Assignment 2 (30%)

Things you will learn

In this assignment, you will develop a more comprehensive event management Android application. This project will equip you with the skills to build such an application.

This app will enable users to create, edit, and manage their events. Users will be able to categorize their events. (Note: the relationship between category and events is one-to-many. i.e. one category can contain one or more events. But one event can only be part of one category).

1. You'll utilize the **Options Menu** to provide users with additional actions and settings.
2. A **Navigation View** will be implemented for clear and consistent in-app navigation.
3. To promote user interaction, you'll integrate **Floating Action Buttons (FABs)** for quick and accessible actions.
4. The application's core content will be structured using **Fragments**, a modular approach for managing different UI elements.
5. To display scrollable lists of data efficiently, you'll leverage **Recycler View**.
6. For persistent data storage, you'll implement **ArrayLists** and **SharedPreferences**, allowing the app to save and retrieve data even after it's closed.

By combining these elements, you'll create a well-rounded and user-friendly Android application.

Assignment 2 Specifications ⚙

Assignment 2 is due on Friday of Week 7, 11:55 PM. This is an individual assignment, submission via Moodle only. What to submit on Moodle?

Assignment 2 interview will be conducted in labs in week 8. Please note, a **"no interview no marks"** policy is applied across all assignments.

Generative AI tools cannot be used in this assessment task
*In this assessment, you must **not** use generative artificial intelligence (AI) to generate any materials or content in relation to the assessment task.*

As mentioned in the Assignment Theme section, this assignment is about developing an Events Management App (EMA) and building upon Assignment 1's output by adding the functionalities listed below under the tasks section.

Implementing Data Storage (Entities)

- As part of Assignment 2, entities must be implemented as **JAVA classes**.
- ArrayList will be used to save multiple events & event categories.
- ArrayList data will be saved/restored from **SharedPreferences**.
- *Hint: use GSON library to convert ArrayList to plain text and vice-versa. SharedPreferences only supports plain text, so to store ArrayLists you need to use a library like GSON.*
- No database implementation is required for Assignment 2, SharedPreference will be the data store for this assignment.

A) Event Category

"Yes" under the **Required** column means, the user must specify a valid value for the attribute.

B) Event

"Yes" under the **Required** column means, the user must specify a valid value for the attribute.

Tasks

Login & SignUp Activities remain unchanged. Once users successfully log in they will be redirected to Dashboard Activity as usual. From there on majority of assignment 2 functionalities will be added as per below.

1) Navigation drawer

1.1 Add a navigation drawer to the Dashboard Activity previously developed as part of assignment 1, that lists the following navigation menu items (1.2 to 1.5).

1.2 View All Categories: On click of this menu item, a new Activity "ListCategoryActivity" must load, see task 4.3.

1.3 Add Category: On click of this menu item, the user will navigate to the "New Event Category" activity, previously developed as part of A1, for changes required to "New Event Category" see task 5.2.

1.4 View All Events: On click of this menu item, a new Activity "ListEventActivity" must load, see task 4.4.

1.5 Logout: This option takes the user back to the login screen. You would need to ensure that the "back" button if pressed after logout should not allow navigation back to Dashboard Activity.

2) Toolbar & Options Menu

Add an options menu to the Dashboard Activity with the following menu items:

2.1 Refresh: On click of this button reload the including the list of categories in the Dashboard Activity.

2.2 Clear Event Form: this operation clears all the fields (EditTexts) in Dashboard.

2.3 Delete All Categories: this operation deletes all the saved categories in SharedPreferences. ***Note****: dont delete any other data (including events or login data).*

2.4 Delete All Events: this operation deletes all the saved events in SharedPreferences. ***Note****: dont delete any other data (including categories or login data).*

3) FAB - Floating Action Button

3.1 On Dashboard Activity, add a Floating Action button which allows event managers to quickly save an event.

3.2 Display a Snackbar with "UNDO" action, this will allow the user to undo their recent save

operation.

4) New Fragments & Activities

A `Fragment` represents a reusable portion of your app's UI. A fragment defines and manages its own layout, has its own lifecycle, and can handle its own input events. Fragments can't live on their own. They must be *hosted* by an activity or another fragment. The fragment's view hierarchy becomes part of, or *attaches to*, the host's view hierarchy. Read more.

