FACULTY OF SCIENCE,
ENGINEERING AND COMPUTING

School of Computing and Information Systems

BSc DEGREE

IN

Computer Science

PROJECT DISSERTATION

Name: Alec McEwan

ID Number: K1810131

Project Title: COVID Track and Trace System

System Project Type: Build

Date: 5th April 2021

Supervisor: Prof. Graeme Jones

Kingston University London

Declaration

I have read and understood the University regulations on plagiarism, and I understand the meaning of the word *plagiarism*. I declare that this report is entirely my own work. Any other sources are duly acknowledged and referenced according to the requirements of the School of Computing and Information Systems. All verbatim citations are indicated by double quotations marks ("…"). Neither in part nor in its entirety have I made use of another student's work and pretended that it is my own. I have not asked anybody to contribute to the project in the form of code, text or drawings. I did not allow and will not allow anyone to copy my work with the intention of presenting it as their own work.

Date: 05/04/2021

Signature: Alec McEwan

Table of Contents

1. Introduction

1.1 Problem Statement

As of March 2021, the world is currently experiencing a pandemic in the form of a virus called the Coronavirus. This has been going on for over a year and the world is finally seeing the light at the end of the tunnel. All countries have tried different methods to slow down the spread of the virus, some successful and some not so much. In the United Kingdom multiple lockdowns have taken place to try and help the cause. As well as this, the government decided to implement a system to track members of the publics interactions. They hoped to track infected members of the public and asking them to self-isolate, this would hopefully result in the decline in number of infected people. The idea was for users to scan barcodes when they enter shops and businesses. However, research has shown that members of the public chose not to scan the QR codes independently. This is due to the system not being compulsory, and so very little people used the application. The engagement rate of people using the system was low and so the rate of infection did not decrease. This is the problem we are going to attempt to fix during this project.

1.2 Proposed Solution

The solution we are proposing is not completely different to the governments approach. The proposed system would reverse the roles of the old system. This time the members of the public all get unique QR codes and business employees would scan QR codes as members of the public enter. For example, an employee could scan the QR codes as they enter the store or as they are checking out/leaving. Of course, it would be most efficient to scan everyone's QR code as they enter the business. However, if someone scans QR codes when someone pays at a shop, there will still be a significant increase of QR scans. The aim of the proposed solution is to try and increase the engagement of people using the system.

1.3 Development Approach

The current government created system is a mobile phone application. The new proposed system will be a web application and the chosen languages are HTML,CSS and PHP. These have been chosen based on prior experience using the languages. The new system will still easily be accessible by mobile phones while being a web application. This will be crucial to allow members of the public to access their QR codes.

After comparing different development approaches. The system will take the MVC approach (Model View Control). This methodology segments the system files into three sections. This will contribute to a faster development process as the tasks can easily be split up, making them more manageable. This approach also helps us debug the program faster and more efficiently.

1.4 Project Management Approach

There are multiple project management methodologies that can be followed when building an application which include methods such as Waterfall and Kanban. However, this project will follow an Agile methodology. The iterative approach to the project will benefit greatly. As well as the flexibility of being able to alter sprints on short notice. The small sprints and constant feedback all help with the constant development.

2. Literature Review

2.1 Introduction

When browsing the web you receive a responsive and seamless experience. However, behind the scenes there are hundreds of little pieces that work together to deliver you, the end user, an enjoyable experience. These technologies all glue together to produce one complete product. For example, you will never see what's inside a company's backend database, however this piece of web technology gets used constantly.

2.2 Web Browsers

Web browsers are platforms that display requested information to an end user. You could think of a web browser as a compiler. It will request data from a web server, usually in the form of written code, compile it and display the result to an end user. As well as receiving files of code, a web browser will also receive the relevant assets to be displayed on a web page. For example, the asset could be a photo that the publisher wishes to display on the page.

There are many different web browsers that offer a range of features to suit different types of users. Currently the most popular web browser is Google Chrome. As of January 2021 Google Chrome has a usage share of "63.63%" (StatCounter Global Stats - Browser, OS, Search Engine including Mobile Usage Share, 2021). Google chrome offers a wide range of features. For example, there is a store of extensions you can add to your browser, some free and some paid. This allows you to further customize your experience when browsing the web. One very popular extension is an add blocker, this essentially blocks any advertisements that may popup on websites making it easier to view the content you requested without getting distracted.

The second most popular web browser is Safari. This is a proprietary piece of software that was created by Apple. You can only access this web browser if you own an Apple Computer. Because of this, it only has a usage share of "19.37%" (StatCounter Global Stats - Browser, OS, Search Engine including Mobile Usage Share, 2021). If this browser were not only accessible to certain people, the usage share could be way higher as they are limiting the number of people that have access to is. Safari offers a plethora of features that are unique to this web browser. For example, "Picture in Picture" allows the user to easily multitask, like watching a video while browsing the web. Another interesting feature is the ability to preview a URL without fully visiting the website. With this you can check if the website you are going to is the correct one.

2.3 Databases

Most websites will use a database in the backend to store important information. This could be account details as an example. Databases can be created relationally, meaning that each set of data will have its own table linking to another. Separating the data is useful when trying to search for specific information. If a database has hundreds of thousands of entries all cluttered together, it would be more efficient to split up the data so when you try and fetch an entry you don't have to search through everything trying to find it.

Different databases will have different features to suit specific websites. "For example, one database management system (DBMS) may have built in encryption" (Comparing Database Management Systems: MySQL, PostgreSQL, MSSQL Server, MongoDB, Elasticsearch and others, 2021). This may be tailored to a government website such as the NHS that contains lots of sensitive information. MySQL is a very common database platform that is predominantly used for small consumer-based websites. As of “January 2020 55.6%” of developers worldwide had used MySQL (Most popular database technologies for developers 2020, Statista, 2021). It offered free instillation and had simple syntax designed for average developers to be able to use. This database solution is widely supported by many cloud-based platforms such as Amazon Web Services (AWS) and Microsoft Azure.

Another common database platform is MongoDB. It is a free, open-source, non-relational DBMS. As mongoDB does not use the SQL language to access the database, it is fast and offers easy data operation. “The non-relational database aspect means that it will use up more processing power because all entries are in one big table but will also mean that it is faster to access data. The negative side to this type of database is that the administration is more complicated” (Relational vs non-relational databases: advantages and disadvantages - Clockwise Software, 2021).

2.4 Frameworks

Frameworks are software tools that build and run web apps such as web services, web servers and web APIs. These are extremely useful to programmers as they don’t have to spend time and errors searching for computational errors and bugs. A lot of frameworks and models have libraries that can implement templates for programmers to use. These libraries can also contain easy database access and testing frameworks, all useful for web developing.

There are two different types of frameworks. The first is server-side (backend) frameworks. This can include simple web pages, landing pages and different types of databases. The second framework is client-side (frontend). The is the opposite to the backend and will have no connection with business logic. It only works inside the web browser and includes the user interfaces.

“ExpressJS is an example of a server-side framework that works well with the Node.js language. It is easy to maintain and focuses on high speed. It is used widely by companies like IBM, Uber and Accenture” (Mistry, 2021).

React is an example of a client-side framework that contains a lot of JavaScript frontend libraries. This feature makes it highly desirable by web developers. React is an example of a framework that uses the MVC (Model Control View) approach. This approach to development was established and maintained by Facebook and Instagram.

2.5 Languages

Every piece of web technology will have some connection to one or more programming languages. Some technologies will only require one programming language, for example, most databases will use SQL as the query language. Whereas a basic web page will use HTML and CSS.

HTML is used in “92.2% of websites” (Usage Statistics and Market Share of HTML5 for Websites, April 2021, 2021). Its popularity is justified as it is very simple to learn and use. It is supported by all browsers and is very lightweight. This language is ever evolving, its most recent iteration is HTML5. Its aim is to keep up with new web technologies and include support for them. Most of the time HTML is accompanied by CSS. CSS is a styling language. Its main usage is to design the frontend page. Without it, websites would simply be vertical text on a blank screen.

HTML and CSS are front end languages. PHP is an example of a backend language, but it can also be implemented into frontend pages. PHP is an open-source language that “is often used on data-heavy websites or for app development” (Common Web Design Languages, What They Do and Why You Need Them, 2021). PHP will be implemented so as the processing will occur before the HTML is displayed. PHP also works very efficiently with database implementation as well as protection, its built in PDO allows seamless database access.

The alternative language to HTML is a templating language. An example of this is HAML. HAML can be simpler than HTML in some ways, however it will just get converted into HTML at the point of compiling.

2.6 Conclusion

In conclusion, each and every piece of web technology is hugely important for the maintaining and developing of web applications. These technologies are still evolving and will only get better as time goes on. After completing the literature review, a stronger understanding of each technology has been achieved. This is a much stronger foundation to now decide what technologies will be implemented into the track and trace system. After comparing different languages, HTML and PHP will be used to develop the web application. As well as choosing an SQL database over a MongoDB database. This was due to the easy integration of SQL databases with PHP backends.

3. Requirements Chapter

3.1 Introduction

Requirements are the functionality that a client would like to be present in the system. Requirements are split up into functional and non-functional requirements. Functional requirements are a description of what the system should do. Non-functional requirements are how the system should behave. There will also be discussions about the internal and external stakeholders of our project. This will give us a better understanding of what a client might want out of the track and trace system.

3.2 S.W.O.T Analysis

We are undertaking a SWOT analysis to try and identify the purpose of this project and whether it is worth doing. We will be identifying the strengths, weaknesses, opportunities and threats to our proposed track and trace system.

3.2.1 Strengths

In the grand scheme of things, we have a massive advantage over our competitors. Due to the government already making their own track and trace system, being able to reconstruct something and alter the negative aspects of their outcome is a big advantage. We will also be able to catch anyone that is meant to be self-isolating. If someone who is self-isolating goes to a shop, they will get their QR code scanned when they enter, this will cause a big red flag as we know that they have been in contact with someone positive with COVID-19.

We will also know what works well in the public’s opinion. This will mean getting feedback from the public about the current track and trace system is a necessity. Once we gather feedback from the general public about what they liked and disliked about the current system, we can then design our track and trace system solely based on positive feedback.

3.2.2 Weaknesses

A massive weakness we have over our competitors is our budget. The government is always going to have a bigger budget than a Student for a track and trace system. With the bigger budget the competitors can pay for different technologies that we may not be able to afford, as well as being able to hire multiple employees. Therefore, the timespan that we take to create the product is another weakness that we will have to overcome.

