**LifeLink – Blood Donation Mobile App**

*Team Members****: 22SW030 & 22SW063***

*Semester****: 6th (3rd Year)***

*Submitted to****: Ma’am Mariam***

**1. Real World Problem Identification**

In emergency situations, patients often face difficulties finding blood donors quickly. The process is usually slow and depends on manual communication, resulting in critical delays. Many existing platforms are either not user-friendly or lack proper verification of donors, leading to unreliable responses. Therefore, there is a need for a digital solution that enables instant and trustworthy communication between blood donors and recipients.

**2. Proposed Solution**

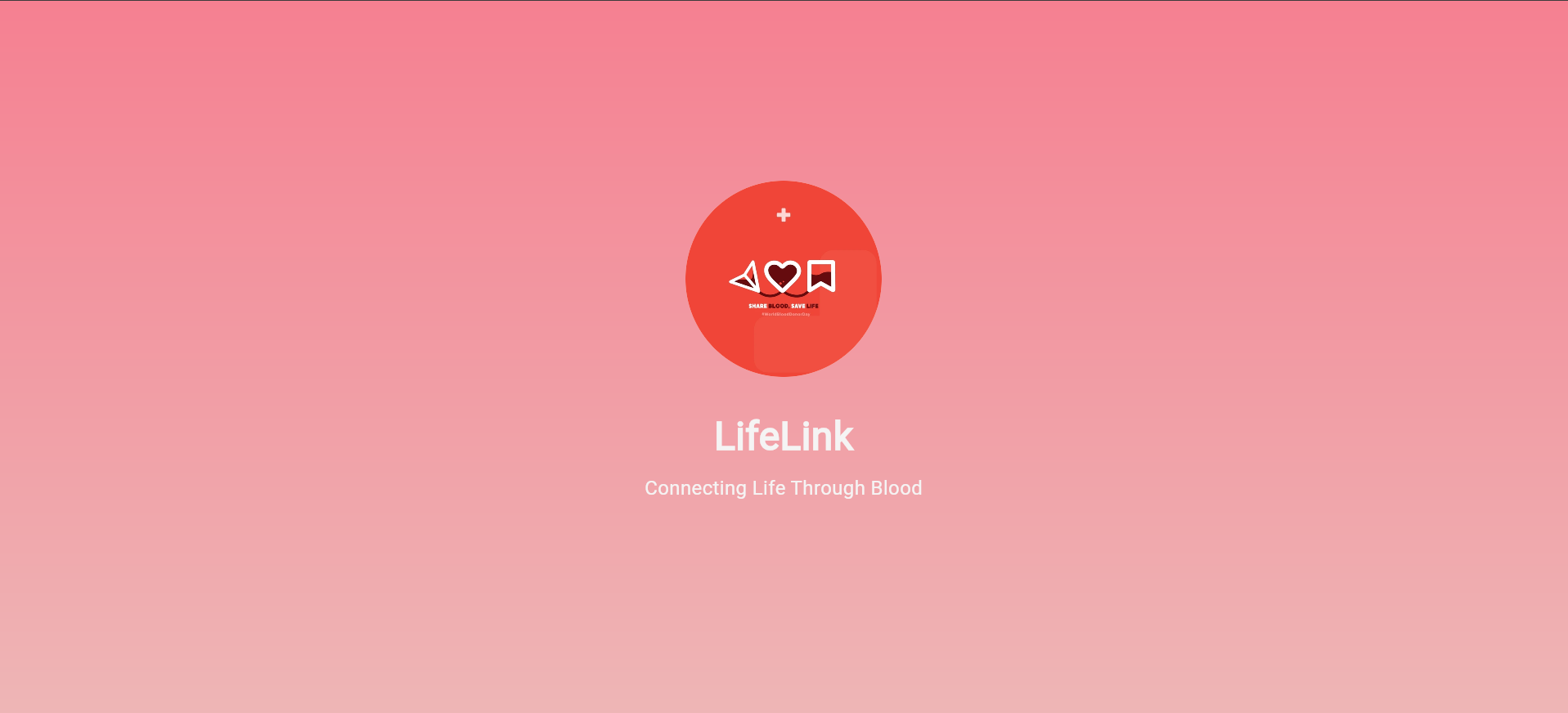
LifeLink is a Flutter-based mobile application designed to connect blood donors and recipients efficiently. The app provides an easy-to-use interface for users to register, log in, and either search for available donors or request blood when needed. The application ensures secure authentication and data management through Firebase integration. Its goal is to make blood donation faster, more accessible, and reliable for everyone.

**3. Responsive User Interfaces**

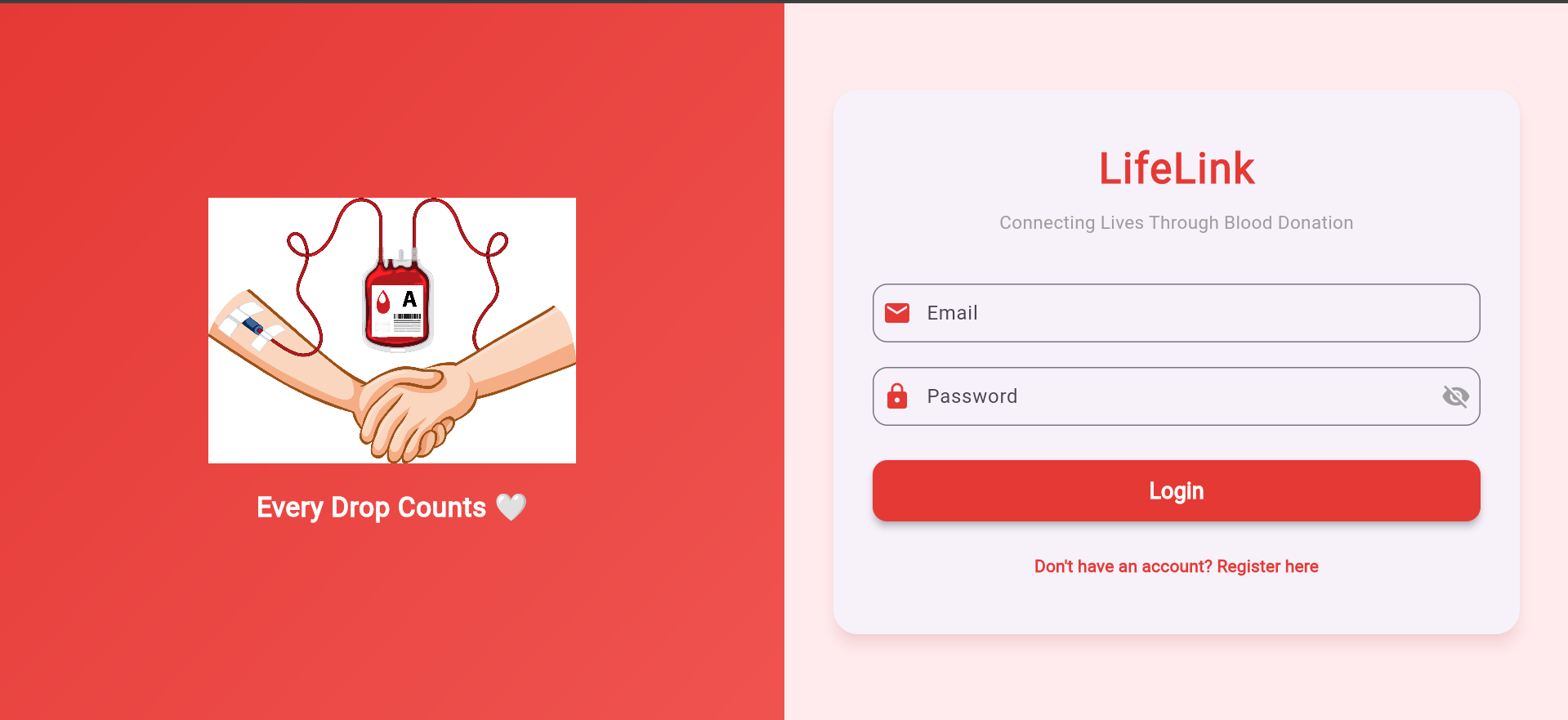
The application interface is designed to be fully responsive across different devices and screen sizes. Each screen maintains a consistent color scheme, typography, and spacing to ensure visual harmony and ease of use.

