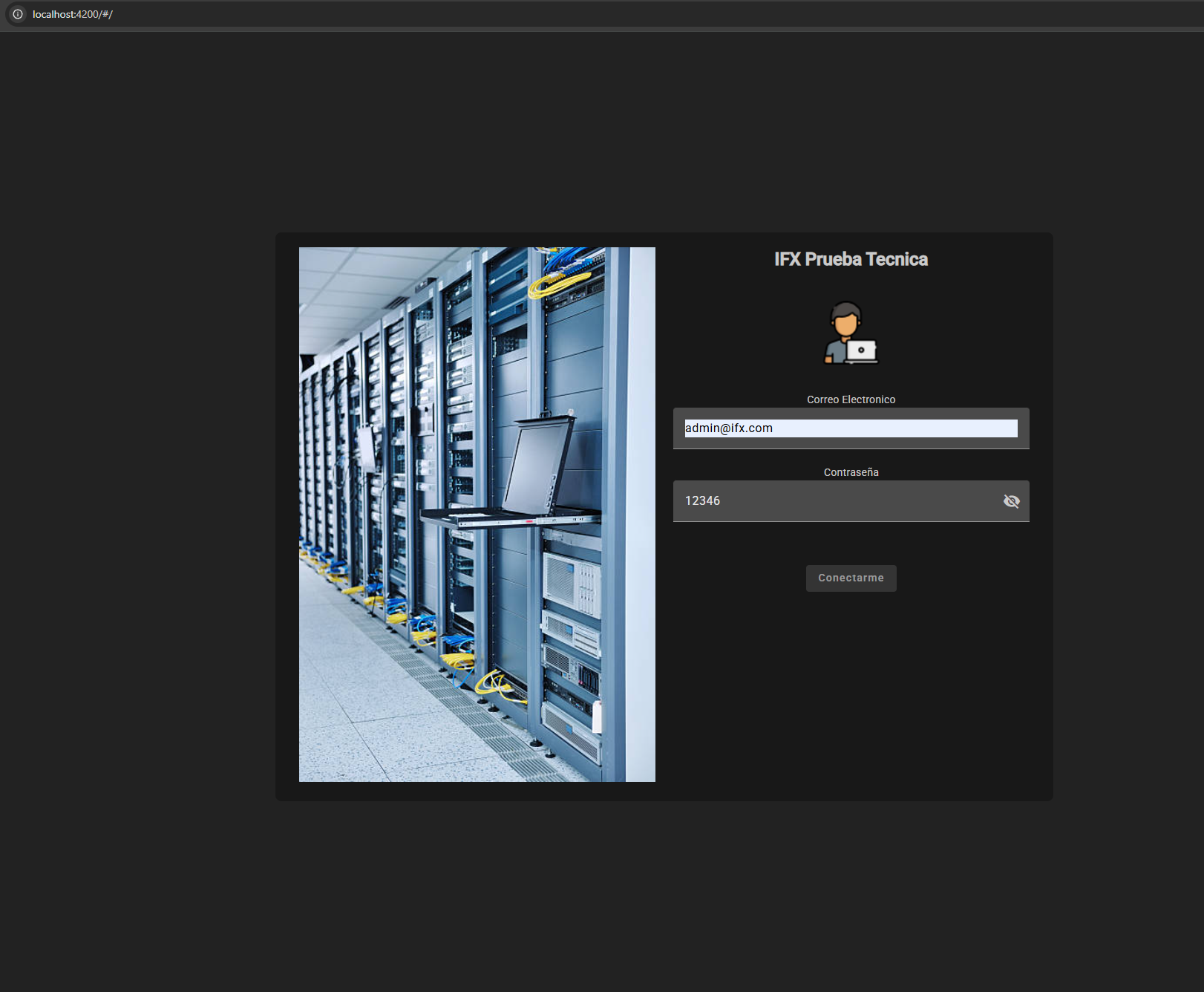
**Documentación FrontEnd (Angular 18)**

URL: http://localhost:4200/#/

Se agrega un login para el proyecto aprovechando la implementación del TokenJwr, proyecto basado en estilos .SCSS, basado en mi experiencia en angular y su suite Angular Material, aprovechando el concepto de tematización.

