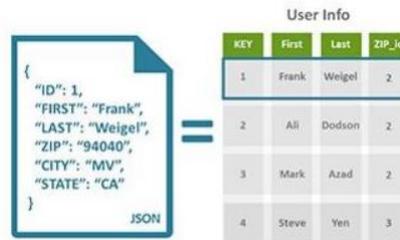


Fig. 11: Base de datos documental
Fuente:

1.8 CouchDB

1.8.1 Historia de CouchDB

Es un proyecto creado por Damien Katz en abril de 2005, lo definió como un “sistema de almacenamiento para una base de datos objeto de gran escala”. Sus objetivos eran que la base de datos se transformara en la base de datos de la Internet y que se diseñara desde un principio para prestar servicios a aplicaciones web. En febrero de 2008, pasó a ser un proyecto de Apache Incubator y la licencia se cambió a la Apache License, pocos meses más tarde, se graduó como proyecto top-level. Así se llegó a la publicación de la primera versión estable en julio de 2010. A principios de 2012, Damien Katz abandonó el proyecto a fin de concentrarse en Couchbase. El proyecto ha seguido adelante, liberando su versión 1.2 en abril de 2012. (CouchDB, 2013)

1.8.2 Definición

CouchDB es un gestor de base de datos de código abierto, almacena sus datos con documentos JSON, accede a sus documentos con un navegador web a través de HTTP, consulta, combina y transforma los documentos con JavaScript. CouchDB no almacena los datos y sus relaciones en tablas, cada base de datos es una colección de documentos independientes con sus propios datos y esquema. Una aplicación puede acceder a múltiples bases de datos. (CouchDB A. , 2014)

1.8.3 Arquitectura:

- HTTP CLIENT: permite realizar al cliente peticiones al servidor de la base de datos.
- EARLANG HTTP: permite el control y gestión de HTTP como gestión de peticiones, concurrencia, cabeceras HTTP, encriptado, autenticación.

- MOD_COUCH: es el core de la máquina virtual de Erlang de CouchDB, este componente implementa la API y es el encargado de entender y gestionar las diferencias operacionales solicitadas por el cliente HTTP.
- VIEW_ENGINE: realiza este componente todo el proceso de las vistas, el diseño de los documentos, la indexación utilizando arboles B.
- STORAGE_ENGINE: este componente se comunica activamente con ViewEngine y el Replicador ya que las vistas son incrementales y deben reflejar los cambios escritos a disco, gestionan las escrituras en disco.
- REPLICADOR: permite a la replicación de los documentos, ayuda a la distribución entre diferentes dispositivos. Solo actúa si es activado manual o programando, pero no funciona de manera continua. (CASTILLO, 2016)

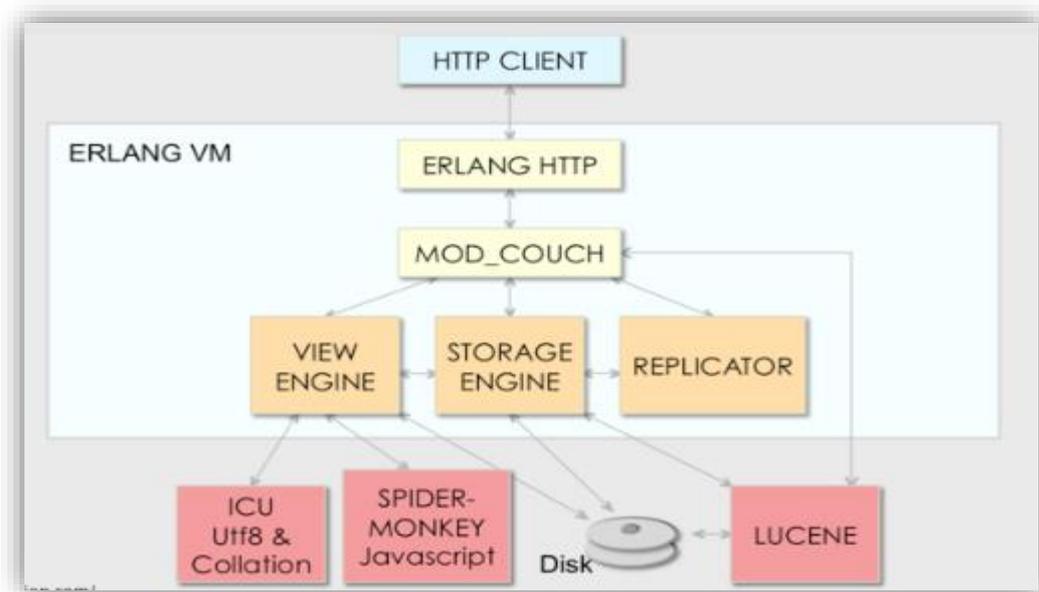


Fig. 12: Arquitectura CouchDB
Fuente: (CASTILLO, 2016)

1.8.4 Características CouchDB

CARACTERÍSTICAS DE COUCHDB

- Replicación fácil entre servidores a través de instancias.
- Soporte para resolución de conflictos y configuración del master.
- Actualizaciones de alimentación de datos por suscripción.
- Indexación, búsqueda y recuperación rápida.
- Interfaz web RESTful.
- Los documentos son accesibles a través de navegadores y los índices pueden ser consultados a través de HTTP.
- Operaciones de indexación, combinación y transformación realizadas con JavaScript.
- Operaciones para crear, leer, actualizar, borrar documentos simples (CRUD).
- MapReduce avanzado

Tabla 4: Características de CouchDB
Fuente: (Rouse, 2014)

1.8.5 Documentos CouchDB

DOCUMENTOS COUCHDB

Documentos con identificaciones únicas, y formado por objetos hechos de varios campos.

Los valores almacenados pueden ser cadenas, números, fechas, booleanos, listas, mapas u otros tipos de datos.

No hay límite para el número de campos.

Incluye metadatos. Ejemplo: números de revisión.

No impone mecanismos de bloqueo en los datos.

Tabla 5: Documentos CouchDB
Fuente: (Espinoza, 2016)

1.8.6 Ventajas

- CouchDB logra consistencia eventual entre las bases de datos usando replicación incremental, un proceso en que los cambios en los documentos son periódicamente copiados entre los servidores.
- CouchDB no almacena sus datos y sus relaciones en tablas.
- Cada base de datos en una colección de documentos independientes.

- Cada documento mantiene sus propios datos y su esquema auto contenido.
- La replicación en CouchDB es una de estas piezas básicas.
- El protocolo HTTP es transparente y está estudiado en profundidad.
- Esta base de datos no tiene un esquema por lo que contiene datos semiestructurados y permite la inclusión de nuevos tipos de documentos sin implicar la modificación de los anteriores. (Colbert, 2014)

1.8.7 Desventajas

- El código abierto puede significar una mancha en el soporte para las empresas.
- No están lo suficientemente maduros para algunas empresas.
- Generalmente un vendedor de código abierto no tiene el alcance global, servicios de soporte, y la credibilidad de Oracle a IBM.
- Los críticos señalan la falta de madurez de NoSQL y los posibles problemas de inestabilidad, mientras que citan la madurez, y una gran funcionalidad y estabilidad de los RDBMS. (Colbert, 2014)

1.9 JSON

Java Script Object Notation - Notación de Objetos de Java Script (JSON), es un formato de texto ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML. (Wikipedia, 2018) Una de las mayores ventajas que tiene el uso de JSON es que puede ser leído por cualquier lenguaje de programación. Por lo tanto, puede ser usado para el intercambio de información entre distintas tecnologías.

1.10 APACHE

Es un software libre, apache es un servidor web HTTP de código abierto, multiplataforma es decir puede trabajar con diferentes sistemas operativos y mantener un excelente rendimiento, siendo este un servidor seguro, eficiente y extensible que proporcionen servicios HTTP en sincronización con los estándares HTTP actuales. (APACHE, APACHE HTTP SERVER PROJECT, 2017)

Sus características son:

- Multiplataforma.
- Modular se adapta a diferentes entornos y necesidades con los diferentes módulos de apoyo que proporciona y con la API de programación de módulos.

- Extensible ya que es modular se han desarrollado diversas extensiones entre las que se destaca PHP, un lenguaje de programación del lado del servidor. (APACHE, 2014)



Fig. 13: Logo de Apache
Fuente: (Apache, 2014)

1.11 Servicios REST

REST (Representational State Transfer) es un estilo de arquitectura de software para sistemas hipermedias distribuidos tales como la Web. El término fue introducido en la tesis doctoral de Roy Fielding en 2000, quien es uno de los principales autores de la especificación de HTTP. (Navarro Marset)

Los servicios REST son un estilo de arquitectura para desarrollar servicios, tienen las siguientes características.

Ciente-Servidor: definen una interface de comunicación entre cliente y servidor separados completamente las responsabilidades entre ambas partes.



Fig. 14: Servicios Rest – Cliente/Servidor
Fuente: (Álvarez Caules, 2013)

Sin estado: son servicios web que no mantienen estado asociado al cliente. Cada petición que se realizan a ellos es completamente independientes de las siguientes. Todas las llamadas al servidor serán idénticas.

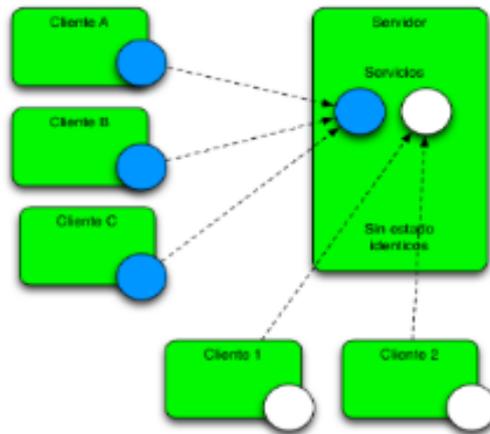


Fig. 15: Servicios Rest - Sin Estado
Fuente: (Álvarez Caules, 2013)

Cache: los servicios Rest se pueden cachear de tal forma que una vez realizada la primera petición al servicio el resto puedan apoyarse en la cache si fuera necesario.

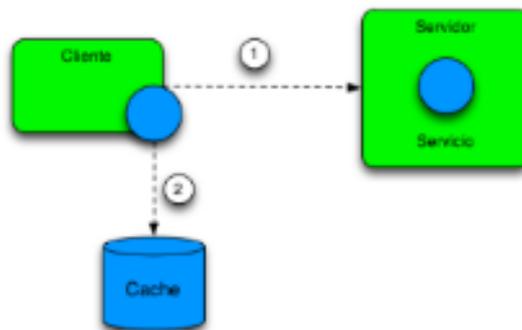


Fig. 16: Servicios Rest – Cache
Fuente: (Álvarez Caules, 2013)

Servicios uniformes: los servicios REST compartirán una forma de invocación y métodos uniformes que son: GET, PUT, POST, DELETE.

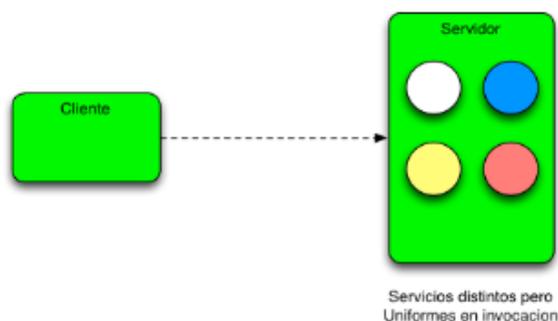


Fig. 17: Servicios Rest – Servicios uniformes
Fuentes: (Álvarez Caules, 2013)

GET: se usa para obtener información del servidor.

PUT: actualiza un archivo que ha sido subido al servidor.

POST: es el encargado de crear un nuevo recurso.

DELETE: es el único que debe ser usado para borrar un recurso del servidor.

(CESC1989, 2013)

Arquitectura en capas: los servicios REST están orientados hacia la escalabilidad y un cliente REST no es capaz de distinguir entre, si está realizando una petición directamente al servidor o se lo está devolviendo un sistema de cache intermedio.

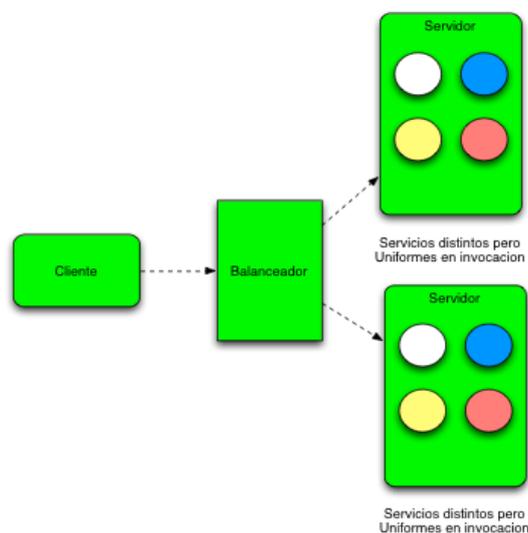


Fig. 18: Servicios Rest – Arquitectura en capas
Fuente: (Álvarez Caules, 2013)

1.12 HTML5

Hyper Text Markup Language – Lenguaje de marcado de hipertexto, es el último estándar que permite mostrar e interactuar con páginas web. Principalmente hace que sea más fácil para los desarrolladores web, reduce la necesidad de Java Script y código extra con nuevos elementos nativos, proporciona consistencia entre navegadores y dispositivos, realiza su trabajo lo más transparente posible. (Refman, 2016)

1.13 CSS3

Cascading Style Sheets – Hojas de estilo en cascada, es un lenguaje usado para definir las reglas y estilos de presentación de un documento estructurado escrito en HTML. (MediaWiki, 2015)

1.14 Estudio de la Integración del lenguaje PHP y CouchDB

Para el estudio de integración de la base de datos NoSQL CouchDB con el lenguaje de programación PHP, se presentan los gráficos de arquitectura del sistema y de integración. Además es necesario realizar la instalación de CouchDB, para esto se debe revisar el manual de usuario.

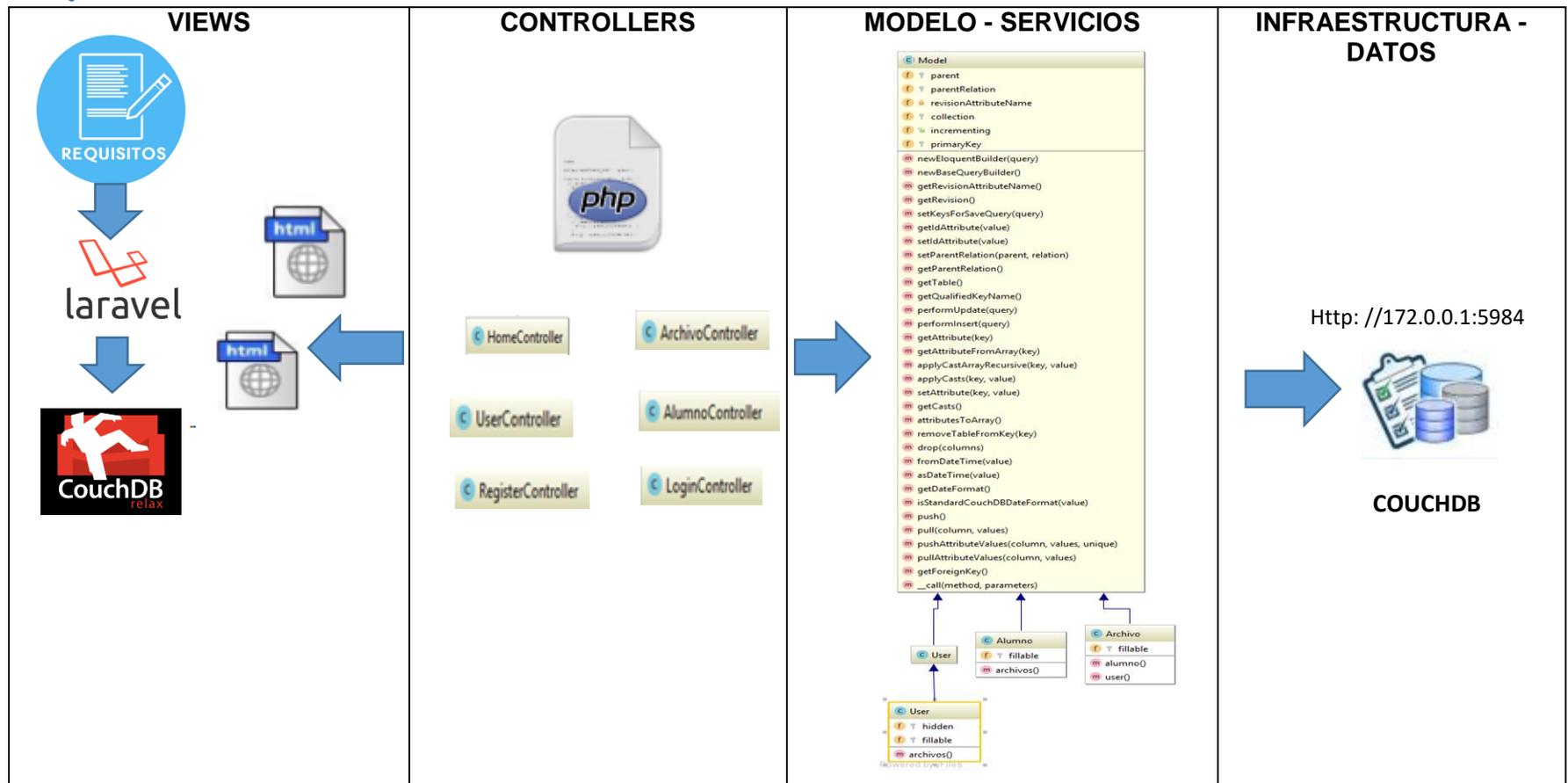


Tabla 6: Arquitectura del Sistema.
Fuente: Propia.

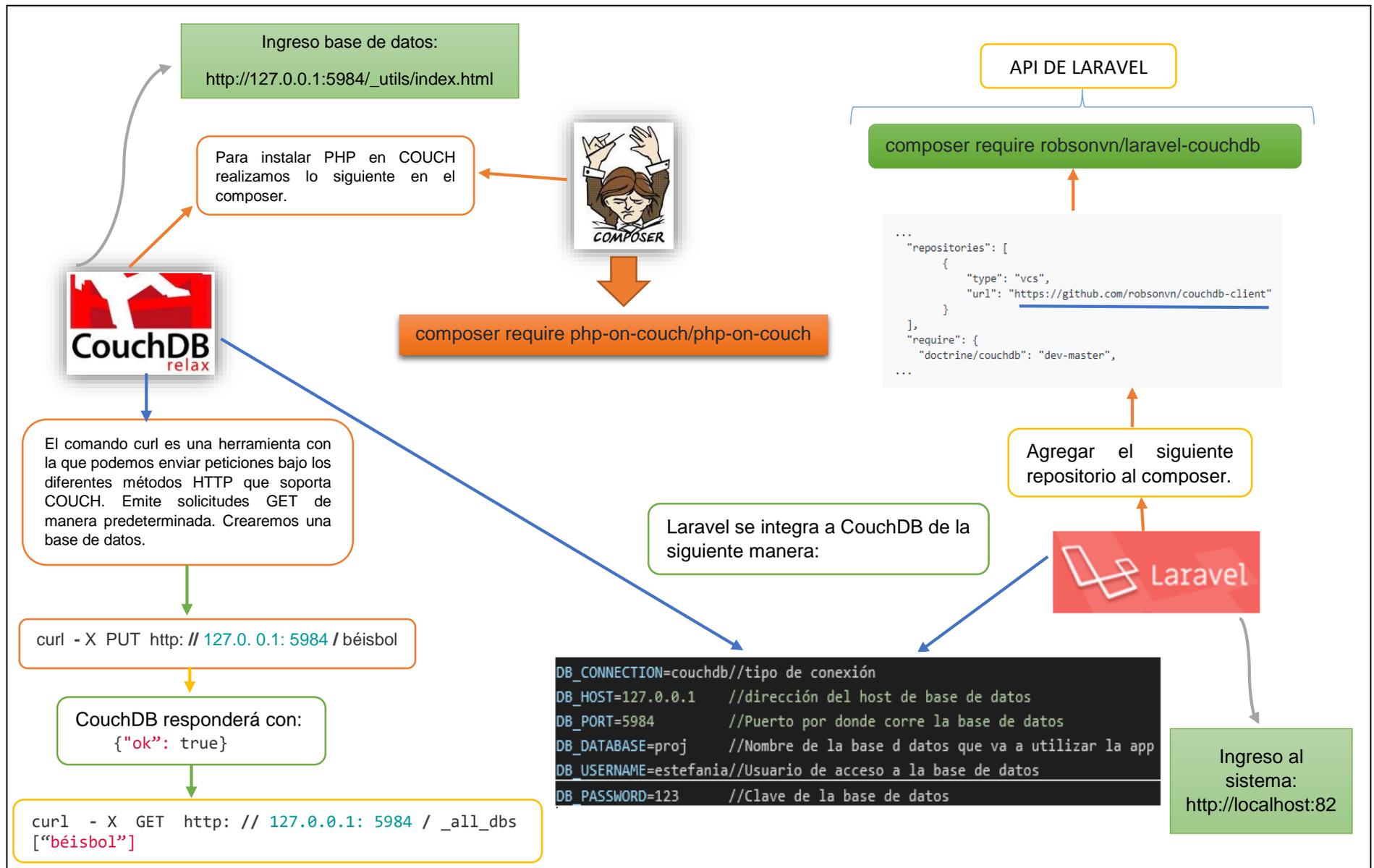


Fig. 19: Diagrama de integración de la base CouchDB con Laravel PHP.