|  |  |  |
| --- | --- | --- |
| **Screen Name** | **Developer** | **Brief Description** |
| Splash Screen | 22SW063 | Displays the app logo with a smooth transition to the login screen. |
| Login Screen | 22SW030 | Provides secure login using email and password with Firebase Authentication. |
| Register Screen | 22SW030 | Enables new users to create an account by entering name, email, and password. |
| Home Page | 22SW030 | Main dashboard with options to find donors, request blood, and view user profiles. |
| Profile Page | 22SW063 | Displays user information such as name, email, and blood group. |
| Edit Profile Page | 22SW063 | Allows users to update and save personal information. |
| Search Donor Page | 22SW030 | Enables users to search for registered blood donors nearby. |
| Request Donor for Blood Page | 22SW063 | Provides a form for recipients to request specific blood types. |

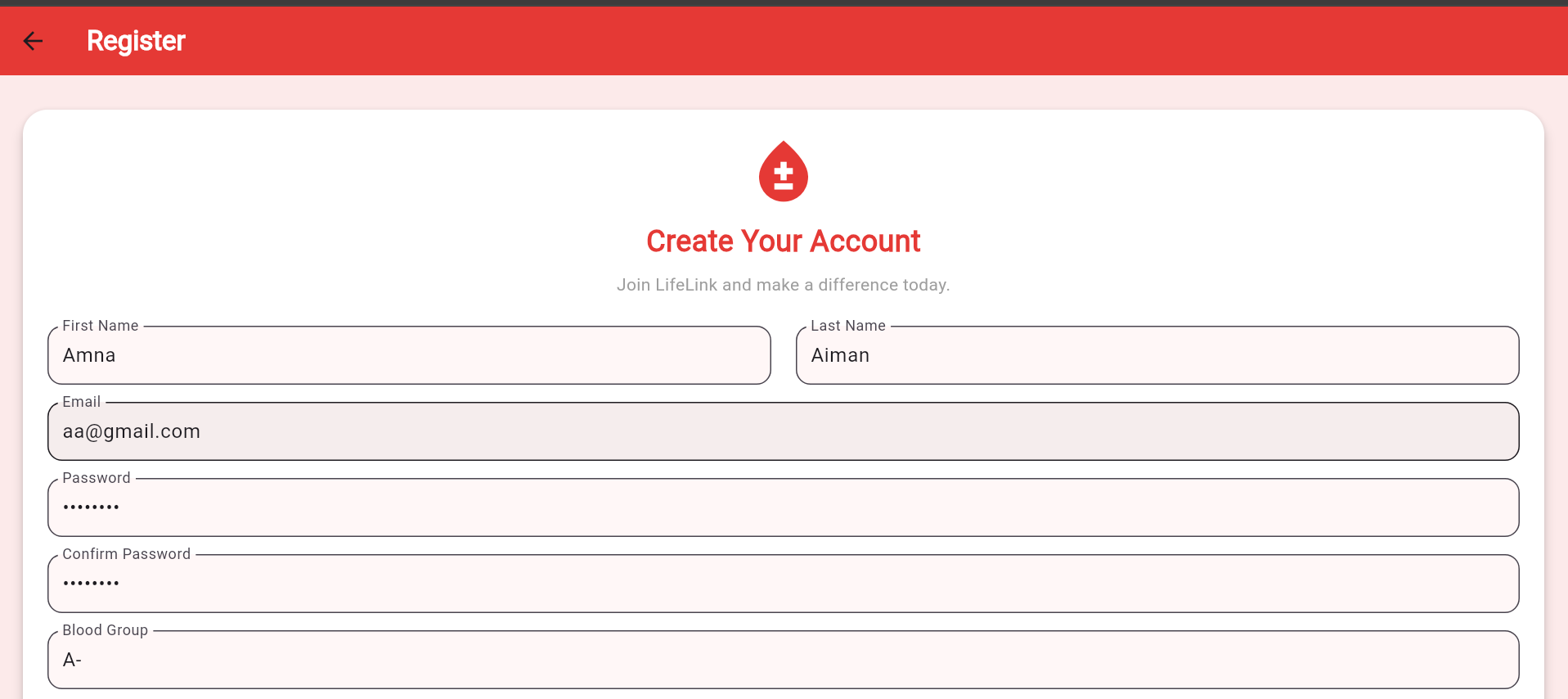
1. **Splash Screen – 22SW063**

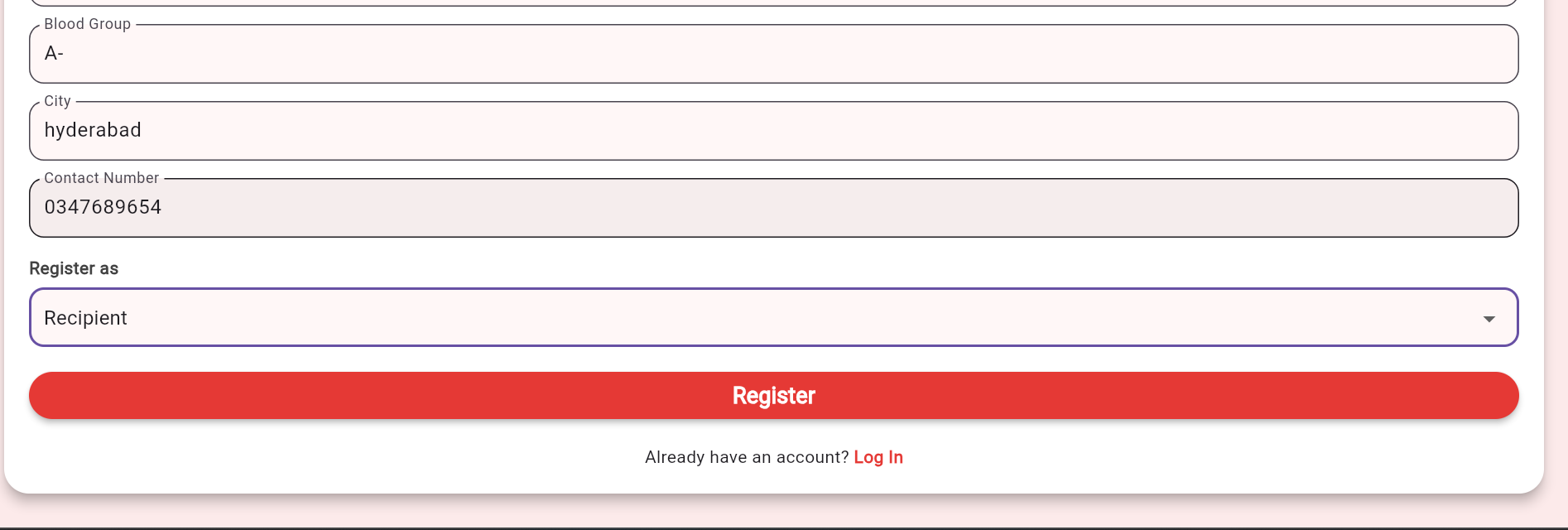


1. **Login Screen – 22SW030**

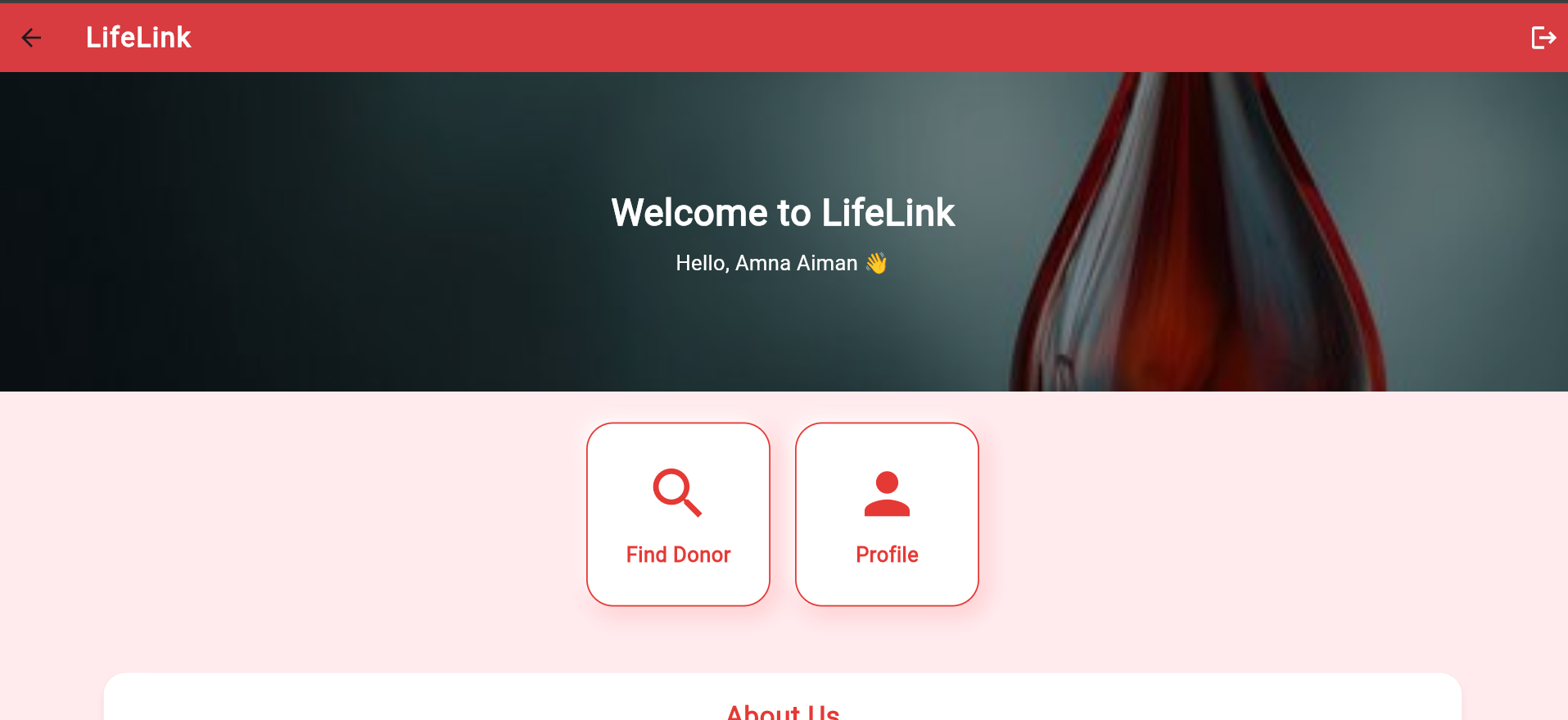


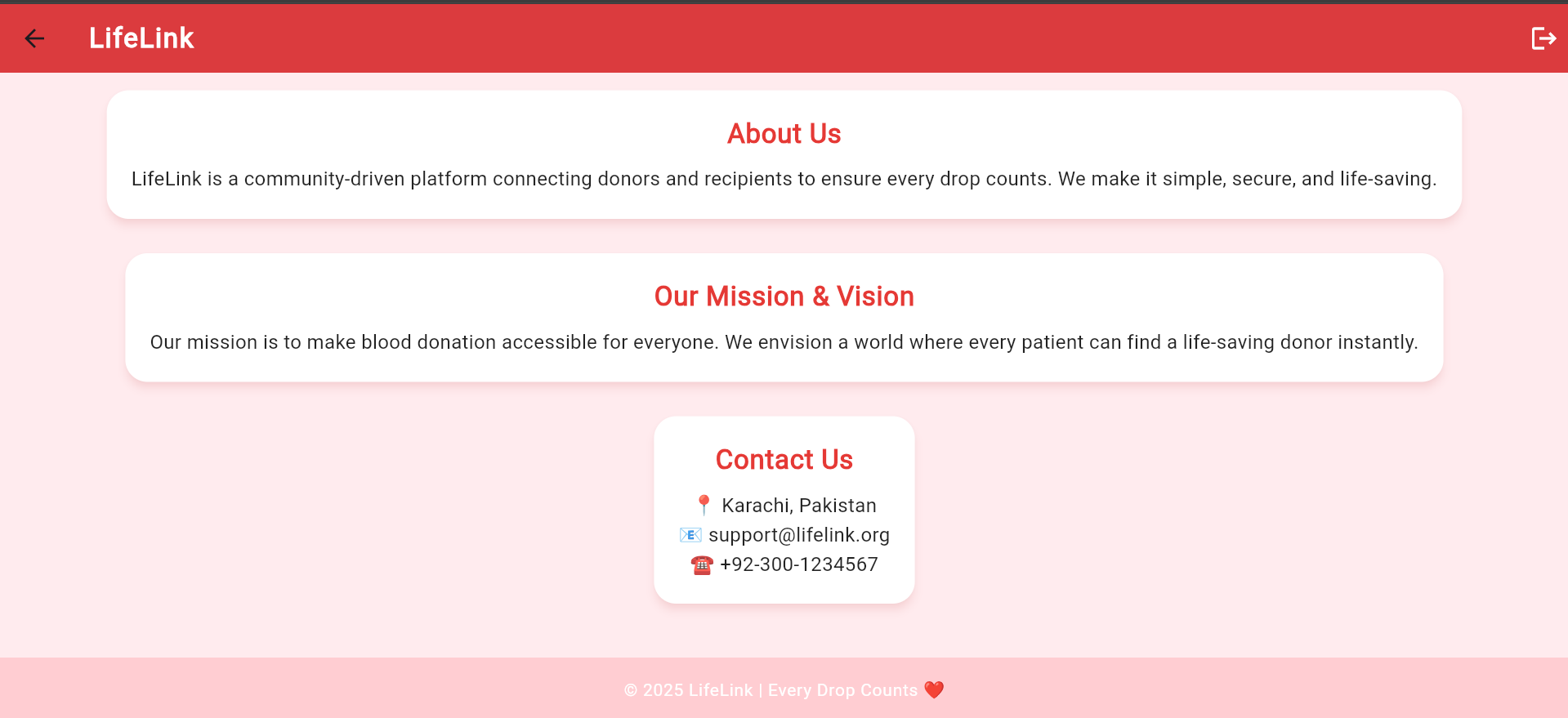
1. **Register Screen – 22SW030**



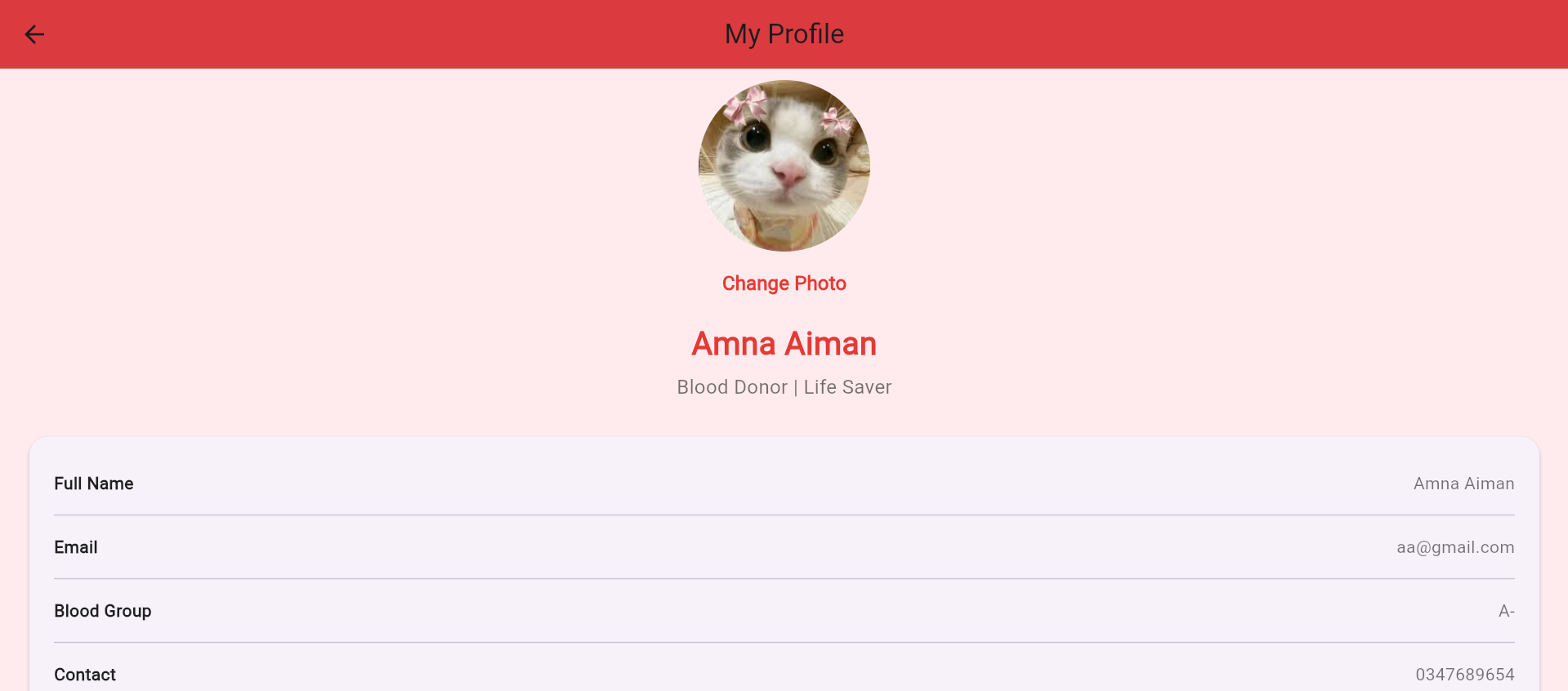


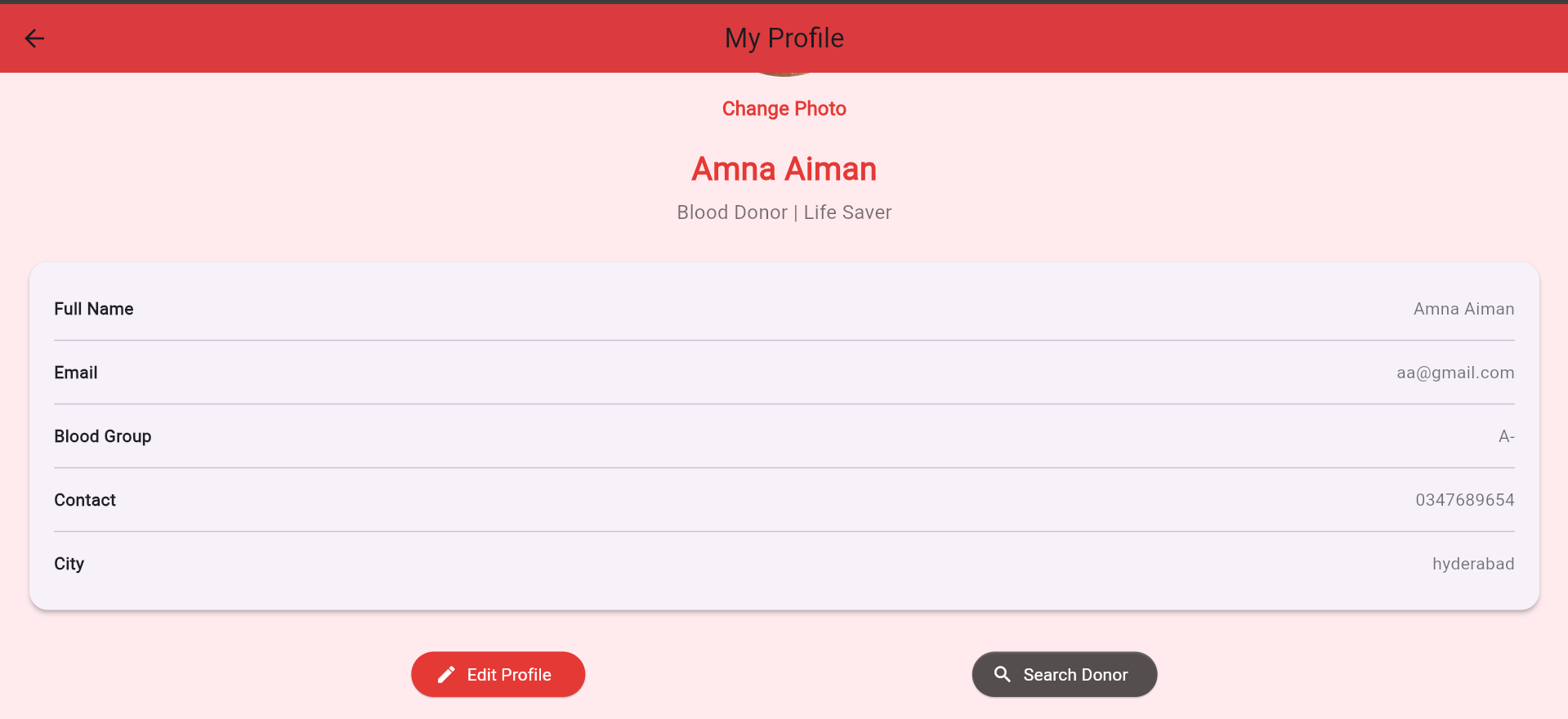
1. **Home Screen – 22Sw030**



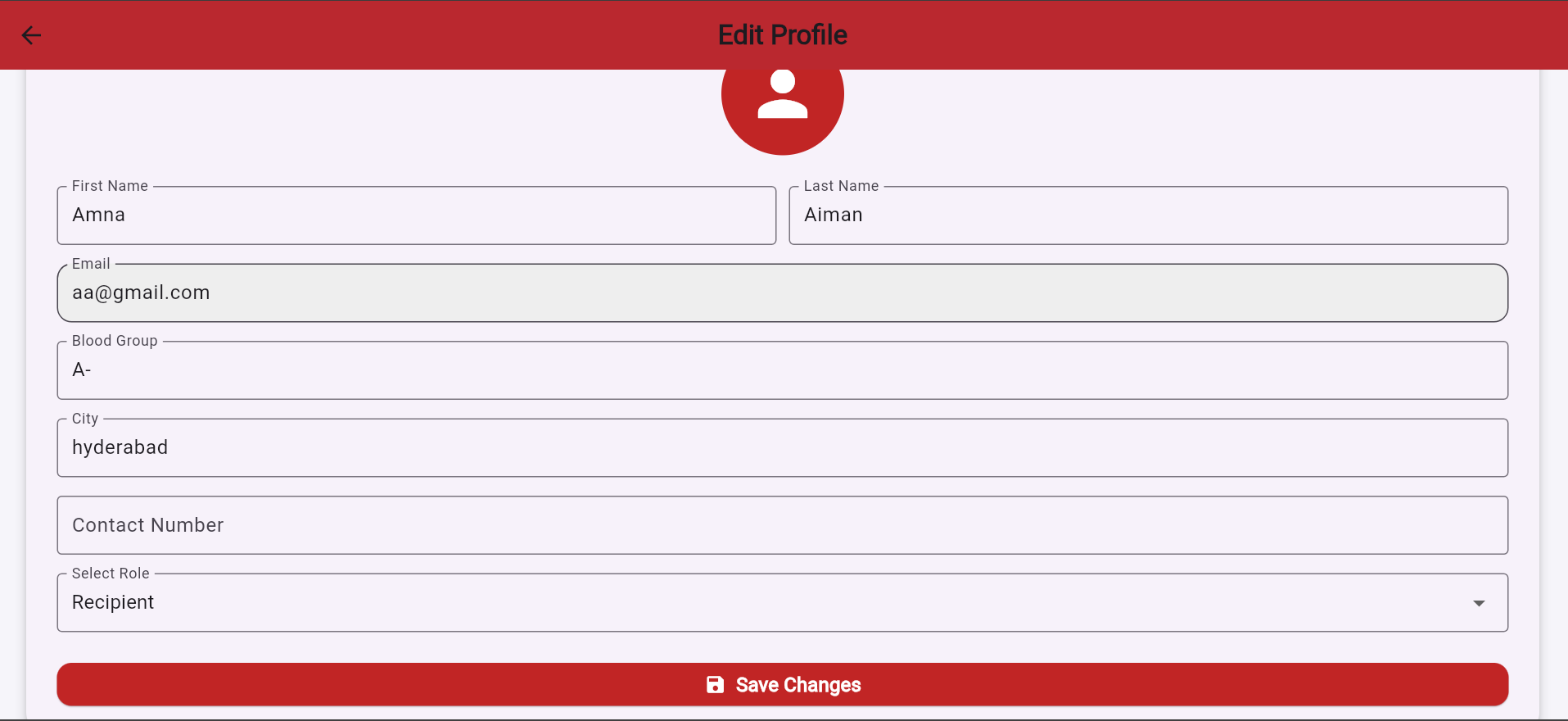


1. **Profile Screen – 22SW063**

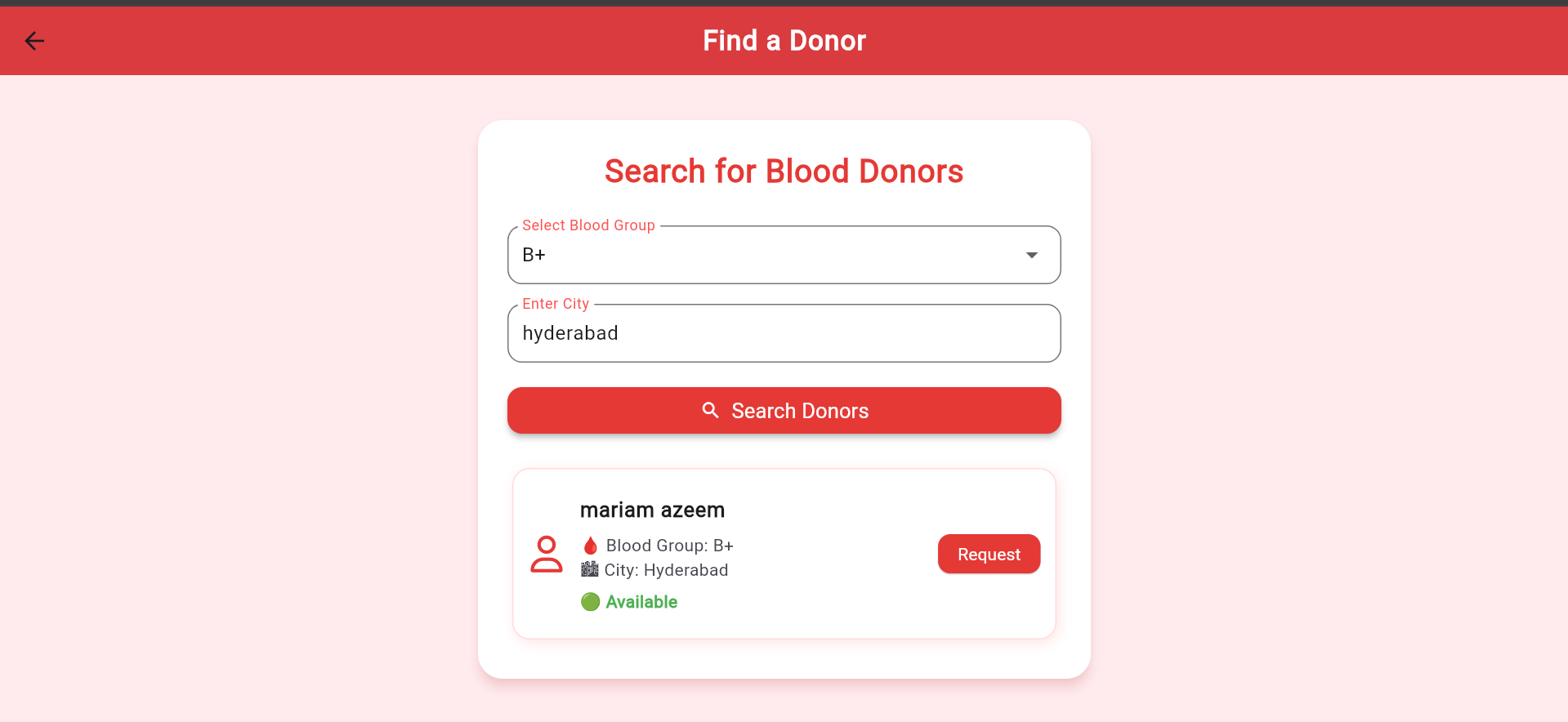




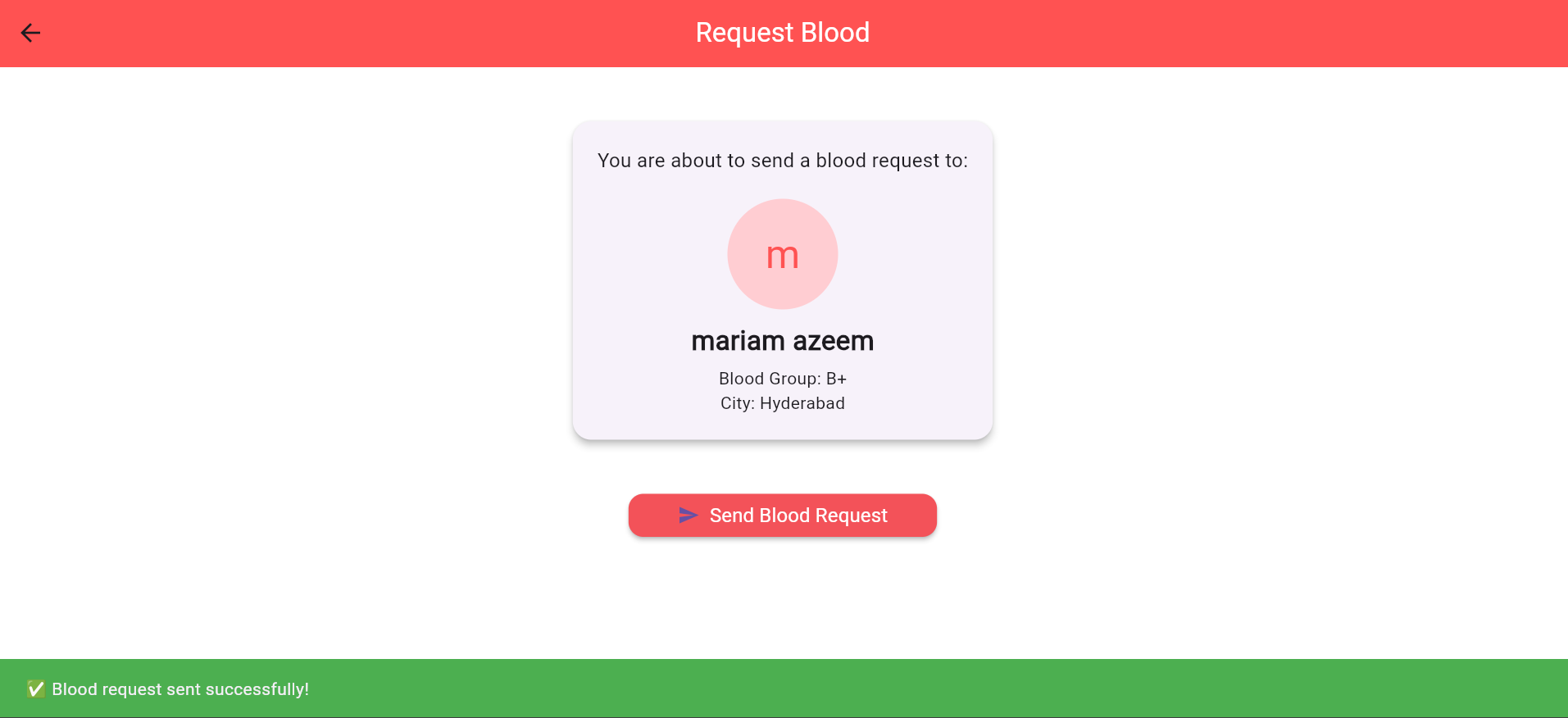
1. **Edit Profile Screen – 22SW063**