Aplicación basada en los mejores conceptos de programación Client, Maneja LazyLoading “Carga diferida”, estilos reutilizables, iconos en modo SVG para no saturar peso, se agrega opción de ver o no la contraseña



Nota “Se crearon 2 perfiles diferentes para poder ingresar, cada uno tiene su validacion”

Perfil: Administrador (Permite editar / eliminar / crear)

Email: [admin@ifx.com](mailto:admin@ifx.com)

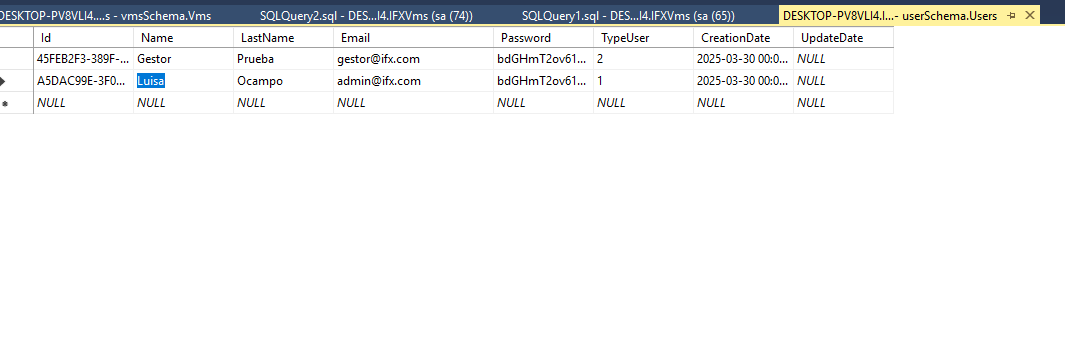
Contraseña: 123456

Perfil: Gestor (No permite editar / eliminar / crear)

Email: [gestor@ifx.com](mailto:gestor@ifx.com)

