

Contact Tracing Application Prototype Documentation

This document contains information on the Covid-19 contact tracing app hosted at [this](#) github repo. The document aims to outline the development process of the application further than the GitHub repos commit history.

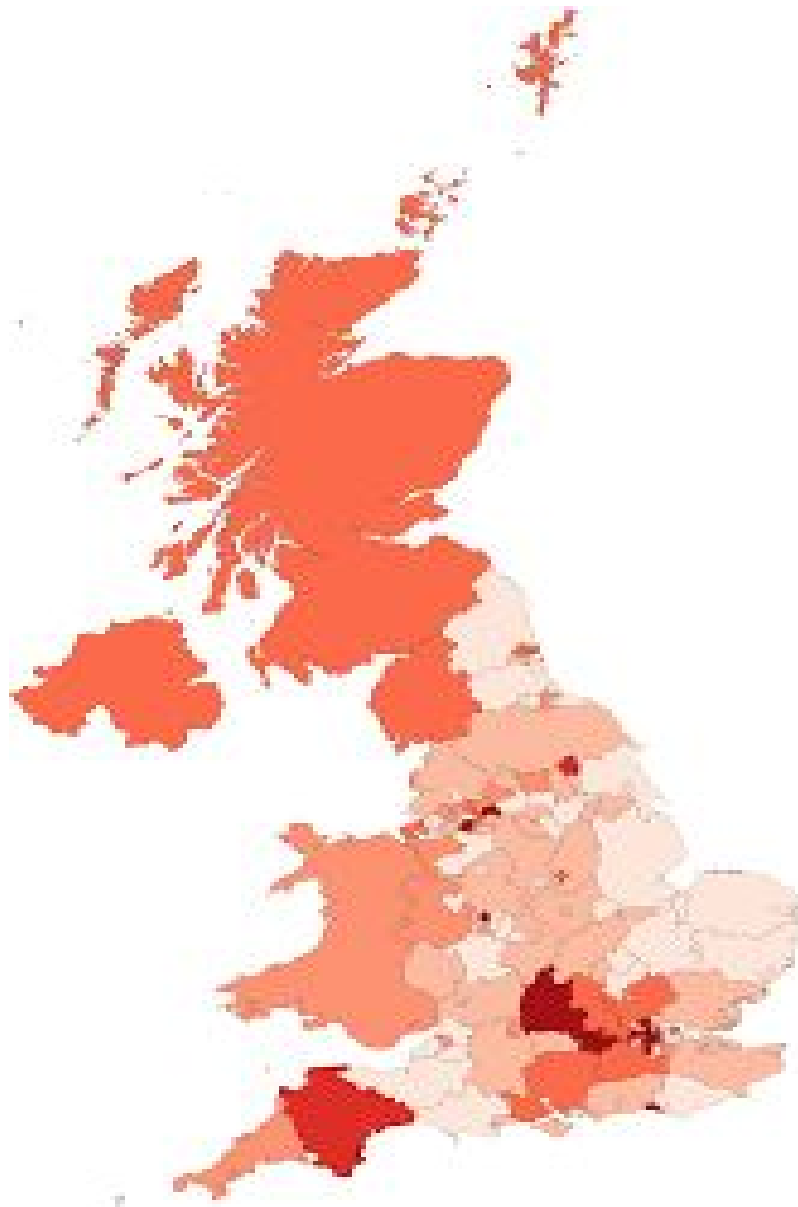


Table of contents

Table of contents	2
Planning	3
Main activity	3
Setup activity	4
Report Symptoms Activity	5
Finalised design	6
Main activity	6
Setup activity	7
Report symptoms	8
Server-Client Configuration	9
Problems encountered during application development	10
Scaling	10
Setup activity intent	10
Threading issue	10
Permission issue	11
Final conclusions	12

Planning

In this section any planning that was made for the application will be outlined in order to aid in the understanding of the development process and how the application changed to suit its use case. The justification of these changes will also be expressed here.

