

# DriverAssistant

## Introducción

La labor del conductor de camiones de carga implica estar expuesto a numerosos riesgos en la carretera, que atentan contra la seguridad y el bienestar físico y mental de quienes se dedican a ella.

Consideramos que el bienestar y seguridad de los conductores que mueven el sistema logístico es un punto central para el desarrollo eficiente de dicha industria, lo que nos impulsó a buscar una propuesta enfocada a esta problemática.

## Nuestra propuesta

Diseñar y desarrollar un asistente virtual para los conductores de camiones de carga, una poderosa herramienta que, ofreciendo una amplia variedad de funcionalidades, contribuya a mejorar la seguridad y bienestar físico y emocional de los conductores, buscando prevenir accidentes y minimizar los efectos psicológicos negativos mediante el uso de tecnologías emergentes como el análisis de datos y la inteligencia artificial generativa.

### **Aplicación para Android**

Los dispositivos móviles como celulares y tabletas poseen características de hardware perfectas para DriverAssistant. Sus componentes para la geolocalización, conexión a datos móviles y micrófono integrado los convierten en herramientas perfectas para este caso de uso.

Además, versiones de Android como Android Automotive, que cuentan con acceso directo a indicadores del vehículo como su nivel de gasolina, geolocalización, nivel de aire en llantas, entre otros, aumentan las posibilidades de lo que puede lograr al desarrollar el proyecto para esta plataforma.

## Gran oferta de funcionalidades

DriverAssistant es un asistente multiusos. A futuro y con el trabajo suficiente, puede ser equipado con todas las funcionalidades que un conductor pueda necesitar en el día a día de sus labores. La propuesta que hoy se presenta contempla las siguientes funcionalidades:

- Compañero inteligente que lee mensajes al conductor utilizando TTS para proveerle de información importante y acompañamiento durante el viaje.
- Alertas sobre el camino en base a una base de datos donde se registran puntos de riesgo en la carretera, como tramos en mal estado o curvas peligrosas.
- Monitoreo de ubicación del conductor, para que sus familiares autorizados y personal dentro de la empresa en la que trabaja pueda estar pendiente de su recorrido.
- Direcciones, funcionalidad fundamental para cualquier conductor, donde se calcula la ruta más óptima para llegar al punto de destino.
- Monitor de indicadores físicos obtenidos por medio de periféricos externos, en particular el ritmo cardiaco resulta de utilidad para detectar si el conductor entra en estado de somnolencia, dando la oportunidad de actuar para evitar que se quede dormido.
- Buscador de lugares, para que el conductor pueda solicitar direcciones para llegar a un negocio de comida, tienda de autoservicio, farmacia, hotel, entre otros.

## Arquitectura de microservicios

Una arquitectura distribuida, donde las funcionalidades del sistema se dividen en pequeños servicios responsables de un conjunto de responsabilidades limitado, proporciona valiosos atributos como la disponibilidad y la escalabilidad, cruciales para un sistema como [proyecto].

## Tecnologías emergentes

El compañero inteligente de [proyecto] utiliza el poder de la IA generativa para brindar mensajes personalizados, generados al momento para el conductor, tomando en cuenta datos relevantes como la ubicación, clima, tamaño y peso de la carga del vehículo, horas de viaje y de descanso, entre otros.

Existe también una amplia área de oportunidad para el análisis de datos, de tal manera que [proyecto] pueda conocer el estado de las carreteras, y sea consciente de áreas de peligro como curvas peligrosas o zonas con peligro de desbordamiento de rocas, con ello, el compañero inteligente pueda advertir de ellas de manera oportuna al conductor, evitando así accidentes y salvando vidas.

