UNIVERSITI MALAYSIA TERENGGANU
FACULTY OF OCEAN ENGINEERING TECHNOLOGY & INFORMATICS

Project Management and Monitoring App

CSM3114 FRAMEWORK BASED MOBILE APPLICATION DEVELOPMENT

Prepared for

DR. MOHAMAD NOR HASSAN

Prepared by

OMAR ISMAIL ABDJALEEL ALOMORY(S63955)

Table of Content

EXECUTIVE SUMMARY

The Project Management and Monitoring App is a critical addition to our mobile application portfolio, tailored to meet the evolving demands of efficient project management. Geared towards project managers, this app streamlines project activities, facilitates real-time monitoring, and enhances overall project efficiency.

1) Key Features:

Personalized Task Management:

- Create, organize, and prioritize tasks specifically tailored to your workflow.
- Assign tasks to different projects and set clear deadlines for each.
- Effortlessly track task completion and visualize progress.

Visual Progress Tracking:

- Monitor project status in real-time with customizable dashboards and timelines.
- Employ visual cues like color-coding and progress bars to highlight task urgency and completion.
- View upcoming deadlines and critical milestones at a glance.

2) Target Audience:

The Project Monitoring and Tracking App is tailored for project managers, team leads, and project stakeholders seeking an intuitive and comprehensive tool to streamline project workflows. This audience will benefit from real-time monitoring, and dynamic project management features.

3) Objectives and benefits:

Objectives: Stay organized, prioritize tasks, and boost efficiency.

Benefits: Clear visuals, improved focus, optimized workflow, better estimations, on-time delivery.

Focus: Personal task management and progress visualization.

