CST2335 Graphical Interface Programming

# Final Project Assignment

## 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 ***July 9th, 2021. 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 an AsyncTask to retrieve data from an http server, along with a progress bar to show that there is something downloading.
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 – July 19 – 23 (create GUI) | 1, 4, 5, 6, 11, 12 | 1 |
| #2 – July 26 – 30 (load data) | 7, 8, 9, 10, 11, 12 | 1 |
| #3 – August 2 – 6 (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 July 19 – 23 – demonstrate the Milestone 1 requirements in your lab period for bonus
* Week of July 26 – 30 – demonstrate the Milestone 2 requirements in your lab period for bonus
* Week of August 2 – 6 – demonstrate the Milestone 3 requirements in your lab period for bonus
* Week of August 9 – 13 - 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 members 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.

**OCTranspo Bus Route App**

* The user can query the OCTranspo web servers for bus route information. The previously queried bus routes should be displayed in a listview where the user can select. The can also add, or remove bus stop numbers which they want to get information about, which gets stored in a database. The bus stop number for Algonquin College (Baseline) is #3017. You can see other station numbers here: <https://www.octranspo.com/en/plan-your-trip/travel-tools/bus-stop-number-list#560Stations>
* Connect to the web server to retrieve the bus routes through that station number and display it in a list.
* The user can then select a bus route from the list, and view:
  + The trip destination
  + The bus’s latitude and longitude
  + The bus’s gps speed
  + The trip start time
  + The adjustedScheduleTime (how late it is)
* The api documentation is hosted here: <http://www.octranspo.com/developers/documentation> and you will need to use the functions: ***GetRouteSummaryForStop***, and ***GetNextTripsForStop***. Use the appId: 223eb5c3 and applicationID: ab27db5b435b8c8819ffb8095328e775. For example, getting the route summary for stop #3017 would be:

<https://api.octranspo1.com/v2.0/GetRouteSummaryForStop?appID=223eb5c3&&apiKey=ab27db5b435b8c8819ffb8095328e775&stopNo=3050>&format=xml . The next trips for the bus #95 from stop #3017 would be:

<https://api.octranspo1.com/v2.0/GetNextTripsForStop?appID=223eb5c3&&apiKey=ab27db5b435b8c8819ffb8095328e775&stopNo=3017&routeNo=95&format=xml>

**Electric Car Charging Station finder**

The user can search for electric car charging stations within a given area. The documentation is posted here: <https://openchargemap.org/site/develop/api?ref=public-apis>

* The user can input a latitude & longitude location, and the server returns a list of charging stations near that location. For example, this URL returns locations near Algonquin College: https://api.openchargemap.io/v3/poi/?output=xml&countrycode=CA&latitude=45.347571&longitude=-75.756140&maxresults=10
* The results should be loaded in a list of results. The user can then select an item from the list and see details about that charging station. You should display the stations:
  + Location Title
  + Latitude
  + Longitude
  + Contact Telephone number
  + A “load directions in google maps” button which loads the latitude and longitude in the google maps activity. Here are instructions how to do that: <https://developers.google.com/maps/documentation/urls/android-intents>
* The user should be able to save a charging station into a list of favourites, saved in a database. This list should be accessible from a menu item. The user should be able to remove items from the list and database.
* Your application should save the last item searched to display the next time the application is launched.

**Movie Information**

* This application tracks information about movies. The user can search for movie titles and get information back. Use: http://www.omdbapi.com/?apikey=6c9862c2&r=xml to query information. You should 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.

**Soccer games api**

* The user can download a list of soccer news articles from <http://www.goal.com/en/feeds/news?fmt=rss>. This will have a list of <item> tags which have a title, date, and a thumbnail image. These news articles should be shown in a list on the left hand side. Clicking on a title should show the image, date, news article URL and description text. There should be a button to save the article to a favourites database, and another button to load the article’s URL in a browser.
* There should be a button to show the saved news articles in a list, like above. When you select a saved news article, it shows the same details as above, the instead of a “save” button, it is replaced with a “remove from favourites” button which removes the story from the database.
* When you first start your application, there should be a dialog box which asks to rate the application using 5 stars. You should save the rating in SharedPreferences to show the next time you start the application.