3.2.3 Opportunities

We are taking advantage of this opportunity because we are aware of the lack of competition in the area. The only competitor we have are the government that have already made their own track and trace system. We believe that our system will be a considerable step up from the current track and trace system that is in place currently.

3.2.4 Threats

If the government changes any regulations on track and trace then we might be at threat, but currently we are all good. There are also no emerging competitors in this area.

If the country goes back into lockdown and no one can go to restaurants, then we could be at risk as we will not be able to test the final system in a real-world environment. This is crucial for finishing the final system and then deploying to the public.

3.2.5 Project Recommendation

After identifying our strengths and weaknesses in our SWOT analysis. We have determined that the strengths that we hold outweigh the threats from our competition. Our main task after completing our SWOT analysis if to gather feedback about the current track and trace system from the member of the public. After this is completed, we will have a strong understanding of what would need to be changed to make the system more user friendly in the publics opinion. The opportunities to progress and upgrade the current track and trace system are massive. Therefore, we have decided to go ahead and carry out the project.

3.2.6 Mitigating the Threats and Weaknesses

One big weakness that we have against our competitors is our budget for the system. Our competition has a massive budget whereas we are using the resources provided to us by Kingston University London. Unless we identify a stakeholder that is willing to invest a lot of money into the system, we will be using what we have. All this means is that there is scope to upgrade the track and trace system in the future. Because we will not be using the fastest servers to host our system, we can easily upgrade this in the future once we have a bigger budget.

3.3 Stakeholder Identification and Analysis

A stakeholder is someone who can affect or is affected by a business or project. Stakeholders can be internal or external, meaning that they could be a part of a business or outside the business. For example, in our scenario the public that will be using our system are an external stakeholder. On the other hand, the owners/managers of shops and restaurants that will adopt our system will be internal stakeholders because they are a part of a business. This stakeholder analysis is undertaken to give a better understanding of the desired features and requirements of the system. It will also identify and risks and problems that may occur during the development. Each stakeholder will have their own dashboard that they will be able to navigate through.

3.3.1 General User

One stakeholder we can identify straight away would be the members of the public. They will be directly impacted by the track and trace system as they will be the daily users of the system. The members of the public will need the system to be simple to use with very few steps required to sign up. There are plenty of members of the public that may not know how to use the technology involved and so will not be able to register and use the system.

3.3.2 Business Employee

Another stakeholder would be employees of shops and restaurants that will be using the system to scan customers QR codes. They will be impacted highly as they may need compulsory training to teach them how to use the system. Just like the members of the public, I am sure that there are many employees that will not know how to use the system; therefore, some training may be required.

3.3.3 Business Owner

The owners of businesses like shops and restaurants are our next stakeholder. They will be affected as they will need to adopt and embrace the system if they want their shop or restaurant to stay open. It would be a requirement of the shop or restaurant to use the track and trace system if they want to stay open. As a manager/owner of a business, it is their objective to make as much profit as possible. Therefore, shop owners will want the system to be fast and responsive to get customers in and out of their shop or restaurant fast. They do not want people wasting time trying to scan QR codes.

3.3.4 Trace Operative

The penultimate stake holders are the trace operatives that work on completing the COVID cases. Their role will be crucial in tracing the members that have been in contact with COVID positive users. Therefore, they will require lots of functionality to complete their job as quickly and efficiently as possible.

3.3.5 Trace Manager

The last stakeholder is the trace manager. This role is simple as they are just managing the trace operatives and assigning them with the relevant COVID cases.

3.3.6 Power/Interest Analysis

In figure C.1 bellow, there is a power/interest grid showing the different user types/stakeholders of the system. From the figure you can see that the General Public has the most interest and influence over the system. This is because they want the pandemic to be over and would be interested in any effective way of going back to normal life. They also influence the project the most as the public will only use the system if it meets their requirements and contains the functionality they want.

A trace operative has low interest in the project; however, they will be the ones completing the COVID cases and so will have a strong influence over the functionality. The better the functionality for the trace operatives, the easier it will be for them to trace users and complete cases.

Business employees will have high interest in the project as they will be adopting the system as part of their job. They have low influence as there is limited functionality that could benefit them. The employees will only be scanning QR codes and wont need lots of features.

The role of the trace manager is to create and delete trace operative accounts, as well as assigning COVID cases. They have the least influence and interest because their role is fairly simple.

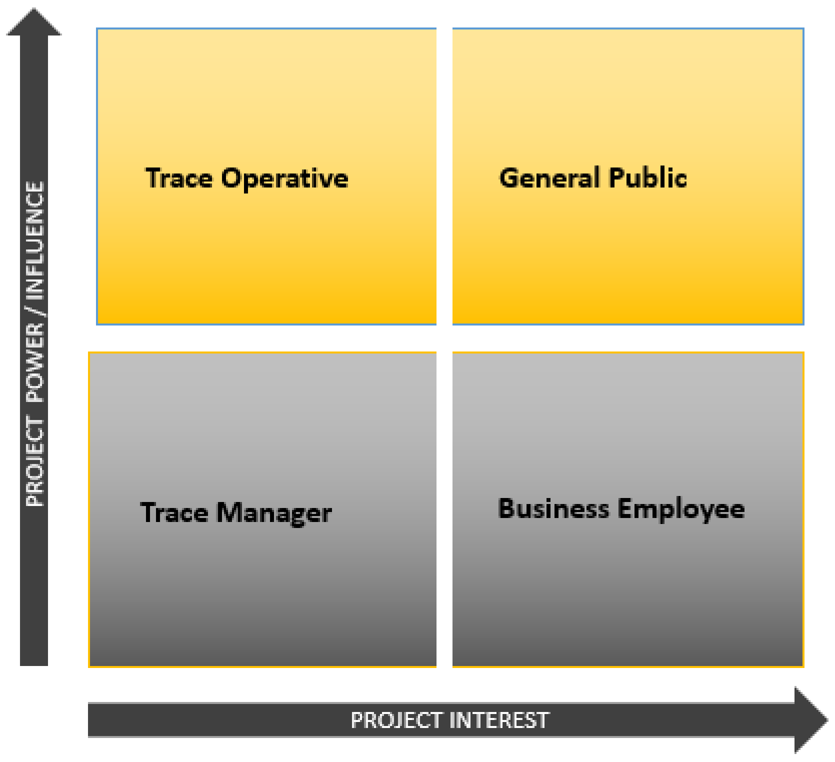


Figure C.1 Use Case Diagram

3.4 Use Case Analysis

User stories are used all the time in Agile Software Development. They are a technique used to extract software features from an end-user point of view. Firstly, you must identify each user type. Then interviews with each user type can take place. During these interviews, questions can be asked about what features that user type may like to see in the system. From the interviews, requirements can be established. These are crucial for a designer to fully understand what the stakeholder want in the system and then attempt to implement it.

Next the use cases and use case diagram can be created. The use case diagram identifies the individual and shared tasks of each user. All this analysis is undertaken to help understand the full system requirements. The use case texts are a more detailed look at each task. They include a step by step on how the user may undertake the task. From the use case diagram, the individual and shared tasks can be correctly designed to make the system streamlined and efficient.

3.4.1 User Stories

There are overall four different types of users. Firstly, we have the average normal user. This would be the members of the public that go into shops. The second user type is the employee point of view, the person scanning QR codes. The third user type is the tracer, this is the person contacting people that have been in contact with someone positive with COVID-19. The final user type is a Trace Manager, they create trace operative accounts as well as assign COVID-19 cases to the operatives. The user stories will give us a better understanding how the project can help the user. After that it is then the developer's job to develop the system to contain as many requirements as possible.

Normal User Story

“As an everyday user, I would like to browse shops without wasting time. Specifically, I would like for my QR code to be the first thing I see when I login to the track and trace system. I also need to be able to report symptoms quickly and efficiently. After submitting my symptoms, if my results say I should self-isolate, I would like to order my ”at home test” with fast shipping so I am not self-isolating for longer than I need to be. Finally, if I have been in contact with someone who has tested positive, I need to be notified immediately so as I don’t leave the house.”

Employee User Story

“As an employee to a local business, I need the system to be fast so I can get customers in and out of the shop. I also do not want to be waiting for my customers to get out their QR codes so it should also be fast on their end. It would be nice if I got some sort of audial or visual confirmation after scanning each QR code, that way I will know when I am done scanning. I would like the system to work on a range of different devices, If I leave my phone at home, I still want to be able to scan customers QR codes.”

Trace Operative User Story

“As a trace operative, I need to be able to view a list of all confirmed cases of COVID-19. I will need to know their location and the date and time. I would also like to be able to query the database quickly so I can identify anyone that was at the same location. I also need to ability to send emails to groups of people to tell them too self-isolate.”

Trace Manager User Story

“As a trace manager, I am responsible for overlooking the progress of the trace operatives. I will need to be able to create multiple operative accounts whenever we get a new employee. I will also need to delete accounts of past employees. I will be overlooking the progress of the COVID-19 cases, so I would like to know if a case is complete, as well as assign specific cases to different operatives.”

3.4.2 Use Cases

Use Case Diagram

Use case diagrams are created to help identify the tasks that are shared and the tasks that are unique to each user type. The box is used to represent the system, inside the system you can see circles containing the possible actions that the user could take. Figure C.2 below is the use case diagram for each user type. You can observe the shared and unique tasks for each user.