USE CASE

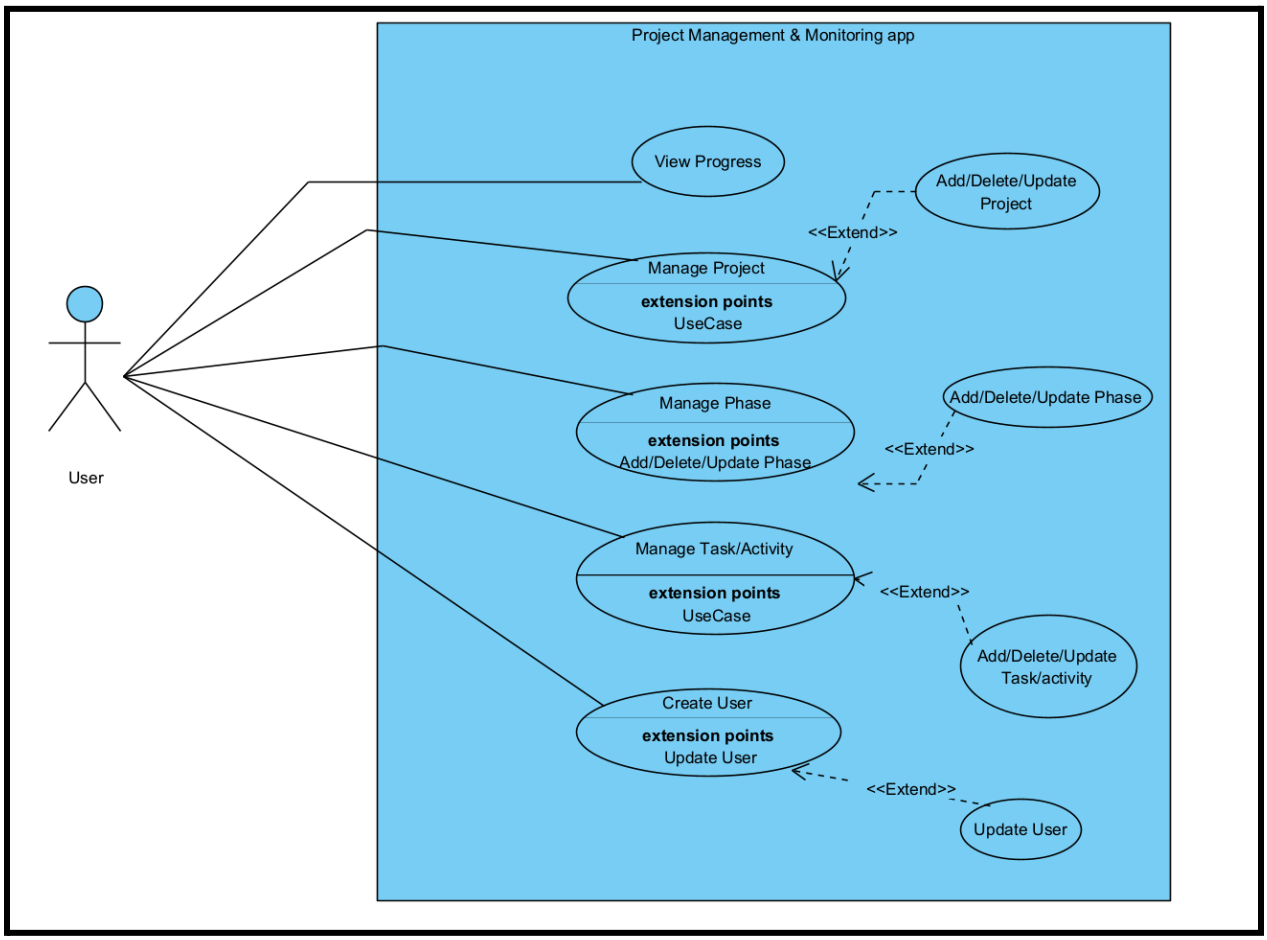

Figure 1: Use Case diagram

Create user : Users have to make an account in order to login into the system, if he already has an account then he can update his information.

Manage Project: User can update/create/delete projects from the database.

Manage Phase: in each project there is phase/s, user can add/delete/update from the projects.

Manage Task/Activity: in each phase/s, user can either add/delete/update tasks and task progress.

View Progress: User can track the progress of the project, which tasks have been completed/not started/ or in progress and sort tasks accordingly.

THE COMMON STRUCTURE OF TREE WIDGET USED WHEN DESIGNING AND DEVELOPING THE APPLICATION

The application adopts a widget tree methodology by implementing the concept of rows and columns to effectively structure the screen. The screen is divided into rows and columns, and widgets are strategically placed based on the UI specifications in the design. To enhance responsiveness, additional widgets are placed at the top of the tree. For example, a SingleChildScrollView is integrated to allow for scrollability, preventing potential app crashes when users add more phases or tasks.

