IT2010 – Mobile Application Development
BSc (Hons) in Information Technology
2nd Year
Faculty of Computing
SLIIT

2024 – Lab Exam 04

Start Date: May 6, 2024
Deadline: May 12, 2024, Midnight
Viva Date: Starts from May 13, 2024

Overview:

In this lab assessment you need to develop a task management application. Throughout the app development process, you need to commit your work using git. Create a public git repository in GitHub and you need to push your work to the GitHub remote repository regularly.

Objectives:

The objective of this assignment is to develop a mobile application using Android Studio and Kotlin that implements various key components of Android development, including SQLite databases with Room library, Kotlin Coroutines, RecyclerView, and ViewModel. Students will create a dynamic task management app that allows users to add, delete, update, and view tasks using a modern and efficient architecture.

Requirements:

1. **Task Management Interface:**
    - Design and implement an intuitive user interface for managing tasks.
    - Include options for adding new tasks, viewing existing tasks, updating task details, and deleting tasks.
    - Utilize RecyclerView to display a list of tasks in a scrollable format.
2. **SQLite Database Integration:**
    - Implement a SQLite database using Room library to store task data locally on the device.
    - Define an Entity for tasks, including properties such as task name, description, priority, deadline, etc.
    - Set up a Data Access Object (DAO) for performing CRUD (Create, Read, Update, Delete) operations on the task database.

3. **Kotlin Coroutines:**
    - Utilize Kotlin Coroutines for performing asynchronous database operations.
    - Implement Coroutines for tasks such as inserting, updating, deleting, and querying tasks from the database.
    - Ensure that long-running operations are executed off the main UI thread to maintain a responsive user experience.
4. **ViewModel Architecture:**
    - Implement the ViewModel architecture to separate UI-related data from UI controller logic.
    - Create a ViewModel class that encapsulates the app's UI data and business logic related to task management.
    - Utilize LiveData or Flow to observe changes in the task data and update the UI accordingly.
5. **User Interaction:**
    - Implement user interaction functionalities such as:
        - Adding new tasks with details such as name, description, priority, deadline, etc.
        - Viewing the list of tasks with options to sort or filter based on priority, deadline, etc.
        - Updating task details such as name, description, or priority.
        - Deleting tasks from the list.
6. **Enhancements:**
    - You need to add additional features or enhancements to make the app more user-friendly or efficient.
    - Suggestions include adding search functionality, implementing task categories or tags, setting reminders for tasks, etc.

Submission:

- You need to submit a report using the template provided before the deadline.
- Include the GitHub Repository (Double check the link)
- Include Screenshots of the app.

Evaluation:

- Code quality and organization.
- Functionality: How well your app works and adheres to the specified requirements.
- Creativity and user interface design.
- Git Commits throughout the project.
- Viva: Be prepared to explain your code and design decisions during the viva session starting from May 13, 2024.

Plagiarism Violation Notice:

Plagiarism is strictly prohibited and will result in severe consequences. All submissions must be your original work, and any resources or references used must be appropriately cited. Cases of academic dishonesty, including but not limited to copying code, using unauthorized materials, or helping others to plagiarize, will lead to disciplinary actions as per institutional guidelines.

Marking Guide for Assignment

1. Task Management Interface (1.5 marks)
    - Intuitive UI design (0.5)
    - Options for adding, viewing, updating, and deleting tasks (0.5)
    - Utilization of RecyclerView (0.5)
2. SQLite Database Integration (1.5 marks)
    - Proper implementation of Room library (0.5)
    - Correct definition of Entity and DAO (0.5)
    - CRUD operations functioning correctly (0.5)
3. Kotlin Coroutines (1.5 marks)
    - Effective use of Kotlin Coroutines (0.5)
    - Asynchronous database operations (0.5)
    - Responsiveness maintained through off-main thread execution (0.5)
4. ViewModel Architecture (1.5 marks)
    - Implementation of ViewModel architecture (0.5)
    - Proper encapsulation of UI data and business logic (0.5)

- Utilization of LiveData or Flow for observing data changes (0.5)

5. User Interaction (1.5 marks)
    - Implementation of user interaction functionalities (1.0)
    - Task addition, viewing, updating, and deletion functioning correctly (0.5)
6. Enhancements (1 mark)
    - Addition of user-friendly or efficient features (1.0)
7. Viva (1.5 marks)
    - App is working (1.0)
    - Confidently answering the questions (0.5)