PONTIFICIA UNIVERSIDAD CATOLICA DEL ECUADOR

FACULTAD DE INGENIERIA

ESCUELA DE SISTEMAS

ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA LA GESTIÓN DE CARNETIZACIÓN PARA ORGANIZACIONES.

CASO DE ESTUDIO: CONSULTORES TECNOLÓGICOS Y DESARROLLO EMPRESARIAL

Autor: RICARDO PAÚL URVINA CÓRDOVA

DISERTACIÓN DE GRADO PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SISTEMAS Y COMPUTACIÓN

Director: Ing. OSWALDO ESPINOSA

QUITO, 2020

Dedicatoria.

A mi madre,

Pilar fundamental en mi proceso de formación estudiantil, por su apoyo incondicional, por sus valores inculcados en mi persona por enseñarme el amor y la empatía con el resto , pero por sobre todo por la perseverancia y paciencia que siempre tiene conmigo con el fin de que logre todos mis objetivos.

A mi padre,

Por haberme criado con un carácter fuerte, por ser mi mentor dentro de casa, fuera de casa y en este privilegio que tuve que sea mi mentor dentro de las aulas de la PUCE, por enseñarme que lo ético y moral es tan necesario en la vida para lograr los objetivos en base de esfuerzo y perseverancia.

A mi hermano,

Por siempre estar conmigo como amigo cuando el resto me falta, por siempre darme ese hombro y el ánimo necesario que necesité cuando parecía que todo se derrumbada.

Agradecimiento.

Agradezco principalmente a mi núcleo familiar que siempre han sido mi sitio seguro, a mis padres por darme la oportunidad privilegiada de estudiar este pregrado, por guiarme en este camino llamado vida, por enseñarme todo lo bueno que una persona puede aprender desde el día que nace hasta la actualidad, agradezco en todo aspecto a ellos y a mi hermano por siempre presionarme y enseñarme a nunca rendirme, por ayudarme a salir adelante personal y profesionalmente.

Agradezco a mi enamorada Ingrid Medina, por haber compartido la mitad de este episodio llamada Ingeniería, agradezco tu apoyo incondicional, por siempre creer en mi, por darme tu amor y tu mano para cualquier necesidad que tuve durante la carrera.

Al Ing. Oswaldo Espinosa, director del presente trabajo, por la colaboración y apoyo durante el proceso de titulación.

Finalmente agradezco a todas las personas que de cierta manera me brindaron una mano, una palabra de apoyo, una conversación o me brindaron una ayuda para cualquier tema o problema dentro y fuera de la universidad.

Tabla de contenidos

Dedicatoria.....ii

Agradecimiento.....iii

Tabla de Ilustraciones vii

Resumen. ix

Capitulo 1: Metodología y Herramientas 1

Objetivo General:..... 1

Objetivos Específicos 1

1.1. Metodología..... 1

1.1.1. SCRUM 1

1.2. Herramientas..... 5

1.2.1. Power Designer 5

1.2.2. MySQL..... 6

1.2.3. WorkBench CE 6

1.2.4. PHP..... 6

1.2.5. Laravel..... 7

1.2.6. JavaScript 8

1.2.7. Front End 9

Capitulo 2: Situación actual de la empresa 10

2.1. Historia de la empresa..... 10

2.2. Enfoque de la empresa...... 11

2.3. Situación actual de la empresa...... 13

Capitulo 3: Análisis y Diseño 18

3.1. Análisis del sistema...... 18

3.1.1. Requerimientos Funcionales..... 20

3.1.2. Requerimientos No Funcionales 20

3.1.3. Casos de Uso..... 21

3.2 Diseño de la Base de Datos 23

a. Modelo Conceptual 24

b. Diagrama Físico 26

c. Diagrama Entidad Relación 27

3.3 Diseño del BackEnd. 28

3.4 Diseño de FrontEnd (Interfaz Gráfica). 30

a. Diseño Ingreso al Sistema 31

b. Diseño recuperación Contraseña 31

a. Diseño Tablero Principal (Administrador) 32

b. Diseño Tablero Principal (Usuario/Estudiante) 32

c. Diseño Visualización de Listas de Registros 33

d. Diseño Visualización Datos Usuario/Estudiante 34

e. Diseño Visualización Instituciones/Organizaciones 34

f. Diseño Ingreso de Datos 35

Capitulo 4: Desarrollo del Sistema Web e implementación. 36

4.1. Desarrollo del BackEnd. 36

a. Modelos. 36

b. Controladores. 37

c. Vistas. 39

d. Request 40

e. Middleware. 41

f. Policies 41

g. Rutas 42

h. Migraciones. 43

Realización del carné. 44

Control de versionamiento 46

4.2. Desarrollo de FrontEnd (Interfaz Gráfica). 48

4.3. Implementación del Sistema Web en el dominio de la empresa. 51

Capitulo 5: Conclusiones y Recomendaciones. 60

5.1 Conclusiones. 60

5.2. Recomendaciones. 60

Referencias 62

Anexos 65

Anexo 1. Script Base De Datos. 65

Glosario de términos 74

Tabla de Ilustraciones

Ilustración 1 Funcionamiento de un sprint en la metodología Scrum 5
Ilustración 2 Logo Cotedem 11
Ilustración 3 Diagrama General 22
Ilustración 4 Diagrama General 23
Ilustración 5 Modelo Conceptual 24
Ilustración 6 Diagrama Lógico 25
Ilustración 7Diagrama Físico 26
Ilustración 8 Diagrama Entidad Relación 27
Ilustración 9 Funcionamiento API Rest 28
Ilustración 10 Ingreso al Sistema 31
Ilustración 11 Recuperar Contraseña 32
Ilustración 12 Tablero Principal (Administrador) 32
Ilustración 13 Tablero Principal (Usuario/Estudiante) 33
Ilustración 14 Visualización Listas de Registros 33
Ilustración 15 Visualización de Datos 34
Ilustración 16 Visualización Instituciones/Organizaciones 35
Ilustración 17 Ingreso de Datos 35
Ilustración 18 Modelo Student 37
Ilustración 19 StudentController 38
Ilustración 20 Vista Index Students 39
Ilustración 21 StudentRequest 40
Ilustración 22 Middleware CheckRoles 41
Ilustración 23 StudentPolicy 42
Ilustración 24 Web.php 43
Ilustración 25 Migración Estudiantes 44
Ilustración 26 Realización Carné 45
Ilustración 27 Formato Carné 46
Ilustración 28 Punto de Entrada FrontEnd 49
Ilustración 29 SideBar Sección de FrontEnd 50
Ilustración 30 Vistas CRUD Student 51
Ilustración 31 compresión Carnet-Cotedem 52
Ilustración 32 Cpanel BlueHost 52
Ilustración 33 Carpeta idCard en public_html 53
Ilustración 34 Carpeta carnet-cotedem 54
Ilustración 35 Configuración index.php 55
Ilustración 36 Configuracion .env 56

ILUSTRACIÓN 39 CORRECTO FUNCIONAMIENTO DEL APLICATIVO EN EL SERVIDOR 58
ILUSTRACIÓN 40 CORRECTO FUNCIONAMIENTO DEL APLICATIVO EN EL SERVIDOR 58

Resumen.

Cotedem es una empresa que inicia formalmente sus actividades en el 2014 con el fin de realizar distinto tipo de actividades, prestando servicios y venta de productos tecnológicos, actualmente el principal giro de la empresa es la venta de servicios tecnológicos con proyectos ya establecidos como auditorias informáticas, entre otros servicios en el afán de expandir los servicios que la empresa presta surge la idea de carnetizar a instituciones educativas y organizaciones, mediante esta idea empieza este proyecto de disertación , el cual consiste en crear una aplicación web que permita hacer una carnetización controlada para cualquier tipo de organización .

Ya que en la actualidad realizar una aplicación web es de gran ayuda para automatizar procesos dentro de una empresa, esta no es la excepción y mediante la realización de este proyecto de titulación se logró hacer un sistema web basado en la arquitectura MVC y estándares de programación y del propio framework en el que se lo desarrolló.

El sistema es capaz de registrar a instituciones u organizaciones con sus respectivos alumnos o usuarios a los cuales se les genera un carné con los datos personales e imágenes propias de cada organización, el carné se lo genera a manera de PDF para su posterior impresión en una impresora especializada para la impresión de este tipo de identificaciones.

Capitulo 1: Metodología y Herramientas

Para la realización de este proyecto se planteó un objetivo general y objetivos específicos.

Objetivo General:

Desarrollar un Sistema Web que permita llevar una gestión controlada de carnetización.

Objetivos Específicos

- Recolectar toda la información necesaria sobre las actividades y procesos que realiza la empresa "COTEDEM" de modo que permita definir los requerimientos del sistema.
- Diseñar el sistema web en base a la lista de requerimientos específicos de la empresa.
- Desarrollar el back-end y front-end tomando en cuenta el diseño del sistema web.
- Implementar el sistema de gestión controlada de carnetización dentro de la empresa.

La metodología para realizar el presente trabajo será: levantamiento de requerimientos mediante charlas con el dueño de la empresa, después se analizará el trabajo a realizar, con en el análisis completo se procederá a realizar el diseño de la base de datos, con la base de datos diseñada se diseñará el *BackEnd* y el *FrontEnd*, para posteriormente realizar el *BackEnd* con *PHP* mediante el *Framework* de *Laravel*, para el *FrontEnd* se utilizará *JavaScript*, *HTML5, CCS3*, y el motor de plantillas de Laravel, el motor de base de datos a utilizarse es *MySQL* con *Workbench*.

1.1. Metodología

1.1.1. SCRUM

Es una metodología ágil, es un marco de trabajo iterativo e incremental, la cual se centra en equipos multifuncionales los cuales tienen decisión propia, consta de iteraciones rápidas con el cliente. Se desarrolla mediante ciclos llamados Sprint, Iteraciones de 1 a 4 semanas, y que suceden una tras otra. Los Sprint son fijos y nunca se alargan.

Al principio de cada iteración o sprint el equipo multifuncional establece y selecciona los requisitos del cliente llamados elementos en una lista priorizada. Y se compromete con el cliente a terminar los requisitos al final del sprint. Durante la iteración no se puede hacer cambios en los requisitos establecidos para el sprint. La manera que SCRUM hace su trabajo semanal es: todos los días el equipo se reúne para comunicar como ha avanzado el progreso, se actualizan las gráficas que restan del trabajo, al final del Sprint se revisa todo lo realizado con los interesados del proyecto y se muestra lo que se ha construido, se recaba comentarios de los interesados para corregir o poner en marcha en el siguiente sprint. Al final se conseguirá un producto con código integrado, completo y probado. (Pete Deemer, 2009)

El proceso

Empieza mediante la lista de objetivos o requisitos que prioriza al producto, el cual será el plan del proyecto. En la lista el cliente (Product Owner) prevalece los objetivos dependiendo del valor que aportar a su coste, el equipo estima estos valores, y se reparte las iteraciones y posteriores entregas.

Planificación de la iteración

Al momento de realizar la reunión el primer día se consta de dos partes:

1. **Selección de requisitos:** en un tiempo estimado de dos horas el cliente propone al equipo una lista con requisitos, los cuales están priorizados dependiendo del producto o proyecto. El equipo de desarrollo pregunta al cliente las dudas acerca de los requisitos y se seleccionan los más prioritarios para poder completar la iteración, de manera que se puedan entregar en los tiempos establecidos con el cliente.
2. **Planificación de la iteración:** en un tiempo máximo de dos horas el equipo elabora una lista de tareas de la iteración indispensables para que se desarrolle los requisitos establecidos. Las tareas deben tener un balance dentro del equipo para que el esfuerzo sea equitativo y se logren mejores resultados, con las tareas asignadas cada integrante se organiza de manera individual para desarrollar las tareas asignadas, en ciertas ocasiones se hace en grupos de dos o más personas con la finalidad de compartir conocimiento creando un equipo las resiliente.

Ejecución de la iteración

En cada día de cada sprint todo el equipo se reúne durante máximo 15 minutos, delante de un tablero (Scrum Taskboard). Se inspecciona el trabajo realizado, se revisa todas las dependencias entre tareas, el progreso de la iteración, las dificultades que pueden impedir llegar al objetivo. Para poder hacer las correcciones necesarias para cumplir lo previsto para mostrar los objetivos al final de la iteración. Se responden tres preguntas claves:

¿Qué hizo desde la última reunión sincronizada?

¿Qué va a hacer para cumplir con el objetivo?

¿Cuáles son los impedimentos que se tiene para cumplir el objetivo?

Mientras dura cada iteración el facilitador (Scrum Máster) se encarga de que no se pierda el foco para llegar a cada objetivo.

- Elimina los impedimentos que el equipo no pueda resolver.
- Se encarga que el equipo no sufra interrupciones externas que puedan afectar el cumplimento de la iteración y disminuya la productividad.

El cliente reunido con el equipo en la iteración revisa los requisitos y si es necesario se re afinan, de tal manera que estén listos para las siguientes iteraciones, si esto se cumple de manera objetiva se logra maximizar la utilidad y el retorno de la inversión.

Inspección y adaptación.

El último día de la iteración se revisa todo el trabajo realizado durante el sprint, la cual consta de 2 partes:

1. **Revisión y demostración:** durante un máximo de 1.5 horas el quipo presenta todo el trabajo realizado al cliente, con los requisitos completados durante la iteración. En función de los resultados y de los cambios que hayan ocurrido en el contexto del proyecto, el cliente analiza e informa las adaptaciones necesarias.
2. **Retrospectiva:** durante un máximo de 1.5 horas el equipo analizar el rendimiento y la manera que se trabajó, cuáles fueron los problemas en la anterior iteración y cuáles podrían ser los problemas futuros que impedirían el progreso del proyecto, se hace todo este análisis para mejorar de manera

continua la productividad. El ScrumMaster elimina o escala los obstáculos que no formen parte del ámbito de acción del equipo. (Pete Deemer, 2009)

Principales características.

Se tiene una alineación entre el cliente y el equipo de desarrollo que permite levantar las expectativas del cliente, flexibilidad y adaptación.

Roles en Scrum

1. Product Owner

Es el responsable de decidir el trabajo a realizar bajo sus necesidades, maximiza el valor del producto o proyecto que se lleve a cabo.

Para este proyecto será el Gerente general de Consultores Tecnológicos y Desarrollo Empresarial el cual ayuda al usuario a escribir las historias de usuario, prioriza y coloca en Product Backlog.

2. ScrumMaster

Es el líder que ayuda al equipo y lo organiza para usar de manera efectiva Scrum, focaliza el negocio y es el responsable de la rentabilidad del proyecto. Encargado de indicar la visión del proyecto al equipo, formaliza las prestaciones en historias las cuales se incorporan en el Product Backlog.

Para este proyecto será el Gerente general de Consultores Tecnológicos y Desarrollo Empresarial. (IT, s.f.)..

3. Equipo de desarrollo

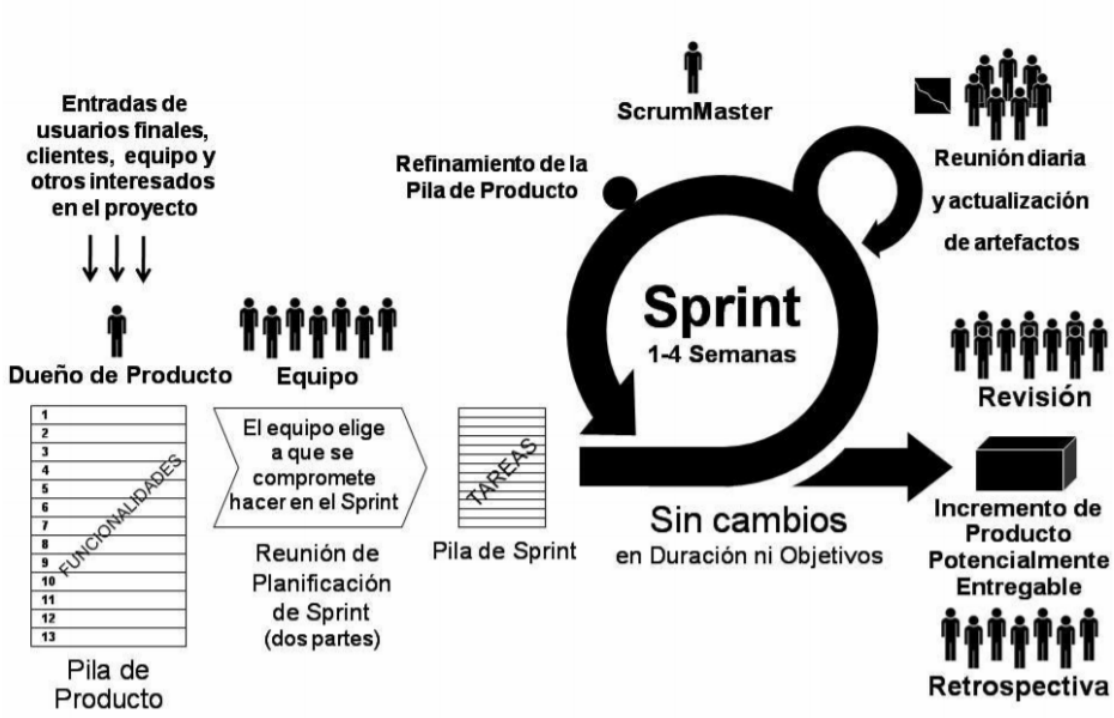
Personas profesionales con conocimientos bastos para desarrollar el proyecto de manera conjunta.

El equipo de desarrollo estará conformado por: Ricardo Paul Urvina Cordova

Documentos.

- Product Backlog: Documento de alto nivel, contiene descripciones genéricas de funcionalidades deseables y priorizadas.
- Sprint backlog: Subconjunto de objetivos que serán desarrollados durante cada ciclo del Sprint, manera en que el equipo desarrollará cada Sprint.

En la ilustración 1, se presenta el funcionamiento de la metodología Scrum.



Fuente: Pete Deemer, Gabrielle Benefield, Craig Larman, Bas Vodde

Ilustración 1 Funcionamiento de un sprint en la metodología Scrum

(Pete Deemer, 2009)

1.2. Herramientas

1.2.1. Power Designer

Power designer es una herramienta para modelar datos, la cual nos permite analizar, manipular y visualizar datos de manera más organizada e inteligente, con una construcción solida de bases de datos, desarrollando modelos orientados a nivel físico y conceptual, (PowerDesigner, 2019).

Entre sus principales características tenemos:

- Visualizar, analizar y manipular metadatos para lograr una efectiva arquitectura empresarial de información.
- Enfoques en modelos para alinear el negocio con las tecnologías de información, esto facilita la implementación de arquitecturas efectivas.
- Brinda estándares de modelamiento con herramientas líderes en desarrollo como .NET, Sybase WorkSpace, Java, Eclipse entre otras, con lo cual permite dar soluciones de análisis de negocio y diseño formal de base de datos.

1.2.2. MySQL

MySQL es un sistema gestor de base de datos relaciones, es de código abierto pensado para entornos de desarrollo web, una base de datos relacional almacena datos en tablas separadas en vez de poner todos los datos en una sola tabla o un gran archivo, lo cual permite que la base de datos sea veloz y flexible, (Oracle, dev.mysql, 2019).

1.2.3. WorkBench CE

MYSQL Workbench es una herramienta que nos permite visualizar de manera unificada las arquitecturas de bases de datos, desarrolladores y DBA, brinda modelados de datos, desarrollo de SQL, herramientas de administración y configuración para servidores, usuarios y copias de seguridad, (Oracle, mysql, 2020).

Entre sus principales características tenemos:

- Diseño: permite diseñar, modelar, generar y administrar bases de datos, nos provee todo lo necesario para crear modelos complejos de ER (Entidad-Relación), provee características claves para realizar tareas, gestionar y documentar todo lo necesario en una base de datos.
- Desarrollar: permite crear, ejecutar y optimizar consultas SQL. El editor SQL ayuda con el resaltado de sintaxis de color, autocompleta y la reutilización de fragmentos de SQL.
- Administrar: proporciona una consola para administrar fácilmente los entornos de MYSQL y obtener visibilidad de las bases de datos.
- Panel de rendimiento visual: provee conjunto de herramientas para mejorar el rendimiento de las aplicaciones MySQL.
- Migración de base de datos: proporciona una solución compleja y fácil de usar, para migrar bases de datos de otros motores de bases de datos como Microsoft SQL Server, Access, Sybase, entre otros.

1.2.4. PHP

Acrónimo de Hypertext Preprocessor es un lenguaje de programación de código abierto utilizado para el desarrollo web y que se incrusta en HTML, se centra en la programación de scripts del lado del servidor aun que es utilizado para muchas otras cosas, (Ferré, 2004).

1.2.5. Laravel

Es un framework de desarrollo creado en 2011, se influencio en frameworks como Ruby on Rails, Sintra y ASP.NET MVC pensado en PHP, implementa el Modelo Vista Controlador, desarrollado por Taylor Otwell, para desarrollar sistemas en laravel se necesita un servidor web Apache. Laravel facilita su uso, es potente y tiene una interfaz elegante y fácil de usar, (Guetierrez, 2018).

El principal objetivo del uso de Laravel es el uso de sintaxis refinada y expresiva para crear códigos de manera más sencilla, evita el código espagueti y permite muchas funcionalidades, utiliza también las últimas versiones de PHP.

Sus características generales son: el sistema de ruteo, también RESTful, Blade, Motor de plantillas, peticiones Fluent, Eloquent ORM, basado en Composer, soporte para el caché, soporte para MVC, usa componentes de Symfony.

Laravel permite almacenar sus rutas en caché y middleware, permite inyección de dependencias en rutas y controladores, permite usar una autenticación preinstalada, permite controlar Oahu de forma óptima este componente se llama Socialice.

Permite estructurar las carpetas, cambios en el Blade, contracts para que sirva como documentación, comandos y eventos, fylesystem el cual permite cambiar a control remoto.

En conclusión, laravel es una herramienta poderosa para desarrollar software, no necesita de requerimientos raros sobre el stack, la documentación existente es completa y la ayuda que se puede recibir en la comunidad laravel es amplia, (Baquero Garcia, 2015)

Framework de desarrollo

Marco de trabajo que permite el desarrollo de software de una manera más sencilla, dependiendo del lenguaje de programación existen varios FrameWorks que suelen incluir librerías para facilitar el desarrollo de software, mejorando la eficiencia en la creación de nuevo software, mejoran la calidad, fiabilidad y robustez. Se centra en las necesidades únicas de la aplicación en lugar de la infraestructura de aplicaciones. Permite la creación de programas mediante librerías de código y módulos ya listos para la implementación según el requerimiento. La mayoría adapta la forma de Modelo Vista Controlador (MVC).

Aplicación Web

Las aplicaciones web son la evolución de las aplicaciones tradicionales, comúnmente llamada web 2.0. son aplicaciones enfocadas al usuario final, colaborativas y que brinden servicios que remplacen a las tradicionales aplicaciones de escritorio. Este término aparece en una conferencia gracias a Dale Dougherty donde se hablaba del renacimiento y evolución de la web.

El primer evento de la web 2.0 tiene lugar en 2004 y hoy en día se lo hace anualmente en San Francisco, Estados Unidos.

Las aplicaciones web están relacionadas con muchos conceptos como: la web es una plataforma, la información es lo que le da vida al internet, etc, (Henst, 2012).

Back-End

El back-end es la manera que un sistema web hace las cosas detrás de un sitio web como: consultas a la base de datos, la recopilación de datos y procesamiento de la información. Toma la información y la envía al usuario, se encarga de la conexión con el servidor. Se puede desarrollar en varios lenguajes de programación como: PHP, Ruby, Python, JavaScript, SQL, MongoDB, MySQL, etc, (Guevara, 2016)..

