Moods – App for Champlain Students

By

Amir Osman

Étienne Bérubé

William Perron-Lafleur

Presented to

Mr. Amin Ranj Bar

420-204-RE, Integrative Project in Computer Science and Mathematics, section 00871

Tuesday, April 23th, 2017

Champlain College Saint-Lambert

**Table of Contents**

**Written Report**

**Project Overview**…...…………………………………………………………………3

**Project Design**………..………………………………………………………………..5

**Analysis**…………………………………………………………………………...5

**Algorithm**…………………………………………………………………………6

**Explaning Design**…………………………………………………………………8

**UML Diagrams**………………………………..………………………………...16

**Documentation**………………………………………………………………….17

**Graphical User Interface**……………...……………………………………….17

**Methods of Evaluation**……………………………………..………………………18

**Results: System Quality**……………………………………..……………………..19

**Project Management**……………………………………………………………….20

**Timeline**………………………………………………………………………….20

**Features**……………………………………………………………..…………..22

**Software**…………………………………………………………………………21

**Conclusion**…………………………………………………………………………….23

**Manuals**

**System Manual**…...……………………………………………………………………3

**Installation Instructions**…………………………………………………………9

**User Documentation**………..………………………………………………………..5

**Procedures help section**………………………………………………………….5

**Information Reference Guide**…………………………………………………...8

**Tutorial**…………………………………………………………………………...9

**Appendix A**

**Documentation**……………………………………………………………………………….1



**Written Report**

**Project Overview**

As our final project we want to create a social network on android phones exclusively dedicated for Champlain students. Communication has always been something of paramount importance in the everyday life. During our college experience, we have noticed a lack of utilities to facilitate contacting the persons we care about. There is no way for students to get instant information about their college peers. Therefore, students who are looking to connect with other students are unable to do so to their liking. Considering students spend a lot of time at school, it is undoubted that such application would be extremely relevant and helpful in order for their life at school to be at its best state.

Additionally, we have also noticed that education is an unexploited domain in the programming world, even though education is essentially a domain everyone is involved at some point in their life. Hence, our goal is to make everyone’s experience at schools the best possible. We want people to have an optimal educative experience by providing them with a service that will allow them to facilitate their communication at school. Incidentally, what makes Moods special is that not only will it target the educative area, but also entertainment. Moods will offer students tools to help people study, but ultimately it will allow students to hang out together. Students wanting to eat for instance, will be able to connect with other students that are hungry. Students wanting to play outdoor soccer, will be able to organize an event. Students wanting to relax, will be able to reserve a room to lay back in. Moods has the opportunity to be successful because it provides simple solution to problems students meet at school. Our goal is not to provide something complex to students, as simplicity is complementary to efficacy. This is why each user is going to be able to reach out to other students within a click. We want students to be able to meet their friends and new people in the College more easily by using our application.

The lack of communication between students at school is essentially due to the fact that schools are too big. It is often difficult for people to know where their friends are, what they are doing, who they are with, what’s their schedule, etc. Therefore, the simple idea of meeting up with a friend is difficult, since you absolutely need to have their phone number and to have them use it exactly when you attempt to communicate with them. On the other hand, Moods would solve that problem by allowing students to publish their location and status/mood so that people don’t waste time anymore asking basic questions such as ‘Where are you?’ ‘What are you doing?’ to every single one of their friends. Instantly, the user will be able to see which students are available, what they are doing, and at what location they are (cafeteria, library, computer lab, etc.).

**Project Design**

**Analyze, define, and understand the problem**

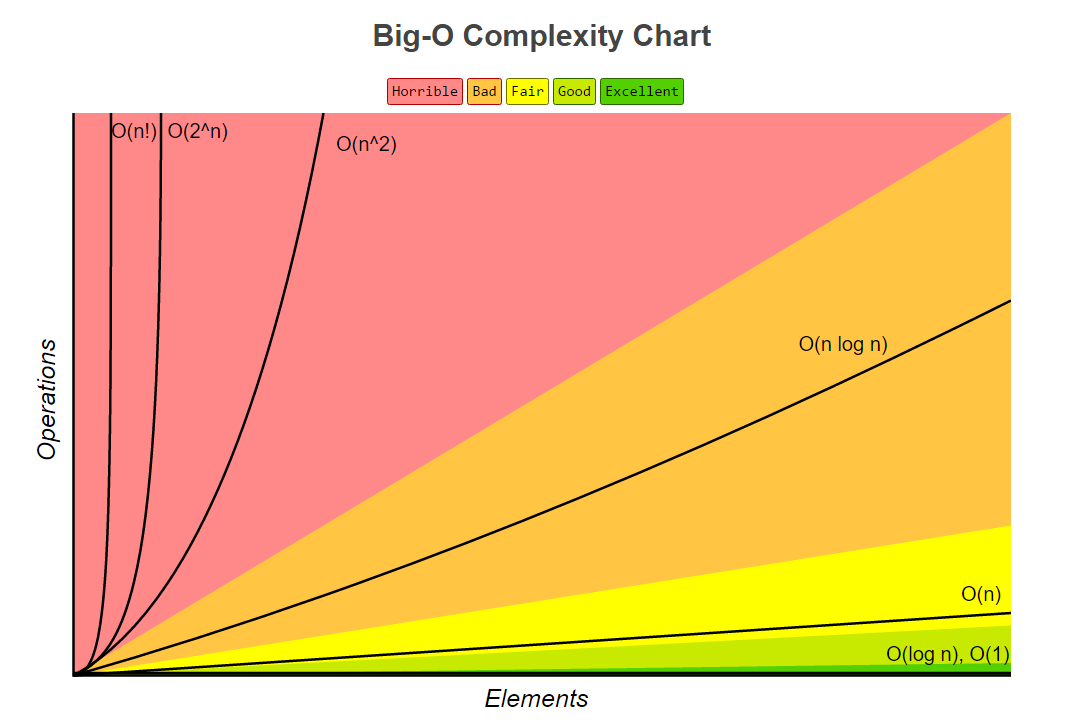
The major problem is that students are having trouble connecting quickly with their friends within their college. The main reason why is that they have no way of consulting their peers’ schedule, thus it is extremely inconvenient as students don't know when they have common breaks with their friends. This prohibits the students from doing things they would prefer doing together, such as studying, eating, learning, relaxing, and the list goes on.

Moods, our application, is the perfect solution to that problem as one of its features involves storing every student’s breaks in the database. Using an efficient mathematical algorithm, our tool shows the current status of each student, telling to other students if they are in break or not and if not, when their next break is. In other words, our users are able to know who's available when they are to do anything they want, easily. In order to display what is that thing users wants to do, users are provided a tool to update their mood, thus their friends know what they are up to.

Additionally, colleges are important in size, thus locating friends can most of the times be time-consuming for students. Therefore, our application has a remedy to that problem by providing an option to share the user’s location with friends, so that they know exactly where the user is when they need, in the matter of seconds.

