

**Mobile Applications Development**

Year 2 (2019/20), Semester 3

Year 3 (2019/20), Semester 5

***SCHOOL OF INFOCOMM TECHNOLOGY***

Diploma in Financial Informatics

Diploma in Information Technology

**ASSIGNMENT**

**Duration:**  1 July 2019 to 29 July 2019

**Weightage :** 30% of total coursework

**Individual/Team:** Team (3 students)

**Format:** Stage 1(20%)

Stage 2 (80%)

**Cut-Off Date/Time: Monday, 29 July 2019, 9 AM** (week 16)

**Penalty for late submission**:

10 marks per day (including Sunday and public holiday)

No report will be accepted after **5 Aug 2019, 9 AM**.

There is a total of 8 pages (including this page) in this hand-out.

|  |
| --- |
| ***WARNING***  ***If a student is found to have submitted work not done by him/her, he/she will not be awarded any marks for this assignment. Disciplinary action will also be taken.***  ***Similar action will be taken for the student who allows other student(s) to copy his/her work.***  ***Please refer to the following URL for more details:***  *http://www.np.edu.sg/antiplagiarism* |

**MAD Assignment Report**

**Chosen category**: Social

Contents

[Finalised features: 2](#_Toc15430182)

[Context 2](#_Toc15430183)

[Application’s features, design and user guide 2](#_Toc15430184)

[Login Feature + Create New User Feature 3](#_Toc15430185)

[Main Page 4](#_Toc15430186)

[Direct Donation Feature 5](#_Toc15430187)

[Crowdfunding Feature 6](#_Toc15430188)

[Donation History Feature 7](#_Toc15430189)

[Firebase 8](#_Toc15430190)

[Requirements 9](#_Toc15430191)

[Manifest Customization 9](#_Toc15430192)

[Responsive Layout 9](#_Toc15430193)

[Intents 9](#_Toc15430194)

[ListView / RecyclerView 9](#_Toc15430195)

[Persistent Memory (SharedPreferences / SQLite) 9](#_Toc15430196)

[Regular Expression 9](#_Toc15430197)

# Finalised features:

1. Login + Create New User (Yu Hong)
2. Donation, Main page and Firebase (Deuel)
3. Donation History (Jun Wei)

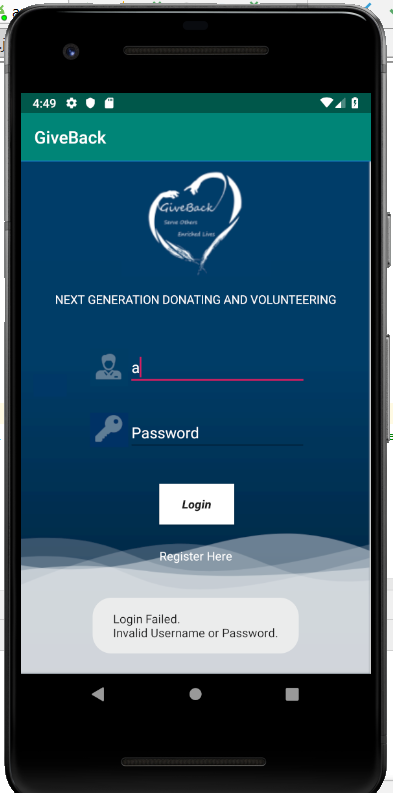
# Context

Our application aims to provide users with the ease of donating to any organization. With technology rapidly evolving and changing the way we do things, the world today is substantially more interconnected as before. As such, this increases the scope of organisations that users can donate to. Thus, using our application, we aim to provide users with a convenient and seamless donation experience without having to physically visit the organisation to make a donation. As of now, our application only allows donations to a single organisation, St Luke Elder Care.

