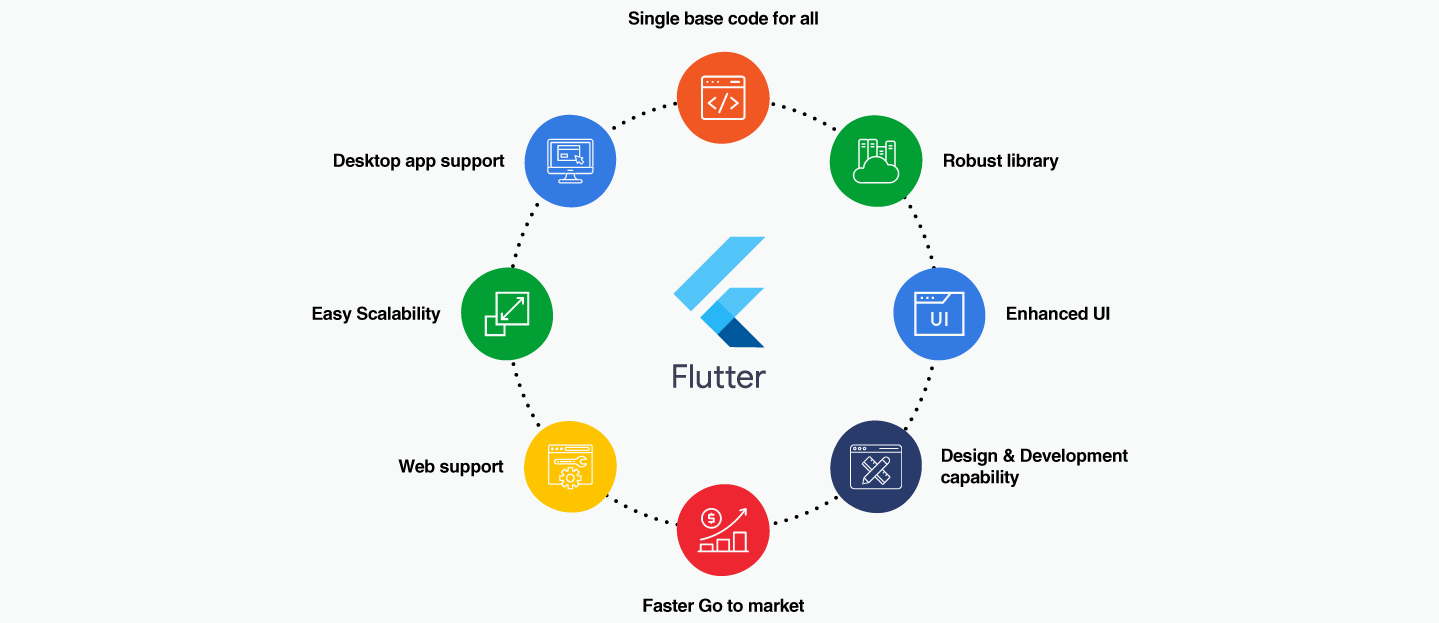


Mehran University of Engineering &

Technology Jamshoro, Pakistan

***Complex Engineering Problem Report on Travely App***



|  |  |
| --- | --- |
| **Roll No** | **22SW028, 22SW040** |
| **Section** | **01** |
| **Subject** | **MAD** |
| **CEP Project** | **Travely App** |
| **Teacher** | **Ma’am Maryam** |

**1. Real-World Problem Identification**

Tourists and travelers often struggle to find reliable information about **notable attractions** within a city, especially when traveling to unfamiliar regions. Existing travel apps are either too complex, lack local coverage, or display irrelevant global data.

Furthermore, travelers in developing regions face challenges like:

* Inconsistent internet access or poor data availability.
* Scattered information across multiple sources (Wikipedia, blogs, etc.).
* No centralized, user-friendly interface tailored to quick discovery.

This problem affects **casual travelers, students, and professionals** alike, who need a quick, lightweight, and visually appealing solution to explore nearby attractions without manual searching or unreliable sources.

**2. Proposed Solution**

To address this, we developed **Travely:Your Simple Travel Mate**, a **cross-platform mobile application** that helps users:

* Discover notable **tourist attractions, landmarks, and monuments** of any city.
* **Search globally** by city name (e.g., “Paris”, “Lahore”, “Tokyo”).
* View summarized **Wikipedia data** with images from **Unsplash API**.
* Save favorite places to the cloud using **Firebase Firestore**.
* Access a clean, responsive interface consistent across devices.

**Key Features**

1. **Smart City Search:** Retrieves real, verified attractions from Wikipedia API and Unsplash.
2. **Dynamic UI:** Auto-adapts to screen sizes, orientations, and themes.
3. **Favorites System:** Syncs user favorites via Firebase authentication.
4. **Offline Resilience:** Data persists locally when network drops.
5. **Cross-Platform Consistency:** Built in Flutter; works seamlessly on Android & iOS.

**System Architecture**

* **Frontend:** Flutter (Material 3) with Provider for state management.
* **Backend:** Firebase Authentication + Firestore Database.
* **APIs:** Wikipedia REST API and Unsplash API.
* **Hosting:** Firebase and GitHub for version control.

**3. Responsive User Interfaces**

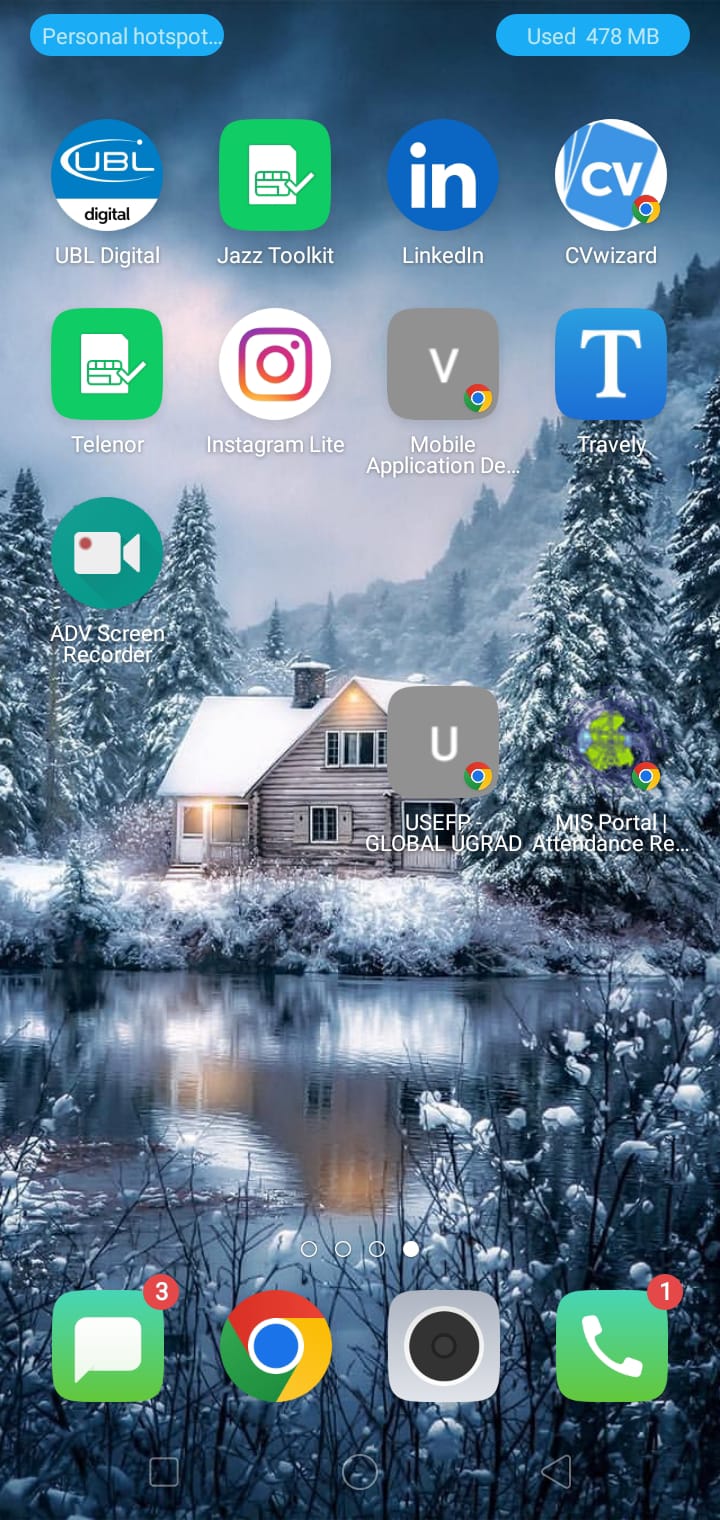
Travely’s UI was designed for simplicity and modernity using Flutter’s Material Design principles.

