**Análisis de Requerimientos – Product Baclog**

Rastreo de paquetes en tiempo real (Prioridad alta, Importancia alta)

* Rastrear la ubicación exacta de los paquetes en tiempo real.
* Notificar al cliente el estado del paquete en tiempo real.

Historia

* **Como** cliente **quiero** poder verificar la ubicación exacta de mis paquetes en tiempo real **para** estar seguro de la compra que hice.

Historias de envío anteriores (Prioridad media, Importancia media)

* Acceder al historial completo de los envíos anteriores.
* Ver el número total de paquetes enviados.

Historia

* **Como** cliente **quiero** recibir notificaciones de estado del paquete real **para** no estar siempre pendiente de la APP

Comunicación en tiempo real (Prioridad baja, Importancia baja)

* Enviar y recibir mensajes para comunicarse directamente con el conductor.

Historia

* **Como** cliente **quiero** recibir y enviar mensajes **para** comunicarme directamente con el conductor

**Arquitectura**

Se utiliza una Arquitectura de “The Clean Arquitectur” que nos permite ser escalables y mantener un código limpio y ordenado lo cual significa trabajar con buenas prácticas como solid.

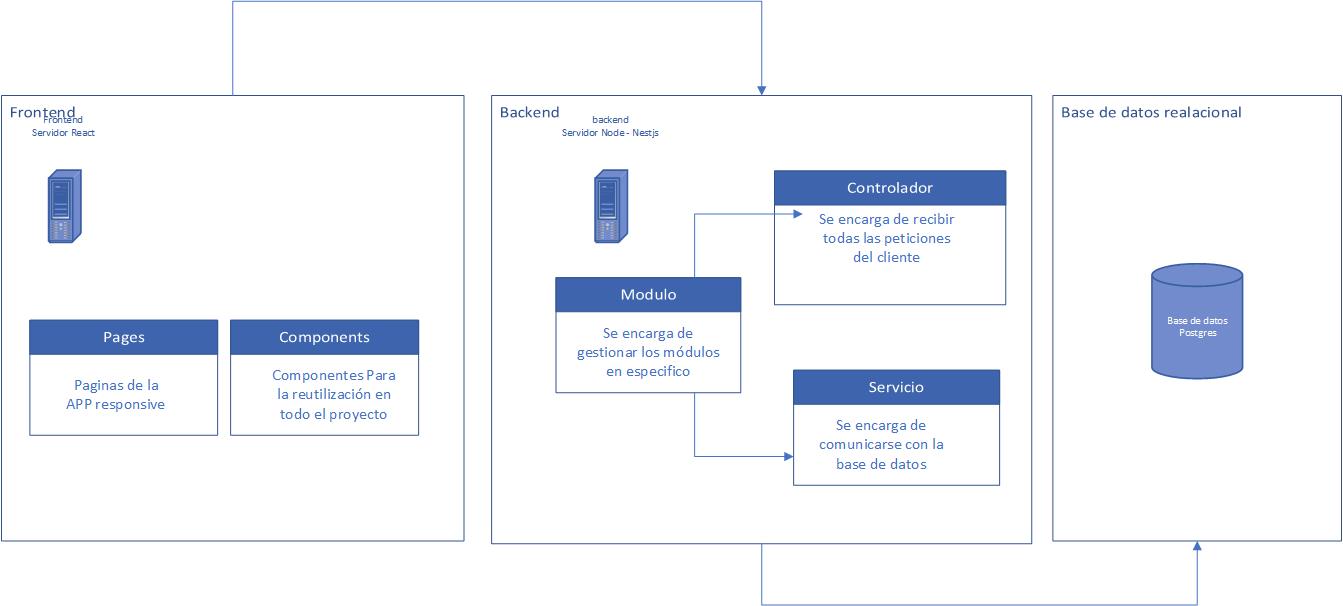
Backend( Nestjs)

Nos permite construir Api progresivas y de alto nivel y su implementación es su fácil se base en tres fundamentos importantes.

