**UNIVERSIDAD MARIANO GÁLVEZ.**

**Ingeniería en Sistemas de Información y Ciencias de la Computación.**

**Ingeniería de Proyectos.**

**Ing. Ivan de León**

**Onceavo Semestre.**

****

**“AutoParking”**

**San Pedro Sacatepéquez, San Marcos, Mayo de 2018.**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **NOMBRE COMPLETO** | **No. CARNÉ** | **Cargo** |
| 1 | Eddy Estuardo Agustín Marroquín | 0903-13-13796 | Director del Proyecto |
| 3 | Erwin Jonathan Navarro Fuentes | 0903-09-5855 | Developer |
| 4 | Julio Noe de Leon Soc | 0903-13-12163 | Developer |
| 5 | Alma Maria Cordón | 0903-13-8610 | Product Owner |

**1. TÍTULO DEL PROYECTO:**

“**Auto Parking”**

**2. TIPO DE PROYECTO:**

Proyecto de Tecnología aplicada que tiene como fin integrar el internet con el entorno, a fin de mejorar situaciones cotidianas, implementando tecnología y el internet de las cosas para desarrollar un sistema que ayude a la integración compleja del entorno con lo virtual

**3. Requerimientos del Proyecto**

* Se busca simplificar y facilitar la ubicación de un parqueo para vehículos en un entorno en el cual siempre esta congestionado y evitar las pérdidas de tiempo en búsquedas fallidas.
* Ayudando a un negocio que brinde servicios de parqueo y aumentando su productividad
* Simplificando y facilitando la búsqueda de un parqueo disponible para un conductor.

**4. DESCRIPCIÓN DEL PROYECTO:**

AutoParking es una aplicación de tecnologías integradas que hace uso del modelo colaborativo para su funcionamiento, el cual será una aplicación desarrollada para plataformas androide integrada con tecnologías como API’s desarrolladas para la aplicación integrando apis ya existente

AutoParking funcionara como una red Social Vehicular el cual ayudará al conductor que busca parqueo para su vehículo y al Parqueo que tiene espacios en su estacionamiento Optimizando y reduciendo búsquedas fallidas tanto del conductor como espacios vacíos con el parqueista, el sistema funcionara de manera colaborativa y con gestión y actualización de datos en tiempo real , en la que el conductor buscará estacionamiento disponible y más cercano y el parqueista brindara información de la cantidad de espacio disponible que tenga en cierto momento brindando también indicaciones para llegar al lugar.

**PROPUESTA GRUPO DE DESARROLLO API “AutoParking”**

# **1. Propuesta de desarrollo**

El proyecto consiste en desarrollar una API para el Proyecto Denominado como AutoParking que integre y gestione los datos desde una plataforma androide hasta un Servidor BD implementando las funciones y procedimientos sin la necesidad de programarlas de nuevo. Con posibilidad tanto de enviar información como recibirla, desarrollada por desarrolladores para desarrolladores.

## **b. Descripción del proyecto**

AutoParking es una aplicación de tecnologías integradas que hace uso del modelo colaborativo para su funcionamiento, el cual será una aplicación desarrollada para plataformas androide integrada con tecnologías como API’s desarrolladas para la aplicación integrando apis ya existente

AutoParking funcionara como una red Social Vehicular el cual ayudará al conductor que busca parqueo para su vehículo y al Parqueo que tiene espacios en su estacionamiento Optimizando y reduciendo búsquedas fallidas tanto del conductor como espacios vacíos con el parqueista, el sistema funcionara de manera colaborativa y con gestión y actualización de datos en tiempo real , en la que el conductor buscará estacionamiento disponible y más cercano y el parqueista brindara información de la cantidad de espacio disponible que tenga en cierto momento brindando también indicaciones para llegar al lugar.

El proyecto consiste en desarrollar una API para el Proyecto Denominado como AutoParking que integre y gestione los datos desde una plataforma androide hasta un Servidor BD implementando las funciones y procedimientos sin la necesidad de programarlas de nuevo. Con posibilidad tanto de enviar información como recibirla, desarrollada por desarrolladores para desarrolladores.

