Zahid Amaar

Student ID: Bh19go

Registration Number: 1349107102

Programme: Computer Science

University of Sunderland

Contents

1. Functionality

For this module the given task was to create a professional notes management application. During the first assessment the task was to create a design for the application which would have the required activities set up and allow for navigation. However during this professional practice week the functionality was to be expanded upon, the application was to be at a stage where it could have the following functionality mentioned in this report, This report will cover the functionality of the final application, and the testing conducted for the application to compare expected results and final results to allow for an accurate evaluation on how well this assessment, and the application have progressed.

1.1 Home Screen

The point of Entry for the application is the home screen, this is a screen which will allow for the user to login to the application, or to select register, which will send the user to a registration activity. The philosophy followed when creating the application and the home screen specifically was to create a warm, familiar and welcoming design, that akin to the likes of social media. The design uses warm colours such as purples and pinks, along with rounded buttons, these are created using custom layout files, which are then called into the main layout file as a background. The core functionality of the home screen is to allow for users who are logged out or have freshly installed the app to navigate to creating an account, or to login, which will automatically send the user to their Task or note list. Logging in is essential as it allows for separation of users items.

The Home screen login function uses FireStore to allow for the creation of user accounts, this is done by FireStore's authentication services. This allows for users to login using Email addresses and passwords. Once a user has created an account the email and the user id for that account is also sent to a collection in the FireStore database. This is done simply to allow for querying which is used to show only the data to specific users who uploaded that data. This users notes from being public view and access. Extra validation has been added to the login system, which prevents users from entering empty email and passwords when registering or logging in, the application also has validation for passwords, requiring at least 6 characters, additionally if the same email is used to register an account the application will inform the user. Once a user has logged in the application will send them to the main activity, which even upon closing the application, if the user did not log out they will still be signed in to the application, and the application can check for the current user, which if the user is not null, then the application will send them straight to their task list.

The below images show validation which has been added to the login activity for the app

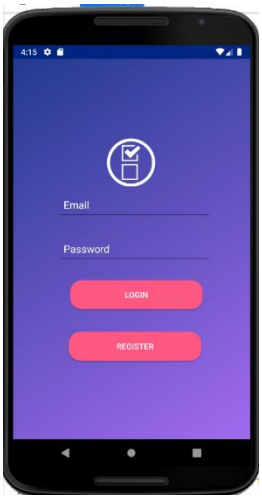

Fig 1. Home Screen

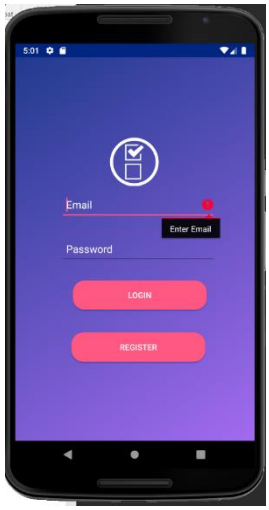

Fig2. No email entered

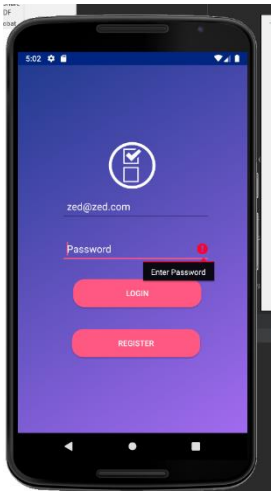

Fig 3. No Pw entered

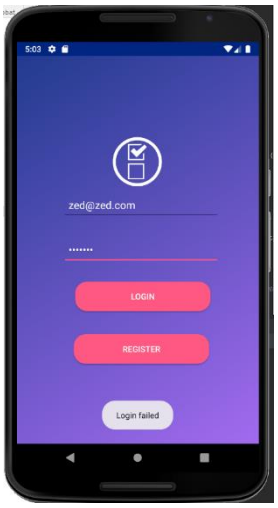

Fig 4. Incorrect details

1.2 User registration

For user registration as mentioned in the previous section FireStore authentication was used, this was used in both the login and registration activities as it simplifies logins, however, when an account is created, a collection of users within the FireStore database is also updated with the new account information, this includes, user email, and their user id, however, passwords have been exhumed from being sent to the database as it is best practice to avoid saving these into the database. Once saved the information can be used for a query which is why it is important to store the user account as a collection. Similar to the login activity, the registration activity uses validation, which disallows users from creating accounts with the same email, empty passwords, passwords which are below 6 characters, and valid email addresses. The below images show the validation and the account creation process.

