**Final Project Report for Even Semester 2020**

*Submitted as course project of*

**OBJECT ORIENTED PROGRAMMING**

**COMP6699**



*Written by:*

**Karel Bondan Andoro Herdito - 2440032373**

*Under the Guidance of:*

**JUDE MARTINEZ**

**Lecturer of Object-Oriented Programming**

**School of Computer Science**

**Binus International University**

**Project Requirements**

**Functional Requirements**

Specifically, the final project application MUST include examples of the following AT A MINIMUM:

* Use of primitive data
* Use of instance variables and objects
* Use of imported classes
* Use of custom-built classes & methods
* Use of Java Collection
* Use of exception handling
* Use of inheritance, polymorphism and interfaces
* Detailed Documentation Commenting
* Detailed Commenting of Methods
* Detailed Code Commenting

**Non-Functional Requirements**

Creating a working implementation is far from sufficient to achieve full marks!

* Stick strictly to the above specifications
* Don’t forget to add helpful comments
* Use (long) meaningful identifiers
* Use ample white space and a consistent indentation scheme
* Make code easy to read by keeping it simple
* Avoid duplicating similar code
* Ensure that all domain classes are modeled properly
* Use a proper access control to variables and methods

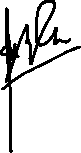
**Plagiarism/Cheating**

Binus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

**Declaration of Originality**

By signing this assignment, I understand, accept and consent to Binus International terms and policy on plagiarism. Herewith I declare that the work contained in this assignment is my own work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student:



Karel Bondan Andoro Herdito

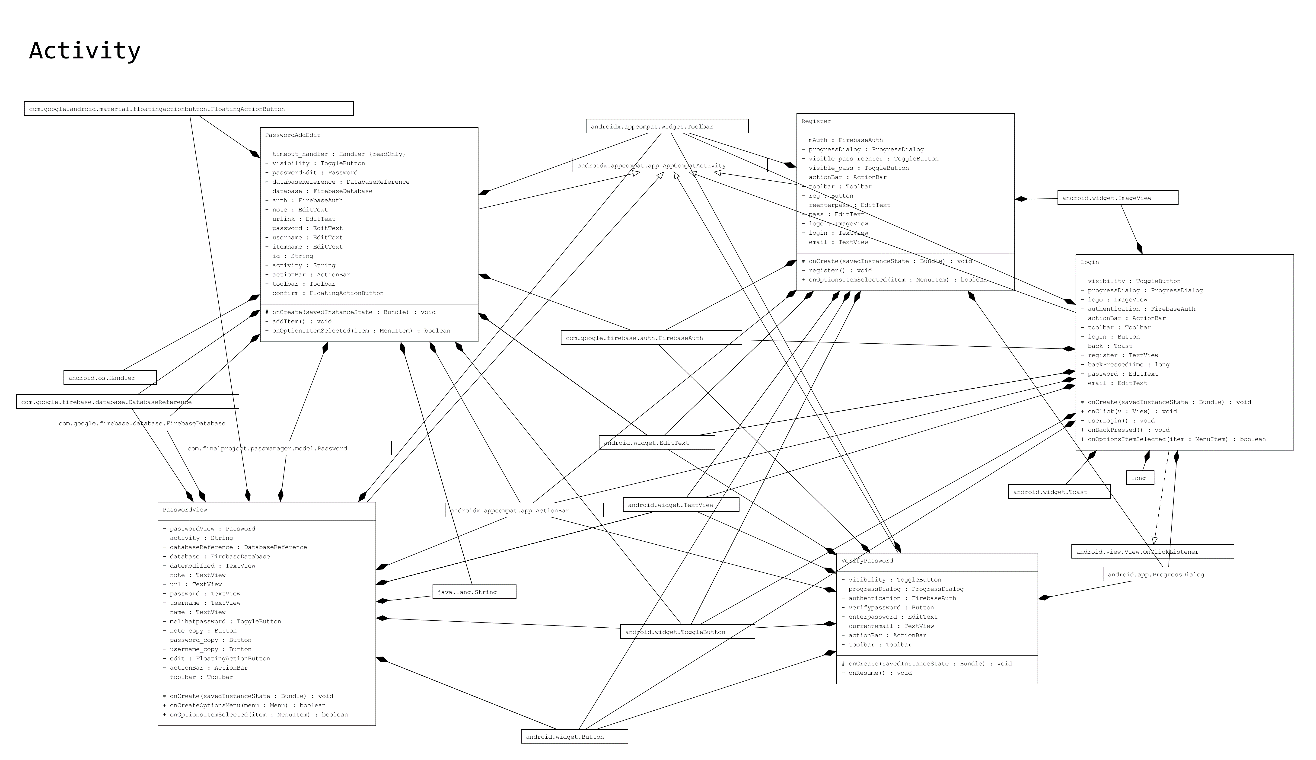
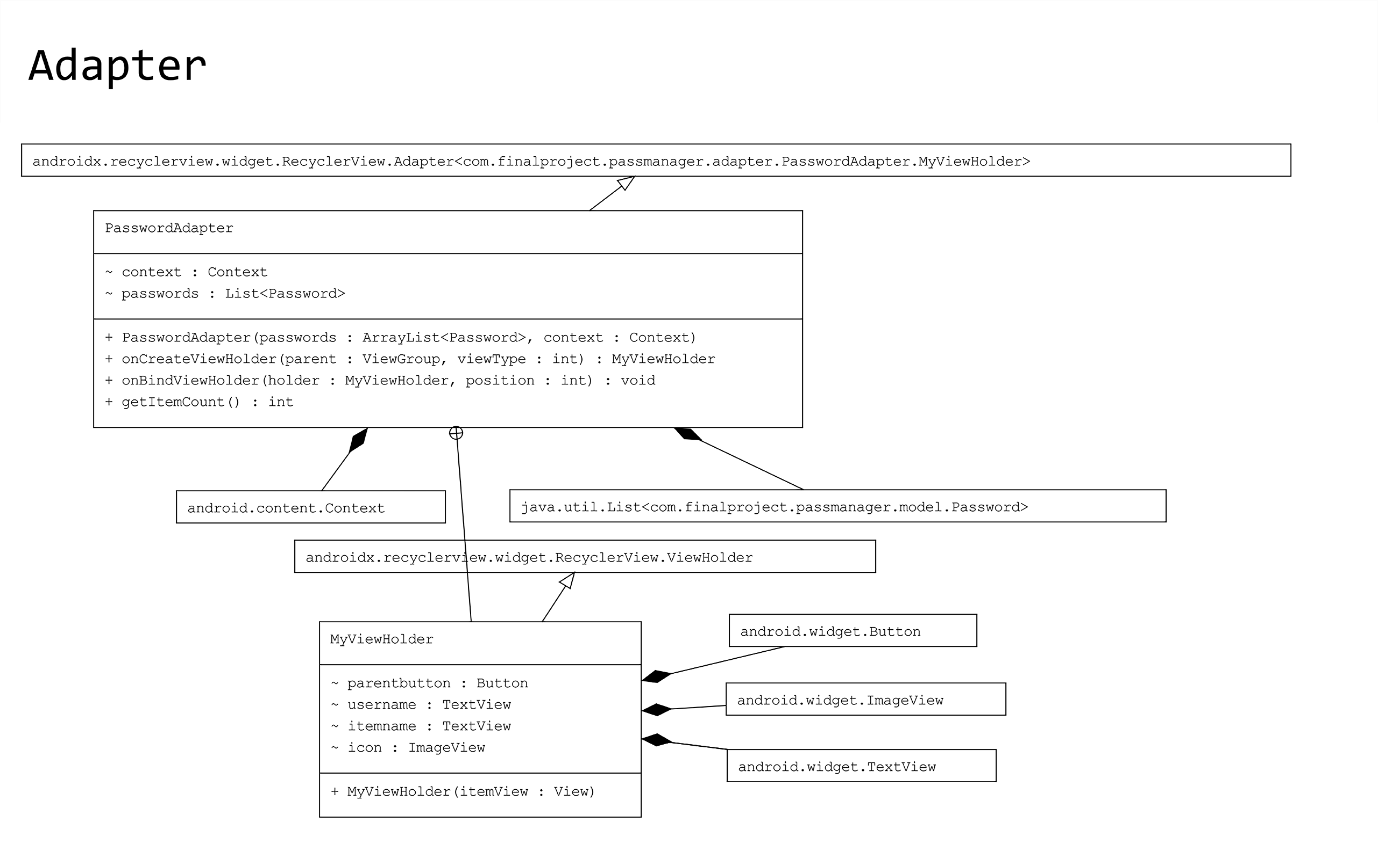
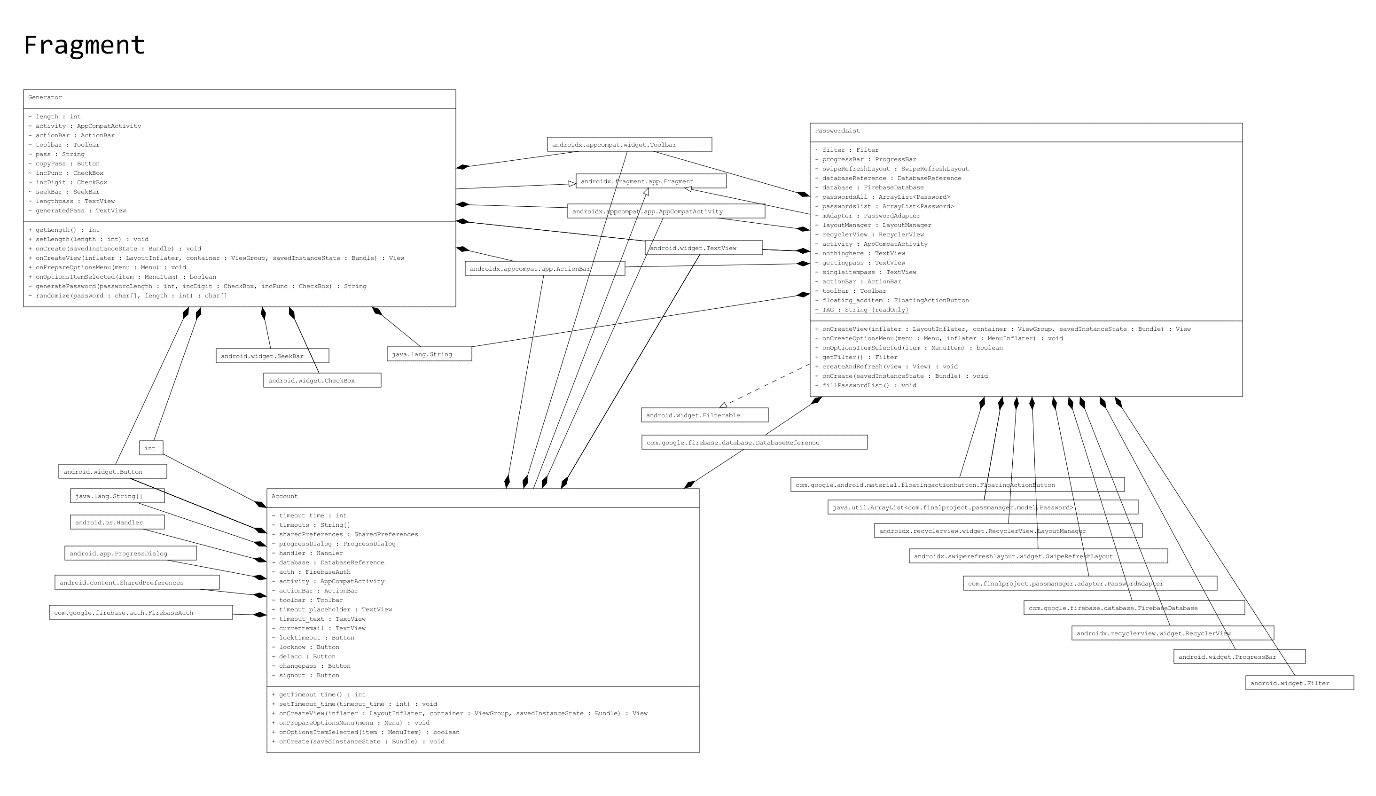
1. **Project Background**

