TESTING MANUAL

Smart City

Conor Reilly - 16478326
Eoin Clayton - 16326173

Supervisor - Alistair Sutherland

Table of Contents Page no.

1. User Testing

1.1 Implementation of Survey

The implementation of user testing was carried out using a Google Form including a short survey regarding the application, its usability and the results certain people expected versus what they actually received.

The survey included several questions where a user was expected to give their rating of the application on various features from 1-10 as well as simple Yes/No questions. Finally we included several open ended questions giving users to give in-depth feedback on the application and their experiences with it. This survey was approved by the ethics board in DCU to assure all data privacy rules were kept in order.

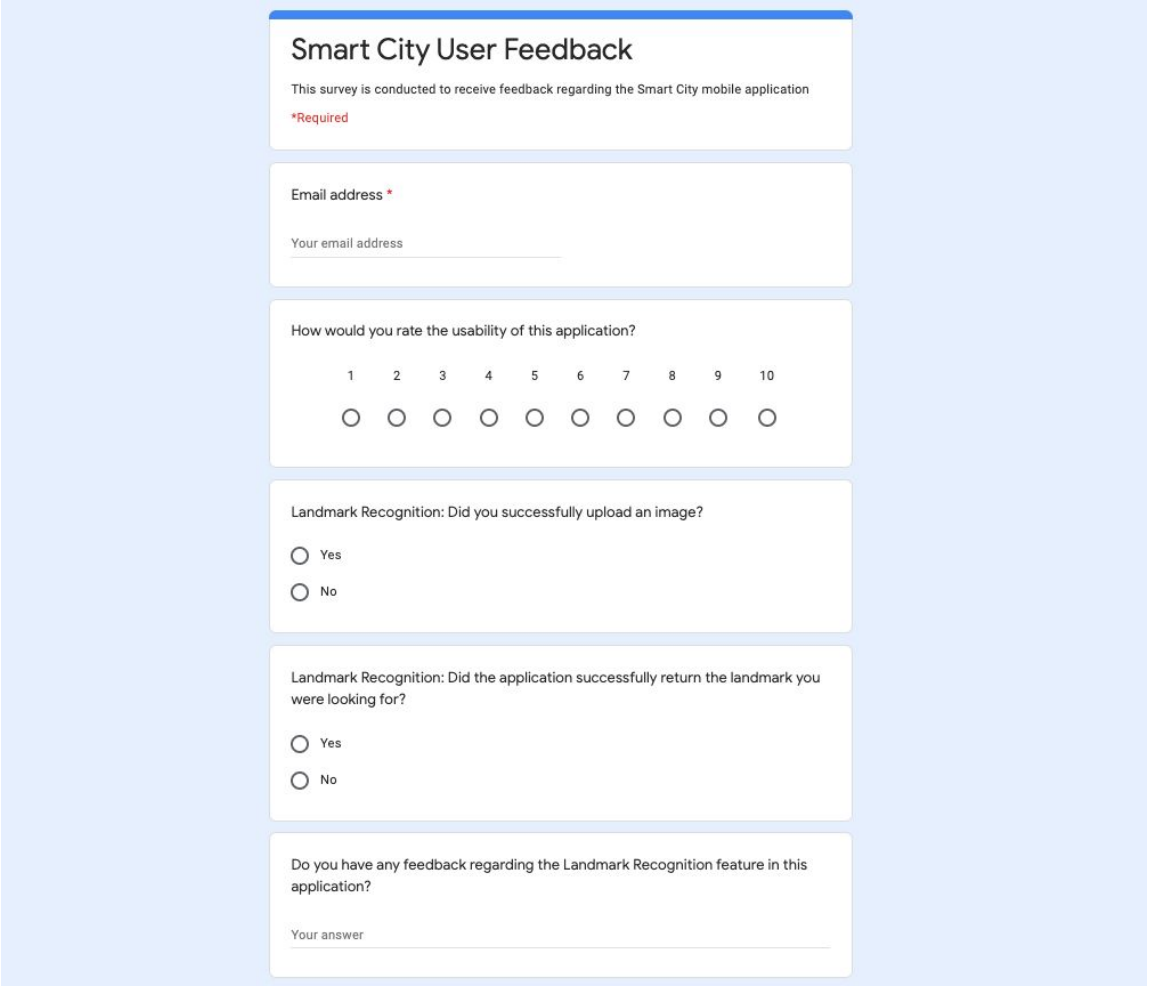
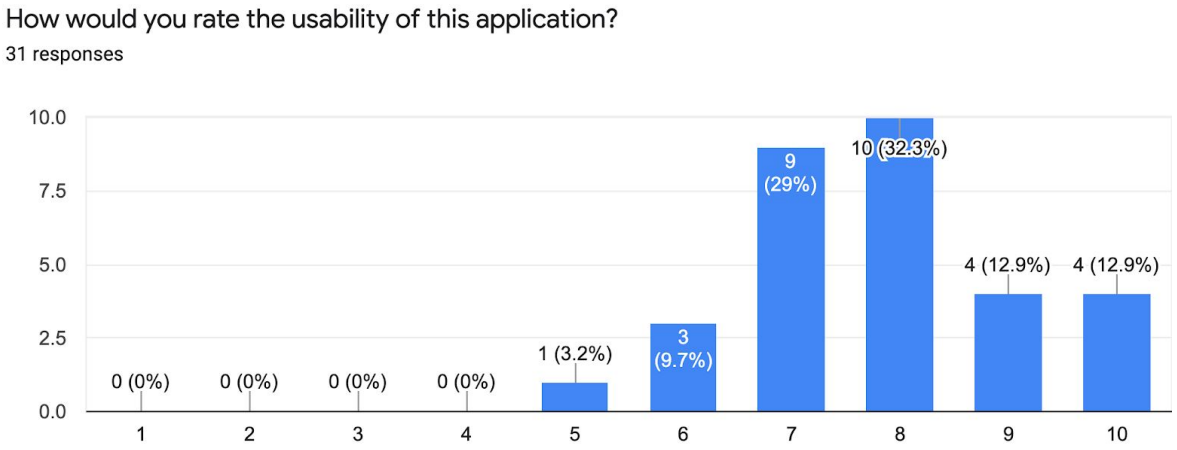


