Department of Computing
Goldsmiths, University of London

Augmented Reality Navigation System for Commercial Spaces

Report

by

Arif Kharoti, Nicholas Orford-Williams, Hardik Ramesh,
Gabriel Sampaio Da Silva Diogo, Hamza Sheikh, Jonathan Tang
Software Projects – Group 14

Spring 2019

Submitted in partial fulfillment for the degree of
Bachelor of Science in *Computer Science*

Abstract

The use of mobile augmented reality by consumers, and research in the field has become more prominent in the last decade through social media, and games. This has allowed for completely new approaches in solving current problems using this technology as there is a year-on-year increase on smartphone users across the world. As a navigation-based application for smartphones that is appropriate for the application of augmented reality technology, the project develops a museum navigation system. Museums are complex commercial spaces that have many intricacies in its own architecture, making it easy for visitors to lose their sense of direction. These complex buildings are suitable for applying the augmented reality with the additional challenge of mapping user location in real-time. Current solutions to indoor navigation within these attractions only consist of 2D paper maps, lacking a sense of realism. This project develops an augmented reality navigation system for museums that can provide users with real-time directions to their desired location.

This report covers the approach taken by the group to explore a theoretical solution to the problem at hand using Bluetooth, and the A* path finding algorithm. By developing a system that allows for real-time navigation with augmented reality assisted technology, this project shows the potential of its use in extensive commercial spaces, not only in museums. Thus, this report details the processes of requirements gathering with potential end users and other key stakeholders. Prototyping and design took place with various AR libraries, and operating systems, concluding with using Android with ARCore to implement the product. Front and back-end implementations are described and implemented through the Agile methodology with Scrum framework. Product testing is outlined with the use of test-driven development, and the deployment of the application to market. Not all initial approaches and decisions taken the by the group were efficient, evaluating implementation decisions were crucial in the overall development of the system. An overall evaluation, and analysing the future of the project and its use in the market are also discussed.

Contents

List of Figures

List of Tables

Nomenclature

AR	Augmented Reality
CI/CD	Continuous Integration/Continuous Deployment
GDPR	General Data Protection Regulation 2016/679
IP	Intellectual Property
MVP	Minimal Viable Product
SDK	Software Development Kit
SDLC	Software Development Life Cycle
TDD	Test Driven Development
UI	User Interface

Acknowledgements

We are extremely thankful and grateful to our supervisor, Dr. Basil Elmasri who gave us timely advice, guidance, and encouragement throughout the project despite the great technical challenge that was set. His willingness to help us come up with coherent and presentable ideas is unfathomable. Also to Matt Isherwood, Managing Director at Pathfindr who gave us his expert opinion surrounding current research done in the augmented reality field - signposting potential pitfalls, and helped developed our ideas into a feasible project. Finally, to students at the Ruskin School of Art, University of Oxford, museum staff, and members of the public for putting aside some of their time to give crucial feedback along with direction throughout the project.

April, 2019

Chapter 1

Introduction

1.1 Motivation

Often, people find themselves lost in unfamiliar spaces such as museums; usually immersed by the culture around them. This project aims to tackle this issue by allowing users to restore their orientation to have an augmented reality (AR) platform, routing users to their destination. Users will have access to navigational assistance at all times as the platform will be available on their mobile devices. Through the means of an application featuring substantial indoor navigational support through AR assisted technology. This approach is considerably more extensive, and dynamic in comparison to the existing 2D solutions available in the market. The platform will use the device's camera to work out its surrounding, and produce a highlighted line on the screen to their destination in real time.

This concept has various applications to other scenarios such as finding products in a supermarket, or books in a library. Furthermore, the concept could also use machine learning in identifying user traits in places visited in a museum in order to give personalised recommendations at other similar exhibitions.

1.2 Purpose & Scope

The purpose of the project is to provide a solution to a real-time navigation system to assist users to their desired location within a museum. The significant part of this project is achieved by calculating the shortest route to the user's destination based from the user's current location, and displaying this route on the user's device through the implementation of AR-assisted superimposition of navigational lines that the user would follow.

1.3 Assumptions

It is assumed that the reader is familiar with the supporting documents of this report, including the proposal, system requirements specification, the documentation, testing, and deployment plans (see appendices). These documents outline the concept of the project, and detail the procedures to achieve the purpose of the project.

1.4 Coverage

This report fully describes the project undertaken, containing eight main sections:

1. Introduction - Outlines the motivation, purpose and scope of the project. It includes any assumptions based on the reader regarding any prior reading and the coverage of this report.

2. Background - Analysis of current projects and literature available in this area. Analysis of AR libraries which held potential to be implemented, and a discussion of the technologies behind each of the proposed libraries as well as their respective operating system (OS). Further, an analysis on the use of Arduinos in current solutions.

3. Project Management - Providing a breakdown of the development methodology and justification of adopted approaches. This section of the report will analyse the approach to Software Development Life Cycle (SDLC).

4. Requirements - Outlining the stakeholder, system, functional and non-functional requirements of the project; the gathering of user requirements to be able to develop a Minimal Viable Product (MVP) for end users.

5. Design - Defines how the system will achieve its purpose. Analysing the different use cases and activities of the application. Discussing the technical architecture, user interface, and accessibility features of the application whilst acknowledging which parts of process user consultation was necessary.

6. Implementation - Discussion of the implementation and justification of the decisions made. Detailed reports with reference to the backlog of the development, outlining sprints, front-end & back-end development, and system hardware. Included in this section is also reports of how the team enforced ethical audit, and evaluation of the challenges that were faced during implementation.

7. Testing & Quality Assurance - Analysis of the test-driven development approach adopted by the team, and discussion of the various types of testing conducted including code reviews. A formative evaluation is given in subject of testing and detailed reviews of the functional, and non-functional requirements in regards to the original purpose of the project.

8. Project Evaluation - Evaluation of the success of the implementation in meeting the needs discovered in the background research, given in quantitative, and qualitative terms. Analysis of the successes, and failures of the project, and discussion of the advances made. Possible extensions, and further work that could be undertaken are then discussed.

In addition to these main sections there is a glossary, and bibliography of references at the end of the report, along with a number of appendices. These appendices contain:

- Appendix A - Research conducted around end-users and stakeholders during requirements gathering.
- Appendix B - User stories and scenarios in which a user would use the application. Screenshots of the user interface and camera activity and the procedure in which the user would use the application are given.
- Appendix C - System Requirements Specification: includes the purpose, scope, system overview, references, definitions, use cases, functional and non-functional requirements.
- Appendix D - Documentation Plan: outlines the strategy for creating all documentation associated with the software release.
- Appendix E - Testing Plan: in accordance to the IEEE Standard for Software Test Documentation (ANSI/IEEE Standard 829-1983), outlines and describes the testing approach and overall framework that was followed during the implementation phase.
- Appendix F - The Deployment Plan: designed to ensure that the system successfully reaches its users and maintenance of the system allows for new features to be delivered seamlessly. Details further information about the development of the system.
- Appendix G - The testing conducted during the implementation and development phase of the project.
- Appendix H - User and stakeholder feedback given after each sprint about the product.

Chapter 2

Background and Literature Review

2.1 Background

A new exciting segment of the technology sector has sprung up recently – indoor navigation with a catalyst of growth in indoor navigational research. Current solutions on the market cater very well for basic navigation of large open public spaces, though failing to display competencies to more complex navigational requirements coupled with even proportions of navigational, and interactive content with well-presented data. Through the use of AR, this concept can provide an interactive solution for museums.

Since most museums use portable audio guides, user experiences can be vastly improved by the use of a smartphone. Currently only a few solutions can be found; the Orpheo group provide a unique app for each individual museum though their solution is somewhat cumbersome to regular museum users who wish to have a hassle-free setup. The appeal to museums and by virtue of this, museum-goers having one app whereby the user can simply walk into a museum or exhibition, and be greeted with relevant information would be a key differentiating factor [1]. Even though the intention is to create one app for every museum and exhibition for the project, each client would have a large

input on the content of their institution within the app.

If a museum wanted a solution for navigation, due to the low number of museum-specific competitors, would choose standard indoor mapping softwares [2]. While there are many options out there from Google, and Mapspeople [3] who provide this, they lack important exigences that are imperative for museums like heavily integrated augmented reality, and intelligent tour guiding from your location.

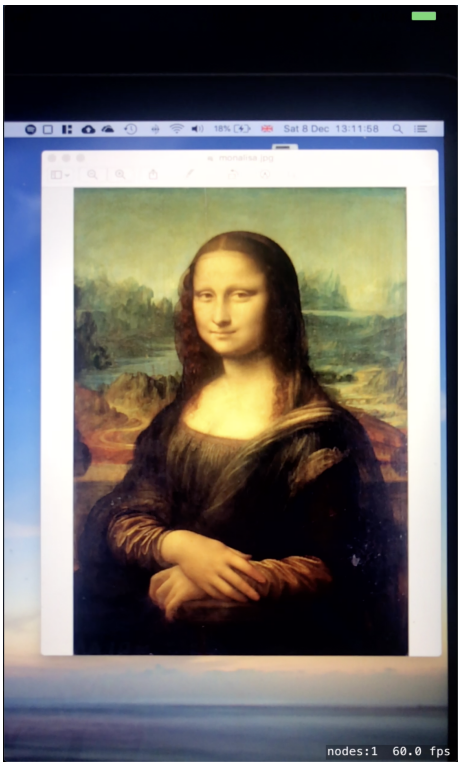
2.2 AR Libraries

The group evaluated three libraries before developing any software. ARKit developed by Apple for its native iOS platform, the ARCore library by Google for Android systems, and Vuforia - a cross-platform SDK built by PTC for Unity/Android, specialising in image recognition.

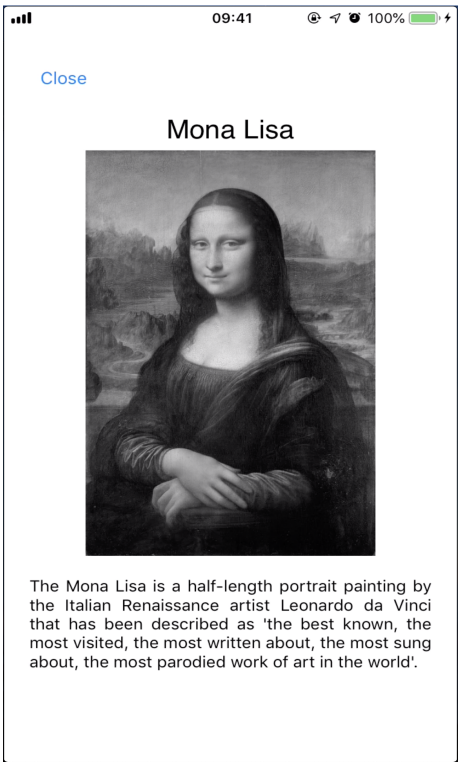
2.2.1 ARKit

ARKit was a strong candidate in becoming the library of choice because of its nativity to iOS devices. The prototype built would hover over an image, in this scenario - the Mona Lisa, and the device would recognise the painting. Subsequently, it would display information about it, and if scanned again, a green tick would be displayed on the screen.

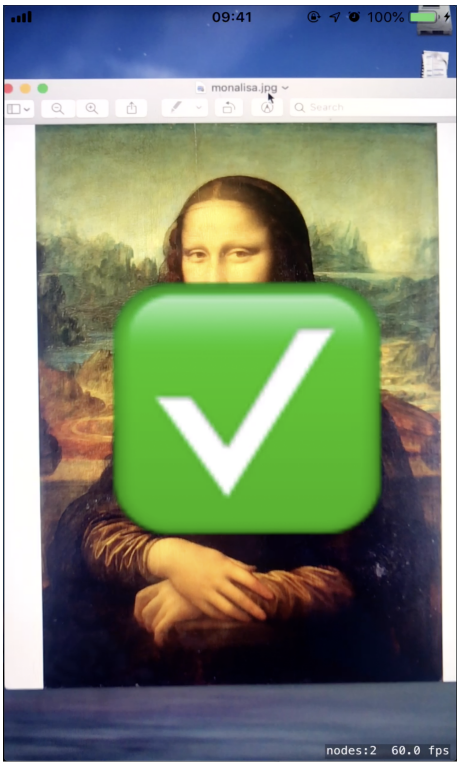
Although the application is only written in Swift, it has more thorough, and straight-forward documentation than Android's ARCore and PTC's Vulforia. This would make the idea easier to manifest because the debugging process would likely run more smoothly. However, due to the lack of the group's familiarity with the platform, only one person was able to partake in creating the image recognition prototype (Figure 2.1), making it an unfit library to move forward with.



(a) Camera over image



(b) Image recognised and displaying information



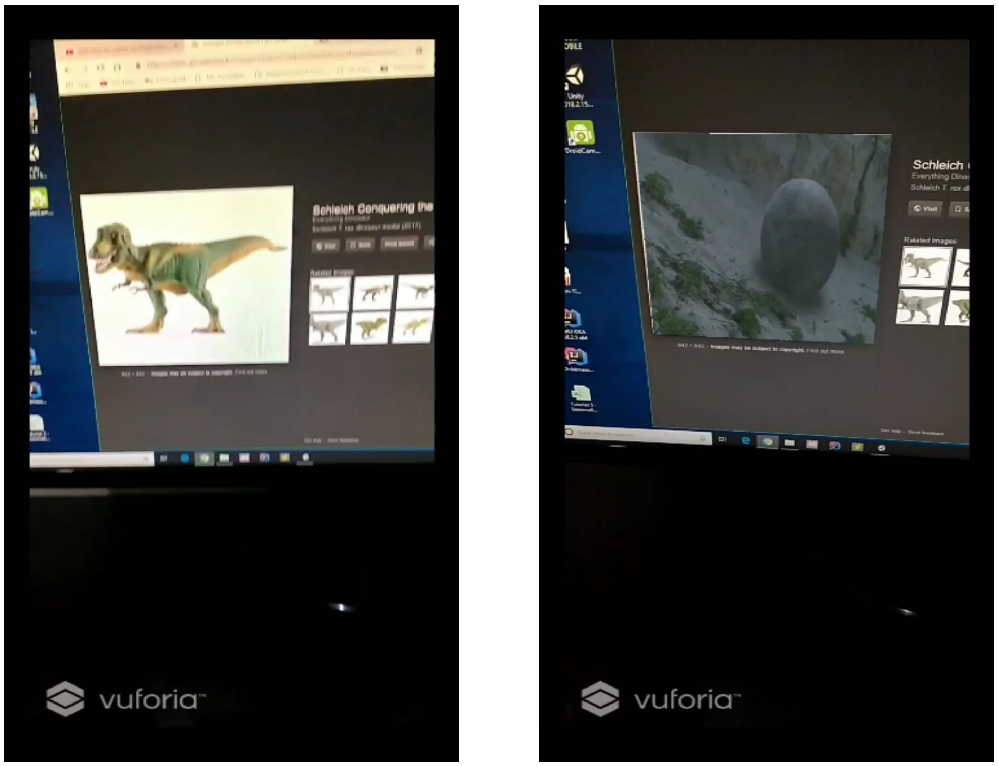
(c) Image scanned before; showing the green tick

Figure 2.1: ARKit prototyping on iOS device

2.2.2 Vuforia

Unity is a cross-platform game engine, used to test a simple AR camera prototype where the device's camera hovers over an image, and displays information about that image on the device. The application was built using Vuforia, an SDK that enables recognition, and tracking of image targets. This library can be used for the exploration case in the use case model. Although, there are a lack of tools for locating user's current location compared to Android.

The Vuforia library was used to test a simple AR camera prototype where the device's camera hovers an image, and displays information about that image on the device.



