SIT708 – Assessment 3 Sprint2: Project Handover -Written Report

Utsav Sharma

218528291

Abstract-

This document represents the final stage of my unit and summarises my android mobile application “Memento Puzzler’s” working. The document focuses on the applications different functionalities and explains how the whole application works. I’ve done my coding on Android Studio and used Kotlin as my programming language. I’ve made a lot of changes in my project as compared to my initial Project Plan. I’ve gone through various different tutorials to come up with different features for my android application and came up with a few. One of those features include, creating your own custom game with the help of Firebase and its data storage features.

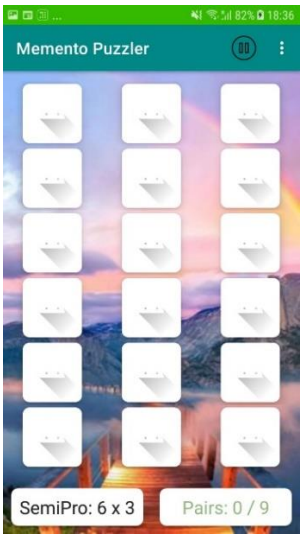
As per my initial project proposal, many of the pages I had created in my Figma prototype have not made my final application. This includes pages such as the login and registration page, share your score page. The reason for them not making the final application is because I designed a custom game feature, which took most of my time for coding it. At the end I had no time left to complete these pages, which were also a common feature in many applications.

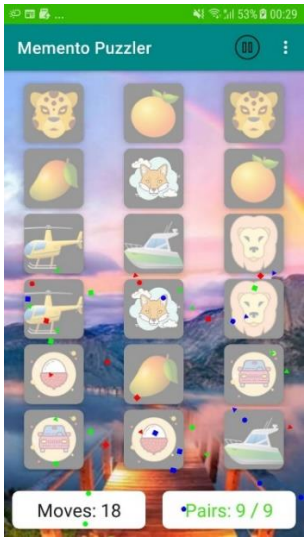
Overall Description-

Game Manual-

- The user clicks on the application’s icon to open the game.
- The first page of the game is the Beginners Level. The player simply has to tap on the cards and match each pair in order to win the game.
- The game has three different gaming levels. The Beginner Level, this level consists of 4 pairs and in total 8 cards. The second level is the Semi Pro Level, this consists of 9 pairs and in total 18 cards. The third and most difficult level is the Professional Level, this consists of 12 pairs and in total 24 cards.
- A player can play these different levels through the Game Centre feature in the menu section. The images used in the game were used I the initial prototype design of the game and can be found in the drawable section.
- While playing, the number of moves the player makes and the number of pairs the player matches is accounted as their score. Interestingly, one move is counted once the player taps on two cards and reveals them and it doesn’t matter if those two cards are same or not. Another cool feature of the game is the score of the game changes in colour when the player is about to win. In short, it represents the progress the player has throughout the game.
- A pause button is situated next to the menu section, which on clicking pauses the game and ask the player whether they want to quit their game or not.
- The main feature of this game is the “Create Your Own Custom Game” feature. The player can simply create a game of any level and use images from their own gallery. The application asks the players permission for accessing their gallery. On choosing the images, the player can then name the game and save it.

- The next feature is the “Download Your Custom Game”. Once the player has created their custom game, they can then download that particular game and play it.
- Two snapshots are attached below. The first one represents the Semi Pro level game and the second one represents the ending of the game.

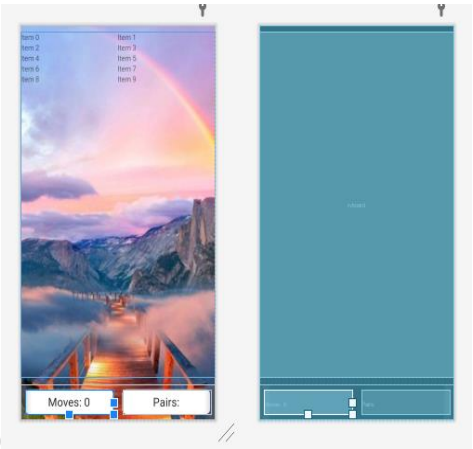




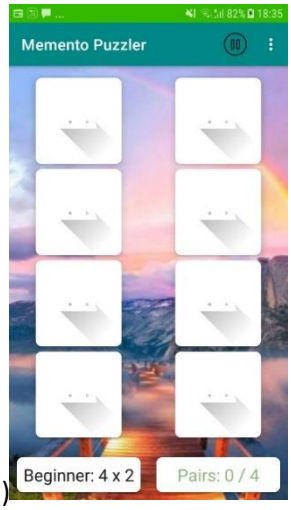
Explanation of coding concepts and design-

