Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

TESIS

Diseño de un sistema para el seguimiento en línea de la ruta de Lobobus

para obtener el título de:

Licenciatura en Ingeniería en Ciencias de la Computación

Presenta:

Pablo Cerón Cervantes

Asesor:

Dr. Pedro Bello López

Puebla, Pue. Septiembre 2023

Agradecimientos

Para la realización del presente trabajo de tesina, agradezco a la institución que fue participe en mi formación profesional y personal, la Facultad de Ciencias de la Computación de la Benemérita Universidad Autónoma de Puebla. Así como a todos los profesores que a lo largo de la carrera compartieron conmigo su experiencia y conocimientos.

De manera particular, agradezco a mis profesores del diplomado, por sus enseñanzas y profesionalismo. De igual forma a mi asesor, el Dr. Pedro Bello López, por su valioso apoyo para la realización de mi proyecto de tesina y mi proceso de titulación.

A quienes hicieron que hoy sea la persona que soy, quienes me formaron con valores, principios y espíritu, mis padres, Mayra Cervantes Bonilla y Pablo Cerón Rosas, por sus esfuerzos y lucha constante para darme lo mejor que podían en todos los aspectos y a pesar de que existieran épocas complicadas, el amor que siempre existió en nuestra familia es lo que al final prevalece y une nuestro hogar. A mi padre, la persona que más me entiende, pilar principal de mi carácter y determinación. Y de forma especial, agradezco y admiro la resiliencia, valentía, aliento y amor de mi madre, muchas gracias, mamá, mi apoyo y cariño siempre contigo.

A la persona que me hace inmensamente feliz, Yasmin Morales Zitlalpopoca (Yas), por su cariño, confianza e inspiración para dar lo mejor de mí. El camino apenas comienza para nosotros, estaré a tu lado comprometido para hacer de esto lo mejor, darnos fuerza y aliento para crecer juntos, cumplir nuestras metas y seguir cultivando este gran amor.

Y finalmente, dedico principalmente este proyecto a la persona que, junto a mi madre, más admiro en el mundo, mi querida abuelita, la Mtra. Estela Guadalupe Bonilla Solís y le agradezco el amor, ayuda, paciencia y enseñanzas que me ha brindado, este primer logro es para ella, Muchas gracias.

Tabla de Contenido

Capítulo I

Introducción

"Este Capítulo presenta un resumen con los objetivos y antecedentes del proyecto, así como una síntesis que enfatiza su potencial aportación para con los usuarios del sistema interno de transporte escolar -Lobobus-"

El presente proyecto consiste en establecer los requerimientos y construir el diseño de un sistema web que funcione como referencia informativa a usuarios del servicio de *Lobobus* (Sistema de transporte interno de la Benemérita Universidad Autónoma de Puebla en Ciudad Universitaria), esto, con el fin de generar una guía para el desarrollo de un sistema que pueda mejorar la experiencia del usuario de cara a darle certeza en los tiempos de espera de las unidades, así como de forma interna, proporcionar información valiosa sobre la eficiencia y estado del servicio de movilidad interna.

La propuesta de diseño del sistema estará pensada para hacer seguimiento en tiempo real de las unidades del sistema de transporte interno *Lobobus*, mostrándole al usuario del servicio, sus ubicaciones y el tiempo estimado de llegada a cada una de las paradas que realizan dentro del área de Ciudad Universitaria, así como sentar las bases para un control estadístico de las unidades y lo relacionado con estas.

Debido a la naturaleza del proyecto y con el fin de conseguir la mayor precisión posible en los datos, así como dinamismo en la lectura de estos, el diseño de este sistema deberá construirse pensando en que se haría uso de una base de datos en tiempo real, así como de una aplicación en *Android* para el control y seguimiento de las unidades.

Dada la descripción de la aplicación a desarrollar, el objetivo general del presente trabajo de tesina es:

Generar una propuesta de diseño de un sistema web que realice el seguimiento a las unidades del sistema de trasporte interno *Lobobus*, mostrando sus ubicaciones y estimando tiempos de llegada a las diferentes paradas que realizan.

Los objetivos específicos que contribuirán a cumplir con el general son:

- Realizar, mediante un análisis, la especificación de requerimientos para la propuesta del sistema de seguimientos de la ruta del *Lobobus*.
- Definir las estructuras de datos necesarias para la propuesta del sistema
- Con base a los requerimientos diseñar la propuesta del esquema de base de datos para almacenar la ruta y realizar el seguimiento de las unidades del sistema *Lobobus*, así como para la generación de estadísticas, definiendo así también los indicadores clave de rendimiento iniciales.
- Diseñar la interfaz gráfica de las aplicaciones del sistema haciendo uso de herramientas de maquetado.
- Definir los elementos tecnológicos del sistema para conseguir el rastreo en tiempo real de las unidades.

Capítulo II

Estado del Arte

"El presente Capítulo contiene el resultado de la investigación sobre la bibliografía de proyectos que guardan relación con el presente. El propósito del capítulo es el de identificar problemáticas técnicas, en el diseño o desarrollo de aplicaciones similares para proponer soluciones a estas, aplicables en el presente, así como, con base a la comparación, recolectar ideas, conceptos, tecnologías y metodologías igualmente aplicables al presente proyecto."

2.1 Aplicación web para el seguimiento de bicicletas robadas en Bogotá

Se trata de una aplicación web dirigida a ciclistas de la ciudad de Bogotá, con la que pueden localizar su bicicleta en caso de hurto. La aplicación cuenta con un registro y una sección gráfica de visualización de ubicaciones. El registro de la ubicación es consultado en un visor geográfico de funciona mediante un servidor conectado al dispositivo GPS instalado en la bicicleta [1].

2.1.1 Características técnicas

En cuanto a los aspectos técnicos del proyecto en revisión, denotando los concernientes a la aplicación web, se enlistan los siguientes:

- Para el diseño Front-end sólo se menciona el uso de HTML5.
- En el manejo de mapas y servicio de geolocalización se optó por el uso de la API de código abierto *Geoserver*.
- En relación con el rastreo de las bicicletas, se realiza mediante un módulo GPS estratégicamente instalado en el vehículo [1].

2.1.2 Metodologías de desarrollo

El equipo de desarrollo eligió el hacer uso de dos metodologías:

- SCRUM, para un desarrollo rápido y ágil [1].
- Acceptance Model, para la validación, ya que predice como los usuarios aceptan y utilizan una herramienta tecnológica [1].

2.1.3 Observaciones del desarrollo

El equipo de desarrollo continúa trabajando en la aplicación, sin embargo, en el artículo publicado, que presenta su reporte de avances, comparten algunas observaciones de su desarrollo con el propósito de guiar a otros investigadores que trabajen en temas o tecnologías similares [1], así pues, se enlistan a continuación las más relevantes:

- Conforme el desarrollo del proyecto fue avanzando, se pudo observar que la invocación de los servicios web a través de un dispositivo móvil era complejo, por ello, integraron el uso de JavaScript para simplificar este proceso [1].

- De cara a la construcción de los servicios web, se plantearon el utilizar SOAP o REST, sin embargo, optaron por REST debido a su sencillez al ser implementado [1].

2.2 Aplicación para localización de taxi a través de GPS en tiempo real

En este proyecto se aborda la necesidad de un sistema web que facilite la asignación de unidades de taxi con los clientes del servicio, esto, en ciudad Juárez [2].

El sistema consta de tres elementos:

- Una aplicación móvil para el cliente, con las que solicita el servicio.
- Una aplicación móvil para el taxista, para recibir asignaciones de servicio.
- Una aplicación web que controle la asignación de unidades con clientes, operada por la central de taxis.

2.2.1 Características técnicas

A continuación, se presentan las características técnicas más relevantes.

- Las aplicaciones móviles son desarrolladas para Android con Java mediante el entorno de desarrollo de Eclipse.
- La aplicación web está desarrollada con el uso de PHP, JavaScript nativo y HTML5.
- Base de datos mediante el sistema gestor MySQL.
- Uso de la API de Google Maps para la geolocalización de clientes y unidades.

2.2.2 Metodología de desarrollo

Para el desarrollo se hizo uso de la metodología ágil XP (*Extreme Programming*), misma que consta de cinco fases: exploración, planificación, proceso de desarrollo, puesta en producción y mantenimiento [2].

2.2.3 Observaciones del desarrollo

En el apartado de conclusiones los autores puntualizan algunas observaciones sobre el desarrollo realizado, entre las que se pueden destacar:

- Debido a que el enfoque de desarrollo está basado sobre un paradigma de programación imperativa, se pierde el dinamismo, es decir, la visualización en tiempo real de todas las acciones de los usuarios que afectan al sistema.
- La experiencia de los usuarios, tanto clientes como administradores, se puede ver beneficiada con una mejora visual de los elementos, instrucciones claras de uso y funcionamiento, y elementos dinámicos que mejoren la usabilidad del sistema.

2.3 Conclusiones

En las aplicaciones revisadas, así como en la mayoría de su ramo, más allá de los objetivos de cara a la localización u oferta de servicios, existen dos puntos técnicos principales a considerar:

- La plataforma y el método de localización de las unidades deben ser seleccionados pensando en que se asegure una comunicación estable para el envío de datos, así como que soporte una consulta constante de su ubicación.
- Se debe buscar que el servicio de geolocalización y consulta de mapas sea confiable, tenga alta disponibilidad y posibilidad de escalamiento.

El presente proyecto presentará una propuesta de diseño tomando en cuenta los puntos anteriores, mismos que podrían comprometer el correcto funcionamiento de la aplicación.