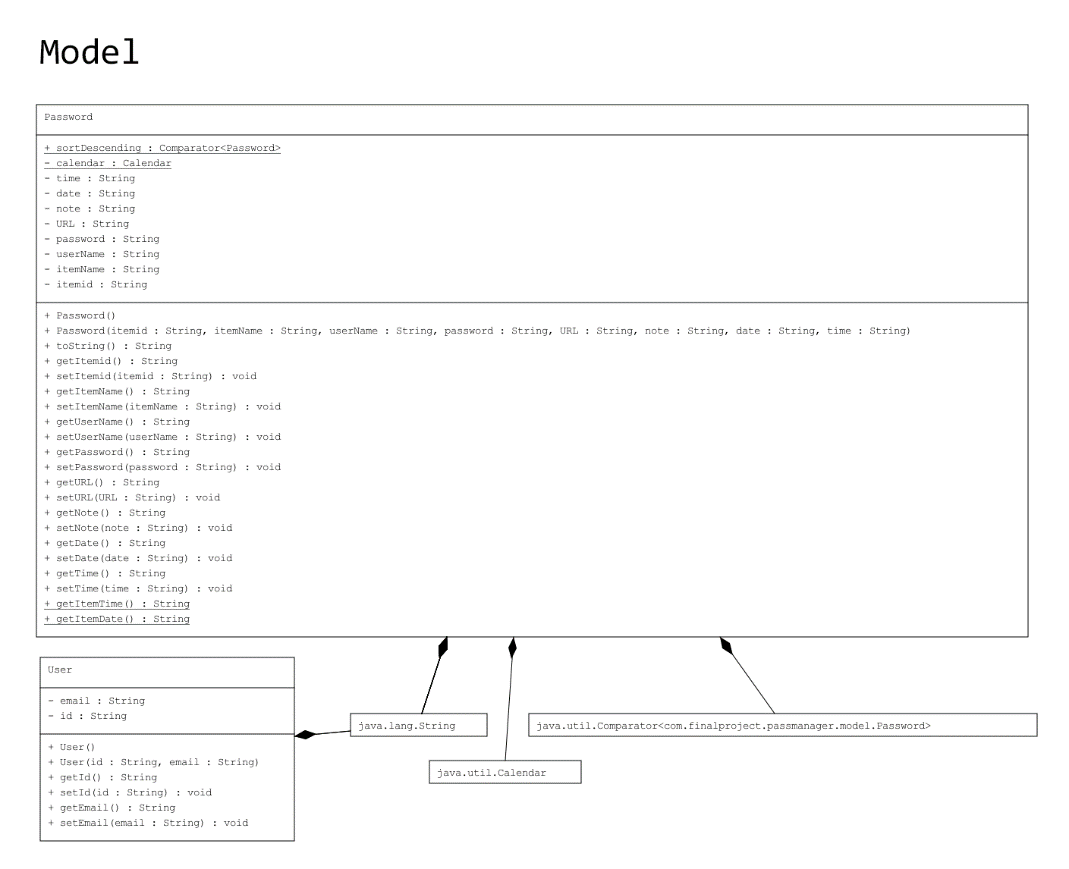
I made an GUI application called “Password Notes”. It is an Android application that can save and store your password into the cloud so you can access it from anywhere, anytime you want.