Contraseña: 123456

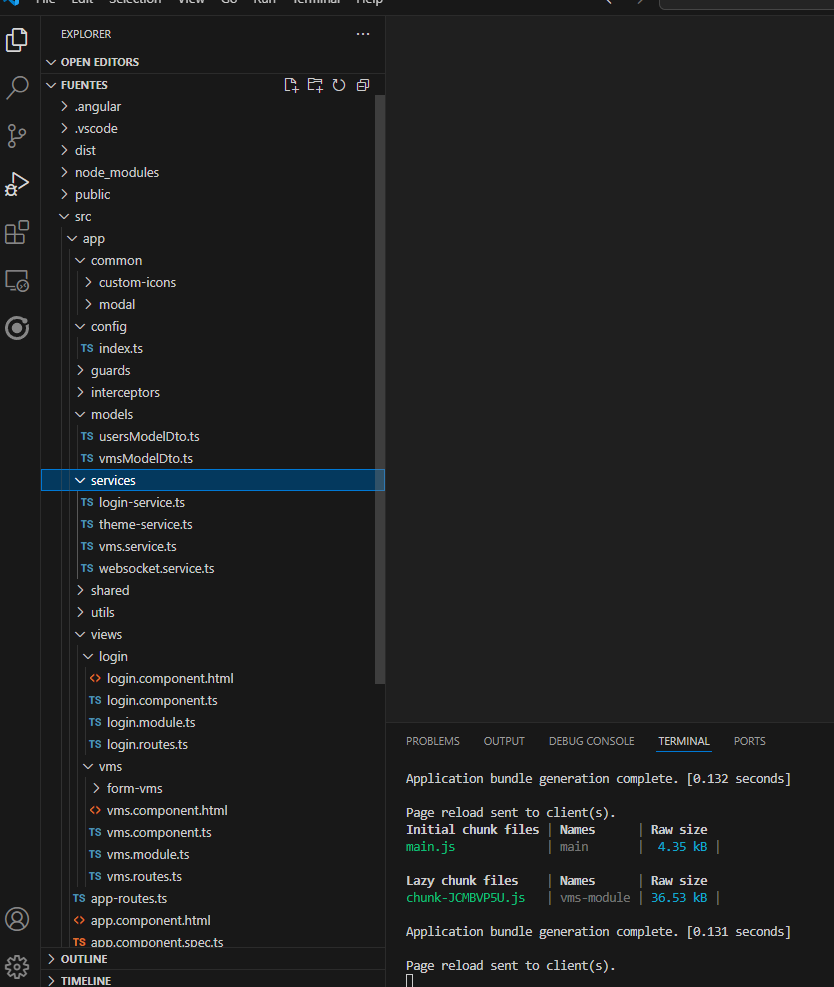




Perfil Gestor – Restringidas acciones de edición, creación o eliminación

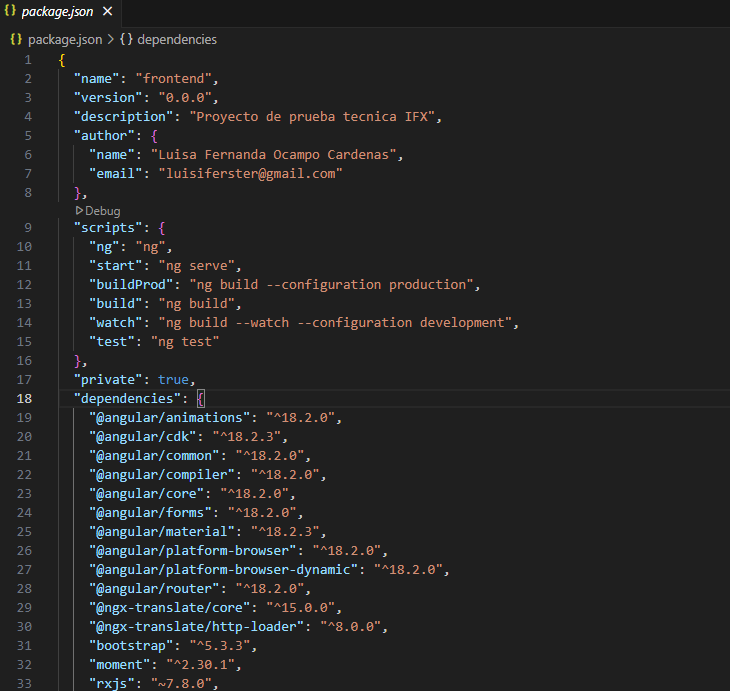


Estructura



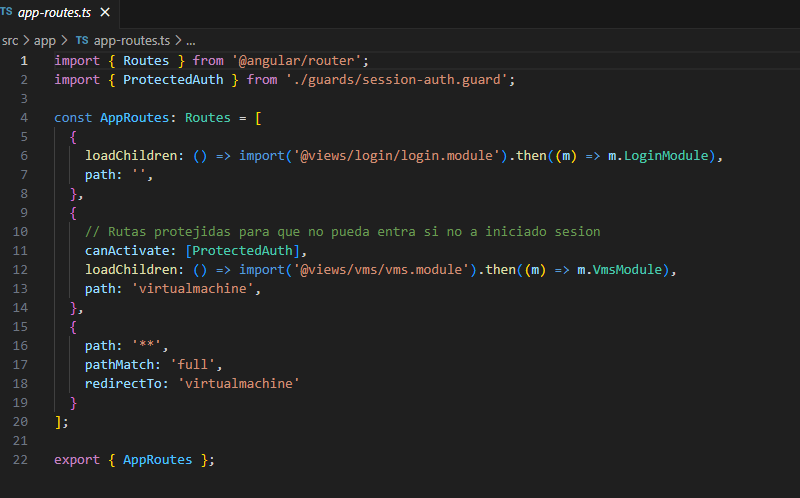
* Package.json

Configuración del proyecto, despliegues en modo producción y modo de lint con el cual se puede verificar reglas de sintaxis tanto en SCSS como en TS.



