

Universidad de Oriente

Sede “Julio Antonio Mella”

Facultad de Ingeniería en Telecomunicaciones Informática y Biomédica

Trabajo de Diploma

En opción al título de Ingeniero en Informática

Autor: Fidel Alejandro Cepero Salazar

Tutor: MSc. Dionis López Ramos

Título: Simulación del tránsito de una ciudad usando técnicas de BigData

2023

“Año 65 de la Revolución

**Resumen**

La población necesita de sistemas que generen en tiempo real la ubicación del transporte. En los tiempos actuales de desarrollo tecnológico, las formas tradicionales no satisfacen está demanda. Para resolver esta necesidad se crearon

Las ciudades de alrededor de medianas dimensiones (de 50 000 a 1 000 000 habitantes) deben gestionar los servicios de transporte de estos núcleos poblacionales. Centrando el problema desde el punto de vista de los ciudadanos para conocer el estado del transporte público para una gestión mejor del tiempo y los servicios es crucial para la vida diaria.

El flujo de información que genera el conocimiento en tiempo real del estado del servicio de transporte público en la ciudad es grande. Este encierra el estado o posición geográfica de los vehículos de transporte, cantidad de personas, cantidad de personas en las paradas, pagos realizados entre otro tipo de información**.** Lo que hace que el manejo de grandes volúmenes de datos sea tan útil, que proporciona respuestas a muchas preguntas que los ciudadanos como usuarios finales que ni siquiera sabían que tenían, partiéndose la captura del mayor número de datos posible con la idea de poder utilizar estos en un futuro para estudiar y mejorar la materia de la cual se han obtenido estos. En otras palabras, proporciona un punto de referencia**.**

Palabras clave: transporte, tiempo, volúmenes de datos

ABSTRACT

The medium-sized cities of 50,000 to 1,000,000 inhabitants must manage the transport services of these population centers. Focusing on the problem from the point of view of citizens, knowing the state of public transport for better time management and services are crucial for daily life.

The flow of information generated by real-time knowledge of the state of the public transport service in the city is great. This contains the state or geographic position of the transport vehicles, number of people, number of people at stops, payments made, among other types of information. What makes handling large volumes of data so useful is that it provides answers to many questions that citizens as end users didn't even know they had, starting to capture as much data as possible with the idea of ​​being able to use these in the future to study and improve the matter from which they have been obtained. In other words, it provides a reference point.

Keywords: transport, time, data volume

**Introducción**

Los grandes volúmenes de datos básicamente son el centro de lo conocido como Ciencia de Datos, siendo el estudio focalizado en la manipulación de datos registrados, permitiendo realizar análisis de los mismos con herramientas estadísticas, consiguiendo explicaciones de la situación actual de un mercado, realizar proyecciones a futuro sobre los precios o una idea de lo que podría pasar al realizar una nueva decisión. Por esta razón diversas entidades por el mundo están empezando a utilizarla. [1]

El objetivo de la presente investigación es proponer varias contribuciones en este campo de investigación, con el objetivo de brindar una solución a la necesidad de la existencia de una aplicación web que le brinde al usuario la información en tiempo real de la circulación de transporte en una ciudad mediana [2], donde le flujo de información puede alcanzar alrededor de puede 400 megabytes por segundo alcanzando cifras cercanas a 1 terabyte diario, para su puesta en marcha en la ciudad de Santiago de Cuba. En primer lugar, se recogen los principales conceptos teóricos sobre esta tecnología. En segundo lugar, se analizará el aprendizaje de la programación empleando técnicas de los grandes volúmenes de datos en el lenguaje Python. Y en tercer lugar, diseñar la solución de las mismas con elementos de sostenibilidad.

**Problema de la investigación:** Brindar al usuario la información en tiempo real del transporte público en una ciudad de medianas proporciones es una de la metas hoy en día con la nueva tecnología ya que resolvería la aglomeración de personas en las paradas.

**Objeto de Estudio:** La gestión del transporte.

**Campo de Estudio:** Técnicas de BigData.

**Objetivo General:** Creación de una aplicación web, partiendo de la aprehensión de las técnicas de Big Data y su implementación, muestre al usuario mediante un mapa la información del transporte de la ciudad en tiempo real.

**Objetivo Específicos**

* Análisis del estado del arte.
* Selección de las herramientas y las tecnologías a emplear.
* Selección de metodología de desarrollo.
* Definición de las funcionalidades de la aplicación web.
* Desarrollo e implementación de la aplicación web.

**Hipótesis:**

Si se implementará una aplicación web que brinde la información del transporte público en tiempo real en las empresas que brindan este servicio se contribuiría a una mejor gestión del tiempo del usuario del servicio y disminuiría considerablemente el nivel de congestión de personas en las paradas de ómnibus.

**Capítulo 1: Marco Referencial**

En este capítulo se explican los principales aspectos teóricos, los conceptos básicos de las tecnologías y la caracterización de las herramientas utilizadas.

**1.1 Breve Historia de BigData**

La historia del [Big Data](https://www.dynamicgc.es/que-es-big-data/) no tiene una fecha concreta de creación, sino que, con el desarrollo y progreso de aplicaciones y tecnologías, ha ido dando forma y se ha ido adaptando hasta lo que hoy en día conocemos. En el año 1989 **Erik Larson** utiliza por primera vez el término Big Data, en un artículo sobre el marketing y cómo se usarán los datos de los clientes, en los términos que actualmente conocemos.

Comienza a tener cierta trascendencia social, en los años 90, con el nacimiento del primer navegador web WWW (World Wide Web) y con la apertura de internet, en la que se comenzó a crear una red de conexión de computadoras en el mundo donde cualquier persona puede subir datos. Esto supuso la primera generación de datos masivos.