Fuente: Propia.

En la Fig.19 indica la manera como se integra CouchDB y el lenguaje de PHP, CouchDB su utilidad curl, es una forma de comunicarse con la base de datos, esta herramienta de línea de comandos puede enviar peticiones bajo los diferentes métodos HTTP que soporta la base de datos, se utiliza para conectarse de una forma **no gráfica**. Para instalar PHP en CouchDB, se coloca el siguiente comando en composer luego de su instalación: **composer require php-on-couch/php-on-couch**.

Debido a que necesitamos una interfaz gráfica para realizar el proyecto y facilitar el trabajo, se utilizó el framework Laravel PHP luego de una comparativa con otras herramientas, este incluye su propia API, lo que ayuda a la estructura y construcción del proyecto. Para utilizar este framework, agregamos a composer el siguiente repositorio <https://github.com/robsonvn/couchdb-client> y **composer require robsonvn/laravel-couchdb**. De esta manera Laravel se integra a CouchDB de la siguiente manera:

```
DB_CONNECTION = couchdb //tipo de conexión
DB_HOST=127.0.0.1 //dirección del host de base de datos
DB_PORT=5984 //Puerto por donde corre la base de datos
DB_DATABASE=proj //Nombre de la base de datos que se va a utilizar en la app
DB_USERNAME=estefania //Usuario de acceso a la base de datos
DB_PASSWORD=123 //Clave de la base de datos.
```

Luego que se ha realizado la configuración e instalación de las herramientas a utilizar, se explica con un ejemplo la integración de la base de datos CouchDB con PHP:

Se crea el proyecto con el nombre **conexión** con las siguientes clases PHP, en un editor de código.

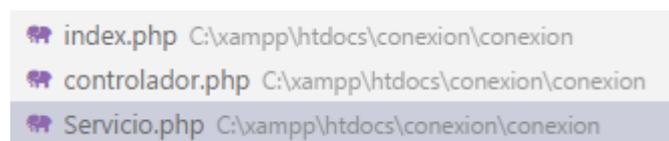


Fig. 20: Clases PHP, del ejemplo Conexión
Fuente: Propia

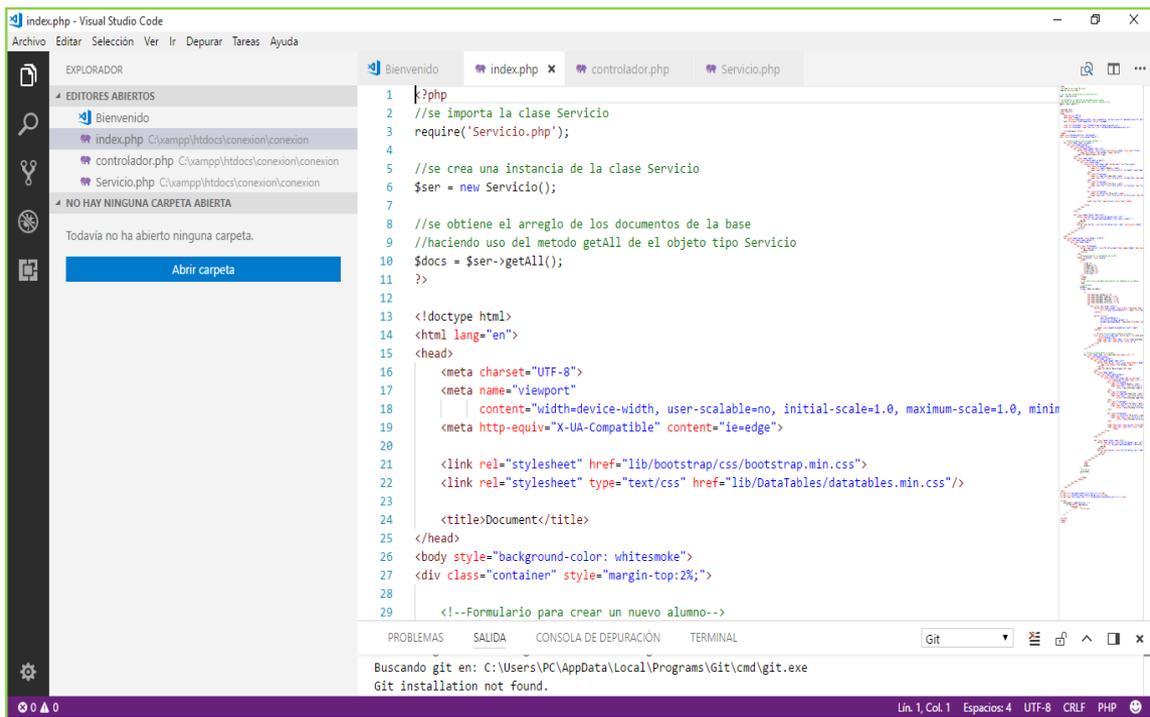
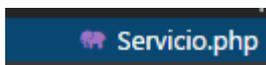


Fig. 21: Proyecto ejemplo, conexión
Fuente: Propia

Para realizar la aplicación es necesario crear tres clases: servicio.php, controlador.php y el index.php. En la clase servicio.php creamos los parámetros de conexión a la base de datos de la siguiente forma:



```

<?php
//Se carga el autoloader generado por composer para poder hacer uso de
las clases ubicadas en vendor.
require "vendor/autoload.php";

//Importa la clase CouchClient para poder usarla.
use PHPOnCouch\CouchClient;

class Servicio
{
    private $couch_dsn; // Url del servidor de bd
    private $couch_db; // Nombre de la bd
    private $client; //Instancia de CouchClient para realizar las
    peticiones al servidor de bd.
    private $options=array(); //Array para las opciones de conexión
    (usuario y contraseña del servidor)

```

```

//Constructor que inicializa los atributos del servicio
function __construct()
{
    $this->couch_dsn = "http://localhost:5984/";
    //Colocamos el nombre de la BD en este caso cont1
    $this->couch_db = "cont1";
    //Ingresamos el usuario y la contraseña que creamos en la BD
    CouchDB.
    $this->options=[ 'username'=>'estefania', 'password'=>'123' ];
    $this->client = new CouchClient($this->couch_dsn, $this-
    >couch_db,$this->options);
    //Se indica la url se servidor y la bd a la instancia de
    CouchClient
}

```

```

//Método para crear un nuevo documento tipo alumno en la bd
//Se instancia un objeto stdClass el cual contendrá la información del
documento a almacenar en la bd.
//Recibe como parámetros los atributos del documento (los datos del
alumno)

public function crear($nombre, $apellido, $telefono, $direccion)
{
    $doc = new stdClass();
    $doc->nombre = $nombre;
    $doc->apellido = $apellido;
    $doc->telefono = $telefono;
    $doc->direccion = $direccion;
    $doc->type = "alumno";
    try {
        $this->client->storeDoc($doc); //Método que permite guardar
el documento en la base
        //Recibe como parámetro un objeto tipo stdClass
    } catch (Exception $e) {
        die("Unable to store the document : " . $e->getMessage());
    }
}

```

```

//Método para editar un documento guardado en la bd
//Recibe como parámetros los nuevos valores para los atributos del
documento y el id del
//Documento a ser actualizado
public function editar($nombre, $apellido, $telefono, $direccion,$id)
{
    //Se busca el documento a ser actualizado por su id
    $doc=$this->client->getDoc($id);
    $doc->_id = $id;
    $doc->nombre = $nombre;
    $doc->apellido = $apellido;
    $doc->telefono = $telefono;
    $doc->direccion = $direccion;
}

```

```

try {
    $this->client->storeDoc($doc);
    //Se guarda el documento con los nuevos datos
} catch (Exception $e) {
    die("Unable to store the document : " . $e->getMessage());
}
}

```

```

//Método que retorna un arreglo de objetos stdClass que representan a
los documentos, almacenados en la bd.
public function getAll(){
    try{
        $rows= $doc1=$this->client->getAllDocs()->rows;
        //Obtiene un arreglo con la información de los documentos almacenados
en la bd

        $docs=array(); //Array para guardar los objetos stdClass

        //Bucle que recorre el arreglo $rows
        foreach ($rows as $key){
            //Se obtiene el documento mediante el id y se lo agrega al
arreglo $docs
            array_push($docs,$this->client->getDoc($key->key));
        }
        unset($key);
        return $docs;
    }catch (Exception $e){
        die("Unable to get documents : " . $e->getMessage());
    }
}
}

```

```

//Método que permite eliminar un documento de la bd
//Recibe como parámetro el id del documento a eliminar
    public function borrar($iddoc){
        try{
            $doc=$this->client->getDoc($iddoc);
            $this->client->deleteDoc($doc); //Método que permite
eliminar el documento
        }catch (Exception $e){
            die("Unable to delete the document : " . $e->getMessage());
        }
    }
}

```

Creación de la clase **controlador.php** el cual tiene los Métodos http: GET POST PUT DELETE, esto permitirá trabajar de forma paralela con CouchDB, las entradas y salidas de datos serán en formato JSON, ya que CouchDB por servicio nativo presenta respuestas en formato JSON y presenta una API RESTFull.

 controlador.php

```

<?php
//Importar clase Servicio
require ('Servicio.php');

//Crear instancia de Servicio
$serv=new Servicio();

//Acción para crear un Alumno
//si $_POST tiene un elemento 'acción' con el valor 'crear'
if($_POST['accion']==='crear'){
    $nombre=$_POST['nombre'];
    $apellido=$_POST['apellido'];
    $telefono=$_POST['telefono'];
    $direccion=$_POST['direccion'];

    //Se pasan los parámetros del formulario al método
    //Crear del objeto tipo Servicio para guardar un nuevo documento en la
    bd
    $serv->crear($nombre,$apellido,$telefono,$direccion);
    header( 'Location: index.php' ) ; //Redirecciona a index.php
}

```

```

//Acción para borrar un Alumno
//Si $_POST tiene un elemento 'accion' con el valor 'borrar'
if($_POST['accion']==='borrar'){

//Se pasan el parámetro del formulario al método
//Borrar del objeto tipo Servicio para borrar un documento de la bd
    $serv->borrar($_POST['iddoc']);
    header( 'Location: index.php' );
}

```

```

//Acción para editar un Alumno
//Si $_POST tiene un elemento 'accion' con el valor 'editar'
if($_POST['accion']==='editar'){
    $nombre=$_POST['nombre1'];
    $apellido=$_POST['apellido1'];
    $telefono=$_POST['telefono1'];
    $direccion=$_POST['direccion1'];
    $iddoc=$_POST['id1'];
    //Se pasan los parámetros del formulario al método
    //Editar del objeto tipo Servicio para actualizar un documento de
la bd
    $serv->editar($nombre,$apellido,$telefono,$direccion,$iddoc);
    header( 'Location: index.php' ); //Redirecciona a index.php
}

```

Para realizar el formulario se crea un index.php de la siguiente manera.

 index.php

```

<?php
//Se importa la clase Servicio
require('Servicio.php');
//Se crea una instancia de la clase Servicio
$ser = new Servicio ();
//Se obtiene el arreglo de los documentos de la base
//haciendo uso del método getAll del objeto tipo Servicio
$docs = $ser->getAll();
?>
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-
scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

```

```

<meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="lib/bootstrap/css/bootstrap.min.css">
  <link rel="stylesheet" type="text/css"
href="lib/DataTables/datatables.min.css"/>
<title>Document</title>
</head>

<body style="background-color: whitesmoke">
<div class="container" style="margin-top:2%;">

  <!--Formulario para crear un nuevo alumno-->
  <div class="modal fade" id="modalc">
    <div class="modal-dialog">
      <div class="modal-content">
        <div class="modal-header label-info">
          <button type="button" class="close" data-
dismiss="modal" aria-label="Close">
            <span aria-hidden="true">&times;</span></button>
          <span><h3> Nuevo Alumno</h3></span>
        </div>
        <div class="modal-body">
          <div class="panel panel-primary">
            <div class="panel-body">
              <form action="controlador.php" method="post" id="form_create">
                <div class="form-group">
                  <label for="nombre">Nombre:</label>
                  <input type="text" class="form-control" id="nombre" name="nombre">
                </div>
                <div class="form-group">
                  <label
for="apellido">Apellido:</label>
                  <input type="text" class="form-
control" id="apellido" name="apellido">
                </div>
                <div class="form-group">
                  <label
for="telefono">Teléfono:</label>
                  <input type="text" class="form-
control" id="telefono" name="telefono">
                </div>
                <div class="form-group">
                  <label

```

```

for="direccion">Dirección:</label>
  <input type="text" class="form-control" id="direccion"
name="direccion">
  </div>

      <input type="text" name="accion" value="crear" hidden>

          </form>
      </div>
  </div>
</div>
</div>

<div class="modal-footer label-info">
    <button type="button" class="btn btn-default pull-
left"
        onclick="
document.getElementById('form_create').submit();">
        Guardar
    </button>
    <button type="button" class="btn btn-default pull-
right" data-dismiss="modal">
        Cerrar
    </button>
</div>
</div>
</div>
</div>
<div class="panel-group" style="margin: 0 16% 0">
    <div class="panel panel-default">
        <div class="panel-body">
            <div>
                <div class="form-group">
                    <button type="button" class="btn btn-primary"
data-toggle="modal" data-target="#modalc">
                        Nuevo Alumno
                    </button>
                </div>
                <!--Tabla para mostrar los documentos de la bd-->
                <table id="myTable" class="display" >
                    <thead>
                        <tr>
                            <th>#</th>
                            <th>Nombre</th>
                            <th>Apellido</th>
                            <th>Teléfono</th>
                            <th>Dirección</th>
                            <th>Acciones</th>
                        </tr>

```

```

<!--Se recorre el array $docs para mostrar sus elementos en la tablas-->
<?php
    //inicio bucle
    $index=1;
    foreach ($docs as $key) {
?>
<tr>
    <td><?php echo ($index)?></td>
    <td><?php echo($key->nombre) ?></td>
    <td><?php echo($key->apellido) ?></td>
    <td><?php echo($key->telefono) ?></td>
    <td><?php echo($key->direccion) ?></td>
    <td>
        <div style="text-align: center">
            <button data-toggle="modal" data-target="<?php echo('#edit_' . $key-
            >_id) ?>"
                onclick="event.preventDefault()" <span class="glyphicon glyphicon-
                pencil"></span>
            </button>

```

```

<!--Formulario oculto para eliminar un alumno-->
<form action="controlador.php" method="post" id="<?php echo("form_$key-
>_id") ?>"
    style="display: none">
    <input type="text" name="iddoc" value="<?php echo($key->_id) ?>">
    <input type="text" name="accion" value="borrar">
    </form>
    </div>
    </td>
</tr>

```

```

<!--Formulario para editar un alumno-->
<div class="modal fade" id="<?php echo("edit_<$key->_id") ?>">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header label-info">
<button type="button" class="close" data-dismiss="modal" aria-
label="Close">
  <span aria-hidden="true">&times;</span>
</button>
<span><h3> Editar Datos Alumno</h3></span>
</div>
      <div class="modal-body">
        <div class="panel panel-primary">
          <div class="panel-body">
<form action="controlador.php" method="post"
id="<?php echo("form_edit_<$key->_id") ?>">
          <div class="form-group">
            <label for="nombre">Nombre:</label>
            <input type="text" class="form-control" name="nombre1"

value="<?php echo(<$key->nombre) ?>">
          </div>
          <div class="form-group">
            <label for="apellido">Apellido:</label>
            <input type="text" class="form-control" name="apellido1"

value="<?php echo(<$key->apellido) ?>">
          </div>
          <div class="form-group">
            <label for="telefono">Teléfono:</label>
            <input type="text" class="form-control" name="telefono1"

value="<?php echo(<$key->telefono) ?>">
          </div>
        </form>
      </div>
    </div>
  </div>
</div>

```

```

<div class="form-group">
  <label for="direccion">Dirección:</label>
  <input type="text" class="form-control" name="direccion1"
    value="<?php echo($key->direccion) ?>">
</div>
<input type="text" name="accion" value="editar" hidden>
<input type="text" name="id1" value="<?php echo($key->_id)
?>"hidden>
</form>
</div>
</div>
</div>

```

```

<div class="modal-footer label-info">
  <button type="button" class="btn btn-default pull-left"
    onclick=" document.getElementById('<?php echo("form_edit_<key->_id")
?>').submit();">Guardar</button>

  <button type="button" class="btn btn-default pull-right" data-
    dismiss="modal">Cerrar</button>
</div>
</div>
</div>
</div>
</div>

<?php
  $index++;
}
//fin bucle
unset($key)
?>
</tbody>
</table>
</div>
</div>

```

```

        </div>
    </div>
</div>

<script src="lib/jquery/jquery-3.1.1.min.js"></script>
<script src="lib/bootstrap/js/bootstrap.min.js"></script>
<script type="text/javascript"
src="lib/DataTables/datatables.min.js"></script>

<script>
    $(document).ready(function () {
        $('#myTable').DataTable({
            "language": {
                "url": "lib/sp.json"
            }
        });
    });
</script>
</body>
</html>

```

Para acceder al proyecto poner la siguiente dirección: <http://localhost/conexion/index.php> (tomar en cuenta el puerto de su servidor).



Fig. 22: Ingreso al proyecto <http://localhost/conexion/index.php>.
Fuente: Propia

Ingreso a la base de datos CouchDB con la siguiente dirección <http://127.0.0.1:5984/ utils/index.html> y creación de una nueva base de datos con el nombre de **cont1**.

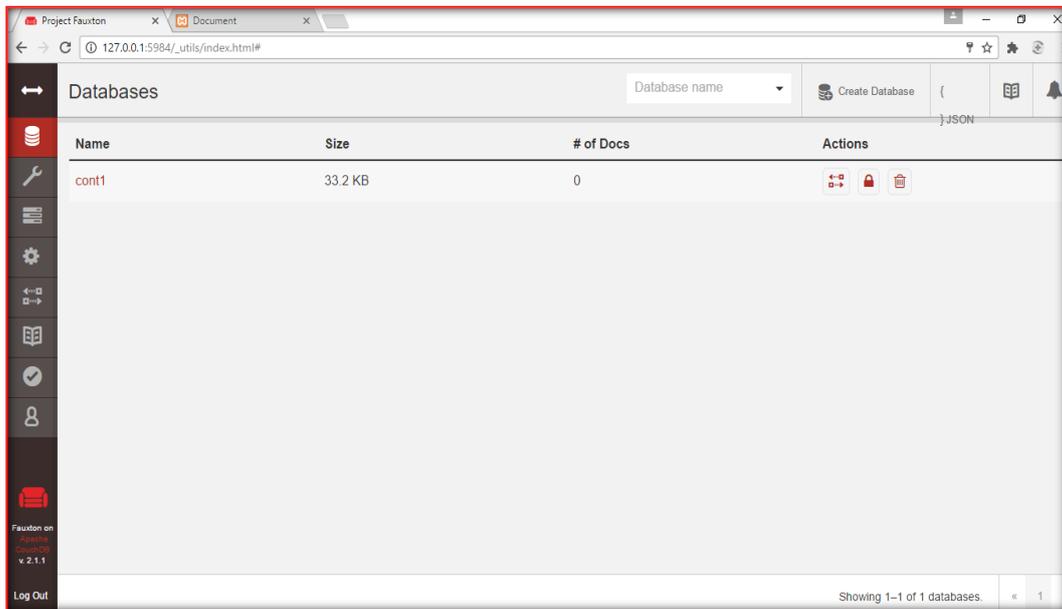


Fig. 23: Ingreso a la base de datos NoSQL CouchDB
Fuente: Propia

Ingreso de un nuevo alumno

Nuevo Alumno

Nombre:
Estefania

Apellido:
Zuleta

Teléfono:
0988803995

Dirección:
Pimampiro, Calles: Bolívar y Garcia Moreno

Guardar Cerrar

Fig. 24: Ingreso nuevo alumno
Fuente: Propia

Como ya se tiene la base de datos integrada a PHP, los datos ingresados se van añadiendo a la base.