1. **Project Specifications**

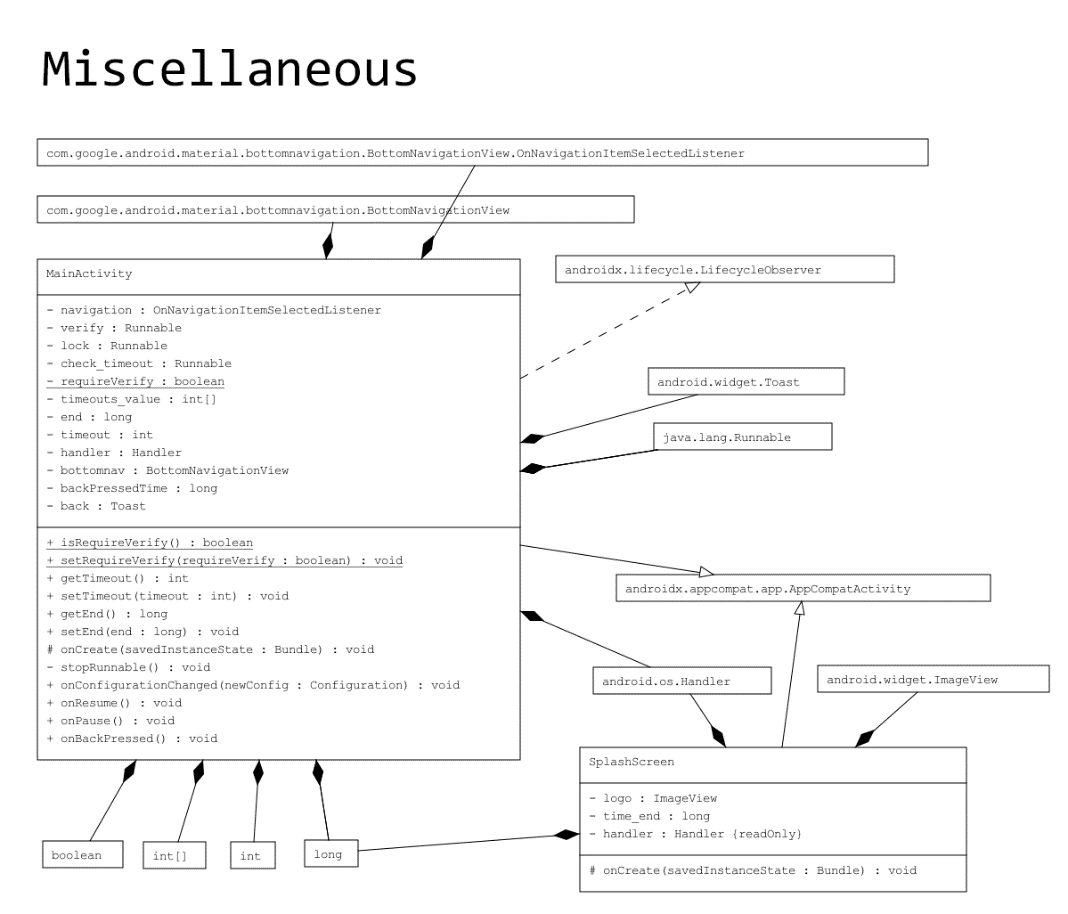
Password Notes is made using Android Studio, a Java compiler which is based on IntelliJ IDEA developed by JetBrains. It uses Android as the main operating system for the base application with user interactions to work. The cloud platform, which used to store the passwords uses Firebase as the main database.

1. **UML Diagram**

* **Activity package**
* **Adapter package**
* **Fragment package**
* **Model package**

****

* **Miscellaneous package**



* **Explanation**
* **Activity and Fragment packages**

The activity and fragment packages are the packages of user activities. For example, in the activity package, there is a class called “Login”. That class contains the activity that will display the login screen to the user. In that screen, user can input their email and password to login into the app. As the name suggests, Fragment means that the classes contained in that package are fragments of the Main Activity of the application. There are three fragments in total, one is “PasswordList” which will display the inputted passwords by the user, the other one is “Generator”, that includes a password generator for the user if they are struggling to make a new password. Lastly, “Account” is where the user can interact with their account. Further explanations will later be discussed in detailed in the Code Explanation part of this project report.

* **Model package**

The model package is where the base of the operation of this app is. There are two classes in this package, they are “Password” and “User”. As their name state, “Password” contains the instances of a password editor, and “User” contains the instances of a user. Further explanations will be discussed in the Code Explanation part of this project report.