**Modulo**. Los módulos son como islas aisladas una de cada de ellas pero cada una de ellas cumple un propósito en general, cada modulo tienes sus propios controladores y servicios

**Controlador.** Son las encargadas de gestionar las peticiones PUT, POST, PATH, DELETE

**Servicios.** Nos permite definir toda la lógica y conectarse con los inyecciones de base de datos



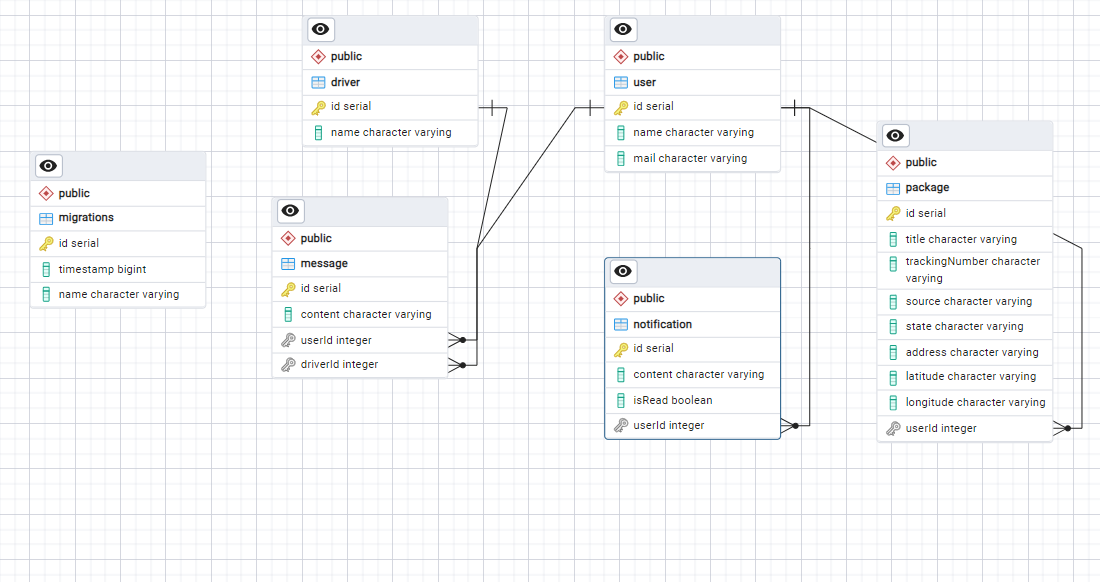
**Ventajas**

La arquitectura de aplicaciones web es un esquema que define cómo interactúan los componentes de una aplicación web. Puede ser desde una relación simple entre cliente y servidor hasta interrelaciones complejas entre múltiples servidores backend, equilibradores de carga, pasarelas API y frontends orientados al usuario.

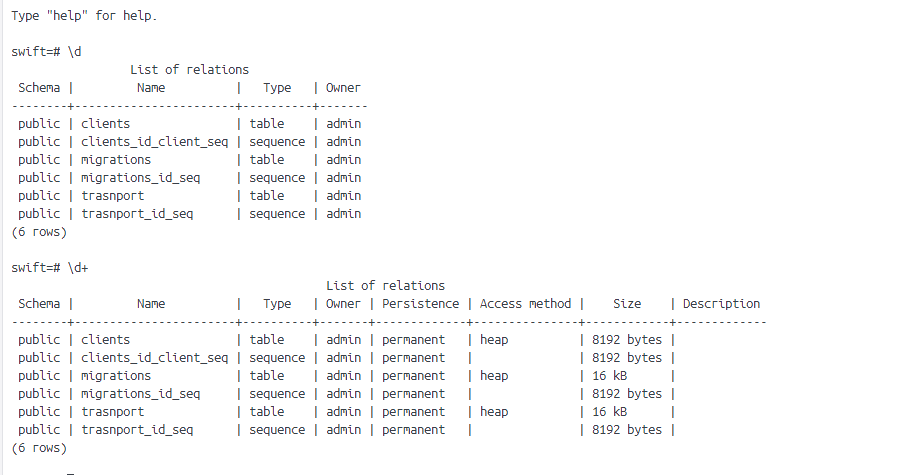
* **Flexibilidad**: Una arquitectura adecuada de la aplicación web debe ser lo suficientemente flexible como para adaptarse a las necesidades cambiantes de tu negocio.