**--> See the SS below:**

**Icon of my Tavely App**

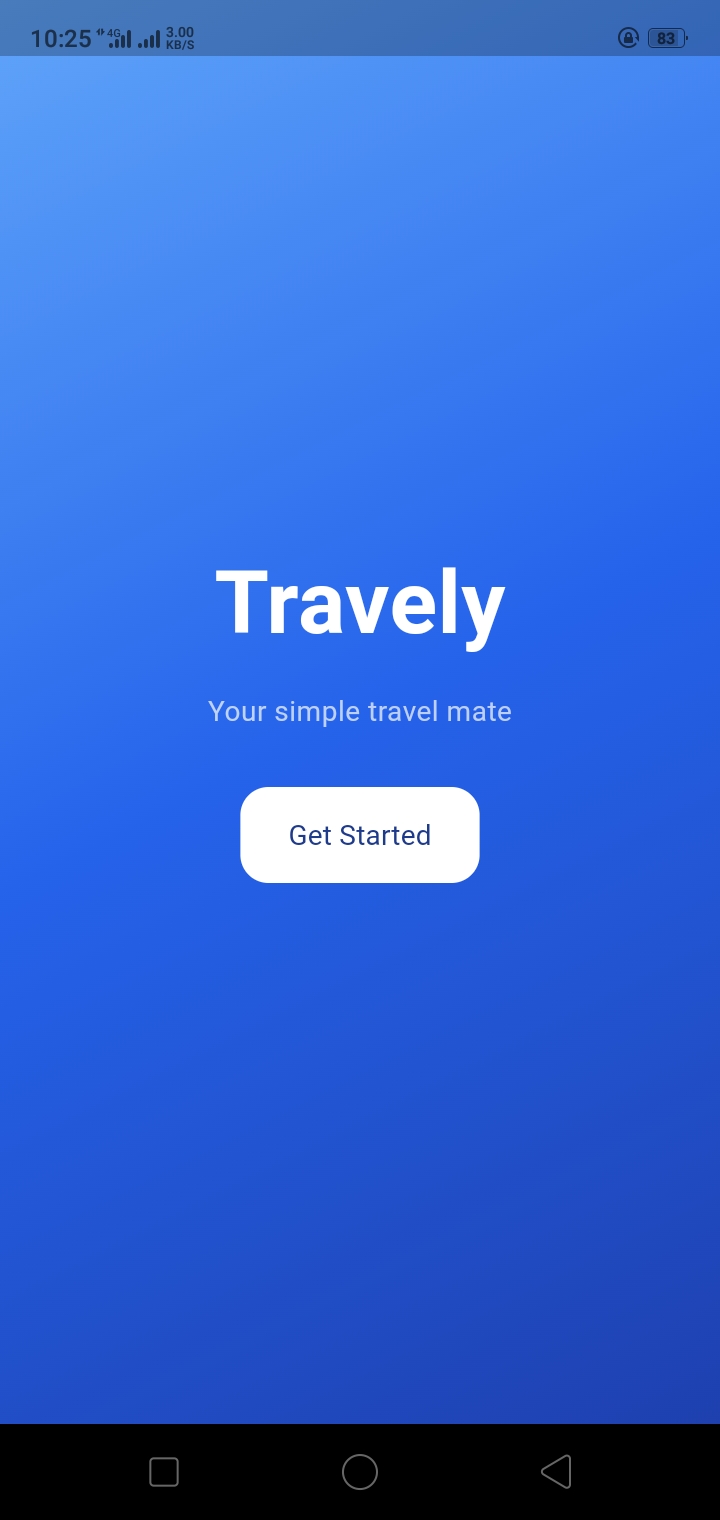
****

**See my App**

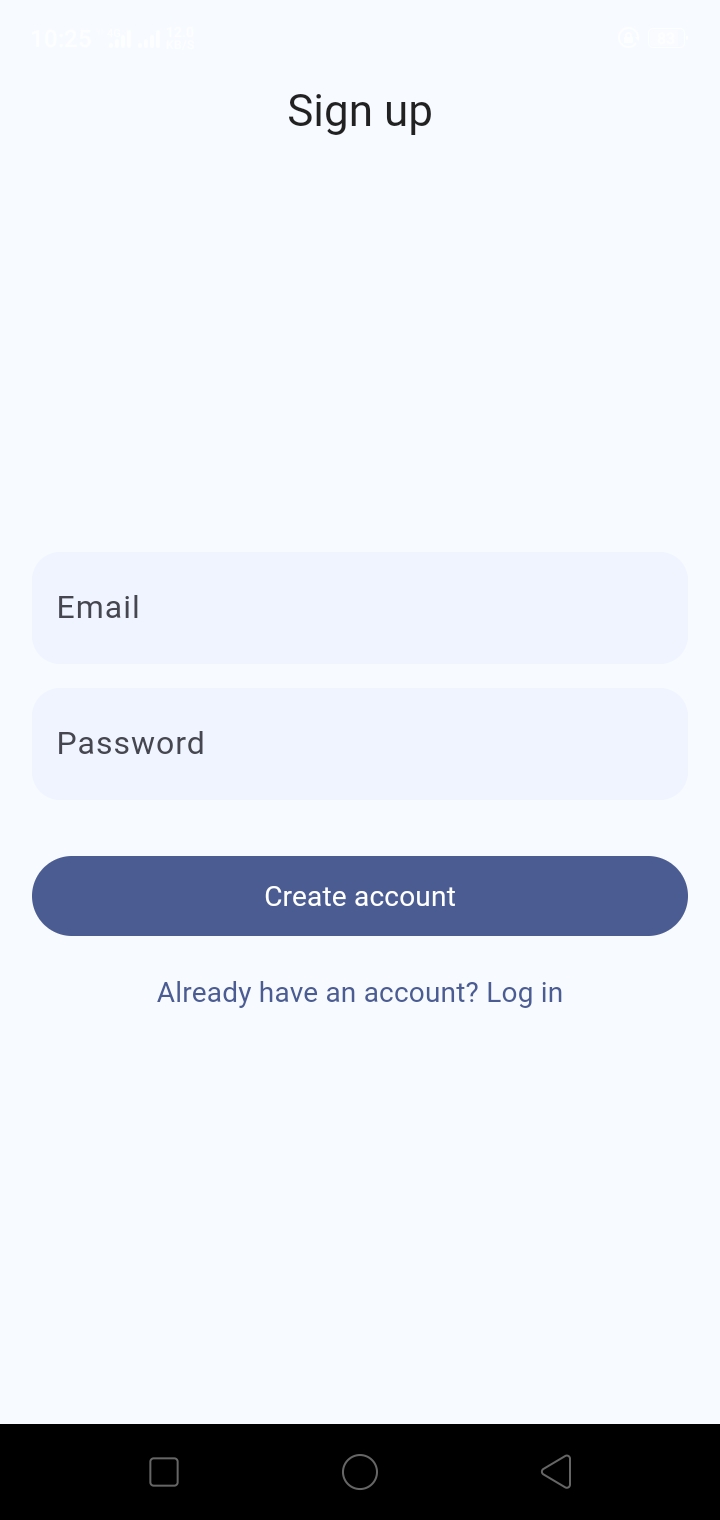
****

**First see the app in mobile**

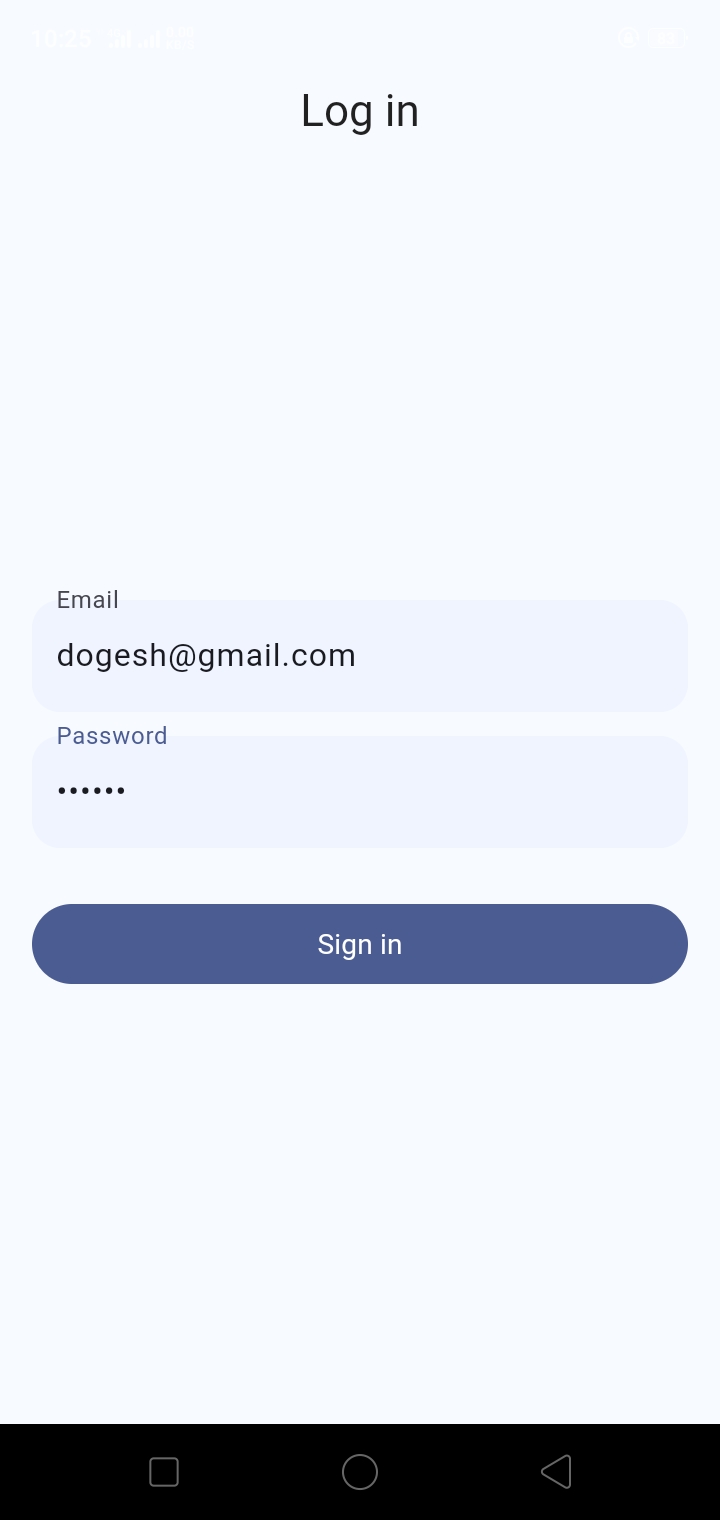
**The Splash Screen(Travely Branding)**

****

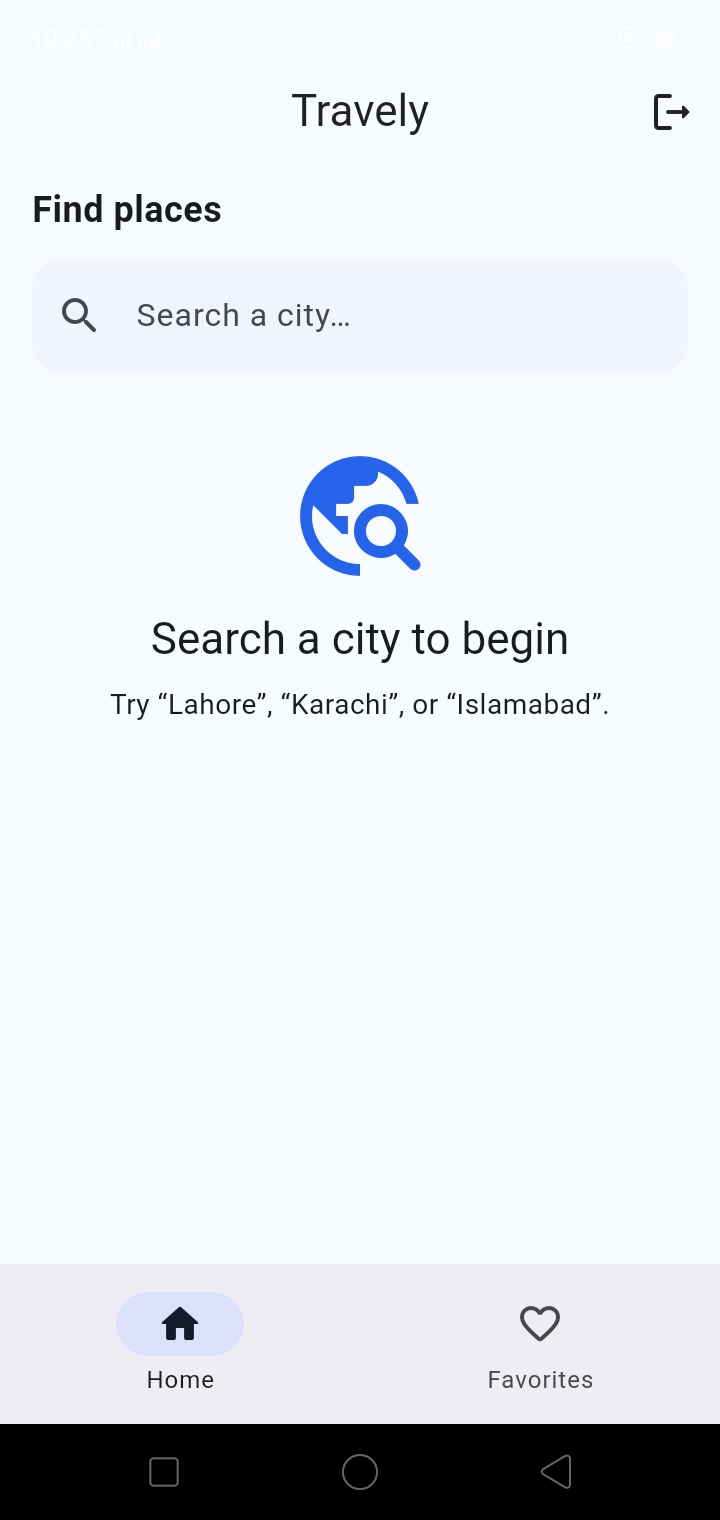
**The Sign Up Screen(with email and Password)**

****

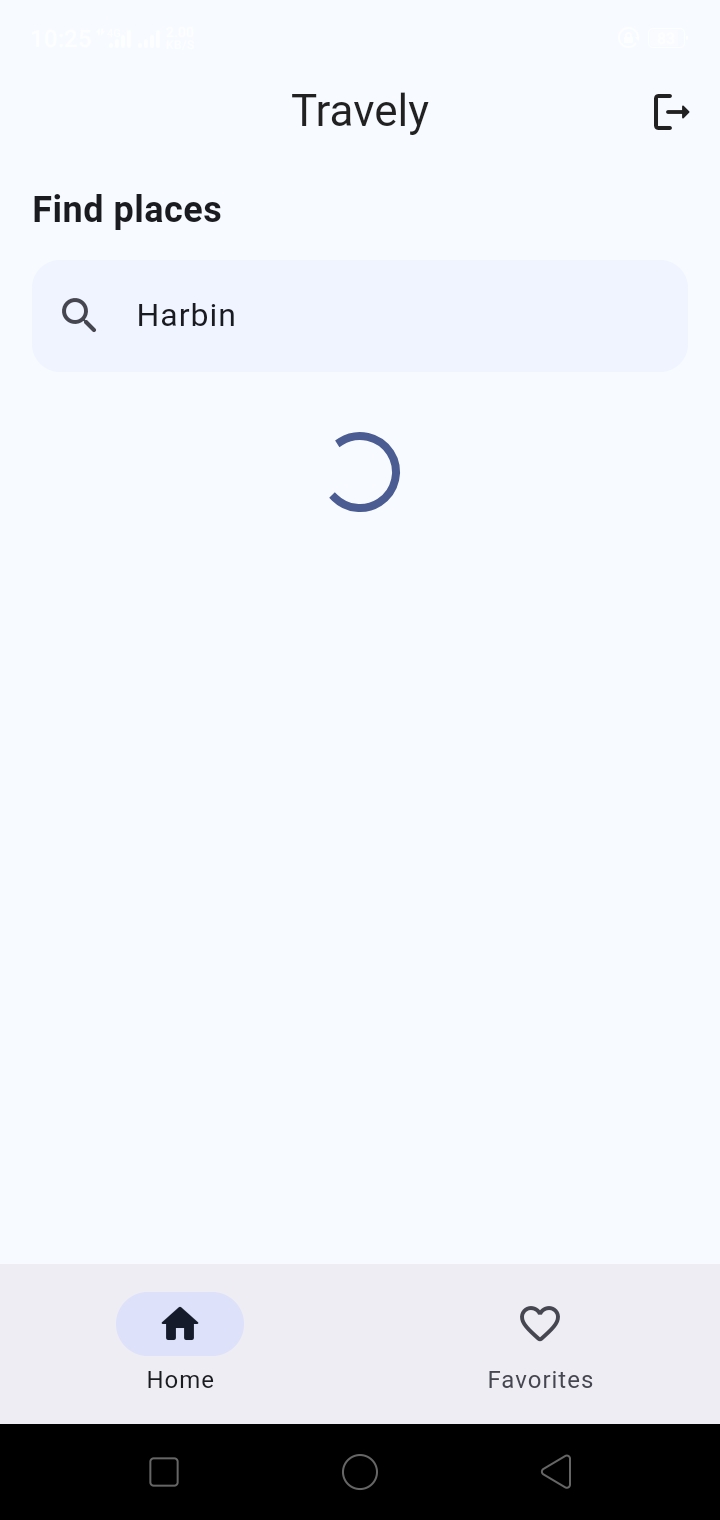
**The Log in Screen(dogesh@gmail.com,12345d)**

