# CalPAL Design Document

**Students**: Christopher Noblett (C22454222), Aimee McGrane (C22393606)

**1. Overview**
CalPal is a scheduling and productivity application designed to streamline daily activity management through features like event scheduling, location tracking, notifications, and notes. The app ensures users can efficiently organize tasks and stay prepared for their day.

✅ **3 Activities with Data Sharing (Intents)**

- At least three activities must share data using intents home page, profile page, notes page and settings page. Sign up page as well.

✅ **Use of Databases or Files**

- Store and manage app data within the mobile device using databases (specifically Room KSP). Entities, DAOs, database, repository and main view model all used to interact with fragments, activities and classes.

✅ **Use of Sensors**

- Implement one or more sensors (GPS).

✅ **Gestures and Activity Callbacks**

- Incorporate gestures and use activity callbacks effectively.

✅ **Custom Views**

- Utilize custom views for each individual user logged in.

✅ **Continuous GitHub Repository Use**

- Ensure both team members have regular commits throughout the project. (175 progressive commits)

✅ **Dark Mode and Accessibility Features**

- Include support for dark mode and accessibility features. High contrasting colours and big and easy to read font sizes(Patrick Hand).

✅ **State Management**

- Manage app states effectively for logged in and out users. Events, notes and the home page all interact and operate in different states. Notifications also has state checks.

✅ **Notifications, Security, and Performance Optimizations**

- Include notifications, security measures, and performance optimizations (optional but encouraged). Live notifications and kotlin coroutines used for improved performance and reducing workload of the main activity.

## 2. Design Philosophy

- **User-Centric Layout**: Prioritized ease of use and intuitive interaction.
- **Visual Hierarchy**: Organized content for quick comprehension.
- **Accessibility**: Designed with large touch targets, high contrast, and dynamic feedback.
- **Consistency**: Maintained uniform layouts and navigation patterns.

## 3. UI and Navigation Design

### 3.1 Navigation Choices

- **ViewPager2 & BottomNavigationView**: Enables intuitive swipe and tap navigation between core sections (Home, Profile, Notes, Settings).
- **Dynamic Updates**: Ensures that actions like login or event selection dynamically refresh the navigation state.

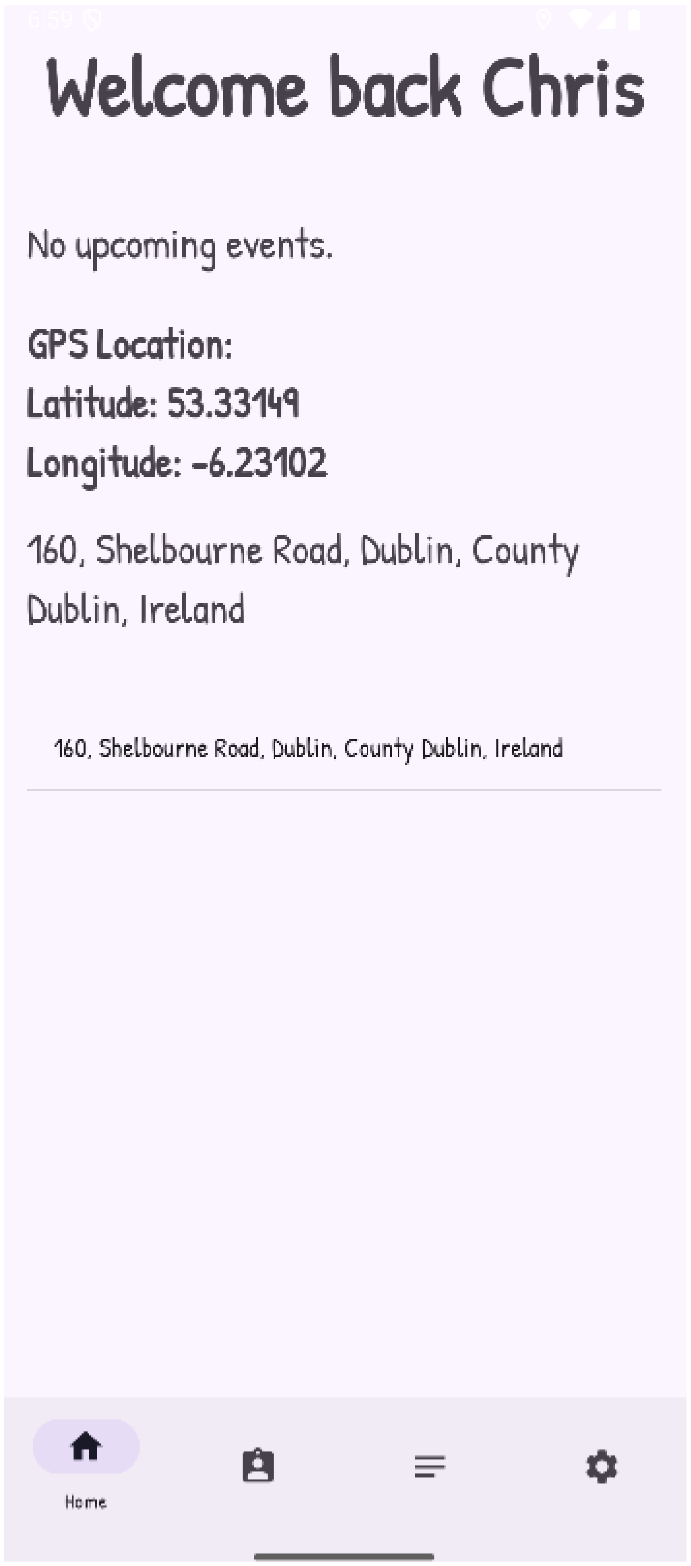
### 3.2 Layout and Interaction Decisions

- **Home Page**:
  - Personalized greetings and next event overview placed prominently to offer immediate value.
  - Event cards provide detailed but concise information.
- **Profile Page**:
  - Calendar integration ensures quick date-based event access.
- **Login Page**:
  - Simple layout focuses on credential input and clear error messages for user feedback.

## 4. Features and Functionalities

### 4.1 Home Page Features

- **Personalized Greeting**: Fetches and displays user details dynamically.
- **Next Event Notification**: Prioritizes important information with real-time updates.
- **Location Tracking**: Converts GPS coordinates to human-readable addresses for convenience.



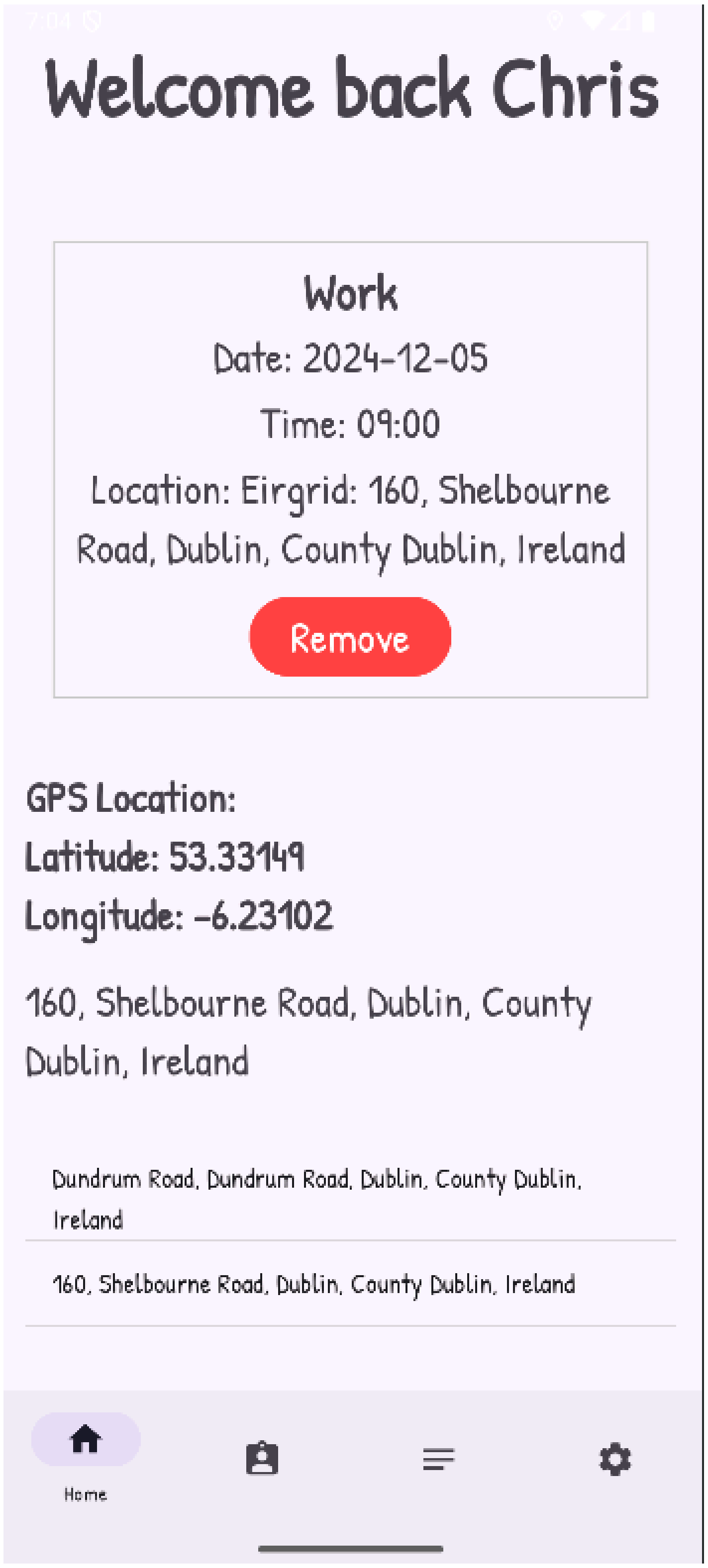
Welcome back Chris

No upcoming events.

