

**UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE
MORELOS**

**DIVISIÓN ACADÉMICA DE TECNOLOGÍAS DE LA
INFORMACIÓN Y DISEÑO**

**SISTEMA DE CONTROL DE TRÁMITES –
MÓDULO WEB SEGUIMIENTO DE
VENTANILLA**



REPORTE DE ESTADÍA

QUE PARA OBTENER EL GRADO DE:

**TÉCNICO SUPERIOR UNIVERSITARIO EN
TECNOLOGÍAS DE LA INFORMACIÓN
ÁREA DESARROLLO DE SOFTWARE
MULTIPLATAFORMA**

**PRESENTA:
BAHENA CASTILLO LUIS EDUARDO**

ASESOR EMPRESARIAL

**T.S.U. OBED ARIEL HURTADO
HERNÁNDEZ**

ASESOR UNIVERSITARIO

**I.T.I. ERICK MIRELES
MERCHANT**



EMILIANO ZAPATA, MOR., SEPTIEMBRE DE 2023



CONTENIDO

Índice de figuras

Índice de tablas

Agradecimientos

Resumen

Summary

CAPÍTULO 1. DESCRIPCIÓN DEL PROYECTO9

1.1 Datos generales de la empresa 9

1.2 Antecedentes del proyecto 9

1.3 Objetivo general..... 10

1.4 Objetivos específicos 10

1.5 Justificación 11

1.6 Alcances 12

1.7 Restricciones 12

CAPÍTULO 2. MARCO DE REFERENCIA13

2.1 Conceptos, metodologías y herramientas..... 13

2.1.1 Conceptos básicos de programación 13

2.1.2 Metodologías..... 16

2.1.3 Herramientas..... 18

2.2 Propuesta de solución 25

2.2.1 Tecnologías seleccionadas 26

2.2.2 Metodología seleccionada..... 28

CAPÍTULO 3. DESARROLLO29

3.1 Inicio	29
3.2 Planeación	30
3.3 Ejecución	33
3.3.1 Análisis	33
3.3.2 Diseño	36
3.3.3 Programación	45
3.3.4 Pruebas	61
3.4 Control	70
3.5 Cierre	71
CAPÍTULO 4. CONCLUSIONES	72
4.1 Cumplimiento de objetivos	72
4.2 Resultados	73
4.3 Contribuciones	73

REFERENCIAS

ANEXOS

ÍNDICE DE FIGURAS

Figura 1.1 Logotipo de la empresa.....	9
Figura 2.1 Estructura de envío de datos mediante API REST.....	14
Figura 2.2 Ciclo de Metodología SCRUM (Calvo, 2018)	16
Figura 2.3 Inicialización de un proyecto en Spring Initializr	20
Figura 2.4 Ciclo de vida de un componente React JS (Ismaellopez.dev, 2022)	21
Figura 3.1 Diagrama EDT.....	30
Figura 3.2 Cronograma de Planeación.....	31
Figura 3.3 Representación Conceptual	34
Figura 3.4 Diagrama de Casos de Uso	35
Figura 3.5 Diseño de Tablas de la Base de Datos	36
Figura 3.6 Diseño de Vista Inicio de Sesión	37
Figura 3.7 Diseño de Vista Menú Administrador	37
Figura 3.8 Diseño de Vista Usuario.....	38
Figura 3.9 Diseño de Vista Crud Administradores.....	38
Figura 3.10 Diseño de Vista Crud Solicitantes	39
Figura 3.11 Diseño de Vista Crud Servicios.....	39
Figura 3.12 Diseño de Vista Gestión Citas.....	40
Figura 3.13 Diseño de Vista Gestión Horarios	40
Figura 3.14 Diseño de Vista Reservar Cita	41
Figura 3.15 Diseño de Vista Citas registradas	42
Figura 3.16 Correo de Confirmación de Creación de Cuenta	43
Figura 3.17 Estado de las Citas	43
Figura 3.18 Cambiar Contraseña	44
Figura 3.19 Modal Citas	44
Figura 3.20 Conexión a la Base de Datos.....	45
Figura 3.21 Entidad Administrador.....	46
Figura 3.22 Clase AdminDto	47
Figura 3.23 Servicio Administrador	48

Figura 3.24 Controller del Administrador	49
Figura 3.25 Entidad Ventanilla	50
Figura 3.26 Clase VentanillaDto	51
Figura 3.27 Servicio Ventanilla	51
Figura 3.28 Controller Ventanilla	52
Figura 3.29 Entidad Solicitante	53
Figura 3.30 Clase SolicitanteDto	54
Figura 3.31 Servicio Solicitante	55
Figura 3.32 Controller Solicitante	56
Figura 3.33 Entidad Servicio	57
Figura 3.34 Entidad Cita	58
Figura 3.35 Entidad Horario	59
Figura 3.36 Ejemplo del Servicio de Correo en Ventanilla	60
Figura 3.37 Cuerpo de Confirmación del Correo	61
Figura 3.38 Solicitud POST Administrador	62
Figura 3.39 Solicitud GET Administrador	62
Figura 3.40 Solicitud PUT Administrador	63
Figura 3.41 Solicitud PATCH Administrador	63
Figura 3.42 Solicitud POST Ventanilla	64
Figura 3.43 Solicitud PUT Ventanilla	65
Figura 3.44 Solicitud PATCH Ventanilla	65
Figura 3.45 Solicitud POST Servicio	66
Figura 3.46 Solicitud GET Servicio	67
Figura 3.47 Solicitud GET Citas	67
Figura 3.48 Solicitud PATCH Citas	68
Figura 3.49 Solicitud POST Horarios	68
Figura 3.50 Confirmación del Correo	69
Figura 3.51 Modal de Confirmación de Registro del Usuario Administrador	69
Figura 3.52 Cronograma de Avance	70

ÍNDICE DE TABLAS

Tabla 3.1 Involucrados y sus funciones	29
Tabla 3.2 Módulos a implementar en el Sistema de CITAT	33

AGRADECIMIENTOS

Mis agradecimientos a las siguientes personas:

A mi papá y mi mamá, ya que ellos fueron el motor de seguimiento en mi etapa universitaria, donde ellos han visto mis victorias y mis fracasos, alentándome a ser una mejor persona y que sea alguien en la vida, que sea una persona honesta y trabajadora, y también porque me han brindado el apoyo que he necesitado en todo momento tanto económico como moral.

A los profes de la Prepa CETis NO. 44, ya que ellos me enseñaron lo básico de la carrera de programación y de otras materias; y al final de cuentas, sus enseñanzas me sirvieron mucho de utilidad en la universidad.

A los profesores de la UTEZ, que me brindaron apoyo para la aclaración de dudas e impartido asesorías para un mejor desarrollo como estudiante y para aumentar mi nivel de autonomía.

A mi Asesor de Estadías el I.T.I. Erick Mireles Merchant por su apoyo que me ha brindado durante mi desarrollo en las estadías y en el asesoramiento impartido logrando desarrollar como persona y que me brindo su confianza hacía mi esfuerzo.

A la empresa en general, que me dio la oportunidad de poder desarrollarme como profesionista y que pudiera desenvolverme en las actividades realizadas para el desarrollo del proyecto.

Y a mis amigos, tanto de la secundaria como de la preparatoria y la universidad, por haberme dado su apoyo incondicional en esta etapa universitaria.

RESUMEN

En el transcurso de la realización del proyecto, se espera que la empresa **Acción TI** cuente con un nuevo sistema para la gestión de citas para el control de trámites, basado en las agendas de la fecha y la carga de la documentación anexa a la misma, a lo cual se espera poder desarrollar competencias autónomas para la instancia laboral y fomenta un nuevo sistema para una escuela privada en el que se solicita; y el resultado de este proyecto brindará soluciones a la empresa para poder realizar futuros proyectos.

Capítulo 1. Incluye datos relevantes acerca de la empresa solicitante del proyecto, su ubicación, responsables a cargo, así como los objetivos generales y específicos del proyecto. También se mencionan los alcances y las restricciones que se deben tener en cuenta durante su ejecución.

Capítulo 2. Se detallan y examinan todos los conceptos, metodologías y herramientas de la programación y al final se seleccionarán las necesarias para proponer una solución y guiar el desarrollo del proyecto.

Capítulo 3. Se llevan a cabo actividades y se crean documentos relacionados con la gestión del proyecto a realizar, especialmente durante la etapa de ejecución, aplicando las fases necesarias para construir el producto o alcanzar los objetivos establecidos.

Capítulo 4. Se expone el cumplimiento de los objetivos del proyecto, se presentan los resultados obtenidos y las contribuciones realizadas a la organización, así como los beneficios logrados en términos de la gestión de las citas.

SUMMARY

Throughout the course of the project execution, it is expected that the company Acción TI will have a new system for appointment management for process control. This system will be based on scheduling dates and loading relevant documentation. The aim is to foster independent competencies for the work instance and promote a new system for a private school, as requested. The outcome of this project will provide solutions for the company to undertake future projects.

Chapter 1. This chapter includes pertinent information about the requesting company, its location, responsible individuals, as well as the overall and specific project objectives. The scope and constraints that must be considered during its implementation are also mentioned.

Chapter 2. All programming concepts, methodologies, and tools are detailed and examined. The necessary elements will be selected to propose a solution and guide the project's development.

Chapter 3. Activities related to project management are carried out, especially during the execution phase. Necessary phases are applied to construct the product or achieve established objectives.

Chapter 4. The fulfillment of project objectives is presented, along with the obtained results and contributions to the organization. Additionally, the benefits achieved in terms of appointment management are highlighted.

CAPÍTULO 1. DESCRIPCIÓN DEL PROYECTO

1.1 Datos generales de la empresa

Acción TI es una compañía comprometida con los servicios de Mercadotecnia Digital y las Tecnologías de Información, apoyando el crecimiento empresarial y ofreciendo soluciones para las Tecnologías de Información, Mercadotecnia Digital e E-Learning. Además, ofrece diferentes soluciones tecnológicas; desde el diseño de páginas web personalizadas, la administración de contenidos “CMS”, y catálogos de productos en línea, tiendas virtuales y redes sociales.



Figura 1.1 Logotipo de la empresa

Acción TI se encuentra ubicada en Blvd. Benito Juárez, #58, Col. Las Palmas, Cuernavaca Morelos C.P. 62050. Esta empresa cuenta aproximadamente con un total de 15 empleados, distribuidos en diferentes áreas: recursos humanos, administración, operación, soporte técnico, desarrollo, marketing y así como en el área de capacitación a los nuevos integrantes o estudiantes de estadías. Este proyecto está dirigido y administrado por el Lic. David Alejandro Silva Magallón, licenciado en administración y mercadotecnia, desempeñando el cargo de director comercial.

1.2 Antecedentes del proyecto

En el marco del contexto de la institución educativa donde se solicita el proyecto, es algo común que cualquier persona interesada que labore, requiera realizar trámites administrativos o solicitar algunos servicios específicos. Todos estos trámites pueden

incluir desde realizar inscripciones hasta solicitar certificaciones de estudios. Anteriormente no había un sistema que gestionará las citas de los trámites, puesto que estos procesos se realizaban de manera presencial, lo que implicaba largas filas, pérdida de tiempo y mucha desorganización en la atención a los usuarios. Además, se requería que los interesados llevaran consigo una serie de documentos en formato físico para entregarlos durante el trámite y también al momento de realizar pagos en tarjetas de crédito/débito, no contaban con una terminal para realizar ese servicio de pago. Este proyecto se distribuye de acuerdo con el Documento Formal de Requerimientos (DFR) que se proporcionó por el responsable empresarial, por lo que se solicita una aplicación web y móvil para garantizar el control y la gestión del sistema de citas. También está experimentando las soluciones con el uso de las nuevas tecnologías para desarrollar aplicaciones, así como también de nuevos métodos para agilizar el tiempo y coste para el desarrollo del proyecto y tener una gestión más eficiente de las citas y los trámites.

1.3 Objetivo general

Optimizar el proceso de programación y seguimiento de citas para trámites, garantizando una gestión eficiente y ordenada que permita a los usuarios realizar sus trámites de manera rápida y efectiva, reduciendo tiempos de espera y mejorando la experiencia del usuario.

1.4 Objetivos específicos

1. Diseñar la base de datos que se utilizará para la aplicación web y móvil, así como definir los métodos de acceso a los datos mediante la implementación de servicios Web REST.
2. Implementar los módulos de control de administrador, ventanilla y solicitante para los servicios que se ofrecerán el sistema de control de citas.

3. Implementar un sistema de seguimiento de citas que permita a los usuarios y al personal encargado del trámite visualizar y gestionar el estado de las citas programadas, desde la solicitud hasta la finalización del trámite.
4. Configurar e implementar el aprendizaje automático en un entorno virtual.
5. Definir e implementar los reportes requeridos por la empresa para realizar un seguimiento de los proyectos y acciones de mantenimiento planificadas y finalizadas.

1.5 Justificación

Debido a la falta de una buena organización y control en el sistema de citas, se realiza este proyecto para mejorar la eficiencia, agilizar los trámites, brindar una mejor experiencia al usuario, aumentar la transparencia y permitir la toma de decisiones basadas en datos, así como brindar a la institución la posibilidad de gestionar las citas de manera automatizada y digital. Esto implica permitir a los interesados reservar citas para realizar trámites específicos, como la certificación de estudios mencionada anteriormente, a través de una plataforma en línea.

El desarrollo de este proyecto cubrirá las necesidades y/o expectativas tanto de la empresa como de la institución educativa, manteniendo la simplicidad y el correcto flujo de los elementos manejados y haciendo más óptima la gestión de citas para poder satisfacer las necesidades al cliente mediante los módulos a realizar para implementarse en la aplicación web y móvil, así como gestionar las citas; y será de utilidad este mismo al momento de realizar pagos y cargar documentación anexa, esto agilizará el proceso y evitará la necesidad de que los usuarios se trasladen físicamente a la escuela para realizar el pago y llevar la documentación de manera presencial, permitiendo una experiencia más cómoda y conveniente. La organización del equipo constante dentro de la empresa será un desarrollo clave para poder cumplir los objetivos de este proyecto, puede ser una solución temporal, pero, se requiere el desarrollo de la aplicación propia para poder brindar un mejor servicio.

1.6 Alcances

- El proyecto a desarrollar incluirá una aplicación web y móvil como entregables finales.
- El sistema incluirá módulos de gestión del administrador, de ventanilla y de solicitante para poder gestionar y agendar las citas.
- Se realizarán pruebas de conectividad, usabilidad y funcionalidad tanto la aplicación web como móvil antes de montarlo a un servidor que se nos proporcionara.
- Se tendrá un manual de usuario en forma digital y con un único formato PDF.

1.7 Restricciones

- Las reuniones para realizar el proyecto serán todas en modo virtual en un horario de 8:00 a.m. a 12:00 p.m.
- El proyecto no contará con extensiones de tiempo y se tendrá que entregar montado en un servidor, donde se deberá adquirir o tener acceso a uno adecuado para alojar el sistema ya sea por el cliente o la empresa.
- Se deberá de utilizar una herramienta llamada “pasarelas de pago” para poder realizar pagos en tarjeta de crédito/débito y así mismo debe cumplir con los estándares y regulaciones de seguridad de la industria de pagos, para garantizar la protección de la información confidencial de los usuarios durante las transacciones.
- Los horarios laborales serán de manera virtual en horario de 8:00 a.m. a 5:00 p.m.

CAPÍTULO 2. MARCO DE REFERENCIA

2.1 Conceptos, metodologías y herramientas

Para la realización de este proyecto, el primer elemento fundamental para adentrarnos en el desarrollo de este mismo es comprender los conceptos clave. Estos serán los pilares fundamentales sobre los cuales se construyen todas las metodologías (abarca desde reducir el nivel de dificultad hasta agilizar procesos y mejorar el resultado final del proyecto) y las distintas herramientas a trabajar. Los conceptos pueden brindar, en esta sección, una comprensión básica y una visión general de los temas relevantes del proyecto necesarios para poder saber acerca de la gestión de citas.

2.1.1 Conceptos básicos de programación

Aplicación

Especie de "herramienta digital" diseñada para realizar tareas específicas y resolver problemas de manera más sencilla y eficiente. Es como tener un compañero tecnológico que simplifica nuestras vidas y nos conecta con el mundo digital de manera más amigable. Mencionando lo anterior, existen dos tipos de aplicaciones para poder interactuarlas en los diferentes dispositivos tecnológicos, estos se les conoce como aplicaciones web y aplicaciones móviles.

La aplicación web es aquella en donde se ejecuta en un navegador web y está disponible a través de Internet. Se puede acceder desde cualquier dispositivo (computadora, dispositivo móvil u otro dispositivo digital) con conexión a Internet y con un navegador compatible. Estas suelen ser accesibles a través de una URL y no requieren una descarga o instalación en el dispositivo.

La aplicación móvil es diseñada para funcionar específicamente en dispositivos móviles (como teléfonos inteligentes o tabletas) y estas se descargan e instalan directamente en el dispositivo a través de una tienda de aplicaciones (dependiendo del sistema operativo del móvil, como Google Play Store para Android o App Store para iOS). Están optimizadas para poder aprovechar las características y capacidades específicas de los dispositivos móviles, como el uso del GPS, la cámara y las notificaciones en tiempo real.

API REST

Interfaz de Programación de Aplicaciones Representacional de Estado Transferencia, es la forma de comunicarse con una aplicación o sistema informático. También se dice que, “es una interfaz de comunicación entre sistemas de información que usa el protocolo de transferencia de hipertexto (hypertext transfer protocol o HTTP, por sus siglas en inglés) para obtener datos o ejecutar operaciones sobre dichos datos en diversos formatos, como pueden ser XML o JSON.”, (Coppola, Qué es una API REST, para qué sirve y ejemplos, 2022). Basado en lo anterior, son el conjunto de reglas y convenciones que permite que diferentes sistemas y aplicaciones se comuniquen entre sí de manera eficiente y estandarizada, y establece un conjunto de reglas que definen cómo se debe solicitar y enviar información entre distintas aplicaciones. Utiliza un conjunto de métodos o verbos estándar, como GET, POST, PUT y DELETE, para indicar qué tipo de acción se quiere realizar, tal como se muestra en Figura 2.1, además de que para transmitir datos, puedes realizar cualquier solicitud que quieras hacer y se recibirá la respuesta con los datos solicitados.

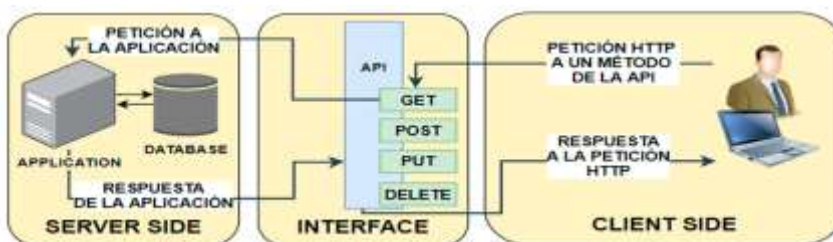


Figura 2.1 Estructura de envío de datos mediante API REST

Framework

Estructura sólida y versátil que ayuda a los desarrolladores a construir aplicaciones y sitios web de manera más rápida y eficiente. Es como una caja de herramientas que contiene todo lo necesario para llevar a cabo un proyecto de desarrollo de software. Cuando se quiere utilizar un framework (puede ser cualquiera), no se tiene que empezar desde cero. En lugar de eso, “los frameworks proporcionan una estructura para el desarrollo de software y garantizan que el código se escriba de forma coherente y lógica. Un framework de desarrollo de software suele incluir una arquitectura, una filosofía de diseño y una selección de elementos como clases de utilidad y ayudantes”, (Suárez, 2023). Con lo anterior, así se puede aprovechar las plantillas y bibliotecas que ya están creadas, lo que ahorra tiempo y esfuerzo, además de que un aspecto clave es que establecen una estructura de trabajo que guía a los desarrolladores a seguir ciertas convenciones y prácticas recomendadas. Esto no solo facilita la colaboración entre diferentes programadores, sino que también mejora tanto evolución y el mantenimiento del software como la adaptación del software a lo largo del tiempo.

Bases de Datos

Sistema organizado y estructurado que permite almacenar y gestionar grandes cantidades de información de manera eficiente. Se define como una especie de almacén virtual donde se guardan datos relacionados de forma organizada para su posterior consulta y manipulación, tanto que “las bases de datos son un elemento fundamental en el entorno informático, en la actualidad tienen una aplicación en la práctica casi total en algunos campos, además, son de utilidad para toda disciplina o área de aplicación donde exista la necesidad de gestionar datos”, (Pulido Romero, Escobar Domínguez, & Núñez Pérez, 2019). Por eso mismo, se utiliza para almacenar grandes volúmenes de datos de manera estructurada para poder crear tablas o colecciones que representen diferentes entidades o conceptos, como clientes,

productos o transacciones. Cada fila de la tabla representa una instancia de esa entidad y cada columna almacena un atributo específico de esa instancia.