****

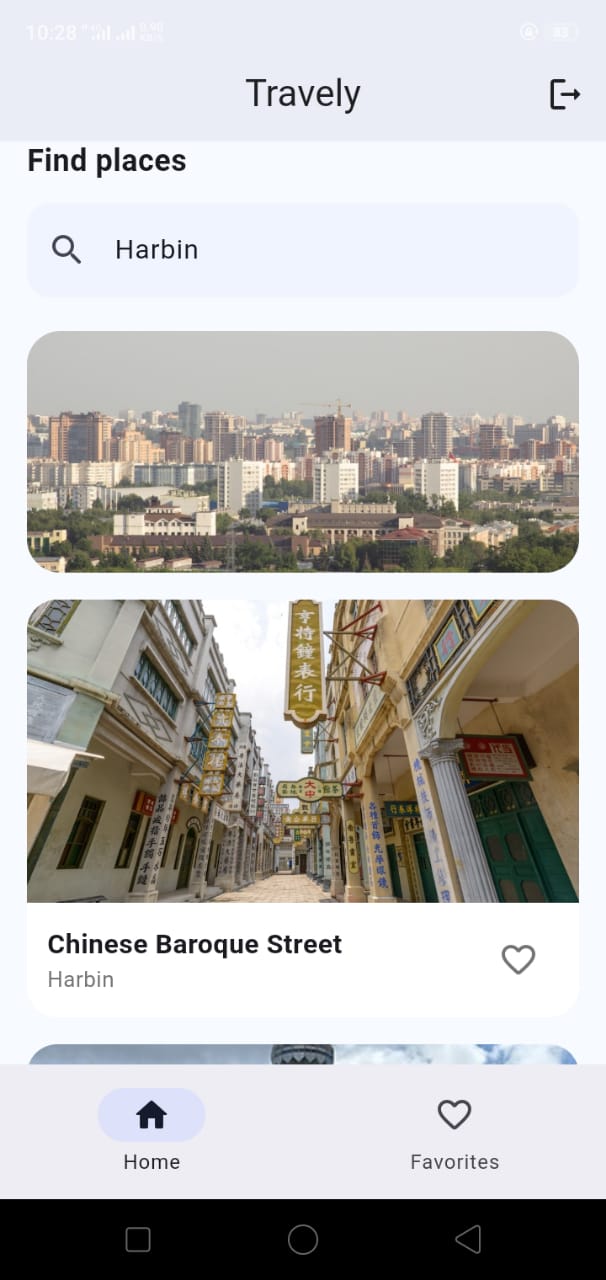
**Our primary/main screen**

****

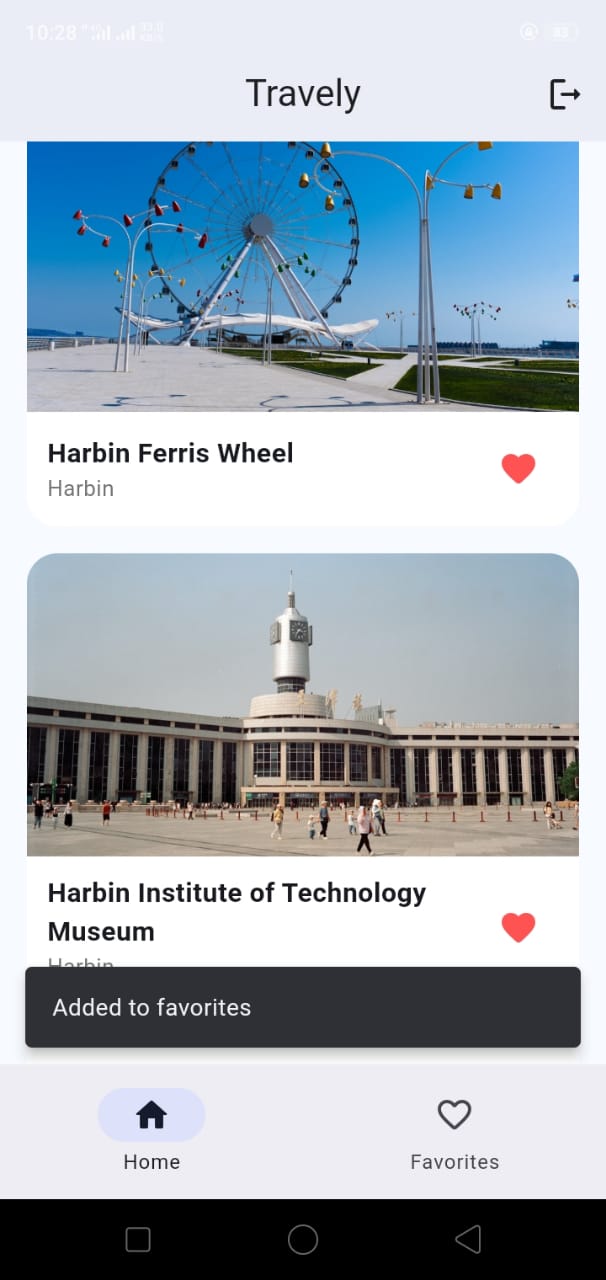
**Now Searching for City in China (Harbin)**

