**CHAPTER FOUR**

**IMPLEMENTATION AND TESTING/RESULTS AND DISCUSSION**

**4.1 Implementation**

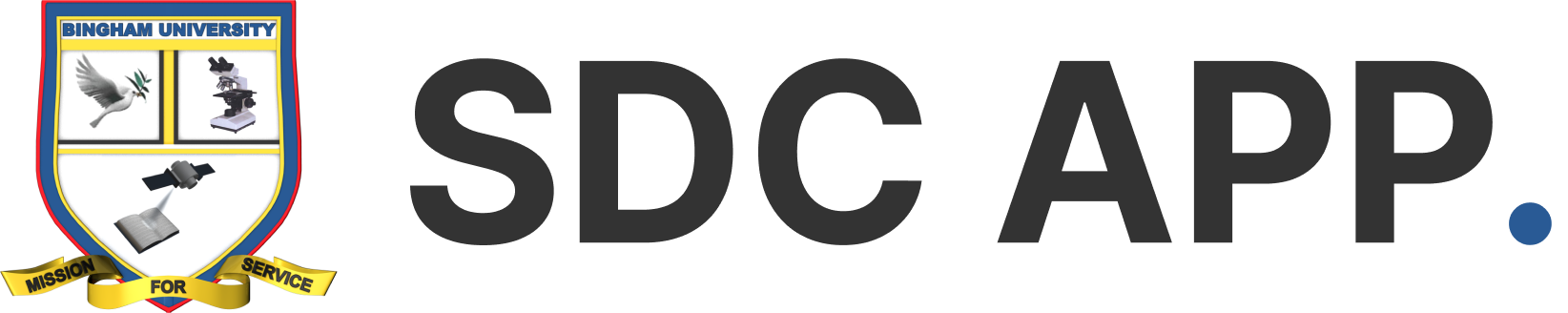
The implementation of the Student Disciplinary Committee (SDC) Application contains several key aspects, including the design of screens for the mobile application, the use of open-source tools, implementing the mobile app and hosting and tracking of changes to the codebase using GitHub.

**4.1.1 Design**

The design of the mobile app screens was a critical aspect of the SDC Application development. Using Figma, a collaborative design tool, allowed for the creation of intuitive and user-friendly interfaces to be used in managing SDC cases. The design focused on ease of use, with clear navigation and a visually appealing layout. Screens were designed to allow users to create new cases, add case offenders, specify case types, view case details, and communicate what a case is about using clear descriptions. The below are some of the screenshots.

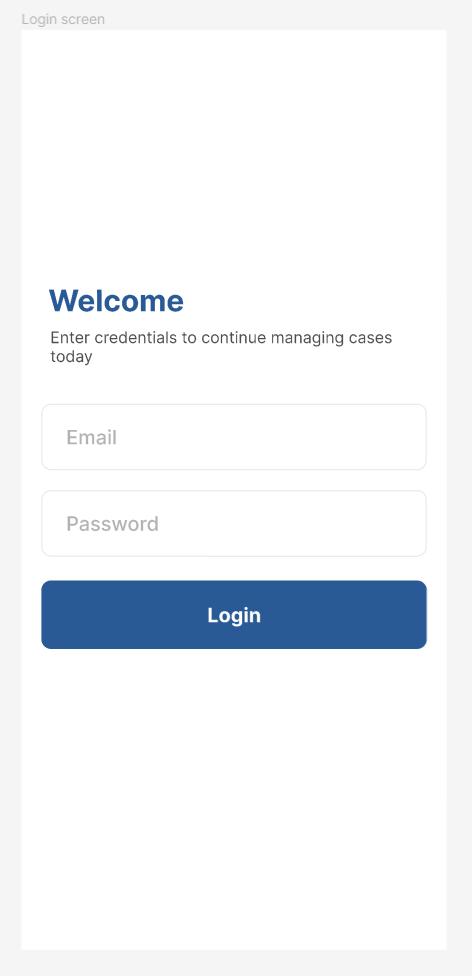
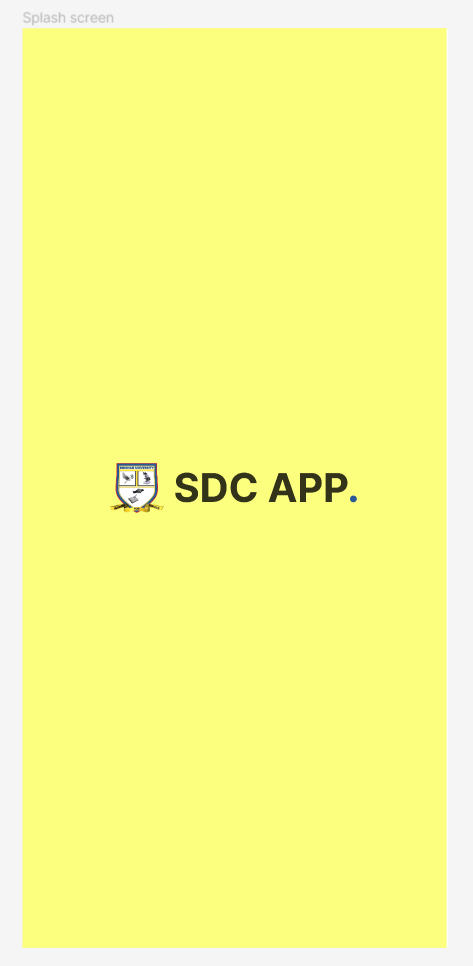
**Figure 1**

*Logo design*

****

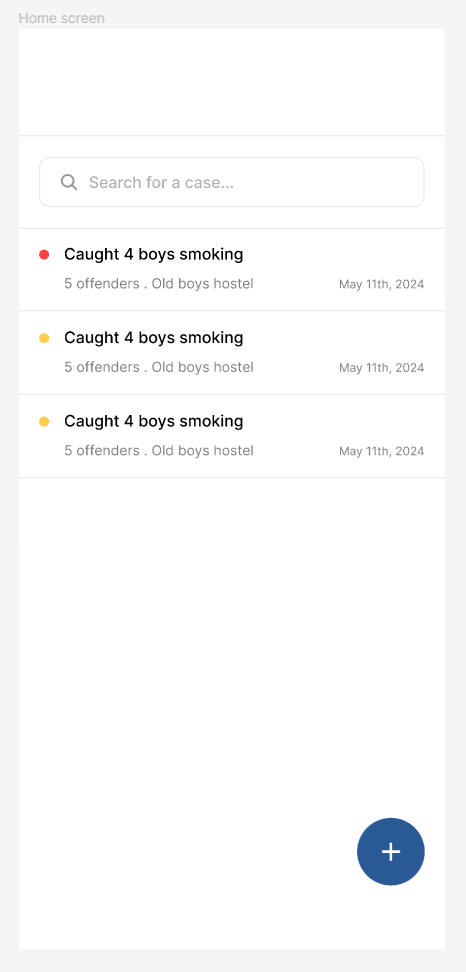
**Figure 2**

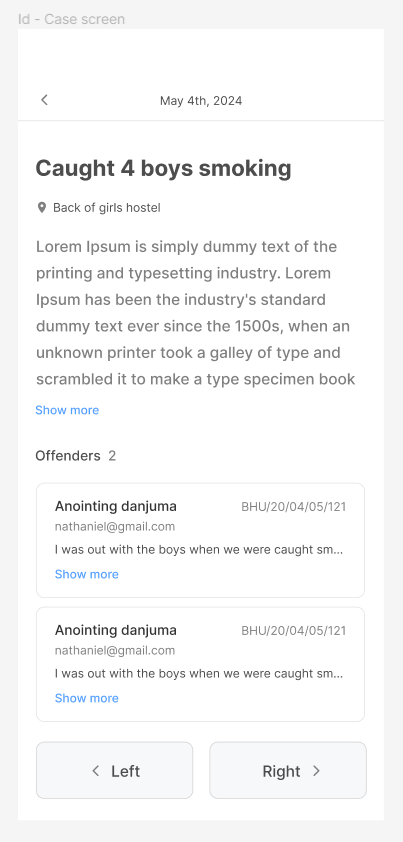
*Splash screen and the login screen design*