**Modelo de la base de datos (Prototipo)**

La base de datos utilizada para el proyecto es PostgreSQL que es relacional que nos permite mantener la gestión de solicitudes de envío, seguimiento y control en tiempo real



Estructura de la base PostgreSQL

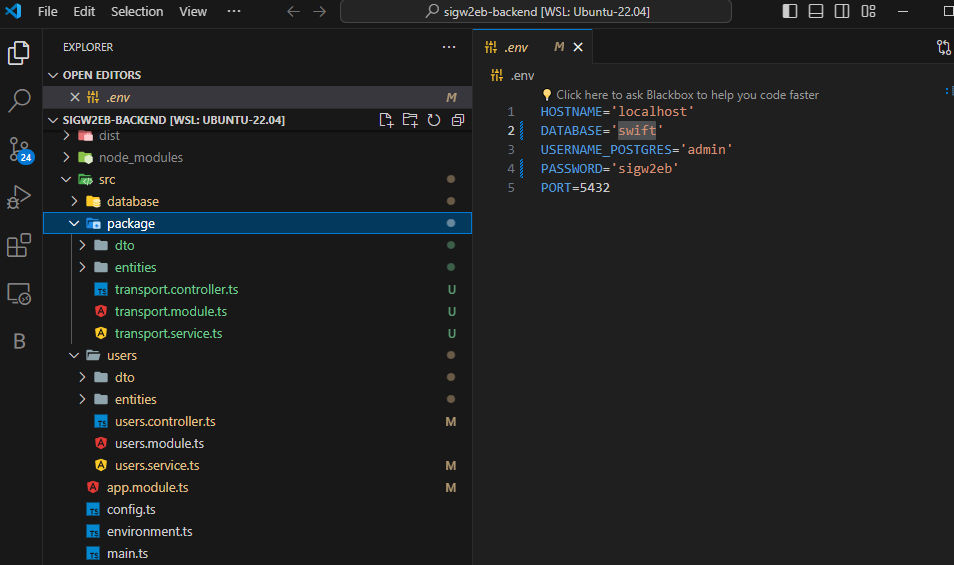


**Lenguajes de programación**

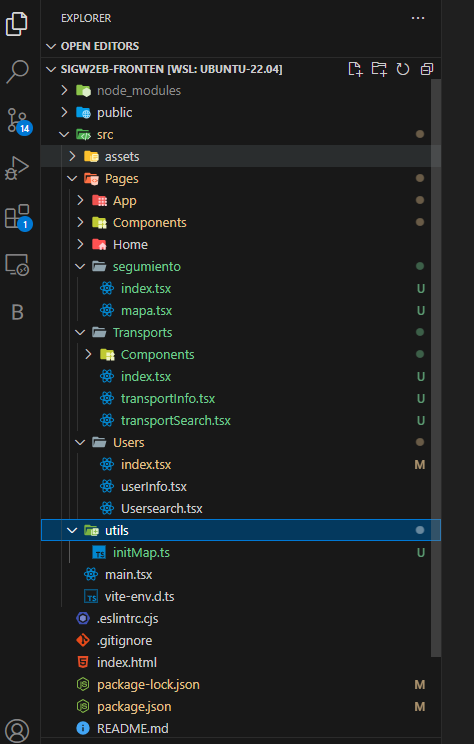
Se eligió utilizar NestJS como backend y React en el frontend para el proyecto de seguimiento de paquetes. NestJS es un marco de aplicaciones web de Node.js que ofrece una arquitectura limpia y eficiente, ideal para aplicaciones escalables y mantenibles.

Por otro lado, React es una biblioteca de JavaScript de código abierto diseñada para crear interfaces de usuario interactivas y reutilizables, lo que la convierte en una excelente opción para el frontend.