API

Proviene de las siglas inglesas Aplicación Programming Interface que su traducción al español es: Interfaz de Programación de Aplicaciones. Este concepto permite saber los procesos, funciones y métodos que brinda una biblioteca de programación, para que sea utilizada por otro programador. Es un código el cual provee de aplicaciones para mantener comunicación entre sí, con reglas que permiten la interacción del programa, disminuyen significativamente el trabajo de un desarrollador. Un API establece la comunicación con la base de datos y su propio protocolo de comunicaciones. Es de fácil integración con un FrontEnd, (Porto, 2017).

1.2.6. JavaScript

Fue creado por el equipo Pluralsight, es un lenguaje de programación que entienden los navegadores, capaz de aportar muchas soluciones eficaces en muchos ámbitos de la tecnología, destinado al desarrollo de FrontEnd y aplicaciones web modernas, se lo considera un lenguaje universal puesto que, es el mas utilizado en la actualidad en su ámbito.

Algunas librerías como jQuery o React o frameworks como Angular, Vue o Ionic están basados en JavaScript, (Dusseldorf, 1997).

1.2.7. Front End

Es todo en cuanto el usuario puede interactuar y lo que ve en un navegador. Para realizar un FrontEnd dinámico y con buena impresión se puede realizar con HTML5, CSS3 y JavaScript, (Guevara, 2016).

HTML5

Lenguaje de etiquetas que aparece a partir de su versión anterior de HTML 4.0 y xhtml, por sus siglas significa lenguaje de marcado de hipertexto y hace referencia a la estructura en una página web, tiene estándares establecidos de estructura, html5 tiene funcionalidades que eran limitadas en HTML convencional.

(Van Lancker, 2012) afirma: “Las novedades de Html5 y sobre todo del trinomio conceptual ´Html5 +CSS3 y aplicaciones JavaScript son numerosas”.

CSS3

Css3 provee junto con html5 estilos como tamaño, color, fondo a los elementos de un documento web, (Gauchat, 2012).

Capitulo 2: Situación actual de la empresa

La situación actual de una empresa provee un entendimiento del entorno de acción, donde muestra sus fortalezas y debilidades. Mediante esto es posible que una empresa conozca las mejoras que se puedan adoptar para mejorar cada día.

Con el proceso de conocer la situación actual de esta empresa se obtendrá los pasos claros que se debe seguir para lograr exitosamente el desarrollo del sistema deseado.

En el presente capitulo se analizará la situación actual de la empresa COTEDEM (Consultores Tecnológicos y Desarrollo Empresarial CIA. LTDA), se detallará su historia visión, misión y sus actividades principales de negocio. De esta manera, se pretende que el sistema web que se desarrollará, facilite una de las actividades de la empresa.

2.1. Historia de la empresa

La empresa COTEDEM nace un ocho (08) de abril del dos mil catorce (2014) en la ciudad San Francisco de Quito ante la notaria tercera donde comparecieron los señores Darío Genaro Urvina López y Patricia Alexandra Maya Dávila, donde por sus propios y personales derechos decidieron crear esta la compañía de responsabilidad limitada la cual se rige por la Ley de Compañías y por el Estatuto Social.

La compañía tiene por objeto social brindar prestación de servicios de diseño, análisis, desarrollo, implementación y mantenimiento de software, adaptación, representación, arrendamiento, instalación, distribución, compra, venta y comercialización de equipos tecnológicos de información, software y hardware e investigación tecnológica en sistemas o proyectos empresariales y de desarrollo de tecnología informática, así como también actividades de importación y exportación de accesorios de tecnologías de información. La compañía brinda asesoría y consultoría técnica e integral de Software y/o Hardware, en los ámbitos empresariales, productivos, tecnológicos en investigación y estudios socioeconómicos, de organización y de mercado; supervisión y evaluación de proyectos sistemáticos, cooperación nacional e internacional, para el desarrollo de empresas, microempresas, pequeñas y medianas empresas, corporaciones y fundaciones, en general entidades públicas y/o privadas, nacionales y/o extranjeras en las áreas económica, estratégica, marketing, comunicación, operativo y tecnológico, valoración de empresas estructuración de planes de negocio y procesos a través de gestión de la innovación, vigilancia tecnológica y asesoría en

explotación de patentes, promoción y mercadeo de productos y servicios tecnológicos y sistemáticos en desarrollo multimedia. Los servicios que brinda la compañía están encaminados a lograr el fortalecimiento de las capacidades institucionales o empresariales, a través de la realización de actividades de capacitación entrenamiento y formación en modalidades presenciales y en e-learning es decir formación virtual en temas de tecnologías de información y desarrollo organizacional, desarrollo en sistemas y/o programas informáticos; las actividades de capacitación no se refieren a las de nivel superior, reguladas por la Ley Orgánica de Educación Superior. Se dedica también de manera habitual, dentro del giro ordinario de sus negocios, a realizar actividades de asesoría e implementación de dispositivos de seguridad, gestión y manejo de medios de seguridad sistemática; también la prestación de servicios de publicidad, marketing, creatividad, diseño en todas sus áreas y por cualquier medio que fuere requerido, impresión, organización de eventos y asesoría en mercadeo, imagen, relaciones públicas y comerciales. En general, la compañía realiza toda clase de actos, contratos y operaciones lícitas que fueren necesarias, relacionados con su objeto social con todos los medios lícitos que permitan conseguirlo. La Compañía, para cumplir con su objeto social, observa y cumple con las disposiciones de la Ley de Compañías y demás Reglamentos.

2.2. Enfoque de la empresa.

COTEDEM es una empresa que se caracteriza por brindar soluciones tecnológicas integrales, la meta principal es optimizar la competitividad a las empresas que brinda su servicio. En la Ilustración 2, se presenta el logo de la empresa Consultores Tecnológicos y Desarrollo Empresarial.



Fuente: http://www.cotedem.com

Ilustración 2 Logo Cotedem

(COTEDEM, 2020)

COTEDEM proporciona soluciones IT para el comercio, consultoría y otras áreas afines de negocio, mediante la metodología sistemática y estandarizada de procesos y tecnologías de última generación. Uno de los enfoques principales de la empresa es distribuir gran variedad de productos tecnológicos y proveer los servicios

necesarios para satisfacer las demandas actuales en tanto a tecnología que necesitan las empresas. La empresa para brindar el mejor servicio a los clientes mediante capacitaciones hacia la mejora continua capacitando a su personal de forma permanente, a través de programas de educación y formación.

La misión de la empresa, (COTEDEM, 2020) afirma que: “Proporciona soluciones tecnológicas integrales, a medida de las necesidades de las empresas que confíen en nosotros. Con el objetivo de optimizar su competitividad.”

La visión de la empresa, (COTEDEM, 2020) afirma que: “Convertirnos en aliados estratégicos de nuestros clientes, para solucionar sus problemas y satisfacer sus necesidades, siempre de forma eficaz y transparente. Ser una empresa de referencia a nivel de Ecuador y América Latina, en soluciones tecnológicas integrales.

Mediante requerimientos que tiene COTEDEM por parte de sus clientes para tener identificaciones personales, surge la idea de crear un módulo dentro de la empresa la cual servirá para crear identificadores para las empresas que lo requieran, ya que, en un principio cuando nació esta necesidad por parte de la empresa COTEDEM la realizó de una manera no sistematizada. Primero, se identificó el requerimiento real de la empresa que solicitaba este servicio, con lo cual se concluyó que requerían identificadores a manera de carné para poder identificar al personal de la empresa cliente ya que, esta empresa trabaja en un edificio donde funcionan más empresas y se podía fácilmente confundir al personal de la empresa con personas ajenas a la misma.

Con esto COTEDEM se acercó a la empresa para conocer todos los detalles que ellos requerían en las identificaciones, luego procedió a la toma de datos de todo el personal, con los datos guardados COTEDEM contrato una empresa para la elaboración e impresión de los identificadores con lo cual primero se obtuvo muchos problemas, entre los más importantes que el precio fue muy alto, en total al sub contratar otra empresa para la impresión de las identificaciones, se tuvo retrasos en las entregas de los carnés a la empresa cliente, también se tuvo un problema de impresiones dobles, datos mal ingresados por la empresa subcontratada y de personas que no habían asistido en el día de toma de datos por lo cual esas personas lograron obtener su identificación después de mucho tiempo. Con todo esto COTEDEM empieza la idea de tener un sistema propio de carnetización, para lo cual

también la empresa decidió adquirir impresoras especializadas para la impresión de identificaciones de calidad y con estándares de seguridad.

El sistema de carnetización que se planteó tener la empresa consiste en un sistema web en el cual principalmente servirá para instituciones educativas o cualquier tipo de organizaciones. El funcionamiento para cualquiera de estas 2 tipos de organizaciones será similar por lo cual cada organización, desde ahora en adelante, se podrá registrar, registrar a cada uno de sus integrantes, en el caso de ser una institución educativa de secundaria o primeria se facilitara para que los representantes de los alumnos se puedan registrar y con esto controlar que los datos de sus hijos estén correctos, después cada institución puede registrar su parametrización para el diseño del carné, finalmente cada cliente puede solicitar una o más veces la impresión de uno o varios carnés.

2.3. Situación actual de la empresa.

La situación actual de COTEDEM es brindar una gran variedad de servicios que son de ayuda y utilidad para una empresa cliente, con la experticia que tiene en brindar soluciones tecnológicas sus principales servicios son:

1. ***Servicios de Cableado Estructurado.***
   Mediante el cableado estructurado se crea una infraestructura de red inteligente mediante redes de fibra óptica y conectividad universal (UCG), que pueden respaldar para futuras tecnologías. El cableado estructurado combina cables de par trenzado protegidos, fibra óptica o cable coaxial que tiene un mando de control desde un cuarto de telecomunicaciones.
2. ***Servicios de Consultoría y Auditoría Informática.***
   en la actualidad toda empresa requiere de una consultoría y auditora informática para conocer los detalles de infraestructura de TI y como los empleados utilizan estos recursos, también se podrá conocer soluciones para las áreas problemáticas de la infraestructura de TI, todo esto con el fin de optimizar recursos.
3. ***Consultoría para Servidores***
   Mediante la asesoría técnica especializada para servicios y soluciones en servidores se puede conocer los detalles y el estado de los servidores físicos y virtuales con el fin de descubrir donde se pueden encontrar problemas y dar

soluciones a las mismas, para asegurar que estos estén funcionando de manera óptima.

4. ***Sistema de Digitalización de Archivo Físico.***

   En la actualidad ya no cabe que una empresa solo tenga respaldos de toda su documentación de forma física por lo cual COTEDEM ofrece el servicio de digitalización de archivos de toda la empresa encriptándola por medidas de seguridad en servidores físicos o virtuales.

5. ***Gestión de Departamentos de TICS.***

   Con la gestión de TICS se pueden tomar decisiones operativas dentro de una empresa para que la misma funcione de manera más eficaz.

6. ***Cámaras de Seguridad (CCTV e IP).***

   Un módulo fuerte de la empresa donde es una de las especialidades es brindar soluciones de videovigilancia para ser monitoreadas desde cualquier parte del mundo mediante aplicaciones móviles.

7. ***Servicios de Manteamiento Preventivo y Correctivo de Equipos de cómputo.***

   El no hacer un mantenimiento preventivo y correctivo de equipos de cómputo e informáticos en toda empresa puede incurrir en pérdidas económicas al siempre adquirir equipos nuevos en vez de darles mantenimiento preventivo y correctivo, es por esto por lo que, COTEDEM da mantenimientos desde la infraestructura tecnología hasta los equipos informáticos más pequeños que pueda poseer una empresa para maximizar el tiempo de productividad.

8. ***Soluciones WI-Fi.***

   COTEDEM implementa infraestructura de conectividad WIFI a todo tipo de empresas con equipos capaces de soportar aplicaciones de datos, voz y video. Incluye soluciones Wifi incluyen enlaces inalámbricos, siempre con el objetivo de anular las distancias en las comunicaciones entre diferentes hosts.

9. ***Administración de redes.***

   La disponibilidad de las redes es un punto clave para el negocio de una empresa, por eso dar soluciones completas ayudan a la administración de una red y mantenerla funcionando, evitando interrupciones largas en los servicios. Por eso la empresa realiza labores preventivas de los sistemas, actualizando periódicamente e instalación de parches de seguridad evitando posibles vulnerabilidades de los sistemas. También realiza procesos de copias de

seguridad tanto locales como remotas con esto una empresa evita la perdida de datos en caso de catástrofe.

10. ***Servicios de impresión de credenciales en alta seguridad.***

    Como se mencionó anteriormente este módulo de la empresa surge bajo las peticiones de varios clientes para la impresión de credenciales, al nacer esta necesidad la empresa requería un desarrollador para lograr desarrollar un sistema web que cubra las necesidades que la empresa requiere, mediante el desarrollo de este proyecto de titulación se podrá cubrir en la totalidad los requerimientos, para que el servicio de impresión funcione en su totalidad, ya que, la empresa cuenta con impresoras de calidad para la impresión de las identificaciones.

11. ***Desarrollo de software a medida***

    En la actualidad para mantener una ventaja competitiva entre empresas es necesario tener aplicaciones que optimicen tiempo y recursos en todo sentido dentro de una empresa es por esto que el desarrollo de software a medida es aquel que se diseña a medida del usuario y de su forma de trabajo. Es decir, busca complacer todas las necesidades y adaptarse de manera posible a lo que una empresa necesita. La principal ventaja es la personalización de los requerimientos de cada cliente para aprovechar el 100% del potencial que brinda un sistema, consiguiendo la satisfacción del usuario.

Entre los principales clientes a los cuales ha brindado servicios o productos tecnológicos tiene:

- Huawei
- Agencia Nacional de Transito
- BMI
- Clínica de urología.
- Edimca.
- Fideval.
- Gloapsa
- Y entre otros clientes particulares que se han beneficiado con la experticia y servicios de COTEDEM.

COTEDEM con el fin de satisfacer de mejor manera las necesidades de los clientes siempre ha trabajado con las mejores marcas a nivel nacional e internacional.

Entre ellas se encuentran:

- Kodak alaris.
- Adobe.
- APC.
- Arcserve.
- Audiocades.
- Autodesk.
- Cloud.
- Corel.
- Epson.
- Hewlet Packard.
- IBM.
- Jabra.
- Kaspersky.
- Bititan.
- Xerox.
- Lenovo.
- MakerBot.
- McAfee.
- Microsoft.
- Oki.
- Polycam.
- Redhat.
- Sophos.
- Symantec.
- Vetiv.
- ViewSonic.
- Wacom.
- Oracle.

Con todas estas marcas la empresa ha logrado culminar de manera satisfactoria la mayoría de todos los proyectos que ha tenido en frente, y actualmente con el nuevo proyecto de generación e impresión de credenciales para lo cual se desarrolla este proyecto de titulación.

Capitulo 3: Análisis y Diseño

El desarrollo del sistema web propuesto, al ser una herramienta que pretende ser utilizada en la empresa, para solucionar la problemática propuesta, tiene que ser desarrollado de manera disciplinada, mediante un análisis profundo de todos los requerimientos para su posterior diseño que proporcionará los respectivos cimientos bajo los cuales se desarrollará el sistema web. De esta manera, todas las etapas del sistema se vuelven más formales y facilita el desarrollo del sistema.

Para que el sistema propuesto sea culminado de manera exitosa, es fundamental realizar el proceso de análisis, para tener una vista completa y plena compresión de los sistemas requerimientos. Con el análisis culminado de manera efectiva se puede realizar un diseño completo y real en cuanto al sistema para traducir los requerimientos en una representación de software. En este capítulo, se detalla el proceso de ingeniería de software, análisis y diseño para realizar el sistema web de manera exitosa.

3.1. Análisis del sistema.

Para hacer un análisis completo del sistema web, se inicia con un levantamiento de requerimientos del sistema que se desarrollará, mediante charlas con el gerente de la empresa. Esto incluye la identificación de los procesos que pueda realizar el sistema, de manera que se logre recolectar toda la información en cuanto a las necesidades que las organizaciones o instituciones educativas requieran. Los usuarios de este sistema web son el gerente de la empresa como administrador del sistema, representantes de cada organización o institución educativa y usuarios de la empresa o institución educativa que soliciten este servicio.

Finalmente, con toda la información levantada, se realizó el análisis para llevar a cabo el respectivo diseño del sistema web. Esto incluye los diagramas de casos de usos para los procesos principales para el entendimiento del funcionamiento del sistema a desarrollar.

Después de todo el levantamiento de información se concluye que el sistema será capaz de:

- <u>El administrador del sitio puede:</u>
    - Administrar las instituciones educativas y organizaciones.

- Administrar los cursos de las instituciones educativas.
- Administrar los estudiantes de las instituciones educativas.
- Administrar las áreas de trabajo de las organizaciones
- Administrar los usuarios de las organizaciones
- Administrar los fondos de las instituciones.
- Administrar los logos de las instituciones.
- Administrar las fotos de los usuarios de las organizaciones.
- Administrar las fotos de los estudiantes de las instituciones educativas.
- Administrar los usuarios del sistema.
- Administrar los roles de los usuarios del sistema.

Administrar refiere a las acciones de un CRUD que puede realizar el administrador del sitio dentro del sistema esto es: realizar las acciones de crear, editar, actualizar y eliminar registros.

- Un represéntate de una institución educativa puede:
  - Ver la lista de estudiantes de la institución a la que pertenece
  - Ver su perfil de usuario.
  - Ver los cursos de la institución.
  - Editar su usuario.
  - Solicitar la impresión de su carné.
  - Solicitar impresión de todos los identificaciones de todos los estudiantes de la institución.
- Para un represéntate de una organización puede:
  - Ver la lista de usuarios de la organización a la que pertenece
  - Ver su perfil de usuario.
  - Editar su usuario.
  - Solicitar la impresión de su carné.
  - Solicitar impresión de todos los carnés de todos los usuarios de la organización.
- Para un usuario o estudiante registrado en el sitio puede:
  - Visualizar su información personal.
  - Editar su información personal.
  - Solicitar la impresión de su carné.

Con toda la información adquirida en la empresa se definen los siguientes requerimientos funcionales para el sistema que se desarrollará para la empresa.

3.1.1. Requerimientos Funcionales

Se definió los siguientes requerimientos funcionales para el sistema:

F1. Administración de Instituciones.

F2. Administración de Cursos.

F3. Administración de Estudiantes.

F4. Administración de Áreas de Trabajo

F5. Administración de Usuarios.

F6. Administración de Fondos.

F7. Administración de Logos.

F9. Administración de Fotos de Usuarios.

F10. Administración de Fotos de Estudiantes.

F11. Administración de Usuarios del Sistema.

F12. Administración de Carnets.

F12. Administración de Solicitudes de Impresión.

F13. Administración de Solicitudes Aprobadas.

3.1.2. Requerimientos No Funcionales

Se definió los siguientes requerimientos no funcionales para el sistema:

1. El sistema en su totalidad será desarrollado y codificado en el lenguaje PHP.
2. El sistema será desarrollado en una arquitectura de 2 capas: BackEnd y FrontEnd.
3. El BackEnd será desarrollado bajo los estándares de desarrollo que provee Laravel el cual será el framework de desarrollo para este.
4. El FrontEnd será desarrollado mediante HTML5, CSS3, JavaScript

5. El sistema brindará seguridad para el manejo de la información, requiriendo una validación de usuario y contraseña cada vez que se desee ingresar al mismo.

3.1.3. Casos de Uso

Para representar todos los requisitos funcionales del sistema se utilizó UML el cual es un Lenguaje de Modelado Unificado, este, nos permite representar los requerimientos mediante diagramas. UML es un lenguaje más común en la actualidad, entre las características principales tenemos que: es aprobado por la Organización Internacional de Normalización, con esto nos permite hacer cualquier tipo de diagrama para que el mismo sea interpretado por cualquier persona que conozca del estándar.

3.1.3.1 Diagrama General

Un diagrama de caso de uso general explica mediante la modelación la funcionalidad a la manera más general del sistema propuesto y el acceso por parte de los usuarios a las distintas funcionalidades. Los casos de uso so n funciones para el sistema para sus usuarios. En las ilustraciones 3 y 4, se presenta el diagrama general de casos de uso del sistema,

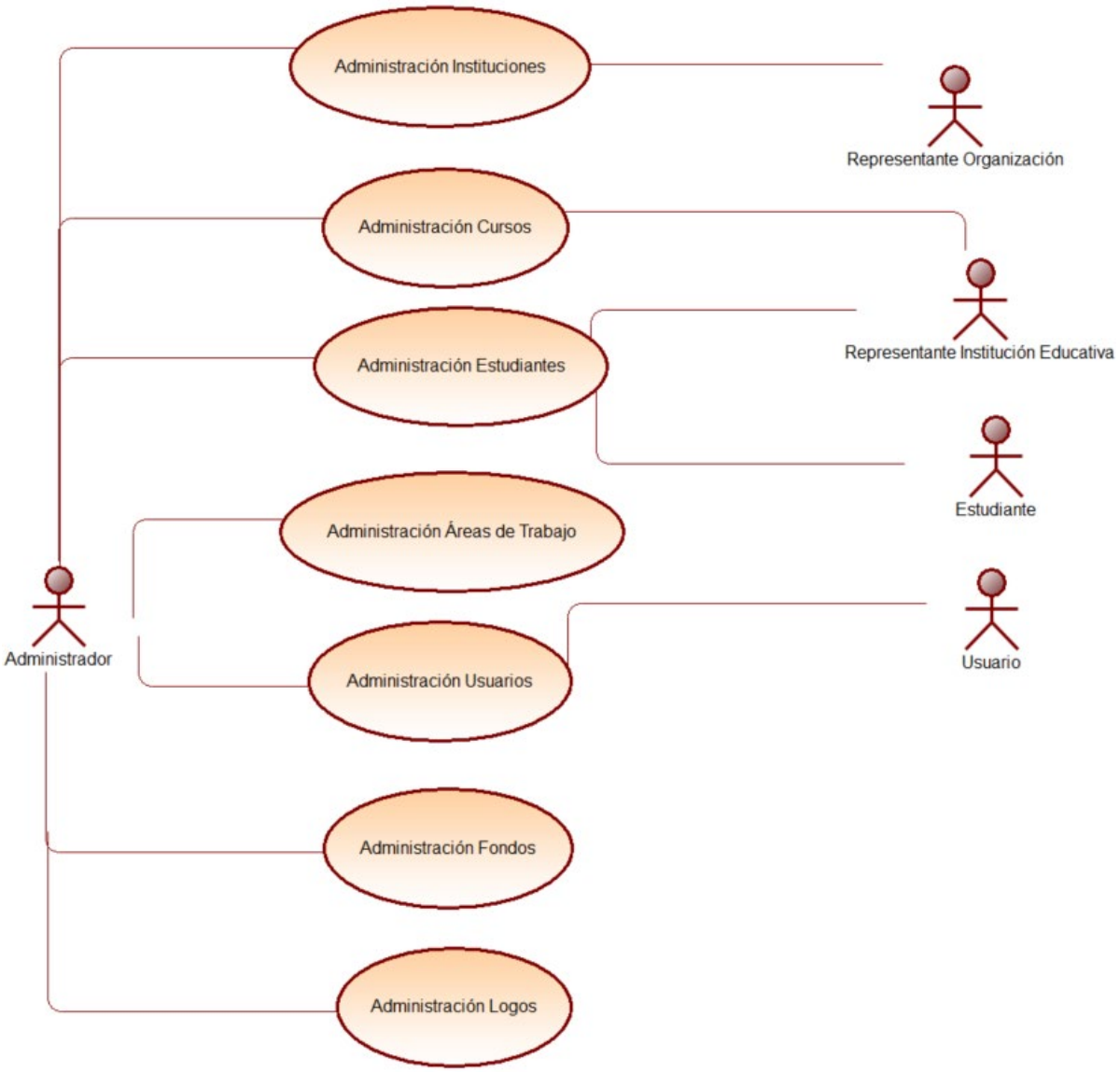


Ilustración 3 Diagrama General

(Urvina Cordova, 2020)