(a) Camera over image (b) Video superimposed on top of image

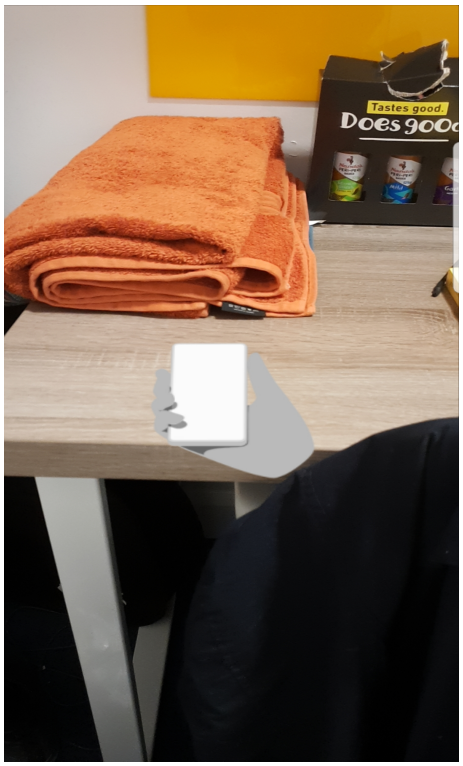
Figure 2.2: Vuforia prototyping on Android device

2.2.3 ARCore

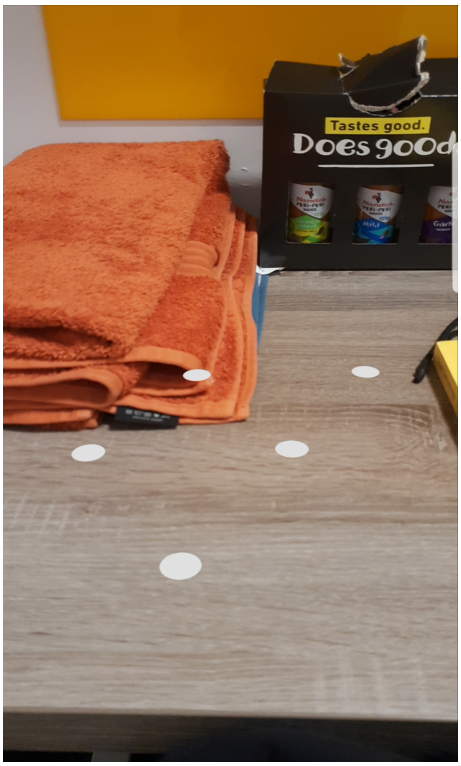
ARCore was used to create an AR experience (Figure 2.3) giving the user the ability to superimpose a 3D object when the camera is focused on a flat

surface. This prototype simply recognised a flat surface, and when detected the user could place the Android robot on the surface.

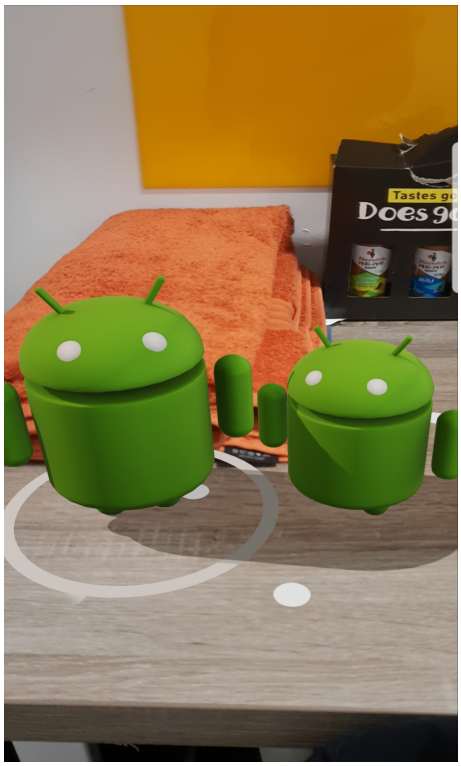
During the process it was found that unlike ARKit, the library is suitable with Java/Kotlin, hence the group would have an easier time acclimating the nature of ARCore. As ARCore is a native Android SDK, it can be integrated with other Android features such as Android's GPS, and Android's WiFi libraries. Despite this, the documentation was scattered and vague, making it difficult to debug with. After exploring this prototyping process, and discussing the accessibility of ARCore, and Android to the stakeholders, the group distinguished it as the most suitable library for the project.



(a) Initial view



(b) Detection of surface



(c) Objects superimposed on surface

Figure 2.3: ARCore prototyping on Android device

2.3 Software Architecture

To aid with selecting AR technologies, two operating systems were analysed to identify the potential risks, and verify the quality of software prototypes in design.

2.3.1 Android

An open-source OS developed by Google, it provides an SDK for development. It uses the *.apk* extension that can be downloaded, and installed on mobile phones. The Android emulator is nearly three time faster in CPU speed compared to Apple, allowing for smoother development of non-camera features. GPS location data is easier to use in Android due to how packages are imported into files, so there is low coupling. Further, the creation of user interface elements is straightforward, and can be executed on a wide variety of devices. However, it is harder creating more complex layouts and animations, and there is a steeper learning curve in Android development compared to other operating systems.

2.3.2 iOS

iOS is a mobile operating system developed by Apple, providing the ARKit SDK for development of the application. Only Swift is compatible with iOS, though this is notoriously an easy language to learn, and develop apps on. Apple devices have more accurate accelerometers, and gyroscope, enabling stable and fast motion tracking [4]. There are also fewer dependencies compared to Android, hence application load time is faster. Yet, there are limitations of the ARKit map, and the application would only support fewer devices.

2.4 Hardware

As part of the exploration it was recommended, based off lengthy research, and the project's domain expert's advice the need for a physical beacon.

This role of this beacon was to broadcast a signal; three main current technologies exist exist: optical infra-red, Bluetooth and, WiFi. Speaking to the project's domain expert, the former – underpins their navigation technology and yields pin-point accuracy – important in dealing with small indoor spaces, but this is expensive; Bluetooth, while less accurate is far cheaper to manufacture, and develop; the latter-most offers the least in accuracy, and not too expensive. From this probing, Bluetooth became the team's solution.

Looking at relatable past projects, the favourable direction was using a Arduino (UNO R3 Mega 2560) or the Raspberry Pi 3 Model B+. Both solutions gave a clear, small, and light-weight canvases on which to design. With a smaller physical footprint – and in keeping with the project's eco-friendly policy, the Arduino was cheaper. This was a priority to give the technology a wide-spread appeal.

The Arduino platform was kitted out with a Bluetooth HC-05 RF transceiver. This setup was able to broadcast a simple Bluetooth signal where software can decipher a signal strength from which a distance could be determined. It became apparent that there was a need to further invest in more beacons to give more accurate, and reliable results via triangulation. This was not a suitable solution as the technology was not as accessible, and available to a wide range of applications – an increase in cost would hinder this.

Chapter 3

Project Management Processes

3.1 Development Methodology

In order to produce high quality software, a development methodology was selected to monitor, and control all aspects of the project deliverable according to the stakeholder requirements within the expected milestones.

Six variables were identified that should be managed throughout the project [5]. The **cost** of the project has to be affordable, and need to factor actions that lead to overspending, or opportunities to cut cost. Allied to this, **time** is the only cost in the project. Ensuring the product delivers on its **quality**, and is fit for purpose along with the **scope** where there is an agreement between stakeholders. It is important not to deliver beyond the scope as this is a common source of delays, and uncontrolled change, known as 'scope creep'. Managing the impact of **risks** is essential, but to assess how much risk there is pivotal. Finally, maintaining the **benefits** of the stakeholders is of interest so that there is clear understanding of the purpose.

With these factors in mind, the waterfall, agile, and lean methodologies were considered for the project. Waterfall is a fixed sequential framework where it

is linear in nature. Each of the eight processes in Waterfall needs to be completed before moving onto the following stage. Despite the easy understanding of the methodology due to strong documentation approaches, this did not suit the project as changes cannot be easily accommodated [6]. If there were errors during the implementation stage discovered during testing, it would be expensive, and difficult to go back. Further, there is greater reliance over the initial requirements; if these are incorrect they can be detrimental to the project.

Lean focuses on removing waste from a process in order to deliver products quicker, lowering delivery costs without sacrificing quality, and providing a flexible response to users. This model narrows down to completing the MVP before considering extra features or content, though sometimes the full potential of the project is not reached. Similar to waterfall, the documentation needs to be explicitly precise in the requirements. Further, there were many skills to be acquired by the team throughout the project, and learning on to go is not possible in the lean methodology.

Agile is an iterative, incremental approach that is open to changing requirements with frequent feedback from end users throughout the process. By breaking down the project into small chunks, it makes it easy to prioritise, and add or drop features mid product, which is impossible in traditional waterfall projects. Breaking down tasks into iterations can allow the team to concentrate on high quality implementation. Such an approach delivers measurable results quickly, engages stakeholders earlier, facilitates a more frequent, and reliable product release cycle. However, due to the quick space of the process, the quality of the delivered software could decrease.

With these benefits in Agile, the team decided to adopt this methodology with the scrum framework. Scrum allows for a highly adaptive change in requirements during the entire cycle, and high transparency. The product owner is constantly in touch with the team [7], and daily stand-up meetings eliminates any misconceptions and confusion. Risk is controlled since issues are identified in advance before getting unmanageable. Kanban was not used

since the project required iteration, and prioritising tasks on hand is harder with a Kanban board as all tasks are placed at equal importance with this framework.

3.2 Software Development Life Cycle

In the scrum team, the project supervisor served as the product owner responsible for maximising the value of the product resulting from work of the development team. This ensured the end product delivers value to the end user. The project lead took on the role of scrum master to ensure that stakeholder requirements were met, and to be accountable for the delivery of the sprint goals. The scrum master also took on the role to set and adjust the priorities of the product backlog since they had a greater understanding of the project than the product owner. Having a flat line structure allowed the development team to be self-organising, and cross-functional, with all the skills as a team necessary to create the product increment. According to the framework, scrum recognises no sub-teams in the development team [8], and this was avoided during implementation. The whole team was accountable for all actions rather than those who have specialised areas of focus. Further, this allowed for dynamic allocation in resources to certain tasks when problems arose.

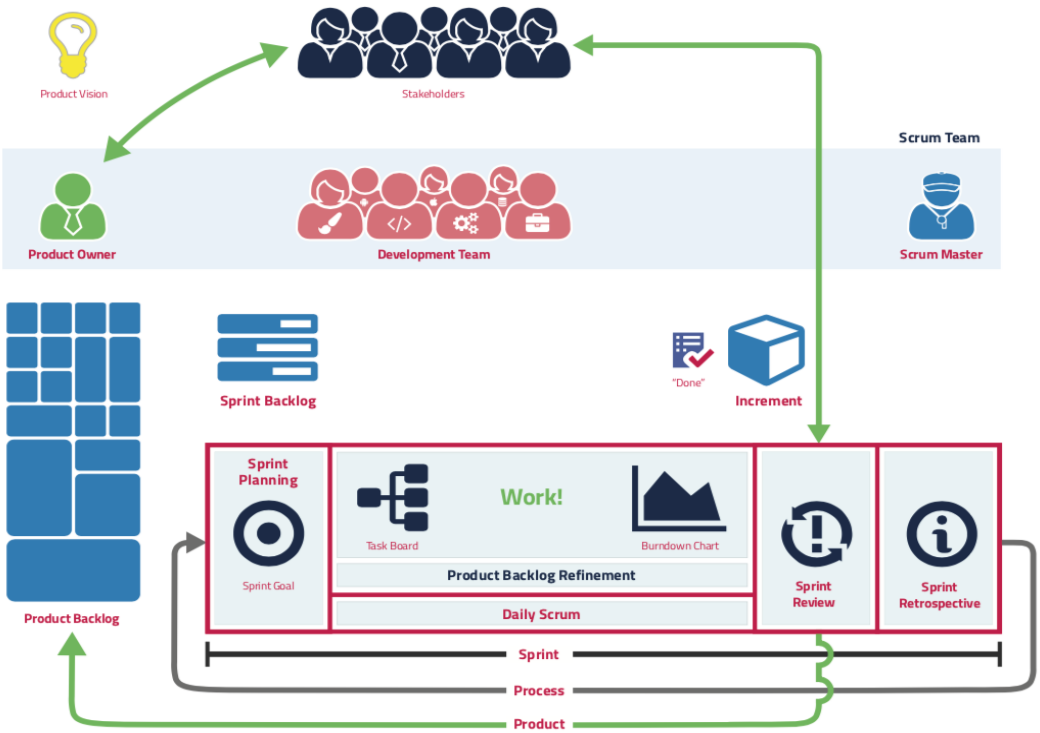


Figure 3.1: The Scrum Workflow [9]

Scrum events took place such as sprint planning, daily scrum, review, retrospective, and backlog refinement. Sprint planning always took place on the first day of the sprint where three key indicators are outlined, what can get done in the sprint, how the work is done, and the sprint goal [8], providing the development team an objective. This event usually lasts about one to two hours for a two week sprint. Due to careful planning at the start of every sprint, successful sprints ensued during implementation, and able to deliver on requirements.

Conducting daily scrum proved challenging from the beginning as members were not always on campus at the same time; to resolve this, stand-up would take place at least four times a week. In both terms at the start of every lab, and every supervisor meeting, a ten minute stand-up would happen in person. The other two meetings would happen virtually on Google Hangouts for ten minutes, the day before a milestone deadline (Thursday), and either Saturday or Sunday - depending on the group's availability. Using this model allowed for quick decision-making, and removing any obstacles the team faced.

Sprint reviews took place to audit the work completed, positives, negatives, and problems of the sprint. At this point, the group would decide if the tasks assigned were at "done" status, subject to review of the product owner. All requirements outlined within a reasonable time constraint must be reached to achieve this status. Outstanding tasks to be completed would filter through to the following sprint planning, and the product backlog will be edited accordingly. This formed part of the sprint retrospective, but mainly focused on how well the team administered the methodology. Other key stakeholders are consulted here, giving any important feedback to the completed work.

The team used a scrum board in order to track tasks and make changes during sprints, promoting transparency amongst the group, and ensured accountability was retained. The project used other scrum artifacts discussed in chapter five and six.

3.3 Test-Driven Development

Another development methodology used was test-driven development (TDD). Before completing implementation for each sprint, unit tests were written so the focus when writing source code is to make all the tests pass. This allows for software architecture to be thought out beforehand, and ensure the core functionality is acting on the requirements. This methodology is proved to reduce defect density, and reduce cost since testing time is dramatically reduced [10]. Further, TDD fits into Agile's iterative approach, providing ease of allocating resources in sprints. Business-driven development, and acceptance test-driven development were considered, but a more thorough and rigorous testing system that focused mainly on the implementation was favored to deliver a working MVP.

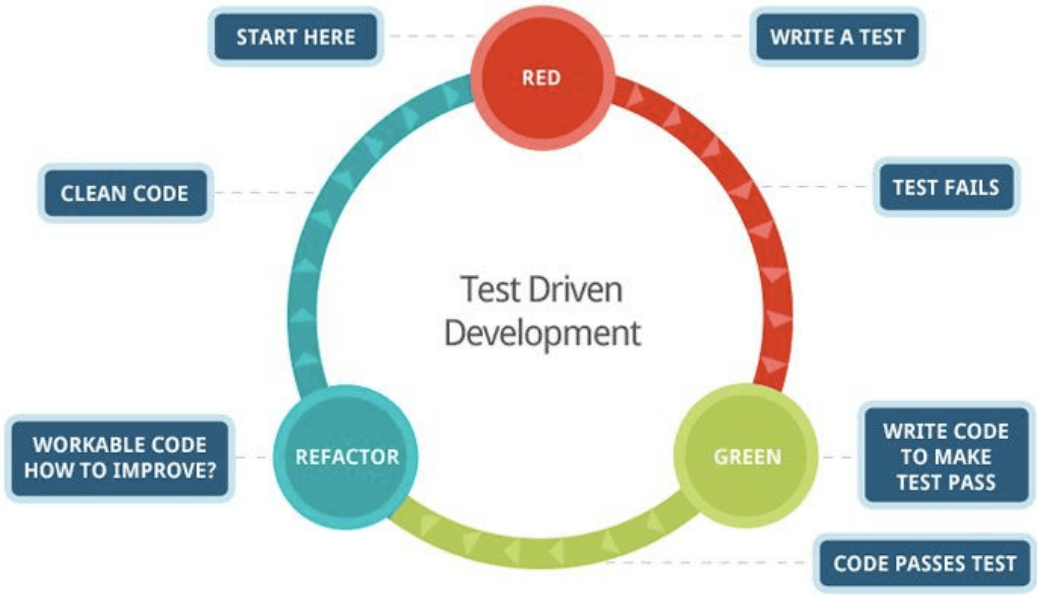


Figure 3.2: The TDD Workflow [11]

Chapter 4

Requirements

4.1 Stakeholders

Stakeholders hold a very important position during a product's development; influencing the outcome of a project by the way of its scope, means of function, overall effectiveness, and success. All appropriate audiences to this rank were detected in foresight of this impact.

Fundamentally, the group at the centre is the end user; in near-to-every circumstance, if those at the bottom on any sort of hierarchy reject an idea or practice, those above will not benefit through its implementation. Furthermore, this is exacerbated as this project will largely change the end user's experience of a museum which if rejected, may lead to a decline in visits, and thereafter, in funding. To understand the project's potential impact, expectations from visitors and members of museum staff of the Science Museum, and the Natural History Museum were noted.

