Callum McLaughlin

24678805  Games Programming

Mobile Apps Coursework 2

Contents

[Overview 2](#_Toc71721749)

[View Implementation 2](#_Toc71721750)

[Data and Storage Architecture 3](#_Toc71721751)

[Classes 3](#_Toc71721752)

[Reflection 4](#_Toc71721753)

[Testing 5](#_Toc71721754)

[User Manual 8](#_Toc71721755)

[Bibliography 12](#_Toc71721756)

# Overview

Although I have been able to achieve parts of the solution, I have not been able to implement a complete solution or one that I am happy with, I only have parts of it that are not connected cohesively. I was able create each activity quite easily and the layout for each remains consistent, I created a Date picker so under the registration page you can click on the button “Select Date” beside the DOB text input and a calendar will pop up. When you click on a date it will input that date in shorthand text format in the text box referring to DOB. I conducted some research for this and found videos such as “Android Beginner Tutorial #25 - DatePicker Dialog [Choosing a Date from a Dialog Pop-Up]”, (CodingWithMitch, 2017) and learning materials on a website called android developer very useful and insightful, (Android Developer, 2020).

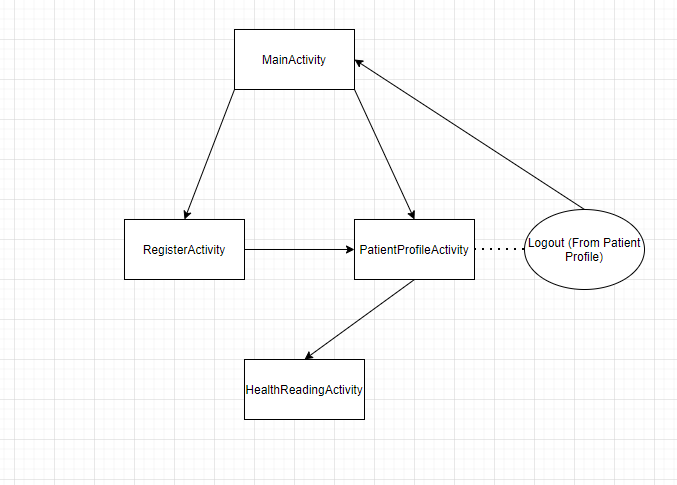
At the start of the project I had decided to use Firebase as the data storage model and I did some research on how it could be implemented into my project, videos such as “Store Data in Firebase real time database in Android Studio - Android Firebase # 1 - 2020”, (Tea, 2020) and the official firebase documentation.

I was also able to implement the data storage requirement using SQLite but before I decided to go ahead and work on implementing the database, I did some background research on SQLite and the number of ways it can be used to achieve what I wanted. I found a tutorial that was quite useful in explaining how the data is written and read from the database, (Simplified Coding, 2017) I created a database programmatically and was able to get data stored on the database through the registration page and I can also read back from the database as shown through the patient profile page. I have also implemented the Add Health Readings page however it is not as intended as the patient number is not automatically inserted based on the logged in user. I could not get the login process working properly however I have completed code which I believe should work. The logout function was also implemented and takes the user back to the main menu login screen.

# View Implementation

I was able to create the applications views by creating multiple different activities and linking them all together through button events. For example when the register new user button is clicked on the main login page, that starts the registerActivity.

As shown in the diagram below you can see that the mainActivity is the login page, then we have the registerActivity which links to that and the profileActivity. If the user is logging in, when they press the login button, they will go straight to the profile page, however if it is a new user, they will register first then go to the profile page. From the profile page the user can then access the Health Readings page.



# Data and Storage Architecture

The first iteration of my implementation used Firebase, but I had many problems setting it up, so I decided to change to SQLite around 3 weeks into the project. I believe that changing the design while developing is not always the best idea as it is better to be prepared for the work that is about to be done but when it is necessary, I believe it is acceptable as long as the final product is not impacted.

I used similar classes to the ones shown in the lectures in week 8 which were used to create a database programmatically. I used SQLite and through creating the database inside the project using a MyDBHelper class to facilitate this. I am able to store data from the text inputs by assigned the text to variables and then passing those variables through an “insertPatient” or “insertHealthReading” method which inserts the values int the relevant table in the database.