In this assignment you will be creating two Fragments (4.1 and 4.2). Each of these fragments will be displayed in a separate screen (4.3 and 4.4). *Note: 4.3 (Fragment for viewing all categories) is also displayed on the Dashboard.*

4.1 FragmentListCategory

- Create a new Fragment named "FragmentListCategory", responsible for listing all categories using a RecyclerView.
- Each card of the RecyclerView must display all the attributes of a category (see entity table above)
- Code to read category data from SharedPreferences should be implemented/executed as part of the fragment

4.2 FragmentListEvent

- Create a new Fragment named "FragmentListEvent", responsible for listing all the events using a RecyclerView.
- Each card of the RecyclerView must display all the attributes of an event (see entity table above)
- Code to read category data from SharedPreferences should be implemented/executed as part of the fragment

4.3 ListCategoryActivity

4.3.1 If the user launches this activity using the navigation menu item (1.2), display "FragmentListCategory" within the layout of this activity.

4.3.2 Add a toolbar to this activity and implement a navigation feature so that the user can navigate 'back' to the previous activity (i.e. Dashboard)

4.4 ListEventActivity

4.4.1 If the user launches this activity using the navigation menu item (1.4), display "FragmentListEvent" within the layout of this activity.

4.4.2 Similar to 4.3.2, implement a toolbar to help with user navigation.

5) Events Dashboard

After the user logs in successfully, in your app they should be taken to a Dashboard Activity (as seen in Assignment 1). In this assignment, make the following changes. See expected output for what the UI should look like.

5.1 Load the "FragmentListCategory" fragment at the top of the Dashboard Activity, see expected output for more details.

5.2 Underneath the FragmentListCategory, show the form to add a new event as developed previously. However no "save" button is required, FAB will be replacing the save event button

5.3 On saving an event increment the "Event Count" attribute of the category specified by the user. e.g. If the user saves an event to the category "Social" with a current event count of 5, the new value for the event count will be 6 after a successful save operation.

5.4 If invalid data is specified while saving an event show an error Toast message

- "Invalid event name" if invalid event name examples: "**1111111**", "Melbourne **%** Game", etc
- "Invalid 'Tickets available'": this field can only contain 0 or positive numbers. Do not allow users to enter negative values. If user enters negative number, default to zero.
- "Category does not exist" if the user specified a non-existing category Id

6) New Event Category

Use the same Activity created as part of Assignment 1 to save a category and make the following changes.

- On successful save of category, the user should automatically redirect to Dashboard Activity
- If invalid data is specified while saving a category show an error Toast message
    - "Invalid category name": invalid category name examples: "1111111", "Melbourne % Centre", etc.
    - "Invalid Event Count": this field can only contain 0 or positive numbers. Do not allow users to enter negative values. If user enters negative number, default to zero.

Notes

- *Incoming SMS functionality for filling event forms is **optional** for this assignment (see Activity 5: New Event).*
- *If your app fails to build tutors will try to resolve bugs to an extent, and you will be penalised for build failures (minimum -2 or no marks for features due to which the app fails to build).*

Expected Output

Marking Rubric

Important Notes

Generative AI tools cannot be used in this assessment task
*In this assessment, you must **not** use generative artificial intelligence (AI) to generate any materials or content in relation to the assessment task.*