## **c. Arquitectura a utilizar y definir por qué se eligió esta arquitectura**

La Arquitectura a utilizar es la **Arquitectura orientada a la Web** ya que su objetivo es de acelerar la creación de aplicaciones móviles y web. El diseño de la tecnología, en el que se fundamenta para el desarrollo de nuestra API será pública ya que permite que desarrolladores terceros accedan a sus APIs, y así poder generar una gran base de usuarios.

El estilo Servicio web es una estrategia basada en operaciones y válida para diferentes transportes para el diseño de API, que utiliza Lenguaje de descripción de servicios web (WSDL) para describir interfaces. La gran cantidad de herramientas que existen para los servicios web también significa que las aplicaciones cliente a menudo se pueden desarrollar de manera rápida y simple.

**i. Métodos a utilizar**

El navegador emplea normalmente los métodos HTTP más usados por los desarrolladores, GET y POST. Para los que no saben, el método GET lo usamos cuando escribimos en la barra de direcciones del navegador un enlace válido, al presionar enter, el navegador envía una petición al servidor por medio del método GET. Por otro lado, cuando nos damos de alta en un sitio web o hacemos un pago a través de Internet, el navegador usará el método POST.  
  
Además de GET y POST, están también PUT, DELETE, HEAD y OPTIONS.

* **GET** Se usa para obtener información del servidor, puede ser algún archivo HTML, una imagen, un archivo de texto, un XML, etc. Este método solo debe usarse para obtener información del servidor de acuerdo a los estándares de HTTP. El método GET no debe cambiar el estado del servidor, es decir, no debe hacer ninguna modificación a cualquier archivo que en éste se encuentre. En términos de CRUD, GET sería el Read (Leer).
* **POST** Se podría decir que es el método HTTP más empleado, eso en parte a que permite hacer variedad de operaciones del CRUD, aunque por estándar, es el encargado de crear un nuevo recurso y, por consiguiente, modificar el estado del servidor. En términos de CRUD, el método POST se confunde en ocasiones con el método PUT, aunque la diferencia radica en una cualidad que se llama idempotencia.
* **PUT** No es tan usado como lo son GET y POST. Su uso podría darse en el caso de que, por ejemplo, quisieras hacer una operación que haga una acción sobre el estado del servidor, pero que la siguiente vez que suceda, no cambie ese estado. El ejemplo que da el autor del artículo que traduzco se basa en la operación de actualización.  
  Teniendo esto en cuenta, podemos relacionar el método PUT con la acción del CRUD, UPDATE (actualizar).
* **DELETE** No tiene mucho que explicar. Teniendo en cuenta los estándares de HTTP, el método DELETE es el único que debe ser usado para borrar un recurso del servidor, sin embargo, algunos desarrolladores emplean los métodos GET o POST para esa labor. En cuanto a CRUD, DELETE sería la acción, que lleva su mismo nombre, DELETE (borrar).
* **HEAD** Se usa para obtener la cabecera de respuesta que devuelve el servidor al hacer una petición sobre éste. Similar a GET, ambos no cambian el estado del servidor, aunque HEAD solo devuelve los metadatos. Se puede usar para saber si cierto recurso está en el servidor.
* **OPTIONS** Este método se usa para saber que otros métodos HTTP están disponibles, para determinado recurso, en el servidor. Por ejemplo, para saber si alguna imagen acepta los métodos GET o POST, haces la petición REST sobre ese recurso y en la cabecera de respuesta, obtendrás la respuesta Allow con los métodos que pueden usarse sobre dicho recurso.

**ii. Como se documentará la API**

La documentación será documentada por medio de la aplicación POSTMAN ya que en ella podemos generar toda la documentación y su sandbox correspondiente. Además, existen distintos módulos para enganchar a nuestro proyecto Node.js, Grails, Scala Play, Spring MVC, Symfony o Ruby, entre otros muchos.