2.1.2 Metodologías

Metodología SCRUM

Forma de metodología ágil y efectiva de trabajar en equipo para desarrollar proyectos y productos de manera colaborativa. Aquí, “la metodología Scrum permite abordar proyectos complejos desarrollados en entornos dinámicos y cambiantes de un modo flexible. Está basada en entregas parciales y regulares del producto final en base al valor que ofrecen a los clientes”, (Hurtado, 2021), lo que quiere decir que es un enfoque utilizado en la gestión de proyectos y es especialmente adecuado para abordar proyectos que son complejos y se desarrollan en entornos que cambian constantemente. Una de las ventajas clave es que permite una entrega rápida y continua de resultados, lo que significa que los clientes o usuarios pueden obtener valor en etapas tempranas del proyecto en lugar de esperar hasta el final.

En la Figura 2.2, podemos observar el flujo de trabajo donde durante un sprint de requerimientos, se descubre cuando un enfoque diferente es más efectivo, el equipo se puede ajustar su plan en el siguiente sprint sin complicarse.



Figura 2.2 Ciclo de Metodología SCRUM (Calvo, 2018)

Metodología Kanban

Enfoque de gestión visual que ayuda a los equipos a organizar y optimizar su trabajo de manera eficiente. Es como una pizarra o tablero donde se colocan tarjetas o notas que representan las tareas pendientes y en progreso. Esta metodología “se implementa por medio de tableros Kanban. Se trata de un método visual de gestión de proyectos que permite a los equipos visualizar sus flujos de trabajo y la carga de trabajo. En un tablero Kanban, el trabajo se muestra en un proyecto en forma de tablero organizado por columnas. Tradicionalmente, cada columna representa una etapa del trabajo”, (Martins, 2022). Se refiere a una forma de implementar utilizando tableros visuales y ayuda a los equipos a visualizar el flujo de trabajo y a identificar posibles cuellos de botella o áreas donde se necesita más atención. También promueve una mayor colaboración y comunicación entre los miembros del equipo, ya que todos pueden ver claramente qué se está haciendo y quién está trabajando en cada tarea.

Metodología Lean

Enfoque de gestión que se centra en eliminar el desperdicio y optimizar los procesos para lograr una mayor eficiencia y calidad en la entrega de productos o servicios. Es como una búsqueda constante de mejorar y hacer más con menos recursos, o también se puede definir como “Una forma innovadora de gestionar los procesos de una empresa. Su objetivo es eliminar actividades que no aportan valor, para así poder obtener un producto o servicio de mayor calidad y que mejore la experiencia de los clientes”, (Humanes, 2019). Resumiendo lo anterior, se basa en la idea de la mejora continua, lo que significa que los equipos se esfuerzan por perfeccionar constantemente sus procesos y eliminar ineficiencias para ser más ágiles y eficaces. Otro aspecto fundamental es el empoderamiento y la participación de los miembros del equipo en la toma de decisiones y en la identificación de áreas de mejora.

Metodología XP

También conocido como Metodología de Programación Extrema, es el enfoque ágil y colaborativo para desarrollar aplicaciones de software. Es como un equipo de programadores trabajando juntos en una atmósfera cercana y comunicativa. También se dice que, “es un conjunto de técnicas que dan agilidad y flexibilidad en la gestión de proyectos, se centra en crear un producto según los requisitos exactos del cliente. De ahí que se involucre al máximo durante el método de gestión del desarrollo del producto”, (Canive, 2020) y esta metodología, quiere decir que se enfoca en crear un producto y se prioriza la satisfacción del cliente donde se cumpla con los requisitos precisos y la entrega constante de pequeñas funcionalidades valiosas. En lugar de esperar hasta el final del proyecto para presentar el producto terminado, se realizan entregas frecuentes y regulares, poniendo gran énfasis en la participación activa del cliente durante todo el proceso de gestión y desarrollo del producto.

2.1.3 Herramientas

Java

Lenguaje de programación versátil y robusto que se utiliza ampliamente en el desarrollo de software. Es el conjunto de instrucciones y reglas que permiten a los programadores crear aplicaciones y sistemas que funcionan en diferentes dispositivos y plataformas. La belleza de esto es que está orientado a objetos, lo que significa que podemos organizar en representaciones de entidades o conceptos del mundo real, además de que es un lenguaje común que permite a los desarrolladores escribir programas una vez y ejecutarlos en múltiples plataformas. Y por si fuera poco, “En la actualidad, Java se utiliza para desarrollar aplicaciones empresariales a gran escala, para mejorar la funcionalidad de los servidores Web (las computadoras que proporcionan el contenido que vemos en nuestros navegadores Web), para proporcionar aplicaciones para los dispositivos de uso doméstico (como teléfonos celulares, teléfonos inteligentes, receptores de televisión por Internet y mucho más) y

para muchos otros propósitos”, (Deitel & Deitel, 2016). Esto quiere decir que se emplea en diversas áreas, como el desarrollo de aplicaciones empresariales a gran escala, mejorando la funcionalidad de servidores web y creando aplicaciones para dispositivos domésticos, como teléfonos celulares y receptores de televisión por Internet, entre otros usos.

JavaScript

Lenguaje de programación que da la vida a los sitios web, permitiendo a los usuarios interactuar con los elementos de la página y realizar acciones en tiempo real. Se ejecuta directamente en el navegador web, lo que quiere decir que no es necesario instalar algo adicional para que funcione. Lo convierte en la herramienta poderosa para crear las experiencias interactivas en el lado del cliente, y esto mejora la usabilidad y la experiencia del usuario en una aplicación web o móvil, se puede interactuar con el contenido del Lenguaje de Marcado de Hipertexto (HTML) y de las Hojas de Estilo en Cascada (CSS), donde se manipula elementos y responde a eventos a través de su estructuración y presentación de una página web. “Sin embargo, JavaScript también puede resultar difícil de usar, debido a algunas diferencias importantes entre las formas en las que los diferentes diseñadores de navegadores han elegido implementarlo. Esto sucedió principalmente cuando algunos fabricantes trataron de incluir funcionalidades adicionales a sus navegadores a expensas de la compatibilidad con sus rivales”, (Nixon, 2020), puede ser complicado de utilizar debido a las diferencias en su implementación entre los distintos navegadores, ya que algunos fabricantes agregaron funciones adicionales que afectaron la compatibilidad con otros navegadores y generó dificultades para los desarrolladores. Esto causó discrepancias y dificultades en el uso de este lenguaje. A pesar de estos desafíos, sigue siendo una herramienta esencial para el desarrollo web y su versatilidad lo mantiene como uno de los pilares fundamentales de la programación en internet para poder ser útil en la conexión del navegador web al servidor local.

Spring

Framework de desarrollo de aplicaciones basado en el lenguaje de programación Java que ayuda a crear aplicaciones empresariales sólidas y escalables. Resuelve varios problemas comunes en el desarrollo de software, como la organización de componentes, la seguridad y el acceso a bases de datos. Es como una caja de herramientas que proporciona a los desarrolladores un conjunto de funciones y componentes predefinidos para crear aplicaciones de forma rápida y eficiente. Esto genera que es uno de los frameworks más populares del mundo, y “la razón por la que esta práctica de marco de trabajo se ha popularizado tanto en el mundo de la programación Java, es que permite agilizar el proceso de desarrollo ya que todas las herramientas disponibles fueron desarrolladas en Java”, (Roca, 2023). Esto quiere decir que esta práctica de marco de trabajo ha ganado mucha popularidad en el mundo de la programación Java porque agiliza el proceso de desarrollo al ofrecer herramientas desarrolladas derivadas de ese lenguaje, lo que facilita la creación de aplicaciones con mayor eficiencia y compatibilidad. Además de que se basa en el principio de inversión de control y la inyección de dependencias, lo que facilita la construcción de aplicaciones flexibles y mantenibles. En la Figura 2.3, se puede observar la herramienta de Spring Initializr, donde en automático se puede generar un proyecto con las dependencias necesarias para la construcción de los servicios de API REST.

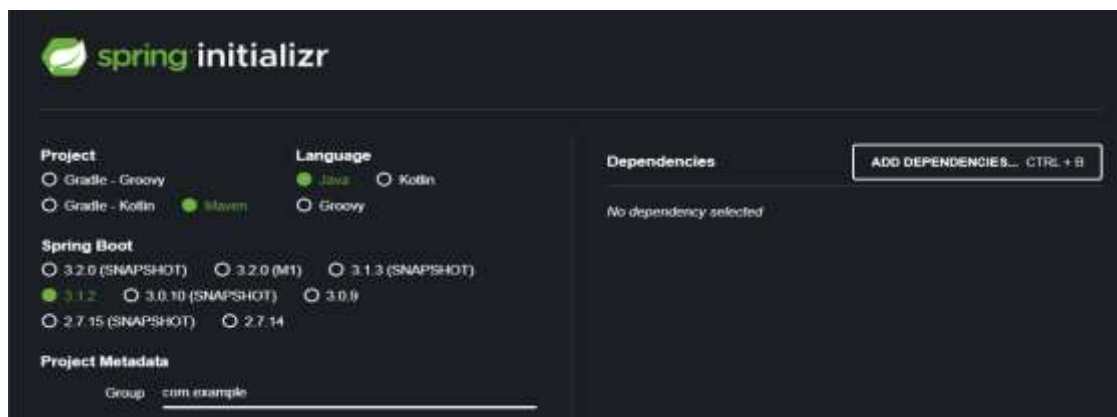


Figura 2.3 Inicialización de un proyecto en Spring Initializr

React JS

Poderosa biblioteca de JavaScript que brinda a los desarrolladores a construir interfaces de usuario interactivas y dinámicas para aplicaciones web. Es como un conjunto de herramientas que simplifica la creación de componentes reutilizables y modernos que pueden actualizarse en tiempo real sin necesidad de recargar toda la página. "La función principal de React es desarrollar páginas web de una manera gratuita y sencilla gracias a sus componentes reutilizables. Estos hacen posible usar un mismo elemento en varias partes del sitio o en otros sitios sin necesidad de volver a escribir todo el código. Como resultado, los programadores web ahorran gran cantidad de tiempo y trabajo." (Coppola, ¿Qué es React y para qué sirve?, 2022). Quiere decir que permite desarrollar páginas web de manera fácil y gratuita utilizando componentes reutilizables y pueden ser utilizados varias veces en el mismo sitio o en otros sin tener que reescribir el código, lo que ahorra mucho tiempo y esfuerzo a los programadores, se enfoca en mantener la interfaz de usuario sincronizada con el estado actual de la aplicación en el que cualquier cambio en los datos se refleja automáticamente en la interfaz de usuario sin que se tengan que realizar acciones adicionales. Esto proporciona una experiencia más fluida y receptiva, como en la Figura 2.4, donde se observa el ciclo de vida del componente de React JS, con el método 'hook useState' (especie de memoria privada para almacenar información para luego modificarlo).

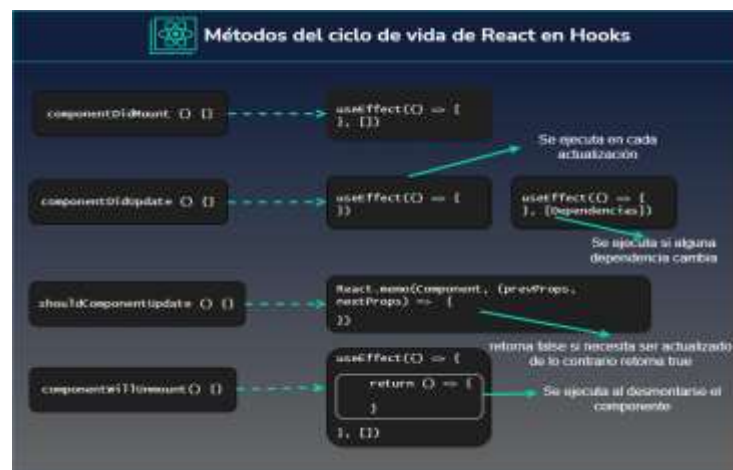


Figura 2.4 Ciclo de vida de un componente React JS (Ismaellopez.dev, 2022)

React Native

Potente plataforma parte también de biblioteca de JavaScript que permite a los desarrolladores crear aplicaciones nativas para dispositivos móviles (iOS y Android) utilizando JavaScript y React JS. Es una plataforma versátil que permite construir aplicaciones móviles con una base de código compartida, lo que ahorra tiempo y esfuerzo en el desarrollo. Así mismo, utiliza componentes reutilizables al igual que React JS, lo que facilita crear interfaces de usuario consistentes y modernas en las plataformas. Estos componentes se combinan para formar la estructura de la aplicación, permitiendo una fácil personalización y una ventaja clave es su capacidad para acceder a características nativas de los dispositivos móviles, como la cámara o el GPS, lo que posibilita desarrollarlos con funcionalidades avanzadas y de alto rendimiento.

Node JS

Entorno de ejecución de código derivado del lenguaje de programación JavaScript que permite a los desarrolladores utilizarlo tanto en el navegador como en el servidor, así como poder construir aplicaciones del lado del servidor de manera eficiente y escalable (capacidad de aportación de una aplicación). Así que, “en la actualidad, Node.js cuenta con un gran número de módulos y componentes orientados y optimizados para networking (redes), que sirven de soporte para el manejo de los estándares y protocolos más populares de Internet, como DNS, HTTP, TCP, TLS/SSL y UDP. Por todo esto, cuando escuchamos hablar sobre Node.js, es muy probable que lo asociemos con la capa de servicios de aplicaciones orientadas a Internet”, (Puciarelli, 2020). Esto se resume a que tiene una amplia variedad de módulos y componentes diseñados para trabajar con redes donde los módulos proporcionan soporte para diversos estándares y protocolos de Internet, Lo interesante es que se puede manejar muchas peticiones de forma eficiente y responder rápidamente a cada una de ellas.

Spring Boot

Marco de trabajo de desarrollo de aplicaciones para Java que simplifica enormemente la creación de aplicaciones empresariales. No debe confundirse con Spring, ya que este concepto es la extensión que simplifica la configuración y la puesta en marcha de aplicaciones Spring. Por ende, “Spring Boot contiene una infraestructura ligera que elimina la mayor parte del trabajo de configurar las aplicaciones basadas en Spring. El objetivo de Spring Boot es proporcionar un conjunto de herramientas para construir rápidamente aplicaciones de Spring que sean fáciles de configurar.” (Gonzalez, 2021). De acuerdo con lo anterior, significa que proporcionan una variedad de funciones y herramientas para crear rápidamente aplicaciones web con servicios API REST, aplicaciones de microservicios y mucho más. También se incluyen bibliotecas y dependencias que ayudan con aspectos comunes como el manejo de bases de datos, la seguridad y la gestión de la configuración. Esto ahorra tiempo y esfuerzo, ya que no es necesario configurar manualmente muchas de las tareas comunes.

MYSQL

Sistema de gestión de bases de datos de código abierto muy popular que permite almacenar y gestionar grandes cantidades de información de manera estructurada. “Es el contenedor de datos detrás de websites enormes con cientos de páginas de datos, y pequeños sites con unas pocas páginas. También se utiliza en muchas aplicaciones que no están basadas en Web. Es rápido, estable, y pequeño cuando se necesita.” (Huillcen Baca, Palomino Valdivia, & Soria Solís, 2022). Es como una enorme hoja de cálculo en la que puedes organizar y recuperar datos rápidamente de manera eficaz y se puede utilizar para guardar una variedad de datos, como nombres de usuarios, contraseñas, registros de ventas, información de productos y mucho más, lo destacado es su habilidad para manejar muchas solicitudes de acceso a la base de datos al mismo tiempo, permitiendo a múltiples usuarios o aplicaciones acceder y actualizar datos simultáneamente sin dificultades y su alto rendimiento como velocidad lo hacen

una opción ideal para aplicaciones web, sistemas de gestión de contenido y proyectos que necesitan acceder eficientemente a grandes volúmenes de datos.

Postman

Herramienta de desarrollo que facilita la tarea de probar, probar y depurar servicios web y API. Es como un asistente digital que facilita la interacción y su análisis, permitiendo a los desarrolladores probar, documentar y compartir sus servicios de manera eficiente. “Postman cuenta con varias herramientas gratuitas que nos hará el mundo de las API algo más sencillo. Algunas tareas que podemos llevar a cabo de forma gratuita serían la creación de peticiones a APIs internas o de terceros, elaborar test que sirvan para validar el comportamiento de dichas APIs y crear entornos de trabajo diferentes”, (Pérez, 2021). En pocas palabras, esta herramienta facilita su trabajo, donde se pueden enviar diferentes tipos de solicitudes HTTP (en este caso GET, POST, PUT o DELETE) para interactuar y ver cómo se responden. También se puede utilizar variables, entornos y pruebas automatizadas para asegurarse de que las respuestas sean las esperadas y de que no haya errores o problemas en el flujo de datos.

Visual Studio Code

Entorno de desarrollo de software muy popular diseñado para facilitar y mejorar el proceso de escritura y edición de código en los diferentes lenguajes de programación, es el taller digital donde los desarrolladores pueden escribir, editar y depurar su código de manera eficiente. “Es un editor de código fuente desarrollado por Microsoft. Es software libre y multiplataforma, está disponible para Windows, GNU/Linux y macOS. VS Code tiene una buena integración con Git, cuenta con soporte para depuración de código, y dispone de un sinnúmero de extensiones, que básicamente te da la posibilidad de escribir y ejecutar código en cualquier lenguaje de programación”,

(Flores, 2022). Lo interesante de lo mencionado, es que es altamente personalizable y se puede instalar extensiones y complementos que se adapten a necesidades específicas con la integración de Git (facilita el control de versiones) y puede ayudar a ser más productivo en tu trabajo o desarrollo de un proyecto.

GitHub

Plataforma en línea que es utilizada principalmente por programadores y equipos de desarrollo para compartir, colaborar y administrar proyectos de software utilizando el sistema de control de versiones Git. Es una especie de red social para desarrolladores, donde se puede almacenar y organizar código de manera centralizada. En cada proyecto se pueden crear repositorios para almacenar y gestionar el código fuente de sus proyectos. Los cambios en el código se registran en "commits", lo que permite realizar un seguimiento detallado de las modificaciones a lo largo del tiempo y facilita la colaboración al permitir que múltiples usuarios trabajen en el mismo proyecto al mismo tiempo.

2.2 Propuesta de solución

Después de describir los conceptos y teorías para el desarrollo del proyecto, se comentarán las principales herramientas y tecnologías seleccionadas. Esto se hace para incluir un plan detallado de implementación, que abarcará desde el análisis y diseño inicial hasta la ejecución y puesta en marcha de la solución propuesta. Se presta especial atención a la gestión del cambio y se establecen estrategias para garantizar una transición exitosa hacia la nueva solución tecnológica. El objetivo es proporcionar a la organización una estrategia clara y efectiva para abordar los desafíos y necesidades específicas del proyecto de sistema de trámites para citas.

2.2.1 Tecnologías seleccionadas

React

Como estrategia, se decidió implementar el framework de React, ya que está ayudará a mejorar significativamente el rendimiento al actualizar y renderizar elementos en las interfaces de usuario (tanto móvil como web), así como reutilizar los componentes que se van a implementar en el sistema, y lo mejor de todo es que solo necesitas tener conceptos de los lenguajes para crear una página web (HTML, CSS, y Javascript) para manipular la herramienta FrontEnd. Se utilizará React JS para el desarrollo de aplicaciones web y React Native para la aplicación móvil.

Node JS

Se decidió utilizar Node JS, ya que esta herramienta es fundamental para poder ejecutar con éxito el framework de React a través del lenguaje de Javascript (con el comando npm), además de que posee un gran rendimiento para desarrollar aplicaciones de servidor modernas y manejar una gran cantidad de conexiones concurrentes con una utilización eficiente de los recursos.

Spring Boot

Para poder gestionar los servicios mediante API REST para realizar las acciones de las aplicaciones, se usará el framework de Spring, así como también la extensión de Spring Boot, y está va a proporcionar un enfoque rápido y sencillo para crearlos a través de las solicitudes, así como su integración de dependencias de trabajo, lo que facilita la gestión de las dependencias entre los componentes de la aplicación y esto permitirá una mejor reutilización de código y una mayor facilidad de prueba.

MYSQL

Se utilizará este gestor de base de datos como complemento a Spring Boot, ya que para la creación de las tablas se necesita la seguridad para poder proteger los datos que se registraran en el sistema, así como se manejará grandes cantidades de datos y alto tráfico de usuarios que se den de alta de manera eficiente, entre otras operaciones (se ocupará el programa SQLyog Community para trabajar de forma más rápida y cómoda, derivado de este lenguaje).

Postman

Como parte de la integración de pruebas, esta herramienta será de utilidad para poder realizar tests y verificar el funcionamiento de una solicitud (request) de los módulos a implementar en el sistema, y esto evaluará las respuestas recibidas del servidor y validar si cumplen con los criterios especificados. Además de que las colecciones que serán testeadas, se cuenta con una interfaz de usuario fácil de usar e intuitiva en la que se podrán realizar solicitudes HTTP/HTTPS sin necesidad de escribir código manualmente. Esto agilizará el proceso de prueba y desarrollo de la API en ejecución.