GPS Location:
Latitude: 53.33149
Longitude: -6.23102

160, Shelbourne Road, Dublin, County Dublin, Ireland

160, Shelbourne Road, Dublin, County Dublin, Ireland



Welcome back Chris

Work
Date: 2024-12-05
Time: 09:00
Location: Eirgrid: 160, Shelbourne Road, Dublin, County Dublin, Ireland

GPS Location:
Latitude: 53.33149
Longitude: -6.23102

160, Shelbourne Road, Dublin, County Dublin, Ireland

Dundrum Road, Dundrum Road, Dublin, County Dublin, Ireland

160, Shelbourne Road, Dublin, County Dublin, Ireland

**4.2 Profile Page Features**

- **Calendar View**: Fetches events for selected dates dynamically.
- **Event Management**: Users can view, add, and delete events seamlessly.
- **Notifications**: Prompts users about events within 24 hours.

## 4.3 Login Page Features

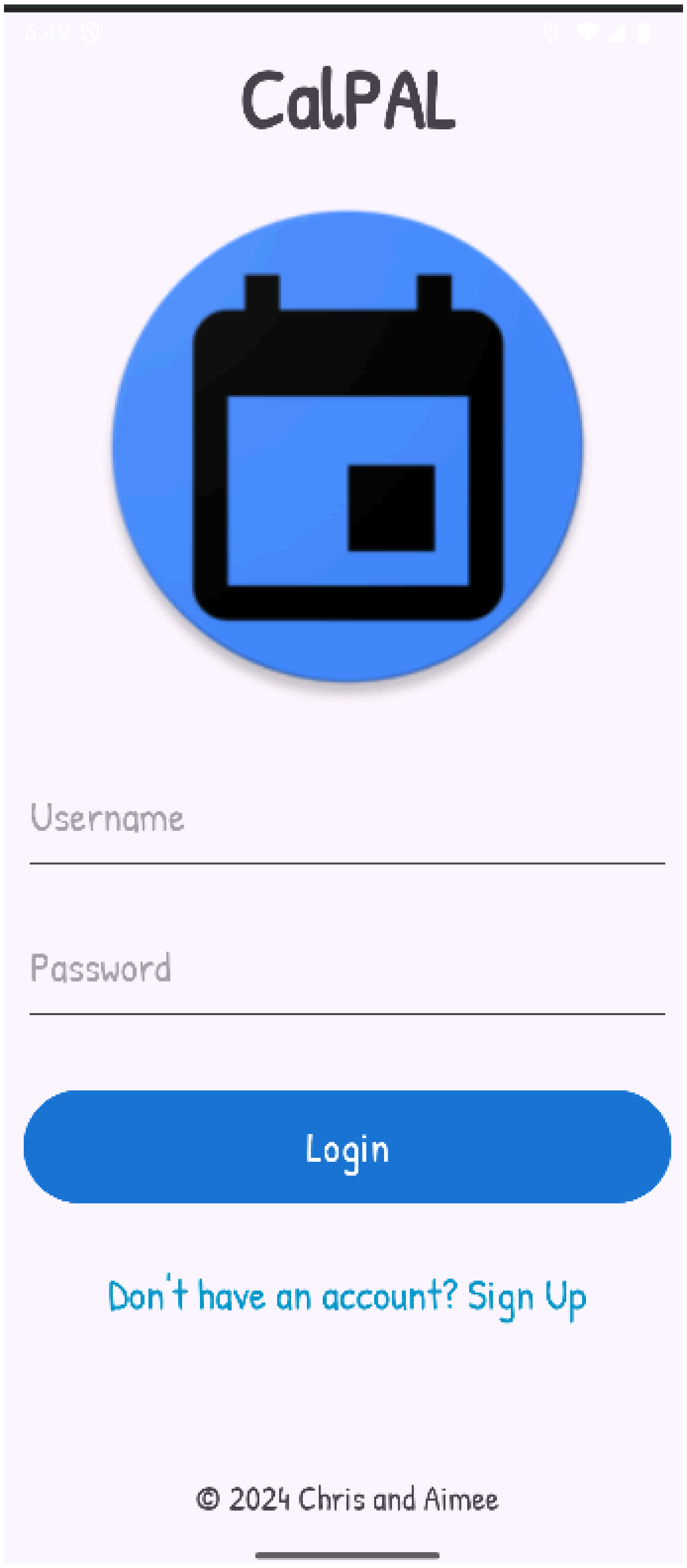
- **Credential Validation**: Ensures secure login with hashed password comparison.
- **Session Management**: Handles logged-in state persistently across app sessions.

## 5. High-Level Method Overview

- **onLocationChanged()**
  - **Purpose**: Updates user location in real-time.
  - **Input**: Location data.
  - **Output**: Updated UI.
  - **Description**: Converts GPS coordinates to a human-readable address using Geocoder.
- **fetchEventsForDate()**
  - **Purpose**: Retrieves events for a specific date.
  - **Input**: Selected date.
  - **Output**: Event list.
  - **Description**: Leverages the ViewModel to fetch data dynamically from the backend.
- **handleLogin()**
  - **Purpose**: Validates user credentials and manages login.
  - **Input**: Username, Password.
  - **Output**: Login status.
  - **Description**: Manages session state securely, including credential validation via hashed password comparison.

## 6. Testing During Development

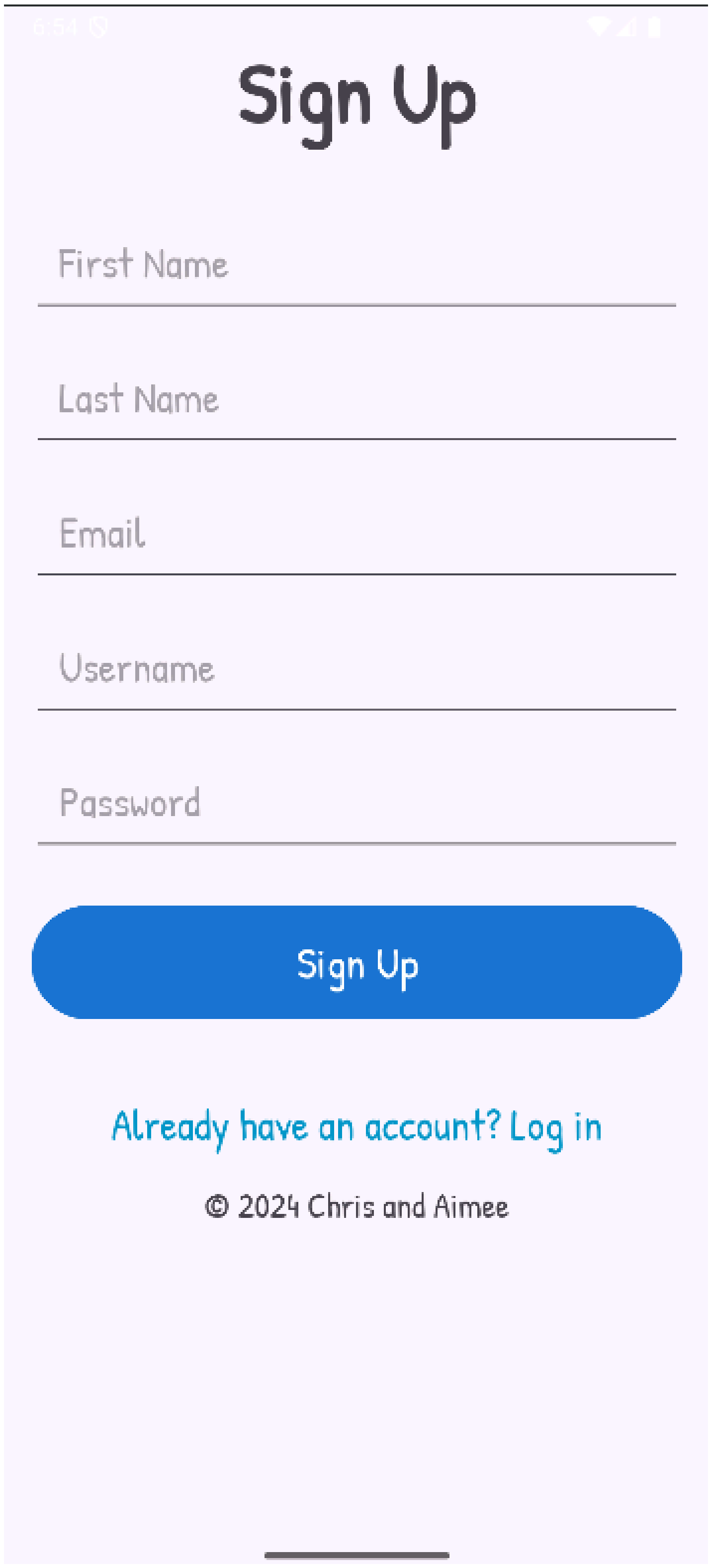
- **Unit Testing**: Validated individual components.
- **Integration Testing**: Checked module interactions.
- **Performance Testing**: Evaluated under peak loads.
- **Functionality Testing:** Extensive error checking and bug finding done.
- **Security Testing**: Protected user data and prevented breaches.



