Recipes App - Documentación Técnica Completa

📖 1. Introducción y Visión General

Recetas App es una aplicación web moderna y responsive diseñada para que los amantes de la comida exploren, descubran y organicen sus recetas favoritas. Resuelve el problema de tener recetas dispersas, proporcionando una plataforma centralizada vinculada a **TheMealDB** para buscar comidas y guardar favoritos personales en la nube.

> 🚀 ***Proyecto Final**: Esta aplicación representa el trabajo final integrador del **Bootcamp de Desarrollo Frontend Moderno** dictado por Código Facilito.*

✨ Funcionalidades Principales

- **Exploración de Recetas**: Navega por categorías (Carne, Pollo, Vegetariana, etc.).
- **Búsqueda Inteligente**: Encuentra recetas instantáneamente por nombre.
- **Detalle de Receta**: Visualiza ingredientes, instrucciones y videos en un modal accesible.
- **Autenticación de Usuarios**: Login y registro seguros mediante Firebase (Email/Password y Google).
- **Sistema de Favoritos**: Colección personalizada de recetas guardadas en la nube (Firestore).
- **Diseño Responsive**: Totalmente optimizada para móviles, tablets y escritorio.
- **Accesibilidad (a11y)**: Cumplimiento WCAG 2.1 AA (Gestión de foco, soporte ARIA).

🛠️ Stack Tecnológico

Tecnología	Rol	Motivo de la Elección
React 18+	UI	Arquitectura basada en componentes.
Vite	Build Tool	HMR extremadamente rápido.
Zustand	Estado	Gestión de estado ligera y sin boilerplate.
TailwindCSS	Estilos	Utility-first para desarrollo rápido.
Firebase	Backend	Auth y DB NoSQL (Firestore).
TheMealDB	API	Fuente de datos de recetas.

🏗️ 2. Arquitectura del Proyecto

El código base sigue la metodología **Atomic Design**, promoviendo la reutilización y la separación de responsabilidades.

📂 Estructura de Carpetas

```
src/
├── components/          # Componentes UI (Atomic Design)
│   ├── atoms/           # Elementos indivisibles (Button, Input, Spinner)
│   ├── molecules/       # Grupos de átomos (RecipeCard, SearchBar)
│   ├── organisms/       # Secciones complejas (RecipesGrid, Navbar)
│   ├── templates/       # Estructuras de página (Layout)
│   └── ui/              # Componentes genéricos (Modal)
├── config/              # Configuración de servicios (firebase.js)
├── constants/           # Constantes globales (Rutas, Mensajes, Config)
├── hooks/               # Custom Hooks (Lógica de negocio y estado)
├── pages/               # Vistas principales (Lazy loaded)
├── routes/              # Definición de rutas y protecciones
├── services/            # Comunicación con APIs externas
├── store/               # Estado global (Zustand)
└── utils/               # Funciones auxiliares
```

🧠 Manejo de Estado (Zustand)

El estado se divide en tres stores especializados para evitar "God Stores":

1. `recipesStore.js` : Datos de recetas, categorías y filtros.
2. `userStore.js` : Estado de sesión y perfil del usuario.
3. `uiStore.js` : Estado efímero (modales, notificaciones).

🚦 Routing

Gestionado por `react-router-dom` con **Lazy Loading**:

- **Públicas**: `/` (Home), `/login` , `/register` .
- **Protegidas**: `/profile` (Requiere autenticación).

🚀 3. Guía de Instalación y Configuración

Requisitos Previos

- Node.js (v18+)
- npm o yarn
- Cuenta de Google (para Firebase)

Pasos de Instalación

1. **Clonar el repositorio**

```
git clone https://github.com/Oliver-92/Recipes_App.git
cd Recipes_App
```

2. **Instalar dependencias**

```
npm install
```

3. **Configuración de Firebase (CRÍTICO)**

- Ve a Firebase Console y crea un proyecto.
- **Auth**: Habilita "Email/Password" y "Google" en Authentication.
- **Firestore**: Crea una base de datos en modo de prueba.
- **Reglas de Firestore**: Copia y pega las siguientes reglas en la pestaña "Reglas":