****

**Our App is Fetching the travel places in (Harbin)**

****

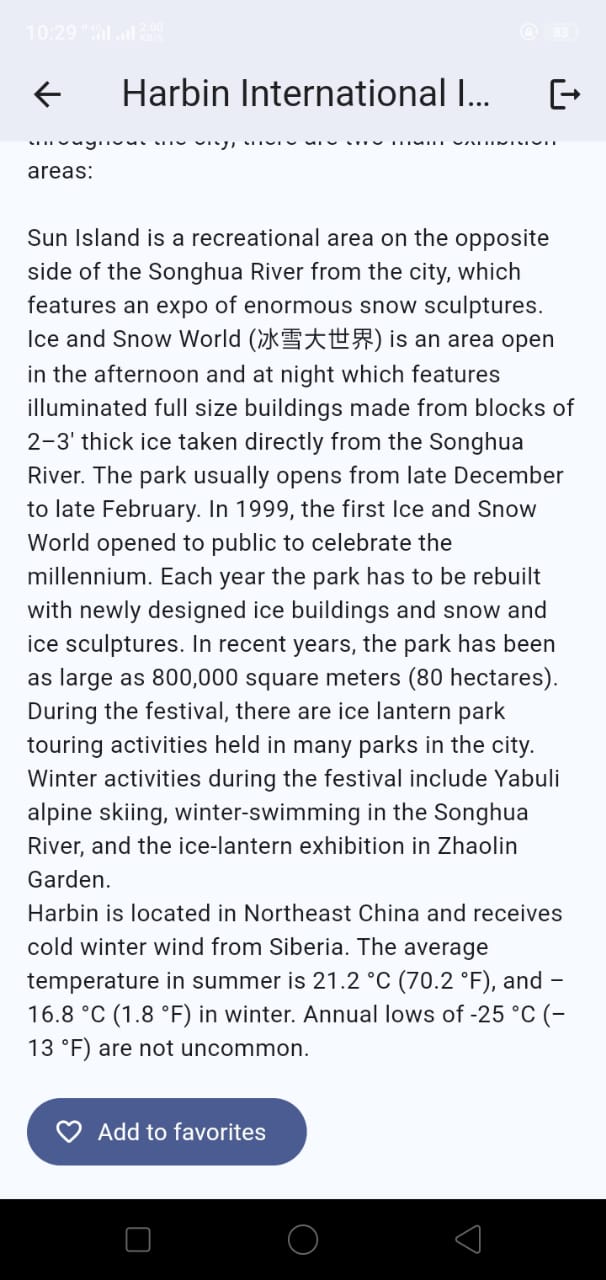
**Adding Some Places to my Favorites**

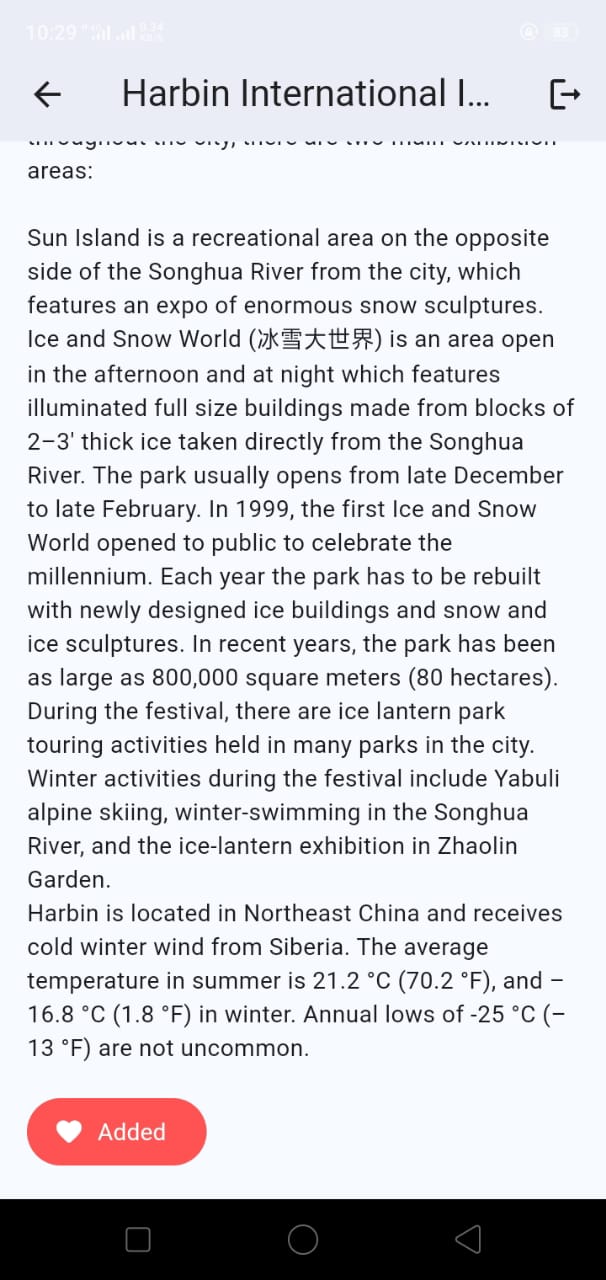