**Sign-Up Page**
The SignUpActivity provides an intuitive interface for account creation, validating user details before saving them to the database. Users can navigate back to the login screen if needed.

**Key Features:**

1. **UI Elements:**
    - Input Fields: EditText fields for first name, last name, email, username, and password.
    - Sign-Up Button: Initiates the sign-up process.
    - Login Link: Navigates back to the login screen.
2. **MainViewModel Integration:**
    - Handles sign-up logic, including checking for duplicate data and saving new user records.
3. **Sign-Up Process:**
    - Data Validation:
        - Empty fields trigger a toast prompting users to fill all details.
        - Duplicate checks for username, email, and password using MainViewModel.
        - Displays appropriate messages if data is already in use (e.g., "Username is already taken").
    - **Successful Sign-Up:**
        - Creates a new user record with insertUserSignUps().
        - Shows "Sign-up successful!" toast and closes the activity to return to the login screen.
4. **Navigation to Login:**
    - The Login Link returns users to the LoginActivity via an Intent, closing the current activity


Sign Up

First Name

Last Name

Email

Username

Password

Sign Up

Already have an account? Log in

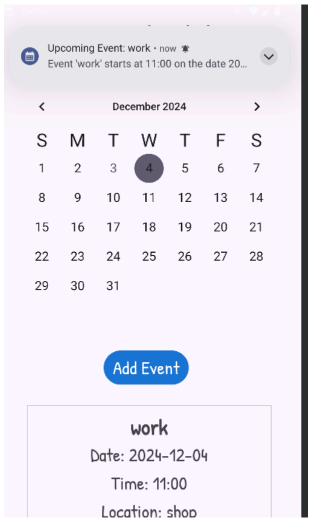
© 2024 Chris and Aimee

**Gestures and Activity Callbacks:**
The MainActivity uses ViewPager2 and BottomNavigationView to provide seamless navigation between four pages: Home, Profile, Notes, and Settings, allowing users to swipe or tap to switch between pages effortlessly. Callbacks are included in dialogs and navigating between login and signup pages if users change their mind.

Key Features:

1. **ViewPager2 Setup**
   - Configured with a FragmentStateAdapter to manage the fragments.
   - Supports horizontal swiping between fragments.
2. **Fragments**
   - Four fragments: HomeFragment, ProfileFragment, NotesFragment, SettingsFragment, displayed in sequence.
3. **Swipe Gestures**
   - Swiping left or right navigates between fragments.
   - The onPageSelected callback tracks the current fragment and updates the navigation view.
4. **Synchronization with BottomNavigationView**
   - Swipe Actions: Swiping updates the selected item in BottomNavigationView.
   - Taps: Tapping a navigation item triggers viewPager.setCurrentItem() to display the corresponding fragment.

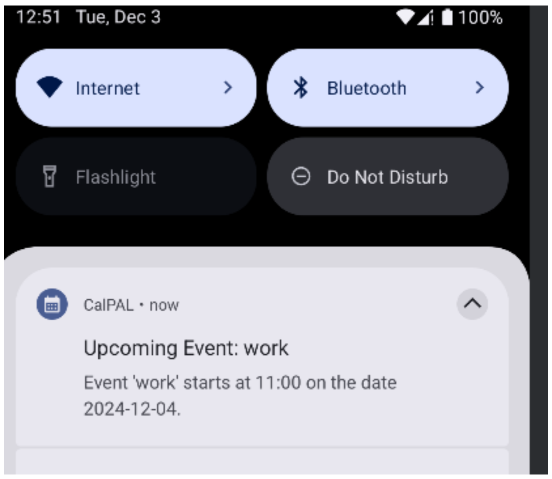
**Notifications:**
The notification system in our app is designed using Android's NotificationManager and WorkManager frameworks to deliver timely reminders about upcoming events. A notification channel, "Event Notifications," is created programmatically for devices running Android Oreo (API 26) or later using the NotificationChannel class. This channel is configured with IMPORTANCE_HIGH to ensure prominent visibility and an alerting behavior for notifications. Periodic notifications are scheduled using PeriodicWorkRequestBuilder, which configures a worker to run every 15 minutes via the WorkManager API. The worker fetches event data, leveraging the app's ViewModel to access stored user event information, and then uses the NotificationCompat builder to create and display notifications.


Upcoming Event: work • now
Event 'work' starts at 11:00 on the date 20...

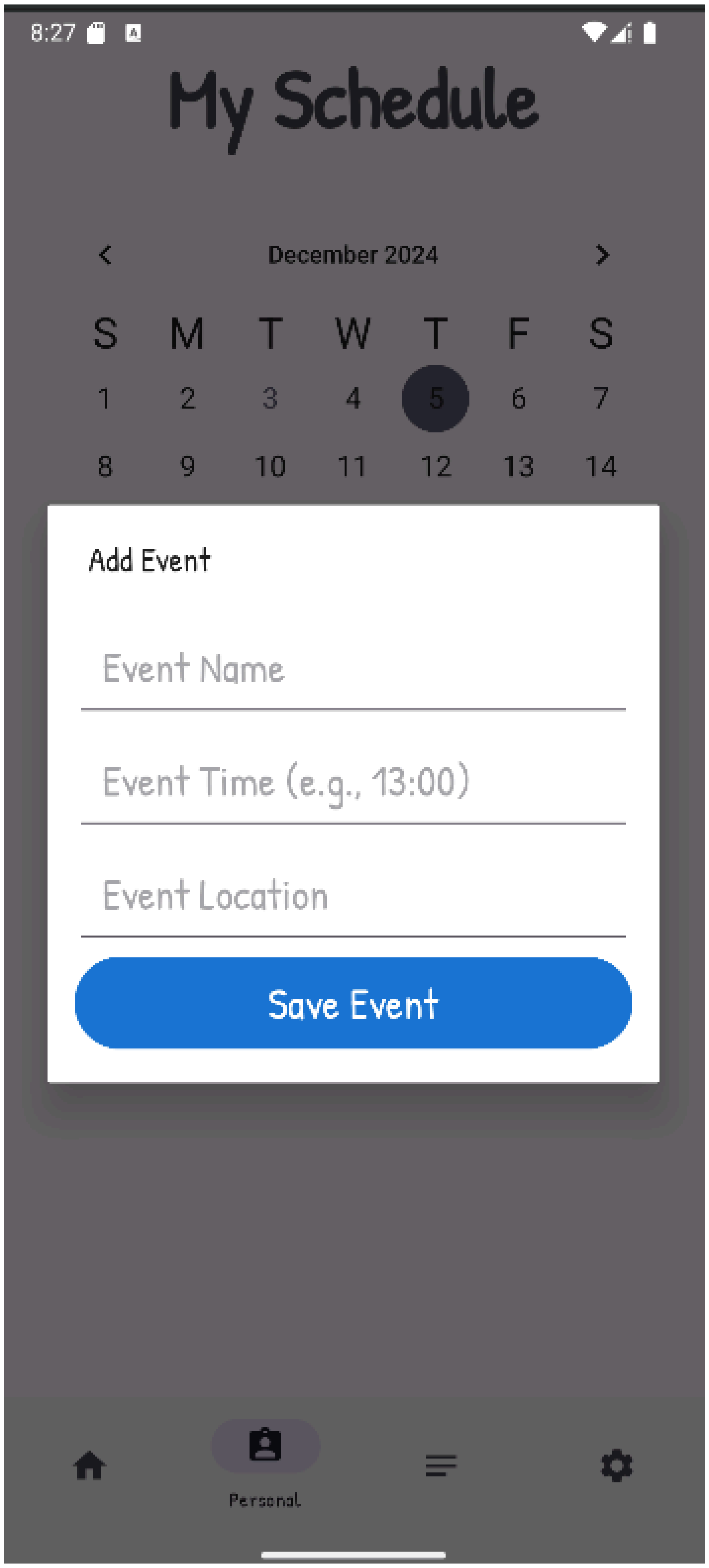
December 2024

| S | M | T | W | T | F | S |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | | | | |

work
Date: 2024-12-04
Time: 11:00
Location: shop



CalPAL • now
Upcoming Event: work
Event 'work' starts at 11:00 on the date 2024-12-04.


