FixShops Technical Documentation

Architecture Overview

FixShops is a vertical SaaS platform designed for the repair economy. It utilizes a modern **Serverless Architecture** where the frontend is a Single Page Application (SPA) that communicates directly with a managed Backend-as-a-Service (Convex).

- **Frontend**: A React application built with Vite. It handles all UI rendering, routing, and client-side logic.
- **Backend & Database**: **Convex** serves as the realtime database, file storage, and backend function host. It replaces traditional SQL databases and Node.js servers.
- **Authentication**: **Clerk** manages user identities, sessions, and secure access,

tightly integrated with Convex.

- **Payments**: **Stripe** handles subscription billing and checkout sessions via webhook integrations.

Tech Stack

Frontend Core

- **Framework**: React v19
- **Build Tool**: Vite v6
- **Language**: TypeScript
- **Routing**: React Router DOM v7
- **State Management**: Convex (React Query-like hooks)

UI & Styling

- **Styling**: Tailwind CSS (Utility-first CSS)
- **Animations**: Framer Motion
- **Icons**: Lucide React

Backend & Services

- **Database/Functions**: Convex (Realtime relational database)
- **Auth Provider**: Clerk
- **Payment Gateway**: Stripe
- **Generative AI**: Google Gemini (via `**@google/genai**`) used for dynamic image generation.
- **Mock Data**: FakerJS (used for database seeding)

Project Structure

```
FLIPPA-Fixshops/
├── components/          # Reusable UI components (Header, Footer, Forms)
├── convex/              # Backend Logic
│   ├── _generated/      # Auto-generated Convex types
│   ├── auth.config.ts   # Clerk integration config
│   ├── bookings.ts      # Booking CRUD & Notifications
│   ├── http.ts          # HTTP Webhook endpoints (Stripe)
│   ├── platform.ts      # White-label configuration logic
│   ├── providers.ts     # Provider/Shop logic
```

```
│   ├── schema.ts        # Database Schema definition
│   ├── services.ts      # Repair Service CRUD
│   ├── stripe.ts        # Stripe checkout actions
│   ├── users.ts         # User management
│   └── seed.ts          # Database seeder
├── lib/                 # Utilities
├── pages/               # Application Route Views
│   ├── AdminDashboardPage.tsx  # Platform Admin
│   ├── DashboardPage.tsx       # Provider Dashboard
│   ├── ConsumerDashboard.tsx   # User Booking History
│   ├── MarketplacePage.tsx     # Search & Discovery
│   └── ...              # Static pages (Home, About, Legal)
├── public/              # Static assets & .htaccess
├── App.tsx              # Main Router & Auth Guards
├── index.tsx            # Entry point & Provider wrappers
└── vite.config.ts       # Build configuration
```

Database Schema

The database is defined in `convex/schema.ts`. It is a relational document schema.

Table Name	Description	Key Fields
users		`clerkId` `email` `role` `subscriptionActive`
providers		`storeName` `storeSlug` `isPublished` `operatingHours` `serviceAreas` `logoUrl`
services		`providerId` `name` `price` `duration` `description`
bookings		`providerId` `serviceId` `consumerEmail` `date` `time` `status`
platform_config		`platformName` `logoUrl` `subscriptionPlanPrice`

Setup & Installation

Follow these steps to set up the development environment locally.

1. Prerequisites

- Node.js (v18 or higher)

- NPM or Yarn
- A Convex account (convex.dev)
- A Clerk account (clerk.com)
- A Stripe account (stripe.com)

2. Installation

1. **Clone the repository**:

```
git clone [repo-url]
cd FLIPPA-Fixshops
```

2. **Install dependencies**:

```
npm install
```

3. **Initialize Convex**:

```
npx convex dev
```

This will prompt you to log in and create a new Convex project. It will generate a ***`.env.local`*** *file with your* ***`CONVEX_DEPLOYMENT`*** *and* ***`VITE_CONVEX_URL`****.*

3. Environment Configuration

Create or update **`.env.local`** in the root directory. You will need keys from Clerk, Convex, and Google Gemini.

Client-side variables (`.env.local`):

```
VITE_CONVEX_URL=https://[your-project].convex.cloud
VITE_CLERK_PUBLISHABLE_KEY=pk_test_...
API_KEY=[Your_Gemini_AI_Key]
```

Convex Dashboard Variables: Go to your Convex Dashboard > Settings > Environment Variables and set:

- **`CLERK_SIGNING_KEY`**: (From Clerk JWT Templates)
- **`STRIPE_SECRET_KEY`**: **`sk_test_...`**
- **`STRIPE_WEBHOOK_SECRET`**: (From Stripe Webhook settings)

4. Database Seeding

To populate the database with mock shops and services:

```
npx convex run seed:seed
```

5. Run Local Server