Though this project was the product of an academic environment, it is not too unlike the real world in that there were directions set by this project's supervisor, and product owner, Dr. Basil Elmasri and standards set by Dr. Nick Hine – the module leader and the project sponsor.

As this project is framed around being a main point of interaction between

museums and visitors, it could affect the experience, and potentially having an adverse effect. Those in industry, the technology sector and creators of exhibits or those who know about the representation thereof have a stake in this project. It was a priority to get experienced understanding, so a line of communication was created with a company called Fuse who have an incubator company, Pathfindr (navigate), providing indoor navigation solutions, via their Managing Director, Matt Isherwood. He served as the project's domain expert.

Seeking opinions from those in the business of being responsible for exhibits or creative works, a priority was to gauge this stakeholder via showing prototype designs and ideas. Getting the full scope of their opinion in relation to this project as the project will directly change how the end users interact with exhibits.

Finally, the development team behind this project, have a large stake in its success. Not only that if there was not a quality report, but this new part of the technology sector is in its infancy so needs innovative ideas.

4.2 Gathering

End users, and other key stakeholders such the domain expert were consulted during the design stage to clarify user needs and requirements. In addition to online interactions with stakeholders, the group conducted field research, and focus groups with potential users at the Natural History Museum, Science Museum, and V&A museum to obtain more nuanced inputs from stakeholders. During prototyping, drawing, and designing the user interface was not as simple as bringing ideas to life - it was imperative that users were consulted when making these decisions. More specifically, the group consulted fine art students studying at the University of Oxford who provided a more in-depth perspective on user experience in a museum setting. Some students said that they would like for the application to provide further information about a

specific exhibit that the museum does not have on display.

Upon consulting the users, it became easier to actualise the designs put forward - acknowledging the fact that users had all required a system that was relatively simple to navigate around. However, users were consulted when it came to the technical aspect of the project. Deciding which medium would be best to track user location was a question that arise, Isherwood was consulted about this, and brought to light the plethora of solutions that were available to begin development; primarily highlighting that the use of Bluetooth beacons would be most efficient to implement given the scope of the project.

Feedback was positive and encouraging from staff describing the current system of charging for maps, and their current app solution as inadequate, often citing to help visitors having simple issues navigating.

4.3 System

The system requirements is an Android mobile application, written in Java, using Android Studio. The group chose Android because it is accessible on more devices and ARCore creates a greater AR experience. The SRS was designed to show the structured collection of the requirements of the system along with its operational environments, and external interfaces. The purpose, scope, and system overview serve to provide an outline of the overall platform. Definitions of both the system, and user requirements describes fully what the software will be. This document creates a bond between stakeholders, allowing them come to a conclusion about what should be implemented, and how it should be implemented - maintaining transparency throughout the process.

Constraints, Assumptions and Dependencies

1. **Internet Connection**: The application would not be able to query mapping services or have access to exhibit information otherwise.

2. **Android**: Users of this application are Android device users that requires assistance in museum navigation. Devices that support basic dependencies of the application is expected for proper user experience.

3. **ARCore enabled device**: User's device must run on Android version 5.0 or later, and have API level 24 or higher, for them to have access to the application's AR navigation.

4.4 Functional

Fundamentally, the user should be able to use the application to navigate between the given current location and the required destination. The application should display navigational routes in real-time from the users given location, calculating the shortest possible route. Displaying navigational routes in real-time is made possible by the superimposition of a 3D navigational line rendered on the user's camera screen through the use of augmented reality technology. The application should make use of camera recognition, detecting artwork/exhibits, and displaying additional information about the exhibit of interest. Additionally, based on stakeholder requests, when the user arrives at their destination, the system should give a recommendation based on their route.

4.5 Non-Functional

This considers the usability of the application, describing aspects such as performance, user actions, and safety.

1. **Performance**: The system should respond quickly to user input, e.g user wants to find more information about an art piece or whenever they search up a location. The system should not require extensive CPU usage, it should not slow the device down inconveniencing the user.

2. **Usability**: The system should have a simple layout, with appropriate colour used in appropriate contexts. The language used on the app should be easy to understand for the users. Having done research based on the user preference, it was discovered that users dislike too much text on their menu screen.

3. **Data Usage**: Data usage should be kept to a minimum, only querying the relevant information (user location and exhibit information).

4. **Safety**: The system needs to have the ability to detect immutable objects obstructing the user's path. This can reduce common user accidents when using a mobile phone whilst walking.

5. **Security**: Ensuring that the device's current location cannot be obtained by unauthorised third-party users is crucial in ensuing the security of using the platform.

Chapter 5

Design

5.1 Technical Architecture

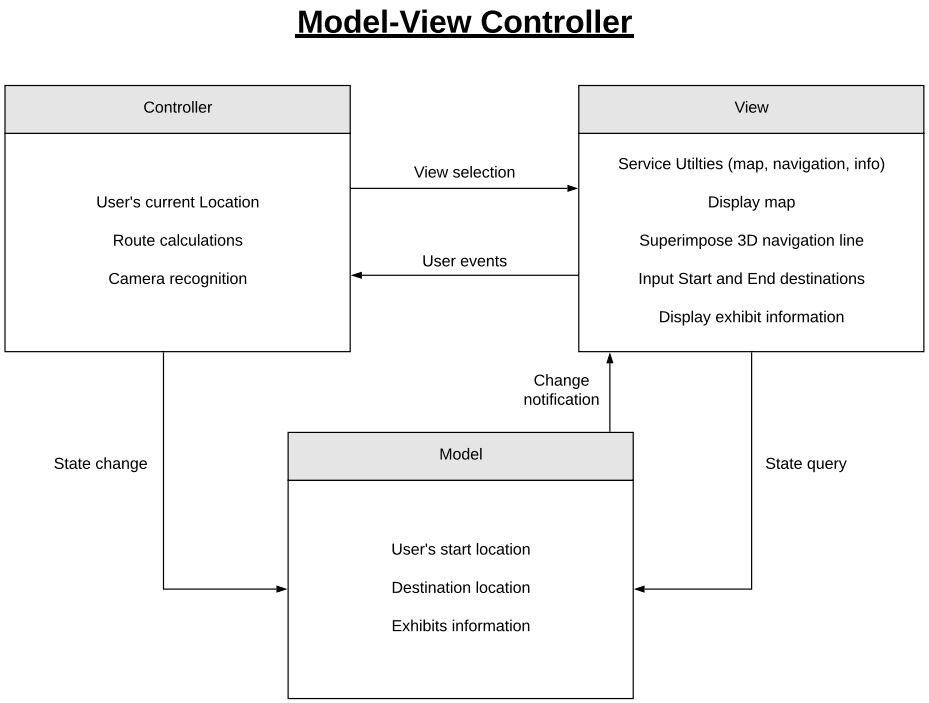


Figure 5.1: MVC

As a user-based application, a robust and reliable technical foundation is necessary in order for it to work successfully. It was quickly agreed upon that, the Model-View Controller (MVC) architecture (Figure 5.1) would be the most

relevant to creating the optimal version of the application.

5.1.1 Model

The model of the MVC is the core component of an application. Data is stored and manipulated in this case, the users location, destination, and information held by the exhibit. These were to be manifested as a hard-coded location within a virtual geo-space to simulate the venue, and strings together data regarding the destination.

5.1.2 View & Controller

Android devices were the conclusion from prototyping as the platforms to develop for. The construction of the front-end used Android Studio, and the back-end used ARCore, Java, and Kotlin. The front-end involved a login screen, a map screen, an information screen, and navigation screen; all were which controlled by the Java, and Kotlin code in the back-end.

5.2 Models

The following models are based on user interaction with the application.

5.2.1 Use Case

The use case diagram is crucial because it clarifies the lifecycle of a user during application runtime. Developing user-based applications like this becomes more manageable.

The use-case diagram (Figure 5.2) deals with the two states that the user could be in; the lost state, and the exploration state.

The lost state is initialised by picking up the user's current location (once they have requested a destination), and calculating the shortest route between them, and their requested destination. This is finalised by collecting the user's current location, and displaying it to them via the superimposition of a 3D line in their camera view. From this step, the user just needs to follow the route until they arrive at their destination.

If the user considers another destination, or starts the application with no destination in mind. They have indirectly declared themselves to be in the exploration state. This state initialises itself by looking at the users past visits (if they have any), and current location to calculate recommendations. From this point on, once the user chooses a destination, the exploration state is identical to the lost state. The user follows the superimposed line and reaches their destination for the lifecycle to either begin again, or end.

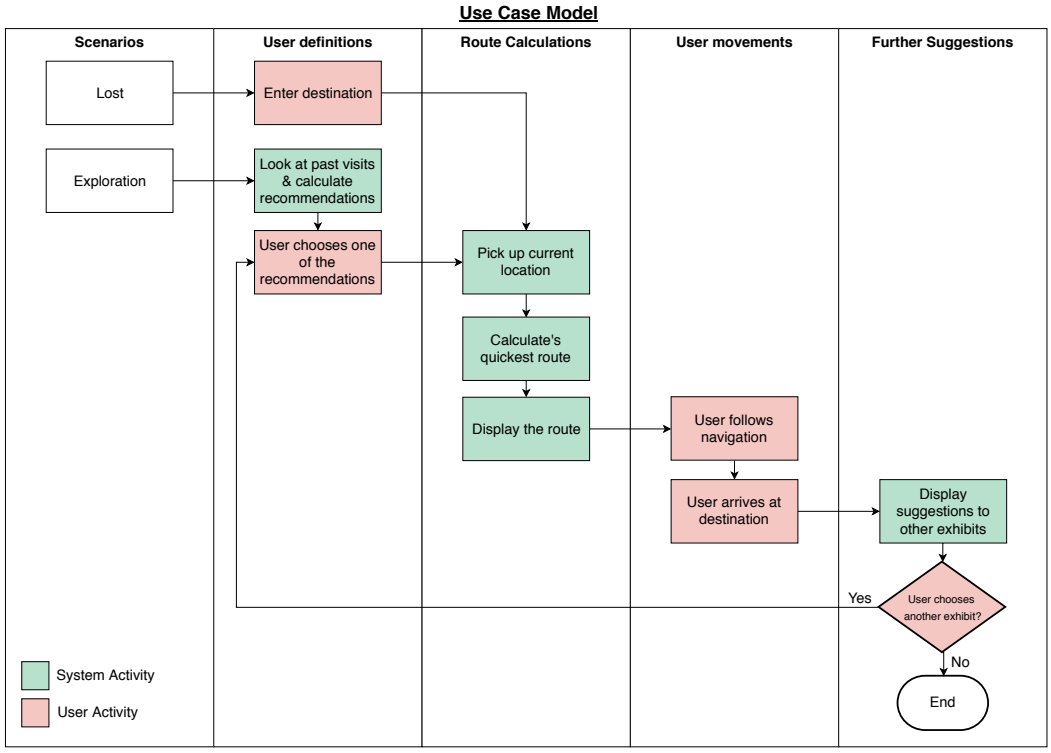


Figure 5.2: Use case diagram

5.2.2 Activity

The activity diagram (Figure 5.3) distinguishes the details in two sections of the use case diagram, user definitions, and route calculations.

The route calculation process in the application is linear (Figure 5.3). The user location is received from the GPS - distinguishing what venue the user is in. The building makeup is then requested from the server in order to make an analysis on the geospacial coordinates that the user will have to traverse. After the analysis is complete, the route between the user and destination requested in the user definition section can be calculated, and the route is superimposed.

User definitions that the activity diagram (Figure 5.3) deals with are case specific to the exploration case. If the user would like to explore their venue, and they do not have one on the application; they are prompted to make an account so that the application can provide optimised recommendations. Inf they do, their previously visited locations can be used in presenting recommendations.

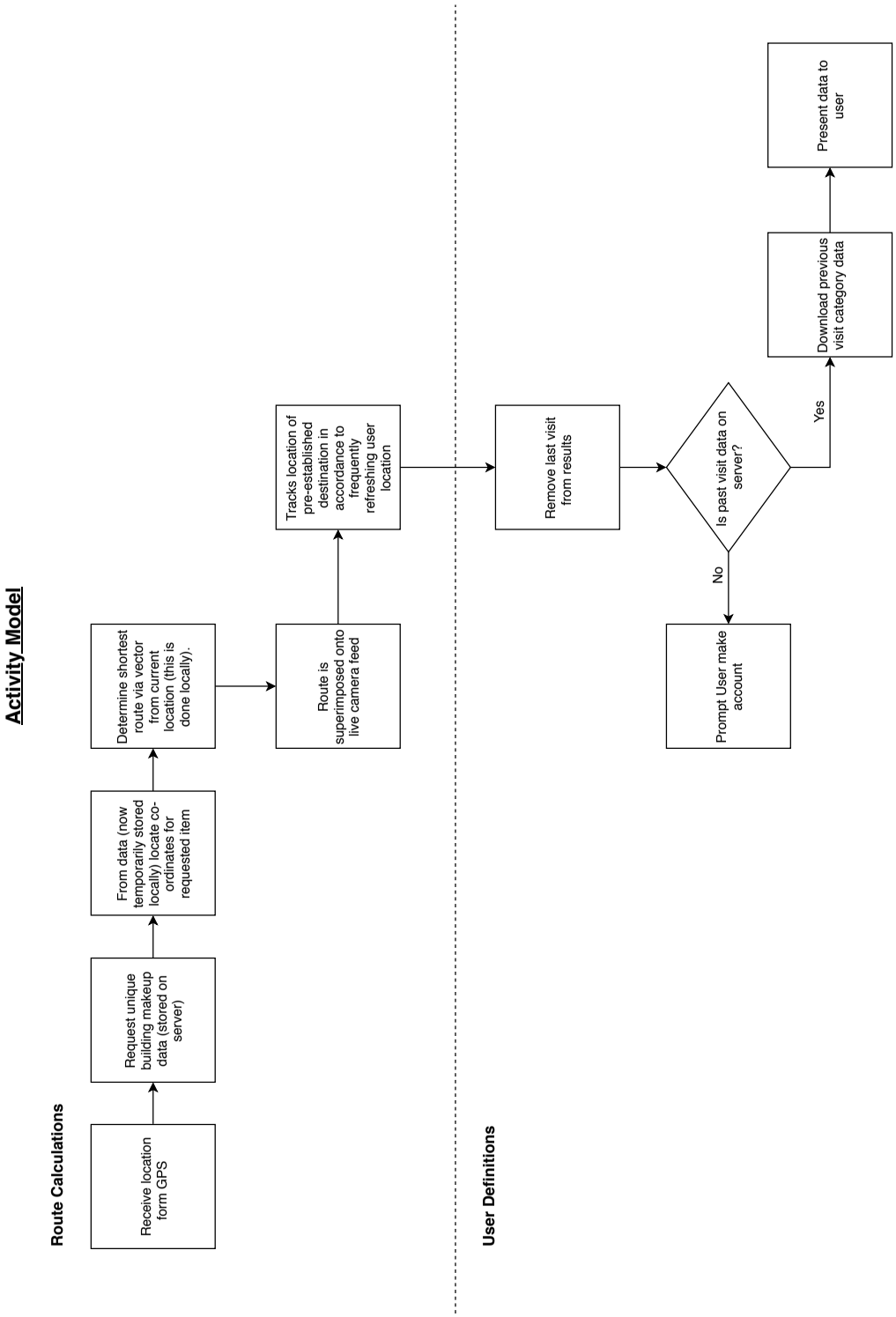


Figure 5.3: Activity diagram

5.3 User Interface

The project lends substantial importance to its user interface, and experience. As it will be used by people with a wide range of technical ability, the aim will be to make the app as simple as possible without having an impinging effect on any service the end product will feature. This prerequisite was clearly outlined in the surveying of museum visitors, and staff alike. The first mission was determining what interfaces, and experiences currently exists within the museum sector. Many museums employed simple interfaces but due to their mass-manufacturing, their design felt unoptimised with simple bare-bones media not beyond text and images. Furthermore, this design would fail to deliver anything more complex than texts and images.

The approach to the UI/UX prototyping was to create different interface mock ups, and exhibit them alongside existing solutions. An initial storyboard, and lo-fi prototypes was drawn up, and three potential interfaces (Figure 5.4, 5.5, 5.6) were designed and shown to stakeholders. The feedback gained from the stakeholders was invaluable in the process as it allowed for the group to consider all aspects of the prototypes. It was decided that a final version of the application's user interface would be designed, implementing all the positive attributes, and combining it into one (Figure 5.7), whilst also considering the negative attributes.