- Con base en la revisión realizada de proyectos que incluyen seguimiento de ubicaciones, se notó que en algunos casos optan por la construcción de hardware propio, haciendo uso de dispositivos *GPS*, comunicación a la red mediante chip telefónico y procesamiento sobre tarjetas de desarrollo como *Raspberry Pi*; mientras que en otros hacen uso de *Smartphones* para realizar el rastreo. La decisión de la plataforma a utilizar recae en las necesidades y alcance del proyecto, la primera opción responde a la necesidad de un

sistema de rastreo desatendido que opera de forma autónoma, sin embargo, requiere de un tiempo de desarrollo más largo, así como una inversión para la compra e instalación de los componentes. Por otra parte, el uso de *Smartphones* responde a una solución que ofrece un desarrollo más rápido, una plataforma confiable y una implementación más simple.

Para el presente diseño, se propone el uso de una aplicación móvil para *Smartphone*, que además de ofrecer las ventajas antes descritas, se integra fácilmente a los servicios web que se podrán usar en el sistema.

- Debido al alcance que tendrá el sistema, el cual se centrará únicamente en el monitoreo de las unidades de transporte dentro de Ciudad Universitaria y cuyo flujo de consultas sería contenido a razón del número de personas que acuden al complejo, lo ideal es hacer uso de los servicios de geolocalización y mapas ofrecidos por la *API de Google Maps*, ya que tiene una integración nativa con los sistemas operativos móviles, además de ser estable, confiable y ofrecer un plan gratuito con el volumen de datos y consultas suficientes para soportar el servicio.

Capítulo III

Marco Teórico

"En este capítulo se revisarán conceptos teóricos cuyo entendimiento resulta esencial para la construcción del diseño del sistema, además de que algunos de ellos nos ayudarán, en conjunto con lo investigado dentro del estado del arte, a elegir la propuesta de la metodología y herramientas de desarrollo más adecuadas."

3.1 Aplicación web

Un sistema web o también conocido como aplicación web es un software al que se accede a través de internet o intranet y conforman una clase especial de aplicaciones de software que se construye de acuerdo con ciertas tecnologías y estándares [3].

De acuerdo con [3], las categorías de aplicaciones web se dividen en:

- Sitios web centrados en documentos
- Aplicaciones web interactivas, transaccionales, basadas en flujos de trabajo, colaborativas, orientadas a portales o de web ubicua.
- Portales generales y especializados
- Web semántica

En la gran mayoría de modelos tradicionales de aplicaciones web, estas se dividen en dos capas: *BackEnd* y *FrontEnd*. La primera hace referencia al control, manejo y comportamiento de los datos almacenados en la nube y la segunda a la presentación y manejo de estos por parte del usuario.

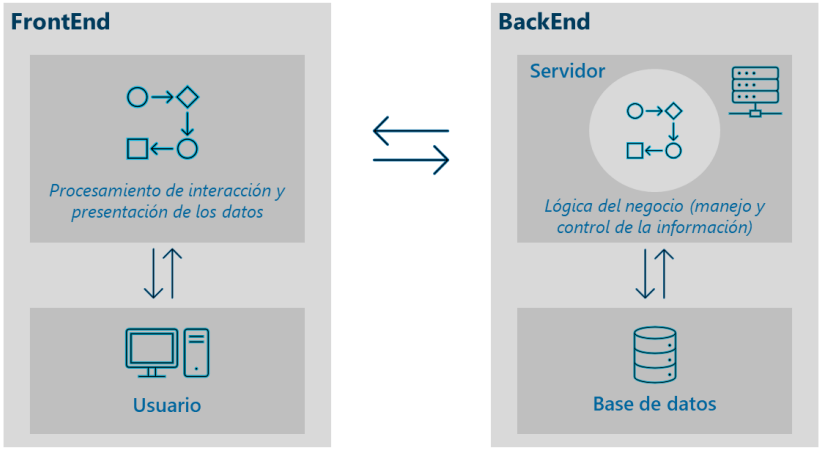

Figura 1. Flujo de comunicación entre BackEnd y FrontEnd (Elaboración propia)

3.1.1 BackEnd

Se denomina BackEnd a la capa de acceso a los datos de un software que no es accesible para el usuario final. Es en esta capa es donde se localiza la lógica de negocio, es decir, toda la lógica de la aplicación que describe el

comportamiento y flujo de los datos. Es en esta capa donde se realiza la conexión y consumo de datos con el servicio gestor de base de datos con el que se trabaje [4].

3.1.1.1 Node.js

Node es un entorno de ejecución orientado a eventos asíncronos[1] diseñado para construir aplicaciones web escalables haciendo uso del lenguaje *JavaScript* [5].

Una de las principales ventajas del uso de *Node.js* como plataforma para el *BackEnd* es que es impulsado por eventos asíncronos lo que de forma particular para el presente proyecto resulta de mucha utilidad pues mantiene una correcta integración con bases de datos de tiempo real.

3.1.2 FrontEnd

El FrontEnd hace referencia a la capa que se encarga de controlar la interacción y la presentación de la información al usuario, esto, procurando ofrecer una buena experiencia de usabilidad [5].

Para el desarrollo de *FrontEnd* existen muchas tecnologías con las que se puede trabajar, entre la más popular se encuentra *JavaScript* que se puede complementar con el uso de *frameworks* que establecen un marco de trabajo predefinido para agilizar el desarrollo como por ejemplo *Angular*, *BackboneJS*, *ReactJS* [5]. También existen lenguajes de transferencia de información como *XML*, *JSON* y *Ajax* para hacer solicitudes al servidor sin necesidad de refrescar la página completa, lo que viene muy bien para aplicaciones dinámicas y reactivas como la del presente proyecto.

3.1.2.1 React JS

React es una biblioteca JavaScript de código abierto usada para construir interfaces de usuario [6]. Tiene mantenimiento y respaldo por parte de la empresa *Facebook* y una comunidad de desarrolladores independientes. Utiliza el patrón de diseño MVC[2] y

---

[1] *Eventos asíncronos:* Comunicación en donde el emisor no espera una respuesta inmediata, dejando libre el canal de comunicación para más peticiones [17].

[2] *MVC (Modelo Vista Controlador):* Es un estilo de arquitectura de software que separa los datos de una aplicación, al interfaz de usuario y la lógica en tres componentes independientes [6].

entre las ventajas de su uso se tiene que es simple, declarativa, fácil de combinar, además que de acuerdo con [6]:

- Permite crear interfaces de usuario interactivas de una manera menos tediosa.
- Renderiza y actualiza los componentes correspondientes en función del cambio en la información, gracias al uso de estados por componentes.
- Permite generar código mas predecible y de fácil depuración.
- Permite la construcción de componentes encapsulados que poseen su propio estado que funciona de forma independiente al DOM[3].

Es por todo esto que el uso de esta librería enfocada al *FrontEnd* es adecuada para tomarse en cuenta en el diseño del presente proyecto, ya que apoya ese punto de dinamismo y facilidad de implementación que se busca en el desarrollo.

3.2 Dispositivos móviles

Los dispositivos móviles inteligentes, a grandes rasgos, se pueden definir como todo aparato electrónico que sea reducido en tamaño, incorpore pantalla táctil y posea conexión inalámbrica [7], sin embargo, al formar un grupo principalmente heterogéneo, no existe un acuerdo amplio para la clasificación de estos. De acuerdo con [7], la clasificación estaría dada por los siguientes grupos:

- Dispositivos de comunicación: ofrecen servicios de comunicación telefónica, incluyendo envío de mensajes de texto y multimedia.
- Dispositivos de computación: poseen mayor capacidad de procesamiento de datos, buscando ofrecer así, una experiencia similar o superior al de computadoras de escritorio.

[3] *DOM (Modelo de Objetos del Documento):* Representación de documentos HTML mediante objetos estandarizados [6].

- Dispositivos reproductores de multimedia: tienen la capacidad de reproducir varios formatos digitales de contenido multimedia.
- Dispositivos móviles grabadores de multimedia: permiten la grabación de datos en audio y vídeo.
- Consolas portátiles: se trata de dispositivos móviles que proporcionan una experiencia dinámica de juego.

Cabe destacar que las funcionalidades de los teléfonos inteligentes (*smartphones*) cubren todas las categorías anteriormente mencionadas. Estos dispositivos permiten desde realizar llamadas telefónicas, interactuar con contenido multimedia, hasta consumir servicios web [7].

Los teléfonos inteligentes desde hace algunos años, al verse masificado su uso, son unas de las principales plataformas en donde se enfocan múltiples desarrollos multidisciplinarios, desde los que cubren necesidades de comunicación e interacción social, hasta algunos que proveen herramientas educativas.

Las características de hardware que acompañan a estos dispositivos varían en cuanto a sus capacidades de procesamiento, de almacenamiento y la inclusión de ciertos sensores; de igual forma son variadas las versiones del sistema operativo que presenten, por todo esto es que los desarrollos de aplicaciones de alcance ambicioso deben de diseñarse contemplando estos factores para conseguir un correcto funcionamiento en la mayoría de los dispositivos.

3.2.1 Aplicaciones móviles

Una aplicación móvil es una herramienta diseñada para desarrollar una función específica en una plataforma móvil [8]. De acuerdo con [8], existen tres tipos de aplicaciones móviles:

- Aplicación nativa: son aquellas que son desarrolladas de forma específica para un determinado sistema operativo. Su interacción con las características del hardware se simplifica, además de que no requiere de una conexión a internet para que funcione.
- Aplicación web: se trata de una aplicación web tradicional con la que se puede interactuar con facilidad desde el dispositivo móvil accediendo por medio del navegador.

- Aplicación híbrida: combinan aspectos de las web y nativas según más convenga. Conlleva un desarrollo más rápido y cuenta con un acceso directo al hardware del dispositivo.

La aplicación por desarrollar se encuentra catalogada como una aplicación nativa, sin embargo, el uso de la plataforma de desarrollo de Unity, como se verá más adelante en la revisión del software, permite la exportación de aplicaciones nativas para distintas plataformas a partir de un solo proyecto. Por lo que, concretando, la aplicación presentada será desarrollada en un entorno multiplataforma que nos brinda la oportunidad de, según sea necesario en el futuro del proyecto de origen, generar compatibilidad con otros sistemas móviles como, por ejemplo, iOS[4].

