**Exposición del Frontend de Project Pets**

**1. Introducción al Frontend**

Bienvenidos. En esta presentación, explicaré la arquitectura y el funcionamiento del **frontend** de **Project Pets**, una plataforma de gestión para clínicas veterinarias.

El **frontend** ha sido desarrollado con **React y TypeScript**, utilizando una arquitectura **modular** y **escalable**, con una interfaz **intuitiva** para diferentes roles:

* **Administrador**: Gestiona usuarios, inventario y servicios.
* **Veterinario**: Atiende consultas y actualiza historias clínicas.
* **Cliente**: Administra sus mascotas y solicita servicios.

También explicaré **cómo interactúa con el backend** y las tecnologías clave que lo soportan.

**2. Tecnologías Utilizadas en el Frontend**

El stack tecnológico del frontend incluye:

1. **React 18 + TypeScript**
2. **Vite** → Herramienta de compilación rápida.
3. **React Router** → Manejo de rutas y navegación.
4. **Axios** → Conexión con el backend a través de API REST.
5. **JWT Storage** → Manejo de autenticación con Tokens.

**3. Arquitectura y Organización del Proyecto**

El código del frontend se organiza en carpetas, siguiendo una estructura **modular** para facilitar la escalabilidad y mantenimiento.

📂 **src/ (Codigo del proyecto)**

└── 📂 **assets/** *(Imágenes, logos, íconos, estilos globales)*  
├── 📂 **components/** *(Botones, modales, tablas reutilizables...)*  
├── 📂 **pages/** *(Pantallas principales, divididas por rol)*

├── 📂 **routes/** *(Definición de rutas y navegación con React Router para la autenticación de usuarios)*  
├── 📂 **services/** *(Funciones para llamadas HTTP al backend)*

├── 📂 **utils/** *(Funciones auxiliares, como formateo de fechas)*  
├── 📂 **context/** *(Context API para autenticación y estado global)*

Esto permite **separar responsabilidades**, haciendo que cada módulo sea independiente y fácil de mantener.

**4. Manejo de Rutas y Navegación (React Router)**

El sistema de navegación está basado en **React Router**, lo que permite:

* **Navegación protegida** según el rol del usuario.
* **Rutas dinámicas** con parámetros (ej. /mascotas/:id).
* **Redirecciones automáticas** al iniciar sesión según el tipo de usuario.

Ejemplo de rutas definidas:

* **/login** → Página de autenticación.
* **/admin/usuarios** → Gestión de usuarios (solo ADMIN).
* **/cliente/mascotas** → Listado de mascotas (solo CLIENTES).
* **/veterinario/servicios** → Servicios pendientes (solo VETERINARIOS).

Un **componente de protección de rutas** (PrivateRoute) se encarga de verificar que el usuario tenga acceso antes de cargar la página.

**5. Manejo de Autenticación y Seguridad (JWT)**

La autenticación se basa en **JSON Web Tokens (JWT)**.

1️⃣ El usuario ingresa credenciales en /login.  
2️⃣ El frontend envía una petición al backend /api/auth/login.  
3️⃣ El backend responde con un **token JWT**.  
4️⃣ Se almacena en localStorage y se usa en cada petición futura.  
5️⃣ Si el token expira, se redirige automáticamente al Login