1. **Donor Search Screen – 22SW030**



1. **Request Blood Screen – 22SW063**



**4. Data Storage**

LifeLink uses **Firebase** as the primary backend service for authentication and data storage.

**4.1 Justifications for Using Firebase:**

* **Security:** Firebase Authentication ensures that all user data is safely managed.
* **Real-Time Syncing:** It provides instant data synchronization across users and devices.
* **Easy Integration:** Flutter’s compatibility with Firebase simplifies setup and coding.
* **Scalability:** Firebase can handle both small and large user bases without additional configuration.
* **Cloud Storage:** Enables safe and centralized storage of user and donor data.
* Firebase services utilized include:
* **Firebase Authentication** – to manage secure user login and registration.
* **Cloud Firestore** – to store donor and recipient data dynamically.

**5. APIs / Packages / Plug-ins (Optional Section)**

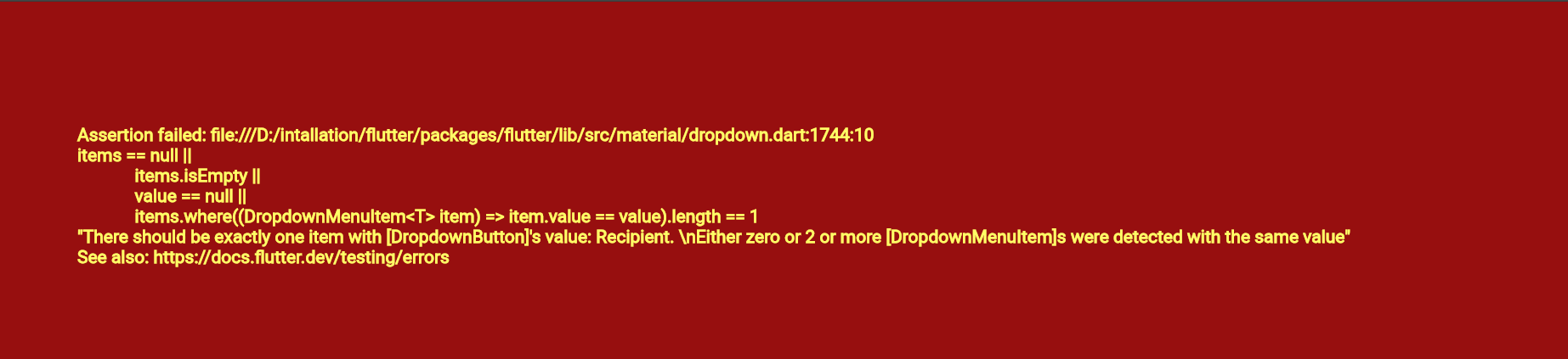
The project uses several Flutter packages to enhance functionality and streamline development.

|  |  |
| --- | --- |
| **Package / Plug-in** | **Purpose / Justification** |
| firebase\_core | Initializes Firebase services in the Flutter application. |
| firebase\_auth | Manages authentication for login and registration. |
| cloud\_firestore | Stores and retrieves data for donors and recipients. |
| flutter/material.dart | Provides core Flutter UI widgets and layout structures |
| image\_picker | Allows users to select and upload profile images. |