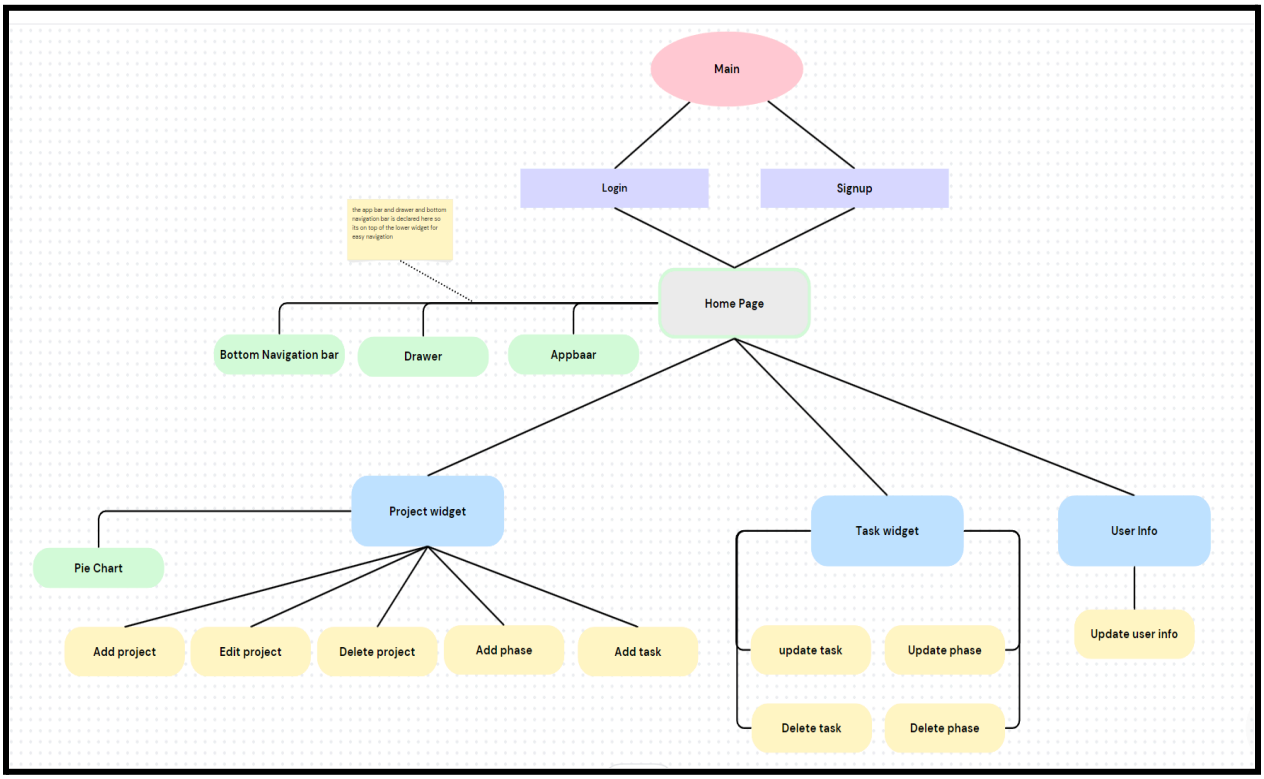

Figure 2: Sitemap of tree widget to show the navigation and flow

Given the limitation of small smartphone screens, specific widgets are utilized to optimize screen usage. Instead of using ListTile, the application opts for ExpansionListTile to enable users to view added phases. Clicking on a phase triggers the expansion of an ExpansionTile, revealing detailed information about the phase, including associated tasks. For user interactions such as

adding or updating, widgets are dynamically generated at the top of the screen using showModalBottomSheet. This design decision aims to improve user convenience and navigation.

In summary, the overall structure of the application revolves around these widget compositions. The construction of each widget aligns with the user interface and functionality requirements. At the topmost layer, the application features a singular AppBar after the user completes the authentication process. This ensures smooth integration with both the drawer and the bottom navigation bar. Below the AppBar, a Container is utilized to contain the background image used throughout the app. The subsequent content comprises the aforementioned widgets, providing a coherent and user-friendly application structure.

FLUTTER WIDGETS AND FEATURES ADOPTED IN THE APPLICATION

1) Widgets Used:

- **StatefulWidget and State:** The app utilizes StatefulWidget and its corresponding State class to manage the mutable state within the app.
- **Scaffold:** The overall app structure is built using the Scaffold widget, providing the basic structure for the visual interface, including the app bar, body, and bottom navigation bar.
- **AppBar:** Customized with a flexible space and an image background.
- **Drawer:** The Drawer widget is used to implement a slide-in menu for navigation, providing quick access to different sections of the app.
- **BottomNavigationBar:** The BottomNavigationBar widget is adopted for easy navigation between the main screens of the app, offering a clean and intuitive user interface.
- **InkWell/GestureDetector :** The InkWell widget is utilized to create interactive ink splashes for the "Create Project" button, making it visually responsive to touch.
- **ListView.builder:** ListView.builder is used dynamically to generate project, phase, and task tiles efficiently based on the length of the project data.
- **ExpansionTile:** The ExpansionTile widget is employed to represent hierarchical data, allowing users to expand and collapse projects, phases, and tasks for a clear and organized view.
- **Container:** Container is used for styling and layout purposes, providing customization for various UI elements.
- **Card:**

Employed for displaying cards containing project information and phase details.

- **IconButton and ElevatedButton:**

Used for various actions like editing, deleting, and updating tasks and phases.

- **Column and Row:**

Organizing and arranging widgets in a column or row structure for better layout.

- **Radio:**

Utilized for selecting task status (Not Started, In Progress, Completed).

- **PieChart (from the pie_chart library):** The PieChart widget from the pie_chart library is adopted for visualizing data in a pie chart format. This is likely used to represent project or task progress in a graphical manner.
- **DatePicker:** Utilized for selecting due dates for tasks.

2) Features:

Project Management:

- Users can create projects using the "Create Project" button.
- Projects are displayed in a list with project details in the dashboard.

- Task Management:

- Tasks are categorized as "Not Started," "In Progress," and "Completed."
- Tasks are displayed in the Tasks section.
- The user can sort tasks using a dropdown menu.

- Firebase Integration:

- Utilizing Firebase for storing and retrieving project, phase, and task data.
- Making use of HTTP requests (http package) for CRUD operations on Firebase.