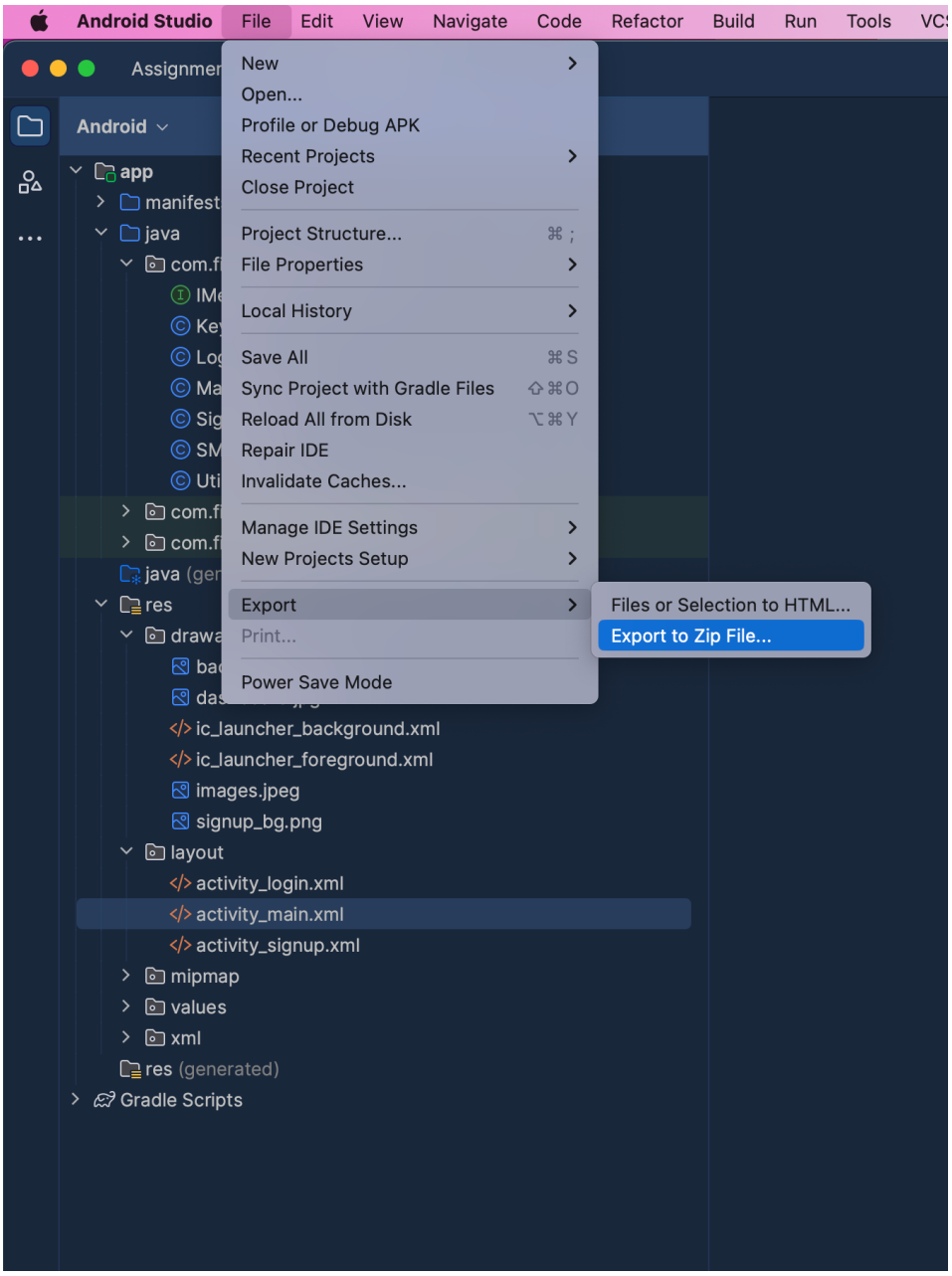
- No marks if you do not **submit your code** to Moodle
- Google Drive submissions will **NOT** be considered
- No marks if you miss attending your interview
- No marks for any sections that you cannot explain during the interview
- If you are unable to answer most of the interview questions, your submission might be escalated to Academic Misconduct

What to submit?

For Moodle, you must ZIP your files into a single file and upload it to the assignment-2 submission link that can be found on Moodle-->Assessments.

How to ZIP your Android project?

From Android Studio-->File-->Export-->Export to Zip FIle...



Save the exported file and upload it to Moodle.

What if my zipped file is larger than the allowed submission size on Moodle?

In case your output zip file is larger than 20MB, try the following:

- Make sure there are no external files accidentally placed inside your project folder.
- Try examining the content of the zipped file and identify which subfolder is resulting in a larger file size. In most cases, your application code will not exceed 1MB.
- What if still unable to figure out the issue? complete your work in advance and seek help during consultation or labs.
- Worst case scenario, you can just zip the "app" folder and submit it on Moodle.
- **Google Drive submissions will not be considered.**

If you are unable to submit on Moodle, due to file size, this will not be deemed eligible for special consideration.

Late Submission