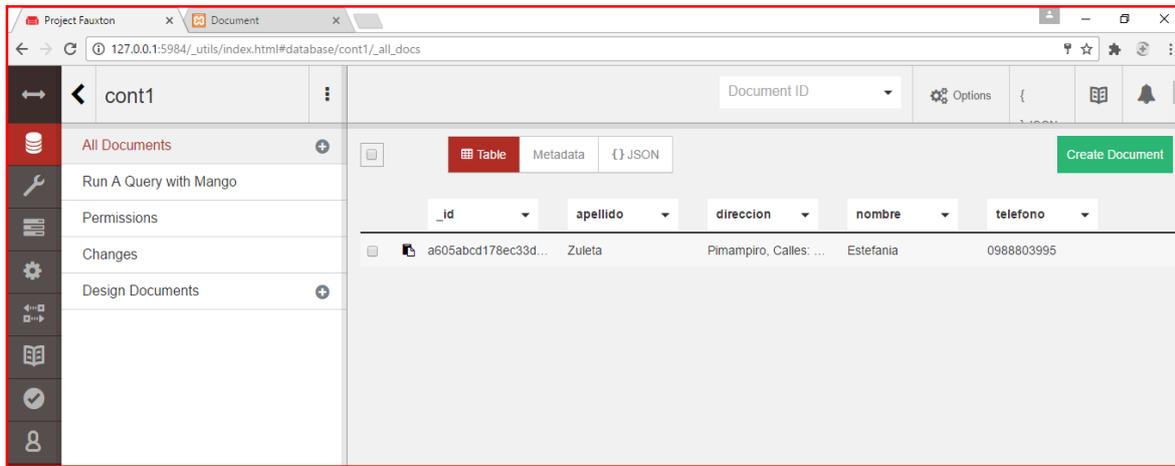


Fig. 25: Creación de la base de datos **cont1** en CouchDB
Fuente: Propia

Se tiene la opción de Editar datos de un alumno, como se lo puede ver en la siguiente figura.

Editar Datos Alumno

Nombre:

Apellido:

Teléfono:

Dirección:

Fig. 26: Editar Datos Alumno
Fuente: Propia

#	Nombre	Apellido	Teléfono	Dirección	Acciones
1	Nataly	Zuleta	0988803995	Pimampiro, Calles: Bolívar y Garcia Moreno	 
2	Wilma	Sevilla	0989413959	Ibarra, Calles: Bolívar y Sucre	 

Mostrando registros del 1 al 2 de un total de 2 registros

Anterior 1 Siguiente

Fig. 27: Edición del nombre
Fuente: Propia

Ingreso de un nuevo alumno y su eliminación.

Nuevo Alumno ×

Nombre:

Apellido:

Teléfono:

Dirección:

Guardar
Cerrar

Fig. 28: Ingreso de un tercer alumno para su eliminación
Fuente: Propia



Fig. 29: Eliminación del alumno ingresado.
Fuente: Propia

Eliminación.



Fig. 30: Eliminación
Fuente: Propia

Aceptar y se elimina el alumno ingresado.



Fig. 31: Alumno eliminado
Fuente: Propia

En la base de datos igual se actualiza en el momento que se edita, ingresa y elimina alumnos. Además los datos se muestran mediante

Tabla

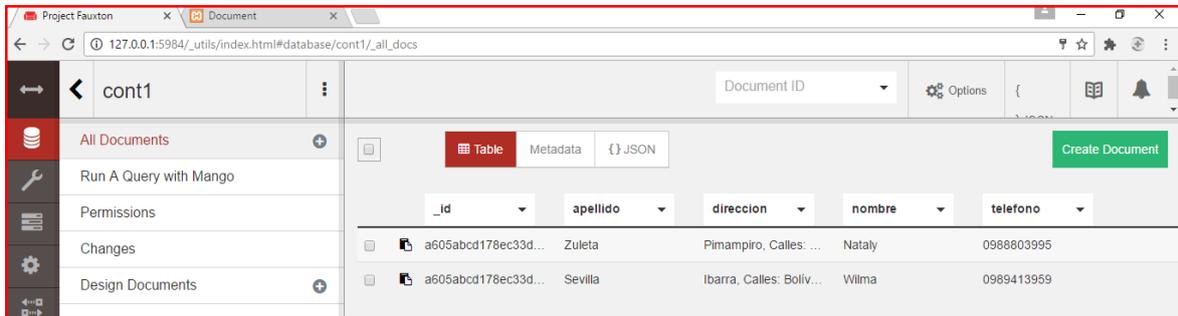


Fig. 32: Base de datos cont1 mediante tabla
Fuente: Propia

Metadata



Fig. 33: Base de datos cont1 mediante Metadata
Fuente: Propia

JSON

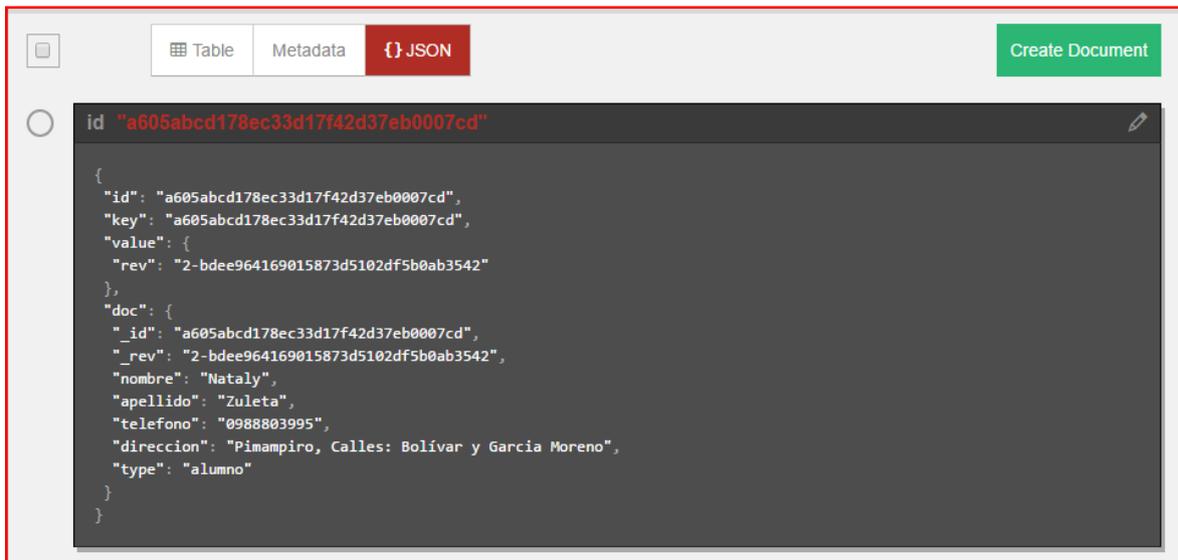


Fig. 34: Base de datos cont1 mediante JSON
Fuente: Propia

Opción buscar

The screenshot shows a web application interface for managing students. At the top left, there is a blue button labeled 'Nuevo Alumno'. Below it, there is a 'Mostrar' dropdown menu set to '10 registros'. To the right, there is a search input field labeled 'Buscar:' containing the text 'Wilma'. Below the search field is a table with the following columns: '#', 'Nombre', 'Apellido', 'Teléfono', 'Dirección', and 'Acciones'. The table contains one row with the following data: '# 2', 'Nombre Wilma', 'Apellido Sevilla', 'Teléfono 0989413959', 'Dirección Ibarra, Calles: Bolívar y Sucre', and 'Acciones' (with edit and delete icons). Below the table, there is a status message: 'Mostrando registros del 1 al 1 de un total de 1 registros (filtrado de un total de 2 registros)'. At the bottom right, there are pagination controls: 'Anterior', a button with the number '1', and 'Siguiete'.

Fig. 35: Opción Buscar
Fuente: Propia

1.15 Metodología Scrum

Scrum es un proceso ágil que se puede usar para gestionar y controlar desarrollos complejos de software y productos usando prácticas iterativas e incrementales. Es un proceso incremental iterativo para desarrollar cualquier producto o gestionar cualquier trabajo. En Scrum un proyecto se ejecuta en bloques temporales (iteraciones sprints) de un mes (pueden ser de dos o tres semanas, si así se necesita). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto que sea susceptible de ser entregado con el mínimo esfuerzo cuando el cliente lo solicite. El Sprint es el ritmo de los ciclos de Scrum. Está delimitado por la reunión de planificación del sprint y la reunión retrospectiva. Una vez que se fija la duración del sprint es inamovible. La mayoría de los equipos eligen dos, tres o cuatro semanas de duración. Diariamente durante el sprint, el equipo realiza una reunión de seguimiento muy breve. Al final del sprint se entrega el producto al cliente en el que se incluye un incremento de la funcionalidad que tenía al inicio del sprint. El proceso parte de la lista de requisitos priorizada del producto, que actúa como plan del proyecto. En esta lista el cliente ha priorizado los requisitos balanceando el valor que le aportan respecto a su coste y han sido divididos en iteraciones y entregas. (Ulloa, 2014)

1.15.1 Beneficios

- 1.- Potenciación responsable de organizar el trabajo por parte del equipo, que es quien mejor conoce cómo realizarlo.
- 2.- Define las tareas necesarias para poder completar cada requisito, creando la lista de tareas de la iteración.
- 3.- Realiza una estimación conjunta del esfuerzo necesario para realizar cada tarea.
- 4.- Es el equipo quien asume la responsabilidad de completar en la iteración los requisitos que selecciona.
- 5.- Es cada una de las personas la que se responsabiliza de realizare las tareas a las que se asigna.
- 6.- Soporte a la innovación.
- 7.- Gestión del ROI
- 8.- Una estimación conjunta es más fiable, dado que tiene en cuenta los diferentes conocimientos, experiencias y habilidades de los integrantes del equipo.

Tabla 7: Beneficios de Scrum
Fuente: (Queralt, 2014)

1.15.2 Componentes

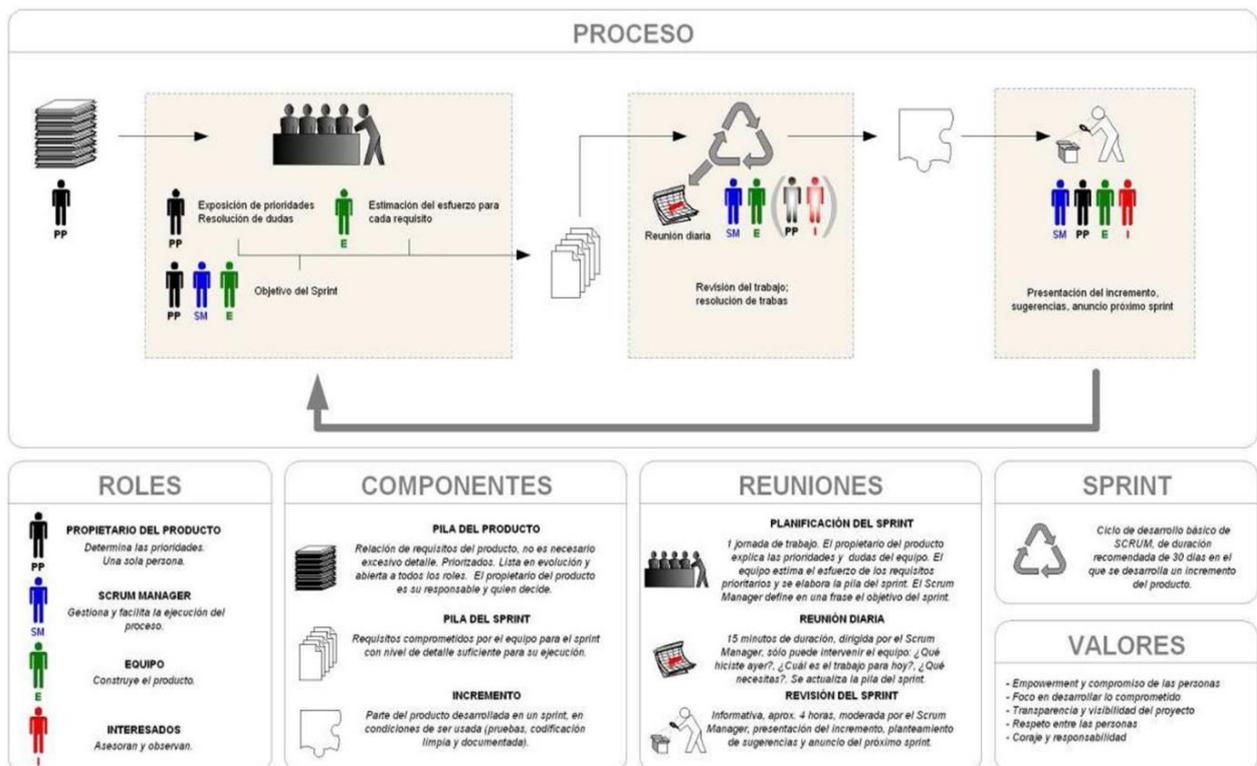


Fig. 36: Componentes Scrum
Fuente: (Ulloa, 2014, pág. 14)

1.15.2 Roles y responsabilidades

Responsables del producto: “Product Owner” (propietario del producto)

- Persona conocedora del entorno de negocio del cliente y de la visión del producto.
- Representa a todos los interesados en el producto final, financia el proyecto, provee los requisitos del sistema, prioriza las funcionalidades, adapta los incrementos de cada Sprint, define los objetivos del proyecto.
- Es el responsable del “Product Backlog” (pila de producto), y proponiendo los requisitos más prioritarios a desarrollar.
- Debe estar disponible durante la iteración para responder las posibles preguntas que pudieran aparecer. (Ulloa, 2014, págs. 14,15)

Responsable del desarrollo: “Scrum Team” (Equipo)

- Equipo multidisciplinario que cubre todas las habilidades necesarias para generar el resultado requerido por el “Product Owner”.
- Se auto-gestiona y auto-organiza de máximo de 5 a 10 personas.
- Dispone de atribuciones suficientes para toma de decisiones sobre cómo realizar su trabajo y seleccionan los requisitos que pueden completar en cada iteración realizando al cliente las preguntas necesarias. (Ulloa, 2014, pág. 15)

Responsables del funcionamiento de Scrum: “Scrum Master” (Director)

- Garantiza el funcionamiento y cumplimiento de los procesos y metodologías que se emplean es el director del proyecto. Es decir se encarga de servir como guía para los integrantes del equipo a través de las reuniones.
- Dirección de la empresa, con el conocimiento de gestión y desarrollo ágil y facilitando los recursos necesarios: “Responsables del Departamento y el área de gestión de proyectos”. (Ulloa, 2014, pág. 15)

Usuarios

- Son los usuarios finales de la aplicación. A partir del progreso de la aplicación, pueden aportar nuevas ideas de modo que la aplicación final se adapte a sus necesidades personales. (Ulloa, 2014, pág. 16)

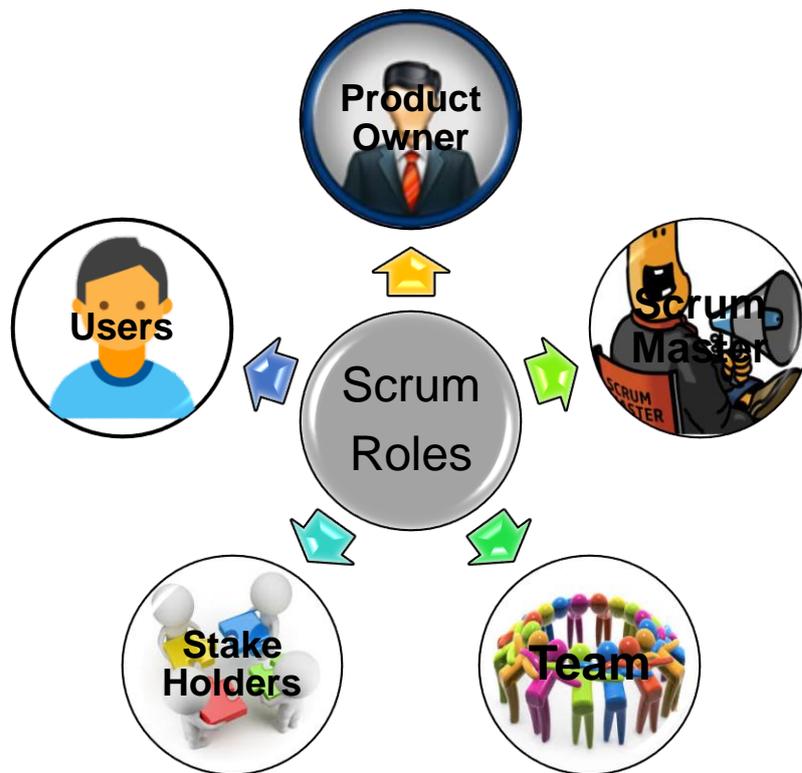


Fig. 37: Roles Scrum
Fuente: Propia

1.15.3 Elementos

“Product Backlog” (Pila del producto)

- Es una lista de todas las tareas, funcionalidades o requerimientos del proyecto. El Product Owner se encarga de priorizar estas tareas y de actualizar la lista con los objetivos conseguidos con la ayuda del Scrum Master. Se parte del resultado que se desea obtener evolucionando durante el desarrollo.
- Todos los integrantes del equipo de desarrollo podrán acceder a él aportando ideas. Pero solo el dueño puede tomar la decisión final. (Ulloa, 2014, pág. 16)

“Sprint Backlog” (Pila del sprint)

- Lista de trabajos que realizará el equipo durante el sprint. Cada iteración tiene que proporcionar un resultado completo, un incremento de producto que sea susceptible de ser entregado con el mínimo esfuerzo cuando el cliente lo solicite.
- Compromiso de ejecución. Asignación de tareas de forma personal con estimación de tiempos y recursos necesarios. (Ulloa, 2014, pág. 16)

“Scrum Daily (Reunión equipo)”

- Demostración de los objetivos alcanzados en cada sprint. Asistencia de todos los roles, “Product Owner” e incluso usuarios.
- Es una tarea iterativa que se realiza todos los días que dure el “Sprint Backlog” con el equipo de desarrollo o de trabajo. Se trata de una reunión operativa, informal y ágil de un tiempo máximo de 30 minutos, donde se da respuesta a 3 preguntas que se hacen a cada integrante.
 - ¿Qué se ha hecho desde la última reunión?
 - ¿Qué se va hacer?
 - ¿Qué ayuda necesita? (Ulloa, 2014, págs. 16,17)

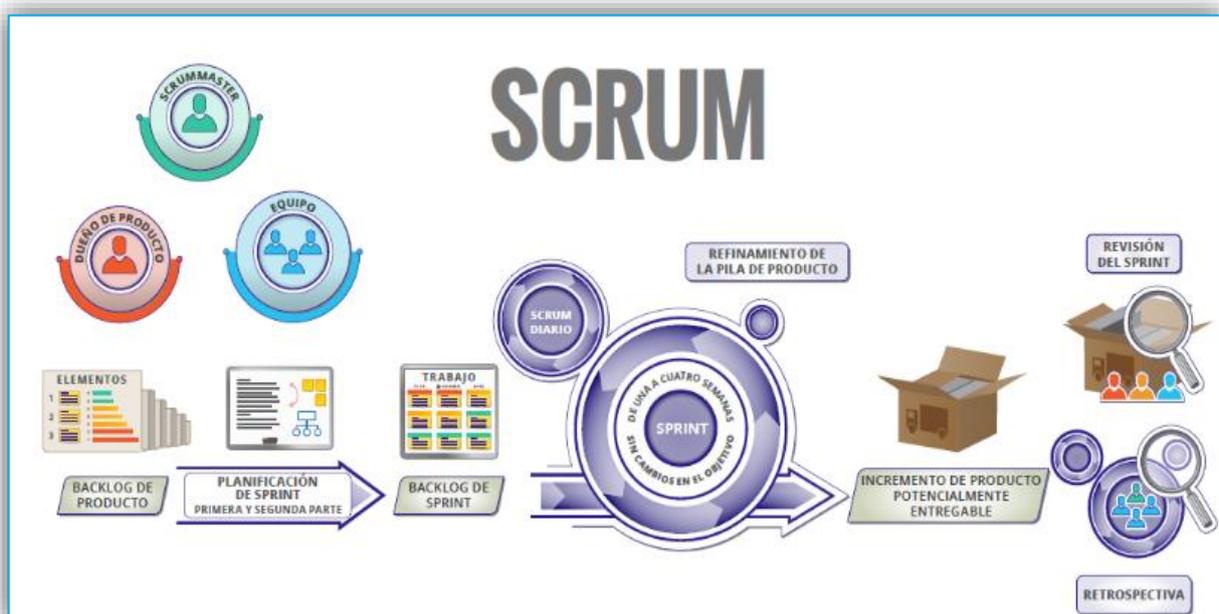


Fig. 38: Elementos principales del Scrum

Fuente: (Editor, ¿Cuál es la metodología Scrum ?, 2016)

1.15.4 Reuniones

Sprint. – un sprint es un periodo de tiempo durante el que se desarrolla un incremento de funcionalidad. Constituye el núcleo de Scrum, que divide de esta forma el desarrollo de un proyecto en un conjunto de pequeñas carreras.

Duración máxima del sprint: 30 días, pero se recomienda 15 días.

Durante el sprint no se puede cambiar el curso de un sprint, abortándolo, y solo puede hacer el Scrum Master si decide que no es viable por alguna de las siguientes razones:

- La tecnología acordada no funciona

- Las circunstancias del negocio han cambiado
- El equipo ha tenido interferencias. (Ulloa, 2014, pág. 18)



Fig. 39: Sprint
Fuente: (ACAPELA, 2016)

Planificación del Sprint.- El equipo planifica la iteración, dado que ha adquirido un compromiso, es el responsable de organizar su trabajo y es quien mejor conoce cómo realizarlo. Intervienen todos los roles y define las tareas necesarias para poder completar cada requisito, creando la listas de tareas de la iteración. Se genera el “Sprint Backlog” o lista de tareas que se van a realizar.

Se determina el objetivo del Sprint (funcionalidad del negocio que se va a generar).