Main activity

In this sketch you can see the home page of the application. This is the page that the user will spend the majority of their time using the app.

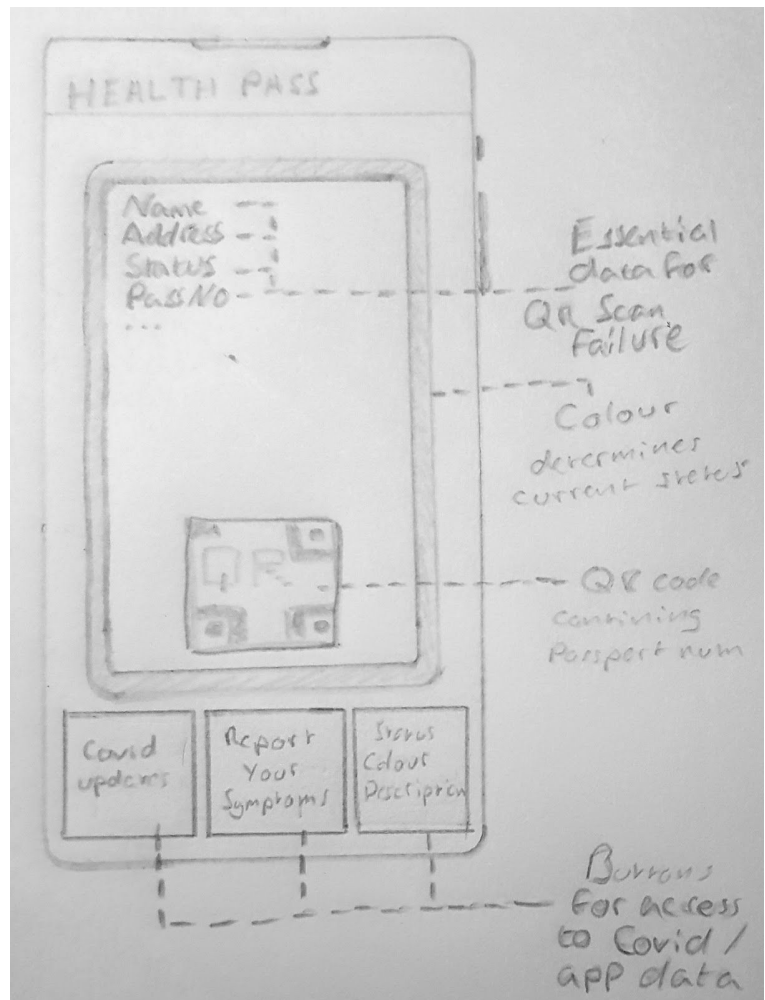
Outlined are some of the elements that will be created in the Android Studio project. Elements such as TextViews, Buttons, ImageViews(QR) and Containers will be needed for this prototypes production.

The QR code Element will have the users JSON data encoded on it for easy scanning by checkpoints or other authorities.

The same data is presented above for faulty QR readers or those without.

Buttons are present at the bottom of the screen. These (in order of left to right) will open the UK government's current guidelines, Allow you to report yourself as having symptoms. Or view the colour description of potential health pass statuses.

The rectangular shape will change colour and a 'Status' attribute will change its value based on your current status of exposure.