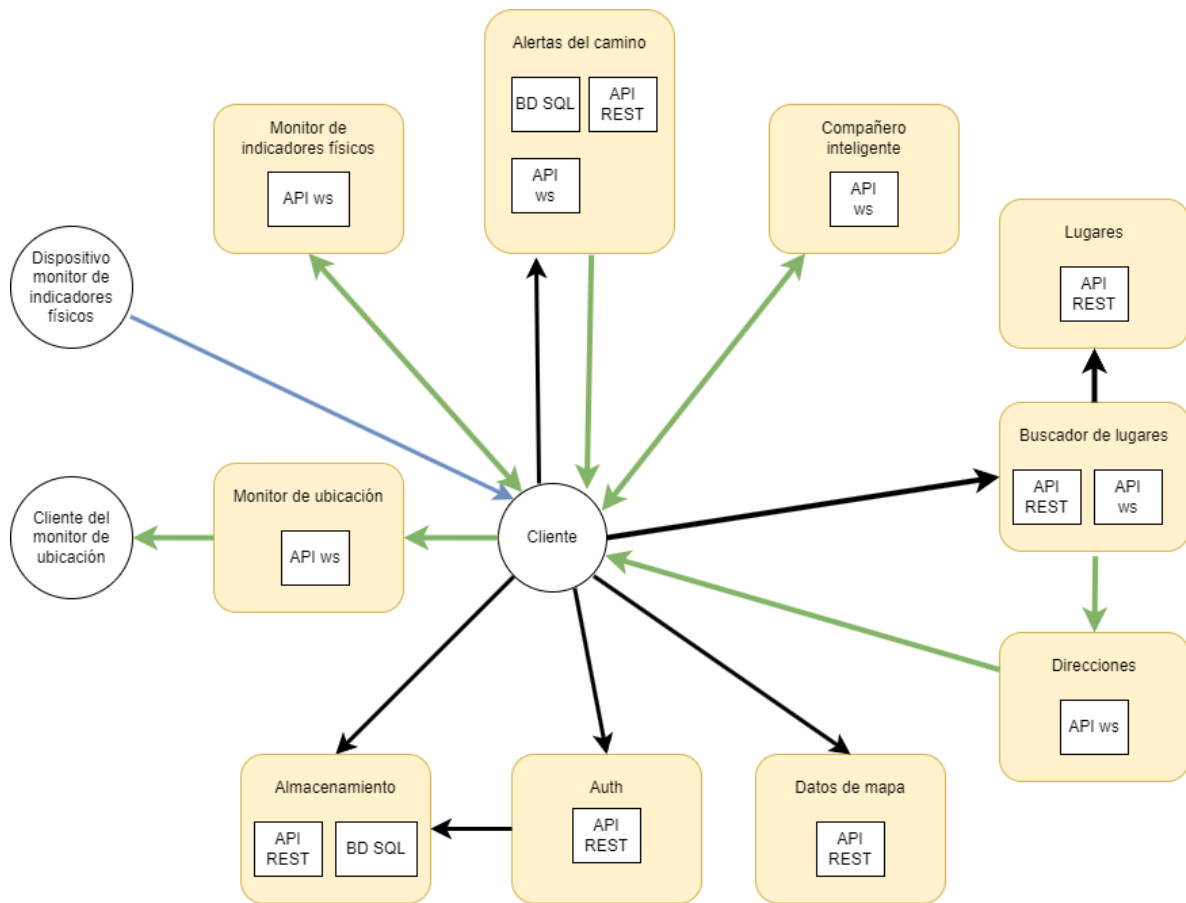
## Detalles técnicos

### Microservicios y sus responsabilidades

Nombre del microservicio	Descripción	Interfaz
<b>Almacenamiento</b>	Almacena datos acerca de los conductores, empresas, vehículos y familiares de los conductores en una base de datos relacional.	REST
<b>Auth</b>	Verifica la validez de las credenciales de acceso proporcionadas y genera un Token de acceso para el cliente.	REST
<b>Monitor de ubicación</b>	El servicio recibe constantemente la ubicación del conductor y la comunica a los interesados, como pueden ser los familiares del conductor y personal de su empresa.	WebSockets
<b>Datos de mapa</b>	Proporciona los datos del mapa, necesarios para visualizarlo en el cliente. Utilizamos herramientas de Google Maps Platform.	REST
<b>Direcciones</b>	Se encarga de obtener la ruta más óptima y comunicarla al cliente. Debe ser capaz de reaccionar ante solicitudes del <i>Buscador de lugares</i> para recalcular la ruta. Internamente utiliza herramientas de Google Maps Platform.	WebSockets
<b>Compañero inteligente</b>	El servicio recibe regularmente datos del conductor, tales como la ubicación, clima, tamaño y peso de la carga del vehículo, horas de viaje y de descanso, entre otros. Utilizando IA	WebSockets

	generará y enviará mensajes al conductor cuando lo considere conveniente, con el objetivo de cuidar su bienestar físico y mental. Utilizamos la API de OpenAI.	
<b>Alertas del camino</b>	<p>Contando con un amplio registro de lugares peligrosos en la carretera, como curvas peligrosas o áreas con riesgo de desprendimiento, alerta al conductor de peligros en el camino.</p> <p>Los conductores podrán contribuir advirtiendo anomalías en la carretera, como baches, accidentes, objetos obstruyendo la vía, entre otros.</p>	REST WebSockets
<b>Monitor de indicadores físicos</b>	Recibe indicadores físicos del conductor y los analiza en busca de anomalías, en caso de encontrar alguna, alerta al cliente.	WebSockets
<b>Lugares</b>	Servidor encargado de proporcionar datos de lugares, como negocios y sitios de interés. Internamente utiliza Google Maps Platform.	REST
<b>Buscador de lugares</b>	<p>Permitirá al cliente ubicar lugares cercanos cuando lo solicite, por ejemplo, una gasolinera, tienda de autoservicio o restaurante.</p> <p>Elige el lugar ideal para el conductor y lo comunica a <i>Direcciones</i> para modificar la ruta.</p>	REST WebSockets

El siguiente diagrama ilustra el sistema, donde las flechas negras indican el uso de HTTP (REST), las verdes WebSockets y la azul una conexión por Bluetooth.