3.2.2 Sistemas operativos móviles

Se conocen como sistemas operativos móviles a los sistemas operativos[5] incorporados en dispositivos móviles inteligentes, como son los *smartphones* [9]. Son sistemas orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las distintas maneras de ingresar información en dispositivos móviles [9].

3.2.2.1 Android

Android es un sistema operativo y una plataforma software, con base en Linux para uso principal en teléfonos inteligentes [10]. Dado que Android es de código abierto, se simplifica el desarrollo de aplicaciones, *widgets*[6] y modificaciones del sistema [10].

Fue desarrollado por la empresa Android Inc., la cual sería comprada por Google en el año 2005, sin embargo, no fue hasta el año 2008 que el sistema de código abierto ganó popularidad por su inclusión al proyecto de Open Handset Alliance, un consorcio conformado por varias empresas de desarrollo del ramo tecnológico [10].

---

[4] *iOS*: Sistema operativo actual de todos los dispositivos móviles de Apple [16].
[5] *Sistema Operativo*: Software que permite interactuar con hardware, gestionar datos y aplicaciones [9].
[6] *Widget:* Pequeña aplicación que facilita el acceso a funciones frecuentes [10].

Para comprender la estructura de Android, esta se puede ilustrar como un sistema en capas (*figura 3.1*), en donde, para funcionar, las capas superiores dependen de las inferiores. Como se puede observar en la *figura 3.1*, el *Kernel* de *Linux* es la base de Android, por lo que tiene acceso a sus recursos como lo son, los controladores de hardware del dispositivo.

Figura 2. Sistema de capas de Android (Elaboración propia)

3.3 Base de datos de tiempo real

Un sistema de base de datos (SBD) maneja datos persistentes garantizando durabilidad y consistencia a través de gestión de transacciones. Una transacción es un conjunto de operaciones que implementan una función lógica indivisible dentro de un SBD. En [11] se define a una transacción como: *"Una parte de un programa que accede y posiblemente actualiza varios datos con un fin específico."*. Todo conjunto de acciones ordenadas que debe llevarse a cabo para lograr un resultado en una base de datos (BD) está comprendido dentro de una transacción [11].

Las transacciones se definen con cuatro propiedades fundamentales, conocidas como propiedades *ACID*:

- Atomicidad: Una transacción se ejecuta por completo o no se ejecuta. Confirma todas sus operaciones satisfactoriamente (commit) o deshace sus operaciones si no pudo completarse (rollback).
- Consistencia: Una transacción que altera datos garantiza que la BD pasa de un estado consistente a otro, cumpliendo las reglas de integridad.
- Aislamiento: Las actualizaciones parciales de los datos no son visibles por otras transacciones hasta que la transacción se confirme.
- Durabilidad: Una vez que una transacción se ha confirmado el resultado debe persistir en la base de datos, aunque se produzcan fallas posteriores.

En los sistemas gestor de base de datos tradicionales se usa un enfoque de serialización, es decir, que se controlan las secuencias de operaciones que acceden a los datos compartidos para que su ejecución sea en serie y no concurrentes en caso de conflictos, lo que garantiza que la ejecución concurrente de transacciones no ponga en riesgo la consistencia de la información [11]. Sin embargo, este enfoque tradicional no es suficiente para dar solución a necesidades específicas, como, por ejemplo, las del presente proyecto, en donde se busca garantizar tiempos de ejecución y de respuestas estrictos.

Para dar solución de cara a cubrir las necesidades de sistemas de tiempo real, surgen los sistemas de base de datos en tiempo real los cuales consideran el vencimiento de las transacciones, así como la consistencia temporal entre los datos almacenado y los datos externos [11].

En los sistemas de bases de datos en tiempo real se necesita la capacidad de validad la consistencia temporal tanto absoluta como relativa y para esto se necesita registrar una marca de tiempo de lectura que recuerde la última actualización de cada dato respecto a su contraparte en el exterior, esto, incrementa las validaciones sobre las transacciones, ya que para que sean consideradas como correctas, no solo tienen que cumplir con la consistencia lógica si no también con la consistencia temporal [11].

3.3.1 Firebase Realtime Database

Se trata de un servicio parte de la plataforma de desarrollo web *Firebase* ofrecida por la empresa *Google*, la cual proporciona una base de datos en tiempo real y acceso simplificado mediante una *API Rest*, con la posibilidad de fácil integración con herramientas de JavaScript como *Angular*, *React JS*, *Ember JS* y *Backbone.js* [12].

Entre las funciones clave de este servicio de base de datos en tiempo real, sen enlistan las siguientes:

- *Tiempo real:* Usa el enfoque de sincronización de datos, esto es, cada vez que cambian los datos, los dispositivos conectados reciben esa actualización en milisegundos.
- *Sin conexión:* Permite que los datos persistan del lado del cliente en un estado de falta de conexión a la red, cuando esta se restablece, se reciben los cambios que faltaban y los sincroniza con el estado actual del servidor.
- *Escalamiento:* Según las necesidades de la aplicación se tiene la posibilidad de dividir la información en diversas instancias de bases de datos dentro del mismo proyecto.

Es por todo esto que el uso de este servicio de cara al desarrollo de la aplicación, cuyo diseño se presenta en el presente proyecto, resulta viable ya que responde a las necesidades del aplicativo, es de fácil uso mediante la API, además que el tiempo de desarrollo se vería reducido.

3.4 Metodología de desarrollo de software (RUP)

La metodología de desarrollo RUP (Proceso Unificado de Desarrollo) define una estructura de trabajo para el desarrollo de software que tiene por objetivo ordenar y estructurar el proceso de desarrollo [13]. Esta metodología es extensa y describe a detalle las fases tanto para proyectos pequeños como para proyectos más grandes [14].

El RUP se basa en principios iterativos e incrementales, es decir, el proyecto se divide en pequeñas iteraciones o ciclos, y cada iteración se enfoca en completar un conjunto de requisitos específicos [14]. En cada iteración, el equipo de desarrollo evoluciona el producto a partir de los resultados completados en las iteraciones previas, añadiendo nuevos requisitos, o bien, mejorando los que ya fueron completados [14].

Cada una de las fases que componen al RUP poseen sus propias actividades, roles y responsabilidades y se enfocan en cumplir con los objetivos específicos previo al avance de la siguiente fase [15].

Según [15], el proceso de desarrollo RUP consta de cuatro fases principales, cada una con sus objetivos y entregables específicos. A continuación, se describen dichas fases.

- **Inicio:** Es en esta fase donde se establece el alcance del proyecto, se identifican los interesados y se define una visión general del sistema. Los objetivos principales son establecer el problema a resolver, identificar los requisitos iniciales y determinar la viabilidad del proyecto.
- **Elaboración:** Fase en la que se realiza un análisis más detallado del sistema, se identifican y describen los casos de uso y se elabora una arquitectura sólida. Los objetivos principales son refinar los requisitos, definir la arquitectura del sistema y elaborar un plan detallado del proyecto.
- **Construcción:** En esta fase, se realiza la implementación del sistema, se elaboran pruebas y se refinan los componentes. Los objetivos principales son desarrollar y probar el software, verificar la calidad y preparar la entrega del producto.
- **Transición o cierre:** Fase en la que se hace la entrega final al cliente, se realiza la capacitación y se da soporte posterior a la implementación. Los objetivos principales son asegurar una transición exitosa del producto a un ambiente de producción, así como evaluar su rendimiento.

Como parte de la propuesta de desarrollo para el diseño del presente proyecto, se incluye la metodología de desarrollo RUP, ya que puede adaptarse a las necesidades y características de cada proyecto, permitiendo una mayor flexibilidad y adaptabilidad a los cambios en los requisitos que el cliente pueda solicitar. Del mismo modo, es preciso destacar que RUP puede ser combinado con otras metodologías ágiles como *SCRUM* o *XP*.

Capítulo IV

Sistema para el seguimiento en línea de la ruta de Lobobus

"El presente capítulo detalla la descripción de la propuesta de diseño del sistema, repasando los principales puntos técnicos, así como estructurales."

4.1 Análisis del sistema

4.1.1 Requerimientos del sistema

4.1.1.1 Alcance

Como se denotó con anterioridad, el presente proyecto tiene por objetivo generar una propuesta de diseño de un sistema web que permita ver el recorrido de las unidades en la ruta del servicio *Lobobus* dentro de Ciudad Universitaria y establecer los requerimientos de hardware necesarios para su posible puesta en marcha de forma real. Por ello, crear un marco de referencia, tanto en elementos técnicos como de diseño para un sistema completo que realice la tarea del seguimiento de unidades del transporte interno en tiempo real, permitiendo así a la comunidad estudiantil mejorar su movilidad por el complejo universitario.

4.1.1.2 Descripción general

Concretando, en cuanto a la funcionalidad del sistema, el flujo de actividades, por la parte de usuarios administrativos, comienza por el registro de unidades, es decir, el alta en el sistema de las unidades de transporte, esto, para realizar su seguimiento. Ya teniendo las unidades registradas, cada que alguna entre en servicio deberá de cambiar en el sistema su estatus para que éste reconozca que entró en operación y comience a realizar su seguimiento y mostrarlo en la página.

Cabe destacar que el cambio de estatus para el inicio de seguimiento de la unidad deberá realizarse por medio de una aplicación móvil, parte del sistema web, esto, debido a que, como se vio en el marco teórico, resulta accesible y de fácil implementación una solución móvil para el rastreo de las unidades, ya que los dispositivos móviles inteligentes ya cuentan con el hardware necesario para un seguimiento por *GPS* en tiempo real, además de que el sistema operativo móvil *Android* ya cuenta con la plataforma de desarrollo para obtener y enviar de forma sencilla esta información. Por tanto, el flujo general de operación del sistema, teniendo en cuenta el uso

de estas dos plataformas para con el sistema web se vería descrito como el representado en la *figura 3*.