**6. Bugs Encountered and Resolved during Development**

During the development of the LifeLink application, several issues and bugs were encountered. These primarily occurred during UI design, Firebase integration, and debugging phases. Each issue was carefully analyzed and resolved to ensure the app’s functionality and user experience remained stable.

* 1. **DropdownButton Value Assertion Error – 22SW063**



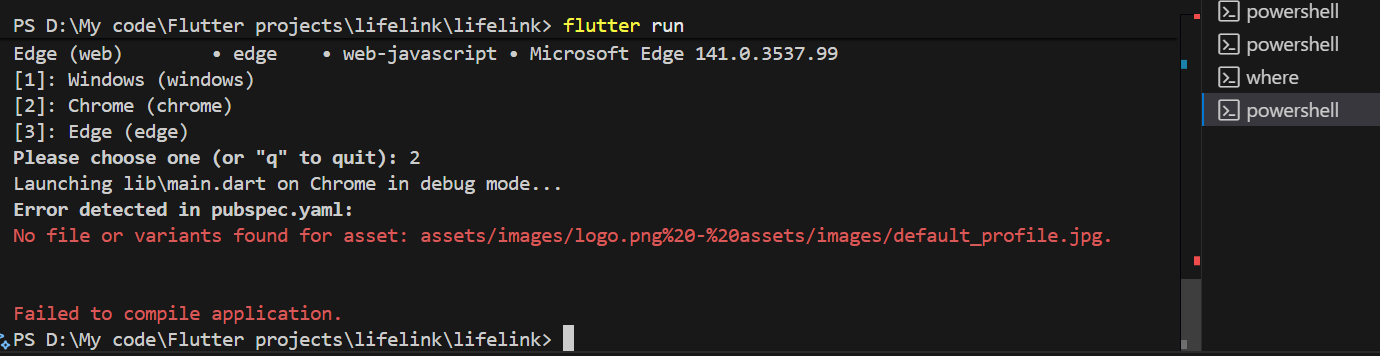
**Description:**  
While testing the profile or registration screen, the app crashed with a red error screen related to the DropdownButton widget. The issue occurred because the dropdown’s current value (Recipient) did not match any of the available items in the dropdown list.