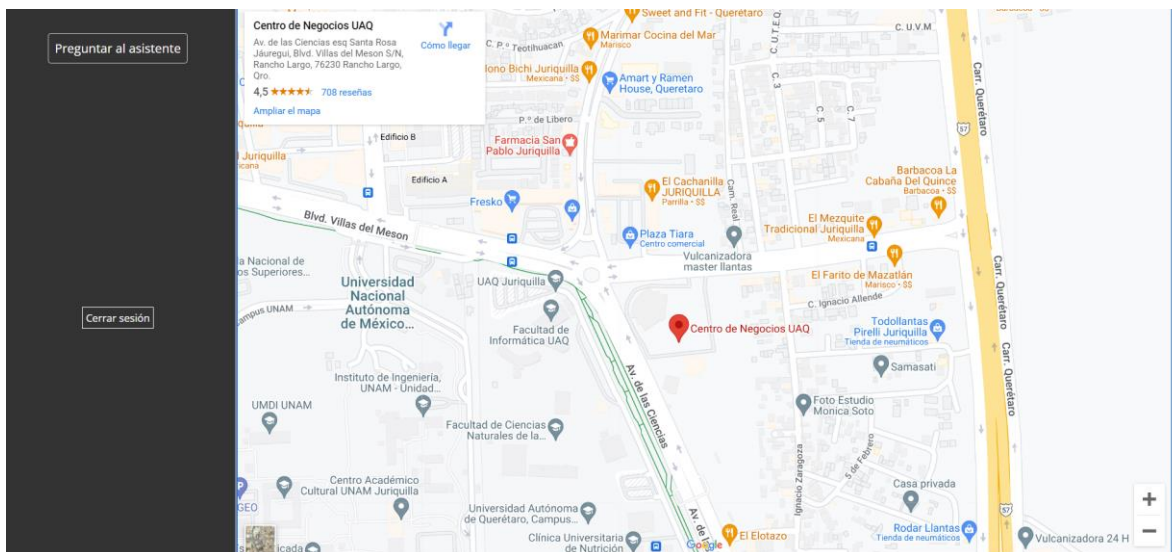


## Avances de implementación

### Cliente

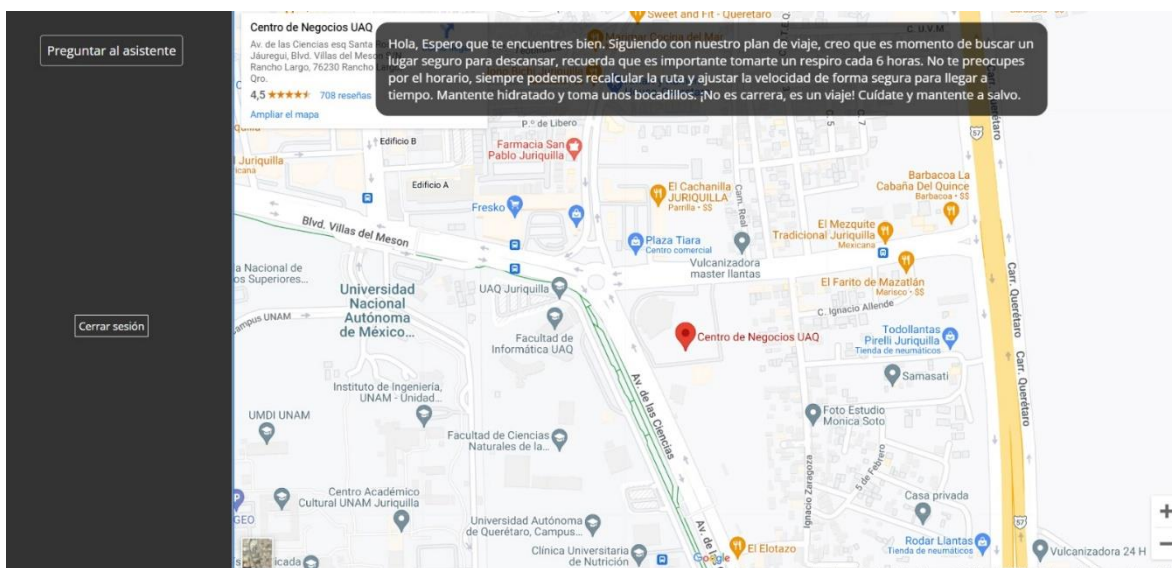
El cliente es la aplicación que utiliza el conductor, desarrollada para su uso en dispositivos Android, provee de vistas y controles para interactuar con el sistema.

## Vista principal:



Se muestran mensajes emergentes en un dialogo que aparece en la parte superior de la pantalla.