En 1997 Google lanza su motor de búsqueda y es el más usado, este acontecimiento favoreció a la creación de datos masivos. En la década de los años 2000 surgen empresas que generan y almacenan de forma segura grandes cantidades de datos concretamente para dar facilidades a las empresas. Con el nacimiento de la Web 2.0 o Web Social en 2005 se creaban un conjunto de páginas donde se pueden compartir la información, promoviendo así la colaboración y la interacción entre los usuarios, suponiendo una evolución trascendental en el uso de internet. En 2012 el Big Data se usa por primera vez en política para la [campaña de Barack Obama](https://www.researchgate.net/publication/295624658_How_President_Obama's_campaign_used_big_data_to_rally_individual_voters), conociendo las opiniones de los votantes. En los años siguientes el número de expertos que analizan y gestionan los datos **crece exponencialmente** en el mercado laboral, siendo de los empleos más demandados en la actualidad por las empresas. [3]

* 1. **Lenguaje de Programación, Herramientas y tecnologías de desarrollo**

**Python** es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML). Los desarrolladores utilizan Python porque es eficiente y fácil de aprender, además de que se puede ejecutar en muchas plataformas diferentes. El software Python se puede descargar gratis, se integra bien a todos los tipos de sistemas y aumenta la velocidad del desarrollo.

Python permite que los desarrolladores sean más productivos, ya que pueden escribir un programa de Python con menos líneas de código en comparación con muchos otros lenguajes. Cuenta con una gran biblioteca estándar que contiene códigos reutilizables para casi cualquier tarea. De esta manera, los desarrolladores no tienen que escribir el código desde cero. Hay muchos recursos útiles disponibles en Internet si desea aprender Python. Por ejemplo, puede encontrar con facilidad videos, tutoriales, documentación y guías para desarrolladores. [4]

**TypeScript** se define como una especie de superaste de JavaScript, cuyo resultado final es un código de JavaScript.

Lo que hace es encapsular varios elementos [5]:

* Una serie de funcionalidades de JavaScript 5, que es considerada hoy en día como un estándar y es la que todos los navegadores comprenden, incluso Node.js a nivel de servidor lo comprende.
* Como JavaScript ha seguido creciendo y apareció ECMAScript 6 (ES6), que añade nuevas funcionalidades, y que también está englobado dentro de TypeScript.
* Añade el uso un tipado muy estricto, en lugar del tipado dinámico de JavaScript, para intentar solucionar una serie de problemas. Aunque estos tipados no son requeridos, es recomendable utilizarlos para tener un código mucho más limpio.
* Añade la posibilidad de poder utilizar interfaces, para poder definir nuestros propios tipos o intentar aplicar programación orientada objetos luego.
* También añade la funcionalidad de genérica, que permite poder definir funciones que sean reutilizables, independientemente del tipo de datos que vayamos a tratar.

**Anaconda Navigator** es una interfaz gráfica de usuario (GUI) de escritorio incluida en la distribución Anaconda que le permite iniciar aplicaciones y administrar fácilmente paquetes, entornos y canales conda sin usar comandos. Navigator puede buscar paquetes en Anaconda.org o en un repositorio local de Anaconda. Está disponible para Windows, macOS y Linux.

Conda es tanto un administrador de paquetes como un administrador de entorno. Esto ayuda a los usuarios a asegurarse de que cada versión de cada paquete tenga todas las dependencias que requiere y funcione correctamente. Navigator es una forma fácil de trabajar con paquetes y entornos sin necesidad de escribir comandos en un terminal. Puede usarse para encontrar los paquetes que desee, instalarlos en un entorno, ejecutar los paquetes y actualizarlos, todo dentro del Anaconda Navigator [6].

**Visual Studio Code (VS Code)** 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. [7]

**Apache Spark** es un framework de programación para procesamiento de datos distribuidos diseñado para ser rápido y de propósito general. Como su propio nombre indica, ha sido desarrollada en el marco del proyecto Apache, lo que garantiza su licencia Open Source. Además, podremos contar con que su mantenimiento y evolución se llevarán a cabo por grupos de trabajo de gran prestigio, y existirá una gran flexibilidad e interconexión con otros módulos de Apache como Hadoop, Hive o Kafka. Parte de la esencia de Spark es su carácter generalista. Consta de diferentes APIs y módulos que permiten que sea utilizado por una gran variedad de profesionales en todas las etapas del ciclo de vida del dato. Dichas etapas pueden incluir desde soporte para análisis interactivo de datos con SQL a la creación de complejos pipelines de machine learning y procesamiento en streaming, todo usando el mismo motor de procesamiento y las mismas APIs [8].

**Angular** es un framework JavaScript potente, muy adecuado para el desarrollo de aplicaciones frontend modenas, de complejidad media o elevada. El tipo de aplicación JavaScript que se desarrolla con Angular es del estilo [SPA](https://desarrolloweb.com/articulos/que-es-una-spa.html) (Single Page Application) o también las denominadas [PWA](https://desarrolloweb.com/articulos/que-son-progressive-apps.html) (Progressive Web App). El framework Angular ofrece una base para el desarrollo de aplicaciones robustas, escalables y optimizadas, que promueve además las mejores prácticas y un estilo de codificación homogéneo y de gran modularidad. El desarrollo en Angular se hace por medio de [TypeScript](https://desarrolloweb.com/home/typescript), un superset del lenguaje JavaScript que ofrece muchas herramientas adicionales al lenguaje, como el tipado estático o los decoradores. [9]