8:27
My Schedule
December 2024
S M T W T F S
1 2 3 4 5 6 7
8 9 10 11 12 13 14
Add Event
Event Name
Event Time (e.g., 13:00)
Event Location
Save Event
Personal

**Settings and Notes Page**

**1.1 Design Principles**

- **User-Centric Layout**: Designed with the user in mind, prioritizing intuitive navigation and ease of interaction.
- **Visual Hierarchy**: Organized layout with a clear distinction between primary and secondary information to guide user attention effectively.
- **Accessibility**: Incorporates large text sizes, high contrast, and intuitive touch targets to ensure usability across diverse user groups.
- **Consistency**: Maintains a uniform design language across fragments for a seamless user experience.

**1.2 Rationale for Layout Choices**

- **Vertical Linear Layout**: Ensures a logical flow, allowing users to process settings sequentially from top to bottom.
- **Widget Placement**: Frequently used actions (e.g., Dark Mode toggle) are positioned for quick access, while irreversible actions (e.g., Delete Account) are distinctly marked and placed to avoid accidental interaction.
- **Text and Typography**:
    - Large headers (50sp for titles) establish context immediately.
    - Smaller but bold section headers (25sp) guide users through specific sections.
- **Color Scheme**:
    - Red text for critical actions (e.g., Logout, Delete Account) emphasizes caution.
    - Neutral tones for toggle switches and general settings maintain a professional look.

**2. Navigation Choices**

- **Fragment Navigation**:
    - The **Settings Fragment** acts as a hub for user preferences, enabling users to toggle themes or manage accounts efficiently.
    - Back navigation seamlessly returns users to the previous activity, preserving the app's flow.
- **Contextual Visibility**: Buttons like "Add Note" in the **Notes Fragment** only appear when relevant (e.g., after an event selection), reducing visual clutter.

**3. Wireframe and Layout Analysis**

- **Settings Fragment Wireframe**:
    - Title centered at the top, followed by Dark Mode toggle in a horizontal layout.
    - Logout and Delete Account buttons at the bottom with adequate spacing.
- **Notes Fragment Wireframe**:
    - Title at the top ("My Notes").
    - Scrollable event list below the title.
    - Notes dynamically appear after an event is selected, with the "Add Note" button contextually placed.

## 4. High-Level Method Descriptions

### 4.1 Settings Fragment

**onCreateView()**

- **Purpose**: Initializes the UI components and inflates the layout.
- **Data Structures**: Accesses the SharedPreferences instance to retrieve the current theme state.

**handleLogout()**

- **Purpose**: Logs out the user and redirects to the login screen.
- **Implementation**:
    - Coroutine-based logout process to ensure non-blocking execution.
    - Clears user session data.

**handleDeleteAccount()**

- **Purpose**: Deletes the user account permanently.
- **Implementation**:
    - Displays a confirmation dialog to prevent accidental deletions.
    - Uses asynchronous database operations to remove user data comprehensively.

### 4.2 Notes Fragment

**loadUserEvents()**

- **Purpose**: Fetches and displays the list of events for the logged-in user.
- **Data Structures**: Uses a list of event objects, each containing attributes like name, date, and location.

**displayEvents()**

- **Purpose**: Dynamically generates event cards based on fetched data.
- **Data Structures**: Iterates over the event list to create individual UI components.

**loadNotesForEvent()**

- **Purpose**: Fetches notes associated with a selected event.
- **Data Structures**: Filters a list of notes by event ID.

**addNoteToEvent()**

- **Purpose**: Adds a new note to the currently selected event.
- **Implementation**: Validates user input before inserting data into the Room database.

## 5. Accessibility and Usability Enhancements

### 5.1 Accessibility Features

- **Large Touch Targets**: Ensures buttons are easy to tap, reducing errors.
- **Descriptive Text**: Each action is labeled clearly to minimize ambiguity.
- **Dynamic Feedback**: UI updates instantly reflect user actions, providing confirmation for changes.

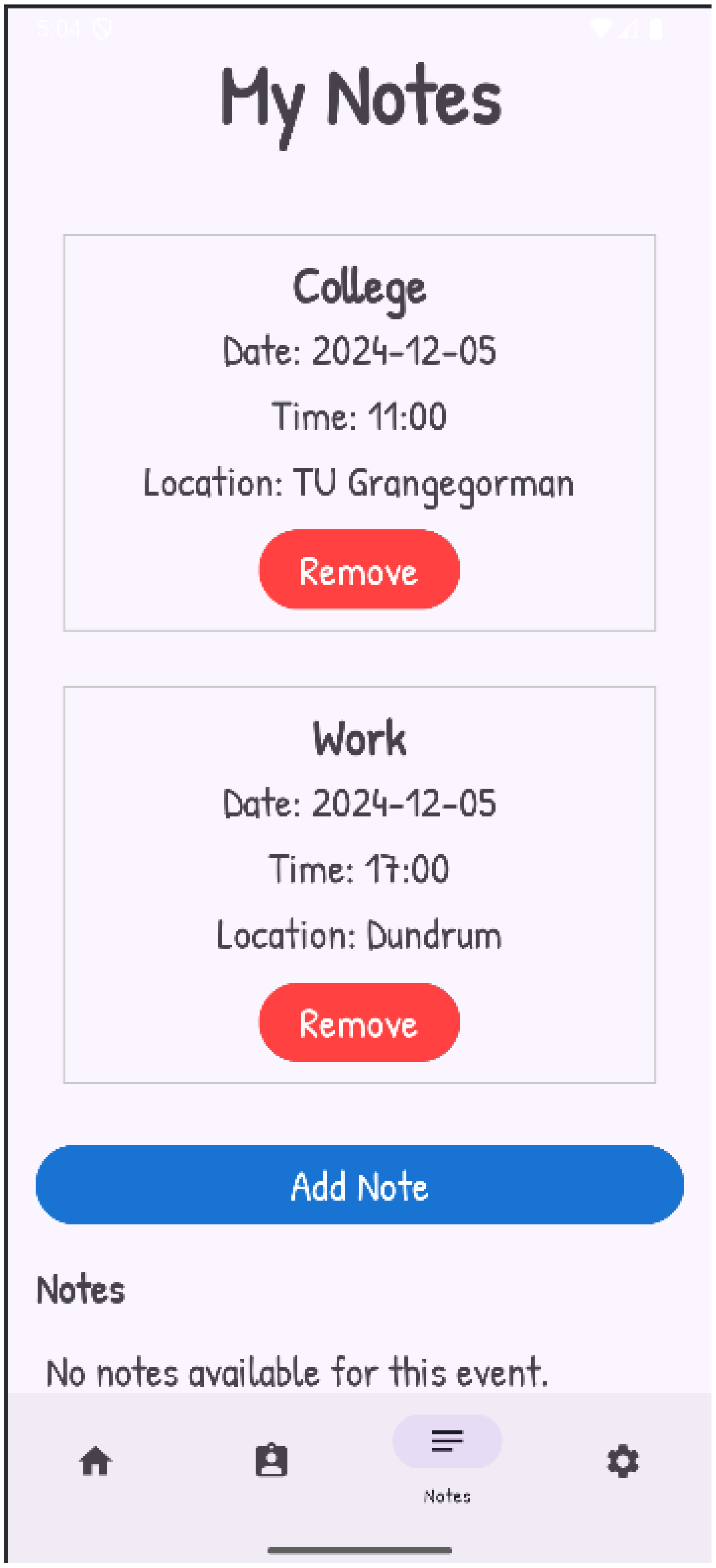
## 6. Performance Considerations

### 6.1 Optimized Data Handling

- Utilizes Kotlin Coroutines for database operations to prevent main thread blocking.
- Implements lazy loading for event and note data to optimize memory usage.

### 6.2 Efficient UI Updates

- ViewModel observers ensure UI components are only updated when necessary, reducing redundant operations.



