Neiva, _**20 de octubre 2020**_

Señores

CENTRO DE INFORMACIÓN Y DOCUMENTACIÓN

UNIVERSIDAD SURCOLOMBIANA

Ciudad **Neiva – Huila**

El (Los) suscrito(s):

__**Jeferson de Jesus Gutierrez Zúñiga** _____, con C.C. No. _**1104873878**_____,

___**Jaime Sebastián Murcia Tovar**_____, con C.C. No. _**1075308202**_____,

_____, con C.C. No. _____,

_____, con C.C. No. _____,

Autor(es) de la tesis y/o trabajo de grado o _____

titulado_ **Plataforma Web "Sistema de ingresos de personal a la Universidad"**
_____

_____

_____

presentado y aprobado en el año __**2020**_____ como requisito para optar al título de **Ingeniero de Software** _____;

Autorizo (amos) al CENTRO DE INFORMACIÓN Y DOCUMENTACIÓN de la Universidad Surcolombiana para que, con fines académicos, muestre al país y el exterior la producción intelectual de la Universidad Surcolombiana, a través de la visibilidad de su contenido de la siguiente manera:

- Los usuarios puedan consultar el contenido de este trabajo de grado en los sitios web que administra la Universidad, en bases de datos, repositorio digital, catálogos y en otros sitios web, redes y sistemas de información nacionales e internacionales "open access" y en las redes de información con las cuales tenga convenio la Institución.

- Permita la consulta, la reproducción y préstamo a los usuarios interesados en el contenido de este trabajo, para todos los usos que tengan finalidad académica, ya sea en formato Cd-Rom o digital desde internet, intranet, etc., y en general para cualquier formato conocido o por conocer, dentro de los términos establecidos en la Ley 23 de 1982, Ley 44 de 1993, Decisión Andina 351 de 1993, Decreto 460 de 1995 y demás normas generales sobre la materia.

- Continúo conservando los correspondientes derechos sin modificación o restricción alguna; puesto que, de acuerdo con la legislación colombiana aplicable, el presente es un acuerdo jurídico que en ningún caso conlleva la enajenación del derecho de autor y sus conexos.

De conformidad con lo establecido en el artículo 30 de la Ley 23 de 1982 y el artículo 11 de la Decisión Andina 351 de 1993, "Los derechos morales sobre el trabajo son propiedad de los autores", los cuales son irrenunciables, imprescriptibles, inembargables e inalienables.

EL AUTOR/ESTUDIANTE:

Firma: _____

EL AUTOR/ESTUDIANTE:

SEBASTIÁN MURCIA.

Firma: _____

TÍTULO COMPLETO DEL TRABAJO: **Plataforma Web "Sistema de ingresos de personal a la Universidad Surcolombiana"**

AUTOR O AUTORES:

Primero y Segundo Apellido	Primero y Segundo Nombre
Gutierrez Zúñiga	Jeferson De Jesus
Murcia Tovar	Jaime Sebastián

DIRECTOR Y CODIRECTOR TESIS:

Primero y Segundo Apellido	Primero y Segundo Nombre
Rojas Rojas	Fernando

ASESOR (ES):

Primero y Segundo Apellido	Primero y Segundo Nombre

PARA OPTAR AL TÍTULO DE: Ingeniero de Software

FACULTAD: Ingeniería

PROGRAMA O POSGRADO: Ingeniería de Software

CIUDAD: Neiva **AÑO DE PRESENTACIÓN:** 2020 **NÚMERO DE PÁGINAS:** 34

TIPO DE ILUSTRACIONES (Marcar con una X):

Diagramas_X__ Fotografías___ Grabaciones en discos___ Ilustraciones en general__X_ Grabados___ Láminas___ Litografías___ Mapas___ Música impresa___ Planos___ Retratos___ Sin ilustraciones___ Tablas o Cuadros_X_

SOFTWARE requerido y/o especializado para la lectura del documento:

MATERIAL ANEXO:

PREMIO O DISTINCIÓN *(En caso de ser LAUREADAS o Meritoria)*:

PALABRAS CLAVES EN ESPAÑOL E INGLÉS:

Español	inglés	español	inglés
1. Programación web	Web Programing	6. _____	_____
2. _Node.js___	__Node.js___	7. _____	_____
3. _MongoDB__	__MongoDB__	8. _____	_____
4. _COVID-19___	_COVID-19__	9. _____	_____
5. Ingeniería de software	Software engineering	10. _____	_____

RESUMEN DEL CONTENIDO: (Máximo 250 palabras)

Se trata del desarrollo de un software específico para hacer eficiente la implementación de los protocolos de salud impuestos por la Universidad Surcolombiana tras la pandemia COVID-19 en busca de la adaptación a la nueva normalidad dada por la necesidad de reabrir las distintas sedes de la institución a lo largo del sur del país colombiano.

Esta plataforma que es de uso fácil e intuitivo para el usuario permite llevar un control y registro de las entradas y salidas de las personas a la institución por parte de los encargados previamente delimitados, así como también ingresar los datos requeridos por el ministerio de salud del país en relación a los síntomas esenciales para la detección del COVID-19 y de esta manera evitar la

propagación del virus y dar seguimiento a aquellas personas que presenten los síntomas en pro de evitar un nuevo brote en el país. El resultado obtenido luego de un proceso de desarrollo de software fue una Página Web que deben usar los encargados de las entradas de las instalaciones de la universidad. Toda esta data servirá además para reportar de manera periódica el tráfico en el establecimiento, reporte requerido obligatoriamente para la reapertura de instituciones a nivel nacional.

ABSTRACT: (Máximo 250 palabras)

It is about the development of an specific software to make more efficient the implementation of the health protocols imposed by the Southcolombian University after the COVID-19 pandemic in search of adaptation to the new normal given by the need to reopen the different headquarters of the institution all around the south of the Colombian country.

This platform is easy and intuitive for the user and also allows to keep control and record of the entrances and exits of people to the institution by the previously defined managers, as well as to enter the data required by the Ministry of Health of the country in relation to the essential symptoms for the detection of COVID-19 and in this way to avoid the spread of the virus and to follow up those people who present the symptoms in order to avoid a new outbreak in the country. The result after of a development process was a web site that it's pretended to be used by the managers on every entrance

of the university. All this data will also serve to periodically report the traffic in the establishment, a mandatory report required for the reopening of institutions nationwide.

APROBACION DE LA TESIS

Nombre Presidente Jurado: Fernando Rojas.

Firma: Rojas

Nombre Jurado: Eumpides Triana

Firma:

Nombre Jurado:

Firma:

Proyecto Plataforma Web

"Sistema de ingresos de personal a la Universidad Surcolombiana"

Jaime Sebastián Murcia Tovar y Jefferson de Jesús Gutiérrez Zúñiga.