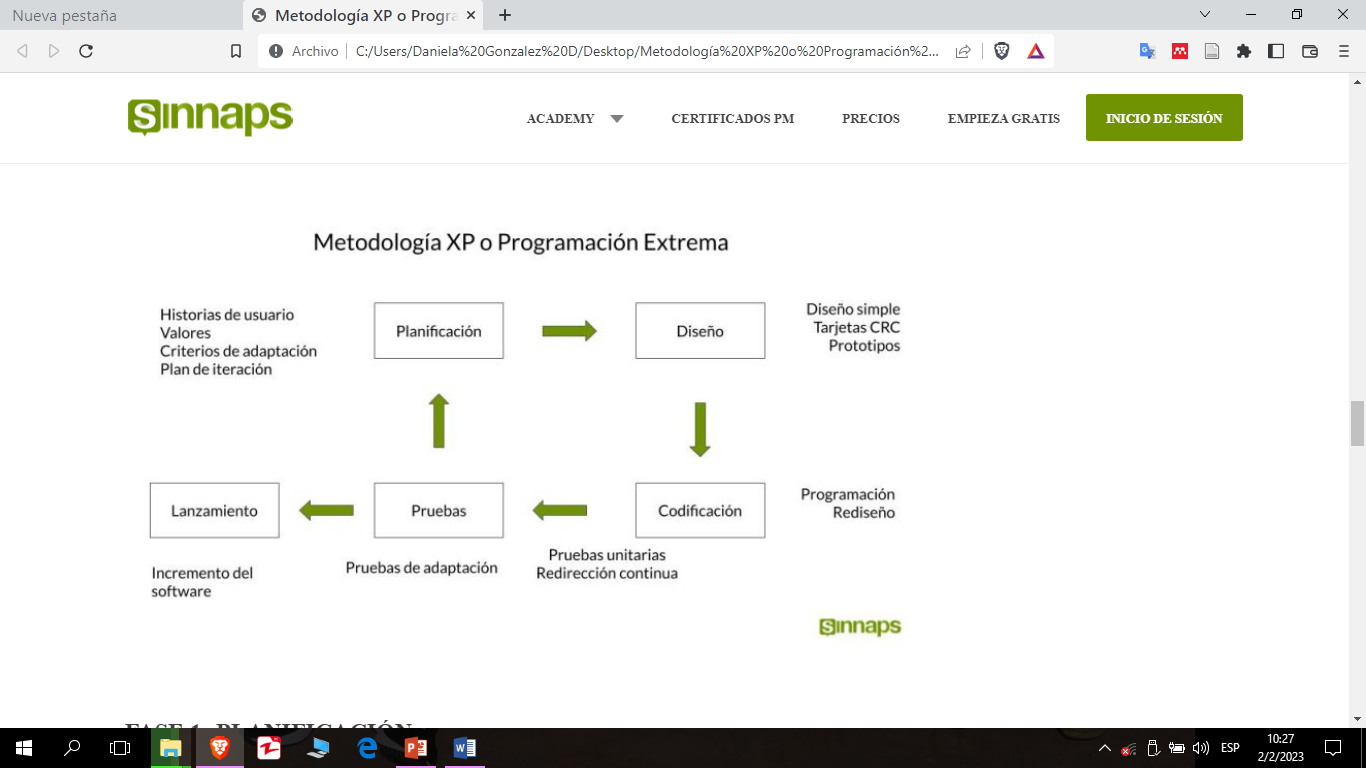
**Flask Python** es un sencillo marco de trabajo web, idóneo tanto para principiantes como para profesionales. Flask incluye únicamente lo esencial, pero los usuarios pueden implementar bibliotecas externas para ampliar sus funcionalidades. Mientras que los marcos de trabajo como Django proporcionan sus propias bibliotecas a los desarrolladores, Flask Python adopta un enfoque diferente. El marco de trabajo web lanzado por el desarrollador austriaco Armin Ronacher en 2010, es más minimalista. Flask solo incluye el motor de plantillas Jinja y una biblioteca llamada “tool”. Sin embargo, ofrece la posibilidad de integrar funciones de terceros. El marco de trabajo Flask está bajo una licencia BSD. Es gratuito y de código abierto. [10]

**1.3 Metodología de desarrollo de software**

**Programación Extrema (XP)**

La metodología XP es un conjunto de técnicas que dan agilidad y flexibilidad en la gestión de proyectos. También es conocida como Programación Extrema (Extreme Programming) y se centra crear un producto según los requisitos exactos del cliente. De ahí, que le involucre al máximo durante el método de gestión del desarrollo del producto.

La primera vez que oímos este tipo de metodología fue a través del libro *Extreme Programming Explained: Embrace Change* (1999), escrito por el ingeniero de software Kent Beck. El uso de esta metodología supone, para muchos teóricos, una aproximación a la calidad óptima del producto. Pues durante el ciclo de vida del software, ocurren cambios naturales. Es más, cuanto más cambios, puede que más cerca estemos del mejor resultado que espera nuestro cliente. Por eso, este cambio constante en el proyecto se llega a considerar como favorable. Y si podemos aplicar una manera dinámica de gestionarlos, mejor. Esta forma es conocida como metodología XP.



## Imagen 1: Metodología XP: Fases

### Fase 1: Planificación

Según la identificación de las historias de usuario, se priorizan y se descomponen en mini-versiones. La planificación se va a ir revisando. Cada dos semanas aproximadamente de iteración, se debe obtener un software útil, funcional, listo para probar y lanzar.

### Fase 2: Diseño

En este paso se intentará trabajar con un código sencillo, haciendo lo mínimo imprescindible para que funcione. Se obtendrá el prototipo. Además, para el diseño del software orientado a objetos, se crearán tarjetas CRC (Clase-Responsabilidad-Colaboración).

### Fase 3: Codificación «de Todos»

La programación aquí se hace «a dos manos», en parejas en frente del mismo ordenador. Incluso, a veces se intercambian las parejas. De esta forma, nos aseguramos que se realice un código más universal, con el que cualquier otro programador podría trabajar y entender.  Y es que deber parecer que ha sido realizado por una única persona. Así se conseguirá una programación organizada y planificada.

### Fase 4: Pruebas

Se deben realizar pruebas automáticas continuamente. Al tratarse normalmente de proyectos a corto plazo, este testeo automatizado y constante es clave. Además, el propio cliente puede hacer pruebas, proponer nuevas pruebas e ir validando las mini-versiones.

