Mobile platform development Design Report

Cameron Crawford

GitHub Link to Repository: https://github.com/cameroncrawford1/MobileCoursework

GitHub Link to video: https://github.com/cameroncrawford1/MobileCoursework/blob/master/MobilePDVideo.mp4

GitHub Link to .apk file:

CCRAWF203@caledonian.ac.uk

S1628376

Design Report

In this report, a description and explanation will be provided, detailing the layout used inside of the application created. What was implemented will be discussed and then what wasn’t will be discussed as to why it didn’t work and how it could have changed how the result of the application would have been different.

User Interfaces are designed to make the system look presentable and to attract the user to wanting to use that application. Thus, throughout the development of the application a consistent and simplistic User Interface was always the intention. It is important that User Interface designs are easy to understand, from the moment you get on the application you are not given a guide, you have to understand what it does and how to use it and a simple but effective User Interface is key to making sure users can understand and enjoy the experience of the application. When the application is run and the orientation changes from portrait to landscape, the layout adapts to cope with the change in the orientation, so if a user had their device side on then the application will be able to cope with it and the functions will not overlap, they will resize to fit the changes.

# Home Page

Firstly, is the home page. When the home page has been loaded, two buttons are displayed. These two buttons are links to the functions which the application provides. The first button is the planned roadworks button. The user selects this button to see all planned roadworks. The button has a background colour of blue, the option to use set colours throughout the application was intended to help users understand in the navigation of the system what pages they were on, so the background of the planned roadworks page was set the same as the button to allow users, especially those driving, to understand if they are on the wrong page without much attention needed. The second button is the current incidents button, this links to the current incidents page and is red, just like the roadworks button, the current incidents page is set to the button colour. Both buttons are located in the middle of the page and the width is set to match the parent, which means that it will stretch to the relative container it is held in. This was done to ensure users can clearly understand the options that are provided, the buttons are in plain sight, they are not tucked away in a corner and are not too small to click on.

# Roadworks Page

Within the planned roadworks page there is a lot of visible functions that are displayed. The main function is the one to get the parsed data, done by the XmlPullParser, and display it. After a few attempts at trying out views, it was concluded that the List View was the best for the job. The List View is a view where by items are displayed in a vertical scrollable list. In order to populate the list an adapter is used, this adapter originates from an array, the view in the application takes all roadwork values from the parser and displays them. When populating the list, the adapter is responsible for getting the data to the user interface from the java class. Within the code the adapter is initialised specifying the constructor and linking to the layout file that is used to display the data. On the roadworks page there are two list views, one for the initial results of the parsed data and one for the results of a search which the user can carry out. It was decided that through the property XML page there would be a way to provide a template of a preview format, allowing the user to see a snippet of the details, this will let the user understand what they are looking for instead of having to click each roadwork instance. When the List View displays the results, each separate roadwork displayed is clickable, this will take the user to a recycle view, this recycle view gets the data of the selected value and displays the full details of that specific roadwork.

Above the list view there is a search bar in which the user can enter the road that they would like to view planned roadworks of. When the user enters their search word they will then press the search button which takes their search word and loops through all of the roadworks, stores the matching results into an array which is then placed into the list view. When the results have been returned a toast is returned to inform the user that their results have been produced. Alternatively, if the user has entered a search word that does not return a value then a toast is displayed, this time informing the user that the search cannot be found. Finally, there is a colour coding system in place for the roadworks page, this provides the user with a way to understand the roadworks that are displayed. Red is used to show roadworks lasting longer than 7 days, yellow is used to show between 2 and 6 days and green is used for roadworks that last 1 day. This function was added to allow users to clearly understand the severity of roadworks and if they are a constant constraint on their daily commute or just a one-off closure. This will help the driver to plan their trip, whether it is on the day or planning it in advance.

Overall, the layout of the Planned Roadworks page is designed to be a simplistic approach to displaying a large amount of data and with a function like the search bar it will allow the user to filter though all of the roadworks to find the information they require.