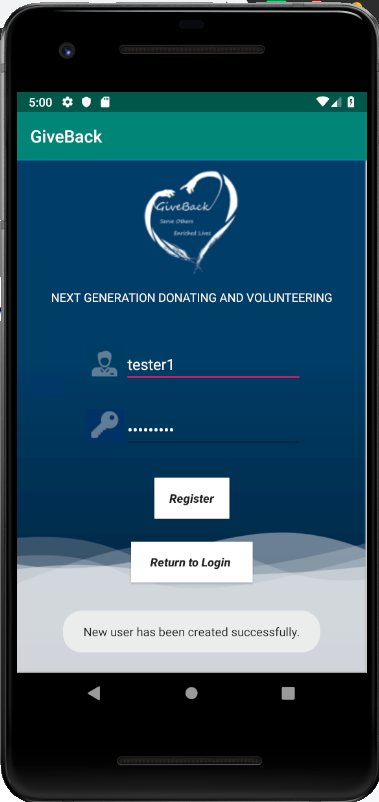
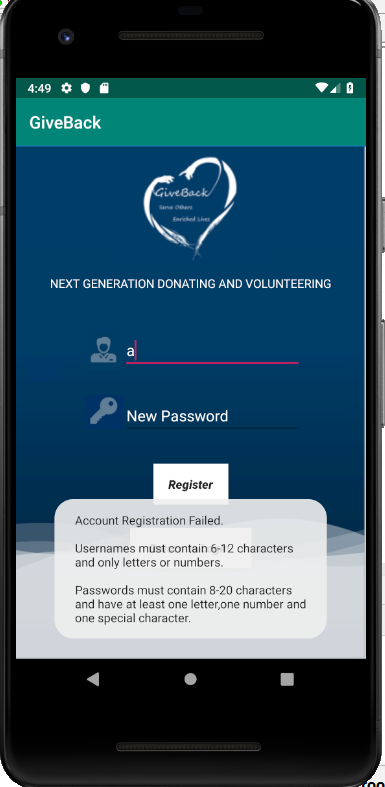
# Application’s features, design and user guide

There are various features embedded in our application to provide users with a holistic experience