### Fase 5: Lanzamiento

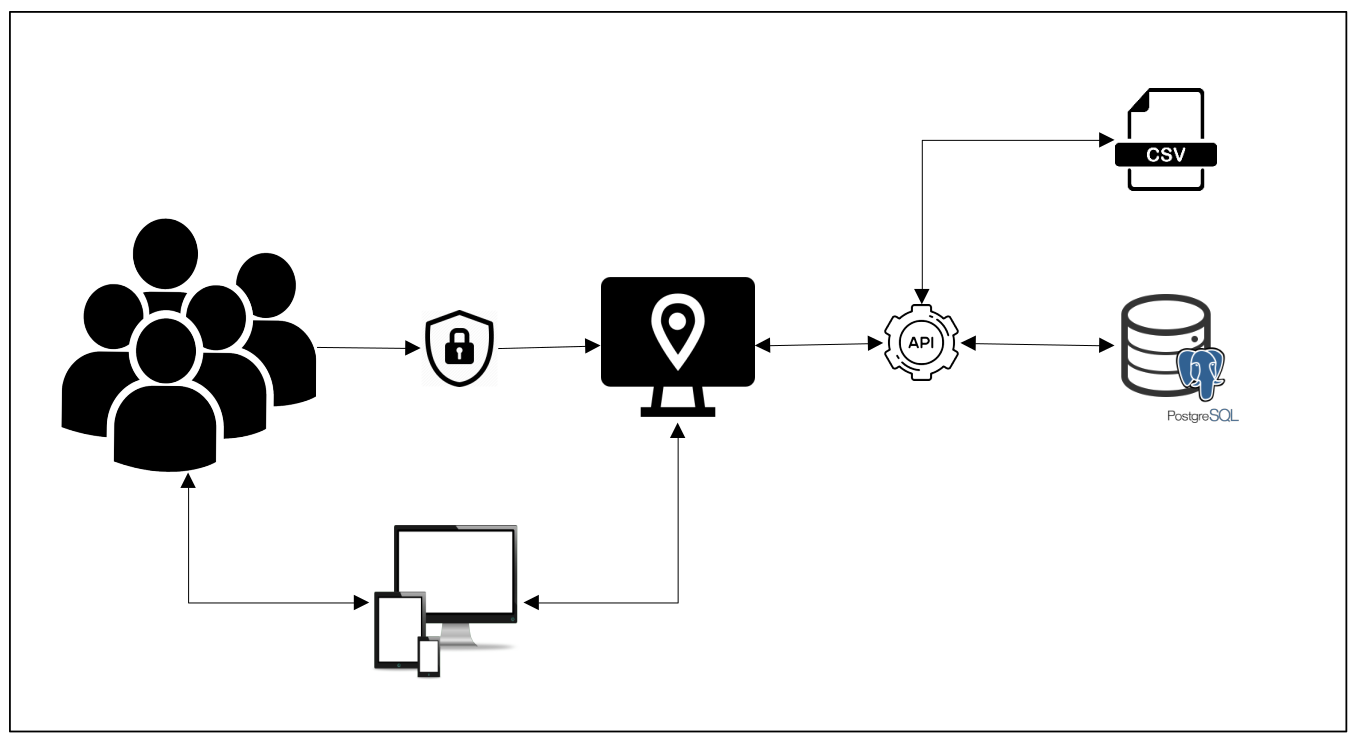
Si hemos llegado a este punto, significa que hemos probado todas las historias de usuario o mini-versiones con éxito, ajustándonos a los requerimientos del cliente. Tenemos un software útil y podemos incorporarlo en el producto. [11]

# **Capítulo 2. Planificación y Diseño**

En este capítulo se documentan los aspectos más relevantes que fueron identificados durante el estudio del contexto del software y el sistema a partir de las necesidades encontradas. Se detallan los requerimientos del sistema mediante las Historias de Usuarios, así como aspectos relacionados con el diseño del *software*, tales como: modelo de clases, diagramas físicos de la base de datos y la arquitectura.

## **2.1 Propuesta del Sistema**

Con el objetivo de dar solución al problema planteado anteriormente se propone desarrollar un sistema para la “Simulación del tránsito de una ciudad usando técnicas de BigData” que muestre al usuario mediante un mapa la información del transporte de la ciudad en tiempo real y de esta forma se resolvería la aglomeración de personas en las paradas.



**2.2** **Roles y responsabilidades**

Para llevar a cabo las diferentes tareas durante el desarrollo del software, es necesario definir los roles y las responsabilidades que le corresponde a cada miembro del equipo de trabajo. La tabla 2.1 muestra los roles y las responsabilidades adquiridas durante todo el proceso.

**Tabla 2.1:** Roles y responsabilidades

|  |  |  |  |
| --- | --- | --- | --- |
| **Roles** | **Nombre y Apellidos** | **Estatus** | **Descripción** |
| **Jefe de proyecto** | Dionis López Ramos | Doctor en Ciencias | Es el encargado de dirigir todo el proceso vinculado al desarrollo e implementación del sistema. |
| **Cliente** | Ministerio de Transporte Provincial | Entidad | Determina qué construir y brinda la información para la confección de los requisitos del negocio. |
| **Programador** | Fidel Alejandro Cepero Salazar | Estudiante | Responsable de las decisiones técnicas, construcción del sistema, diseño y realización de las pruebas. |
| **Analista** | Fidel Alejandro Cepero Salazar | Estudiante | Reúne toda la información referente al problema y especifica sus requisitos en cada fase de desarrollo. |
| **Diseñador** | Fidel Alejandro Cepero Salazar | Estudiante | Se encarga de todo el diseño del software en correspondencia con los requisitos del cliente. |
| **Arquitecto** | Fidel Alejandro Cepero Salazar | Estudiante | Define las partes integrantes del sistema |
| **Probador** | Fidel Alejandro Cepero Salazar | Estudiante | Ejecuta las pruebas y difunde los resultados de las mismas. |

## **2.3 Usuarios del Sistema**

Los usuarios son todas aquellas personas que interactúan de alguna forma con esta herramienta o que desempeñan algún rol específico. A continuación, se describen las actividades que realiza cada rol de usuario.

**Tabla 2.2 Usuarios del Software**

|  |  |
| --- | --- |
| **Usuario** | **Descripción** |
| Administrador | Acceso a funciones internas del sistema, base de datos y procesos en general del sistema. |
| Cliente | Acceso a realizar las operaciones comunes para lo que fue diseñada la herramienta, dichas operaciones se realizan desde la interfaz gráfica de la misma. |
| Funcionario | Modifica y Crea rutas en la Base de Datos |

## **2.3 Planificación del desarrollo del Sistema**

La planificación es la etapa inicial de todo proyecto. Es aquí donde se comienza a interactuar con el cliente para descubrir los requerimientos del sistema y realizar los ajustes a la metodología según las características del software. Se definen los requisitos funcionales y no funcionales.

### **2.3.1. Funcionalidades**

**F1:**  Crear Usuario: Para crear un usuario con datos válidos para poder interactuar con las funcionalidades de la página web.

Entrada: Correo electrónico y Contraseña

Salida: Redirección a la vista de logueo

**F2:** Logueo del Sistema: Para registrarse en la página web dependiendo a los roles del sistema.

Entrada: Datos de Usuario.

Salida: Redirección a la vista principal atendiendo a los roles de usuario.