Universidad Surcolombiana

Jaime Sebastián Murcia Tovar, Ingeniería de Software Universidad Surcolombiana.

Jefferson de Jesús Gutiérrez Zúñiga, Ingeniería de Software Universidad Surcolombiana.

Universidad Surcolombiana, Avenida Pastrana Borrero - carrera 1 Neiva - Huila.

Contacto: sebastianmurcia16@gmail.com – jepetoto18@gmail.com

Contenido

Tabla de Imágenes

1. Resumen

Se trata del desarrollo de un software específico para hacer eficiente la implementación de los protocolos de salud impuestos por la Universidad Surcolombiana tras la pandemia COVID-19 en busca de la adaptación a la nueva normalidad dada por la necesidad de reabrir las distintas sedes de la institución a lo largo del sur del país colombiano.

Se pretende que esta plataforma de uso fácil e intuitivo para el usuario permita llevar un control y registro de las entradas y salidas de las personas a la institución por parte de los encargados previamente delimitados, así como también ingresar los datos requeridos por el ministerio de salud del país en relación a los síntomas esenciales para la detección del COVID-19 y de esta manera evitar la propagación del virus y dar seguimiento a aquellas personas que presenten los síntomas en pro de evitar un nuevo brote en el país.

Toda esta data servirá además para reportar de manera periódica el tráfico en el establecimiento, reporte requerido obligatoriamente para la reapertura de instituciones a nivel nacional.

2. Línea de investigación

Ingeniería de Software – Desarrollo web

3. Problema

La llegada del COVID-19 al mundo y en especial a Colombia ha traído consigo diferentes retos que afrontar en los campos donde el ser humano se desarrolla, obligando así a cambiar las distintas formas y costumbres de trabajo y desarrollo de las personas. Como era de esperarse, esta pandemia lamentablemente también ha afectado al departamento del Huila y más concretamente la ciudad de Neiva la cual no se podía quedar atrás en desarrollo respecto a otros departamentos colombianos. En este orden de ideas, el sector de la educación fuertemente golpeado por esta pandemia y otros factores que se han fortalecido con el aislamiento social (poca conectividad, pobreza), se han tenido que idear diferentes formas de enseñanza, ya que el aprendizaje y la educación nunca paran, siendo así el desarrollo tecnológico el aliado más importante para las personas a la hora de controlar y contener esta pandemia que aqueja a la sociedad. La Universidad Surcolombiana, en busca de nueva normalidad en su labor educativa se ve en la necesidad de reabrir sus sedes en todo el sur colombiano, teniendo en cuenta que existen medidas de prevención sugeridas por el gobierno nacional y por la máxima autoridad en el campo de la salud, la OMS. En este sentido la universidad debe buscar la forma de seguir estos lineamientos y de crear métodos para contribuir al cuidado de la salud pública y también poder volver a una normalidad académica y administrativa para toda la familia Surcolombiana.

Pregunta de la investigación

¿Cómo se puede regular el ingreso y salida de personas de la Universidad Surcolombiana, además de tener un reporte diario de las personas que asisten a la universidad, aplicando la tecnología y el desarrollo de software?

4. Antecedentes

GLOBAL

Artículo: Corona-100m: Funciona con datos públicos del gobierno de Corea y datos brindados voluntariamente por los usuarios. Utiliza un sistema que se conoce como 'geofencing', que lanza un aviso al entrar en una zona delimitada en un mapa lo que permite detectar los lugares donde ha acudido una persona afectada por el virus, informando al resto de usuarios para evitar que pasasen por allí. La aplicación te puede informar de un caso detectado en un edificio en tal fecha, o de que en la zona hay un centenar de casos, o cualquier otro dato público disponible, siempre con una supervisión oficial detrás.

Autor: Ministerio de Salud y Bienestar de Corea del Sur

Ubicación: Seúl, Corea del Sur

Artículo: COVID-19MX: El gobierno de México a través de la Secretaría de Salud puso a disposición de sus habitantes un servicio para celulares en el que se ofrece información actualizada de la pandemia, además sirve para monitorear su estado de salud, guiar en un auto diagnóstico y obtener instrucciones ante un posible contagio. En caso de contagio pone a su servicio un número de emergencia al que se puede comunicar a través de la app y ubica los centros de salud cercanos dependiendo del tipo de servicio que ofrezcan los hospitales. Cuenta también con datos estadísticos, noticias y una sección de preguntas frecuentes.

Autor: Secretaría de Salud del Gobierno Federal

Ubicación: CDMX, México

Artículo: Google lanza en España su web con recursos e información sobre el COVID-19: La página de Google sobre el COVID-19 ofrece un ‘hub’; centralizado en el que se incluyen por un lado información pública del Gobierno de España y de la Organización Mundial de la Salud (OMS) sobre la prevención del COVID-19, información sobre los síntomas y los lugares oficiales para pedir ayuda. Las herramientas de información incluyen una alerta SOS en el buscador de Google para incluir noticias de última hora sobre el virus, así como información autorizada de la OMS para orientar en las búsquedas. Entre los recursos que se ofrecen se encuentran también otras herramientas útiles para profesores proporcionadas por Google para facilitarles que puedan seguir su día a día desde casa durante el confinamiento, como Enseña desde casa de GSuite y Aprende en casa de YouTube.

Autor: Google LLC

Ubicación: Madrid, España

COLOMBIA

CoronApp: De las Apps más descargadas en la historia del país, con más de 1 millón y medio de descargas, fue creada por la Agencia Nacional Digital. CoronApp hace parte de un grupo de aplicaciones desarrolladas como mecanismos de prevención, en donde la información ciudadana sirve para determinar cuáles son los focos epidemiológicos. También entrega cifras en tiempo real sobre casos de contagio, recuperados, puntos y líneas de atención, además de recomendaciones para su prevención. CoronApp está disponible de forma gratuita en las tiendas Android y iOS y entre sus principales características se encuentran en que no

consume datos. Finalmente, para acceder a toda la información, una vez descargada, debe hacerse un registro y aceptar los términos de uso.

Autor: Agencia Nacional Digital

Ubicación: Bogotá, Colombia

Artículo: coronaviruscolombia.gov.co: El Gobierno colombiano lanzó una página web donde se puede hacer un autodiagnóstico para descartar síntomas y evitar que se colapsen las líneas de atención o los hospitales con personas que realmente no tienen síntomas de Covid-19. Además, pone a disposición datos estadísticos y noticias actualizadas, así como los protocolos de prevención y protocolos de contagio, con líneas de atención las 24 horas del día y una sección de preguntas frecuentes.

Autor: Instituto Nacional de Salud, Gobierno de Colombia

Ubicación: Bogotá, Colombia.