Fig 1.1 Example questions asked in survey

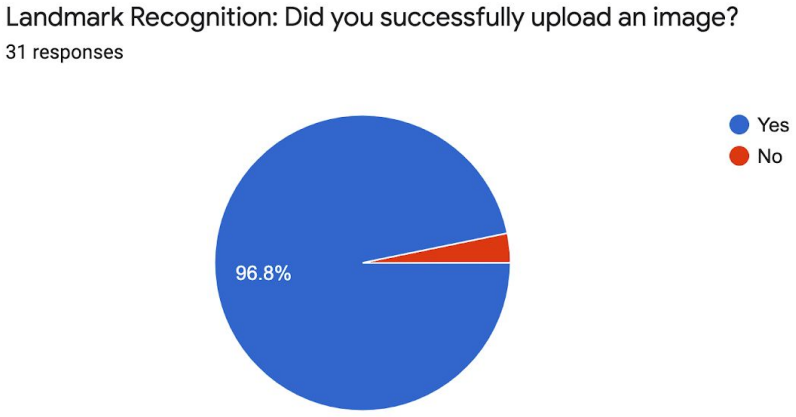
1.2 User Testing Results

The results obtained from user testing were very interesting and helped us gain a better understanding of our application from the eyes of a user.

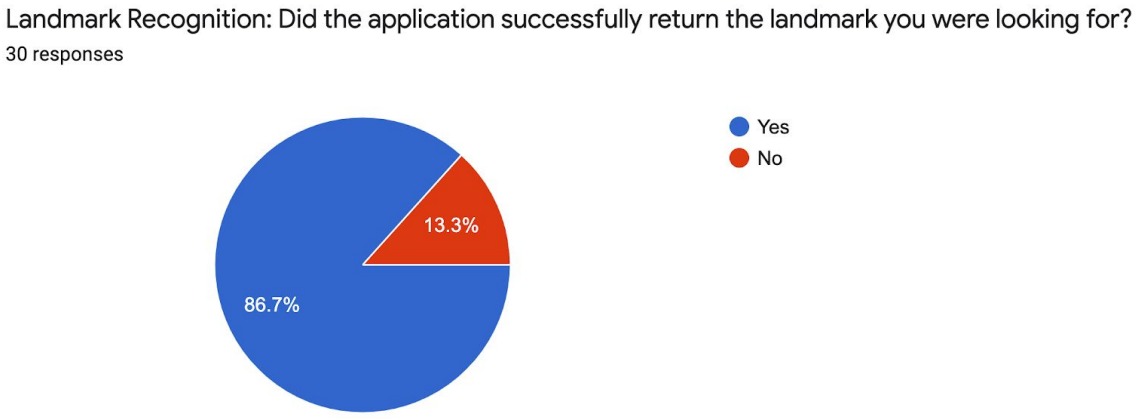
For example, we found that the modal usability score was 8 out of 10. However, we can see there are examples where users had a negative response to the application giving the application a score of 5.