**Root Cause:**  
 The dropdown items list was defined as:

final List<String> \_roles = ['Donor', 'Receiver', 'Guest'];

so i replace Receiver with Recipient.

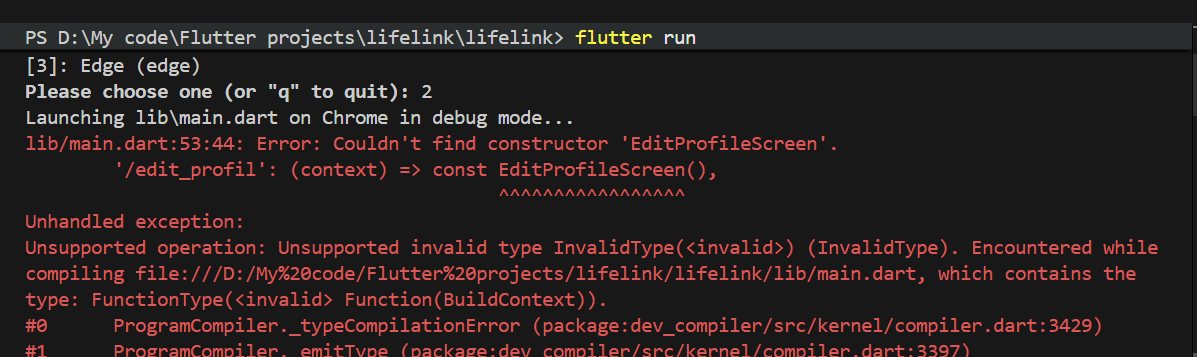
* 1. **Asset Loading Error —Indentation Issue in pubspec.yaml – 22SW063**

****

**Description:**  
The application failed to compile due to an incorrectly formatted asset entry in the pubspec.yaml file. The error appeared because of **improper indentation or spacing**, causing Flutter to misread multiple asset paths as a single invalid path.

**Resolution:**  
 Fixed the indentation by properly aligning all asset paths.

**6.3 Missing Import Causing Compilation Failure in main.dart – 22SW063**

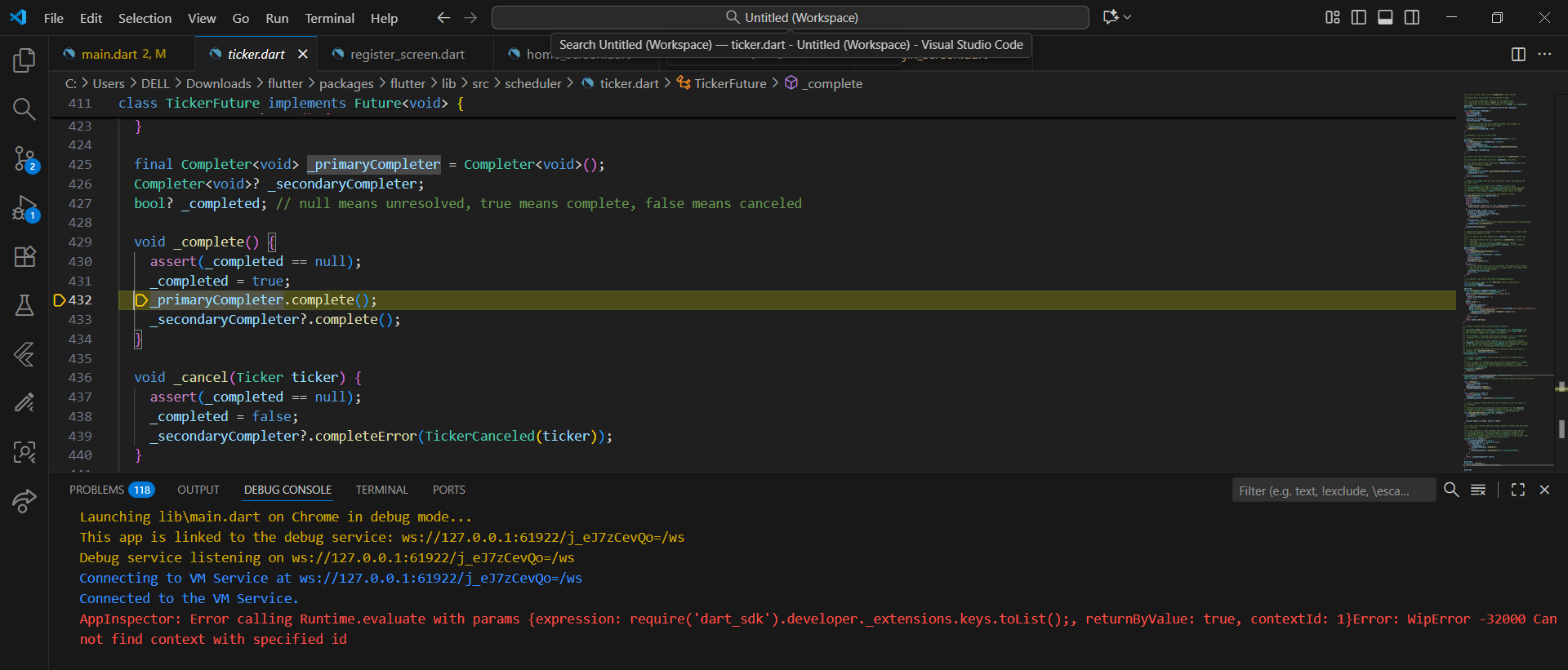
****

**Description:**  
The Flutter app failed to compile when running on Chrome. The error indicated that the EditProfileScreen constructor could not be found. This happened because the edit\_profile\_screen.dart file was not imported in main.dart, even though it was being used inside the route configuration.

**Resolution:**  
 Added the missing import statement at the top of main.dart:

import 'screens/edit\_profile\_screen.dart';