Again, there is an obvious need to simplify the students' social life at college by helping them connect together easily and quickly.

**Algorithm**

The mathematic is behind the scene. The application by itself is not based on any kind of science. The application is a social media, hence does not have a scientific purpose. However, a lot of mathematics is used since the program has to deal with a lot of lists and arrays. For example, the search algorithm for finding profiles is a linear search. This means that the average efficiency of the algorithm has a Big O notation of n. The efficiency is O(n). This means that the same amount of comparison is equal to the number of elements in the array itself. For our app, we do not need a better search system since we do not have a lot of users. As explained by the chart below, the time complexity for the linear search is in the fair tier. Thus, for a big number of user profiles, it is a good algorithm for this given situation.

Moreover, this application also requires sorting of profiles in order to return an adequate list of friends, profile searched and notifications. It is Firebase that takes care of keeping track of a good sorting. a normal merge sort.

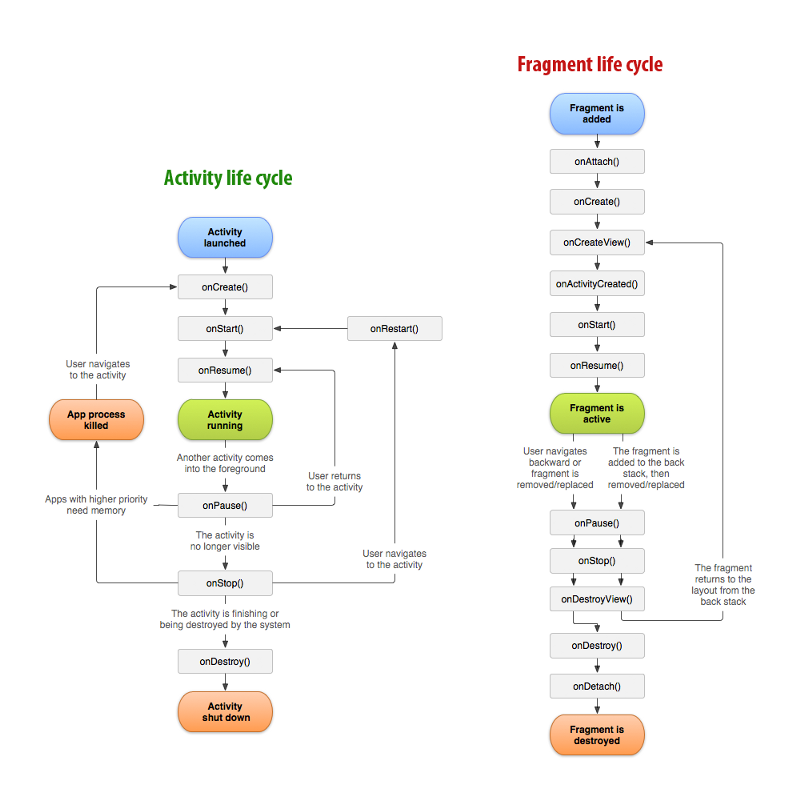
**Explaning Design**

The design of this project lies on two different key aspect. The first one is the direct JAVA code responsible for all of the application’s behavior and visual aspect. The second one is the database that incorporate an online component to this project in order to store the user’s information on a server instead of each and every device. The first part of the design will be focused on the JAVA local code and the second part will look at the database and its behavior.

 The first thing to understand when dealing with android application is the core components of an application and how they interact with the Android environment and the JVM (Java Virtual Machine). Comparatively to a “Scene” in “JavaFx” or similar environments, Android uses an “Activity” in order for the user to see components of the app. Each page that the user will interact with is or extends an Activity. The super-class Activity is provided by Google and comes with specific methods that needs to be overridden in order for the android environment to communicate with the specified page. This instance of Activity follows a rigid lifecycle. At each point of this cycle, a method is called in order to achieve certain actions. Therefore, the class Activity can be extended in order to gain access to such methods. Using those, a given Activity can perform custom tasks.

Image 1 Activity’s lifecycle

In this application, every page the user will interact with is an instance of “AppCompatActivity”. Its behavior is similar to a typical Activity, however it contains certain fields and method in order to accommodate its code to older devices or android versions. In UML diagrams, the fact that every page extends this specific class have been omitted for clarity. Otherwise every UML class would be connected to this super-class and it would then be difficult to read the other relationships from other components of this project.

Another type of class is used in this project. Indeed, “Fragments” have also been used in this project. The purpose of a Fragment is to run alongside an activity, while providing a more dynamic behavior. For example, tabs in a “TabActivity” are instances of Fragments. They have a different, yet similar, lifecycle than an Activity (Image 2). Thus, interacting with them is noticeably more complex. Since they attach to an Activity, sharing the same “Context” object, it can become tricky to make it do what one might really want to do with it. Most of the methods on android require the Context as a parameter. Therefore, one must ensure that each of the fragments is properly attached. Also, when one wants to access a field of the parent Activity, within an attached Fragment, it becomes difficult to actually get the desired results.

Moreover, the way of interacting inside a Fragment is completely different from a typical Activity. This lead to hours of frustration an incomprehension as one tried to properly understand the meaning behind such behavior. For instance, in an Activity, the visual layout is set with simple method in the onCreate() method. However, the layout is inflated in a Fragment, which requires a completely different approach. Since this method sends an object of type View when inflating, one must keep this view as a field in the current class in order to be able to interact with the desired visual element. On the other hand, the fact that the lifecycle is different and more complex gives a broader range of action to the developer in order to achieve certain actions. Since it is a more dynamic object, it gives the ability to program certain actions in very particular context. For example, when a tab is changed, what should the Fragment do? This is something that does not have to be dealt with while working with Activities.

Image 2 Fragment lifecycle

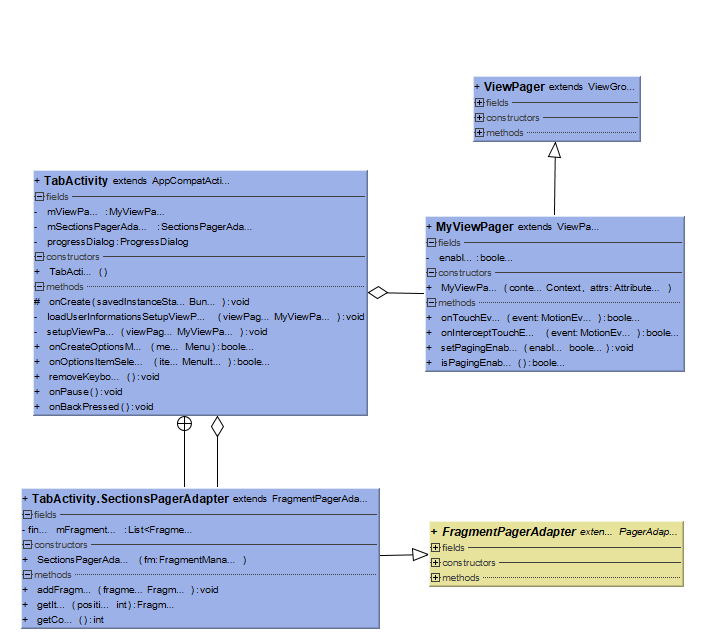
**UML Diagrams**

The UML diagrams have been separated in distinct categories for clarity. Since each class have different super-classes and properties, they will be regrouped in different blocks. Thus, the Fragments and their respective Adapters and ViewHolders will be grouped together, the Database models will share the same section and the Activities will also have their own part. It is also important to realise that Fragments and Activities do not have any kind of relationship together. They exist in the same application but rarely interact with each other. As explained in the previous part, some of the super-classes have been omitted for a reason of clarity. Only the important relationships are shown.