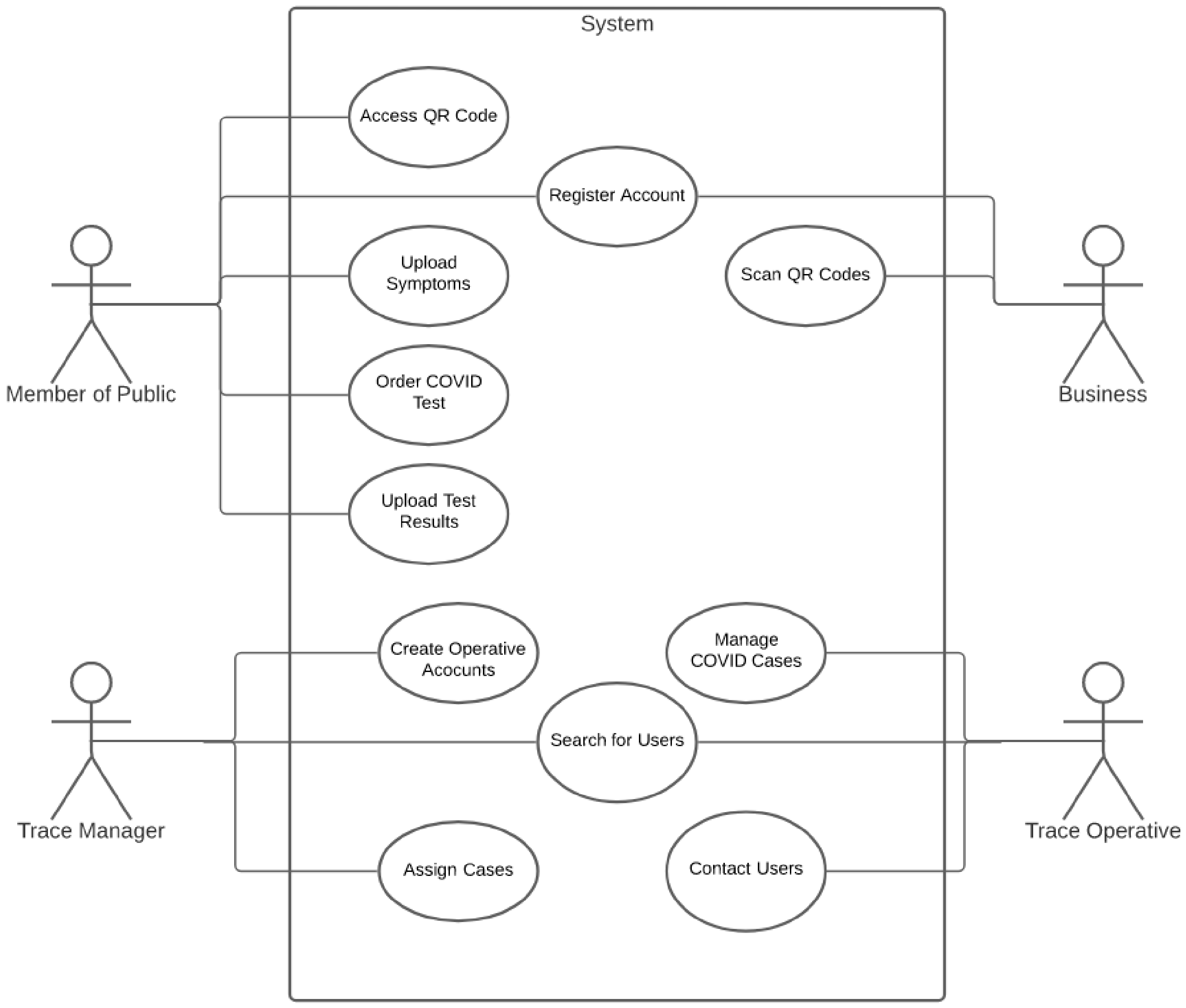


Figure C.2 Use Case Diagram

Use Case Text

Use case Text are an expansion of the Use Case Diagram. Below is an example of one of the use cases. The actor row refers to the type of User that may experience this use case. In the example the actor is a Shop Employee as they will be the only user type that scans QR codes. The description row helps describe the whole activity. To setup the task a few “pre-conditions” must be met before the use case can be carried out. These conditions could also be expanded out. Finally, the flow of events is a step-by-step guide on how to complete the use case.

Use Case 1

Below is an example of one of the use case texts. The user type that would be completing this task would be a shop employee. This example shows the steps needed for a shop employee to scan a user’s QR code. As you can see, this task requires for the user to have already logged in before completing the task. The flow of events shows the necessary buttons that need to be clicked on to complete the task.

ID	**01**
Name	Scan Customer
Actor	Shop Employee
Description	Employee scans customers QR code
Pre-Condition	Employee is not logged in Customer has QR code ready
Flow of Events	1. Click on “Log In” 2. Enter details in login 3. Click on “Start Scanning” 4. Hold camera over customers QR code 5. Get confirmation of scanning
Post-Condition	N/A

Table C.1 Use Case

A comprehensive set of use case texts are listed in Appendix A.

3.5 List of Requirements

The user stories and use case diagram have helped to identify the key features for the system. Below, those features have been broken down into functional and non-functional requirements. There is also a description of the requirement as well as a link to the user type that it corresponds to.

Functional requirements refer to a physical action that the system has to perform. Non-functional requirements refer to how the system should complete a task. “Non-functional requirements do not affect the basic functionality of the system.” (Functional vs Non-Functional Requirements: The Definitive Guide - QRA Corp, 2021)

3.5.1 Functional Requirements

ID	**Requirement**	**Description**	**User Type**
1	Create General User and Business Accounts	A General User and a Business User should be able to create separate accounts, the system should record personal info during this time	General User/Business
2	Create/Delete Trace Operative Accounts	A trace manager should be able to create and delete trace operative accounts	Trace Manager
3	Log In	All users should be able to log into the system and access their respected areas	All Users

4	Log Out	A User should be able to log out of their account once they are done	All Users
5	Generate Barcode	A user should be generated a QR code whenever they signup, they should also be able to view it from their home screen	General User
6	Scan Barcode	A user should be able to get their barcode scanned by a shop employee	Business
7	Report Symptoms	The user should be able to report their current symptoms	General User
8	Order COVID Test	Users that show symptoms should be able to order a test kit	General User
9	Upload Test Results	A user should be able to upload their COVID test results	General User
10	Assign COVID Cases	A trace manager should be able to assign positive COVID cases to specific trace operatives	Trace Manager
11	View/Manage Assigned Cases	A trace operative should be able to view and manage all of their cases	Trace Operative
12	Trace Users	A trace operative should be able search for specific users	Trace Operative
13	Add Case Members	A trace operative should be able to add members to a COVID case	Trace Operative
14	Contact Users by Email	A trace operative should be able to contact users by email	Trace Operative
15	Contact Users by Phone Number	A trace operative should be able to contact users by texting their phone number	Trace Operative
16	Update case status	A trace operative should be able to update the status of a case to complete	Trace Operative
17	Case Reports	A trace manager should be able to create case reports	Trace Manager

Table C.2 Functional Requirements

3.5.2 Non-Functional Requirements

- The system should be responsive and not take longer than 2 seconds of delay
- The database should be protected from SQL injection attacks
- The system should log out users after a period of inactivity
- The interface should be simple for any user to understand
- The system should lock users out after multiple failed attempts to log in

3.6 Discussion

The whole requirements chapter had an end goal of identifying comprehensive list of functional and non-functional requirements. All the sections in the requirements chapter helped contribute to the design chapter.

The chapter started off with the S.W.O.T analysis. The aim of this section is to identify if it is worth undertaking the project. This was achieved by identifying our strengths, weaknesses, opportunities, and threats of the project. All this information helps us determine if it is worth carrying out the project in the first place. For the project to go ahead we would want to have more strengths than weaknesses in our S.W.O.T analysis, in our case this is true. By identifying our opportunities, we now hope to action upon the opportunities to give us the best chance. After identifying our threats, we can now take action to avoid the threats.

Next was the stakeholder identification and analysis. In this section the internal and external stakeholders were identified. With our stakeholders, a power/interest grid was completed to identify which user types should be prioritised. This section had a strong influence on the next section which was the User stories.

The next section was our Use case Analysis. This was the extraction of use cases based on the different user types.

The first subsection of the analysis were the user stories. We undertook User stories to find out what sort of features the different users may want as a part of the system. This was done by interviewing each user type. From the interviews, desired features could be extracted. These features were very low level however they were useful to know for the Use Cases.

A Use case diagram was created based on the interviews with the different users. Different tasks were identified as either individual or shared. For example, registering for an account is a shared task between a general user and a business employee.

The next subsection was the use case texts. More detail was put into each user activity. This included the flow of events that the user must undertake to complete the task. This detail helps the designers and developers. Now a more detailed database can be designed and developed now that they know the different user types and tasks that must be completed.

The last section was simply extracting the comprehensive list of functional and non-functional requirements from the user stories and use cases. More detail was put into the descriptions of each requirement, as well as linking each functional requirement to a user type.

4. Design Chapter

4.1 Introduction

This section of the report will be split up into two main sections. The first is the database design, this is where all of our data will be stored including personal data and data about QR code scans. It will also include some reasoning as to why the database is designed the way it is. The second section will be the interface design. This will mostly consist of the GUI and what the user will see and interact with. It will tackle the design of each web page and include wireframes/prototypes of each page.

4.2 Database Design

This part of the design chapter will be based around how the database is created to ensure we remain the integrity of our database and ensure data redundancy it achieved. The activity diagrams created demonstrate the flow of data from the start of a task to the end. It also includes the decisions paths that users could take. The entity relationship diagram gives a visual representation of the relationships between the tables in the database. Finally, the data dictionary contains all the meta data about our entities in the database.

4.2.1 Activity Diagram

Login

The activity diagram in Figure 4.1 bellow shows the flow of data as the user attempts to login to their account. We can observe the decisions the user has to take to complete the action. For example, if the user cannot remember their password, they will have to follow the steps to get their password reset, after this they can then attempt to login again.

A comprehensive set of Activity Diagrams are listed in Appendix B

Uploading Test Results

Figure B.1.1 in Appendix B shows the sequence of events as a user attempts to upload their COVID-19 test results to the system. We can see in the figure that the user is taken to another page if they have not received a test. This is because this activity cannot be completed if they have not already ordered a test, and to do this they must first show symptoms of having Coronavirus.

User Search

Figure B.1.2 in Appendix B shows how a trace operative can search for users. This can be done by inputting the search criteria that they want. This could be any form of data, for example they could input a post code and the system would return a list of users registered with that post code. The same could be done with the users' name or/and address.

Contact Users

Figure B.1.3 in Appendix B shows how a trace operative would contact a user. This can be done by inputting the users "UserID". The "UserID" could have been acquired by searching for users that are part of a case, as seen in the previous activity diagram.