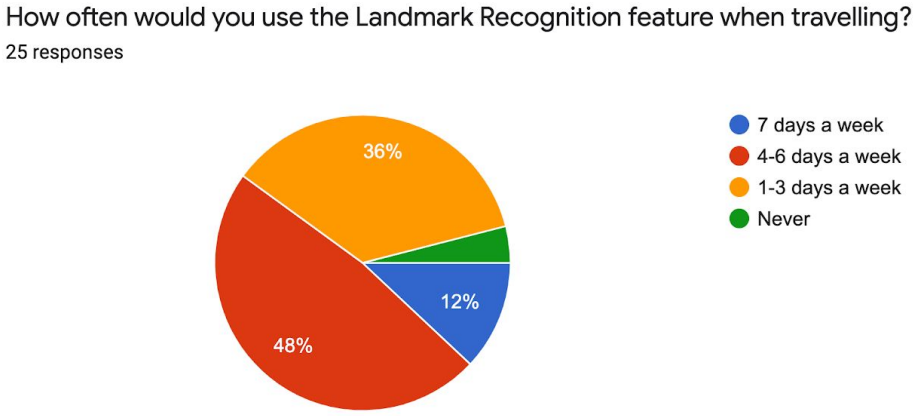
One result that was very positive was that 96.8% of users were able to access the application and upload a photo to the system. This indicates that our initial design is clear in its functionality and clarity for almost all users.



Another interesting result from our survey is that 86.7% of users received the correct landmark recognition in their using of the application. Although this is close to our projected testing of the Neural Network it is possible that some users became confused at this stage and backed out of the application or misread the information given to them.



We also asked users how often they would use various services offered on the application. This gave us a better understanding of how viable our application would be in the open business market and where we should utilise our time when adding future features as well as bug fixing.



Finally we asked several open ended questions that gave users a chance to express their feelings and suggest improvements to the application. We found this result to be very helpful when assessing exactly what each person had issues with when using the application and gave us future ideas on how to improve in design and implementation.

Do you have any feedback regarding the Landmark Recognition feature in this application?