My Notes

College
Date: 2024-12-05
Time: 11:00
Location: TU Grangegorman
Remove

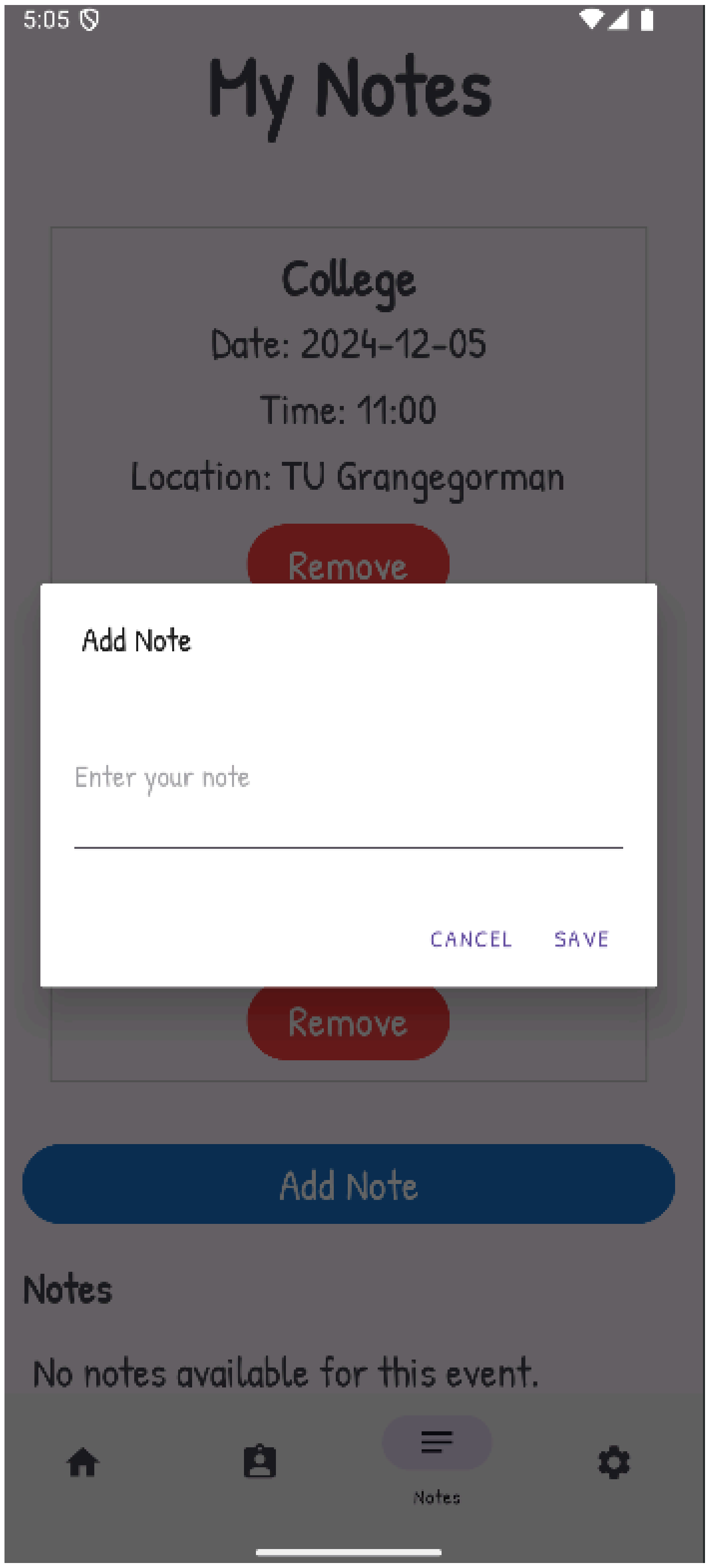
Work
Date: 2024-12-05
Time: 17:00
Location: Dundrum
Remove

Add Note

Notes

No notes available for this event.

Notes



5:05

My Notes

College
Date: 2024-12-05
Time: 11:00
Location: TU Grangegorman
Remove

Add Note

Enter your note

CANCEL SAVE

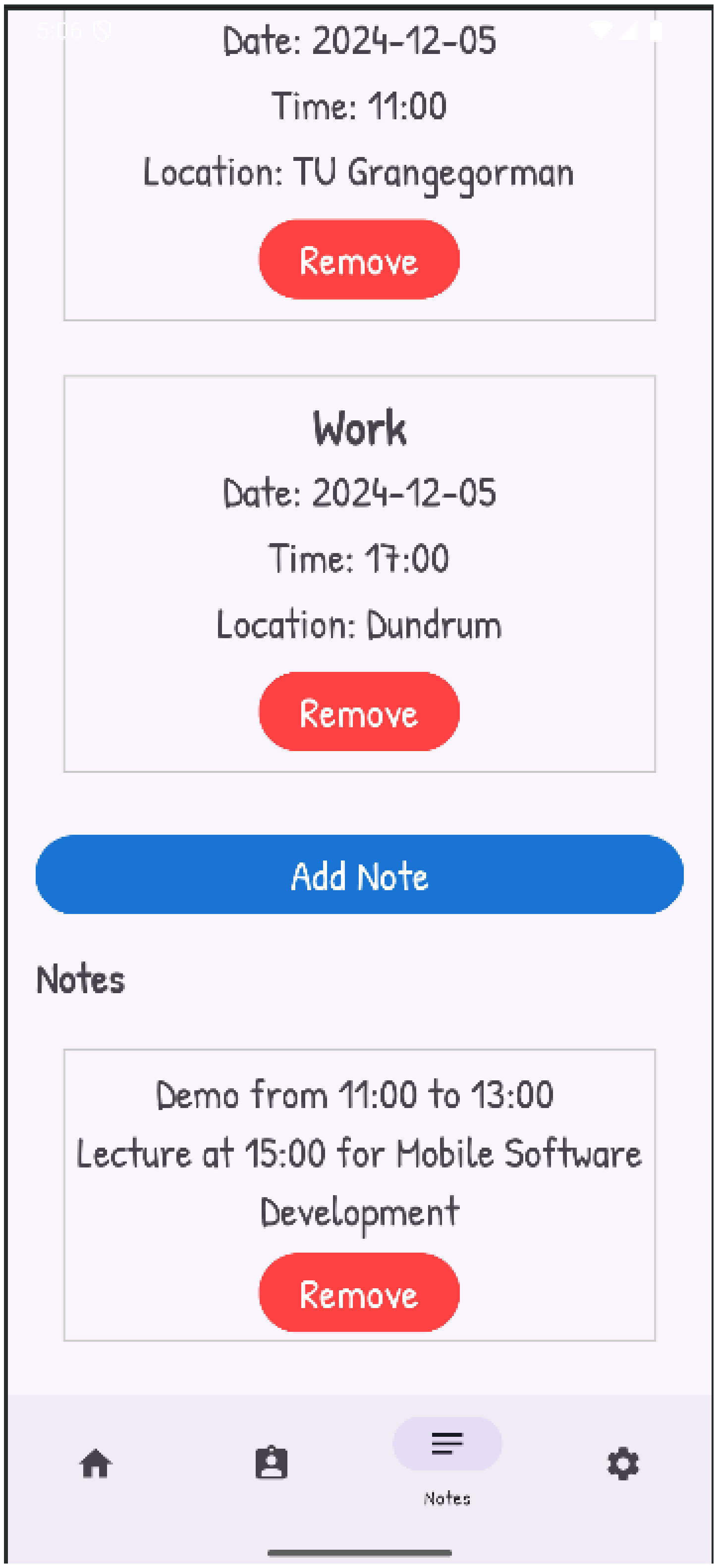
Remove

Add Note

Notes

No notes available for this event.