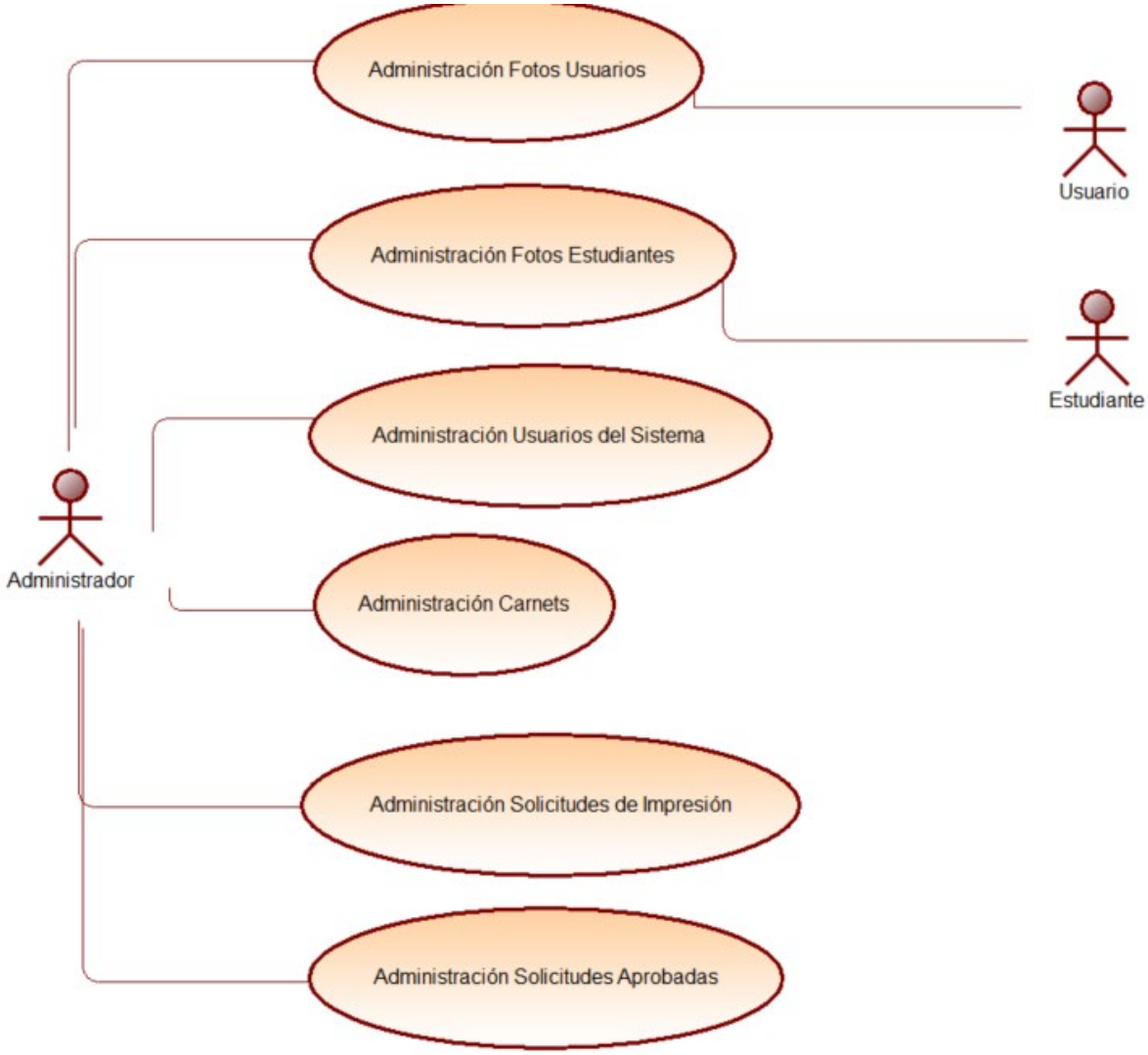


Ilustración 4 Diagrama General

(Urvina Cordova, 2020).

3.2 Diseño de la Base de Datos

La base de datos se diseñó mediante reuniones constantes con el dueño de la empresa se procedió a realizar el modelo conceptual para conseguir finalmente el modelo entidad relación. La estructura de la base de datos es el diseño físico que tendrán los datos dentro del sistema, se representa mediante tablas relacionas que identifican los atributos que vinculan los registros en cada tabla.

a. Modelo Conceptual

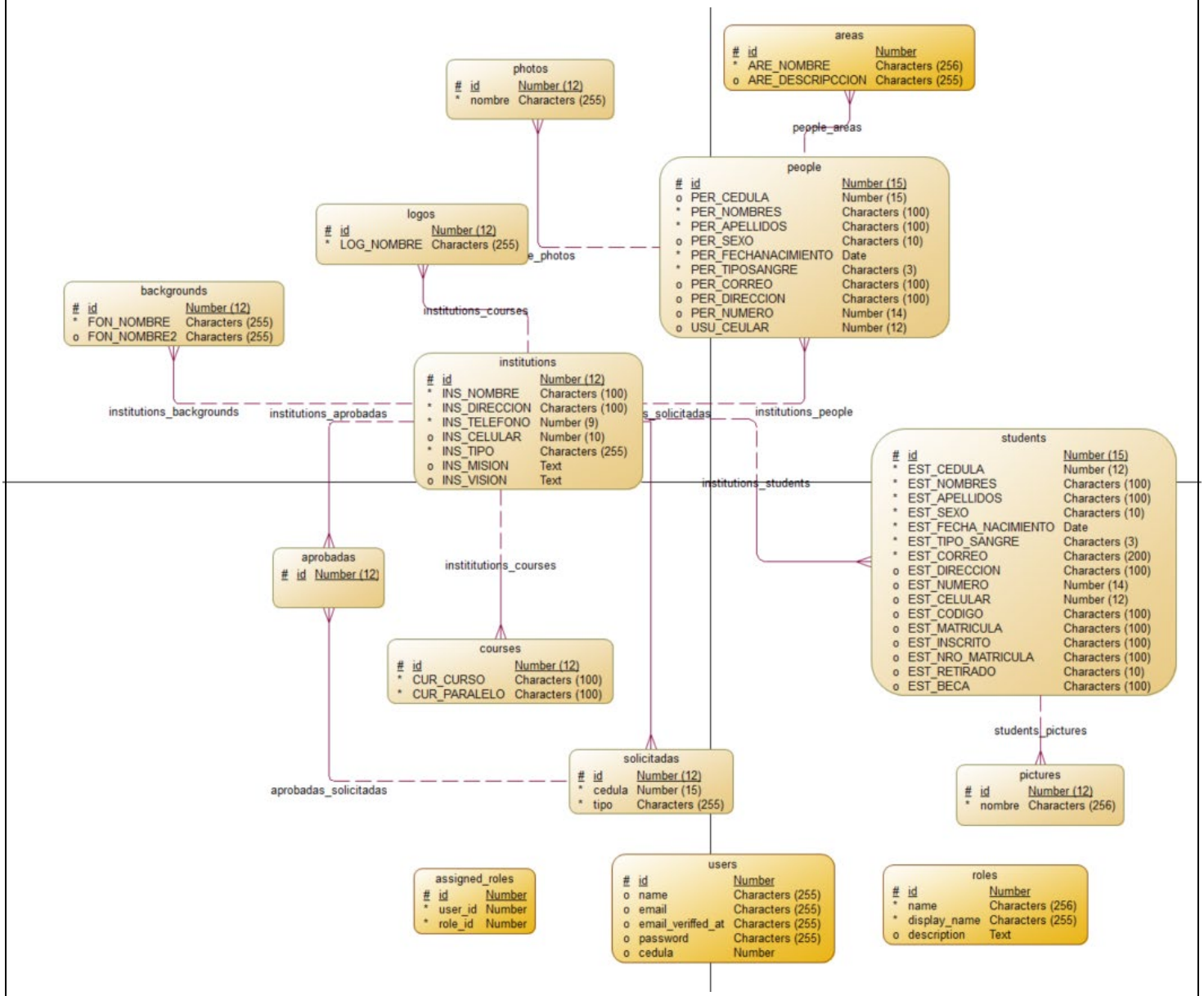


Ilustración 5 Modelo Conceptual

(Urvina Cordova, 2020)

Diagrama Lógico

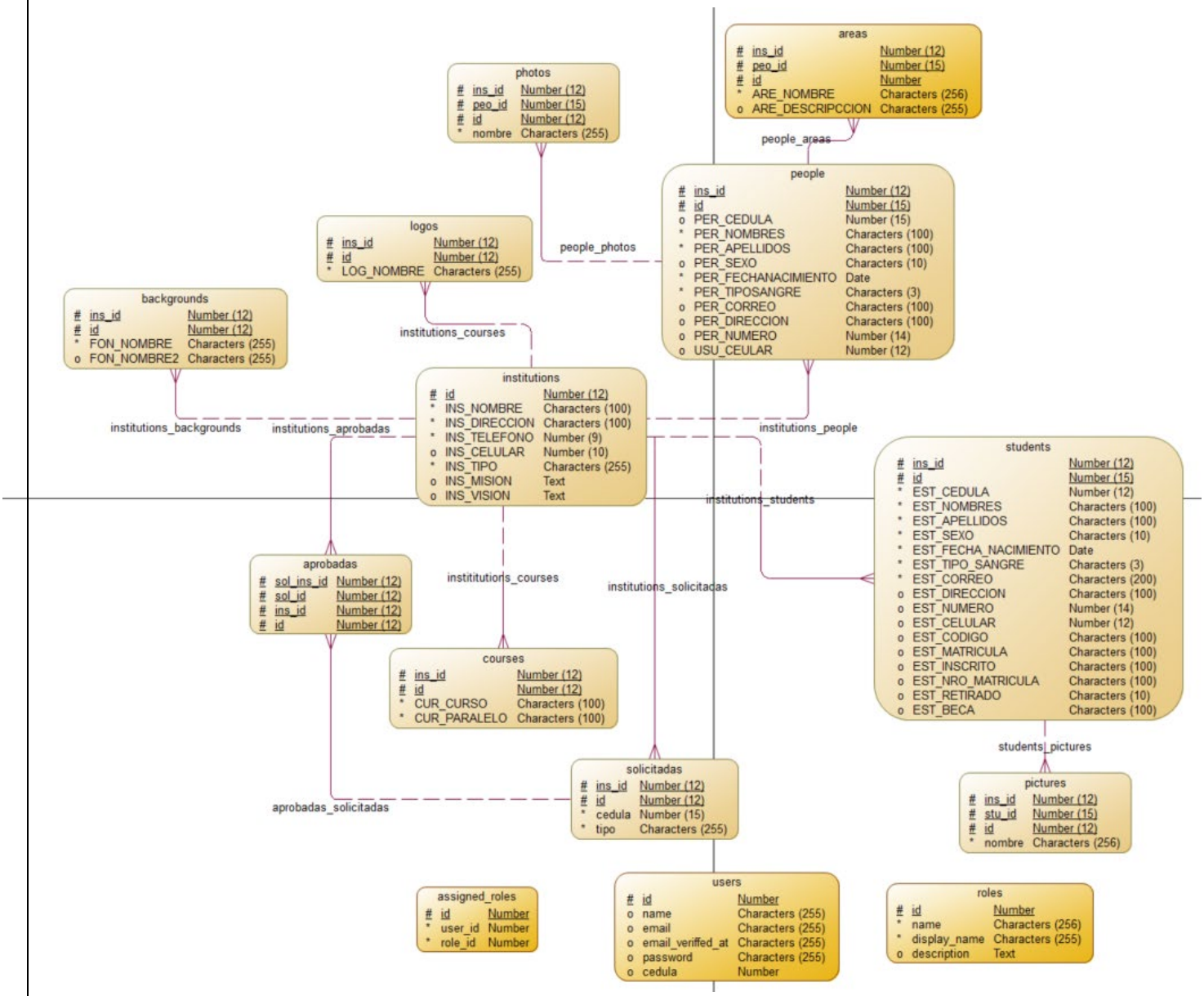


Ilustración 6 Diagrama Lógico

(Urvina Cordova, 2020)

b. Diagrama Físico

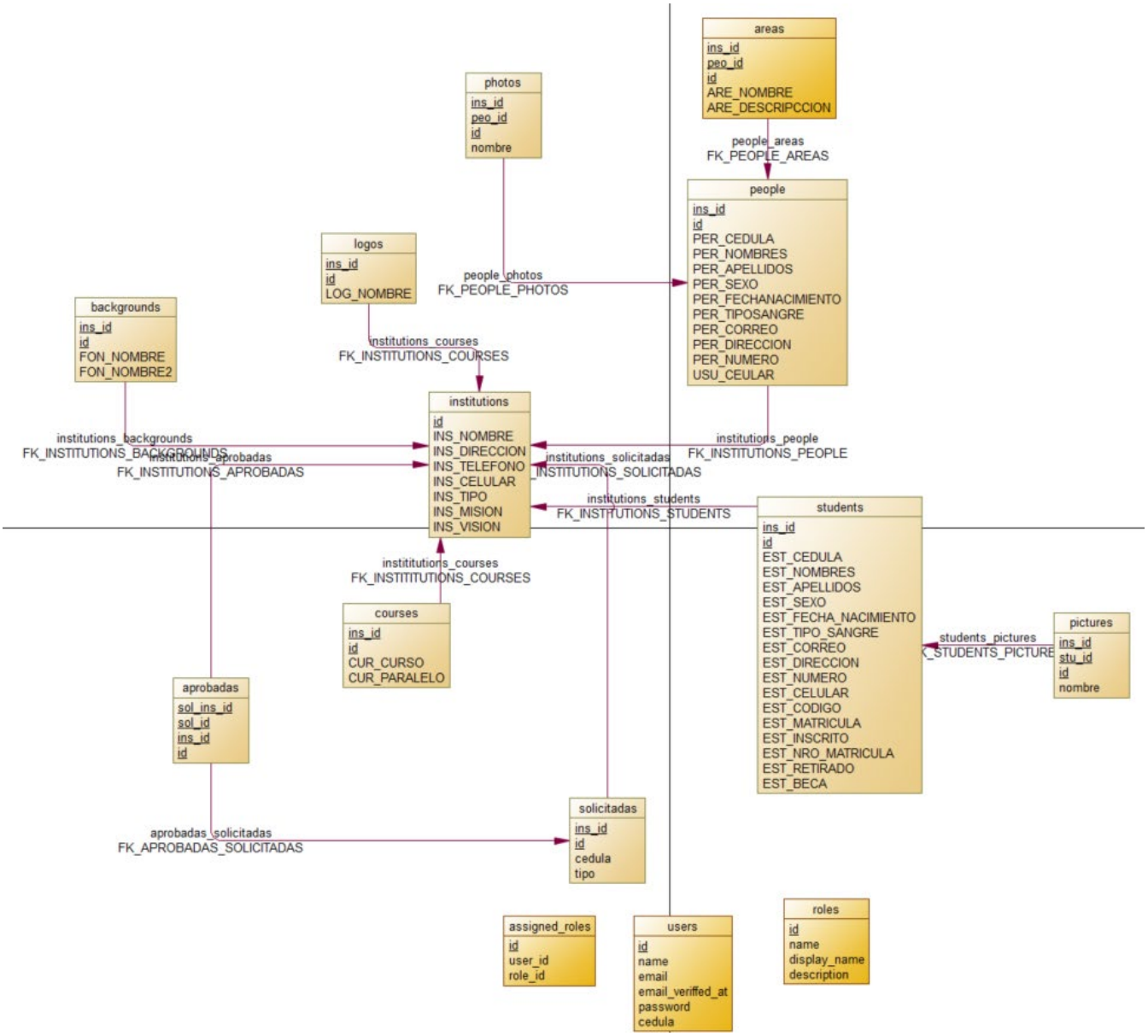


Ilustración 7Diagrama Físico

(Urvina Cordova, 2020)

c. Diagrama Entidad Relación

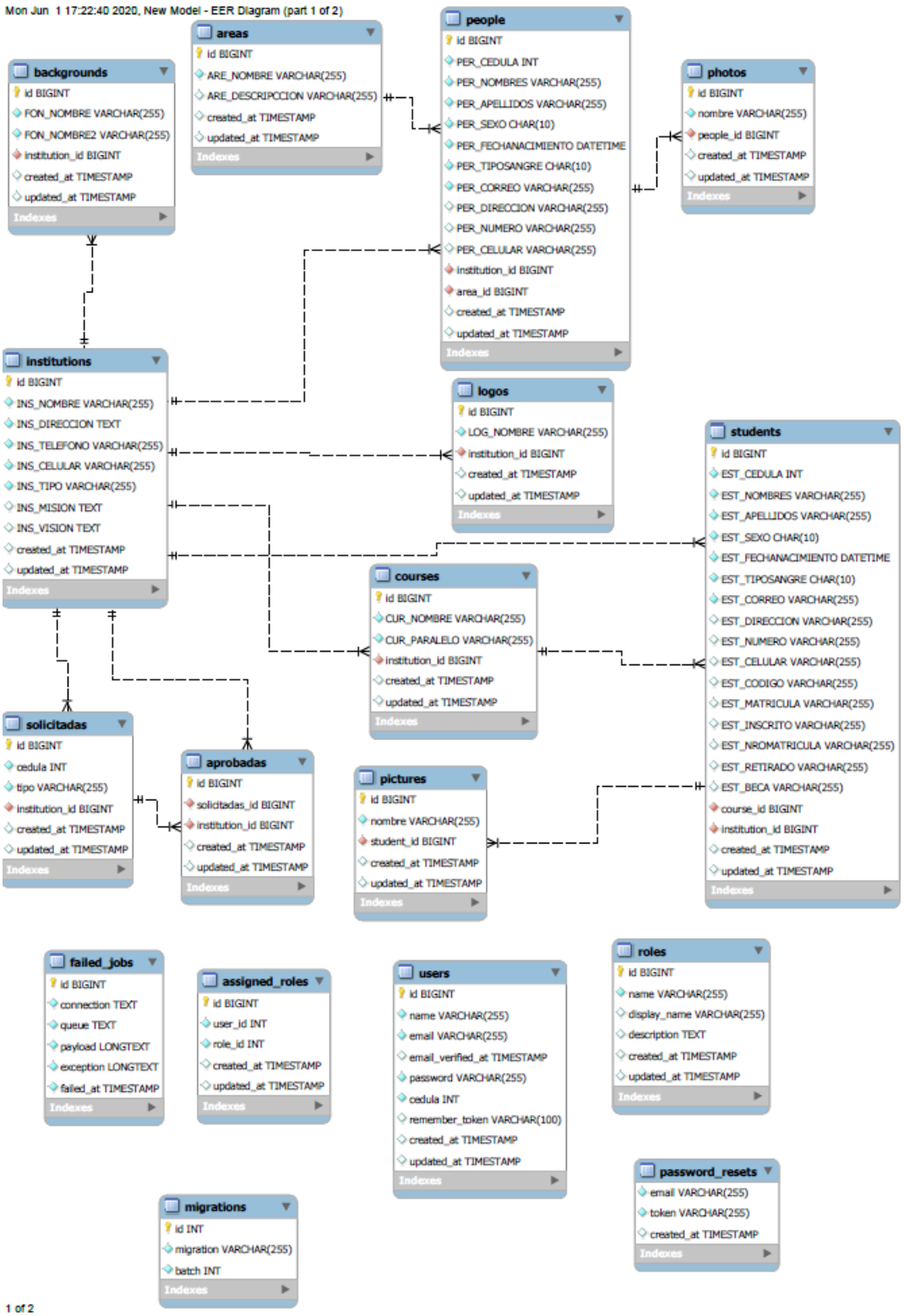


Ilustración 8 Diagrama Entidad Relación

(Urvina Cordova, 2020)

El script de la base de datos puede ser consultado en el Anexo 1: Script Base De Datos.

3.3 Diseño del BackEnd.

Se decidió usar una arquitectura REST, el cual permite crear un servicio y aplicación que puede ser consumida por cualquier cliente que permita HTTP, al ser un tipo de arquitectura de desarrollo web que se apoya en el estándar HTTP, esta arquitectura es la ideal para el sistema web a desarrollar, porque se desarrollaran una serie de recursos que serán aprovechados por el frontend (portal web), y pensando en la evolución del sistema para que este pueda funcionar en un aplicativo móvil que se podría desarrollar a futuro dentro de la empresa, se usa REST porque nos proporciona ciertas ventajas. Para entender de mejor manera las ventajas y el funcionamiento de un API REST se explicará mediante la ilustración 9.

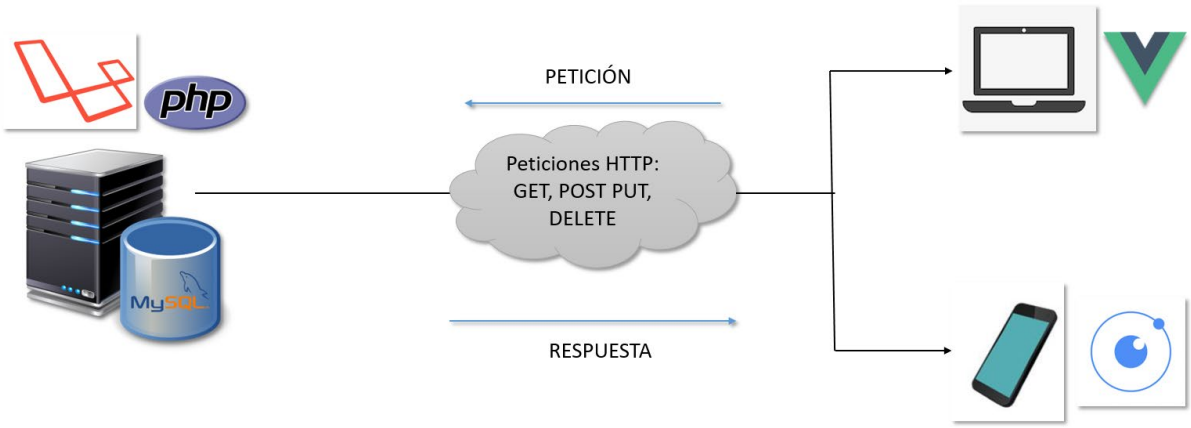


Ilustración 9 Funcionamiento API Rest

(Urvina Cordova, 2020)

El cliente en este caso el frontend realiza una solicitud hacia el servidor, la cual pasa por JavaScript. El lenguaje del lado del cliente el cual realiza la solicitud de un recurso que este disponga.

El servidor y el cliente web realizan la comunicación en un formato de información como JSON, XML, entre otros. En sí, el cliente no recibe un HTML para renderizar, si no recibe los datos generados y guardados que se han generado como respuesta a una solicitud, es decir, el servidor solo genera los datos para enviar al cliente y que este los procesé.

Con una visión del funcionamiento del desarrollo basado en REST las ventajas son las siguientes:

1. **Se separa el cliente y el servidor**

   Como son sistemas independientes, esto quiere decir que el intercambio de datos se produce mediante JSON o cualquier otro tipo de formato de intercambio, el sistema se desarrollara de manera autónoma, en otras palabras, es irrelevante para el cliente lo que el API vaya a hacer con la información que el frontend brinda. Es decir, se puede realizar muchos más frontend aparte del que se va a diseñar y realizar en este trabajo con un único backend, como se mencionó antes a futuro se piensa hacer una aplicación móvil que consuma los servicios de este mismo API, con esto el uso del API será único y no se tendrá que volver hacer todo un programa para la aplicación móvil.

2. **Independencia en la aplicación de tecnologías y lenguajes de desarrollo de software**

   Como el frontend y el backend son sistemas independientes, estos se pueden desarrollar en cualquier framework de programación en este caso con el motor de plantillas de laravel y Laravel con PHP respectivamente, de igual manera para desarrollar otro frontend, no es indispensable que se desarrolle en la misma tecnología que se usó en este trabajo.

3. **Fiabilidad, escalabilidad, flexibilidad**

   La fiabilidad en un API refiere solo al nexo que se tiene con el cliente, este debe ser correcto, se puede realizar cambios en el servidor, lenguajes o bases de datos, mientras que, la información que se mande al cliente sea correcta todo funcionará de manera adecuada.