5. Justificación

Desde la llegada del COVID-19 al mundo y a Colombia ha cambiado la forma de vivir de las personas. el contagio del virus ha crecido exponencialmente, por esta razón la Organización Mundial de la Salud promueve medidas de protección básicas contra el virus como el lavado de manos frecuentes y distanciamiento social, en Colombia se declaró la emergencia sanitaria mediante la Resolución 385 de 2020 de 12 de marzo de 2020. La situación actual del nuevo coronavirus en Colombia el 21 de agosto de 2020 en el reporte dado en la página web del ministerio de salud es de Casos Confirmados 522.138, casos activos 155.576, muertes: 16.568 y recuperados 348.940 en el departamento del Huila 3.325 casos confirmados.

Una de las medidas que establece el gobierno nacional es la Resolución No. 000666 del 24 de abril de 2020 del Ministerio de Salud y Protección Social donde se busca comprometer a los empleadores o contratantes para que tomen todas medidas posibles de bioseguridad en función de mitigar y controlar la pandemia del coronavirus en Colombia.

En la Universidad Surcolombiana se mencionan en el documento de MEDIDAS DE PREVENCIÓN Y PROTECCIÓN PARA PREVENIR EL CONTAGIO POR COVID -19 los protocolos que los funcionarios deben cumplir dentro y fuera de la universidad.

En el apartado 5 de dicho documento especifican 10 medidas para brindar espacios de trabajo sanos y seguros por parte de la comunidad universitaria para prevenir el contagio del COVID-19 en las medidas de prevención para ingresos y salida de los trabajadores se ve la necesidad de un sistema que facilite el control de ingreso y toma de datos de la comunidad universitaria, en consecuencia, el programa de ingeniería de software plantea una aplicación web de control de ingreso y salida (proporcionan todos los datos necesarios para el seguimiento del contagio del virus en caso dado.)

6. Objetivo general

Desarrollar e implementar una herramienta tecnológica para contribuir a los protocolos de bioseguridad propuestos por la universidad Surcolombiana para las Medidas de prevención en el ingreso y salida de comunidad universitaria.

Objetivos específicos

- Analizar datos, noticias, necesidades y decretos para la recolección y formación de los requerimientos necesarios.
- Diseñar el esquema de base de datos, dependencias de información que se utilizará y diferentes vistas que se van a tener en la aplicación web.
- Implementar tecnologías necesarias en el desarrollo de la aplicación web, sus requerimientos y funcionalidades.

8. Alcance

Se creó una herramienta en un entorno web donde las personas encargadas de vigilar el ingreso y salida de las personas de la universidad (estudiantes, docentes, administrativos, etc.) puedan ingresar diferentes datos requeridos para llevar un control sobre síntomas esenciales a la hora de detectar el COVID-19, con esta información se facilitaría hacerle seguimiento a las personas que directa o indirectamente fueron afectadas por un brote de contagios dentro de la Universidad. También se puede crear un reporte del historial de ingreso y salida de todas las personas, ya que el ministerio de Salud ha requerido esto para la reapertura de diferentes establecimientos a nivel nacional.

9. Limitaciones

Las limitaciones que se tuvieron en cuenta para este desarrollo son de tipo técnicas y de salud. En este sentido, se tuvo en cuenta los problemas que se dieron respecto al lenguaje de programación utilizado y los diferentes frameworks y librerías que se fueron usando, sumado a los diferentes entornos y versiones de navegadores donde se pruebe la herramienta web. Por otro lado, existe la limitación principal, en nuevo brote del COVID-19 la herramienta no cumpliría su propósito principal, ya que se extendería una cuarentena o aislamiento en la cuidad y no se permitiría la apertura de las instalaciones de la universidad.

10. Metodología

El proyecto se desarrolló bajo la metodología de ingeniería inversa, se partió con la generación de requisitos principales y construcción del programa y a medida que se avanzó en estos requisitos se fueron ideando y desarrollando otros. Se utilizaron varias tecnologías modernas y estables para el desarrollo web.

Esta herramienta se desarrolló con lenguaje de programación JavaScript, en backend y base de datos se utilizó NodeJS y MongoDB, para frontend se utilizó CSS y HTML5, Bootstrap, JQuery, varias librerías de JS como bcryptJS, Express (handlebars, session, validator, override), mongoose, passport y para la creación del informe en pdf se utilizó la librería pdfmake. Además, otras dependencias globales que se usaron como nodemon, dotenv y el entorno de instalación e inicialización y manejo de Node NPM.

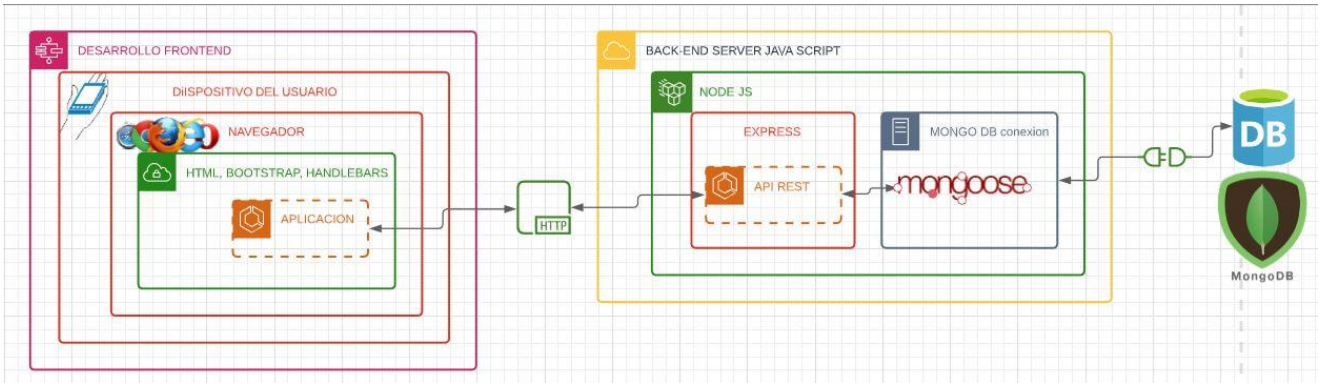


***Ilustración 1**. Modelo de Desarrollo (autoría Propia).*

La estructura de backend para este trabajo usando Node como entorno de ejecución de JavaScript fue de la siguiente manera. Primero fue necesario crear un servidor de Node donde se ejecutó el código JavaScript. Con Node Package Manager o Npm se instalaron todos módulos (librerías) necesarios para trabajar con Node.