Notes



Date: 2024-12-05

Time: 11:00

Location: TU Grangegorman

Remove

Work

Date: 2024-12-05

Time: 17:00

Location: Dundrum

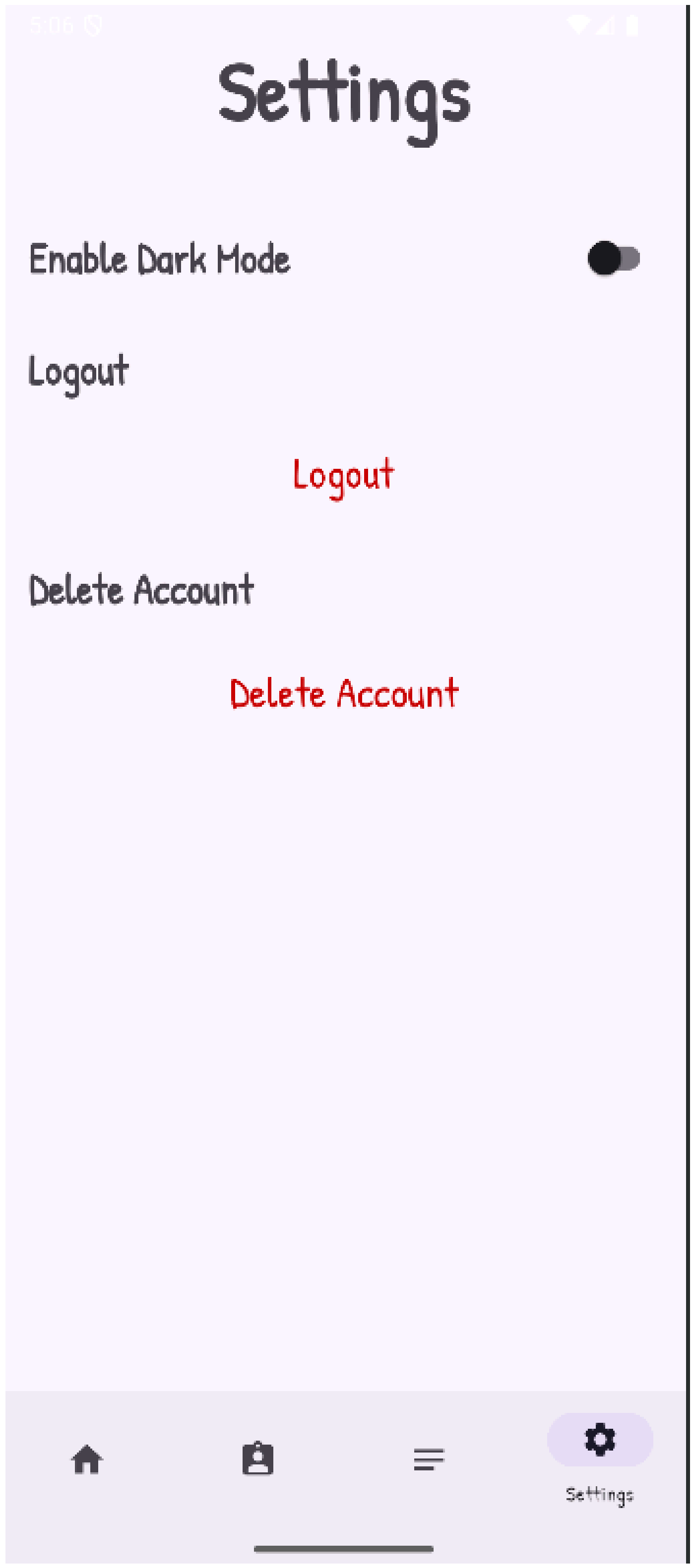
Remove

Add Note

Notes

Demo from 11:00 to 13:00
Lecture at 15:00 for Mobile Software Development

Remove



Settings

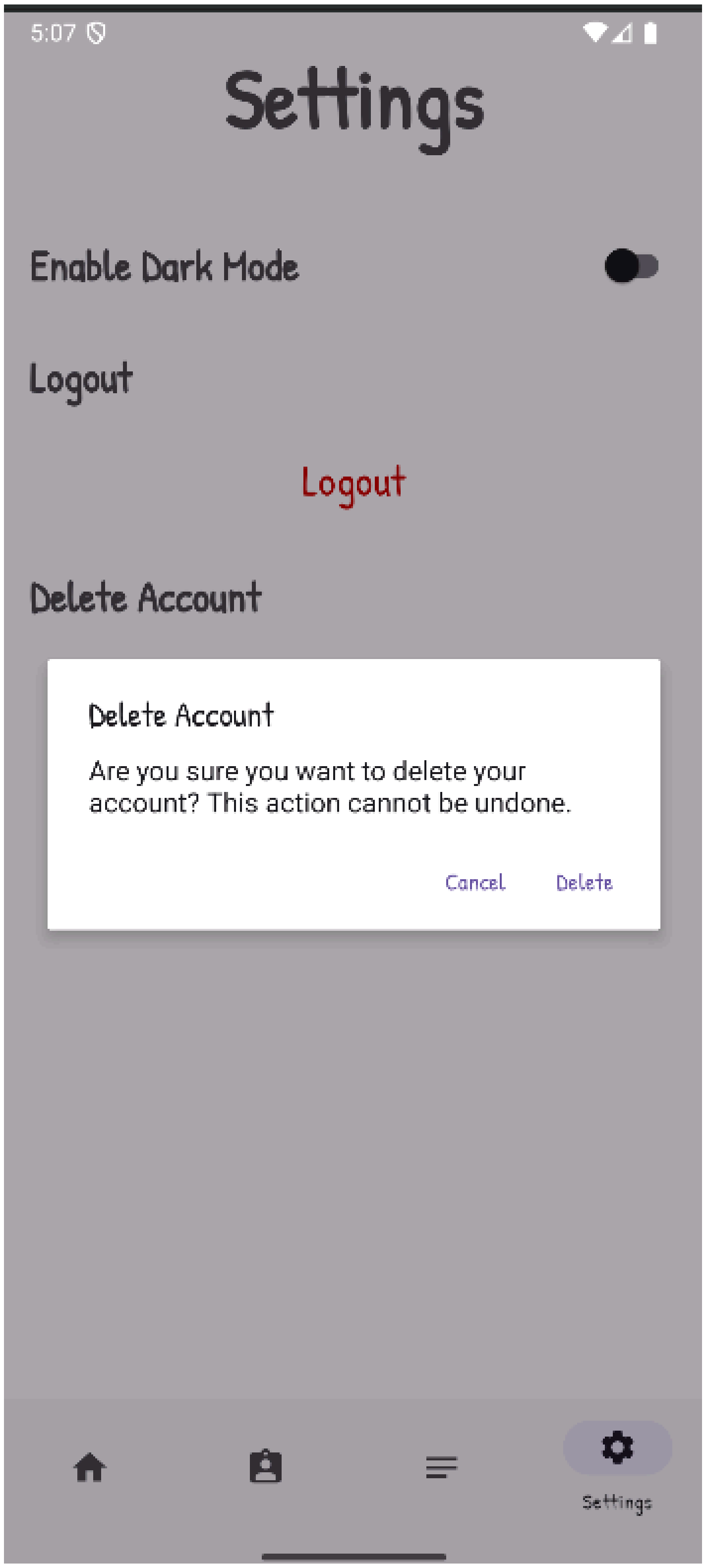
Enable Dark Mode

Logout

Logout

Delete Account

Delete Account



5:07

Settings

Enable Dark Mode

Logout

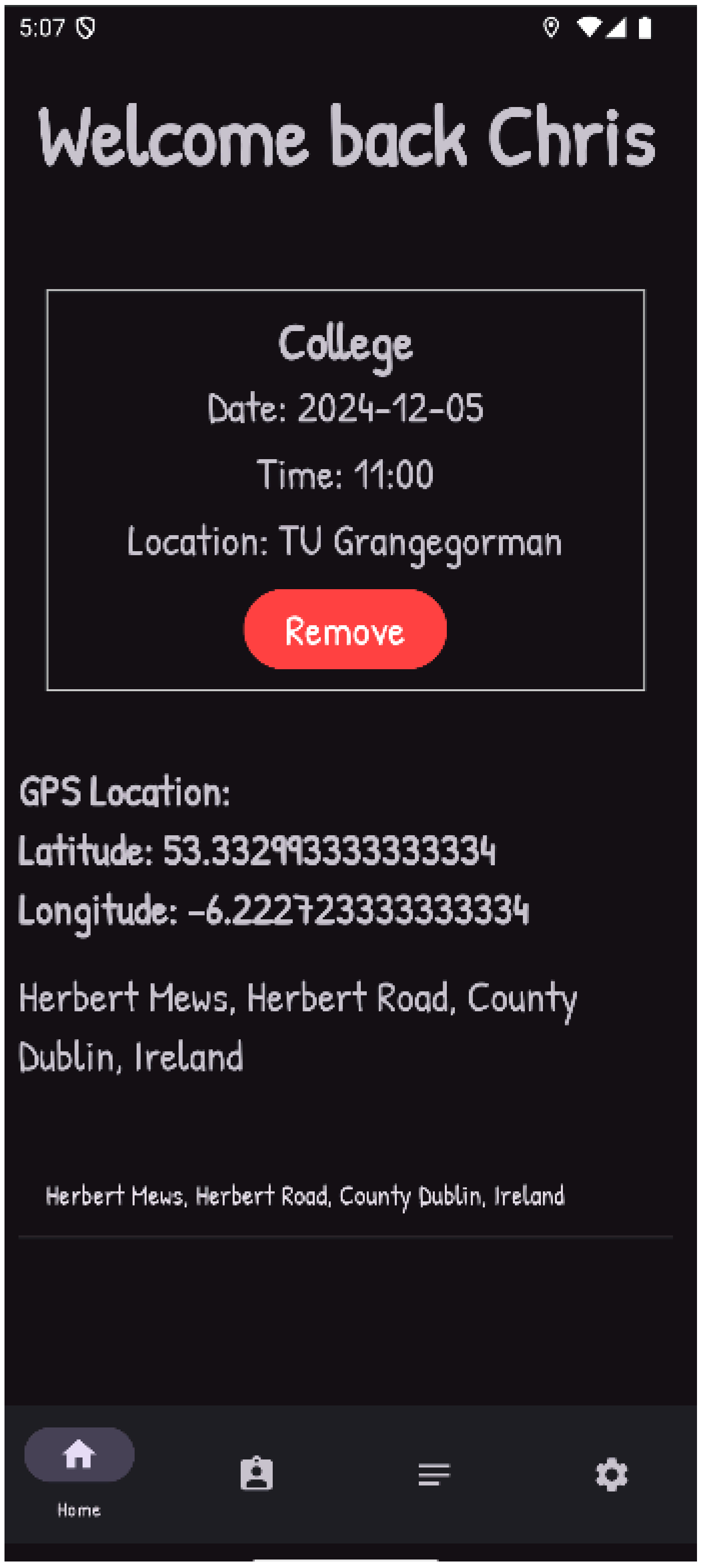
Logout

Delete Account

Delete Account

Are you sure you want to delete your account? This action cannot be undone.