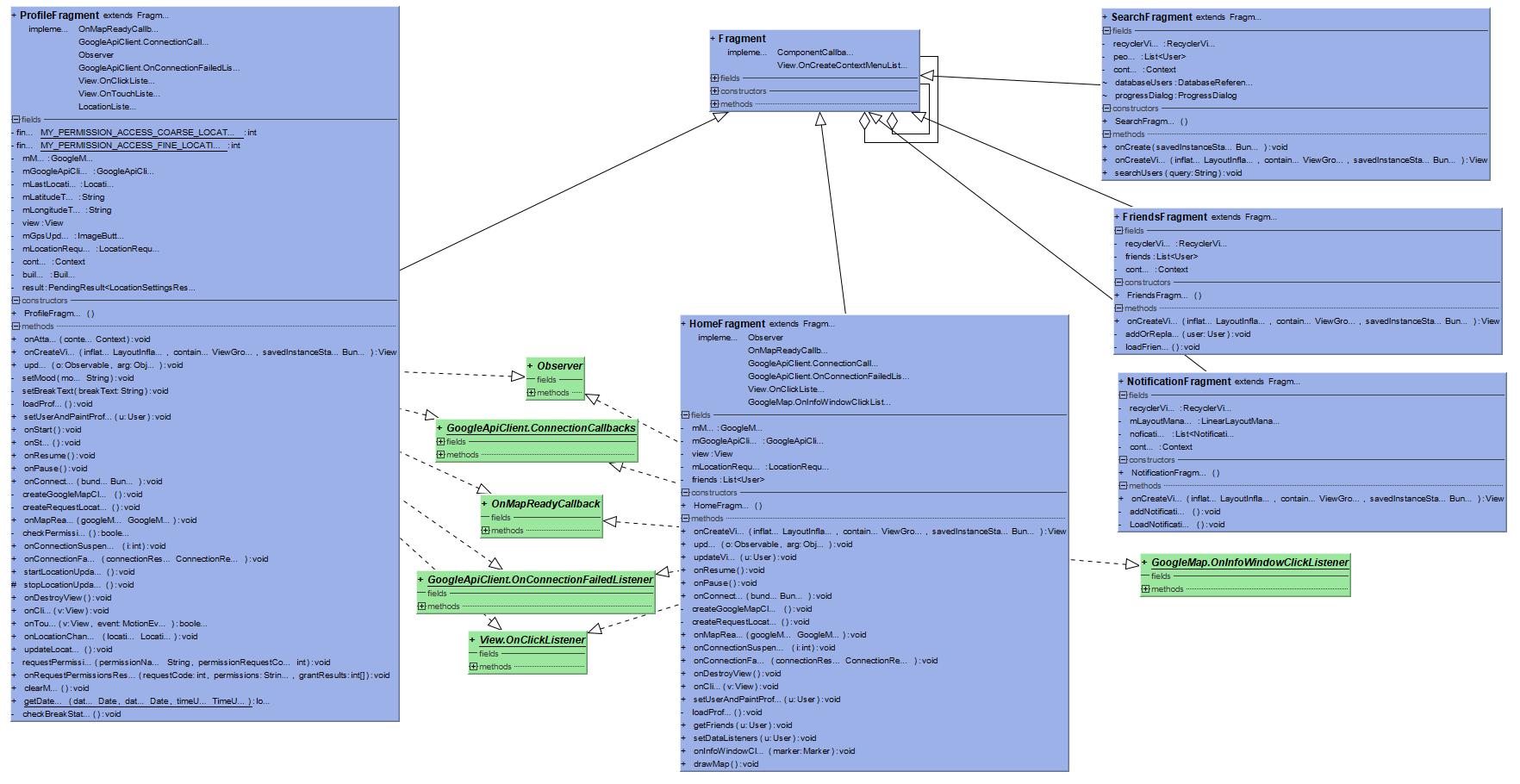
Activities:

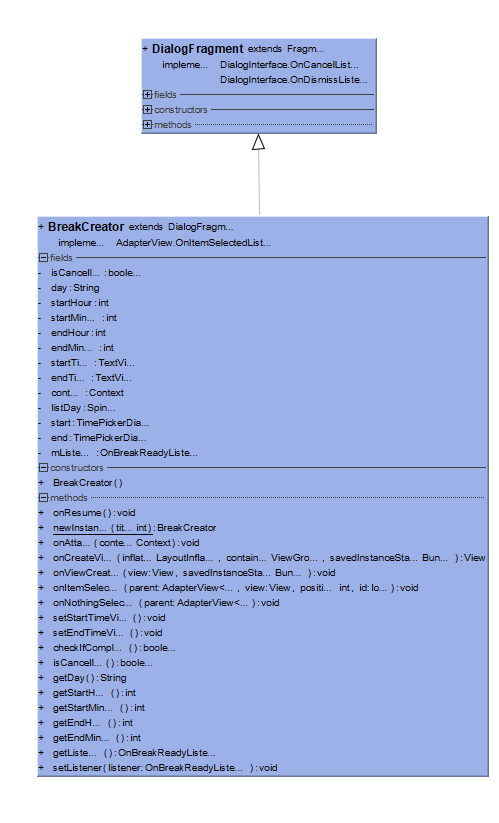
Note: There is no relationship between them.