- Modal Bottom Sheets:

- Displayed when editing tasks and phases, allowing users to modify details.

- ExpansionTile:

- Providing an expandable view for each phase, showing associated tasks.

- Task Sorting:

- Implemented a dropdown for sorting tasks based on status (Not Started, In Progress, Completed, All Tasks).

- Pie Chart:

- Utilized the pie_chart library for visualizing task completion statistics.

- Navigation:

- Bottom navigation bar is used for quick navigation between dashboard, tasks, and user info.

- Styling and Theming:

- The app uses a custom theme with specific color choices for a visually appealing design.

- Error Handling:

- Error messages are displayed using snack bars when there is an issue with the project.

SAMPLES OF THE INTERFACE WITH UI

1) Authentication Screens

In order to use the app, user have to either signup or register if account already exists

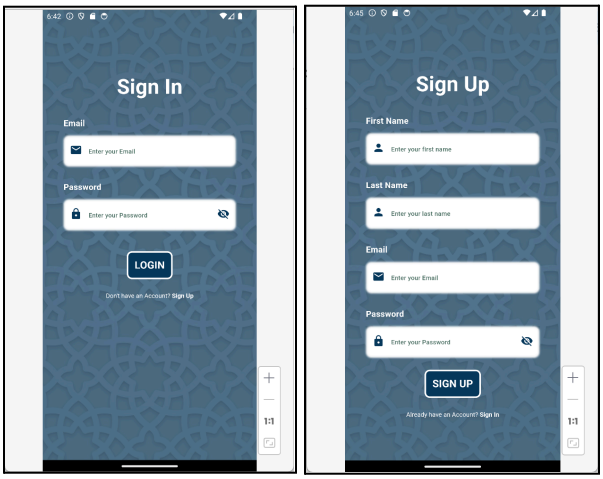

Figure 3&4: showing authentication screens for the application

2) Dashboards screens

The dashboards are the home for project management and monitoring applications, it would display views of the entire project progress and other functionalities such as adding/updating/deleting project/phase/task. The drawer function is to aid users in logout, display and navigate projects, and create new projects.

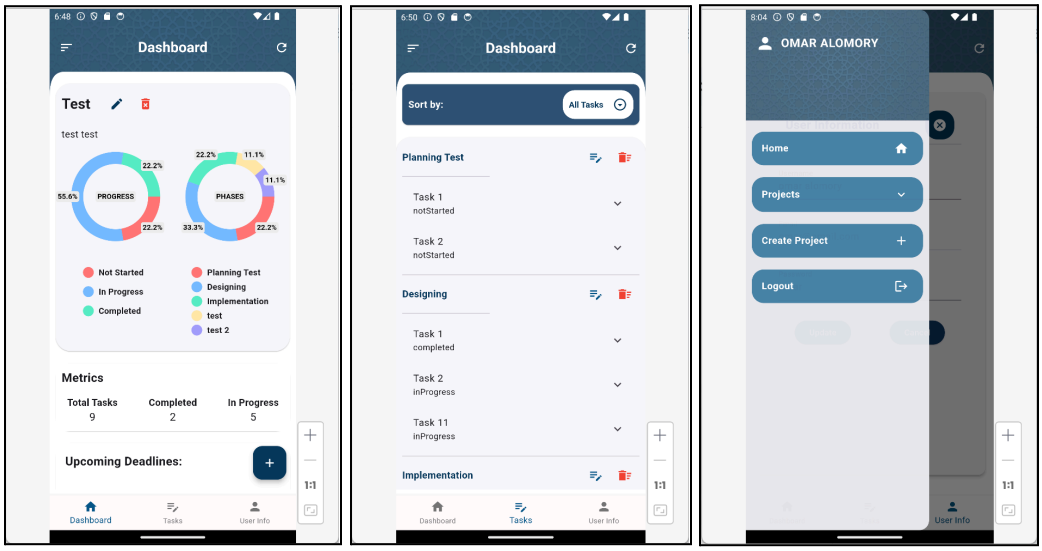

Figure 5&6&7: Dashboards, consist of (Dashboard, Tasks, Drawer)

2) Add/Edit Project

These widgets are provided to aid users in creating new projects or updating existing ones.