**Figure 3**

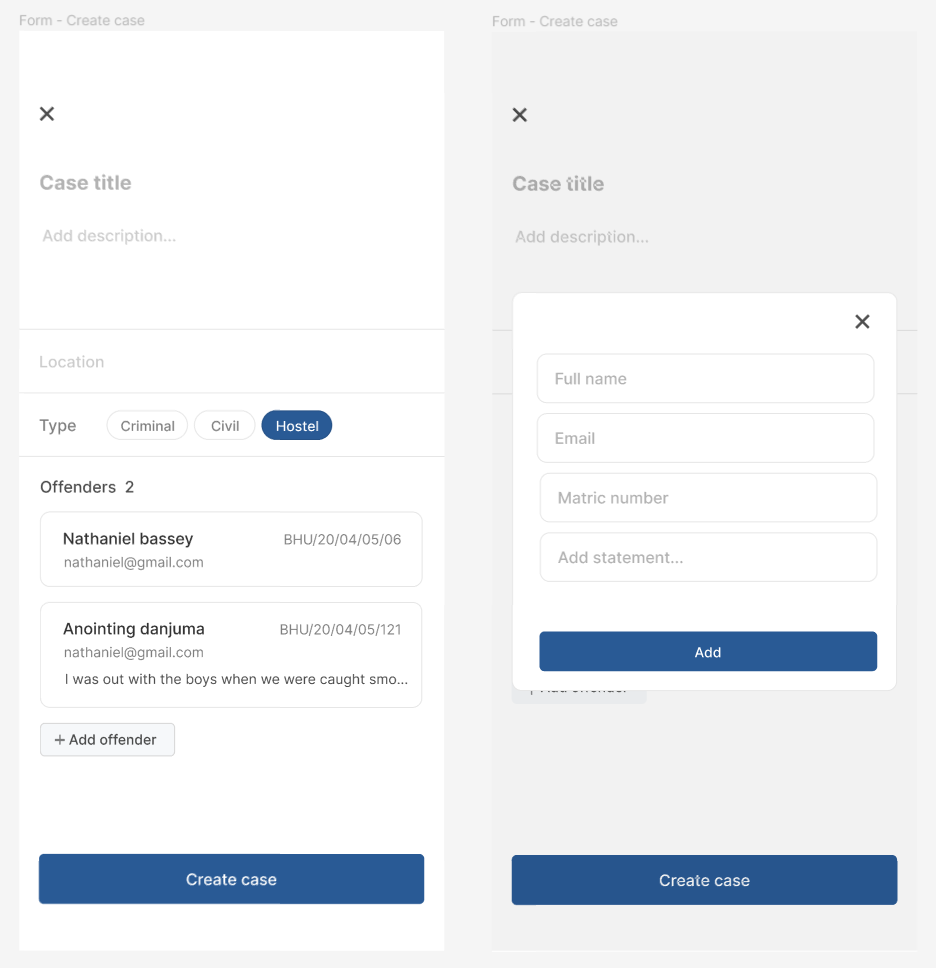
*The home screen design and a particular case screen design*

****



**Figure 4**

*The case form design and an offenders modal design*

****

**4.1.2 Tools**

Several open-source tools were instrumental in building the SDC Application:

**DrizzleORM:** Used for managing the application's database queries, providing a simple, lightweight and efficient way to handle interactions. And it’s also type safe, which makes it that much indispensable.

**TursoDB:** Employed as a serverless SQLite for production, offering a lightweight and scalable database solution.

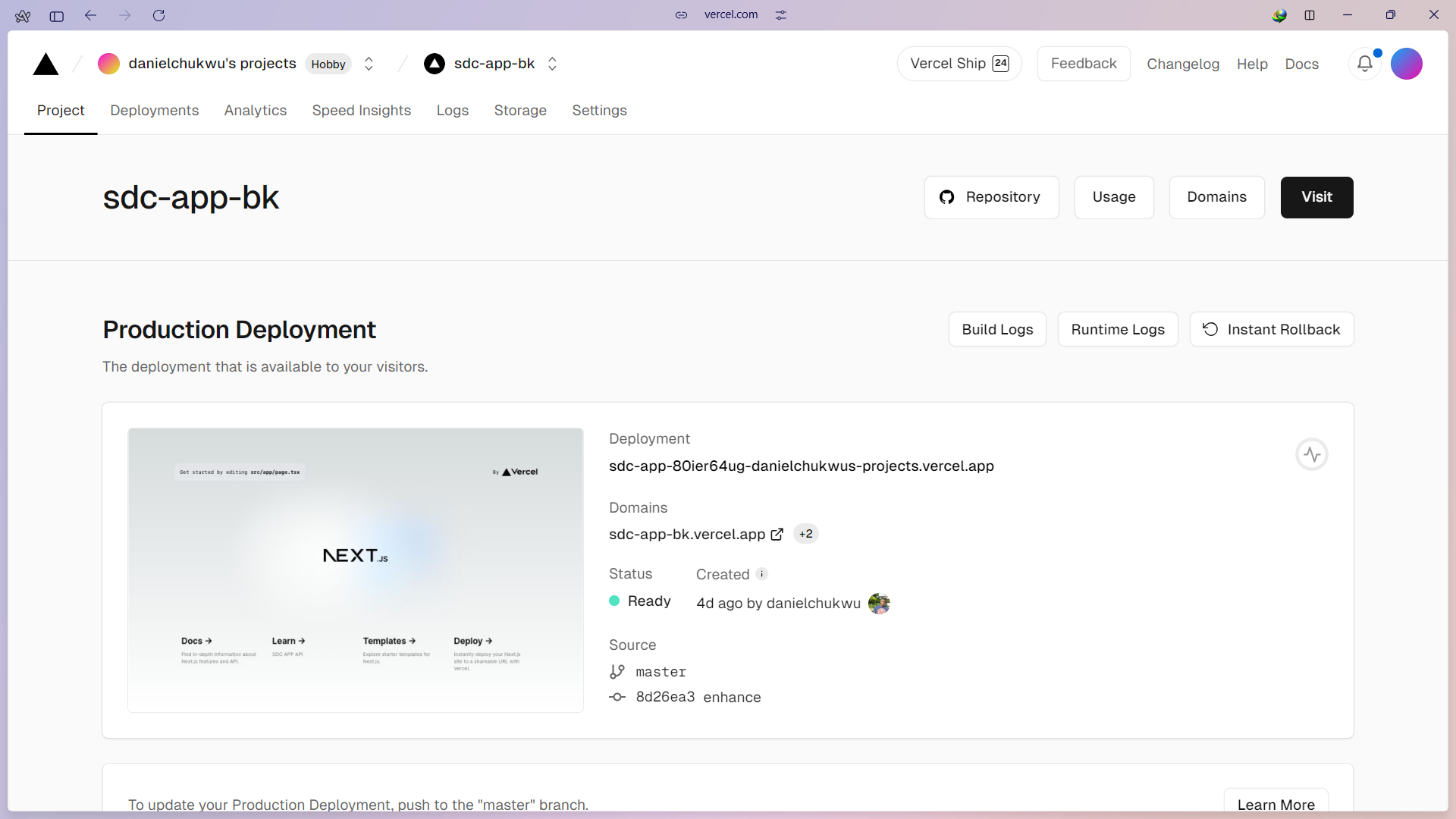
**Zustand & React Query:** Used for managing the application's state and handling data fetching, providing a robust and efficient way to manage data (Very important).