Visual Studio Code

Se deberá de utilizar esta herramienta, ya que es un editor ligero que se inicia rápidamente y tiene un bajo consumo de recursos (en este caso para computadora de gama baja será necesario utilizarlo por el rendimiento), además de que soportará la codificación del proyecto en los lenguajes que se van a utilizar (Java y JavaScript), y para el caso de utilizar el framework de Spring Boot, se instalará una extensión llamada "Spring Boot Extension Pack" para gestionar los módulos de control, y esta deriva de las extensiones "Spring Boot Tools" (permitirá la gestión más sencilla de las entidades

Java), “Spring Initializr” (permitirá montar el proyecto de Spring Boot al editor) y “Spring DashBoard” (permitirá gestionar de forma cómoda los proyectos desarrollados).

Github

Se optó por usar esta herramienta, para poder almacenar el código fuente del proyecto del Sistema de Gestión de Citas, esto ayudará tanto el Asesor Universitario y Asesor Empresarial como el Estudiante a poder saber el progreso de construcción del proyecto que se está gestionando, tal como la fecha y hora en la que se subió un commit (mensaje de confirmación) al repositorio y cómo fue que se realizaron los cambios.

2.2.2 Metodología seleccionada

Metodología SCRUM

Se implementará esta metodología, porque permitirá a los equipos adaptarse rápidamente a los cambios en los requisitos del proyecto y a las necesidades del cliente, así según lo requiera la empresa, además de que fomenta una comunicación efectiva entre los miembros del equipo y los interesados. También en esta metodología, hay reuniones diarias (a través de Google Meet) y las retrospectivas periódicas ayudarán a mantener a todos los integrantes informados y permiten resolver problemas de manera oportuna. En este caso, los integrantes tienen los papeles de Sponsor (persona con autoridad y recursos que respalda activamente el proyecto y tiene un interés en su éxito), Cliente (representa al usuario final quien recibirá el servicio desarrollado), Asesor (parte del rol interno que brinda experiencia técnica al equipo), Administrador del Proyecto (conocido como Scrum Master, responsable de asegurar que el equipo siga correctamente este marco de trabajo) y Colaboradores (miembros del equipo de desarrollo que trabajan activamente en la creación del servicio).

CAPÍTULO 3. DESARROLLO

3.1 Inicio

Durante la fase inicial del proyecto, se estableció con precisión el alcance completo del mismo, incluyendo los objetivos generales y específicos, así como las restricciones potenciales. Este proceso se documentó en el ANEXO A - PROJECT CHARTER, que fue fundamental para recopilar la información necesaria de los módulos del sistema a trabajar, incluyendo fechas clave de entrega, duración proyectada e hitos para la presentación de avances. Para asegurar una gestión efectiva de los indicadores críticos, se definió detalladamente el modelo que se implementaría mediante la Metodología SCRUM. Esto permitiría mantener un control adecuado sobre los elementos esenciales del proyecto. Además, se asignaron con precisión los roles de cada uno de los interesados involucrados en el desarrollo del proyecto, lo cual quedó reflejado de manera clara en la Tabla 3.1, donde se describieron los diversos actores y se especificaron sus funciones y responsabilidades respectivas.

Tabla 3.1 Involucrados y sus funciones

INVOLUCRADOS EN EL PROYECTO	
Nombre de la persona	Función
Luis Enrique Trujillo Campos	Sponsor <i>Interno</i>
Lic. David Alejandro Silva Magallón	Cliente <i>Externo</i>
T.S.U. Obed Ariel Hurtado Hernández	Asesor Empresarial <i>Externo</i>
I.T.I. Erick Mireles Merchant	Asesor Universitario <i>Interno</i>
Jorge Ángel Ruiz Juárez	Administrador del proyecto <i>Interno</i>

Luis Eduardo Bahena Castillo	Colaborador <i>Interno</i>
Andrea Michelle Estrada Hernández	Colaborador <i>Interno</i>

3.2 Planeación

Dentro de este apartado se llevó a cabo una minuciosa planificación del proceso de desarrollo del proyecto, el cual tiene como objetivo principal la creación de una innovadora aplicación web y móvil a través del Sistema de Gestión de Trámites y Citas (CITAT). Esta aplicación estará destinada a gestionar de manera eficiente y efectiva las citas de una escuela privada que fue otorgada a la prestigiosa empresa Acción TI, garantizando así una mayor productividad y satisfacción para todos sus usuarios y colaboradores. El proceso de planificación incluyó la definición detallada de los pasos a seguir en la creación de la aplicación a través de una Estructura de Desglose del Trabajo (EDT o WBS), como se muestra en la Figura 3.1.

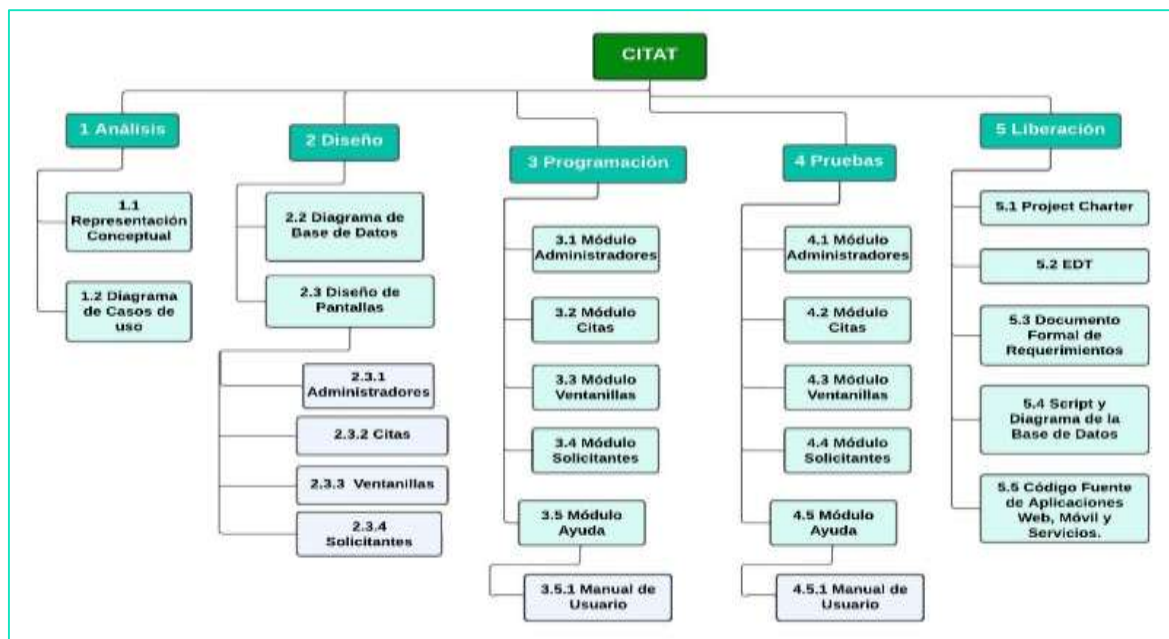


Figura 3.1 Diagrama EDT

Este esquema implica la clasificación valiosa para visualizar y organizar un proyecto en partes más pequeñas y manejables, lo que facilita la planificación, ejecución y seguimiento efectivos para cada uno de los resultados entregables, agrupándolos según las diversas fases de desarrollo del proyecto.

Cronograma de Planeación

En esta Tabla 3.1 se planteó la elaboración de un Cronograma de Planeación de acuerdo con los módulos que se van a emplear de acuerdo a la Estructura de Desglose del Trabajo para gestionar con precisión los tiempos asignados a cada una de las fases del proyecto, y con su finalidad de tener un seguimiento y control efectivos de los plazos establecidos.

N	EDT	Fase	Tarea	Días Plan	Fecha Inicio Plan	Fecha Final Plan
1	1.0	Análisis	Documento Formal de Requerimientos (DFR)	10	1/5/2023	12/5/2023
2	1.1	Análisis	Diagrama EDT	11	12/5/2023	26/5/2023
3	1.2	Análisis	Project Charter	11	12/5/2023	26/5/2023
4	2.1	Análisis	Diagrama de Casos de Uso	11	12/5/2023	26/5/2023
5	2.2	Diseño	Diagrama de Base de Datos	26	29/5/2023	3/7/2023
6	2.3.1	Diseño	Pantalla de Administradores	26	29/5/2023	3/7/2023
7	2.3.2	Diseño	Pantalla de Citas	26	29/5/2023	3/7/2023
8	2.3.3	Diseño	Pantalla de Ventanillas	26	29/5/2023	3/7/2023
9	2.3.4	Diseño	Pantalla de Solicitantes	26	29/5/2023	3/7/2023
10	2.4.1	Diseño	Informes de Citas	26	29/5/2023	3/7/2023
11	2.4.2	Diseño	Informes del Historial de Ventanillas	26	29/5/2023	3/7/2023
12	3.1.1	Programación	Módulo de Administradores	32	1/6/2023	14/7/2023
13	3.1.2	Programación	Módulo de Citas	32	1/6/2023	14/7/2023
14	3.1.3	Programación	Módulo de Ventanillas	32	1/6/2023	14/7/2023
15	3.1.4	Programación	Módulo de Solicitantes	32	1/6/2023	14/7/2023
16	3.2	Programación	Módulo Informe general de Citas	27	12/6/2023	18/7/2023
17	3.3	Programación	Módulo Manual de Usuario	27	12/6/2023	18/7/2023
18	3.3.1	Pruebas	Módulo de Administradores	20	17/7/2023	11/8/2023
19	3.3.2	Pruebas	Módulo de Citas	20	17/7/2023	11/8/2023
20	3.3.3	Pruebas	Módulo de Ventanillas	20	17/7/2023	11/8/2023
21	4.1.1	Pruebas	Módulo de Solicitantes	20	17/7/2023	11/8/2023
22	4.1.2	Pruebas	Módulo Informe general de Citas	20	17/7/2023	11/8/2023
23	4.1.3	Pruebas	Módulo Manual de Usuario	17	20/7/2023	11/8/2023
24	5.3	Liberación	Documento Formal de Requerimientos (DFR)	12	1/8/2023	16/8/2023
25	5.4	Liberación	Script y Diagrama de la Base de Datos	12	1/8/2023	16/8/2023
26	5.5	Liberación	Aplicación Móvil	12	1/8/2023	16/8/2023
27	5.5	Liberación	Aplicación Web	12	1/8/2023	16/8/2023

Figura 3.2 Cronograma de Planeación

Además, se desarrolló el ANEXO B - DOCUMENTO FORMAL DE REQUERIMIENTOS (DFR), un documento crucial que establece los módulos generales que contendrá la aplicación, asegurando de esta manera que el desarrollo del proyecto esté alineado con los objetivos y necesidades tanto de Acción TI como de la escuela privada.

El equipo de desarrollo, con gran sabiduría, optó por emplear la tecnología de Spring Boot para la creación de los servicios. Esta elección se fundamentó en la versatilidad y facilidad que brinda el framework de Spring en la construcción de las entidades y programación de la lógica y respuesta del servicio tipo REST, una solución ideal para la comunicación entre la aplicación y sus usuarios.

En cuanto a la elección del gestor de base de datos, el equipo tomó la decisión acertada de utilizar MySQL como entorno para almacenar la información. Esta elección se respaldó por la familiaridad y experiencia del equipo del gestor, lo que asegura un manejo óptimo de la base de datos durante todo el desarrollo del proyecto. El uso de SQLyog Community como plataforma permitirá una administración eficaz de la base de datos y facilitará el trabajo de forma más rápida y cómoda, derivado de este lenguaje SQL.

Finalmente, y tras un análisis detenido, el equipo de desarrollo llegó a una decisión unánime en cuanto a las tecnologías a emplear para la creación de la aplicación. Se optó por utilizar la prestigiosa librería de React, la cual se encuentra en la vanguardia del desarrollo web con React JS y de desarrollo móvil con React Native. Con gran entusiasmo, el equipo trabajará utilizando NPM como centro de comandos para la creación y ejecución del proyecto web, y la aplicación Expo CLI para la ejecución de la aplicación móvil. Esta elección asegura que el desarrollo se realice de manera ágil y efectiva, ofreciendo un alto grado de flexibilidad y eficiencia en la construcción de la aplicación.

Es relevante destacar que todas las tecnologías seleccionadas, a excepción de la base de datos, se basan en el lenguaje JavaScript, lo que aportará coherencia y homogeneidad al código del proyecto. Esta decisión permite unificar los conocimientos del equipo y garantiza un flujo de trabajo armonioso y colaborativo.

En resumen, gracias a la cuidadosa elección de tecnologías y a la minuciosa planificación del equipo de desarrollo, se ha determinado que el lenguaje de

programación óptimo para llevar a cabo este emocionante proyecto es JavaScript. Con esta sólida base tecnológica, se prevé un desarrollo exitoso y una aplicación innovadora que cumplirá a cabalidad con los requerimientos y expectativas de la empresa Acción TI, brindando así beneficios significativos para todos sus usuarios y colaboradores.

En la Tabla 3.2 se presentan los requerimientos generales del proyecto a desarrollar. Esta destaca los elementos esenciales necesarios para su desarrollo exitoso:

Tabla 3.2 Módulos a implementar en el Sistema de CITAT

No	Descripción
1	Inicio de sesión / Recuperación de Contraseña
2	Gestión de Administradores
3	Gestión de Ventanillas
4	Gestión de Solicitantes
5	Gestión de Citas

3.3 Ejecución

3.3.1 Análisis

En el transcurso de la tarea de comprender tanto la aplicación web como la variante para dispositivos móviles, resulta de vital importancia disponer de una Representación Conceptual que nos ofrezca una perspectiva holística de su operatividad. Dicha representación, presentada en la Figura 3.3, nos brinda la oportunidad de contemplar de manera visual el contexto operacional y funcional en el cual ambas aplicaciones desempeñarán sus funciones. encapsula en su diseño los distintos módulos y componentes esenciales que dan forma a estas aplicaciones. Desde la perspectiva de la aplicación web, se pueden identificar los componentes clave que permiten la interacción con los usuarios a través de navegadores, así como la gestión de bases de datos y la lógica de negocios subyacente.

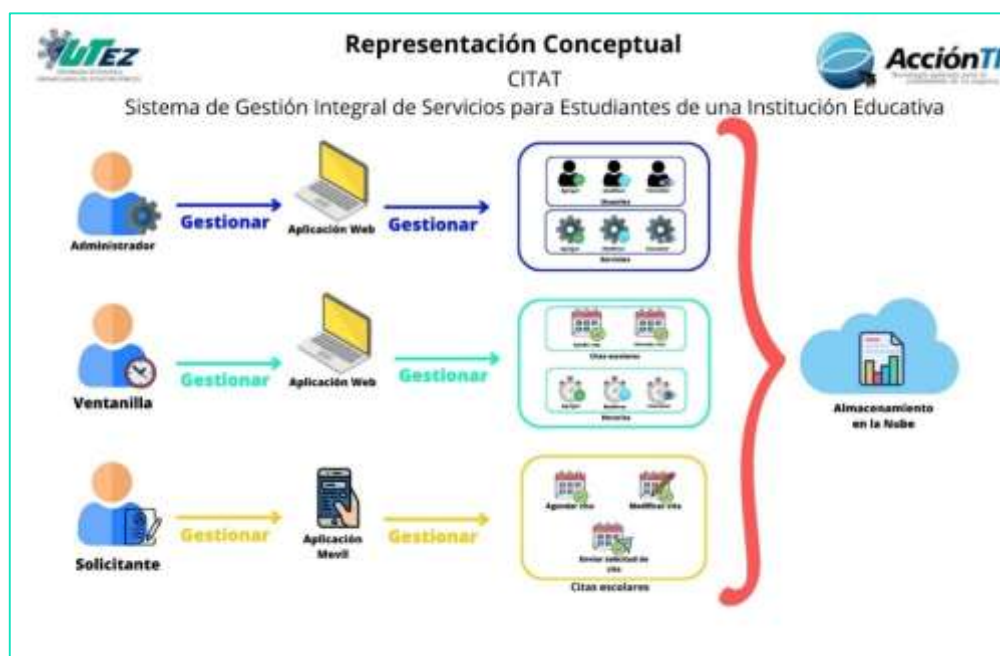


Figura 3.3 Representación Conceptual

La Representación Conceptual juega un papel esencial al desentrañar la forma en que los distintos elementos del sistema interactúan y establecen vínculos mutuos. Gracias a esta representación, tenemos la capacidad de identificar los flujos de información, las características fundamentales y las conexiones que emergen entre los usuarios y las interfaces que se emplearán en el Sistema de Citas.

En continuidad con la explicación previamente proporcionada, se procede a la elaboración del Diagrama de Casos de Uso, una herramienta que nos permitirá adentrarnos en las funciones respectivas de cada usuario y en las limitaciones que los rodean en el contexto de la aplicación.

El enfoque principal recae en la creación del Módulo Administrador, cuya tarea principal consistirá en administrar a los usuarios roles de Ventanilla y Solicitantes, además de supervisar los servicios educativos ofrecidos. Su función principal radica en garantizar la eficiencia en las interacciones entre los solicitantes y los servicios, manteniendo intacta la integridad y la seguridad del sistema en todo momento.

Por último, el Módulo Solicitante emerge como un componente esencial, ya que posee la capacidad de acceder al calendario de citas, registrar y adjuntar documentos, así como efectuar pagos relacionados con los servicios, todo desde un único espacio. Esta consolidación de funciones no solo mejora la experiencia del usuario, sino que también optimiza los procedimientos dentro del sistema. El diagrama completo de estas operaciones se encuentra detallado en la Figura 3.4.



3.3.2 Diseño

Base de datos. En esta etapa del procedimiento, se enfoca en la fase de concepción, elaboración y ejecución de la base de datos que desempeñará el papel de depósito centralizado y meticulosamente estructurado para la recolección de datos en el marco del sistema de citas. Está asentada infraestructura se encuentra representada de manera visual en la Figura 3.5 que proporciona una representación gráfica de la esencia del proceso de construcción de la base de datos.

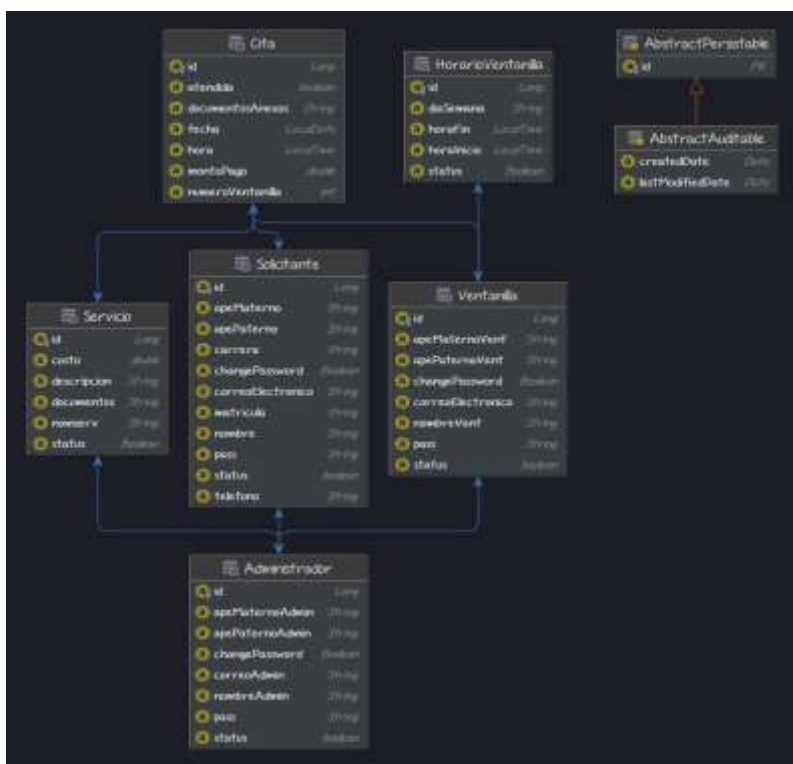


Figura 3.5 Diseño de Tablas de la Base de Datos

Pantallas de Aplicación Web

Vista Inicio de Sesión. El Módulo Administrador necesita suministrar sus credenciales para acceder al perfil, habitualmente compuestas por un nombre de usuario y una contraseña exclusiva y sólida. Estas credenciales se utilizan para acceder a la Aplicación Web, tal como se ilustra en la Figura 3.6.



Figura 3.6 Diseño de Vista Inicio de Sesión

Vista Menú. Dentro de este menú, se encuentra una herramienta de vital importancia para la gestión de los variados roles de usuarios categorizados como Usuarios y Servicios. Al ofrecer un acceso meticuloso y simplificado a las funciones exclusivas de cada rol, el administrador tiene la capacidad de supervisar y mantener un control óptimo sobre el desempeño y la funcionalidad integral del sistema. Este proceso se visualiza de manera ejemplificada en la Figura 3.3.4.



