Diseño e implementación de una aplicación web para el agendamiento de visitas de instalación en la empresa Follow Up

Jhon Stiven Diaz Santa
Jhon.diazssa@cun.edu.co
Corporación unificada nacional de educación superior (CUN).

***Resumen*— El presente documento expone una vista general y un panorama claro sobre el diseño y desarrollo de la aplicación web para el agendamiento de visitas de instalación en la empresa Follow Up, compartiendo las metodologías, técnicas y lenguajes aplicados durante el proceso de desarrollo de la misma.**

Palabras clave: SCRUM, NODEJS, BDD, Back-End, Front-End y Backlog.

I. INTRODUCCIÓN

En la actualidad, la gestión de visitas de instalación puede ser un proceso tedioso debido a la cantidad de información que debe manejarse y los diversos inconvenientes que pueden presentarse a la hora de ingresar la información y ejecutar la instalación de forma correcta. Por lo cual el presente documento expone y explica la solución empleada para poder ingresar la información de una forma correcta, disminuyendo la cantidad de complicaciones que este proceso pueda emplear.

El presente documento se enfocará en el diseño e implementación de esta aplicación en el área de operaciones de la empresa. De la herramienta, se espera que permita a la empresa mejorar la satisfacción del cliente y aumentar la productividad al reducir el tiempo y esfuerzo dedicados a la gestión de instalaciones.

A. *Planteamiento del problema.*

La empresa FollowUp se dedica a instalación de sensores de conteo y equipos de seguimiento en diversas ubicaciones. Como parte de sus operaciones, FollowUp debe gestionar eficazmente el agendamiento de las visitas de instalación de sensores de conteo en los distintos lugares de Colombia. Actualmente esta gestión se realiza principalmente a través de un documento en formato Excel. Este está almacenado en drive y se comparte a todos los integrantes del área de operaciones sin importar el rol que ejecute el individuo. Lo anterior contiene varios desafíos y limitaciones que afectan la eficiencia y calidad del servicio que brinda la empresa. A continuación, se presentan el subconjunto de problemas que se derivan de la ineficiencia y poca calidad del servicio:

- *Pérdida de Datos y Errores Manuales*: La dependencia de hojas de cálculo manuales aumenta el riesgo de pérdida de datos y errores humanos en la programación de visitas. Esto puede resultar en insatisfacción del cliente y costos adicionales para corregir los errores.

- *Ineficiencia en la Gestión de Agendas*: El proceso actual de agendamiento de visitas se realiza mediante el ingreso de información a un archivo Excel, lo que dificulta la coordinación de horarios y recursos. Esto puede resultar en solapamientos, donde al estar usando simultáneamente el archivo y guardar la información genera que se pise la información, retrasos y una asignación ineficiente de personal y recursos.

- *Falta de Acceso Centralizado*: Los datos de agendamiento no están centralizados, lo que hace difícil el acceso a la información al área de operaciones. Esto puede llevar a malentendidos y confusiones en la programación de visitas.

- *Dificultad en la Comunicación Cliente-Empresa*: La falta de un sistema de agendamiento centralizado dificulta la comunicación efectiva entre la empresa y sus clientes, lo que puede llevar a malentendidos, falta de transparencia y una experiencia insatisfactoria para los clientes.

- *Imposibilidad para integrar datos*: Es posible que FollowUp utilice otros sistemas de gestión empresarial o bases de datos para otros aspectos de su operación. La falta de integración con el sistema de agendamiento actual puede generar duplicación de datos y falta de coherencia. Una aplicación web podría integrarse con otros sistemas para una gestión más efectiva.

- *Falta de Respaldo y Recuperación de Datos*: El uso de una hoja de cálculo única para gestionar información crítica puede representar un riesgo en términos de pérdida de datos debido a errores humanos o problemas técnicos. Una aplicación web ofrecería una mejor gestión de respaldo y recuperación de datos para garantizar la integridad de la información.

Por lo tanto, la implementación de una aplicación web para el agendamiento de visitas de instalación de sensores de conteo, evitara los problemas anteriormente mencionados, logrando mayor eficiencia y generando calidad en la información.