****

**Now Opening one of the places by taping the Card (To see more Info about that Travel Place)**

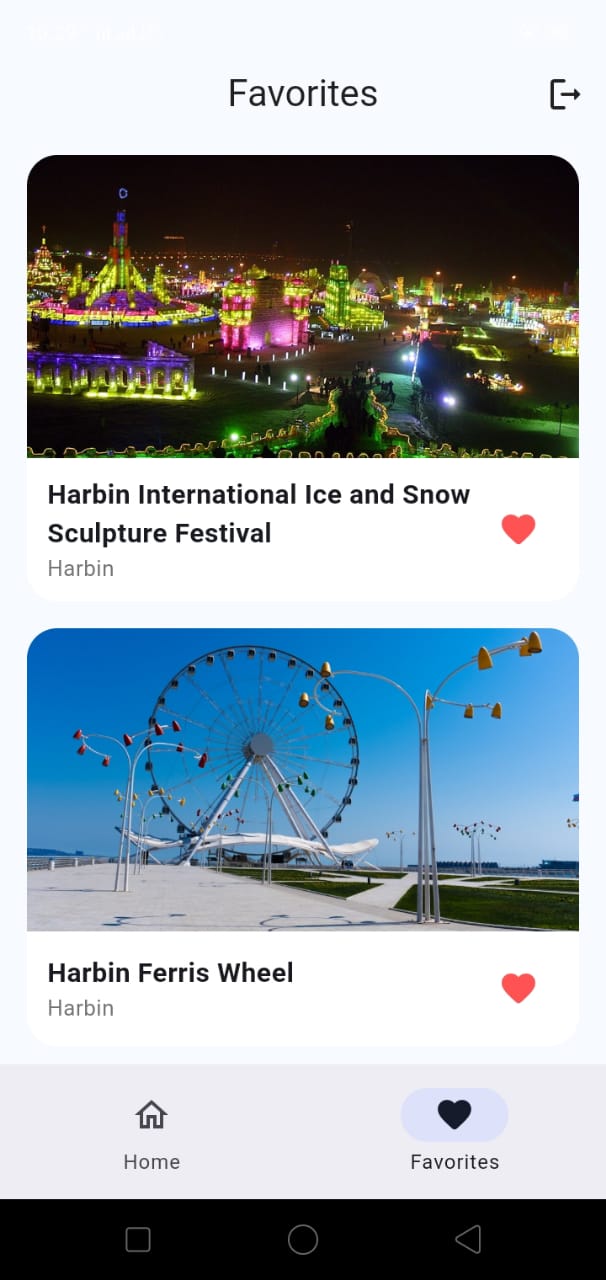
****

**Also adding this one in my Fav places**

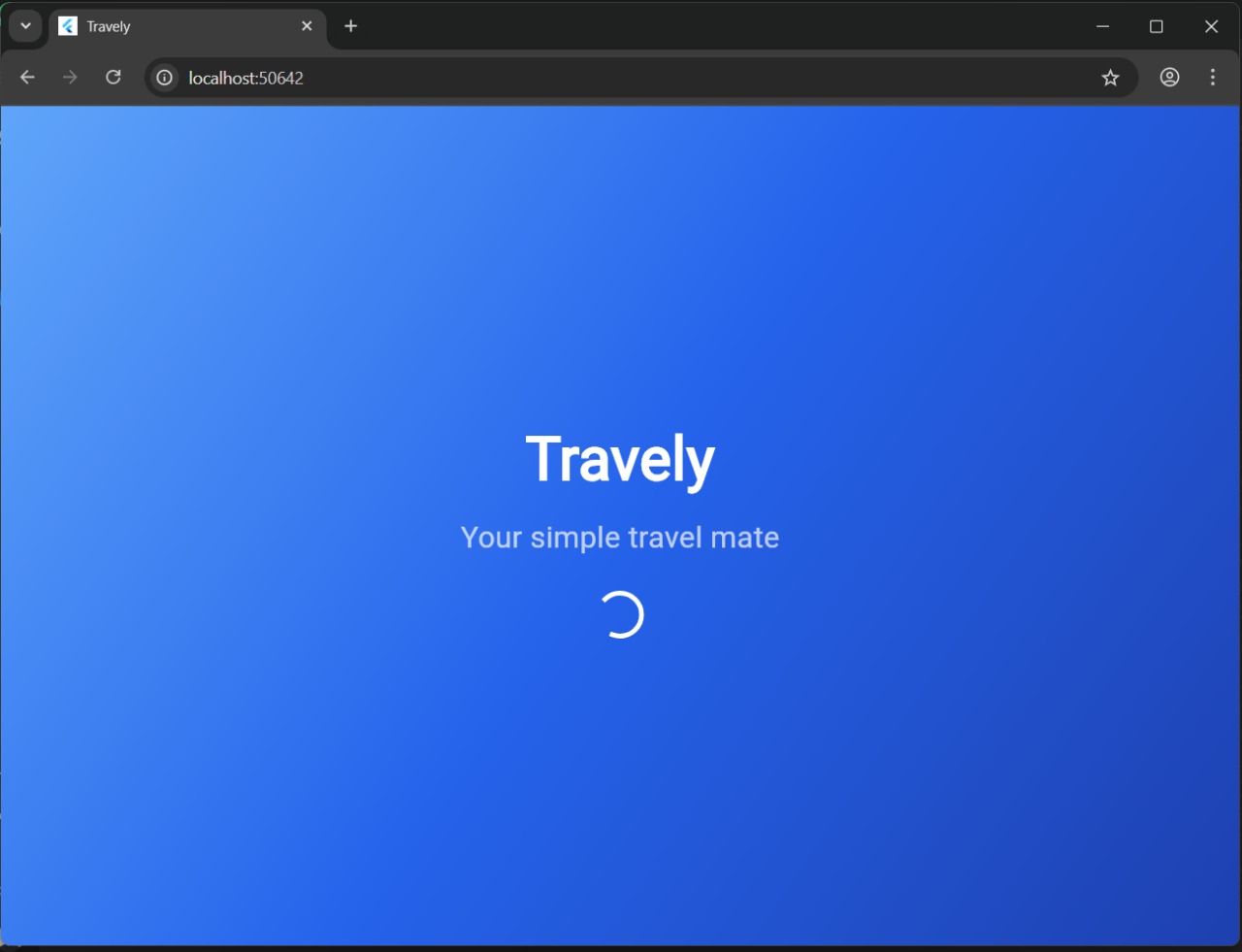