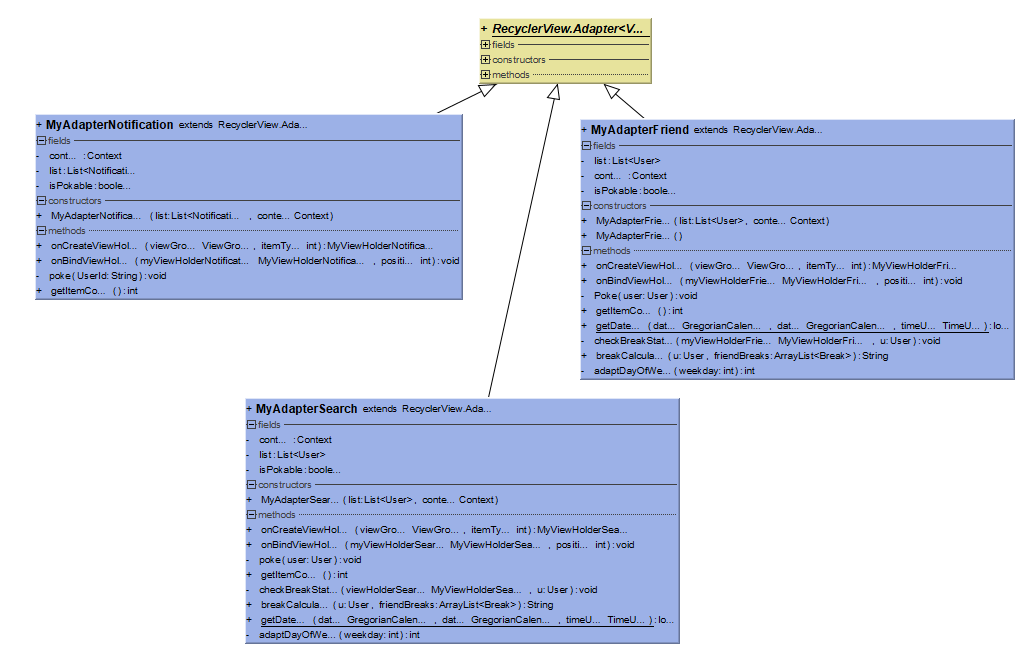


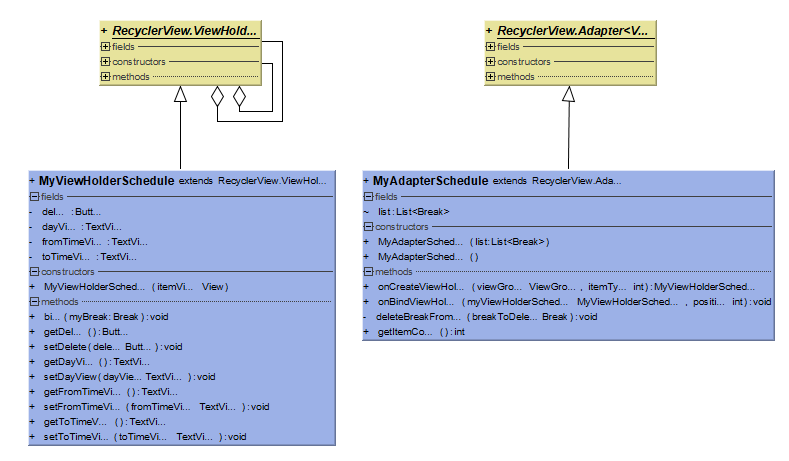


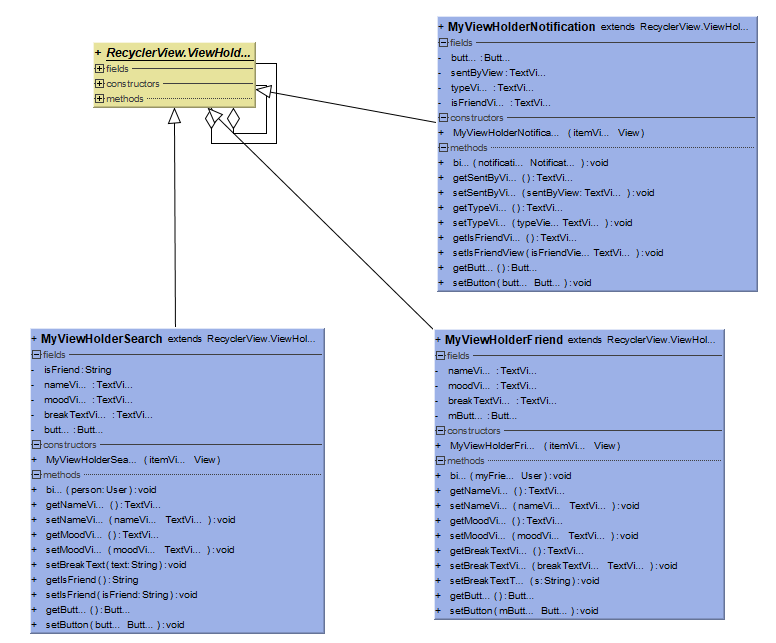
Fragments:

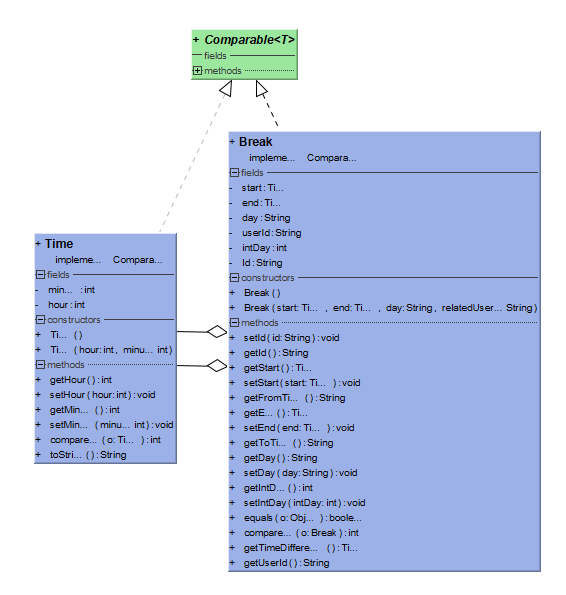


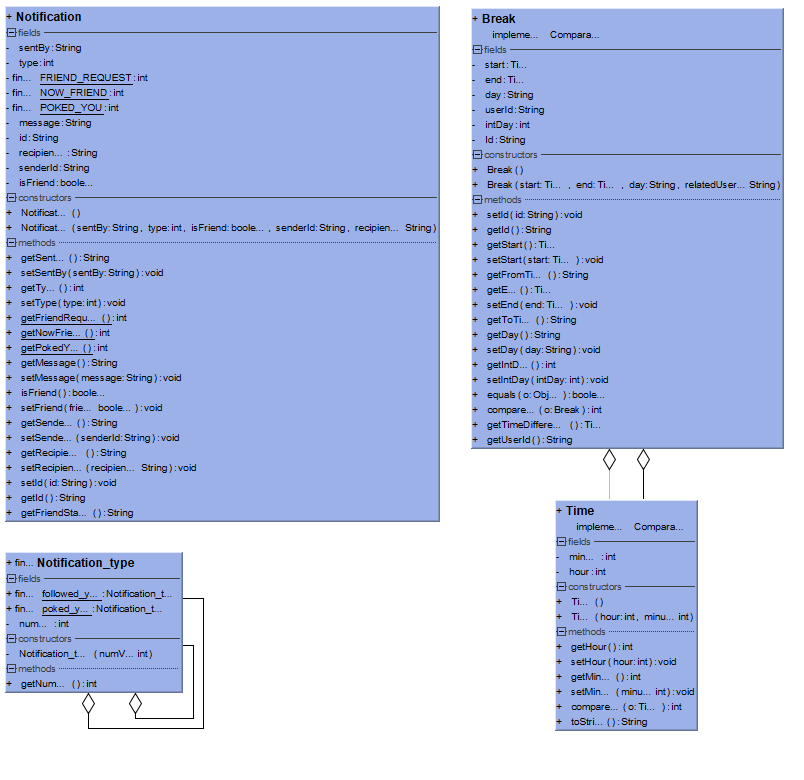


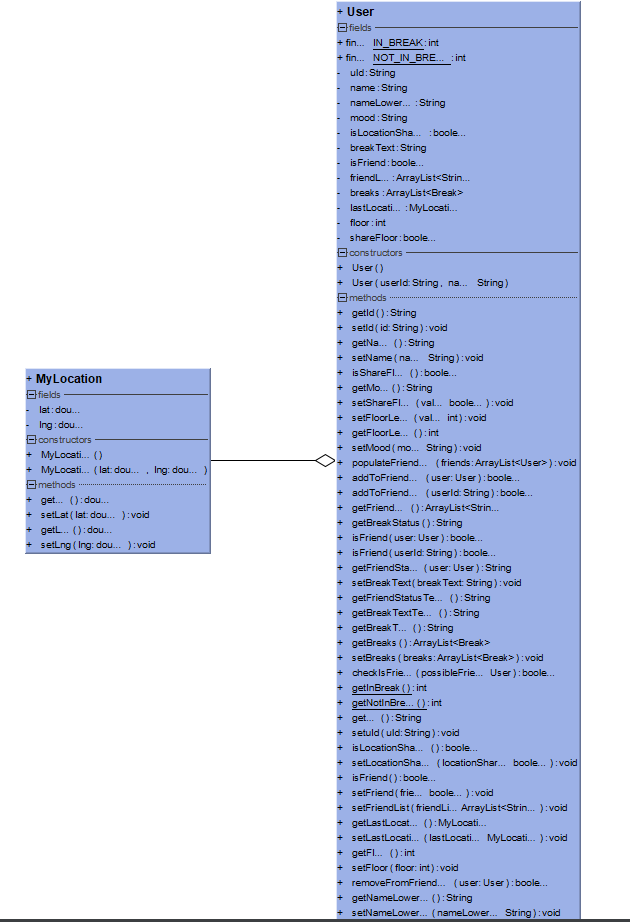
Adapters and View Holders



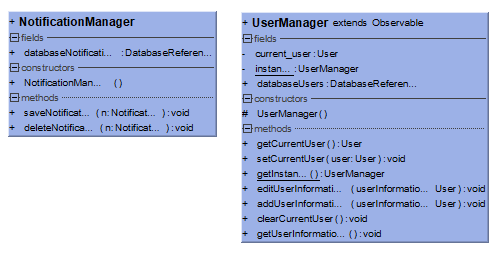


Models:





Managers:



**Documentation**

Since the documentation is extremely large (i.e. more than 300 pages long), it has been put as an appendix to this document. Therefore, if the JavaDoc needs to be consulted, it will be found at the end of this document, see Appendix A.

Note: The format of this documentation is not the best due to a large amount of conversions (HTML to PDF to RTF to DOCX). However, the key elements are still available to the reader.

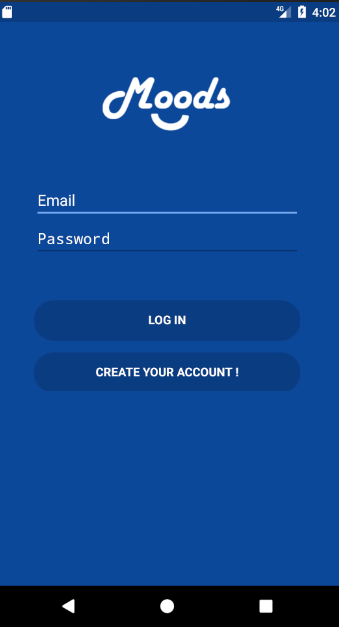
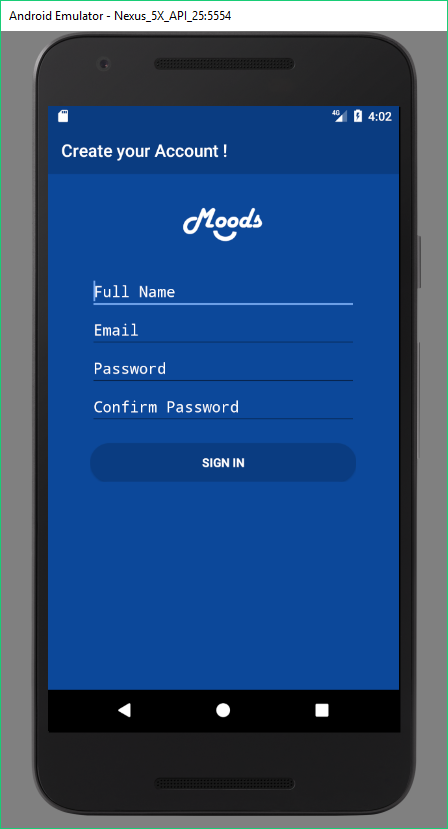
**Explaining Database Design**