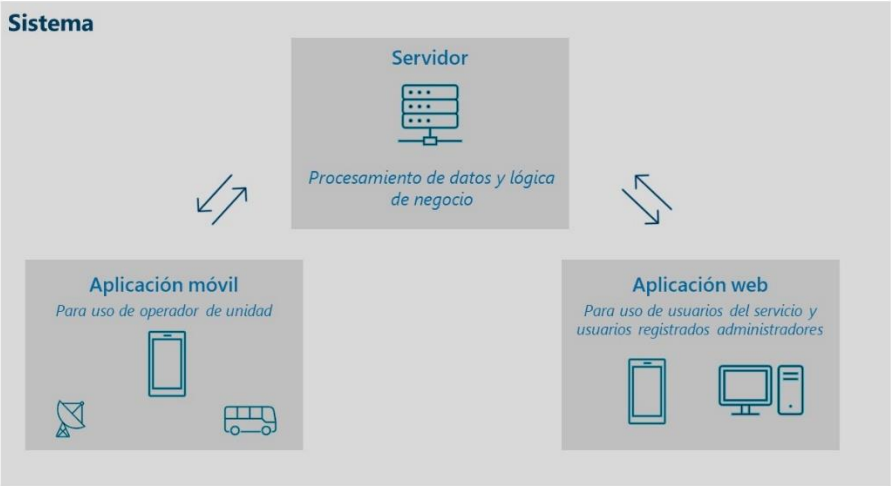

Figura 3. Flujo de operación del sistema (Elaboración propia)

Por parte del usuario del servicio, solo deberá entrar a la página del sistema, resolver un captcha[7], y se le mostrará las ubicaciones de las unidades, así como un estimado de cuánto tiempo tardará en llegar una unidad a cada una de las paradas del servicio. La inclusión de un captcha, para comenzar a visualizar la información responde a una medida de seguridad para evitar ataques.

[7] Captcha: Medida de seguridad que consiste en una prueba para determinar sí el acceso al servicio responde a uno automatizado.

4.1.1.3 Requerimientos funcionales del sistema

A continuación, se presentan los principales requerimientos funcionales identificados del sistema.

- El sistema debe contar con un módulo de control de usuarios y roles, esto, para identificar a administradores y operadores.
- El sistema deberá permitir el control de registro de las unidades del sistema de transporte *Lobobus*, esto es, su alta, baja y edición.
- Los operadores deben tener un módulo específico y de rápido accionar para cambiar el estatus de su unidad asociada indicando su inicio o fin de operaciones.
- El usuario del servicio, por lo general la comunidad estudiantil, debe de ver de forma clara y rápida las ubicaciones de las unidades en ciudad universitaria.
- El sistema deberá realizar un cálculo del tiempo de llegada por unidad a las diferentes paradas de la ruta.
- El sistema deberá contener una sección de estadísticas con indicadores clave para que administradores e incluso de forma eventual todos los usuarios puedan ver el estatus del servicio.

4.1.1.4 Requerimiento no funcionales del sistema

Entre los requerimientos no funcionales identificados del sistema, se destacan los siguientes:

- El sistema debe de estar diseñado pensando en su uso en dispositivos móviles, por lo que debe ser responsivo y de fácil uso en pantalla chicas.
- El diseño gráfico del sistema debe de apegarse a los estándares gráficos de la universidad.
- El control de los usuarios y contraseñas solo podrá controlarse por usuarios administradores.
- La página de acceso a usuarios generales del sistema debe estar protegida por un captcha para evitar ataques al servidor.

4.1.2 Casos de uso del sistema

A continuación, se presenta el diagrama de casos del uso del sistema, el cual describe las actividades que se realizan y que usuarios las realizan.

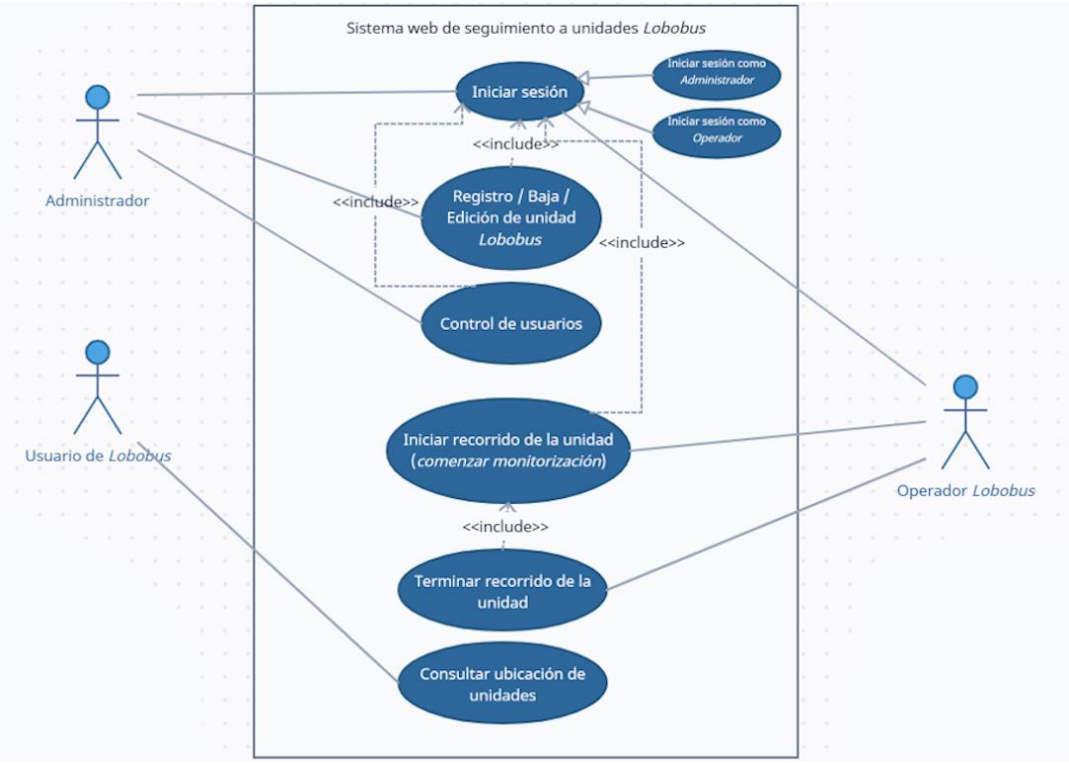

Figura 4. Diagrama de casos de uso del sistema

4.1.3 Especificación de casos de uso

4.1.3.1 Registro y baja de información de unidad *Lobobus*

4.1.3.1.1 Descripción

Se trata del CRUD[8] de los registros asociados a las unidades de *Lobobus*, los cuales tienen asociados datos como identificador, operador u operadores asociados y datos generales del vehículo. Esta acción podrá ser realizada únicamente por usuarios de tipo administrador.

[8] CRUD: Acciones de Crear, Leer, Actualizar y Borrar sobre elementos o entidades de un sistema, haciendo referencia a su información almacenada en una base de datos.

4.1.3.1.2 Flujo de eventos

El flujo de eventos para completar la acción del caso de uso es el siguiente:

1. Inicio de sesión con usuario de rol administrador
2. Acceso al módulo de control de unidades registradas
3. Ejecución de edición seleccionada
4. Guardado de cambios por parte del sistema

4.1.3.1.3 Precondiciones

La acción del caso de uso únicamente podrá ser efectuada sí el usuario con el que se inició sesión corresponde al rol de administrador.

4.1.3.2 Control de usuarios

4.1.3.2.1 Descripción

Se trata del control de datos de los usuarios registrados en el sistema, manejo de roles y contraseñas. Esta acción es realizada por usuarios de tipo administrador.

4.1.3.2.2 Flujo de eventos

El flujo de eventos para completar la acción del caso de uso es el siguiente:

5. Inicio de sesión con usuario de rol administrador
6. Acceso al módulo de control de usuarios
7. Ejecución de edición seleccionada
8. Guardado de cambios por parte del sistema

4.1.3.2.3 Precondiciones

La acción del caso de uso únicamente podrá ser efectuada sí el usuario con el que se inició sesión corresponde al rol de administrador.

4.1.3.3 Iniciar/Terminar recorrido de la unidad

4.1.3.3.1 Descripción

Esta acción corresponde a indicar al sistema que una de las unidades registradas está activa, es decir, efectuando su recorrido, lo que accionará el seguimiento de esta.

4.1.3.3.2 Flujo de eventos

El flujo de eventos para completar la acción del caso de uso es el siguiente:

1. Inicio de sesión con usuario de rol administrador u operador
2. Acceso al módulo de control de estatus de unidades
3. Ejecución de cambio de estatus
4. Guardado de cambios por parte del sistema e inicio de procedimiento de seguimiento, en el caso del presente proyecto, simulación de seguimiento.

4.1.3.3.3 Precondiciones

- La acción solo podrá ser realizada sí antes la unidad ya ha sido registrada y asociada a uno o más operadores.
- Un operador sólo podrá cambiar el estatus de la o las unidades que tiene asociadas.

4.1.3.3 Consultar ubicación de unidades

4.1.3.3.1 Descripción

Se trata de la acción principal del sistema, efectuada principalmente por los usuarios de la comunidad estudiantil. Para esta acción no es necesario iniciar sesión.

4.1.3.3.2 Flujo de eventos

El flujo de eventos para completar la acción del caso de uso es el siguiente:

1. Resolución de captcha

2. Visualización de la información
3. Posibilidad de consulta de parada especifica, indicando tiempo de espera aproximado para arribo de unidad, esto, por parte del sistema

4.2 Diseño del sistema

4.2.1 Diagrama de clases

Se presenta el diagrama de clases, el cual, ejemplifica la estructura general del sistema.