Al dar clic en el botón “Preguntar al asistente” del menú lateral, se obtiene un mensaje del Compañero inteligente generado en el momento con IA:



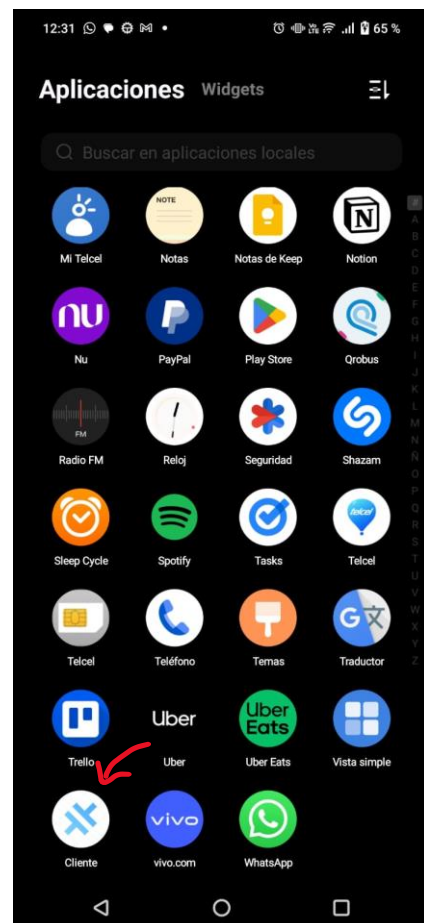
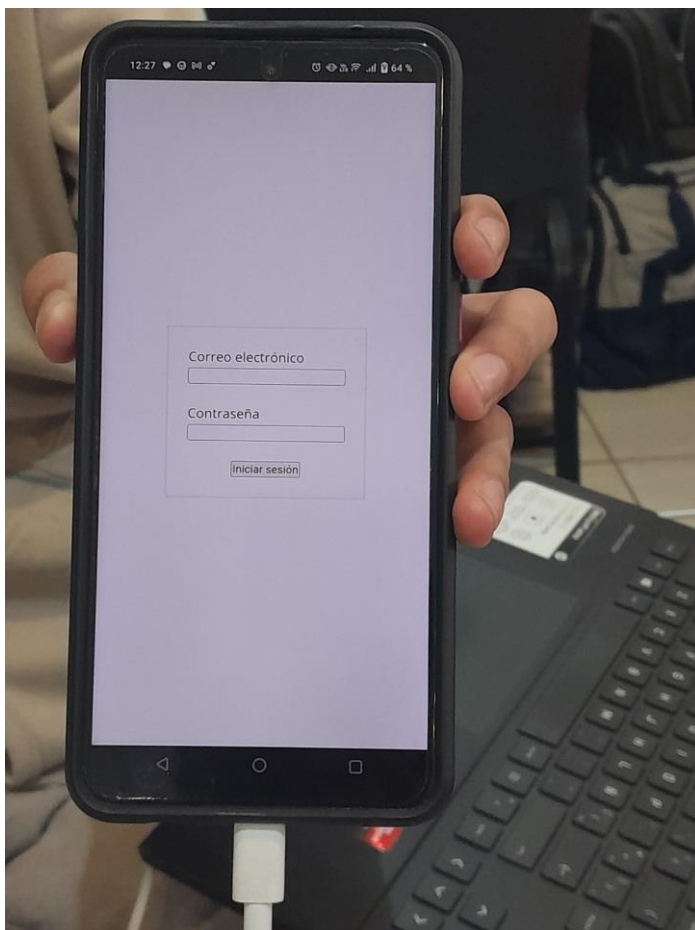
## Formulario de inicio de sesión:



A screenshot of a web browser displaying a login form. The browser's address bar shows "http://localhost:5173". The form is centered on the page and contains the following elements:

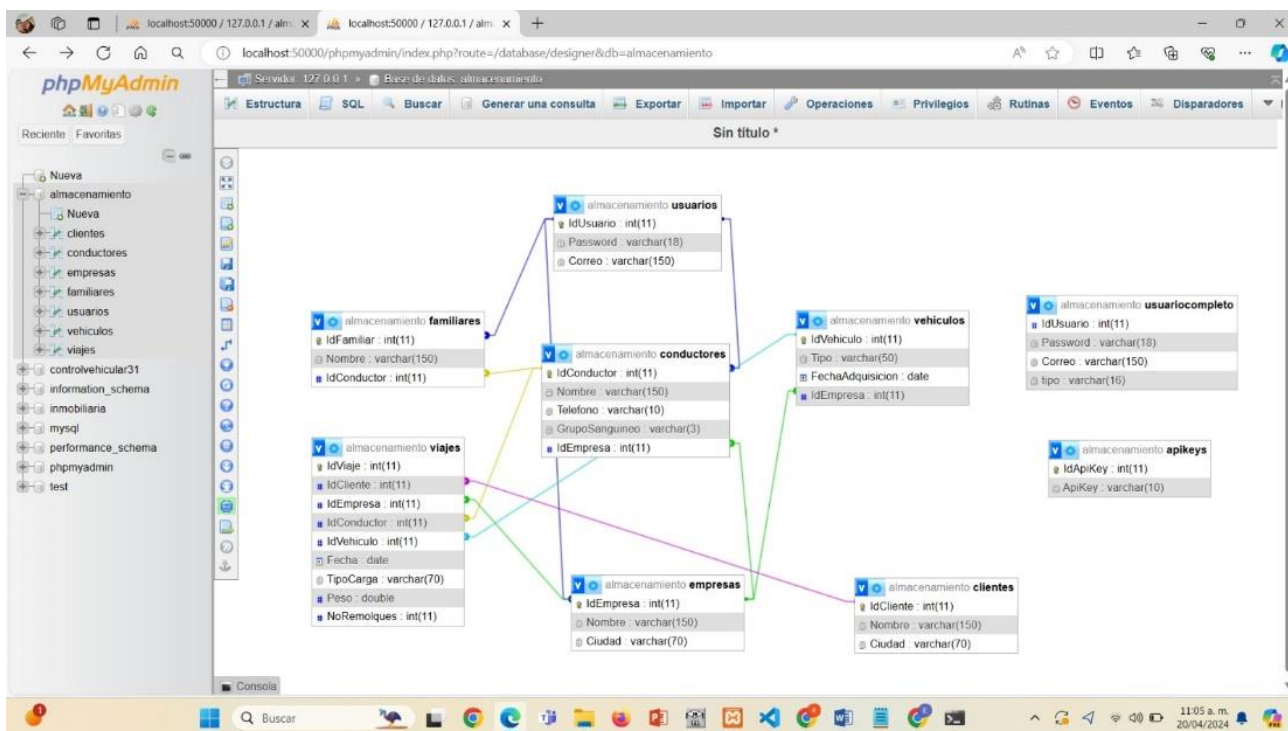
- A label "Correo electrónico" above a text input field containing the email "conductor@logistica.com".
- A label "Contraseña" above a password input field with masked characters "\*\*\*\*\*".
- A button labeled "Iniciar sesión" located below the password field.