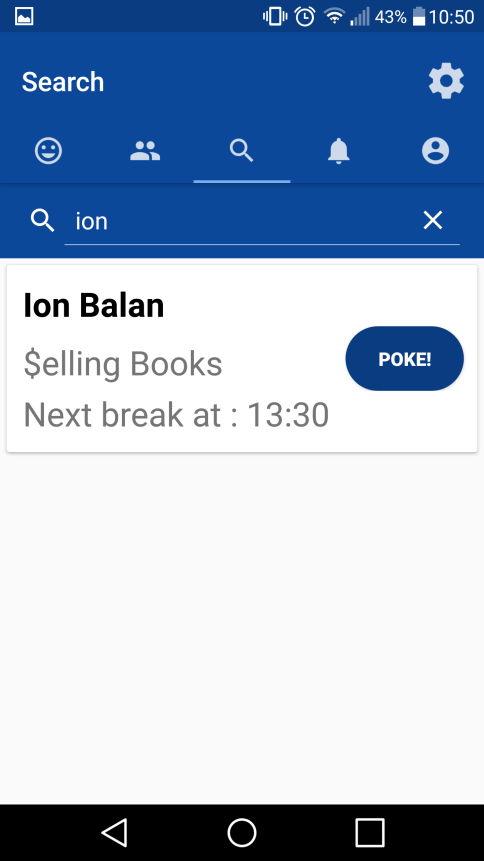
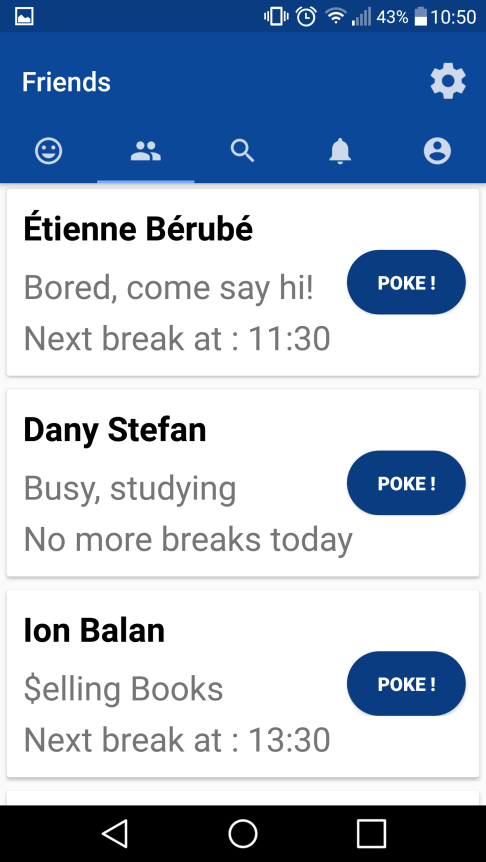
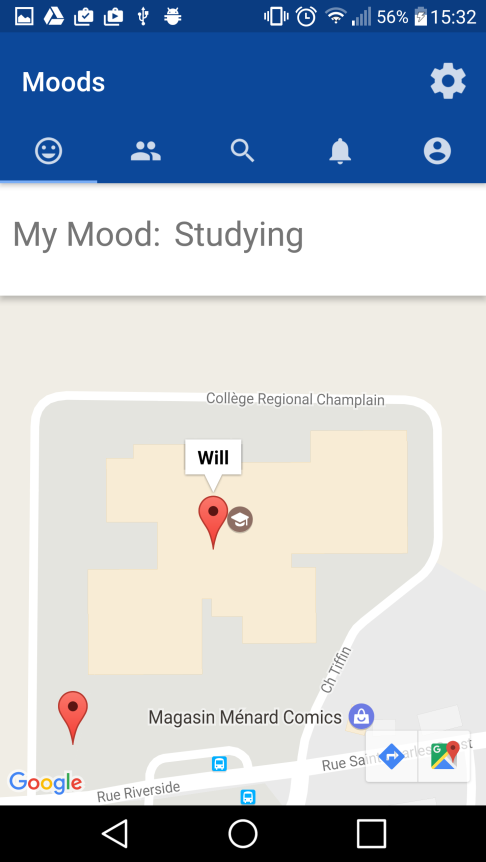
**Graphical User Interface**

Loading Page

Create Account Page

Login Page

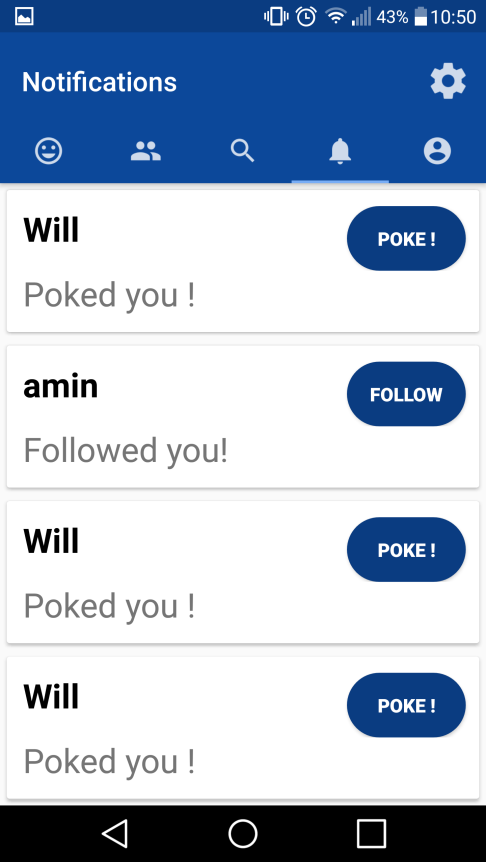
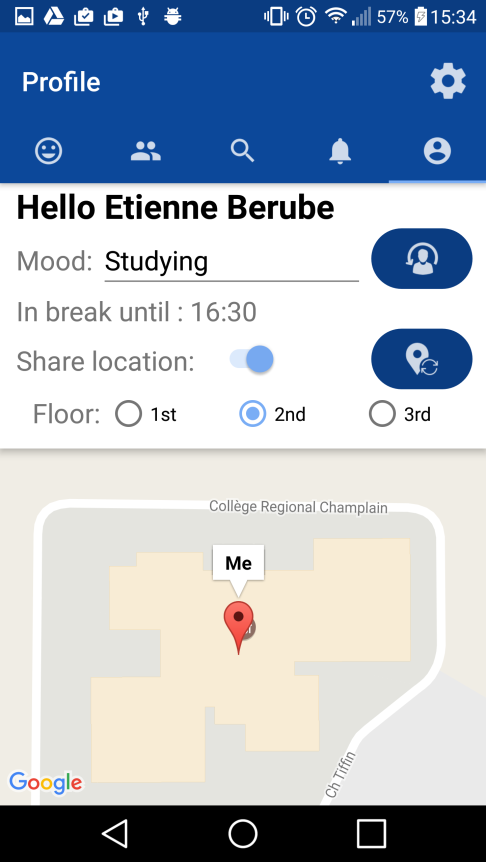
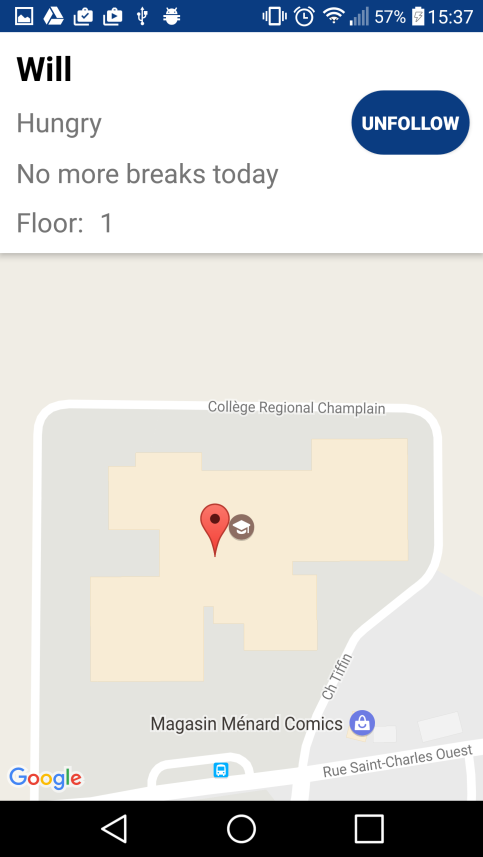