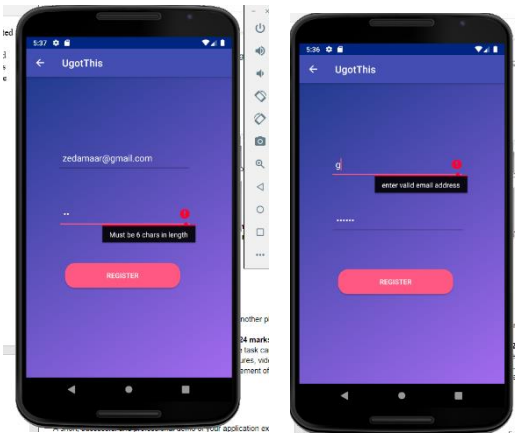

Fig 5. Password too short

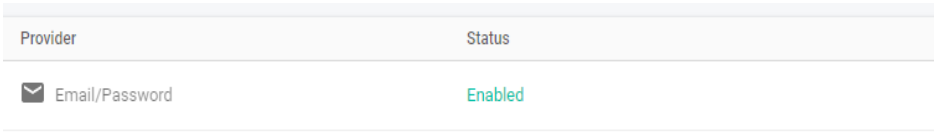

Fig 6. FireStore Authentication enabled

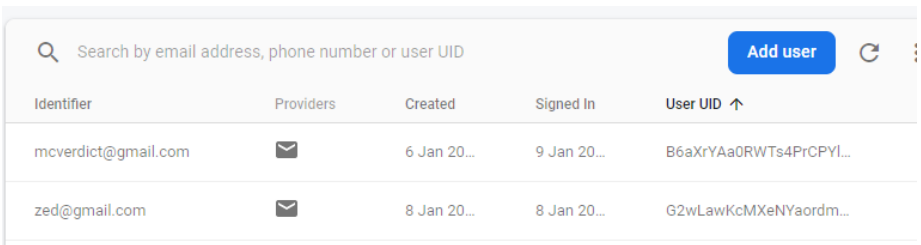

Fig 7. FireStore Accounts shown

To ensure account creation works, it is important to enable authentication in FireStore, the above image shows the FireStore authentication for this application.

1.3 Managing Tasks

One of the most important features of the application was to allow for users to manage their tasks, this included editing and deleting their tasks once they were added to the application. The application completed during this professional practice week had functionality which would allow for deletion along with the ability to edit tasks within the application once created to fulfil these requirements, but to also make ease of use and accessibility of the application more user friends were integrated with gestures. For example the deletion of tasks can be done with a simple swipe to the right, and the action of marking tasks as complete can be done with a simple swipe to the left.

In addition a feature including a shake was implemented allowing for users to shake their device to go to the add task activity. These simple ease of use changes were the additional feature selected for this application as they make the application much more user friendly.

1.3.1 Delete Tasks

The deletion of tasks can be achieved in this application by swiping to the right, however, the application allows for the user to click on the task, which brings up a context menu, once clicked the position of the recycler view is passed, upon clicking delete task, the task in the given position in the view is deleted from the FireStore database, which is then updated real time in the recycler view. This essentially makes a much friendly and cleaner experience for the user as swiping to delete tasks can be effortless and responsive.

Below are screenshots showing these functions:

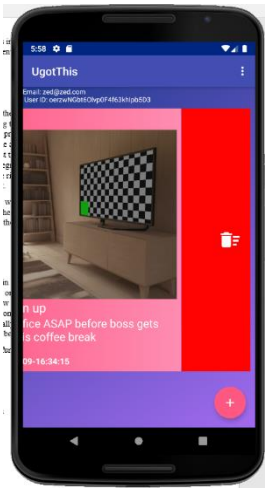

Fig 8. Showing Swipe to delete

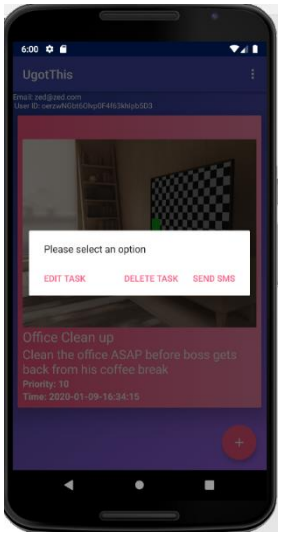

Fig 9. Edit menu

1.3.2 Edit Tasks

To allow users to edit tasks, there has been an activity implemented, this activity pulls in the task object from the FireStore database, which can then be overwritten. To edit a task, the user must first click on the desired task, which they would want to edit, once done a context menu appears, which allows for the user to select Edit task, once selected the user is sent to a new activity, where they change the fields for description, title, priority, and upload a new image. Each of these fields have been enforced with validation which requires entry of information and a new image to be uploaded when editing a task. Below are screenshots showing the edit activity:

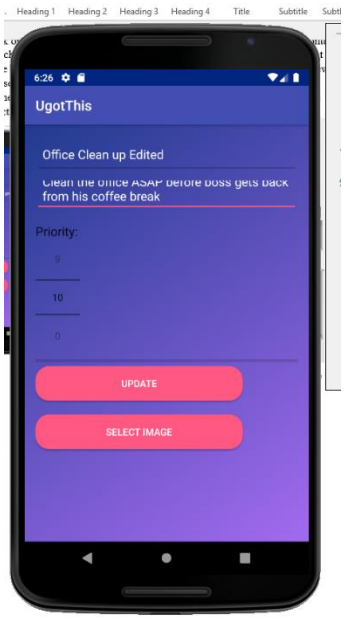

Fig 10.

Getting desired note in the edit activity

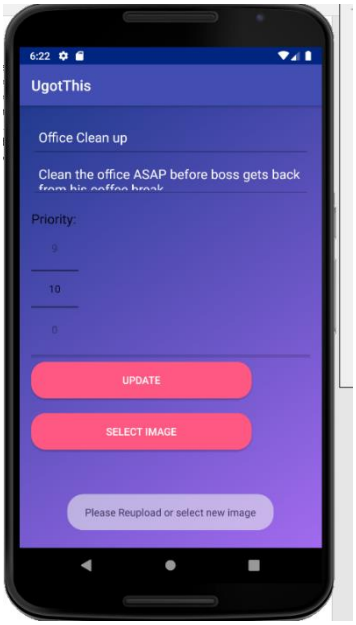

Fig 11. Showing new image required for update

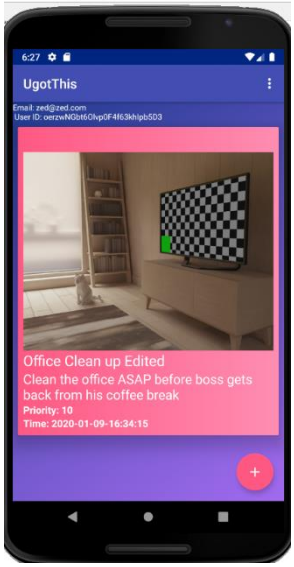

Fig 12. Showing Task has been edited

1.3.3 Mark As complete tasks

Another important core functionality for the application was to ensure that it is able to mark tasks as complete, this was done by creating a new collection in the FireStore database. A new adapter is created which then pulls in the information from the completed tasks collection and queries it against the current user to ensure only the completed tasks for the logged in user are shown.

To mark a task as completed the user must swipe to the left, this will then send the task to a completed tasks activity where the user can view their completed tasks and delete them from there, this activity can be viewed by going into the top right menu and clicking completed tasks. The below screenshots show the process of completing a task and the completed task activity where tasks can be viewed:

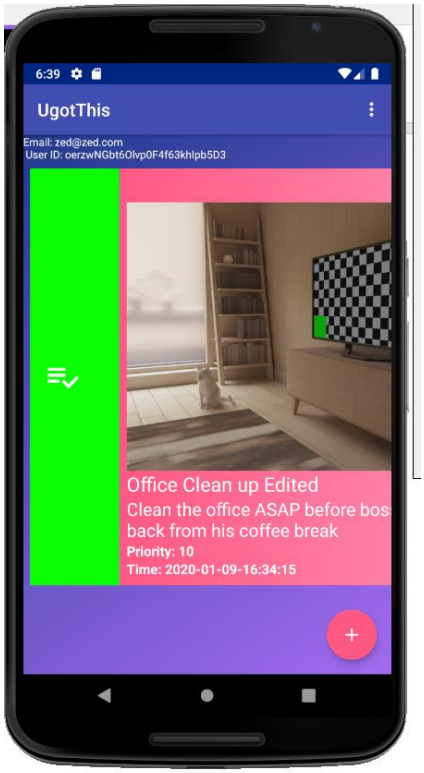

Fig 13. Shows image being swiped as complete

Fug 14. Shows Completed task in completed activity

1.4 Creating Tasks

One of the most important core features of this app is to allow for the creation of tasks, because of this a task creation activity has been created for this application, without this the application would not be able to fulfil its requirements. When the user is logged in and clicks on the floating action button they are directed to the add task activity, which then allows for users to add a title, description, set a priority which the tasks are ordered by, and an image, all

of which have validation inserted. Once a task is created it is uploaded to the collection, which is read by the adapter in main activity.

