# **Prueba Técnica Desarrollador Full Stack**

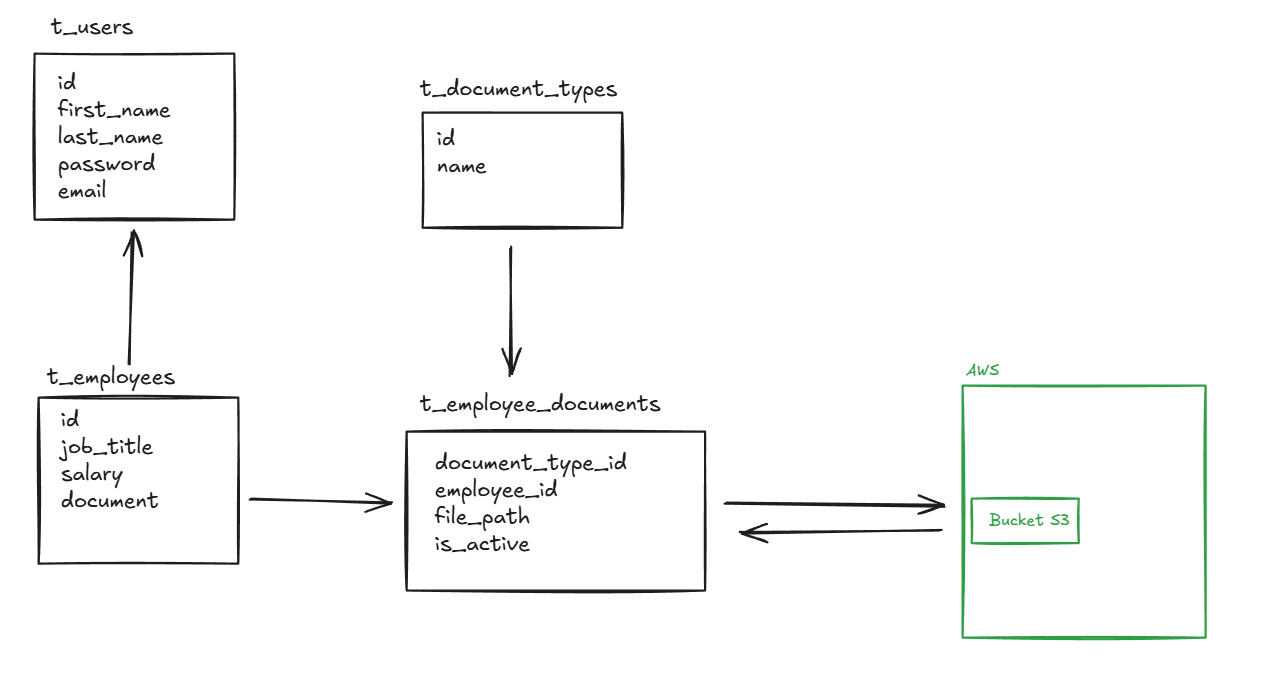
**Datgeo Technology**

## **Descripción del proyecto**

El proyecto consiste en desarrollar una aplicación web para la gestión de empleados en una empresa. El área de Recursos Humanos será responsable de registrar a los empleados en la aplicación, tras lo cual cada empleado deberá iniciar sesión y subir tres documentos escaneados (documento de identidad, licencia de conducir y CV) requeridos por la empresa en formato PDF. Para el backend deberá utilizar NestJS con TypeORM y PostgreSQL como base de datos, y en el frontend Next.js y Material UI.

**Es importante destacar que el backend debe estar preparado para integrarse con una aplicación móvil. Para ello, la autenticación y autorización de los usuarios se deben implementar utilizando JWT.**

**Nota:** Las tablas de la base de datos se crearán utilizando migraciones, y mediante un seeder se creará un usuario encargado de Recursos Humanos para gestionar el alta de empleados.Tomar como referencia el siguiente diagrama.



## **Requisitos técnicos**

* NestJS, TypeORM, PostgreSQL y AWS S3
* NextJS y Material UI

## **Funcionalidades**

1. La ruta raíz de la aplicación debe mostrar un formulario de inicio de sesión, donde el usuario pueda ingresar su correo electrónico y contraseña. Al hacer clic en el botón "Iniciar", la aplicación debe redirigir a la ruta */empleados* si el usuario es responsable de Recursos Humanos; en caso contrario debe redirigir a la ruta /mi-perfil.
2. En la ruta */empleados*, la aplicación debe mostrar una tabla de empleados con paginación de 50 filas. Cada fila debe incluir opciones para editar y dar de baja a un empleado. Además, en esta ruta debe haber una opción para dar de alta a un nuevo empleado.
3. Al hacer clic en *“Alta de empleado”*, la aplicación debe mostrar un formulario para ingresar los datos correspondientes, como nombres, apellidos, correo electrónico, contraseña, puesto, documento de identidad y el salario.
4. En la ruta */mi-perfil*, la aplicación debe mostrar los datos registrados en la funcionalidad n°3, así como los documentos subidos y los pendientes por subir. Al subir un documento, la aplicación debe almacenarlo en un Bucket S3 (usando un access y secret key) y guardar la ruta junto con el resto de datos en una tabla de la base de datos. Además, un documento subido debe poder previsualizar en la misma aplicación.

## **Criterios de evaluación**

* Código (buenas prácticas, optimizaciones, naming, modularidad, escalabilidad, componentes reutilizables, etc.)
* Funcionalidad
* Seguridad
* Diseño UI y UX
* Estructura de tablas en la base de datos
* Documentación de los endpoints con OpenAPI
* Control de cambios con Git

## **Entregables**

* Código fuente (link del repositorio en Github)
* Readme del proyecto
* Json de permisos para Bucket S3
* Link de deploy (opcional)