   La escalabilidad es el punto más importante en lo que refiere a un API REST ya que nos permite crecer en todo momento, el API puede versionarse tantas veces como se deseé, por otro lado, los frontend que se conecten al API pueden evolucionar las veces que se requiera mientras las peticiones al API sean las adecuadas.

4. **Mejor experiencia al usuario.**

A pesar de que esto depende en la gran mayoría de como este hecho el frontend, el desempeño del sistema completo es más rápido que uno tradicional. Al momento de hacer una solicitud al servidor por parte del cliente la respuesta es datos planos, que requiere de un determinado tiempo de transferencia, los cuales para su presentación dependen del diseño que tenga el frontend.

5. **REST requiere menos recursos del servidor.**
   El API no mantiene el estado con esto no requiere memoria ya que puede seguir atendiendo más peticiones por parte del cliente.
   De igual manera el API no escribe nada de HTML, con lo que se gana menos procesamiento en el servidor.

En conclusión, las ventajas que provee un API REST son muy favorables frente a sistemas web tradicionales, dado que un sistema así nos permite evolucionar sin tener que cambiar todo lo hecho anteriormente.

Para el sistema propuesto el backend será desarrollado en lenguaje PHP con la ayuda del framework Laravel.

De manera general la arquitectura que usa Laravel será la implementada en este sistema, el modelo MVC (Modelo, Vista, Controlador).

3.4 Diseño de FrontEnd (Interfaz Gráfica).

Para el diseño de la interfaz gráfica se lo realizó de manera separada, esto posteriormente en el desarrollo del BackEnd es de gran utilidad para tener separados todos los componentes, con esto se logra un fácil entendimiento del sistema y se evita tanto como en el FrontEnd y en el BackEnd el código desordenado (Espagueti).

La realización de las interfaces gráficas para el sistema se realizó con la ayuda de la herramienta Adobe Dreamweaver, la cual nos permite diseñar, editar y construir sitios web con los estándares deseados, la ventaja principal de este software de diseño es el poder de personalización y ampliación mediante la utilización de JavaScript, de igual manera soporta Html5, CSS.es de gran utilidad ya que, mediante se va haciendo la construcción de la plantilla se puede visualizar en tiempo real los cambios, controla la sintaxis de los archivos y la introspección de código, Dreamweaver CC es una de las herramientas más utilizadas para el diseño de FrontEnd por su versatilidad en el diseño y desarrollo, al tener archivos finalizados mediante FT, SFTP o WebDAV carga

los archivos directamente en un servidor deseado, pero para el desarrollo de este trabajo se utilizará el modelo MVC del BackEnd, (Microsoft, 2020).

a. Diseño Ingreso al Sistema

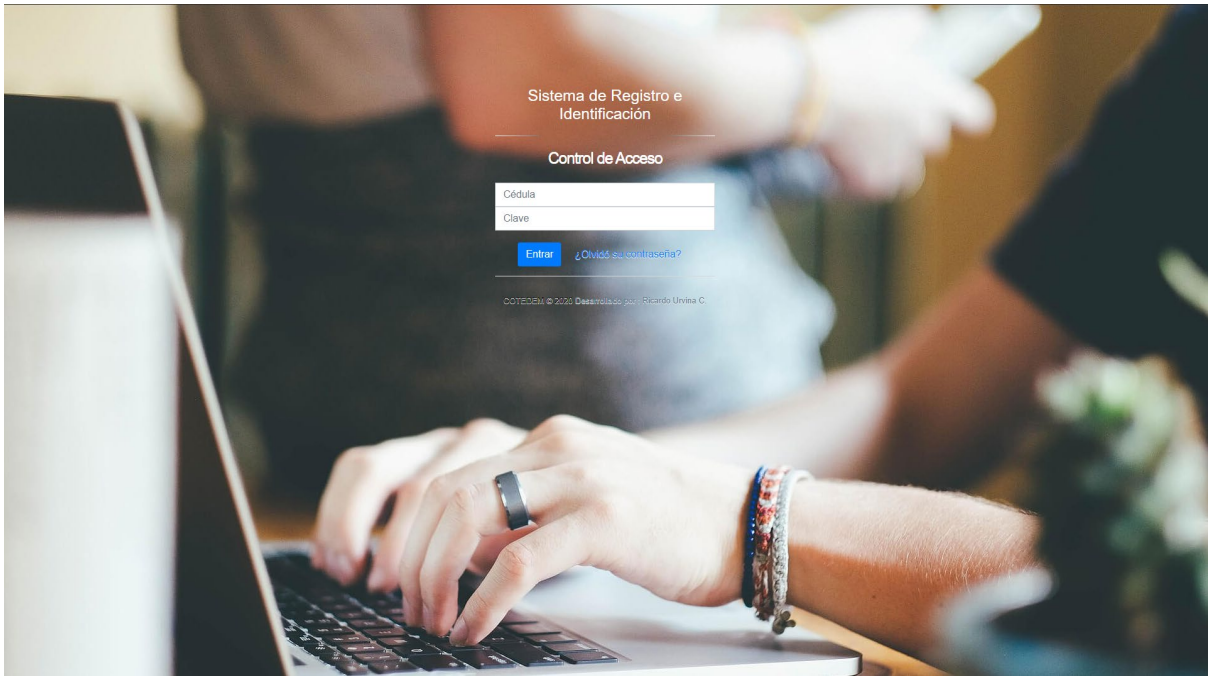


Ilustración 10 Ingreso al Sistema

(Urvina Cordova, 2020)

El usuario que desee ingresar al sistema debe iniciar sesión mediante su cédula y la clave generada por el administrador del sitio.

b. Diseño recuperación Contraseña

2. En el caso que un usuario no recuerde su contraseña debe en la pantalla de iniciar sesión dar clic sobre ¿Olvido su contraseña?, con la cual se mostrará la siguiente pantalla en la cual el usuario ingresa su cédula y solicita el cambio de contraseña con lo cual al administrador del sitio le llegará un correo informando que el usuario solicito el cambio de contraseña

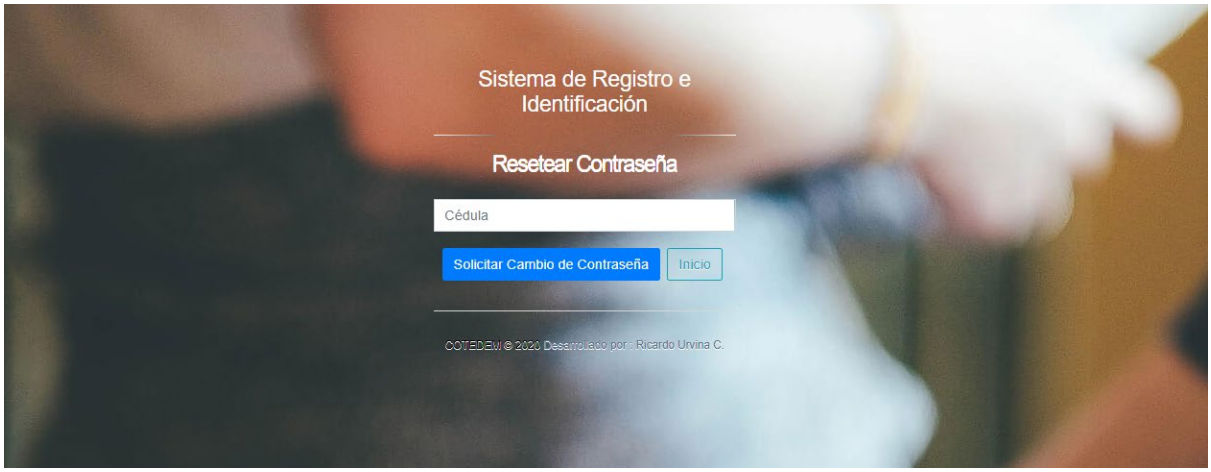


Ilustración 11 Recuperar Contraseña

(Urvina Cordova, 2020)

a. Diseño Tablero Principal (Administrador)

El administrador del sitio dentro del panel principal tiene acceso a todo el sistema.

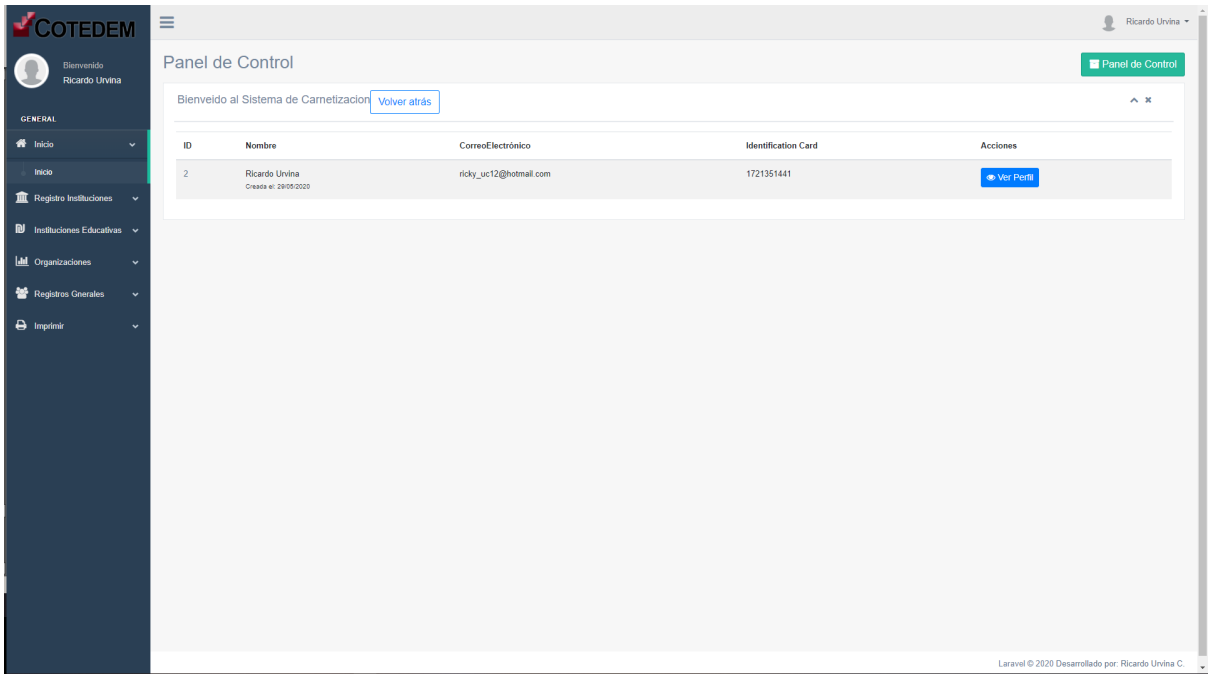


Ilustración 12 Tablero Principal (Administrador)

(Urvina Cordova, 2020)

b. Diseño Tablero Principal (Usuario/Estudiante)

Dentro de esta pantalla el usuario o estudiante únicamente podrá ver su perfil y editar sus datos o fotografía si fueran necesarios. El usuario estudiante puede solicitar la impresión de su carné dando clic sobre Solicitar Impresión.

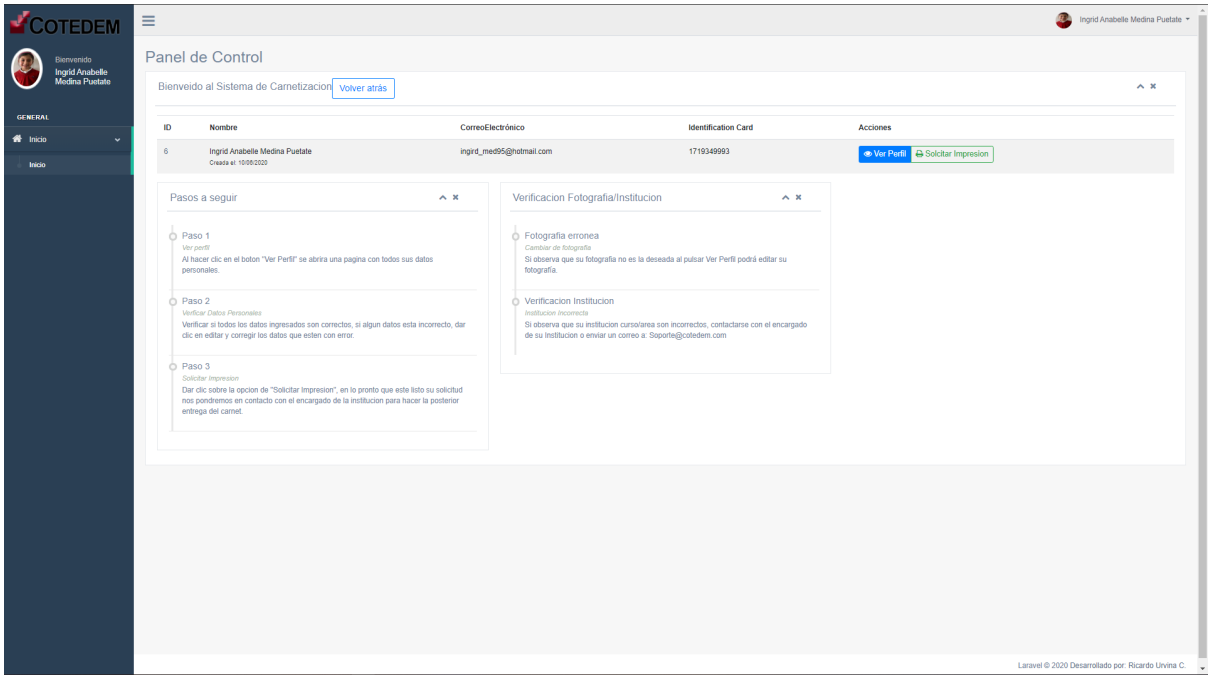


Ilustración 13 Tablero Principal (Usuario/Estudiante)

(Urvina Cordova, 2020)

c. Diseño Visualización de Listas de Registros

Esta pantalla muestra cómo es la visualización de los datos de estudiantes, usuarios, instituciones, cursos, áreas y de más modelos que necesiten mostrar listas de sus registros. En este caso en el listado de estudiantes el administrado puede hacer el CRUD de estudiantes y ver el carné de un estudiante deseado.

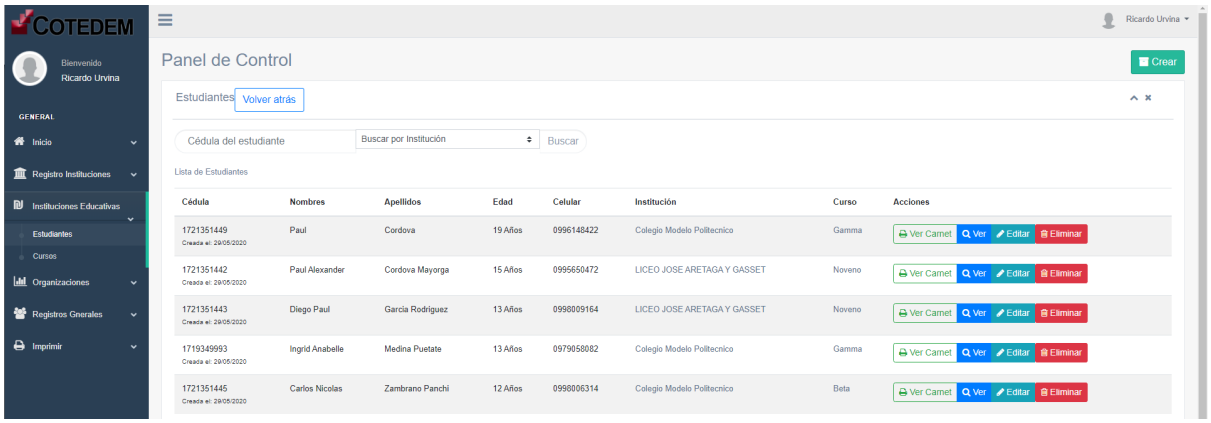


Ilustración 14 Visualización Listas de Registros

(Urvina Cordova, 2020)

d. Diseño Visualización Datos Usuario/Estudiante

Esta pantalla muestra la manera que se despliegan los datos de un usuario o estudiante.

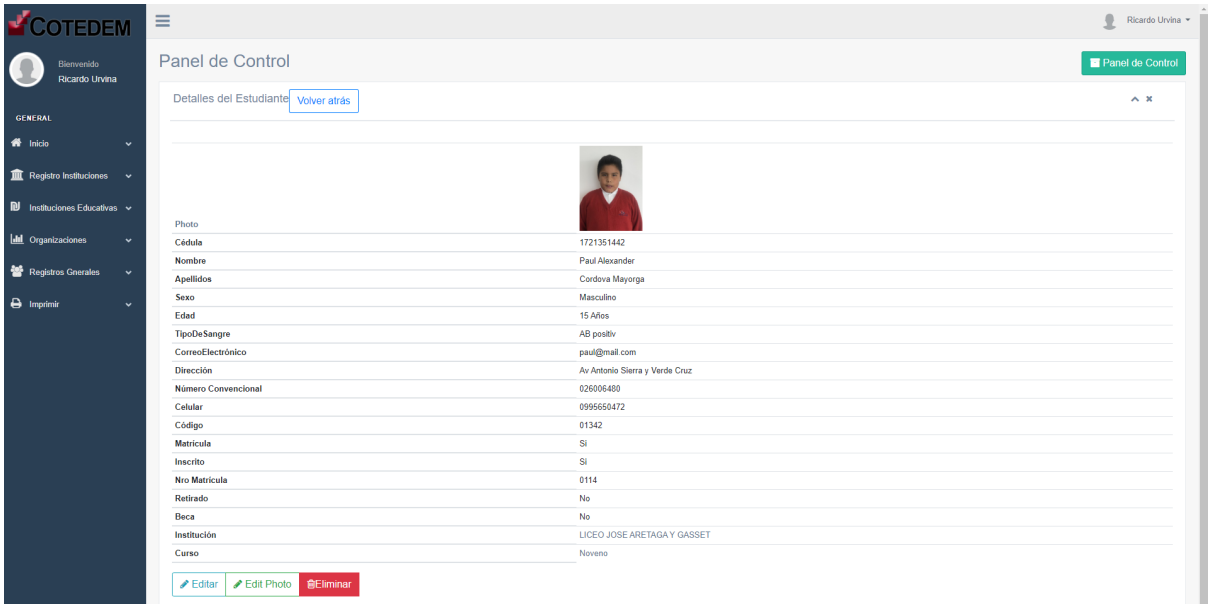


Ilustración 15 Visualización de Datos

(Urvina Cordova, 2020)

e. Diseño Visualización Instituciones/Organizaciones

Esta pantalla muestra cómo se indican los datos de una institución u organización, con sus cursos respectivos en el caso de ser una institución educativa.

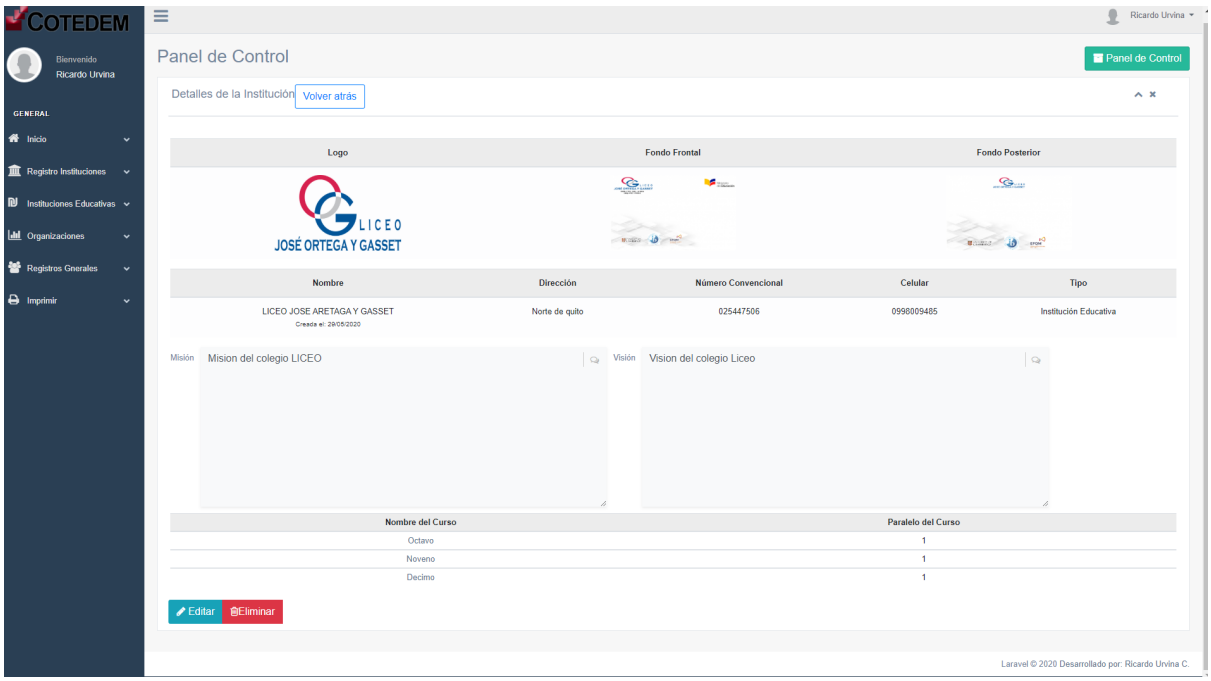


Ilustración 16 Visualización Instituciones/Organizaciones

(Urvina Cordova, 2020)

f. Diseño Ingreso de Datos

Esta pantalla muestra un formulario de ingreso de datos, el cual es semejante para cualquier ingreso de datos de cualquier modelo.

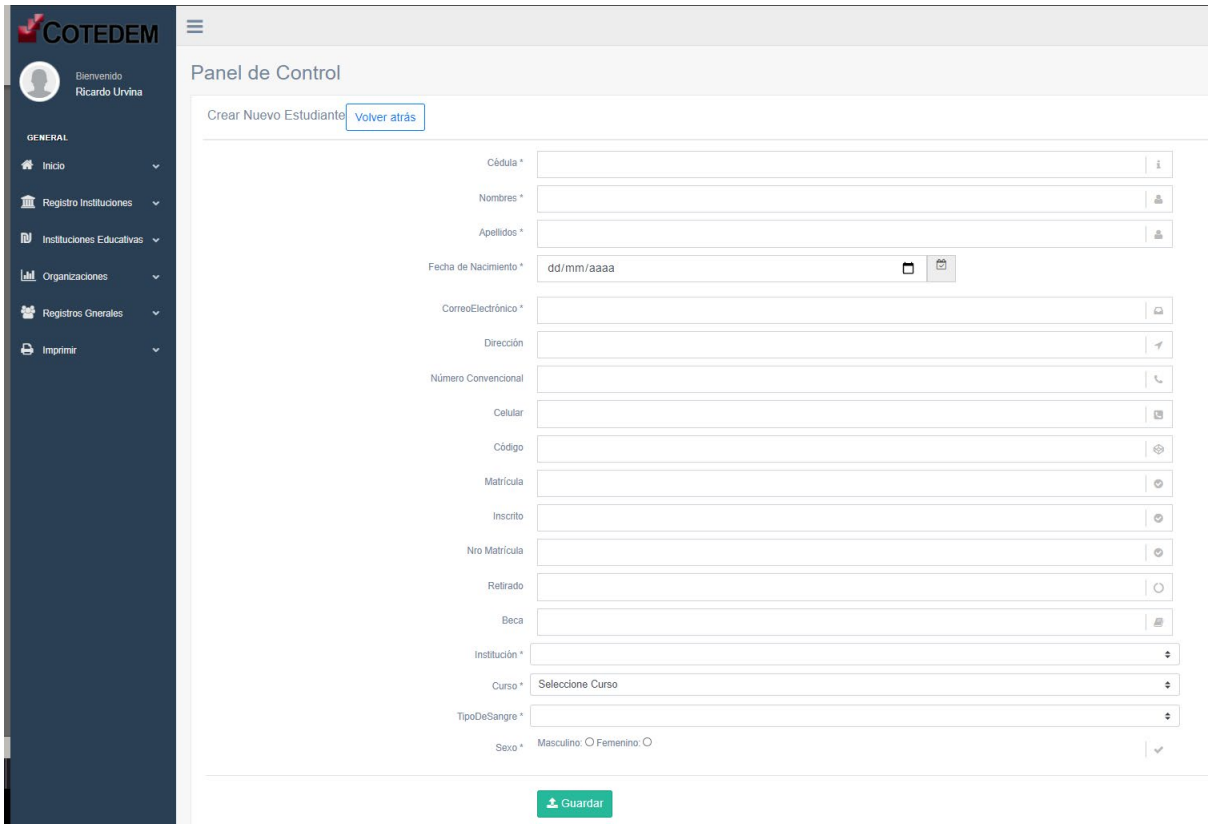


Ilustración 17 Ingreso de Datos

(Urvina Cordova, 2020)

Capitulo 4: Desarrollo del Sistema Web e implementación.