Prototype 1

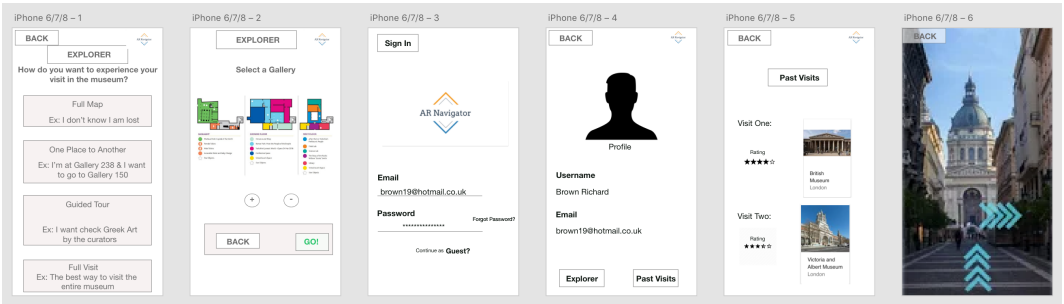


Figure 5.4: Overview of UI Prototype 1

Prototype 2

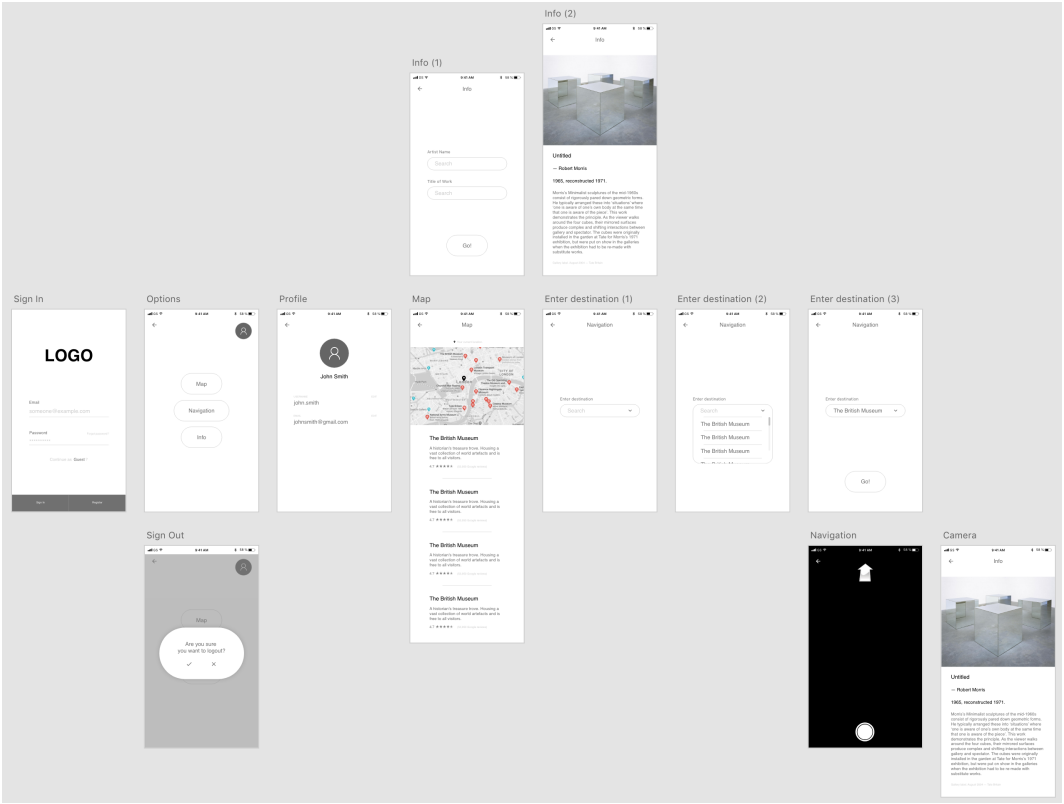


Figure 5.5: Overview of UI Prototype 2

Prototype 3

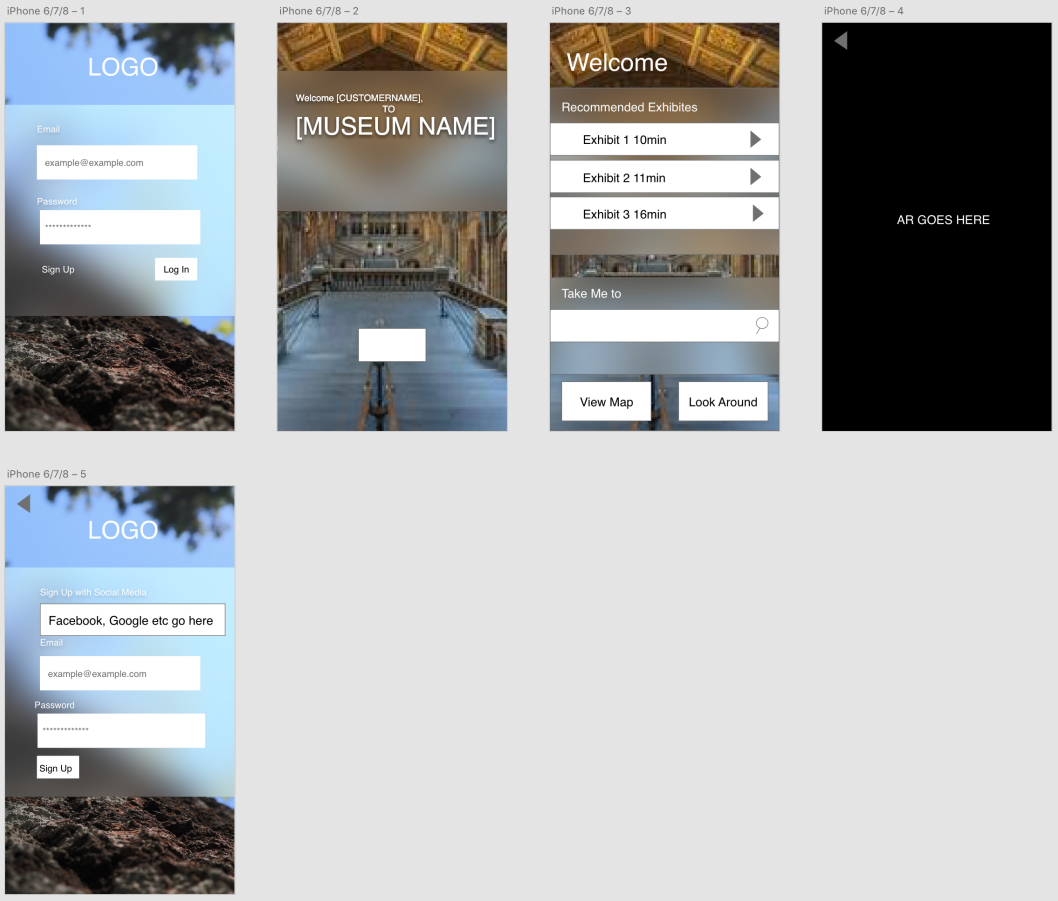


Figure 5.6: Overview of UI Prototype 3

Final Prototype

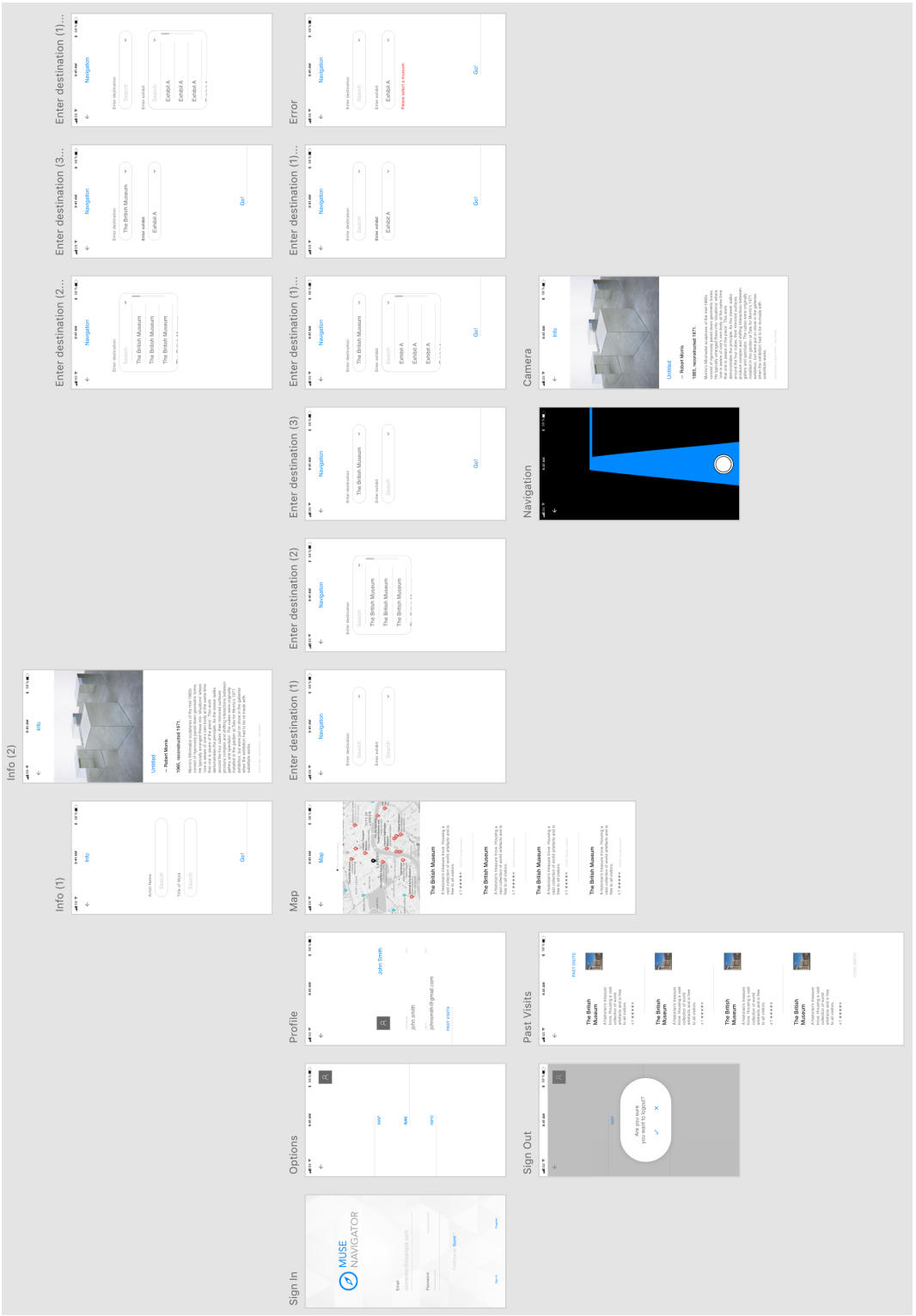


Figure 5.7: Overview of final UI prototype

5.4 Accessibility

The Web Accessibility Standard outlines key accessibility features for the group to consider. Some accessibility options would include features such as relating to mobility, colour perception, and literacy. Almost all museums have physical accessibility features managed, and features designed in the application have them incorporated in the design.

1. **Screen Reading**: Users tend to rely on screen readers to help them interact with the app, including UI elements. This feature would allow the user to use the app without any difficulties since the layout is simple and straightforward.

2. **Scale**: Scale was used to allow for users to zoom, and resize elements in order to help those with visual impairments, particularly for images that include words. Font sizes across the application are not too small to ensure users of all ages can use it. Future iterations will include more scaling options through the app settings to allow for this.

3. **Vision**: The text and UI in the app is designed to support high-contrast theme. Whilst colour is important, it must not be the only channel of communicating information. For instance, users who are colour blind would not be able to distinguish some colour status indicators from their surroundings.

Chapter 6

Implementation

6.1 Backlog

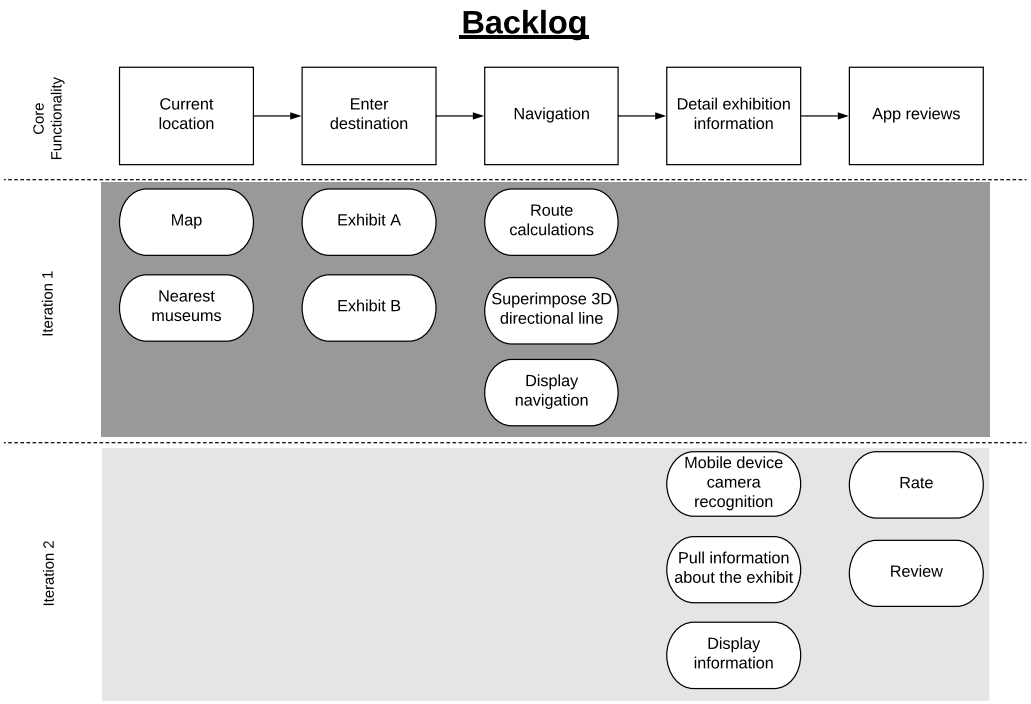


Figure 6.1: Inital Product Backlog

The product backlog was used as the single source of requirements in order to prioritise releases, and allowed for transparent iteration planning. From the requirements and project scope, each of the five core functionalities were broken

down into its major components. Considering the time estimates, the scrum team decided that to demonstrate originality in the project, and to build the base functionalities first, the route calculations and the AR implementation were to form the minimal viable product. Features which required a database such as exhibition item/image recognition, user profiles/logins, and building servers were placed in future releases. These components would take longer than the amount of development time available along with the base functionalities. The team recognised the added time for the construction and integration of a database to the app would increase the risk in not delivering a functional MVP. As a living scrum artifact, the backlog changed accordingly when there were requirements, enhancements, and fixes during the implementation and testing phases of each sprint. In practice, the backlog was maintained on the scrum board hosted on Trello.