Una estimación conjunta es más fiable, dado que tiene en cuenta los diferentes conocimientos, experiencias y habilidades de los integrantes del equipo. (Ulloa, 2014, pág. 18)

Seguimiento del Sprint. – breve reunión diaria para repasar cada una de las tareas y el trabajo previsto de la jornada. Sólo interviene el equipo de desarrollo.

Cada miembro responde a tres preguntas:

- ¿Trabajo realizado desde la reunión anterior?
- ¿Trabajo que se va a realizar hasta la próxima reunión de seguimiento?
- ¿Problemas que se deben solucionar para realizar el trabajo propuesto? (Ulloa, 2014, pág. 18)

Revisión del Sprint. – el objetivo de la reunión de revisión es presentar el producto o proporcionar del producto desarrollo por el equipo a los usuarios y la misma se utiliza para detectar inconformidades y se los resuelve en el siguiente sprint. Análisis y revisión del incremento generado en si constituye la presentación de resultado. (Ulloa, 2014, pág. 18)

1.15.5 Características

- Se orientan más a las personas que a los procesos.
- Acepta requisitos cambiantes.
- Basado en iteraciones y revisiones.
- Enfocado a conseguir pequeños incrementos de software.
- Su prioridad es la satisfacción al cliente. (Ulloa, 2014, pág. 45)

1.15.6 Ventajas

- Incremento de la productividad.
- Mejoras constantes.
- Existe una gran comunicación en el equipo.
- El cliente siempre está presente en cada mejora del producto.
- Entrega de un producto funcional después de cada sprint.
- Capacidad de aceptar modificaciones.
- Trabaja con iteraciones cortas.
- Equipos integrados y comprometidos. (Ulloa, 2014, pág. 46)

1.15.7 Desventajas

- Dependencia de las personas.
- No es apto para todos los proyectos.
- Delega responsabilidades fijas al equipo de trabajo.
- Problemas con las fechas de entrega bien restringidas mediante un contrato.
(Ulloa, 2014, pág. 46)

CAPÍTULO 2

2 Desarrollo del Sistema Gestión de Expedientes.

2.1 Desarrollo

El presente capítulo es desarrollado a través de la metodología SCRUM, como primer punto se hace un análisis del equipo de trabajo del proyecto especificando los roles del sistema de cada persona, a partir de esto se crea una pila de productos que son los requerimientos de los usuarios, de la pila de productos se deriva una pila de tareas que corresponde a realizar el análisis de requerimientos del sistema, posteriormente a esto se planifica cada iteración, cada iteración consta de fecha de entrega y un resultado como: cambio en el sistema o documentación.

2.2 ROLES DEL SISTEMA

En Scrum, el equipo se concentra en construir un software de calidad, para lo cual el trabajo en equipo es esencial; cada integrante debe comprometerse a una colaboración efectiva por lo que deben tener a su conocimiento sus roles y funciones en el sistema.

El equipo de trabajo para el desarrollo de la aplicación es presentado a continuación:

Rol	Nombre	Cargo
Jefe de Proyecto SCRUM Master	Estefanía Zuleta	Programadora del desarrollo del sistema.
Propietario del Producto Product Owner	Ing. Pedro Granda Coordinador de la escuela CISIC.	Entidad que auspicia el desarrollo del sistema
Externos Interesados Stake Holders	Ing. Mauricio Rea	Asesor de trabajo de grado 2

Tabla 8: Equipo de trabajo para el desarrollo del sistema
Fuente: Propia

2.3 HISTORIAS DE USUARIOS Y CRITERIOS DE ACEPTACIÓN

En la metodología ágil SCRUM la forma de levantar los requerimientos de usuarios es mediante el uso de historias de usuarios, las cuales se enfocan en lo que el usuario necesita hacer, las historias de usuarios pueden tener los siguientes campos

- Identificador (ID) de la Historia: Identificador que se le asigna a la historia de usuario.
- Rol: Muestra el rol en el que el usuario puede realizar la tarea que se va a describir a continuación.
- Funcionalidad: muestra la funcionalidad que se va a poder realizar en la aplicación.
- Número de escenario: los números de escenarios que se puedan presentar realizando la funcionalidad antes mencionada.
- Criterio: Descripción de los diferentes escenarios que se puedan presentar.
- Resultado: Es el comportamiento que el sistema tomaría en cada criterio.

		ENUNCIADO DE LA HISTORIA			CRITERIOS DE ACEPTACIÓN	
ID HISTORIA	SOLICITANTE	ROL	FUNCIONALIDAD	ESCENARIO	CRITERIO	RESULTADO
001	Ing. Pedro Granda	Como secretaria	Ingreso al sistema como administrador.	1	Puede crear usuarios y designar roles.	En la parte superior derecha exista un icono con el perfil, el cual el administrador puede subir su foto y editar su información.
002	Ing. Pedro Granda	Como pasante	Ingreso al sistema como usuario.	2	Encargado de crear alumnos.	De igual manera el usuario tenga un perfil con su foto e información.
003	Ing. Pedro Granda	Como pasante	Registrar alumnos	3	Encargado del ingreso, edición y eliminación de alumnos.	Subir los expedientes de cada alumno escaneados y en formato .pdf y con su respectiva fotografía.
004	Ing. Pedro Granda	Como secretaria y pasante.	Información de alumno.	2 3	Subir documentos escaneados del	<ul style="list-style-type: none"> • Módulos: • Académicos. • Administrativos.

					alumno en cada módulo.	<ul style="list-style-type: none"> • Programa Académico. • Titulación. • Personales.
005	Ing. Pedro Granda	Como secretaria y pasante.	Necesito que cuando ya se tenga la información del alumno.	4	Estadísticas de documentos individuales.	Debe indicar el porcentaje de documentos escaneados subidos del alumno.
006	Ing. Pedro Granda	Como secretaria y pasante.	Necesito que tenga una estadística de documentos por periodos académicos	5	Debe mostrar el porcentaje de documentos por periodos.	Mostrar el 100% si ha completado con toda la documentación, caso contrario indicar los documentos que faltan por subir.
007	Ing. Pedro Granda	Propietario del sistema	El producto final debe funcionar en coordinación de la carrera CISIC.	6	Subir al computador de Coordinación de la carrera CISIC.	Debe el sistema funcionar en la computadora de la Srta. Secretaria de la coordinación de la carrera CISIC para su funcionamiento.

Tabla 9: Historias de usuario y criterios de aceptación
Fuente: Coordinación de la carrera CISIC

2.4 PILA DE PRODUCTOS

En esta pila de detallan los requisitos en los que se estará trabajando en la fase de iteraciones.

ID	NOMBRE	OBSERVACIÓN
R1	Análisis y estructuración del proyecto.	Creación del proyecto.
R2	Registro y creación del administrador.	Creamos la interfaz del registro e ingresamos los datos del administrador del sistema, él puede editar su perfil y será el encargado de dar privilegios.
R3	Seguridad y Control	Implementación para accesos al sistema, a continuación de registrarse como administrador debe ingresar al sistema y este crear nuevos usuarios y designar su rol según sus actividades.
R4	Creación de alumnos con sus módulos e ítems.	<ul style="list-style-type: none"> • Creación, edición, eliminación, actualización y subida de documentos de los alumnos. • Estadísticas – Documentos que se suben una sola vez. • Estadísticas – documentos que se suben cada período.
R5	Implementación del Sistema	Implementación y pruebas del sistema, instalación en la computadora de la Coordinación de la carrera CISIC.

Tabla 10: Pila de productos
Fuente: Propia

2.5 PILA DE TAREAS

En esta pila se realiza el análisis de requerimientos, haciendo el desglosamiento de los mismos en pequeñas tareas para que los desarrolladores puedan cumplir con su implementación en un periodo corto de tiempo.

Análisis y Estructuración del proyecto		
ID	NOMBRE	OBSERVACIÓN
T1	Instalación de la base de datos NoSQL Apache CouchDB 2.1.1	Instalación y configuración del motor de la base de datos con la cual se va a trabajar.
T2	Instalación de Apache	Instalación del servidor apache para poder testear la aplicación.
T3	Instalación de Laravel	Instalación y configuración del framework de PHP Laravel.
T4	Creación de la base de datos	Creación de las diferentes tablas para el sistema.
T5	Creación del Proyecto	Se crea el proyecto con los módulos e ítems ya mencionados.

Tabla 11: Análisis y Estructuración del proyecto
Fuente: Propia.

Registro y creación del administrador		
ID	NOMBRE	OBSERVACIÓN
T6	Creación de la Interfaz de registro para el ingreso al sistema.	Se crea un formulario de registro para ingreso al sistema.
T7	Crear administrador.	Se crea el administrador con sus datos personales este tendrá un perfil en donde puede poner su foto y modificar sus datos.

Tabla 12: Registro y creación del administrador
Fuente: Propia

Seguridad y control		
ID	NOMBRE	OBSERVACIÓN
T8	Creación de la Interfaz para el ingreso al sistema.	Se crea primero un registro y luego el login para el ingreso al sistema.
T9	Añadir seguridad al sistema.	Ninguna
T10	Crea usuario y designa un rol.	Crea usuario y da un rol.

Tabla 13: Seguridad y control
Fuente: Propia

Creación de Usuarios		
ID	NOMBRE	OBSERVACIÓN
T11	Creación de usuarios.	Se crea el usuario, este podrá ingresar, editar y eliminar entrando desde su perfil.
T12	Usuarios.	El administrador puede ver, editar o eliminar. El usuario puede crear alumnos.

Tabla 14: Creación de Usuarios.
Fuente: Propia

Creación de Alumnos		
ID	NOMBRE	OBSERVACIÓN
T13	Creación de alumnos.	Se crea los usuarios por el administrador, este puede ingresar, editar y eliminar al usuario creado.
T14	Creación de módulos con sus diferentes ítems.	Módulos: <ul style="list-style-type: none"> • Académicos. • Administrativos. • Programa Académico. • Titulación. • Personales. Se debe subir la documentación de cada alumno en los módulos, escaneados y en formato pdf.
T15	Creación de estadísticas – documentos que se suben una sola vez	Indica el porcentaje de cada documento que se sube.
T16	Creación de Estadísticas totales.	Indica el porcentaje total de todos los expedientes subidos de los alumnos.

Tabla 15: Creación de alumnos.
Fuente: Propia.

Implementación del sistema		
ID	NOMBRE	OBSERVACIÓN
T17	Ingreso de alumnos.	Se crea los usuarios por el administrador, este puede ingresar, editar y eliminar al usuario creado.
T18	Subida de documentos escaneados a los módulos con sus diferentes ítems.	Módulos: <ul style="list-style-type: none"> • Académicos. • Administrativos. • Programa Académico. • Titulación. • Personales. Se debe subir la documentación de cada alumno en los módulos, escaneados y en formato pdf.
T19	Visualización de las Estadísticas – documentos que se suben una sola vez.	Indica el porcentaje de cada documento que se sube.
T20	Visualización de las Estadísticas – documentos que se suben cada periodo.	Indica el porcentaje total de todos los expedientes subidos de los alumnos.

Tabla 16: Implementación del sistema.
Fuente: Propia.

2.6 PLANIFICACIÓN DEL PROYECTO

En esta sección del proyecto se planifica cada iteración o sprint, con el que se define qué tareas y requisitos se van a cumplir en cada iteración, además se determina las fechas de inicio y fin para la ejecución de las mismas.

ID	NOMBRE	INICIO	FIN	REQUERIMIENTOS
11	Análisis y estructuración del proyecto.	13-10-2017	20-10-2017	R1, T1, T2, T3.
12	Registro y creación del administrador.	22-10-2017	28-10-2017	R2, T4, T5, T6.
13	Seguridad y Control	30-10-2017	23-11-2017	R3, T7, T8, T9.
14	Creación de usuarios	25-11-2017	15-12-2017	R4, T10, T11, T12.
15	Creación de alumnos	16-12-2017	31-12-2017	R5, T13, T14, T15.
16	Implementación del Sistema	03-01-2018	23-02-2018	R6, T16, T17, T18, T19, T20.

Tabla 17: Planificación del Proyecto
Fuente: Propia.

2.7 ITERACIONES

2.7.1 ITERACIÓN 1 ANÁLISIS Y ESTRUCTURACIÓN DEL PROYECTO

Requerimiento Nro. 1. – Análisis y estructuración del proyecto.

Tarea Nro. 1. – Instalación de CouchDB.

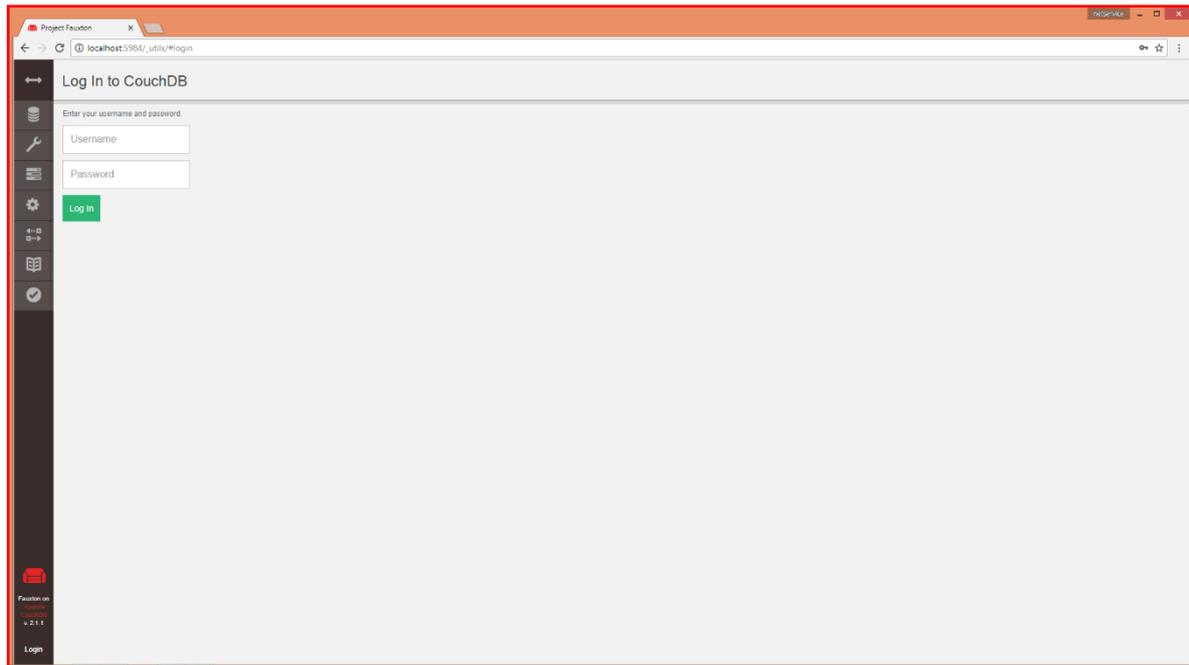


Fig. 40: Instalación de CouchDB
Fuente: Propia

Interfaz visual de administración de CouchDB, en la cual se va a crear la base de datos que estará integrada a nuestro lenguaje de programación PHP en este caso el framework Laravel.

TAREA NRO. 2. – Instalación de Apache.

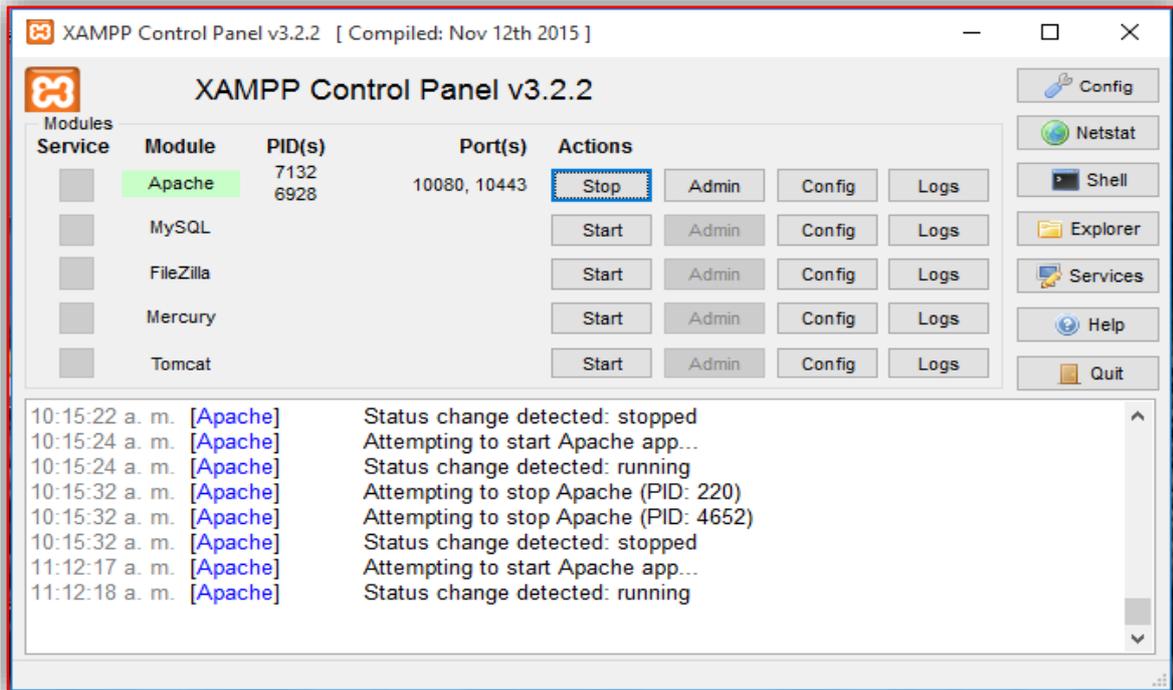


Fig. 41: Instalación de Apache
Fuente: Propia.

El panel de control XAMPP permite ejecutar el sistema también en este panel de control se puede modificar los archivos de configuración del servicio que utilizamos (Apache).

TAREA NRO. 3. – Instalación de Composer y Laravel.

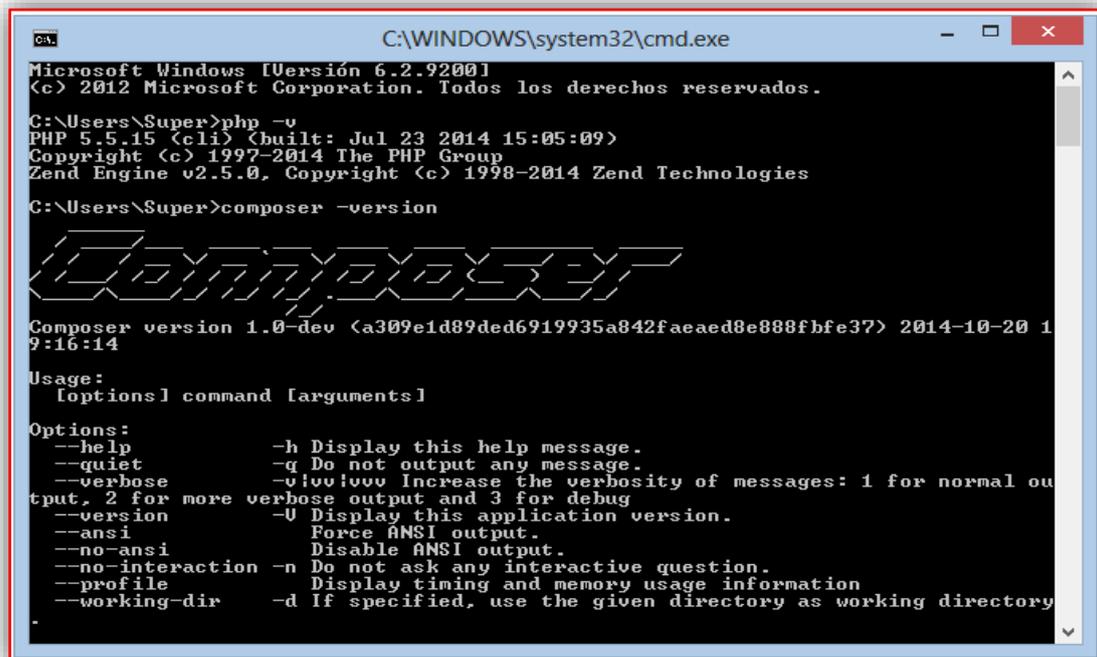
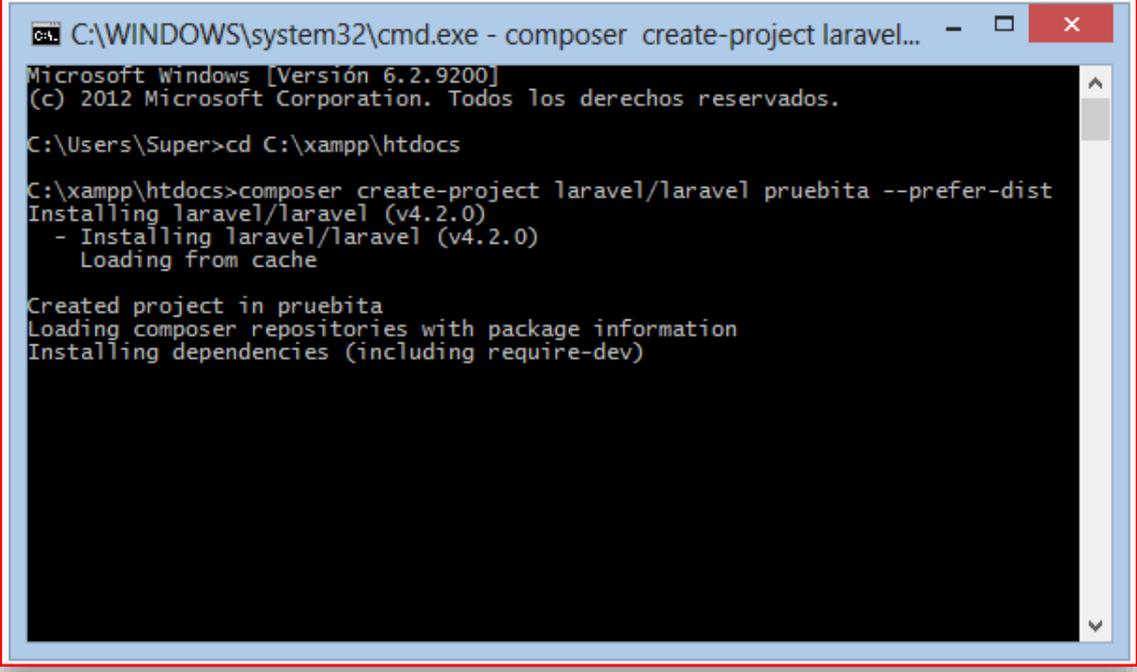


Fig. 42: Instalación de Composer
Fuente: Propia.