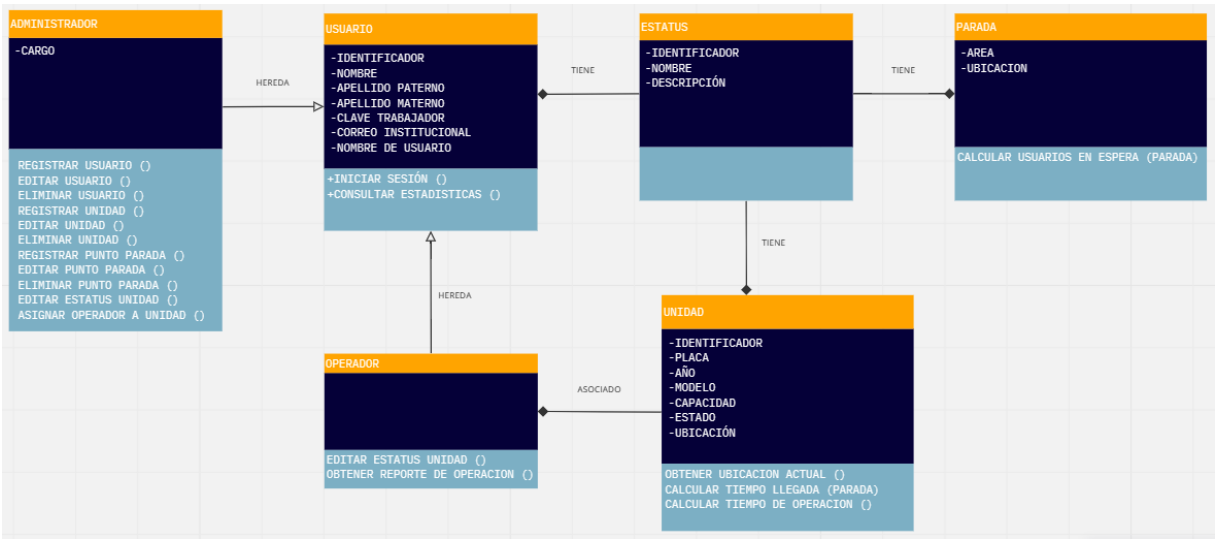

Figura 5. Diagrama de clases del sistema

A continuación, se describen las diferentes clases identificadas que compondrían la lógica general del aplicativo:

- *Usuario:* Hace referencia a los usuarios que controlan de alguna forma la operación del sistema, estos, deberán iniciar sesión para acceder a las funciones que les corresponden.
- *Administrador:* Se trata del usuario con el rol de *Administrador*, este tipo de usuario es el encargado de controlar los catálogos de usuarios y de unidades.
- *Operador:* Describe al usuario con rol de *Operador*, quien es la persona que opera una unidad designada previamente por un usuario *Administrador*.

- *Unidad:* Hace referencia a la unidad de transporte de la cual nos interesa conocer su ubicación, así como el tiempo estimado de llegada a cierta parada.
- *Parada:* Se trata de los puntos de parada en la ruta de las unidades. Cabe destacar que, al tener las coordenadas de esta, al momento de ingresar a la aplicación por parte del usuario final (*comunidad estudiantil*), se puede solicitar la ubicación para poder determinar el número de usuarios que está esperando en determinada parada.
- *Estatus:* Describe el *estatus* que guardan los diferentes objetos del sistema, por ejemplo, en el caso de la *unidad* sería si esta se encuentra en operación o no; para el caso de la *parada* sería si se encuentra activa o no ya que podría estar ese tramo de la ruta no disponible.

4.2.2 Diagrama de secuencia

Se presenta el diagrama de secuencia, que describe la interacción que surge en el proceso base del uso del sistema.

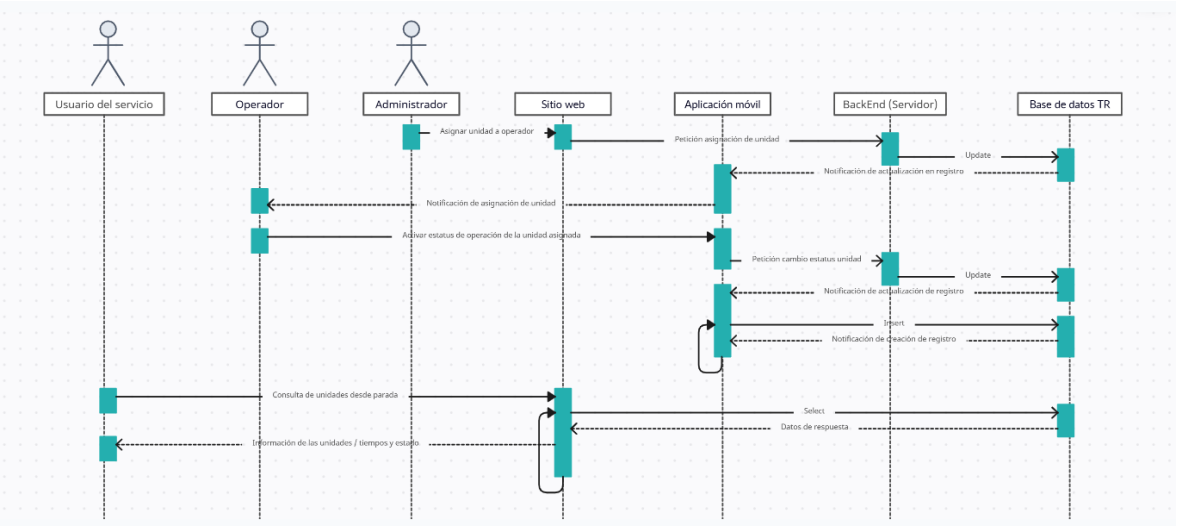

Figura 6. Diagrama de secuencia general del sistema

Como se puede notar en la *Figura 6*, se describe el comportamiento de las diferentes parten que conforman el sistema. A continuación, se enlistan observaciones que resultan importantes de denotar, con base en lo presentado en el diagrama de secuencia:

- Las peticiones que se manejan son asíncronas, esto para garantizar mayor dinamismo en el sistema, prevención de errores al fallar en algunas de las peticiones por intermitencias en la red o alguna otra

causa externa, además de que las tecnologías propuestas para la posible implementación exigen este tipo de comportamiento como es el caso de la base de datos de tiempo real de *Firebase*, o bien porque son herramientas diseñada para usar este comportamiento de forma nativa como es el caso de *NodeJS*.

- La base de datos de tiempo real dispara diferentes acciones al notificar a los clientes activos de cambios en los registros.
- Existen procesos que mantienen una recursividad mientras un cliente acceda al recurso. Esta se encuentra sujeta a los estatus de sus dependencias.

4.2.3 UI del sistema

A continuación, se presenta el diseño de la interfaz de usuario para las dos aplicaciones que conformarían el sistema.

4.2.3.1 Aplicación móvil

Recordemos que la aplicación móvil es necesaria para llevar el seguimiento de la unidad de transporte haciendo uso del GPS integrado en el dispositivo móvil inteligente, así como haciendo uso de las herramientas para la obtención y envío de estos datos ofrecidas por el sistema operativo móvil *Android*. Así mismo, en la aplicación el operador controlará el estatus de su unidad, sí se encuentra activa en recorrido o no, además de que le notificará que unidad tiene asignada.

Figura 7. Pantalla de inicio de sesión – Aplicación Móvil

El acceso a la aplicación móvil solo estaría habilitado para operadores de las unidades de transporte, por lo que desde el portal de administración en el sitio web del sistema se le tendría que asignar las credenciales para su acceso, mismas que el operador introduciría en la pantalla de inicio de sesión de la *App* (*Figura 7*).

Figura 8. Pantalla principal (Sin iniciar recorrido) – Aplicación Móvil

Figura 9. Pantalla principal (En ruta) – Aplicación Móvil

Una vez iniciada sesión en la aplicación móvil por parte del operador se encontraría con la pantalla principal, la cual le presenta información sobre su perfil, así como la posibilidad de iniciar o terminar ruta, lo que indica que se comience o finalice el rastreo de su ubicación para efectos de seguimiento de la ruta por parte del sistema. (*Figuras 8 y 9*)

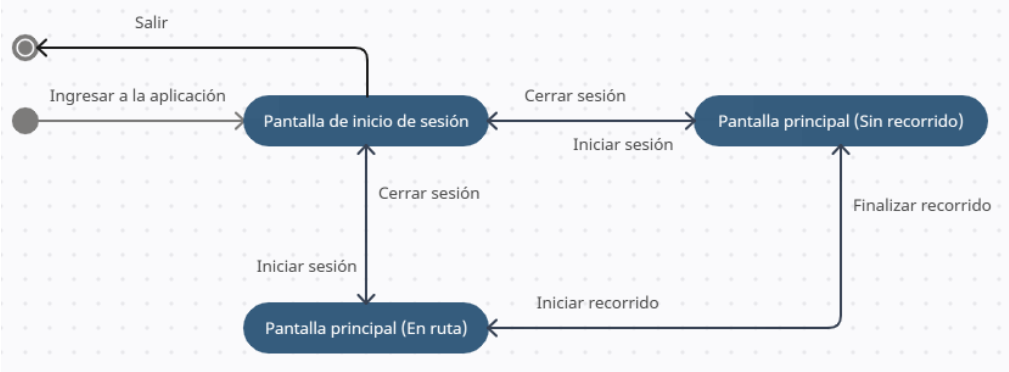

Figura 10. Diagrama de navegación – Aplicación móvil

La navegación de la aplicación resulta entonces simple, ya que la función principal de esta es la recolección de datos más que la interacción con el usuario, esta navegación responde a la descrita en la *figura 10*.

4.2.3.2 Sitio web

Es el sitio web el medio por el que los usuarios del servicio de transporte, así como los administradores podrán acceder al sistema. Cabe destacar que solo las operaciones de administrador requerirán de inicio de sesión a diferencia de consultar la ruta que no lo solicita (*Figuras 11 y 12*).