B. *Justificación*

Debido al constante cambio tecnológico y la necesidad de mantener la integridad de la información se presentan una serie de razones fundamentales que respaldan la necesidad de poner en marcha el desarrollo y la implementación de una aplicación web para el agendamiento de visitas de instalación de sensores de conteo, para el área de operaciones de la empresa FollowUp. La implementación de una aplicación web dedicada para el agendamiento de visitas de instalación permitirá a la empresa

Followup automatizar y agilizar significativamente sus procesos de programación. Esto resultará en una gestión más eficiente de los recursos y tiempos, reduciendo costos operativos y mejorando la productividad general.

Actualmente en el mercado no hay una aplicación que permita gestionar las visitas de instalación de sensores de conteo.

El proyecto en cuestión ofrece la posibilidad de modificar y adaptar el sistema de información de acuerdo las necesidades y nuevos tipos de datos que el cliente tenga durante el desarrollo del área de operaciones.

Una aplicación web proporcionará un entorno controlado para el ingreso y almacenamiento de datos relacionados con las visitas de instalación. Esto minimizará la posibilidad de errores humanos y garantizará la integridad de la información crítica, evitando pérdidas de datos y visitas mal programadas.

Al brindar a los clientes finales una plataforma en línea para ingresar y confirmar visitas de instalación, la empresa mejorará la experiencia del cliente al hacer que el proceso sea más transparente y accesible.

La aplicación web permitirá a Followup adaptarse con mayor facilidad a las cambiantes necesidades y demandas del mercado. Podrá gestionar mejor las solicitudes de cambios de horario y reasignaciones de recursos, lo que es fundamental en un entorno empresarial dinámico.

El aplicativo en cuestión centralizará todos los datos relacionados con las visitas de instalación en un solo lugar, lo que facilitará el acceso y la gestión de la información por parte de los empleados y directivos de la empresa. Esto contribuirá a una toma de decisiones más informada y estratégica.

Los datos estarán estructurados en una BDD la cual estará almacenada en el servidor para que el personal autorizado pueda acceder y obtener la información necesaria para realizar algunos análisis o integrar estos datos con otras aplicaciones.

La herramienta proporcionará un medio efectivo de comunicación interna entre los técnicos, el personal administrativo y de gestión, lo que reducirá la confusión y los malentendidos relacionados con las programaciones de visitas.

La aplicación web permitirá un mayor control y seguimiento de las visitas de instalación programadas, lo que facilitará la supervisión en tiempo real del progreso de las instalaciones y la capacidad de tomar medidas correctivas de manera oportuna. Con una solución basada en web, Followup estará mejor preparada para manejar su crecimiento futuro y expandirse a nuevas ubicaciones geográficas sin las limitaciones que impone un sistema manual.

La adopción de tecnología avanzada para la gestión de visitas de instalación puede mejorar la competitividad de Followup en el mercado al ofrecer un servicio más eficiente y profesional en comparación con competidores que aún utilizan métodos manuales.

C. Objetivo General

Desarrollar e implementar una aplicación web altamente funcional y personalizada, diseñada específicamente para la gestión integral y automatización del proceso de agendamiento de visitas de instalación de sensores y dispositivos de seguimiento para la empresa Followup. La aplicación deberá proporcionar un sistema de agendamiento en tiempo real, permitiendo la asignación eficiente de recursos, la comunicación efectiva con los clientes, la capacidad de realizar cambios y reprogramaciones de manera ágil, así como la recopilación de datos para análisis y mejora continua. El objetivo es mejorar sustancialmente la eficiencia operativa de la empresa, reducir errores en el agendamiento, mejorar la satisfacción del cliente y mantener la competitividad en el mercado, respaldado por una solución tecnológica robusta y escalable. Para llevar a cabo el cumplimiento del objetivo mencionado, se implementará la metodología Ágil SCRUM.

D. Objetivos Especificos

Fase 1:

- Crear producto Backlog. (Lista de requisitos y funcionalidades para la aplicación de lista de tareas)

Fase 2: Sprint 1

- Desarrollar interfaz de inicio de sesión de cliente final.

Fase 3: Sprint2

- Desarrollar interfaz para el ingreso de datos de las visitas de instalación de sensores de conteo.

