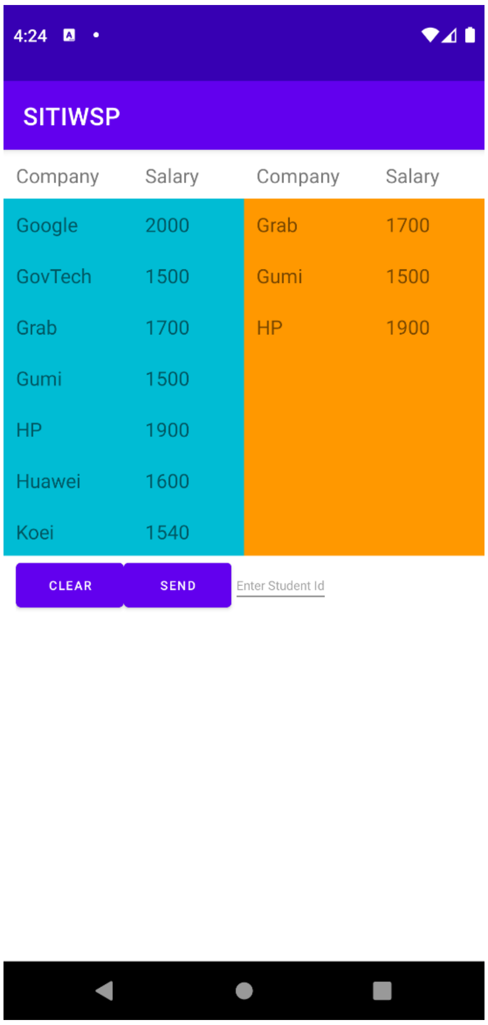
IWSP

CSC2007 Mobile Application Development Spring 2023

Fork the repo **csc2007-quiz02-2023** and download the job list in xsite. IWSP is on the horizon and the readytalent team wants you to help to develop a mobile app for IWSP job selection. Each student is required to choose 3 jobs and rank them in order of preference and submit.

Design the UI

This app has only one activity: MainActivity. Design the layout of the screen for the MainActivity to be similar to the following, the views need to be visible after rotation.



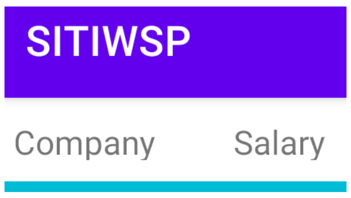
4:24

SITIWSP

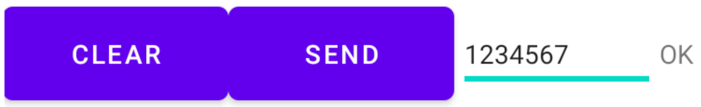
Company	Salary
Google	2000
GovTech	1500
Grab	1700
Gumi	1500
HP	1900
Huawei	1600
Koei	1540

Company	Salary
Grab	1700
Gumi	1500
HP	1900

- **TextView** has to be used to display the header texts: **Company**, **Salary** (ids: **headerCompany, headerSalary)**

