Pakistan Ordnance Factories (POF)

Internship Weekly Report – Week 3

Name: Ahmer Moon Majid

Instructor: Madam Ansa

Department: IT Department

Technology: Flutter (Mobile App Development)

Duration: 6 Weeks

Week: 3

Report Date: 8/27/2025

Task 1: To-Do List Application with Shared Preferences

Assigned Task:
The task was to develop a full-featured To-Do List application capable of creating, reading, updating, and deleting tasks. The application needed to include task prioritization (Low, Medium, High), mark tasks as complete, and organize them into tabs (All, Pending, Completed). A critical requirement was to persist all task data locally on the device using the shared_preferences package so that the user's data remains available between app sessions.

Key Widgets and Approach:

1. **Data Model (**Task **Class)**
    - Created a custom Task class with properties for title, description, priority (using an enum), and isDone status.
    - Implemented toJson() and fromJson() methods for seamless serialization and deserialization to store tasks as JSON strings.

2. **Local Data Persistence (**TaskStorage **Service)**
    - Utilized the shared_preferences package to save and load data from the device's local storage.
    - The saveTasks() method converts the list of tasks to a JSON string and saves it.
    - The loadTasks() method retrieves the string, decodes it, and converts it back into a list of Task objects.

3. **Tabbed Navigation**
    - Implemented a TabBar and TabBarView within the AppBar for organizing tasks into three categories: All, Pending, and Completed.
    - Used a TabController to synchronize the tab bar and the tab view

4. **Home Screen (**HomeScreen **Widget)**
    - Managed the main application state using a List<Task> and various functions for CRUD operations (_addTask, _updateTask, _toggleTask, _deleteTask).
    - Used ListView.builder to efficiently render the list of task items.
    - Implemented a FloatingActionButton to navigate to the AddTaskScreen.

5. **Add/Edit Screen (**AddTaskScreen **Widget)**
    - Created a reusable form screen that functions for both adding new tasks and editing existing ones, determined by the presence of a optional task parameter.
    - Used TextEditingController to manage input fields for title and description.
    - Implemented a DropdownButtonFormField for selecting task priority.

- Passed the new or updated Task object back to the HomeScreen using Navigator.pop(context, newTask).

6. **Custom List Item (**TaskTile **Widget)**
   - Designed a reusable TaskTile widget using a Card and ListTile to display each task consistently.
   - Included a Checkbox for toggling completion status, which strikes through the task text.
   - Added IconButtons for editing and deleting the task.
   - Used an Icon(Icons.flag) whose color changes based on the task's priority (Colors.red for high, Colors.orange for medium, Colors.green for low).

7. **State Management**
   - Utilized core Flutter state management (setState) within the HomeScreen to manage the list of tasks and trigger UI rebuilds after any data changes.

Output Screenshots:

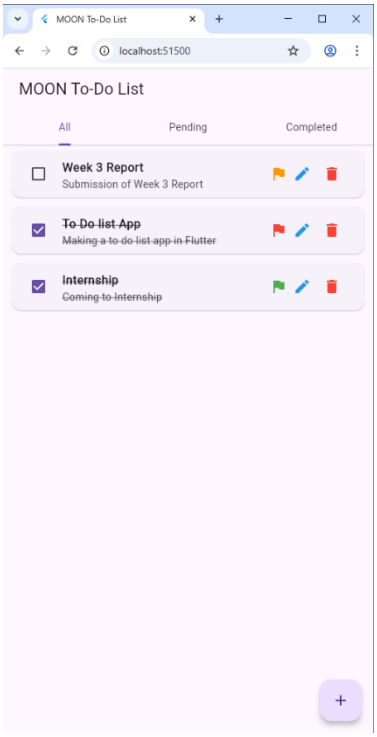

Figure 1: Homescreen

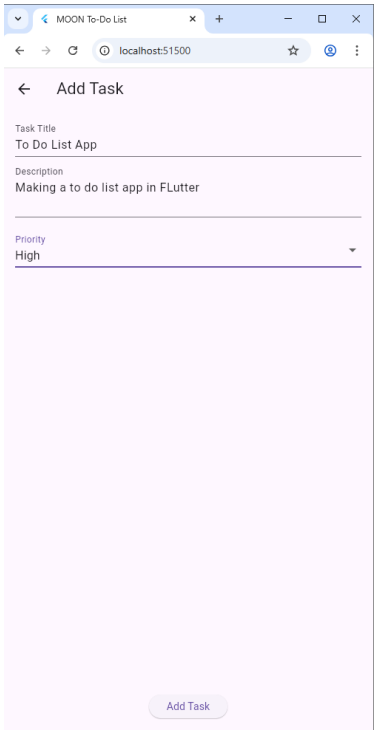

Figure 2: Add Task Screen

Figure 1: Edit Task Screen

Task 2: POF e-Services App Clone

Assigned Task:
The task was to develop a multi-screen clone of the Pakistan Ordnance Factories (POF) e-Services employee portal application. The objective was to replicate the official app's look, feel, and core navigation flow, including a branded splash screen, a functional login interface, a main dashboard with a grid of services, and specific service screens like Complaint Registration, CMA Pay Slips, and a Change Password form. The focus was on achieving high visual fidelity and a seamless user experience.

Key Widgets and Approach:

1. **Splash Screen (**SplashScreen**)**
    - Utilized an AnimatedContainer to create a smooth scaling animation for the app logo, growing from 100.0 to 200.0 pixels.
    - Implemented a Future.delayed timer to automatically navigate to the login screen after a 5-second duration using Navigator.pushReplacementNamed.
    - Set a semi-transparent background image (AssetImage) to match the authentic app's aesthetic.
2. **Login Screen (**LoginScreen**)**
    - Designed custom text fields for email and password by using a Container as the prefixIcon. This container housed both an icon and a vertical divider, creating a polished, branded input style.
    - Used OutlineInputBorder with borderSide: BorderSide.none and a white fillColor to achieve the modern, card-like input field design.
    - Implemented a prominent green ElevatedButton for login and a text OutlinedButton linking to the Login Instructions screen.
3. **Login Instructions Screen (**LoginInstructionsScreen**)**
    - Created a visually instructional page with a numbered list of steps using Text widgets.
    - Built a mock video player UI using an AspectRatio widget, a Container with a BoxShadow, and a Stack containing a play icon (Icons.play_circle_fill) and a custom progress bar built with a LinearProgressIndicator to simulate video controls.

4. **Home Screen / Dashboard (**HomeScreen**)**
    - Constructed the main service grid using GridView.count with a crossAxisCount of 3 to create a 3-column layout of service tiles.
    - Developed a reusable _buildGridItem() function that returns an InkWell wrapped in a Card for each service item. This function displays the service's icon (Image.asset) and label.
    - Implemented navigation logic within _buildGridItem() using conditional checks (if (label == 'Station Engineer')) to push the appropriate screen (e.g., ComplaintScreen, CmaScreen) onto the navigation stack when a tile is tapped.
5. **Complaint Screen (**ComplaintScreen**)**
    - Designed a summary dashboard using a Column of Card widgets to display "Total Complaints", "Open Complaints", and "Closed Complaints".
    - Used a Row with Expanded widgets to ensure the "Open" and "Closed" complaint cards share the available space evenly.
    - Added a bottom row of action buttons (Register Complaint, Open Complaints, Closed Complaints) using the same _buildActionButton method for consistency.
6. **CMA Screen (**CmaScreen**)**
    - Displayed a scrollable list of the last 12 months for pay slips using ListView.builder for efficient rendering.
    - Each pay slip month is displayed in a ListTile within a Card, featuring a leading icon and a trailing download button (IconButton with Icons.download).
7. **Change Password Screen (**ChangePasswordScreen**)**
    - Built a form with three password fields using custom _buildPasswordField helper method.
    - Implemented basic validation logic to check for empty fields and password matching before showing a success SnackBar.
    - Styled the text fields with a white background, shadow, and a green prefix icon to maintain UI consistency.

8. **Navigation & Theming**
    - Configured app-wide navigation using named routes (/login, /home) in the MaterialApp widget.
    - Set a global app theme (ThemeData) with a green primary swatch and the 'Roboto' font family to ensure design consistency across all screens.
    - All AppBars were standardized with a green background (backgroundColor: Colors.green) and white text.