Fase 4: Sprint3

- Creación de Base de datos para el almacenamiento de información sobre las visitas de instalación de sensores de conteo.

Fase 5: Sprint4

- Desarrollar interfaz para la edición o eliminación de los datos ingresados para las visitas de instalación de sensores de conteo.

Fase 6: Sprint5

- Desarrollo back-end de inicio de sesión, adición de información y edición de información.

Fase 7: Sprint6

Creación de manual de usuario.

II. METODOLOGIA

A partir de esta sección, se desarrollan los contenidos del tema, de una forma ordenada y secuencial. Nótese que la sección debe ir organizada usando títulos como el anterior para cada tema nuevo incluido. Aparte, se incluyen subtítulos comoel siguiente.

A. Metodología SCRUM:

SCRUM es un marco de trabajo ágil utilizado para gestionar proyectos, especialmente en el desarrollo d software, pero también se aplica en otros contextos. Su objetivo principal es mejorar la productividad, la calidad y la satisfacción del cliente a través de un enfoque colaborativo y adaptable. Scrum se basa en varios principios y roles clave:

Roles en Scrum:

- *Product Owner:* Es responsable de definir y priorizar los elementos del producto en el Backlog del Producto. Debe representar los intereses del cliente final y tomar decisiones sobre qué funcionalidades deben incluirse en el producto.
- *Scrum Master:* Es el facilitador del equipo Scrum. Su función principal es eliminar obstáculos y garantizar que el equipo siga los principios de Scrum. No es un jefe ni un gerente, sino un coach.
- *Equipo de Desarrollo:* Este equipo es responsable de crear el producto. Debe ser autoorganizado y multidisciplinario, lo que significa que debe incluir todas las habilidades necesarias

para llevar a cabo el trabajo.

Artefactos en Scrum:

- *Product Backlog:* Es una lista priorizada de todas las funcionalidades, mejoras y tareas que deben realizarse en el proyecto. El Product Owner es responsable de mantener y priorizar este backlog.
- *Sprint Backlog:* Para cada iteración, conocida como Sprint (usualmente de 2 a 4 semanas), el equipo selecciona un conjunto de elementos del Product Backlog para trabajar. Estos elementos se convierten en las tareas a realizar durante ese Sprint.
- *Incremento:* Es el resultado del trabajo del equipo al final de cada Sprint. Debe ser una versión potencialmente entregable del producto con un conjunto de funcionalidades completas y probadas.

Eventos en Scrum:

- *Sprint Planning*: Al comienzo de cada Sprint, el equipo se reúne para seleccionar elementos del Product Backlog y planificar cómo los implementarán durante el Sprint.
- *Daily Scrum:* Es una reunión diaria de 15 minutos en la que el equipo actualiza su progreso y discute cualquier obstáculo que esté impidiendo su avance.
- *Sprint Review:* Al final de cada Sprint, el equipo muestra el Incremento al Product Owner y otros interesados para obtener retroalimentación y discutir los próximos pasos.
- *Sprint Retrospective*: Después de la Sprint Review, el equipo realiza una retrospectiva para analizar lo que funcionó y lo que no funcionó en el Sprint y definir mejoras para el próximo Sprint.

B. UML de secuencia.

Por medio de este se explicarán los componentes en la arquitectura del sistema, el cual nos permite mostrar relaciones de asociación. Adicionalmente nos permite tener un panorama claro del sistema de información.

III. RESULTADOS

Fase 1: Crear producto Backlog. (Lista de requisitos y funcionalidades para la aplicación de lista de tareas)

Para la creación del producto Backlog programo una reunión con los integrantes que están a cargo de las visitas de instalación, para poder entender completamente la necesidad y las sugerencias que se desean emplear en el diseño e implementación de una aplicación para el agendamiento de visitas de instalación de sensores en la empresa Follow Up.

Se emplea un formato el cual contiene los siguientes campos:

- *N' Sprint:* El cual indica el número del sprint.
- *N':* El cual indica el numero de la Épica
- *Historia de usuario:* Este campo indica las sugerencias y necesidades con las que cuenta el usuario.
- *N' de tarea:* Este campo indica el número de tareas para solucionar la Épica.
- *Tareas:* Es te campo se emplea para describir la tarea asociada a la historia de usuario.
- *Estimación en horas:* Este campo indica el número de horas necesarias para desarrollar y completar la tarea.