# Current Incidents Page

The current incidents page is very similar to the planned roadworks page in that it uses a list view to display the data that has been retrieved from the XmlPullParser and then it displays the data, by using the adapter getting the data and then giving it to the list view. This allows the user to scroll through the results easier, especially since there can be a potential large amount of data. When the user selects an incident, they are taken to a details page which gives the user the details about the incident. This is a helpful way to provide the user with information, especially if they are planning to travel, this will allow them to know what route is best for them to take. Unlike the planned roadworks page, though, the current incidents do not have a colour code scheme. They are just displayed as a normal list; the background colour is the same as the rest of the page. This could be a scope for future work.

# Details page for both

When the user selects one of the values from the List View on either of the roadworks or current incidents pages, it displays a details page. The details page is used to provide the full details of each value. It takes the data that has been parsed by the XmlPullParser, which is then stored, it is then displayed through the recycler view. Once a specific item has been selected, the details of the selected item are passed to the recycler view, this will then display the values for that specific instance.

The layout of the details page for both the planned roadworks and the current incidents is structured in a list format, each value is given a label and then the populated data is displayed next to that label. The title is displayed in clear bold text, this is used to ensure that the title is clearly displayed and visible to the user. Next is the description of the incident or the roadwork, then the location is displayed in the longitude and latitude format and a link is provided to Traffic Scotland in case the user need to access the original page for the value. For roadworks, there is a potential diversions section to provide users a way to access diversion information before they drive.

# Future Work

If the development of the application was to be carried out again or if the project was to continue past the deadline, then there would be a few routes which would be suitable to go down to improve the current functionality of the existing application and add more tools for the users to have more options on planning their journeys.

The first could be amending the current incidents page, which could have a severity counter that displays a colour coded warning to drivers on the severity of the incident. This can then be used by the user to plan what roads they should use on their journey, this would be of benefit to users who, for example, are at work and are about to head home and want to know if there are any incidents that they can avoid. The colour coding will allow the user to understand if there will be a big hold up, a moderate hold up or if traffic is managing to flow but slightly.

Another could be to add a search onto the current incidents page, allowing the user to search a road that they could potentially be about to use.

Also, a view roadworks by date filter can also be a function to add, allowing a user to select a date which they plan on travelling and the user will be shown the roadworks that are being carried out on that day and they will be able to search that, like they can now for a specific road.

Testing Report

In this report, there will be an insight into how the Traffic Scotland application was tested. Applications must be thoroughly tested, this is done to ensure that all of the functions that are contained within the application works as they should and the behaviours which they are expected to have when executing a specific function. Applications are also tested to ensure that it meets the requirements that it has been set at the start of the project.

# Methods Used

When testing the application, two main testing methods were used, the first was Black Box Testing and the other was Usability Testing.

## Blackbox Testing

The first testing method is black box testing, this type of testing is done on the running of the application, the tester has no knowledge of the code behind or what interactions should happen. The tests are carried out and are compared to the requirements which has been set at the start of the project. In terms of the development of this application, the testing was carried out with the knowledge of the initial document set out at the start of the project, this gave an understanding of what functions are required and if the functions implemented were doing what the program was indented to do.

When the testing began, there was a set of test cases which are predefined, when these are being carried out, documentation is produced, showing the results of each individual test, indicating whether they have passed or failed the test that was carried out. The testing is a thorough run through of the application, the tester tests the application using normal, extreme and exceptional data, to ensure that the application has sufficient try and catches and also has appropriate toast messages to inform the user when an error has been produced, as opposed to the application crashing while doing a process.

## Usability Testing

Usability testing is where the application is presented to potential users. Users have been assigned tasks to complete when they are on the system, they have no knowledge of the system at all and they have to get a hang of the application on their own and it is part of the test to get an insight into how easy the application is to understand and how well different users can interact with the functions of the application.