Search Page

Friends Page

Moods Page



Notifications Page

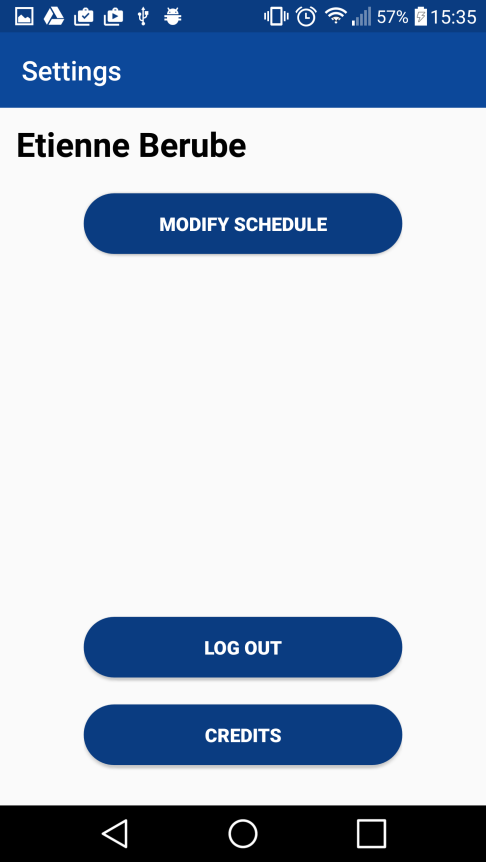
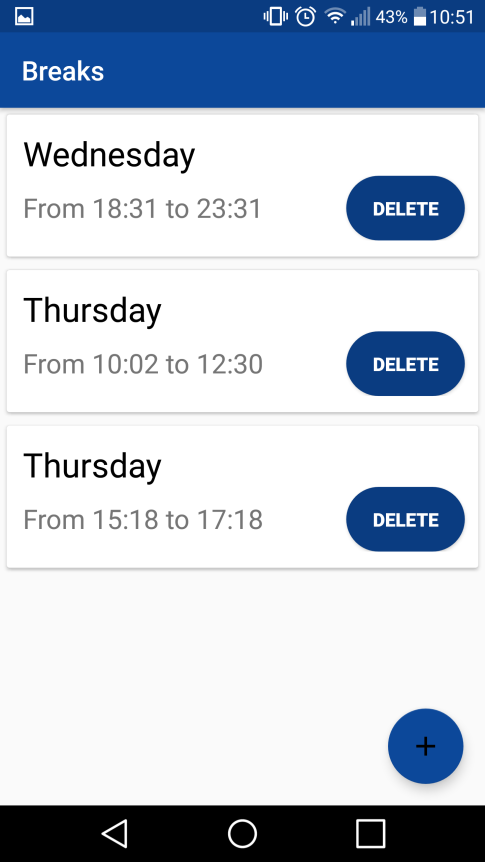
Friend Profile Page

My Profile Page

Credits Page

Breaks Page

Settings Page



For the graphic user interface, we used a modern look inspired by other social medias like Instagram and Facebook. We used blue as our main colour because it is the Champlain College’s main color. Our GUI makes using the app easy, it is user-friendly and intuitive. More information is given in the User Documentation Manual as to how users interact with the GUI.

**Methods of Evaluation**

**Results: System Quality**

**Developer Perspective**

We are really satisfied with the final version of our app. We were able to implement the great majority of the features we wanted in the beginning of the project. On some aspects, we may have overestimated ourselves in the beginning because the project seemed easier than it was and we thought we would maybe have more time than reality. However, for three students that did not know anything of XML and Firebase, we did a great job in implementing the features. Also, we put a lot of time on the layout throughout the project, so it is certain that we are proud of it. This is why we are greatly satisfied with the final result.

**Objective measure**

It is sure that there are still minor bugs with our app, but we put a lot of time on testing and debugging it. It would take a lot more time to make the app flawless. Overall, we still think of our app as being highly functional.

**Testing**

**Project Management**

**Timeline**

**Software**

Platform**:** Android Java

The application runs on Android Devices. All devices using Android Lollipop and above have access to the application.

Menu: User account

The user interacts with the app throughout multiple pages using an account. There is a home page, a profile page and multiple other pages for the user to interact with the account and other users’ account.

Accounts: Firebase

The users are able to log from a database using Firebase. Firebase takes care of the user’s security. Using this database, the application is able to retrieve the user’s information.

Code sharing platform: GitHub

The GitHub platform was used in order to efficiently work on this project. Using this service, one may share his codes and resources in real time with his coworkers.

Code IDE: Android Studio

Android Studio is used in order to manage and program everything that is in relationship with the activities. This program is optimized for a visual environment in order to code XML and Java files efficiently.

\

**Features**

* Mood (Information as to eat, study, tutor, lift/go home, chill, etc.)
  + Helps user communicate with students (tells what the user is doing)
  + Stored in database
* Breaks
* User enters their own breaks
* Other users can see if the user is in break or not
* Location of users (sends to friends)
* Uses GPS location from device (Google API integrated in Android Studio)
* Asks user to tell on which floor the user is
* Map of Champlain
  + Use Google Map API
* Networking
  + Use Firebase
  + Store user information on Firebase
    - User ID
    - User name
    - Email
    - Mood
    - Breaks
    - Location

**Conclusion**

In conclusion, as explained in the Design, for the application to properly work, it must connect to a web service to store and retrieve information from a database. The architecture of the system lies within 2 layers. The application layer and the web service layer. The design has allowed us to see how we will exactly proceed in order to build the application. The most challenging part was to understand how the web server will work, as it is something that we have not studied in class. Nonetheless, after hours of research, we have now a fair understanding of how our server will interact with the application layer. The design was overall far more complex than what we thought, as we have to handle every single interaction between the user and other users. Therefore, we have an overwhelming amount of methods that we will need to treat properly in order for the application to run seamlessly. Considering we have less than two months to implement the coding, it will require a lot of work and discipline, as we have to be the most efficient possible in order to achieve what we initially planned.

Finally, we are a team of 3 devoted college students who want to change the way Champlain students interact with each other. Our goal is to design an efficient application called Moods who will aim on the wellbeing of the student’s relationships. This program will allow Champlain students to connect more easily together during breaks. Each user will be able to share his current mood (a small sentence that describes his current status i.e. hungry, need help is [Course name], bored, etc.). Our project uses a SQL database to store each of the user’s course schedule, friends and mood. Most of the program’s interactions will be between the database, the user and the user’s friends. There will be a certain amount of Android pages (activities), but most of the work is in the networking. This application will have a scientific purpose, yet it will use a lot of algorithms in order to efficiently deal with the lists of elements and arrays of data. The overall application should be finished for the 27th of April and should be tested and fully operation for the 9th of May.

****

**Manuals**

**System Manual**

**Installation Instructions**

Installing an app from another source than the Google play store might sound difficult and intimidating but it is quite simple. There are only few steps in order to get this done.