4.1. Desarrollo del BackEnd.

El framework Laravel antes de ser instalado necesita de requisitos para poder empezar su ejecución y uso, (LARAVEL, 2018).

- PHP> = 7.2.5
- Extensión PHP BCMath
- Extensión PHP Ctype
- Extensión de Fileinfo PHP
- Extensión PHP JSON
- Extensión PHP Mbstring
- Extensión PHP OpenSSL
- PDO PHP Extension
- Tokenizer PHP Extension
- Extensión XML PHP

Para el desarrollo de este sistema se instaló los programas:

- XAMPP
- Composer
- MySQL Workbench
- Node Js

En el desarrollo de este sistema como ya se mencionó anteriormente se usó la arquitectura MVC.

a. Modelos.

Son los archivos PHP necesarios que interactúan con la base de datos, se puede hacer cualquier consulta a la base de datos desde el modelo correspondiente a la tabla que se requiera consultar.

Para la creación de un modelo se lo hace mediante la línea de comandos con la siguiente línea: php artisan make:model NombreDelModelo -a

El nombre del modelo por estándar se lo hace en singular y con la primera letra en mayúscula el final de línea “-a” es una opción de ejecución al crear el modelo para

crear una migración, un seed, un factory y un Controlador con todas las funciones REST correspondientes al modelo.

Se indica a manera de ejemplo las líneas de código del modelo Student en la ilustración 18.

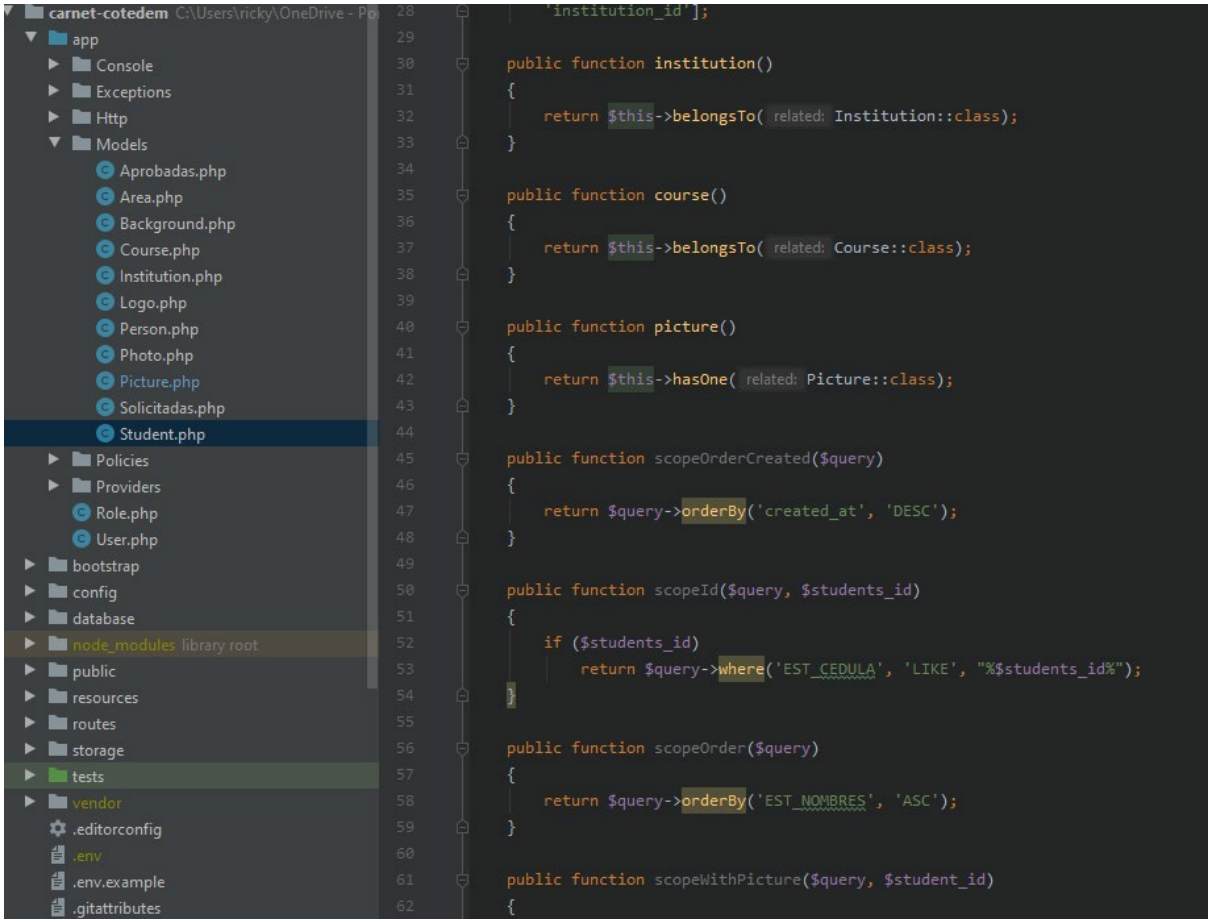


Ilustración 18 Modelo Student

(Urvina Cordova, 2020)

Dentro de cada modelo del sistema se crearon las funciones requeridas para gestionar las relaciones entre las tablas y las consultas SQL necesarias para la ejecución del sistema.

b. Controladores.

Son los archivos PHP que permiten al sistema agrupar la lógica de peticiones HTTP relacionadas, en si el controlador recibe peticiones del frontend, delega las peticiones y devuelve un resultado después de receptar las respuestas de las peticiones delegadas.

Se indica a manera de ejemplo las líneas de código del controlador StudentController en la ilustración 19.

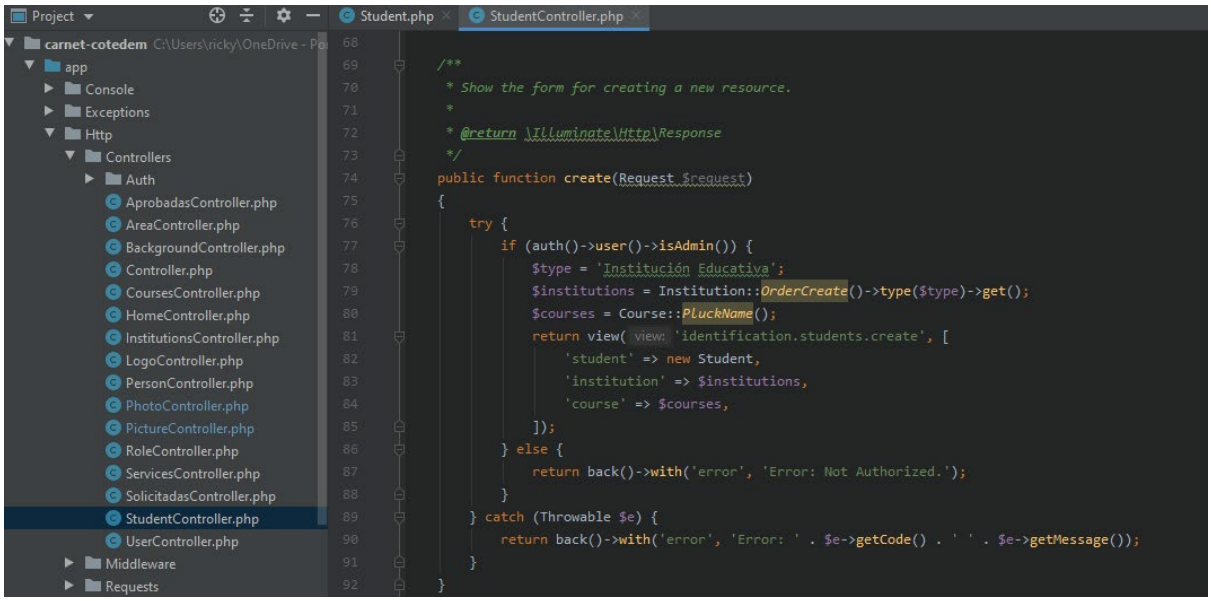


Ilustración 19 StudentController

(Urvina Cordova, 2020)

Dentro de los controladores del sistema se implementaron todas las funciones REST necesarias para cada modelo con el fin de dar cumplimiento a todos los requerimientos del sistema.

Las funciones REST que se crean dentro del controlador son:

Index(): dentro de este método se tiene el código necesario para visualizar la lista de registro del modelo, dentro de la cual dependiendo del modelo se pueden hacer algunos filtros de búsqueda.

Créate(): este método se tiene el código que llama a la vista créate del modelo.

Store(): dentro de este modelo se tiene el código necesario para guardar los datos que se ingresen dentro de la vista créate. Validándolos con un request personalizado para cada modelo.

Show(): este método llama a la vista show del modelo la cual nos muestra todos los datos del registro solicitado.

Edit(): este método hace la llamada a la vista edit del modelo cargando los datos del registro solicitado dentro de la vista.

Update(): este método actualiza y almacena la información que se haya modificado dentro de una vista edit.

Destroy(): este método elimina el registro solicitado de la base de datos.

c. Vistas.

Laravel tiene un Blade, el cual es un motor de plantillas, mediante el uso de vistas separamos la lógica de la presentación donde se receptan datos y se devuelve información.

Se indica a manera de ejemplo las líneas de código de la vista de la lista de estudiantes en la ilustración 20.

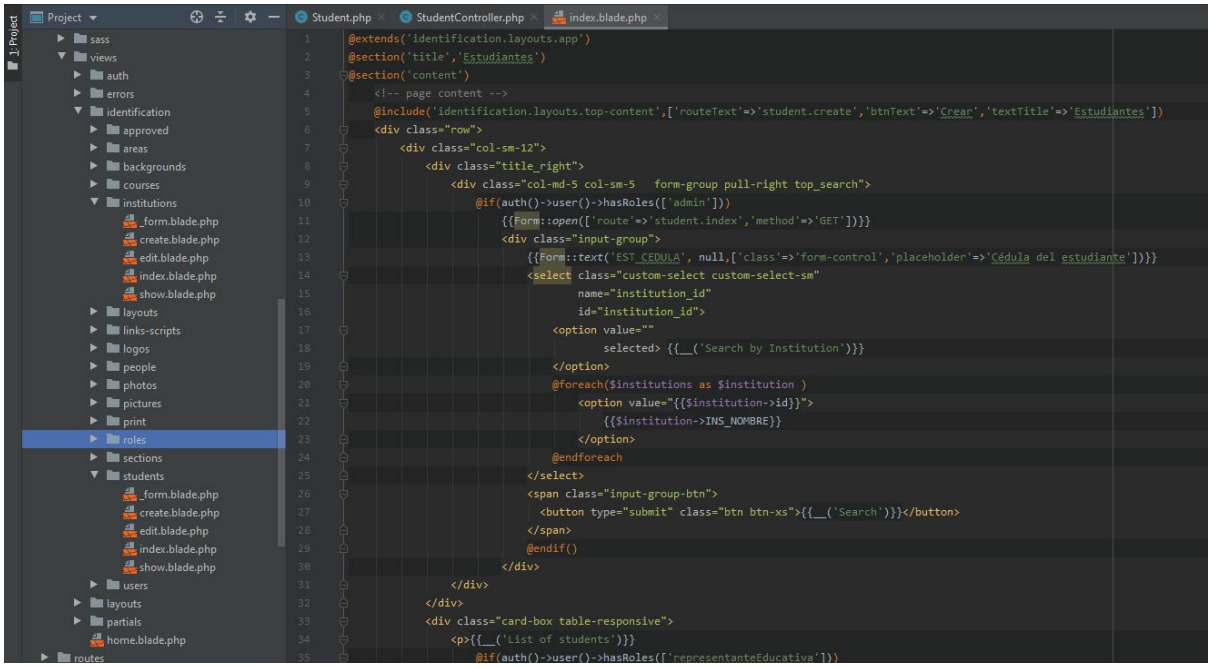


Ilustración 20 Vista Index Students

(Urvina Cordova, 2020)

Se crearon todas las vistas necesarias para satisfacer los requerimientos solicitados.

Estas 3 carpetas contienen todos los archivos necesarios para cumplir con la arquitectura MVC.

Para el correcto uso y aplicación de la arquitectura MVC se utilizó los siguientes recursos que provee laravel:

d. Request

Permite la validación de datos, mensajes de error y autorizaciones que el controlador necesite. Esta capa intercepta la solicitud y valida los datos provenientes de una petición HTTP antes de ir al controlador.

Se indica a manera de ejemplo las líneas de código del request StudentRequest en la ilustración 21.

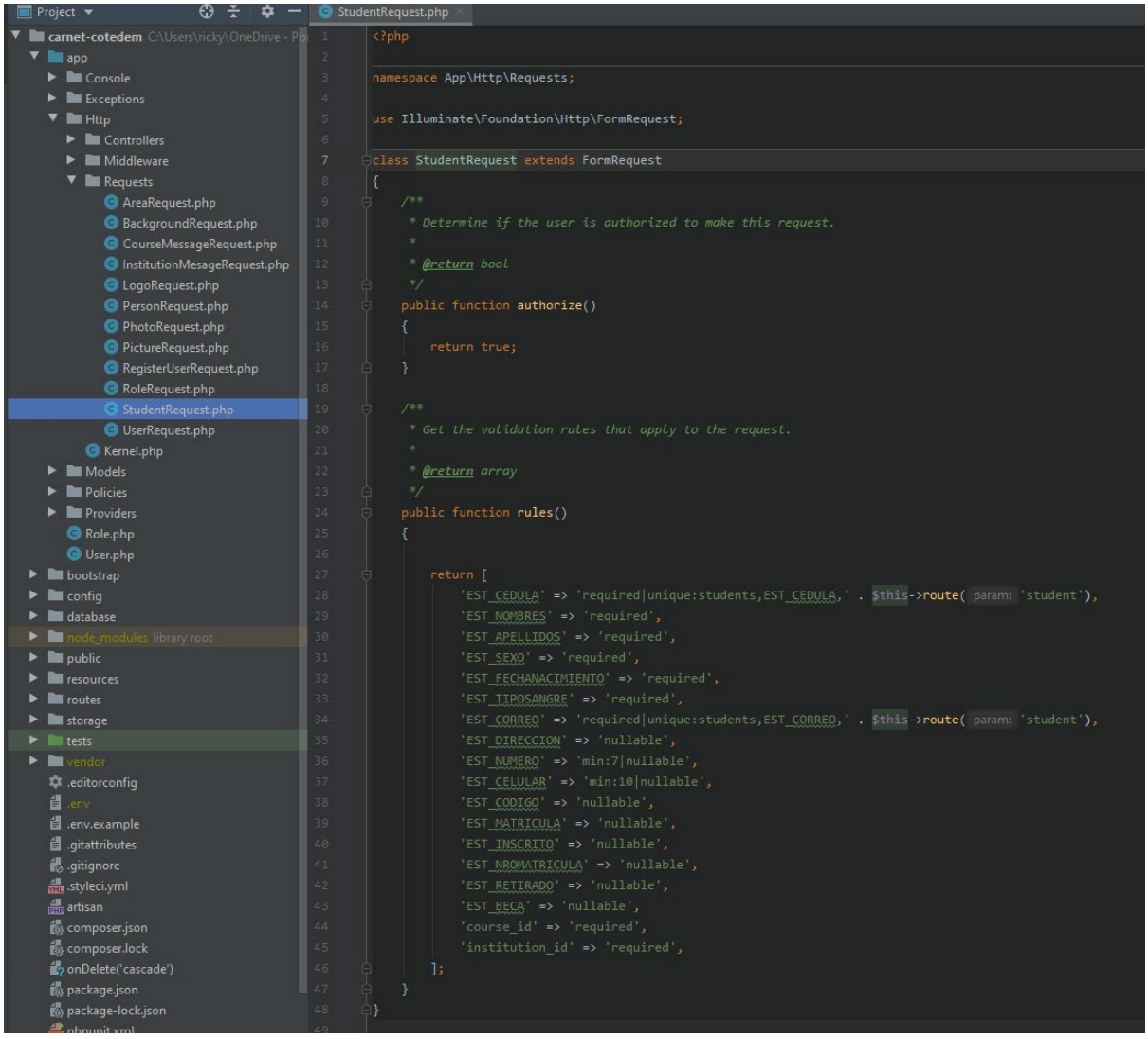


Ilustración 21 StudentRequest

(Urvina Cordova, 2020)

Para este sistema se realizaron todos los request personalizados para cada modelo, facilitando la validación de datos fuera del controlador.

e. Middleware.

Dentro de laravel el middleware cumple una función importante ya que es el encargado de filtrar las solicitudes HTTP que ingresan en la aplicación.

Se indica a manera de ejemplo las líneas de código del middleware CheckRoles en la ilustración 22.

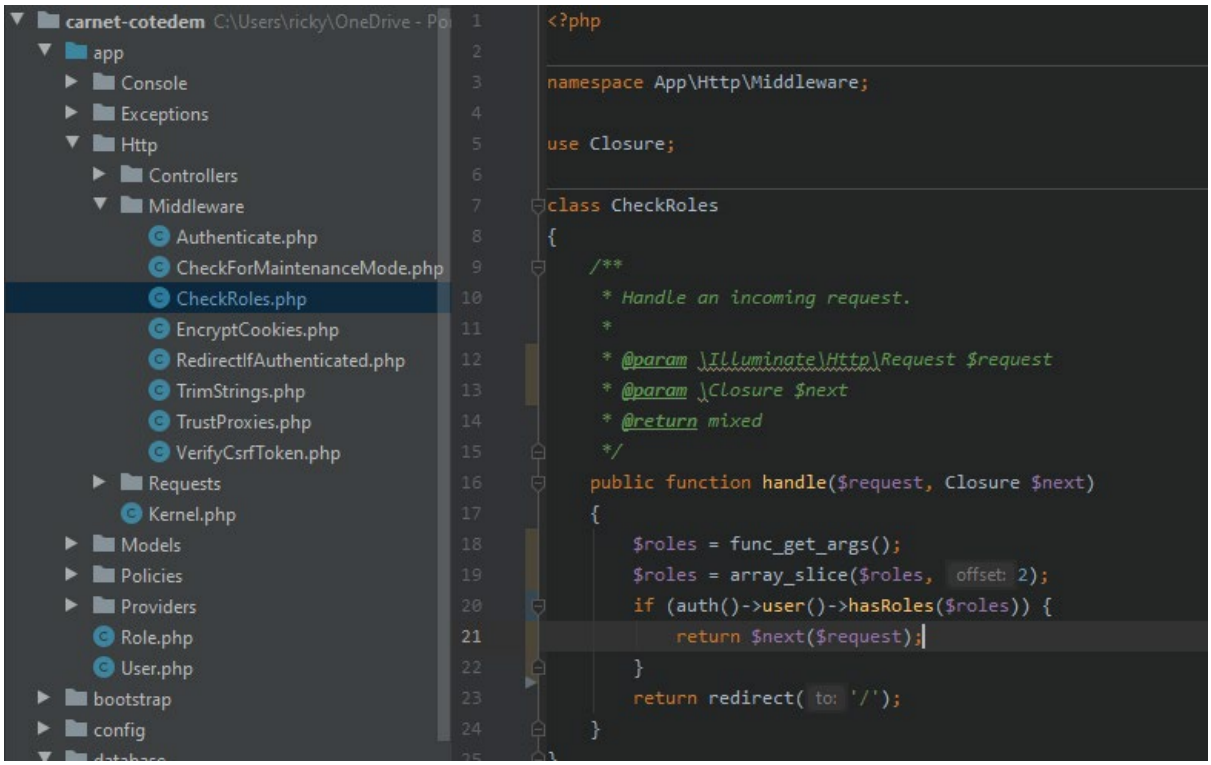


Ilustración 22 Middleware CheckRoles

(Urvina Cordova, 2020)

Los middlewares que se utilizan dentro de este sistema son el middleware que controla quien esta autenticado, y se creó un Middleware personalizado “CheckRoles” para comprobar el rol del usuario que está dentro del sistema.

f. Policies

Para controlar las acciones de autorización dentro del sistema se implementaron políticas que agrupan las acciones de autorización para un modelo especifico o algún recurso que necesita de privacidad.

Se indica a manera de ejemplo las líneas de código del la política de privacidad StudentPolicy en la ilustración 23.

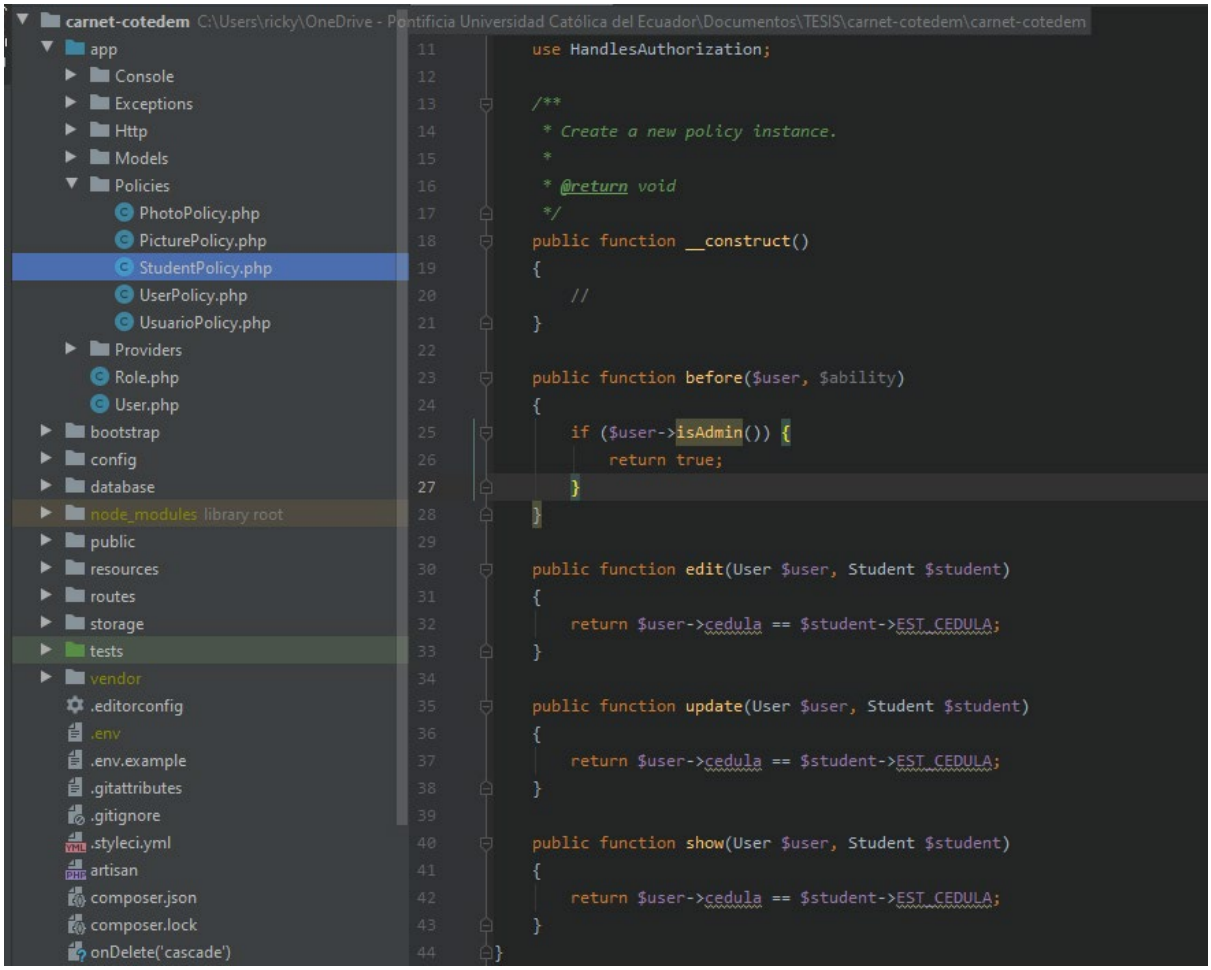


Ilustración 23 StudentPolicy

(Urvina Cordova, 2020)

Dentro del sistema se crearon las políticas de privacidad para los modelos de estudiantes usuarios y fotos de los anteriormente mencionados.

g. Rutas

Es la capa de laravel donde se diferencian las rutas api y web, como su nombre lo dice son las rutas URL que se ejecutaran en la barra de direcciones del navegador para acceder a los distintos servicios de la aplicación.

