CST2335 Graphical Interface Programming - Draft

# Final Project Assignment

Although this is a group project, you must only work on your part of the project that you have selected. You must write all of your own code that makes up your part of the project. If you use code that someone else, including your teammates, then that is considered plagiarism and you will get a mark of 0 for the final project.

## Purpose:

The Project is assigned to give you experience in:

* Developing software in a group environment, including using Github to merge code into 1 project.
* Dividing workload to meet deadlines.
* Designing modular software that allows for that division of work.

## Part 1 – Choosing your team

This is a group project for groups of 4 members and can be from any lab section in this course. If you would like to be assigned to a group, then send an email to your lab instructor. If you have chosen partners yourself then please email the names of the group to your lab instructor. Once you have a team for the final project, each of you should select one (and only one per person) of the given topics. You should then fill out the Group Activity Worksheet together to exchange contact information amongst your group (Algonquin student email addresses at a minimum). You must then upload the Excel worksheet on Brightspace, under the FinalProject assignment link on or before ***November 6th, 2022. This is worth 5% of your final project mark***

## Part 2 – Programming your application. Requirements:

Once you have chosen your topic, you must implement this list of the requirements for the final project:

1. Each person’s project must have a RecyclerView somewhere to present items in a list. Selecting an item from the RecyclerView must show detailed information about the item selected.
2. Each person’s project must use a fragment somewhere in its graphical interface to show details about a selected item from the RecyclerView.
3. Each person’s project must use a database to store items. The user must be able to add and delete items, which should be displayed somehow in a RecyclerView. It should work similarly to the chat room labs you did this semester.
4. Each activity must have at least 1 Toast, 1 Snackbar, and 1 AlertDialog notification.
5. Each activity must have at least 1 edit text with appropriate text input method and at one button.
6. Each activity must use SharedPreferences to save something about what was typed in the EditText for use the next time the application is launched.
7. The entire project must have at least 1 or more activities written by each person in your group (so at least 1 activity per person). Your activity must be accessible by selecting a graphical icon from a Toolbar, and NavigationDrawer. The top navigation layout should have the Activity’s title, author, and version number.
8. Each person’s project must have a help menu item that displays an AlertDialog with instructions for how to use the interface.
9. There must be at least 1 other language supported by your part of the project. If you are not bilingual, then you must support both British and American English (words like colour, color, neighbour, neighbor, etc). If you know a language other than English, then you can support that language in your application and don’t need to support American English.
10. Each activity must use Volley to retrieve data from an http server. You cannot use Executor or AsyncTask.
11. All activities must be integrated into a single working application, on a single device or emulator. You should use GitHub for merging your code by creating pull requests.
12. The interfaces must look professional, with GUI elements properly laid out and aligned.
13. The functions and variables you write must be properly documented using JavaDoc comments. You must create the JavaDocs in a JavaDocs folder like you did in the labs.

***Milestones:***

Bonus marks will be awarded for demonstrating correct functionality during your lab period by the following dates:

|  |  |  |
| --- | --- | --- |
| Milestone # and date | Requirements implemented # | Bonus Marks available |
| #1 – Nov. 14 – 18 (create GUI) | 1, 4, 5, 6, 11, 12 | 1 |
| #2 – Nov. 21 – 25 (load data) | 7, 8, 9, 10, 11, 12 | 1 |
| #3 – Nov 28 - Dec. 2 (save data) | 2, 3, 11, 12, 13 | 1 |

## Beginning Steps

* One person should create a new project for the team and then upload it to github using the menu option “VCS” -> “Import Into Version Control” -> “Share project on GitHub”.
* That group member must then invite the other group members to contribute. This is done by clicking on the “Settings” tab in Github, then click “Collaborators” on the left side menu, and search the group member names to add them to the project. Other team members should then clone that project to their computer and start making branches for their work. From AndroidStudio, select “File” -> “New” -> “Project from version control” -> “Git” and then paste the git URL from the main github repository from the previous step. ***You will not be able to integrate your work if you do not start by first cloning the project!***
* Then write your own code on your own branch and then merge that branch on Github (after each requirement is finished). Don’t try to merge the code only on the last week.

## Grading Guide

* Each student is graded on his or her application separately. ***This counts for 85% of your project mark.***
* Week of Nov. 14 – 18 – demonstrate the Milestone 1 requirements in your lab period for bonus
* Week of Nov. 21 – 25 – demonstrate the Milestone 2 requirements in your lab period for bonus
* Week of Nov. 28 – 2 – demonstrate the Milestone 3 requirements in your lab period for bonus
* Week of Dec. 5 – 9 - Project demonstration during your scheduled lab demonstration found on the demonstration schedule. You will show each of the 13 requirements from the list. Arrange a single submission of the group deliverable by one of the group member’s computer on behalf of the entire group. You must be in the lab in person to answer questions about your work. Code submitted on Brightspace will not be marked.