Setup activity

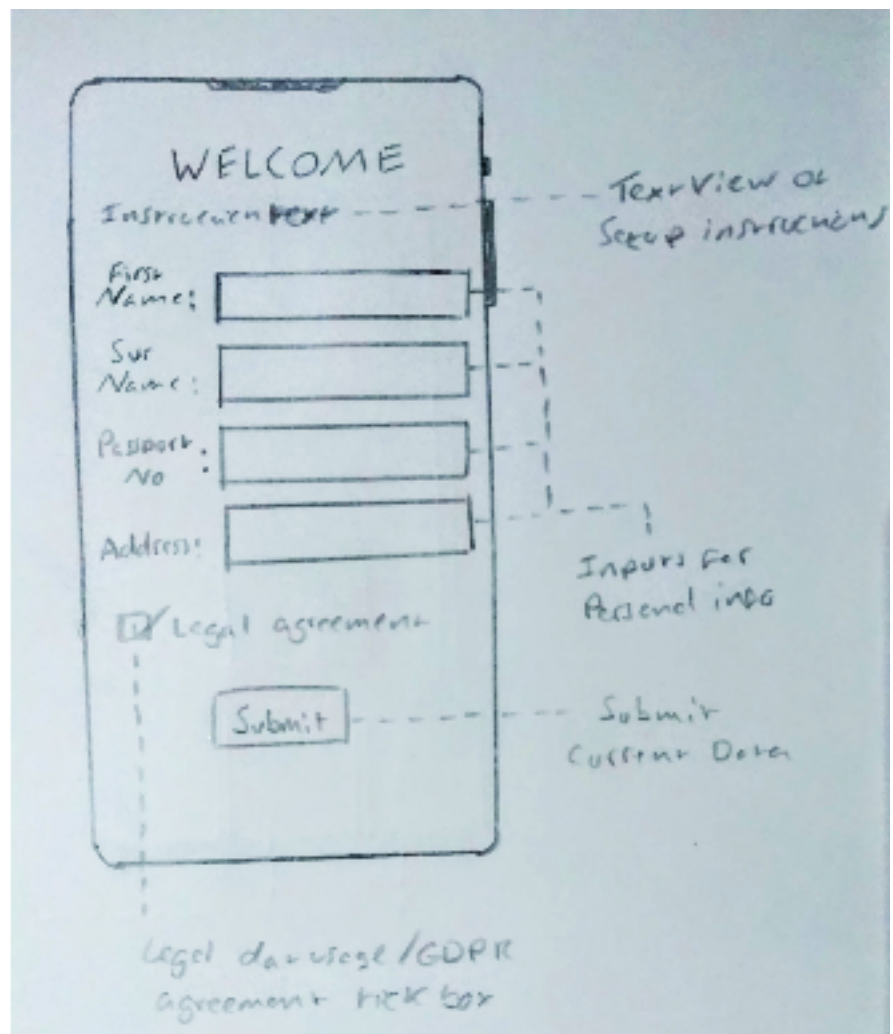
In this sketch you can see the setup activity sketch prototype. This activity will show if the app is being launched for the first time.

Below the welcome TextView there are helpful instructions on what to do to set up the app, informing the user to enter their details and submit.

The boxes represent inputs with adjacent tags that allow the user to enter their details. These boxes should have a level of defensiveness in which only a certain type / amount of data can be entered. For example the Passport No field should only be integers.

A checkbox element can be seen on the sketch above the submit button. This element will allow the user to agree to the use of their bluetooth data in restriction to the GDPR rules and instructions.

The submit button will allow the user to submit their data. However, if there is missing field data in the input boxes then the submit button should not allow the data to be submitted partially and should inform the user of the error. Additionally if the submit button is pressed while the legal agreement is not ticked, an error should appear as well.