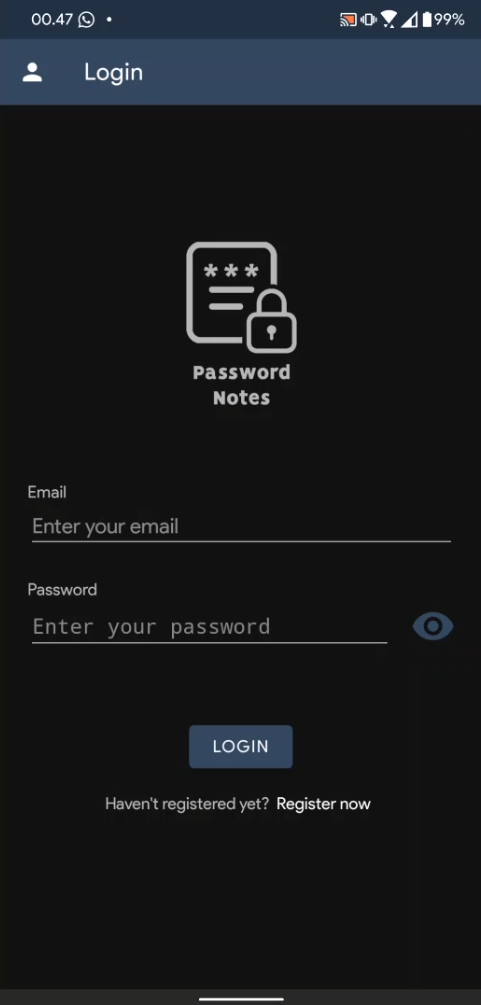
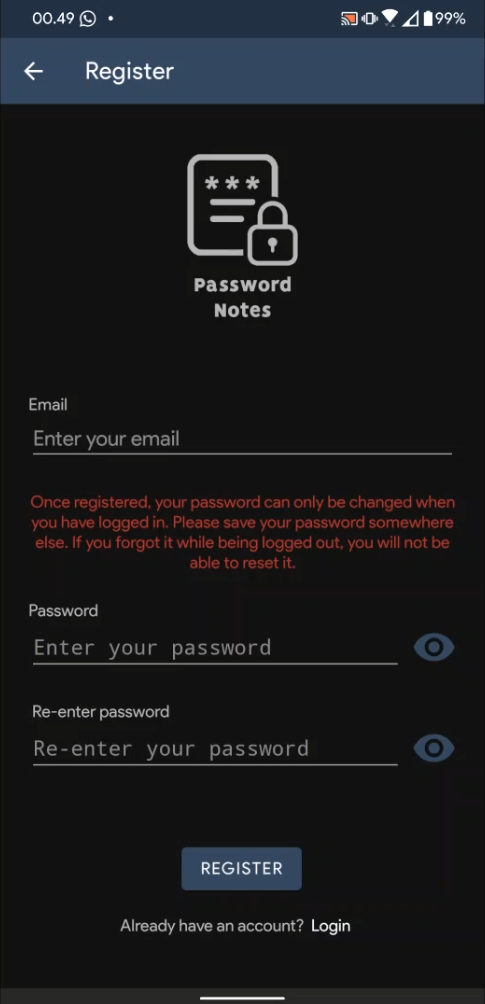
* **Adapter package**

The class contained in this package acts as an adapter for the “PasswordList”’s RecyclerView class. RecyclerView is one of the core elements in Android app development. It is widely used as a way to display selectable items in a form of a dropdown list. The simplest example of RecyclerView is the “Settings” app in the Android operating system.

* **Miscellaneous package**

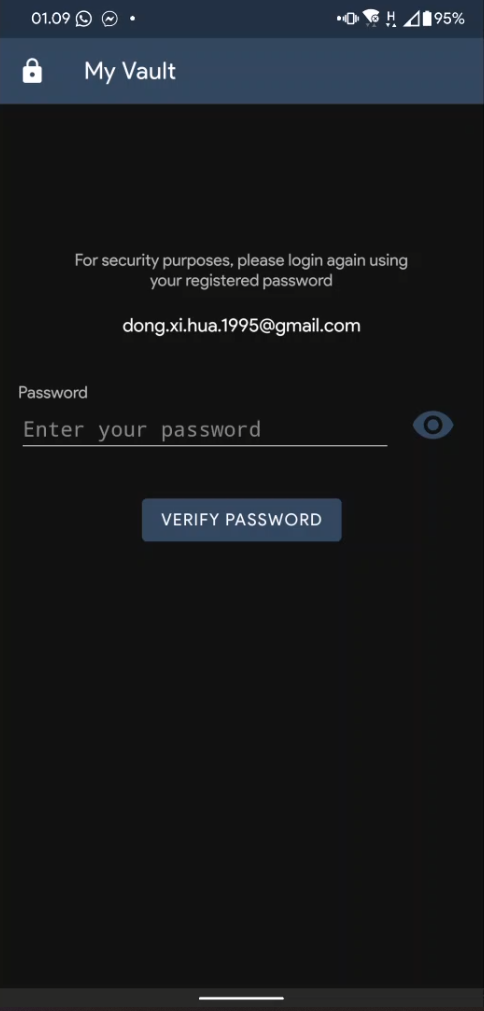
This contains only two classes, they are MainActivity and SplashScreen. MainActivity is the main activity of this application. It contains the three fragments that has been explained above. SplashScreen is there to check whether the app already has a signed in user or not. If there is already a signed in user, then it will tell the application to go to the VerifyPassword activity, else, it will tell the application to go to the Login activity.

