Documentación Profesional del Proyecto CONCRETEWARE

Autor: Concreteware  
Fecha: Julio de 2025

# Concreteware - Documentación Profesional Del Proyecto

# Índice

1. Descripción General

2. Arquitectura y Tecnologías

3. Estructura del Proyecto

4. Modelos de Datos

5. Funcionalidades Principales

6. API REST - Endpoints Backend

7. Autenticación y Seguridad

8. Configuración y Despliegue

9. Troubleshooting y Buenas Prácticas

# 10. Créditos y Contacto

# 1. Descripción General

Concreteware es una plataforma tecnológica integral para la gestión de plantas de concreto premezclado. Permite administrar clientes, obras, productos, vehículos, conductores y pedidos, centralizando la logística y operación de la empresa. El sistema está compuesto por un frontend web responsivo y un backend robusto, integrados mediante una API REST y autenticación segura con Firebase.

# 2. Arquitectura Y Tecnologías

* \*\*Frontend:\*\* React 18, TypeScript, Tailwind CSS, React Router, Axios, React Hook Form, Firebase Auth, SPA (Single Page Application)
* \*\*Backend:\*\* Java 21+, Spring Boot, Firebase Admin SDK, Firestore, Maven
* \*\*Autenticación:\*\* Firebase Authentication (JWT)
* \*\*Despliegue:\*\* Firebase Hosting (frontend), cualquier servidor Java (backend)
* \*\*Base de datos:\*\* Firestore (NoSQL, en Firebase)

# 3. Estructura Del Proyecto

* \*\*frontend/\*\*
* src/
* components/ (Componentes UI reutilizables)
* contexts/ (Contextos globales, ej. Auth)
* pages/ (Vistas principales: Login, Clientes, Conductores, Productos, Vehículos, Obras, Pedidos, Mapa, Setup)
* services/ (Lógica de conexión a la API)
* types/ (Modelos TypeScript)
* utils/ (Utilidades)
* \*\*backend/src/main/java/com/concreteware/\*\*
* administrador/controller/ (Controladores CRUD de entidades)
* auth/controller/ (Controladores de autenticación y plantas)
* core/model/ (Modelos de dominio)
* common/enums/ (Enumeraciones de estados y tipos)
* config/ (Configuración Spring y Firebase)

# 4. Modelos De Datos

## A. \*\*Usuario (abstracto)\*\*

* id: string
* nombre: string
* email: string
* telefono: string
* dni: string

## B. \*\*Cliente\*\* (extiende Usuario)

* nitEmpresa: string
* nombreEmpresa: string
* obrasAsociadasIds: string[]

## C. \*\*Conductor\*\* (extiende Usuario)

* licenciaConduccion: string
* activo: boolean

D. \*\*Producto\*\*

* idProducto: string
* nombreComercial: string
* descripcion: string
* precioBase: number

E. \*\*Vehiculo\*\*

* idVehiculo: string
* placa: string
* tipo: string
* capacidadM3: number
* activo: boolean
* ubicacionActual: { latitud, longitud, descripcion }
* idConductor: string

F. \*\*Obra\*\*

* idObra: string
* nombreObra: string
* direccion: string
* municipio: string
* ubicacion: { latitud, longitud, descripcion }
* estado: ACTIVA | COMPLETADA | SUSPENDIDA
* fechaInicio: string (YYYY-MM-DD)
* idCliente: string

G. \*\*Pedido\*\*

* idPedido: string
* idCliente: string
* idObra: string
* idVehiculo: string
* idConductor: string
* productos: ProductoPedido[]
* fechaEntrega: string
* horaEntrega: string
* estado: PENDIENTE | EN\_PROCESO | EN\_CAMINO | ENTREGADO | CANCELADO
* ubicacionActual: { latitud, longitud, descripcion }

H. \*\*ProductoPedido\*\*

* idProducto: string
* cantidadM3: number
* precioUnitario: number
* observaciones: string

I. \*\*Planta\*\*

* idPlanta: string
* nombre: string
* baseUrl: string

# 5. Funcionalidades Principales

* \*\*Autenticación segura\*\* (Firebase Auth, JWT)
* \*\*Gestión de plantas\*\* (selección y persistencia de planta)
* \*\*CRUD de Clientes\*\* (alta, baja, modificación, listado, búsqueda)
* \*\*CRUD de Conductores\*\* (incluye estado y licencia)
* \*\*CRUD de Productos\*\* (catálogo, precios, descripción)
* \*\*CRUD de Vehículos\*\* (capacidad, estado, asignación de conductor)
* \*\*CRUD de Obras\*\* (por cliente, estado, ubicación)
* \*\*CRUD de Pedidos\*\* (multi-producto, asignación de conductor y vehículo, estados)
* \*\*Mapa de ubicaciones\*\* (visualización de obras, vehículos y pedidos)
* \*\*Búsqueda y filtrado en tablas\*\*
* \*\*Formularios con validación y feedback visual\*\*
* \*\*Notificaciones y manejo de errores\*\*
* \*\*Diseño responsivo y moderno\*\*