The first page of my application is the Beginner’s Level gaming page. The design for this page was created under file “activity_main.xml”. It consists of a Constraint Layout with an image in its background, the same image was used in my Figma design. The Constraint Layout is followed by a Linear Layout ‘llgame’ (its id), followed by two CardViews which themselves consist of two separate TextViews, ‘tvnummoves’ and ‘tvnumpairs’. The tvnummoves counts the number of moves the player makes and the tvnumpairs count the number of pairs the player matched. Lastly, a Recycler View ‘rv_board’ is used for holding each of our gaming cards. Two snapshots are provided below, (1) Design of my activity_main.xml file and (2) Screenshot of my running application.

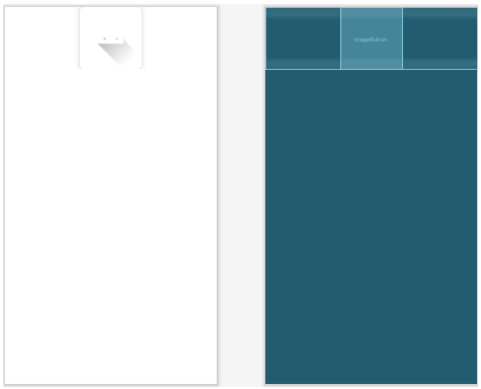
(1)



(2)



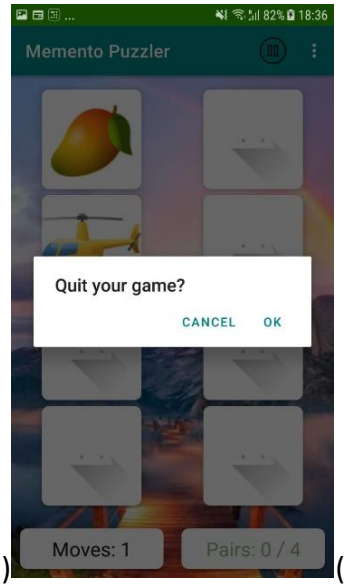
The cards or hidden box's, which contain hidden images are the essence of my application. They were created under file "hidden_card.xml". It consists of a Linear Layout, which has a Card View under which an Image Button 'imageButton'. Image Button is simply a button with an image, which can easily be clicked by the user. A snapshot of my hidden_card.xml file's design view is attached below. The same design can be found on the cards of the actual game.

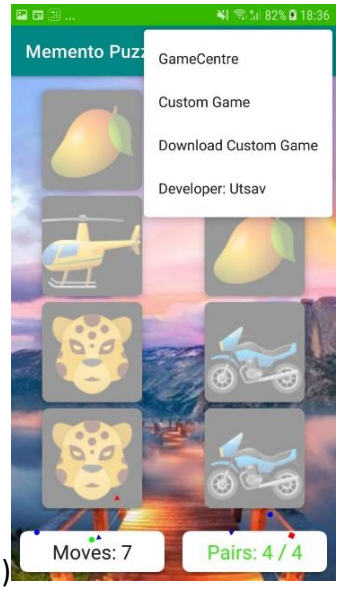


The next important part of my application is the menu section. It consists of different features my application provides. I created a separate resource file named menu and a new layout file "homepage.xml". The first option is the pause button. The pause option can be found at the top of the menu bar, but it has a different function than its name. On clicking the pause button, the player is asked whether they want to continue playing their game or quit the game. The remaining four features consist of the Game Centre, Create your Custom Game, Download your Custom Game and the Developer page.

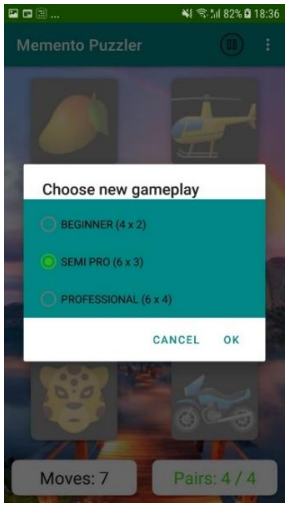
Interestingly, each of the features in the menu section are hidden features except for the Pause Button, as they were creating overloading problems in my code. Three snapshots are provided below, (1) Design of the homepage.xml file, (2) Screenshot of pause button running in the application and (3) Screenshot of the remaining features being displayed.

