**Herramientas:**

**Backend: Python (framework Django con Django-ninja para el manejo de APIs)**

**DB: PostgreSQL**

**Frontend: React**

**Sprint 1 – Configuración y cimientos del proyecto**

**Backend (Python)**

* Definir arquitectura básica (API FIRST - RESTAPI)
* Crear entorno virtual y estructura base del proyecto (venv)
* Instalación de librerías a utilizar
* Configuración de db en Python
* Seleccionar framework (Django - ninja)
* Configurar base de datos inicial (PostgreSQL)
* Crear algunos modelos iniciales y migrate la db
* Crear primer endpoint de prueba (/health, /docs y /redocs)
* Documentar cómo levantar el proyecto

**Frontend (React)**

* Crear proyecto React
* Estructurar carpetas (components, pages, services, etc.)
* Configurar rutas
* Maquetar páginas básicas (solo estructura)
* Definir y aplicar sistema de estilos (css, typography y paleta de colores según exigencias de pablo)

**Sprint 2 – Autenticación y layout base**

**Backend**

* Implementar registro y login con JWT
* Crear modelos de usuario y migraciones
* Endpoints: /register (no creo que se utilice pero lo voy a implementar), /login, /me
* Proteger rutas con autenticación
* Escribir pruebas básicas de endpoints

**Frontend**

* Crear formulario de login y registro
* Integrar login con el backend
* Manejo de jwt en localStorage o cookies
* Crear layout general con navegación lateral o topbar (diseño según lo comentado por pablo)
* Redireccionar usuarios no autenticados

**Sprint 3 – CRUD inicial**

**Backend**

* Definir y crear un modelo clave del sistema (ej: "Documentos", "Usuarios", "Perfiles", etc.)
* Crear endpoints: GET, POST, PUT, DELETE
* Validaciones de entrada (según el framework)
* Añadir paginación y ordenamiento

**Frontend**

* Crear lista principal (ej: lista de tareas o proyectos)
* Crear formularios para crear/editar ítems
* Integrar con el backend (CRUD)
* Validaciones en frontend

**Sprint 4 – Optimización y mejoras UX**

**Backend**

* Mejorar estructura modular del proyecto
* Añadir permisos por rol (si aplica)
* Optimizar queries o filtros
* Documentar API con Swagger/OpenAPI

**Frontend**

* Añadir feedback visual (toasts, loading states)
* Mejorar accesibilidad básica
* Componentes reutilizables (inputs, botones)
* Proteger rutas (Redirect si no está autenticado)