Function	**Estimate Time (Days)**
Displaying map	0.5
Finding nearest museums	3
Getting start and end locations	5
Route calculations	10
Superimpose 3D directional line	12
Displaying navigation	10
Mobile device camera recognition	2
Retrieve exhibit information	10
Display exhibit information	1
Rating the app	1
Reviewing the museum visit	3

Table 6.1: Table of time estimates for each function

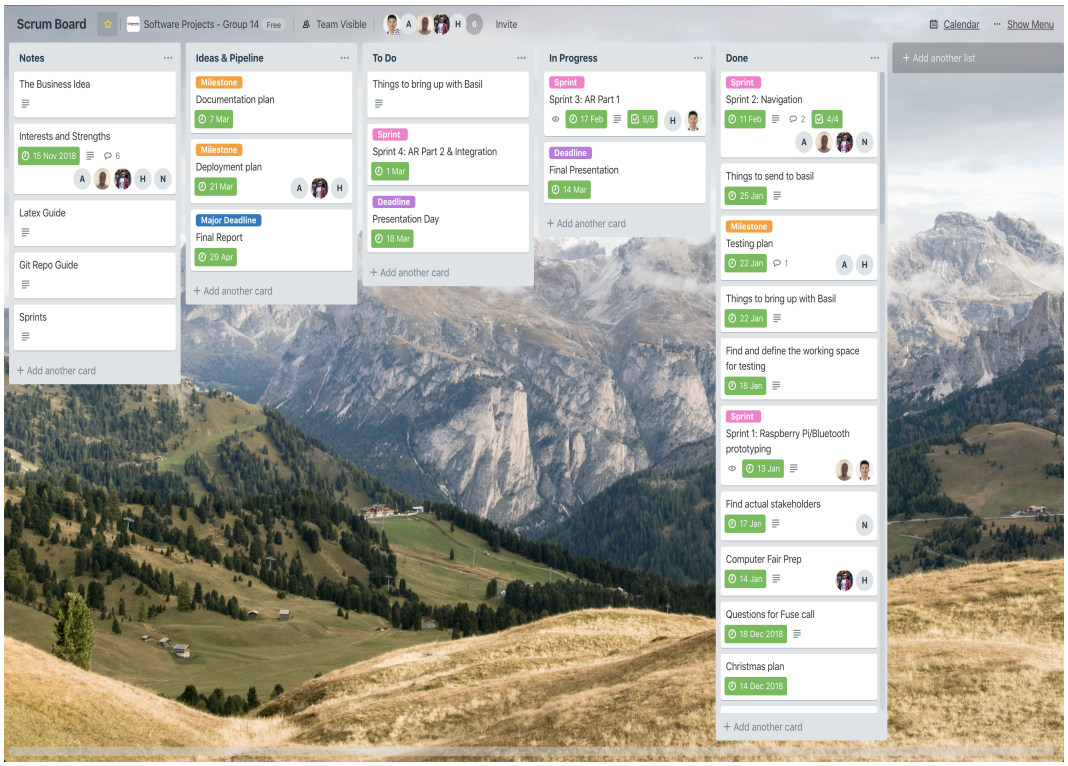


Figure 6.2: Snapshot of scrum board in action

6.2 Sprint Outlines

During implementation, four sprints were conducted each lasting one to two weeks. The first sprint lasting one week with the goal of building a beacon for calculating distances between users and objects. The following sprint focused on route calculations; being able to calculate the route between two points in a given space. For the AR part, this was divided into two separate sprints, with the initial one focusing on rendering, and anchoring objects in real-time, and afterwards the integration with all base functionalities.

6.3 Front-end

Front-end development was carried out using Java on Android Studio.

It was carried out in the early stages of implementation due to the fact that the team had already received user feedback for the UI/UX prototypes. After

combining the positive features of the initial design prototypes into one, it became easier to visualise the end result. With aims to create an user interface that is user friendly, and intuitive, various features of the application are self-explanatory.

As soon as the application is opened, the user is faced with the login page asking for credentials. The user can keep track of their past visits to museums and review various sites; this put in place future developments but not included in the MVP. However, if the user does not have an account they are able to 'Continue as a Guest', and use the services freely without saving any data. When the user has passed the login page, they see the menu.

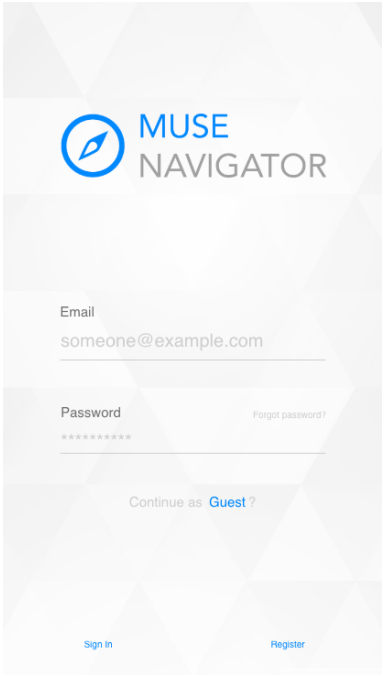


Figure 6.3: Log-In Screen

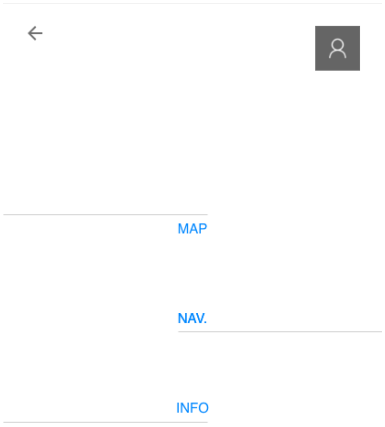


Figure 6.4: Menu Screen

There are three different services that the user can utilise. The 'Map' which shows the user their current location, and surroundings on a 2D map. The 'Nav' menu takes the user to the screen which the user has to then enter their desired destination or exhibit. After all the information has been entered by the user, the user's device displays the AR-enabled camera with the highlighted navigational line directing the user to their destination. However, if the user's device is not compatible with Google's ARCore then the user will not see this

screen, and instead faced with an error message. Lastly, the 'info' page will allow the user to see additional information about an exhibit of their choice - this also requires the user to fill out a form.

```
void enableArButton() {
    ArCoreApk.Availability availability = ArCoreApk.getInstance().
    checkAvailability(this);
    if (availability.isTransient()) {
        // Re-query at 5Hz while compatibility is checked in the
    background.
        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                enableArButton();
            }
        }, 200);
    }
    if (availability.isSupported()) {
        Intent intent = new Intent(MenuActivity.this, ARActivity.
    class);
        startActivity(intent);
        // indicator on the button.
    }
    else { // Unsupported or unknown.
        Context context = getApplicationContext();
        CharSequence text = "Android version not compatible";
        int duration = Toast.LENGTH_SHORT;

        Toast toast = Toast.makeText(context, text, duration);
        toast.setGravity(Gravity.TOP|Gravity.LEFT, 0, 0);
        toast.show();
    }
}
```

Listing 6.1: Enabling AR from MenuActivity

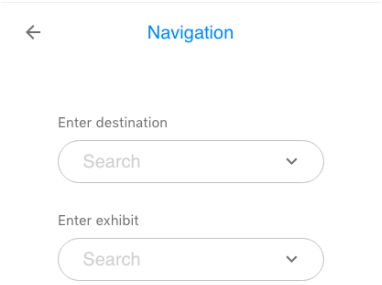


Figure 6.5: Enter Destination

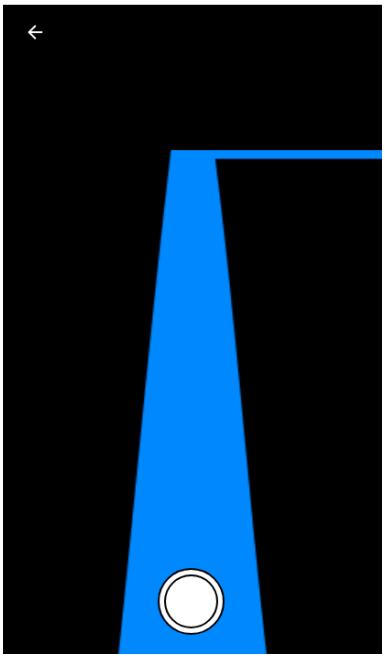


Figure 6.6: Navigation Screen

As a result of all the planning and prototyping, when it came to actualising the design in Android Studio it was as simple as writing code for implementation - no further designing was needed, with exception of minor changes such as font sizes.

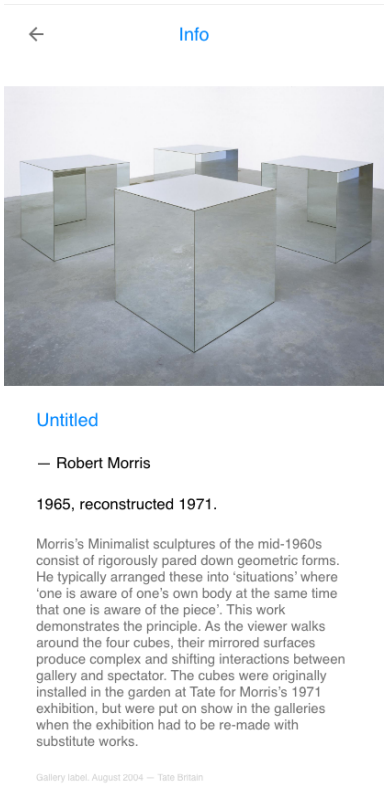


Figure 6.7: Additional information

6.4 Back-end

The back-end implementation consisted of developing the application's AR, and route calculations systems, along with integrating, and testing the two constantly throughout the process. Since most of the development team were unfamilar with Android Studio, time over the Christmas vacation was used to gain a firmer understanding of the IDE to reduce the amount of time during implementation to learn basic new skills.

6.4.1 Route Calculations

The scrum team delegated both a sprint, and a consultation with the domain expert in order to ensure this was done correctly.

This concluded in the route calculations using the A* path finding algorithm.

Bluetooth was an option in development however, the errors that Bluetooth navigation provided were difficult to handle. Also, the A* algorithm is a better fit for the back-end, since they are both complimentary to augmented reality.

Developed in Kotlin (for development speed), and then converted to Java (to be integrated with the AR segment), the user first enters a destination. That destination becomes a goal node in the context of the route calculations. An attempt to calculate the shortest route using a graph, then outputs that route via the 3D line. As the user traverses that line, their location is constantly updated relative to the goal node so that when the user reaches it, and an alert is shown. It also ensures that the path constantly updates in the case that they ever lose their way.

6.4.2 Augmented Reality

The application's AR research began with the group understanding how Android's ARCore operates with the camera. It was found that the library evaluates the camera's pixels as a set of 2D nodes - it then maps those 2D nodes onto a 3D plane using depth perception This meant that in order to superimpose the 3D line. The output of the route calculations would have to have little, to negligible margin for error. Otherwise, the user could be guided incorrectly.

Once the research was completed, the group delegated a sprint and some research time beforehand to setting up the AR environment that would be utilised by the route calculation system. Therefore, over the sprint, a testing plan was formulated with seven tests to ensure the environment worked in a sound fashion.

```
1 @Override
2 protected void onCreate(Bundle savedInstanceState) {
3     super.onCreate(savedInstanceState);
4     setContentView(R.layout.ar_activity);
5
```

```
6       // Identify AR environments
7       baseArFragment = (BaseArFragment)getSupportFragmentManager().
    findFragmentById(R.id.sceneforem_ux_fragment);
8       arSceneView = findViewById(R.id.sceneform_ar_scene_view);
9
10      setupModel();
11
12      baseArFragment.setOnTapArPlaneListener(new BaseArFragment.
    OnTapArPlaneListener() {
13          @Override
14          public void onTapPlane(HitResult hitResult, Plane plane,
    MotionEvent motionEvent) {
15              Anchor anchor = hitResult.createAnchor();
16              AnchorNode anchorNode = new AnchorNode(anchor);
17
18              anchorNode.setParent(baseArFragment.getArSceneView().
    getScene());
19              createModel(anchorNode);
20          }
21      });
22  }
```

Listing 6.2: Creating AR Scene from ARActivity

The AR sceneform (which assists in rendering straightforward 3D scenes without OpenGL) is retreived from the UX XML fragment and the AR Activity fragment. The model is then setup and rendered with an sfb file (the line), and if unrenderable, an exception is thrown. Using the device's camera, when a flat surface is recognised, the object's position is anchored on the device's screen. After, transformable functions are added to the line including qualities such as scale, and position. Finally, the line is then displayed.

```
1   private void setupModel() {
2       ModelRenderable.builder()
3           .setSource(this, R.raw.line)
4           .build().thenAccept(renderable -> navLine1Renderable =
    renderable)
5           .exceptionally(throwable -> {
```

```
                Log.e(TAG, "Unable to load line one.", throwable);
                return null;
            }
        );
}
```

Listing 6.3: Setup Model from ARActivity

```
private void createModel(AnchorNode anchorNode) {
    TransformableNode navLine1 = new TransformableNode(
    baseArFragment.getTransformationSystem());
    navLine1.setParent(anchorNode);
    navLine1.setRenderable(navLine1Renderable);
    navLine1.select();
}
```

Listing 6.4: Creating Model from ARActivity

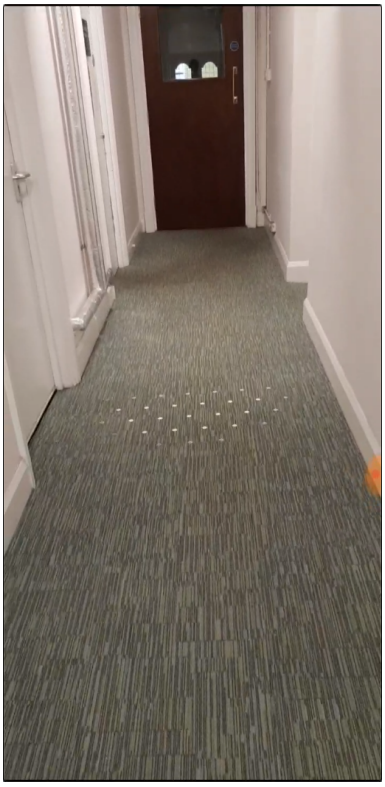


Figure 6.8: Detecting flat surface

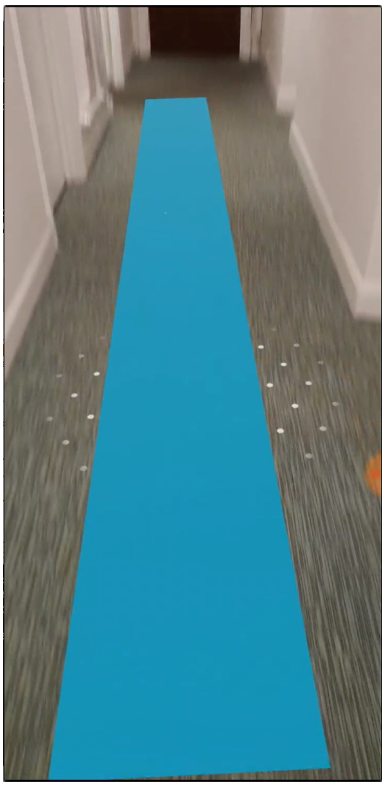


Figure 6.9: Anchoring, and displaying straight line object

6.5 Hardware

In order to transmit the user's location to compute a route solution, utilising the Arduino (UNO R3 Mega 2560)'s small foot-print, a simple beacon plan was sought consisting of the Arduino coupled with the Bluetooth HC- 05 RF transceiver.

Figure 6.10 displays the basic layout of the beacon aside from power input via the board's USB-B socket. This configuration was able to send the strength of the Bluetooth frequency thereby allowing the computation of a distance.

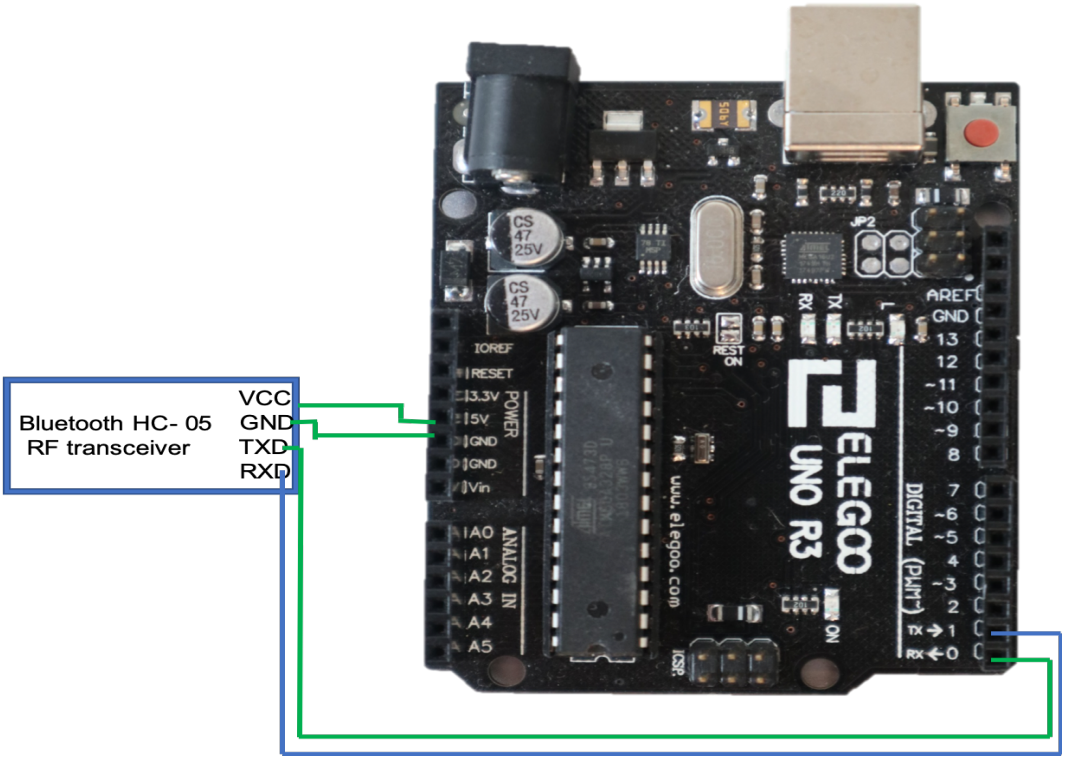


Figure 6.10: Arduino layout

However, to get more accurate readings, the project would need at least three beacons to demonstrate triangulation wherein three signals are emitted from three angles to get an accurate location. During the testing there was only sufficient means for the procurement of one beacon – this led to an inability to fairly showcase the ideal usage, hindering the projects in this method.