As strange as it sounds, the aim of usability testing is to highlight as many errors with the application as possible, this will be done prior to launch and if the developers can encounter the problem through users who they trust to test the application then instead of it going public and the application can end up getting hacked or could have an error which crashes the system or returns the wrong outcome from a function. These are all found through usability testing.

In terms of this application, numerous potential users were given the opportunity to look through the application, try and understand how the application worked and if there were any unexpected errors with the application that could be flagged up to be fixed before the submission. The testing returned no errors, the application reacted as it should to all of the tests.

## What was tested?

In order to make sure a successful testing process was carried out on the application, there was a clear process of what needed to be tested but also what each test was looking to do.

On the home page, each button had to be checked to ensure that when they are initiated then they link to the correct page.

On the Planned Roadworks page, a few tests had to be carried out, the first test was to ensure that the planned roadworks actually loaded from the rss feed and had been correctly parsed into application and that the adapter had then passed the information to the user interface to be able to display all of the information that is required on the application’s List View. The next was testing the search function, the user would be able to enter in a search word and once the user pressed search, there had to be a test that the correct query had been made and that the results weren’t the results already present. A test was carried out on the details page of the application, one before it was searched and then one on the results of a search, the user would select a roadwork and then the details of that roadwork was loaded, displaying its details. The back-navigation functionality was also checked to ensure it went back to the page before.

The incidents page was loaded and the test on whether the incidents were displayed was carried out. The incidents were displayed on the page in a list view and they were made selectable, so a test was carried out when the user had selected one of the incidents and then the details were displayed. This was done to ensure that the correct data was being pulled from the feed and displayed on the application.

A test was also done on the orientation management of the system. This was done to ensure that when a user twists their screen then the application will manage the orientation and will adjust the screen to the appropriate layout design for the landscape mode. Most applications, if they cannot cope with this, are marked down on their quality, so it was essential to make sure that the application could cope with the change.

Finally, a test was also carried out on whether the application can display the appropriate message when the device loses its access to the internet. The purpose of this is to ensure that if the device loses connection then there is no access to the rss feed but at the same time the application does not crash. Also, the test was used to ensure that an appropriate error message was displayed when the device did lose connection, informing the user of the problem and that it is not a system error.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Number | Test Case | Input | Expected Output | Actual Output (Y/N) |
| 1 | Test that the application loads, displaying the home page | Load Application | Application Loads | Y |
| 2 | Test that the roadworks button can be selected, and the roadworks page is loaded | Click Roadworks button | Roadworks Page loads displaying the current roadworks | Y |
| 3 | Test that the roadworks has been displayed showing a colour coordinated list | N/A | Roadworks are colour coordinated | Y |
| 4 | Test that a user can select one of the roadworks on the list view, which displays the details page for that roadwork | Select a roadwork | Roadwork details page is displayed | Y |
| 5 | Test that the back button takes you back a page | Click back button | Page goes to the previous page | Y |
| 6 | Test that when a user enters an incorrect value to the search field then an error message is displayed | “#” | Error Toast is displayed | Y |
| 7 | Test that when a user enters a search, the appropriate details are produced | “M74” | All roadworks for M74 is displayed | Y |
| 8 | Test that when the user selects on one of the results from the search, that it loads the appropriate details page | Select Roadwork | Appropriate roadwork details page is displayed | Y |
| 9 | Test that the current incidents page can be displayed | Select current incidents page | Current Incidents page loads, displaying all incidents | Y |
| 10 | Test that when the user selects one of the incidents then the details of that incident is displayed | Select an incident | Appropriate incident details is displayed | Y |
| 11 | Test that the user interface displays the appropriate view when you switch the application | Change Orientation to Landscape | Screen adjusts to the size | Y |
| 12 | Test that when the virtual device has no internet access, the user is informed they cannot connect to the internet | Turn off internet and load roadworks | An Error message is produced informing the user they do not have access to the internet | Y |