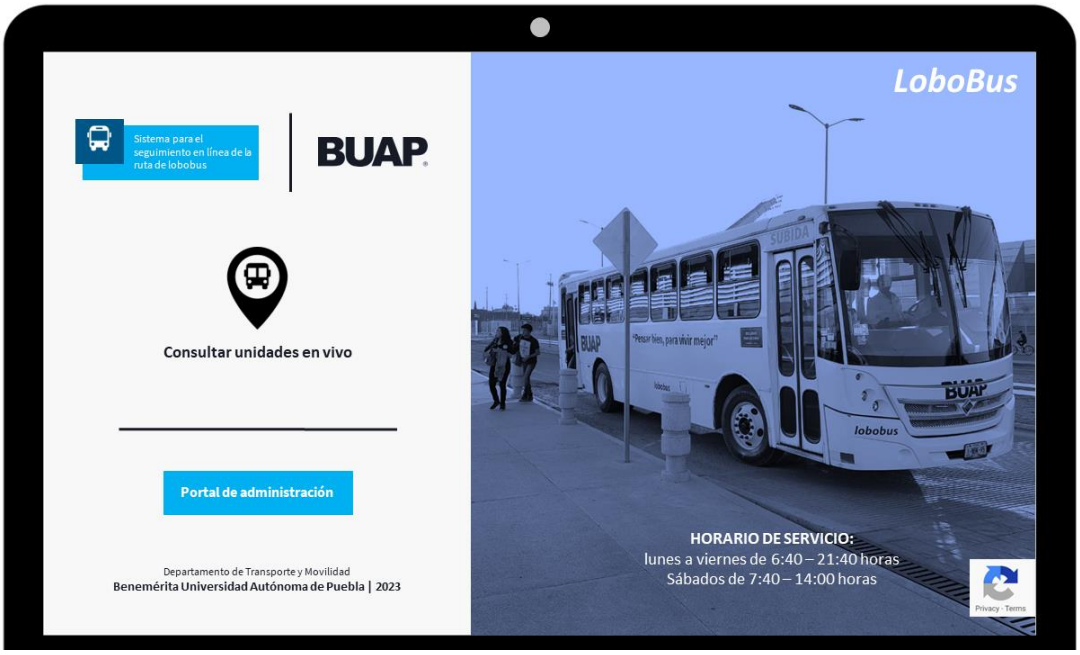

Figura 11. Página inicial – Sitio Web

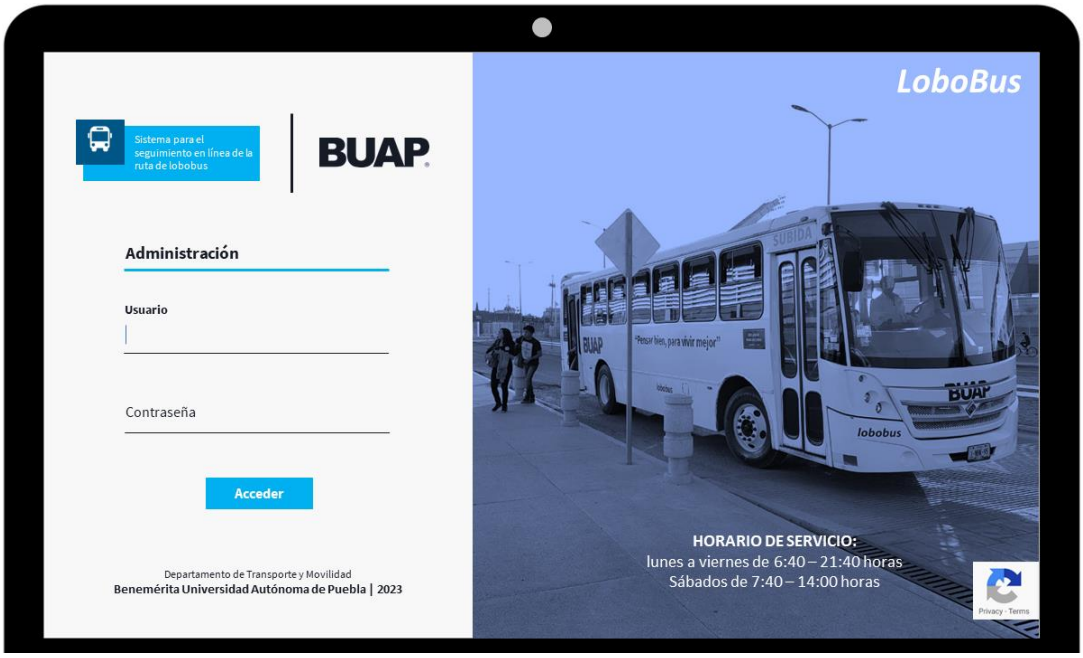

Figura 12. Página de Login – Sitio Web

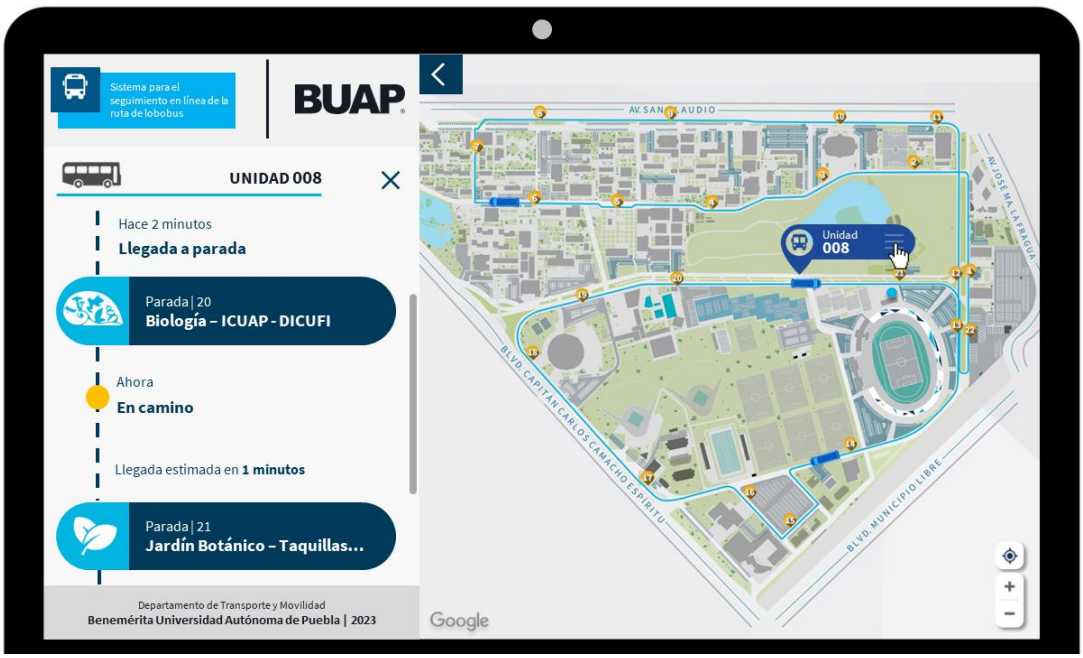

Figura 13. Página de consulta de unidades en vivo – Sitio Web

Como se observa en la *Figura 13*, la pantalla de consulta de unidades, que es la que se les mostrará a los usuarios del servicio, presentará un mapa dinámico en donde se podrán visualizar las unidades que se encuentran en ruta. Al seleccionar alguna de ellas, el usuario podrá ver el detalle de la ruta de la unidad, así como las diferentes paradas que va realizando y un estimado de llegada para las siguientes.

La navegación del sitio web debe tener en cuenta las dos modalidades de visualización, esto es, el portal de administración el cual exige un inicio de sesión, y el portal de consulta de unidades que no requiere de una autenticación. La navegación sobre el sitio web, se ve descrita en el diagrama de navegación indicado en la *Figura 14*.