Figura 3.7 Diseño de Vista Menú Administrador

Vista Usuario. Dentro de esta sección del menú en el apartado correspondiente a usuarios, se encuentra una herramienta de suma importancia para administrar de manera eficaz los diversos roles de usuarios en el sistema, los cuales se dividen en categorías tales como Administrador, Ventanilla y Solicitante. Esta herramienta, al

ofrecer un acceso meticuloso y simplificado, facilita la creación de usuarios dentro del sistema de manera organizada. Este proceso se ilustra claramente en la Figura 3.8.



Figura 3.8 Diseño de Vista Usuario

Vista Crud Administradores. Se trata de una interfaz especialmente elaborada para la administración efectiva de usuarios con capacidades de gestión en un sistema. A través de esta visualización, los administradores tienen la capacidad de llevar a cabo acciones cruciales, como la creación de nuevos administradores, la modificación de datos preexistentes y la eliminación de registros de administradores que ya no sean necesarios. Esta operativa queda representada gráficamente en la Figura 3.9



Figura 3.9 Diseño de Vista Crud Administradores

Vista Crud Solicitantes. Esta interfaz ha sido diseñada con el propósito de administrar de manera eficaz a los usuarios que cuentan con el rol de solicitante en el sistema. A

través de esta vista, los usuarios con este rol pueden llevar a cabo acciones esenciales, como la creación de nuevos solicitantes, la edición de información ya registrada y la eliminación de registros de solicitantes que ya no sean necesarias. Esta dinámica está representada gráficamente en la Figura 3.10.



Figura 3.10 Diseño de Vista Crud Solicitantes

Vista Crud Servicios. Es la interfaz concebida para permitir que los administradores administren de manera eficiente los servicios proporcionados en el sistema. Mediante esta visualización, los administradores tienen la capacidad de establecer servicios nuevos, modificar la información de aquellos ya existentes y descartar aquellos que ya no sean de utilidad. Esta operación se presenta gráficamente en la Figura 3.11.



Figura 3.11 Diseño de Vista Crud Servicios

Vista Gestión Citas. Es la interfaz diseñada con el propósito de permitir que los empleados en esa función puedan acceder y visualizar de manera rápida y sencilla los detalles de las citas programadas. A través de esta visualización, los usuarios en este rol pueden acceder a información clave de cada cita, brindando únicamente los datos necesarios para que puedan mantener un manejo efectivo de las citas y ofrecer un servicio eficaz a los clientes. Esta se presenta en la Figura 3.12.

No. Ventanilla	Fecha	Hora	Documentos	Monto Pago	Estado
1	2023-07-01	09:30	Credencial y matrícula	150	Resuelto

Figura 3.12 Diseño de Vista Gestión Citas

Vista Gestión Horarios. Esta interfaz se adapta al usuario ventanilla donde permite que los empleados que trabajan en ventanilla puedan administrar los horarios de atención de manera eficiente. Desde esta vista, el usuario ventanilla puede visualizar y editar los horarios disponibles para citas y servicios. Así como se muestra en la Figura 3.13.

Día de la semana	Hora de Inicio	Hora de Fin	Estado
Sunday	09:00	09:30	Activo
Monday	11:00	12:00	Activo

Figura 3.13 Diseño de Vista Gestión Horarios

Pantallas de Aplicación Móvil

Vista Reservar Cita. La plataforma incluye un par de menús desplegables; el primero se destina a que el usuario elija su horario preferido, mientras que el segundo le permite seleccionar el tipo de trámite que está por llevar a cabo. Con base en la selección de trámite, se exhibirá tanto el monto requerido para el proceso como la lista de documentos necesarios para proceder con éxito. Una vez que todos los pasos se han completado, el usuario estará en condiciones de agendar su cita. Esta dinámica se ilustra en la Figura 3.14, ofreciendo una representación visual de la secuencia.



Figura 3.14 Diseño de Vista Reservar Cita

Vista Citas registradas. La visualización presenta un despliegue de las fechas programadas para las citas, acompañadas de una descripción del tema a tratar. Además, se detallan los documentos necesarios, el mostrador designado para la

atención, y el costo asociado, indicando si el pago se ha realizado o no. Justo debajo de cada cita, se brinda la posibilidad de cancelarla en caso necesario. Esta presentación es ilustrada en la Figura 3.15, mostrando gráficamente el diseño de esta disposición.



Figura 3.15 Diseño de Vista Citas registradas

Correo de confirmación. Una vez que el usuario de cualquier rol introduce sus credenciales de correo electrónico y contraseña para poderse registrar, recibe un mensaje de bienvenida en la pantalla principal del sistema y se envía al correo registrado. Estas credenciales necesitan haber sido registradas con antelación para habilitar la entrada al sistema. También es relevante mencionar que el logotipo del sistema se encuentra visible en la interfaz, lo que proporciona a los usuarios una identificación visual evidente mientras interactúan con la plataforma. El ejemplo de una

prueba llevada a cabo para verificar la confirmación del correo electrónico se ilustra en la Figura 3.16.



Figura 3.16 Correo de Confirmación de Creación de Cuenta

Estado de la cita (Ventanilla). Este componente en cuestión se refiere al apartado de citas programadas en la ventanilla, el cual presenta un listado de citas en cola que aguardan ser atendidas. Una vez que una cita ha sido atendida, se proporciona acceso a los datos pertinentes. Es crucial resaltar que en esta vista no se añade información adicional, ya que esta interfaz se reserva exclusivamente para el solicitante, como se muestra en la Figura 3.17.



Figura 3.17 Estado de las Citas

Cambiar Contraseña. En esta interfaz se puede observar la creación de un usuario, en la que posteriormente se realiza una modificación de contraseña en caso de que este no lo recuerde, como se muestra en la Figura 3.18, donde al momento de seleccionar a un usuario registrado, cuando se modifica posteriormente da clic a la leyenda “Modificar Contraseña” para seguir el proceso.

The screenshot shows a web application interface titled 'Administrador'. It features a light blue sidebar on the left with a user icon and a gear icon. The main content area is white and contains a form for user management. The form has several input fields: 'Nombre(s)' with the value 'Andrea Michele', 'Apellido Materno' with 'Estrada', 'Apellido Paterno' with 'Ramirez', 'Correo Institucional' with 'micheferubick15@gmail.com', and 'Contraseña' with a masked value. A red 'Inactivo' button is located next to the user icon. A 'Modificar contraseña' button is positioned at the bottom right of the form.

Figura 3.18 Cambiar Contraseña

Modal Citas. En esta interfaz se puede observar la creación del modal para poder visualizar el registro de una cita por parte del solicitante, en la Figura 3.19 se puede observar como el modal muestra los datos de la fecha y hora, documentos requeridos, la ventanilla que brindará el servicio y la cantidad por el servicio

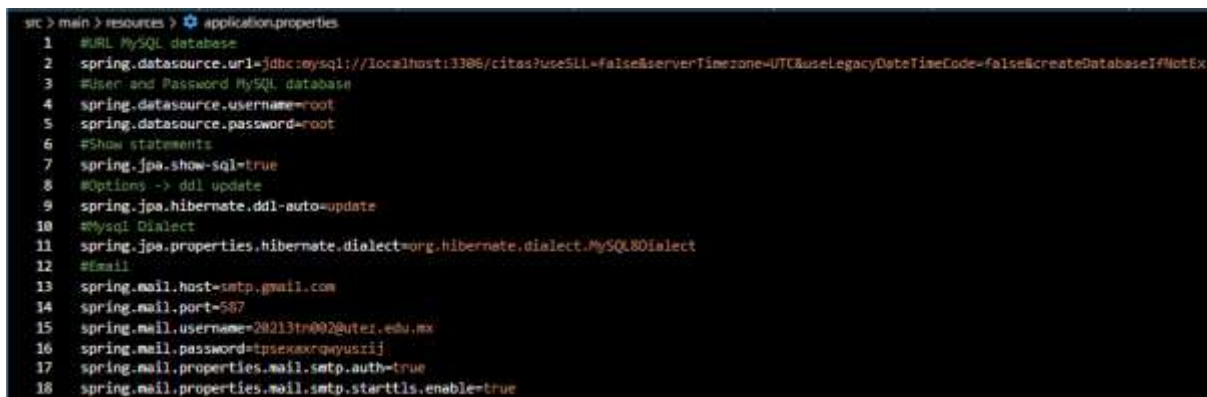
The screenshot displays a modal window titled 'Cita por atender'. It contains a table with the following data: 'No. Ventanilla' (1), 'Fecha' (2023-07-03), 'Hora' (10:30), 'Documentos requeridos' (Credencial y matricula), and 'Monto' (150). At the bottom of the modal, there are two buttons: 'CERRAR' (Close) and 'ATENDER' (Attend).

Figura 3.19 Modal Citas

3.3.3 Programación

Asignación de módulos. Para dar inicio a la fase de codificación, tanto para la aplicación web como la versión móvil, se llevó a cabo una reunión en la que participaron el asesor empresarial y el equipo de desarrollo que se asignó. En esta sesión, se establecieron los lineamientos y enfoques para el proceso de desarrollo en donde una de ellas fue un horario de trabajo de casi 5 horas al día y se tomó la decisión de asignar a un compañero especializado en diseño digital para realizar la tarea de crear prototipos para la interfaz de usuario dirigida al cliente. Asimismo, se planteó una estrategia en la que todos estaríamos al frente del proyecto, y se determinó que cada uno de nosotros se enfocaría en áreas específicas del proyecto: en este caso, me hice responsable de la parte de la creación del BackEnd y tuve la oportunidad de poner desarrollar el módulo de Ventanilla.

Configuración de la base de datos. Este conjunto de ajustes se emplea con el propósito de crear una conexión directa con la base de datos MySQL y para establecer las preferencias en la aplicación construida utilizando el framework Spring, tal y como se observa en la representación visual presentada en la Figura 3.20.



```
src > main > resources > application.properties
1  #URL MySQL database
2  spring.datasource.url=jdbc:mysql://localhost:3306/citas?useSSL=false&serverTimezone=UTC&useLegacyDateTimeCode=false&createDatabaseIfNotExist=true
3  #User and Password MySQL database
4  spring.datasource.username=root
5  spring.datasource.password=root
6  #Show statements
7  spring.jpa.show-sql=true
8  #Options -> ddl update
9  spring.jpa.hibernate.ddl-auto=update
10 #MySQL Dialect
11 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
12 #Email
13 spring.mail.host=smtp.gmail.com
14 spring.mail.port=587
15 spring.mail.username=20213tr002@utez.edu.mx
16 spring.mail.password=tpsexaxrqayusij
17 spring.mail.properties.mail.smtp.auth=true
18 spring.mail.properties.mail.smtp.starttls.enable=true
```

Figura 3.20 Conexión a la Base de Datos

La dirección URL determina la ubicación y el puerto para la conexión a la base de datos, e incluye otros ajustes, como la activación de SSL, la definición de la zona horaria y la posibilidad de generar automáticamente la base de datos en caso de que

no exista alguno. Además, se especifican los datos de autenticación necesarios para acceder al servidor MySQL y se permite la visualización de las sentencias SQL con el fin de simplificar el proceso de seguimiento y depuración. Las configuraciones "ddl-auto" y "dialect" otorgan la capacidad a Hibernate de actualizar la estructura de la base de datos de manera automática y de emplear el dialecto adecuado para la comunicación. Por último, se realiza la configuración del servicio de correo electrónico, el cual utilizará el protocolo SMTP de Gmail para el envío de notificaciones a través de una cuenta existente de Gmail.

Módulo del Administrador. A partir de los requerimientos para este módulo, se diseñó de acuerdo a sus atributos correspondientes, en este caso fueron el nombre, apellidos, correo, contraseña y el status en el que se encuentra el administrador. Como se puede ver en la Figura 3.21, se establece una relación @OneToMany, que se utiliza para establecer una relación uno a muchos entre las entidades de Ventanilla, Solicitante y Servicio, esto para que el parámetro "mappedBy" especifica el mapeo de las entidades en relación inversa para poder acceder a estas mismas.

```
17 @Entity
18 @Table(name = "admins")
19 @AllArgsConstructor
20 @NoArgsConstructor
21 @Setter
22 @Getter
23 @IdGeneratorInfo(
24     generator = ObjectIdGenerators.PropertyGenerator.class,
25     property = "id"
26 )
27 public class Admin {
28     @Id
29     @GeneratedValue(strategy = GenerationType.IDENTITY)
30     private long id;
31     @Column(name = "nombreAdmin", nullable = false)
32     private String nombreAdmin;
33     @Column(name = "apePaternoAdmin", nullable = false)
34     private String apePaternoAdmin;
35     @Column(name = "apeMaternoAdmin") //No todos tienen dos apellidos
36     private String apeMaternoAdmin;
37     @Column(name = "correoAdmin", nullable = false, unique = true)
38     private String correoAdmin;
39     @Column(name = "passwordAdmin", nullable = false)
40     private String passwordAdmin;
41     @Column(nullable = false, columnDefinition = "tinyint default 1")
42     private Boolean status;
43
44     ///Gestiona Ventanillas
45     @OneToMany(mappedBy = "admin")
46     private List<Ventanilla> ventanillas;
47
48     ///Gestiona solicitantes
49     @OneToMany(mappedBy = "admin")
50     private List<Solicitante> solicitantes;
51
52     ///Gestiona servicios
53     @OneToMany(mappedBy = "admin")
54     private List<Servicio> servicios;
55 }
```

Figura 3.21 Entidad Administrador

Con el objetivo de agilizar el flujo de información entre la capa de interfaz y el controlador, se diseñó la clase "AdminDto" como un elemento específico de transferencia de datos (DTO). Esta entidad se creó con la finalidad de mejorar la eficiencia en la comunicación entre estas dos capas, facilitando así el intercambio de datos de manera más efectiva. Adicionalmente, se incorporaron anotaciones de validación en la clase para garantizar que los datos introducidos se ajusten a ciertos criterios predefinidos. La Figura 3.22 ofrece una representación visual de este proceso.

```
15 @AllArgsConstructor
16 @NoArgsConstructor
17 @Setter
18 @Getter
19 public class AdminDto {
20
21     private Long id;
22     private String nombreAdmin;
23     private String apeMaternoAdmin; //No todos tienen dos apellidos
24     private String apePaternoAdmin;
25     //@ValidEmailDomain
26     private String correoAdmin;
27     private String passwordAdmin;
28     private Boolean status;
29
30     private List<Ventanilla> ventanillas;
31     private List<Solicitante> solicitantes;
32     private List<Servicio> servicios;
33
34     public Admin getAdmin(){
35         return new Admin(
36             getId(),
37             getNombreAdmin(),
38             getApeMaternoAdmin(),
39             getApePaternoAdmin(),
40             getCorreoAdmin(),
41             getPasswordAdmin(),
42             getStatus(),
43             getVentanillas(),
44             getSolicitantes(),
45             getServicios()
46         );
47     }
48 }
```

Figura 3.22 Clase AdminDto

La función del servicio consiste en desempeñar el rol de intermediario entre el controlador y el repositorio del módulo. Dentro de esta clase se encuentran implementados diversos métodos que facilitan la realización de las acciones demandadas, tales como recuperar el listado de administradores (tanto si están activos o no), eliminar o modificar un registro, activar o desactivar la cuenta y enviar correos electrónicos de confirmación al momento de realizar un nuevo registro. Estos

procedimientos se encuentran ilustrados visualmente en la Figura 3.23 donde se retornan mediante un objeto de la Clase “CustomResponse” (es la encargada de validar errores internos que ocurran en la ejecución del BackEnd).

```
15 @Service
16 @Transactional
17 public class AdminService {
18     @Autowired
19     private AdminRepository adminRepository;
20
21     @Transactional(readOnly = true)
22     public CustomResponse<List<Admin>> getAll(){
23         return new CustomResponse<>{
24             this.adminRepository.findAll(),
25             error: false,
26             StatusCode: 200,
27             message: "ok"
28         };
29     }
30
31     ///Servicio para los activos
32     @Transactional(readOnly = true)
33     public CustomResponse<List<Admin>> getAllActive(){
34         return new CustomResponse<>{
35             this.adminRepository.findAllByStatus(status: true),
36             error: false,
37             StatusCode: 200,
38             message: "ok"
39         };
40     }
41
42     ///Servicio para los inactivos
43     @Transactional(readOnly = true)
44     public CustomResponse<List<Admin>> getAllInactive(){
45         return new CustomResponse<>{
46             this.adminRepository.findAllByStatus(status: false),
47             error: false,
48             StatusCode: 200,
49             message: "ok"
50         };
51     }
52 }
```

Figura 3.23 Servicio Administrador

Dentro del marco de la clase AdminController, se lleva a cabo la implementación del controlador de API Rest, destinado a llevar a cabo las operaciones CRUD y administrar los datos correspondientes a los solicitantes. En este contexto, se han establecido variadas rutas y se han asignado métodos HTTP específicos para acceder a los recursos vinculados a los solicitantes, tal como se ve en la Figura 3.24 para acceder a una ruta correspondiente de acuerdo a su petición que se va a realizar.

```

17 @RestController
18 @RequestMapping("/api/admin/")
19 @CrossOrigin(origins = {"*"})
20 public class AdminController {
21
22     @Autowired
23     private AdminService adminService;
24     // @PreAuthorize("hasRole('ADMIN')")
25     @GetMapping("/")
26     public ResponseEntity<CustomResponse<List<Admin>>> getAll(){
27         return new ResponseEntity<>{
28             this.adminService.getAll(),
29             HttpStatus.OK
30         };
31     }
32
33     @GetMapping("/getActive")
34     public ResponseEntity<CustomResponse<List<Admin>>>
35     getAllActive(){
36         return new ResponseEntity<>{
37             this.adminService.getAllActive(),
38             HttpStatus.OK
39         };
40     }
41
42     @GetMapping("/getAllInactive")
43     public ResponseEntity<CustomResponse<List<Admin>>>
44     getAllInactive(){
45         return new ResponseEntity<>{
46             this.adminService.getAllInactive(),
47             HttpStatus.OK
48         };
49     }

```

Figura 3.24 Controller del Administrador

Módulo de Ventanilla. Basándonos en los requisitos específicos de este módulo, se procedió a la etapa de diseño, cuidadosamente configurado según sus atributos esenciales. Entre estos atributos se incluyen elementos fundamentales como el nombre, los apellidos, la dirección de correo electrónico, la contraseña, y también el estado de la ventanilla. Una relación del tipo @OneToMany fue implementada con el fin de establecer conexiones sólidas de uno a muchos en la entidad de Citas. Además, se introduce con éxito una relación de muchos a uno (@ManyToOne), esta última representa un eslabón vital que permite al Administrador desempeñar su función de supervisión y control sobre los usuarios que operan en las ventanillas. Esta trascendente relación se encuentra visualizada de manera elocuente en la Figura 3.25.

```

16  @Entity
17  @Table(name = "ventanillas")
18  @AllArgsConstructor
19  @NoArgsConstructor
20  @Setter
21  @Getter
22  @JsonIdentityInfo(
23      generator = ObjectIdGenerators.PropertyGenerator.class,
24      property = "id"
25  )
26  public class Ventanilla {
27
28      @Id
29      @GeneratedValue(strategy = GenerationType.IDENTITY)
30      private Long id;
31      @Column(name = "nombreVent", nullable = false)
32      private String nombre;
33      @Column(name = "apeMaternoVent", nullable = false)
34      private String apeMaterno;
35      @Column(name = "apePaternoVent") //No todos tienen dos apellidos
36      private String apePaterno;
37      @Column(name = "correoVent", nullable = false, unique = true)
38      private String correoVent;
39      @Column(name = "passwordVent", nullable = false)
40      private String password;
41
42      @Column(nullable = false, columnDefinition = "tinyint default 1")
43      private Boolean status;
44
45      @OneToMany(mappedBy = "ventanilla")
46      private List<Cita> citas;
47
48      @ManyToOne
49      @JoinColumn(name = "admin_id", nullable = false)
50      @JsonBackReference
51      private Admin admin;
52  }

```

Figura 3.25 Entidad Ventanilla

El propósito de la creación de la clase "VentanillaDto" fue la optimización de la comunicación, asegurando su agilidad y eficiencia. Para mantener la coherencia y la integridad de los datos en un nivel óptimo, se dotó a esta entidad de elementos adicionales en forma de anotaciones de validación. Estas anotaciones se aplican minuciosamente para certificar que cualquier información introducida esté perfectamente alineada con los criterios establecidos de antemano, generando una precisión y exactitud notables en la entrada de datos. La representación visual de este proceso, plasmada en la Figura 3.26, ofrece un esclarecedor vistazo de cómo estas anotaciones de validación fortalecen la calidad y la idoneidad de los datos transmitidos, fortificando la confiabilidad de la comunicación.

```

15  @AllArgsConstructor
16  @NoArgsConstructor
17  @Getter
18  @Setter
19  public class VentanillaDto {
20
21      private Long id;
22      private String nombre;
23      private String apeMaterno;
24      private String apePaterno;
25      private String correo;
26      private String password;
27      private Boolean status;
28
29      private List<Cita> citas;
30      private Admin admin;
31
32      public Ventanilla getVentanilla(){
33          return new Ventanilla(
34              getId(),
35              getNombre(),
36              getApeMaterno(),
37              getApePaterno(),
38              getCorreo(),
39              getPassword(),
40              getStatus(),
41              getCitas(),
42              getAdmin()
43          );
44      }
45  }