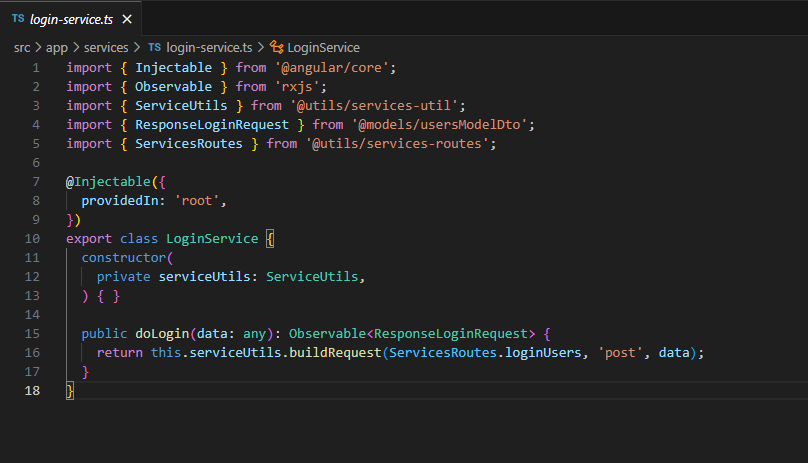
* Routing Login

Se evidencia la implementación de carga diferida



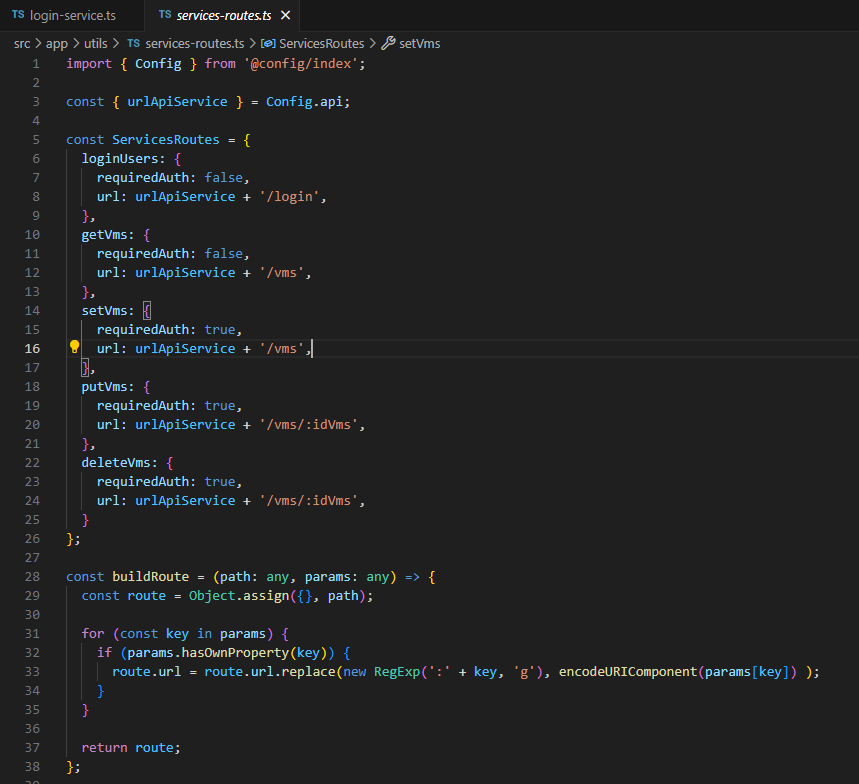
* Ejemplo servicio login-service.ts

Ejemplo de consumo de un servicio con Injectable, y Observables para el correspondiente llamado al servicio API REST .Net Core 3.1