A continuación, se presentará el roduct Backlog empleado con los campos anteriormente mencionados:

Product Backlog

N' Sprint	N'	Historia de usuario	N' de tarea	Tareas	Estimación en horas
Sprint 1	Epica 1	Como gestor de visitas de instalación quiero un inicio de sesion para evitar acceso a personas no autorizadas.	T-1	Diseñar login de usuario	3
			T-2	Desarrollar login de acuerdo Wireframe	12
Sprint 2	Epica 2	Como gestor de visitas de instalación quiero ingresar información referente a las visitas de forma sencilla y clara.	T-3	Diseñar interfaz para ingreso de información	3
			T-4	Desarrollar interfaz de ingreso de información	14
Sprint 3	Epica 3	Como gestor de visitas de instalación quiero que la información ingresada quede guardada correctamente en un sitio el cual se pueda consultar nuevamente.	T-5	Diseñar base de datos	3
			T-6	Crear base de datos	6
Sprint 4	Epica 4	Como gestor de visitas de instalación quiero una interfaz en donde pueda editar la información cargada por si se presentan cambios con el cliente.	T-7	Diseñar interfaz de edición de datos	3
			T-8	Desarrollar interfaz de edición de datos	17
Sprint 5	Epica 5	Como gestor de visitas de instalación quiero todo lo que se ingrese y se edite quede guardado y que cuando se consulte las visitas de instalación para editar, se obtengan todos los registros guardatos.	T-9	Desarrollo back-end	17
Sprint 6	Epica 6	Como cliente final, deseo un manual en donde se ecplique como funciona el aplicativo web	T-10	Crear manual de cliente final	4

Fig 1.

Product Backlog

Fase 2: Sprint 1

A. Arquitectura de software.

Para explicar y exponer el funcionamiento del aplicativo web se emple un diagrama UML de secuencia, el cual explica de una forma sencilla el funcionamiento del aplicativo web. A continuación, se presenta el diagrama expuesto en la figura 2:

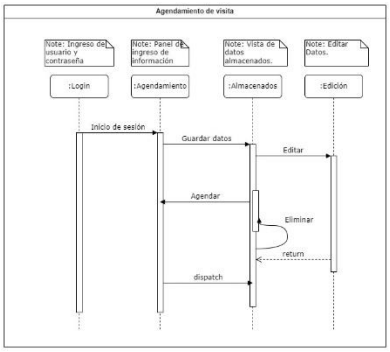

Fig 2
Diagrama UML de secuencia

Desarrollar interfaz de inicio de sesión de cliente final.

Para el diseño del inicio de sesión del cliente se emplea el software Moqup el cual cuenta con distintos objetos, imágenes e iconos para lograr diseñar de la manera más fiel posible el Wirefrme de inicio de sesión.

A continuación, se presenta el Wirefrime diseñado para el inicio de sesión del usuario final:

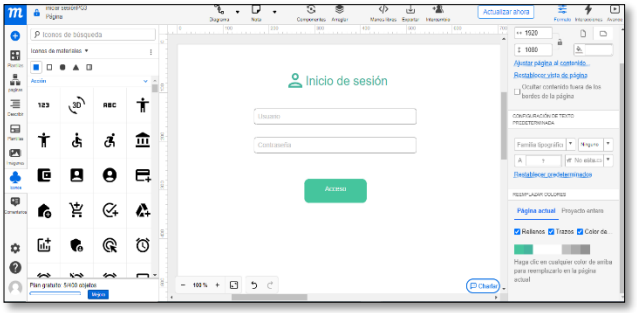

Fig 3.

Wireframe Login

Para el desarrollo de este inicio de sesión se emplea principalmente como “kernel” el lenguaje de programación nodejs, la librería express y el motor de plantillas ejs de la librería mencionada para poder trabajar con vistas dinámicas.