Cancel Delete



5:07

Welcome back Chris

College

Date: 2024-12-05

Time: 11:00

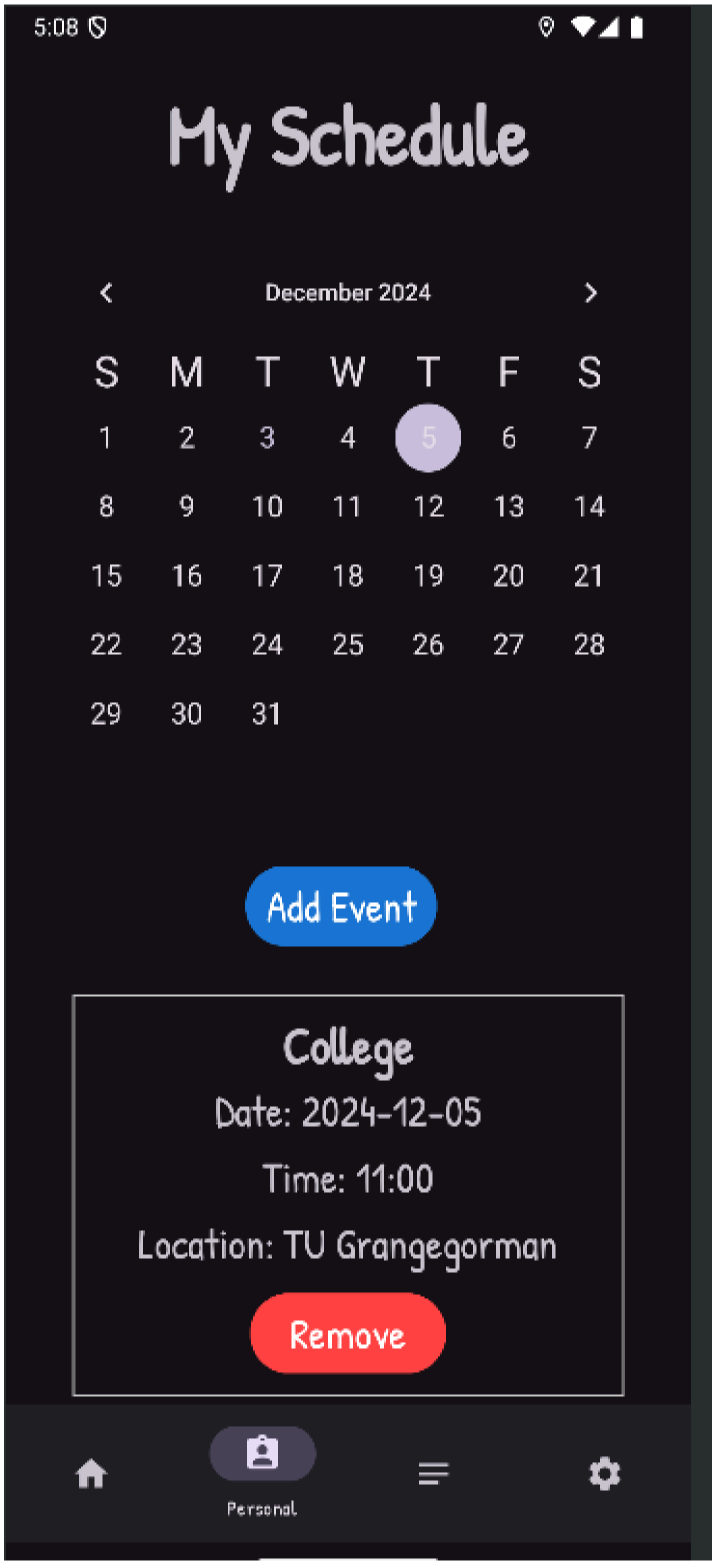
Location: TU Grangegorman

Remove

GPS Location:
Latitude: 53.332993333333334
Longitude: -6.222723333333334

Herbert Mews, Herbert Road, County Dublin, Ireland

Herbert Mews, Herbert Road, County Dublin, Ireland



5:08

My Schedule

December 2024

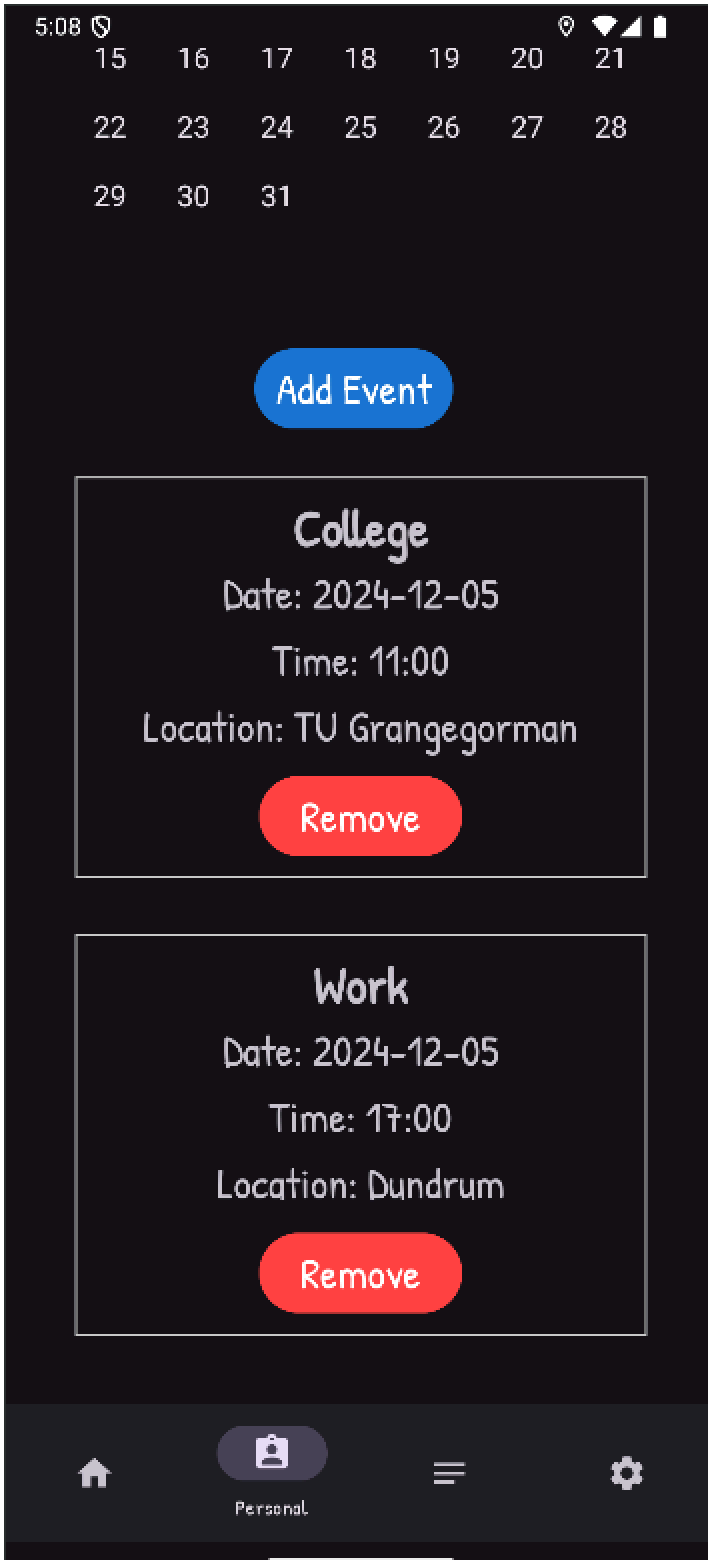
| S | M | T | W | T | F | S |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | | | | |

Add Event

College
Date: 2024-12-05
Time: 11:00
Location: TU Grangegorman

Remove

Personal



5:08

| | | | | | | |
|---|---|---|---|---|---|---|
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | | | | |

Add Event

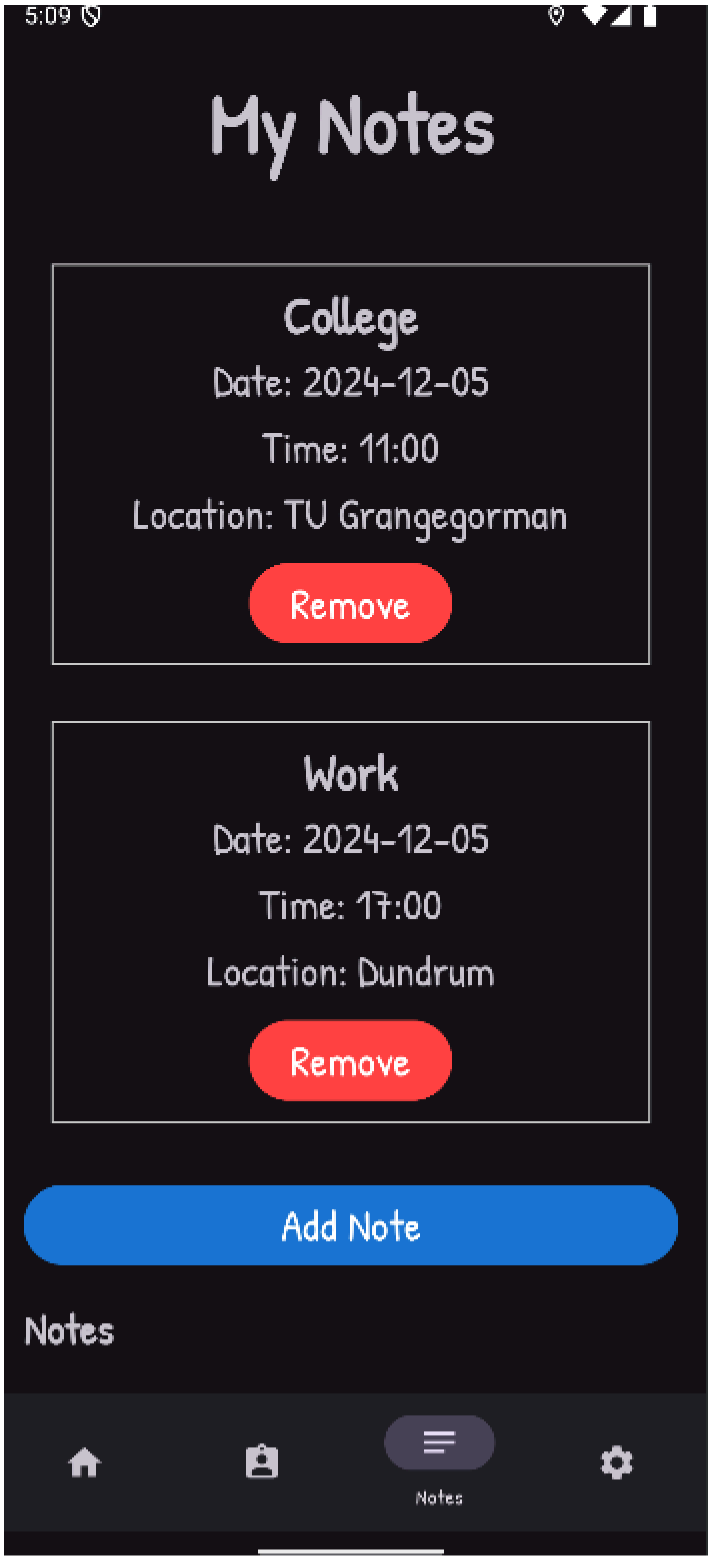
College
Date: 2024-12-05
Time: 11:00
Location: TU Grangegorman