****

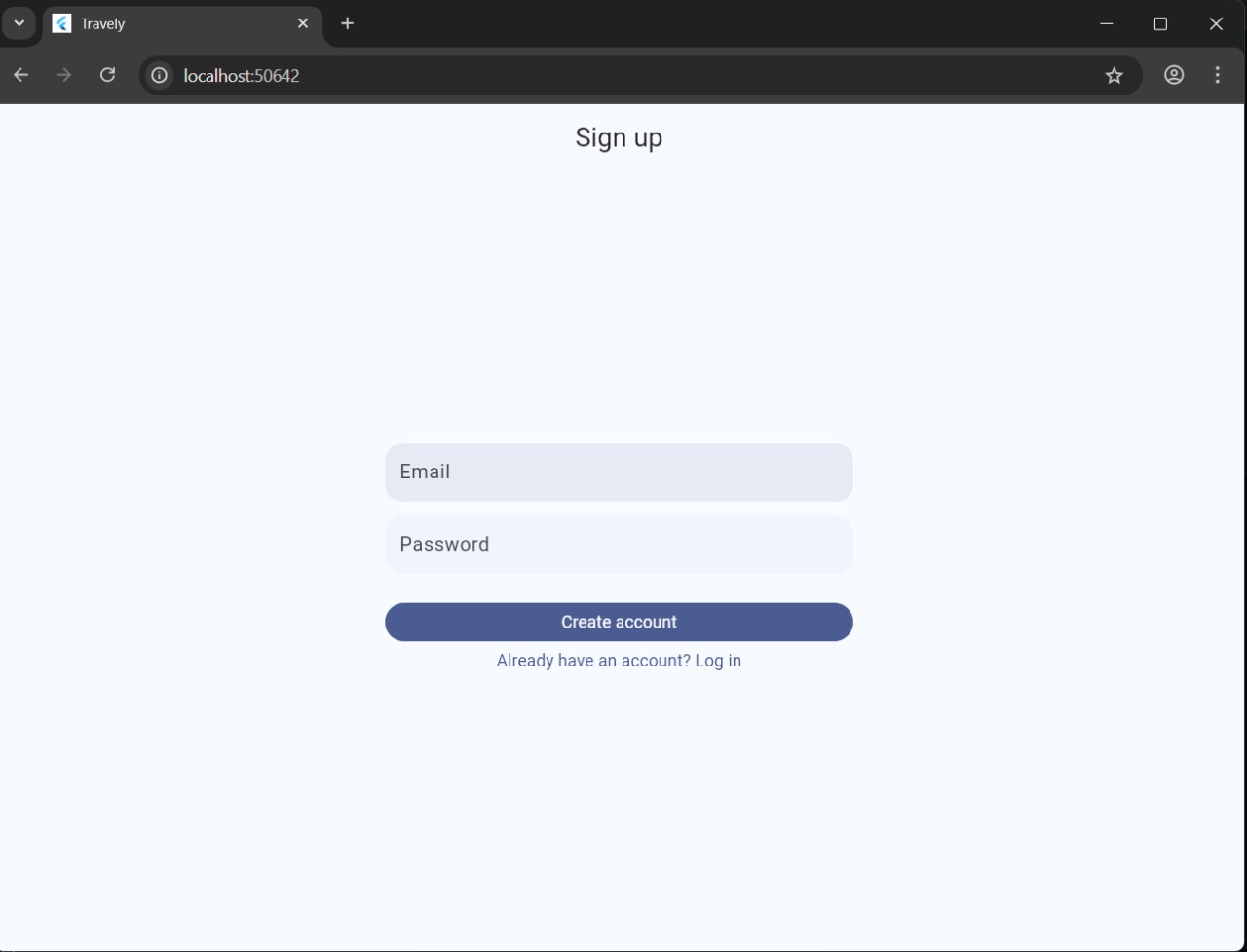
****

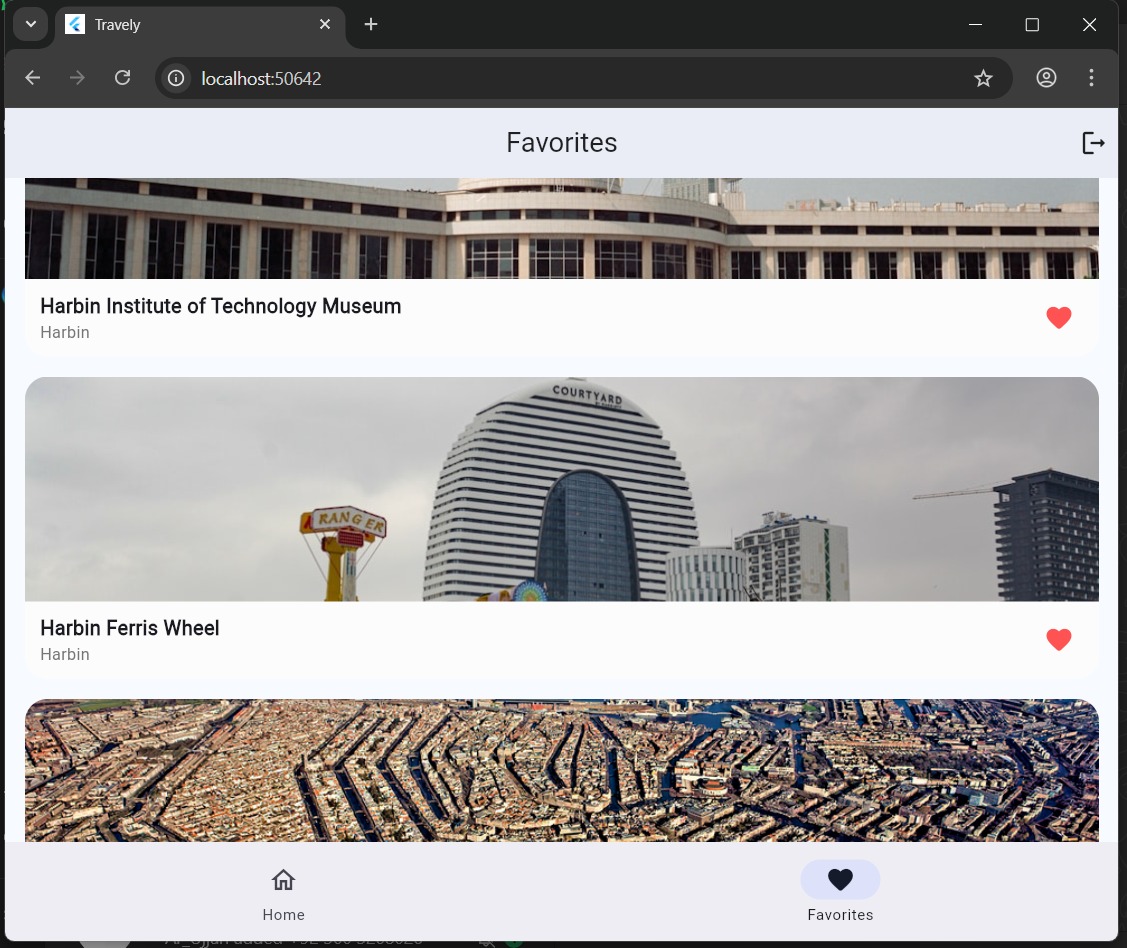
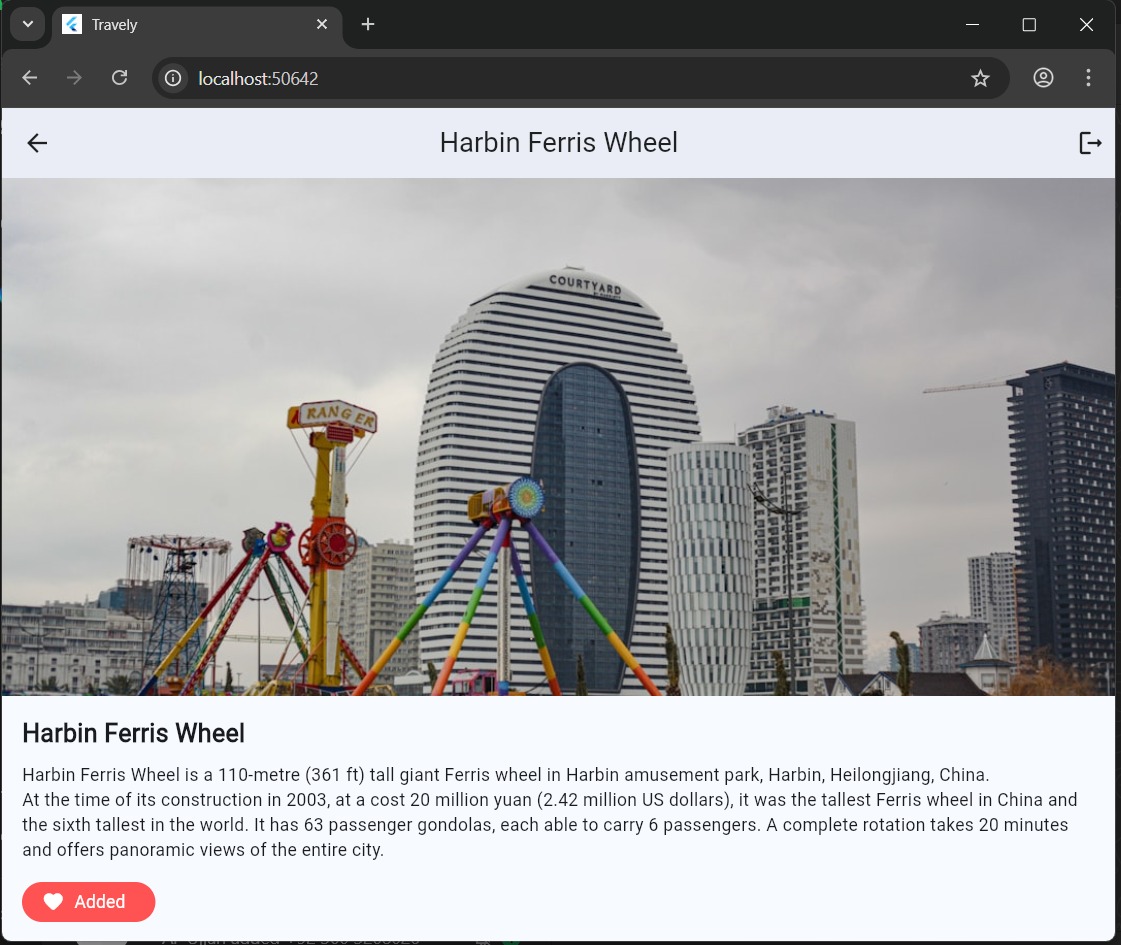
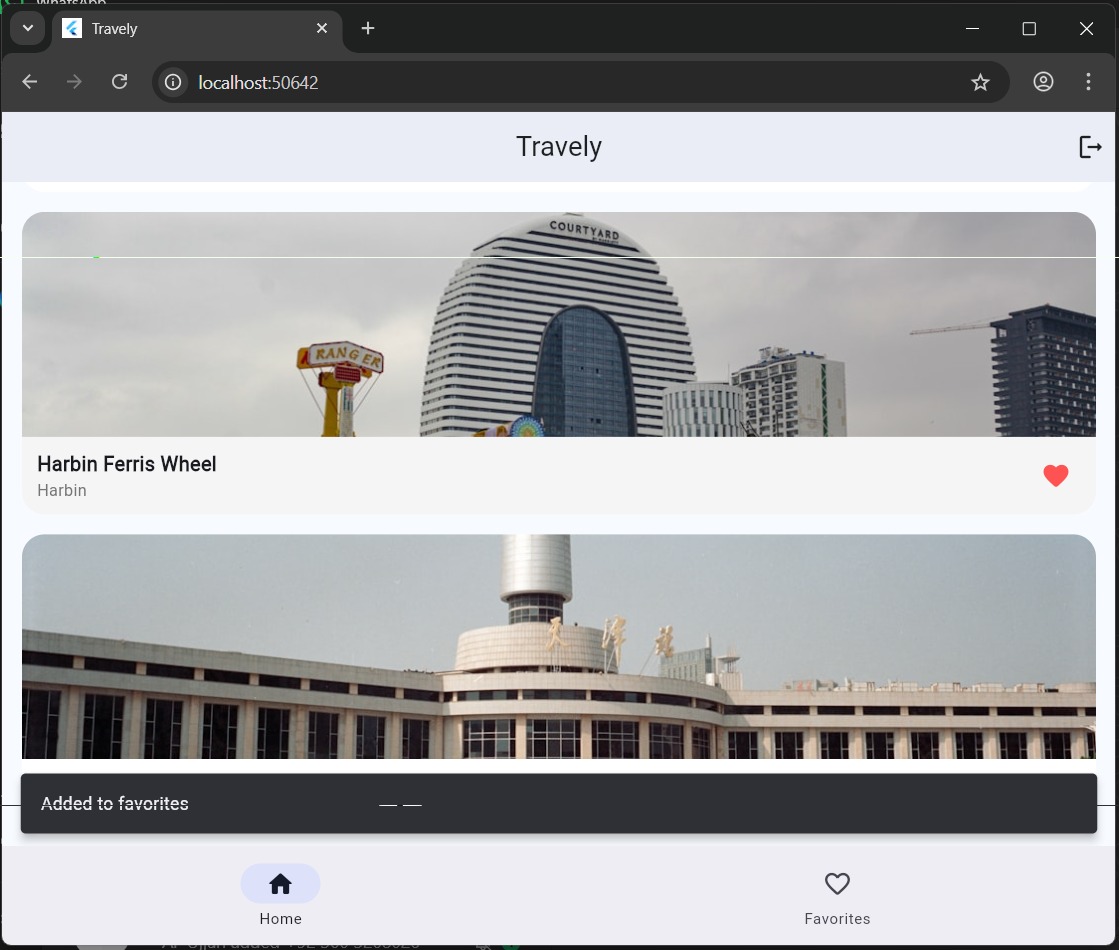
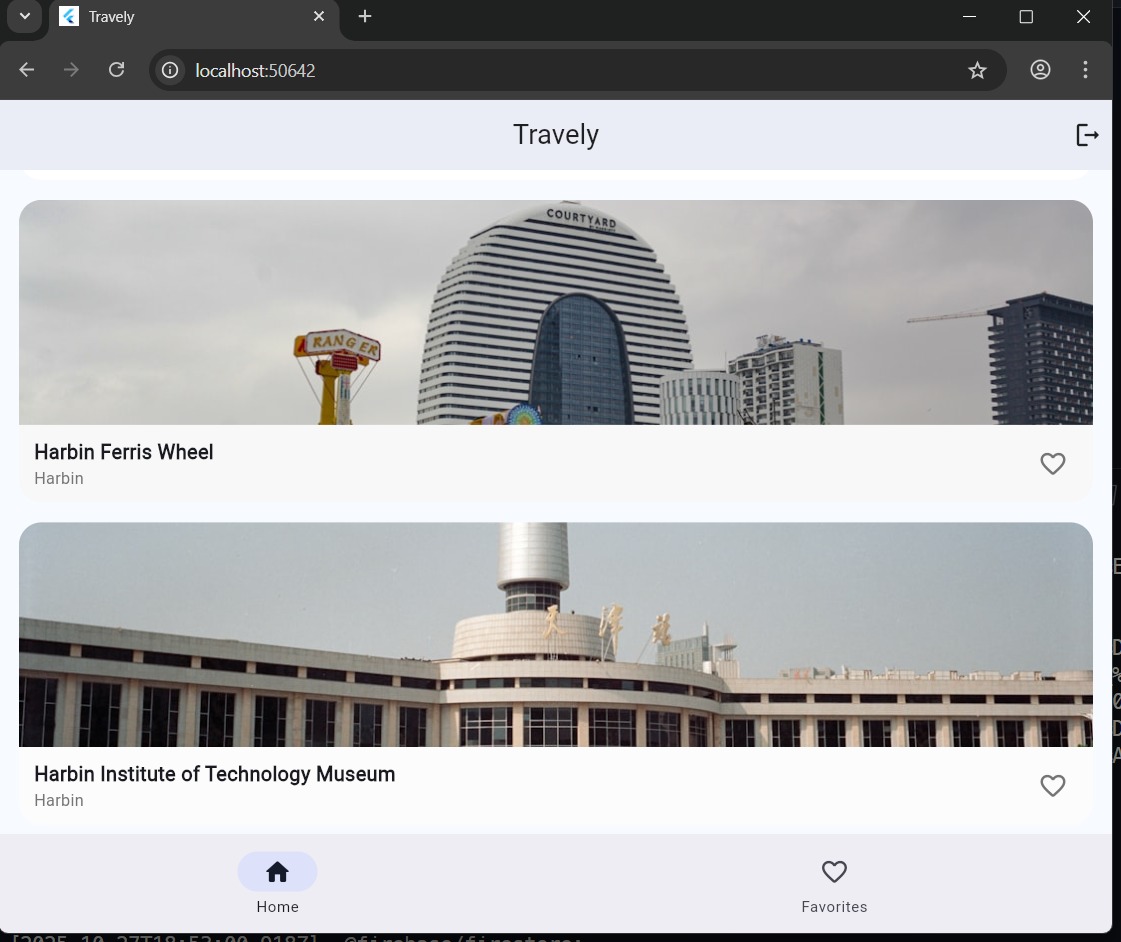