**6.4 VM Service Connection Error**

****

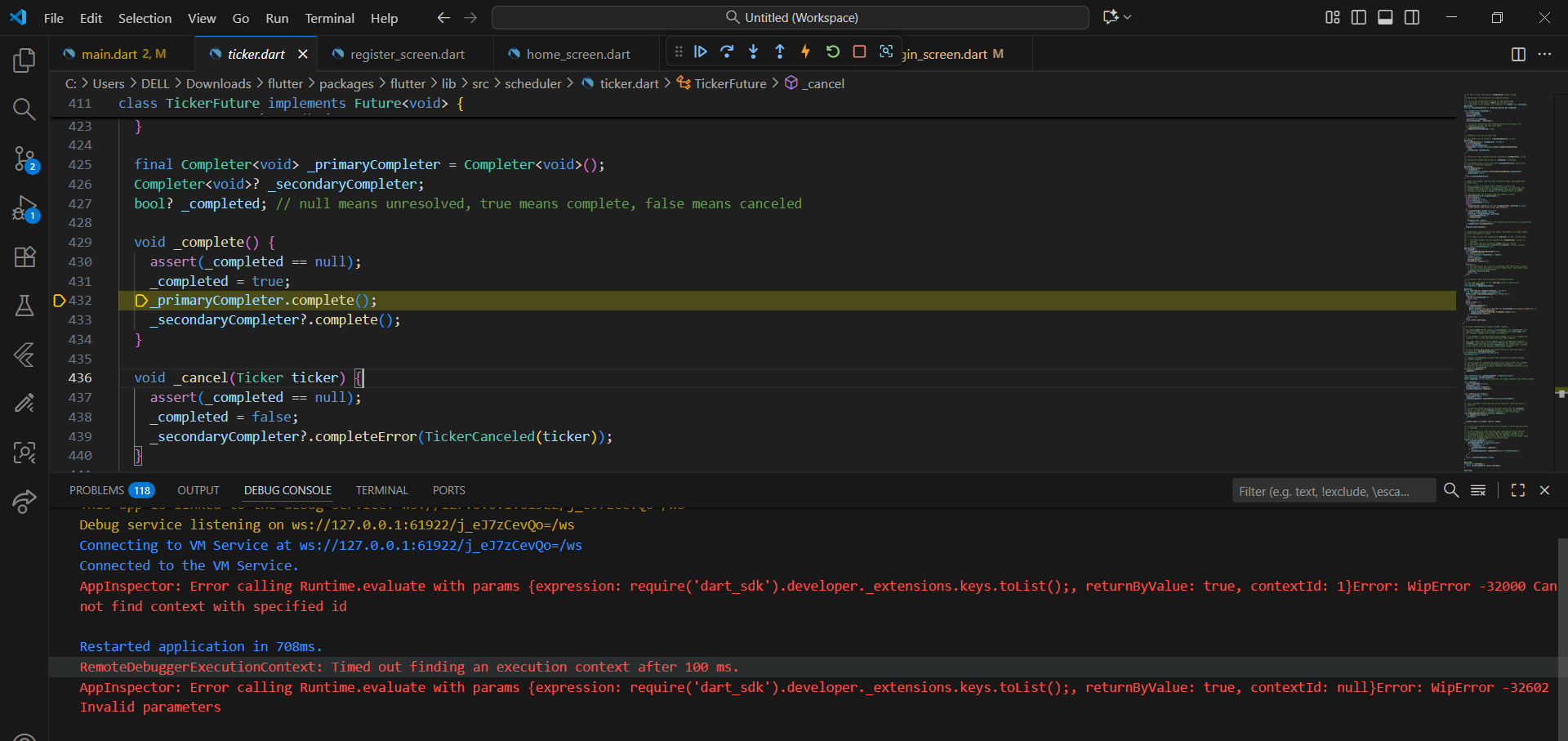
**Description**

A VM Service connection error that occurred during Flutter app debugging. The error indicates that Flutter DevTools lost synchronization with the running app's execution context, likely due to a hot reload operation. This is a common, non-critical debugging tool issue that doesn't affect the app's functionality. The error was resolved by performing a hot restart of the application

**Resolution**

Perform a hot restart of the application to re-establish synchronization between Flutter DevTools and the app.

**6.5 Flutter Failing to Connect to Dart Debugger**

****

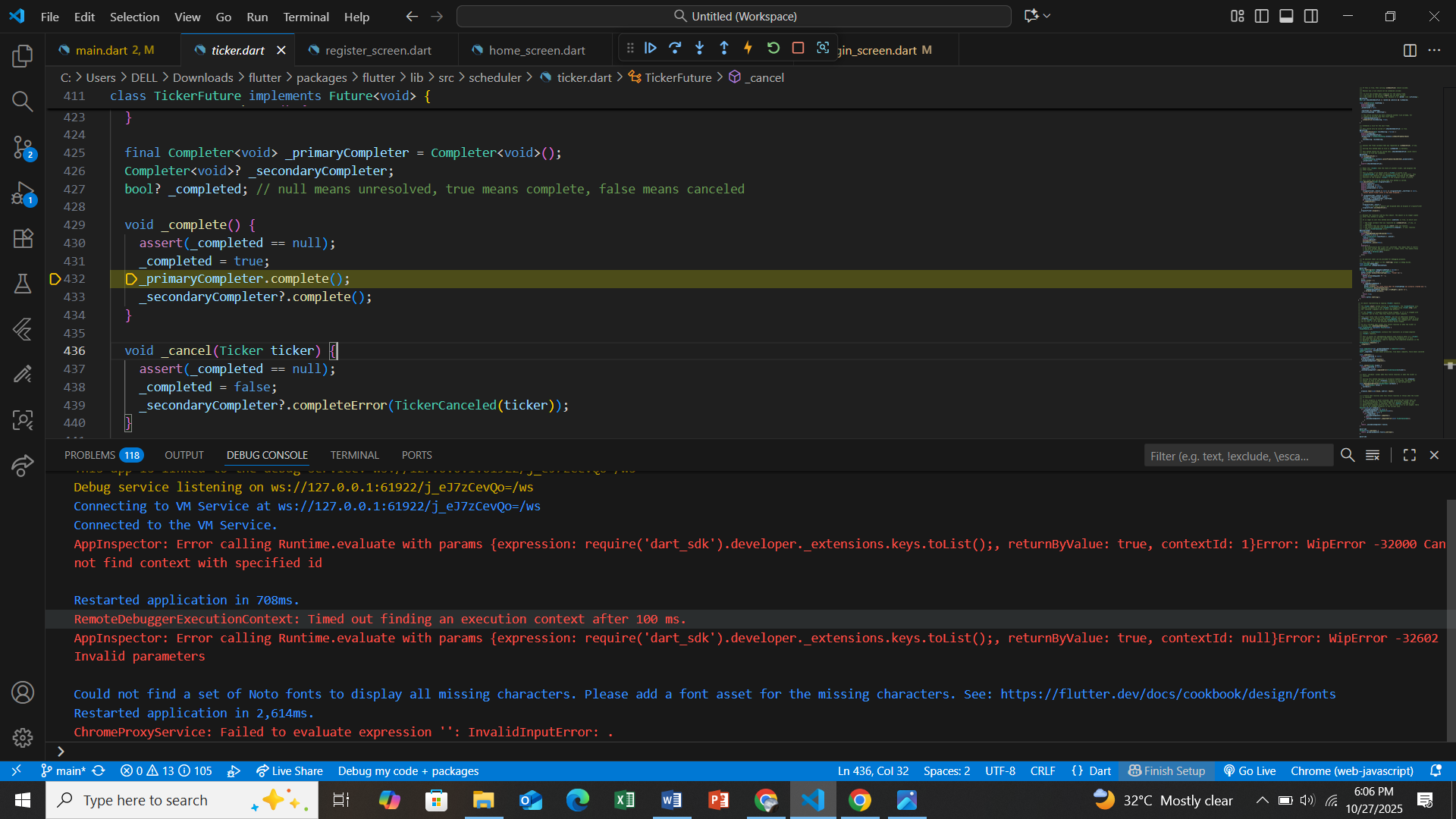
**Description**

Encountered this error due to Flutter failing to connect to the Dart debugger after restarting the app. It happens due to a timeout or invalid context issue and can be fixed by restarting the debug session or updating the Flutter and Dart extensions.

**Resolution**

* Restarted the ****debug session**** in your IDE.
* The issue persisted, updated both the ****Flutter**** and ****Dart extensions**** to their latest versions.

**6.6 Missing Noto Fonts**

****

**Description**

This occurred because Flutter couldn’t find the required Noto fonts to display certain characters. It usually occurs when specific Unicode characters are missing from the default font set and can be fixed by adding a compatible font asset in the pubspec.yaml file.

**Resolution**

Add a compatible font asset (such as Noto fonts) in the pubspec.yaml file to ensure proper rendering of all characters**.**

**7. Conclusion**

The LifeLink project successfully demonstrates the use of Flutter and Firebase to build a responsive and purpose-driven mobile application. By focusing on clean design, secure authentication, and effective collaboration, the project provides a practical solution to an important social issue — connecting blood donors and recipients efficiently.  
 This experience enhanced our understanding of Flutter app development, Firebase integration, and teamwork using GitHub.