

GlobStaff Technical Documentation

Architecture Overview

GlobStaff is a modern, real-time marketplace application designed to connect service providers with consumers. The platform utilizes a decoupled architecture where a rich Single Page Application (SPA) frontend communicates with a serverless backend-as-a-service (BaaS).

The system operates on a role-based architecture supporting three distinct user types:

1. **Consumers:** Browse the marketplace, book services, and manage hires.
2. **Providers:** Create storefronts, list services, manage booking requests, and track payouts.
3. **Administrators:** Oversee platform operations, manage content/branding, control pricing, and moderate users.

Payments and payouts are securely handled via integrated Stripe connect flows, while all data operations feature real-time synchronization between the database and active clients.

Tech Stack

Based on the project configuration, the application utilizes the following

Based on the project configuration, the application utilizes the following modern web stack:

- **Frontend Framework:** React (via Vite)
- **Language:** TypeScript
- **Backend & Database:** Convex (Serverless backend, real-time database, and API generation)
- **Authentication:** Convex Auth (integrated identity management)
- **Payments Processing:** Stripe (Payments, Subscriptions, and Connect Payouts)
- **Routing:** React Router (implied by the extensive `pages/` directory structure)
- **Styling:** Modern CSS solutions (likely Tailwind CSS, standard with this ecosystem)

Project Structure

The codebase is organized into a feature-based structure for scalability and maintainability:

```
GlobStaff/
├── components/           # Reusable presentation components (UI, Landing
├── context/             # React Context providers (e.g., StaffContext f
├── convex/              # Backend logic and database configuration
│   ├── _generated/      # Auto-generated TypeScript types and API hooks
│   ├── schema.ts        # Database schema definitions
│   └── *.ts             # Backend mutation/query functions (bookings, u
├── pages/              # Application routes and views
│   ├── admin/          # Admin dashboard and management components
│   ├── consumer/       # Consumer-facing dashboard and billing
│   ├── dashboard/      # Provider-facing dashboard, payouts, and servi
│   └── *.tsx           # Public pages (Home, Blog, Auth, Marketplace,
├── public/             # Static assets (images, fonts) and Apache .hta
├── scripts/            # Utility scripts for maintenance and Stripe se
├── types.ts            # Global TypeScript interface definitions
└── vite.config.ts      # Vite bundler configuration
```

Database Schema

The database is built on Convex. Based on the backend file structure, the schema consists of the following core tables (collections):

- ``users``
 - Stores authentication profiles, user roles (Admin, Provider, Consumer), and contact information.
- ``services``
 - Stores provider offerings, including title, description, category, pricing, and provider reference ID.
- ``bookings``
 - Manages the lifecycle of a hire. Links a ``consumerId`` to a ``serviceId`` and ``providerId``. Tracks statuses (pending, accepted, completed, canceled).
- ``reviews``
 - Stores feedback and ratings linked to specific ``bookings``, ``services``, and ``providers``.
- ``pricing``
 - Platform configuration for fees, subscription tiers, or global pricing rules managed by admins.
- ``branding``
 - Dynamic application configuration (logos, colors, platform names) allowing white-labeling directly from the admin dashboard.
- ``content``
 - Headless CMS entries for dynamic pages like Blog Articles, FAQs, and Policies.

Setup & Installation

Follow these steps to configure the project in a local development environment.

Prerequisites

- Node.js (v18 or higher recommended)
- npm or yarn
- A Convex account (free tier is sufficient for development)
- A Stripe account for payment testing

1. Install Dependencies

Clone the repository and install the required packages:

```
npm install
```

2. Configure Environment Variables

Create a ``.env.local`` file in the root directory. You will need to add your Convex deployment URL and Stripe API keys:

```
VITE_CONVEX_URL=your_convex_deployment_url  
STRIPE_SECRET_KEY=your_stripe_secret_key  
VITE_STRIPE_PUBLIC_KEY=your_stripe_public_key
```

3. Initialize Backend (Convex)

Run the Convex development server. This will sync your `convex/schema.ts` and functions to the cloud:

```
npx convex dev
```

4. Start the Frontend

In a separate terminal window, start the Vite development server:

```
npm run dev
```

Admin Credentials

To access the administrative dashboard and manage the platform, use the following login credentials:

- **Username / Email:** ``codeblix``
- **Password:** ``codeblix``

Deployment Guide