**React Native & Expo SDK 51.0:** Utilized for building the mobile app, allowing for the development of cross-platform applications with a single codebase.

**Bun, Next API Routes & Hono:** These tools were used to build and run the backend code on the edge using the vercel platform, enabling faster performance and improved user experience.

**Figure 5**

*The backend deployed on vercel*

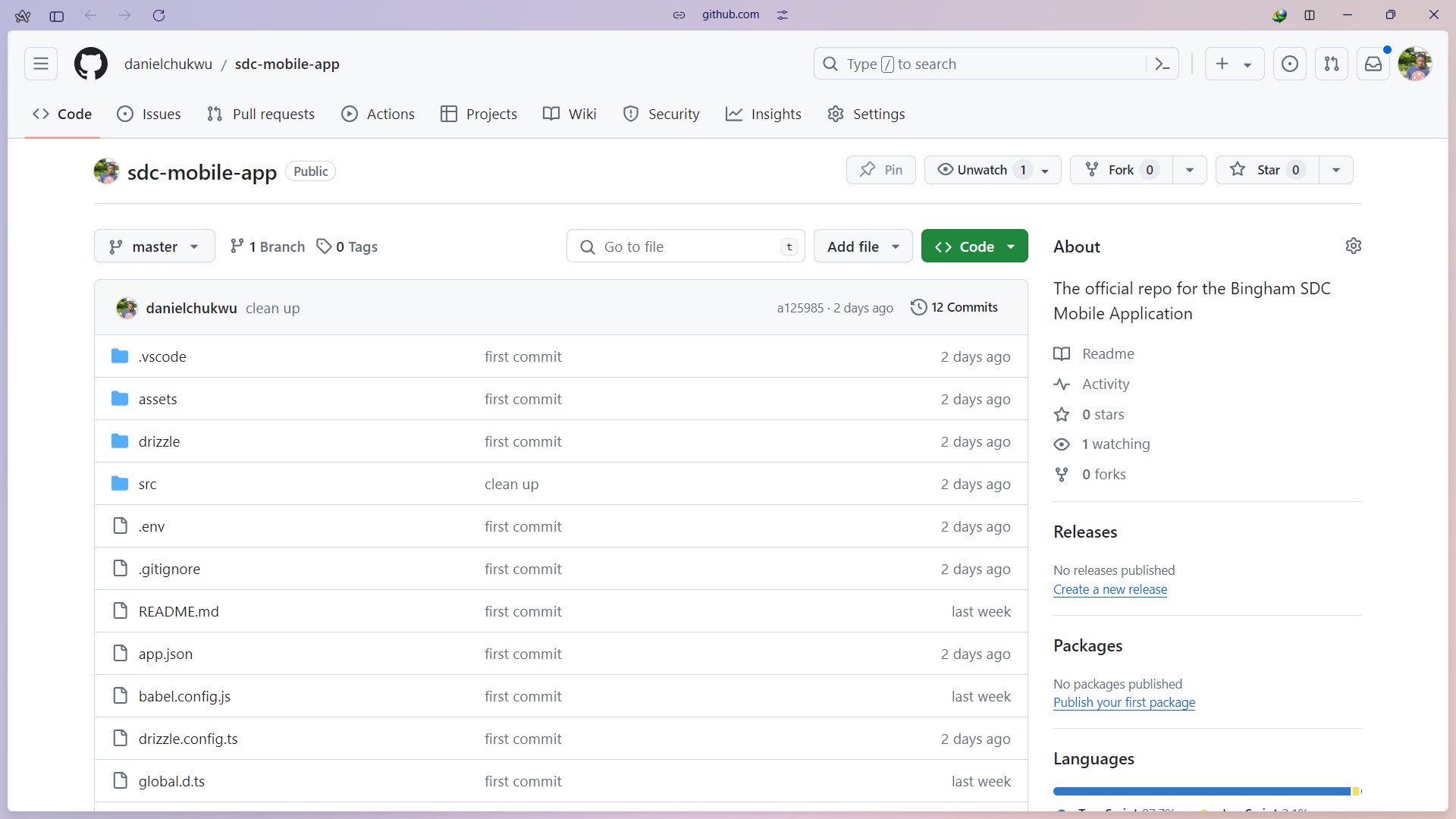


**4.1.3 GitHub Repository Codebase Screenshots**

The backend and frontend codebases for the SDC Application were hosted in separate GitHub repositories. The backend repository contains the server-side code, including API endpoints and database interactions, while the frontend repository contains the code for the mobile app interface.