# Classes

The class design was much different from what I had initially designed but more so because I had to reduce the scope of the project in order to get some functionality working before the deadline. The classes that were used in the project on top of the other activities were the InputValidation class and the MyDBHelper class.

Although the InputValidation class was only used a few times as the other parts of the project were not implemented. The InputValidation class had methods that check if editText are filled, if editText is an email and if editTextInput matches another editTextInput. There is also a hideKeyboard method.

The MyDBHelper class contains everything needed to create the database. Included methods for inserting patients and health readings into the database, retrieving all patients and health readings as well as a method to retrieve specific records. There is also a checkUser method which was created for the login process and although I believe it should work, the login process crashed when the login button is clicked and I was no able to fully implement it properly although all the code is there, an error is there somewhere that I could not fix.

# Reflection

Although this module was challenging and I enjoy being tested, I did not enjoy Mobile App Development. At the end of the day I want to do game development and I do not believe I will be doing it on mobile as I do not enjoy it and it just does not click for me like regular programming does or C# Unity scripting. Having said that, I have learnt a fair bit from this module, I know understand how mobile applications are created and the various data storage methods they used as well as the software that is used to create them, using an emulator was also a first for me as I would usually use my phone on previous projects during my Software Development Degree.

The major challenge of the project was the read and write to the database, I had many problems with it and even when I would fix it, I would edit my code then elsewhere and it would not be working again so I found myself constantly fixing that which led to it being my biggest challenge throughout the project.

If I had more time, I would try and implement the Display Health Readings activity and link the health readings to each user automatically as well as having the login page work as planned. The best idea I had during this task was creating the date picker button which displayed a calendar, and the chosen date is converted to the textInput for DOB on the user registration page.

# Testing

|  |  |
| --- | --- |
| **Use Case Title** | ***User Registration*** |
| **Pre-conditions** | User is not registered. |
| **Assumptions** | * Device is connected to the internet |
| **Success Criteria** | User can successfully register on the mobile app |
| **Failure Criteria** | User is unable to register on the app. |
| **User Role** | New App User |
| **Steps** | 1. User starts the app and then the login screen appears. 2. User clicks on register new user and follows steps |
| **Post-conditions** | User details are stored within SQLite Database |
| **Test Status** | Pass |
| **Risks and constraints** | * None |

|  |  |
| --- | --- |
| **Use Case Title** | ***User Details are stored*** |
| **Pre-conditions** | User has signed in |
| **Assumptions** | * User is registered. * User is signed in |
| **Success Criteria** | Details are stored on database |
| **Failure Criteria** | Details not stored on database |
| **User Role** | Existing App User |
| **Steps** | 1. User starts the app. 2. Users’ details are shown on profile screen |
| **Post-conditions** | User details are stored within mobile app |
| **Test Status** | Pass |
| **Risks and constraints** | * None |

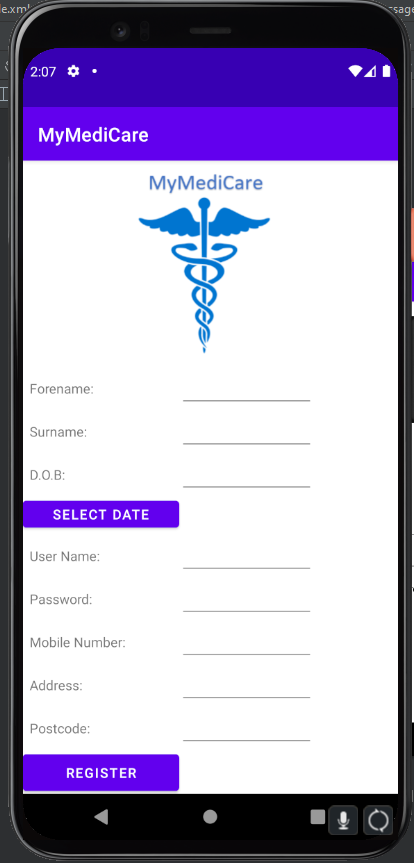
|  |  |
| --- | --- |
| **Use Case Title** | ***User can send SMS to GP*** |
| **Pre-conditions** | User has signed in |
| **Assumptions** | * User is registered. * User is signed in |
| **Success Criteria** | Message is received by test device |
| **Failure Criteria** | Message is not received by test device |
| **User Role** | Existing App User |
| **Steps** | 1. User starts the app. 2. User logs in 3. User selects the “SMS GP” button 4. User enters message and clicks send. 5. Message is received by test device/ toast message shown |
| **Post-conditions** | None |
| **Test Status** | Pass |
| **Risks and constraints** | * None |

