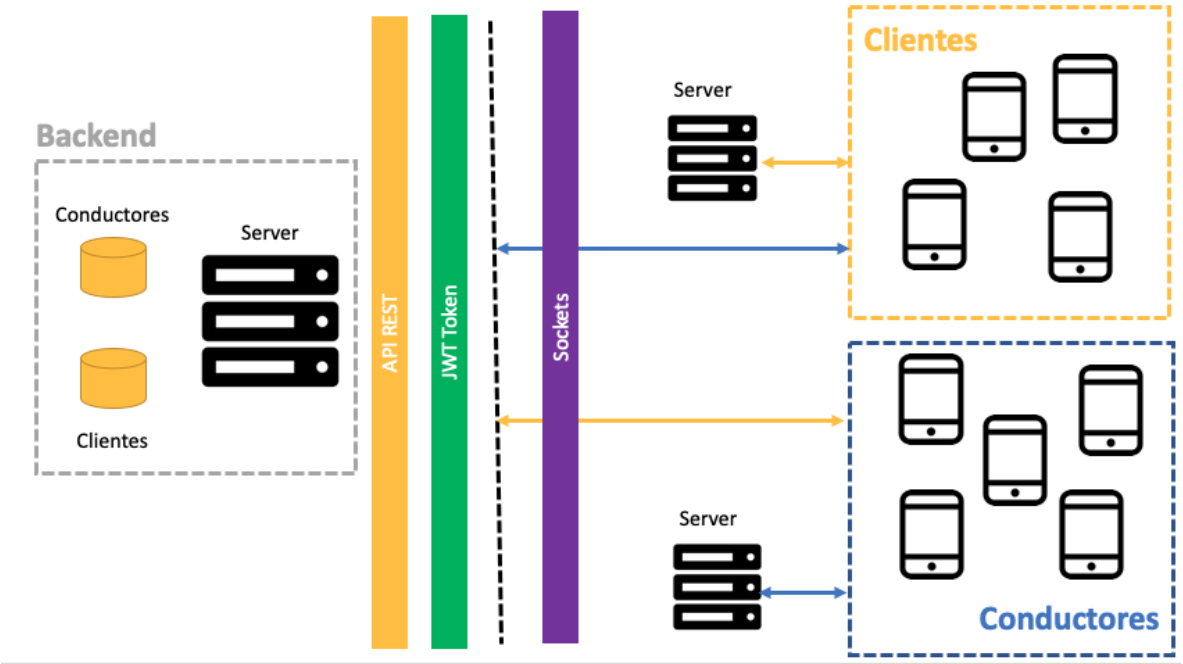
ARQUITECTURA DISTRIBUIDA
Conductor-Usuario

Para realizar este tipo de arquitecturas es necesario tener conocimientos en el desarrollo de **APIREST**, **WebSockets** y **Microservices**, acontinuación se explica, como se puede llevar a cabo el desarrollo de este tipo de arquitecturas.

Arquitectura de Modulación:



Debemos realizar un modelo de arquitectura moderno, utilizando herramientas y tecnologías de software que permitan realizar conexiones remotas haciendo uso del protocolo **HTTP** para realizar operaciones **CRUD** a las bases de datos.

Debemos tener en cuenta el concepto **FRONTEND** y **BACKEND**, el cuál es la base para desarrollar este tipo de aplicaciones.

Para el desarrollo de esta plataforma, implementaremos 3 módulos los cuales serán enlazados mediante **WebSockets** para cumplir con el requerimiento **RealTime,** estos son:

App Frontend **ConductoresApp**
App Frontend **ClientesApp**
API **REST Backend Server**

¿Porque **Javascript**?
Es un lenguaje por excelencia su exclusividad es nata y es muy robusto, además es multiplataforma y su rendimiento es 100% compatible con cualquier navegador del mercado actual y es ejecutable también del lado del servidor.

Tecnologías y Frameworks:

NPM: Gestor de paquetes de NodeJS.

MongoDB: Gestor de Bases de datos documentales basadas en clases, es el modelo NoSQL de bases de datos por excelencia.