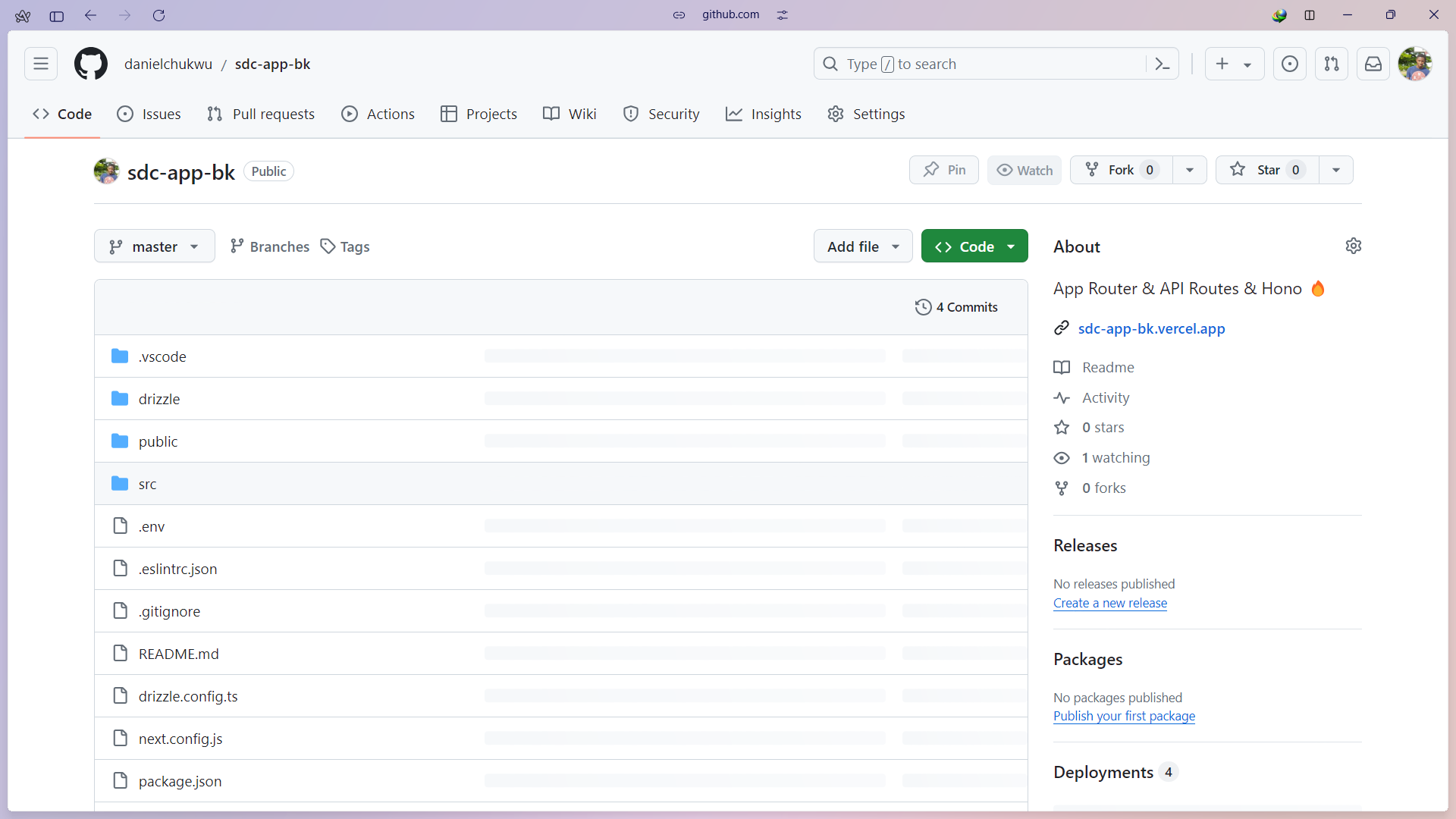
**Figure 6**

*The mobile interface github repository*



**Figure 7**

*The backend github repository*

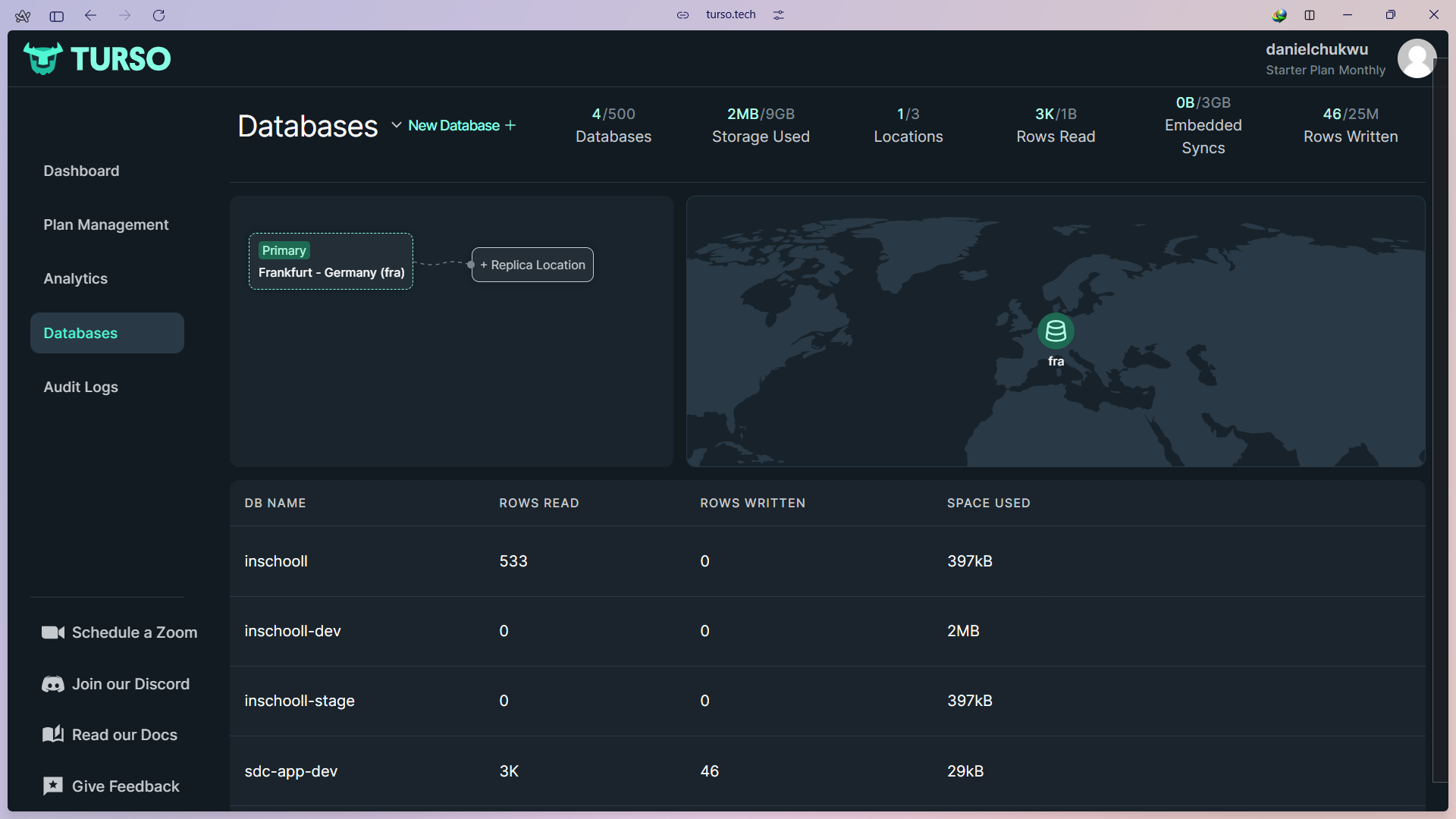


**4.1.4 Database Implementation**

The database used for the Student disciplinary committee application (SDC) is Turso, which provides an SQLite for Production service, it’s also a serverless solution that offers scalability and reliability for data storage. Turso SQLite for Production provides a CLI tool that enables developers to develop locally and easily switch to the hosted database in production.