No

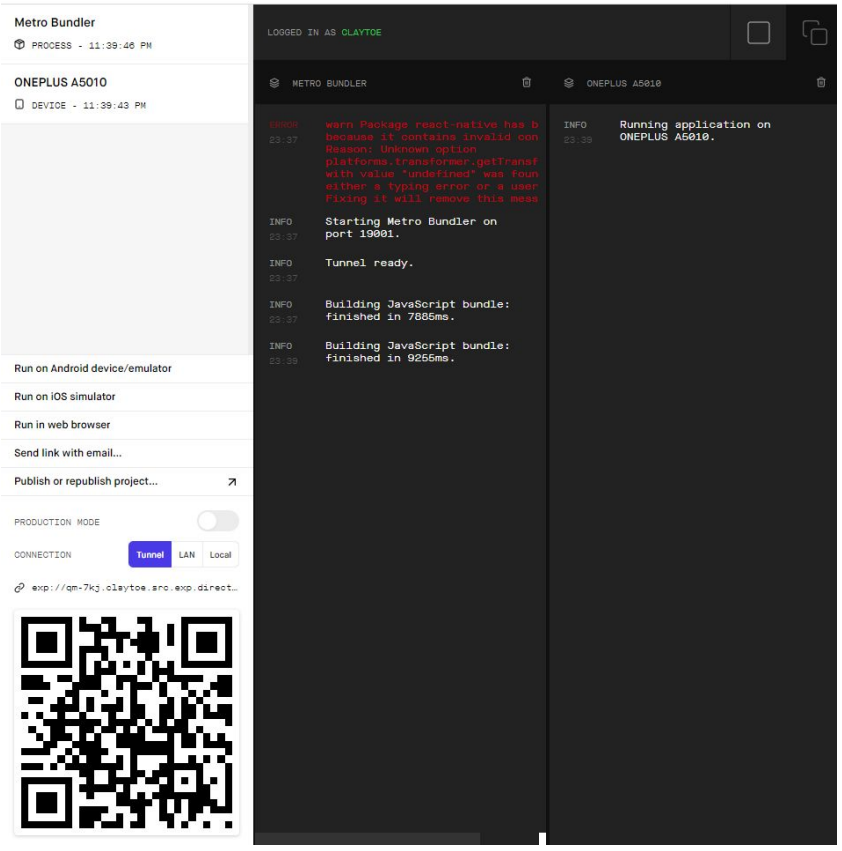
Very clean design

Very interesting and useful when travelling!

Did not return landmark correctly

Very handy!

We were able to track each user's success rates and connections to our system using the React Native system. This allowed us to see when users were accessing our platform and how long they stayed connected for.



1.3 User Testing Conclusions

From this short survey we gained a greater understanding of how usable our application was to all types of people. We gained a better understanding of how people interact with each section of the application independently. We also learned where our app needs to be improved in terms of reliability, clarity and accuracy. Given current circumstances we found it difficult to find a large database of people to test our application. Finding out that 35 people took our survey is something we are happy with but in future we would have liked to have a more reliable source of information.

2. Unit Testing

Throughout this project we also implemented Unit Testing whenever possible. This involves splitting and dividing individual units/ components of software into sections to be tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. This is often the first level of testing but usually the most important as it sets a solid foundation for the system to build on.

We found in the testing at this phase it was best to use white box testing. White Box Testing, also known as Clear Box Testing involves the developer choosing a specific input, where the tester knows the function of the system and determines the known output. This is a perfect style of testing for the early implementations of our system.

We decided to test the react native code using snapshot testing in Jest. Snapshot testing is useful to ensure the UI does not change unexpectedly.
This test renders a component in react native and then takes a snapshot. This snapshot is then compared to the snapshot which it already has on file. If the two snapshots match then the test will pass.

```
1  import 'react-native';
2  import React from 'react';
3  import WelcomeScreen from '../../Screens/WelcomeScreen';
4
5  import renderer from 'react-test-renderer';
6
7  test('Welcome SnapShot', () => {
8
9      const snap = renderer.create(<WelcomeScreen />).toJSON();
10     expect(snap).toMatchSnapshot();
11
12 });
13
14
15
```

```
PASS  __tests__/Screens/WelcomeSnapShot.js
  √ Welcome SnapShot (497 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   1 passed, 1 total
Time:        4.887 s
Ran all test suites.

C:\Users\eoinc\source\repos\2020-ca400-claytoe2-reillc54\src>
```

3. Use Case Testing

Use Case Testing is a functional black box testing technique that helps testers to identify test scenarios that exercise the whole system on each transaction basis from start to finish. Here we will look at a number of scenarios and describe the expected results.

Reference Number	Scenario	Result	Developers comments/ Proposed solution
001	User clicks “Camera”	User smartphone camera opens	As intended
002	User takes photo	Photo stored in application	As intended
003	User clicks “Submit Photo”	Image is sent to database and processed	As intended
004	User receives correct landmark in response	Image contains information about landmark	As intended
005	User receives incorrect landmark in response	Image contains information about incorrect landmark	Not as intended, User can resubmit new photo
006	User selects “Predesigned Route”	User is directed to list of selectable routes	As intended
007	User selects a route	User is directed to Google Maps	As intended
008	User selects Nearby Locations	User is directed to google maps of pinned locations to choose from	As intended
009	Selects a nearby location	Destination is selected	As intended
010	User exits app	App closes	As intended

4. Agile Testing

Throughout this project our main type of software testing was to use Agile Testing methods. We believe this suited us best as there was a small number of participants and given we were often together while programming large sections of our project it was the optimal choice.

We intended on building weekly scrums to tackle a new challenge each week. This coincided with our lectures taking place daily giving us time to complete sections of the system one by one. We set up small sprints for each individual person to tackle as well as splints we both would pursue for the topics that were more difficult to comprehend on our own.