|  |  |
| --- | --- |
| **Use Case Title** | ***User customises background colour*** |
| **Pre-conditions** | User has signed in |
| **Assumptions** | * Device is connected to the internet. * User is registered. * User is signed in |
| **Success Criteria** | User can change theme from light to dark |
| **Failure Criteria** | User cannot change theme from light to dark |
| **User Role** | Existing App User |
| **Steps** | 1. User starts the app. 2. User logs in 3. User selects settings button. 4. User selects new colour for background. 5. Application theme changes colour |
| **Post-conditions** | Colour changes to selected colour |
| **Test Status** | Pass |
| **Risks and constraints** | * Mobile device might not be connected to the internet. |

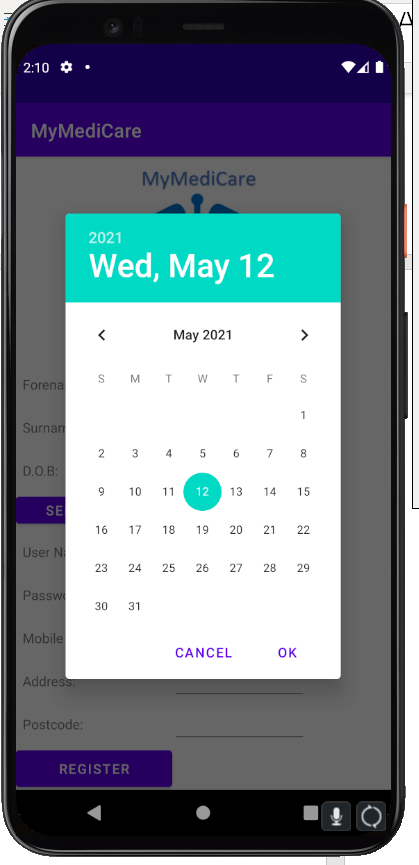
|  |  |
| --- | --- |
| **Use Case Title** | ***Check that all navigation buttons work*** |
| **Pre-conditions** | User has signed in |
| **Assumptions** | * User is registered. * User is signed in |
| **Success Criteria** | Each button works as intended |
| **Failure Criteria** | Faults found in navigation buttons |
| **User Role** | Existing App User |
| **Steps** | 1. User starts the app. 2. User logs in 3. User selects settings button. 4. User selects add health reading. 5. User selects send GP message |
| **Post-conditions** | Each button takes the user to the right page |
| **Test Status** | Pass |
| **Risks and constraints** | * None |

# User Manual

The Application is quite straightforward and easy to use. The login process I not implemented completely and crashes upon pressing the login button. To register a new user click on the register button on the main start-up page shown on the left.



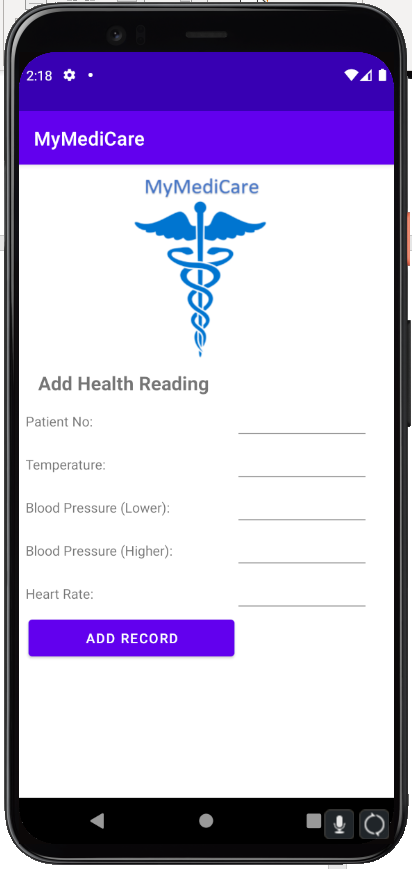
You will be taken to the registrations screen where you can enter your personal details and click on the submit button to register your account. You will then be taken to the account page.