## Login Feature + Create New User Feature



*Figure 1: Toast message when the login fails*

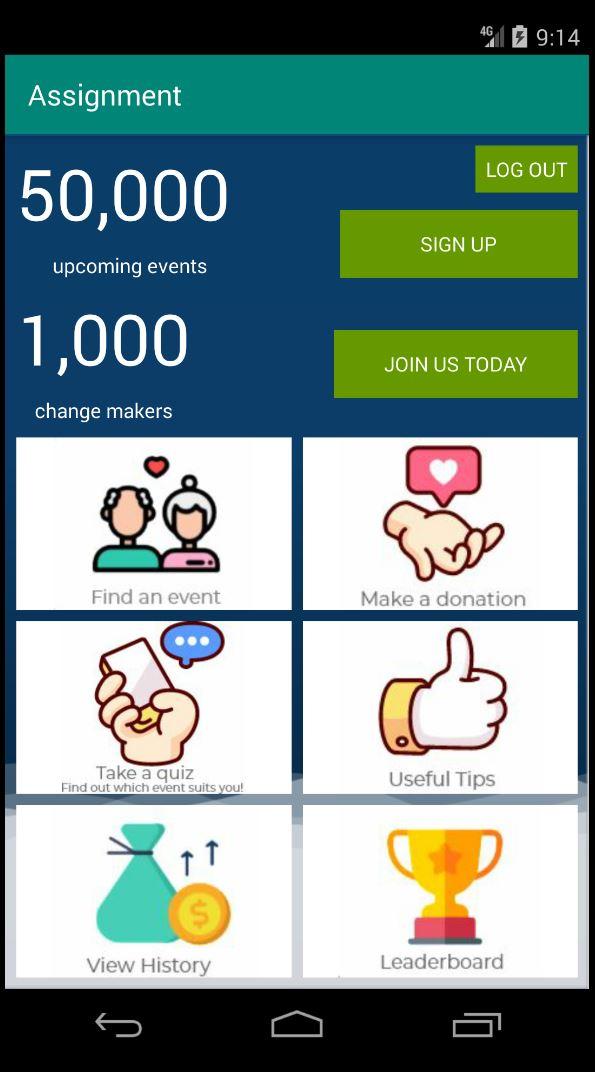
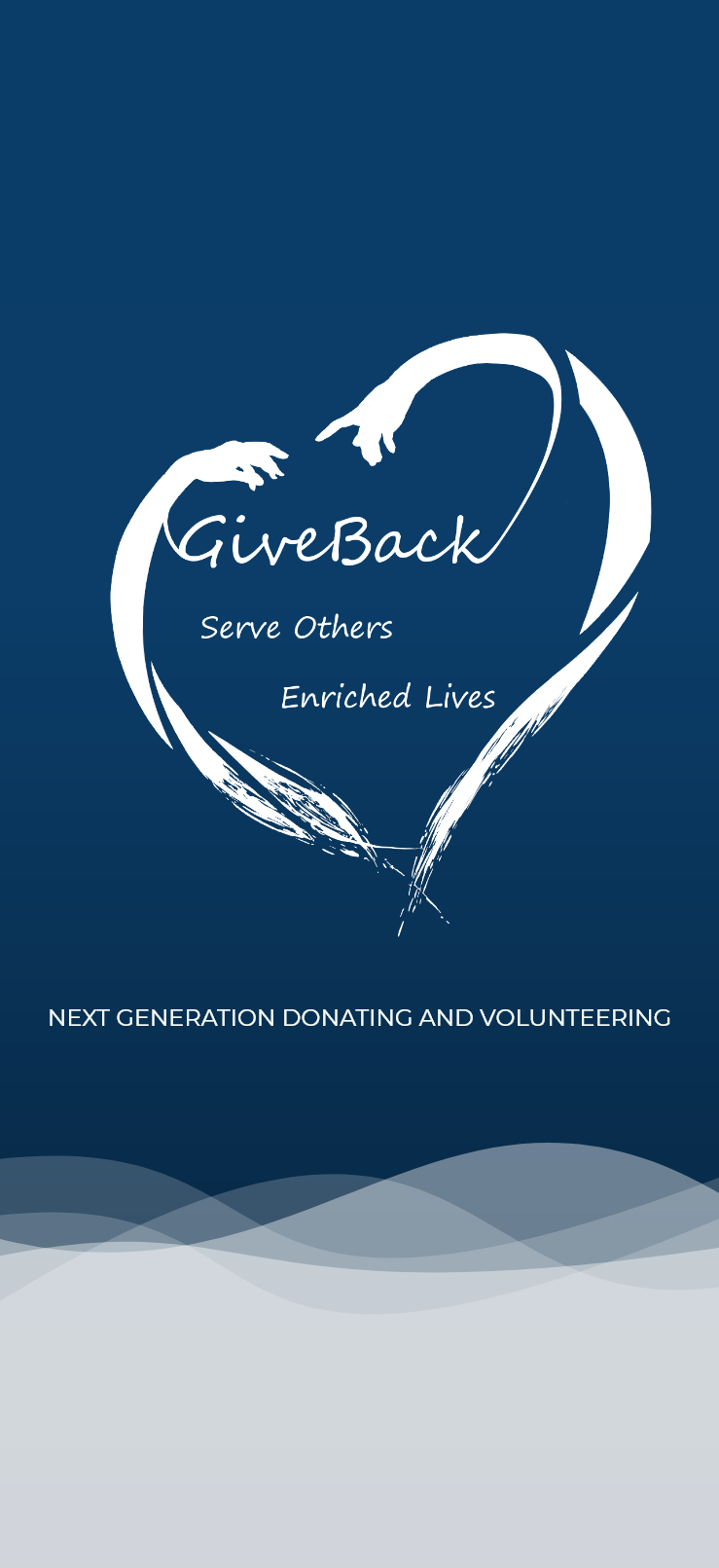


*Figure 2: Toast message when account registration fails or is successful*

The first feature would be the login function. In the login page, users will be prompted for their username and password which would act as a form of authentication. We created an onClick event handler method (onLogin) and assigned it to the Login button. When the user clicks on the Login button, a verify account method is called (verifyAccount()) from the Dbhandler class in the application to check if the username and password entered by the user match any account records in the SQLite database. If the account exists and is valid, the user is directed to the main page where the user can perform other functions such as making a donation or viewing his/her donation history. If the account does not exist or username or password is invalid, a toast message “Login Failed. Invalid Username or Password” will be displayed on the bottom of the user's screen.

We created an setOnTouchListener event handler and assigned it to the textView(Register Here).If users do not own an existing account, users may click on the textView(Register Here) to navigate to the create new user layout page. Users are required to register an account with the application by clicking on the “Register” button at the bottom of the screen. They are required to enter a new Username and Password to register an account. For usernames, they must contain 6-12 characters and only letters or numbers are allowed. For passwords, they must contain 8-20 characters and have at least one letter, one number and one special character. If the new username and new password entered is valid, the account is registered and inserted into the database successfully and a toast message will show to inform the user that the Account registration is successful. Otherwise, a toast message will be displayed to notify the user that account registration has failed and the proper validations needed for username and password.