La primera librería para montar el servidor es ExpressJs. Express proporcionó métodos de enrutamiento también gestión de sesiones y cookies. Para usar de manera más fácil estos métodos existen librerías como Express- session y Method-Override estas dos van muy de la mano cuando se manejan datos de usuarios en sesiones estos datos se envían a través de formularios que con Method Override se extienden y podemos usar métodos de envío de datos adicionales como por ejemplo PUT y DELETE. Otra librería de ExpressJs que se usa muy comúnmente es Express-Handlebars que sirve para integrar un motor de plantillas en Node y extender el código HTML del navegador. Con este módulo se pueden usar bucles y condicionales sobre HTML.

La información que recibe el servidor de los usuarios es necesario almacenarlo para garantizar su permanecía y persistencia dentro del sistema que se desarrolla, por esto es necesario elegir un gestor de bases de datos compatible con Node para desarrollos web. Se escogió MongoDB, una base de datos distribuida, basada en documentos y de uso general y diseñada para el desarrollo de aplicaciones modernas que ofrece un muy alto nivel de productividad. Es importante mencionar que mongo es una base de datos documental esto significa que almacena datos en documentos tipo JSON por esto es NoSQL. La librería que facilita la conexión y utilización de mongo en con Node se llama Mongoose. Mongoose proporciona una solución directa basada en esquemas (tablas en SQL) para modelar los datos de la aplicación. Incluye conversión de tipos integrada, validación, creación de consultas, enlaces de lógica y mucho más.

Los datos que el usuario proporciona y con los cuales se crea una cuenta para poder hacer uso de las funcionalidades de la plataforma que se desarrolló son procesados en el servidos

y guardados en la base de datos. El esquema que define que los tipos de datos y los objetos dentro de la base datos son hechos en Node con mongoose.

```
name: { type: String},
email: { type: String },
password: { type: String,},
date: { type: Date }
```

Para la seguridad de los datos en especial la contraseña de la cuenta se usa una librería de Express llamada bcryptJS la cual permite aplicar algoritmos para encriptar un campo con la función `bcrypt.genSalt(10)` la cual encripta 10 veces el String dado en este caso el campo password. Por lo tanto, la contraseña del usuario se guarda encriptada, al momento de usar su contraseña cuando se inicia una sesión con bcryptJS campara el campo de la base de datos y la contraseña que envía el usuario y retorna error si no coincide la información encriptada. Con esta explicación se toca la parte de autenticación de los usuarios que van usar la plataforma. Hoy en día en el desarrollo web hay muchas formas y métodos para hacer una autenticación, en muchas páginas implementan Google, Facebook, Apple como formas iniciar sesión si el usuario tiene una cuenta en estas plataformas o redes sociales. En vista que la universidad no cuenta con una API de inicio de sesión se optó por una autenticación local, con un registro muy sencillo.

El diseño e implementación de la interfaz con la el usuario va interactuar fue creada con código estándar de HTML, los estilos y colores fueron dados con uno de los frameworks más usados y versátiles del mundo, Bootstrap en su versión 4, todo trabajado sobre un motor de plantillas compatible con Node el cual es Handlerbars con posibilidades de ser escalado o migrado a otros motores de plantillas o frameworks como REACT o ANGULAR los cuales tiene funcionalidades y posibilidades de desarrollo de interfaces muy similares a

Handlerbars. Todo este entorno permite que la plataforma sea capaz de ser usada en cualquier entorno web, desde cualquier navegador compatible con estas tecnologías.

11. Desarrollo

Habiendo aclarado en la metodología algunos de los módulos importantes para el desarrollo de la plataforma seguiremos con el camino que se tomó para la creación es decir explicaremos como se usaron dichos módulos y los resultados que se obtuvieron.

El primero de los pasos que se siguieron para la creación fue el de la Configuración Del Servidor con NodeJS. Para esto fue necesario como se mencionó antes hacer uso de Node Package Manager o NPM para instalar las dependencias y las librerías necesarias. Esta es la lista de todas módulos o dependencias se puede miran en el archivo de nombre *Package.json* que es precisamente el archivo de configuración de módulos que se crea cuando se inicializa un proyecto de Node que se muestra en la ilustración 2.

Ilustración 2. *Módulos Package.json*

Una vez instaladas las dependencias se puede empezar a escribir código para el servidor. El primer paso fue crear un archivo de nombre *server.js* donde se van inicializar todas las dependencias, se configura el puerto local, se configuran las rutas de visualización, la configuración de middlewares y por último se definen las variables globales para enviar mensajes y respuestas entre las interfaces de usuario. Posterior es necesario crear un archivo de nombre *index.js* que ejecute el código del servidor y escuche el puerto definido para obtener respuestas del servidor. También este archivo ejecutara código que indicara si la base de datos está en línea. La conexión con la base de datos se hizo en un archivo llamado *database,js* donde se inicializa el módulo Moongose y se define una MongoDB_URI para hacer dicha conexión. Se debe definir un puerto y el nombre de la base de datos a la cual se quiere conectar. Hay que aclarar que la configuración en esta parte se debe hacer pensando en que el puerto en un futuro no será local si no dado por un servicio de host cuando la plataforma este en producción. Este es un ejemplo de la configuración de la conexión para una base de datos de forma local llamada *Notesdb,* mostrado en la Ilustración 3.

```
const mongoose = require("mongoose");

const { NOTES_APP_MONGODB_HOST, NOTES_APP_MONGODB_DATABASE } = process.env;

const MONGODB_URI = `mongodb://${
  NOTES_APP_MONGODB_HOST ? NOTES_APP_MONGODB_HOST : "localhost"
}/${NOTES_APP_MONGODB_DATABASE ? NOTES_APP_MONGODB_DATABASE : "notesdb"}`;

mongoose
  .connect(MONGODB_URI, {
    useNewUrlParser: true,
    useUnifiedTopology: true,
    useFindAndModify: false
  })
  .then((db) => console.log("Mongodb is connected to", db.connection.host))
  .catch((err) => console.error(err));
```

Ilustración 3. *Configuración de Conexión a bases de datos de Mongo con Moongose en Node.*

Hasta este punto del desarrollo tenemos que una carpeta llamada *AppUscoWeb* donde están contenidos los archivos creados para la configuración del servidor y de la base de datos de esta manera Ilustración 4.