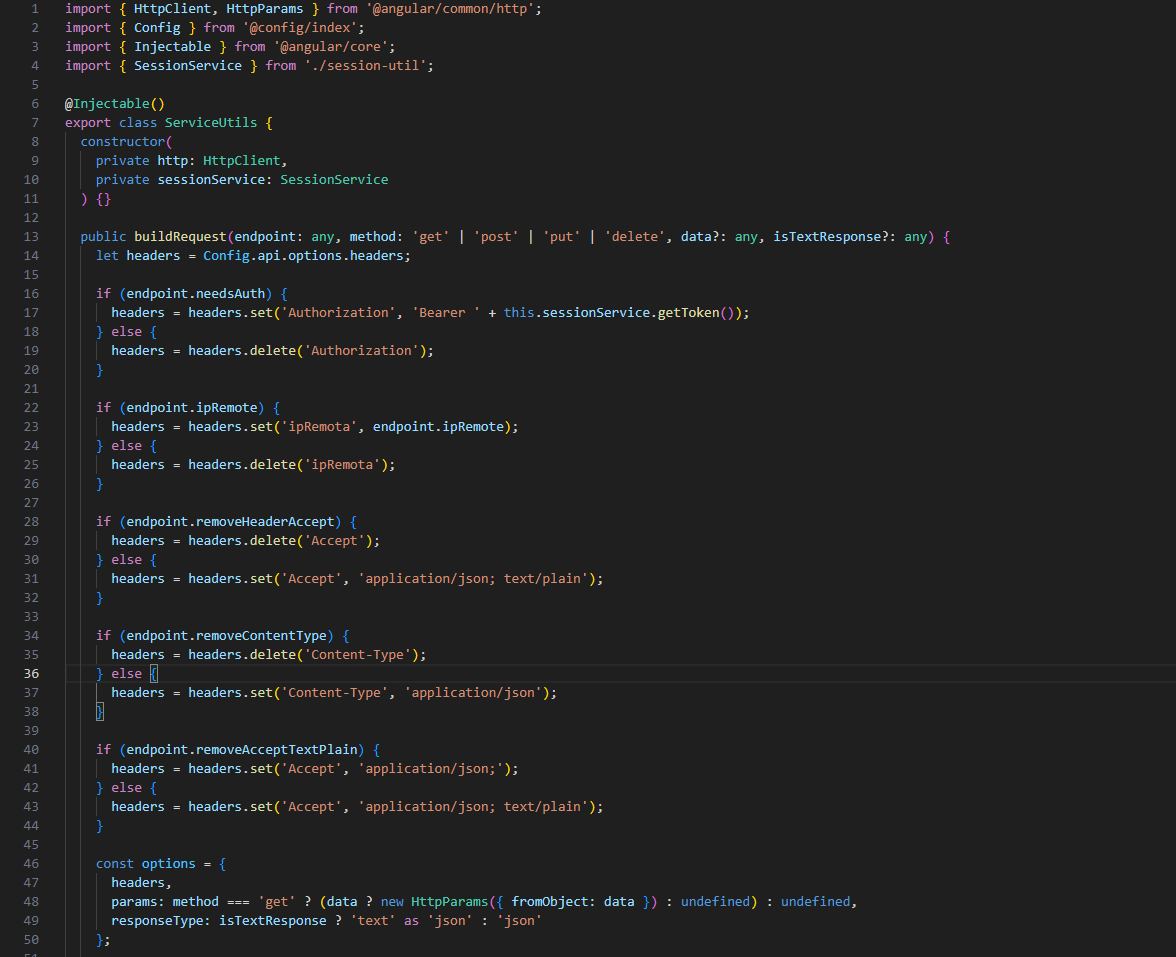
* Ejemplo service-routes.ts

Se evidencia la clase donde se almacenan todas las peticiones creadas en el API, junto con una opción reutilizable llamada requiredAuth la cual es por si se requiere envió de token en las cabeceras Http



* Services-utils.ts

Utilidad reutilizable la cual arma todos los llamados al servicio, y que según su lógica organiza los arrays para peticiones GET, SET, POST, DELETE, y las cabeceras necesarias de cada uno



