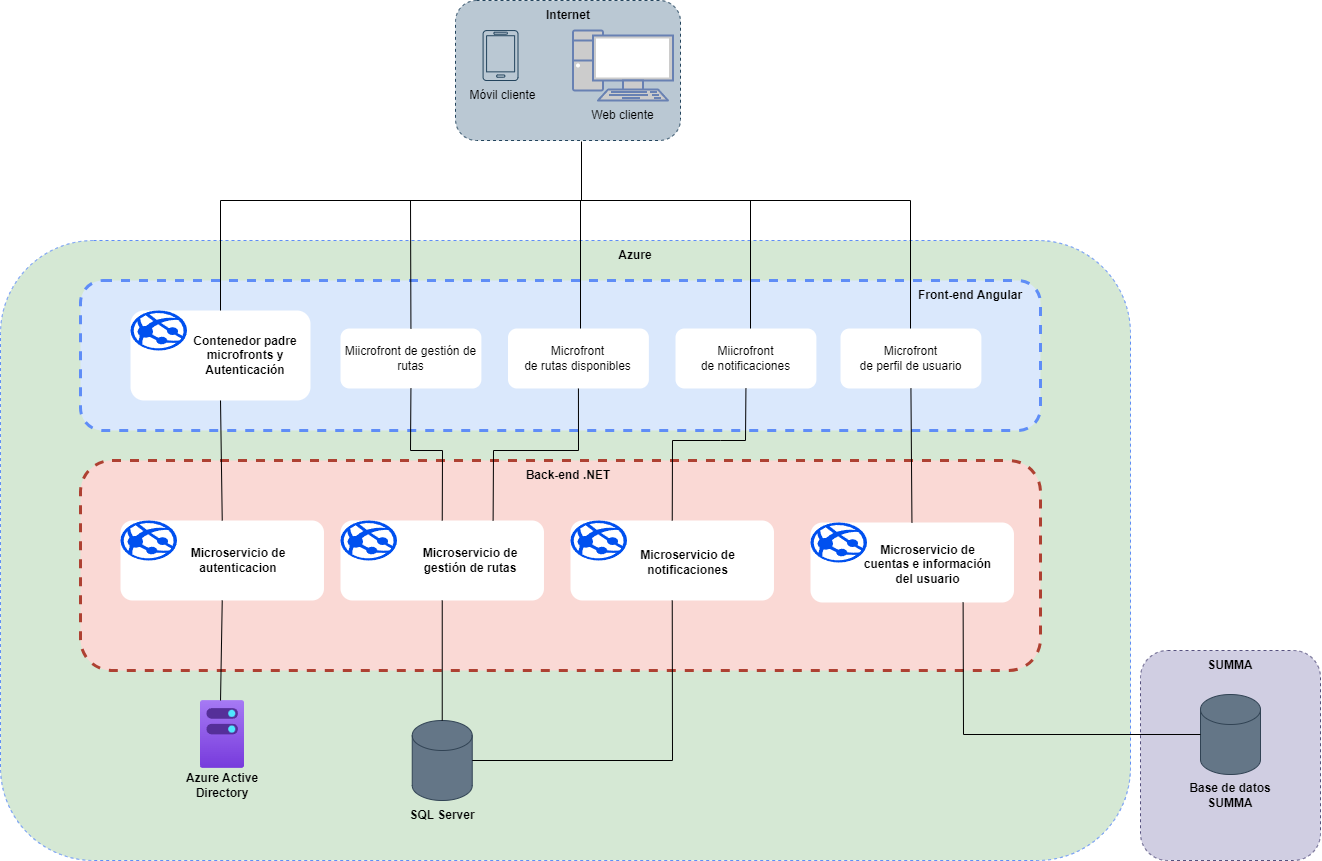
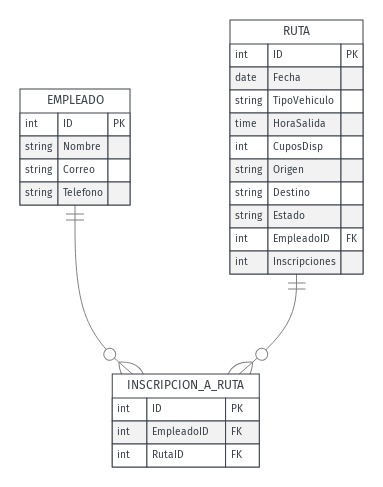
***Desarrollo prueba técnica Santiago Franco Rendón***

Parte 1

**Diagrama de arquitectura**



**Diagrama de clases**



**Tecnologías para utilizar**

Arquitectura de microservicios en .net, esto con el fin de generar desacoplamiento del sistema ya que al tener las funcionalidades separadas en diferentes proyectos hace mas sencillo su mantenimiento, escalabilidad y desarrollo. Cada microservicio puede ser implementado usando ASP.NET para aprovechar algunos frameworks que nos presenta la plataforma .NET, como por ejemplo EntityFramework para el manejo mas sencillo de entidades de base de datos. Realizar la implementación de un API Gateway para hacer el enrutamiento de las solicitudes de todos los microservicios.