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /favorites/{document=**} {
      allow read, write: if request.auth != null && request.auth.uid ==
resource.data.userId;
      allow create: if request.auth != null;
    }
  }
}
```

4. **Variables de Entorno**

- Copia el archivo de ejemplo: `cp .env.example .env.local`
- Obtén tus credenciales en *Project Settings* > *General* > *Your apps* en Firebase Console.
- Rellena el archivo `.env.local`:

```
VITE_FIREBASE_API_KEY=tu_api_key
VITE_FIREBASE_AUTH_DOMAIN=tu_id.firebaseapp.com
VITE_FIREBASE_PROJECT_ID=tu_project_id
VITE_FIREBASE_STORAGE_BUCKET=tu_id.appspot.com
VITE_FIREBASE_MESSAGING_SENDER_ID=tu_sender_id
VITE_FIREBASE_APP_ID=tu_app_id
```

5. **Ejecutar la aplicación**

```
npm run dev
```

💻 4. Guía de Desarrollo

Convenciones

- **Componentes**: PascalCase (`RecipeCard.jsx`).
- **Hooks**: `use` + PascalCase (`useRecipes.js`).
- **Servicios**: camelCase (`mealService.js`).
- **Constantes**: UPPER_SNAKE_CASE (`API_BASE_URL`).

Patrones de Uso

Consumo de Stores:

```
import { useRecipesStore } from '@/store/recipesStore';
// Selector para evitar re-renders innecesarios
```

```
const recipes = useRecipesStore(state => state.recipes);
```

Uso de Servicios:

```
import { getMealsByCategory } from '@/services/mealService';
// Llamadas asíncronas manejadas siempre con try/catch
try {
  const data = await getMealsByCategory('Beef');
} catch (error) {
  showNotification('Error fetching meals', 'error');
}
```

🧪 5. Testing Manual

Lista de verificación para asegurar el correcto funcionamiento:

Autenticación

- [ ] Registro exitoso con email válido.
- [ ] Validación de errores (email inválido, contraseña corta).
- [ ] Login correcto y redirección al Home.
- [ ] Logout funcional.

Funcionalidad Core

- [ ] Carga inicial de categorías.
- [ ] Filtrado de recetas al cambiar categoría.
- [ ] Búsqueda por texto retorna resultados coherentes.
- [ ] Modal de detalles abre correctamente con info de la receta.

Favoritos

- [ ] Click en "Corazón" guarda la receta (con usuario logueado).
- [ ] Click en "Corazón" elimina la receta si ya existe.
- [ ] Vista `/profile` muestra la lista correcta de favoritos.
- [ ] Persistencia de datos al recargar la página.

🏆 6. Equipo de Evaluación de Código Facilito

Un agradecimiento especial al increíble equipo de evaluación del Bootcamp de Código Facilito:

Eva Ferreira

🗞️ Twitter Eva es una de las personas más importantes de la comunidad tech, organizó por varios años la primer conferencia de CSS en LATAM. Es UI Lead Developer, y Google Developer Expert en web. Ha participado en más de 30 eventos en 12 países.

Bel Rey

🔗 Twitter Más de 10 años de experiencia en distintos roles dentro del mundo de sistemas, desde cofundadora hasta ingeniero de software. Cuenta con años de experiencia en el área edTech, y es una de las principales creadoras de contenido en español sobre programación.

Juan José Ortiz

🔗 Twitter Juan es uno de los profesores favoritos de la comunidad de Código Facilito, es Senior Software Engineer en X-Team, trabajando para clientes como Intel, Sony, RIOT, y Twitter. Referente enseñando React y Frontend.

Vanessa Aristizabal

🔗 Twitter Vanessa es Senior Software Engineer, Google Developer Expert en Angular y tecnologías web. Es GitHub Star y Embajadora de Women Techmaker Medellín. Entusiasta de la comunidad y creadora de contenido educativo.

Uriel Hernández

🔗 Twitter Ingeniero en desarrollo de software y actualmente CTO (Chief Technology Officer) de Código Facilito. Uriel es un apasionado de la tecnología y ha trabajado en la industria del desarrollo de software durante más de 10 años.

Desarrollado por Ezequiel Oliver