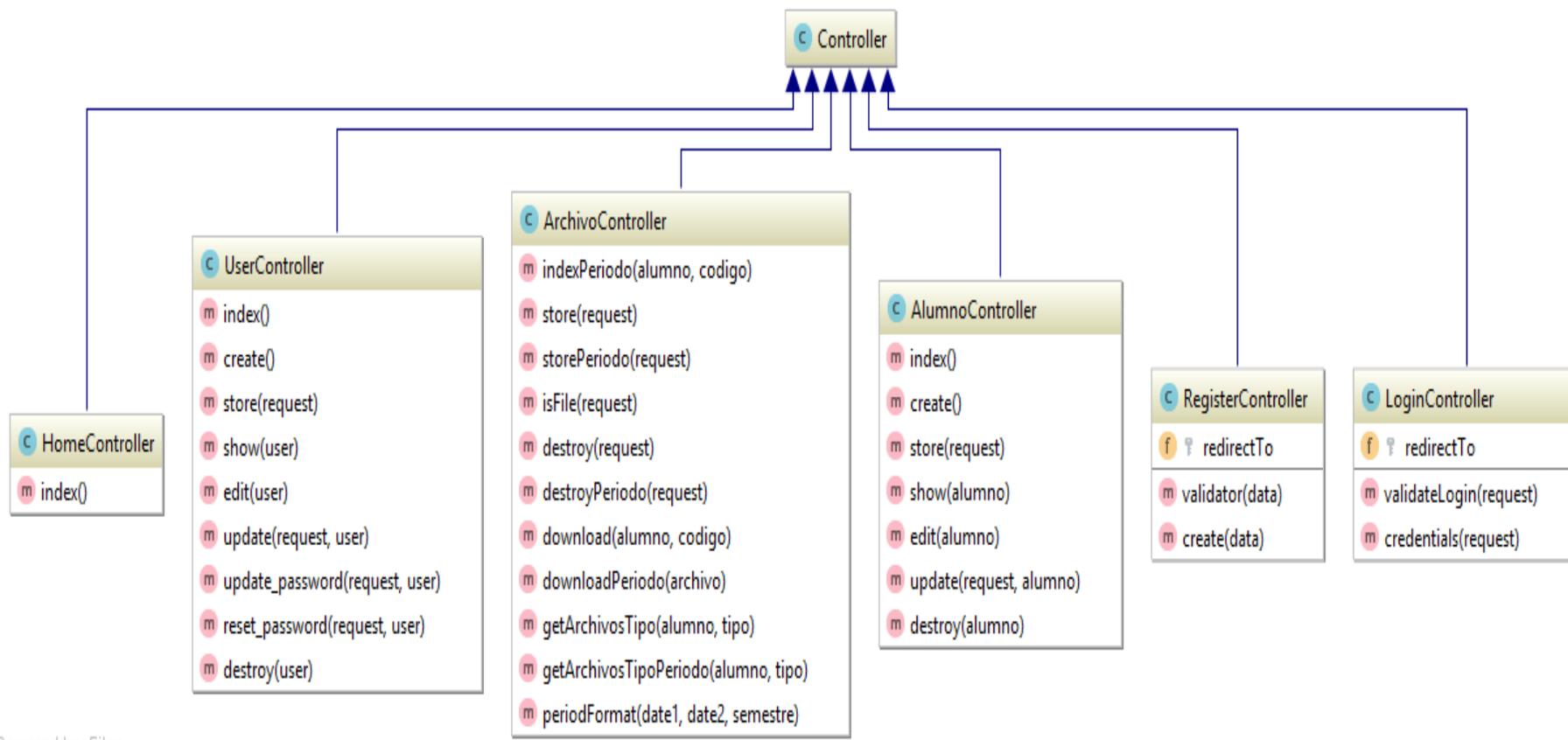


```
C:\WINDOWS\system32\cmd.exe - composer create-project laravel...  
Microsoft Windows [Versión 6.2.9200]  
(c) 2012 Microsoft Corporation. Todos los derechos reservados.  
C:\Users\Super>cd C:\xampp\htdocs  
C:\xampp\htdocs>composer create-project laravel/laravel pruebita --prefer-dist  
Installing laravel/laravel (v4.2.0)  
- Installing laravel/laravel (v4.2.0)  
  Loading from cache  
  
Created project in pruebita  
Loading composer repositories with package information  
Installing dependencies (including require-dev)
```

Fig. 43: Instalación de Laravel
Fuente: Propia.

La instalación de Laravel se realiza mediante el código: *composer create-project laravel/laravel proj --prefer-dist*.

TAREA NRO. 4. – Creación de la base de datos.



Powered by yFiles

Fig. 44: Diagrama de Clases, Controller
Fuente: Propia.

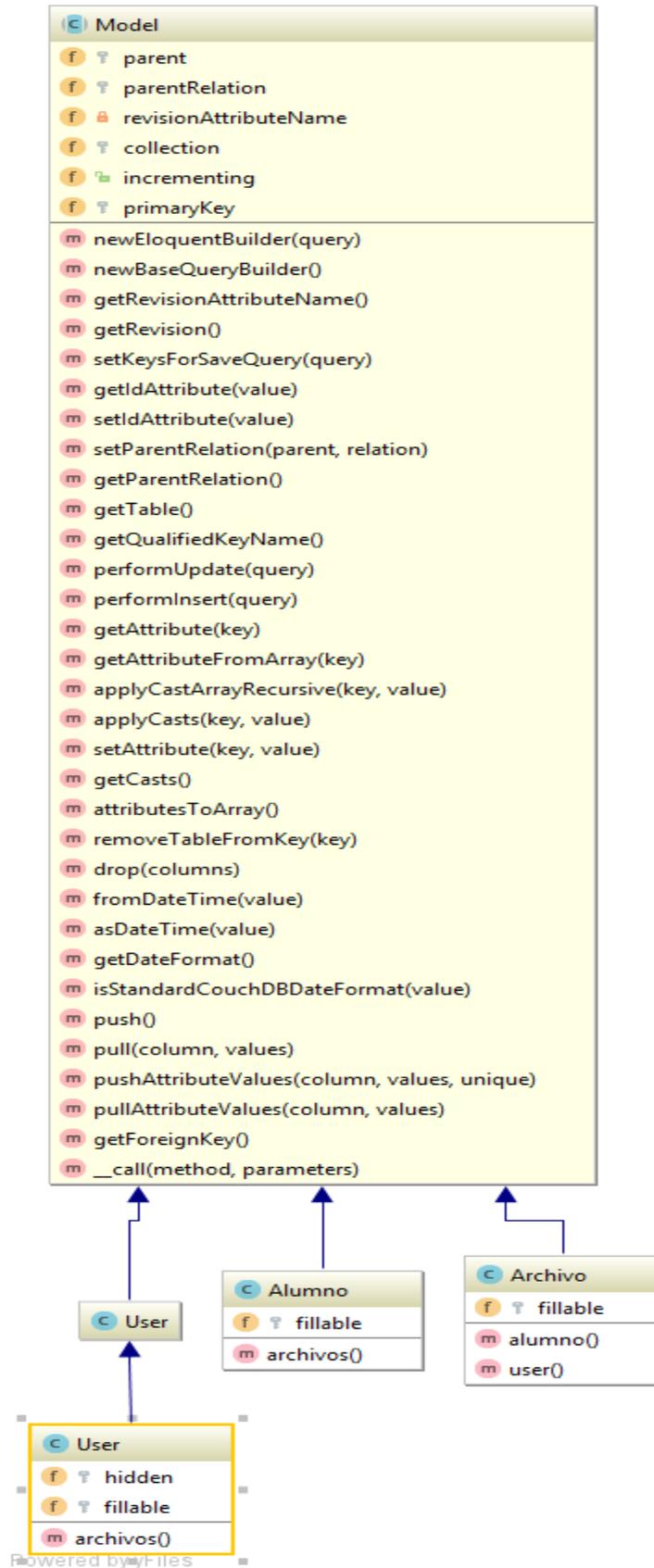


Fig. 45: Diagrama de clases, Model
Fuente: Propia.

TAREA NRO. 5. – Creación del proyecto.

Desde la consola, dirigirse al directorio donde se guardan los proyectos, en este caso como utilizamos XAMPP la ruta es C:\xampp\htdocs.

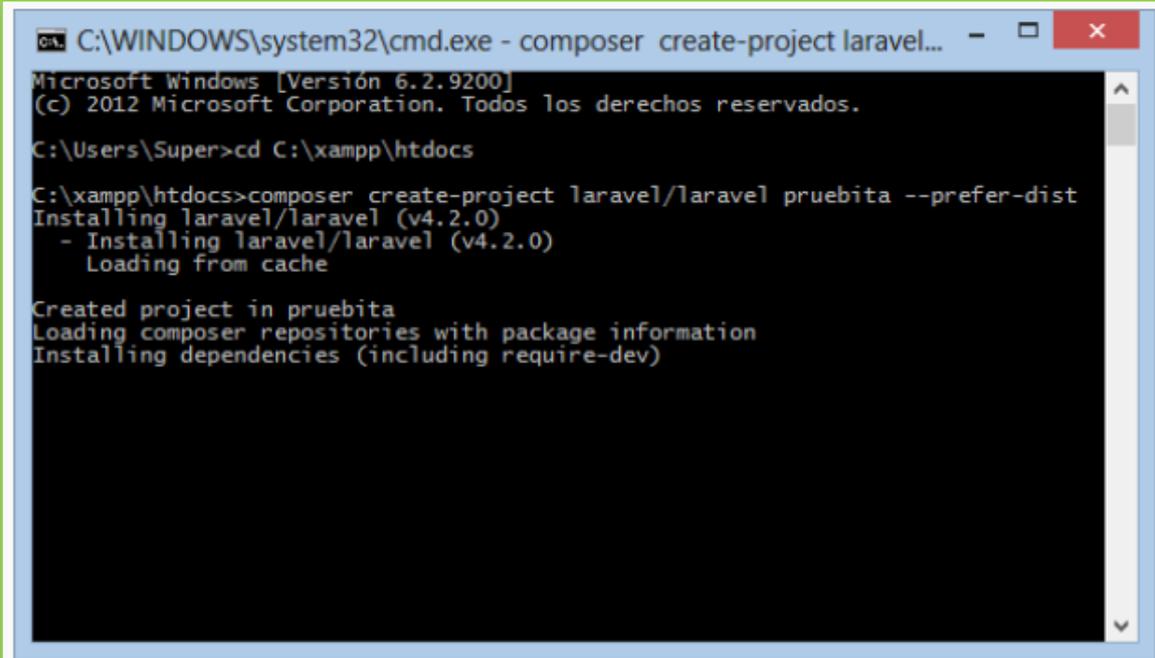
```
cd C:\xampp\htdocs
```

Fig. 46: Comando ingreso al directorio XAMPP
Fuente: (Cruz, 2014)

Ahora creamos el proyecto Laravel con el siguiente comando

```
composer create-project laravel/laravel nombre_del_proyecto  
--prefer-dist
```

Fig. 47: Comando para crear proyecto en Laravel.
Fuente: (Cruz, 2014)



```
C:\WINDOWS\system32\cmd.exe - composer create-project laravel...  
Microsoft Windows [Versión 6.2.9200]  
(c) 2012 Microsoft Corporation. Todos los derechos reservados.  
C:\Users\Super>cd C:\xampp\htdocs  
C:\xampp\htdocs>composer create-project laravel/laravel pruebita --prefer-dist  
Installing laravel/laravel (v4.2.0)  
- Installing laravel/laravel (v4.2.0)  
  Loading from cache  
  
Created project in pruebita  
Loading composer repositories with package information  
Installing dependencies (including require-dev)
```

Fig. 48: Creación del proyecto.
Fuente: (Cruz, 2014)

Composer empezará a descargar las librerías necesarias para el proyecto.

```
C:\WINDOWS\system32\cmd.exe - composer create-project laravel...  
- Installing nikic/php-parser (v0.9.5)  
  Loading from cache  
- Installing jeremeamia/superclosure (1.0.1)  
  Loading from cache  
- Installing filp/whoops (1.1.3)  
  Loading from cache  
- Installing ircmaxell/password-compat (1.0.3)  
  Loading from cache  
- Installing d11wtq/boris (v1.0.8)  
  Loading from cache  
- Installing symfony/filesystem (v2.5.6)  
  Loading from cache  
- Installing classpreloader/classpreloader (1.0.2)  
  Loading from cache  
- Installing laravel/framework (v4.2.11)  
  Loading from cache
```

Fig. 49: Instalación de librerías Laravel.
Fuente: (Cruz, 2014)

Fig. 50: Finalización de instalación librerías de Laravel.
Fuente: (Cruz, 2014)

Cuando ya se haya instalado las librerías, verificamos que el proyecto se crea correctamente acceder al navegador: <http://localhost/proj/public>

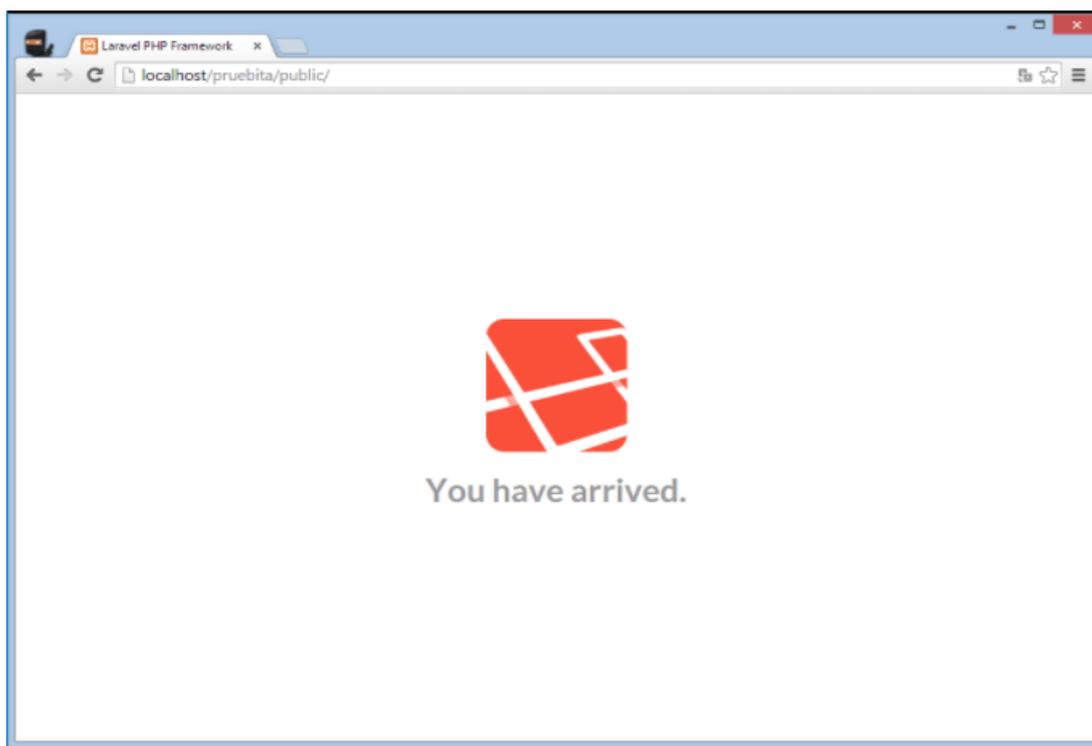


Fig. 51: Vista de inicio del proyecto en Laravel

Fuente: (Cruz, 2014).

SPRINT : 1 FECHA DE INICIO: 13 de Oct. De 2017.			
			13 Oct.
			Tareas Pendientes 0
REQUERIMIENTOS	TAREA	REQUERIMIENTOS	ESTADO
Análisis y estructuración del proyecto.	Instalación de la base de datos NoSQL Apache CouchDB 2.1.1	Estefanía Zuleta	TERMINADO
	Instalación de Apache	Estefanía Zuleta	TERMINADO
	Instalación de Laravel	Estefanía Zuleta	TERMINADO
	Creación de la base de datos	Estefanía Zuleta	TERMINADO
	Creación del Proyecto	Estefanía Zuleta	TERMINADO

Tabla 18: Sprint 1 – Hoja Electrónica
Fuente: Propia

PENDIENTE	EN PROCESO	TERMINADA
_____	_____	Instalación de la base de datos NoSQL Apache CouchDB 2.1.1
_____	_____	Instalación de Apache
_____	_____	Instalación de Laravel
_____	_____	Creación de la base de datos
_____	_____	Creación del proyecto

Tabla 19: Sprint 1 – Pizarrón
Fuente: Propia.

2.7.2 ITERACIÓN 2 REGISTRO Y CREACIÓN DEL ADMINISTRADOR.

Requerimiento Nro. 2. – Registro y creación del administrador.

Tarea Nro. 6. – Creación de la Interfaz de Registro para el ingreso al sistema.

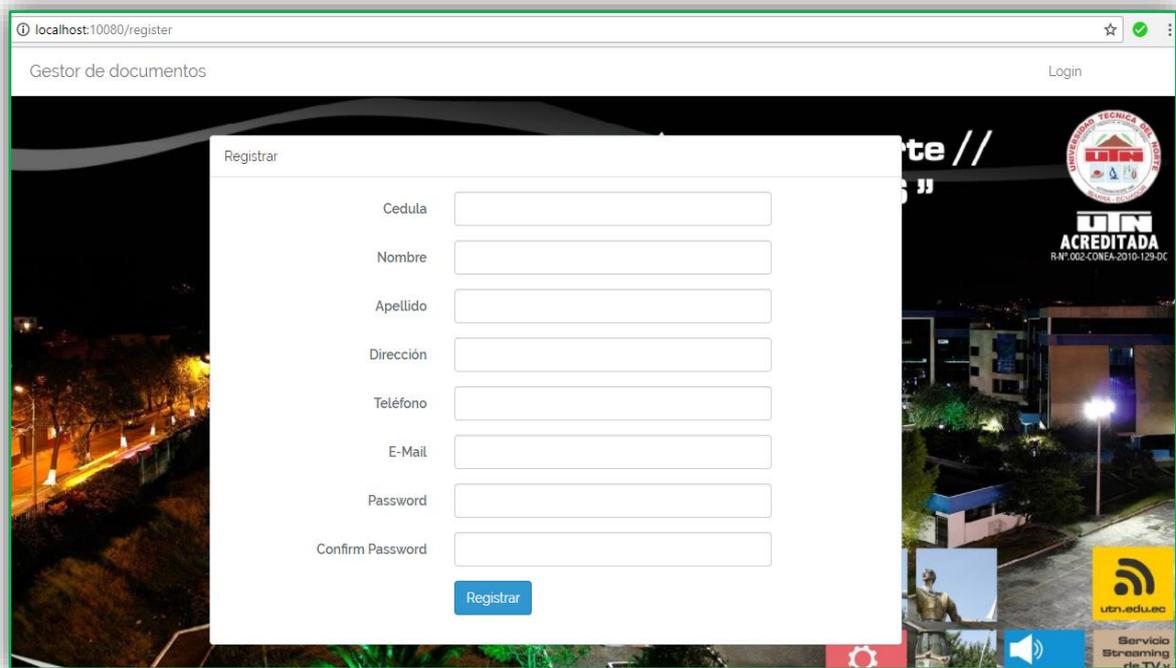


Fig. 52: Interfaz de Registro para el ingreso al sistema
Fuente: Propia.

Interfaz de registro para el ingreso al sistema, en este formulario se ingresaran los datos personales.