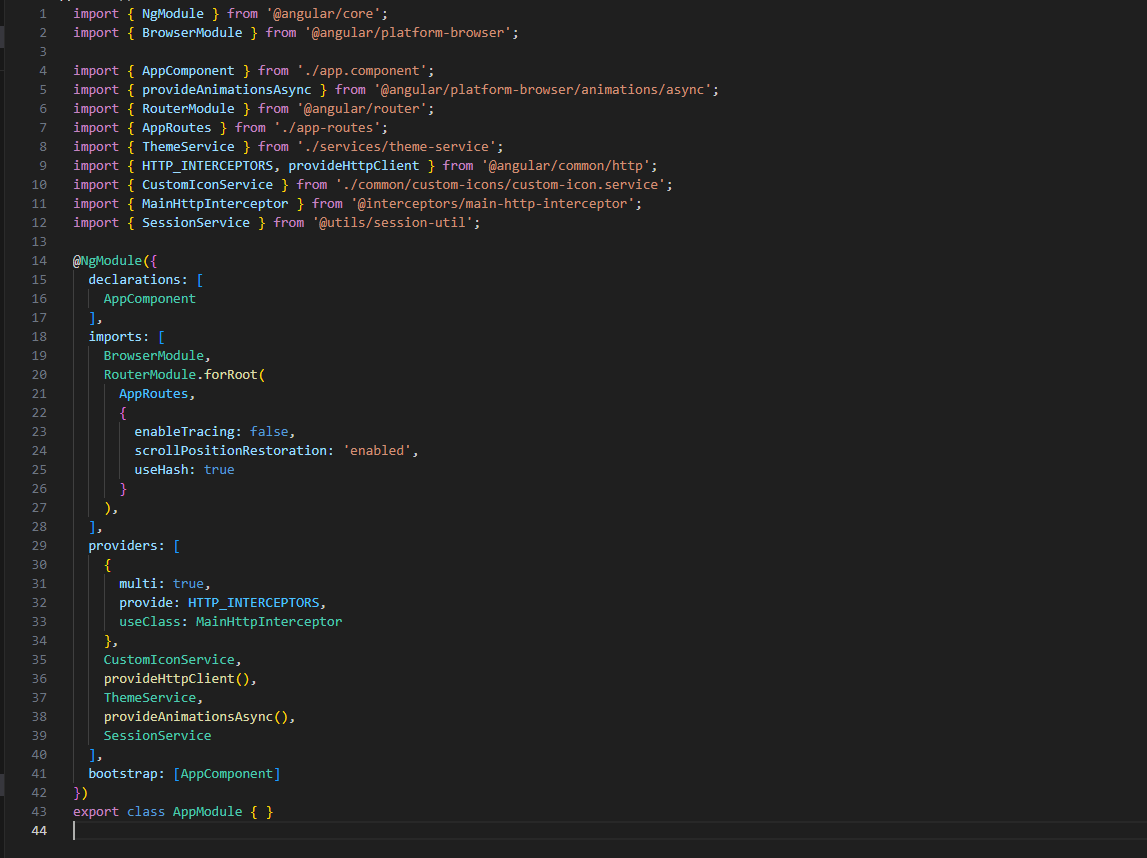
* Sesión-service.ts

Esta clase es de importancia la cual almacena a las variables local.storage y session.storage del navegador para con ello poder guardar datos globales como lo puede ser el token que iría en las cabeceras cuando se requiera



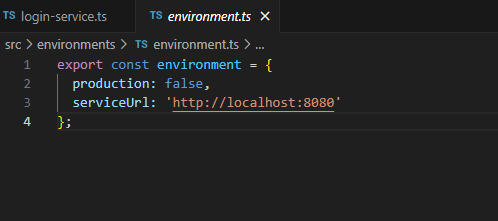
* app.module.ts

Modulo principal del proyecto el cual carga los módulos y que se utilizan en todo el proyecto, como lo puede ser el HttpClientModule que en la versión Angular 18 es provideHttpClient(), o el llamado a los iconos SVG que se usa, adicional para interceptores.

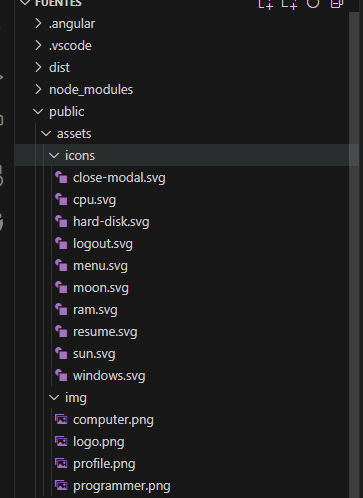


* enviroment.ts

clase la cual tiene le ruta del api que consume



* **Assets**
* Iconos



* Dashboard o Home

Sección que accede luego de ingresar al login, como se observa se utilizan todos los temas de Angular material para que la experiencia visual del usuario sea lo mejor posible, se destaca en esto la utilizacon de theming de angular logrando así una experiencia visual cambiante de colores.