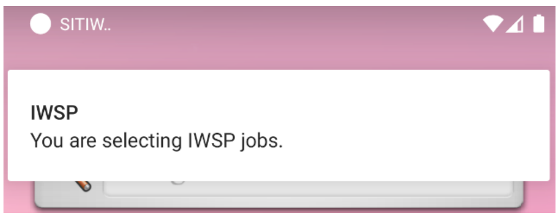


- A **RecyclerView** has to be used to display all the jobs (id: **recyclerviewleft) (**You may need to define a job_item.xml). Each job contains two fields: company and salary. Clicking on a job indicates one preference. It occupies about 50% of the screen in height and width. The job list is given to you in the **data.txt** in xsite, please ensure its default order, when the app is firstly launched, you can choose to hardcode the jobs.
    - **TextView** has to be used to display the Company (id: **jobCompany**) and Salary (id: **jobSalary**).
- Another **RecyclerView** has to be used to display all the jobs selected (id: **recyclerviewright) (**You may need to reuse the job_item.xml). Each job contains two fields, company and salary which indicates your preference. It occupies about 50% of the screen in height and width. It is empty by default when firstly launched.
    - **TextView** has to be used to display the Company and Salary. (ids: **jobCompany**, **jobSalary**)
    - Use a Preferences DataStore with name "**IWSPOptions**" (**Context.IWSPOptions)**, to store these options.
- **Button** has to be used to display **Clear** and **Send** (ids: **selectionClear**, **selectionSend**).
- **EditText** has to be used for Student input (id: **studentId**).
- **TextView** has to be used to display error/send message (id: **errorMsg**), the error/send messages are: **Invalid Id, Invalid Choices, OK**

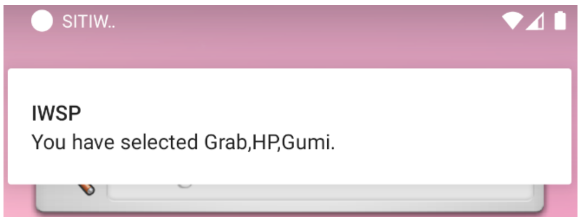


- If the app is at background (i.e., the back button pressed), a foreground service with a notification will be started to indicate you are away from selection jobs, the title of the notification will be **IWSP**. When it is started, its content text will be "**You**

are selecting IWSP jobs.”, every 10 seconds, the notification pops up, and the notification texts will be “**You have selected x,y,z.**”, where **x,y,z** are the company names the selected, if no jobs are selected, the notification texts will be “**You haven't selected any jobs.**”


SITIW..

IWSP
You are selecting IWSP jobs.


SITIW..

IWSP
You have selected Grab,HP,Gumi.

IMPORTANT: Ensure there are no spelling errors on the texts and buttons.

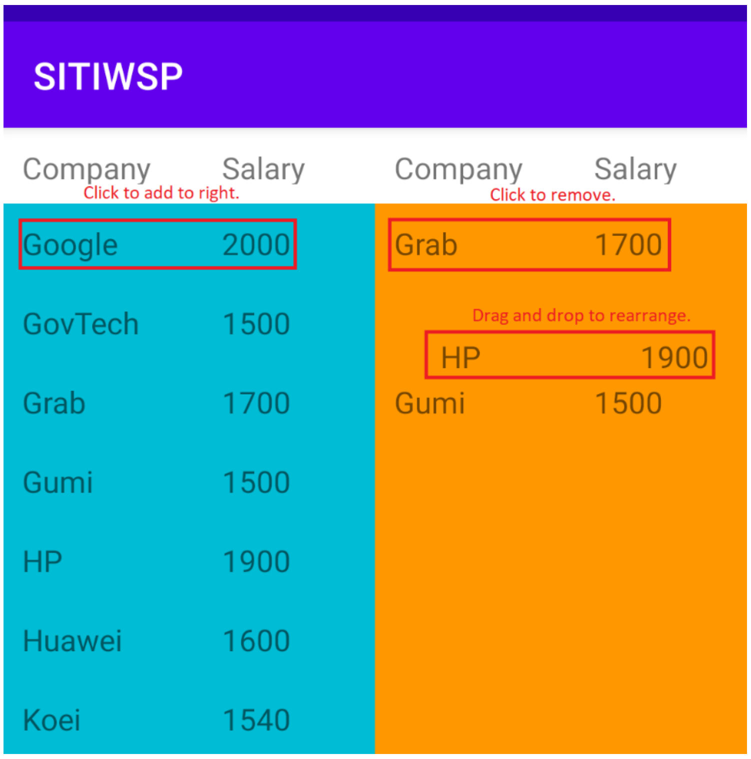
IMPORTANT: Ensure the above IDs within the XML.

IMPORTANT: Ensure the above views are visible after rotation.