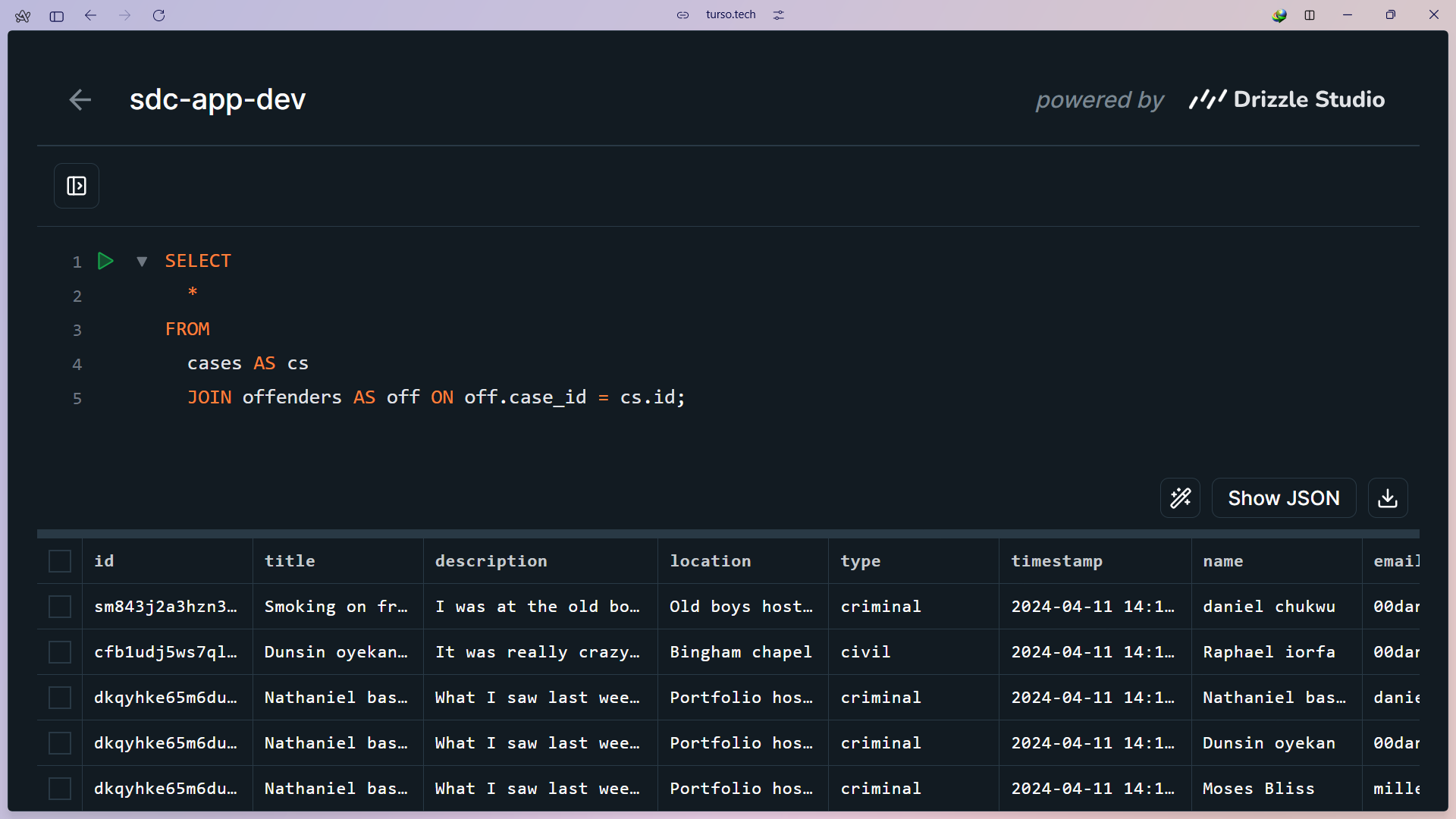
**Figure 8**

*SDC application turso database*



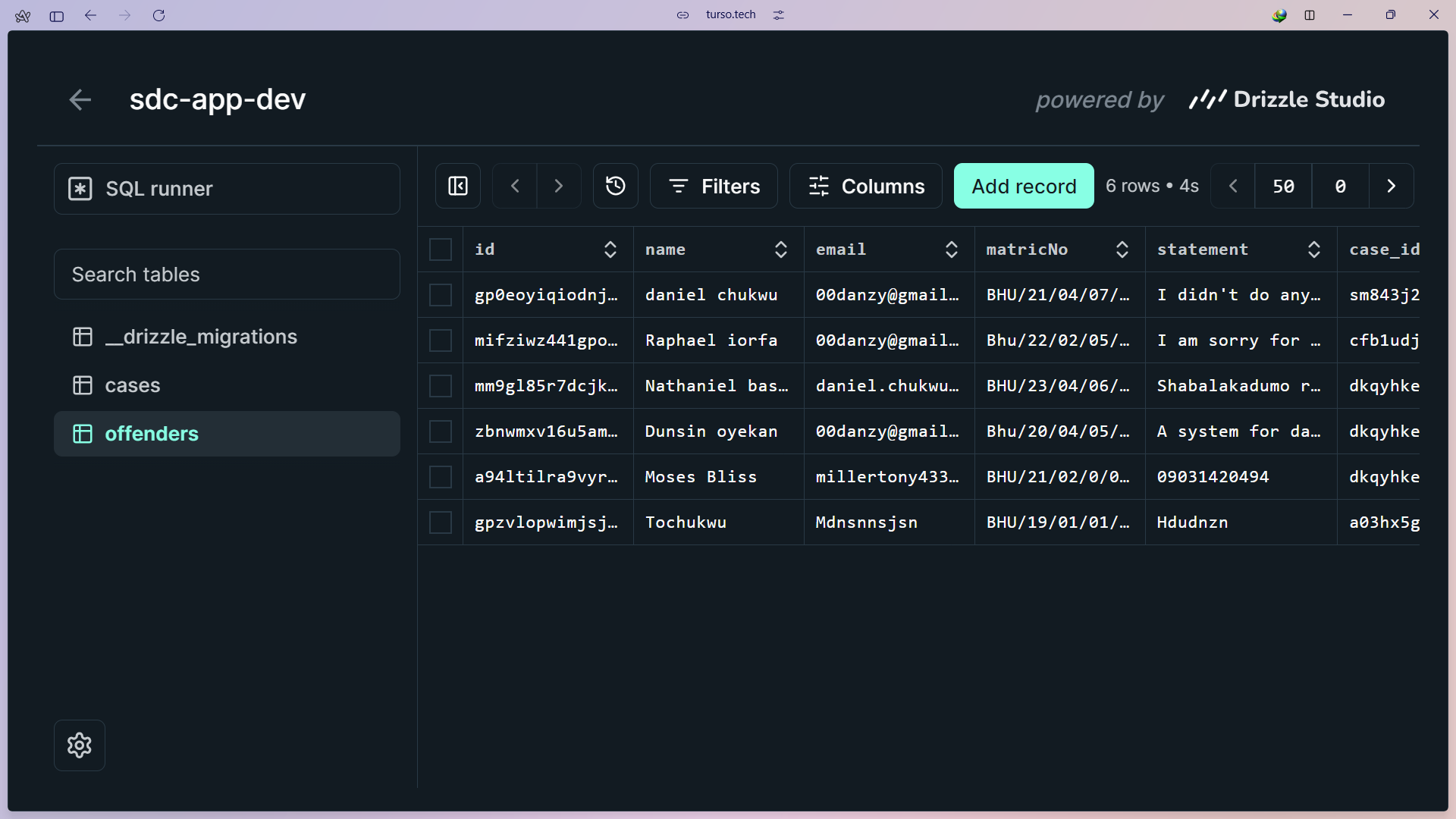
