**App Design Document – CleanFix Web Platform.**

## Índice – App Design Document: CleanFix

1. Visión General del Proyecto.
2. Funcionalidades Principales.
3. Arquitectura Técnica.
4. Integración de Inteligencia Artificial.
5. Métricas y Visualización.
6. Seguridad y Configuración.
7. Pruebas y Calidad.
8. Roadmap de Desarrollo.

### 1. Visión General del Proyecto

**Nombre de la aplicación:** CleanFix.

**Tipo de solución:** Plataforma web empresarial.

**Propósito:** Optimizar la gestión de servicios de mantenimiento y limpieza mediante una solución digital que centraliza solicitudes, incidencias, tareas, facturación y comunicación.

**Objetivos estratégicos:**

* Reducir tiempos de respuesta ante incidencias.
* Mejorar la trazabilidad de tareas y solicitudes.
* Automatizar procesos administrativos como facturación.
* Ofrecer una experiencia moderna y eficiente a clientes y operarios.

### 2. Funcionalidades Principales.

#### 2.1 Gestión de Solicitudes.

* Formulario dinámico para crear solicitudes.
* Visualización en tabla con filtros por estado, cliente, fecha.
* Acceso a detalles, historial y comentarios.

#### 2.2 Gestión de Incidencias.

* Registro de incidencias con prioridad, tipo y estado.
* Asociación a solicitudes o tareas específicas.
* Seguimiento y resolución por parte del equipo técnico.

#### 2.3 Tareas Completadas.

* Listado paginado de tareas finalizadas.
* Acceso a evidencias (imágenes, comentarios).
* Filtros por técnico, cliente, fecha y tipo de servicio.

#### 2.4 Facturación.

* Generación automática de facturas en PDF.
* Cálculo de IVA, desglose por servicio.
* Envío por correo electrónico al cliente.
* Registro de facturas emitidas y estado de pago.

#### 2.5 Chat Asistido por IA.

* Asistente conversacional integrado con Azure OpenAI.
* Capacidad para responder preguntas frecuentes.
* Generación de acciones automatizadas (ej. crear factura, consultar estado).

#### 2.6 Panel de Administración.

* Dashboard con KPIs: número de solicitudes, incidencias abiertas, tareas completadas.
* Gestión de usuarios y roles.
* Configuración de parámetros del sistema.

### 3. Arquitectura Técnica.

#### 3.1 Frontend.

* **Framework:** Angular.
* **Estilos:** Tailwind CSS + Prettier.
* **Componentes clave:**
  + ServiceCard, StatsSection, ChatBox, LoginForm, Snackbar, WhyUsSection.
* **Características:**
  + Diseño responsivo.
  + Animaciones suaves.
  + Navegación intuitiva.

#### 3.2 Backend.

* **Lenguaje:** C# (.NET).
* **Arquitectura:** Clean Architecture:
  + Capas: Domain, Application, Infrastructure.
* **Patrones utilizados:**
  + CQRS (Command Query Responsibility Segregation).
  + Mediator Pattern.
* **Base de datos:** Azure SQL:
  + Migraciones con Entity Framework.
  + Uso de claves INT y GUID según contexto.

#### 3.3 Seguridad.

* Autenticación mediante formulario de login.
* Validación de campos y control de sesión.
* Configuración segura de variables de entorno.
* Envío de correos mediante SMTP (Gmail).
* Webhooks para integración con otros sistemas.

### 4. Integración de Inteligencia Artificial.

#### 4.1 Asistente IA.

* Basado en Azure OpenAI.
* Capacidad para comprender lenguaje natural.
* Generación de respuestas y acciones automatizadas.

#### 4.2 Plugins IA.

* DBPlugin: Acceso a datos estructurados para enriquecer respuestas.
* AssistantService: Orquestación de peticiones y generación de contenido.
* ChatBoxIAController: Controlador que gestiona la lógica conversacional.

### 5. Métricas y Visualización.

* Componente AppStatsSection para mostrar KPIs.
* DTOs personalizados para paginación y visualización:
* PaginatedDataDto, SolicitationBriefDto, IncidenceDto, InvoiceDto.
* Visualización de estadísticas en tiempo real.

### 6. Pruebas y Calidad.

* Pruebas unitarias en componentes visuales y servicios.
* Validaciones automáticas con Husky y lint-staged.
* Generación de datos ficticios para entornos de desarrollo.
* Validación de estructura de código antes de cada commit.

### 7. Configuración y Entorno.

* .gitignore para exclusión de archivos sensibles.
* Variables de entorno diferenciadas por entorno (dev, prod).
* Configuración de SMTP para envío de correos.
* Control de versiones con Git.

### 8. Roadmap de Desarrollo.

El desarrollo de CleanFix se ha organizado en seis fases progresivas. Primero se configuró el entorno técnico y el repositorio del proyecto. Luego se construyó el backend con arquitectura limpia y patrones como CQRS y Mediator, seguido por el desarrollo del frontend en Angular con componentes visuales reutilizables.

Posteriormente se integró el asistente de inteligencia artificial con Azure OpenAI, capaz de generar facturas y responder consultas. Se optimizó la experiencia de usuario con mejoras visuales, paginación y filtros. Actualmente, el proyecto se encuentra en la fase final de pruebas, documentación y preparación para el despliegue en producción.