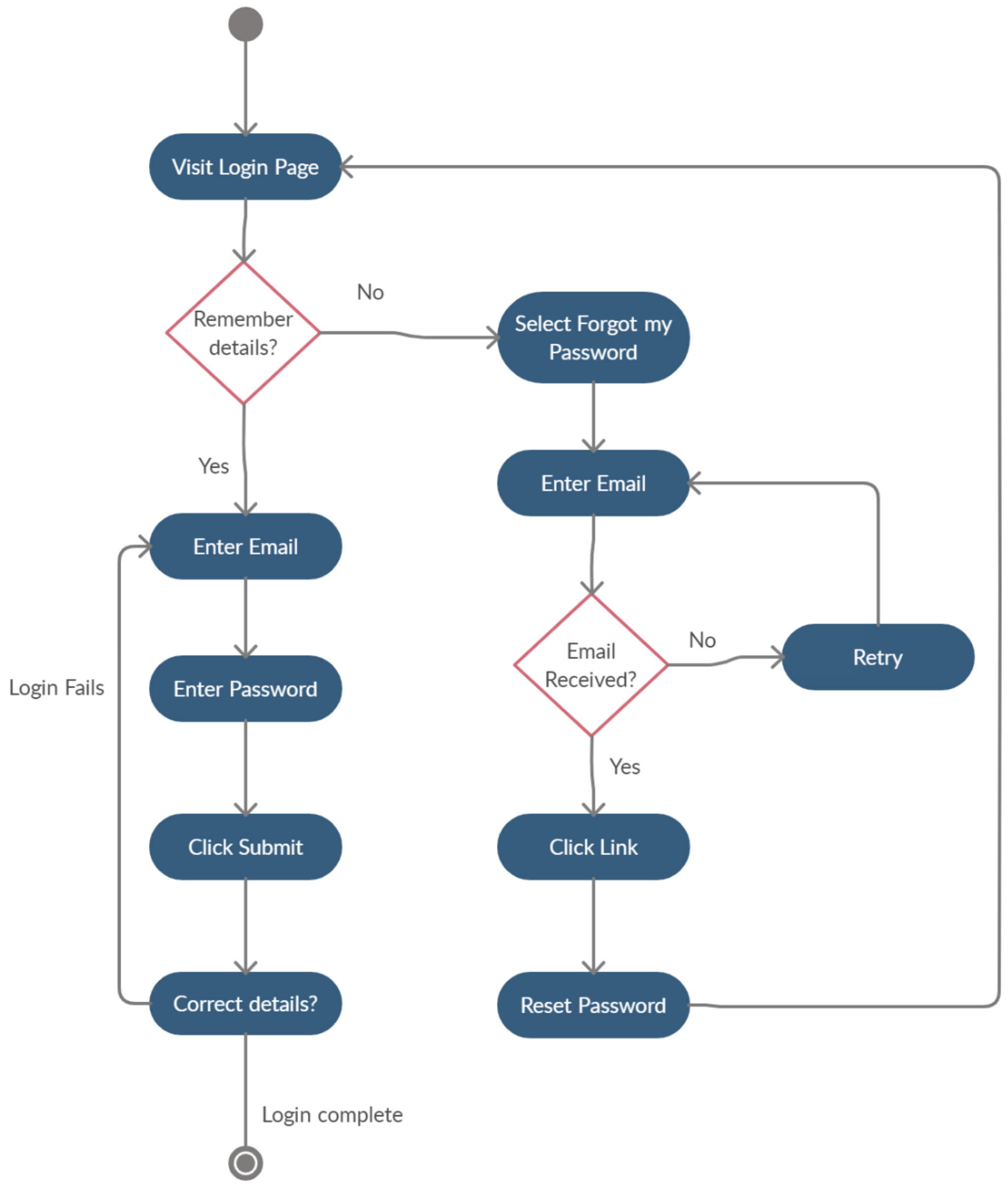


Figure 4.1 Activity Diagram - Login

4.2.2 Entity Relationship Diagram

The entity relationship diagram in Figure 4.2 bellow shows the relationships between the individual tables in the database. You can also observe the entities that will be stored in the corresponding table.

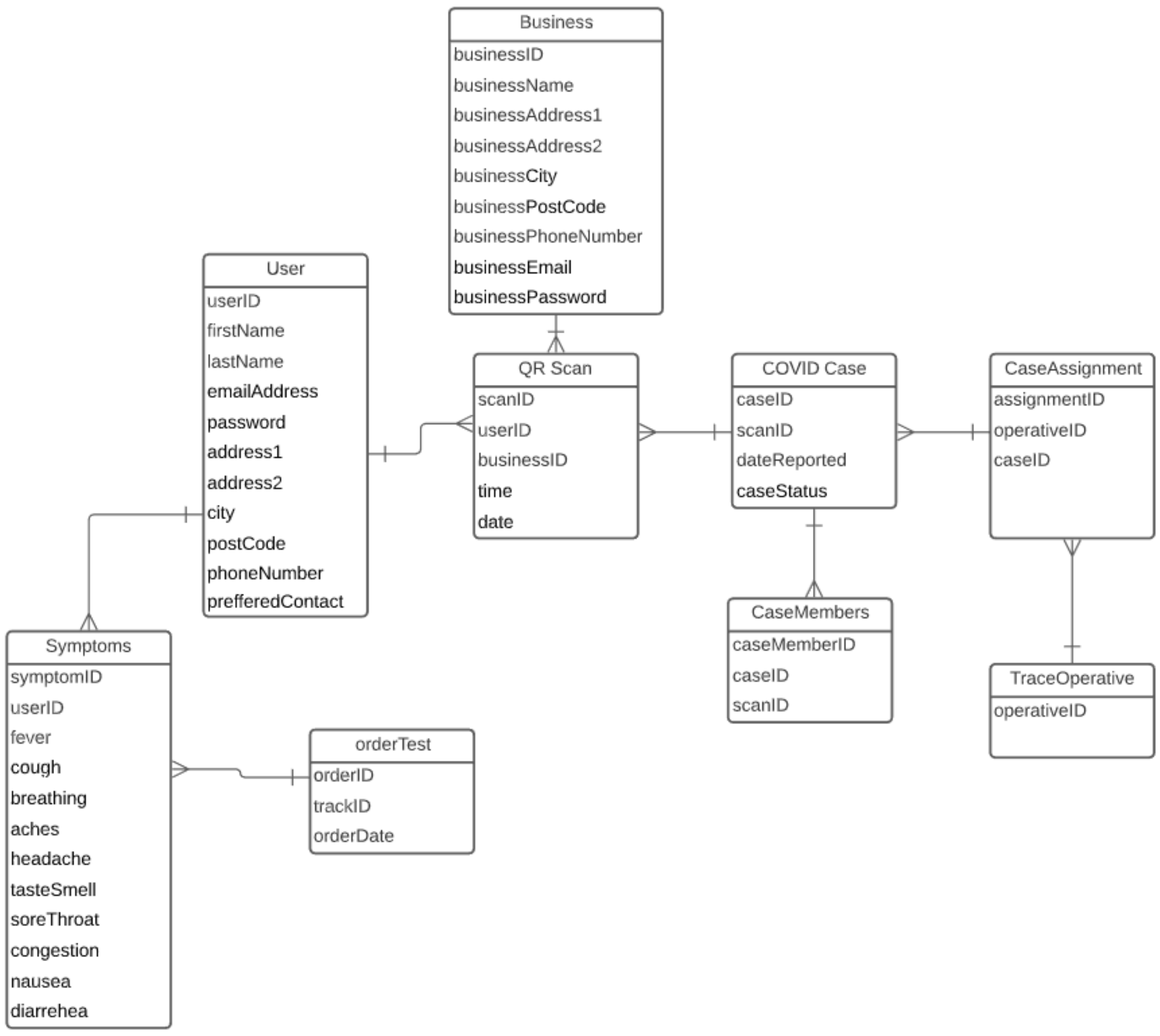


Figure 4.2 Entity Relationship Diagram

4.2.3 Data Dictionary

A data dictionary is useful as it gives all the data about each field in a database table. If someone wanted to recreate our database, the data dictionary would help them reconstruct each table to match the original table identically. For example, in figure 4.3 bellow we can see that in the “userID” row, it has a comment saying that it is the primary key of the table.

A comprehensive set of Data Dictionary tables are listed in Appendix C

Column	Type	Null	Default	Comments
userID	Int(10)	No	-	Primary Key
firstName	Varchar(50)	No	-	
lastName	Varchar(50)	No	-	

dateOfBirth	date	No	-	
emailAddress	Varchar(150)	No	-	
password	Varchar(50)	No	-	
address1	Varchar(200)	No	-	
address2	Varchar(200)	Yes	NULL	
city	Varchar(50)	No	-	
postcode	Varchar(8)	No	-	
phoneNumber	int(11)	No	-	
preferredContact	Varchar(10)	Yes	"Email"	

Figure 4.3 Data Dictionary – User Table

4.3 Interface Design

4.3.1 Wireframes

Login Page

This login page is very basic, but it also does not need to be complicated. The user should easily be able to input their email and password and click submit to login. If the credentials do not match what is stored in the database, the user will receive a message saying that the details were incorrect.

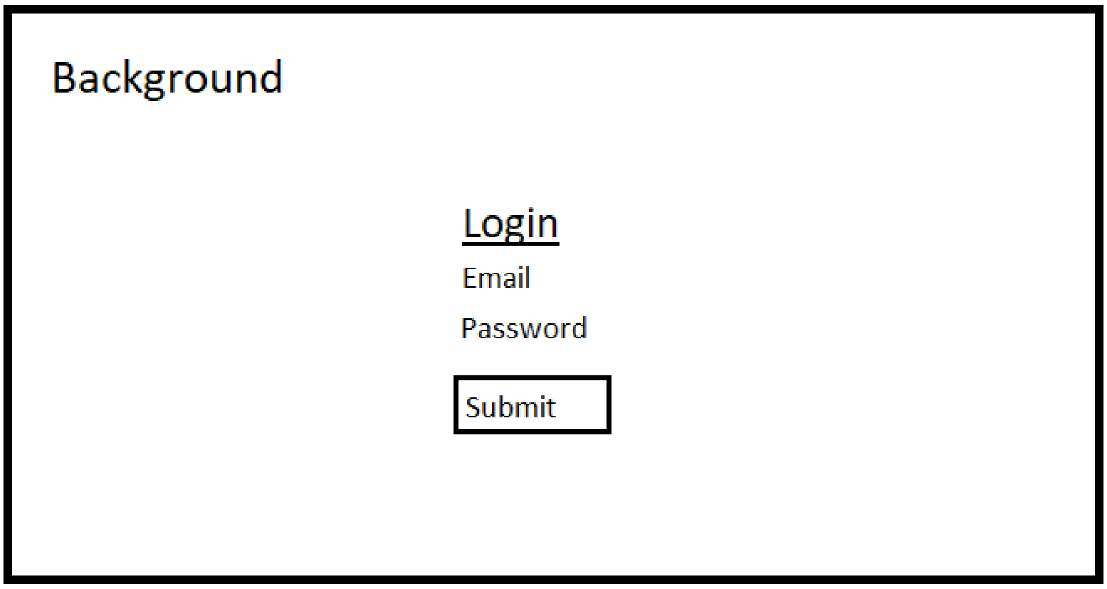


Figure 4.4 Wireframes – Login Page

Home Screen (User)

Once a User has logged in, they will be greeted with the home page. We have decided to have the users QR code on the home screen for ease of use. This way the user will be able to easily pull up their QR code if they are waiting to get scanned.