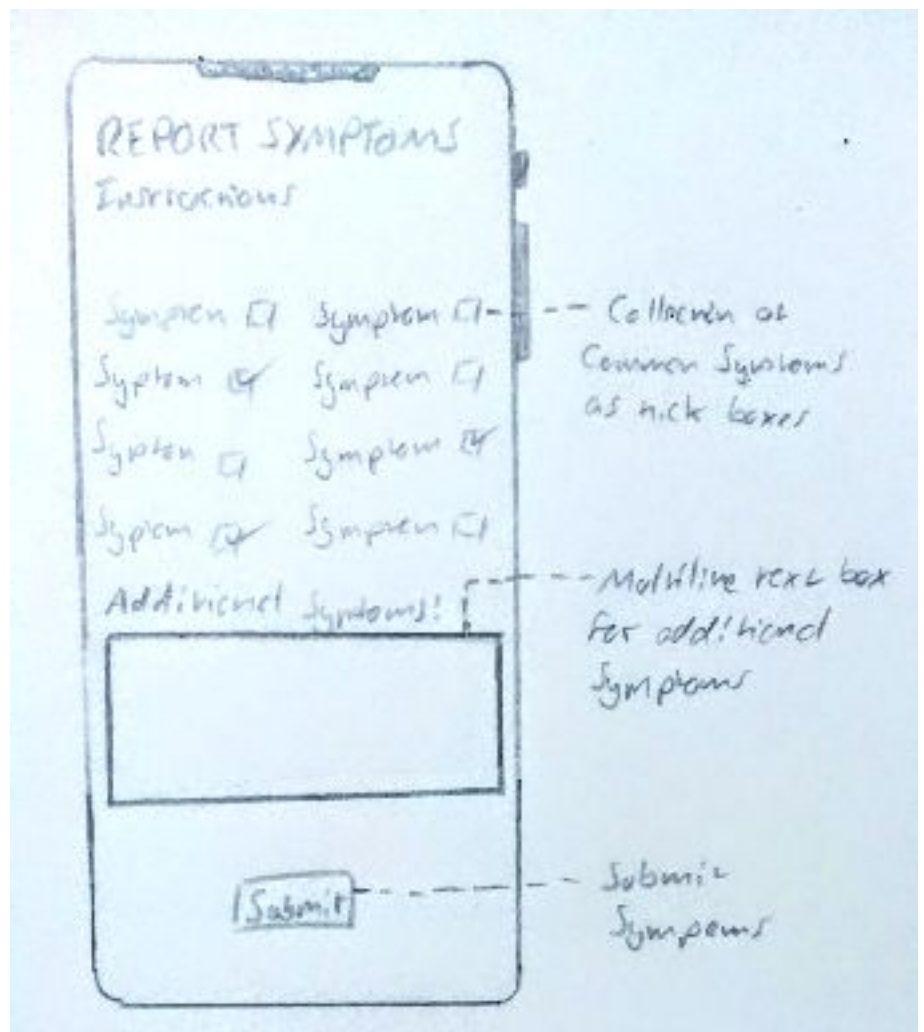
Report Symptoms Activity

This sketch is for the Report symptoms activity in which the user can report any symptoms they have recently begun to experience.

Below the “Report symptoms” title there are instructions to aid the user on what to do to submit their new symptoms.

The collection of checkboxes are each a different symptom that is common to the coronavirus. If the user has one or more of these they can check the boxes for submission.

If there is a symptom that the app did not display in the check boxes then the user can enter it in the additional symptoms textbox field.



The submit button will submit this data to a central server and or being the process of informing other application users that were in proximity of the user, that they may be at risk of developing symptoms.

This button should only submit if a symptom has been selected or if text is entered in the additional symptoms box. The additional symptoms box data could be sent to a central server, where it could be split into key words and more frequent additional symptoms could be analysed as potential additional symptoms.