Tarea Nro. 7. – Creación del administrador

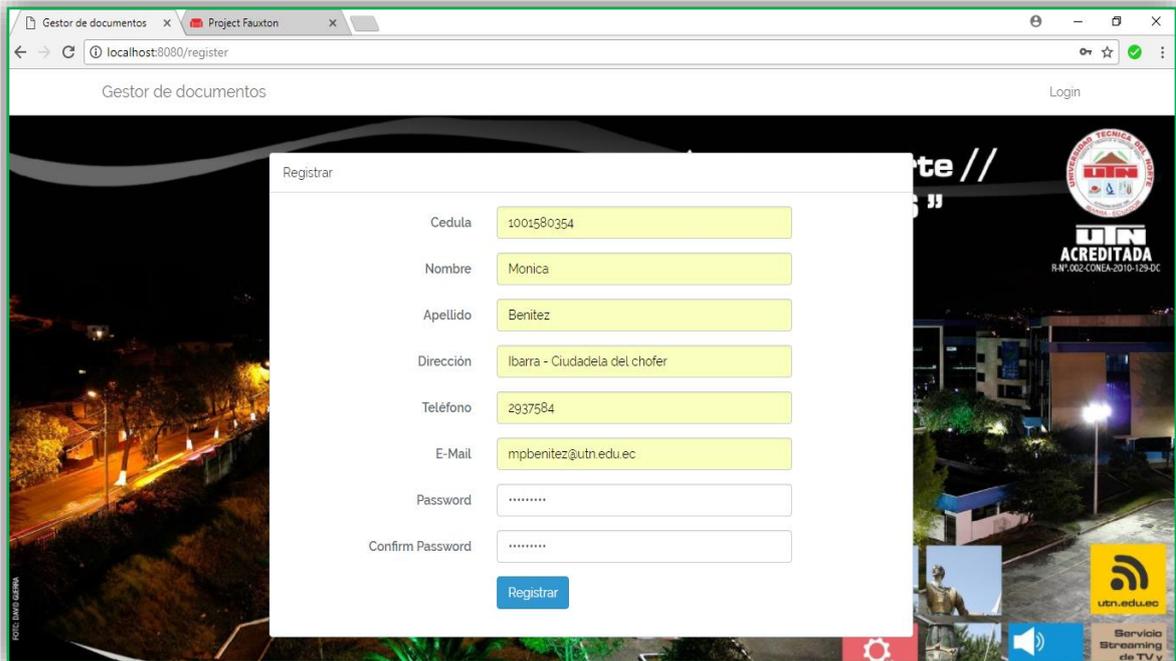


Fig. 53: Ingreso de datos personales.
Fuentes: Zuleta, Estefanía

Ingreso de los datos personales de la persona que va hacer el administrador del sistema.

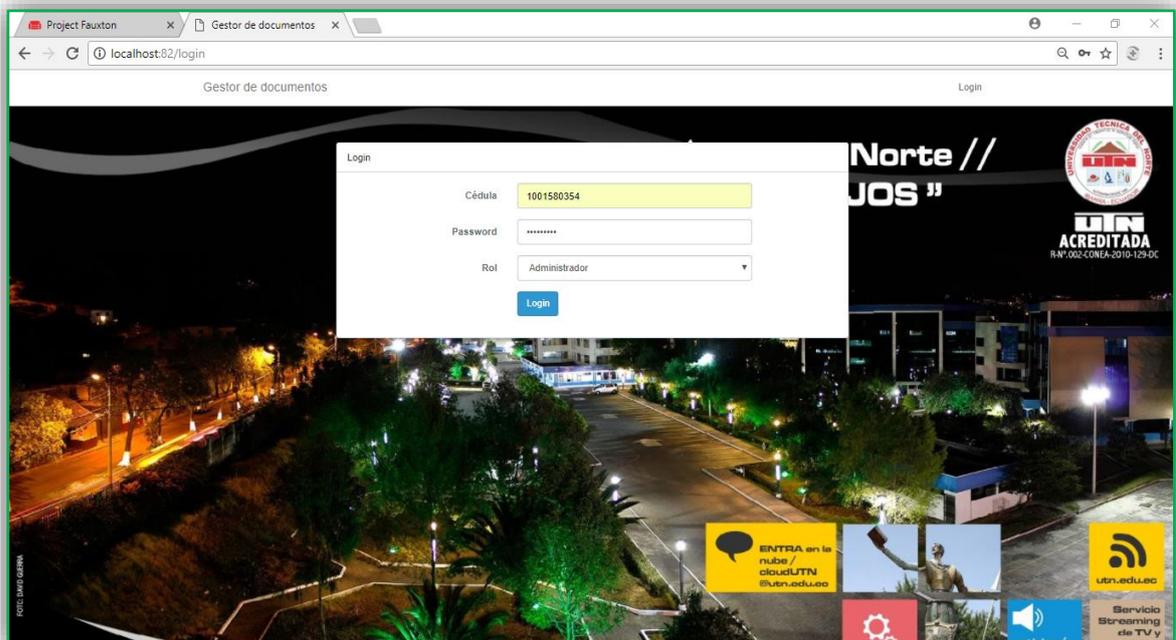


Fig. 54: Ingreso al sistema como administrador.
Fuente: Zuleta, Estefanía

Ingreso al sistema como administrar.

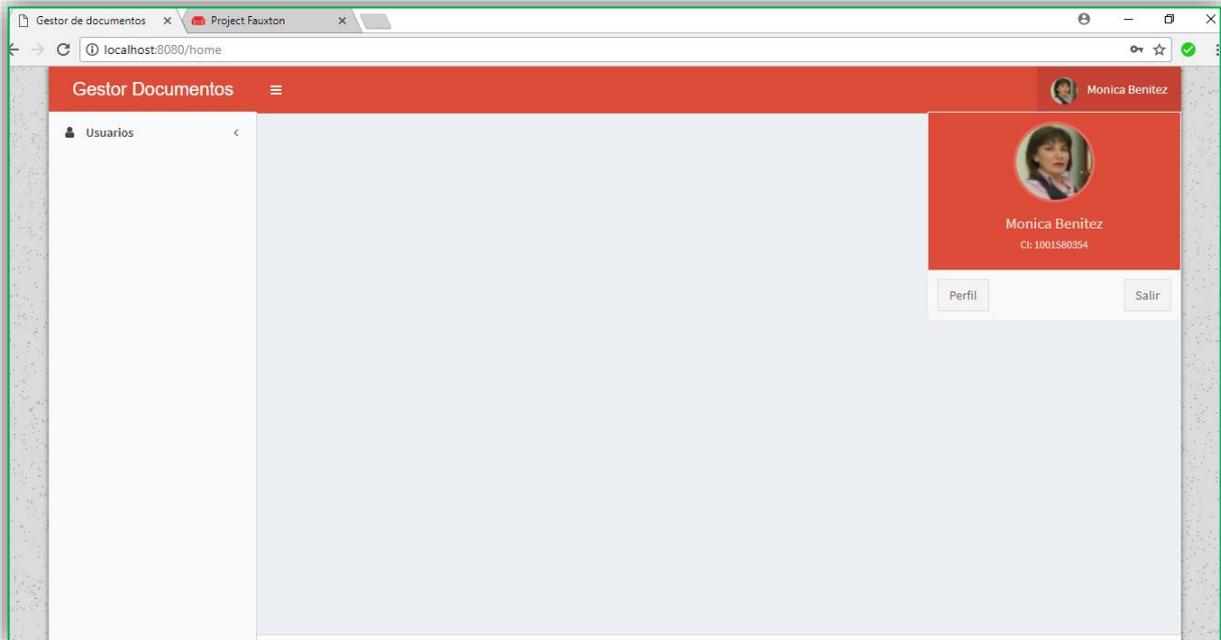


Fig. 55: Perfil del administrador
Fuente: Propia

Perfil del administrador.

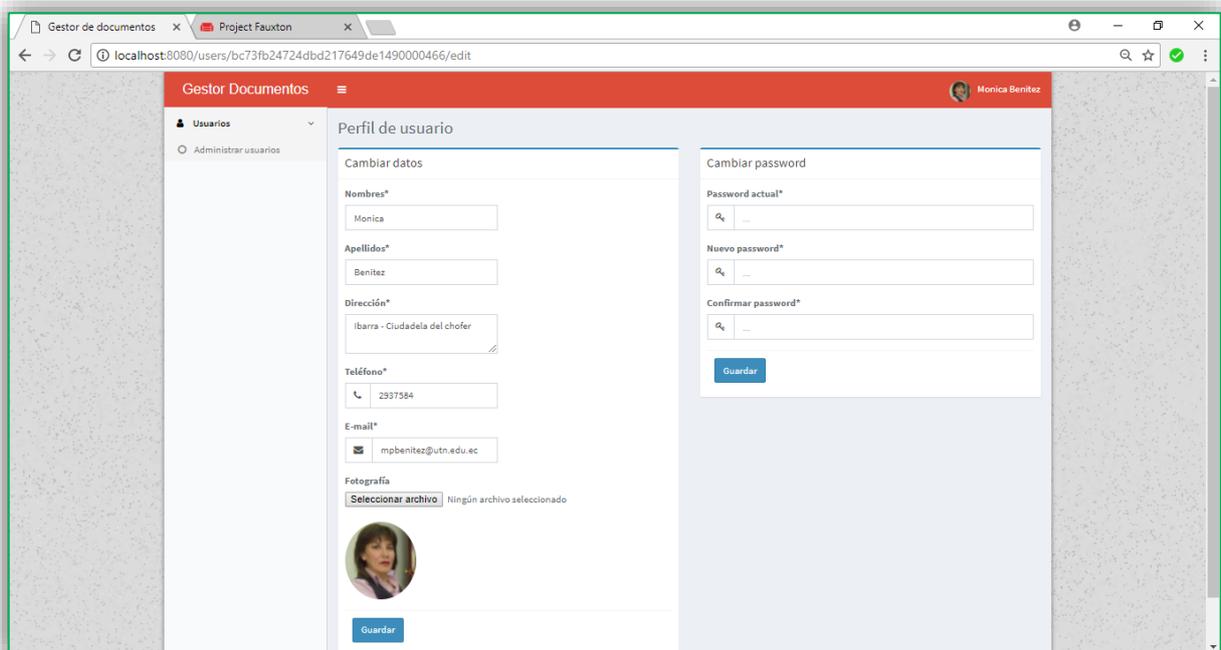


Fig. 56: Ingreso al perfil del administrador.
Fuente: Propia.

El administrador ingresa al perfil, aquí este puede editar su información y cambiar su password.

SPRINT : 2 FECHA DE INICIO: 22 de Oct. De 2017.			
			22 Oct.
		Tareas Pendientes	0
REQUERIMIENTOS	TAREA	REQUERIMIENTOS	ESTADO
Registro y creación del administrador.	Creación de la Interfaz de registro para el ingreso al sistema.	Estefanía Zuleta	TERMINADO
	Crear administrador.	Estefanía Zuleta	TERMINADO

Tabla 20: Sprint 2 – Hoja electrónica.
Fuente: Propia.

PENDIENTE	EN PROCESO	TERMINADA
_____	_____	<div style="border: 1px solid black; border-radius: 15px; background-color: #FFD700; padding: 10px; text-align: center;"> Creación de la interfaz de registro para el ingreso al sistema. </div>
_____	_____	<div style="border: 1px solid black; border-radius: 15px; background-color: #FFD700; padding: 10px; text-align: center;"> Crear administrador </div>

Tabla 21: Sprint 2 – Pizarrón
Fuente: Propia.

2.7.3 ITERACIÓN 3 SEGURIDAD Y CONTROL.

Requerimiento Nro. 3. – Seguridad y control.

Tarea Nro. 8. – Creación de la Interfaz para el ingreso al sistema.

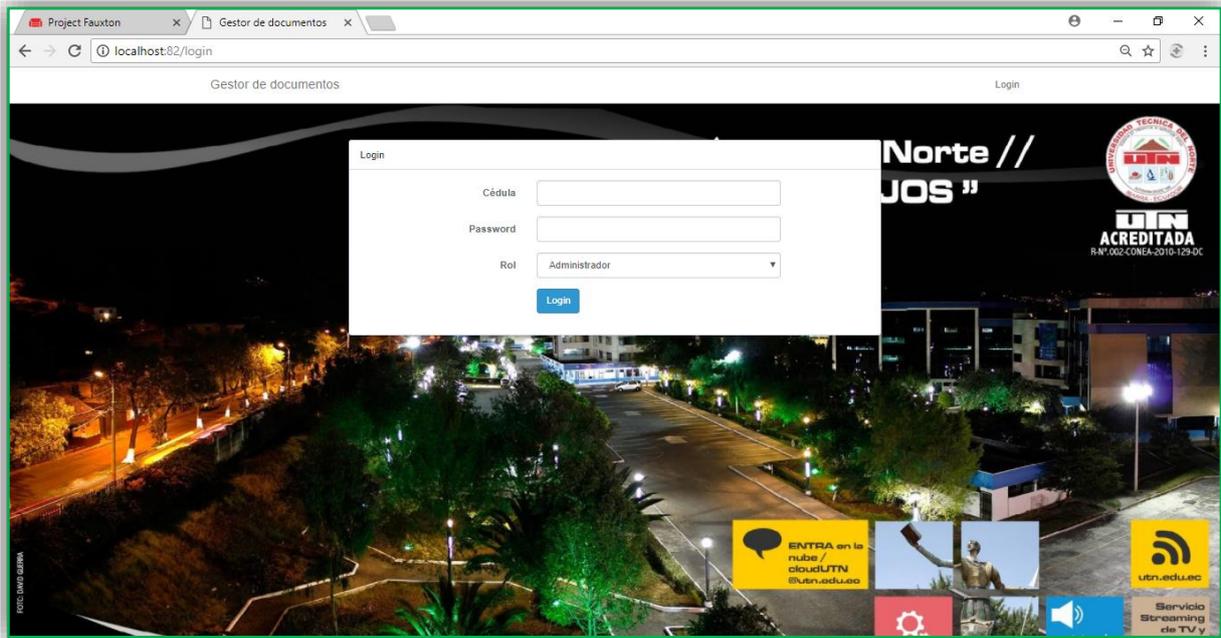


Fig. 57: Login del sistema.
Fuente: Propia.

Creación del Login para el ingreso al sistema con cédula, password y rol el cual contiene el Administrador y Usuario.

Tarea N. 9.- Añadir seguridad al sistema.



Fig. 58: Seguridad del sistema.
Fuente: Propia.

Se usó el módulo de seguridad de Laravel, que nos permite definir los privilegios de cada rol de usuario.

Tarea N. 10. – Crea usuario y designa un rol.

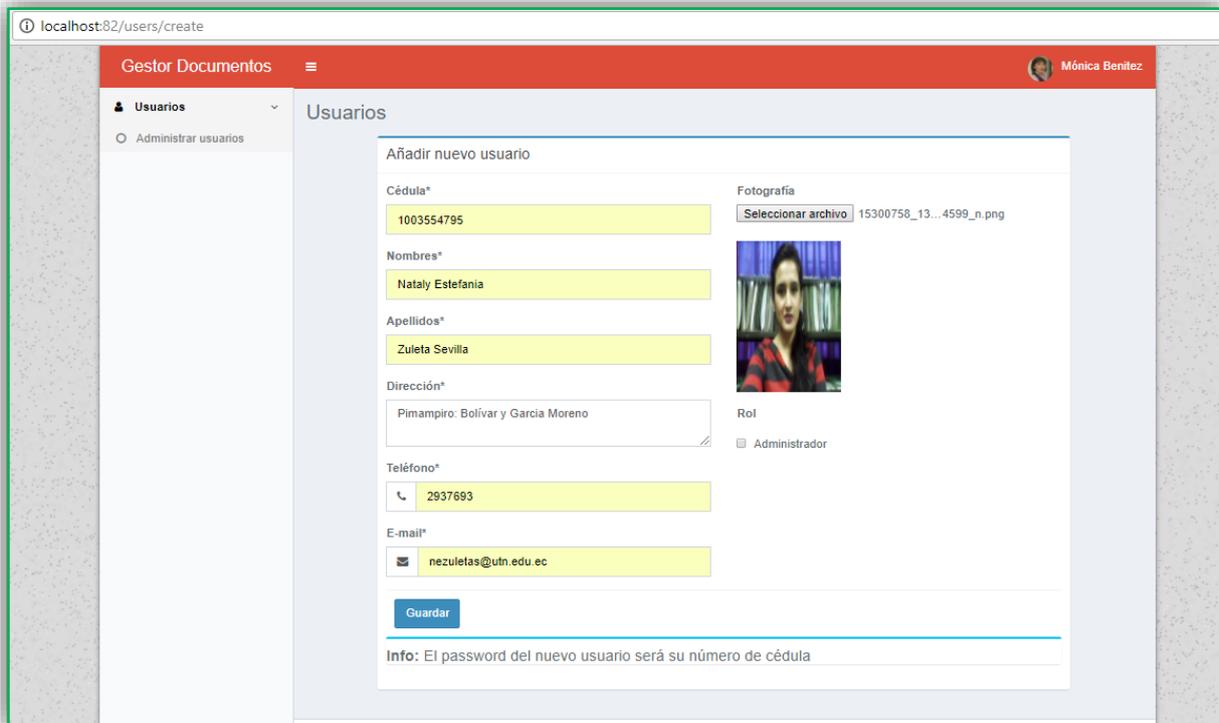


Fig. 59: Creación de usuario.
Fuente: Propia.

El administrador crea usuario con todos los datos personales y puede designar su rol.

SPRINT : 3			
FECHA DE INICIO: 30 de			
Oct. De 2017.			
		Tareas Pendientes	30 Oct. 0
REQUERIMIENTOS	TAREA	REQUERIMIENTOS	ESTADO
Seguridad y control.	Creación de la Interfaz para el ingreso al sistema.	Estefanía Zuleta	TERMINADO
	Añadir seguridad al sistema.	Estefanía Zuleta	TERMINADO
	Crear usuarios y designar un rol.	Estefanía Zuleta	TERMINADO

Tabla 22: Sprint 3 – Hoja electrónica.
Fuente: Propia.

PENDIENTE	EN PROCESO	TERMINADA
_____	_____	Creación de la interfaz para el ingreso al sistema.
_____	_____	Añadir seguridad al sistema
_____	_____	Crear usuarios y designar un rol.

Tabla 23: Sprint 3 – Pizarrón.
Fuente: Propia.

2.7.4 ITERACIÓN 4 CREACIÓN DE USUARIOS.

Requerimiento Nro. 4. – Creación de usuarios.

Tarea Nro. 11. – Creación de usuarios.

Fig. 60: Creación de usuario.
Fuente: Propia.

El administrador crea el usuario con sus datos. Se colocó una información donde indica que el password del nuevo usuario será su número de cédula.

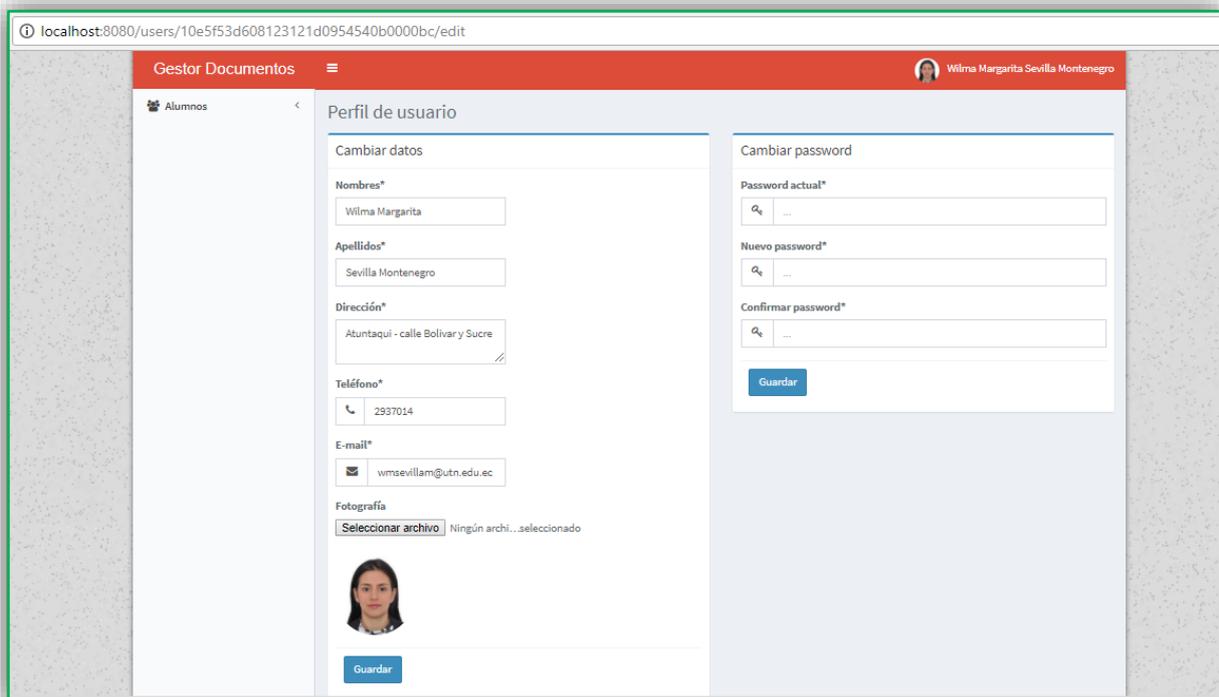


Fig. 61: Ingreso al perfil de usuario creado para ser editado.
Fuente: Propia.