El login esta desarrollado con el lenguaje HTML (Lenguaje de etiquetas de hipertexto). Para dar color y diseño se emplea el lenguaje CSS (Hoja de estilo en cascada) y Bootstap el cual es un framework front-end para el desarrollo de aplicaciones web.

Al iniciar la aplicación, nuestro enrutador (archivo router.js) invocado por la librería express, mostrara la interfaz de inicio de sesión cuando reciba una solicitud get a la dirección raíz (/).

Fase 3: Sprint2

Para el diseño de la interfaz para el ingreso de datos de las visitas de instalación se emplea el software Moqup el cual cuenta con distintos objetos, imágenes e iconos para lograr diseñar de la manera más fiel posible el Wirefrme de inicio de sesión.

A continuación, se presentará el Wireframe de la interfaz para el ingreso de datos de visitas de instalación:

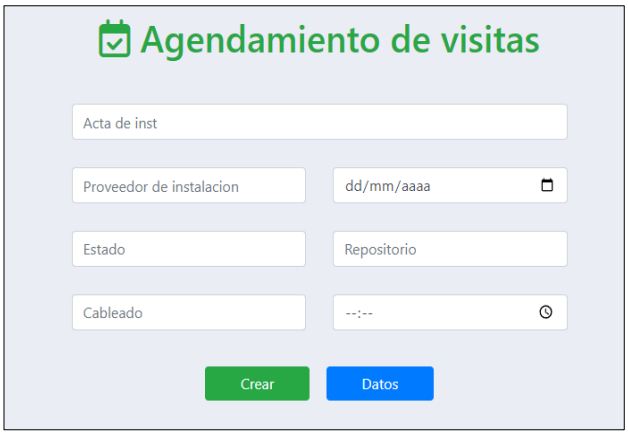

Fig 4.
WireFrame Agendamiento de visitas.

Para el desarrollo de la interfaz para el ingreso de datos de las visitas de instalación se emplea principalmente como “kernel” el lenguaje de programación nodejs, la librería express y el motor de plantillas ejs de la librería mencionada para poder trabajar con vistas dinámicas.

La interfaz para el ingreso de datos de las visitas de instalación esta desarrollado con el lenguaje HTML (Lenguaje de etiquetas de hipertexto). Para dar color y diseño se emplea el lenguaje CSS (Hoja de estilo en cascada) y Bootstap el cual es un framework front-end para el desarrollo de aplicaciones web.

Se emplean dos botones, uno de ellos es crear el cual es de tipo submit y ajusta la url con la dirección /save, por último, envía una solicitud post que será recibida por nuestro enrutador (router.js), esto si se inicia sesión correctamente.

El segundo botón estará destinado para mostrar la vista de edición y eliminación de datos que se hayan guardado en la BDD realizada en PostgreSQL.

Fase 4: Sprint 3

A. Diseño de base de datos.

Para el diseño de la base de datos se emplea un diagrama entidad relación, el cual se realiza mediante el software de uso libre Diagrams.net.

Este cuenta con dos tablas principalmente, en una de ellas se almacenará la información referente a las actas de instalación y en la tabla de Agendamientos se almacenará la información referente a los agendamientos teniendo en cuenta toda la información necesaria para que la tabla cuente con la información necesaria y precisa, para consultar posteriormente por medio del aplicativo web de agendamiento de visitas de instalación.

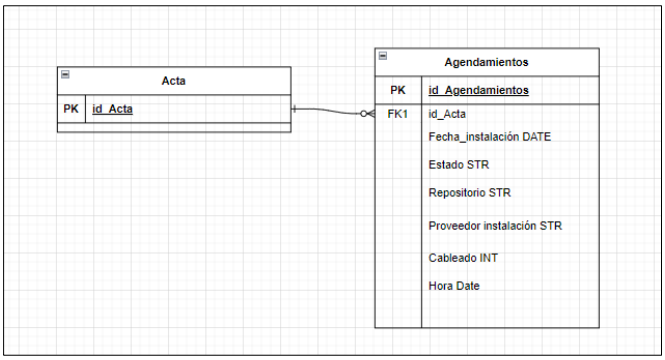

Fig 5.
Diagrama de clases

B. Creación de Base de Datos.

Para la creación de la base de datos se emplea el software de uso libre PostgreSQL, inicialmente se debe instalar el sistema gestor de bases de datos, realizando los ajustes correspondientes en lo que respecta al servidor, al puerto, al usuario y la contraseña.