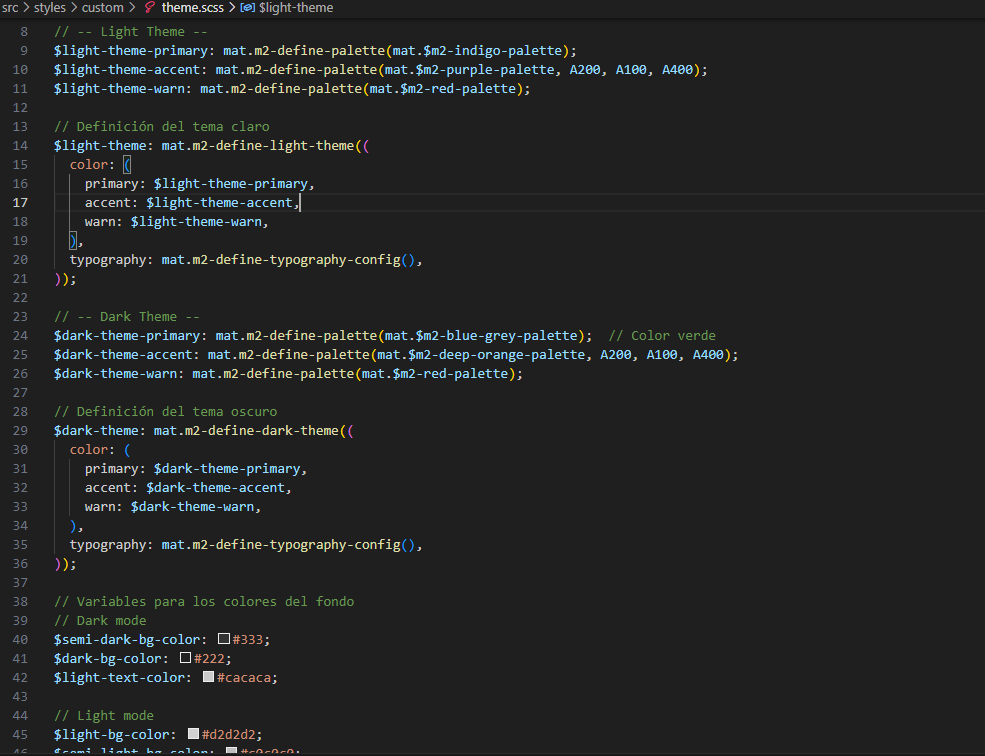




Como se puede ver agregue 2 temas con su correspondiente previsualización de cómo podría quedar

* Theme.scss

Basado en la documentación de la suite Angular material se agregan 2 temas que podrían ser más cambiantes a través de observables y etiquetas [ngClass