# 6. Api Rest - Endpoints Backend

## A. \*\*Autenticación y Plantas\*\*

* POST /{idPlanta}/auth/login (Verifica token Firebase y retorna usuario)
* POST /{idPlanta}/auth/register (Registra usuario en Firebase y Firestore)
* POST /{idPlanta}/auth/setup-existing (Configura usuario existente en Firestore)
* GET /{idPlanta}/auth/check-user/{email} (Verifica existencia de usuario)
* GET /api/plantas (Lista todas las plantas)

## B. \*\*Clientes\*\*

* POST /{idPlanta}/administrador/clientes (Crear cliente)
* GET /{idPlanta}/administrador/clientes (Listar clientes)
* GET /{idPlanta}/administrador/clientes/{id} (Obtener cliente por ID)
* PUT /{idPlanta}/administrador/clientes/{id} (Actualizar cliente)
* DELETE /{idPlanta}/administrador/clientes/{id} (Eliminar cliente)

## C. \*\*Conductores\*\*

* POST /{idPlanta}/administrador/conductores
* GET /{idPlanta}/administrador/conductores
* GET /{idPlanta}/administrador/conductores/{id}
* PUT /{idPlanta}/administrador/conductores/{id}
* DELETE /{idPlanta}/administrador/conductores/{id}

D. \*\*Productos\*\*

* POST /{idPlanta}/administrador/productos
* GET /{idPlanta}/administrador/productos
* GET /{idPlanta}/administrador/productos/{id}
* PUT /{idPlanta}/administrador/productos/{id}
* DELETE /{idPlanta}/administrador/productos/{id}

E. \*\*Vehículos\*\*

* POST /{idPlanta}/administrador/vehiculos
* GET /{idPlanta}/administrador/vehiculos
* GET /{idPlanta}/administrador/vehiculos/{id}
* PUT /{idPlanta}/administrador/vehiculos/{id}
* DELETE /{idPlanta}/administrador/vehiculos/{id}

F. \*\*Obras\*\*

* POST /{idPlanta}/administrador/obras
* GET /{idPlanta}/administrador/obras
* GET /{idPlanta}/administrador/obras/{id}
* GET /{idPlanta}/administrador/obras/cliente/{idCliente}
* PUT /{idPlanta}/administrador/obras/{id}
* DELETE /{idPlanta}/administrador/obras/{id}

G. \*\*Pedidos\*\*

* POST /{idPlanta}/administrador/pedidos
* GET /{idPlanta}/administrador/pedidos
* GET /{idPlanta}/administrador/pedidos/{id}
* PATCH /{idPlanta}/administrador/pedidos/{id}/estado?nuevoEstado=ESTADO
* PATCH /{idPlanta}/administrador/pedidos/{id}/conductor?idConductor=ID

# 7. Autenticación Y Seguridad

* \*\*Firebase Authentication:\*\* Login seguro, tokens JWT, integración con backend
* \*\*Roles y tipos de usuario:\*\* Control de acceso por tipoUsuario (admin, operador, etc.)
* \*\*CORS:\*\* Configurado en backend para permitir peticiones desde frontend
* \*\*Validación de formularios:\*\* En frontend y backend
* \*\*Protección de rutas:\*\* Solo usuarios autenticados acceden a la app

# 8. Configuración Y Despliegue

## A. \*\*Frontend\*\*

* Requisitos: Node.js 16+, npm
* Instalar dependencias: npm install
* Configurar Firebase en src/firebase.ts
* Configurar URL del backend en src/services/api.ts
* Ejecutar: npm start (dev) o npm run build (producción)
* Desplegar en Firebase Hosting (firebase deploy)
* firebase.json: rewrites para SPA (todas las rutas a index.html)

## B. \*\*Backend\*\*

* Requisitos: Java 21+, Maven
* Configurar application.properties (Firebase, puerto, etc.)
* Ejecutar: mvn spring-boot:run o java -jar target/\*.jar
* Desplegar en VPS, Heroku, Docker, etc.
* Abrir puerto 8080 (o configurado)

# 9. Troubleshooting Y Buenas Prácticas

* Revisar configuración CORS si hay errores 403/405
* Verificar credenciales de Firebase y reglas de Firestore
* Limpiar caché y desregistrar Service Workers si hay problemas de SPA
* Mantener sincronizados los modelos entre frontend y backend
* Usar logs y feedback visual para depuración
* Proteger siempre los endpoints sensibles

# 10. Créditos Y Contacto

Desarrollado por el equipo de Concreteware UNAL

Contacto: info@concreteware.com

Para contribuciones, sugerencias o soporte técnico, consultar el archivo README.md o escribir al correo anterior.