1. **Application Interface and How the Application Works**

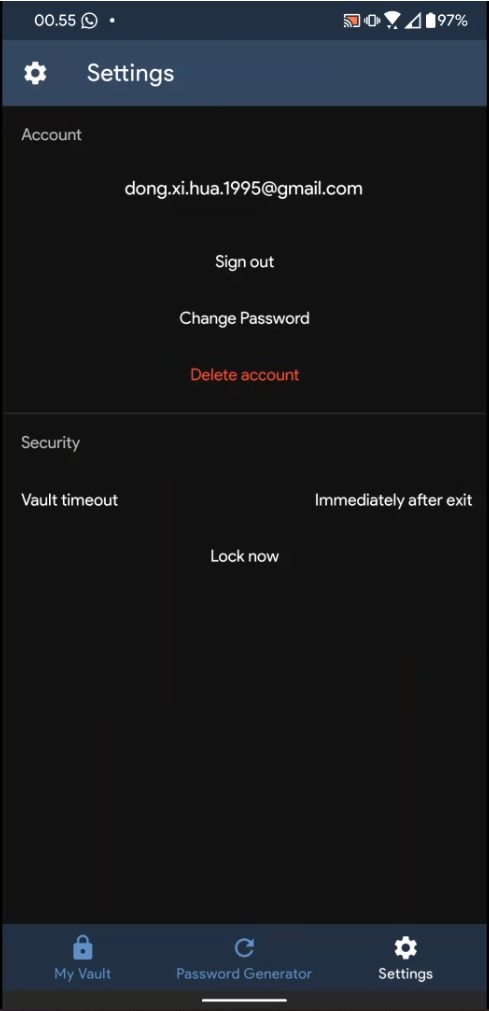
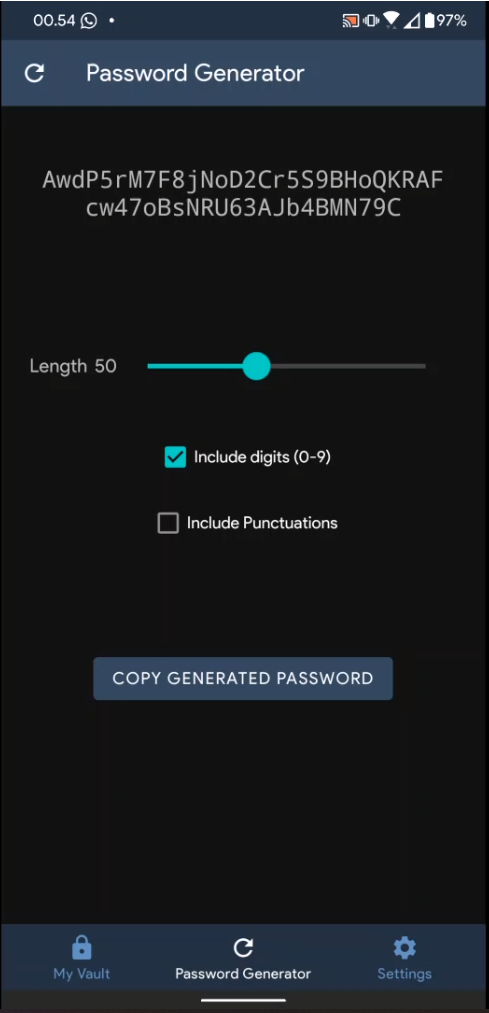
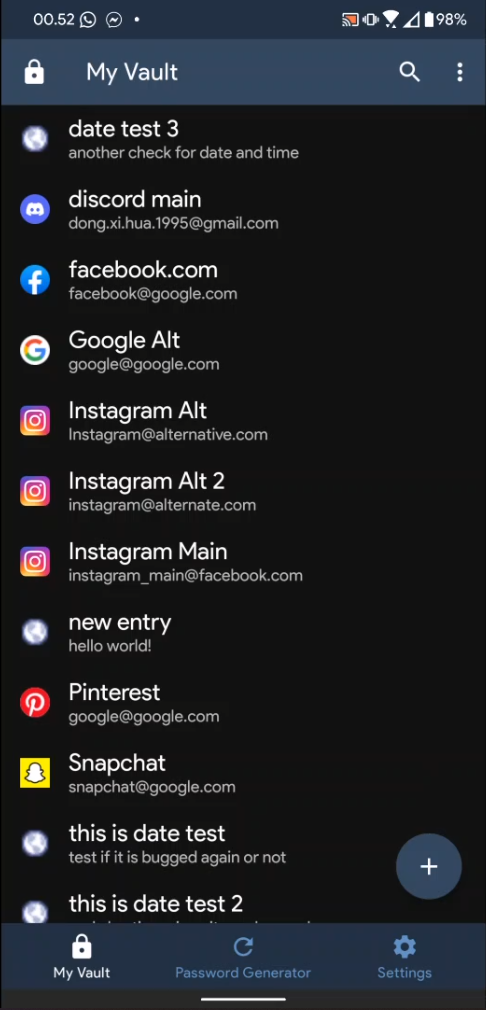
* **Register and Login activities**