## En dispositivo Android:



## Almacenamiento

Se implemento el servicio utilizando PHP y el entorno XAMPP. Utiliza la base de datos relacional SQL y se proporciona una API para obtener su contenido por medio de peticiones HTTP.



## Auth

Este microservicio fue una integración de Java con SpringBoot con JWT, el cual proporciona una base sólida para una implementación sencilla y segura, las credenciales de autenticación serán proporcionadas por el usuario que desee entrar,



este microservicio generará un token de acceso a la aplicación, siendo este token la llave de acceso a los servicios de nuestra aplicación.

#### Estructura del token JWT

```
{
  "subject": number,
  "tipo": "conductor" | "empresa" | "familiar"
  "iat": "fecha expiracion"
}
```

### Monitor de ubicación

Logs del servidor:

```
Servidor escuchando en el puerto 3000
Cliente conectado:
Mensaje recibido desde el cliente: {"latitud":20.70512,"longitud":-100.442875}
Mensaje recibido desde el cliente: {"latitud":20.70512,"longitud":-100.442875}
Mensaje recibido desde el cliente: {"latitud":20.70512,"longitud":-100.442875}
Mensaje recibido desde el cliente: {"latitud":20.70512,"longitud":-100.442875}
Cliente conectado: familiar
Mensaje recibido desde el cliente: iniciar_viaje
Mensaje recibido desde el cliente: iniciar_viaje
Cliente desconectado: familiar
Cliente conectado: familiar
Mensaje recibido desde el cliente: {"latitud":20.705089,"longitud":-100.442952}
Mensaje recibido desde el cliente: {"latitud":20.705089,"longitud":-100.442952}
Mensaje recibido desde el cliente: {"latitud":20.705089,"longitud":-100.442952}
Mensaje recibido desde el cliente: {"latitud":20.705089,"longitud":-100.442952}
Mensaje recibido desde el cliente: {"latitud":20.705089,"longitud":-100.442952}
```

### Cliente del monitor de ubicación

Se desarrolló un cliente web para que familiares de conductores y personal de la empresa pueda observar la ubicación en tiempo real del conductor plasmada en un mapa.

### Seguimiento de Ruta - Conductor

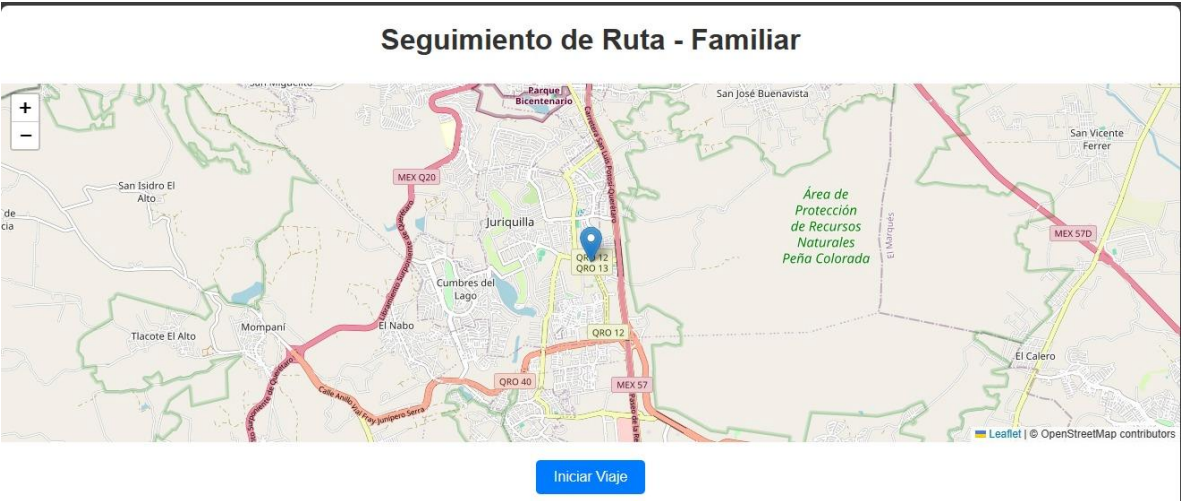
Comenzar Seguimiento
Detener Seguimiento