**F3:** Crear ruta por funcionario: A través de un mapa el usuario traza la ruta y coloca marcadores en la posición en que estarán las paradas de la ruta y se almacenarán en la base de datos

Entrada: Coordenadas de la ruta y las paradas, Nombre de la ruta, Lugar de inicio y lugar donde termina la ruta.

Salida: Mensaje de éxito o error

**F4:** Modificar ruta por funcionario: El funcionario cargará la ruta que desea modificar y trazará en el mapa la nueva ruta a seguir con las respectivas paradas y podra modificar los valores correspondientes al nombre, lugar de inicio y lugar donde termina la ruta y se modificarán en la base de datos

Entrada: Coordenadas de la ruta y las paradas, Nombre de la ruta, Lugar de inicio y lugar donde termina la ruta.

Salida: Mensaje de éxito o error

**F5:** Eliminar ruta por funcionario: EL funcionario seleccionará que ruta desea eliminar y tras confirmar su selección se eliminara de la base de datos toda la informacion perteneciente a la ruta seleccionada

Entrada: Nombre de la ruta

Salida: Mensaje de éxito o error

**F6:** Visualizar rutas en el mapa: Se verán todas las rutas con sus paradas y desplazamiento de los ómnibus en el mapaen tiempo real

Salida: Mapa con todas las rutas trazadas y sus respectivas paradas y un marcador sobre el ómnibus que recorre cada ruta que se despazará a medida que lo haga el ómnibus

**F7:** Visualización de información de las rutas: Se visualizará el nombre de la ruta, lugar de inicio, lugar donde termina y chapa del ómnibus que realiza la ruta, en una tabla

Salida: Tabla con la información

**F8:** Carga de la información en la base de datos: Se carga los datos en tiempo real almacenados en un archivo en formato CSV hacia la base de datos

Entrada: Archivo en formato CSV

Salida: Información en la base de datos

**2.3.2. Características para las funcionalidades**

**Requisitos de Software**

* Manjaro Linux KDE 21.0.3 o Windows 10
* NodeJS 18.17.0

**Requisitos de Hardware**

Los requerimientos mínimos de hardware para correr la aplicación:

* Procesador Intel Corei7 a 4.0 GHz
* 8.0 GB de memoria RAM
* Conexion estable a Internet

### **2.3.3. Historias técnicas**

Las funcionalidades son complementadas por las historias técnicas que se enfocan en el diseño o la implementación. Las historias técnicas constituyen las propiedades o cualidades que el producto debe tener.

**Usabilidad**

La aplicación web debe brindar al usuario una clara navegabilidad de las funcionalidades del sistema.

**Confiabilidad**

La aplicación web debe ser capaz de funcionar correctamente cada vez que se utilice, las funciones para las que fue diseñado deben arrojar los resultados que el usuario espera, dichos resultados deben ser íntegros.

**Seguridad**

El espacio de trabajo de cada usuario será protegido mediante autenticación con usuario y contraseña, y con el empleo de tokens generados a partir de la informacion de usuario valida . De esta forma se garantiza la seguridad e integridad de la información del usuario.

### **2.3.4. Historias de Usuario**

Las Historias de Usuarios (HU) son la técnica utilizada en XP para especificar los requisitos de software, tanto funcionales como no funcionales y se dividen en tareas de ingeniería asignadas a los programadores para ser implementadas durante las iteraciones. Están escritas con el vocabulario del cliente, no con vocabulario técnico. El procedimiento de las HU es muy dinámico y flexible, en cualquier momento las HU pueden romperse, emplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas.

La prioridad de las Historias de Usuario se clasificó de la siguiente manera:

* Alto: requerimientos que no pueden faltar.
* Medio: requerimientos que pueden ser implementados en otra iteración.
* Bajo: requerimientos adicionales.
* HU.1 Crear Usuario
* HU.2 Logueo en el sistema
* HU.3 Crear ruta por funcionario
* HU.4 Modificar ruta por funcionario
* HU.5 Eliminar ruta por funcionario
* HU.6 Visualización de rutas en el mapa
* HU.7 Visualización de información de las rutas

**Historias de Usuario**

**Tabla 2.3 Historia de Usuario: Crear Usuario**

|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número:1** | **Nombre de HU: Crear Usuario** |
| **Usuario:** Cualquier usuario | |
| **Prioridad en negocio:** Alta | **Riesgo de desarrollo:** Bajo |
| **Descripción:** Donde el usuario crea una cuenta para acceder a la plataforma | |
|  | |

**Tabla 2.3 Historia de Usuario: Logueo en el sistema**

|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número:2** | **Nombre de HU: Logueo en el sistema** |
| **Usuario:** Cualquier usuario | |
| **Prioridad en negocio:** Alta | **Riesgo de desarrollo:** Bajo |
| **Descripción:** Donde el usuario inicia sesión para acceder a la plataforma | |
|  | |

**Tabla 2.3 Historia de Usuario: Crear ruta por funcionario**

|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número:3** | **Nombre de HU: Crear ruta por funcionario** |
| **Usuario:** Cualquier usuario | |
| **Prioridad en negocio:** Alta | **Riesgo de desarrollo:** Alto |
| **Descripción:** Donde el funcionario crea las rutas | |
|  | |

**Tabla 2.3 Historia de Usuario: Modificar ruta por funcionario**

|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número:4** | **Nombre de HU: Modificar ruta por funcionario** |
| **Usuario:** Funcionario | |
| **Prioridad en negocio:** Alta | **Riesgo de desarrollo:** Alto |
| **Descripción:** Donde el funcionario podrá modificar las rutas | |
|  | |

**Tabla 2.3 Historia de Usuario: Eliminar ruta por funcionario**

|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número:1** | **Nombre de HU: Iniciar sesión** |
| **Usuario:** Cualquier usuario | |
| **Prioridad en negocio:** Alta | **Riesgo de desarrollo:** Bajo |
| **Descripción:** Donde el usuario inicia sesión para acceder a la plataforma | |
|  | |

**Tabla 2.3 Historia de Usuario: Visualización de rutas en el mapa**

|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número:1** | **Nombre de HU: Iniciar sesión** |
| **Usuario:** Cualquier usuario | |
| **Prioridad en negocio:** Alta | **Riesgo de desarrollo:** Bajo |
| **Descripción:** Donde el usuario inicia sesión para acceder a la plataforma | |
|  | |