These are the Register and Login activities of the application. When the app is first opened and there is no user logged in, the app will go to the Login activity. When the user taps on the “Register Now”, it will go to the Register activity. There, the user can register for a new account. However, if the user is trying to make a new account with an email that is already exists in the database, the app will throw an error displaying to the user that the email entered has already been registered in the database. When the user is registering a new account, they will be prompted to verify their new account via their newly registered email. There will be a link where the user can verify their email.

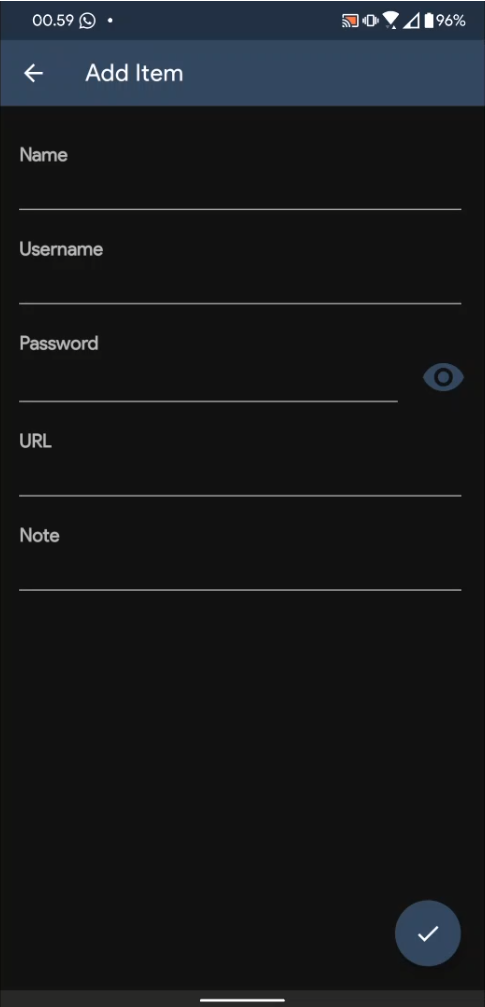
* **VerifyPassword activity**

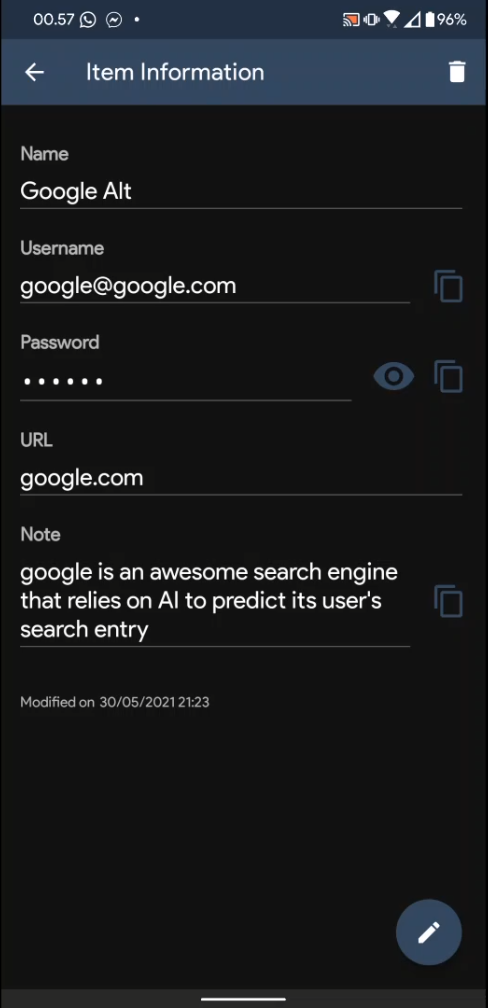
****

After the user has successfully logged in, they will be redirected to the VerifyPassword activity first before they can finally view their passwords. This is an added security so that their password vault will be safe.

* ******PasswordList, Generator, and Account fragments**

After they have entered their password in the VerifyPassword activity, they will be redirected to PasswordList activity which is the home fragment of the MainActivity. This fragment shows as “My Vault” to the user. There, user can interact with their previously added passwords and or add new ones. This application provides a password generator for the users to ease themselves on an account creation process. The user can access this feature by tapping the “Password Generator” tab on the bottom middle of the app screen. This is the “Generator” fragment in the source code. Lastly, the user can also interact with their account and also the application itself. For account control, there are sign out, change password, and delete account options. For the application control, there are vault timeout and lock now options. Vault timeout is basically a setting to lock the app if the time has exceeded the amount of time set by the user. By default, the app’s timeout is “Immediately after exit”, which means that if the user exits the app, they will be greeted by the VerifyPassword activity when they return to the app.