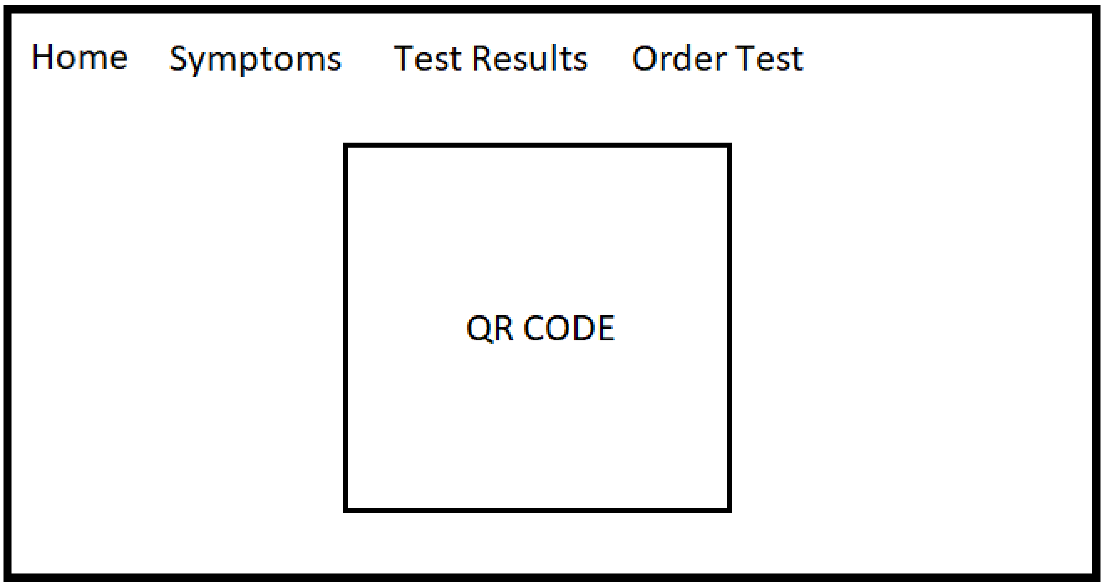


Figure 4.5 Wireframes – User Home Screen

Upload Symptoms

Bellow we can observe how a user can upload what symptoms they have. In order for a user to do this, we list out all of the known and common symptoms of Coronavirus. All the user has to do is check each text box against the symptom they have, and then click save.

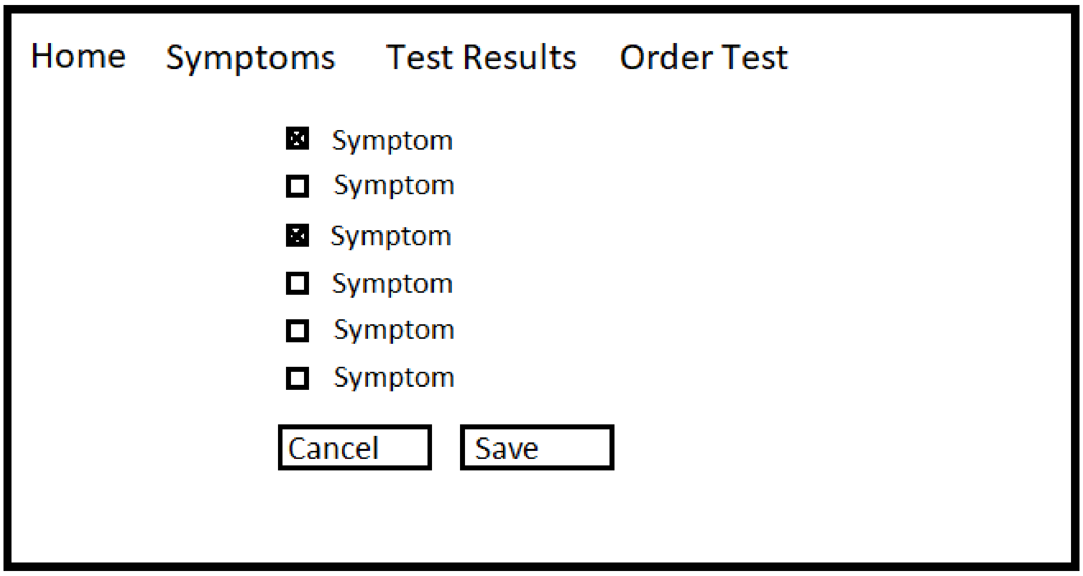


Figure 4.6 Wireframes – Upload Symptoms Screen

Order Test

Here we can see the process for ordering an at home Coronavirus test kit. If the system determines that the user has Coronavirus based on their symptoms, the user will then be able to order a test kit. They will have to input their personal shipping details in order for it to be shipped to them.

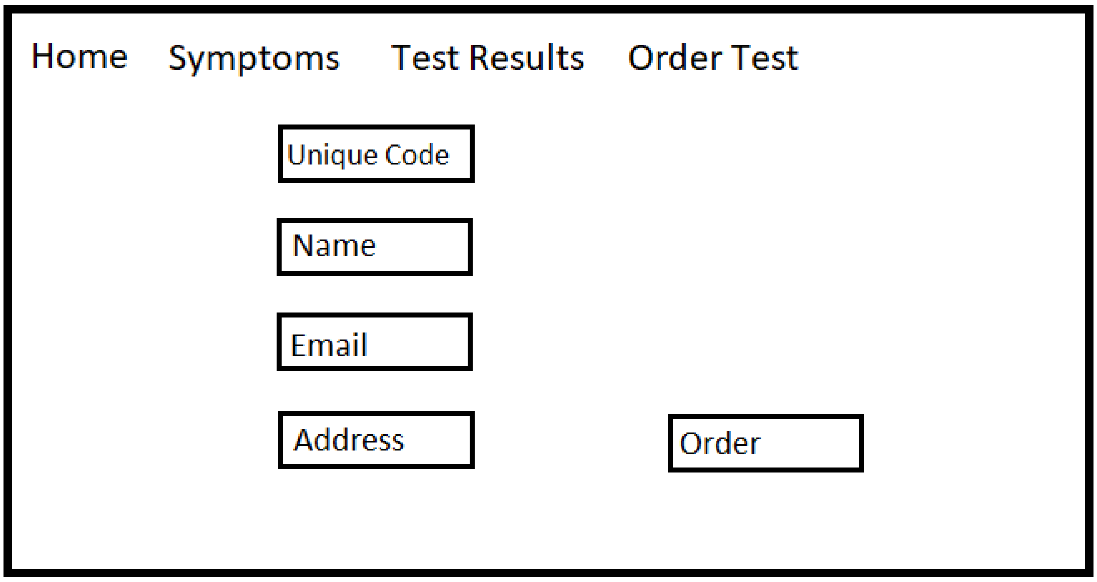


Figure 4.7 Wireframes – Order Test Screen

Upload Test Results

In figure 4.8 we can see the process to upload their test results after they have completed it. The first entry field is for a "Unique Code" this is given to the user with their test kit. This is crucial so as we know which User has taken that test.

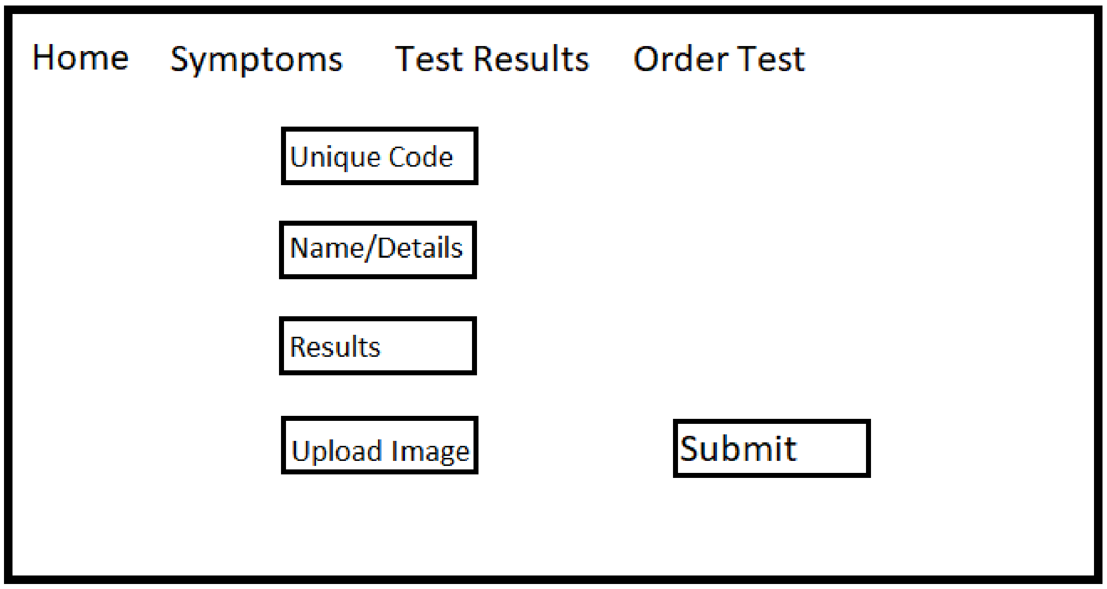


Figure 4.8 Wireframes – Upload Test Results Screen

Home Screen (Business Account)

In figure 4.9 bellow we can see that a business account has a different home screen to a normal user. This layout benefits the employee that is scanning users/ customers. The employee can quickly access their camera to be able to scan a user.