**Figure 9**

*Raw SQL JOIN query*



**Figure 10**

*Offenders’ database table*

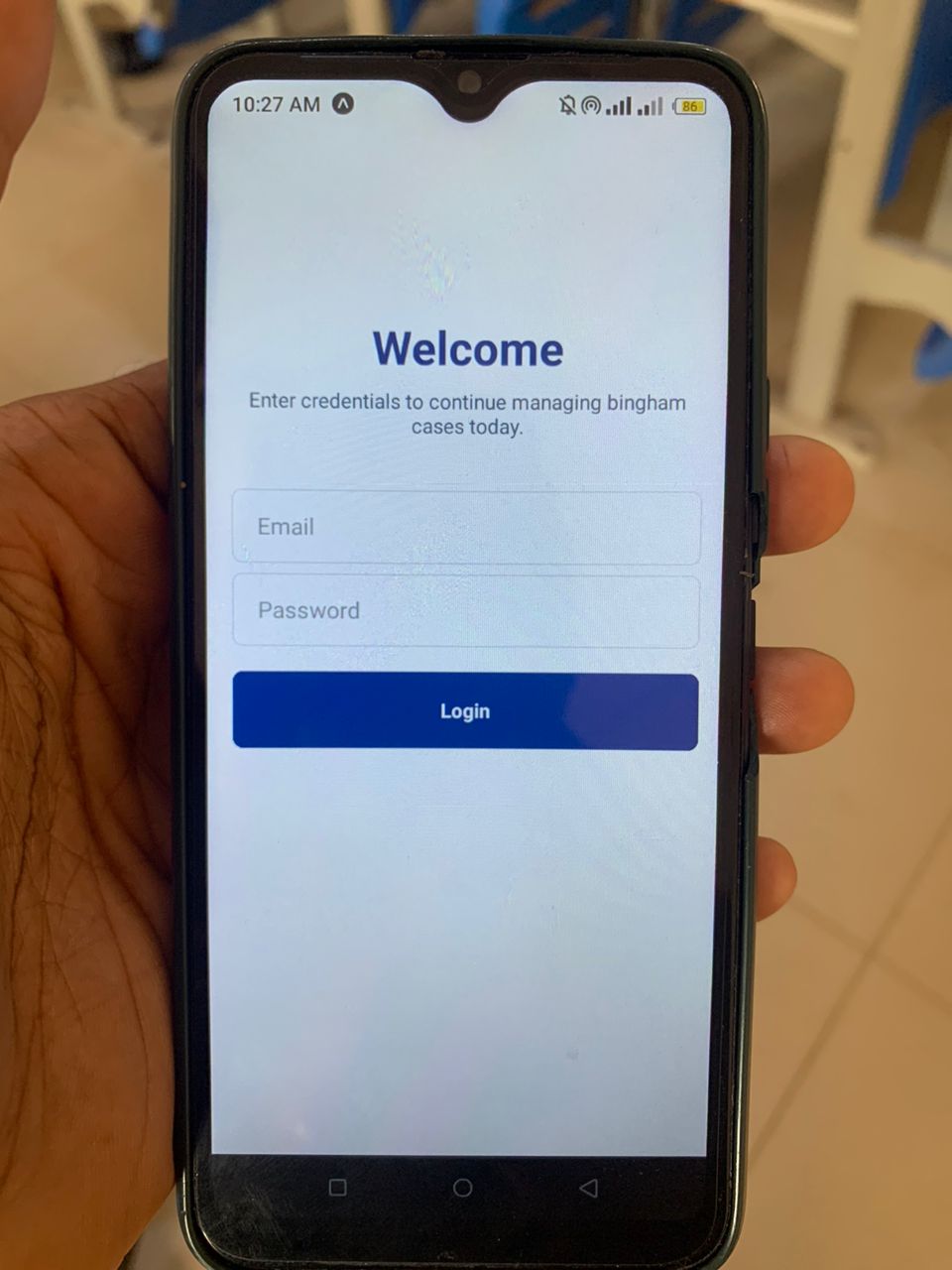
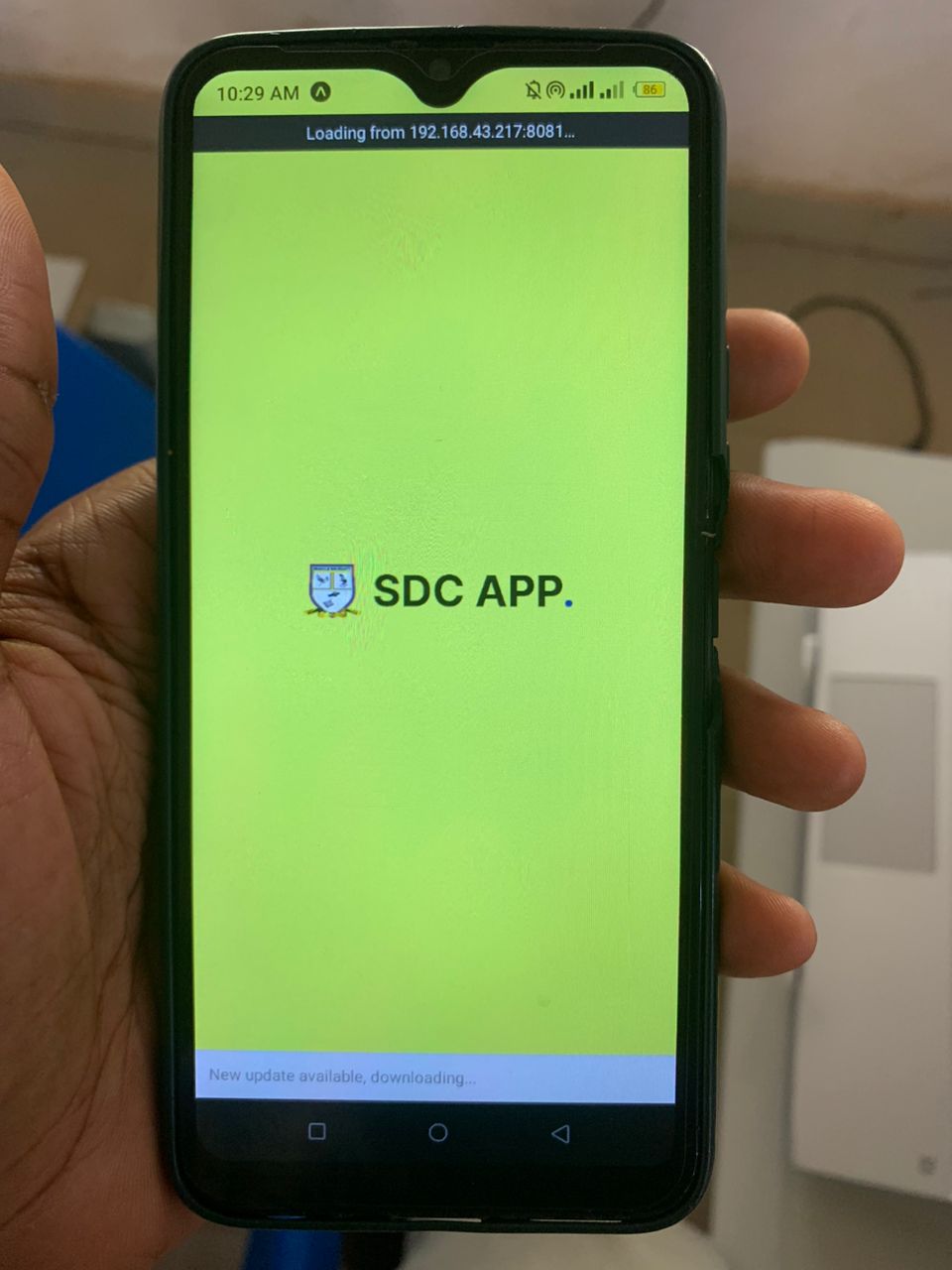