Estructura de carpetas Backend utilizando programación orienta a objetos con typescript

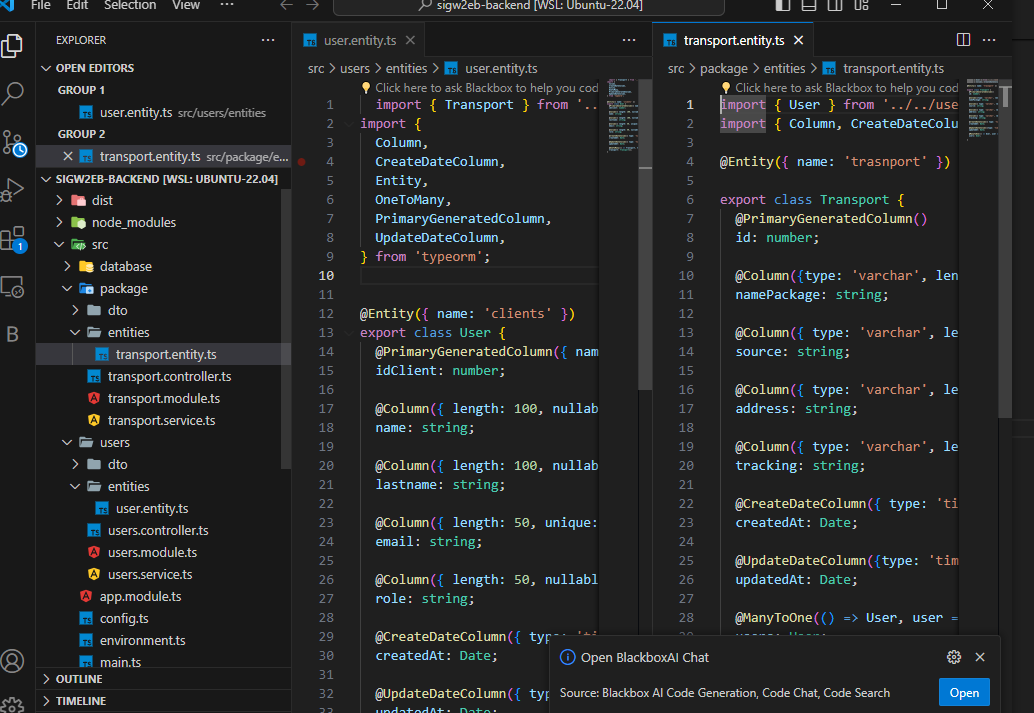


Estructura de carpetas Frontend utilizando typescript