Se indica a manera de ejemplo las líneas de código de las rutas del sistema en la ilustración 24.

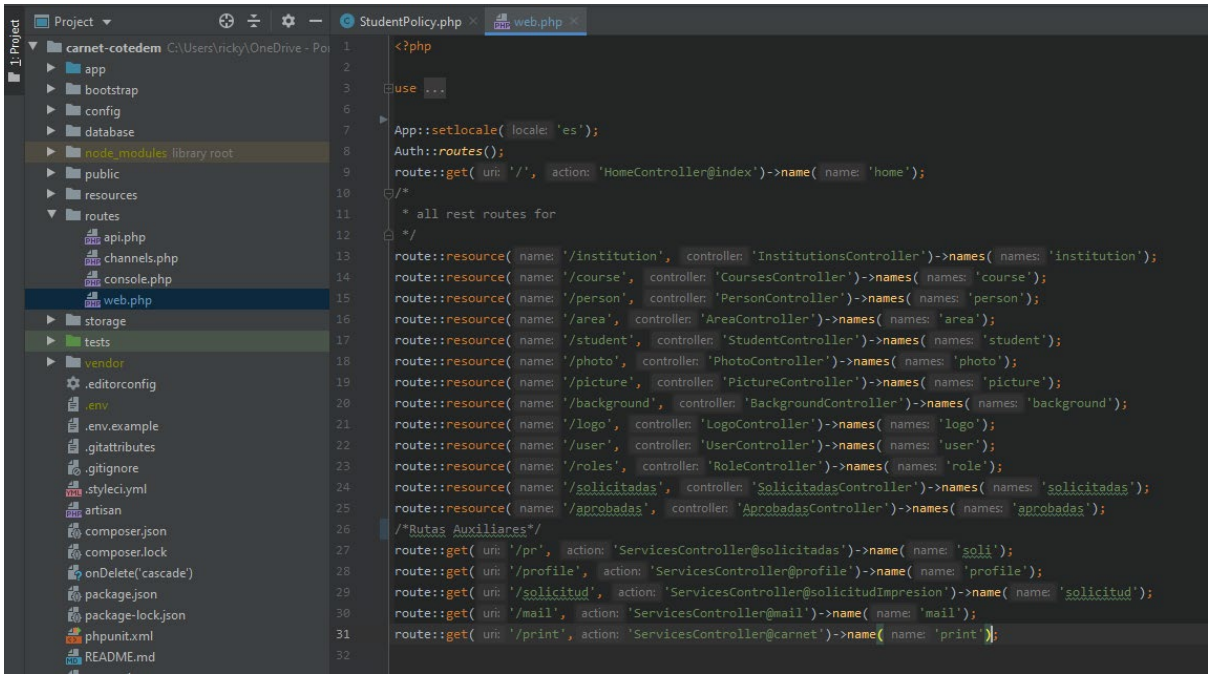


Ilustración 24 Web.php

(Urvina Cordova, 2020)

Para el sistema propuesto se utilizaron rutas web y api, en las rutas web se utiliza el recurso route:resource, que laravel internamente crea las 7 rutas REST necesarias para el modelo requerido, adicional a las rutas web dentro de las rutas api se creó 4 rutas api las cuales son llamadas por unos archivos .JS que realizan peticiones axios dentro del sistema, por ejemplo:

```
route: get('/institution/{id}/courses', 'CoursesController@byInstitution')-
>name('institution.courses');
```

es la ruta api que ejecuta el archivo .js para la carga de cursos dependiendo de la selección de la institución dentro de un selector de instituciones.

h. Migraciones.

Las migraciones permiten realizar acciones DDL dentro de la base de datos, su principal ventaja es el desarrollo del esquema de la base de datos.

Se indica a manera de ejemplo las líneas de código de la migración de estudiantes en la ilustración 25.

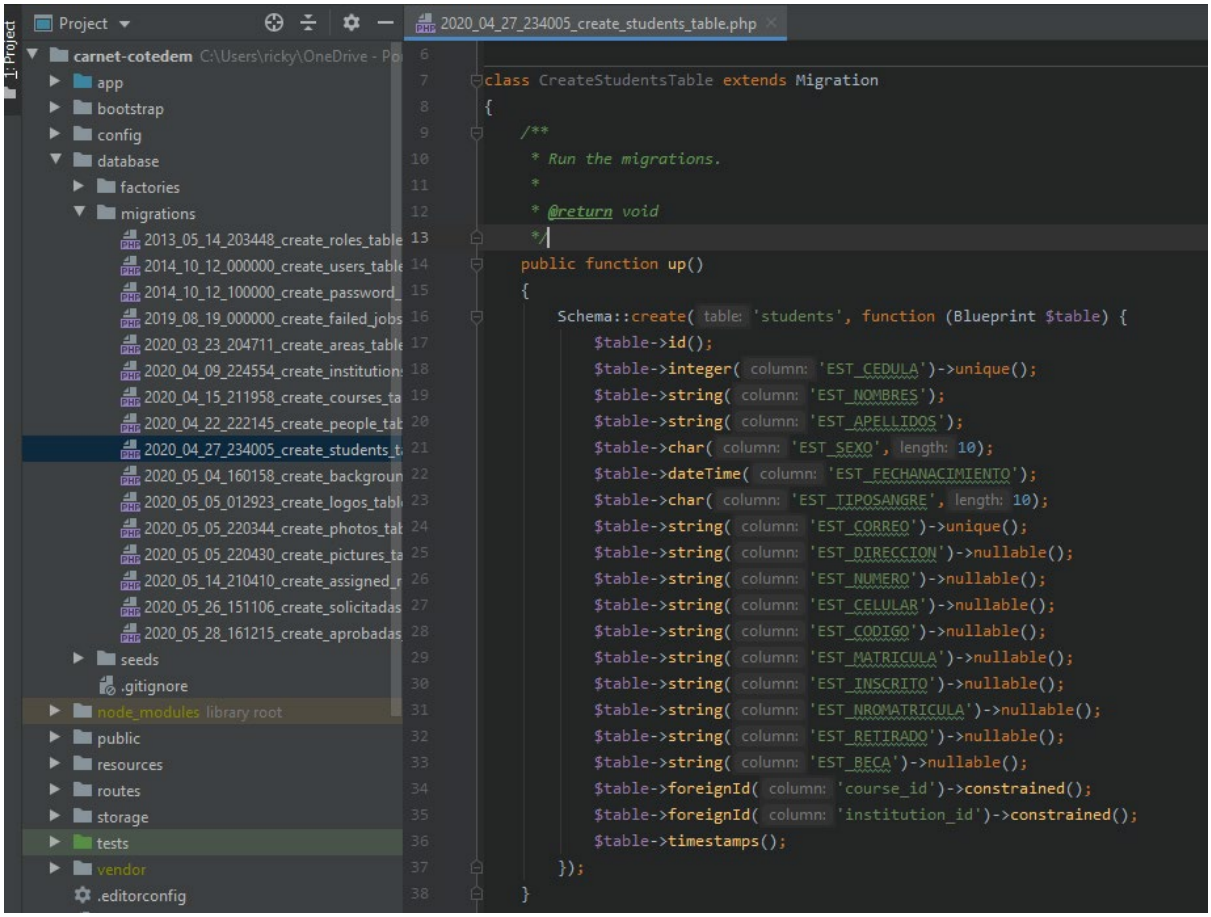


Ilustración 25 Migración Estudiantes

(Urvina Cordova, 2020)

Al crear cada modelo necesario de la aplicación se creó una migración, dentro de las cuales se definieron los campos necesarios y sus relaciones para cada tabla.

Realización del carné.

Se indica a manera de ejemplo las líneas de código para crear un carné en la ilustración 26.

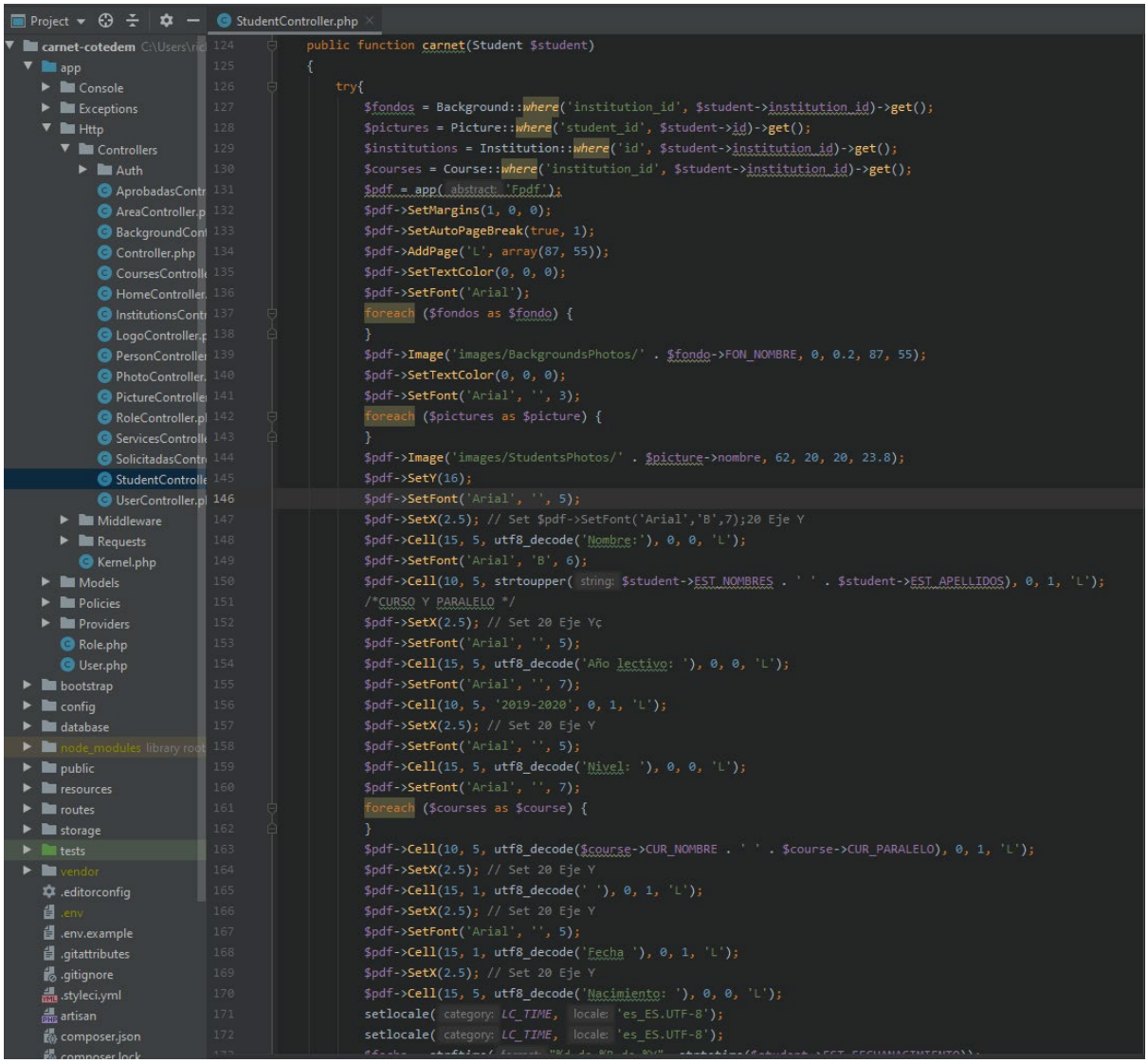


Ilustración 26 Realización Carné

(Urvina Cordova, 2020)

Se utilizó FPDF la cual es una librería de PHP para el manejo de archivos PDF, para lo cual se desarrolló de la siguiente manera:

Se crea la página del pdf con las dimensiones de 87x55 mm la cual es el tamaño que tendrán las identificaciones, dentro de cada página creado se ubican los datos de los usuarios o estudiantes, los fondos y la foto de cada estudiante o usuario, el resultado final de estas líneas de códigos son dos hojas de pdf creadas a manera de carné como se muestra en la ilustración 27.



Ilustración 27 Formato Carné

(Urvina Cordova, 2020)

Control de versionamiento

Para un control de versionamiento se utilizó la herramienta GitHub en el cual se creó dos Branch, uno con el nombre de rickyurvina y el master (Branch principal), esta herramienta permitió durante todo el periodo de desarrollo tener un control de versiones para corrección de errores y tener un respaldo del sistema en la nube.

Con este control de versionamiento se puede instalar el sistema en cualquier entorno local o de producción, solo se requieren de algunos pasos.

Pasos para la instalación del sistema en un entorno local en caso de perder el sistema.

Para poder instalar el sistema se requiere previamente tener instalado Git Bash, Composer, Xampp, Node js, MySQL con Workbench.

Primer paso.

Se inicia git bash en donde se alojará el sistema.

Se escribe el comando Git init dentro de la consola git bash + enter.

Segundo paso.

Se escribe el comando git clone https://github.com/rurvina208/carné-cotedem.git + enter.

Tercer paso.

Abrimos el sistema con cualquier entorno de desarrollo como PHP Storm o VisualCode.

Cuarto paso.

Dentro de la línea de comandos del entorno de desarrollo se escribe el comando Composer install, esto permite que se instalen todas las dependencias de Composer.

Quinto paso.

Dentro de la línea de comandos del entorno de desarrollo se escribe el comando npm install, esto permite que se instalen todas las dependencias de Node js.

Sexto paso.

Se duplica el archivo. Env.example con el nombre de .env. se escriben todas las llaves necesarias de conexión para la ejecución del sistema dentro del archivo .env.

Séptimo paso.

Dentro de la línea de comandos del entorno de desarrollo se escribe el comando php artisan key:generate, esto permite crear la llave única que usa laravel para la ejecución del sistema.

Octavo paso.

Dentro de la línea de comandos del entorno de desarrollo se escribe el comando php artisan:migrate, esto permite migrar la base de datos y su estructura al motor de base de datos.

Con los pasos mencionados se puede tener listo el sistema para el desarrollo y su ejecución de igual manera muchos servidores web permiten montar el sistema de manera muy fácil en un servidor clonando toda la aplicación desde GitHub.

4.2. Desarrollo de FrontEnd (Interfaz Gráfica).

Como base para la interfaz gráfica se utilizó una plantilla llamada: Gentelella Máster la cual es una plantilla gratis de Bootstrap, Gentelella es una plantilla responsive que implementa HTML5, CSS3 y JavaScript la cual se puede descargar gratis en la siguiente dirección:

https://ehantechnology.com/free-items/gentelella-free-bootstrap-admin-template/

Para el uso de esta plantilla en el sistema se la implementó dentro de laravel copiando los archivos css3 y JavaScript de la plantilla en la carpeta publica la cual se encuentra dentro del directorio del proyecto, dentro de la carpeta publica se encuentran gran parte de los archivos necesarios que necesita el sistema para su ejecución, se almacenan los archivos .JS .css e imágenes.

Para empezar el uso de la plantilla dentro del sistema se modularizó la plantilla para evitar el rehúso de código Html5. El punto de entrada para todas las vistas del sistema es el archivo app.blade.php que se encuentra dentro de la carpeta de las vistas del proyecto como se indica en la Ilustración 28.

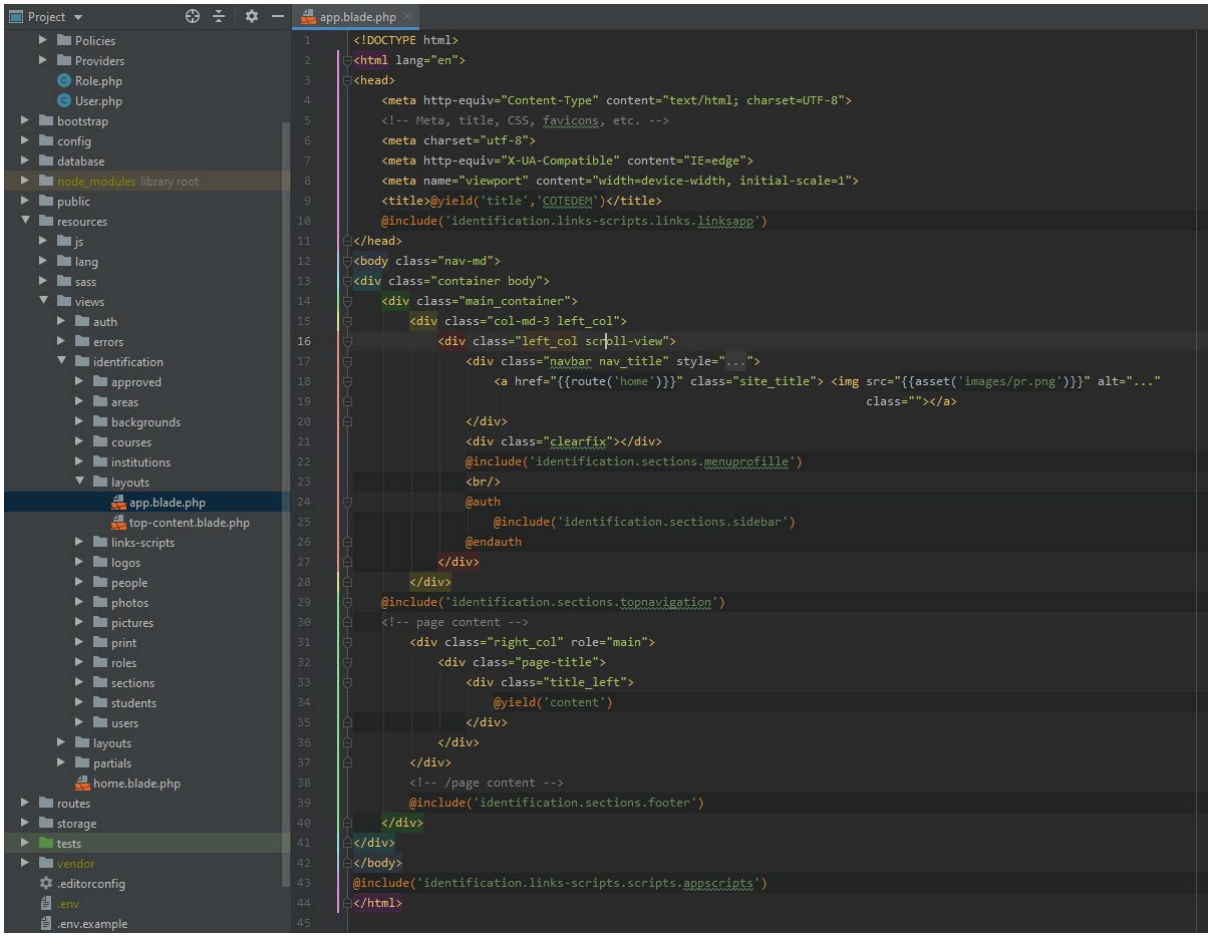


Ilustración 28 Punto de Entrada FrontEnd

(Urvina Cordova, 2020)

Teniendo el punto de entrada del frontend se procedió a modularizar las secciones que requiere el front end para su correcta ejecución como se muestra en la ilustración 29.

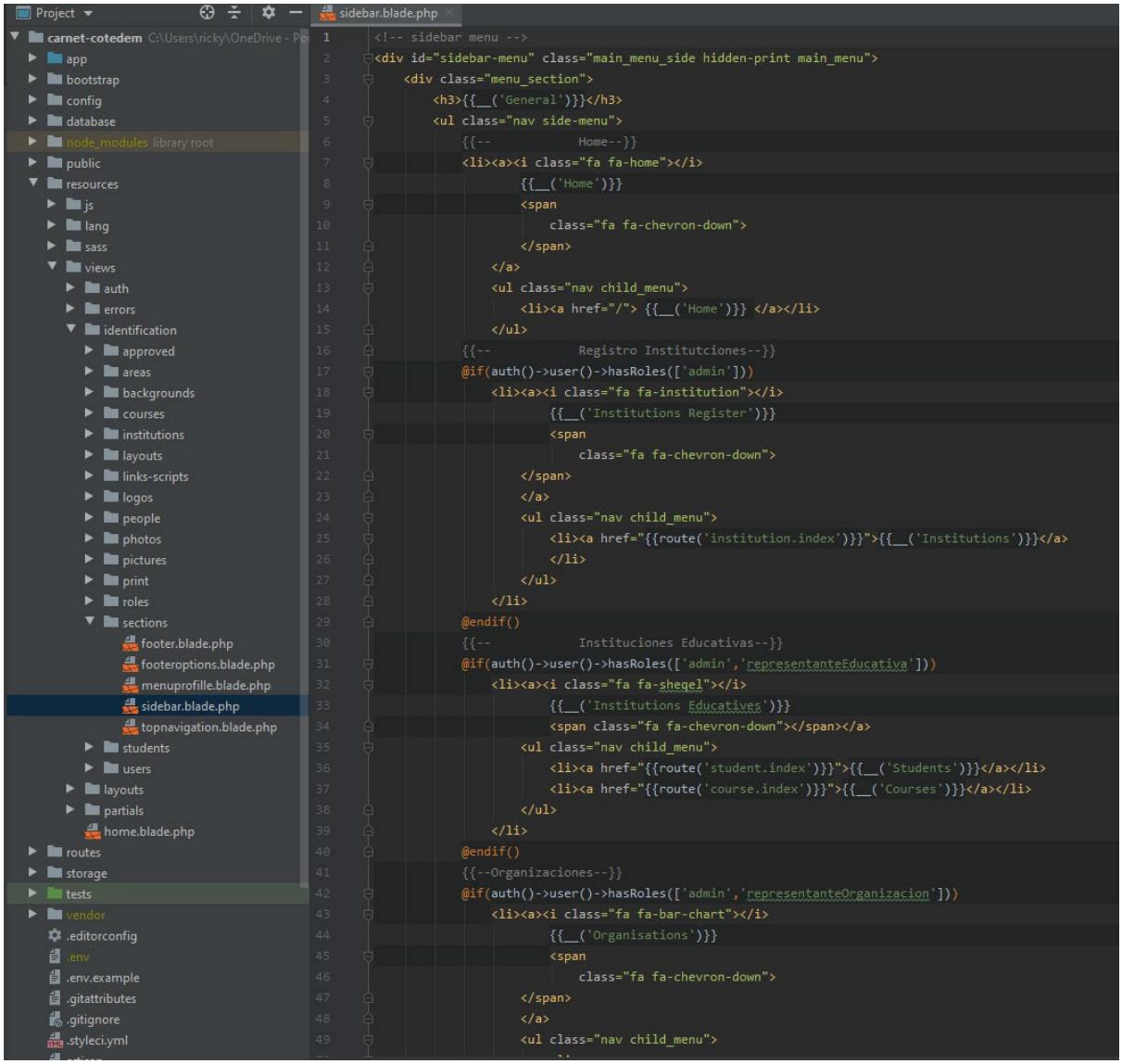


Ilustración 29 SideBar Sección de FrontEnd

(Urvina Cordova, 2020)

Con el punto de entrada y las secciones establecidas lo que se procedió a realizar son todas las vistas necesarias para un CRUD, las cuales son semejantes para todos los modelos del proyecto, como se muestra en la ilustración 30.