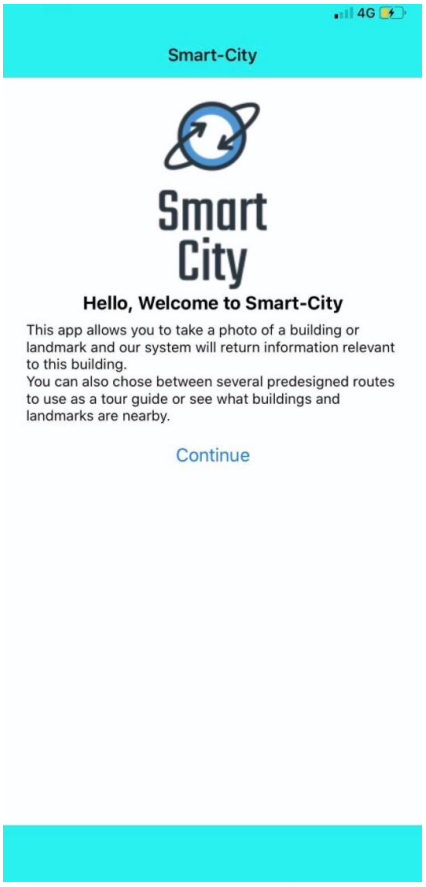
We also set up a Gant chart to help track our progress throughout the course of the project.

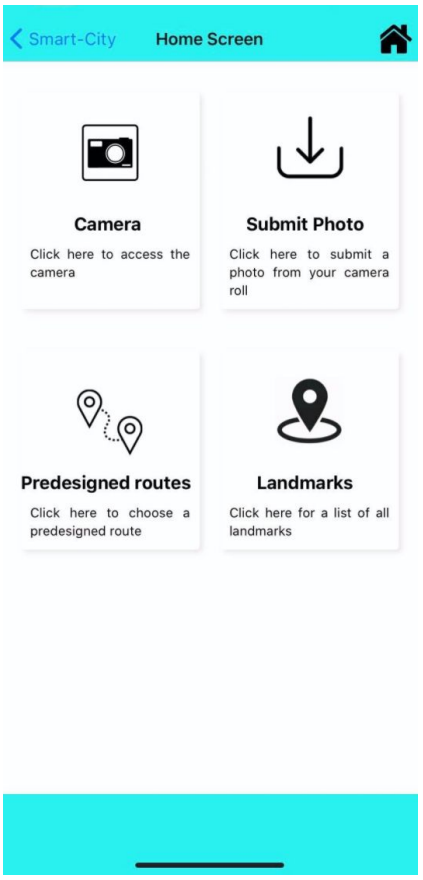
5. Heuristic Testing

5.1 Shneiderman's Rules

Strive for consistency

Throughout our system we ensured that similar technology was used to avoid confusion. We kept to similar colour schemes, button types and other various design aspects visible in the GUI and non-visible in the code itself. We used refactoring of code to help keep systems running in similar ways.





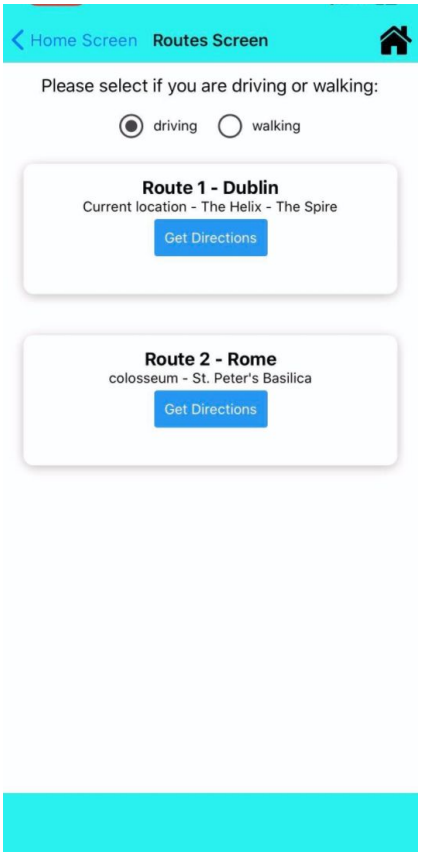
Enable frequent users to use shortcuts
Once a user becomes comfortable with the system there are several shortcuts they can use. For example a user can quickly change between screens within the app without reading the information provided. WIth instant button click options available for those that are comfortable with the application.