While it was advised by the domain expert that this Bluetooth solution would

be the best policy, it was not practicable given the available resources. In view of these drawbacks, the hardware proposal was ceased. The research provided the team with enough of the methods, and models to pursue a theoretical solution.

6.6 Ethical Audit

To comply with GDPR, user locations are recorded during the runtime of the application, and upon termination this data is deleted from memory in order to prevent unauthoried access to the user's location. Users are asked initially to give the Bluetooth, camera, and location permissions to use particular functions.

Since the app uses museum maps, and exhibition details, the intellectual property for those will not be under the jurisdiction of the team. Rather, the project members owns the intellectual properties of all technologies that are being created, and used in conjunction with third-party information. Since assigning data and storing details to specific users was not implemented in this iteration, in future versions the app will need to prevent attacks such as SQL injections.

6.7 Challenges

The groups ideas were difficult to manifest due to a lack of technological resources, and a much larger, but relevant issue; the infancy of the AR, and accurate indoor navigation market-places.

Currently, the market relies on Bluetooth technology in order to bypass the problem. However, for this application, using Bluetooth has disadvantages that can prove crucial success to the project. First, the group would have to have access to multiple Arduinos in order to propagate the necessary number of Bluetooth RSSI signals to get an accurate position on the user. Without

enough Arduinos, the group would be at risk evaluating the signals inaccurately due to the compromising of the signals by metallic surfaces. Another issue with the Bluetooth technology is the risk that it places the user in.

The group initialised the solution by finding a prototype that had already existed [12]. Once found, another challenge arose; the prototype was last maintained in 2016 so the code did not function to begin with. This hurdle was solved by delegating a sprint to the debugging the application.

This halted the software development of the application for a period of time, until the prevailing solution had been devised. A theoretical solution would involve the group hard-coding the geometry of the area to be traversed by the user, and using the coded geospace along with a hard-coded location anchor within that geospace to navigate (A* algorithm with a vector function as the heuristic) the users device.

Finding problems with ARCore library, the group had problems debugging since documentation was vague, and information was scarce. An example of a faced issue was the inability to have the AR section of the application open up on different mobile devices.

A large amount of implementation time was finding usable AR resources, namely the straight line to direct users. Initially, the line file was created, and rendered but did not display on the user's device. To resolve this, a different object file from a GitHub repository was utilised instead. In this case, the line was displayed on the screen, and the group came to resolve that it was an issue to do with the object file, and not sections of the source code.

Further, members of the development team had varying Android versions and SDKs during development, providing inconsistent experiences for each developer. Instead, more pair programming took place so instead of working in an ad-hoc fashion, a collaborative approach was used instead.

Chapter 7

Testing & Quality Assurance

7.1 Testing Conducted

During development various testing rule were created to improve maintainability, and limit bugs. Various testing techniques were explored that were used in SDLC, and TDD (Appendix G). When features were completed by a member of the team, a code review would take place with the scrum master to highlight any defective areas of the source code, improve code quality such as maintainability, and meeting the group's agreed quality assurance practices.

7.1.1 Unit Testing

The unit testing was carried out in house by the team. The aim for this testing was to evaluate the quality of the application, and finding bugs to ensure the product works as expected. 13 unit tests were conducted on the application, and all of them passed. These tests focused on various features that were being implemented, and in some cases being broken down into further tests to maintain integrity.

7.1.2 Integration Testing

Since constructing features were split up into various sprints, to protect from instances where combining features would crash the application, integration testing was used ensuring that on merging features together, the application is running as expected. Two integration tests were conducted each time after sprint 2, and sprint 3 - with both tests passing.

7.1.3 Performance and Stress Testing

Performance testing examines stability, usability, reliability, responsiveness, and speed of the application therefore testing was carried out to assess the performance of the application. Stress testing was carried out to check the upper limits of the application. As application will used in commercial space so the application needed to make sure it does not crash during high volume of users. Since no querying of databases or servers takes place during the application runtime, only stress testing the resources on a user's device during runtime was measured. As expected, it does use a large amount of memory, and CPU power as the camera is in use along with the ARCore package.

7.1.4 Regression Testing

This is the process of testing changes to the application to make sure that the older iteration still works with the new features. Before working on newer versions of application, the old versions are tested against the new version to make sure that all the old features still work with new program. During development, branches are split according to iteration/sprints. By testing and merging two branches, this will show if previous iterations are still compatible with sprint 1 and sprint 2. Two regression tests were conducted and both passed.

7.1.5 User Acceptance Testing (UAT)

The purpose behind UAT is to conform to the system being developed and be ready for operational use according to the specified user requirements. These tests were conducted in focus groups where the team were freely able to ask about various areas that were pointed out in the requirements to be tested with end users and other key stakeholders. By the final sprint, all the user acceptance tests passed measuring whether user actions were used as intended.

7.1.6 Beta Testing

With the application at over 90% completion, field testing was conducted with potential end users, and key stakeholders. Any pertinent issues were identified, and fixed to ensure that the public release version do not contain any fundamental flaws.

7.2 Deployment

After the application is ready, and successfully released, a notification will be sent to stakeholders, and clients. All iterations of the system will be detailed in the changelog.

Notification of deployment procedure (Taken from Appendix F):

1. Check all procedures and ensure everything is done.
2. Email client for meeting.
3. Present the client with the application.
4. Sign off documents.
5. Email client with application information.
6. Release of project and approval of supervisor.

To deliver a solution of quality, the project employs a set of practices of continuous integration and deployment (CI/CD) throughout development. These set of practices (properly pipelined, automated and frequently executed) allows for developers to receive feedback quickly about the quality of the software. To facilitate CI efforts, the project uses GitLab's in house features.

Source Code Control (Management): The project has its own repository in GitLab so that the source code and supporting documents can be tracked, maintained, versioned, and audited.

Build Automation: Build automation is handled using Android Studio. All separate software components of the project are Android projects. For example, the route calculations was considered as a separate project in the repository, and the AR was built in a separate Android project. This allows for a common build interface, and specification of concise instructions for the components testing, building and upload to the project's distribution repository. This is the initial initial step for continuous integration.

Unit Test Automation: Android Studio has JUnit built-in. During development, unit testing becomes much easier to perform as Android Studio allows for this as a feature.

Deployment Automation: Using an online CD/CD platforms such as CircleCI, it is possible to automate the deployment process through the use scripts. Google provides an API to edit a PlayStore listing which will upload the APK and publish it. This can be done directly through HTTP.

7.3 Formative Evaluation

Following TDD throughout implementation allowed for smooth development, and the ability to deliver the outlined requirements. As a result of rigorous testing, the team knew what to develop before developing the feature, hence less time was spent on designing the app. Mosts tests during implementation were resolved in the estimated time frames, though tests relating to sprint three

spent more time than predicted as not enough material was available to the project. However, time spent on these issues could have been reduced if other developers with no tests remaining in the sprint moved over. Nonetheless, the team's flexibility allowed for testing and implementation concurrently; splitting up branches in the repository ensured integration was seamless, whilst keeping relevant tests in their own sprints.

Going back to stakeholders after each sprint allowed for a better understanding of following steps to take, saving time in creating the subsequent sprint plan. Prior to starting the next sprint, any outstanding concerns were addressed and settled with stakeholders. For example, the Bluetooth module was not compatible with some focus group users' device. Regarding this, talks with the domain expert had helped to identify a feasible solution. Issues such as these were resolved before the next sprint. Users also expressed preferences in having an account system to track their activity since it allows for more personalised usage. Stakeholders were consulted, and agreed to include this in further iterations, given the scope of the project.

Feedback from the product owner was largely positive with constant reminders of meeting the set deadlines, allowing for on-target sprints, and effective time-management as a group. Talks with the sponsor who had previous, and relevant experience in the field had introduced the team to 'triangulation', networks used in surveying to determine point locations which led the team to think about implementing this in future iterations of the project where multiple locations are necessary - triangulation allows for path-finding to be updated dynamically, and calculates the shortest path through a sequence of triangles, thus making it ideal for the project.

7.4 Functional Requirements Review

The system successfully navigates a user from one point to another. This was achievable by displaying navigational routes in real-time, and superimposing

a 3D line with AR technology. The shortest route can be calculated by implementing the A* algorithm. However, other requirements have not been reached yet due to discussions made by the team regarding further iterations of the application. For example, although developed in prototyping - the requirement of implementing camera recognition on artworks/exhibits. and displaying further information was decided to be included in the future. Considering the complexity of the application, this was a reasonable decision to make as the MVP agreed upon by the team was to deliver the foundational service of the system Lastly, the requirement of having the system suggest recommendations based on the user's route was decided to not be included in the first release of the application; suggesting recommendations is a feature that can only be implemented with a database. Due to the lack of resources available at hand, having an account database would be too costly as it would require the use of a server. Stakeholder feedback encouraged the introduction of an account system, therefore implementation of an account system is possible in further development of the application.

7.5 Non-Functional Requirements Review

As with the functional requirements, implementing a TDD approach greatly helped the team adhere to the requirements, ensuring non-functional requirements are met. The performance of the system should not be slow - after a new feature was added to the system, developers would then test-run the app to make sure that the new feature did not affect the performance. Similarly usability was tested - going back to the stakeholders to gain feedback on the simplicity of the navigation of the system. In terms of the data usage; the only real data being sent out by the device is the user location, and Internet connectivity. It was imperative the system requested the user for location permissions, and the application is also able to function on Wi-Fi so this requirement was satisfied. For safety, development has not reached a level of complexity where the user would need to concerned about hitting an immutable object in their

path - further developments will house this concern. Having avoided the use of Bluetooth for route calculations, the concern of security in the user's location being accessed by third-party users was invalidated.

Chapter 8

Project Evaluation

8.1 Summative Evaluation

This project aimed to tackle indoor navigation in commercial settings specifically museums. Augmented reality was selected in order to enhance this, providing greater a user experience. Stakeholder gathering was valuable since it gave insight to problems the group did not foresee, and laid out the foundational requirements of the system. The execution of the Agile methodology was good, since two members of the team had worked together in projects using Agile in industry - leading to a greater understanding by everyone of how it works.

Although most stakeholders had Android devices, more extensive research could have taken place on iOS since the team faced various problems regarding the Android operating system. Situations like Bluetooth signals being received by Apple instead of Android devices, and in contrast to Android, Apple offers thorough AR-related documentation which was realised during later stages.

Strong planning across the project was the key to success as all of the team contributed to tasks, allowing for leeway if issues arose. When team members were indisposed for example, re-allocating responsibilities were managed, and con-

trolled the risk of overusing resources. Using Trello ensured responsibility was held to account, and centralised project management. Especially during the design stage of the project, this ensured that ideas were communicated across clearly to stakeholders. Since many Lo-fi prototypes were created, finding positive attributes to form the final design was straightforward as stakeholders were not limited in choice.

Initially, finding one similar prototype for navigation assured Bluetooth was not the technology to use in calculating routes. The project sponsor informed that metal surfaces interferes with RSSI [13], this lead to using path finding algorithms (in the context of graphs) instead as AR better supports it. During the first two sprints the scrum master was involved in the sprints themselves, and that lead to work not meeting requirements. This was rectified in the remaining sprints, where timely interventions at impediments took place by them so that momentum was conserved, and the outlined requirements were achieved.

The decision by the team to control the scope of the MVP ensured that requirements were met within the timeframe. Sections of the app written in Kotlin were converted to Java in the final sprint since Android Studio better supports it. Further, more members were confident with Java, allowing for team members to rigorously test features, and finding resolutions. Controlling the scope allowed the demonstration of originality in this project, and provided discipline to adhere to the MVP; reducing risk in resources enabled for correctly implemented features in the app.

If given more time, further research could have been conducted into various technologies for calculating routes in the context of AR. At the start of the project lifecycle, academic research in the field has been relatively limited given its infancy, but has recently accelerated with large technology companies rolling out beta versions of outdoor AR navigation. Hence during the research stage, a greater understanding of the sector dynamics would benefit outlining requirements, and features to build for the MVP.

8.2 Future Developments

By design, this project is geared towards to meeting the requirement of the museum sector. However, the main objective navigation, being so widely applicable, with adequate research there are many prudent areas of development. With the saturation of navigation software, this project should strive to stick to the indoor avenue.

In a commercial context the most important future development would be retail navigation. As the product is based on finding the shortest route to a point, taking the project to this stage would not require any fundamental change. As in this concept, a retail version would involve finding the best solution in terms of pure navigation, and also proposing certain shops in keeping with the user's preferences should be made clear that based on the current model which involves studying the layout of a candidate area, a need to find a quicker method of digitising a space is desirable.

Areas covered in the project backlog with the remaining features left to implement to create a fully working product would be an area of future development. Building a database to house previous user visits can further enhance the user experience. A more streamlined method of implementing individual floor plans would provide quicker lead times to potential clients in developing prototypes for their buildings, though this would be further down the line.

In addition, there is a plethora of other environments to explore. For instance, the findings of this report can be easily implemented into public libraries wherein you have a catalogue system for which the project could apply a node to which to navigate to. Furthermore, as most library users will look for a general subject or even a very specific book criteria, the efficiency of this project would be very well employed in bringing about unambiguous results and pathway. Other areas can also be exploited where there is a sense of urgency to reach a destination like airports or train stations whereby there is usually an abundance of space only supported by signposts to guide the way. The implementation of the project could have a positive impact in helping

users quickly finding a pathway.

Appendix A

User & Stakeholder Research

Quotes from field research

Statement 1

Visitor I - "Doesn't change the experience for me, maps are helpful already"
Visitor H - "Big tech companies haven't produced products like that"
Staff L - "If successful, it will raise the overall visitor count"

Statement 2

Visitor G - "Exhibits with QR codes allows me to look at more details for a particular work of art"
Visitor D - "Want device to recognise what exhibit I am looking at"
Staff L - "Takes away the need for human interaction"

Statement 3

Visitor B - "Takes away immersive experience and distracting"
Visitor G - "Don't like having mobile phone out in a museum"
Visitor C - "If app was free of charge, more people would use that instead of

paying for maps - particularly those in younger generations"
Staff N - "Simple to use detract experience, make their current jobs harder and target market"

Statement 4

Visitor E - "Will allow to me plan journey more efficient"
Visitor I - "Great to navigate museum with a mobile phone, don't need to ask strangers for help"
Visitor F - "Didn't have much information available on hand"
Staff M - "Too many questions and about the app foreseen, retracts experience for museum-goers"

Ratings

Focus groups answered statements as set out by the group with guidance from the market research proposals. Responses will be a range of agreement from 10, being strongly agree to 1, strongly disagree. Total is multiplied by 2 to ensure equal weighting.

Statement	Museum Visitors								Museum Staff							
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	Total
1. Current Museum Navigation aids are inadequate	6	9	6	3	6	7	8	10	4	10	8	9	7	6	10	73
2. I would like to see more interactivity with exhibits	3	10	8	7	5	9	10	10	7	5	7	2	7	6	8	69
3. I would be happy to use my own device to aid experience	2	5	8	6	9	4	3	7	6	9	10	6	8	3	8	65
4. Often have to ask/answer navigation questions	9	5	9	8	9	7	6	9	8	7	10	10	8	10	9	83

Table A.1: Responses from focus groups on agreement with statements
Interviewees A-J are visitors, K-O are museum staff members

Domain Expert meetings

Snippets from meetings with the project's domain expert

GPS

- GPS indoor not very reliable

Bluetooth

- Bluetooth beacon
  - estimate
  - proximity beacon
  - not to the absolute distance
- most cost effective
- through raspberry pi
- individual desk level/25m range

Optical Solutions

- Rear/forward facing for marker like QR code
- All stored locally on device
- Lo-fi solutions

Other things

- Indoor challenge not completely met currently (e.g. by Google)
- Infra-red approach, differency frequency & delta

  - Point A to point B, not cost effective

- Estimate using accelerometer between markers
- 2D top-down map for displaying optimal route?
- Finite number of routes - do natively
  - Calculate peak times in certain spaces
- Work with environment & orientation
  - Know you fix beacons

QA

For UX

- A/B testing
- In field testing
- Azure wireframing prototypes

Development

- 2 weeks of tuning location
- Monitor data from users to find optimal performance

Appendix B

User Stories

Use Case Model

Two scenarios have been taken into account, where the user gets lost in the museum, and the user wants to explore the museum. When a user is lost, they need to enter their destination where the app will receive their current location, and find the quickest route from the user's current position. The user follows that navigation until they arrive at their destination. For the exploration, the app will show the details where user know what they going to see in the museum.