1. Plug your phone in a computer that has an APK (Android Package file).
2. In the notification bar, ensure that your phone is set on MTP or File Transfer mode instead of charging only.
3. Open your device from the file explorer of your computer.
4. Drag and drop your APK to a folder in your device (the location is not important but remember where it is located).
5. Open the setting app on your phone.
6. Go to the Security menu
7. Enable Apps from unknown sources (the phrasing might change from a device to another).
8. Use the file explorer app of your choice (most devices come with a default one).
9. Find the APK.
10. Click on it and click accept the invitation to install it.

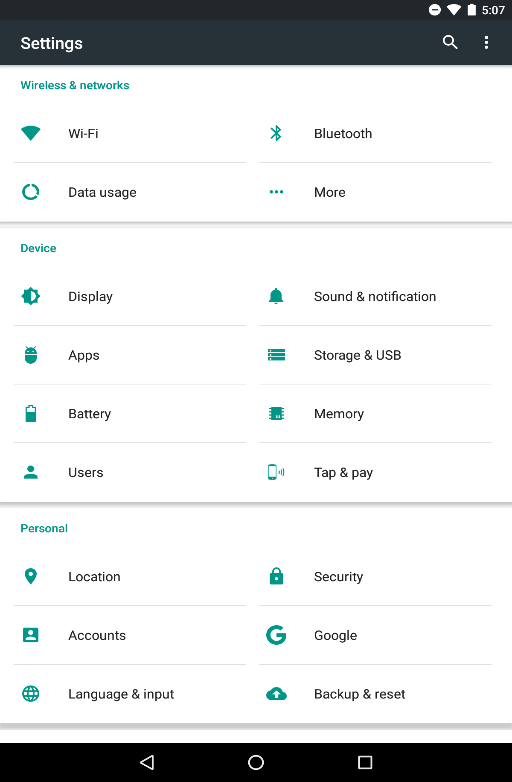


Image 1 Security menu

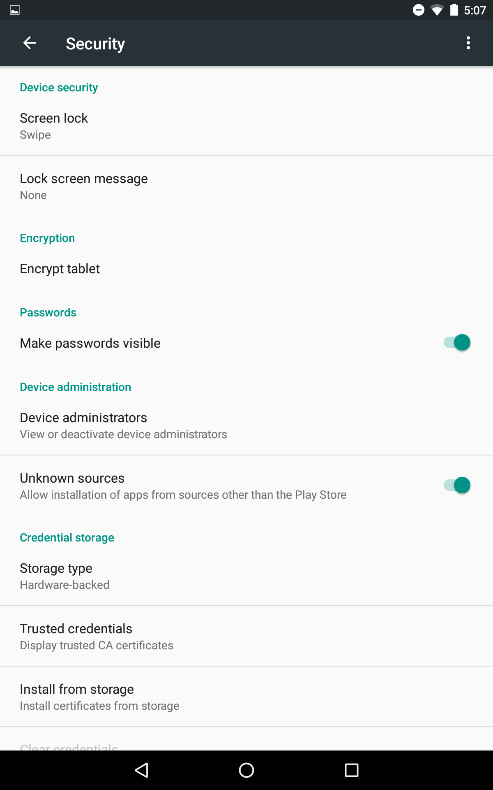


Image 2 Unknown Sources

**User Documentation**

**Procedures help section for performing particular system tasks**

From the Login page:

To create your account, click on “Create Your Account !”. Then, fill out the required text fields and click on “Sign in”.

To log in, fill out the required text fields with your own account information click “Log in”.

When Logged in:

To go to the home page (the Moods page), click on the smiley icon.

To go to your friend list (the Friends page), click on the people icon.

To go to the Search page, click on the magnifying glass icon.

To go to your notifications (the Notifications page), click on the bell icon.

To go to your profile (the Profile page), click on the person icon.

To go to the settings (the Settings page), click on the cogwheel icon.

From the Moods page:

To see who is associated to the pin location, click on the pin.

To go to your friend’s profile, click on your friend’s pin location and click on your friend’s name.

From the Friends page:

To go to a friend’s profile, click on your friend’s name.

To poke your friend, click on “Poke !”.

From the Search page:

To search for a student, enter the student’s name in the search bar and press enter.

To go to the student’s profile, click on the student’s name after the search.

To poke your friend, click on “Poke !” after the search.

To follow a student, click on “Follow” after the search.

From the Notifications page:

To go to a student’s profile, click on the student’s name.

To poke your friend, click on “Poke !”.

To follow a student, click on “Follow”.

From the Profile page:

To change your mood, enter it in the text field and click on the update mood icon.

To share your location, click on the switch to put it on, select the floor you are currently on and click the update location icon.

From a student’s profile:

To follow or unfollow the student, click on “follow” or “unfollow”.

From the Settings page:

To modify your schedule, click on “Modify Schedule” and click on the plus sign from the Breaks page. Then, add your breaks one at a time by selecting the day, start time and end time of your break, and click on “Save”.

To log out, click on “Log out”.

To see the credits, click on “Credits”.

**Information Reference Guide**

Moods uses a toolbar to navigate between the Moods page, the Friends page, the Search page, the Notifications page and the Profile page. The Moods page shows your mood and pins on the map that represents your friend’s location in Champlain. The Friends page shows your friend list, the mood of each of your friends and a phrase telling when their next break is. When clicking on a student’s name anywhere in the Friends, Search and Notifications page, the student’s profile is shown where you can access their information. The Search page makes it easy to find students and see their information. The Notifications page shows all your notifications of friend requests and poke, by telling who send it. The Profile page shows your name, a phrase telling when your next break is and your shared location. The Settings page is where you can choose to modify your schedule, log out or read the credits. The Breaks page shows when each of your breaks is.

**Tutorial**

In this tutorial, we will guide you through five steps: create account, follow a student, change your mood, share your location and add a break to your schedule.

Firstly, open the app by clicking on the app icon in your Android device. From the Login page, click on “Create your Account !”. Then, fill out the required text fields with your own information and click on “Sign in”. Welcome to Moods!

Secondly, click on the magnifying glass icon, to go to the Search page. Then, enter the name of a student you want to follow in the search bar and press enter. To follow this student, click on “Follow” after the search. This student has received a notification telling that you now follow the student.

Thirdly, click on the person icon, to go to your profile. Then, to change your mood, enter your mood, which is what you are currently doing or want to do, in the text field and click on the update mood icon. You have now updated your mood!

Fourthly, stay on your profile. Then, to share your location, click on the switch to put it on, select the floor you are currently on and click the update location icon. Your location is now updated!

Fifthly, click on the cogwheel icon, to go to the settings. Click on “Modify Schedule” and click on the plus sign at the bottom of the Breaks page. Then, add a break by selecting the day, start time and end time of your break, and click on “Save”.

You now follow someone, have a mood that people can see, have shared your location to your friends and have a break in your schedule!

**Appendix A**

**Documentation**