Cuando el usuario este creado, este ingresará mediante el login con su número de cédula y su password al sistema, en el cual ya estará creado su perfil en el que puede ingresar para modificar sus datos.

Tarea N. 12. – Usuarios

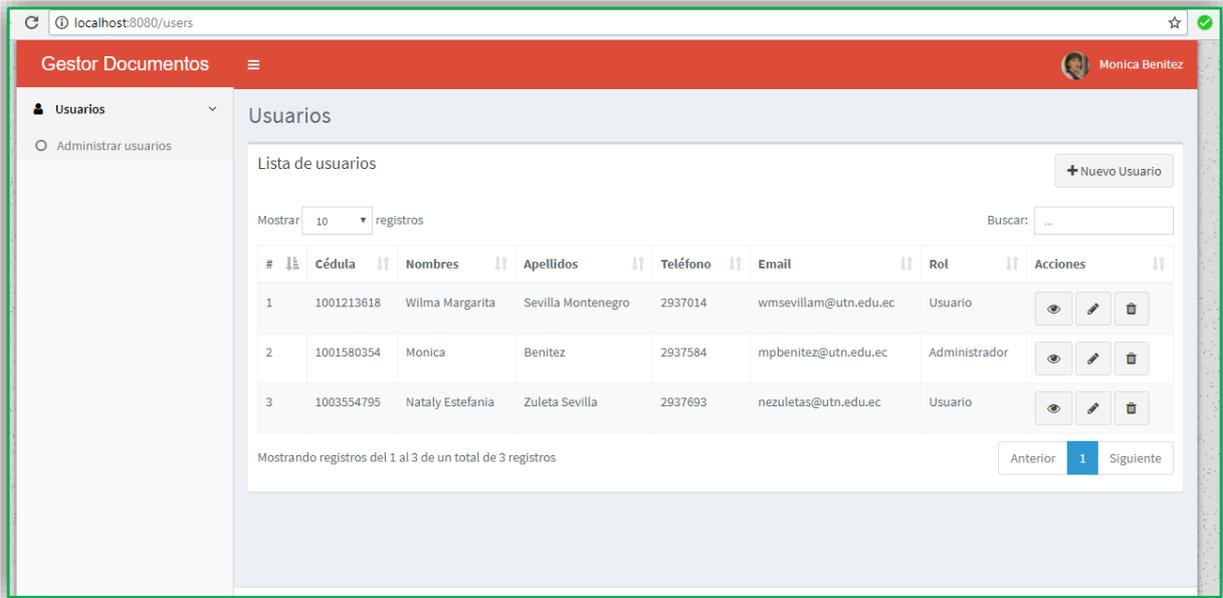


Fig. 62: Usuarios.
Fuente: Propia.

Acciones que puede realizar el administrador: ver, editar o eliminar al usuario.

SPRINT : 4			
FECHA DE INICIO: 25 de Nov. De 2017.			
			25 Nov.
			Tareas Pendientes 0
REQUERIMIENTOS	TAREA	REQUERIMIENTOS	ESTADO
Creación de usuarios.	Creación de usuarios.	Estefanía Zuleta	TERMINADO
	Usuarios.	Estefanía Zuleta	TERMINADO

Tabla 24: Sprint 4 – Hoja electrónica.
Fuente: Propia.

PENDIENTE	EN PROCESO	TERMINADA
_____	_____	Creación de usuarios.
_____	_____	Usuarios.

Tabla 25: Sprint 4 – Pizarrón.
Fuente: Propia.

2.7.5 ITERACIÓN 5 CREACIÓN DE ALUMNOS.

Tarea 13. – Creación de alumnos.

Fig. 63: Creación de alumno.
Fuente: Propia.

Interfaz para el ingreso de los datos personales del alumno.

Tarea 14. - Creación de módulos con sus diferentes ítems.

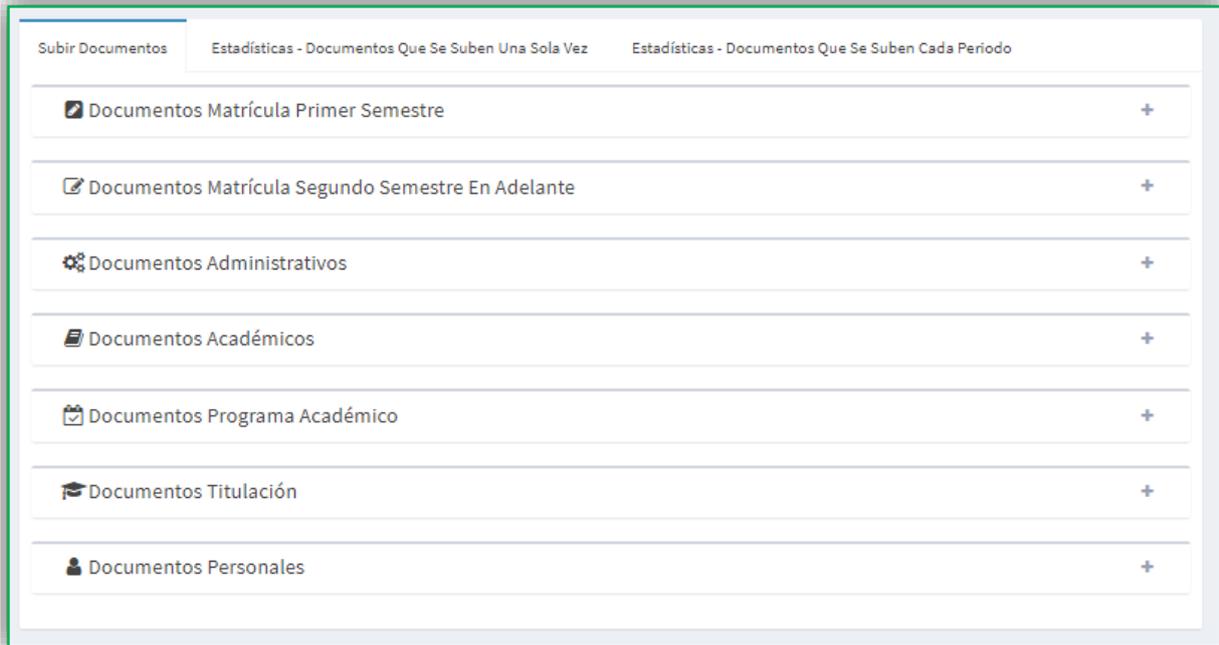


Fig. 64: Creación de módulos con sus diferentes ítems
Fuente: Propia.

Interfaz de los módulos para subir los expedientes de los alumnos, aquí se escanearan y se subirán en formato pdf los documentos que están mencionados en cada módulo.

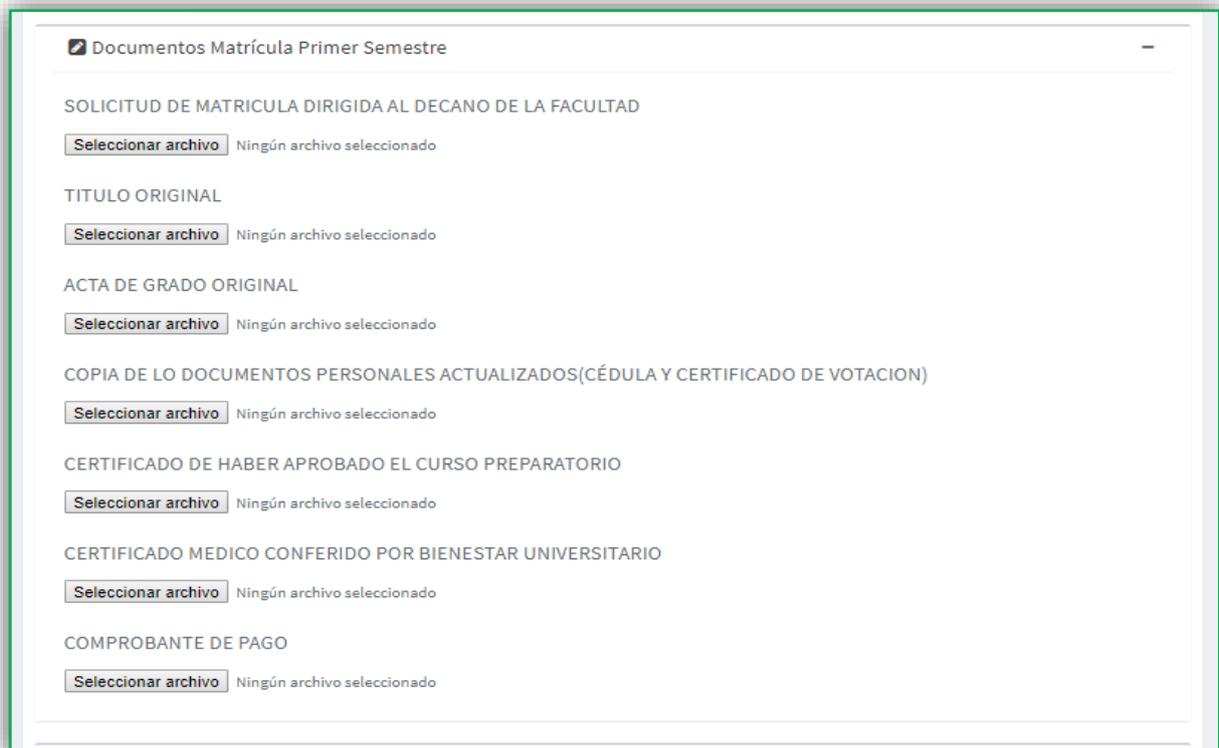


Fig. 65: Ítems del módulo Documentos Matricula Primer Semestre.
Fuente: Propia.

Cada módulo contiene ítems, en los cuales hay que subir los documentos escaneados en formato pdf.

Tarea 15. – Creación de estadísticas – documentos que se suben una sola vez.

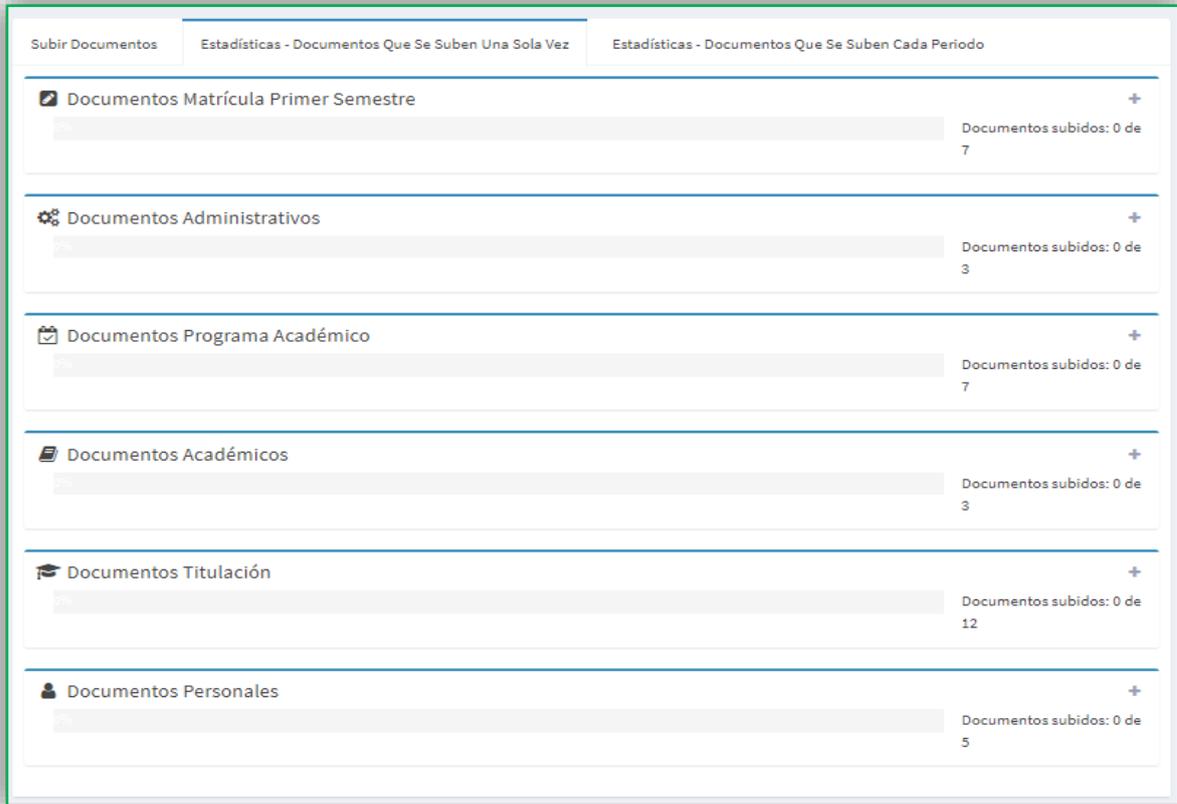


Fig. 66: Creación de estadísticas – documentos que se suben una sola vez.
Fuente: Propia.

Indica el porcentaje de documentos que tiene el alumno.

Tarea 16. – Creación de estadísticas – documentos que se suben cada periodo.

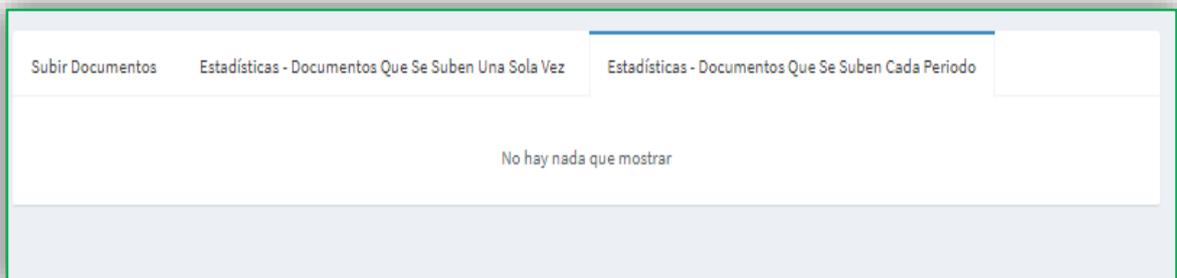


Fig. 67: Creación de estadísticas – documentos que se suben cada período.
Fuente: Propia.

Se muestra el porcentaje total, de los documentos que se suben cada período.

SPRINT : 5			
FECHA DE INICIO: 16 de Dic. 2017			
			16 Dic.
			Tareas Pendientes 0
REQUERIMIENTOS	TAREA	REQUERIMIENTOS	ESTADO
Creación de alumnos.	Creación de alumnos.	Estefanía Zuleta	TERMINADO
	Creación de módulos con sus diferentes ítems.	Estefanía Zuleta	TERMINADO
	Creación de estadísticas – documentos que se suben una sola vez	Estefanía Zuleta	TERMINADO
	Creación de estadísticas – documentos que se suben cada periodo.	Estefanía Zuleta	TERMINADO

Tabla 26: Sprint 5 – Hoja electrónica.
Fuente: Propia

PENDIENTE	EN PROCESO	TERMINADA
_____	_____	Creación de alumnos.
_____	_____	Creación de módulos con sus diferentes ítems.
_____	_____	Creación de estadísticas – documentos que se suben una sola vez.
_____	_____	Creación de estadísticas – documentos que se suben cada periodo.

Tabla 27: Sprint 5 – Pizarrón
Fuente: Propia.

2.7.6 ITERACIÓN 6 IMPLEMENTACIÓN DEL SISTEMA.

Requerimiento Nro. 1. – Implementación del sistema.

Tarea Nro. 17. – Ingreso de alumnos.

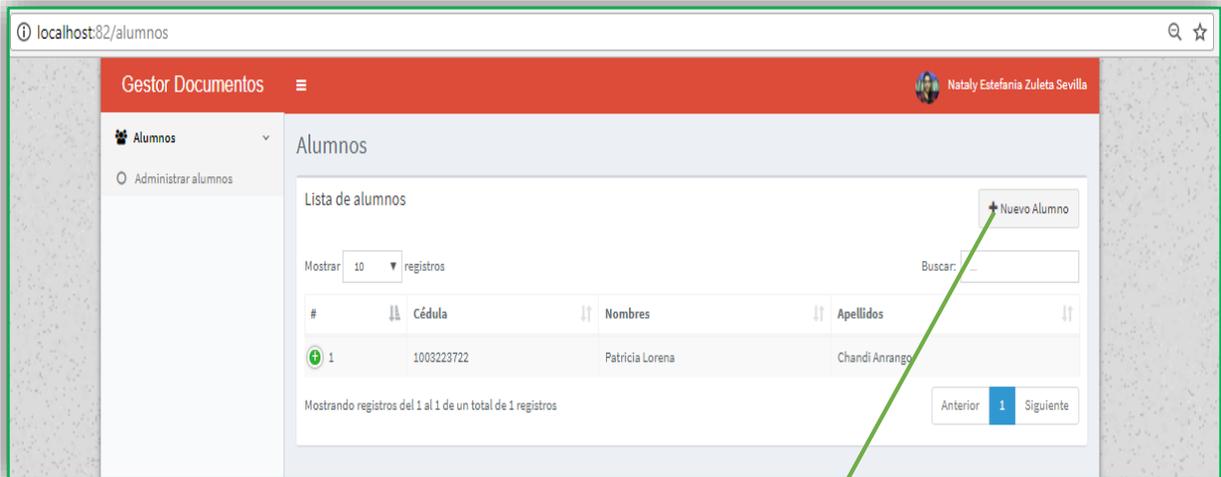


Fig. 68: Botón para crear nuevo alumno.
Fuente: Propia.

Damos clic en la opción **nuevo alumno** y vamos a registrar.

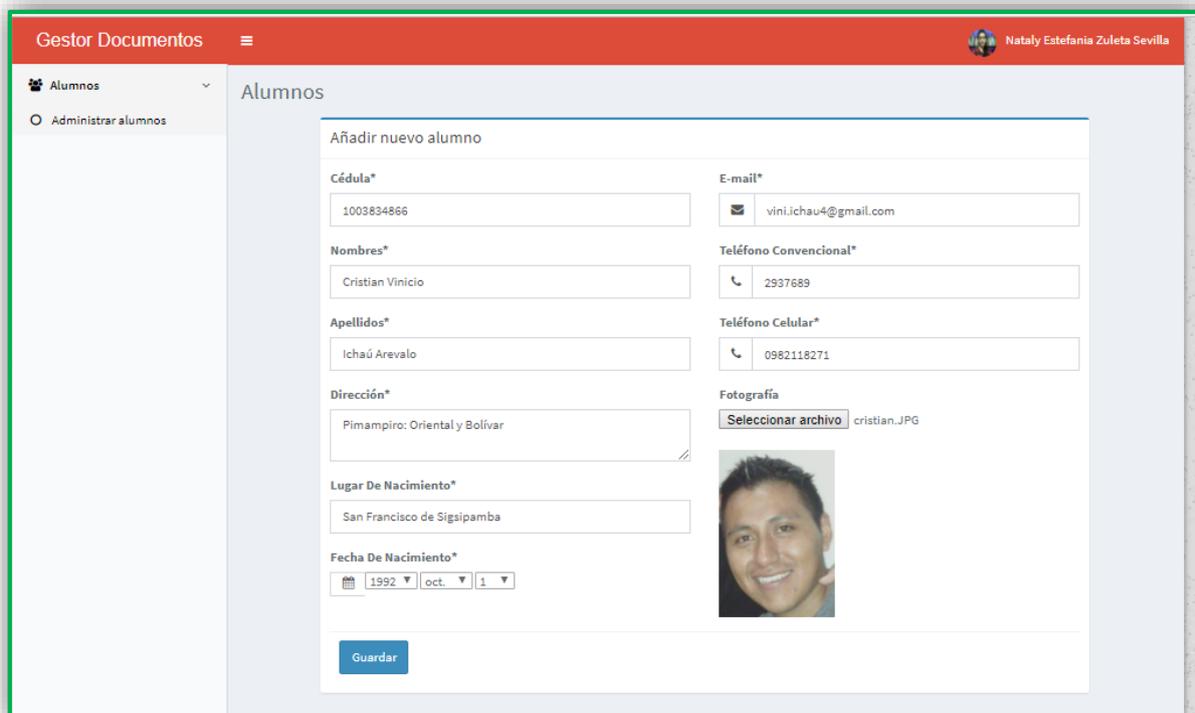


Fig. 69: Creación de alumno.
Fuente: Propia.

Tarea Nro. 18. – Subida de documentos escaneados a los módulos con sus diferentes ítems.