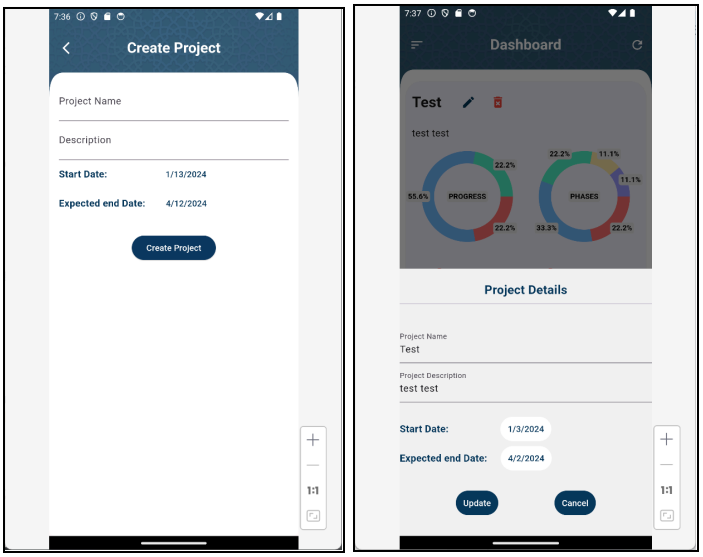

Figure 8&9: Creating/Editing project

3) Add/Edit Phase

After the user finishes adding a project, then the functionality of adding phase will show up, if the user wants to update it, the functionality is added in the task tab.

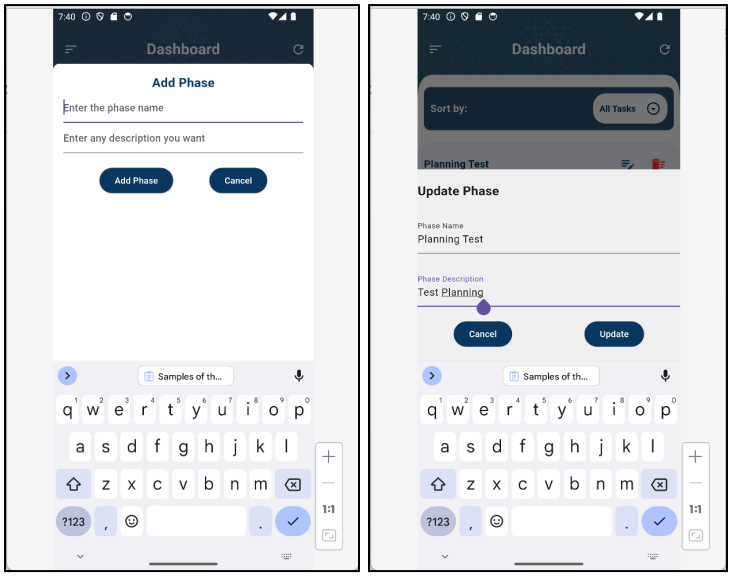

Figure 10&11: Adding/Editing phase

4) Add/Edit Task

This is the core functionality and what makes the app worthy, after the user adds phase, then he has the functionality to add or update tasks. The tasks will be stored and then monitored. Based on the app status, the progress of the entire project will change in dashboard(view progress).