La autenticación se puede realizar mediante Azure Active Directory que es un servicio que nos facilita la gestión de accesos y realizar una integración con JSON Web Tokens para la creación de un token que pueda ser usado por las APIS posteriormente como parámetro obligatorio para tener un nivel de seguridad.

Para la generación de los microfronts se usará el framerwork de JavaScript Angular, ya que su trabajo por módulos facilita el desarrollo y ayuda a tener un proyecto más organizado, es fácil de manejar su enrutamiento para las diferentes vistas. Para la creación de los microfronts se cuentan con varias librerías que facilitan el desarrollo de estos como single SPA o ng-module-federation.

El proceso de implementación y despliegue se realizará en Azure App Service para la aplicación web y Azure Mobile Apps para la aplicación móvil.

Para el control de versiones se usará la herramienta Git que permite un trabajo mas organizado ya que cada desarrollador puede tener su rama de trabajo según lo que este desarrollando sin afectar lo que ya se tenga hasta el momento.

**Metodología de desarrollo**

La metodología de desarrollo estará basada en SCRUM con algunas de sus ceremonias como los dailys, sprint review, planning. Los sprints serán de 1 semana para llevar un mejor control de las tareas del cada sprint. Se podrán acordar entregas parciales de módulos o funcionalidades con fechas y tiempos que se acuerden durante el planning. Se podrá llevar mejor seguimiento del desarrollo usando la herramienta Azure DevOps en su apartado de Boards que permite crear ahí todas las historias de usuario que requiera el proyecto, hacer la creación de los spints y llevar mejor control de las horas estimadas y empleadas de cada historia de usuario.

**Buenas prácticas metodológicas**

Se pueden acordar las entregas parciales de diferentes historias de usuario, esto con el fin de que cada historia pueda tener una etapa de pruebas unitarias mas detallada que permita la detección temprana de bugs y su corrección antes de la integración con toda la funcionalidad.

Tener un ambiente dedicado a la etapa de QA donde el desarrollador también pueda llevar a cabo las pruebas unitarias en un ambiente lo más parecido al de producción.

Tener un paso a paso de los puntos críticos que se tuvieron a lo largo del desarrollo y pudieron ser detectados en la etapa de pruebas para facilitar posteriormente la salida a producción y que esta no presente inconvenientes.

Desarrollar con código limpio basado en los principios solid que permita una mejor sostenibilidad del codigo.

**Infraestructura y plataformas**

Microsoft Azure

Azure App Service: este servicio se usará para hacer el alojamiento de los microservicios.

Azure Active Directory: este servicio es necesario para la gestión de accesos, la autenticación y la autorización de los usuarios.

Azure DevOps: será necesario para poder trabajar sobre los pipelines que faciliten la implementación y la integración continuas, además de los repositorios de cada proyecto, documentación y seguimiento de cada desarrollo.

ASP.NET Core: esta plataforma de .NET será utilizada para el desarrollo de los microservicios del backend.

Angular: este framework se empleará para el desarrollo de los microfrontend.

Visual Studio y Visual Studio Code: estos IDE facilitan el trabajo con los lenguajes de programación escogidos para el desarrollo.

**Riesgos**

Cambios en los requisitos del sistema o falta de definición de alguna historia de usuario, este tipo de problema es evitable con una revisión detallada de cada historia por parte de todos los actores del proyecto y que siempre se tenga una buena comunicación durante el proceso de implementación.

La actualización y los despliegues pueden ser un punto critico del proyecto ya que si no se tiene un buen plan de actualización y despliegue es posible que este salga mal y termine afectando la disponibilidad del sistema.

La carga de usuarios puede ser un problema y se debe tener un monitoreo del desempeño del sistema para prevenir posibles colapsos.

La integración de los microfronts y microservicios con algunos servicios externos puede suponer un reto técnico para el equipo de desarrollo, por lo que se debe hacer un buen análisis de cada servicio que se requiera.