## Part 3 – Submit your source code on Brightspace

* Before your group demonstration, each member of the group must submit their final code as a record of what was finished at the end of the project. From your github repository, there is a link for “Clone or Download”. Select the Download option and save your code as a zip file on your computer. Then upload that zip file to Brightspace using the FinalProject link. ***This is worth 10% of your final project mark.***

## The Application Topics

Each of the applications (as they are intended) requires similar programming techniques. Each application takes information from the user and stores it in a database. They can then view the data saved to a list of favourites and delete items from that list. Beyond that you are free to get creative.

**Pexels**

* Your application should have an EditText for entering a search term. There should also be a “search” button which sends the search term to the server and returns a list of photos that matches that term.
* The URL for searching is “[https://api.pexels.com/v1/search?query=](https://api.pexels.com/v1/search?query=https://api.pexels.com/v1/search?query=) ” followed by the search term. This API requires that you send the API key in an “Authorization” header before you open your URL connection: ("Authorization", YourApiKey );
* Your application must show a list of results in a RecyclerView, with each row showing a thumbnail of the picture and the photographer’s name.
* If the user selects a picture from the list, your application will load a fragment showing the original picture, the picture height and width, and URL to the image. If the user clicks on the URL, your application should launch the URL in a web browser.
* Also in the detail fragment, the user can save the original picture to the device for offline viewing. The user must be able to view a list of their saved pictures and remove them from the database of favourites if they choose.
* The SharedPreferences should save the user’s search term so that the next time you start the application, the previous search term is shown.
* The documentation on Pexels can be found here: <https://www.pexels.com/api/documentation/>, and you will have to sign up for a free API key. It is limited to 200 queries per hour, or an average of 3 per minute.

**Movie Information**

* This application tracks information about movies. The user can search for a movie title and get a list of movies back that match the title. Use: <http://www.omdbapi.com/?apikey=6c9862c2> and add “&t=” followed by the movie title to search for. You will need to use the function URLEncoder.encode( aString, "UTF-8") to change the title to URL encoded strings. URLs can’t have spaces in them so the encode() function will change “The Matrix” to “The+Matrix”.
* The information stored includes movie title, year, rating, runtime, main actors, plot, and URL of the movie poster.
* The Movie poster should be saved locally to the device for later viewing.
* The user can save a movie description to the device for later viewing, or can delete a saved movie.
* The saved movies should appear in a list where the user can choose to view details on a details page.

**TicketMaster event search**

* The user can search for upcoming events near a given city. The user can enter the city name, and a search radius for the events. The server will return a list of events scheduled in that city. Your program should show the list of the names of the events. Clicking on an event name should show the starting date, the price range of tickets, the URL from ticketmaster, and the promotional image. The user should also be able to save the event’s details in a database of saved events.
* The SharedPreferences should save the name of the last city that was searched. It should display that city name the next time the application is launched.
* There should be an option to view the list of saved events. Clicking on the name of an event should show the same information as from the server. However this time there should be a button to delete the city from the list of favourites.
* You should sign up for an API key from this website: https://developer-acct.ticketmaster.com/user/register. The API looks like: <https://app.ticketmaster.com/discovery/v2/events.json?apikey=XXXXXX&city=YYYYY&radius=100>
* You would replace the XXXX with your API key that you signed up for, city is the city where the events will be taking place, and the radius is how far from the city the event is taking place to be included.

**Soccer match highlights**

* Retrieve a list of soccer matches that were recently played from: <https://www.scorebat.com/video-api/v1/>
* You should show a list of titles of the games. When a user selects a game from the list, you should show a details page with the date of the match, the two teams that are playing.
* There should be a button that allows the user to go to the page listed under the URL field for viewing the match highlights. For this, you can either just use an intent to view the URL on another page, or there could be a VideoView embedded right on the page. The source URL is found under the “videos” array, under “embed”, and then look for the string after ***src=***.
* The user can save the details of the match in a database so the user can retrieve it later. There should be a button to show the list of favourite matches.
* If the user selects one of the favourite matches, there should be a details page which shows the details about that match. There should also be a delete button to remove the match from the list of favourite matches.
* For SharedPreferences, if the user selects to watch a video in a web browser, then you should save that URL as a shared preference. The next time when you load the page, check to see if there is a URL in the shared preference (if the string length != 0). If there is a URL saved, then launch a browser immediately to go to that page but make sure to use startActivityForResult so that it will call onActivityResult() when the user returns. In onActivityResult() , set the saved URL in the SharedPreferences to an empty string: “” so that the next time, your application won’t go directly to the video.