Para crear una nueva base de datos se puede hacer desde SQL Shell empleando el siguiente comando:

```
CREATE DATABASE Agendamientos;
```

Fig 6.
Comando de creación BDD

Este proceso también se puede realizar desde pgAdmin por medio de la opción créate:

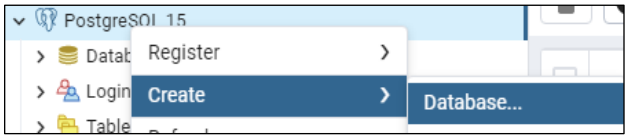

Fig 7.
Creación de Base de datos por PGAdmin.

Al crear la base de datos se deben emplear los distintos campos que esta necesita para poder almacenar la información de acuerdo al tipo de dato establecido.

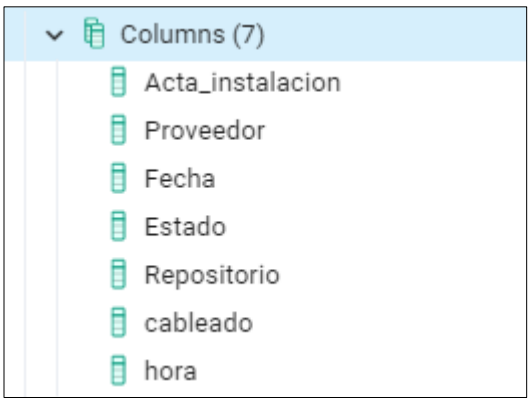

Fig 8.
Empleo de columnas por PGAdmin

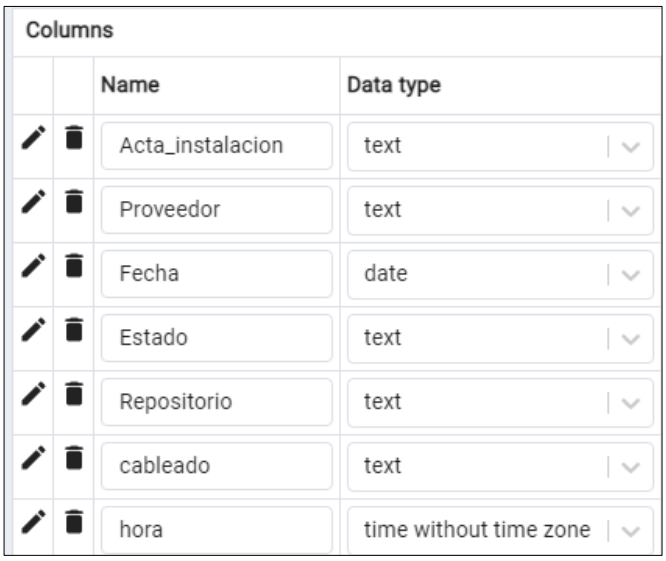

Fig 9.

Definición de campos en PGAdmin.

De acuerdo a la imagen anterior se puede reflejar el tipo de dato de cada uno de los campos que el usuario ingresara en el formulario de Agendamiento de visitas.

Fase 5 : Sprint 4

A. Diseñó de interfaz de edición

En esta parte hay dos formularios a tener en cuenta, el primero de ellos, es desde el cual se va a mostrar la información para seleccionarla y por medio de un botón de edición redirigir a un formulario de edición.

El diseño se realiza mediante el software Moqup y quedan del siguiente modo:

Fig 10.

Vista general de datos y de edición

La vista mostrada anteriormente se realiza mediante HTML y EJS o Embedded JavaScript templates, donde al recibir una solicitud get hacia route, este renderiza index al cual le pasa un objeto proveniente de la BDD el cual se iterara para poder mostrar la información de la BDD:

Codigo desde archive route:

```
res.render('index',{results:result});
```

Codigo de iteración desde la vista EJS llamada index:

```
<% results.rows.forEach(element => { %>
```

Al seleccionar el botón de edición se presentará el siguiente formulario:

Fig 10.