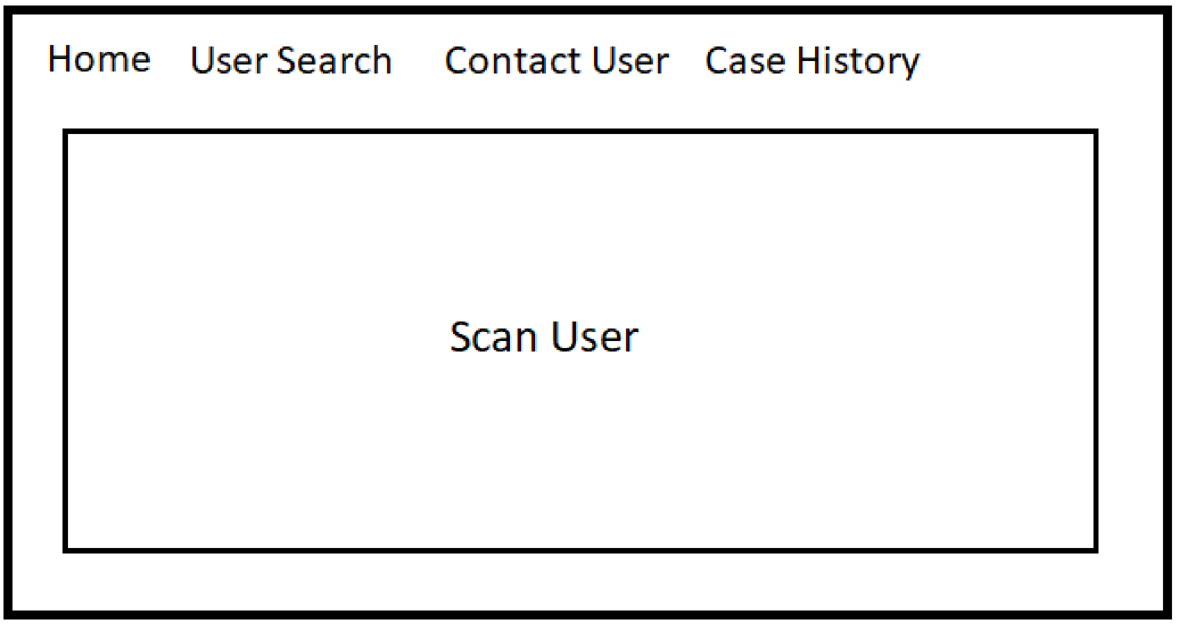


Figure 4.9 Wireframes – Employee Home Screen

User Search (Trace Operative)

In figure 4.10 bellow we can see the user search screen for the trace operatives to use. They can easily enter any search criteria and anyone that matches will be displayed as a result in the "Search Results" box.

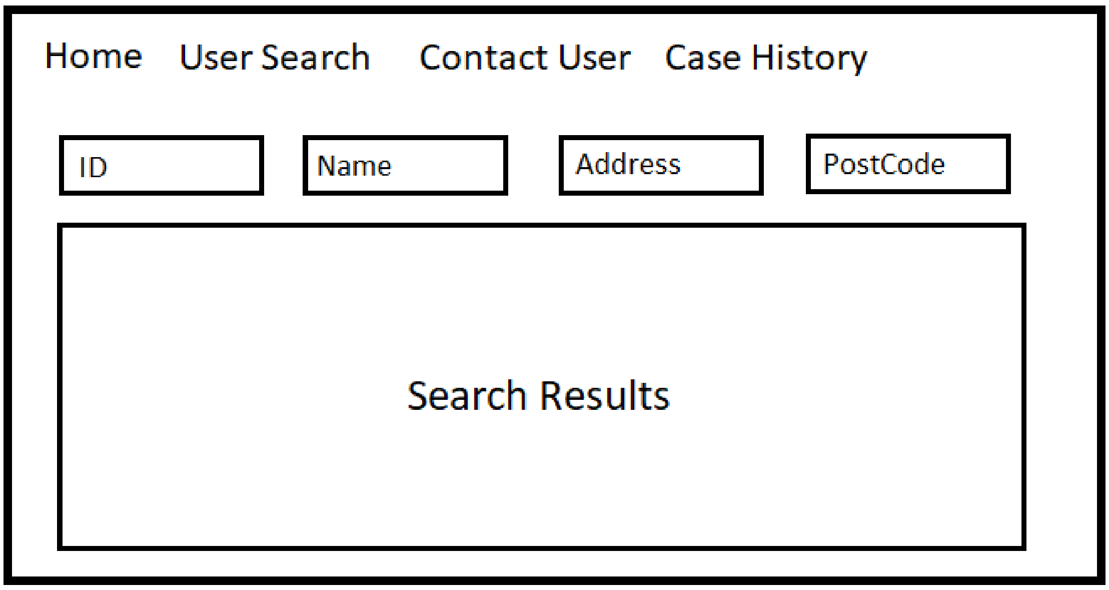


Figure 4.10 Wireframes – Trace Operative User Search

View Cases (Trace Operative)

In figure 4.11 bellow we can see the screen for Trace Operatives to view the cases that have been assigned to them. From here they will be able to select and view a case. They will also be able to edit and alter cases by adding selected users to a case.

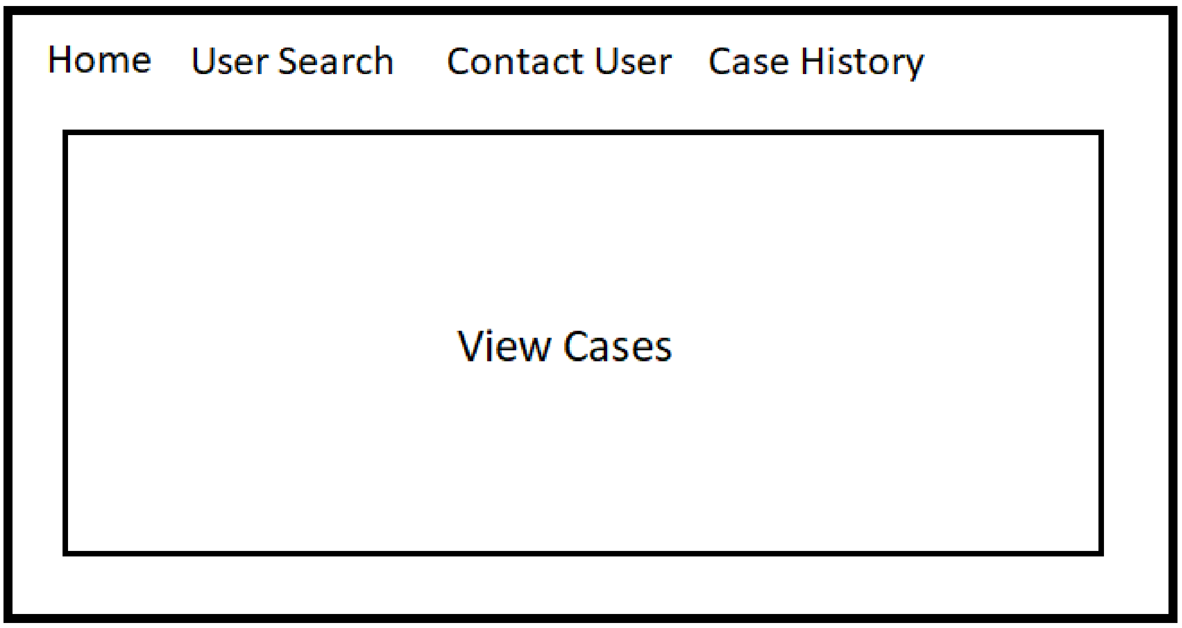


Figure 4.11 Wireframes – Trace Operative View Cases

Contact Users (Trace Operative)

In figure 4.11 bellow we can see the screen for Trace Operatives to contact users. They can do this easily by entering the UserID’s of each user. They could acquire these by viewing their assigned cases or by searching for users.

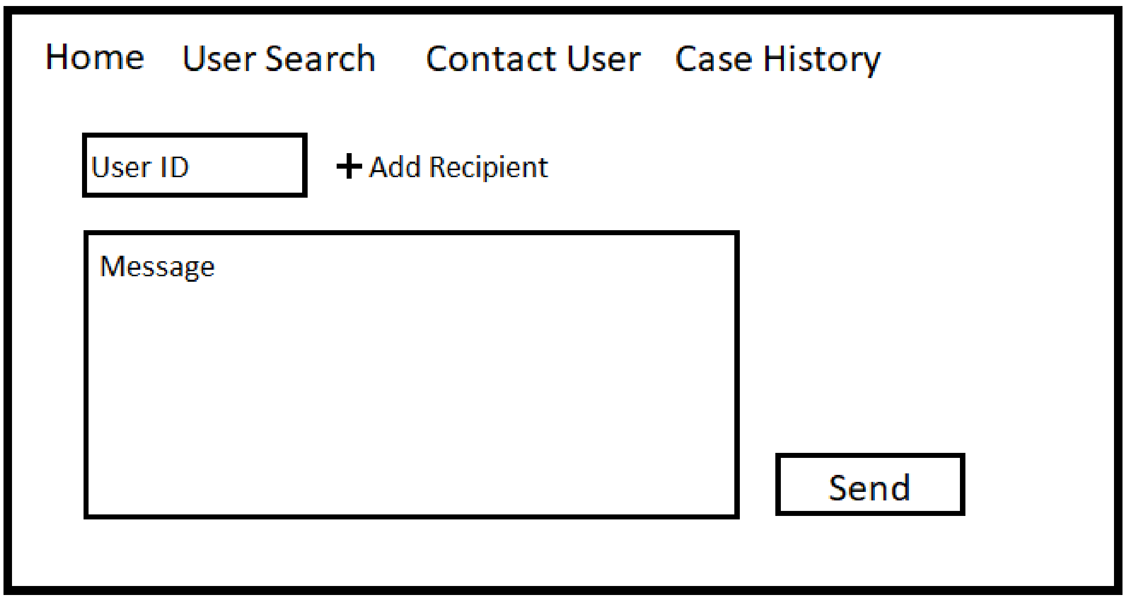


Figure 4.12 Wireframes – Trace Operative Contact Users

4.4 Conclusion

This chapter focused on the design of the application and how it meets our functional requirements. Based on our comprehensive design chapter we can now easily create and populate our database knowing that we will not encounter data redundancy. While designing the sketches and wireframes for each page of the system, difficulties arose that were tackled successfully. For example, the home screen for Users is always designed for ease of use. We ended up not including two factor authentications when logging into the system, as this could waste time when trying to get their QR code scanned.