**Tabla 2.3 Historia de Usuario: Visualización de información de las rutas**

|  |  |
| --- | --- |
| **Historia de usuario** | |
| **Número:1** | **Nombre de HU: Iniciar sesión** |
| **Usuario:** Cualquier usuario | |
| **Prioridad en negocio:** Alta | **Riesgo de desarrollo:** Bajo |
| **Descripción:** Donde el usuario inicia sesión para acceder a la plataforma | |
|  | |

**Conclusiones del capítulo**

En el presente capítulo se mostraron aspectos relacionados con las fases de planificación y diseño del desarrollo del software. Quedaron expuestas las funcionalidades a implementar definidas por los artefactos que propone la metodología XP, se mencionan los requisitos tanto funcionales como los no funcionales, las historias de usuarios y las historias técnicas.

# **Capítulo 3. Implementación y Pruebas del Sistema**

En este capítulo se presentarán algunas de las funcionalidades del sistema diseñado y detalles de su implementación, explicando las herramientas de software utilizadas y aspectos relevantes que han tenido lugar en su elaboración. También se realizan las pruebas pertinentes a algunas de las funcionalidades del sistema, evaluando de esta manera el software y su correcto funcionamiento.

## **Estructura del proyecto**

Antes de empezar a programar la aplicación, fue necesario adaptar todas las funcionalidades y esquemas a la filosofía de trabajo de Anaconda y Flask lo cual aseguró el desarrollo e implementación del sistema lo más eficientemente. Truffle es un marco de trabajo para el desarrollo de aplicaciones descentralizadas y contratos inteligentes en la cadena de bloques. A continuación, se describen algunos de los elementos principales de Truffle:

* **Anaconda** facilita la gestión de entornos de Python y la instalación de paquetes. Además, Anaconda incluye una amplia gama de bibliotecas y herramientas preinstaladas que son útiles para la ciencia de datos y el análisis. También permite crear entornos virtuales aislados para proyectos específicos, lo que facilita la administración de dependencias y versiones de paquetes.[12]
* **Flask Framework** es un microframework web de Python que se utiliza para crear aplicaciones web simples y rápidas. Es minimalista y flexible, lo que significa que proporciona solo lo esencial para crear aplicaciones web, permitiendo a los desarrolladores elegir las herramientas y bibliotecas que desean utilizar. Flask es conocido por su simplicidad y facilidad de uso, lo que lo hace ideal para proyectos web pequeños y medianos, así como para crear APIs (interfaces de programación de aplicaciones) web.[13]

La aplicación BusVisor (Visor de Omnibus) se desarrolla para la creación administracion y visualizacion de las rutas del transporte publico de una ciudad, resolviendo el problema de la s largas esperas x las pernosas en la parada y el desconicimento de si las rutas están en funcionamiento. La misma está compuesta por un conjunto de directorios, dentro de los que se destacan:

* El directorio **“src/api”:** está el código de la API-REST.
* El directorio **“src/app”:** está el código de los componentes de la aplicación .

## **Instalación de las herramientas utilizadas**

Manjaro Linux:

* Instalación de NodeJS en el LTS versión 18.13.0 que es uno de los últimos, para esto uso nvm:

1. En primer medida debemos de actualizar los repositorios de Manjaro para esto ejecutamos:

* sudo pacman –Syy

1. Si el comando anterior no presenta errores saltar este paso siguiente:

* sudo pacman –Syyu

1. Después de tener actualizado los repositorios de Manjaro se procede a la instalación nodejs para esto utilizamos el comando;

* sudo pacman -S nodejs npm

1. Luego de esto se descargara e instalara node.js para comprobar que quedo bien instalado puedes comprobar el comando:

* npm –version
* Instalación de Angular:
* En la terminal: npm install -g @angular/cli

Luego de esto se descargara e instalara node.js para comprobar que quedo bien instalado puedes comprobar el comando:

angular version

* Instalación Python:
* En primer medida debemos de actualizar los repositorios de Manjaro para esto ejecutamos:

sudo pacman –Syy

* Si el comando anterior no presenta errores saltar este paso siguiente:

sudo pacman –Syyu

* Después de tener actualizado los repositorios de Manjaro se procede a la instalación nodejs para esto utilizamos el comando;

sudo pacman -S python

* Luego de esto se descargara e instalara node.js para comprobar que quedo bien instalado puedes comprobar el comando:

python --version

* Instalación Anaconda:
* **Descargar Anaconda:**

**En el sitio web oficial de Anaconda (**<https://www.anaconda.com/products/distribution>**) y descargar la versión de Anaconda que corresponda con el sistema**

* **En la terminal:** 
  + **cd <ruta\_de\_la\_descarga>**
  + **chmod +x NombreDelArchivo**
* **Ejecuta el script de instalación:**

**./NombreDelArchivo**

* Luego de esto se descargara e instalara node.js para comprobar que quedo bien instalado puedes comprobar el comando:

conda --version

* Instalación opcional de GitHub Desktop: Ejecutamos Instalador, una vez instalado pudimos poner nuestra cuenta personal en GitHub y alojar nuestro proyecto allí para tenerlo salvado y controlar todos los cambios que se hicieron durante la implementación del mismo.

## **3.3 Implementación de Funcionalidades**

## **3.4 Análisis económico**

En la realización de un proyecto se hace necesaria la planificación y el control del esfuerzo, costo y tiempo que tomará llevarlo a cabo. Con la utilización de métodos de estimación de costos, se puede determinar una aproximación de los recursos necesarios, así como el total de tiempo que gastaría una persona o un equipo, en el desarrollo de un producto de software específico. Además, se determina la viabilidad económica, ambiental, técnica y de mercado. A continuación, se realiza un análisis de costos para el sistema.

### **3.4.1. Estimación de costo y tiempo**

Para determinar los costos de los sistemas desarrollados se usó el método de puntos en casos de uso [14]. Este es un método de estimación prometedor que se adapta bien al enfoque de caso de uso para la descripción de los requisitos. En sus bases yace el concepto de transacción de caso de uso, la unidad más pequeña de medición. Se realizó este análisis teniendo en cuenta las Historias de Usuario proporcionadas por la metodología XP.