```

Figura 3.26 Clase VentanillaDto

El servicio cumple la función de intermediario entre controlador y repositorio del módulo. Este contiene diversos métodos que simplifican acciones como obtener la lista de ventanillas (activas o no), modificar o eliminar registros, habilitar o deshabilitar cuentas y enviar correos de confirmación en el momento de nuevos registros. Una representación visual en la Figura 3.27 donde demuestra estos procesos mediante el objeto de la Clase "CustomResponse".

```

14  @Service
15  @Transactional
16  public class VentanillaService {
17
18      @Autowired
19      private VentanillaRepository ventanillaRepository;
20
21      ///Todos las ventanillas
22      @Transactional(readOnly = true)
23      public CustomResponse<List<Ventanilla>> getAll(){
24          return new CustomResponse<(){
25              this.ventanillaRepository.findAll(),
26              error: false,
27              StatusCode: 200,
28              message: "Ok"
29          };
30      }
31
32      ///Servicio para los activos
33      @Transactional(readOnly = true)
34      public CustomResponse<List<Ventanilla>> getAllActive(){
35          return new CustomResponse<(){
36              this.ventanillaRepository.findAllByStatus(status: true),
37              error: false,
38              StatusCode: 200,
39              message: "ok"
40          };
41      }

```

Figura 3.27 Servicio Ventanilla

En el contexto de VentanillaController, se implementa el controlador de API Rest para realizar operaciones CRUD y administrar datos de ventanillas. Rutas variadas y métodos HTTP específicos se asignan para acceder a recursos relacionados con las ventanillas, evidenciado en la Figura 3.28.

```
17 @RestController
18 @RequestMapping("/api/ventanillas/")
19 @CrossOrigin(origins = {"*"})
20 public class VentanillaController {
21
22     @Autowired
23     private VentanillaService ventanillaService;
24
25     // @PreAuthorize("hasRole('VENTANILLA')")
26     @GetMapping("/")
27     public ResponseEntity<CustomResponse<List<Ventanilla>>> getAll(){
28         return new ResponseEntity<>{
29             this.ventanillaService.getAll(),
30             HttpStatus.OK
31         };
32     }
33
34     @GetMapping("/getActive")
35     public ResponseEntity<CustomResponse<List<Ventanilla>>>
36     getAllActive(){
37         return new ResponseEntity<>{
38             this.ventanillaService.getAllActive(),
39             HttpStatus.OK
40         };
41     }
42
43     @GetMapping("/getAllInactive")
44     public ResponseEntity<CustomResponse<List<Ventanilla>>>
45     getAllInactive(){
46         return new ResponseEntity<>{
47             this.ventanillaService.getAllInactive(),
48             HttpStatus.OK
49         };
50     }
51 }
```

Figura 3.28 Controller Ventanilla

Módulo de Solicitante. Apoyados en los requerimientos específicos delineados para este módulo, se procedió a un proceso de diseño que abrazó en su totalidad los atributos fundamentales inherentes. Entre estos atributos se encuentran elementos de importancia vital tales como el nombre, la matrícula, los apellidos, la dirección de correo electrónico, la contraseña, el número telefónico y el estado en el cual el solicitante se encuentra en el sistema. Una estrategia de diseño eficaz se perfiló con la creación de una relación del tipo @OneToMany, esta conexión se dirige como un enlace robusto que establece una relación de uno a muchos con la entidad de Citas. A esta configuración se añade otra capa de relación mediante @ManyToOne, una vez más, con el fin de que el Administrador asuma el control y la gestión de los usuarios ligados a los solicitantes. Esta mecánica se visualiza y se compendia de manera

gráfica en la representación que ofrece la Figura 3.29.

```
17 @Entity
18 @Table(name = "solicitanter")
19 @AllArgsConstructor
20 @NoArgsConstructor
21 @Setter
22 @Getter
23 @JsonIdentityInfo(generator = ObjectIdGenerators.PropertyGenerator.class, property = "id")
24 public class Solicitante {
25
26     @Id
27     @GeneratedValue(strategy = GenerationType.IDENTITY)
28     private Long id;
29     @Column(name = "matricula", nullable = false, unique = true)
30     private String matricula;
31     @Column(name = "nombre", nullable = false)
32     private String nombre;
33     @Column(name = "apeMaterno", nullable = false)
34     private String apeMaterno;
35     @Column(name = "apePaterno", nullable = false)
36     private String apePaterno;
37     @Column(name = "correoSoli", nullable = false, unique = true)
38     private String correoSoli;
39     @Column(name = "passwordSoli", nullable = false)
40     private String passwordSoli;
41     @Column(name = "telefono", nullable = false)
42     private int telefono;
43     @Column(nullable = false, columnDefinition = "tinyint default 1")
44     private Boolean status;
45
46     @ManyToOne
47     @JoinColumn(name = "admin_id", nullable = false)
48     @JsonBackReference
49     private Admin admin;
50
51     @OneToMany(mappedBy = "solicitante", cascade = CascadeType.ALL)
52     private List<Cita> citas;
53 }
```

Figura 3.29 Entidad Solicitante

Con el objetivo primordial de impulsar y dinamizar la transferencia de información esencial entre la capa de interfaz y el controlador, se procedió a la creación meticulosa de la clase "SolicitanteDto". Este componente específico, en la forma de un DTO, se forjó como una herramienta integral diseñada con el fin de amplificar de manera significativa la eficiencia en la interacción entre estas dos capas cruciales del sistema. La creación y aplicación de este elemento específico fue dirigida a optimizar, de manera sobresaliente, el flujo de datos, generando una sinergia más efectiva en la comunicación. Asimismo, se implementaron con esmero anotaciones de validación en la mencionada clase. Estas anotaciones cumplen un papel fundamental al asegurar que todos los datos ingresados cumplan con los rigurosos criterios predefinidos, agregando un nivel adicional de rigurosidad y calidad en la entrada de información, véase la Figura 3.30 para más precisión.

```

15  @AllArgsConstructor
16  @NoArgsConstructor
17  @Getter
18  @Setter
19  public class SolicitanteDto {
20
21      private Long id;
22      private String matricula;
23      private String nombre;
24      private String apeMaterno;
25      private String apePaterno;
26      private String correo;
27      private String passwordSoli;
28      private int telefono;
29      private Boolean status;
30
31      private Admin admin;
32
33      private List<Cita> citas;
34
35      public Solicitante getSolicitante(){
36          return new Solicitante(
37              getId(),
38              getMatricula(),
39              getNombre(),
40              getApeMaterno(),
41              getApePaterno(),
42              getCorreo(),
43              getPasswordSoli(),
44              getTelefono(),
45              getStatus(),
46              getAdmin(),
47              getCitas()
48          );
49      }
50  }

```

Figura 3.30 Clase SolicitanteDto

En el proceso integral del flujo de datos, el servicio emerge como un punto de conexión esencial entre el controlador y el repositorio. En esta crucial clase, se ejecutan y despliegan una variedad de métodos estratégicos, diseñados con el propósito primordial de simplificar y agilizar la ejecución de acciones de vital importancia. Entre estas acciones se encuentran la obtención de la lista de solicitantes, abarcando tanto los activos como los inactivos, la habilidad para modificar o eliminar registros, además de la capacidad para habilitar o deshabilitar cuentas según sea necesario. Una función que destaca es la habilidad para emitir correos electrónicos de confirmación en el momento de registrar nueva información.

Este proceso de funcionalidad y ejecución está gráficamente representado en la Figura 3.31, donde la ilustración captura de manera visual cómo estos procedimientos

interactúan y se desarrollan de manera coordinada. Además, estos procedimientos son canalizados y presentados a través de un componente fundamental, el objeto de la clase "CustomResponse", el cual asume la importante responsabilidad de validar y gestionar cualquier error interno que pudiera surgir en la ejecución del BackEnd.

```
14 @Service
15 @Transactional
16 public class SolicitanteService {
17
18     @Autowired
19     private SolicitanteRepository solicitanteRepository;
20
21     ///Todos los solicitantes
22     @Transactional(readonly = true)
23     public CustomResponse<List<Solicitante>> getAll(){
24         return new CustomResponse<>{
25             this.solicitanteRepository.findAll(),
26             error: false,
27             StatusCode: 200,
28             message: "Ok"
29         };
30     }
31
32     ///Servicio para los activos
33     @Transactional(readonly = true)
34     public CustomResponse<List<Solicitante>> getAllActive(){
35         return new CustomResponse<>{
36             this.solicitanteRepository.findAllByStatus(status: true),
37             error: false,
38             StatusCode: 200,
39             message: "ok"
40         };
41     }
42 }
```

Figura 3.31 Servicio Solicitante

Entrando en el ámbito de la clase SolicitanteController, la acción de implementar el controlador de la API Rest se materializa. Este controlador, que opera en un contexto específico, desempeña un papel crucial en la ejecución de operaciones CRUD y la administración precisa de datos vinculados con los solicitantes. La configuración en este escenario es minuciosa, con la asignación de rutas específicas y la designación de métodos HTTP apropiados para acceder a los recursos íntimamente relacionados con los solicitantes. Este diseño se visualiza y se estructura de manera precisa en la figura 2, brindando una representación gráfica tangible de cómo esta etapa concreta se despliega en la realidad del proceso global.


```

17 @RestController
18 @RequestMapping("/api/solicitantes/")
19 @CrossOrigin(origins = {"*"})
20 public class SolicitanteController {
21
22     @Autowired
23     private SolicitanteService solicitanteService;
24
25     // @PreAuthorize("hasRole('SOLICITANTE')")
26     @GetMapping("/")
27     public ResponseEntity<CustomResponse<List<Solicitante>>> getAll(){
28         return new ResponseEntity<> (
29             this.solicitanteService.getAll(),
30             HttpStatus.OK
31         );
32     }
33
34     @GetMapping("/getActive")
35     public ResponseEntity<CustomResponse<List<Solicitante>>>
36     getAllActive(){
37         return new ResponseEntity<> (
38             this.solicitanteService.getAllActive(),
39             HttpStatus.OK
40         );
41     }
42
43     @GetMapping("/getAllInactive")
44     public ResponseEntity<CustomResponse<List<Solicitante>>>
45     getAllInactive(){
46         return new ResponseEntity<> (
47             this.solicitanteService.getAllInactive(),
48             HttpStatus.OK
49         );
50     }

```

Figura 3.32 Controller Solicitante

Módulos de Servicio, Citas y Horarios. Los requerimientos que fundamentan los subsiguientes módulos presentan una semejanza intrínseca. Tomando como base estos requerimientos, se delinearon configuraciones que comparten una estructura básica.

En el caso de la entidad "Servicio", el diseño se forjó con una cuidadosa consideración de sus atributos característicos, los cuales abarcan elementos como el nombre, la descripción, los documentos asociados, el costo y el estado del servicio. Esta estructura se refleja con claridad en la Figura 3.33 que se presenta. De forma destacada, se implementa una relación del tipo @OneToMany, la cual establece un vínculo de uno a muchos con la entidad de "Citas". Dentro de esta relación, el parámetro "mappedBy" desempeña un rol crucial al especificar cómo se mapean las entidades en una relación inversa, permitiendo un acceso fluido a estas instancias. A

este entramado se agrega una dimensión adicional con la incorporación de la relación @ManyToOne, que facultará a los Administradores asumir la gestión y el control de los servicios escolares.

```
16 @Entity
17 @Table(name = "servicios")
18 @AllArgsConstructor
19 @NoArgsConstructor
20 @Setter
21 @Getter
22 @JsonIdentityInfo(
23     generator = ObjectIdGenerators.PropertyGenerator.class,
24     property = "id"
25 )
26 public class Servicio {
27     @Id
28     @GeneratedValue(strategy = GenerationType.IDENTITY)
29     private Long id;
30     @Column(name = "nomserv", nullable = false)
31     private String nomserv;
32     @Column(name = "descripcion", nullable = false)
33     private String descripcion;
34     @Column(name = "documentos", nullable = false)
35     private String documentos;
36     @Column(name = "costo", nullable = false)
37     private int costo;
38
39     @Column(nullable = false, columnDefinition = "tinyint default 1")
40     private Boolean status;
41
42     @OneToMany(mappedBy = "servicio")
43     private List<Cita> citas;
44
45     @ManyToOne
46     @JoinColumn(name = "admin_id", nullable = false)
47     @JsonBackReference
48     private Admin admin;
49 }
```

Figura 3.33 Entidad Servicio

En el contexto de la entidad "Cita", el proceso de diseño siguió una línea similar al considerar sus atributos intrínsecos. Estos atributos comprenden la fecha y la hora de la cita, el monto de pago asociado y el estado de la cita. Este diseño se refleja con claridad en la figura proporcionada. En este escenario, se despliega una capa de relación mediante la anotación @ManyToOne, con conexiones que abarcan las entidades de "Ventanilla", "Servicio" y "Solicitante". Este enfoque busca permitir que estas instancias sean capaces de gestionar las citas, asegurando una coordinación eficaz. Esta representación también se encuentra ilustrada en la Figura 3.34.

```

18 @Entity
19 @Table(name = "citas")
20 @AllArgsConstructor
21 @NoArgsConstructor
22 @Setter
23 @Getter
24 @JsonIdentityInfo(
25     generator = ObjectIdGenerators.PropertyGenerator.class,
26     property = "id"
27 )
28 public class Cita {
29
30     @Id
31     @GeneratedValue(strategy = GenerationType.IDENTITY)
32     private Long id;
33     @Column(name = "fechaCita", nullable = false)
34     private LocalDateTime fechaCita;
35     @Column(name = "horaCita", nullable = false)
36     private LocalTime horaCita;
37     @Column(name = "montoPago", nullable = false)
38     private double montoPago;
39     @Column(nullable = false, columnDefinition = "tinyint default 1")
40     private Boolean status;
41
42     @ManyToOne
43     private Ventanilla ventanilla;
44
45     @ManyToOne
46     private Servicio servicio;
47
48     @ManyToOne
49     @JoinColumn(name = "solicitante_id")
50     private Solicitante solicitante;
51
52 }

```

Figura 3.34 Entidad Cita

En el contexto de la entidad "Horario", el proceso de diseño sigue una dinámica similar al considerar sus atributos característicos, que comprenden elementos como el día de la semana, la hora de inicio y finalización del trabajo, la cantidad de repeticiones y el estado del horario. Esta configuración se visualiza de manera clara en la figura proporcionada y nuevamente se implementa la anotación `@ManyToOne` para establecer una relación con la entidad "Ventanilla". Este enlace se desarrolla con el propósito de permitir una gestión eficiente por parte de la ventanilla. Esto, a su vez, contribuye a optimizar la administración y el control, como se encuentra representada en la Figura 3.35 con los nombres de los campos en la entidad.

```

16 @Entity
17 @Table(name = "horarios")
18 @AllArgsConstructor
19 @NoArgsConstructor
20 @Setter
21 @Getter
22 @JsonIdentityInfo(
23     generator = ObjectIdGenerators.PropertyGenerator.class,
24     property = "id"
25 )
26 public class Horario {
27
28     @Id
29     @GeneratedValue(strategy = GenerationType.IDENTITY)
30     private Long id;
31     @Column(name = "diaSemana", nullable = false)
32     private String diaSemana;
33     @Column(name = "horarioInicio", nullable = false)
34     private LocalTime horarioInicio;
35     @Column(name = "horarioFin", nullable = false)
36     private LocalTime horarioFin;
37     @Column(name = "cantidadRepeticiones", nullable = false)
38     private int cantidadRepeticiones;
39
40     @Column(nullable = false, columnDefinition = "tinyint default 1")
41     private Boolean status;
42
43     @ManyToOne
44     private Ventanilla ventanilla;
45
46 }

```

Figura 3.35 Entidad Horario

Correo de Confirmación. El proceso de codificación es llevado a cabo, tal como se presenta de manera visual en la Figura 3.36. Iniciamos este procedimiento realizando una verificación rigurosa para determinar si la dirección de correo electrónico proporcionada por el solicitante ya se encuentra presente en nuestra base de datos. En caso afirmativo, respondemos de manera inmediata con un mensaje personalizado, indicando que el correo electrónico en cuestión ya ha sido registrado previamente. Como medida de precaución y para mantener la integridad de la base de datos, no continuamos con el proceso de registro. En contraste, si no encontramos una coincidencia en la base de datos, avanzamos con determinación en el proceso de registro, asegurando una respuesta efectiva y consistente a las solicitudes de los usuarios.

```

89 //Insertar un Ventanilla
90 @Transactional(rollbackFor = {SQLException.class})
91 public CustomResponse<Ventanilla> insert(Ventanilla ventanilla) throws MessagingException{
92     if (this.ventanillaRepository.existsByCorreoVent(ventanilla.getCorreoVent())) {
93         return new CustomResponse<>{
94             null,
95             true,
96             400,
97             "¡El correo de esta Ventanilla ya existe!"
98         };
99     }
100
101     //envio de correos electronicos
102     MimeMessage mimeMessage = javaMailSender.createMimeMessage();
103     MimeMessageHelper messageHelper = new MimeMessageHelper(mimeMessage, true, "UTF-8");
104     messageHelper.setTo(ventanilla.getCorreoVent());
105     messageHelper.setFrom(mail);
106     messageHelper.setSubject("Mensaje de Confirmación de Creación de Cuenta");
107
108     // Configurar el contenido del mensaje con diseño personalizado
109     String htmlContent = "<html>" +
110         "<head>" +
111         "<style>" +
112         ".card {" +
113         "width: 500px;" +
114         "height: 300px;" +
115         "margin: 0 auto;" +
116         "background-color: #f8fbfe;" +
117         "border-radius: 8px;" +
118         "border: 1px solid #ccc;" +
119         "}" +
120         ".tools {" +
121         "display: flex;" +
122         "align-items: center;" +
123         "padding: 9px;" +
124         "}" +
125         ".circle {" +
126         "padding: 0 4px;" +

```

Figura 3.36 Ejemplo del Servicio de Correo en Ventanilla

Posteriormente, procedemos a generar un correo electrónico de confirmación, diseñado meticulosamente en un formato HTML personalizado. En este mensaje, se incluirá un cálido saludo de bienvenida, acompañado de los detalles específicos de cualquier rol en específico. Entre estos detalles, se mencionarán el nombre, el apellido paterno, la dirección de correo electrónico y la contraseña asignada. Es relevante mencionar que el logotipo de nuestro sistema se incorpora como una imagen en línea directamente en el contenido del correo electrónico, para brindar una identificación visual clara.

Una vez construido el contenido del correo electrónico de verificación, se inicia el proceso de envío. Utilizamos el servicio de envío de correos que ha sido cuidadosamente configurado en nuestro proyecto. La Figura 3.37 proporciona una

vista parcial de este proceso, centrando la atención en los aspectos más cruciales del mensaje. Esta representación visual destaca cómo los datos se insertan con precisión en el cuerpo del correo, logrando una comunicación coherente y personalizada con cada usuario.

```
186     "<div class=\"card_content\">" +
187     "</div>" +
188     "<h1 style=\"color: #ff605c;\">Hola! " + ventanilla.getNombre() + " " + ventanilla.getApePaterno() + "</h1>" +
189     "<h4 style=\"color: #264899;\">Bienvenido al Sistema CITAT</h4>" +
190     "<h4 style=\"color: #588EC4;\">Usuario Ventanilla</h4>" +
191     "<h4 style=\"color: #BAB98C;\">" +
192     "Este es tu correo para ingresar sesión " + ventanilla.getCorreoVent() + " con tu contraseña " + ventanilla.getPassword() +
193     "</h4>" +
194
195     "</div>" +
196     "</div>" +
197     "</div>" +
198     "</body>" +
199     "</html>";
200     messageHelper.setText(htmlContent, html: true);
201
202     // Adjuntar la imagen como recurso en línea
203     FileSystemResource imageResource = new FileSystemResource(new File("src/main/resources/static/images/citat.jpeg"));
204     messageHelper.addAttachment(attachmentFilename: "citat.jpeg", imageResource);
205
206     // Enviar el correo electrónico
207     this.javaMailSender.send(mimeMessage);
208
209     return new CustomResponse<>() {
210         |         this.ventanillaRepository.saveAndFlush(ventanilla),
211         |         error: false,
212         |         StatusCode: 200,
213         |         message: "Ventanilla registrado correctamente"
214     };
215 }
```

Figura 3.37 Cuerpo de Confirmación del Correo

3.3.4 Pruebas

Módulo Administrador. Durante el proceso de experimentación llevado a cabo, se logró establecer a través de la vía denominada "/administrador" empleada en la herramienta conocida como Postman. Estas pruebas consistieron en la simulación de diversos escenarios, entre los que se incluye la inserción de un nuevo registro utilizando la solicitud POST. Además, se evaluó la capacidad de recuperación de información correspondiente a todos los usuarios que hubieran sido previamente registrados en el sistema. Asimismo, se puso a prueba la funcionalidad de marcar el estado de este mismo registro y llevar a cabo modificaciones en un campo que ya se encontrara presente en los registros existentes.