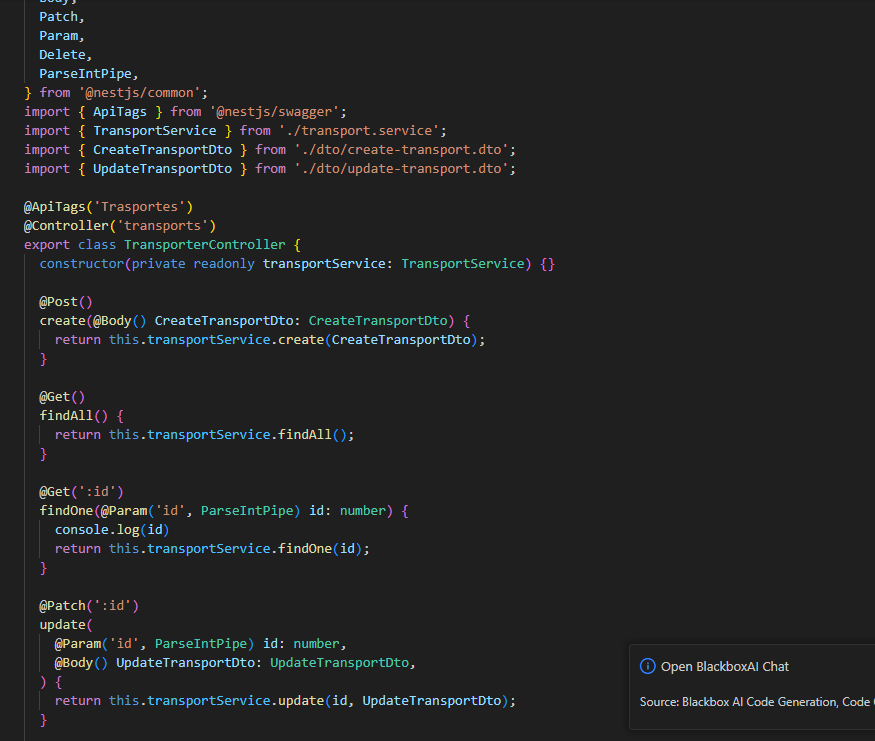


**Código Fuente Api con Nets JS – App web responsive**

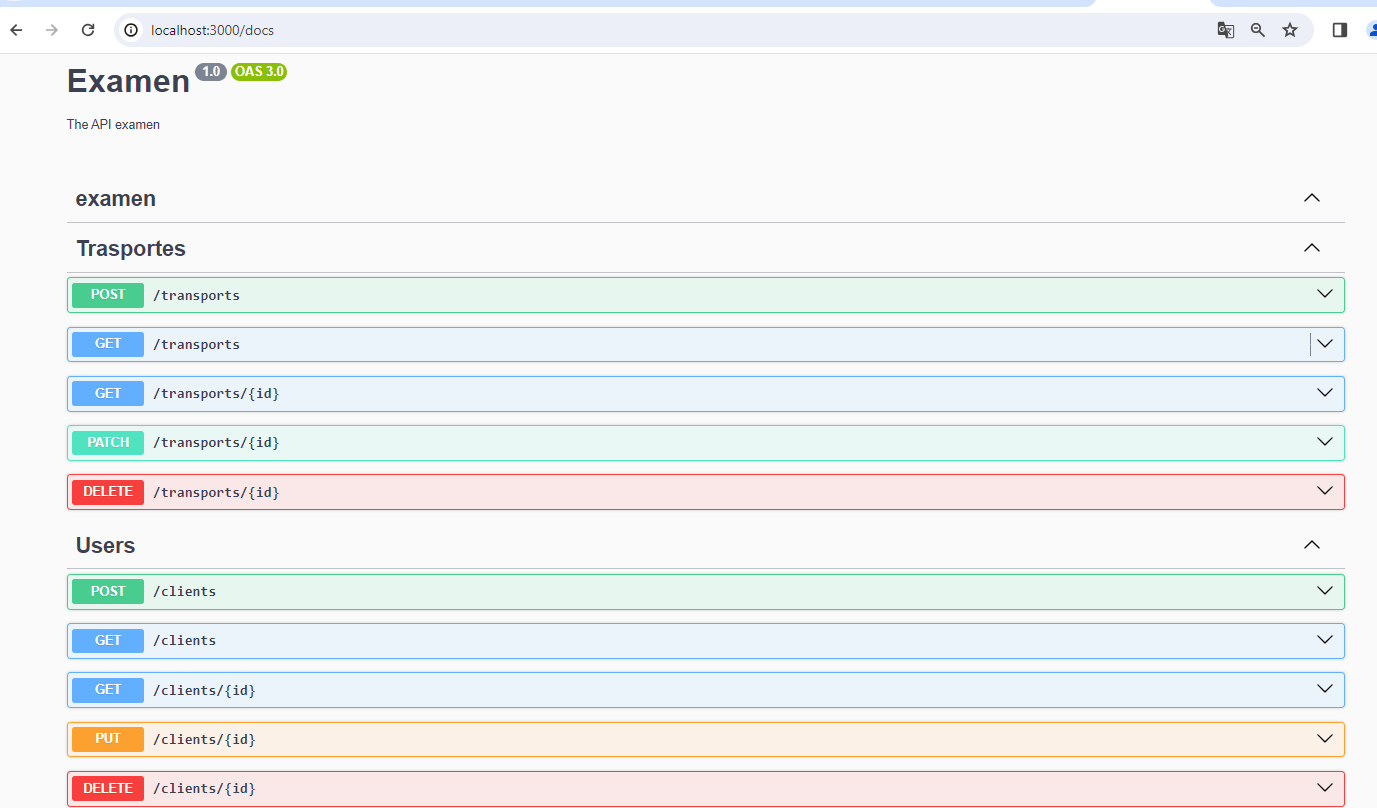
**Entidades para la construcción las tablas con TypeOrm**