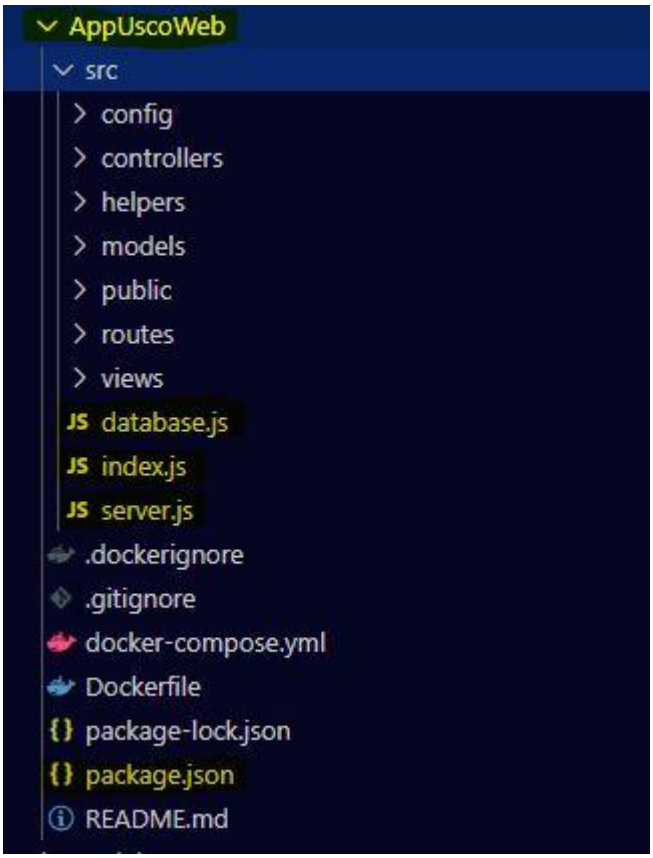


Ilustración.4. *Árbol de Directorios AppWebUsco.*

Para hacer las primeras visualizaciones de una interfaz o un hola mundo en la pantalla del usuario es necesario definir en el servidor unas RUTAS o *Routes* que se puedan ingresar en el navegador web. Las cuales fueron creadas en una carpeta de nombre *routes* dentro de una subcarpeta dentro de *AppUscoWeb* llamada *src*. La configuración de las *routes* en el archivo del servidor puede ven en la porción de código mostrada en la Ilustración 5.

```
// routes
app.use(require('./routes/index.routes'));
app.use(require('./routes/users.routes'));
app.use(require('./routes/visitas.routes'));
app.use(require('./routes/pdfMake.routes'));
```

Ilustración 5. *Configuración routes, archivo del servidor.*

Los archivos de configuración de las rutas contienen el código que renderiza las VISTAS o *views* disponibles para el usuario y también recibe y procesa los datos enviados a través de esas vistas por los usuarios.

Los archivos que gestionan las vistas y todo su diseño son configurados con el motor de plantillas instalado en las dependencias. El cual permite extender el código HTML y facilitar la forma de imprimir los resultados de las consultas hechas por el usuario. El motor de plantillas se configura en el archivo del servidor donde se cambia la extensión de los archivos de las vistas de *.HTML* a *.hbs* que corresponde a Handelbars como se puede ver en la fracción de código de la Ilustración 6.

```
// settings
app.set('port', process.env.PORT || 4000);

app.set('views', path.join(__dirname, 'views'));
app.engine('.hbs', exphbs({
  defaultLayout: 'main',
  layoutsDir: path.join(app.get('views'), 'layouts'),
  partialsDir: path.join(app.get('views'), 'partials'),
  extname: '.hbs'
}));
app.set('view engine', '.hbs');
```

Ilustración 6. *Configuración de Archivos de wiews*

Los archivos que se crearon para contener todo el código de diseño de las vistas están en una carpeta dentro de *src* llamada *views*

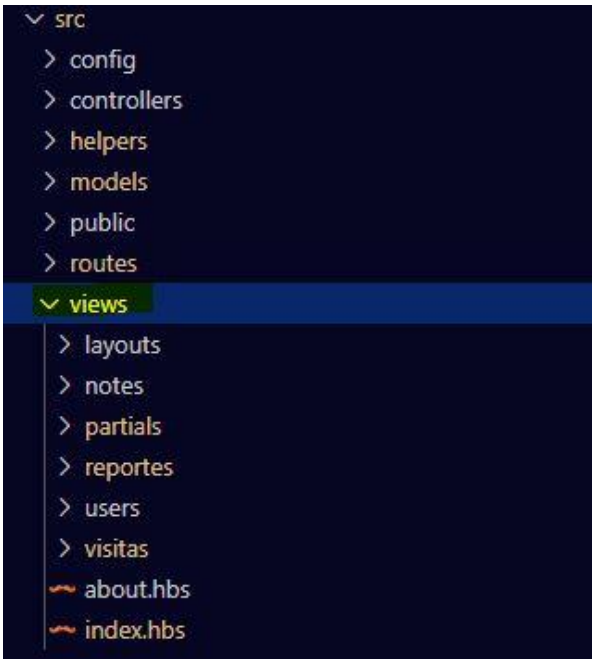


Ilustración 7. *Árbol de directorios.*

La carpeta *layouts* contiene el archivo con el código del cuerpo principal de la página. Es decir, el código HTML que es común en todas las *views* y donde se muestran los demás *módulos de views* que se crearon para la plataforma como el header o la navegación de la página el cual se muestra a continuación.

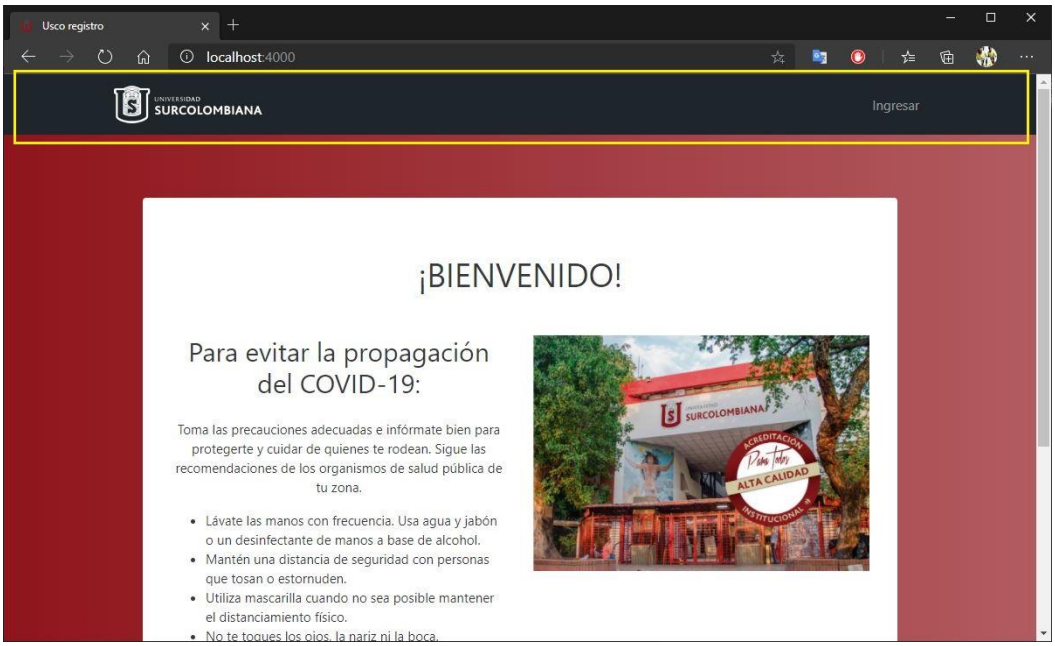


Ilustración 8*. Header (encabezado de la página)*

Luego de iniciar sesión en la plataforma cambia en menú de la navegación donde ya se pueden ver todas funciones con las que dispone la página.

