Data Management & Architecture Components

DataStore, Room, StateFlow/LiveData & ViewModel

CSD3156 Mobile and Cloud Computing Spring 2026

Overview

This lab provides guidelines to create a simple app to gain familiarity with Android Data Management and Architecture Components, namely DataStore, Room, StateFlow/LiveData & ViewModel.

Outcomes

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

- Create a simple interactive interface
- Implement Android app architecture using StateFlow/LiveData & ViewModel
- Create and populate a simple SQLite database using the Room architecture component
- Save app settings using Preferences DataStore

Tutorials & Codelabs

Please refer to the demo given in the class:

https://github.com/csd3156/jetpackarchtest/

The following are references to help you learn about the concepts in this exercise:

Android Architecture

https://developer.android.com/topic/architecture/intro

https://developer.android.com/courses/pathways/android-architecture

ViewModel

https://developer.android.com/topic/libraries/architecture/viewmodel

https://developer.android.com/courses/pathways/android-basics-compose-unit-4-pathway-1
https://developer.android.com/codelabs/basic-android-kotlin-compose-viewmodel-and-state
Room:
https://developer.android.com/training/data-storage/room
https://developer.android.com/courses/pathways/android-basics-compose-unit-6-pathway-2

Preferences DataStore:

https://developer.android.com/topic/libraries/architecture/datastore
https://developer.android.com/codelabs/android-preferences-datastore

StateFlow
https://developer.android.com/topic/architecture/ui-layer/state-production

LiveData
https://developer.android.com/topic/libraries/architecture/livedata

https://developer.android.com/develop/ui/compose/state

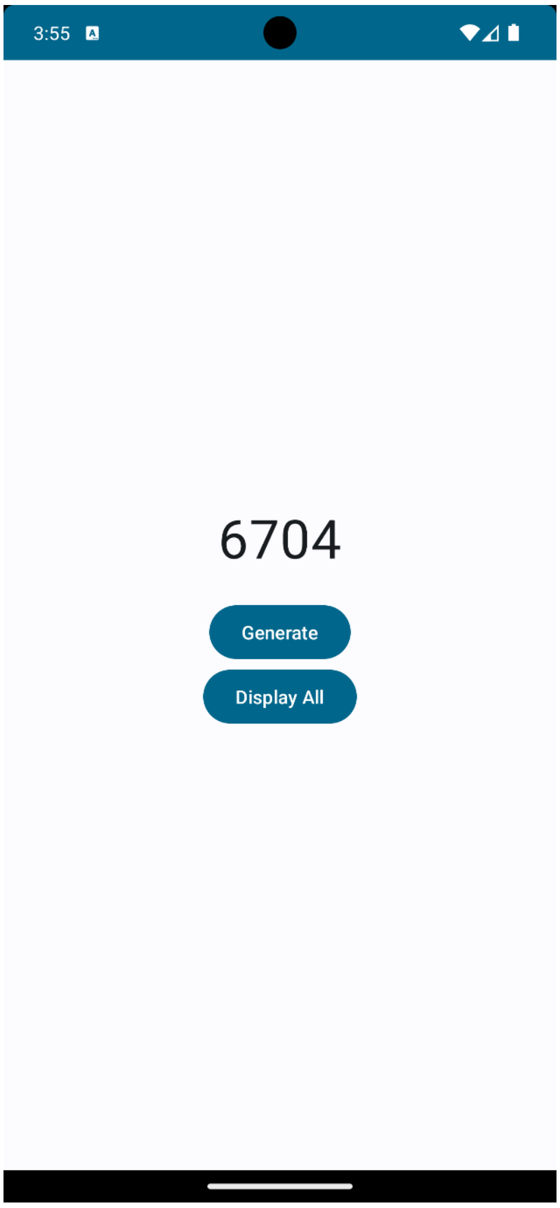
The random four-digit generator application

This section provides an exercise to build a simple four-digit generator application (this has nothing to do with a certain popular 4D lottery game in Singapore). It consists of a user interface to randomly generate a four-digit number and save it to a SQLite relational database, using the Room architecture component.