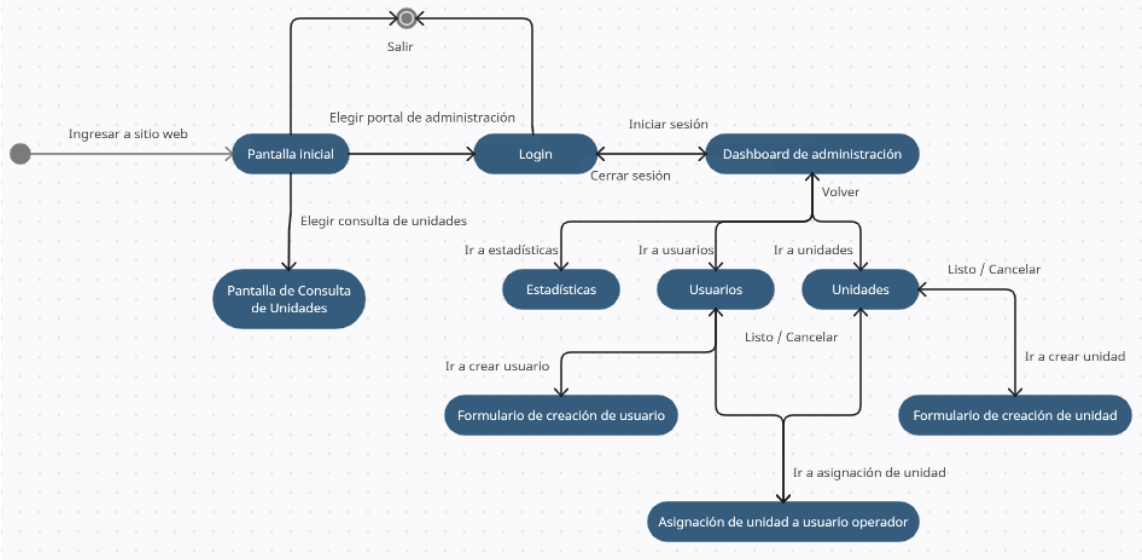

Figura 14. Diagrama de navegación – Sitio Web

4.2.4 Base de datos del sistema

Como se revisó con anterioridad, la base de datos propuesta a utilizar para el proyecto trata de una base de datos en tiempo real, particularmente la ofrecida por el servicio de *Firebase* de nombre *Realtime Database*, la cual es una base de datos no relacional, es decir No SQL, por lo que el modelado de esta no se puede expresar en un diagrama entidad relación convencional. A continuación, entonces, se presenta el modelado de las colecciones que conforman la base de datos, estos refieren a las entidades u objetos que contendrán la información del sistema.

```
usuarios: {
        identificador_usuario (string),
        nombre_usuario (string),
        tipo_usuario (string),
        datos_personales: {
                nombre (string),
                apellido_paterno (string),
                apellido_materno (string),
                telefono: (string)
        },
        datos_internos: {
                clave_trabajador (string),
                correo_institucional (string)
        },
        estatus: {
                identificador (string),
                nombre (string),
                descripcion (string)
        },
        fecha_creacion (marca de tiempo),
        fecha_actualizacion (marca de tiempo)
}
```

```
administradores: {
        identificador_usuario (string),
        cargo (string),
        fecha_actualizacion (marca de tiempo)
}

operadores: {
        identificador_usuario (string),
        identificador_unidad (string),
        fecha_actualizacion (marca de tiempo)
}
```

```
unidades: {
        identificador_unidad (string),
        datos_vehiculo: {
                placa (string),
                año (number),
                modelo (string),
                capacidad (number),
                estado (string)
        },
        estatus: {
                identificador_estatus (string),
                nombre (string),
                descripcion (string)
        }
        fecha_creacion (marca de tiempo),
        fecha_actualización (marca de tiempo)
}
```

```
paradas: {
        area: {
                facultad (string),
                nombre (string)
        },
        estatus: {
                identificador_estatus (string),
                nombre (string),
                descripcion (string)
        },
        ubicacion: {
                latitud (number),
                longitud (number)
        },
        fecha_creacion (marca de tiempo),
        fecha_actualización (marca de tiempo)
}
```

```
ubicaciones_unidades: {
        identificador_unidad (string),
        ubicación: {
                latitud (number),
                longitud (number)
        },
        tiempo (marca de tiempo)
}
```

Figura 15. Modelo de base de datos del sistema – BD no relacional

En relación al modelo de datos presentado anteriormente, con el propósito de tener un mayor control sobre los registros guardados, inclusive pensando en temas de auditoría al sistema, se añadió la subcolección de *estatus* que servirá para realizar eliminaciones lógicas, es decir, cambiar el estatus del dato asociado cuando éste sea eliminado desde la aplicación, por lo que ya no se mostrará, sin embargo, el registro seguirá existiendo en la base y podrá ser recuperado.

La colección principal, que tendrá mayor número de documentos, es la de nombre "*ubicaciones_unidades*", ésta hace referencia a las ubicaciones registradas por la aplicación móvil de las unidades, es por ello por lo que cada cierto tiempo, dependiendo de la configuración en la aplicación, se estaría escribiendo un documento, mismo que se reflejaría de forma inmediata en los usuarios activos.

En cuanto a los datos de acceso para inicio de sesión, como es la contraseña, no se incluyen en el modelado, ya que parte de la propuesta incluye el uso del módulo de autenticación ofrecido por el servicio de *Firebase*, el cual solo se debe configurar para que controle los accesos a los aplicativos.

4.2.5 Propuesta de implementación del sistema

Una vez que todas la bases técnicas y partes del modelo del sistema fueron discutidas, a continuación, se enlista la propuesta, de forma general, de los pasos técnicos que habría que seguir para el desarrollo propio de la solución web.

En principio se tendría que trabajar la parte del *BackEnd* en *NodeJS*, que estaría soportada por la plataforma Firebase, la cual además ofrece servicio de alojamiento web, autenticación de usuarios y módulo de base de datos en tiempo real. Para ello se tendría que iniciar sesión en el portal oficial del servicio (*https://console.firebase.google.com/*) y seleccionar la creación de un nuevo proyecto (*Figura 16*).

Figura 16. Creación de proyecto en Firebase, 2023, Firebase Console (https://console.firebase.google.com)

Posterior a la creación del proyecto, se crearía la base de datos (*Figura 17*), en el apartado de *compilación* > *Realtime Database*, en donde se iniciaría con la configuración básica recomendada por el servicio, y se crearía la estructura inicial siguiendo el modelo propuesto en la sección de base de datos del sistema.

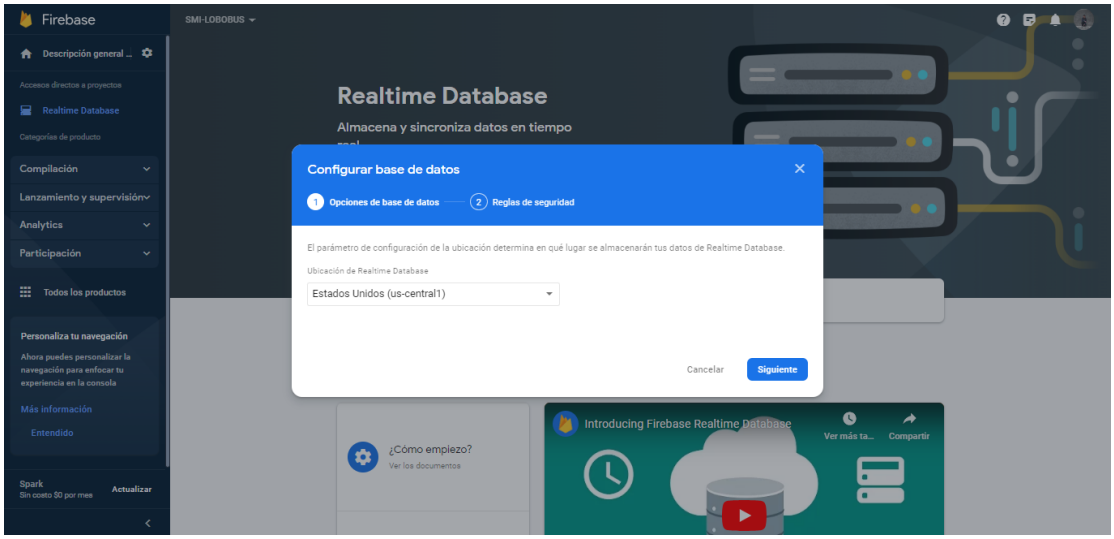

Figura 17. Creación de base de datos en Firebase, 2023, Firebase Console (https://console.firebase.google.com)

Finalmente se activaría y configuraría el módulo de autenticación (*Figura 18*) de usuarios, en donde se tendría que elegir la modalidad de autenticación

por correo y contraseña, para que los usuarios puedan acceder haciendo uso de su correo institucional.

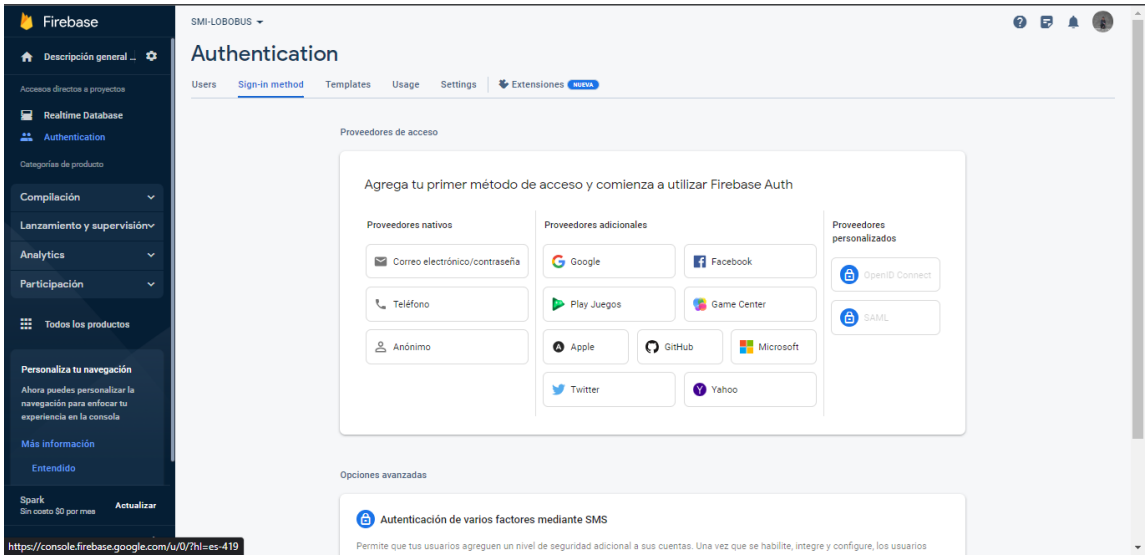

Figura 18. Activación de módulo de autenticación en Firebase, 2023, Firebase Console (https://console.firebase.google.com)

Ahora se comenzaría con el desarrollo de la aplicación web, para su creación en *React* se requiere de la instalación del sistema gestor de paquetes *npm*. En donde se inicializaría el proyecto aplicando las sentencias:

- $ npx create-react-app web-seguimiento-lobobus
- $ npm install firebase
- $ npm start