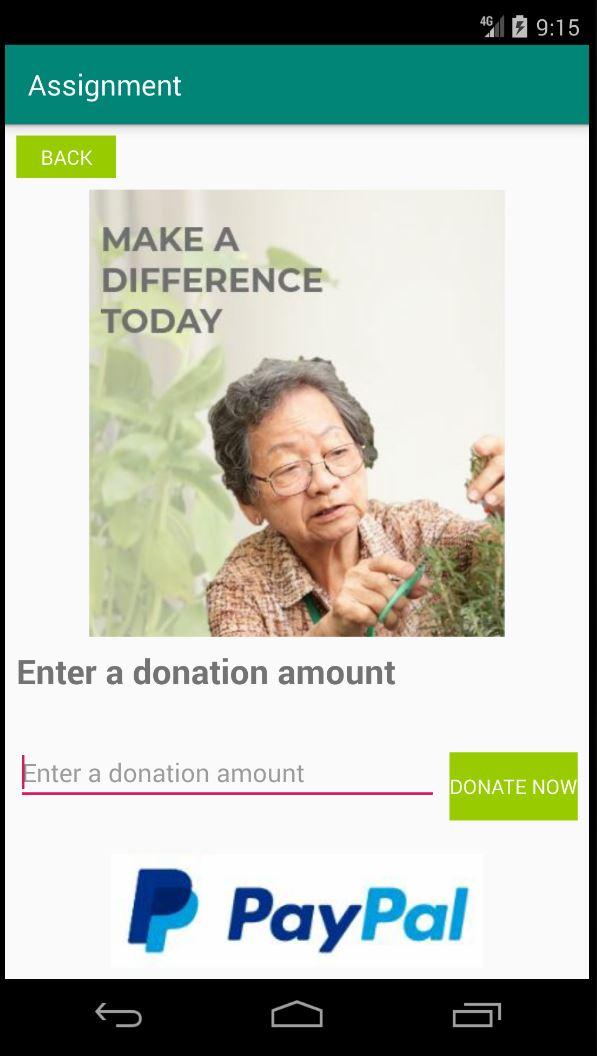
## Main Page

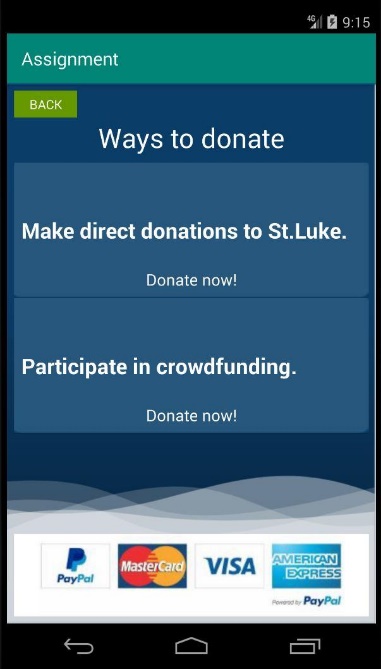
****

*Figure 3: Main page*

After the user has successfully logged in using a set of valid credentials that is verified against the database, the user will see this main page where he can access the main functions of the application such as making donations. Based on Figure 3, users can use the features of “Make a Donation” and “View History”.

## Direct Donation Feature





*Figure 4: Ways to Donate Page Figure 4.1: Direct Donation Page*

Users can access and use this feature by clicking on the “Make a Donation” icon in the main page and subsequently, clicking on the “Make direct donations to St.Luke” as shown in Figure 4. Thereafter, the application will redirect the user to the Direct Donation page as shown in Figure 4.1. There is also a back button in every page to allow users to return back to the previous page. To facilitate the transfer of the user between each page, we made use of the Intent activity and specified the names of the classes that the user will be redirected to in the parameter

In the Direct Donation page, there will be an EditText for the user to enter their donation amount. We made the design of the EditText intuitive for the user by adding the text “Enter a donation amount” in the hint property of the EditText control.

After the user has decided on an amount to donate, he can click the “Donate Now” button to confirm the donation. We created an onClick event handler method (onDirectDonate) and assigned it to the button. In the onDirectDonate method, each time the user clicks the button, the event handler method will be triggered and a new Donation object will be instantiated, with the various attributes such as Title, Amount, Date and Username being parsed in with values. We then called the addDonation method in the DbHandler class to add the newly instantiated donation object into the database using an Insert SQL Query Statement. If the donation is successfully appended into the database, a toast message with the text “Donated Successfully” will appear, alerting the user.