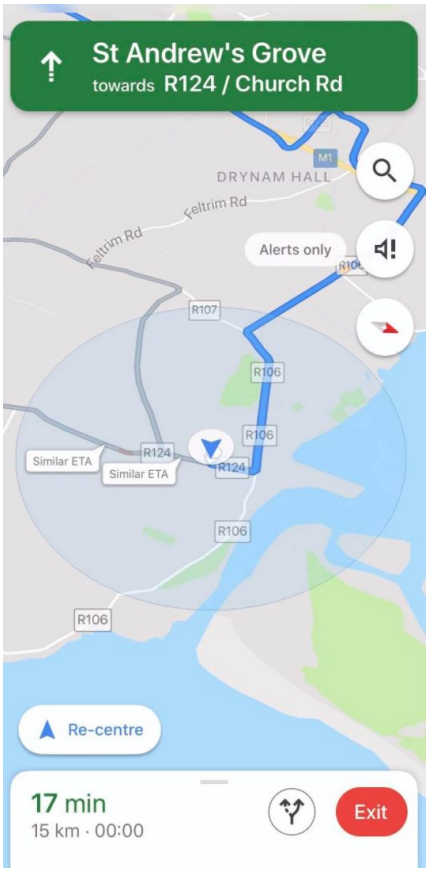
Design dialog to yield closure
Within the application, each page has an end result relating to the request the user has made. When searching for a landmark, a user will be presented with a prediction as to what the landmark is. If it is incorrect, the system allows a user to take another photo.

Offer informative feedback
On each screen a user is presented with the path they have specifically chosen. Within the application, each screen offers text based explanations to inform users of the purpose for the feature.

Permit easy reversal of actions
The permit of easy reversal of actions is possible at each stage of app. Although there is no clear reversal a user is free to navigate the application to suit their needs while in use.

Support internal locus of control
Users have full control of the application. They can customise any route they would like to take or submit any image of a landmark they would like. Users initiate all actions within the application.





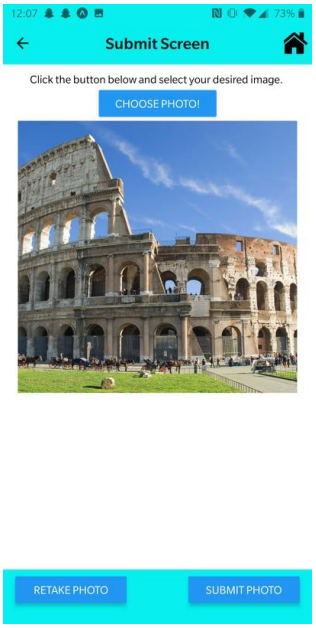
Reduce short-term memory load
Given that our application is fully customisable it heavily reduces the amount of short-term memory required.

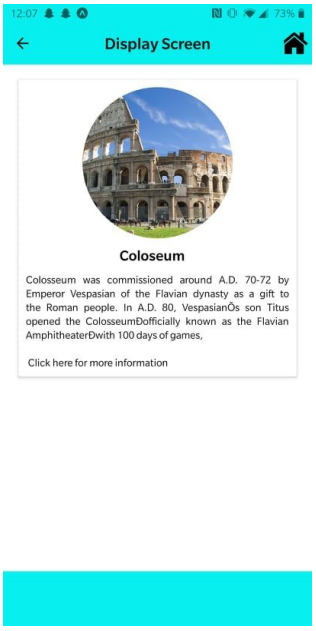
6. Adhoc Testing

We used Adhoc to test many different features of our application.

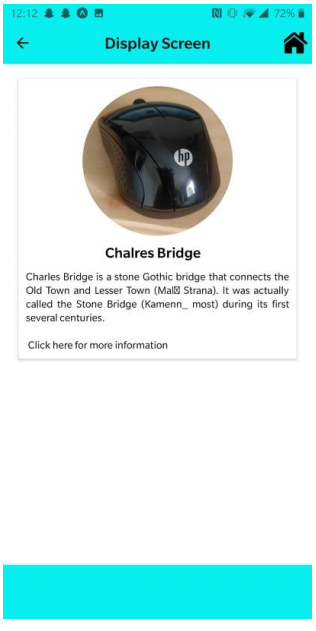
While implementing Adhoc testing I wanted to submit a video to test what the result would be but the app only displays photos that are in the camera roll.

Another method I tried was to submit a photo and close the app. The app still fulfilled the request in the background and when I went back into the app the display screen was loaded with the correct information relating to the building in the photo.