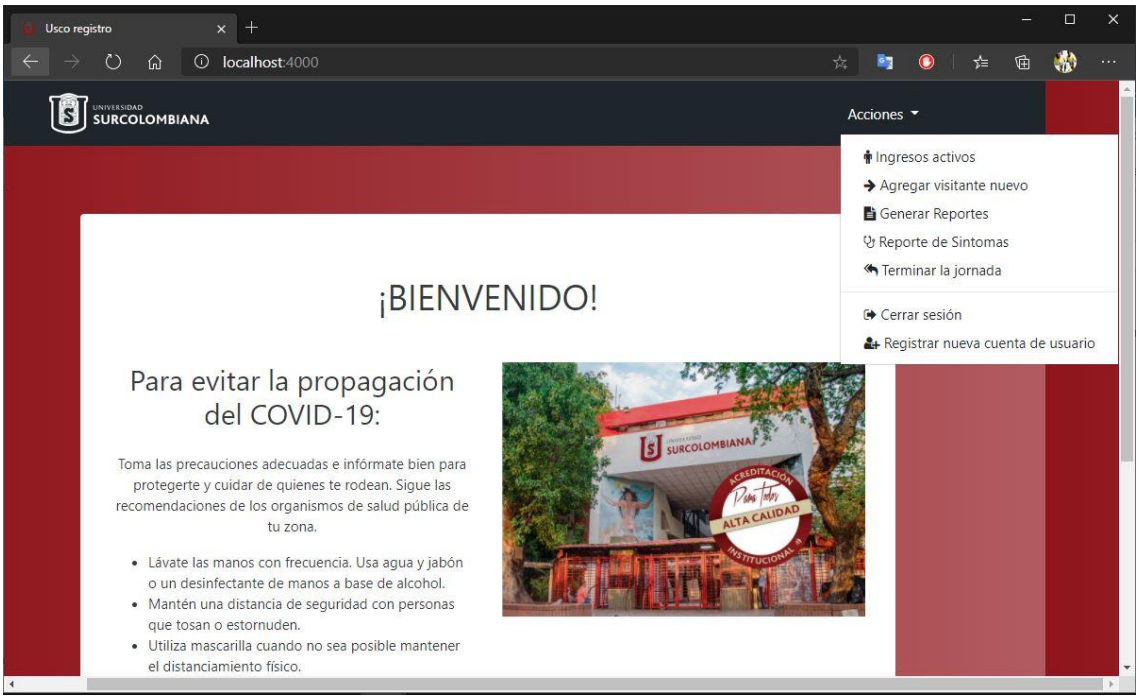


Ilustración 9. *Menú de Acciones (Menú de la página.).*

El enlace de agregar visitante nuevo lleva a una vista con un formulario que autoriza la entrada de la persona que proporciona los datos, a la universidad Surcolombiana. Esta información que el sistema recibe es validada por el servidor y luego guardada en la base datos como un registro dentro de un documento. En la base de datos de mongo como se mencionó anteriormente las tablas son llamados documentos y sus registro o campos pueden ser llamados objetos precisamente porque tienen un formato de objeto JSON.

El formulario de ingreso obedece a un modelo que recibe los datos ingresados en el navegador.

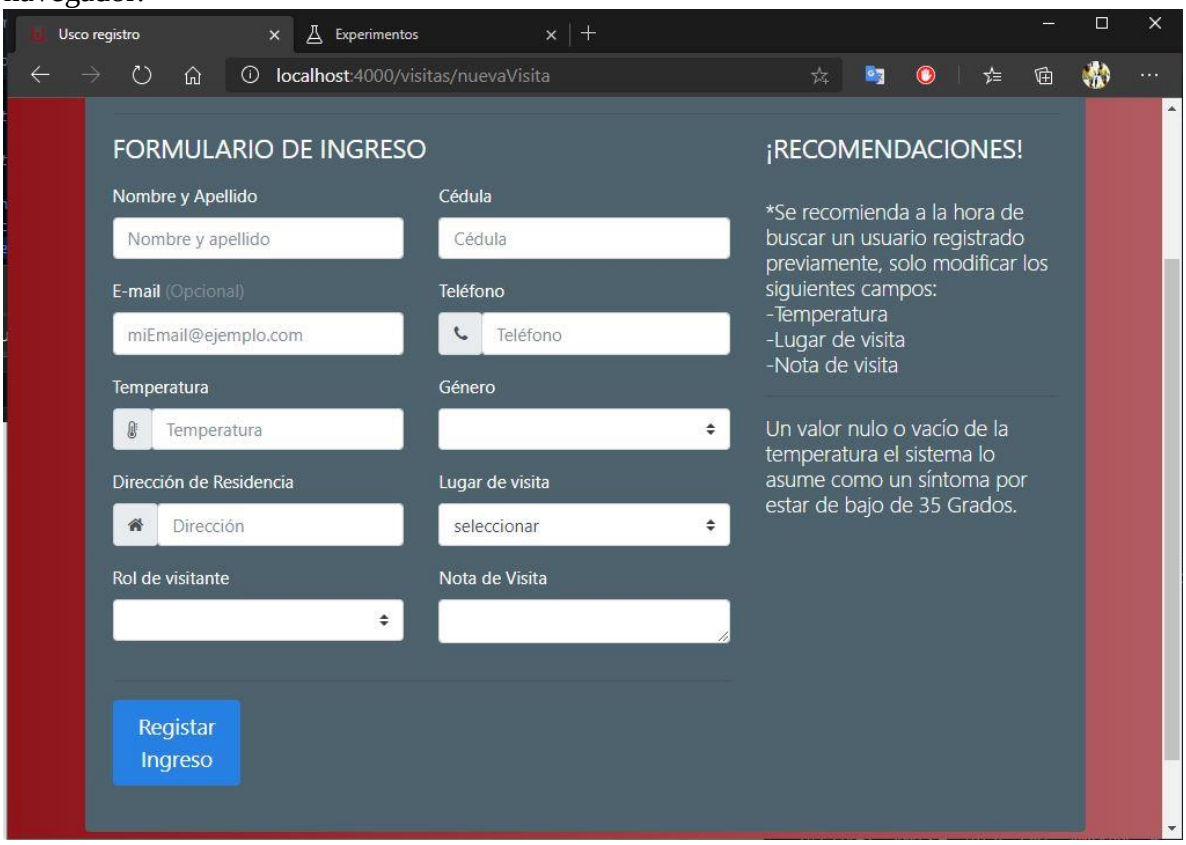


Ilustración 10. *Formulario de ingreso.*

El modelo o Schema es la equivalencia de las tablas cuando se habla de modelos relacionales.