El método de punto de casos de uso consta de cuatro etapas, en las que se desarrollan los siguientes cálculos:

**Ecuación 3.1**: Cálculo de los Puntos de Historias de Usuarios sin ajustar

**UUCP = UAW + UUCW**

Donde:

* **UUCP**: Puntos de Historias de Usuarios sin ajustar.
* **UAW**: Factor de Peso de los Actores sin ajustar.
* **UUCW**: Factor de Peso de Historias de Usuarios sin ajustar.

**Factor de Peso de los Actores sin ajustar (UAW)**

Este valor se calcula mediante un análisis de la cantidad de Actores presentes en el sistema y la complejidad de cada uno de ellos. En la [tabla 3.1](#_Tabla_3.1_Factor) se presenta el Factor de Peso de los Actores sin ajustar.

#### Tabla 3.1 Factor de peso de los actores sin ajustar.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipo** | **Descripción** | **Peso** | **Cant\*peso** |
| Simple | Otro sistema que interactúa mediante una interfaz de programación de aplicaciones. (API) | 1 | 1\*1 |
| Medio | Otro sistema que interactúa mediante un protocolo o una persona interactuando con una interfaz basada en texto. | 2 | 0\*2 |
| Complejo | Una persona que interactúa con el sistema mediante una interfaz gráfica. (GUI) | 3 | 1\*3 |
| Total |  |  | 1+0+3= **4** |

**Para calcular UUCW**

Este valor se calcula mediante un análisis de la cantidad de HU presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los HU se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo según muestra la [tabla 3.2.](#_Tabla_3.2_Peso)

#### Tabla 3.2 Peso de las Historias de Usuarios sin Ajustar

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo | Descripción | Peso | Cant \* peso |
| Simple | La HU contiene de 1 a 3 transacciones. | 5 | 5\*5 |
| Medio | La HU contiene de 4 a 7 transacciones. | 10 | 2\*10 |
| Complejo | La HU contiene más de 8 transacciones. | 15 | 0\*15 |
| Total |  |  | 25+20+0=**45** |

**Cálculo de los Puntos de Historias de Usuarios ajustadas.**

Una vez que se tienen los Puntos de Historias de Usuarios, se debe ajustar este valor como se muestra en la ecuación 3.2.

**Ecuación 3.2**: Cálculo de los Puntos de Historias de Usuarios ajustadas

**UCP = UUCP \* TCF \* EF**

Donde:

* **UCP**: Puntos de Historias de Usuarios ajustados.
* **UUCP**: Puntos de Historias de Usuarios sin ajustar.
* **TCF**: Factor de complejidad técnica.
* **EF**: Factor de ambiente.

**Factor de complejidad técnica** (**TCF**).

Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5, donde 0 significa un aporte irrelevante o nulo y 5 un aporte muy importante. En la siguiente [tabla 3.3](#_Tabla_3.3_Factor) se muestra factor de complejidad técnica con su significado y el peso de cada uno de estos factores:

#### Tabla 3.3 Factor de complejidad técnica

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Factor | Descripción | **Peso** | **Valor** | **(Peso-i \* Valor-i)** |
| T1 | Sistema distribuido | 2 | 1 | 2 |
| T2 | Rendimiento o tiempo de respuesta | 1 | 5 | 5 |
| T3 | Eficiencia del usuario final | 1 | 2 | 2 |
| T4 | Procesamiento interno complejo | 1 | 5 | 5 |
| T5 | El código debe ser reutilizable | 1 | 5 | 5 |
| T6 | Facilidad de instalación | 0.5 | 2 | 1 |
| T7 | Facilidad de uso | 0.5 | 5 | 2.5 |
| T8 | Portabilidad | 2 | 5 | 10 |
| T9 | Facilidad de cambio | 1 | 2 | 2 |
| T10 | Concurrencia | 1 | 4 | 4 |
| T11 | Incluye objetivos especiales de seguridad | 1 | 5 | 5 |
| T12 | Acceso directo a terceras partes | 1 | 2 | 2 |
| T13 | Se requieren facilidades especiales de entrenamiento a los usuarios | 1 | 1 | 1 |
| Total |  |  |  | **46.5** |

**Para Calcular TCF:** Factor de complejidad técnica se muestra la ecuación 3.3 cálculo del factor complejidad técnica.

**Ecuación 3.3:** Cálculo del Factor de complejidad técnica

**TCF = 0.6 + 0.01 \* Σ (Pesoi\* Valori)**

**TCF = 0.6 + 0.01 \*46.5**

**TCF = 1.06**

**Factor Ambiente (EF).**

El factor de ambiente está relacionado con las habilidades y entrenamiento del grupo de desarrollo. Cada factor se cuantifica con un valor desde 0 (aporte irrelevante) hasta 5 (aporte muy relevante).

En la siguiente [tabla 3.4](#_Tabla_3.4_Factor) se muestra factor de ambiente con su significado y el peso de cada uno de estos factores:

#### Tabla 3.4 Factor de ambiente

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Factor** | **Descripción** | **Peso** | **Valor** | **(Peso-i \* Valor-i)** |
| E1 | Familiaridad con el modelo de proyecto utilizado | 1.5 | 3.5 | 5.25 |
| E2 | Experiencia en la aplicación | 0.5 | 1 | 0.5 |
| E3 | Experiencia en orientación a objetos | 1 | 3 | 3 |
| E4 | Capacidad del analista líder | 0.5 | 3.5 | 1.75 |
| E5 | Motivación | 1 | 4.5 | 4.5 |
| E6 | Estabilidad de los requerimientos | 2 | 4 | 8 |
| E7 | Personal Part-Time | -1 | 0 | 0 |
| E8 | Dificultad del lenguaje de programación | -1 | 4 | -4 |
| **Total** |  |  |  | **19** |

|  |  |
| --- | --- |
|  |  |

**Para Calcular EF**: Factor de ambiente se muestra la ecuación 3.4 cálculo del factor ambiente.

**Ecuación 3.4**: Cálculo del Factor de ambiente

**EF** = 1.4 - 0.03 \* Σ (Peso**i** \* Valor**i**)

**EF** = 1.4 - 0.03 \* **19**

**EF** = **0.83**

**Luego: UCP=UUCP \* TCF \* EF**

**UCP = 45 \* 1.06\* 0.83**

**UCP = 39.59**

**Estimación de esfuerzo a través de los Puntos de Historias de Usuarios**.

**Ecuación 3.5**: Esfuerzo estimado en horas hombres.

E = UCP \* CF

Donde:

**E:** Esfuerzo estimado en horas hombres.

**UCP**: Punto de historias de usuarios ajustadas.

**CF**: Factor de conversión.

Para obtener el factor de conversión (CF) se cuentan cuántos valores de los que afectan el factor ambiente (E1 a E6) están por debajo de la media (**<3**), y los que están por encima (**>3**) para los restantes (E7 a E8). Si el total (nos da **0**) es 2 o menos se utiliza el factor de conversión 20 Horas- Hombre / Punto de historias de usuarios. Si el total es 3 o 4 se utiliza el factor de conversión 28 Horas-Hombre / Punto de historias de usuarios. Si el total es mayor o igual que 5 se recomienda efectuar cambios en el proyecto ya que se considera que el riesgo de fracaso es demasiado alto. En este caso:

CF= 28 Horas-hombre / Puntos de historias de usuarios.

Luego

**E = 39.59 \* 28 horas-hombre**

**E = 1108.52 horas-hombre**

En la siguiente [tabla 3.5](#_Tabla_3.5_Distribución) se muestra distribución del esfuerzo por etapas.

#### Tabla 3.5 Distribución del esfuerzo por etapas

|  |  |  |
| --- | --- | --- |
| **Actividad** | **% esfuerzo** | **Valor esfuerzo** |
| Planificación | 10 | 110.852 |
| Diseño | 20 | 221.704 |
| Codificación | 40 | 443.408 |
| Prueba | 15 | 166.278 |
| Sobrecarga | 15 | 166.278 |
| Total | 100 | **1108.52** |

Una vez estimado el tiempo de desarrollo del proyecto y conociendo la cantidad de desarrolladores y el pago que recibe cada uno de estos se puede llevar a cabo una estimación del costo total del proyecto referidos a los recursos humanos; existen otros costos como por ejemplo del equipamiento que se suman al anterior.

**K**: Coeficiente que tiene en cuenta los costos indirectos (1,5 y 2,0).

**THP:** Tarifa Horaria Promedio. El salario promedio mensual de los trabajadores en este caso es de $12 000 CUP dividido entre 176h.

**176 horas** (horas de trabajo para 1 mes, esto se toma a razón de 24 días, ya que no se cuentan los fines de semana ni sábados cortos).

**Tiempo**= 1108.52 horas/176 ≈ equivalente a 7 meses, este es el tiempo que tomaría desarrollar el proyecto empleando una sola persona.

7 meses a razón de 24 días laborables por mes, representan 168 días. En nuestro caso como empleamos 1 trabajador para el desarrollo del proyecto, el tiempo para su culminación quedará reducido a 84 días aproximadamente.

**Entonces el costo total del proyecto:**

**C = E (Total) \* K \* THP**

**C** = 1108.52 \* 2.0 \* 12 000/176 = **$** **151 161.818**

Para el salario a los trabajadores se investigó como se mueve en diferentes organizaciones o sucursales en Santiago de Cuba de diferente sector ya sea estatal o privado, dada las nuevas regulaciones y tasas de cambio. Los datos se obtuvieron a través de trabajadores de las entidades y por anuncios laborales.

Tabla 3.6 Situación actual de pago.

|  |  |
| --- | --- |
| **Organización (Sector)** | **Pago/mensual (Moneda Nacional)** |
| XETID (Estatal) | 10 000 |
| DATYS (Estatal) | 9 600 |
| DESOFT (Estatal) | 10 000 |
| MYPIMES (Privado) | 20 000 |
| Freelancer o persona autónoma (Privado) | 200MLC a 220CUP (tasa oficial actual) son 44 000 |

Referencias Bibliográficas

1. Cripto tendencia. Septiembre 2019. El impacto del Big Data en el mundo. <https://criptotendencia.com/2019/09/21/el-impacto-del-big-data-en-elmundo/> citado [13-11-2022]
2. USLG. 2022. Ciudades intermedias. <https://www.uclg.org/es/agenda/ciudades-intermedias> citado [14-1-2023]
3. Dynamic. 2020. Historia de BigData. <https://www.dynamicgc.es/historia-del-big-data/>
4. Amazon Web Service. 2023. ¿Qué es Python? <https://aws.amazon.com/es/what-is/python/> citado [15-1-2023]
5. Cabrera Barrios, José. 17 diciembre 2020. Que es TypeScript. <https://openwebinars.net/blog/que-es-typescript/> citado[13-1-2023]
6. Rondon, Izary. 4 febrero 2022 ¿Que es Anaconda? <https://eiposgrados.com/blog-python/que-es-anaconda/> citado[13-1-2023]
7. Flores, Franquear. 22 julio 2022. Que es Visual Studio Code y que ventajas ofrece. <https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/> citado[13-1-2023]
8. ESIC. Octubre 2018. Apache Spark: Introducción, que es y cómo funciona. <https://www.esic.edu/rethink/tecnologia/apache-spark-introduccion-que-es-y-como-funciona> citado[13-1-2023]
9. Deymar, A. 11 enero 2023. ¿Qué es Angular y cuáles son sus ventajas? <https://www.hostinger.es/tutoriales/que-es-angular> citado[13-1-2023]
10. Digital Guide IONOS. 2 noviembre 2022. ¿Qué es Flask? Un breve tutorial sobre este microframework. <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/flask/> citado[13-1-2023]
11. Sinnaps. 2020. Metodología XP o Programación Extrema. <https://www.sinnaps.com/> citado [2-2-2023]
12. Anaconda, 2018, Documentacion Oficial, <https://docs.anaconda.com/> citado[10-8-2023]
13. Flask, 2023, Documentación Oficial, <https://flask.palletsprojects.com/en/2.0.x/> citado[10-8-2023]
14. Puntos de caso de uso. (2021). En *Wikipedia, la enciclopedia libre*. <https://es.wikipedia.org/w/index.php?title=Puntos_de_caso_de_uso&oldid=133055102> citado[21-06-2023]