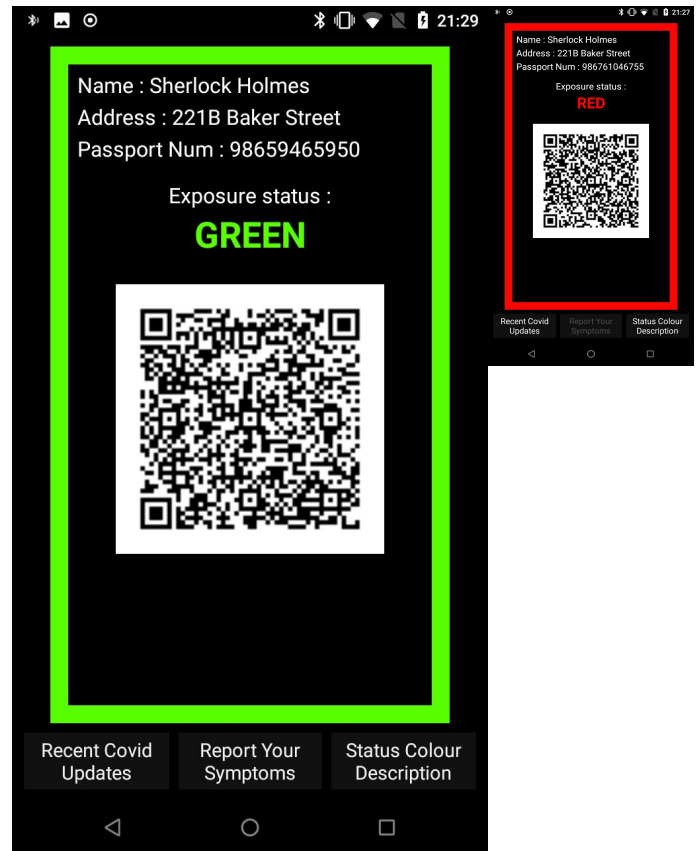
Finalised design

In this section you will see the final designs that were used in the application. They differ partly from the plans as decisions were made while designing them that lead to certain elements being removed or changed.

Main activity

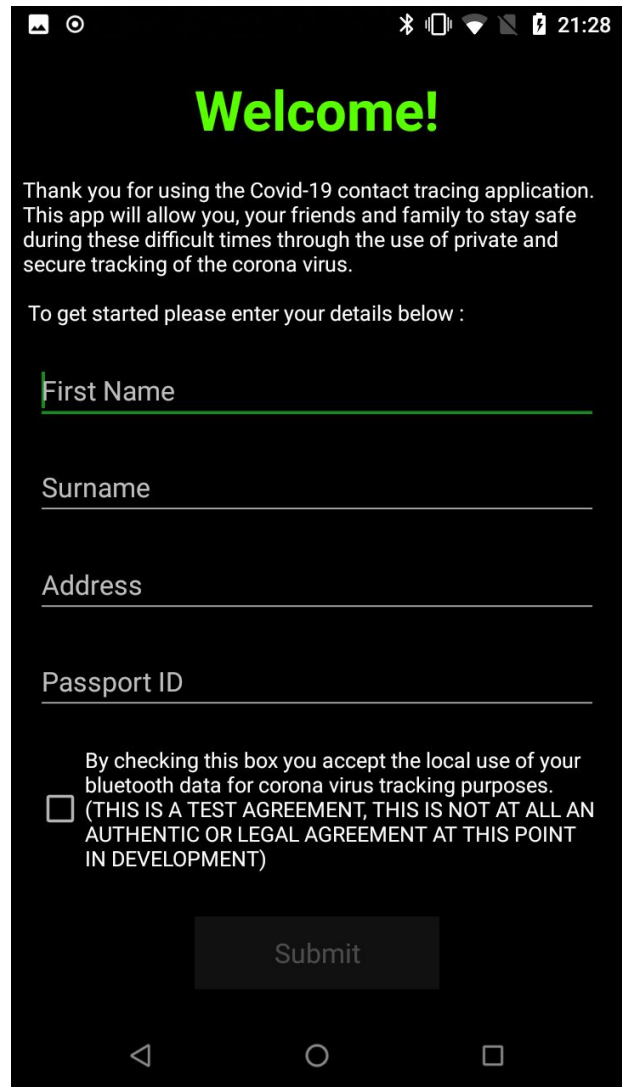
This final design followed the planned design very closely, the only changes really are the height of the buttons and the disabling of the Activity header that i felt simply used up space without reason.

Beside the image on the right you can see the same page, however, the status is changed to red as the user has been exposed.



Setup activity

This final design, much like the main activity, follows the plan very closely also. There were some issues with following the design in a form that allowed the activity to scale however this was easily solved using a vertical chain.