### Iniciar Sesión Conductor

Usuario:

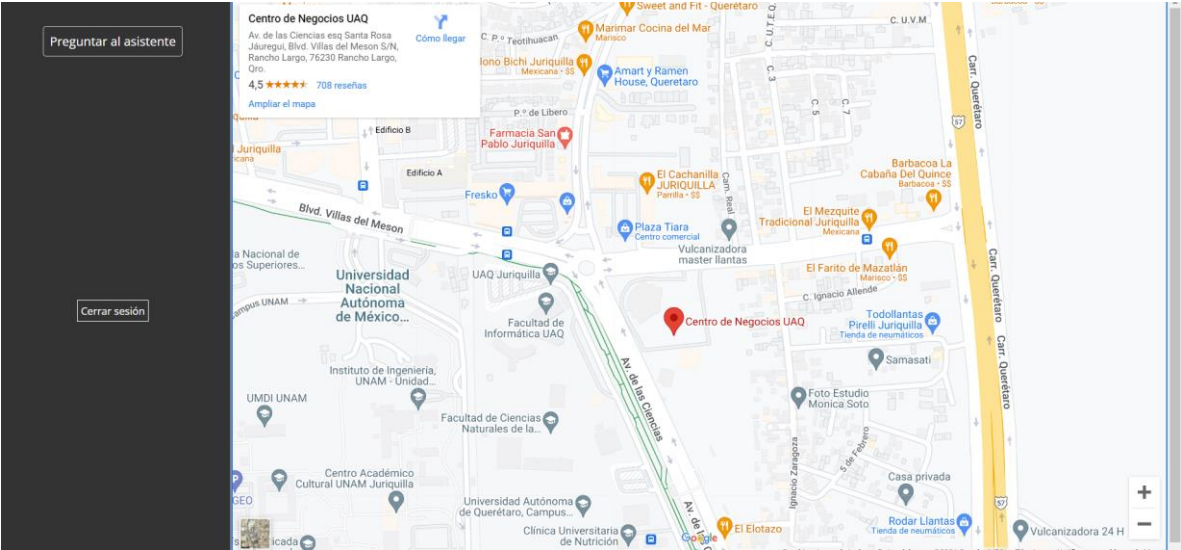
Conductor 1

Ingresar



Datos de mapa

Se incluyó en el cliente un mapa embebido utilizando las APIs de Google Maps Platform, sobre este mapa se dibujarían las direcciones que el conductor debe seguir para llegar a su destino, utilizando datos del microservicio de Direcciones.



## Direcciones

Este microservicio tiene la finalidad de proporcionar la ruta de recorrido y una serie de instrucciones para que la aplicación pueda marcar la ruta a recorrer. Este microservicio trabaja en conjunto con el microservicio de Lugares; recibe un destino el cual tiene que estar lo más descriptivo (conteniendo colonia, calle, número exterior, interior, CP, etc.); el segundo parámetro es la ubicación actual, una vez hecho esto, el microservicio proporcionará un JSON con toda la información del recorrido que el conductor tendrá que hacer; estos resultados son muy completos ya que contiene el destino de inicio y final, tanto en formato de dirección como en formato de coordenadas longitudinales, el tiempo de duración de cada uno de los pasos, entre otras cosas. Dentro de este microservicio se implementaron APIs de Google Maps Platforms.

Realizando peticiones al microservicio:

The screenshot shows a REST client interface with a POST request to `http://localhost:4230/search`. The request body is a JSON object with the following structure:

```
1 {
2   "destination": "Naranjos Punta Juriquilla 1000 Santa
3   Rosa Jáuregui Jurica Acueducto 76230 Santiago de
4   Querétaro Qro Mexico",
5   "origin": "20.704316,-100.443771"
6 }
7
8 }
```