Below are screenshots showing how a task can be added to the application:

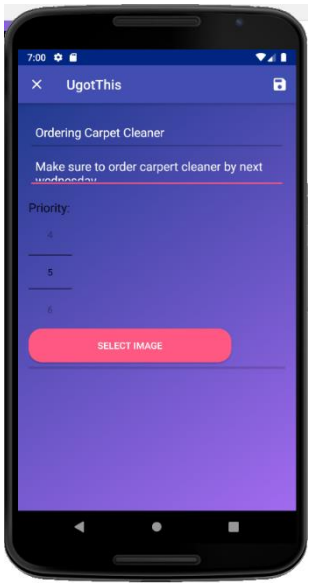

Fig 15 showing adding task

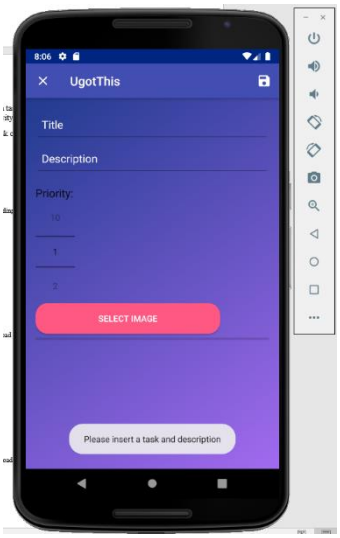

Fig 20.

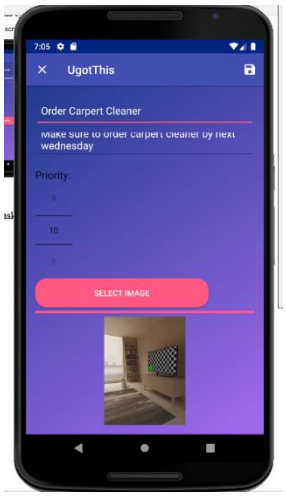

Fig 16. Showing upload of task in progress

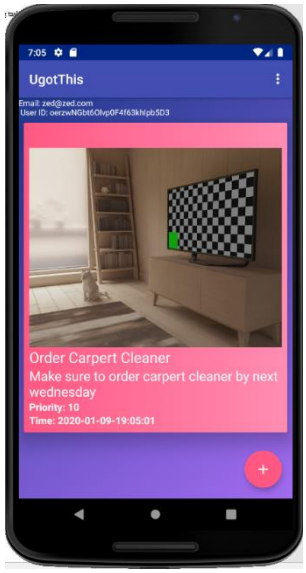

Fig 17. Shows uploaded task in main activity

1.5 Task Delegation

For this assessment one of the core functionalities was to be able to delegate and send a task as an sms message, this can be done by clicking on the desired task and then selecting the send sms option. Below are screenshots showing this process.