5. Implementation Chapter

5.1 Introduction

In the implementation chapter we will be collating all the information written in the previous chapters to start developing the required features and functionality. A discussion will be had about the tools, technology and software architecture used to develop the included features.

5.2 Technologies

5.2.1 WinSCP

WinSCP is the application I am using to access my files stored on the KUnet web server. The build in SFTP (Secure file transfer protocol) makes it easy to upload and download files straight from my webserver. I can also easily create files and folders to store my files. I have setup WinSCP to have my preferred IDE be my default code editor, meaning that I can edit files straight from the web server. Also, WinSCP allows me to encrypt my files when transferring, this uses AES-256 encryption (File Encryption : WinSCP, 2021).

5.2.2 Visual Studio Code

Visual Studio Code is the development environment I have chosen to create my application in. This application offers support for a wide variety of languages. It has a user-friendly interface and isn't cluttered with lots of unnecessary features. The built-in text prediction helps to speed up the process of writing code. I can setup visual studio code to be my default editor for WinSCP meaning that whenever I save my code it will be saving it to the web server. This helps me to be able to check and test that the code I just wrote works. As well as this I can "Customize every feature to your liking and install any number of third-party extensions" (Code, 2021).

5.2.3 phpMyAdmin

PhpMyAdmin is a free open-source administration tool for MySQL databases. I am using the service offered my Kingston University to host my database and store information on. PhpMyAdmin is one of the most popular platforms for small web hosting solutions. In this application you can quickly and easily access your database tables. There is the ability to edit your tables through their user-friendly interface, as well as drop and create new tables. There is a section for you to run your own SQL commands, this makes it useful to test commands before implementing them into my own application.

5.2.4 PHP/HTML

PHP and HTML have been my choice of programming languages for this web application. I have chosen to use the Model View Control (MVC) approach to creating my application. PHP is an open-source scripting language that is very useful for web environments. It has the ability to be embedded directly into HTML code. PHP has a range of features that make it ideal for my project. One of those features are its PDO (PHP Data Objects), this allows for easy and efficient access of databases. HTML is a universally known language that is associated with web development. The HTML aspect of my project will predominantly be the front end, what the user will see.

5.2.5 Mockeroo

This is an amazing tool that will help in the data generation for my project. There are loads of data types you can choose to generate. For example, you can generate region specific addresses and phone numbers. As my project will require lots of dummy data, this tool will definitely help. You can choose exactly how many entries you want to generate.

5.3 Software Architecture

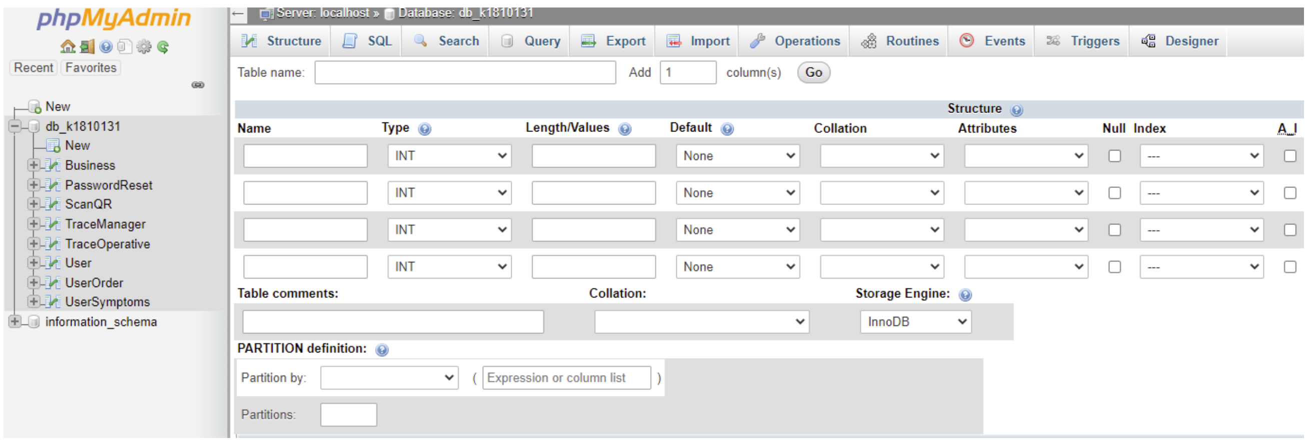
As discussed previously in the design chapter, the MVC (model, view, controller) architecture was proposed for the project. The purpose of this architecture is a split up the front end and backend coding into separate documents. This approach makes the code easy to view and understand, it also doesn't allow users to view any of the backend processing which is helpful for security. "It works well for developing web applications which need to be supported by large teams of developers and for Web designers who wants greater control over the application behaviour" (Top 6 Most Important Benefits of MVC Architecture for Web Application Development Process - Siya Infotech, 2021). Choosing the correct architectural approach can have a big impact on the future of the product. The MVC software design is still very common in modern applications. This approach also allows new developers to easily alter and update the code.

MVC stands for model, view and controller. In the model section you will typically find all of the data access files. This includes all of the SQL statements needed for the application. You can also find the classes in the model, for example there are different classes for a general user and a business account. In the view you will find all of the front end HTML pages. These files/pages contain the HTML that will be displayed for the user. The controller contains the backend processing files. For example, when a user logs in there is a database call to fetch the users personal details if their credentials were correct.

5.4 Database Implementation

5.4.1 Creating the Database

The database was created in phpMyAdmin and contains 8 different tables. They were all created using the built in table creator. This was very helpful as I didn't have to write the SQL to create the database. Some tables include foreign keys to other tables, this makes it easier to perform SQL queries. The table creator can be seen in the picture bellow.



Server: localhost » Database: db_k1810131

- New
- db_k1810131
  - New
  - Business
  - PasswordReset
  - ScanQR
  - TraceManager
  - TraceOperative
  - User
  - UserOrder
  - UserSymptoms
- information_schema

Table name: Add 1 column(s)

Structure

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index	A_I
	INT		None			☐	---	☐
	INT		None			☐	---	☐
	INT		None			☐	---	☐
	INT		None			☐	---	☐

Table comments: Collation: Storage Engine: InnoDB

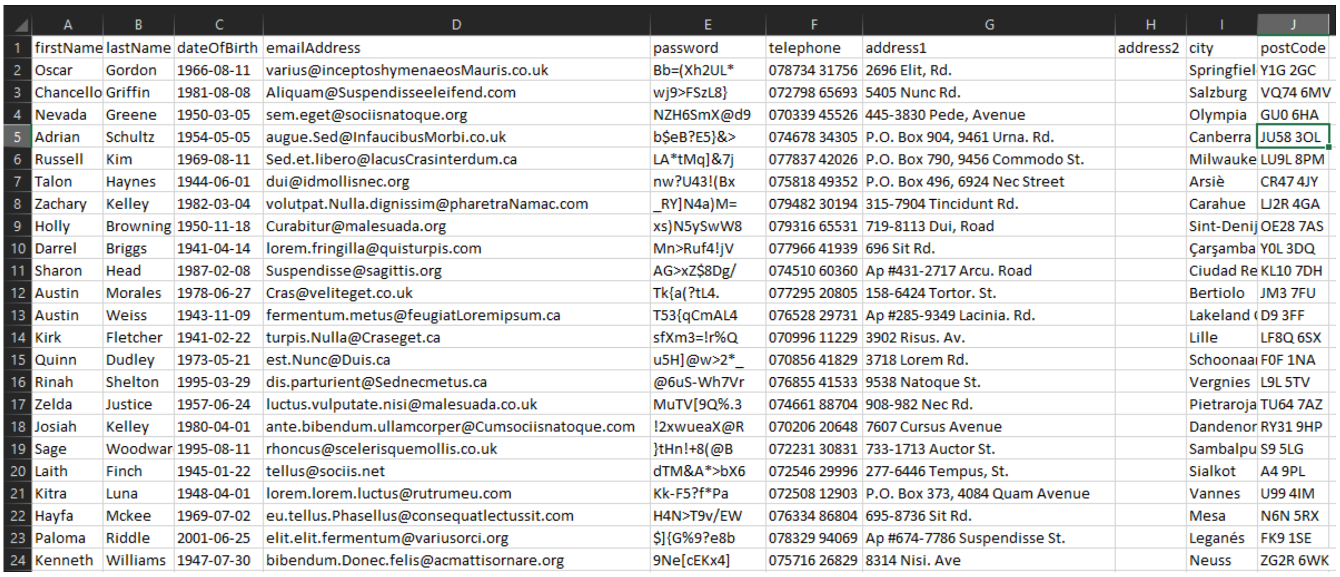
PARTITION definition:

Partition by: (Expression or column list)

Partitions:

5.4.2 Populating the Database

Trying to imitate the number of database entries for the population of a whole country isn't easy. However, mockeroo was used to generate thousands of entries. The entries were stored in specifically structured CSV files. They were then imported to the correct tables using the built in feature in phpMyAdmin. Bellow is a picture of the generated CSV file.