The application also includes a button to display a list of all the four-digit numbers that were saved to the database. This list should be displayed on a separate screen, using **LazyColumn** or **LazyVerticalGrid**.

Fork the repo **csd3156-lab03-2026** to get started. The application has only one activity: **MainActivity**. Design the layout of the screen for the `Generator` to be similar to the following screenshot.

Implement the logic to **randomly** generate a four-digit number between 1000 and 9999.



MainActivity

IMPORTANT: Ensure that the activity is named as **MainActivity**:

- The number generated is a Text with the testTag “FourDigit”, e.g., using Modifier.testTag("FourDigit")
- Buttons with names: “Generate” and “Display All”

Saving four-digit numbers to a Room (SQLite) database

Next, implement architecture components to save every generated four-digit value to a Room database (backed by SQLite), together with a ViewModel and StateFlow or LiveData, such that the user interface will be updated for every new saved value.

- Hitting the "Generate" button will create a new random four-digit number and insert it through ViewModel and Repository into the Room database, cause the data is updated, then Repository and ViewModel will be updated and notify UI to display.
- When the screen is rotated, the last generated four-digit value is maintained on the screen.

There are <u>two columns</u> for the SQLite database, <u>the ID, or primary key</u> for each record in the table (this is auto incremented and auto generated), and the <u>value of the four-digit number</u> itself.

Hint: The following classes and interfaces are most probably needed: **FourDigit, FourDigitDao, FourDigitRepository, FourDigitRoomDatabase, FourDigitViewModel.**

Hitting the "Display All" button will navigate to another screen to the screen for the **`Display.`**

Displaying a list of four-digit numbers

The **`Display`** screen should display the list of randomly generated four-digit numbers that were stored in the database, and the autogenerated <u>database index number</u>.

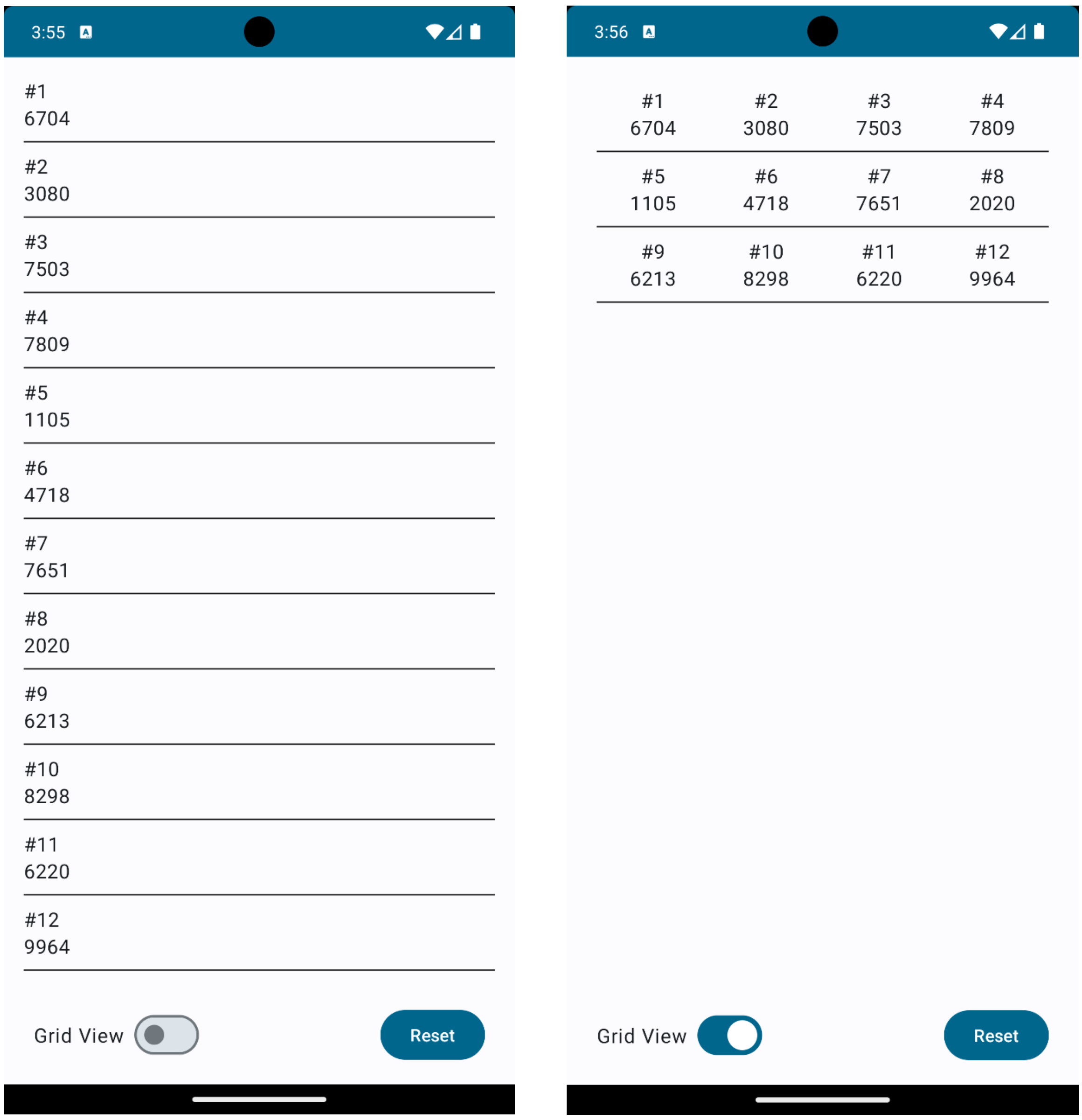
When the **`Display`** screen is rotated, the list of saved four-digit values are maintained on the screen. Hitting the back button will return to the main **`Generator`** screen with the most recently generated four-digit number.