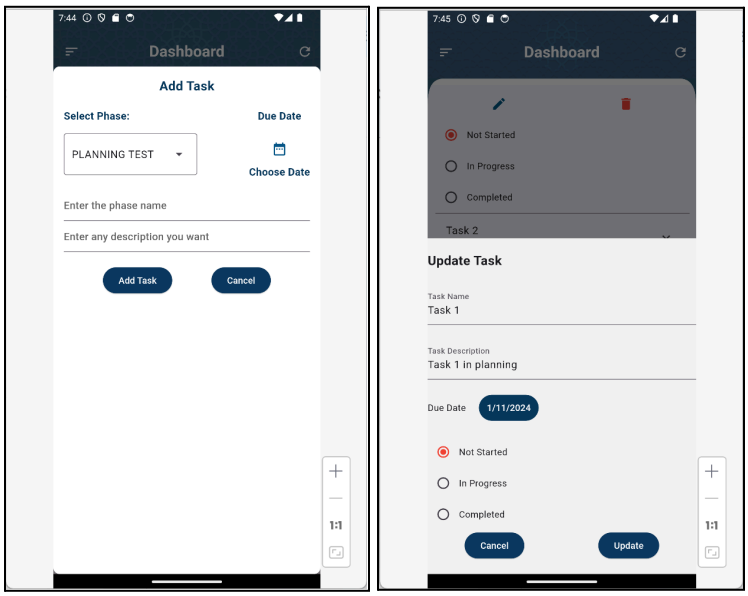

Figure 12&13: Creating/Editing project

5) User information

User can view/modify the personal information he/she have in this section

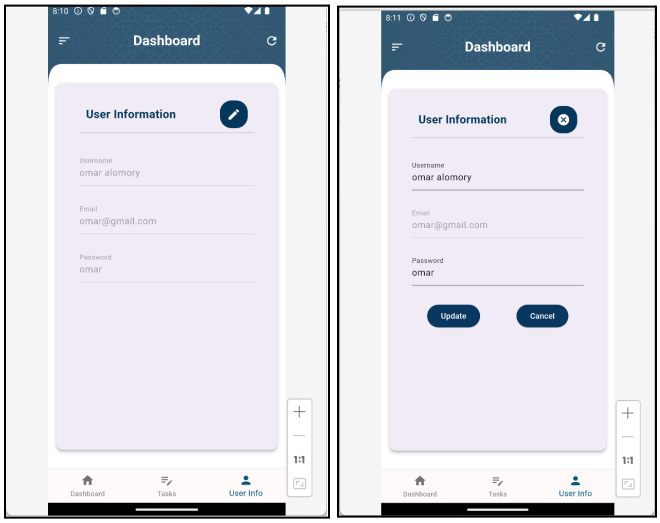

Figure 14&15: User Information

CONCLUSION

Forged on the foundation of Flutter, the Project Management and Monitoring App stands as a pinnacle in project efficiency, empowering both managers and stakeholders. With its sophisticated task management, real-time progress tracking, and seamless Firebase integration, the app not only streamlines workflows but also becomes a catalyst for heightened focus and sustained success. Through an interface designed for user intuitiveness, this application transcends conventional project management, offering an unparalleled journey towards project triumph.

REFERENCES

Darji, P. (2024, January 1). Change drawer icon in flutter [color and size]. FlutterBeads. https://www.flutterbeads.com/change-drawer-icon-in-flutter/#Steps-to-Change-Drawer-Icon-in-Flutter

Jay TilluJay Tillu , MichaelMMichaelM , & Ignacio Tomas CrespoIgnacio Tomas Crespo . (1965a, September 1). Flutter: How to add icon to text?. Stack Overflow. https://stackoverflow.com/questions/57897786/flutter-how-to-add-icon-to-text

Ny RegencyNy Regency. (1964, September 1). Flutter: Appbar background image. https://stackoverflow.com/questions/52160746/flutter-appbar-background-image

JAgüeroJAgüero. (1968, September 1). Change image.asset opacity using opacity parameter in image widget. Stack Overflow. https://stackoverflow.com/questions/73490832/change-image-asset-opacity-using-opacity-parameter-in-image-widget

Ammy KangAmmy Kang (1964, June 1). Flutter navigation drawer hamburger icon color change. Stack Overflow. https://stackoverflow.com/questions/50580234/flutter-navigation-drawer-hamburger-icon-color-change

Free vector: Flat Arabic pattern background. Freepik. (n.d.). https://www.freepik.com/free-vector/flat-arabic-pattern-background_15365516.htm#query=arabic%20pattren&position=1&from_view=search&track=ais&uuid=eb5a555e-1d49-47c5-8a37-dca13b989bf6

Bottomnavigationbar class. BottomNavigationBar class - material library - Dart API. (n.d.). https://api.flutter.dev/flutter/material/BottomNavigationBar-class.html

Pie_chart: Flutter Package. Dart packages. (2023, November 14). https://pub.dev/packages/pie_chart/example

HTTP: Dart Package. Dart packages. (2023a, November 27). https://pub.dev/packages/http

INTL: Dart Package. Dart packages. (2023b, December 7). https://pub.dev/packages/intl

Add a drawer to a screen. Flutter. (n.d.). https://docs.flutter.dev/cookbook/design/drawer

The design of the application: Google. (n.d.). Malaysia Airlines - apps on Google Play. Google.https://play.google.com/store/apps/details?id=aero.sita.lab.resmobileweb.android.mh&hl=en&gl=US

ChatGPT: https://chat.openai.com/share/4ca8fe21-95a6-4631-b64d-7c882e762f75