En la Figura 3.38 se muestra la solicitud POST realizada para ingresar un registro.

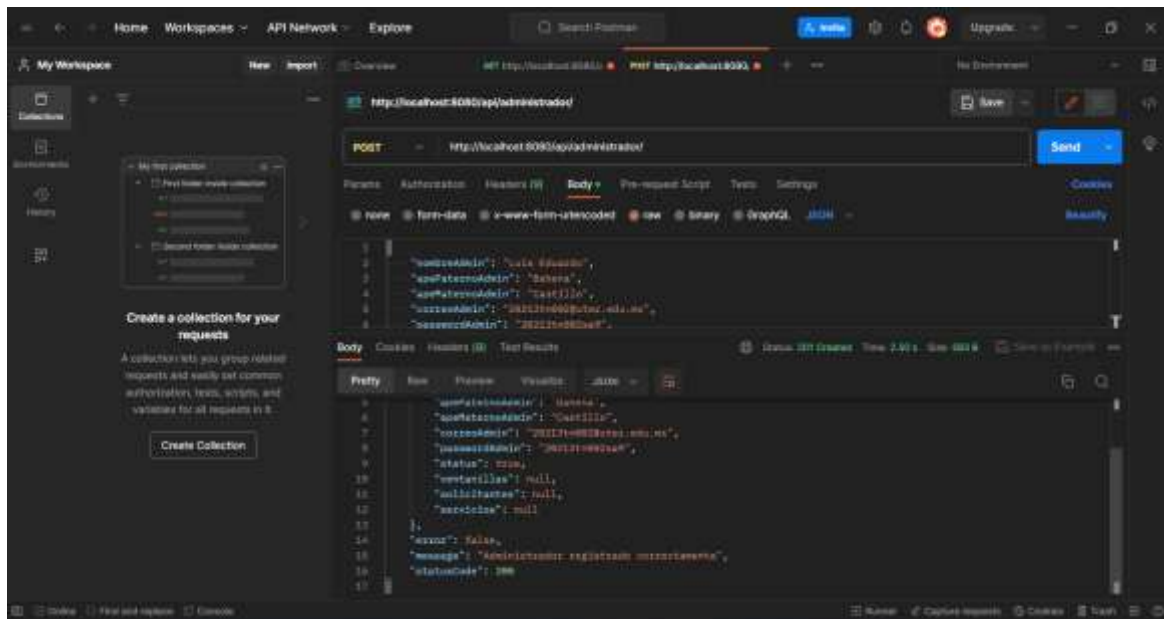


Figura 3.38 Solicitud POST Administrador

En la figura se muestra la solicitud GET realizada para retornar a los usuarios existentes.

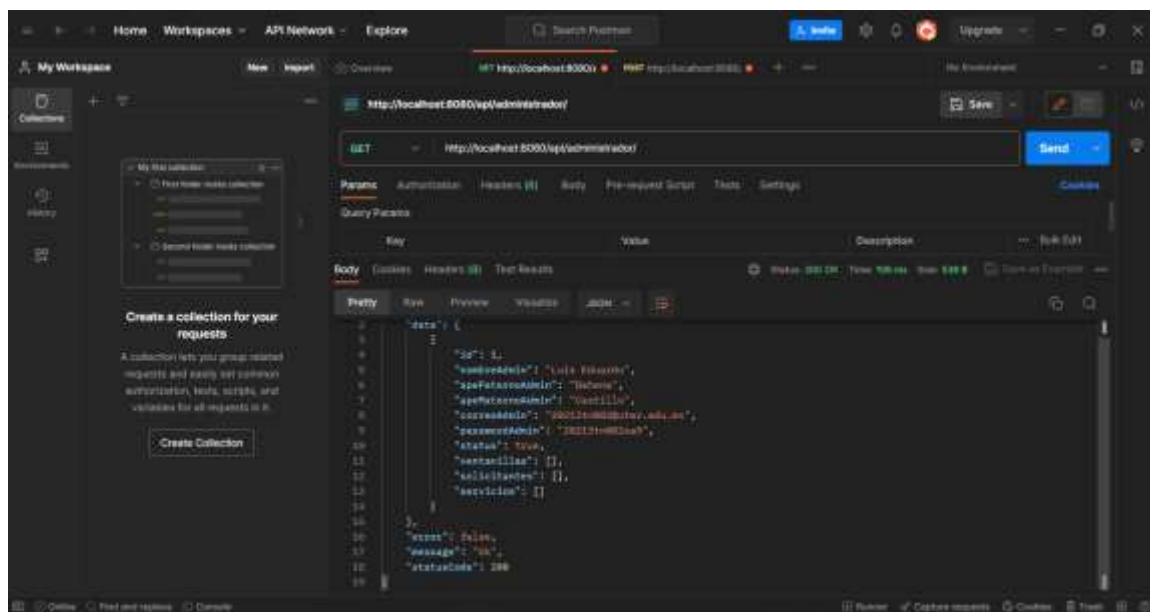


Figura 3.39 Solicitud GET Administrador

En la Figura 3.40 se muestra la solicitud PUT realizada para modificar un campo existente.

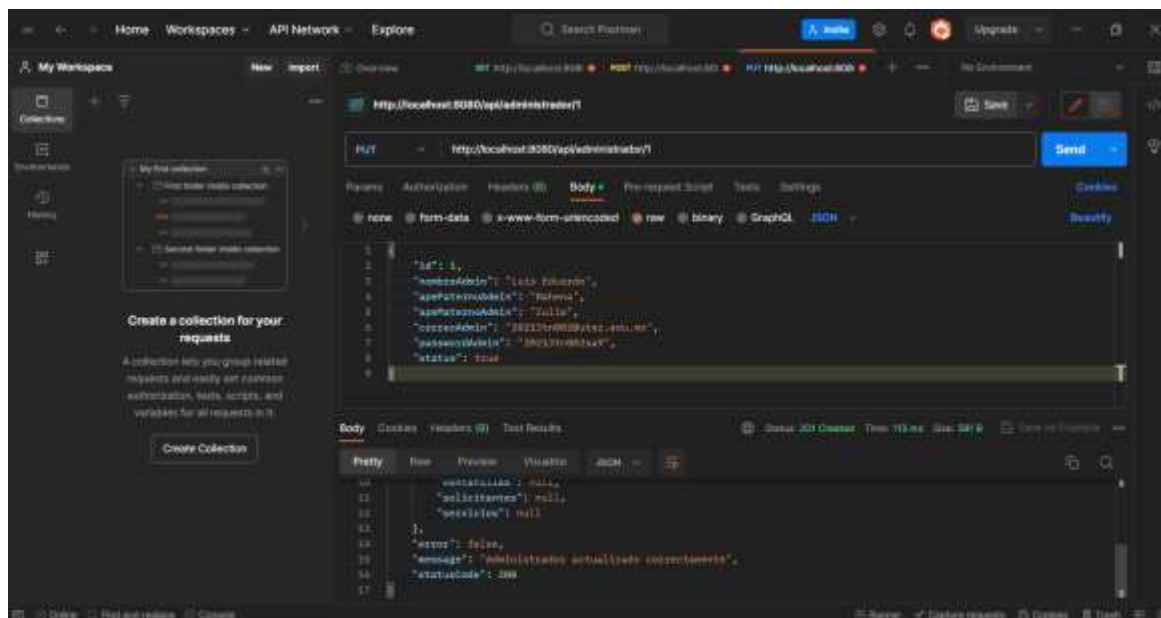


Figura 3.40 Solicitud PUT Administrador

En la Figura 3.41 se muestra la solicitud PATCH realizada para marcar el status del usuario.

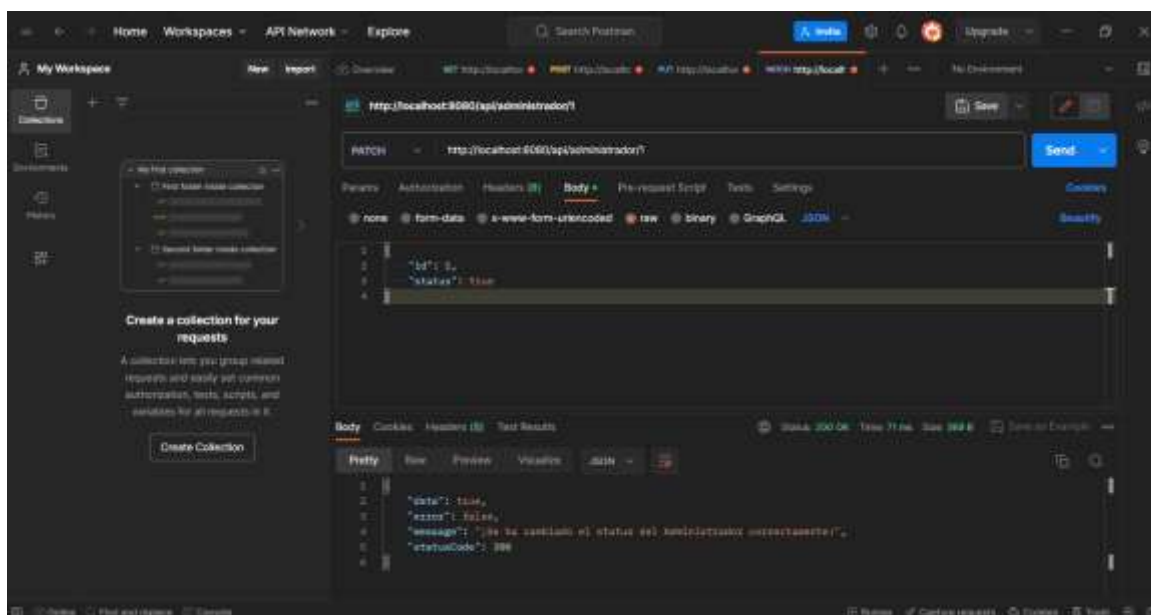


Figura 3.41 Solicitud PATCH Administrador

Módulo Ventanilla. A lo largo del proceso de experimentación que se llevó a cabo, se logró establecer mediante el uso de la ruta designada como `"/ventanilla"` en la herramienta reconocida bajo el nombre de Postman. Estas series de pruebas se desarrollaron al simular una variedad de situaciones, abarcando desde la inclusión de un nuevo registro mediante la solicitud POST hasta la evaluación de la capacidad para recuperar datos relacionados con todos los usuarios que hubieran sido registrados con anterioridad en el sistema. Además, se sometió a examen la funcionalidad inherente a la acción de asignar un estado particular a este mismo registro con método PUT, al tiempo que se pusieron a prueba las facultades de realizar modificaciones en un campo ya existente dentro del conjunto de registros preexistentes con el método PATCH.

En la Figura 3.42 se muestra la solicitud POST realizada para ingresar un registro.

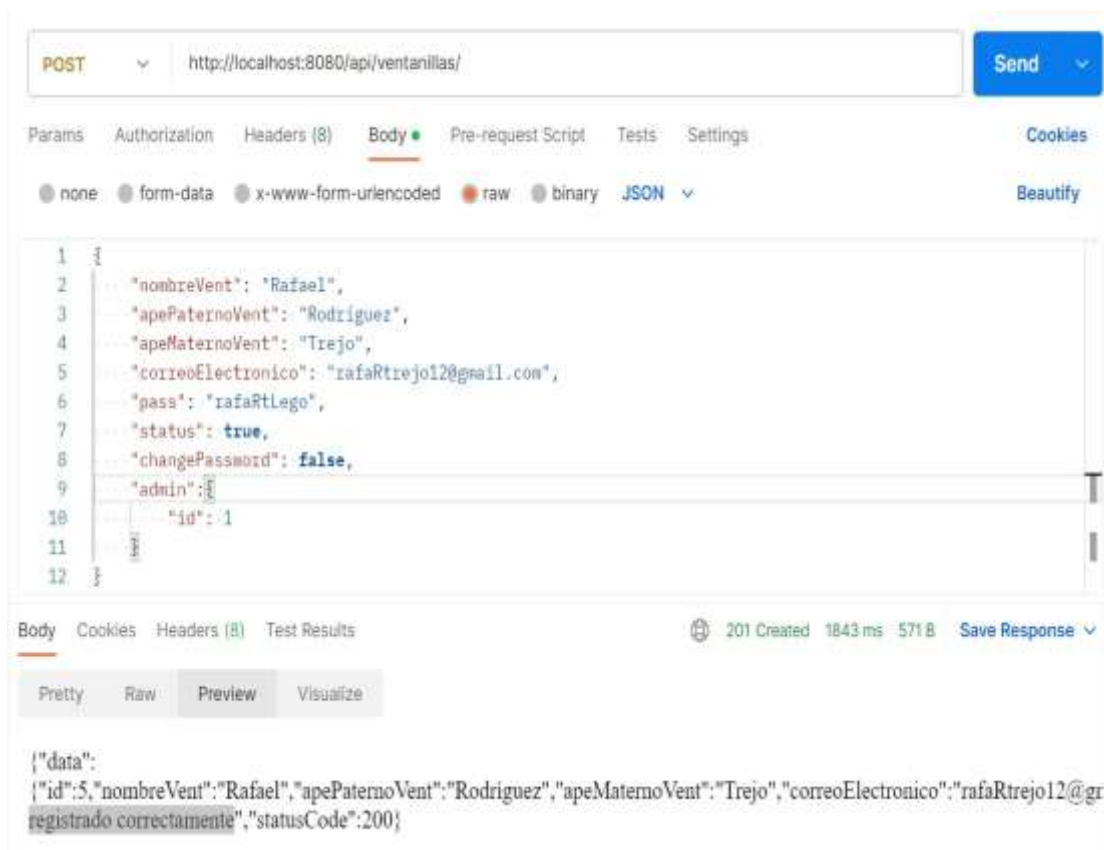


Figura 3.42 Solicitud POST Ventanilla

En la Figura 3.43 se muestra la solicitud PUT realizada para modificar un campo existente.

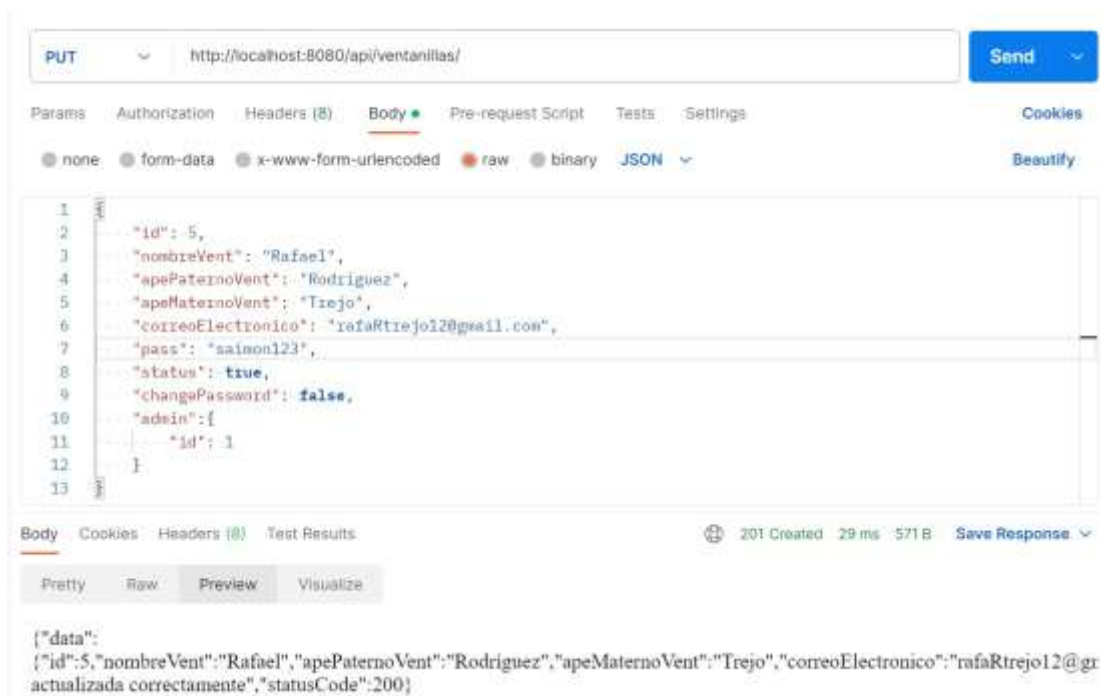


Figura 3.43 Solicitud PUT Ventanilla

En la Figura 3.44 se muestra la solicitud PATCH realizada para marcar el status del usuario.



Figura 3.44 Solicitud PATCH Ventanilla

Módulo Servicio, Citas y Horarios. En el transcurso del proceso extensivo de experimentación que fue aplicado con precisión, se logró establecer por medio de la utilización de rutas específicamente designadas, tales como "/citas", "/horario" y "/servicio", en la altamente reconocida herramienta denominada Postman. Este conjunto pormenorizado de pruebas fue ejecutado mediante la simulación de una amplia variedad de situaciones, que abarcaron desde la inserción de un registro completamente inédito a través de la emisión de solicitudes del tipo POST, hasta la evaluación exhaustiva de la capacidad inherente para recuperar información relacionada con la totalidad de usuarios que hubieran sido registrados previamente de manera adecuada en el sistema en cuestión. Además, en un ejercicio integral de rigurosa verificación, se procedió a poner a prueba con un alto grado de escrutinio las capacidades disponibles para realizar modificaciones en un campo que ya se encontraba previamente establecido dentro del conjunto global compuesto por registros ya existentes, además que se realizan algunos servicios de cada entidad.

En la Figura 3.45 se muestra la solicitud POST realizada para ingresar un registro de la entidad Servicio.

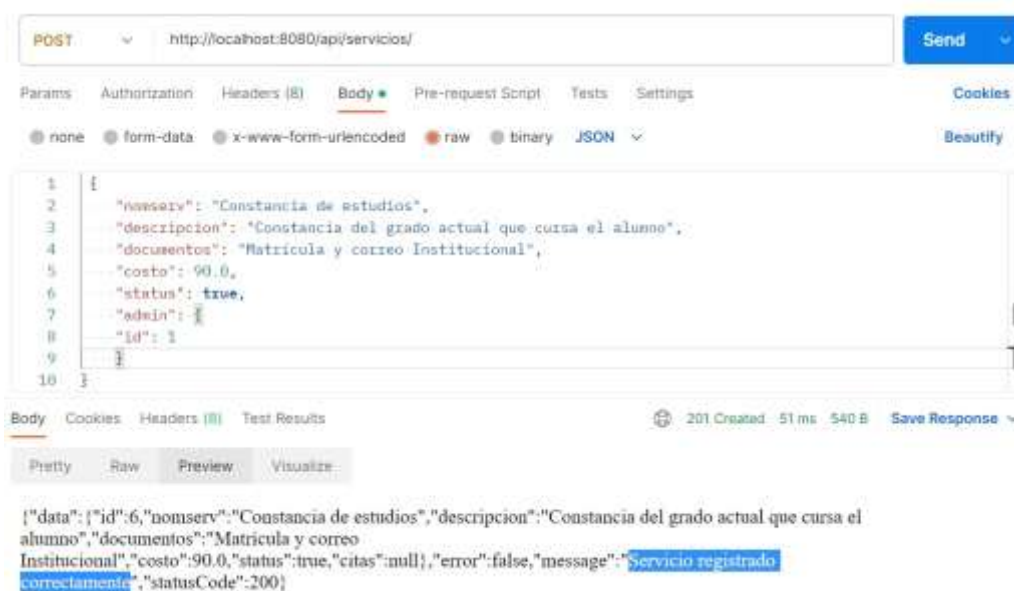


Figura 3.45 Solicitud POST Servicio

En la Figura 3.46 se muestra la solicitud GET realizada para retornar los registros del Servicio.

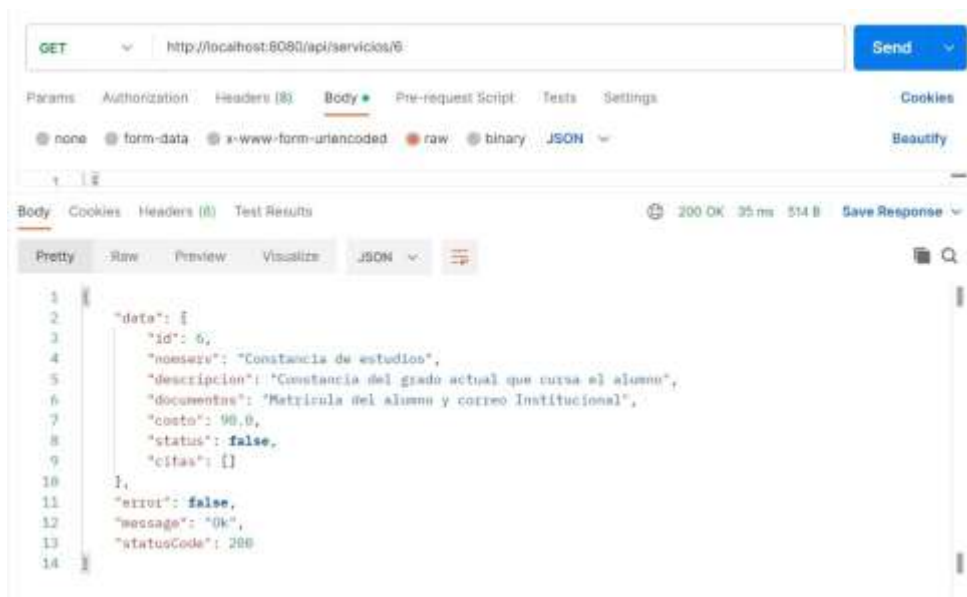


Figura 3.46 Solicitud GET Servicio

En la Figura 3.47 se muestra la solicitud GET realizada para retornar los registros de las Citas.