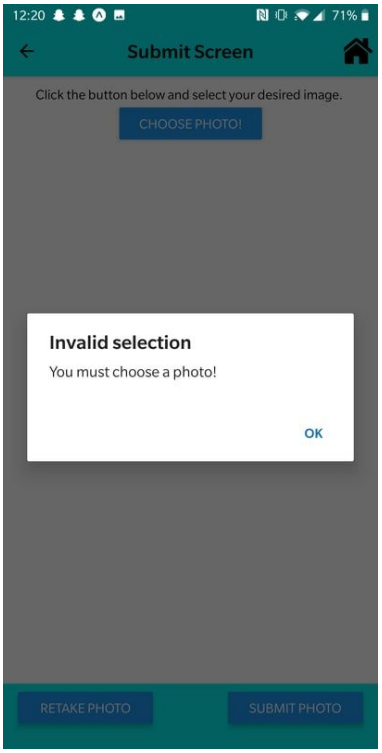




I also tried to submit a general photo which was not associated to a building compatible with the app. The app still looked for similarities in the database and returned information about the building it was most similar with.



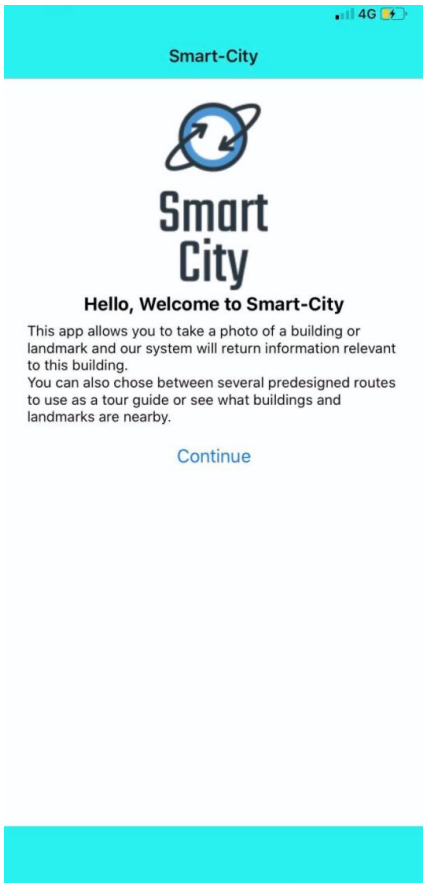
The last method I used was to press submit without choosing a photo. This resulted in an alert telling the user that it was an ‘Invalid selection’ and ‘You must choose a photo!’

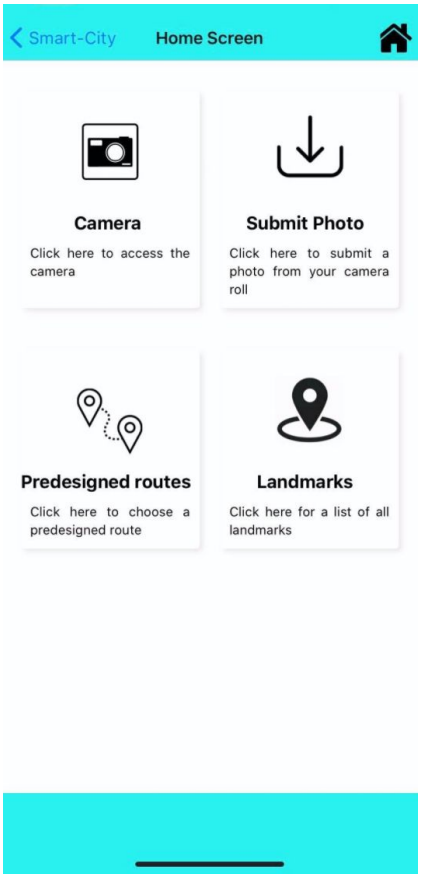


7. Accessibility Testing

Visually Impaired Users

Using React Native allows for screen readers to operate over an application, this means that if a user is visually impaired they can still use the application as intended. When developing we ensured that each piece of text was large and each graphic or icon directly represented the outcome of clicking on it. This should help those who have less visual capabilities. We also decided to use bright neutral colours such as blue to help those who may be colourblind.





Motor Skill Deficiency

Users who suffer from poor motor skills may hopefully find this product manageable to use. Although some of the buttons appear small, each was taken into consideration in terms of design to be clear and precise for all users.