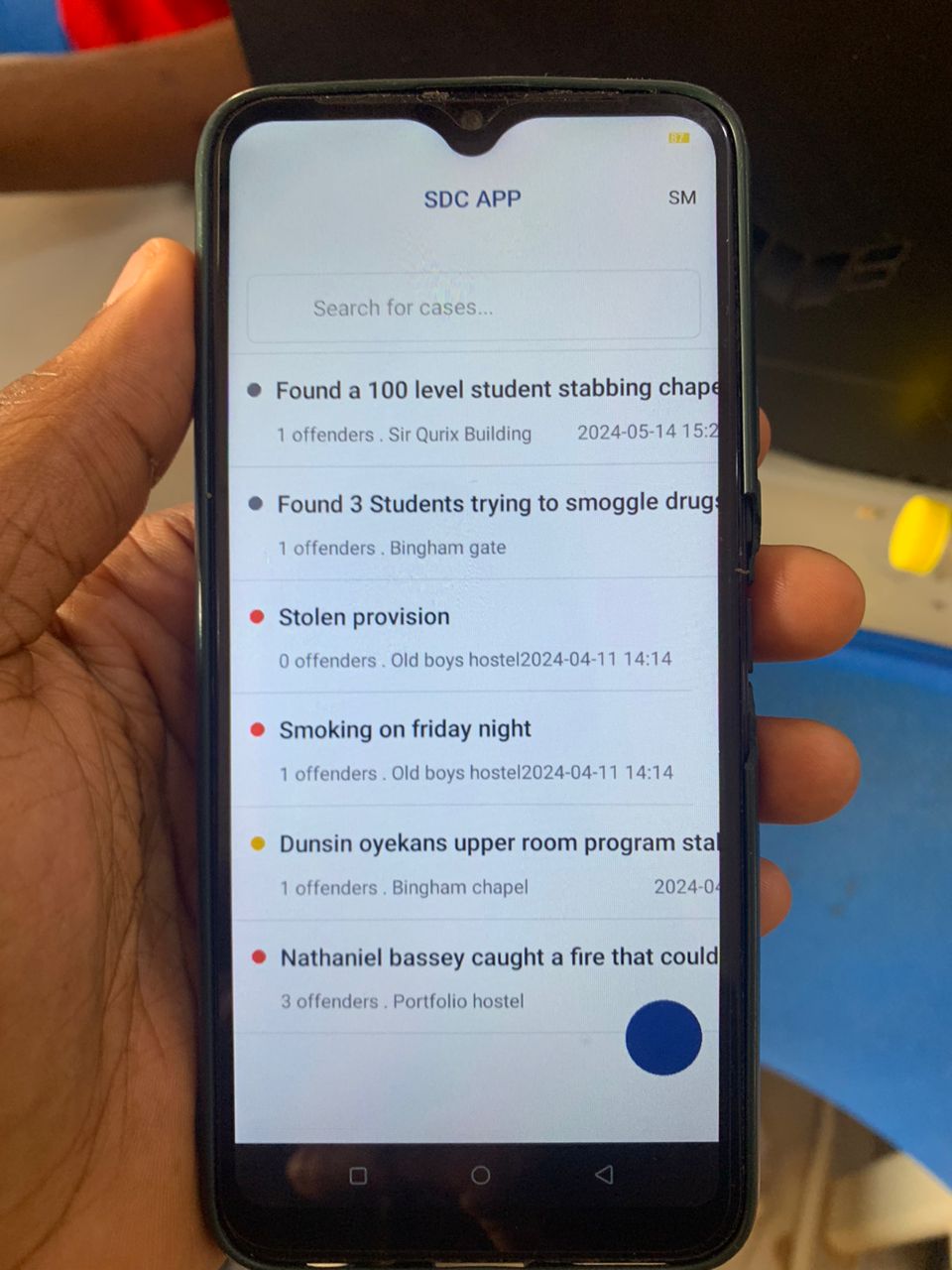
**4.1.5 Final product**

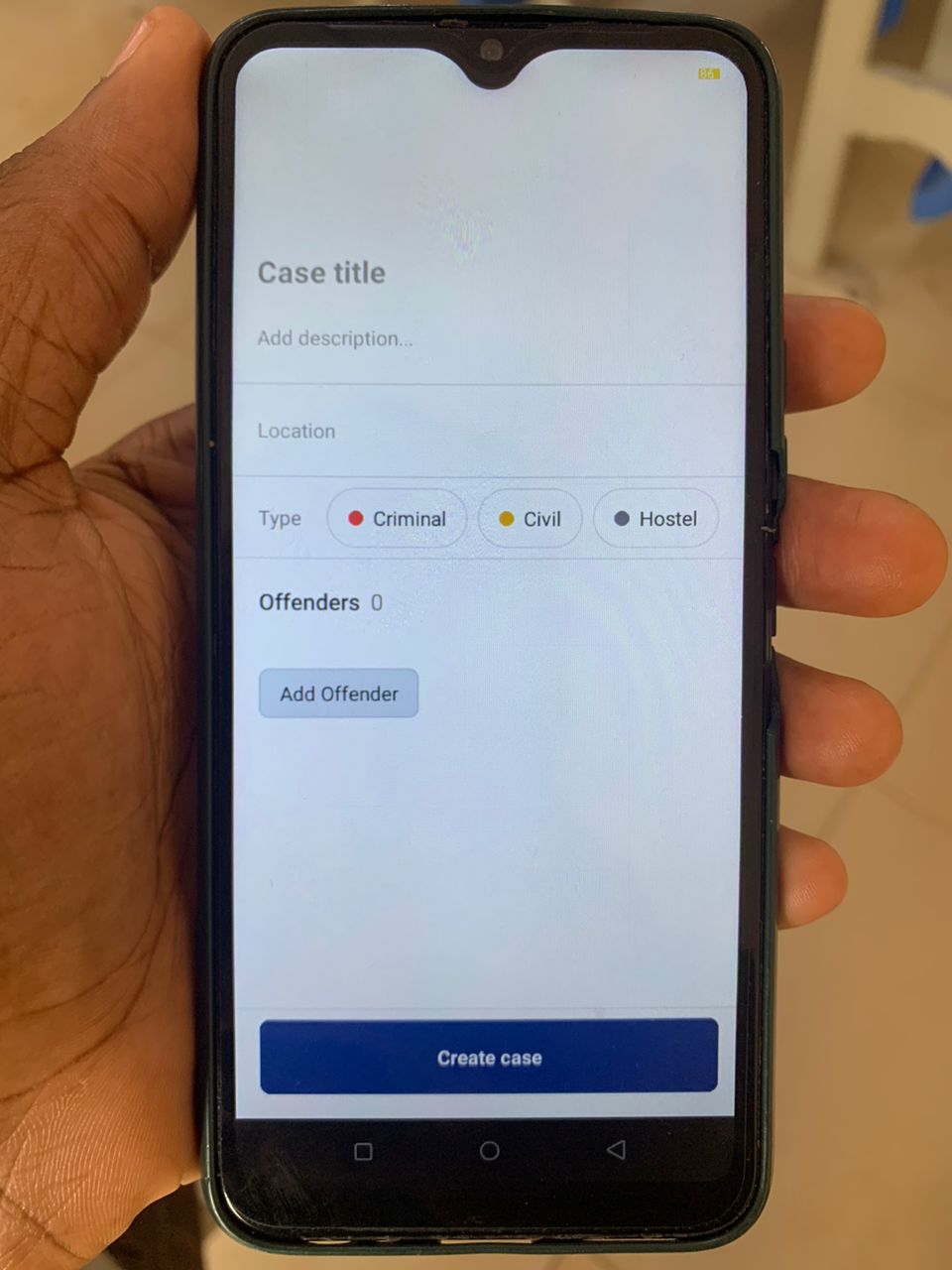
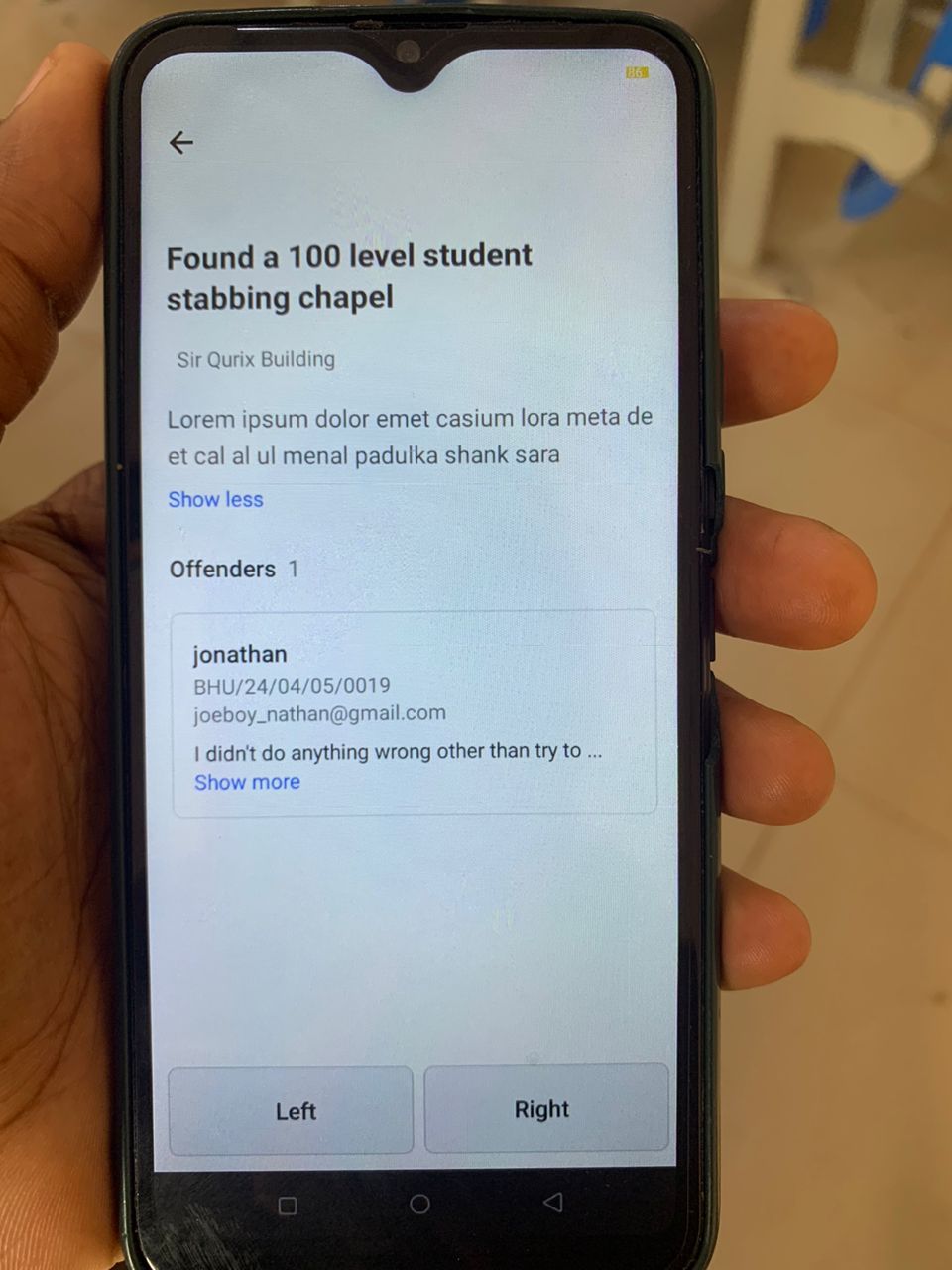
The below figures show the result of the development process so far.

**Figure 9**

*Mobile app screens*







**4.2 Testing**

The testing of the SDC Application was carried out using the following libraries:

1. **jest:** Jest is an amazing, robust, industry-standard JavaScript Testing Framework used by some of the biggest tech companies in the world. It works with projects using: Babel, TypeScript, Node, React, and more. Hence making it a no brainer for the development of the SDC Application.