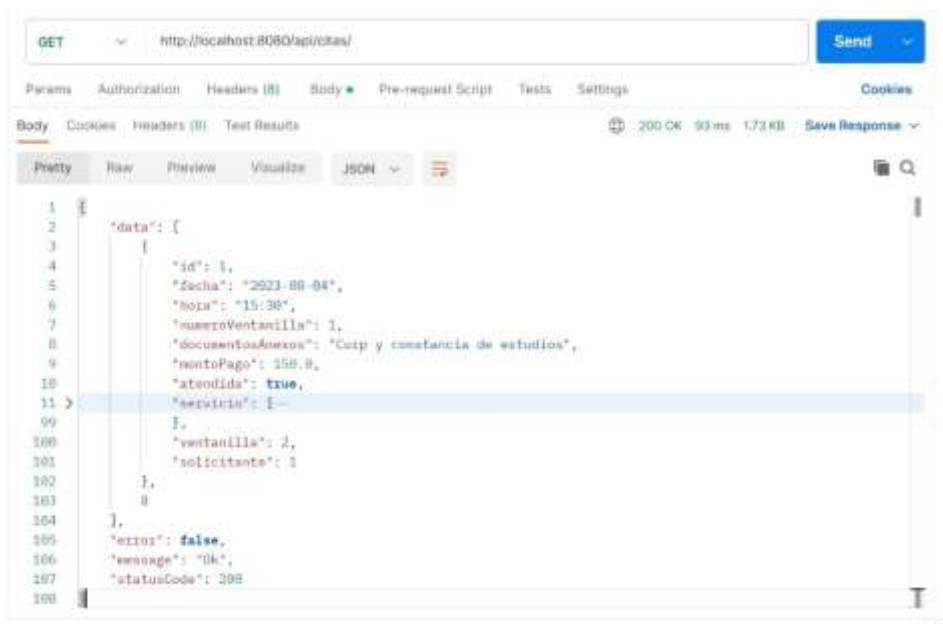


Figura 3.47 Solicitud GET Citas

En la Figura 3.48 se muestra la solicitud PATCH realizada para marcar una cita como atendida de la entidad Citas.

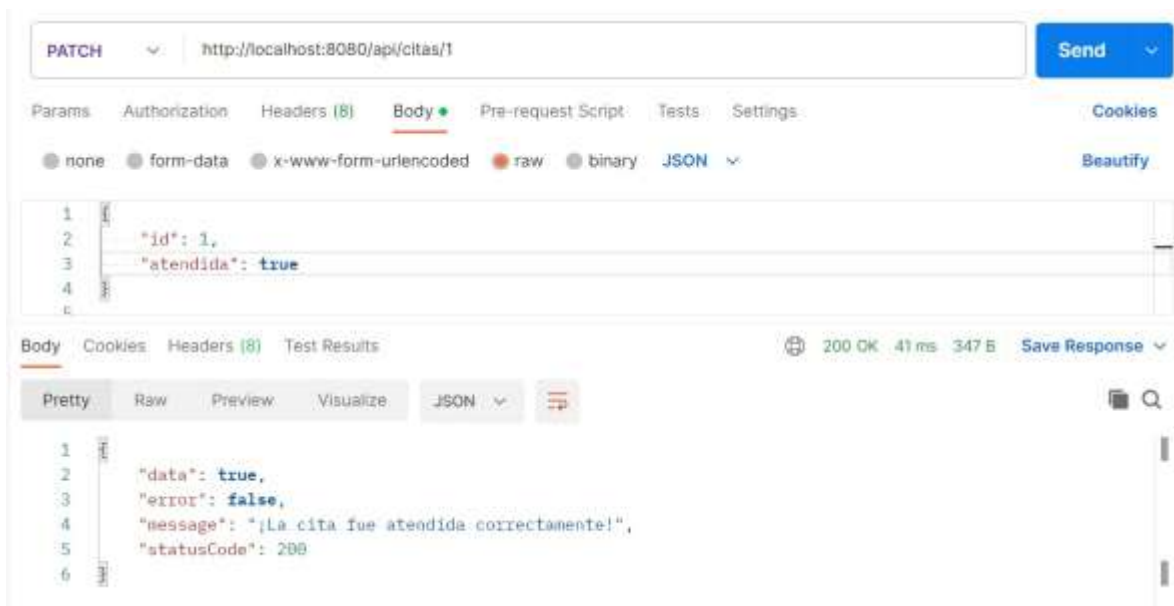


Figura 3.48 Solicitud PATCH Citas

En la figura se muestra la solicitud POST realizada para ingresar un registro de la entidad Horarios.



Figura 3.49 Solicitud POST Horarios

Correo de Confirmación. Cuando se registra un usuario de cualquier tipo, esta manda un correo para confirmar que se ha realizado el registro correctamente y muestra tanto el usuario como la contraseña en la que va a ingresar, como se muestra en la Figura 3.50.



Figura 3.50 Confirmación del Correo

Modal de Registro. Cuando se registra un usuario de cualquier tipo en la parte del Front, esta manda un modal con un mensaje de confirmación de que se ha realizado el registro correctamente, como se muestra en la Figura 3.51 cuando se registra un usuario de Administrador.

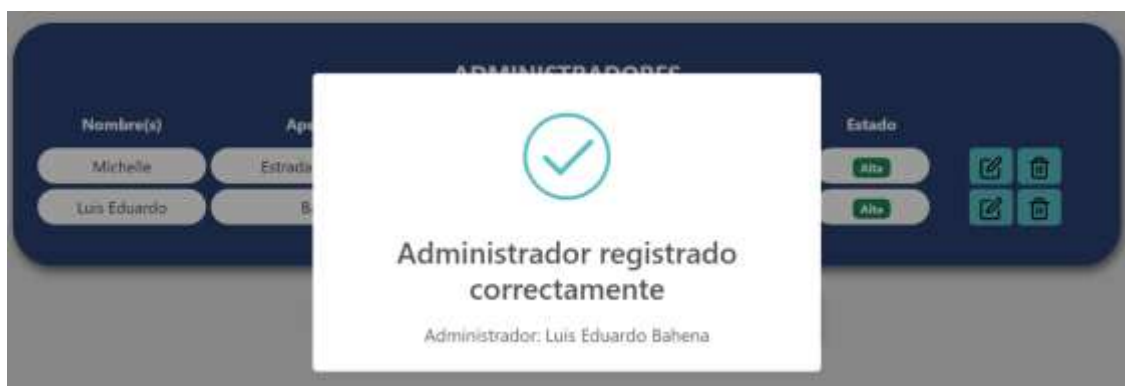


Figura 3.51 Modal de Confirmación de Registro del Usuario Administrador

3.4 Control

Fueron realizadas reuniones de carácter semanal, cada una con una duración de una hora con el desarrollo de la Metodología Scrum, en las cuales participaban tanto el equipo de trabajo como el asesor empresarial que ocupaba una posición destacada en el área de sistemas. Estas reuniones proporcionaron una valiosa oportunidad para evaluar con regularidad el avance del proyecto y recibir comentarios inmediatos. El papel del asesor se reveló esencial en este proceso, ya que su experiencia permitió identificar áreas susceptibles de mejora, resaltar los aspectos más sólidos y señalar posibles ajustes que podrían ser requeridos. Este enfoque facilitó la adaptación de estrategias y tácticas con base en los comentarios obtenidos, contribuyendo a perfeccionar tanto la efectividad como la eficiencia del proyecto.

Como resultado de todo lo expuesto, se logró un considerable avance en la ejecución constante de este proceso. Para ilustrar este progreso, se presenta en la Figura 3.52 donde se observa el cronograma que sigue un esquema visual de seguimiento del proyecto.

N	EDT	Fase	Tarea	Días Plan	Fecha Inicio Plan	Fecha Final Plan	% Avance Plan	% Avance Real	Status
1	1.0	Análisis	Documento Formal de Requerimientos (DFR)	10	1/5/2023	12/5/2023	100%	100%	Terminado
2	1.1	Análisis	Diagrama EDT	11	12/5/2023	26/5/2023	100%	100%	Terminado
3	1.2	Análisis	Project Charter	11	12/5/2023	26/5/2023	100%	100%	Terminado
4	2.1	Análisis	Diagrama de Casos de Uso	11	12/5/2023	26/5/2023	100%	100%	Terminado
5	2.2	Diseño	Diagrama de Base de Datos	26	29/5/2023	5/7/2023	100%	100%	Terminado
6	2.3.1	Diseño	Pantalla de Administradores	26	29/5/2023	5/7/2023	100%	100%	Terminado
7	2.3.2	Diseño	Pantalla de Citas	26	29/5/2023	5/7/2023	100%	100%	Terminado
8	2.3.3	Diseño	Pantalla de Ventanillas	26	29/5/2023	5/7/2023	100%	100%	Terminado
9	2.3.4	Diseño	Pantalla de Solicitantes	26	29/5/2023	5/7/2023	100%	100%	Terminado
10	2.4.1	Diseño	Informes de Citas	26	29/5/2023	5/7/2023	100%	100%	Terminado
11	2.4.2	Diseño	Informes del Historial de Ventanillas	26	29/5/2023	5/7/2023	100%	100%	Terminado
12	3.1.1	Programación	Módulo de Administradores	32	1/6/2023	14/7/2023	100%	90%	Retrasado
13	3.1.2	Programación	Módulo de Citas	32	1/6/2023	14/7/2023	100%	90%	Retrasado
14	3.1.3	Programación	Módulo de Ventanillas	32	1/6/2023	14/7/2023	100%	90%	Retrasado
15	3.1.4	Programación	Módulo de Solicitantes	32	1/6/2023	14/7/2023	100%	90%	Retrasado
16	3.2	Programación	Módulo Informe general de Citas	27	12/6/2023	18/7/2023	100%	85%	Retrasado
17	3.3	Programación	Módulo Manual de Usuario	27	12/6/2023	18/7/2023	100%	85%	Retrasado
18	3.3.1	Pruebas	Módulo de Administradores	20	17/7/2023	11/8/2023	100%	80%	Retrasado
19	3.3.2	Pruebas	Módulo de Citas	20	17/7/2023	11/8/2023	100%	80%	Retrasado
20	3.3.3	Pruebas	Módulo de Ventanillas	20	17/7/2023	11/8/2023	100%	80%	Retrasado
21	4.1.1	Pruebas	Módulo de Solicitantes	20	17/7/2023	11/8/2023	100%	80%	Retrasado
22	4.1.2	Pruebas	Módulo Informe general de Citas	20	17/7/2023	11/8/2023	100%	60%	Retrasado
23	4.1.3	Pruebas	Módulo Manual de Usuario	17	20/7/2023	11/8/2023	100%	60%	Retrasado
24	5.3	Liberación	Documento Formal de Requerimientos (DFR)	12	1/8/2023	16/8/2023	92%	92%	Proceso
25	5.4	Liberación	Script y Diagrama de la Base de Datos	12	1/8/2023	16/8/2023	92%	92%	Proceso
26	5.5	Liberación	Aplicación Móvil	12	1/8/2023	16/8/2023	92%	44%	Retrasado
27	5.5	Liberación	Aplicación Web	12	1/8/2023	16/8/2023	92%	80%	Retrasado

Figura 3.52 Cronograma de Avance

3.5 Cierre

La culminación oficial del proyecto "Sistema de Gestión de Citas para Trámites" se marcó con la presentación exhaustiva de la mayoría de las funcionalidades de la aplicación web y de aplicación móvil. A través de las pruebas unitarias de los servicios web en respuesta a las solicitudes en la herramienta Postman sobre los usuarios, se demostraron los procesos operativos con éxito. La efectividad de cada modelo preexistente quedó validada y verificada, respaldando su eficacia y rendimiento.

En una reunión final de entrega, se consolidó la confirmación de la aprobación tanto de la aplicación web como de los elementos funcionales de la versión móvil. Esta aprobación sentó las bases para la liberación y aprobación del sistema por parte del cliente. Una vez que el cliente otorgó su consentimiento a los módulos y funcionalidades desarrollados, se avanzó hacia la formalización del proceso de cierre a través de la firma del ANEXO C - ACTA DE CIERRE. Este documento abarcó los detalles completos de la entrega general del proyecto, abordando la fecha de inicio y finalización del sistema por parte del cliente, así como los entregables convenidos, entre estos documentos mencionados son: el Project Charter, Diagrama EDT, Informe de Avances, Acta de Cierre y Carta de Liberación.

En última instancia, se llevó a cabo la tramitación del ANEXO D - CARTA DE LIBERACIÓN, lo cual oficializó la entrega de las aplicaciones desarrolladas. Esta carta cumplió el papel de confirmar de manera concreta la conclusión exitosa del período de colaboración entre la Universidad Tecnológica Emiliano Zapata (UTEZ) y la Institución. En su mayoría, se lograron cumplir los objetivos generales del proyecto y los entregables fueron entregados dentro de los plazos estipulados.

CAPÍTULO 4. CONCLUSIONES

4.1 Cumplimiento de objetivos

Una serie de objetivos se lograron dentro del marco de los plazos establecidos, bajo la atenta supervisión tanto del asesor empresarial como del asesor académico. Se llevaron a cabo las correcciones necesarias durante el desarrollo del proyecto y se avanzó de acuerdo con el cronograma previsto. Como culminación, se procedió a entregar lo siguiente:

- El proyecto de la aplicación web como entregable final.
- La inclusión de los módulos de gestión del administrador, de ventanilla y de solicitante para poder gestionar y agendar las citas.
- Las pruebas de conectividad, usabilidad y funcionalidad que se realizaron en la aplicación web.

Es importante destacar que se logró implementar la funcionalidad de envío de correos electrónicos, un aspecto que planteó desafíos debido a la interacción entre el BackEnd y la plataforma de Gmail. Este obstáculo se superó con éxito al crear una clave de API que facilitó la comunicación entre estas dos partes.

Adicionalmente, es crucial enfatizar que se llevaron a cabo pruebas específicas para verificar la funcionalidad integral del sistema. Estas pruebas rigurosas aseguraron que las diferentes funcionalidades sean verdaderamente operativas y que cualquier test realizado por los miembros del equipo sea validado.

En conclusión, se lograron cumplir varios objetivos en los plazos previstos, con una supervisión activa por parte de los asesores. Se logró la entrega de la aplicación web, la implementación de módulos clave y pruebas exhaustivas para asegurar la funcionalidad. La superación de obstáculos, como el envío de correos electrónicos,

demostró la capacidad del equipo para abordar desafíos técnicos. Las pruebas meticulosas también validaron la efectividad del sistema en general y la participación activa del equipo en su desarrollo.

4.2 Resultados

Para poder mostrar los resultados que se obtuvieron durante el desarrollo del proyecto, se organizó una reunión en Google Meet tanto el cliente como el personal de la empresa en la que se presentó todos los proyectos de los diversos equipos de trabajo el día 16 de Agosto del presente año, realizando exhaustivamente todas las pruebas requeridas tanto del Asesor Empresarial y se ejecutó con éxito la aplicación web y parte de la aplicación móvil, durante la reunión se explicó el funcionamiento del sistema y tuvimos la retroalimentación necesaria en la que se llegó a un acuerdo que se podrá resolverlo en un evento futuro, al finalizar la reunión se realizó la entrega de los códigos fuentes del repositorios en GitHub, y un archivo de extensión llamada citat.zip a la carpeta de drive para respaldar una copia de seguridad de lo hecho en este periodo de estadía y facilitara al equipo de trabajo tenerlo de manera más rápida, para su posterior despliegue a producción. Como resultado obtenido al realizar el Sistema de Gestión de Trámites y Citas (CITAT), se agilizó los procesos y lo servicios en las que el personal de la institución educativa logrará poder tener una buena administración en tiempo real de las citas agendadas por los solicitantes, así como la planeación de los horarios de trabajo por parte del personal de ventanilla, sin duda alguna será un gran éxito este proyecto para la institución.

4.3 Contribuciones

La empresa no estaba familiarizada con la herramienta GitHub, la cual permite crear repositorios en la nube y almacenar el código fuente. Sin embargo, esta experiencia sirvió como un punto de partida para considerar la adopción de esta nueva tecnología

en futuros proyectos. La empresa pudo constatar por sí misma que esta herramienta es altamente accesible y sencilla de utilizar.

Al embarcarnos en la realización de la aplicación web, se tomó la decisión de emplear las tecnologías React y Spring. Estas opciones resultaron altamente beneficiosas para diseñar tanto el FrontEnd, a través de la estructura de lenguajes HTML y CSS, como para la parte del BackEnd, donde Java se integró con el framework Spring. La elección de estas tecnologías no solo permitió un desarrollo fluido, sino que también contribuyó a la creación de una aplicación robusta y de alto rendimiento.

El propósito fundamental detrás de la creación del sistema era optimizar los procesos internos de la empresa. Un aporte crucial fue la estandarización de todo el proceso de desarrollo del modelo, lo cual se documentó minuciosamente. En este proceso, se optó por utilizar la metodología ágil SCRUM, que resultó ser muy bien recibida tanto por el equipo interno de la empresa como por el asesor universitario. Como resultado, esta metodología se incorporó con éxito en el desarrollo y el control de proyectos futuros a corto y mediano plazo. Esta adopción también marcó un hito en el plan de crecimiento empresarial.

En síntesis, la empresa aprendió sobre la utilidad de GitHub y consideró su aplicación en proyectos venideros. La elección de las tecnologías React y Spring impulsó un desarrollo eficiente y equilibrado en la aplicación web. El objetivo clave de mejorar los procesos internos se materializó mediante la estandarización y la adopción de la metodología SCRUM. Este enfoque resultó en un proceso de desarrollo más fluido y en una integración exitosa de las mejores prácticas en la gestión de proyectos futuros.