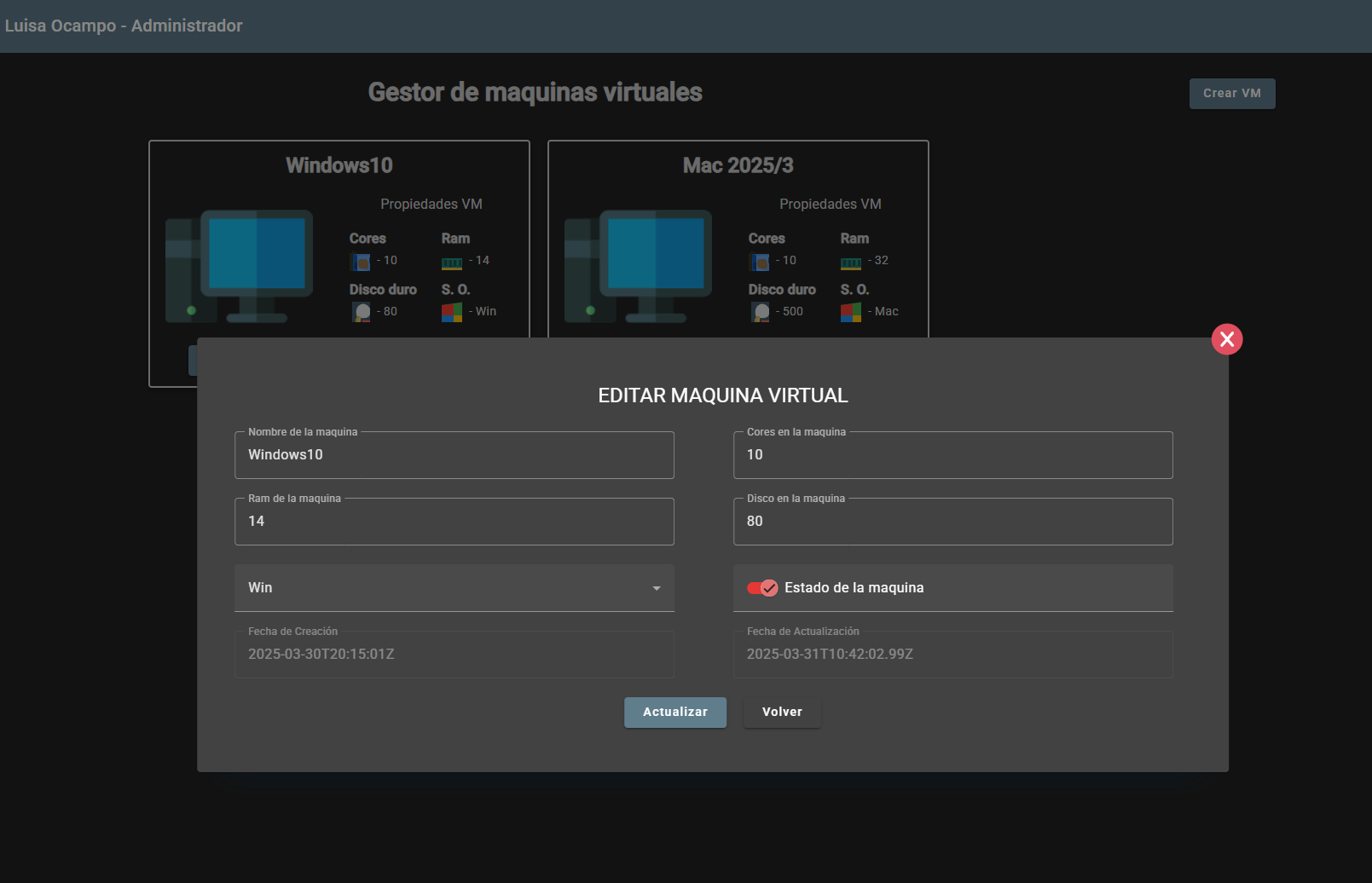
* Componente Vms

Explicación componente Vms, en el cual se implementa en CRUD del api REST, y mejoras visuales por iconos svg para mejorar la experiencia del usuario, así como imágenes que resalten en este caso la integración con máquinas virtuales usando Bootstrap.

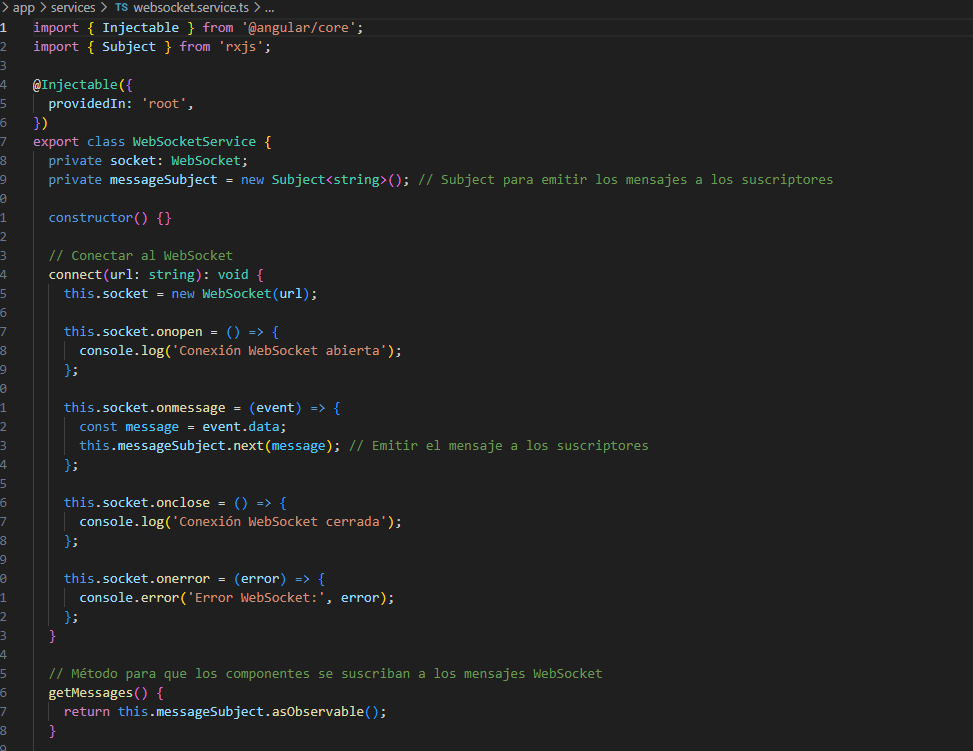


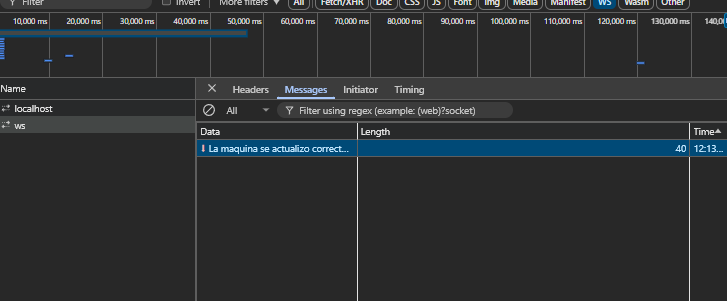
* Editar Máquinas virtuales

Se crea un componente de tipo formulario el cual usa formularios reactivos de angular para validaciones de campos y demás, se agregan los campos necesarios



* Se implementa la funcionalidad de WebSocket, esto para poder manejar en un canal abierto entre el cliente y el servidor, lo que permite cada vez que se emite un cambio relevante, enviar desde el servicio un mensaje para que el o los clientes sepan que deben actualizar registros, en este caso las máquinas virtuales





Se realiza la actualización en el perfil Administrador y el perfil Gestor en este caso recibe también el mensaje y actualiza la lista