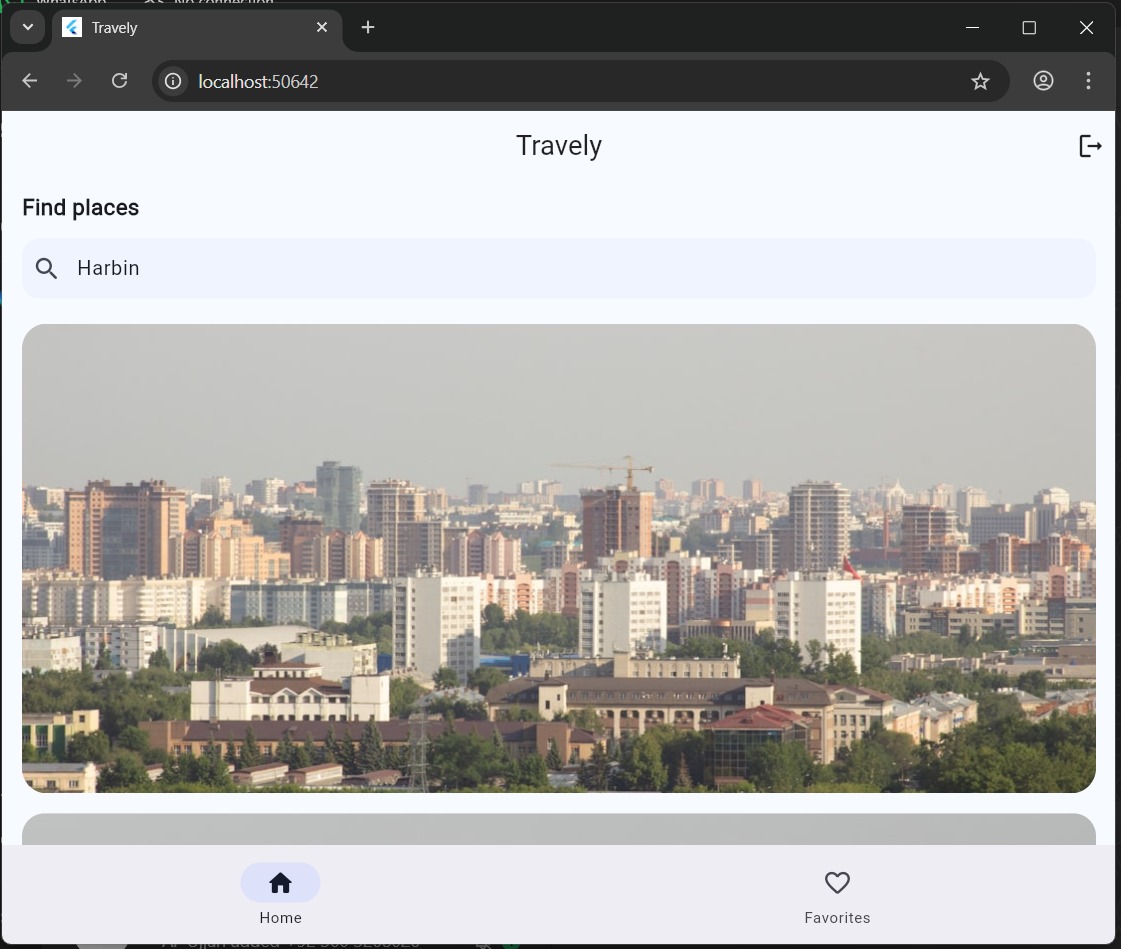
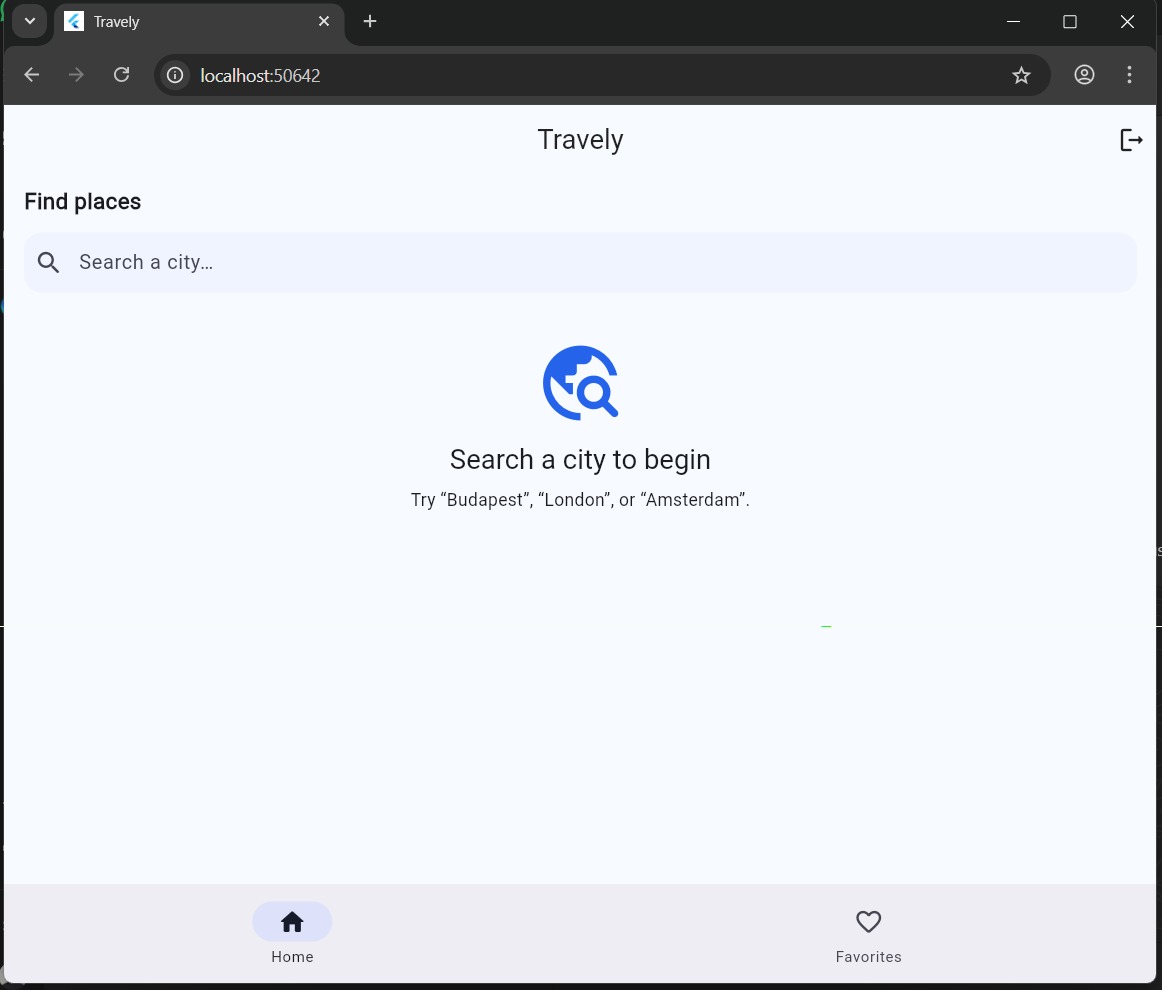
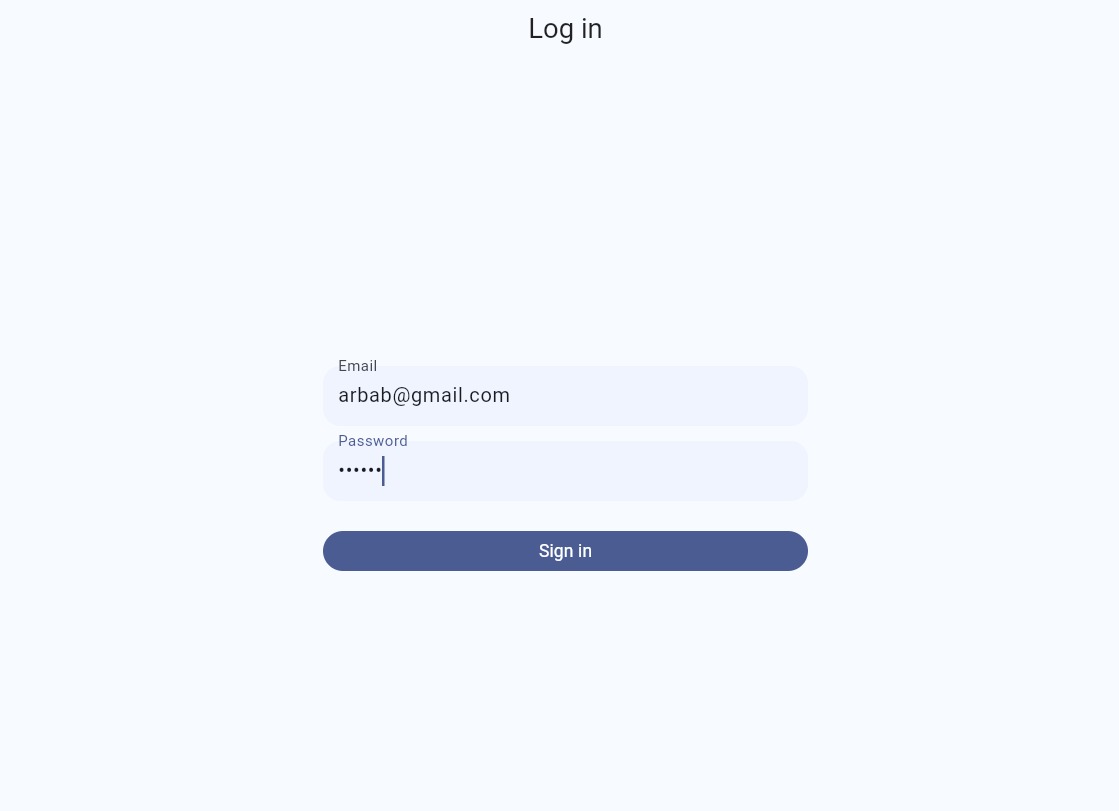
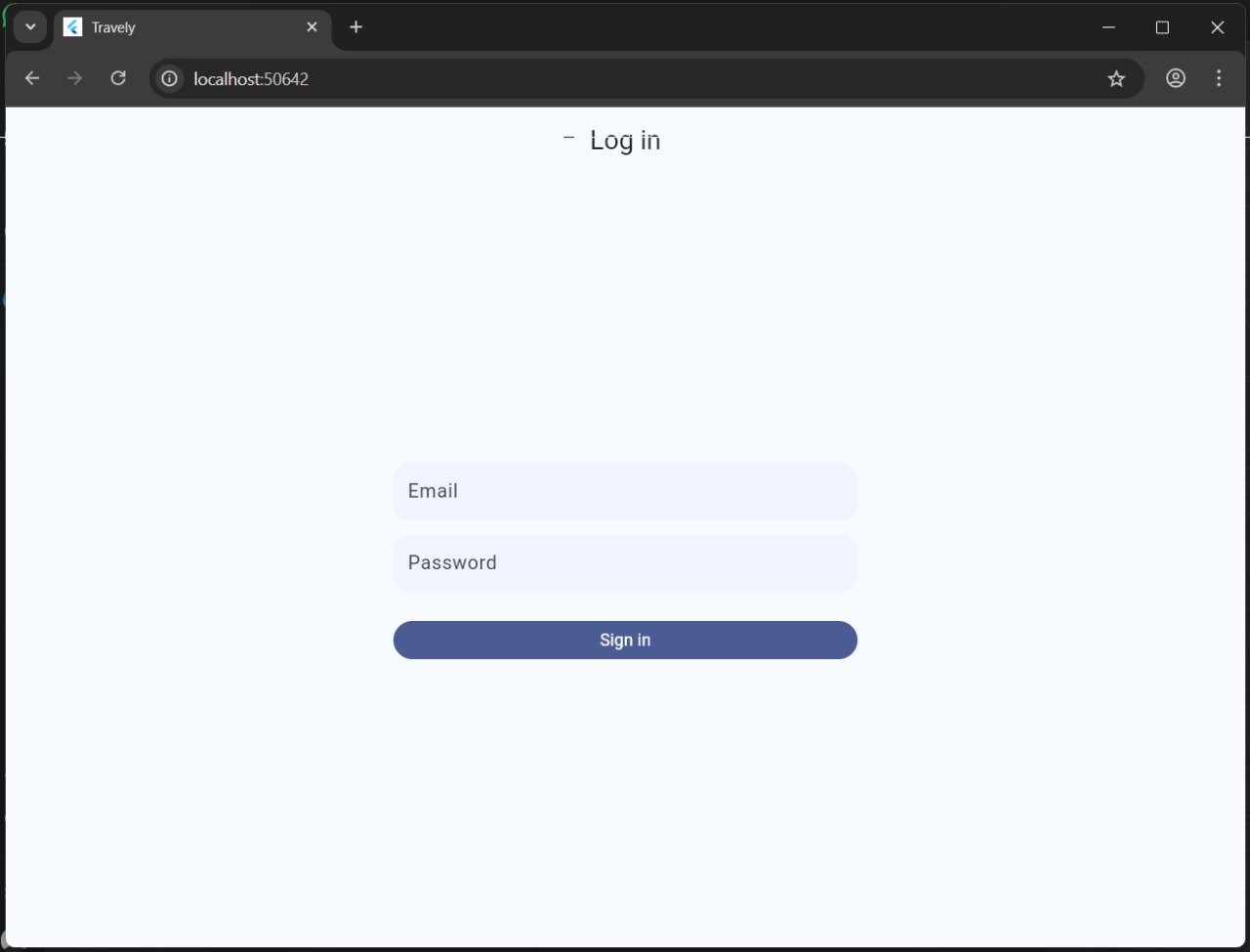
**Now Going to my Favorites Screen (to see all my addes places)**