Una vez que el proyecto de *React* se encuentre inicializado, se comenzaría con la vinculación entre el aplicativo y las dependencias de *Firebase*, de cara a la conexión a la base de datos y a la implementación de las funcionalidades de la plataforma. Para ello en el portal de *Firebase* se deben obtener los identificadores secretos del proyecto para añadirlos al archivo de configuración dentro del proyecto (*Figura 19*).

```
/*CONFIGUARCIÓN PARA USAR FIREBASE EN TODO EL PROYECTO*/

import firebase from 'firebase/app';
import 'firebase/auth';
import 'firebase/realtime';

var firebaseConfig = {
   apiKey: "AIzaSyDOayv4uy34463tv34t3kkgyyOYs",
   authDomain: "web-seguimiento-lobobus-67a6d.firebaseapp.com",
   databaseURL: "https://web-seguimiento-lobobus-67a6d.firebaseio.com",
   projectId: "web-seguimiento-lobobus-69a3d",
   storageBucket: "web-seguimiento-lobobus-67a6d.appspot.com",
   messagingSenderId: "845451210481",
   appId: "1:84525320481:web:5344beec118cff12f3253250",
   measurementId: "Y-QQ7C35YTS3243C"
};

const fb = firebase.initializeApp(firebaseConfig);

const fb_aux = firebase.initializeApp(firebaseConfig, "Secondary");

export const auth = fb.auth();
export const db = fb.realtime();

export const auth_aux = fb_aux.auth();
```

Figura 19. Código configuración Firebase en React

Una vez que se tiene todo configurado, se puede comenzar con el desarrollo propiamente de la aplicación web en el *FrontEnd*, generando los componentes con las funciones de conexión con la base de datos (*Figura 20 y 21*), así mismo por parte del *BackEnd*, para la parte de la lógica del negocio, se puede hacer uso del servicio de "*Cloud Functions*", ofrecido por *Google Cloud* y que viene integrado a *Firebase*. Estas *"Cloud Functons"*, son bloques de código en *NodeJS* a los que se pueden acceder mediante peticiones *HTTP*, por lo que estas pueden ser usadas de forma nativa en el código dentro de *React* (*Figura 22*).

```
import { getDatabase, ref, onValue } from "firebase/database";

const db = getDatabase();
const starCountRef = ref(db, 'posts/' + postId + '/starCount');
onValue(starCountRef, (snapshot) => {
  const data = snapshot.val();
  updateStarCount(postElement, data);
});
```

Figura 20. Consulta básica a Realtime Datatable desde React

```
import { getDatabase, ref, set } from "firebase/database";

function writeUserData(userId, name, email, imageUrl) {
  const db = getDatabase();
  set(ref(db, 'users/' + userId), {
    username: name,
    email: email,
    profile_picture : imageUrl
  });
}
```

Figura 21. Escritura básica a Realtime Datatable desde React

```
//Se actualiza la contraseña de la cuenta del usuario seleccionado (llamado a cloud function)********
fetch("https://us-central1-sistema-escuela-69a3d.cloudfunctions.net/actualizarPassword?uid="+this.datosIniciales.uid+"&pass="+password_temp)
.then(response => response.json())
.then(data => {
        //Sí se completó correctamente la actualización de la contraseña (auth)
        if(data.completado) {
                //---------------------------------------------------------------------------------------------------
                alert_cargando.close();
                //Mensaje de éxito
                swal.fire({
                  title:"<p style='font-size:13pt;font-weight:600;color: #282828;'>Cambio realizado</p>",
                  icon: 'success',
                  iconColor: '#17a2b8',
                  width: '300px',
                  showConfirmButton: false,
                  timer: 2500
                })
                //---------------------------------------------------------------------------------------------------
```

Figura 22. Ejemplo de llamado a Cloud Function desde React

En cuanto al desarrollo de la aplicación móvil, se propone hacer uso del entorno de trabajo *React Native*, que no es más que, de forma general, la versión para desarrollo de aplicaciones móviles de su contraparte web *React*. La ventaja de usar este entorno es que se puede reutilizar el código empleado para la aplicación web, como, por ejemplo, el de configuración para el uso de *Firebase* (*Figura 19*), o el utilizado para la intercomunicación con la base de datos (*Figura 20 y 21*). Sin embargo, la parte que requeriría más atención es la de la interfaz que hará la lectura de los datos brindados por *GPS* (*Figura 23*).

```
watchLocation = () => {
  const { coordinate } = this.state;

  this.watchID = navigator.geolocation.watchPosition(
    position => {
      const { latitude, longitude } = position.coords;

      const newCoordinate = {
        latitude,
        longitude
      };

      if (Platform.OS === "android") {
        if (this.marker) {
          this.marker._component.animateMarkerToCoordinate(
            newCoordinate,
            500 // 500 is the duration to animate the marker
          );
        }
      } else {
        coordinate.timing(newCoordinate).start();
      }

      this.setState({
        latitude,
        longitude
      });
    },
    error => console.log(error),
    {
      enableHighAccuracy: true,
      timeout: 20000,
      maximumAge: 1000,
      distanceFilter: 10
    }
  );
};
```

Figura 23. Función de seguimiento de ubicación en React Native

Conclusiones y trabajo futuro

Para concluir el presente trabajo de tesina, a continuación, se presenta una relación de los objetivos y cómo fue que se abordaron y concluyeron. Por último, se habla sobre el trabajo a realizar en el futuro con base en lo conseguido.

El objetivo general del proyecto, que responde a la generación de una propuesta de diseño de un sistema web que realice el seguimiento a las unidades del sistema de trasporte interno *Lobobus*, queda completado, siendo que el presente trabajo de tesina puede servir como un marco de referencia y guía técnica importante para el desarrollo e implementación de la solución web.

Se cumple además con definir las propuestas de uso de tecnologías para la implementación del sistema, con base en el análisis e investigación del porqué de su inclusión, respondiendo a las necesidades de los aplicativos a desarrollar.

De igual forma, se sentaron las bases del diseño de interfaz de usuario para las aplicaciones involucradas en el sistema, así como de la descripción de la navegación en ellas, todo esto, con base en los casos de uso identificados.

Como trabajo futuro se espera presentar la propuesta de diseño del sistema al área correspondiente (BUAP) y en caso de tener una respuesta positiva, comenzar propiamente con el desarrollo funcional de los aplicativos con la retroalimentación constante del área y adaptando el presente diseño, en caso de ser necesario, en relación con los posibles nuevos requerimientos que sean dados.

Referencias

[1] V. V. A. F. V. S. A. M. L. P. F. N. D. &. P. M. I. M. Fernández, «Bicialert: Aplicación web para el seguimiento de bicicletas robadas en Bogotá, Colombia.,» *Investigación Formativa en Ingeniería,* pp. 344-352, 2020.

[2] A. I. Rentería Ayala, «Aplicación para localización de taxi a través de GPS en tiempo real,» Universidad Autónoma de Ciudad Juárez, Ciudad Juárez, 2014.

[3] A. D. F. J. &. M. M. L. Oliveros, «Prácticas de Ingeniería de Requerimientos en el desarrollo de aplicaciones Web,» de *CIbSE*, 2014, pp. 491-505.

[4] S. G. Q. J. R. M. F. F. &. L. D. A. Pérez Ibarra, «Herramientas y tecnologías para el desarrollo web desde el FrontEnd al BackEnd,» de *XXIII Workshop de Investigadores en Ciencias de la Computación*, Chilecito, La Rioja, 2021.

[5] E. G. T. P. A. O. Z. &. Q. G. N. Haro, «Desarrollo backend para aplicaciones web, servicios web restful: Node. js vs spring boot,» de *Revista Ibérica de Sistemas e Tecnologias de Informação*, 2019, pp. 309-321.

[6] P. Berbel Marín, «Desarrollo de un frontend en ReactJS,» Universitat Jaume I, 2018.

[7] R. A. Garita-Araya, «Tecnología Móvil: desarrollo de sistemas y aplicaciones para las Unidades de Información,» *E-Ciencias de la Información,* vol. 3, nº 2, pp. 1-15, Julio - diciembre 2013.

[8] M. L. Castañeda, «Qué son las app y tipos de apps,» Universidad Tecnológica de Pereira, Pereira, 2015.

[9] S. V. Á. R. Joao Ranieri, «Sistemas operativos,» Academia Accelerating the world's research, 2013. [En línea]. Available: https://www. edu. xunta. gal/centros/iesblancoamorculleredo/aulavirtual2/pluginfile. php/25655/mod_page/content/30/SistemasOperativos_JoaoRanieri_AlvaroRodriguez_SergioVillar. pdf. [Último acceso: 2022].

[10] Á. B. J. C. L. C. M. G. F. H. D. P. J. R. d. L. D. S. M. S. P. T. Á. Z. Manuel Báez, «Introducción a Android,» Universidad Complutense de Madrid, Madrid, 2019.

[11] C. E. U. J. M. &. P. F. E. Buckle, «Transitando hacia las bases de datos de tiempo real,» de *7º Jornada de Informática Industrial (JII 2010)-JAIIO 39*, 2010.

[12] E. U. A. B. M. A. P. P. &. A. I. S. M. A. Aguiar, «Firebase en el desarrollo de aplicaciones móviles,» UNIVERSIDAD POLITÉCNICA DE SINALOA, Mazatlán, Sinaloa, 2018.

[13] L. C. C. M. R. M. L. B. S. R. P. M. Daniel Alexander Vera Paredes, «Análisis de la metodología RUP en el desarrollo de software académico mediante la herramienta DJANGO,» *Revista Científica Mundo de la Investigación y El Conocimiento,* vol. 3, nº 2, pp. 664-679, 2019.

[14] A. &. M. R. Martínez, «Guía a rational unified process,» Escuela Politécnica Superior de Albacete–Universidad de Castilla la Mancha, Albacete, 2014.

[15] R. A. &. P. M. J. A. López Rosciano, «Desarrollo de herramienta de gestión de proyectos RUP usando metodología SCRUM+ XP: Pruebas,» Universidad Politécnica de Madrid, Madrid, 2015.

[16] C. D. J. C. Avila Cruz H. C., «Guía para la realización de aplicaciones móviles en los sistemas operativos Android e IOS,» Universidad Distrital Francisco José de Caldas, Bogotá, 2016.

[17] E. C. R. Navarro Delgado, «Sistemas de comunicación asíncrona aplicados a la web en tiempo real,» Universidad de San Carlos de Guatemala, 2016.