Para esta plataforma y sus funcionalidades fue necesario crear varios Schemas para el manejo de las cuentas de usuarios y el historial de ingreso y de salidas de los visitantes.

este es el Schema de usuarios del sistema, Ilustración 11.

```
const { Schema, model } = require("mongoose");

const bcrypt = require("bcryptjs");

const UserSchema = new Schema({
  name: { type: String, required: true },
  email: { type: String, required: true },
  password: { type: String, required: true },
  date: { type: Date, default: Date.now }
});


module.exports = model("User", UserSchema);
```

Ilustración 11. *UserSchema Modelo de MongoDB.*

Schema de información de los visitantes si aprecia en la ilustración 12.

```
const { Schema, model } = require("mongoose");

const VisitaSchema = new Schema(
{
    nombres: { type: String, required: true },
    cedula: { type: Number, required: true },
    email: { type: String, required: true },
    telefono: { type: Number, required: true },
    temperatura: { type: Number, required: true },
    genero: { type: String, required: true },
    direccion: { type: String, required: true },
    lugarVisita: { type: String, required: true },
    nota: { type: String, default:'sin Notas'},
    Rol_visitante: { type: String, default:'no especificado'},
    date: { type: Date, default: Date.now },
});

module.exports = model("Visita", VisitaSchema);
```

Ilustración 12.*VistasSchema Modelo MongoDB.*

Schema de visitantes que se les permitió la entrada y marcaron una salida o fin de la visita a la universidad se guarda su información junto con la fecha y la hora en la cual se registró esta salida. El modelo de Schema se puede ver en la siguiente Ilustración.

```
const { Schema, model } = require("mongoose");

const VisitaSalienteSchema = new Schema(
{
    nombres: { type: String, required: true },
    cedula: { type: Number, required: true },
    email: { type: String, required: true , default:'Email vacio'},
    telefono: { type: Number, required: true },
    temperatura: { type: Number, required: true },
    genero: { type: String, required: true },
    direccion: { type: String, required: true },
    lugarVisita: { type: String, required: true },
    nota: { type: String, default:'sin Notas'},
    Rol_visitante: { type: String, default:'no especificado'},


    dateEntrada: { type: Date, required: true},
    dateSalida: { type: Date, default: Date.now },
});

module.exports = model("VisitaSaliente", VisitaSalienteSchema);
```

Ilustración 13. *VisitasSalientesSchema Modelo MongoDB.*

Node y Moongose permiten crear estos modelos y guardas la información que reciben en MongoDB los cuales se ven así en el gestor de la base de datos

***Ilustración 14**. Datos en MongoDB.*

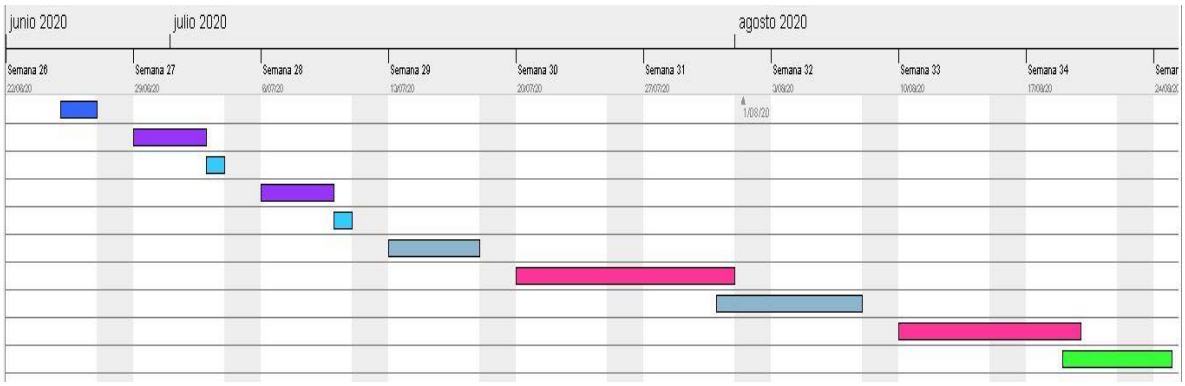
Ilustración 15. *Datos en MongoDB*

"Un registro en MongoDB es un documento, que es una estructura de datos compuesta de pares de campos y valores. Los documentos de MongoDB son similares a los objetos JSON. Los valores de los campos pueden incluir otros documentos, matrices y matrices de documentos." [1]

Por estas razones están grande su versatilidad al momento de almacenar todo tipo de registros.

12. Cronograma

Nombre	Fecha de inicio	Fecha de fin
Lluvia de ideas	25/06/20	26/06/20
Levantamiento de Requisitos	29/06/20	2/07/20
Levantamiento de Informacion(antecedentes e informacion)	3/07/20	3/07/20
Diseño de Interfaces	6/07/20	9/07/20
Diseño de Formularios	10/07/20	10/07/20
Diseño de modelos Bases Datos	13/07/20	17/07/20
Creacion de interfaces y formularios	20/07/20	31/07/20
Desarrollo e implementacion de Funcionalidades	31/07/20	7/08/20
Validaciones y Pruebas	10/08/20	19/08/20
Correccion de errores y pruebas	19/08/20	24/08/20



13. Costos y presupuesto

El proyecto es un desarrollo con motivo social y académico, en consecuencia, no tiene ningún presupuesto monetario asignado. Los costos son de tiempo, recursos humanos y tecnológicos.