Output Screenshot:

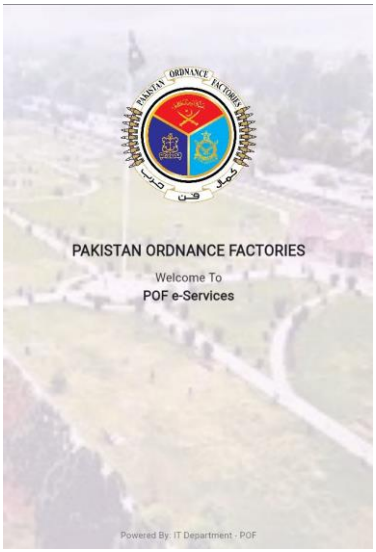

Figure 1: Splash Screen

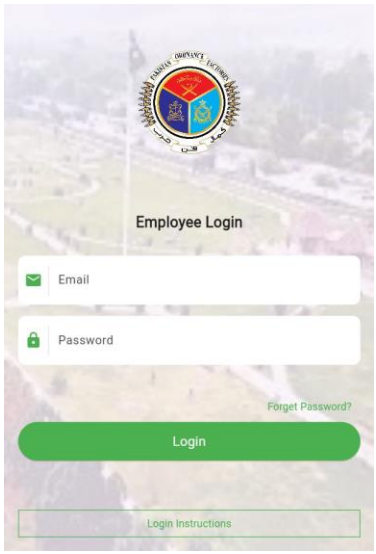

Figure 2: Login Screen

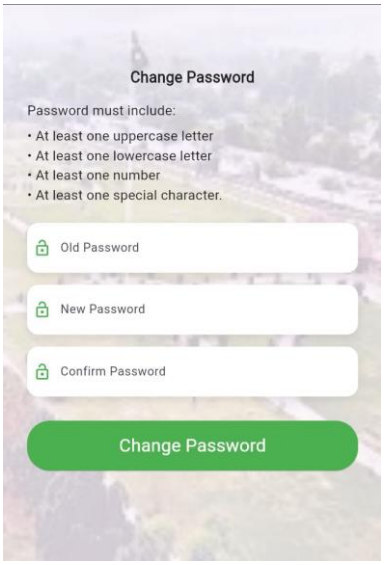

Figure 3: Change Password Screen

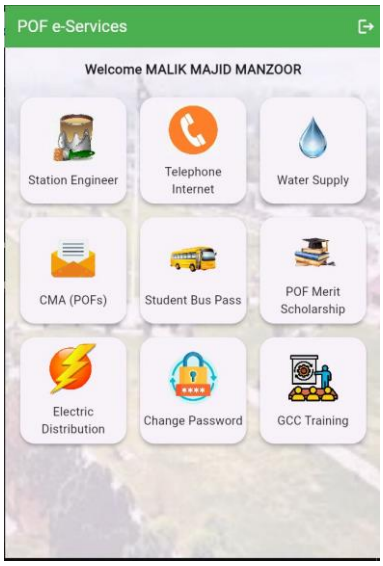

Figure 4: Home Screen

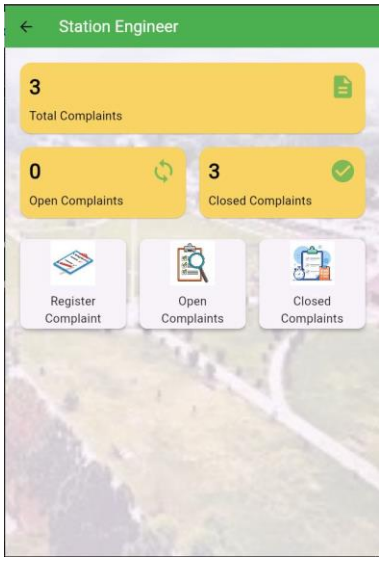

Figure 5: Complaints Screen

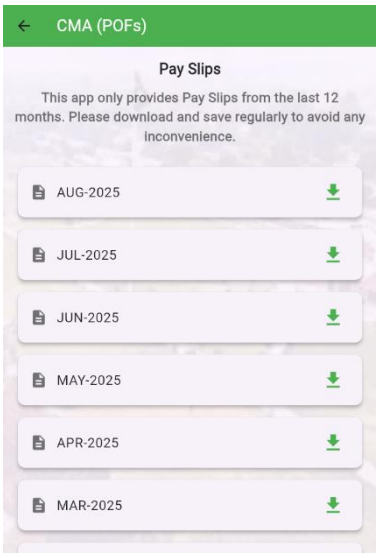

Figure 6: CMA Screen