Vista de Edición de datos.

En este caso como se muestra en la imagen, trae todos los datos asociados al agendamiento guardado. Un punto importante a tener en cuenta es que el id del acta de instalación no podrá ser editado después de su primer ingreso en el primer formulario.

Al seleccionar el botón actualizar se activará el evento click el cual tendrá la siguiente orden en lenguaje pgSQL:

```
conexion.query('UPDATE instalaciones SET
"Acta_instalacion"=$1, "Proveedor"=$2,
"Fecha"=$3, "Estado"=$4, "Repositorio"=$5,
cableado=$6, hora=$7 WHERE "Acta_instalacion"
= $1',
```

```
    [acta,proveedor,fecha,estado,repositorio
,cableado,hora],(error,results)=>{
```

Lo que se hace inicialmente es recibir la solicitud post por medio de route y este se redirige a una función de actualización ubicada en el archivo controllers la cual ejecuta esta consulta para que la información quede correctamente actualizada en la base de datos.

Fase 6: Sprint 5

Durante el Desarrollo del aplicativo web se fue implementando código back-end, el cual es referente a los tipos de solicitudes realizados teniendo en cuenta la solitud POST y GET, ya que la lógica se maneja de acuerdo a los endpoints ubicados en la URL

Inicialmente se emplea una solicitud de inicio GET para mostrar la vista de Inicio de sesión:

```
router.get('/',(req,res)=>{
    res.render('login');
});
```

Al recibir la solicitud de tipo GET hacia el endpoint raíz, se renderiza la vista EJS de login, la cual se llama por medio de una dirección relativa gracias al middleware app.use(express.static('public')), public es el directorio donde se encuentran todas las vistas necesarias para que la aplicación funcione correctamente.

La dinámica se repite para todas las vistas cambiando el tipo de solicitud, por ejemplo al recibir una solicitud de tipo post hacia el endpoint "/save" se realizara una acción lógica, en la cual se llama el modulo de guardado de información ubicado en el archivo controllers, para realizar el insert correspondiente en la Base de datos.

```
const crud = require('./controllers/crud');
router.post('/save',crud.save);
```

Al ejecutar la función save del modulo crud presento una parte del código de inserción:

```
    conexion.query('INSERT INTO
instalaciones ("Acta_instalacion",
"Proveedor", "Fecha", "Estado",
"Repositorio", cableado, hora) VALUES
($1,$2,$3,$4,$5,$6,$7);',
    [acta,proveedor,fecha,estado,repositorio
,cableado,hora],(error,results)=>{
            if(error){
                console.log(error);
            }else{
              res.redirect('/create');
            };
        });
```

En conclusión, el back-end mayormente se realiza desde el archivo route el cual cuenta con un módulo que tiene el mismo nombre y nos permite recibir las solicitudes para realizar las acciones correspondientes.

Fase 7: Sprint 6

Finalmente se realiza un manual de usuario, el cual se adjunta en el presente documento para que en futuras ocasiones los integrantes nuevos puedan entender y utilizar correctamente el aplicativo web de gestión de visitas de instalación para el área de continuidad operacional en la empresa FollowUp.

IV. CONCLUSION

Para concluir, el proyecto expuesto anteriormente toma en cuenta las necesidades del área de continuidad operacional para el agendamiento de visitas de instalación de sensores en la empresa FollowUp, evitando inconvenientes y errores referente al proceso. Adicionalmente presenta impactos positivos.nte cuando sea requerido para casos especiales.

REFERENCIAS

[1] Universidad de Murcia. (2016). JavaScript en el servidor. https://www.um.es/docencia/barzana/DAWEB/2017-18/daweb-NodeJS.pdf

[2] Autentia . (2020-08-24). Guia completa Front.

https://www.autentia.com/wp-content/uploads/libros/Front_GuiaCompleta-Autentia.pdf

[3] Universidad de Murcia. (2005-12-16). https://www.um.es/geograf/sigmur/sigpdf/postgresql.pdf

[4] Ken Schwwaber & Jeff Sutherland. (2020). La Guía Scrum. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish-European.pdf