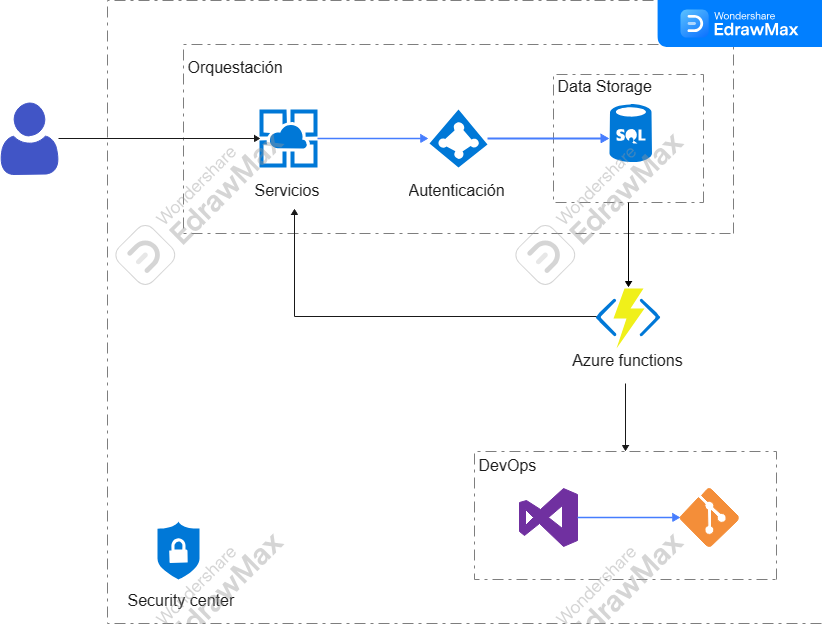
**Proyecto rutas de transporte**

* Diagrama de arquitectura

****

1. Azure App Service:

Aloja las aplicaciones web y móviles. Es el punto de entrada para los usuarios que interactúan con la solución a través de una interfaz web o una aplicación móvil.

Se conecta con otros servicios para procesar datos y manejar la lógica de negocio.

2. Azure Active Directory (AAD):

Proporciona servicios de autenticación y autorización.

Integra con Azure App Service para autenticar a los usuarios antes de permitirles acceder a la aplicación.

3. Azure SQL Database:

Almacena datos relacionados con las rutas de transporte, usuarios y otras entidades relevantes.

Está conectada con Azure App Service para recuperar y almacenar datos necesarios para las operaciones de la aplicación.

4. Azure Functions:

Permite ejecutar código de backend de manera escalable y sin servidor, ideal para operaciones como notificaciones y tareas automatizadas.

Puede interactuar con Azure SQL Database para procesar datos y con Azure App Service para enviar resultados o disparar acciones.

5. Azure DevOps:

Proporciona herramientas para la integración y entrega continuas (CI/CD), lo que facilita la actualización y el despliegue de la aplicación.

Está vinculado al código fuente y al proceso de desarrollo, no interactúa directamente con la aplicación en ejecución, sino que facilita su despliegue y mantenimiento.

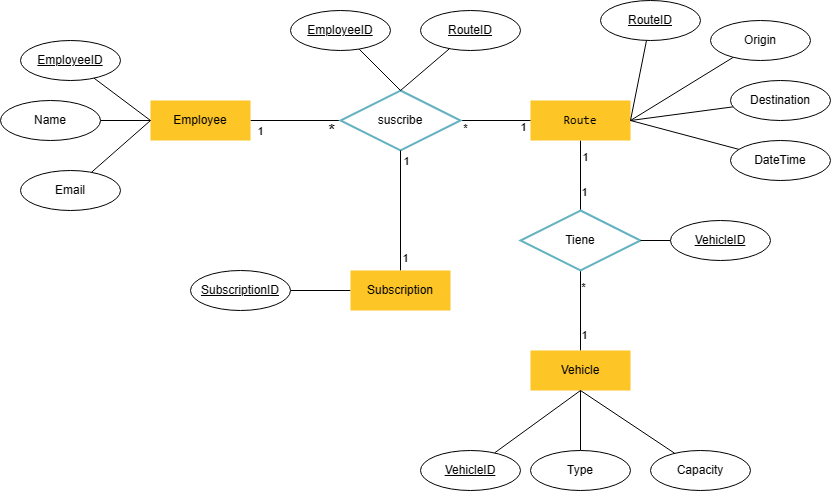
6. Azure Security Center:

Ofrece herramientas de seguridad y monitoreo para proteger la aplicación y los datos.