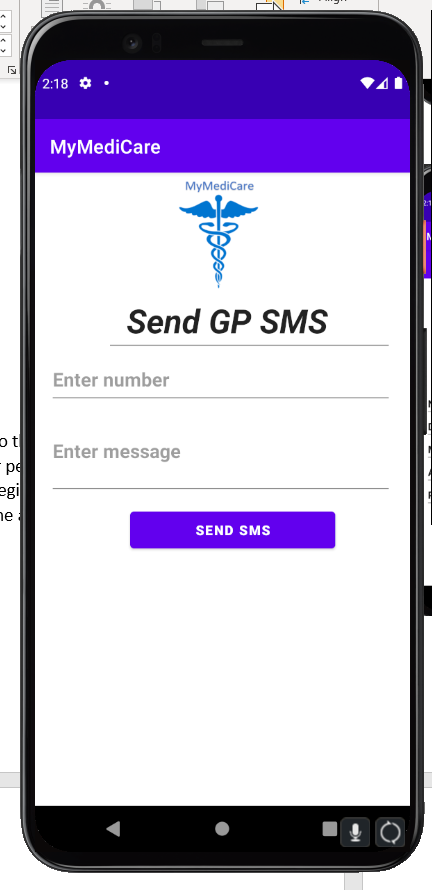
Also on the registration page there is a date picker to help with selecting your Date of Birth. To access it simply click on the “Select Date button” and a calendar will pop up. Pick you DOB and the date will be auto filled into the adjacent text box.



Once you have registered you will be taken to the patient profile page where you can see your personal details displayed and also add health readings, contact GP and also change the theme of the toolbar.

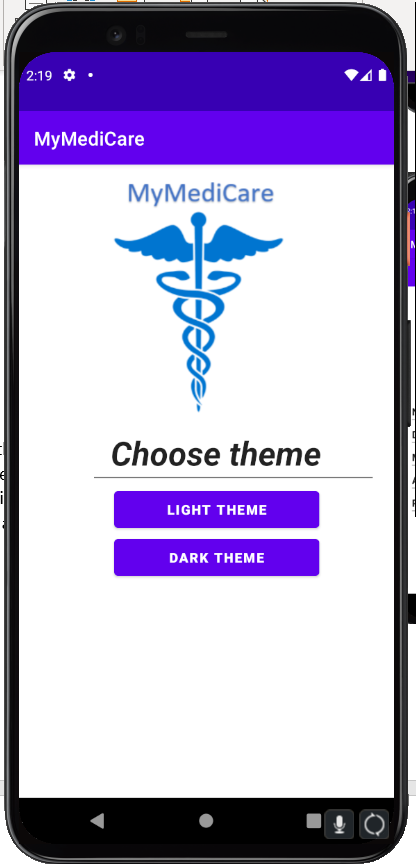


If you click on the “Add Health Reading” button on the profile page it will take you to this screen where you can enter your health details and add it to the database.



If you click on the “Contact GP” button on the profile page it will take you to this screen where you can send a text message to you GP by entering the phone number, adding the message and clicking “Send SMS”.

If you click on the “Change Theme” button on the profile page it will take you to this screen where you can change the color of the toolbar on the profile screen.



# Bibliography

Android Developer, 2020. *Developer.Android.* [Online]   
[Accessed 06 05 2021].

CodingWithMitch, 2017. *Youtube.* [Online]   
Available at: https://www.youtube.com/watch?v=hwe1abDO2Ag  
[Accessed 12 05 2021].

Simplified Coding, 2017. *YouTube.* [Online]   
Available at: https://www.youtube.com/watch?v=Fnf3PImIKa0  
[Accessed 12 05 2021].

Tea, C. W., 2020. *YouTube.* [Online]   
Available at: https://www.youtube.com/watch?v=wa8OrQ\_e76M  
[Accessed 12 05 2021].