**d. Herramientas a utilizar y definir por qué se eligió esta herramienta**

**Postman** es una herramienta que nos facilitara la creación de documentación por lo cual como grupo hemos decidido emplearla para documentar nuestra los diversos procesos de desarrollo de nuestra api.

**Google Drive**: Google Drive es un servicio de sincronización y almacenamiento de archivos desarrollado por Google. Google Drive permite a los usuarios almacenar archivos en sus servidores, sincronizar archivos entre dispositivos y compartir archivos. Además de un sitio web, Google Drive ofrece aplicaciones con capacidades sin conexión para computadoras con Windows y macOS, y teléfonos inteligentes y tabletas con Android e iOS. Google Drive incluye Google Docs, Sheets y Slides, una suite ofimática que permite la edición colaborativa de documentos, hojas de cálculo, presentaciones, dibujos, formularios y más. Los archivos creados y editados a través de la suite Office se guardan en Google Drive.

En el desarrollo de proyecto API, debido a las restricciones geográficas de todos los integrantes del proyecto decidimos implementar la suite de ofimática de google drive, ya que esta herramienta nos permite crear documentación interactiva, desde cualquier lugar en donde nos encontremos ya que lo único que necesitamos es una conexión a internet.

**Git**: Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos.

**Github**: GitHub es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git. Se la utiliza principalmente para la creación de código fuente de programas de computadora. El software que opera GitHub fue escrito en Ruby on Rails. Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. Anteriormente era conocida como Logical Awesome LLC. El código de los proyectos alojados en GitHub se almacena típicamente de forma pública, aunque utilizando una cuenta de pago, también permite hospedar repositorios privados.

En el desarrollo de proyecto API, hemos decidido utilizar las herramientas de git y github, para manejar las versiones y avances de nuestro proyecto, teniendo en cuenta las restricciones geográficas, estas herramientas nos facilitan la interacción de desarrollo entre desarrolladores y al mismo tiempo mantenemos una copia de seguridad de versiones anteriores del proyecto.

**Slack**: Es una herramienta de comunicación en equipo creada por Stewart Butterfield, Eric Costello, Cal Henderson, y Serguei Mourachov. Slack surge como una herramienta interna utilizada por la compañía Tiny Speck en el desarrollo de Glitch, un juego en línea actualmente obsoleto. Slack es lanzado al mercado en agosto del 2013 y consiguió un registro de 8000 clientes en las primeras 24 horas.

En el desarrollo de proyecto API, como herramienta de comunicación, hemos decidido utilizar slack, ya que nos permite sincronizar las diferentes herramientas de desarrollo que en nuestro proyecto emplearemos y así centralizadas en un solo canal de comunicación.

**MySQL**: MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual: Licencia pública general/Licencia comercial por Oracle Corporation y está considerada como la base datos de código abierto más popular del mundo,1​2​ y una de las más populares en general junto a Oracle y Microsoft SQL Server, sobre todo para entornos de desarrollo web.  
  
MySQL fue inicialmente desarrollado por MySQL AB (empresa fundada por David Axmark, Allan Larsson y Michael Widenius).

**NodeJS**:

​

**Atom**: Atom es un editor de código de fuente de código abierto para macOS, Linux, y Windows con soporte para plug-ins escritos en Node.js y control de versiones Git integrado, desarrollado por GitHub.

En el desarrollo del proyecto API, utilizaremos el editor de texto Atom, para codificar lo que nuestro proyecto requerirá para ser funcional, hemos decidido utilizarla para aprovechar todas sus herramientas que nos proporciona.

**e. Miembros del equipo**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **NOMBRE COMPLETO** | **No. CARNÉ** | **Cargo** |
| 1 | Eddy Estuardo Agustín Marroquín | 0903-13-13796 | Director del Proyecto |
| 3 | Erwin Jonathan Navarro Fuentes | 0903-09-5855 | Developer |
| 4 | Julio Noe de Leon Soc | 0903-13-12163 | Developer |
| 5 | Alma Maria Cordón | 0903-13-8610 | Product Owner |