The screenshot shows a mobile application interface for a Covid-19 contact tracing app. At the top, there's a status bar with icons for Bluetooth, signal strength, Wi-Fi, battery, and the time 21:28. The main heading is "Welcome!" in large green text. Below it, a paragraph explains the app's purpose: "Thank you for using the Covid-19 contact tracing application. This app will allow you, your friends and family to stay safe during these difficult times through the use of private and secure tracking of the corona virus." A prompt follows: "To get started please enter your details below :". There are four text input fields labeled "First Name", "Surname", "Address", and "Passport ID". Below these fields is a checkbox with the text: "By checking this box you accept the local use of your bluetooth data for corona virus tracking purposes. (THIS IS A TEST AGREEMENT, THIS IS NOT AT ALL AN AUTHENTIC OR LEGAL AGREEMENT AT THIS POINT IN DEVELOPMENT)". At the bottom is a "Submit" button. The bottom of the screen shows the standard Android navigation bar with back, home, and recent apps icons.

21:28

Welcome!

Thank you for using the Covid-19 contact tracing application. This app will allow you, your friends and family to stay safe during these difficult times through the use of private and secure tracking of the corona virus.

To get started please enter your details below :

First Name

Surname

Address

Passport ID

☐ By checking this box you accept the local use of your bluetooth data for corona virus tracking purposes. (THIS IS A TEST AGREEMENT, THIS IS NOT AT ALL AN AUTHENTIC OR LEGAL AGREEMENT AT THIS POINT IN DEVELOPMENT)

Submit

Report symptoms

Finally, the report page. This too was difficult to scale on devices as the multiple elements (the tick boxes) are vertically restrained on each other.

This made it harder for the elements to align evenly, however the issue was solved through the use of containers and chains.

Report symptoms

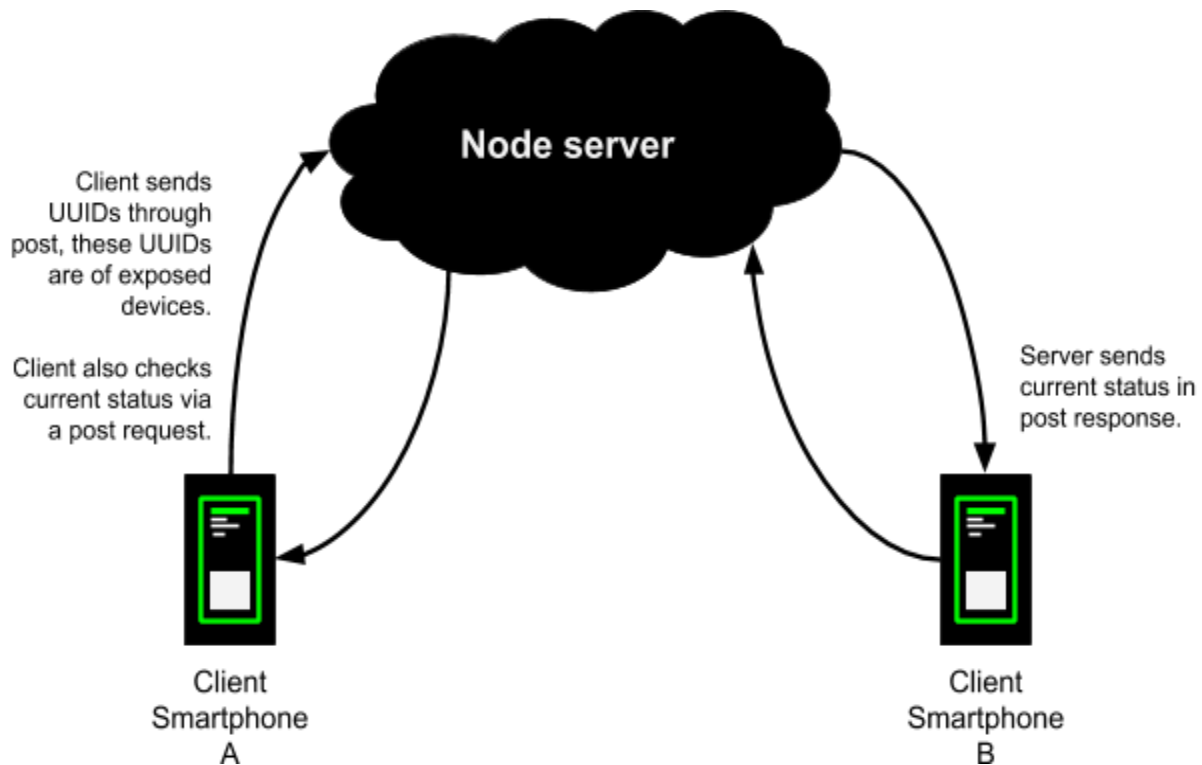
Please tick any of the symptoms that you are experiencing below, if you are experiencing a symptom that is not listed then please enter it in the 'Additional symptoms' field.