****

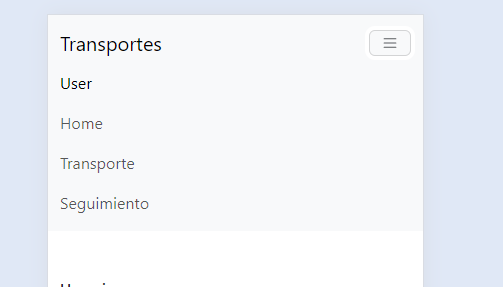
**Controlador Transporte**

****

**Documentación API**

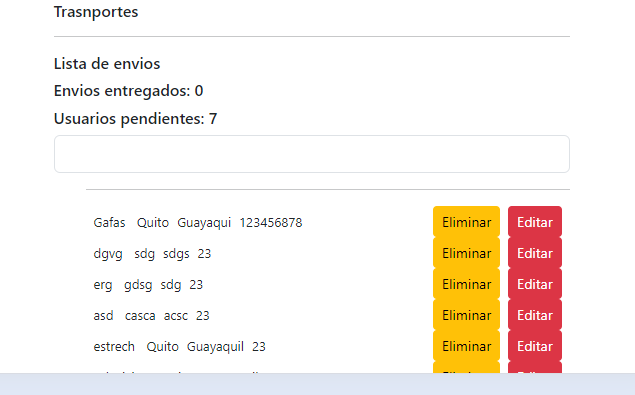
****

**Pages de la aplicación**

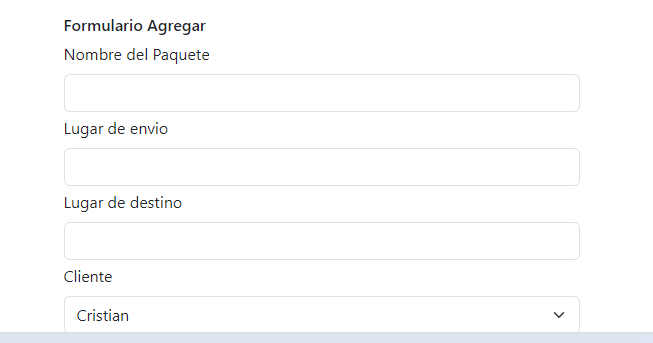
****

**Page de transporte**

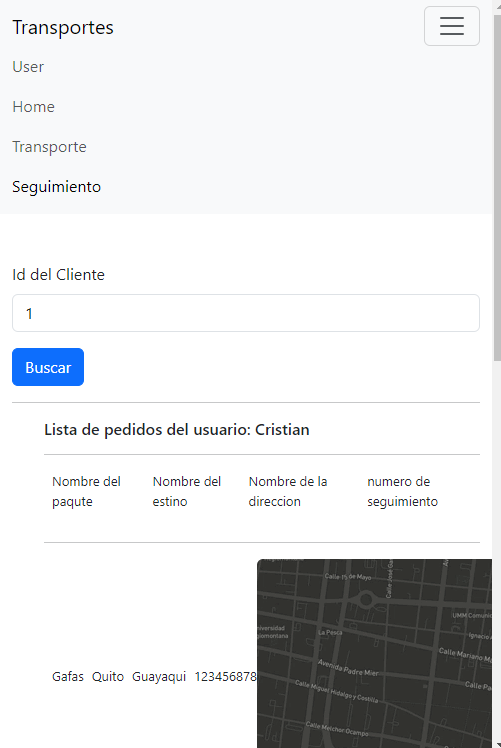
**Lista de envios**

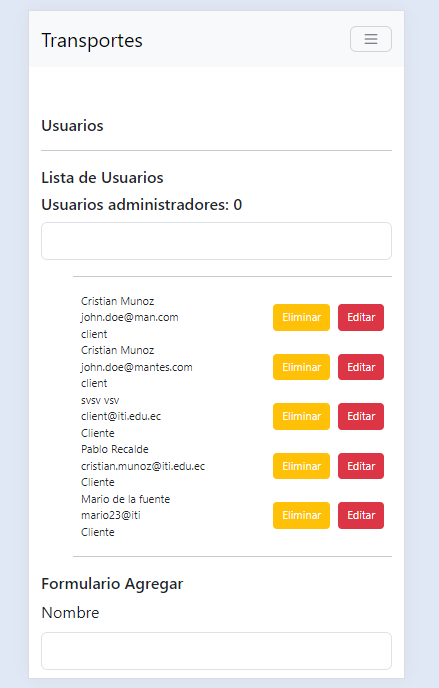
****

**Agregar un Envio**

****

**Pagina de seguimiento por usuario y mapa – se neciesta token en maxlop para esta actividad**

****

****