Remove

Work
Date: 2024-12-05
Time: 17:00
Location: Dundrum

Remove

Personal


5:09

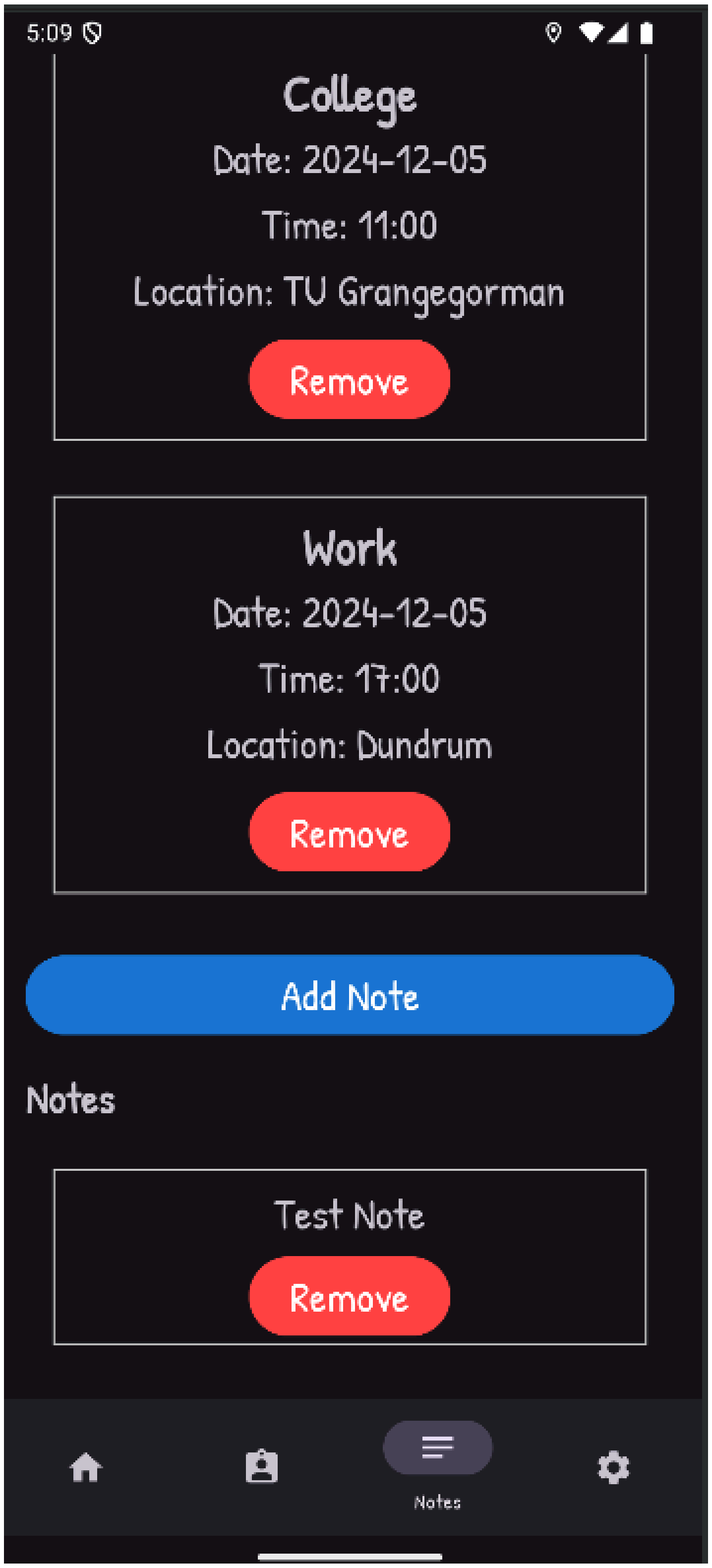
My Notes

College
Date: 2024-12-05
Time: 11:00
Location: TU Grangegorman
Remove

Work
Date: 2024-12-05
Time: 17:00
Location: Dundrum
Remove

Add Note

Notes


5:09

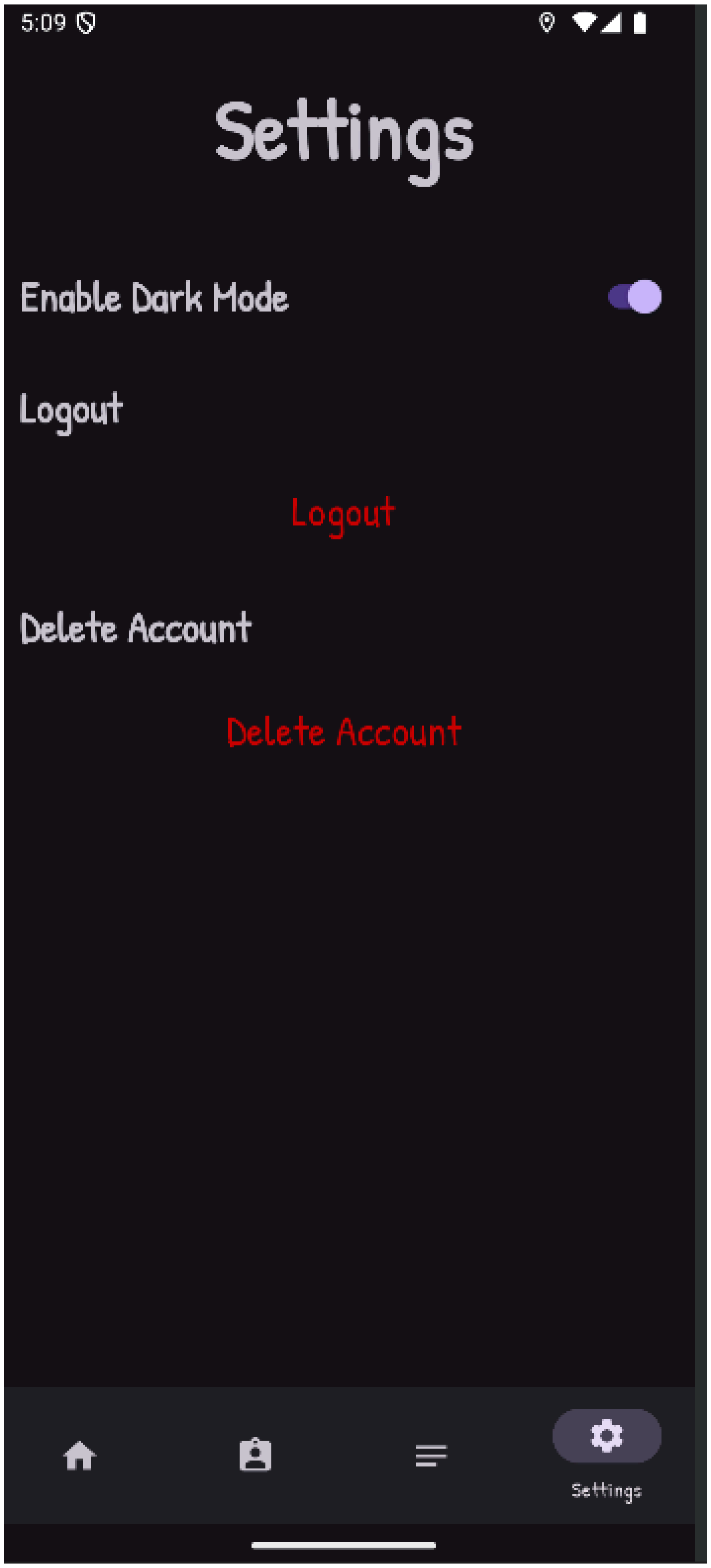
College
Date: 2024-12-05
Time: 11:00
Location: TU Grangegorman
Remove

Work
Date: 2024-12-05
Time: 17:00
Location: Dundrum
Remove

Add Note

Notes

Test Note
Remove


5:09

Settings

Enable Dark Mode

Logout

Logout

Delete Account

Delete Account