The response is a 200 OK status with a 4.66 s response time and 1214 B of data. The response body is a JSON object with the following structure:

```
40 },
41 "end_address": "Naranjos Punta Juriquilla 1000, Santa Rosa Jáuregui, 76230 Santiago de Querétaro, Qro., Mexico",
42 "end_location": {
43   "lat": 20.7089786,
44   "lng": -100.4458008
45 },
46 "start_address": "Naranjos Punta Juriquilla 1000, Santa Rosa Jáuregui, 76230 Santiago de Querétaro, Qro., Mexico",
47 "start_location": {
48   "lat": 20.7089786,
49   "lng": -100.4458008
50 },
51 "steps": [
52   {
53     "distance": {
54       "text": "1 m",
55       "value": 0
56     },
57     "duration": {
58       "text": "1 min",
59       "value": 0
60     },
61     "end_location": {
62       "lat": 20.7089786,
63       "lng": -100.4458008
64     },
65     "html_instructions": "Head",
66     "polyline": {
67       "points": "cvkj8Fiadk"
68     },
69     "start_location": {
70       "lat": 20.7089786,
71       "lng": -100.4458008
72     },
73     "travel_mode": "DRIVING"
74   }
75 ],
76 "traffic_speed_entry": [],
77 "via_waypoint": []
```

## Compañero inteligente

El Compañero inteligente utiliza la API de OpenAI y su modelo gpt-4 para generar mensajes para el conductor. Se incluyen en el prompt datos relevantes como la hora, ubicación, velocidad media y tiempo desde el último descanso para obtener una respuesta útil del modelo. Estos mensajes son enviados al cliente para mostrarlos al usuario.

```
const getMessageForAssistant = () => {  
  return (  
    `Soy un conductor que se encuentra conduciendo un vehículo cargado hasta el norte del país.  
    Tu eres mi asistente, tu trabajo consiste en aconsejarme y apoyarme para que pueda cumplir con mi entrega en tiempo y forma,  
    considerando que la seguridad y bienestar físico y emocional deben ser una prioridad en mi profesión. Tu mensaje debe ser breve y amigable.  
    Te comparto los siguientes datos sobre mi viaje en este momento:  
    Hora actual: ${hours}:${min} (formato 24hrs)  
    Ubicación actual: Juriquilla  
    Velocidad media en la última hora: 100km/hr  
    Tiempo transcurrido desde el inicio del viaje: 12 horas  
    Tiempo desde el último descanso: 6 horas`  
  );  
};
```

## Logs de las respuestas recibidas del modelo:

Hola, veo que estás haciendo un buen progreso hacia el norte. Sin embargo, recuerda que la seguridad es esencial. Has estado conduciendo durante 6 horas desde tu último descanso, sería recomendable que te detengas para un breve descanso, podrías tomar un café, estirar tus piernas y descansar tus ojos un rato. Suena tentador, ¿verdad? ¡Mantente seguro y feliz viaje!

Hola, excelente trabajo hasta ahora, pero es importante que cuides de tu bienestar. Ya llevas 6 horas sin tomar un descanso y es recomendable hacerlo cada 4-5 horas aproximadamente. Te sugiero que busques un lugar seguro para tomar un breve descanso, estirar las piernas y comer algo nutritivo. Mantén una velocidad moderada para garantizar tu seguridad. ¡Sigue adelante, lo estás haciendo genial!

¡Hola! Veo que has estado en camino durante bastante tiempo desde que tomaste tu último descanso y que tu velocidad ha sido constante en la última hora. Quisiera sugerirte que hagas una pausa en la siguiente parada segura, tal vez puedas estirar las piernas y refrescarte un poco. Recuerda que tu salud y seguridad es lo más importante. Sigue haciendo un excelente trabajo.

Hola, parece que has estado en la carretera un buen rato. Recuerda que descansar es fundamental. Sería una buena idea buscar un lugar seguro para tomar un descanso pronto, tal vez en una hora. Conduces a buen ritmo, pero no olvides que la seguridad es lo primero. ¡Mantén la concentración y llegaremos a nuestro destino a tiempo! ¡Vamos!

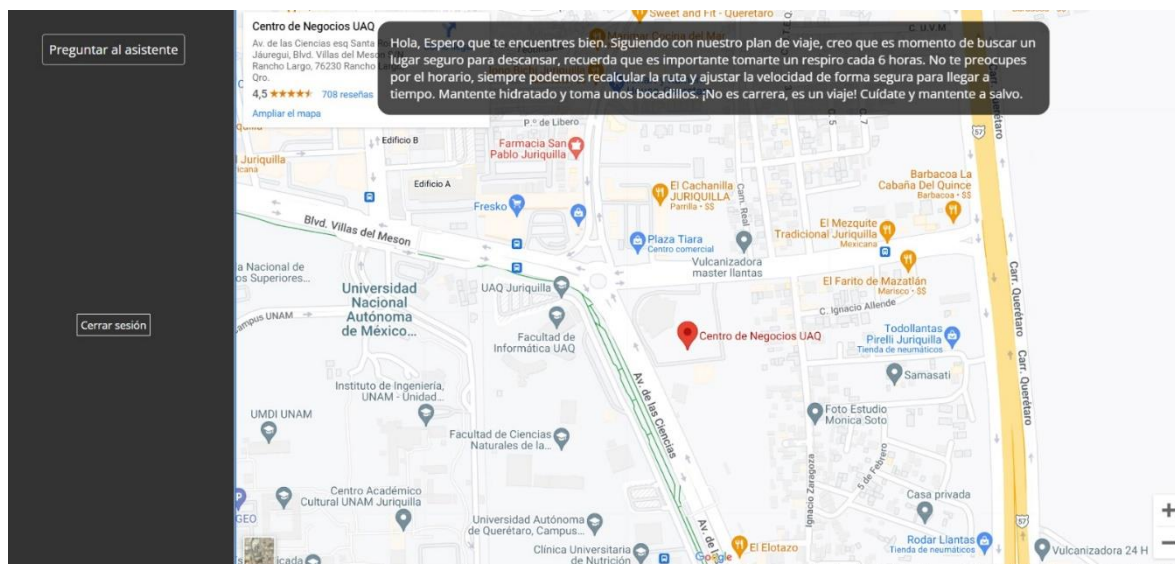
Hola, parece que has estado conduciendo durante varias horas hasta ahora. Queremos asegurarnos de que llegues a salvo a tu destino, por lo que te recomendamos encontrar un lugar seguro para detenerte.