****

**Now see on the Desktop/Tablet Screen**

****

****

****

**Responsiveness Highlights**

* Adaptive layouts using Flexible, Expanded, and MediaQuery.
* Dynamic scaling for phones and tablets.
* Consistent typography and iconography throughout.

**4. Data Storage (with Justification)**

Travely stores user data in **Firebase Firestore**, chosen for its:

* **Real-time updates:** Instantly syncs favorites across devices.
* **Cloud scalability:** Supports growth without server management.
* **Ease of integration:** Works natively with Flutter via cloud\_firestore.
* **Secure access:** Protected by Firebase Authentication & Firestore Rules.

Additionally, user authentication (email/password) ensures that each user’s favorites remain private and persist across sessions.

**5. APIs / Packages / Plug-ins Used**

These dependencies were selected to optimize maintainability, performance, and scalability while keeping the codebase lightweight and modular.

|  |  |  |
| --- | --- | --- |
| **Name** | **Purpose** | **Justification** |
| **http** | For REST API calls to Wikipedia & Unsplash | Lightweight and reliable networking |
| **provider** | State management | Clean, reactive architecture |
| **firebase\_auth** | User authentication | Secure, scalable login/signup |
| **cloud\_firestore** | Cloud database | Real-time sync for favorites |
| **cached\_network\_image** | Efficient image loading | Smooth scrolling and caching |
| **flutter\_dotenv** | API key management | Keeps credentials safe |
| **url\_launcher** | To open external Wikipedia links | Adds convenience |

**6. Issues and Bugs Encountered & Resolved**

|  |  |  |
| --- | --- | --- |
| **Issue** | **Description** | **Resolution** |
| **Geocode 403 Error** | Nominatim requests blocked due to missing headers | Added User-Agent header and optimized query parameters |
| **Firestore Permission Denied** | Unpublished database rules caused write failures | Published Firestore rules and re-tested document writes |
| **Favorites Not Updating** | Heart icon didn’t reflect real-time updates | Implemented stream-based listener via Firestore |
| **Duplicate Image Fetches** | Wikipedia sometimes lacked thumbnails | Added Unsplash fallback with city-based query |
| **Navigation Bug** | Splash page skipped bottom navigation | Routed authenticated users to HomeShell() instead of HomePage() |

Each issue was debugged using Flutter’s console output and fixed with best practices to ensure stable performance and maintainability.

**7. Maintainability and Code Quality**

* The app follows **Flutter’s Clean Architecture principles** with separate layers for UI, logic, and data.
* Code is modular, readable, and versioned on **GitHub**.
* Consistent naming conventions and comments improve maintainability.
* Provider ensures clean separation of concerns between UI and business logic.

--> **My**  **GitHub Repository :-** *[Travely](https://github.com/Farooquekk/Travely)*