Implement the functionality

1. Select 3 job preferences

- By clicking on the job in the **recyclerviewleft**, this job will be selected and added to the **recyclerviewright** (adding from bottom, these are your selections). The **recyclerviewleft** remains no change. The maximum number of selections (items in **recyclerviewright**) is 3, further or duplicated selections from **recyclerviewleft** will be ignored.
- By clicking on one selected job in **recyclerviewright**, this job will be removed from **recyclerviewright.** The **recyclerviewleft** remains no change.
- By dragging and dropping the jobs in **recyclerviewright**, the jobs can be re-arranged. The top job indicates the first option.
- You can use ItemTouchHelper.SimpleCallback ~ onMove(), and attach it to the adapter to handle drag and drop. https://developer.android.com/reference/kotlin/androidx/recyclerview/widget/ItemTouchHelper.SimpleCallback



Company	Salary
Google	2000
GovTech	1500
Grab	1700
Gumi	1500
HP	1900
Huawei	1600
Koei	1540

Company	Salary
Grab	1700
HP	1900
Gumi	1500

2. Sort the job list

- If the app is firstly launched the job list in the **default** order has to be displayed
- If the header text Company (id: **headerCompany**), is firstly hit, the job list will be sorted in **ascending** alphabetical order, if it is hit again, the job list will be sorted **descending** alphabetical order, each hit will switch the sorting order.
- Note that, your previous job choices and their order (in **recyclerviewright**) need to be maintained after sorting.

3. Clear the job choices

- Clear button will clear your selections in **recyclerviewright**

4. Send you job choices

Hitting send button will send your choices with your student id.

Firstly, the student id will be verified, if the student id is invalid, the error message **Invalid Id** will be displayed in the TextView (id: **errorMsg**). Implement the logic to check for valid student id format:

- **Exactly 7 numerical digits**

Secondly, your job choices will be verified, if not all 3 choices are indicated, the error message **Invalid Choices** will be displayed in the TextView (id: **errorMsg**). Otherwise, **OK** will be displayed in the TextView (id: **errorMsg**).

5. Rotation and app relaunch

If your phone is **rotated** or the app is **relaunched**:

- All the job choices and their order in **recyclerviewright** need to be maintained.
- The job list **sorting order in** in **recyclerviewleft** needs to be maintained.

Use ViewModel and Preferences DataStore to implement these.

6. Foreground service

If the app is at background (i.e., the back button pressed), the foreground service (title: **IWSP**) with a notification will be **started**. This service will be **destroyed** (no more notification) when the app resumes to be at foreground by clicking on the notification or relaunching. The service will be **destroyed** when the app is destroyed.

Lab Quiz 2

(14 Marks)

1. Fork the repo **csc2007-quiz02-2023** and download the job list in xsite.
2. Design the layouts the screen similar to the given screenshots. The views need to be visible after rotation. (1 Mark)
3. Implement the logic to check for the validity of studentId for Send button. (1 Mark)
4. Implement a RecyclerView **(recyclerviewleft)** to display the job list. (1 Mark)
5. Implement selecting job on **recyclerviewleft** and implement a RecyclerView (**recyclerviewright**) to display the selected job list. (1 Mark)
6. Implement the logic to check for the validity of job choices for Send button. (1 Mark)
7. Implement item delete (by hitting the item) in **recyclerviewright**. (1 Mark)
8. Implement item drag and drop to re-arrange selections in **recyclerviewright**. (1 Mark)
9. Implement Clear button logic to reset the job selections in **recyclerviewright**. (1 Mark)

10. Implement the logic to maintain job selections after sorting job list when hitting the header text Company as well as maintain job selections and job sorting order and after phone rotation. (1 Mark)
11. Implement the logic to maintain job selections and sorting order after the app is relaunched. (1 Mark)
12. Implement the logic to start the foreground service. (1 Mark)
13. The foreground service behaves correctly according to the activity's lifecycle. (1 Mark)
14. Remember to comment and indent the code and implement it in a modular fashion, using different methods or classes if appropriate. Ensure it conforms to Kotlin coding conventions. (2 Marks)
15. Commit and push all changes to your forked repository **csc2007-quiz02 -2023.**

IMPORTANT: Do not change the activity names or package name. Ensure the job list follows its default order when firstly launched. Ensure there are no spelling errors on all the text. Remember to commit and push to repo!

END OF DOCUMENT