(1) 

(2) 

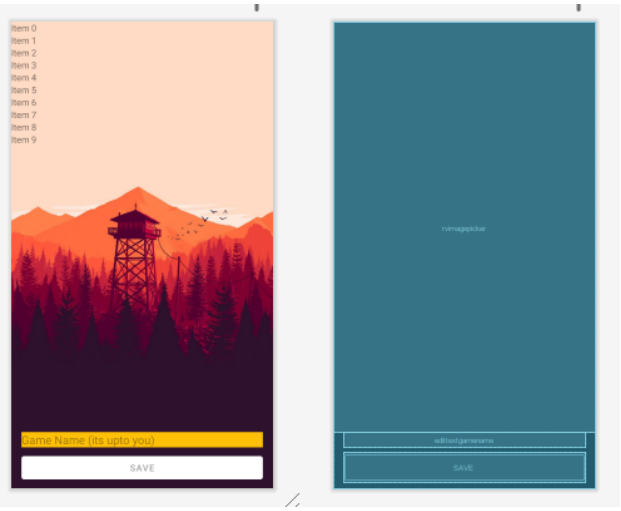
(3) 

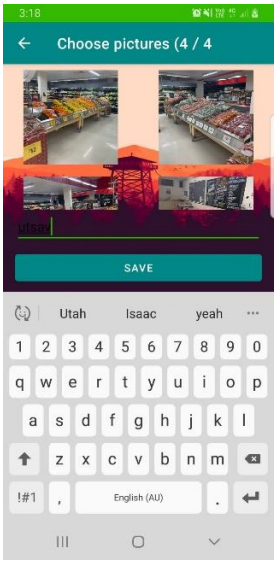
Moving onto the different features in the menu section. The first feature is the Game Centre page. It contains the three levels this game provides. The default level is the Beginner's Level. I created a "dialogboardsize.xml" file in the layout folder to implement this feature. It has a Constraint Layout, with three radio buttons for each level. Radio Buttons allow the user to select one option from a set of options, which are also visible. A snapshot of my xml file's Design view and a screenshot from my running application is attached below.



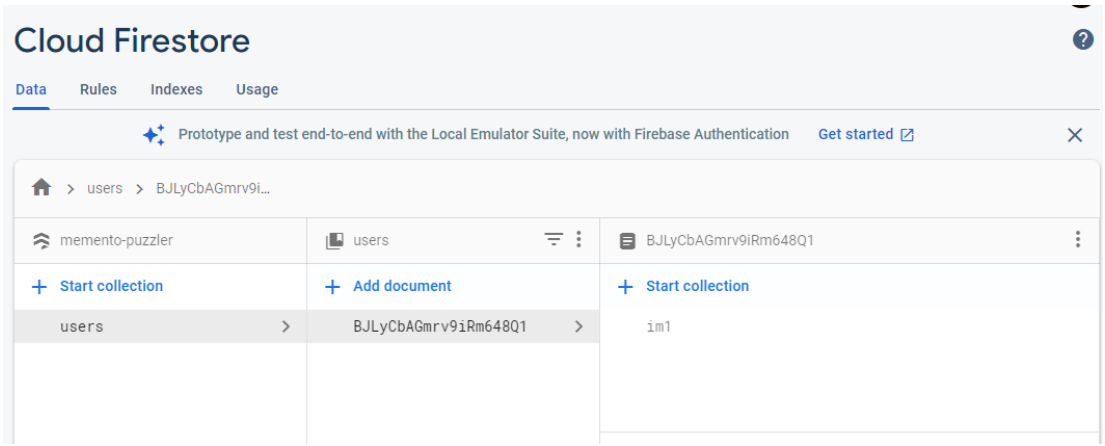
The next feature is the "Create your own custom game". I had to follow numerous tutorials to get this right and even learnt a few topics, which weren't a part of my course. The "Create Your Custom Game" page firstly asks for the user's permisson for allowing access to gallery. Once granted access, the user can select random images from their gallery and use those images for their custom game. Then the user has to enter a name for their game and save it. Firstly, I created a new Empty Activity in my project and named it Create Activity. I then declared the Intent function in my main activity file. The intent function is used for navigating through different pages of an application. In the newly created "Creation" file, a support action bar command is passed to ask the user about the number of images they require for the game and a layout file file "activity_create.xml" is declared. This layout file is where the whole custom game page is designed.