2. **react-test-renderer:** This package provides an experimental React renderer that can be used to render React components into pure JavaScript objects, without depending on the DOM or a native mobile environment in our case. Which is extremely important.

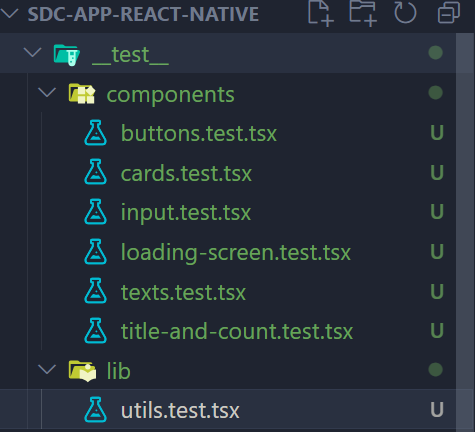
By using Jest and react-test-renderer, the testing process for the SDC Application is streamlined, making it possible to write more comprehensive tests for the app's components and functionalities.

**4.2.1 Unit Testing**

Jest and react-test-renderer libraries were used to unit test all the application components and utility functions. The screenshots below show some of these tests.

**Figure 11**

*Test files*



**Figure 12**

*Unit test code for a card component*

**

**Figure 13**

*Executed test results*