NodeJS: nos permitirá crear un servidor web con Javascript.

ExpressJS: nos permitirá crear un API REST robusta compatible con NodeJS.

Socket.io: Tecnología que permite realizar consultas en tiempo real cliente - servidor, actualmente es la librería más utilizada para este tipo de aplicaciones.

JSONWebToken: Sistema de autenticación basado en sesiones de usuario autenticados por tokens de verificación entre aplicaciones a través de HTTP.

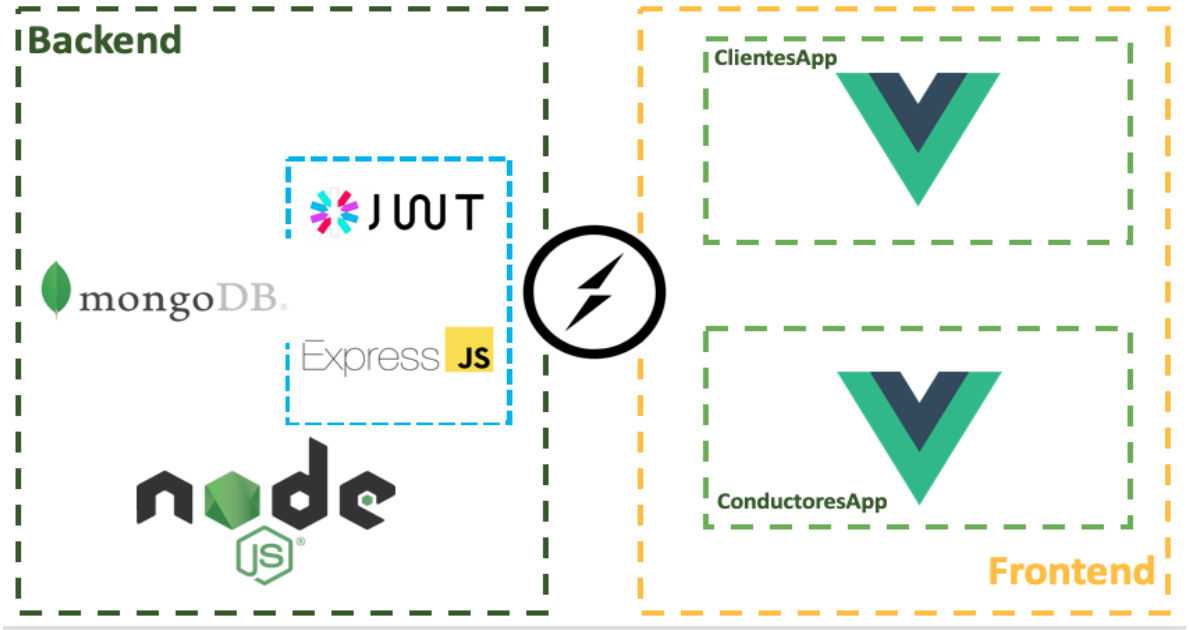
Mongoose: Librería para NodeJS que permite conectarse a MongoDB.

Morgan: Librería para declarar middlewares en NodeJS.

VueJS: Uno de los Frameworks Frontend más utilizado para crear aplicaciones del lado del cliente.

Nodemon: Servicio para correr servidores web en caliente.

Diagrama de Tecnologías:



Descripción Backend:

Se creó un APIREST con autenticación JSONWebToken para realizar las operaciones CRUD básicas, además de dos rutas de autenticación independientes para cada una de las aplicaciones frontend.

Descripción Frontend:

Se crean dos aplicaciones con vue-cli, una para solo clientes del sistema y otra para conductores, ambas tendrán un login y registro, la segunda permitirá internamente tomar mediante socket.io la ubicación actual a través de la ip de navegación y esta se conectara al servidor WebToken de NodeJS en el cual será enviada en tiempo real hacia la aplicación del Cliente en la cual se podrán visualizar todos los conductores con sus ubicaciones y su actual estado, además esta tendrá una lista de los conductores y su estado en la cual se podrá seleccionar el conductor de preferencia.