No se conecta directamente con los otros componentes en términos de flujo de datos, pero supervisa y protege toda la infraestructura.

En resumen, los usuarios interactúan con la aplicación a través de Azure App Service. La autenticación es manejada por Azure Active Directory. Los datos son almacenados y recuperados de Azure SQL Database o Cosmos DB. Azure Functions realiza tareas de backend, y Azure DevOps maneja el ciclo de vida del desarrollo y despliegue de la aplicación. Finalmente, Azure Security Center proporciona seguridad y protección a todos estos componentes.

* Modelo entidad relación

****

* Tecnologías, Lenguajes de Programación y Frameworks
* .NET y C#:

Justificación: .NET es un framework robusto y versátil para el desarrollo de aplicaciones web y móviles. C# es un lenguaje de programación moderno, con un buen soporte para paradigmas de programación orientada a objetos y funcional.

Uso: Para el backend del sistema y la lógica de negocio.

* React o Angular (para la interfaz web) y React Native (para la aplicación móvil):

Justificación: React/Angular es ampliamente utilizado para el desarrollo de interfaces web dinámicas, mientras que React Native permite el desarrollo de aplicaciones móviles multiplataforma con una base de código compartida.

Uso: Para el desarrollo de la interfaz de usuario tanto en la web como en móviles.

* Azure SQL Database o Azure Cosmos DB:

Justificación: La elección entre SQL y NoSQL dependerá de la estructura de los datos y los requisitos de escalabilidad. Azure ofrece servicios administrados que facilitan la escalabilidad y el mantenimiento.

Uso: Como base de datos para almacenar información sobre rutas, vehículos, empleados y suscripciones.

* Azure Services (App Service, Functions, Active Directory, etc.):

Justificación: Ofrecen una plataforma escalable y segura para alojar aplicaciones, autenticación y ejecución de tareas automatizadas.

Uso: Para alojamiento, autenticación y servicios backend.

* Metodología de Desarrollo
* Agile/Scrum:

Justificación: Permite un desarrollo iterativo y adaptativo, facilitando la incorporación de cambios y la entrega continua de valor.

Uso: Para gestionar el proceso de desarrollo, con sprints regulares y revisiones constantes.

* Buenas Prácticas Metodológicas
* Integración y Despliegue Continuo (CI/CD):

Uso: Automatiza la construcción, pruebas y despliegue, mejorando la calidad del software y acelerando el tiempo de lanzamiento al mercado.

* Revisiones de Código y Pair Programming:

Uso: Mejora la calidad del código y facilita la transferencia de conocimientos entre los miembros del equipo.

* Pruebas Automatizadas:

Uso: Asegura que el software funcione como se espera y reduce los errores.

* Infraestructura y Plataformas Necesarias
* Azure Cloud Services:

Uso: Para alojamiento, base de datos, autenticación y otros servicios backend.

* Sistemas de Control de Versiones (como Git):

Uso: Para el seguimiento de cambios y colaboración en el código fuente.

* Posibles Riesgos y Mitigación
* Retrasos en el Desarrollo:

**Mitigación**: Uso de metodologías ágiles para adaptarse a los cambios y priorizar las funcionalidades.

* Sobrecarga del Servidor o Falta de Escalabilidad:

**Mitigación**: Diseñar una arquitectura escalable y realizar pruebas de carga.

* Fallos de Seguridad:

**Mitigación**: Implementar prácticas de seguridad desde el inicio y realizar auditorías y pruebas de seguridad regularmente.

**Se puede agregar adicionalmente:**

* Accesibilidad y Experiencia del Usuario
* Diseño Centrado en el Usuario: Investigar y entender las necesidades y comportamientos de los usuarios finales para diseñar interfaces intuitivas y accesibles.
* Pruebas de Usabilidad: Realizar pruebas de usabilidad para garantizar que la aplicación sea fácil de usar y cumpla con las expectativas de los usuarios.
* Seguridad y Cumplimiento
* Encriptación de Datos: Implementar encriptación tanto en reposo como en tránsito para proteger los datos sensibles.