Recursos	**Costos por unidad**	**Costo total**
1er mes		
Dos ingenieros de software (desarrolladores)	$ 2.500.000	$ 5.000.000
Dos Computadores para desarrollo con procesador intel core i7, 1tb almacenamiento y 16 gb de ram	$ 3.000.000	$ 6.000.000
Fase 1: levantamiento de requisitos y requerimientos	$ 4.000.000	$ 4.000.000
2do mes		
Dos ingenieros de software (desarrolladores)	$ 2.500.000	$ 5.000.000
Fase 2: Desarrollo de requisitos y requerimientos, programación de ítems en página web.	$ 6.000.000	$ 6.000.000
3er mes		
Dos ingenieros de software (desarrolladores)	$ 2.500.000	$ 5.000.000
Fase 3: Pruebas de seguridad y validación de campos, test con	$ 6.000.000	$ 6.000.000
Costo final		**$ 37.000.000**

14. Conclusiones

Las herramientas tecnologías desde hace mucho tiempo vienen ayudando en la vida cotidiana de las personas, dando muchas facilidades y formas de agilizar varias de las actividades que realizan a diario, por ejemplo, en sus trabajos, en sus estudios, en la forma de comunicarse, entre otras. El surgimiento del Covid-19 fortaleció el uso de las herramientas de software, una de las más importantes y relevantes a la hora de combatir este virus son las diferentes plataformas desarrolladas en muchas partes del mundo para implementar protocolos de bioseguridad. En la universidad Surcolombiana la plataforma que se creó con el fin de implementar el "*PROTOCOLO PARA LA PREVENCIÓN Y PROTECCIÓN PARA CONTENER LA INFECCIÓN RESPIRATORIA AGUDA POR COVID-19*"[7] cumple con su principal función que es llevar el control al momento del ingreso y de la salida de las instalaciones, agiliza la manera en la que se toman los datos de las personas y también facilitar la generación de reportes de las visitas realizadas a las instalaciones Surcolombianas, es un gran paso para seguir construyendo el camino hacia la vida normal antes del Covid-19.

15. Bibliografía

[1] I. 2008-present. M. MongoDB, “Introduction to MongoDB — MongoDB Manual.” https://docs.mongodb.com/manual/introduction/ (accessed Oct. 10, 2020).

[2] MINISTERIO DE SALUD Y PROTECCIÓN SOCIAL, “ABECÉ Protocolos de bioseguridad (Resolución 666 del 24 de abril de 2020),” Apr. 2020. Accessed: Sep. 08, 2020. [Online]. Available: https://bit.ly/3hU5JVk.

[3] MINISTERIO DE SALUD Y PROTECCIÓN SOCIAL, “RESOLUCIÓN NÚMERO 385 DEL 12 DE MARZO DE 2020,” COLOMBIA, Mar. 2020. Accessed: Sep. 08, 2020. [Online]. Available: https://www.minsalud.gov.co/sites/rid/Lists/BibliotecaDigital/RIDE/DE/DIJ/resolucion-385-de-2020.pdf.

[4] A. Mardan, *Express.js Deep API Reference*. Apress, 2014.

[5] J. F. (Perú), C. O. (Colombia), A. R. (Bolivia), H. M. (Brasil), A. B. y D. R. (México). EL PAÍS en América Latina. Mar Centenera (Argentina), “Varios países de Latinoamérica lanzan aplicaciones para el autodiagnóstico del coronavirus | Verne México EL PAÍS,” Apr. 07, 2020. https://verne.elpais.com/verne/2020/03/24/mexico/1585084400_954679.html (accessed Sep. 08, 2020).

[6] MINISTERIO DE SALUD Y PROTECCIÓN SOCIAL, “RESOLUCIÓN NÚMERO 000666 DE 2020,” Apr. 2020. Accessed: Sep. 08, 2020. [Online]. Available: https://id.presidencia.gov.co/Documents/200424-Resolucion-666-MinSalud.pdf.

[7] UNIVERSIDAD SURCOLOMBIANA GESTIÓN DE SEGURIDAD Y SALUD EN EL TRABAJO, “PROTOCOLO PARA LA PREVENCIÓN Y PROTECCIÓN PARA CONTENER LA INFECCIÓN RESPIRATORIA AGUDA POR COVID-19 CÓDIGO EV-SST-DA-05 VERSIÓN 1 VIGENCIA 2020 PÁGINA Página 1 de 40 PROTOCOLO PARA LA PREVENCIÓN Y PROTECCIÓN PARA CONTENER LA INFECCIÓN RESPIRATORIA AGUDA POR COVID-19,” NIEVA, 2020.

[8] C. B. Ivan Watson and Sophie Jeong, “Coronavirus mobile apps are surging in popularity in South Korea - CNN,” Feb. 28, 2020.

[9] Ministerio de Tecnologías de la Información y las Comunicaciones de Colombia, “Así funciona CoronApp Colombia, aplicación para detectar y monitorear casos de covid-19,” *Sala prensa MinTic en los medios*, Apr. 2020, Accessed: Sep. 11, 2020. [Online]. Available: http://www.mintic.gov.co/portal/604/w3-article-135648.html.

[10] Ministerio de salud, “El Coronavirus en Colombia,” 2020. https://coronaviruscolombia.gov.co/Covid19/index.html (accessed Sep. 11, 2020).

[11] A. Rafique, D. Van Landuyt, B. Lagaisse, and W. Joosen, “On the Performance Impact of Data Access Middleware for NoSQL Data Stores A Study of the Trade-Off between Performance and Migration Cost,” doi: 10.1109/TCC.2015.2511756.

[12] J.-J. Quisquater and G. Avoine, “Passport Security,” in *Encyclopedia of Cryptography and Security*, Springer US, 2011, pp. 913–916.

[13] *Encyclopedia of Cryptography and Security*. Springer US, 2011.

[14] M. M. Patil, A. Hanni, C. H. Tejeshwar, and P. Patil, “A qualitative analysis of the

performance of MongoDB vs MySQL database based on insertion and retriewal operations using a web/android application to explore load balancing-Sharding in MongoDB and its advantages," in *Proceedings of the International Conference on IoT in Social, Mobile, Analytics and Cloud, I-SMAC 2017*, 2017, pp. 325–330, doi: 10.1109/I-SMAC.2017.8058365.

[15] A. Mardan and A. Mardan, "Working with Middleware," in *Express.js Deep API Reference*, Apress, 2014, pp. 21–48.

[16] C. J. Ihrig, "Databases," in *Pro Node.js for Developers*, Berkeley, CA: Apress, 2013, pp. 217–232.

[17] C. J. Ihrig and C. J. Ihrig, "JavaScript Object Notation," in *Pro Node.js for Developers*, Apress, 2013, pp. 263–270.

[18] C. J. Ihrig and C. J. Ihrig, "The Node Module System," in *Pro Node.js for Developers*, Apress, 2013, pp. 9–27.

[19] C. J. Ihrig and C. J. Ihrig, "The Node Programming Model," in *Pro Node.js for Developers*, Apress, 2013, pp. 29–44.

[20] C. J. Ihrig and C. J. Ihrig, "The Express Framework," in *Pro Node.js for Developers*, Apress, 2013, pp. 189–204.

[21] C. J. Ihrig, *Pro Node.js for Developers*. Apress, 2013.

[22] A. Mardan and A. Mardan, "Template Engines and Consolidate.js," in *Express.js Deep API Reference*, Apress, 2014, pp. 49–61.