You can choose to let the two screens share the same ViewModel, or define another one for the **`Display`** screen.

Additionally, you are to add a *Switch* button at the bottom of the **`Display`** for switching between normal (**LazyColumn**) and grid (**LazyVerticalGrid**) views (**firstly started as off** = normal, on = grid view). *This setting must be saved using Preferences*

DataStore, such that the setting is persisted between instances (i.e., the switch setting should remain as it was set by the user, even if the device is restarted).

Finally, add a button called “Reset” at the bottom that clears all data from the database.


3:55

#1
6704

#2
3080

#3
7503

#4
7809

#5
1105

#6
4718

#7
7651

#8
2020

#9
6213

#10
8298

#11
6220

#12
9964

Grid View Reset

3:56

#1 6704	#2 3080	#3 7503	#4 7809
#5 1105	#6 4718	#7 7651	#8 2020
#9 6213	#10 8298	#11 6220	#12 9964

Grid View Reset

The Display Screen in Normal and Grid View

IMPORTANT: Ensure that the Text “**Grid View**” and Button “**Reset**” are displayed with the correct names.

- The switch is off when app is firstly started, id is of "**GridSwitch**" as the testTag, e.g., using Modifier.testTag("GridSwitch")

Lab Exercise 3

Due Date:

Wed Feb 04, 2026 2359 hrs

1. Fork the repo **csd3156-lab03-2026.**
2. Design the layouts of the screen similar to the given screenshots.
3. Implement the logic on **`Generator`** screen, to generate a random four-digit number each time the "Generate" button is clicked and the last generated number will be maintained even after the rotation of the phone.
4. Implement the navigation to navigate to the **`Display`** screen when "Display All" button is clicked.
5. Implement the logic to create a simple database table for storing the four-digit values. The rows should consist of the id (auto-generated/incremented), and the four-digit number itself.
6. Implement the logic to add a new random number to the database each time the "Generate" button is clicked.
7. Implement the app architecture using Room, StateFlow/LiveData & ViewModel
8. In the **`Display`** screen, all the random numbers in the database are displayed and updated in LazyColumn or LazyVerticalGrid..
9. In the **`Display`** screen, add a *Switch* button for toggling the grid view on or off. This setting should be saved using Preferences DataStore, such that the setting is persisted between instances.
10. In the **`Display`** screen, add a button, "Reset", that clears all four-digit numbers from the database, resetting it back to empty.
11. Commit and push all changes to your forked repository **csd3156-lab03-2026.**

END OF DOCUMENT