The screenshot displays the 'Gestor Documentos' interface. At the top, the title 'Gestor Documentos' is visible on the left, and the user profile 'Nataly Estefanía Zúñiga Sevilla' is on the right. The main content area is titled 'Alumnos' and shows the 'Información de alumno' for a student named Patricia Lorena Chandi Arango. The student's details are as follows:

Cédula	Nombres	Apellidos
1000225722	Patricia Lorena	Chandi Arango
Dirección	Lugar De Nacimiento	Fecha De Nacimiento
Ibarra	Ibarra	1990-05-21
Teléfono Convencional	Teléfono Celular	E-mail
2084054	0992542070	lorechandi@gmail.com

Below the student information are 'Editar' and 'Eliminar' buttons. The main section is titled 'Subir Documentos' and contains a list of document categories with expandable options:

- Documentos Matrícula Primer Semestre (+)
- Documentos Matrícula Segundo Semestre En Adelante (+)
- Documentos Administrativos
- Documentos Académicos
- Documentos Programa Académico (+)
- Documentos Titulación (+)
- Documentos Personales (+)

Annotations on the image include:

- 'Opciones' with an arrow pointing to the 'Administrar alumnos' link in the left sidebar.
- 'Módulos' with a bracket pointing to the list of document categories.
- 'Hacer clic, despliegue de ítems.' with an arrow pointing to the plus sign next to 'Documentos Matrícula Segundo Semestre En Adelante'.

Fig. 70: Módulos e ítems para subir documentación.

Fuente: Propia

Tarea Nro. 19. – Visualización de las Estadísticas – documentos que se suben una sola vez.

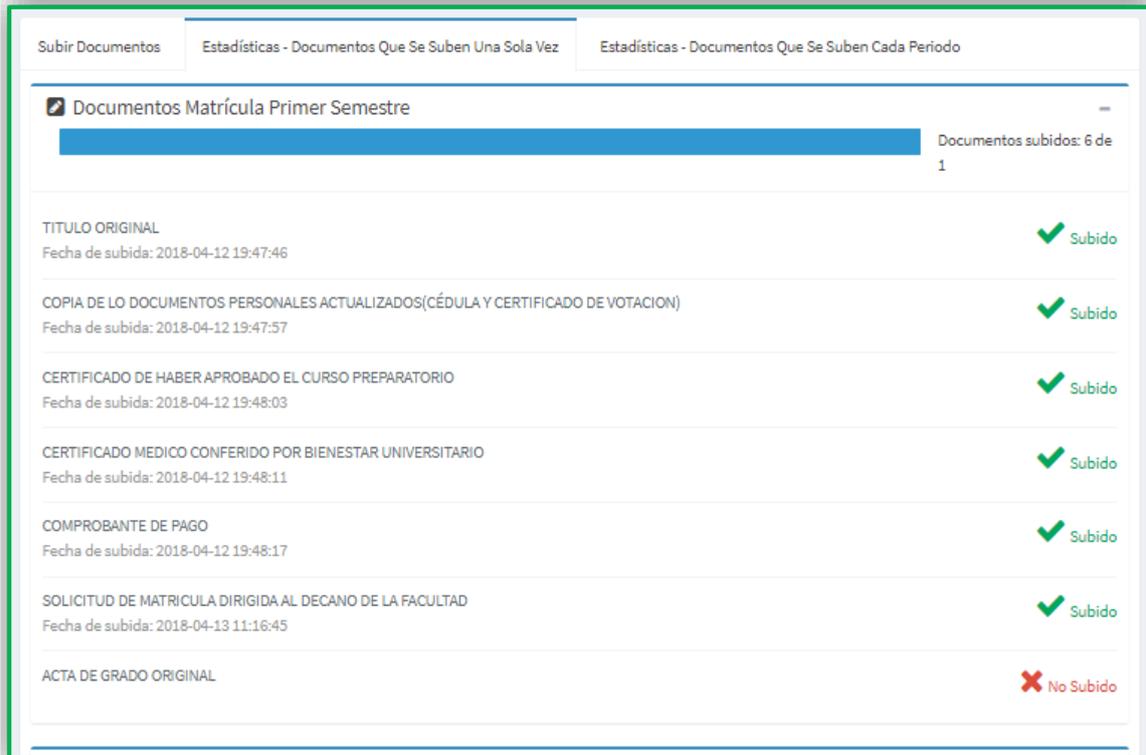


Fig. 71: Estadísticas – documentos que se suben una sola vez.
Fuente: Propia

Tarea Nro. 20. – Visualización de las Estadísticas – documentos que se suben cada periodo.

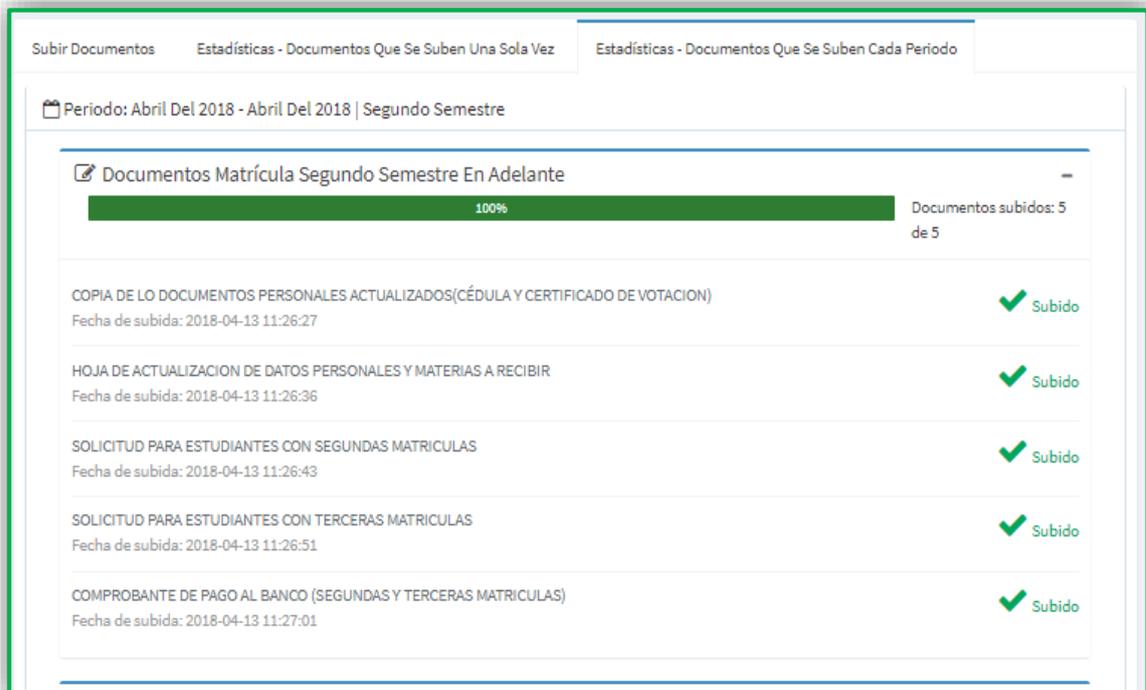


Fig. 72: Estadísticas – documentos que se suben cada periodo
Fuente: Propia.

SPRINT : 1			
FECHA DE INICIO: 03 de			
Ene. De 2018.			
			03 Ene.
		Tareas Pendientes	0
REQUERIMIENTOS	TAREA	REQUERIMIENTOS	ESTADO
Implementación del Sistema.	Ingreso de alumnos.	Estefanía Zuleta	TERMINADO
	Subida de documentos escaneados a los módulos con sus diferentes ítems.	Estefanía Zuleta	TERMINADO
	Visualización de las Estadísticas – documentos que se suben una sola vez.	Estefanía Zuleta	TERMINADO
	Visualización de las Estadísticas – documentos que se suben cada periodo.	Estefanía Zuleta	TERMINADO

Tabla 28: Sprint 6 – Hoja electrónica.
Fuete: Propia.

PENDIENTE	EN PROCESO	TERMINADA
_____	_____	Ingreso de alumnos.
_____	_____	Subida de documentos escaneados a los módulos con sus diferentes ítems.

<p>_____</p>	<p>_____</p>	<p>Visualización de las Estadísticas – documentos que se suben una sola vez.</p>
<p>_____</p>	<p>_____</p>	<p>Visualización de las Estadísticas – documentos que se suben cada periodo.</p>

Tabla 29: Sprint 6 – Pizarrón.
Fuente: Propia.

CAPÍTULO 3

CONCLUSIONES Y RECOMENDACIONES

Conclusiones:

- Luego de realizar la integración entre la aplicación PHP y la base de datos CouchDB mediante la función curl, se puede concluir que debido a que la función mencionada envía peticiones de bajo nivel a los diferentes métodos HTTP de la base de datos CouchDB, resulta muy compleja la implementación de este tipo de sistemas documentales. Por este motivo se buscó un framework PHP que encapsule mediante un API de alto nivel, para poder integrar estas herramientas y diseñar el sistema de una manera más sencilla, siendo el más factible Laravel PHP.
- El sistema gestión de expedientes fue realizado con la herramienta Laravel PHP mediante su API, y permitió desarrollar en forma rápida todos los módulos del sistema de gestión de expedientes.
- Para el desarrollo del sistema se aplicó la metodología SCRUM que permite realizar software complejo y entregarlo en corto tiempo a través de las historias de usuario que luego se traducen en la lista de producto (requisitos funcionales del sistema) y se implementan dentro de las actividades de los sprints.

Recomendaciones:

- Se recomienda utilizar la utilidad Curl como una opción sólo para ejecutar funciones básicas de CouchDB como crear bases de datos mediante métodos PUT, POST y DELETE, ya que se comunica a través del protocolo http internamente con la base de datos CouchDB.
- Se recomienda tener un control estricto en la administración de claves de acceso al sistema gestión de expedientes por parte de los administrativos encargados.
- Se recomienda utilizar la metodología Scrum para un rápido desarrollo de proyectos debido a que, es considerado como el paradigma de la metodología de desarrollo ágil, gracias a la descripción de los roles, sus componentes y la organización de la actividad diaria. Se presenta como una decisión preventiva, ya que esta metodología optimiza los círculos productivos, dinamizando los procesos de producción, dándoles mayor relevancia y acortando drásticamente los periodos de entrega.

- Se recomienda implementar el sistema gestión de expedientes a la nueva carrera de Ingeniería de Software y a todas las carreras de la facultad; en el futuro a las facultades de la UTN debido a que posibilitará el ahorro en tiempo y recursos al disponer en forma inmediata y con precisión la información de los estudiantes.

Referencias

- @AVILLAREJO. (s.f.). *40 de fiebre*. Obtenido de ¿Cómo medir tu ROI en Social Media?: <https://www.40defiebre.com/como-medir-roi-social-media/>
- ACAPELA. (5 de Octubre de 2016). *MADBRICKS*. Obtenido de Esquemas de Producción para Videojuegos: <http://madbricks.co/scrum-kanban/>
- Álvarez Caules, C. (14 de Junio de 2013). *arquitecturajava*. Obtenido de Introducción a Servicios Rest: <https://www.arquitecturajava.com/servicios-rest/>
- andrearrrs. (15 de Mayo de 2014). *Hipertextual*. Obtenido de ¿Qué es API?: <https://hipertextual.com/archivo/2014/05/que-es-api/>
- Apache. (2014). Obtenido de <https://es.opensuse.org/images/c/c4/Apache.jpg>
- APACHE. (2014). Obtenido de <https://es.opensuse.org/Apache>
- APACHE. (23 de Octubre de 2017). *APACHE HTTP SERVER PROJECT*. Obtenido de <https://httpd.apache.org/>
- Artaza, D. (4 de Julio de 2012). *Slideshare*. Obtenido de Base de Datos no Relacional (NoSQL): Cassandra, CouchDB, MongoDB y Neo4j: <https://es.slideshare.net/dipina/nosql-cassandra-couchdb-mongodb-y-neo4j>
- Caballero, G. C., & Montoya, C. R. (2016). *Almacenamiento de la Información e Introducción a SGBD*. Madrid: Ediciones Nobel, S.A.
- CASTILLO, C. (20 de Noviembre de 2016). *SlideShare*. Obtenido de CouchDB - Base de Datos NoSQL: <https://es.slideshare.net/christianecastillo1/couch-db-69323075>
- CESC1989. (22 de Mayo de 2013). *Otro Espacio Blog*. Obtenido de Conoce un poco sobre los Métodos HTTP en REST: <https://otroespacioblog.wordpress.com/2013/05/22/conoce-un-poco-sobre-los-metodos-http-en-rest/>
- Cividas. (2015). *La gestión de expedientes por procesos en cividas*. Obtenido de <http://www.cividas.com/la-gestion-de-expedientes-por-procesos-en-cividas/>
- Cogneau, A. (8 de Agosto de 2012). *Laravel: Is It Really Clean and Classy?* Obtenido de <https://www.sitepoint.com/laravel-really-clean-and-classy/>
- Colbert, J. (27 de Mayo de 2014). *Prezi*. Obtenido de CouchDB: https://prezi.com/aoab9u2hi4z_/couch-db/
- Colimba, P. (21 de Julio de 2016). *Repositorio UTN*. Obtenido de "Desarrollo de un portal web utilizando el framework Laravel y Bootstrap para la difusión turística de la Hosteria Fabricio's en la parroquia de Ambuquí".: <http://repositorio.utn.edu.ec/bitstream/123456789/5698/1/04%20ISC%20427%20TRABAJO%20DE%20GRADO.pdf>
- CouchDB. (2013). *Enciclopedia libre*. Obtenido de CouchDB: <https://es.wikipedia.org/wiki/CouchDB>
- CouchDB, A. (2014). *Apache CouchDB relax*. Obtenido de <http://couchdb.apache.org/>

- Cruz, R. (4 de Noviembre de 2014). *Cómo instalar Composer y Laravel en Windows*. Obtenido de <https://styde.net/instalacion-de-composer-y-laravel-en-windows/>
- Devacademy. (2015). *Tutorial #Laravel: Introducción y Conceptos básicos por @jlinformatico*. Obtenido de <http://blog.devacademy.la/post/94202131491/tutorial-laravel-introducci%C3%B3n-y-conceptos/embed>
- Dimes, T. (2015). *Conceptos Básicos De Scrum: Desarrollo De Software Agile Y Manejo De Proyectos Agile*. Babelcube Inc., 2015.
- Drauta, S. (26 de 10 de 2016). *Laravel*. Obtenido de <https://www.drauta.com/que-es-laravel-eloquent>
- Editor. (7 de Abril de 2014). *Devellar*. Obtenido de WHY NOSQL AND HOW IT WILL DISRUPT BIG DATA: <http://devellar.com/why-nosql-and-how-it-will-disrupt-big-data/>
- Editor. (20 de Mayo de 2016). *¿Cuál es la metodología Scrum ?* Obtenido de <http://www.conocimientosweb.net/dcmt/ficha26242.html>
- Espinoza, P. (27 de Enero de 2016). *Prezi*. Obtenido de CouchDB: <https://prezi.com/xmk3ja33ptub/couchdb/>
- fácil, T. (26 de Julio de 2015). *Tecnología Fácil*. Obtenido de ¿Qué es Open Source?: <https://tecnologia-facil.com/que-es/que-es-open-source/>
- Francia, J. (25 de Septiembre de 2017). *Scrum.org*. Obtenido de ¿Qué es Scrum?: <https://www.scrum.org/resources/blog/que-es-scrum>
- freelancer. (17 de Marzo de 2016). *Laravel: ventajas del framework PHP de moda*. Obtenido de <https://www.freelancer.ec/community/articles/ventajas-del-framework-moda-laravel>
- González de Artaza, D. L. (18 de Julio de 2013). *SlideShare*. Obtenido de NoSQL: Introducción a las Bases de Datos no estructurales: <https://es.slideshare.net/dipina/nosql-introduccion-a-las-bases-de-datos-no-estructuradas>
- Guest, A. (19 de Abril de 2016). *Innovación tecnológica*. Obtenido de Un café con Oracle: <https://blogs.oracle.com/uncafeconoracle/qu-es-una-base-de-datos-nosql>
- Gutiérrez González, E. (s.f.). *¿Qué es PHP? y ¿Para qué sirve? Un potente lenguaje de programación para crear páginas web. (CU00803B)*. Obtenido de [aprenderaprogramar.com: https://www.aprenderaprogramar.com/attachments/article/492/CU00803B%20Que%20es%20PHP%20para%20sirve%20potente%20lenguaje%20programacion%20paginas%20web.pdf](https://www.aprenderaprogramar.com/attachments/article/492/CU00803B%20Que%20es%20PHP%20para%20sirve%20potente%20lenguaje%20programacion%20paginas%20web.pdf)
- Imaginanet. (29 de Septiembre de 2015). *BLOG: SQL vs NoSQL*. Obtenido de <https://www.imaginanet.com/blog/sql-vs-nosql-como-elegir.html>
- J, G. (9 de Octubre de 2017). *Definición de Javascript*. Obtenido de <http://conceptodefinicion.de/javascript/>
- KYOCERA. (27 de Junio de 2017). *¿Qué son las bases de datos documentales?* Obtenido de <https://smarterworkspaces.kyocera.es/blog/las-bases-datos-documentales/>
- LaravelBook. (2014). *Architecture of Laravel Applications*. Obtenido de <http://www.laravelbook.com/laravel-architecture/>

Martín, S. (20 de Abril de 2017). *Bases de datos NoSQL : Guía definitiva*. Obtenido de <https://blog.pandorafms.org/es/bases-de-datos-nosql/>

MediaWiki. (9 de Diciembre de 2015). Obtenido de CSS: <https://www.w3.org/wiki/Es/CSS>

Mera, F. (29 de Junio de 2013). *SlideShare*. Obtenido de Paginas web dinámicas y estáticas: <https://es.slideshare.net/FloydMera98/paginas-web-dinmicas-y-estticas>

Navarro Marset, R. (s.f.). *REST vs Web Services*. Obtenido de ¿Qué es REST realmente?: <http://users.dsic.upv.es/~rnavarro/NewWeb/docs/RestVsWebServices.pdf>

Nilanchala. (24 de Abril de 2017). *stacktips*. Obtenido de Introducción a Laravel PHP Framework Features and Version History: <http://stacktips.com/tutorials/laravel/intro-to-laravel-php-framework-and-features#4-laravel-features>

Norte, U. T. (s.f.). *Blog Ingeniería en Sistemas Computacionales*. Obtenido de http://www.utn.edu.ec/fica/carreras/sistemas/?page_id=6

Queralt, E. (31 de Julio de 2014). *BLOG DE OPEN SCHOOL*. Obtenido de ¿Qué es el método SCRUM y que beneficios y ventajas va a aportar a mi negocio?: <https://www.openiebs.com/blog/que-es-scrum-beneficios-ventajas-para-negocio-agile-scrum/>

Refman, J. (17 de Marzo de 2016). *evatotuts+*. Obtenido de ¿Qué es HTML5?: <https://code.tutsplus.com/es/tutorials/what-is-html5--cms-25803>

Rouse, M. (2014). *TechTarget*. Obtenido de CouchDB: <http://searchdatacenter.techtarget.com/es/definicion/CouchDB>

Sanchez, J. (26 de Julio de 2014). *JSON: ¿QUÉ ES JSON?, ¿CÓMO SE USA? Y ¿PARA QUÉ SIRVE JSON?* Obtenido de <https://frontendlabs.io/1490--json-que-es-json-parse-json-stringify>

TAYLOR, O. (2016). *Laravel*. Obtenido de The PHP Framework For Web Artisan: <https://laravel.com/>

tic.Portal. (2016). *Sistema de Gestión Documental*. Obtenido de <https://www.ticportal.es/temas/sistema-gestion-documental/que-es-sistema-gestion-documental>

Ulloa, D. G. (Noviembre de 2014). *UNIVERSIDAD TÉCNICA DE AMBATO*. Obtenido de ESTUDIO DE METODOLOGÍAS PARA ESTANDARIZAR EL DESARROLLO DE SOFTWARE EN EL DEPARTAMENTO DE INFORMÁTICA EN LA PASTORAL SOCIAL CARITAS DE LA DIÓCESIS DE AMBATO.: http://repositorio.uta.edu.ec/bitstream/123456789/8524/1/Tesis_t937si.pdf

UTN-CREACIÓN. (2010). *Blog Carrera CISIC – UTN*. Obtenido de http://www.utn.edu.ec/fica/carreras/sistemas/?page_id=6

UTN-HISTORIA. (2010). *Blog Carrera CISIC – UTN*. Obtenido de http://www.utn.edu.ec/fica/carreras/sistemas/?page_id=6

UTN-INTRODUCCIÓN. (2010). *Blog Carrera CISIC – UTN*. Obtenido de http://www.utn.edu.ec/fica/carreras/sistemas/?page_id=2103

vBote. (16 de Julio de 2014). *vBote mobilize your business*. Obtenido de Gestión de proyectos, metodología de desarrollo ágil : Scrum: <http://www.vbote.com/actualidad-detalle.html?name=86-gestion-de-proyectos-metodologia-de-desarrollo-agil-scrum>

vBote. (16 de Julio de 2014). *vBote movilize your business*. Obtenido de Gestión de proyectos, metodología de desarrollo ágil: Scrum: <http://www.vbote.com/actualidad-detalle.html?name=86-gestion-de-proyectos-metodologia-de-desarrollo-agil-scrum>

Wikipedia. (2 de Febrero de 2018). *Wikipedia la enciclopedia libre*. Obtenido de JSON: <https://es.wikipedia.org/wiki/JSON>

Xnechijli01. (20 de Agosto de 2016). *ABC DIARIO*. Obtenido de CARACTERISTICAS DE LARAVEL, DESARROLLO WEB: <http://www.diario-abc.com/nota/2461/caracteristicas-de-laravel-desarrollo-web.html>