REFERENCIAS

- Calvo, D. (7 de Abril de 2018). *Metodología SCRUM (Metodología ágil)*. Recuperado el 5 de Julio de 2023, de Diego Calvo: <https://www.diegocalvo.es/metodologia-scrum-metodologia-agil/>
- Canive, T. (27 de Mayo de 2020). *Metodología XP o Programación Extrema*. Recuperado el 4 de Julio de 2023, de Gestor de proyectos online: <https://www.sinnaps.com/blog-gestion-proyectos/metodologia-xp>
- Coppola, M. (6 de Junio de 2022). *¿Qué es React y para qué sirve?* Recuperado el 7 de Julio de 2023, de Hubspot.es: <https://blog.hubspot.es/website/que-es-react>
- Coppola, M. (23 de Noviembre de 2022). *Qué es una API REST, para qué sirve y ejemplos*. Recuperado el 4 de Julio de 2023, de Hubspot.es: <https://blog.hubspot.es/website/que-es-api-rest>
- Deitel, P., & Deitel, H. (2016). *Java™ Cómo Programar* (Décima ed.). (A. V. Romero Elizondo, Trad.) Ciudad de México: Pearson.
- Fernández, Y. (30 de Octubre de 2019). *Qué es Github y qué es lo que le ofrece a los desarrolladores*. Recuperado el 26 de Junio de 2023, de Xataka.com: <https://www.xataka.com/basics/que-github-que-que-le-ofrece-a-desarrolladores>
- Flores, F. (22 de Julio de 2022). *Qué es Visual Studio Code y qué ventajas ofrece*. Recuperado el 13 de Julio de 2023, de OpenWebinars.net: <https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/>
- Gonzalez, M. V. (17 de Marzo de 2021). *¿Qué es Spring Boot?* Recuperado el 10 de Julio de 2023, de Codmind: <https://blog.codmind.com/que-es-spring-boot/>
- Huillcen Baca, H. A., Palomino Valdivia, F. d., & Soria Solís, I. (2022). *Introducción a las Bases de Datos con MySQL*. Arequipa, Perú: Herwin Alayn Huillcen Baca. Obtenido de <https://books.google.at/books?id=xq5wEAAQBAJ>

- Humanes, M. (23 de Julio de 2019). *Metodología Lean: ¿qué es y cómo aplicarla en tu empresa?* Recuperado el 19 de Julio de 2023, de Ekon:
<https://www.ekon.es/blog/metodologia-lean-empresa/>
- Hurtado, J. S. (3 de Diciembre de 2021). Cómo funciona la Metodología Scrum: Qué es y cómo utilizarla. *Thinking for Innovation*. Recuperado el 3 de Julio de 2023, de <https://www.iebschool.com/blog/metodologia-scrum-agile-scrum/>
- Ismaellopez.dev. (3 de Agosto de 2022). *El ciclo de vida de React con React Hooks*. Recuperado el 5 de Julio de 2023, de <https://ismaellopez.dev/blog/el-ciclo-de-vida-de-react-con-react-hooks>
- Mancuzo, G. (10 de Mayo de 2021). *Modelo Evolutivo de desarrollo en espiral*. Recuperado el 4 de Julio de 2023, de Blog - ComparaSoftware:
<https://blog.comparasoftware.com/modelo-de-desarrollo-en-espiral/>
- Martins, J. (10 de Octubre de 2022). *¿Qué es la metodología Kanban y cómo funciona?* Recuperado el 19 de Julio de 2023, de Asana:
<https://asana.com/es/resources/what-is-kanban>
- Nixon, R. (2020). *Aprender PHP, MySQL y JavaScript*. Marcombo. Obtenido de <https://books.google.at/books?id=AExOEAAAQBAJ>
- Pérez, E. G. (13 de Septiembre de 2021). *Qué es Postman: Cómo empezar a usar sus llamadas*. Recuperado el 18 de Julio de 2023, de ParaAndroid:
<https://paraandroid.net/que-es-postman-como-empezar-a-usar-sus-llamadas/>
- Puciarelli, L. (2020). *Node JS - Vol. 1: Instalación - Arquitectura - node y npm*. Buenos Aires, Argentina: RedUsers. Obtenido de <https://books.google.at/books?id=GOfqDwAAQBAJ>
- Pulido Romero, E., Escobar Domínguez, Ó., & Núñez Pérez, J. Á. (2019). *Base de datos*. Azcapotzalco, Mexico: Grupo Editorial Patria. Obtenido de <https://books.google.at/books?id=yZCzDwAAQBAJ>
- Robledano, A. (8 de Octubre de 2019). *Qué es MongoDB*. Recuperado el 11 de Julio de 2023, de OpenWebinars.net: <https://openwebinars.net/blog/que-es-mongodb/>



Roca, C. (9 de Junio de 2023). *¿Qué es Spring Boot y en qué se diferencia de Spring Framework?* Recuperado el 19 de Julio de 2023, de ThePower Business School: <https://www.thepowermba.com/es/blog/que-es-spring-boot>

Stsepanets, A. (17 de Enero de 2023). *Modelo de cascada (Waterfall): qué es y cuándo conviene usarlo.* Recuperado el 3 de Julio de 2023, de Gantt Chart GanttPRO Blog: <https://blog.ganttpro.com/es/metodologia-de-cascada/>

Suárez, I. (14 de Marzo de 2023). *Frameworks de desarrollo de software: tendencias para 2023.* Recuperado el 2 de Julio de 2023, de Apiumhub: <https://apiumhub.com/es/tech-blog-barcelona/frameworks-de-desarrollo-de-software-tendencias-para-2023/>

ANEXOS





ANEXO A. PROJECT CHARTER

 <div> <div>Project Charter</div> <div>  <div>División Académica de Tecnologías de la Información y Diseño</div> </div> </div>		
Proyecto	CITAT	Sistema de gestión integral de servicios para estudiantes de una Institución Educativa
Ciente	LETC	Luis Enrique Trujillo Campos
Objetivo General		
Administrar una Institución Educativa por medio de un Sistema web y móvil para la gestión de servicios para estudiantes.		
Justificación		
Mejorar el proceso de entrega de documentos para hacerlo más eficaz y rápido		
Hacer posible a los usuarios de la Institución Educativa llevar un control adecuado de las citas para los trámites necesarios		
Facilitar el seguimiento y control de la las citas para los servicios de los Solicitantes		
Entregables		
01	Documento Formal de Requerimientos	
02	Acta Constitutiva - Project Charter	
03	Script y diagrama de la base de datos	
04	Código fuente de aplicaciones web, móvil y servicios	
05	Empaquetados de aplicaciones web, móvil y servicios	
06	Estructura de desglose de trabajo (EDT)	
Requerimientos Funcionales		
Disponer de un usuario administrador el cual debe tener acceso total al sistema en el lado administrativo (WEB)		
Disponer de un usuario ventanilla el cual debe tener acceso solo a la gestión de citas (WEB)		
Acceder por correo y contraseña		
Gestionar los datos de perfil		
Usuario Administrador		
Gestionar usuarios (Solicitante, Ventanilla y Administrador)		
Gestionar servicios escolares de solicitantes		
Usuario Ventanilla		
Gestionar citas y horarios disponibles de los solicitantes		
Gestionar documentos anexos de las citas		
Usuario Solicitante		
Consultar calendario de citas disponibles		
Cargar documentación necesaria para ser atendida en ventanilla		
Registrar citas		
Alcance		
La aplicación solamente es para una Institución Educativa		
La aplicación permite visualizar el registro de citas del solicitante		
Los documentos adjuntos por los solicitantes deben de ser solamente pdf, csv y png		
La aplicación permite visualizar el historial de citas		
Usuario Administrador		
Acceso a todas las cuestiones administrativas en la aplicación		

Página 1 de 2

CITAT_ProjectCharter

ANEXO A. PROJECT CHARTER Continuación...

		<h1>Project Charter</h1>			
Proyecto	CITAT	Sistema de gestión integral de servicios para estudiantes de una Institución Educativa			
Cliente	LETC	Luis Enrique Trujillo Campos			
Acceder a la aplicación mediante la página web Usuario Ventanilla Se le permite asignar, visualizar y eliminar citas al usuario solicitante Acceder a la aplicación mediante la página web Usuario Solicitante Visualizar las citas y servicios asignados por el usuario ventanilla y administrador No puede modificar el calendario					
Lider de Proyecto Jorge Angel Ruiz Juárez					
Fecha de Inicio		09 de mayo del 2023		Fecha de Fin	
				18 de agosto del 2023	
Supuestos La computadora y el smartphone donde sea ejecutado debe tener acceso a internet. Los usuarios solicitantes forman parte de la Institución Educativa El cliente proporcionó la libertad de elección del tipo de base de datos					
Lider de Proyecto		Sponsor		Cliente	
 Jorge Angel Ruiz Juárez Firma		I.T.I. Erick Mireles Merchant Firma		 Luis Enrique Trujillo Campos Firma	
Control de Versiones					
Versión	Fecha	Descripción			
1.0	09/05/2023	Versión Inicial			

Página 2 de 2

CITAT_ProjectCharter

ANEXO B. DOCUMENTO FORMAL DE REQUERIMIENTOS (DFR)

Datos del Proyecto

Nombre del proyecto:	Sistema de control de citas para trámites
Nombre del cliente:	AccionTi

Introducción

El presente documento describe los requerimientos para el desarrollo del proyecto "Sistema de control de citas para trámites" para una escuela. El sistema permitirá gestionar las citas para los servicios escolares, brindando a los usuarios la posibilidad de agendar citas, realizar pagos en línea y entregar la documentación necesaria de manera física. El objetivo principal es optimizar la gestión de citas y facilitar el proceso de trámites para los estudiantes.

Requerimientos Funcionales

No.	Descripción	Prioridad Alta / Media / Baja
1.1	Gestión de Usuarios Iniciar Sesión - Todos los roles El sistema requiere una funcionalidad que permita a todos los usuarios iniciar sesión en la aplicación web. Datos de entrada: <ul style="list-style-type: none">• Correo Válido*• Contraseña* Nota: <ul style="list-style-type: none">• El símbolo (*) marca los campos que son obligatorios. Reglas de negocio: <ul style="list-style-type: none">• La sesión tendrá un tiempo límite de una hora antes de que la sesión caduque.• Solo se tendrán 3 intentos para ingresar la contraseña de la sesión.	Alta

ANEXO B. DOCUMENTO FORMAL DE REQUERIMIENTOS (DFR) Continuación...

	Criterios de aceptación: <ul style="list-style-type: none"> Se mostrará una alerta con la siguiente leyenda: "Tu sesión ha expirado". Seguido de el cierre de la sesión. Se mostrará una alerta para anunciar que alguno de los dos campos es incorrecto 	
--	---	--

No.	Descripción	Prioridad Alta/ Media/ Baja
2.1	Gestión de Administrador Administrar el acceso a la aplicación mediante la gestión de usuarios en el sistema. El administrador requiere una funcionalidad que permita el acceso de usuarios previamente creados para poder entrar al sistema. Datos de entrada: <ul style="list-style-type: none"> Nombre completo* Matrícula* Carrera* Correo electrónico* Teléfono de contacto Nota: <ul style="list-style-type: none"> El símbolo (*) marca los campos que son obligatorios. Reglas de negocio: <ul style="list-style-type: none"> El administrador será responsable de visualizar, modificar y eliminar cuentas de usuario y definir los roles y permisos correspondientes. El formato del correo electrónico debe ser válido. Criterios de aceptación: <ul style="list-style-type: none"> El sistema de permisos y roles de usuario es eficaz y se asegura de que los usuarios solo tengan acceso a las funciones y datos que son necesarios para desempeñar sus tareas. 	Alta
2.2	Gestionar los distintos servicios que pueden realizarse en ventanilla, incluyendo sus costos y los documentos necesarios El administrador requiere una funcionalidad que permita el control de los servicios en ventanilla incluyendo los costos y documentos a solicitar. Datos de entrada: <ul style="list-style-type: none"> Nombre del Servicio* Costo del Servicio* Documentos requeridos* Descripción.* 	Alta

ANEXO B. DOCUMENTO FORMAL DE REQUERIMIENTOS (DFR) Continuación...

<p>Nota:</p> <ul style="list-style-type: none"> El símbolo (*) marca los campos que son obligatorios. <p>Reglas de negocio:</p> <ul style="list-style-type: none"> Puede gestionar los sistemas que se realicen en ventanilla con costos y los documentos necesarios. Se checará si el estudiante realizó el pago en línea del servicio antes de realizar el proceso de dicho servicio. <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> La información sobre los servicios disponibles es clara y precisa, y se mantiene actualizada con los últimos cambios en los requisitos de documentos y costos. Los proveedores de servicios están claramente identificados en el sistema, y se asegura que solo los proveedores aprobados realicen los servicios ofrecidos en ventanilla. 	
--	--

No.	Descripción	Prioridad Alta/ Media/ Baja
3.1	<p>Gestión de Ventanilla</p> <p>Atender las citas agendadas por los solicitantes de servicios.</p> <p>El sistema debe permitir a los usuarios con el rol de ventanilla atender las citas agendadas por los solicitantes de servicios.</p> <p>Datos de entrada:</p> <ul style="list-style-type: none"> Nombre del solicitante * Número de identificación * Servicio que solicita * Fecha y hora de la cita * Duración <p>Nota:</p> <ul style="list-style-type: none"> El símbolo (*) marca los campos que son obligatorios. <p>Reglas de negocio:</p> <ul style="list-style-type: none"> El sistema debe permitir al personal de ventanilla acceder a las citas agendadas. El personal de ventanilla debe tener la capacidad de marcar una cita como "atendida" una vez finalizada la atención. <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> El personal de la ventanilla debe poder visualizar la lista de citas agendadas y seleccionar una cita para atenderla. Una vez que la atención ha finalizado, el personal de la ventanilla debe poder marcar la cita como atendida. 	Alta
3.2	Registrar los horarios disponibles para atender a los solicitantes	Alta

ANEXO B. DOCUMENTO FORMAL DE REQUERIMIENTOS (DFR) Continuación...

	<p>El sistema debe permitir a los usuarios con el rol de ventanilla registrar sus horarios disponibles para atender a los solicitantes.</p> <p>Datos de entrada:</p> <ul style="list-style-type: none"> • Día de la semana • Horario inicio • Horario fin • Cantidad de repeticiones <p>Nota:</p> <ul style="list-style-type: none"> • El símbolo (*) marca los campos que son obligatorios. <p>Reglas de negocio:</p> <ul style="list-style-type: none"> • Se debe registrar el horario disponible para atención de solicitudes en la Ventanilla. <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> • El personal de la ventanilla debe poder registrar sus horarios disponibles, especificando las fechas y horas en las que están disponibles para atender citas. 	
3.3	<p>Visualizar la agenda de citas y documentos anexos cargados por los solicitantes.</p> <p>El sistema debe proporcionar una función para visualizar la agenda de citas y los documentos anexos cargados por los solicitantes</p> <p>Datos de salida:</p> <ul style="list-style-type: none"> • Fecha • Hora • Número ventanilla • Servicios solicitado • Documentos anexos • Monto de pago <p>Nota:</p> <ul style="list-style-type: none"> • El símbolo (*) marca los campos que son obligatorios. <p>Reglas de negocio:</p> <ul style="list-style-type: none"> • El sistema debe permitir la gestión de citas agendadas por orden de llegada y asignarlas a los miembros del personal de la ventanilla según su disponibilidad. • Los documentos anexos cargados por los solicitantes deben ser válidos y relevantes para el servicio solicitado. <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> • El personal de la ventanilla debe poder visualizar la agenda de citas, mostrando las citas agendadas en orden cronológico. • El personal de la ventanilla debe poder acceder a los documentos anexos cargados por los solicitantes para cada cita. 	Alta
3.4	<p>Marcar las citas como atendidas una vez finalizada la atención brindada.</p>	Media

ANEXO B. DOCUMENTO FORMAL DE REQUERIMIENTOS (DFR) Continuación...

	<p>El sistema debe permitir a los usuarios con el rol de ventanilla marcar las citas como atendidas una vez finalizada la atención brindada.</p> <p>Datos de salida:</p> <ul style="list-style-type: none"> Identificador o número de la cita* <p>Nota:</p> <ul style="list-style-type: none"> El símbolo (*) marca los campos que son obligatorios. <p>Reglas de negocio:</p> <ul style="list-style-type: none"> Solo se permitirá marcar una cita como atendida una vez finalizada la atención brindada. <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> El personal de la Ventanilla debe poder marcar una cita como atendida una vez finalizada la atención brindada. 	
No.	Descripción	Prioridad Alta/ Media/ Baja
4.1	<p>Gestión de Solicitante</p> <p>Registrar como rol de estudiantes para solicitar servicios</p> <p>El sistema requiere que el solicitante se registre como estudiante para solicitar servicios mediante validación por correo electrónico.</p> <p>Datos de entrada:</p> <ul style="list-style-type: none"> Nombre * Apellidos * Correo Electrónico * Carrera* Contraseña * Teléfono de contacto <p>Nota:</p> <ul style="list-style-type: none"> El símbolo (*) marca los campos que son obligatorios. <p>Reglas de negocio:</p> <ul style="list-style-type: none"> Se deberá utilizar un correo válido. <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> Para que el correo sea aceptable deberá validarse. 	Alta
4.2	<p>Consultar calendario (Citas disponibles)</p> <p>El sistema requiere que el solicitante consulte el calendario de citas disponibles para agendar una atención.</p> <p>Datos de entrada:</p>	Media






ANEXO B. DOCUMENTO FORMAL DE REQUERIMIENTOS (DFR) Continuación...

	<ul style="list-style-type: none"> • Nombre * • Apellidos * • Correo Electrónico * • Carrera* • Contraseña * • Teléfono de contacto <p>Nota:</p> <ul style="list-style-type: none"> • El símbolo (*) marca los campos que son obligatorios. <p>Reglas de negocio:</p> <ul style="list-style-type: none"> • Solo se puede consultar el calendario de citas disponibles a los solicitantes que estén registrados dentro del sistema. <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> • Si el solicitante no se encuentra dentro del sistema no podrá acceder. 	
4.3	<p><u>Agendar citas.</u></p> <p>El sistema requiere que el solicitante pueda agendar una cita para ser atendido con una duración de media hora por defecto</p> <p>Datos de entrada:</p> <ul style="list-style-type: none"> • Fecha • Hora • Número ventanilla • Servicios solicitado • Documentos anexos • Monto de pago* <p>Nota:</p> <ul style="list-style-type: none"> • El símbolo (*) marca los campos que son obligatorios. <p>Datos de salida:</p> <ul style="list-style-type: none"> • Datos de los estudiantes. <p>Reglas de negocio:</p> <ul style="list-style-type: none"> • Solo se puede agendar cita los solicitantes que estén registrados dentro del sistema. • Todo solicitante puede agendar citas en los días y horarios disponibles ya consultado en el calendario <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> • Si el solicitante no se encuentra dentro del sistema no podrá acceder. • Si no hay fecha disponible se verá deshabilitado por el contrario podrá solicitarse 	Alta
4.4	<p><u>Cargar la documentación necesaria para ser atendido en ventanilla.</u></p> <p>El sistema debe permitir a los solicitantes cargar la documentación necesaria para su atención en ventanilla. Se debe proporcionar una</p>	Alta

ANEXO B. DOCUMENTO FORMAL DE REQUERIMIENTOS (DFR) Continuación...

	<p>funcionalidad de carga de archivos que permita a los solicitantes adjuntar los documentos requeridos en un formato aceptado</p> <p>Datos de entrada:</p> <ul style="list-style-type: none"> • Documentos anexos <p>Reglas de negocio:</p> <ul style="list-style-type: none"> • Los documentos anexos cargados por los solicitantes deben cumplir con los requisitos establecidos para cada servicio. <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> • Los solicitantes pueden cargar los documentos requeridos y realizar el pago correspondiente, así se guardan y se visualizan correctamente. 	
4.5	<p>Pagar el monto requerido (en ventanilla)</p> <p>El responsable tendrá que pagar el monto requerido por ventanilla para ser atendidos.</p> <p>Datos de salida:</p> <ul style="list-style-type: none"> • Datos de los estudiantes. • Datos de la cita agendada <p>Reglas de negocio:</p> <ul style="list-style-type: none"> • Solo se puede pagar a una cita ya existente y confirmada. <p>Criterios de aceptación:</p> <ul style="list-style-type: none"> • Si la cita no se encuentra en el sistema, no podrá pagarse 	Alta

ANEXO C. ACTA DE CIERRE

		<h1>Acta de Cierre</h1>			
Proyecto	CITAT	Sistema de gestión integral de servicios para estudiantes de una Institución Educativa			
Cliente	LETC	Luis Enrique Trujillo Campos			
Declaración					
<p>Con fecha del 18/08/2023, por medio de la presente, se deja constancia de la entrega, recepción, y aceptación del proyecto denominado "CITAT", con fecha de inicio: 09/05/2023, y fecha final: 18/08/2023. Por lo que, a mi entera satisfacción doy por liberado al equipo de desarrollo.</p>					
Entregables					
<p>Aplicación Móvil y Web.</p>					
<p>Script de la Base de Datos</p>					
<p>Documento Formal de Requerimientos</p>					
Alcance del Producto					
N	Requerimientos Funcionales				
1	Gestionar módulos de usuarios, servicios, citas y horarios.				
2	Crear módulo ventanilla el cual se encarga de marcar las citas como "atendidas" una vez que el servicio se haya completado.				
3	Notificar mediante un mensaje de confirmación al correo electrónico al registrar un nuevo usuario.				
4	Consultar el historial de citas por parte del solicitante.				
5	Validar el inicio de sesión para entrar a la sesión correspondiente.				
6	Enviar notificaciones por correo electrónico a los solicitantes cuando se haya registrado con éxito una cita o cuando se hayan realizado cambios en la cita.				
7	Diseñar una interfaz de usuario amigable y fácil de usar para los diferentes roles.				
Comentarios					
<p>La aplicación se entregará en una reunión virtual y funcionará con cada uno de los módulos correspondientes.</p>					
Líder de Proyecto		Sponsor		Cliente	
					
Jorge Angel Ruiz Juárez		L.T.J. Erick Mireles Merchant		Luis Enrique Trujillo Campos	
Firma		Firma		Firma	

Página 1 de 1

Acta de Cierre

ANEXO D. CARTA DE LIBERACIÓN



ASUNTO: CARTA DE LIBERACIÓN

M.A. Vivianissel Ariza Batalla
Jefa del Departamento de Estadías y Seguimiento a Egresados
Universidad Tecnológica Emiliano Zapata
del Estado de Morelos
P R E S E N T E

Por medio de la presente, se notificó a Usted, que el estudiante LUIS EDUARDO BAHENA CASTILLO del programa educativo de TECNOLOGÍAS DE LA INFORMACIÓN ÁREA: DESARROLLO DE SOFTWARE MULTIPLATAFORMA, con número de matrícula 20213TN002 cumplió satisfactoriamente su Estadía, participando en actividades propias a su perfil académico, en el periodo Mayo-Agosto 2023, y habiéndolo desarrollado con esmero, dedicación, disponibilidad y puntualidad en las labores asignadas, adscrito (a) al Área de Desarrollo.

Sin otro particular por el momento, hago oportuna la ocasión para reiterarle las seguridades de mi atenta y distinguida consideración.

Cuernavaca Mor. a 18 de agosto de 2023.

ATENTAMENTE

LIC. DAVID ALEJANDRO SILVA MAGALLÓN

Monterrey Nuevo León
C.P. 64550
Tel. 52 1 81 1707 4489
www.accionti.com