Since the Username attribute is defined as a foreign key in the Donation table in the database, to ensure that the username can be retrieved anytime and anywhere as long as the application is running, we created a GlobalClass extending the Application and created a getter and setter method for the username variable (username). We set the value for the username by retrieving the username entered in the EditText once the user has successfully been authenticated. With that, the username is now a global variable and its value can be retrieved by simply calling the GlobalClass class.

## Crowdfunding Feature

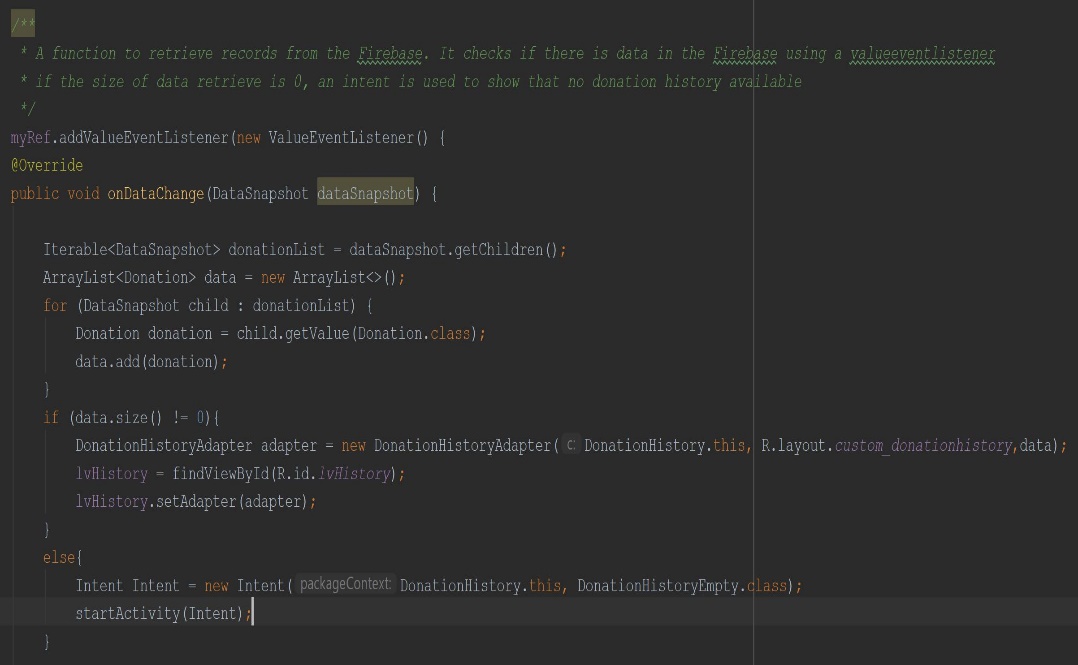
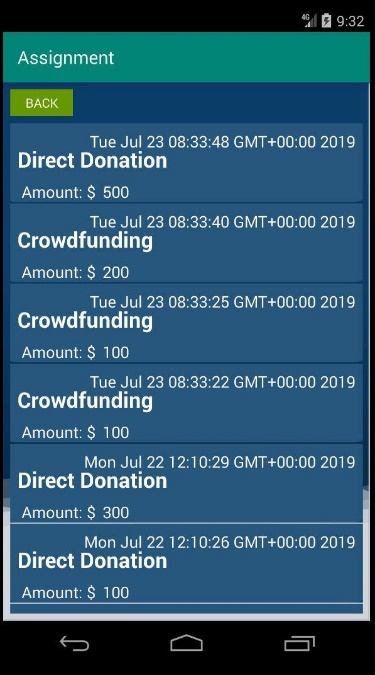
*Figure 4.2: Crowdfunding page Figure 4.3: Crowdfunding with value*

Similar to the Direct Donation Feature, users can access this feature by navigating to the “Ways to Donate” page and select “Participate in crowdfunding”. They will then be redirected to the crowdfunding page as shown in Figure 4.2 through the use of Intent activity. Similarly, the EditText design is made intuitive for the user through the display of hints to guide the user along.