<input type="checkbox"/> Fever	<input type="checkbox"/> Fatigue
<input type="checkbox"/> Dry cough	<input type="checkbox"/> Loss of appetite
<input type="checkbox"/> Body aches	<input type="checkbox"/> Shortness of breath
<input type="checkbox"/> Mucus or phlegm	<input type="checkbox"/> Sore throat
<input type="checkbox"/> Headaches	<input type="checkbox"/> Chills / Shaking

Any additional symptoms :

Submit

Server-Client Configuration



When client A reports themselves as having symptoms. A post request will be sent to the Node server containing the current device's UUID and any proximity UUIDs. This data is parsed and appended to a file server side.

Client B will check its status every 20 seconds, the service will handle this transaction using a POST request that will contain the current device UUID. The UUID is then checked against the server side file of exposed UUIDS, if the UUID is inside the file then the server will return the String "RED" in the POST response. Otherwise it will send "GREEN".

Problems encountered during application development

Scaling

One of the initial problems I encountered in development was the scaling. As i was new to android development I was not entirely educated on the techniques of scaling. With the applications elements overflowing out of the displayed region.

Solution

The solution to the problem was quite trivial. Simply following Android Studio's guide to layouts and elements and application of some of the techniques I learned from web design the app soon started to scale better on devices with lower and higher resolutions as well as scaling.

Setup activity intent

There was an issue with the implementation of the setup page, the page which should run once and only run again if it's not completed. This page would allow the user to enter their personal data. The activity would run when it shouldn't and additionally the user could back press to the main activity without completing the setup.

Solution

The solution was solved with the use of an application preference that would trigger once the setup is complete. The alternative was to check if the personal data JSON was present which used File IO and therefore would have been slower. The back press issue was solved simply after I educated myself on the concepts of the Intent class in Android Studio. I simply had to close the main activity when the setup intent was launched.

Threading issue

A small issue appeared with a button that ran a function which would take a large duration of time to complete. This resulted in errors on the render thread. As the button would run code on the same thread that the application was rendering the drawable elements which is itself the UI.

Solution

The solution again was simply resolved via the Android Studio documentation. Where i learned about the need for running specific code in separate threads just like you would use an async function on JavaScript. This issue was easily resolved due to what I had learned from previous node projects.

Permission issue

An issue arose in which the bluetooth background service would not start on boot when a file read write operation was enabled. Upon disabling the file IO operations in the service, it would load the service just fine on boot.

Solution

After an entire day of creating different solutions to store data for the service, such as using the Android preferences functionality, using standard file IO and trying to modify the strings.xml file. Later that day, after almost giving up hope... I recalled the existence of permissions and their importance. After adding a single line granting me access to android internal storage everything worked just fine. An important lesson on looking outside of what you're currently working on and at the android manifest file.

Final conclusions

The development of this project definitely had its ups, and a few downs. The most interesting parts stemmed from the Bluetooth development side of the project. Whereas the most difficult area was the service Permission issue.

I learnt a lot about android app development and bluetooth as well as threading in java. Additionally design wise, my knowledge of scaling and ergonomics of apps has increased from this exercise.

Finally, to note what i feel would need more development in the project, i think that the defensiveness of the app would need to be improved if it was ever to be deployed. Sanitisation of data going to the server would also need to be implemented. And a timer, client side, would need to be implemented to remove UUIDs from the stored UUIDs file after they have been there for a week. This would prevent the erroneous logging of people outside the exposure time frame.