# Mid-Term Task FALL 2024

### Development of a Comprehensive Task Management Application

**Note:** Copying code from another student will result in **zero marks** for both parties involved due to plagiarism.

**Course:** Mobile Application Development

**Instructor: Muhammad Abdullah (**githubprojectmine@gmail.com**)**

**Submission Date: Due Date.**, 08/11/2024
**Total Marks: 100** [marks distribution provides end of this document]

**Objective:**

The aim of this project is to develop a fully functional task management application using Flutter and SQLite. The application allows users to manage their daily tasks effectively through features like task addition, modification, deletion, categorization, and custom notifications.

**Project Description:**

You are required to create a task management application that not only handles the basic management of tasks but also includes advanced functionality such as task repetition, progress tracking, and data export options. The application should offer a user friendly interface, robust data storage solutions, and customization capabilities.

## Functional Requirements:

1. **Setup and Configuration**:
   - Set up the Flutter environment and create a new Flutter project.
   - Initialize a SQLite database to manage task data persistently.
2. **User Interface**:
   - Design a simple and intuitive interface with the following views:
     - **Today Task:** Tasks due on the current day.
     - **Completed Task:** Tasks that have been completed.
     - **Repeated Task:** Tasks that recur on specified days or intervals.
3. **Task Management**:
   - **Add Tasks**: Users should be able to add tasks with details like title, description, due date, and optional repeat settings.
   - **Edit Tasks**: Users should have the ability to modify task details.
   - **Delete Tasks**: Implement functionality to remove tasks.
   - **Mark as Completed**: Allow users to mark tasks as completed, moving them automatically to the 'Completed Task' category.
4. **Advanced Features**:
   - **Customization Options**: Implement theme customization (e.g., light and dark mode) and allow users to choose notification sounds.

    - **Progress Tracking**: Allow users to add subtasks and display a progress bar or percentage for task completion.
    - **Export Functionality**: Enable users to export tasks to CSV, PDF, and email.
    - **Repeat Tasks**: Tasks can be set to repeat daily or on selected days of the week, automatically resetting based on user-defined settings.
5. **Notifications**:
    - Implement local notifications to alert users of upcoming tasks based on their due dates and times.
6. **Testing**:
    - Thoroughly test the application for functionality, usability, and responsiveness.
    - Identify and fix any bugs or issues.
7. **Documentation and Submission on GitHub**:
    - Submit the complete source code .apk along with any necessary configuration files and a video demonstration of the application.

## Recommended Tools:

- Flutter for UI and application logic.
- SQLite for data management.
- Any additional Flutter libraries or plugins that are found necessary.

## Estimated Time Commitment:

- Total Time: Approximately 42 hours.
- Daily Commitment: Aim to dedicate 1.5 to 2 hours per day.

## Support and Resources:

- Access to Flutter's official documentation and various online tutorials.
- Weekly check-in sessions for project guidance and troubleshooting.

## Marks Distribution for Task Management Application Project

| Component | Description | Marks |
|---|---|---|
| **1. Project Setup and Design** | Includes setting up the Flutter environment and the initial design of the user interface. | 10 |
| **2. Database Integration** | Correct setup of SQLite and schema design for managing tasks. | 10 |
| **3. Task Management Features** | Implementation of adding, editing, deleting, and marking tasks as completed. | 20 |
| **4. Advanced Features** | Customization options, progress tracking, and export functionalities. | 20 |

| | | |
|---|---|---|
| **5. Repeat Functionality** | Correct implementation of repeating tasks based on user defined settings. | 10 |
| **6. Notifications** | Implementation of local notifications for upcoming tasks. | 10 |
| **7. User Interface** | Usability, aesthetics, and responsiveness of the application. | 10 |
| **8. Documentation and Submission on GitHub**: | Quality of the code documentation, user manual, and video demonstration. | 10 |
| **Total** | | 100 |