There is also a onClick button event handler method (onCrowdFunding) assigned to the “Donate Now” button. For each time the user clicks the button, the method will be triggered and a new donation object will be instantiated, and its class attributes such as Title, Amount, Date and Username will be assigned with values. After that, we used a SQL Insert query statement and incorporated it into the the addDonation method that we created in the DbHandler class. We then called the method to add a donation record into the database. For each successful donation being made and inserted into the database, a toast message will appear with the text “Donated successfully” to alert the user.

There is a progress bar being implemented within the page to track the progress of the user’s total donation out of a total of $1,000. The progress bar is white in colour with $0 being donated and gradually, parts of it turn blue as the user donates more money, as illustrated in Figure 4.3. The progress bar will turn fully blue when the user has donated a total of $1,000. The unique thing about the crowdfunding feature is that we have shared preferences implemented in our application, thus even if the user closes our application and launches it again, the progress bar and the amount donated will not be resetted, as the data is saved and retrieved in the form of key, value pair.

## Donation History Feature

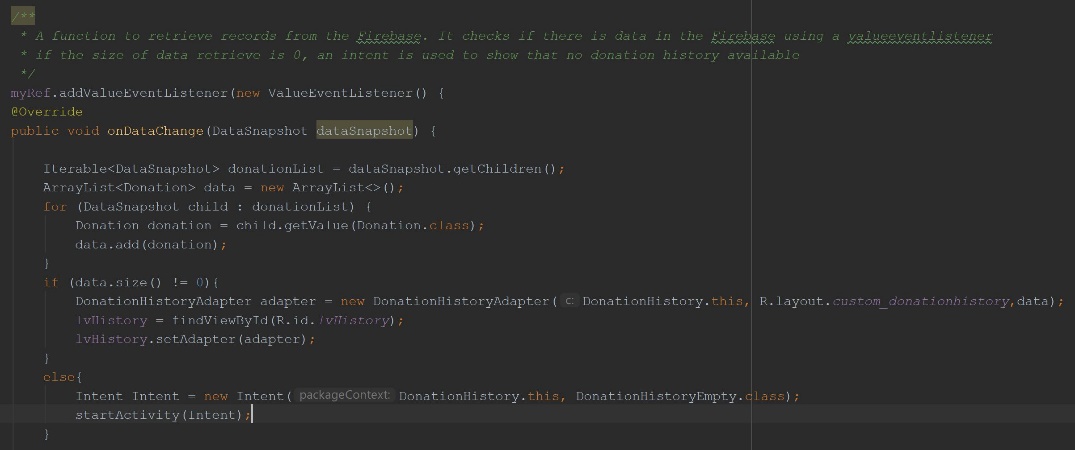
****

*Figure 5: Donation History Page Figure 5.1: Codes for Donation History*

This feature was implemented to provide greater transparency to the user regarding the donations they have made to the organisation. As shown in Figure 5, the donation records are populated into a listview and the first record shown being the latest donation made by the user. Each donation record in the listview shows the donation’s title, amount as well as date to provide context and information to the user.

Since we have implemented firebase in our application which is a non-relational database, we have to model our codes in a way that is workable for a NOSQL database. Thus, to show the donation records for a specific user, we created a function to retrieve records from the firebase. Using a ValueEventListener and specifying the relative path to where the data is stored, the function will check for the size of the data retrieved and store it in an ArrayList<Donation> object. If the size of the data retrieved is 0, an Intent will be used to redirect the user to another page to show that there is no donation history available for display. Whereas on the other hand, if there is data being retrieved, the ArrayList<Donation> object will be specified as a data source in the custom adapter which will subsequently be displayed in the listview.

## Firebase

****

*Figure 6: Firebase codes*

We have also incorporated and integrated the use of Firebase into our application. Having a firebase provides many benefits as it is able to act as a realtime database that provides synchronised data across various parties. In other words, when a person goes offline, the data still remains available in real time. As a result, integrating the Firebase Real-time Database SDKs into our application allows us to create rich, collaborative and secure data with the users of our application and reduces the reliance on complex backend services to manage our database.

For instance with Firebase, now each donation history is able to be stored in the fire base and can be retrieved without resetting. This is previously not achievable without the help of Firebase because there was no synchronised platform to store the donation history that we have created in our individual laptops. Firebase allows the user to push data into a real time database and allows users to retrieve data when it is changed. This allows records to be stored in real time and not stored into an offline database. An online real time database allows users to retrieve those same records even after the user has closed and exited the application.