Constraint Layout is present by default. Then a button with id 'buttonsave' is added at the bottom, which acts as a Save Button for the page. A recycler view 'rvimagepicker' is used for adding images. Lastly, an EditText 'edittextgamename' is used for naming the custom game. The design view of my activity_create file and running application is attached below. A progress bar is also situated below the Save button, but is hidden and can only be seen when the user custom game creation takes place. A new file named Adapter_ImageSelection is created, which defines an interface ImageCLickListener and declares a function in it, so that the user can click on the empty cards on choose random images from their gallery.

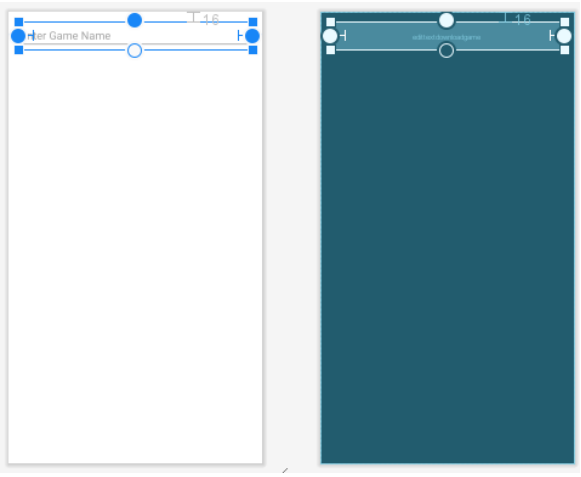




The next step for creating the custom game is to save the images in a cloud storage system, and I've used Firebase. It is the perfect tool as it allows sharing of data between applications. I created a new folder in the Firebase console and added my projects package name. I then downloaded the google-services json file and made the necessary changes in the build.gradle file. In the build.gradle app file, under the dependencies section, I added firebase, firestore, analytics, remoteConfig and crashlytics libraries, which was a new learning for me.



Moving on to the last feature of my game is the "Download Custom Game". A layout file "dialogdownloadbox.xml" file was created to design this page. Once the user creates their custom game, the game is saved in the database and them the user can download that particular game and play it. If the custom game isn't saved in the database, then a message appears at the bottom indicating that no such game exists. The design of this page consists of a Constraint Layout and a EditText 'edittextdownloadgame'. A snapshot of the page's design view and application view is attached below.



Issues faced during application development-

- In the beginning, I wasn't sure on how to begin with my project. I went through a few tutorials, **https://www.youtube.com/watch?v=BGvjScKcW1s** and **https://www.youtube.com/watch?v=U4Wtjewy7EY** to get started with my project.
- I made numerous changes in my final project from my intial prototype. One of the major changes was using Firebase as my cloud storage feature. Through firebase I created a custom game feature and used its firestore, analytics, etc libraries to run my program. The tutorials I followed for getting this done were **https://firebase.google.com/docs/reference/kotlin/com/google/firebase/firestore/FirebaseFirestore** , **https://www.youtube.com/watch?v=uyCZIN_gDIw** , **https://www.youtube.com/watch?v=gqIWrNitbbk** , **https://www.youtube.com/watch?v=_xIWkCJZCu0** and a few more.
- When I finished my coding, I kept getting this error in my Manifest file, "intent filter in android manifest not working" . My Creation.kt file had an error in which, the function placeclickerholder() wasn't working because I missed closing it with a curly bracket. On correcting the error, my application started working again.
- Lastly, the build.gradle file asked for implementing Multidex library because my whole project had surpassed 64K methods. I had no prior knowledge about Multidex, but my code started working after I implemented it.

Future Work-

- I will work on creating a user login and registration page.
- In the gaming screens, I will add a timer as well in order to clock the player's gaming time.
- The UI/UX of the game still needs some changes in it.
- The custom game portal will be availaible for each player.
- More levels will be added in the game as well.