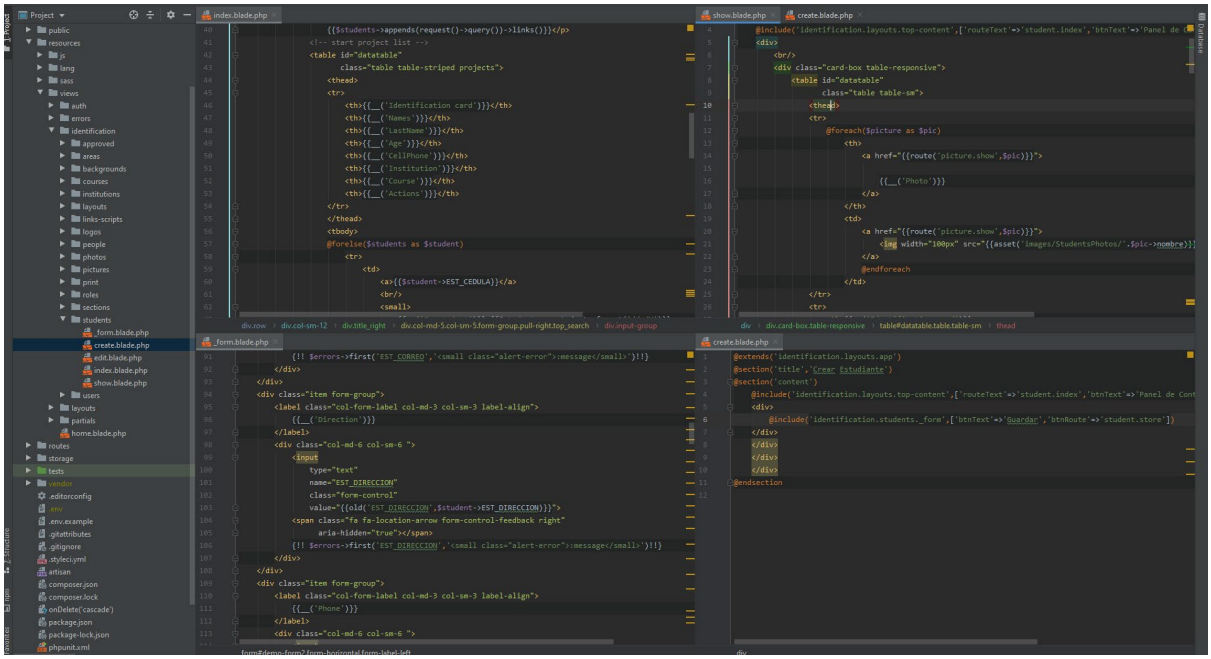


Ilustración 30 Vistas CRUD Student

(Urvina Cordova, 2020)

Mediante el motor de plantillas de laravel se rehúsa el código necesario con directivas, las cuales llaman a otros archivos HTML5 a ejecutarse donde se lo requiera y con los parámetros necesarios. Como se observa la vista créate solo hace una llamada de la vista form, la cual se ejecuta de manera similar en las vistas créate y edit.

4.3. Implementación del Sistema Web en el dominio de la empresa.

Para la implementación del aplicativo web desarrollado dentro del sistema se estableció directamente con el gerente de COTEDEM las normas y directrices necesarias para subir el sistema al servidor.

COTEDEM posee alojamiento web dentro de bluehost, la cual es una empresa de alojamiento web, que incluye servicios de alojamiento compartido, alojamiento de WordPress, alojamiento VPS, alojamiento dedicado, los servidores funcionan con PHP, HTTP, NGINX. Dentro del alojamiento dedicado que tiene contratado la empresa, se procedió a crear un subdominio en el cual se alojo el sistema de manera exitosa.

Para la implementación del sistema web en el dominio de la empresa se requirió de los siguientes pasos.

Paso 1.

Comprimir el aplicativo en .zip, como se muestra en la ilustración 31.

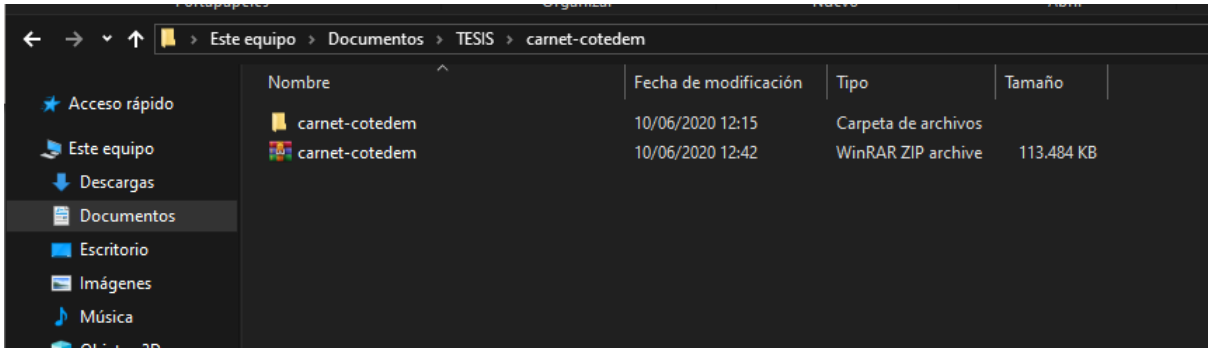


Ilustración 31 compresión Carnet-Cotedem

(Urvina Cordova, 2020)

Paso 2.

Dentro de bluehost iniciamos sesión con las credenciales de COTEDEM, posterior nos dirigimos a cpanel como se muestra en la ilustración 32.

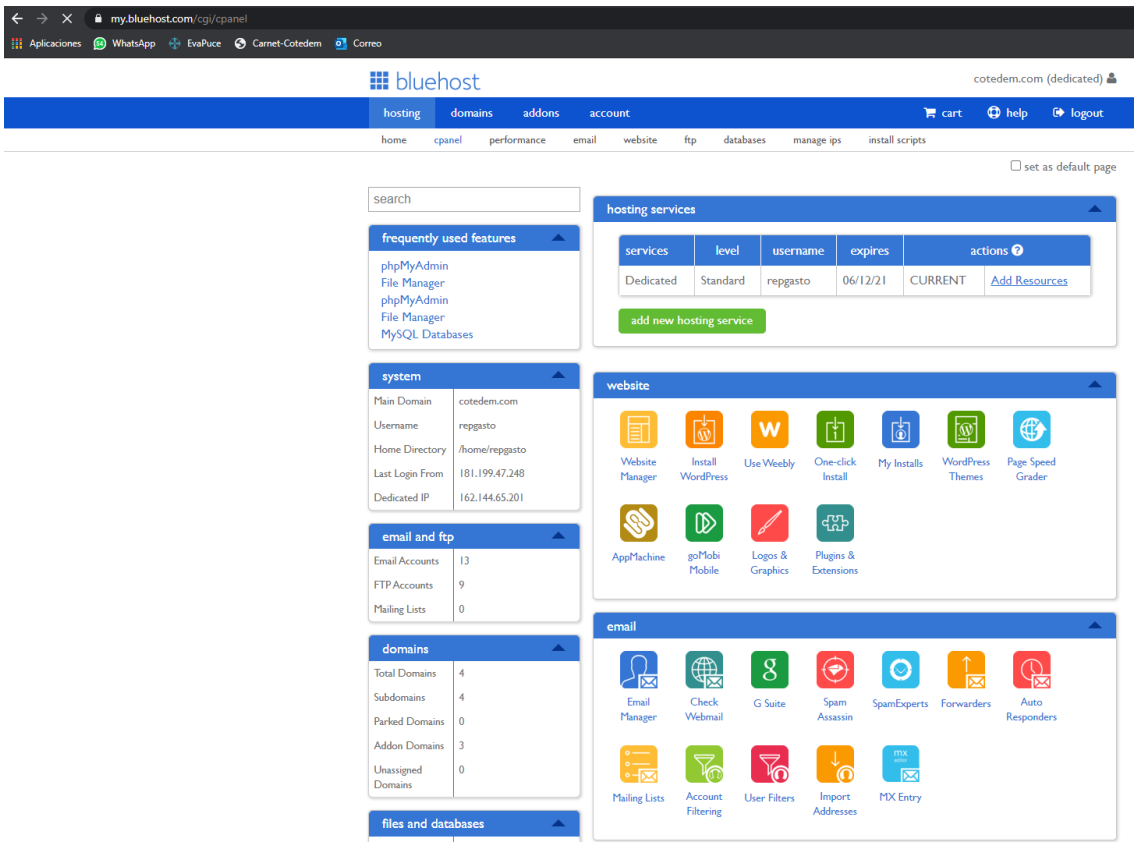


Ilustración 32 Cpanel BlueHost

(Urvina Cordova, 2020)

Paso 3.

Dentro de la sección files, nos ubicamos en file manager, donde se crea una carpeta llamada idCard dentro de la carpeta public_html como se muestra en la ilustración 33.

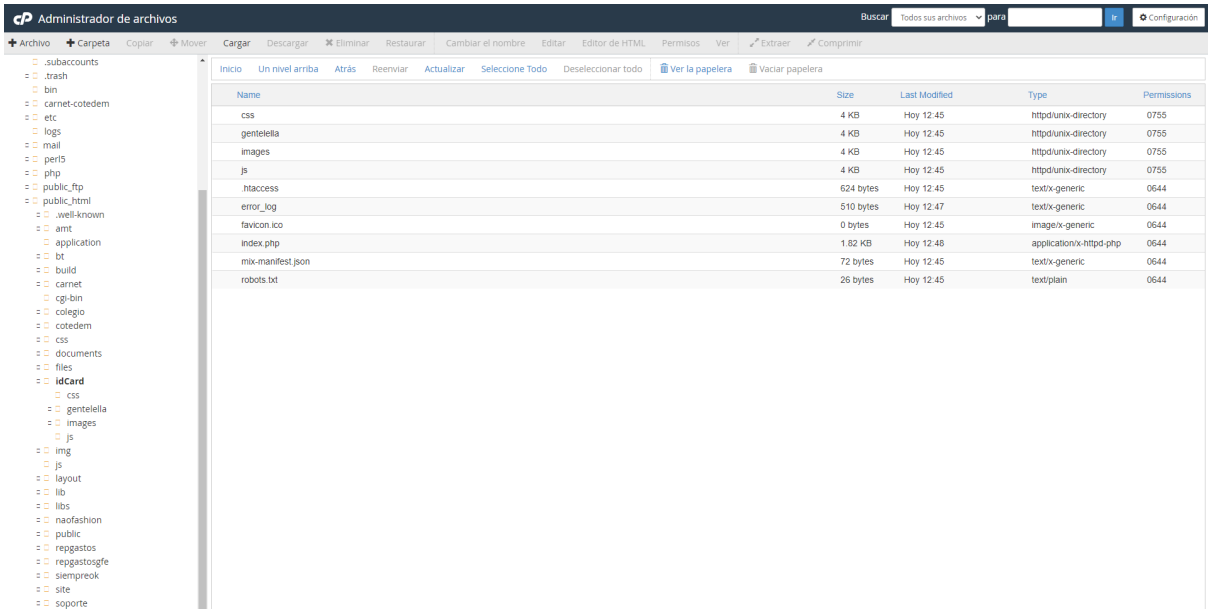


Ilustración 33 Carpeta idCard en public_html

(Urvina Cordova, 2020)

En la carpeta idCard se colocan todos los archivos de la carpeta public del aplicativo de laravel.

Paso 4.

Dentro de la raíz de los archivos se creó una carpeta llamada “carnet-cotedem”, dentro de la misma se colocaron todos los archivos del aplicativo web como se muestra en la ilustración 34.

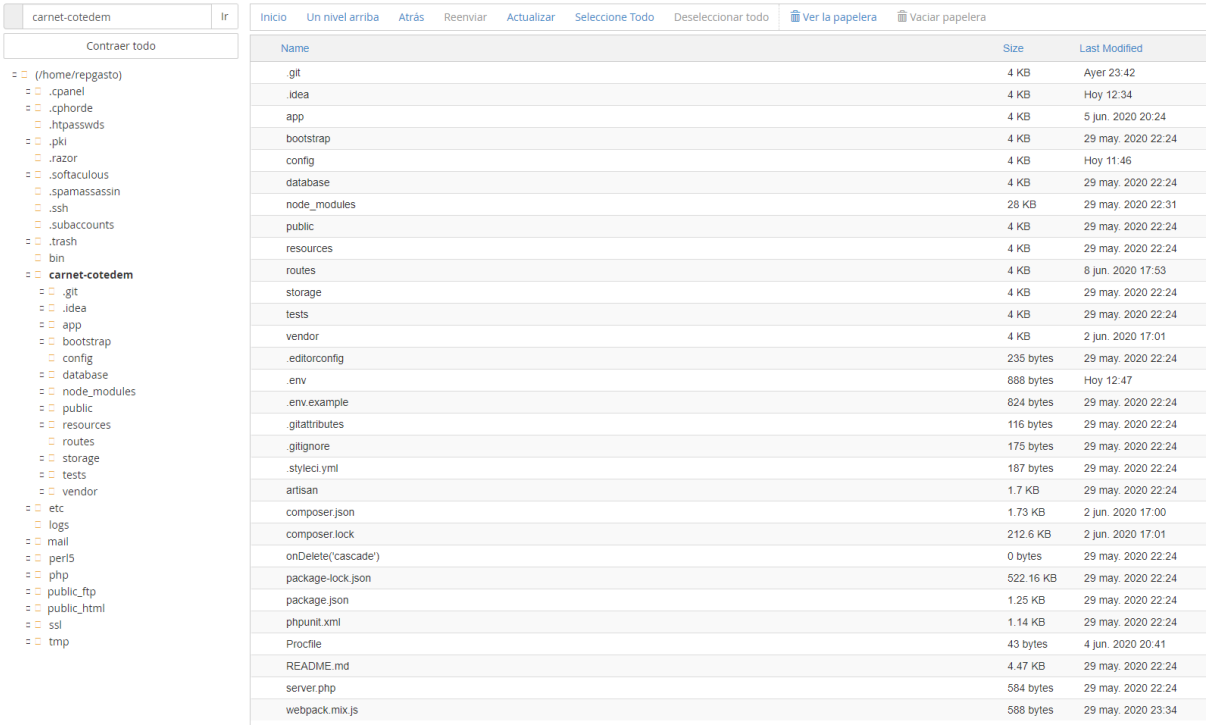


Ilustración 34 Carpeta carnet-cotedem

(Urvina Cordova, 2020)

Paso 5.

Dentro de la carpeta idCard, el archivo index.php, se configura de la siguiente manera para la correcta ejecución del sistema como se muestra en la ilustración 35.

index.php (PHP script text)

```php
<?php

/**
 * Laravel - A PHP Framework For Web Artisans
 *
 * @package  Laravel
 * @author   Taylor Otwell <taylor@laravel.com>
 */

define('LARAVEL_START', microtime(true));

/*
|--------------------------------------------------------------------------
| Register The Auto Loader
|--------------------------------------------------------------------------
|
| Composer provides a convenient, automatically generated class loader for
| our application. We just need to utilize it! We'll simply require it
| into the script here so that we don't have to worry about manual
| loading any of our classes later on. It feels great to relax.
|
*/

require __DIR__.'/../../carnet-cotedem/vendor/autoload.php';

/*
|--------------------------------------------------------------------------
| Turn On The Lights
|--------------------------------------------------------------------------
|
| We need to illuminate PHP development, so let us turn on the lights.
| This bootstraps the framework and gets it ready for use, then it
| will load up this application so that we can run it and send
| the responses back to the browser and delight our users.
|
*/

$app = require_once __DIR__.'/../../carnet-cotedem/bootstrap/app.php';

/*
|--------------------------------------------------------------------------
| Run The Application
|--------------------------------------------------------------------------
|
| Once we have the application, we can handle the incoming request
| through the kernel, and send the associated response back to
| the client's browser allowing them to enjoy the creative
| and wonderful application we have prepared for them.
|
*/

$kernel = $app->make(Illuminate\Contracts\Http\Kernel::class);

$response = $kernel->handle(
    $request = Illuminate\Http\Request::capture()
);

$response->send();

$kernel->terminate($request, $response);
```

Ilustración 35 Configuración index.php

(Urvina Cordova, 2020)

Paso 6.

En el archivo .env ubicado en carnet-cotedem se edita este archivo con las variables de entorno necesarias para su correcta ejecución, como se muestra en la ilustración 36.

```
APP_NAME=Laravel
APP_ENV=production
APP_KEY=base64:lyMZwsWhtAJvjkBz1CkPcLXZfft1a5iuHrt2vcRslME=
APP_DEBUG=false
APP_URL=http://localhost

LOG_CHANNEL=stack

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=repgasto_idcard
DB_USERNAME=repgasto_admin
DB_PASSWORD=Cotedem$1205

BROADCAST_DRIVER=log
CACHE_DRIVER=file
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
SESSION_LIFETIME=120

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379


MAIL_MAILER=smtp
MAIL_HOST=smtp.office365.com
MAIL_PORT=25
MAIL_USERNAME=info@cotedem.com
MAIL_PASSWORD=Cotedem@2020
MAIL_ENCRYPTION=tls
MAIL_FROM_ADDRESS=info@cotedem.com
MAIL_FROM_NAME=User

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
PUSHER_APP_CLUSTER=mt1

MIX_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
MIX_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
```

Ilustración 36 Configuracion .env

(Urvina Cordova, 2020)

Paso 7.

Se crea la base de datos dentro del servidor como se muestra en la ilustración 37.

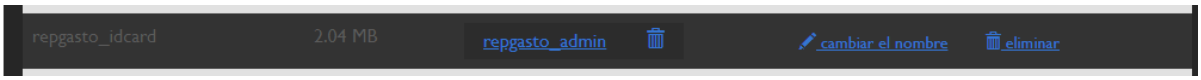


Ilustración 37 Base de Datos Creada

(Urvina Cordova, 2020)

Paso 8.

Se importa el script de creación de la base de datos dentro de phpMyAdmin, para tener configurada y lista la base de datos como se muestra en la ilustración 38.

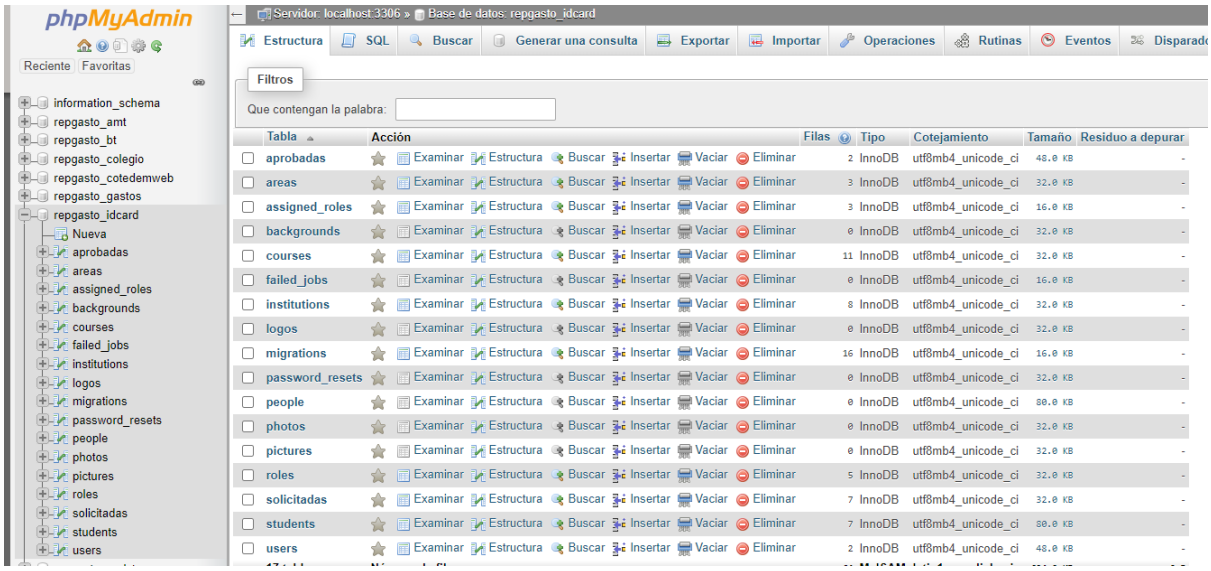


Ilustración 38 Base de Datos

(Urvina Cordova, 2020)

Paso 9.

Se escribe en el navegador la siguiente dirección www.cotedem.com/idCard para comprobar el correcto funcionamiento de sistema, como se muestra en las siguientes ilustraciones 39 y 40.