* **PasswordView and PasswordAddEdit activity**



Moving back to the PasswordList fragment, when the user tapped on the plus button on the bottom right of the app, they will be redirected to the “Add Item” activity above. There, the user can add new entry by filling out the details in the text fields that have been provided. When the user tapped on an existing entry, they will be redirected to the “Item Information” activity above, which contains the information about the password entry itself. There, the user can copy the username and password, and also view the password that by default is censored by dots. The user can edit the entry by tapping the pencil icon on the bottom right of the app. There is also an information about when the modification was last made on the entry on the bottom left of the screen, below the “Note” field.

1. **Libraries**
2. Android SDK 30

Android SDK 30 is the application development codename for Android 11. This library provides the core elements for Android applications.

1. Firebase

Firebase is an API used to connect the project to Firebase database. It is by default included in the Android SDK, but it is a standalone library, meaning that it is not part of the core elements for an Android application to be able to run.

1. **Lessons Taken from Doing the Project**

From the moment that we were asked to think about what to make for the final project, I instantly thought about an Android application. I do not know why but I was so intrigued on Android development. I wanted to make an Android application for the sake of my curiosity on the world of the Android development. And so, I found out that this is the greatest opportunity for me to dive into the world of Android application development. Without hesitation, I wrote in the Google Docs proposal that I wanted to make an Android application, whatever the application will be. I began to look for inspiration, and a password manager application seemed to match my interest. And so, without a single knowledge of Android Studio, I began coding by looking tutorials online. There were ideas that were scrapped because there was not enough time: an ability to login by using fingerprint; a more advanced security, that is to prevent the user from screenshotting the app; the ability to autofill login and password fields on apps and websites (I have looked everywhere, but there were no one that discussed this topic). There are still other projects that needs to be done. Android Studio was totally a new thing for me. I did not know anything related to it. And so, this was the biggest difficulty that I faced. Other than that, the confusing “deprecated” signs that appears on some methods made me confused on what alternatives to use, because the methods being marked as “deprecated” are the ones that are used in the tutorials. The lessons that I can take from this project is that Android development sure is fun, but sometimes it can be a tedious work because something that looks so simple is actually so hard to implement behind the screen. One that might think is simple but very hard to implement is the dropdown selectable items list, like in the device settings app. That one was very tedious to make. From this fact, I can appreciate Android developers even more who made majestic app UIs and animations.

1. **Project Technical Description**

Password Notes is an application that can take note of user inputted credentials that are used in other services. The credentials that have been submitted as entries are going to be stored in a database. The database used for storing the entries is Firebase, a database provider developed by Google (the database structure is different from MySQL. It is simpler and more friendly to be used especially for newbies). Password Manager is an Android application developed in Android Studio using Android SDK 30. In other words, it was developed using Android 11’s SDK. The features of Password Manager are:

1. Make new entries.

The said entry is in a form of user credentials used to log in to other services other than Password Manager. Duplicate entries are allowed since this application is not a specialized application such as dictionary. New entries will be automatically sorted in descending order alphabetically.

1. Edit and delete entries.

Entries that previously have been added by user can be deleted or edited, depending on the user’s need.

1. Search entries

When the entries are growing in size, it can be a tedious work to find a specific entry in the list. To ease the user on finding the entry that they need, Password Notes has successfully implemented a search function to filter the entry that the user is currently looking for from the rest of the entries.

1. Generate password.

Password Notes can generate a new password with custom options depending on user’s needs. User has three options for the password to be generated:

1. Include digits.

User has the freedom to include digits in the generated password. If this option is not checked, the generated password will not contain any digits.

1. Include punctuations.

User also has the freedom to include or exclude punctuations in the generated password. If this option is not checked, the generated password will not contain punctuations.

1. Length

Finally, user has the freedom to set the length of the generated password. The minimum length that the user can set is 6 characters, and the maximum length of the generated password is 128 characters.

1. Full account control.

To use this application, user needs to register for a new account or log in into an existing one. After registering, user has the freedom to sign out from the application, change password, and delete their account. Due to security reasons, user can only change their password once they have logged in to their account. If user forget their password while being logged out from their account, they will lose all their entries. It is highly advised that user always remember their password or store it somewhere safe.

Password Notes will not run if the target operating system is below Android KitKat (Version 4.4; Android SDK 19). As for the reason why version 4.4 was chosen as the minimum, the parent class’ methods and interfaces are much different from the previous versions. As of today, Android SDK 19 is considered as the minimum SDK for major app releases to run, as devices running previous versions of Android before version 4.4 are decreasing and is considered to be deprecated.

1. **Code Explanation**

// Code explanation goes here. This document will be updated ASAP.

1. **Project and Video Link**
2. Project Repository

<https://github.com/karelbondan/password_notes>

1. Video link

<https://www.youtube.com/>

1. **References**