Activity Model

This is based on the back-end of the application for example when the user searches about the museum, this history saved in the server where if the user wants to go to the same place then they can use our function called past visit.

User & Acceptance Stories

This will describe what will be achieved once the application is ready to be used by the user. A diagram has been created based on different scenarios where it can be found if the application has achieved the user needs.

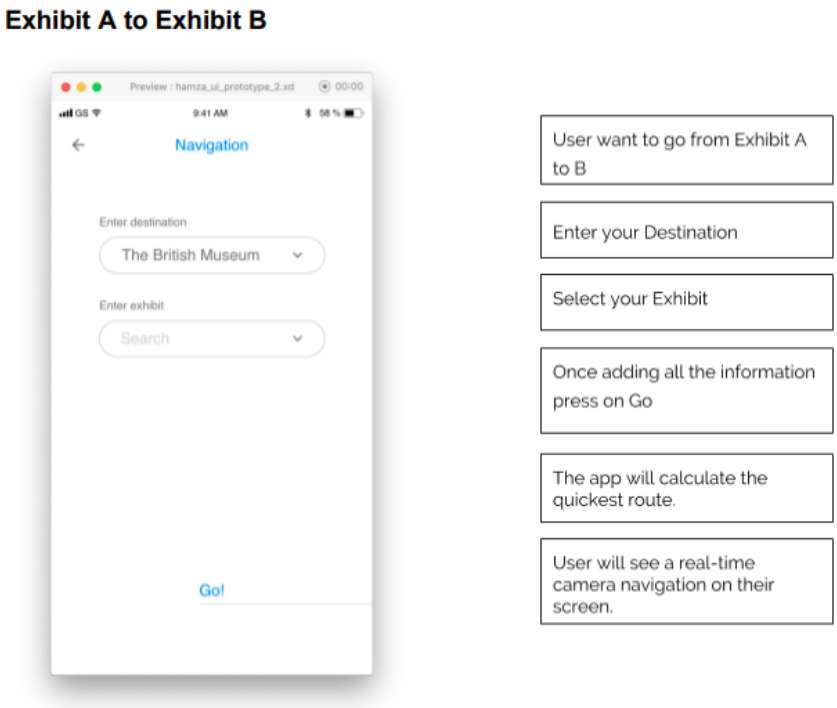


Figure B.1: Going from point A to point B

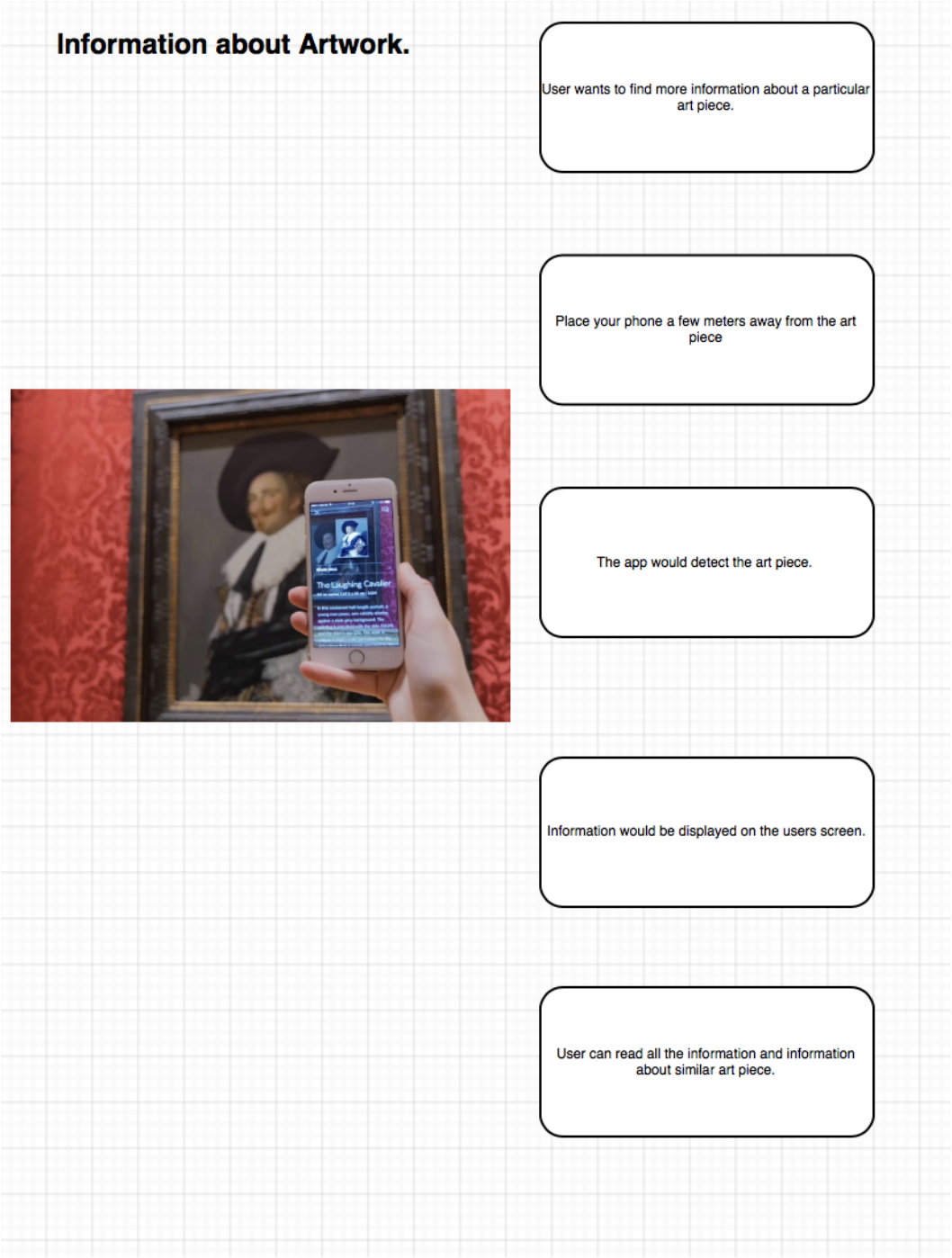


Figure B.2: Getting information from exhibition

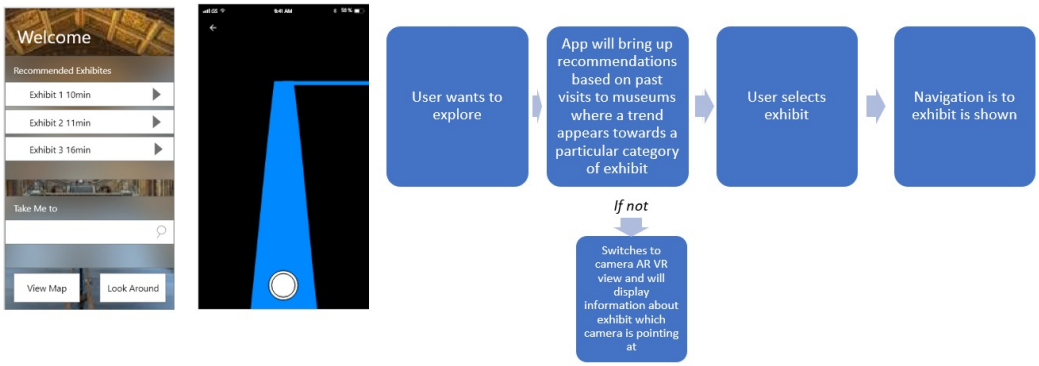


Figure B.3: Exploring the museum

Appendix C

Systems Requirements Specification

Purpose

The main goal of this concept is to provide users a solution to indoor museum navigation through an exciting, and enjoyable experience using augmented reality. It includes users being lost, or searching for a specific location within the museum. It was discovered through field research that the concept would make life easier for users and the museums since it would allow easy access to the information based on exhibitions.

Scope

This project will include creating an AR application for people to get an enjoyable journey in the museum. The project will be completed by 29 April 2019. The AR application will include simple navigation system to direct various part of the museum. Getting information on the user screen using the user's camera, and explore various museum using the system. The platform will be developed on Android due to the high usage of Google's AR library,

ARCore.

System Overview

The application will perform all the basic tasks to help users with their journey in the museum. Such as navigating from point A to B, getting the user back on track in case they are lost, allowing the user to view information based on camera recognition of an exhibit.

References

This specification should be read in conjunction with the following publications:

IEEE 24765-2017 - ISO/IEC/IEEE International Standard - Systems and software engineering–Vocabulary [14]

IEEE Std 29148-2011, ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes –Requirements engineering [15]

IEEE Std 730-2014, IEEE Standard for Software Quality Assurance Processes [16]

IEEE Std 24748-4-2016 - ISO/IEC/IEEE International Standard for Systems and Software Engineering – Life Cycle Management – Part 4: Systems Engineering Planning [17]

Definitions

Activity: A set of cohesive tasks of a process, which transforms inputs into outputs. [ISO/IEC/IEEE 12207:2008]

Augmented reality: A technology that superimposes a computer-generated image on a user's view of the real world, thus providing a composite view.

Functional requirement: A requirement that specifies a function that a system or system component must perform. [ISO/IEC/IEEE 24765:2010]

Non-functional requirement: The measurable criterion that identifies a quality attribute of a function or how well a functional requirement must be accomplished. A non-functional requirement is always an attribute of a functional requirement. [ISO/IEC/IEEE 730:2014]

Performance: Degree to which a system or component accomplishes its designated functions within given constraints, such as speed, accuracy, or memory usage. [ISO/IEC/IEEE 24765:2017]

Stakeholder: Individual or organization having a right, share, claim or interest in a system or in its possession of characteristics that meet their need and expectations. [ISO/IEC/IEEE 15288:2015]

Usability: Extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. [ISO/IEC/IEEE 25064:2013]

User: Individual or group that interacts with a system or benefits from a system during its utilisation. [ISO/IEC 25010:2011]

Use Cases

The use cases have been defined as follows:

1. Use Case Model
2. Activity Model
3. User & Acceptance Stories
    (a) In Exhibit going from A to B

(b) Getting information from an exhibition

(c) Exploring the museum

(d) User get lost in the museum

Use Case Model

Two scenarios have been taken into account, where the user gets lost in the museum, and the user wants to explore the museum.

When a user is lost, they need to enter their destination where the app will receive their current location, and find the quickest route from the user's current position. The user follows that navigation until they arrive at their destination. For the exploration, the app will show the details where user know what they going to see in the museum.

Reference to use case model

Activity Model

This is based on the back-end of the application for example when the user searches about the museum, this history saved in the server where if the user wants to go to the same place then they can use our function called past visit.

Reference to activity model

User & Acceptance Stories

This will describe what will be achieved once the application is ready to be used by the user. A diagram has been created based on different scenarios where it can be found if the application has achieved the user needs.

Reference to the 3 user stories

Functional Requirements

1. Needs to be able to navigate the user from point A to B.
2. The system should be able to display navigational routes in real-time.
3. It should be able to calculate the quickest route.
4. A 3D line should be superimposed through augmented reality to display the navigation to the user's destination.
5. Camera recognition on artwork/exhibits, displaying further information about the exhibit.
6. When user arrives at destination, the system should give a recommendation based on their route.

Non-functional Requirements

1. **Performance**: The system should respond quickly to user input, e.g user wants to find more information about an art piece or whenever they search up a location. The system should not require extensive CPU usage, it should not slow the device down inconveniencing the user.
2. **Usability**: The system should have a simple layout, with appropriate colour used in appropriate contexts. The language used on the app should be easy to understand for the users. Having done research based on the user preference, it was discovered that users dislike too much text on their menu screen.
3. **Data Usage**: Data usage should be kept to a minimum, only querying the relevant information (user location and exhibit information). Also, the app would require internet connection in order to calculate the real-time distance of the final destination from the user's current location.
4. **Safety**: The system needs to have the ability to detect immutable objects obstructing the user's path. This can reduce common user accidents

when using a mobile phone whilst walking.

5. **Security**: Ensuring that the device's current location cannot be obtained by unauthorised third-party users is crucial in ensuing the security of using the platform.

Appendix D

Documentation Plan

Introduction

This documentation plan outlines the strategy for creating all documentation associated with the software release. This document is addressed to project team, and supervisors to inform them about the documentation efforts that is undertaken for the release.

Scope

The documentation plan includes the development of updates to all users and developers that are required for the software release. Specifically, it covers the creation and updating:

- User guides
- Product website
- Release notes

Scope of the development activity providing updates to the above documents. These activities requires the involvement of user testing.

Assumptions

It is assumed that readers of the document are familiar with the previous stages to the project and the associated strategies in place. It is also assumed that the required resources will be available to achieve the objectives of the plan, and that there are no risks other than those identified in the section on Risks.

Constraints

Constraints on this documentation project are the available time from the resources as outlined at the start of the project along with the product delivery schedule. Changes or delays in product delivery will affect the documentation plan.

Existing Documentation

This document should be read in conjunction with:

- Proposal
- Testing plan
- Gantt chart
- Application release notes

Documentation Specifications

Platforms

All documentation is accessible on all platforms via a PDF, and on all browser-compliant platforms.

Distribution & Delivery

PDFs of all documentation will be available on the product's website.

Terminology

Terminology will be maintained throughout the documentation as of the proposal document.

Process & Schedule

Activities

The following activities will be undertaken to produce the documentation:

- Creating indexes for user guides.
- Merger of all application notes from previous releases into guides.
- Documenting source code and approaches.
- Creating testing documentation.
- Creating release notes.
- Creating read me files for each component.

Milestones

Given the diversity of activities, and information streams, estimated milestones are based on the current availability of required resources:

Change Control

Change control for documentation is similar to changes in the source code:

Milestone	**Delivery Date**
Implementation Ends	4th March 2019
Updated files to reviewers	6th March 2019
Initial review complete	29th March 2019
Revisions complete	22nd April 2019
Review Complete	25th April 2019
Program and Report Release	29th April 2019

- During documentation development, changes and error corrections are communicated directly with the appropriate author.
- After the end of the implementation phase, changes or corrections are communicated in the same way as above, but the author is responsible for prioritizing the requested fixes to determine which ones should be made in the remaining time before release.
- Major documentation changes shall be treated the same as bug releases, and will be handled in conjunction with the next applicable major release.

Risks

The risks identified have a potential to affect the delivery schedule:

- Due to the volume of changes and enhancement to the product throughout the development process, so long as the scope has been correctly identified, this document can be time appropriate to all of the development activities.
- If there are changes to the scope, the depth of coverage of the documentation may be amended, or the target date extended.
- Delays in turnaround of reviews prevent on-time delivery. To reduce this risk, authors of the document will have as much advance notices as possible of the requirement for a review.

Issues

None found at the time of publication.

Appendix E

Testing Plan

Introduction

Purpose

This testing plan in accordance to the IEEE Standard for Software Test Documentation (ANSI/IEEE Standard 829-1983), outlines and describes the testing approach and overall framework that will drive the testing of the implementation phase of the project.

The document outlines:

- Testing Strategy: structure and descriptions of testing.
- Execution Strategy: describes how the test will be performed and processed to analyse defects to the platform, and resolutions to the defects.
- Test Management: processing how to deal with testing platforms and events that take place during execution.

Audience

- Project members will conduct tasks specified in this document, and provide working updates to it. All members will be accountable for the

results.

- The project lead plans the testing activities in the overall project schedule, and tracks the performances of the test.
- The project supervisor will ensure that the plan is met by the team and provide further test cases if necessary to important functionalities.

References

This document should be read in conjunction with:

- Proposal
- GitLab repository testing plan
- Gantt chart

Objectives and Tasks

Test Objectives

The objective of testing is to verify the functionality of the platform is in accordance to the outlines of the proposal. Test execute and verify test scripts, identifies and fixes various levels of defects.

Assumptions

General:

- During the execution of testing, the current project plan acts as a precondition.
- Software delivered by from the development side must be in accordance with the development plans so it is functionally usable and in testable units.