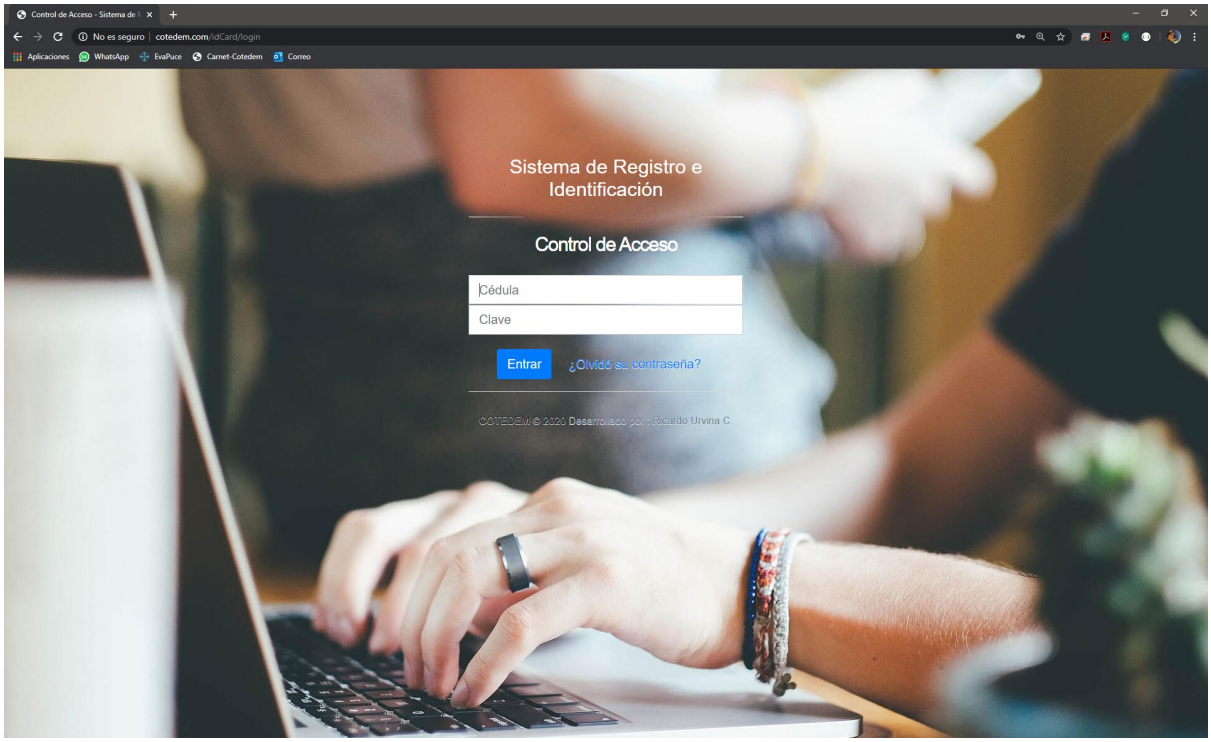


Ilustración 3937 Correcto Funcionamiento del aplicativo en el servidor

(Urvina Cordova, 2020)

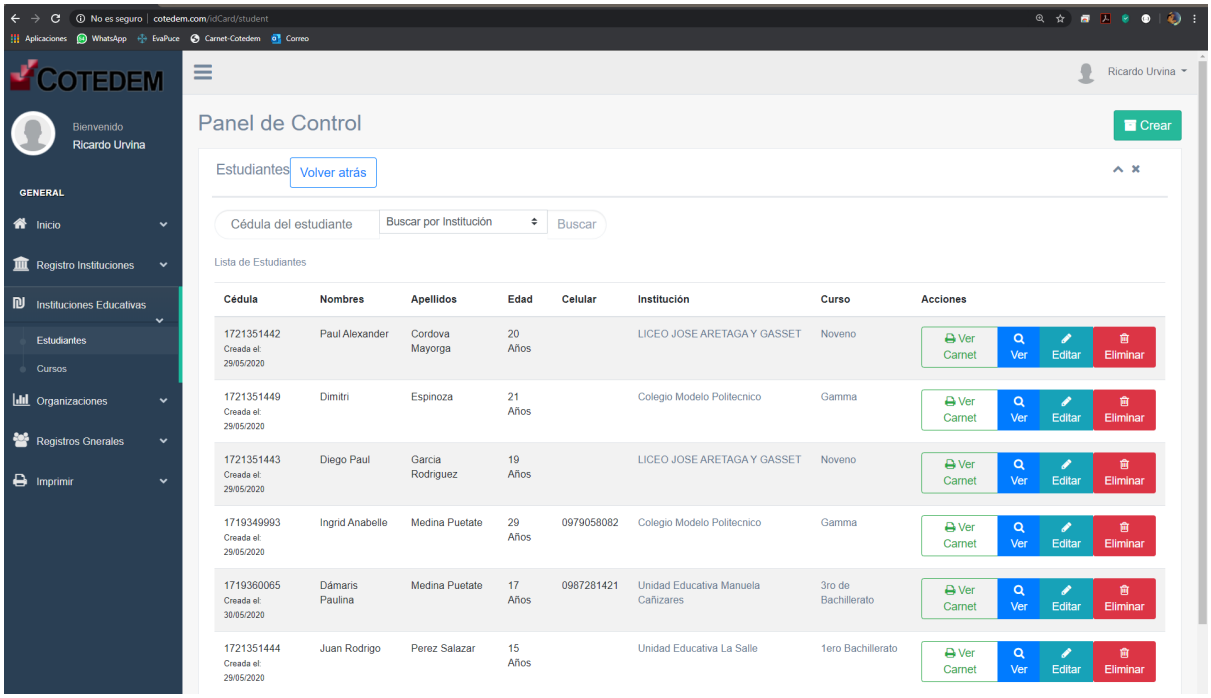


Ilustración 4038 Correcto Funcionamiento del aplicativo en el servidor

(Urvina Cordova, 2020)

El sistema funciona de manera correcta dentro del subdominio de la empresa, la cual ya el administrador o gerente de la empresa harán uso como lo requieran.

Capitulo 5: Conclusiones y Recomendaciones.

En la siguiente sección se detallan las conclusiones que dan cumplimiento al objetivo general y el alcance del sistema.

5.1 Conclusiones.

- Se completo con éxito el análisis de los requerimientos del sistema para el desarrollo para la posterior implementación de la aplicación requerida por COTEDEM.
- Se completo con éxito el diseño del sistema en base al análisis realizado de todos los requerimientos funcionales del sistema.
- Se realizo el diseño del backend y el frontend mediante el diseño propuesto aplicando la arquitectura MVC.
- Se desarrollo el sistema de manera modular, incremental y estable que cubre todos los objetivos y requerimientos planteados.
- Se implemento exitosamente el sistema dentro del dominio y host de la empresa en la dirección . http://cotedem.com/idCard/
- Se utilizo de manera exitosa todas las herramientas tecnológicas para desarrollar el sistema propuesto.
- Gracias a la encriptación de claves de usuarios, el ingreso al sistema es seguro y los datos permanecen alojados de manera segura y fiable.
- La metodología SCRUM ayudo en el desarrollo ágil del sistema gracias a la facilidad del uso frente a otras metodologías.
- Mediante el buen manejo de la plantilla Gentelella se pudo realizar un aplicativo responsive que se puede ejecutar cualquier dispositivo móvil sin que este pierda el diseño.
- Se logro realizar una gestión de carnetización controlada mediante el sistema el cual ahorra un 80% de tiempo a la empresa para realizar este proceso.

5.2. Recomendaciones.

- Mantener el diseño modular de las vistas de la aplicación mediante el uso del motor de plantillas de laravel en versiones posteriores.
- Incrementar la funcionalidad para que un representante de una organización o institución pueda realizar el pedido de carnés seleccionados y no en masa.

- Incrementar la funcionalidad para que los representantes de cada organización o institución puedan editar los datos de los usuarios o estudiantes de la organización que representan.
- Seguir utilizando el control de versionamiento en Git, para la facilidad de los cambios que se necesiten a futuro y hacer rollback si fuera necesario.

Referencias

40defiebre. (2019). Recuperado el 09 de junio de 2020, de ¿Qué es el diseño responsive?: https://www.40defiebre.com/que-es/diseno-responsive

Alvarez, M. A. (19 de Diciembre de 2014). *Desarrolloweb*. Recuperado el 16 de Marzo de 2020, de Ventajas e incovenientes de API REST para el desarollo: https://desarrolloweb.com/articulos/ventajas-inconvenientes-apirest-desarrollo.html

Anónimo. (s.f.). *Proyectos Agiles*. Recuperado el 17 de Marzo de 2020, de Qué es SCRUM: https://proyectosagiles.org/que-es-scrum/

Baquero Garcia, J. (11 de Dciembre de 2015). *arsys*. Recuperado el 17 de Marzo de 2020, de ¿Qué es Laravel?: https://www.arsys.es/blog/programacion/que-es-laravel/

COTEDEM. (04 de Marzo de 2020). *cotedem*. Recuperado el 04 de Marzo de 2020, de https://cotedem.com/

Dario, B. R. (s.f.). *Dario Benabeu eLearning*. Recuperado el 09 de junio de 2020, de Modelo Conceptual: http://troyanx.com/Hefesto/13_modelo_conceptual.html

Dusseldorf, D. B. (1997). *Das grobe Buch JavaScript.* Barcelona: MARCOMBO S.A.

ESPAÑOLA, R. A. (2018). *dle.rae.es*. Obtenido de https://dle.rae.es/?id=ZotYNb8

Española, R. A. (2020). *Real Academia Española*. Recuperado el 09 de junio de 2020, de carné: https://dle.rae.es/carn%C3%A9#3KAsYxe

Ferré, D. B. (2004). *Flash dinámico PHP, XML y bases de datos.* Mdrir: GRUPO ANAYA. S. A.

Gauchat, J. D. (2012). *El gran libro de HTML5, CSS3 y jAVASCRIPT.* Barcelona: MARCOMBO. Recuperado el 01 de Marzo de 2020, de https://books.google.es/books?hl=es&lr=&id=szDMlRzwzuUC&oi=fnd&pg=PA1&dq=css3&ots=0BtHS2uEVi&sig=Kg8Uve7_Vy5fU1ZqcawXvk1WdE0#v=onepage&q=css3&f=false

GitHub. (2020). *GitHub*. Recuperado el 09 de junio de 2020, de https://github.com/

Guetierrez, J. (2018). *isi.* Recuperado el 29 de febrero de 2020, de http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf

Guevara, A. (2016). *BackEnd y FrontEnd.* Recuperado el 29 de Febrero de 2020, de devocode: https://devcode.la/blog/frontend-y-backend/

Henst, V. d. (2012). *edu.xunta.gal*. Obtenido de https://www.edu.xunta.gal/centros/cfrcoruna/aulavirtual2/mod/page/view.php?id=375

IBM. (25 de 04 de 2014). *ibm.com*. Recuperado el 29 de febrero de 2020, de https://www.ibm.com/support/knowledgecenter/es/SSMKHH_9.0.0/com.ibm.etools.mft.doc/ac55710_.htm

IT, I. (s.f.). *Integra IT*. Recuperado el 17 de Marzo de 2020, de Conoce los principales roles de Scrum: https://www.integrait.com.mx/blog/roles-de-scrum/

JS, N. (2020). *Node Js*. Recuperado el 09 de junio de 2020, de https://nodejs.org/es/

Lacker, L. V. (2012). *HTML5 Los Fundamentos del Lenguaje.* Barcelona: Francisco Javier PIQUERES JUAN.

LARAVEL. (2018). *LARAVEL*. Obtenido de https://laravel.com/docs/6.x

laravel-5. (2018). Recuperado el 09 de junio de 2020, de https://richos.gitbooks.io/laravel-5/content/capitulos/chapter10.html

Lucidart. (2019). Recuperado el 09 de junio de 2020, de ¿Qué es un modelo entidad relación?: https://www.lucidchart.com/pages/es/que-es-un-diagrama-entidad-relacion

Microsoft. (Enero de 2020). *Adobe*. Recuperado el 31 de Marzo de 2020, de DREAMWEAVER: https://www.adobe.com/la/products/dreamweaver/free-trial-download.html

Oracle. (2019). *dev.mysql*. Obtenido de https://dev.mysql.com/doc/refman/8.0/en/

Oracle. (2020). *mysql*. Recuperado el 01 de Marzo de 2020, de mysql.com/products/workbench/

Perú, A. (2019). *apptiva*. Recuperado el 09 de junio de 2020, de Axios.js una libreria de promesas.

Pete Deemer, G. B. (2009). *Informacion Basica de Scrum.* San Francisco: Creative Commons.

Porto, J. P. (2017). *definicion.de*. Obtenido de https://definicion.de/api/

PowerDesigner. (2019). *powerdeigner*. Recuperado el 01 de Marzo de 2020, de https://www.powerdesigner.biz/ES/powerdesigner/probar-powerdesigner-source_adw847a.html?gclid=Cj0KCQiA1-3yBRCmARIsAN7B4H03kCuyNNG1PjXC3qrbGx5qkprlVsgXvTP80wv7R_LzU8myIgLrFQoaArXjEALw_wcB

Tecnologías Información. (s.f.). Recuperado el 09 de junio de 2020, de Modelos de Datos: https://www.tecnologias-informacion.com/modelos-datos.html#:~:text=El%20modelo%20de%20datos%20f%C3%ADsicos,en%20la%20base%20de%20datos.&text=Las%20caracter%C3%ADsticas%20de%20un%20modelo,para%20identificar%20relaciones%20entre%20tablas.

Urvina Cordova, R. P. (2020). *ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA LA GESTIÓN DE CARNETIZACIÓN PARA ORGANIZACIONES. CASO DE ESTUDIO: CONSULTORES TECNOLÓGICOS Y DESARROLLO EMPRESARIAL.* Quito, Pichincha, Ecuador.

VUEJS. (2019). *vue.js*. Obtenido de https://es-vuejs.github.io/vuejs.org/v2/guide/

Xampp. (2020). *Xampp*. Recuperado el 09 de junio de 2020, de https://www.apachefriends.org/es/index.html

Yair. (2019). *Styde*. Recuperado el 09 de junio de 2020, de https://styde.net/que-es-composer-y-como-usarlo/

Anexos

Anexo 1. Script Base De Datos.

```
CREATE DATABASE  IF NOT EXISTS `desarrollo` /*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci */ /*!80016 DEFAULT ENCRYPTION='N' */;

USE `desarrollo`;

-- MySQL dump 10.13  Distrib 8.0.19, for Win64 (x86_64)

--

-- Host: localhost    Database: desarrollo

-- ------------------------------------------------------

-- Server version    8.0.19

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;

/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;

/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;

/*!50503 SET NAMES utf8 */;

/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;

/*!40103 SET TIME_ZONE='+00:00' */;

/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;

/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;

/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;

/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--

-- Table structure for table `aprobadas`

--

DROP TABLE IF EXISTS `aprobadas`;

/*!40101 SET @saved_cs_client     = @@character_set_client */;

/*!50503 SET character_set_client = utf8mb4 */;

CREATE TABLE `aprobadas` (

  `id` bigint unsigned NOT NULL AUTO_INCREMENT,

  `solicitadas_id` bigint unsigned NOT NULL,

  `institution_id` bigint unsigned NOT NULL,

  `created_at` timestamp NULL DEFAULT NULL,

  `updated_at` timestamp NULL DEFAULT NULL,

  PRIMARY KEY (`id`),

  KEY `aprobadas_solicitadas_id_foreign` (`solicitadas_id`),

  KEY `aprobadas_institution_id_foreign` (`institution_id`),

  CONSTRAINT `aprobadas_institution_id_foreign` FOREIGN KEY (`institution_id`) REFERENCES `institutions` (`id`) ON DELETE CASCADE,

  CONSTRAINT `aprobadas_solicitadas_id_foreign` FOREIGN KEY (`solicitadas_id`) REFERENCES `solicitadas` (`id`) ON DELETE CASCADE

) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

/*!40101 SET character_set_client = @saved_cs_client */;
```

```
--
-- Table structure for table `areas`
--
DROP TABLE IF EXISTS `areas`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `areas` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `ARE_NOMBRE` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `ARE_DESCRIPCCION` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `areas_are_nombre_unique` (`ARE_NOMBRE`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `assigned_roles`
--
DROP TABLE IF EXISTS `assigned_roles`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `assigned_roles` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `user_id` int unsigned NOT NULL,
  `role_id` int unsigned NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `backgrounds`
--
DROP TABLE IF EXISTS `backgrounds`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `backgrounds` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `FON_NOMBRE` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `FON_NOMBRE2` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
```

```
  `institution_id` bigint unsigned NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `backgrounds_institution_id_unique` (`institution_id`),
  CONSTRAINT `backgrounds_institution_id_foreign` FOREIGN KEY (`institution_id`) REFERENCES `institutions` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `courses`
--
DROP TABLE IF EXISTS `courses`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `courses` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `CUR_NOMBRE` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `CUR_PARALELO` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `institution_id` bigint unsigned NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `courses_institution_id_foreign` (`institution_id`),
  CONSTRAINT `courses_institution_id_foreign` FOREIGN KEY (`institution_id`) REFERENCES `institutions` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `failed_jobs`
--
DROP TABLE IF EXISTS `failed_jobs`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `failed_jobs` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `connection` text COLLATE utf8mb4_unicode_ci NOT NULL,
  `queue` text COLLATE utf8mb4_unicode_ci NOT NULL,
  `payload` longtext COLLATE utf8mb4_unicode_ci NOT NULL,
  `exception` longtext COLLATE utf8mb4_unicode_ci NOT NULL,
  `failed_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```

```
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `institutions`
--
DROP TABLE IF EXISTS `institutions`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `institutions` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `INS_NOMBRE` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `INS_DIRECCION` text COLLATE utf8mb4_unicode_ci NOT NULL,
  `INS_TELEFONO` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `INS_CELULAR` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `INS_TIPO` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `INS_MISION` text COLLATE utf8mb4_unicode_ci,
  `INS_VISION` text COLLATE utf8mb4_unicode_ci,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `institutions_ins_nombre_unique` (`INS_NOMBRE`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `logos`
--
DROP TABLE IF EXISTS `logos`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `logos` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `LOG_NOMBRE` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `institution_id` bigint unsigned NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `logos_institution_id_unique` (`institution_id`),
  CONSTRAINT `logos_institution_id_foreign` FOREIGN KEY (`institution_id`) REFERENCES `institutions` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `migrations`
```

```
--
DROP TABLE IF EXISTS `migrations`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `migrations` (
  `id` int unsigned NOT NULL AUTO_INCREMENT,
  `migration` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `batch` int NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=33 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `password_resets`
--
DROP TABLE IF EXISTS `password_resets`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `password_resets` (
  `email` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `token` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  KEY `password_resets_email_index` (`email`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `people`
--
DROP TABLE IF EXISTS `people`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `people` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `PER_CEDULA` int NOT NULL,
  `PER_NOMBRES` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `PER_APELLIDOS` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `PER_SEXO` char(10) COLLATE utf8mb4_unicode_ci NOT NULL,
  `PER_FECHANACIMIENTO` datetime NOT NULL,
  `PER_TIPOSANGRE` char(10) COLLATE utf8mb4_unicode_ci NOT NULL,
  `PER_CORREO` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `PER_DIRECCION` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `PER_NUMERO` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
```

```
  `PER_CELULAR` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `institution_id` bigint unsigned NOT NULL,
  `area_id` bigint unsigned NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `people_per_cedula_unique` (`PER_CEDULA`),
  UNIQUE KEY `people_per_correo_unique` (`PER_CORREO`),
  KEY `people_institution_id_foreign` (`institution_id`),
  KEY `people_area_id_foreign` (`area_id`),
  CONSTRAINT `people_area_id_foreign` FOREIGN KEY (`area_id`) REFERENCES `areas` (`id`),
  CONSTRAINT `people_institution_id_foreign` FOREIGN KEY (`institution_id`) REFERENCES `institutions` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `photos`
--
DROP TABLE IF EXISTS `photos`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `photos` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `nombre` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `people_id` bigint unsigned NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `photos_people_id_unique` (`people_id`),
  CONSTRAINT `photos_people_id_foreign` FOREIGN KEY (`people_id`) REFERENCES `people` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `pictures`
--
DROP TABLE IF EXISTS `pictures`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `pictures` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `nombre` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `student_id` bigint unsigned NOT NULL,
```

```
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `pictures_student_id_unique` (`student_id`),
  CONSTRAINT `pictures_student_id_foreign` FOREIGN KEY (`student_id`) REFERENCES `students` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `roles`
--
DROP TABLE IF EXISTS `roles`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `roles` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `display_name` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `description` text COLLATE utf8mb4_unicode_ci,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `roles_name_unique` (`name`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `solicitadas`
--
DROP TABLE IF EXISTS `solicitadas`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `solicitadas` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `cedula` int NOT NULL,
  `tipo` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `institution_id` bigint unsigned NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `solicitadas_institution_id_foreign` (`institution_id`),
  CONSTRAINT `solicitadas_institution_id_foreign` FOREIGN KEY (`institution_id`) REFERENCES `institutions` (`id`) ON
DELETE CASCADE
```

```
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `students`
--
DROP TABLE IF EXISTS `students`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `students` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `EST_CEDULA` int NOT NULL,
  `EST_NOMBRES` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `EST_APELLIDOS` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `EST_SEXO` char(10) COLLATE utf8mb4_unicode_ci NOT NULL,
  `EST_FECHANACIMIENTO` datetime NOT NULL,
  `EST_TIPOSANGRE` char(10) COLLATE utf8mb4_unicode_ci NOT NULL,
  `EST_CORREO` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `EST_DIRECCION` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `EST_NUMERO` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `EST_CELULAR` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `EST_CODIGO` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `EST_MATRICULA` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `EST_INSCRITO` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `EST_NROMATRICULA` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `EST_RETIRADO` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `EST_BECA` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `course_id` bigint unsigned NOT NULL,
  `institution_id` bigint unsigned NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `students_est_cedula_unique` (`EST_CEDULA`),
  UNIQUE KEY `students_est_correo_unique` (`EST_CORREO`),
  KEY `students_course_id_foreign` (`course_id`),
  KEY `students_institution_id_foreign` (`institution_id`),
  CONSTRAINT `students_course_id_foreign` FOREIGN KEY (`course_id`) REFERENCES `courses` (`id`),
  CONSTRAINT `students_institution_id_foreign` FOREIGN KEY (`institution_id`) REFERENCES `institutions` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Table structure for table `users`
```

```
--
DROP TABLE IF EXISTS `users`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `users` (
  `id` bigint unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `email` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `email_verified_at` timestamp NULL DEFAULT NULL,
  `password` varchar(255) COLLATE utf8mb4_unicode_ci NOT NULL,
  `cedula` int NOT NULL,
  `remember_token` varchar(100) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `users_email_unique` (`email`),
  UNIQUE KEY `users_cedula_unique` (`cedula`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Dumping events for database 'desarrollo'
--
--
-- Dumping routines for database 'desarrollo'
--
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;
/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;
-- Dump completed on 2020-06-09 19:49:05
```

Glosario de términos

- **CRUD**

Son las 4 operaciones fundamentales de aplicaciones para crear, ver y editar, actualizar y eliminar registros mediante la aplicación en la base de datos.

- **Carné**

Este documento posee generalmente una fotografía del propietario y de información personal la cual le permite Identificarse dentro de un colectivo o para realizar cierta actividad, (Española, 2020).

- **Modelo Conceptual**

El modelo conceptual dentro de un desarrollo describe en alto nivel la estructura de la base de datos, la información se representa mediante objetos, relaciones y atributos, (Dario, s.f.).

- **Modelo Lógico**

Este modelo lógico representa una estructura de base de datos que puede almacenarse y procesarse en algún Sistema Gestor de Base de Datos, (Dario, s.f.).

- **Diagrama Físico**

Este modelo representa como funciona el diagrama dentro de la base de datos, este modelo indica todas las tablas, columnas, tipos de datos, restricciones y relaciones entre tablas, (Tecnologías Información, s.f.).

- **Diagrama entidad relación**

Es un diagrama de flujo que ilustra como las entidades de la base de datos se relacionan, es un diseño que se usa para depurar las bases de datos relacionales, son un reflejo de la estructura gramatical y emplea entidades como sustantivos y relaciones como verbos, (Lucidart, 2019).

- **Script**

Dentro de informática script se conoce como a un documento de texto con una seria de instrucciones o líneas de código que ejecutan alguna función dentro de una aplicación.

- **Xampp**

Es un software libre que gestiona bases de datos MySQL mediante el servidor web Apache y los intérpretes para lenguajes de programación como PHP y Perl, (Xampp, 2020).

- **Composer**

Composer es un software que gestiona paquetes necesarios para programar en PHP, provee de estándares y formatos para el manejo de dependencias y librerías de PHP, (Yair, 2019).

- **Node js**

Es un entono de ejecución en varias plataformas de código abierto, funciona principalmente en la capa del servidor basado en JavaScript, con una arquitectura orientada a eventos y basados en el moto V8 de Google, (JS, 2020).

- **Blade**

Blade es una herramienta del framework laravel el cual nos provee un motor de plantillas para modularizar las vistas del sistema, también permite usar las estructuras de control y variables de PHP directamente en ellas, (laravel-5, 2018).

- **Axios**

Es un cliente HTTP que se basa en promesas para ser ejecutadas por JavaScript, se utiliza dentro del backend y frontend, una petición Axios envía una petición a una operación REST y realiza operaciones CRUD, (Perú, 2019).

- **Responsive**

Es una técnica de diseño para frontend el cual permite que cualquier aplicación se pueda visualizar en varios dispositivos móviles adaptándose al ancho y alto de cada dispositivo, permite una correcta visualización de la aplicación y mejor experiencia de usuario, (40defiebre, 2019).

- **Rollback**

Hacer un rollback tiene distintos significados dentro de una aplicación, en base de datos hacer un rollback refiere a cancelar una transacción en curso y restaura los

datos a su estado anterior a comienzo de la transacción, dentro de la aplicación hacer un rollback refiere a restaurar una versión anterior del código.

- **GitHub**

Es una forja para alojar proyectos de programación o cualquier otro tipo de proyecto, utilizando un sistema de control de versiones, se utiliza principalmente para alojar código fuente de una aplicación, (GitHub, 2020).

- **Git bash**

Es una consola mediante la cual se puede ejecutar comandos principalmente de git, dentro de la consola se hacen todos los comandos necesarios para el control de versionamiento de git.

- **Branch**

Es una rama o apuntador local dentro de un proyecto en la cual se hacen los cambios del código fuente, los cuales requieren de una aprobación para que estos cambios estén reflejados dentro de la rama master la cual aloja el código fuente final que se usará.