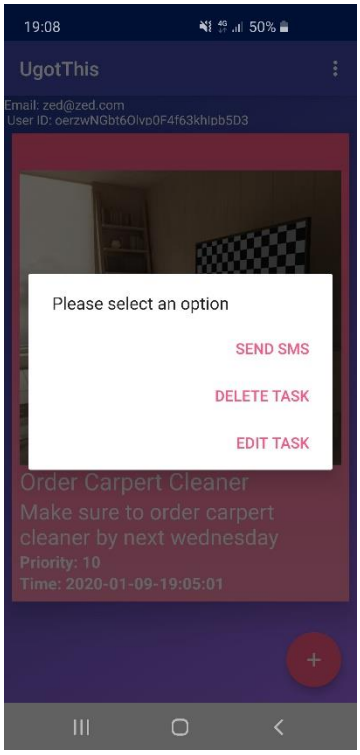

Fig 18, showing the selection of send sms option

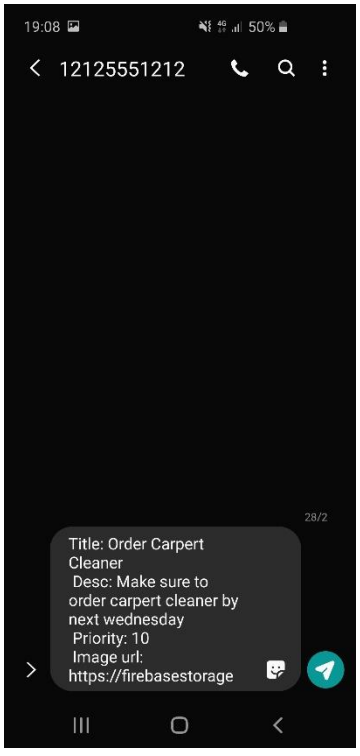

Fig 19. Showing task being sent to sms activity

2. Test Strategy

For this application the asserted test strategy was based off of the application requirements, essentially, taking into account the core functionality for the application, it was important to come up with a strategy which would allow for testing of not only the core features, but the chosen desired feature, in this case, gestures. The strategy was fragmented in the sense that each core functionality was thoroughly tested. For example when testing the register or login functionality, it was important to test for empty fields for both email and password. The following section includes a testing table with the corresponding evidence.

3. Test Results

Test	Predicted result	Actual result	Evidence	Pass or Fail
Opening application	Application opens to home screen	Application opens to home screen	Fig 1.	p
Register account with invalid email	Tells user to fill in valid email	Tells user to fill in valid email	Fig 5.	p
Register with password field null	Shows error in edit text box	Shows error in edit text box	Fig 5.	P
Register with email field null	Shows error in edit text box	Shows error in edit text box	Fig 5.	P
Login with incorrect details	Shows toast incorrect details	Shows toast incorrect details	Fig 4.	P
Login with Password field null	Shows error in edit text box	Shows error in edit text box	Fig 3.	P
Login with Email field null	Shows error in edit text box	Shows error in edit text box	Fig 2.	P
Register with password below 6 characters	Shows error in edit text box	Shows error in edit text box	Fig 5.	P
Adding task with no information	Shows toast fill in all fields	Shows toast fill in all fields	Fig 20.	P
Updating task with no information	Shows toast fill in all fields	Shows toast fill in all fields	Fig 11.	P
Adding task with all information valid	Successfully uploads task	Successfully uploads task	Fig 16. Fig 17	P
Updating task without reuploading or selecting new image	Shows toast fill in all fields	Shows toast fill in all fields	Fig 11.	P
Swipe left to save task	Saves task to completed task activity	Saves task to completed task activity	Fig 13.	P
Swipe right to delete task	Deletes swiped task	Deletes swiped task	Fig 8.	P
Logs in successfully with correct information	Logs user in and sends them to the task list activity	Logs user in and sends them to the task list activity	Fig 12.	P
When clicked item in recycler view shows context menu	Shows context menu	Shows context menu	Fig 18.	P

Clicking delete on context menu deletes item which was clicked on	Deletes task	Deletes task	Fig 9.	P
Clicking send sms sends task to sms message intent	Sends to sms message with task information	Sends to sms message with task information	Fig 19.	P
Clicking edit on context menu sends user to edit activity with note information filled in	Sends user to edit activity with all information filled in	Sends user to edit activity with all information filled in	Fig 9. Fig 10. Fig 11.	P
Update button sends toast when details not all filled in.	Sends toast when not all details filled to fill in details	Sends toast when not all details filled to fill in details	Fig 11.	P
Update button updates task when all fields filled in	Updates the task	Updates the task	Fig 12.	P

4. Evaluation

`Overall for this application during the course of this professional practice week there were many things which went both well and things which could have been improved upon. The overall outcome I believe was positive, and the final application was a well-designed, well-functioning application.

The best aspects of this application were both the core functions and the design. The final application is able to fulfil the requirements for all of the core functions, along with this, each of these functions have a streamlined and easy process to follow, which is made a lot more accessible by the additional feature chosen, which was gesture controls.

One of the things which could have been improved during this professional practice week was the incorporation of multi touch, this was a feature which was hard to find a reason to implement however, it was an additional desired function.

1. References

Android (2019) - Design for Android- https://developer.android.com/design (Accessed on: 09/01/2020)

GitHub- User: probelalkhan (2018) 'https://github.com/probelalkhan/android-cloud-firestore-example' - https://github.com/probelalkhan/android-cloud-firestore-example/commits/master (Accessed on: 09/01/2020)

Codinginflow (2019) 'cloud-firestore'- https://codinginflow.com/tutorials/android/cloud-firestore (Accessed on: 09/01/2020)

YouTube- User: Simplified Coding (2019) '#3 Firestore Android Tutorial - Updating Data' - https://www.youtube.com/watch?v=VHLj-ndsZnY (Accessed on: 04/01/2020)