Hola, parece que estás haciendo un gran trabajo en tu viaje al norte. Dado que han pasado seis horas desde tu último descanso, te sugeriría que en la próxima hora encuentres un lugar seguro para estacionarte y tomar un breve descanso. Un poco de estiramiento y rehidratación son esenciales para mantenerse alerta en el viaje. ¡Mantente seguro!

¡Hola! Parece que llevas un buen ritmo, pero recuerda que es importante cuidar tu salud también. Dado que han pasado 6 horas desde tu último descanso, te sugeriría encontrar un lugar seguro para parar y estirar las piernas, beber agua y refrescar la mente. Juriquilla tiene zonas de estacionamiento seguras, continúa a buen ritmo, pero sin olvidarte de ti. ¡Buen viaje!

Hola, Espero que te encuentres bien. Siguiendo con nuestro plan de viaje, creo que es momento de buscar un lugar seguro para descansar, recuerda que es importante tomarte un respiro cada 6 horas. No te preocupes por el horario, siempre podemos recalcular la ruta y ajustar la velocidad de forma segura para llegar a tiempo. Mantente hidratado y toma unos bocadillos. ¡No es carrera, es un viaje! Cuidate y mantente a salvo.

## Mensaje del Compañero inteligente apareciendo en el cliente.



## Monitor de indicadores físicos

Se implementó una simulación del microservicio utilizando Node.js y la comunicación por WebSockets entre un cliente web y un servidor. Se generan datos de frecuencia cardíaca y se verifica en qué rango se encuentran, se muestra un mensaje si el dato sobrepasa el rango ideal.

### Monitor de Indicadores Físicos

¡Alerta! Se han detectado las siguientes anomalías: Frecuencia cardíaca alta.

Últimos datos de indicadores físicos:

Frecuencia cardíaca: 108

Presión arterial: 91

## Lugares

Se utilizan APIs de Google Maps Platform para obtener datos de lugares (negocios y centros de interés). En la configuración de la implementación de la API en este proyecto, el rango de búsqueda quedó delimitado en un rango de 500 metros, pero, al no encontrar las ubicaciones coincidentes, proporcionará una lista con los resultados más acertados en cuanto a las características proporcionadas. Para poder obtener la lista de resultados de ubicaciones dentro del rango, el microservicio recibirá, además del texto de búsqueda, la ubicación actual del conductor en medidas longitudinales (latitud y longitud). Este microservicio es utilizado por el Buscador de lugares para obtener los datos de lugares.

## Buscador de lugares

El microservicio de lugares permite realizar una búsqueda en lenguaje natural del lugar al que se desea llegar. Puede ser utilizado por el cliente, que generará y proporcionará una entrada con las características más aproximadas de la ubicación a encontrar, tomando en cuenta las instrucciones del usuario. De entre todos los resultados obtenidos, este microservicio elige el lugar ideal para la petición del usuario y envía la decisión al microservicio de Direcciones, quien generará una nueva ruta y la enviará al cliente.

Realizando peticiones al microservicio:

The screenshot displays a REST client interface with the following details:

- Request:** A POST request to `http://localhost:4200/search`. The request body is a JSON object: 

```
{  "text": "Facultad de Informatica uaq",  "latitud": "20.704316",  "longitud": "-100.443771"}
```
- Response:** The response is a 200 OK status with a response time of 276 ms and a body size of 174 B. The response body is a JSON object: 

```
{  "places": [    {      "formattedAddress": "PHB4+6W, Av. de las Ciencias S/N, 76230 Juriquilla, Qro., Mexico",      "displayName": {        "text": "Facultad de Informática UAQ",        "languageCode": "es"      }    }  ]}
```
- Interface Elements:** The interface includes tabs for Parameters, JSON, Auth, Headers (1), and Preview. The Preview tab is currently selected, showing the response body. There is also a 'Send' button and a status bar at the top right with an 'Invite' button and a user profile icon.