Hilt and Retrofit

CSD3156 Mobile and Cloud Computing Spring 2026

Overview

This lab provides exercises and guidelines to gain familiarity with the Android Hilt, Coroutine, and Network.

You are going to build an app to check if an input repository is among the popular **Android** repositories in GitHub by querying GitHub API using and store the found one in a local Room Database, using hilt for dependency injection and flow to implement the data flow in the MVVM/MVI.

Please refer to the demo given in the class:

https://github.com/csd3156/jetpackArchTest/tree/hilt

Hilt
https://developer.android.com/training/dependency-injection
https://developer.android.com/jetpack/compose/libraries#hilt
https://github.com/android/architecture-samples
https://github.com/android/architecture-templates

Coroutines and Flow
https://developer.android.com/kotlin/coroutines
https://developer.android.com/kotlin/flow

Retrofit
https://developer.android.com/codelabs/basic-android-kotlin-compose-getting-data-internet#0
https://developer.android.com/codelabs/android-paging#4/

Outcomes

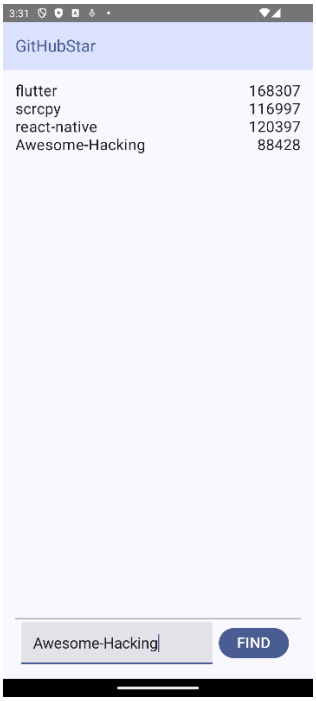
Upon completion of the session, you should be able to:

- Use hilt for dependency injection in Android
- Use Coroutines/flow with Room Database/RESTful API
- Retrieve data using HTTP client, like Retrofit.

Creating and using Hilt, Coroutine, Retrofit and Flow with Room Database

The goal of this exercise is to practice using hilt/coroutine/retrofit/flow in the MVVM/MVI architecture as well as performing HTTP request.

Fork the repo **csd3156-lab04-2026.**



The app has only one Activity: **MainActivity**. Note that the top part is a LazyColumn (testTag: **repoList**), the columns in the LazyColumn are one Row of two Texts for the repo name (testTag: **repoName**) and number of stars (testTag: **repoStar**) (the same testTags are shared for Rows of Texts). The TextField is editable (testTag: **inputRepo**), the name of one repo to search will be entered. Hit the FIND Button (the

Text in the Button is "**FIND**") to start the query. As always, you need to recreate a UI similar to the screenshot above.

IMPORTANT: Ensure that the activity is named as MainActivity:
- The repos found are displayed in a LazyColumn with the testTag "repoList", e.g., using Modifier.testTag("repoList")
- In each row: repo name (testTag: repoName) and number of stars (testTag: repoStar)
- TextField with the testTag "inputRepo"
- Button with the name: "FIND"

Your task is to create an app using MVVM/MVI architecture with a Room Database, HTTP client (e.g., Retrofit), Flow, LazyColumn, Repository, ViewModel, and etc with Hilt for dependency injection.

1. Check if the input Repo name is already in local Room Database.
2. If it is not in the local Room Database, then query https://api.github.com/search/repositories?sort=stars&q=Android to check if it is among the top 50 repositories about Android sorted by the number of stars. You can set the response page-size to realize this.
3. If it is among the top 50, then add it into the Room Database, otherwise ignore the input.
4. The data in Room Database will be displayed in the LazyColumn.

Lab Exercise 4

Due Date: Wed, Feb 11 2026 2359 hrs

Fork the repo **csd3156-lab04-2026** and then clone it.

1. Implement the code needed to create the MVVM/MVI architecture with a Room Database, HTTP client (e.g., Retrofit), Flow, LazyColumn, Repository, ViewModel, and etc with Hilt for dependency injection.
2. Implement the code needed for query GitHub API (e.g., Retrofit).
3. Commit and push all changes to your forked repository.

END OF DOCUMENT