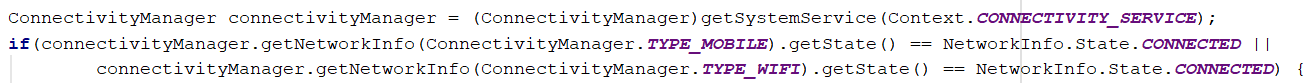


Figure 7: check internet connection codes

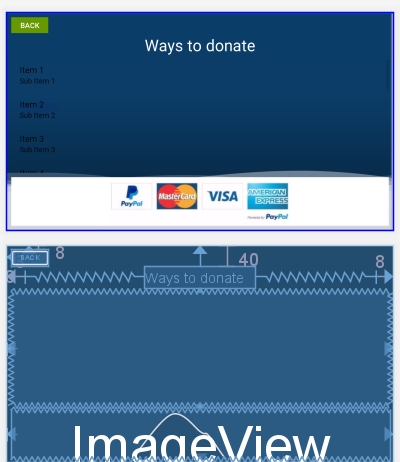
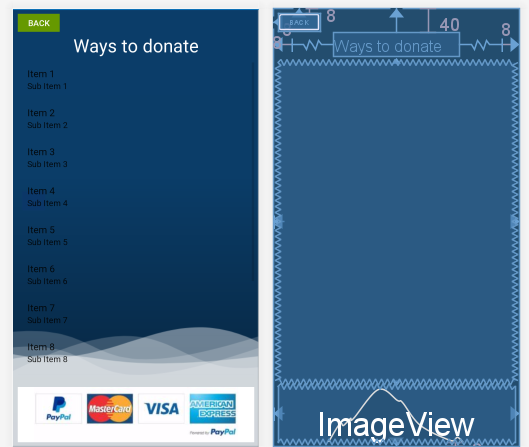
This set of codes allows the application to check if the current user has an active mobile connection. If the user does, the donation history will be pulled from the firebase if not, the donation history will be pulled from the sqllite database.

# Requirements

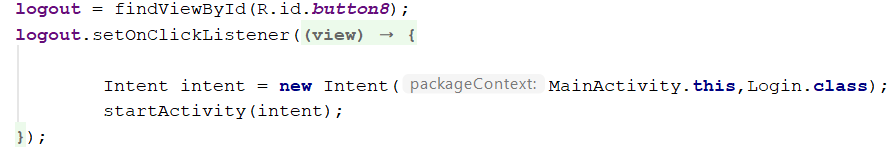
## Manifest Customization



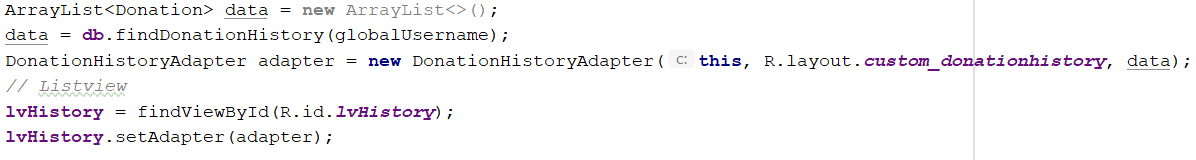
## Responsive Layout



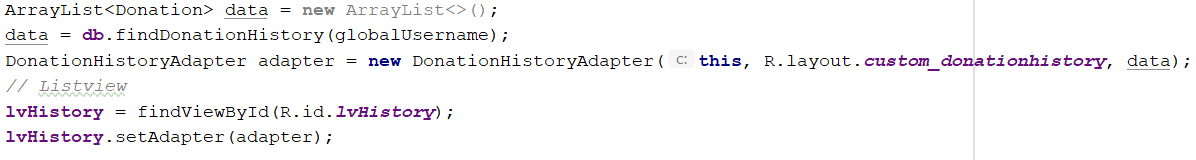
## Intents



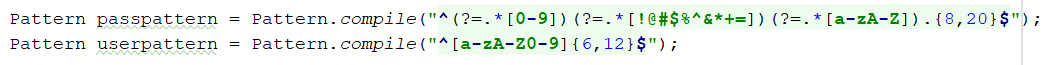
## ListView / RecyclerView



## Persistent Memory (SharedPreferences / SQLite)



## Regular Expression



The screenshots above are just one of the many examples that we have implemented in the application.