- The quality of the development tests are to be performed in the agreed manner and thoroughness.
- Testers for each sprints should be available in accordance with the test schedule.
- Defects will be tracked through GitLab. Any defect fixes planned will be shared with Test Team prior to applying the fixes on the Test environment.
- There is no environment downtime during test due to outages or defect fixes.
- The system will be treated as a black box; if the information shows correctly on device, and in the reports, it will be assumed the program is functioning.

UAT:

- UAT test execution will be performed by end users and the team will create the UAT script.

Testing Strategy

There are three key aspects to this version of the platform. For both route calculations and augmented reality implementation, unit and integration testing will be pivotal in ensuring these functions are implemented rigorously. UI testing will mainly focus on unit testing but user acceptance testing will be heavily featured to match the requirements of the stakeholders.

Unit Testing

The unit testing will be carried out in-house by the team. The application will be tested when development has reached 70% or above. The goals of this

testing will be to evaluate the quality of the application, finding any bugs, ensuring the product works and is ready to be tested by the users.

Entry Criteria

- 70% – 90% complete and assurance to go ahead with unit testing
- Unit test cases should be designed and reviewed
- Testing environment set up and and stability confirmed

Exit Criteria

- All test cycles should be complete
- All the unit tests should be executed and Passed
- Alpha version of the application frozen (i.e., no additional features, no modifications to existing features, no dropping of the existing features)
- Unit test formal Sign-Off

System and Integration Testing

This type of testing will verify the behaviour of the integrated hardware and software environment of the complete system. Helping to evaluate the system's compliance with its specified requirements.

Integration will be tested using incremental testing, taking on the 'top down' approach. First testing each module of the application individually and then continue testing by considering other modules in addition. As the nature of the application is an Augmented Reality navigation app, testing the Bluetooth functionality prior to testing the AR functionality is one example of how integration testing would occur. This would be followed by testing of both these modules combined.

Performance and Stress Testing

Performance testing examines responsiveness, stability, scalability, reliability, speed and resource usage of the software and infrastructure. As development of the application will be done under the agile methodology, continuous testing will be carried out to assess the performance of the application.

Stress testing will be carried out to check the upper limits of the application, testing it under extreme loads. As the software developed will be an application used for navigational purposes in a commercial space, an example definition of a test case would be with a very high number of users, which is known as a 'Spike Test'.

User Acceptance Testing (UAT)

UAT will be performed at the very end of development, prior to the product going live. This testing will be carried out by real users who will decide whether or not the acceptance requirements provided by the team are to be accepted or rejected. One example of an acceptance requirement would be "the route calculation must be accurate". This is an optimal phase for also identifying any bugs.

Acceptance criteria will be gathered by the team with real users comparing the system to the initial requirements. During testing, the team will **Assist In UAT**, who will be on stand-by to help users in the case of any difficulty. However, the main reason for the team to be on stand-by is to record results and log any bugs etc.

Beta Testing

Beta testing is the final stage of the testing phase, where the application will be released to an external test group consisting of real users. The main entry criteria being that the development should be 90% - 95% completed, all

components either fully or almost complete for testing. At this point in testing, the unit testing should be signed off. Any bugs identified will be handled promptly and feedback analysed to ensure application satisfies the user. Test cases written in this phase will be clearly outlined, defining which feature is being examined, such as UI, Bluetooth recognition, location handling, route calculations etc.

Validation and Defect Management

- It is the responsibility of testers to open defects and link them to the corresponding code, and assign an initial severity and status, before retesting and closing the defect. It is the responsibility of the project lead to ensure the defects are fixed in a timely manner and according to the project and testing plan.

Severity categories (from softwaretestinghelp.com):

1. Critical - The bug is critical enough to crash the system, or cause potential data loss. It causes an abnormal return to the operating system. Or it causes the application the hang and requires a re-boot of the system.
2. High - It causes a lack of vital program functionality with workaround
3. Medium - This Bug will degrade the quality of the System. However there is an intelligent workaround for achieving the desired functionality. Or this bug prevents other areas of the product from being tested. However other areas can be independently tested.
4. Low - There is an insufficient or unclear error message, which has minimum impact on product use.
5. Cosmetic - There is an insufficient or unclear error message that has no impact on the product use.

Testing metrics allows the measurements and level of success of test that will be developed with the project lead.

- Test preparation and execution status - To report on % complete [daily/weekly]
- Daily execution - To report on pass, fail, total defects, critical defects [daily]
- Project weekly status - Project driven reporting (as requested by supervisor) [weekly]

Hardware Requirements

- Raspberry Pi with Bluetooth beacon
- Mobile device with Bluetooth beacon
- Android device with Android 5.0 (Lollipop) or higher

Test Schedule

Tests will be executed during each sprints as outlined in the Gantt chart, and a final testing stage will be completed for the testing milestone.

Control Procedures

Problem Reporting

If there are defective software found during testing, the project lead will assign the defect to the team, and fix it before it is sent back to testing. Project lead will require approval to ensure the updated software matches the requirements of the test.

Change Requests

If modifications to the software are made, the project lead is required to sign off the changes and review the changes to the current platform. If there are changes that will affect the existing platform, then these particular modules will have to be identified.

Features to Be Tested

To be read in conjunction to the backlog:

- Receiving user inputs for user destination
- Route calculations
- Superimposing 3D directional line
- Display navigation

Features Not to Be Tested

These features will appear in later iterations. This is due to the short implementation time available for the project.

- Finding museums nearby
- Mobile device camera recognition
- Receiving and displaying information about the exhibit
- Rating and dealing with user reviews of platform

Risks/Assumptions

Tools

All tests will be mainly conducted on the Android Studio testing suite. JUnit will specifically conduct unit testing, and GitLab will have continuous integration and continuous delivery in order to ensure integration to the current platform is successful.

All testing artifacts such as the test cases themselves are stored on the GitLab repository.

All tests should be tested on devices higher than Android 5.0 (Lollipop) that have allowed Bluetooth to be used.

Appendix F

Deployment Plan

Introduction

Purpose

The purpose of the deployment plan is to ensure that the system successfully reaches its users and new features to the system are delivered successfully. The aim of the deployment plan is to provide a detailed schedule of events, persons responsible, and dependencies required to integrate the new version of the app with the previous version. It should minimize the impact of the integration of the new system on the users and stakeholders.

Assumptions

The application would have a place for users to:

- Have a login activity, where the user can enter credentials to login, or continue as a guest.
- To login to the application, enter their location, and destination.
- Route calculation takes place (finding the shortest route to the destination).

- The user can retrieve their current location, and map.
- The user can take a quick snap of a exhibit and the application provides more information about it.

Dependencies

Dependencies that can hinder or slow the process of deployment are:

- Dependent on using an Android device
- Dependent on Google's ARCore Software Development Kit (SDK)
- Dependent on Google's Map services
- Unavoidable change of plans
  - Change of user requirements
  - Research fails
  - Implementation fails
- Time management
  - Group meeting
  - Supervisor meeting
  - Weekly delegated tasks
  - Milestones

Constraints

- Reliance with Google's map services (server can crash)
- Repository could be down
- Group member availability:
  - Not all members are mutually available

  - Conflicting time schedules

- Internet Connection/Wi-Fi issues
- Database error: Existing accounts may not be able to login if the DB server is down

Assessment of Deployment Readiness

1. Verify that the application does not have any broken links and that all content is accessible.
2. Verify that all dependent files have been uploaded to the relevant directories so that they can be accessed from other calls.
3. Supervisor approval

Product Content

Configuration would include the following:

- Accuracy and reliability of separation of programming plans by members.
- How easy to download all the documentation, report or code from gitlab.

Deviations and Waivers

Deviations from the original plan included:

- Implemented Route Calculations using Wi-Fi instead of Bluetooth
- Changed our 3D Model from a directional arrow to a navigational line
- Implemented outside commercial spaces instead of within a museum setting - to deliver applicable concept

Phase Rollout

Phase I

- Map showing user's current location
- Route calculations
- Superimposed 3D directional line
- Display navigation

Phase II

- Show nearest museums to the user's current location
- Camera recognition of exhibits
- Request and pull information about exhibit
- Display the information

Phase III

- Account database
- Registered users can store their visited museums
- User can rate and review the visited museums

Notification of Deployment

After the application is successfully released, a notification will be sent to stakeholders and clients. All iterations of the system will be detailed in the changelog.

Steps

1. Check all procedures and ensure everything is done.
2. Email client for meeting.

3. Present the client with the application.
4. Sign off development plan documents.
5. Email client with application information.
6. Release of project and approval of supervisor.

Deployment Systems

Continuous Integration and Continuous Delivery (CI/CD) will be used to deliver any changes to the system. Automated tests are written to for each new feature to ensure that less bugs are passed to the production stage and captured early by regression, reducing the risks at every release. Gitlab have built-in tools to support CI/CD, which provides a simplified setup and execution of software development using continuous methodology.

DevOps

Much like Agile development, adopting a DevOps culture allows for smoother processes within development.

- **Building the right product**: After code has been written, the team will be able to receive faster feedback as a result of live testing.
- **Improved productivity**: As delivery would be continuous, the developers and testers will be additionally efficient as testing environments are easier to set up (see A/B Testing below).
- **Reliable releases**: Smaller and more frequent releases would allow for less changes made to the code as well as the bugs.

A/B Testing

The main idea behind A/B Testing is that testing can be performed on variants based on live environments. Since development followed the agile methodology as well conforming to a test-driven development approach - testing was performed in conjunction with the coding. After new code has been written and when testing is required, the environment would already be set up for current and future tests.

Appendix G

Testing

Type	Description	Usage	Expected Outcome	Actual Outcome	Status
Unit	1. Allow bluetooth usage from user device	Normal	User permission given	App ask for user permission for bluetooth	Pass
Unit	2. User denies bluetooth permissions	Erroneous	Stop app and send user an alert encouraging bluetooth use	The app will stop working	Pass
Unit	3. User does not follow route	Erroneous	Route is re-calculated according to their current location	Code not written	Pass
Unit	4. Receiving current location as destination of desire from user input	Erroneous	User goes back one screen	User is taken one screen back	Pass
Unit	5. User gets to their desired destination	Normal	Alert conveying that the user has arrived at their destination	User is taken one screen back	Pass
Unit	6. Allow location usage from user device	Normal	User permission given	User permission for location requested	Pass
Unit	7. Get sensor information from user device	Normal	Receive acceleromter and orientation information	Utilise sensor information for route calculation	Pass
Unit	8. User declines sensor information being sent	Erroneous	Application closes	Application terminates	Pass
Unit	9. User allowing camera permissions	Normal	User permission given	The system does give permission to the user	Pass
Unit	10. AR Scene is viewable	Normal	The screen displays AR scene on user's device.	AR fragment and Scene is created successfully	Pass
Unit	11. Navigation activity camera functionality	Normal	Navigation activity loads device camera	Device camera is loaded upon navigation activity	Pass
Unit	12. 3D model is created and viewable on the user's device.	Normal	3D model is rended on the device.	Straight line is shown	Pass
Unit	13. Installation of ARCore	Normal	Upon navigation activity, redirected to ARCore on the Google Play Store	User is redirected and navigation activity works	Pass
Unit	14. Installation of ARCore	Erroneous	User does not install ARCore: Redirected to menu.	Navigation activity closed and user redirected to menu	Pass
Unit	15. User not allowing camera permissions for the app	Erroneous	Alert displayed to force user to allow camera	A message pops up on the users device "the app require camera permission"	Pass

Table G.2: Unit Tests

Type	Description	Usage	Expected Outcome	Actual Outcome	Status
Integration	1. AR & Bluetooth integration	Normal	Bluetooth development recognised by AR	No collisions	Pass
Integration	2. UI & functionality (AR & Bluetooth) integration	Normal	User input works the functionality behind application	No collisions	Pass

Table G.3: Integration Tests

Type	Description	Usage	Expected Outcome	Actual Outcome	Status
Performance & Stress	1. Time taken to load application and features	Normal	Quick responses from system	Quick responses from system, but large amount of device resources used	Pass

Table G.4: Performance and Stress Tests

Type	Description	Usage	Expected Outcome	Actual Outcome	Status
Regression	1. Checking if previous iterations are still compatible with v.0.0.1	Normal	New integration does not affect previous version	No affect to previous versions	Pass
Regression	2. Checking if previous iterations are still compatible with v.0.0.1 and v.0.0.2	Normal	New integration does not affect previous version	No affect to previous versions	Pass

Table G.5: Regression Tests

Type	Description	Usage	Expected Outcome	Actual Outcome	Status
UAT	1. User logging in as a guest	Normal	Access to application services with Guest login	The system allows a Guest user to login	Pass
UAT	2. Select destination	Normal	AR navigation show ups	AR Sceneform is generated, and camera is turned on	Pass
UAT	3. Select destination that does not exist but still registered on the app	Erroneous	Display destination error message	Error message displayed	Pass

Table G.6: User Acceptance Tests

Type	Description	Usage	Expected Outcome	Actual Outcome	Status
Beta	Field testing	Normal	No unexpected or unnormal behaviours during runtime with users	Application runtime ran as expected	Pass

Table G.7: Beta Tests

Appendix H

User & Stakeholder Feedback

Comments about the application after each sprint

After each sprint took place, key stakeholders were consulted through face-to-face meetings. Below are snippets from some meetings that were used as feedback for the following sprints.

Sprint 1

"Bluetooth is not compatible with all devices"

"iphone picked up signal better androids"

"Not all androids picked up the Bluetooth signal"

Sprint 2

"able to pick up location on all devices"

"UI smooth, bigger font sizes"

"Not able to go anywhere"

Sprint 3

"needs to be able to run ARCore on new phones"

"asked for camera permissions"

"can't see any AR yet"

Sprint 4

"fully integrated with route calculations"

"dead links on info page"

"navigates users from a to b"

Bibliography

[1] Microsoft. (2017). Path guide: A new approach to indoor navigation, [Online]. Available: https://www.microsoft.com/en-us/research/blog/path-guide-new-approach-indoor-navigation/.

[2] A. Murphy. (2017). Retail in museums: Continuing the visitor experience through the museum shop, [Online]. Available: https://advisor.museumsandheritage.com/features/retail-museums-continuing-visitor-experience-museum-shop/.

[3] Mapspeople, 2018. [Online]. Available: http://www.mapspeople.com/.

[4] Q. Mourcou, A. Fleury, C. Franco, F. Klopcic, and N. Vuillerme. (2015). Performance evaluation of smartphone inertial sensors measurement for range of motion. sensors.

[5] AXELOS, Manging Successful Projects with PRINCE2. TSO, 2014.

[6] Smartsheet. (2018). What's the difference? agile vs scrum vs waterfall vs kanban, [Online]. Available: https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban.

[7] Globalluxsoft. (Oct. 2017). 5 popular software development models with their pros and cons, [Online]. Available: https://medium.com/globalluxsoft/5-popular-software-development-models-with-their-pros-and-cons-12a486b569dc.

[8] K. Schwaber and J. Sutherland, "The scrum guide - the definitive guide to scrum: The rules of the game", p. 7, Nov. 2017. [Online]. Available:

https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf.

[9] APD, Scrum summary sheet, 2017.

[10] A. Bulajic, S. Sambasivam, and R. Stojic, "Overview of the test driven development", Proceedings of Informing Science and IT Education Conference 2012, pp. 165–187, Jun. 2012. [Online]. Available: http://proceedings.informingscience.org/InSITE2012/InSITE12p165-187Bulajic0052.pdf.

[11] K. D. Science and Engineering. (2018). Software engineering fundamentals - best practices, [Online]. Available: https://blog.k2datascience.com/software-engineering-fundamentals-best-practices-b5105d155c6d.

[12] S. Khasbag, S. Ranade, P. Hiralikar, and A. Wani. (2016). Smart museum design, [Online]. Available: https://github.com/shubhankar30/Smart-Museum-Design.

[13] Apple. (2019). Resolve wi-fi and bluetooth issues caused by wireless interference, [Online]. Available: https://support.apple.com/en-gb/HT201542.

[14] IEEE, "International standard - systems and software engineering -vocabulary", ISO/IEC/IEEE Std 24765-2017, 2017. DOI: 10.1109/IEEESTD.2017.8016712.

[15] ——, "International standard - systems and software engineering – life cycle processes –requirements engineering", ISO/IEC/IEEE Std 29148-2018, 2018. DOI: 10.1109/IEEESTD.2018.8559686.

[16] ——, "Standard for software quality assurance processes", IEEE Std 730-2014, 1998. DOI: 10.1109/IEEESTD.2014.6835311.

[17] ——, "International standard for systems and software engineering – life cycle management – part 4: Systems engineering planning", IEEE Std 24748-4-2016, 2016. DOI: 10.1109/IEEESTD.2016.7470727.