	A	B	C	D	E	F	G	H	I	J
1	firstName	lastName	dateOfBirth	emailAddress	password	telephone	address1	address2	city	postCode
2	Oscar	Gordon	1966-08-11	varius@inceptoshymenaeosMauris.co.uk	Bb=(Xh2UL*	078734 31756	2696 Elit, Rd.		Springfiel	Y1G 2GC
3	Chancello	Griffin	1981-08-08	Aliquam@Suspendisseeleifend.com	wj9>FSzL8}	072798 65693	5405 Nunc Rd.		Salzburg	VQ74 6MV
4	Nevada	Greene	1950-03-05	sem.eget@sociisnatoque.org	NZH6SmX@d9	070339 45526	445-3830 Pede, Avenue		Olympia	GU0 6HA
5	Adrian	Schultz	1954-05-05	augue.Sed@InfaucibusMorbi.co.uk	b$eB?E5}&>	074678 34305	P.O. Box 904, 9461 Urna. Rd.		Canberra	JU58 3OL
6	Russell	Kim	1969-08-11	Sed.et.libero@lacusCrasinterdum.ca	LA*tMq]&7j	077837 42026	P.O. Box 790, 9456 Commodo St.		Milwauke	LU9L 8PM
7	Talon	Haynes	1944-06-01	dui@idmollisnec.org	nw?U43!(Bx	075818 49352	P.O. Box 496, 6924 Nec Street		Arsiè	CR47 4JY
8	Zachary	Kelley	1982-03-04	volutpat.Nulla.dignissim@pharetraNamac.com	_RY]N4a)M=	079482 30194	315-7904 Tincidunt Rd.		Carahue	LJ2R 4GA
9	Holly	Browning	1950-11-18	Curabitur@malesuada.org	xs)N5ySwW8	079316 65531	719-8113 Dui, Road		Sint-Denij	OE28 7AS
10	Darrel	Briggs	1941-04-14	lorem.fringilla@quisturpis.com	Mn>Ruf4!jV	077966 41939	696 Sit Rd.		Çarşamba	Y0L 3DQ
11	Sharon	Head	1987-02-08	Suspendisse@sagittis.org	AG>xZ$8Dg/	074510 60360	Ap #431-2717 Arcu. Road		Ciudad Re	KL10 7DH
12	Austin	Morales	1978-06-27	Cras@veliteget.co.uk	Tk{a(?tL4.	077295 20805	158-6424 Tortor. St.		Bertiolo	JM3 7FU
13	Austin	Weiss	1943-11-09	fermentum.metus@feugiatLoremipsum.ca	T53{qCmAL4	076528 29731	Ap #285-9349 Lacinia. Rd.		Lakeland (	D9 3FF
14	Kirk	Fletcher	1941-02-22	turpis.Nulla@Craseget.ca	sfXm3=!r%Q	070996 11229	3902 Risus. Av.		Lille	LF8Q 6SX
15	Quinn	Dudley	1973-05-21	est.Nunc@Duis.ca	u5H]@w>2*_	070856 41829	3718 Lorem Rd.		Schoonaa	F0F 1NA
16	Rinah	Shelton	1995-03-29	dis.parturient@Sednecmetus.ca	@6uS-Wh7Vr	076855 41533	9538 Natoque St.		Vergnies	L9L 5TV
17	Zelda	Justice	1957-06-24	luctus.vulputate.nisi@malesuada.co.uk	MuTV[9Q%.3	074661 88704	908-982 Nec Rd.		Pietraroja	TU64 7AZ
18	Josiah	Kelley	1980-04-01	ante.bibendum.ullamcorper@Cumsociisnatoque.com	!2xwueaX@R	070206 20648	7607 Cursus Avenue		Dandenor	RY31 9HP
19	Sage	Woodwar	1995-08-11	rhoncus@scelerisquemollis.co.uk	}tHn!+8(@B	072231 30831	733-1713 Auctor St.		Sambalpu	S9 5LG
20	Laith	Finch	1945-01-22	tellus@sociis.net	dTM&A*>bX6	072546 29996	277-6446 Tempus, St.		Sialkot	A4 9PL
21	Kitra	Luna	1948-04-01	lorem.lorem.luctus@rutrumeu.com	Kk-F5?f*Pa	072508 12903	P.O. Box 373, 4084 Quam Avenue		Vannes	U99 4IM
22	Hayfa	Mckee	1969-07-02	eu.tellus.Phasellus@consequatlectussit.com	H4N>T9v/EW	076334 86804	695-8736 Sit Rd.		Mesa	N6N 5RX
23	Paloma	Riddle	2001-06-25	elit.elit.fermentum@variusorci.org	$]{G%9?e8b	078329 94069	Ap #674-7786 Suspendisse St.		Leganés	FK9 1SE
24	Kenneth	Williams	1947-07-30	bibendum.Donec.felis@acmattisornare.org	9Ne[cEKx4]	075716 26829	8314 Nisi. Ave		Neuss	ZG2R 6WK

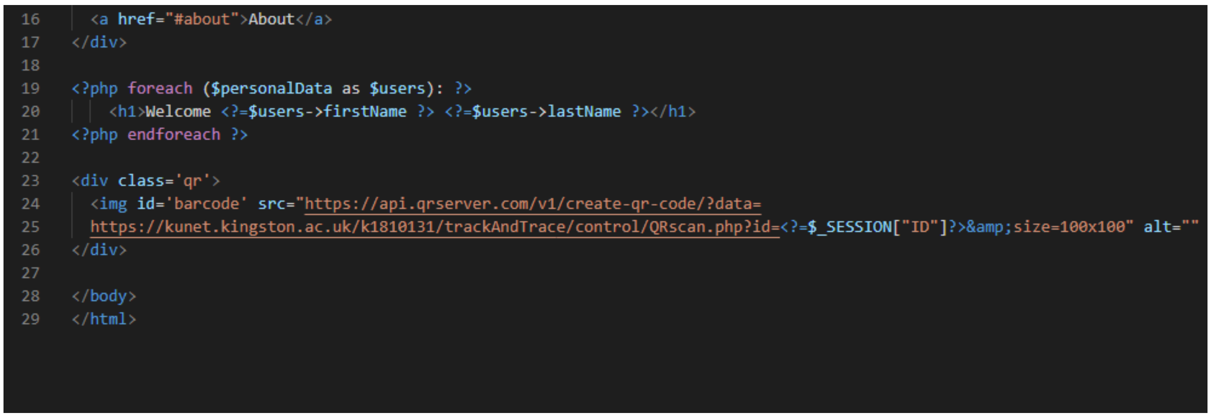
5.4.3 Querying the Database

When querying the database, I used SQL statements to access the database. phpMyAdmin has a helpful tool that allows you to perform queries on the database in your browser. This allowed me to test my statements without implementing them into my code straight away. PHP PDO was utilized to help prevent again SQL injection attacks.

5.5 QR Code Implementation

5.5.1 Generating QR Codes

In order to give every user a unique QR code I had to use a third party API. It was very simple to use and only had to call the API URL to create a QR code. As you can see from the API call in the picture bellow, I had to include some sort of data to be passed to the person scanning the QR codes. My approach to this was to send the shop employee to a specific URL. To be able to distinguish between different users, their UserID is embedded in the URL. Bellow shows the API call to create the QR code.



```
    <a href="#about">About</a>
  </div>

  <?php foreach ($personalData as $users): ?>
      <h1>Welcome <?=$users->firstName ?> <?=$users->lastName ?></h1>
  <?php endforeach ?>

  <div class='qr'>
    <img id='barcode' src="https://api.qrserver.com/v1/create-qr-code/?data=
    https://kunet.kingston.ac.uk/k1810131/trackAndTrace/control/QRscan.php?id=<?=$_SESSION["ID"]?>&amp;size=100x100" alt=""
  </div>

  </body>
  </html>
```

5.5.2 Scanning QR Codes

In a perfect scenario a shop employee would scan everyone’s QR code. This would require the employee to be signed into the website on their device using their default web browser. At this point they can then use their camera application to scan the QR codes. Once a QR code is scanned they will be directed to their home page. During this process a SQL insert query is made that will store the users scan data in the database.

5.6 Conclusion

In conclusion, the implementation of the requirements has been made significantly easier for the developer with the help of specific technologies and code design patterns. The implementation was successful as most of the user requirements were developed and implemented.

6. Testing and Validation Chapter

6.1 Introduction

This chapter will focus on demonstrating the different testing techniques used to test the core functionality that was demonstrated in the implementation phase. Testing is crucial in the software development life cycle. The aim is to test as much implemented functionality as possible. We hope that the testing comes back with no errors, however sometimes we are better off receiving errors during testing and not when the system is live.

6.2 Test Strategy

A crucial step to validation is testing functional and non-functional requirements. There are lots of different methods you can use to test your application. In our case we will be using black box testing for the functional and non-functional requirements. This testing strategy will be used as the foundations to test if the requirements have been met successfully.

In addition to black box testing on our requirements, we will be testing the database validation. This can be accomplished by looking at the design chapter and comparing our database structure to the ERD diagram created. We could also make use of the data dictionary by checking features such as the data type and the length of a field.

6.2.1 Black Box Testing

Black box testing can be done to either functional or non-functional requirements. This type of testing is usually done without any knowledge of the internal code structure. It also mainly focuses on the input and output of data in the system. For example, a post code field should only allow up to 8 letters or numbers. If I attempt to enter 10 characters, I should receive an error. Another example would be if the system is meant to output certain data. You can easily validate that this data is correct.

6.2.2 White Box Testing

White box testing is when you test the internal structure of a software component. An example of this type of test would be to test for security flaws. In our scenario we would accomplish this by testing for SQL Injection attacks. You may also test the flow of data through your code. If you are inefficient then you could clean up your code.

6.3 Database Integrity Validation

This system wouldn't function without implementing a backend database to store crucial user data and QR code data. Without a database some of our requirements would not be met, for example, the login functionality requires us to store the users email and password.

If we refer to the ERD diagram created in the design phase, we can validate some of the data surrounding the database. For example, we can test the integrity of the database.

The database was created with phpMyAdmin and we will be using a range of different SQL statements to test the database. Table 6.1 shows the different test cases and further test cases can be found in the appendix.

Test No.	**Test Case**	**Test Description**	**Expected Result**	**Actual Result**
1	Testing Datatypes	Test inputting the wrong datatype. E.G. Inserting a string into a integer type	An error should occur	An error was displayed
2	Insert NULL value into required field	Test inserting a NULL (blank) value into a field that doesn't allow NULL values. (E.G. PostCode)	The insert should not happen	An error was displayed on the screen
3	Insert record with a foreign key that doesn't exist	Test inputting a record without including a foreign key constraint	The insert should not happen	System output an error

Table 6.1: Test cases for database

6.4 Requirements Testing

The functional requirements specified in the requirement analysis will help us with our testing. We can use our previous requirements as specific test cases. In our test tables there are detailed instructions on how we completed the requirement. Appendix D shows the black box testing applied to the functional requirement tests.

Test ID	**1**	**User Type**	General Public
Test Goal		Register Account	
Test Description		A member of the public should be able to register an account	
Instruction		1. At login screen click "Register" 2. Input details requested 3. Click "Register"	
Expected Results		Account should be created and input into database	

Test Case 6.2

6.5 Conclusion

In conclusion, the tests came back successfully. The database validation is crucial to undertake as the system would not work without the database implementation. However, some of the database validation can be easily solved with some front-end validation. This would mean that data is checked before any SQL queries are completed. The requirements testing was successful and all stakeholder requirements were completed.

7. Critical Review Chapter

7.1 Introduction

The following chapter will review the whole project. We will discuss how it was completed, also what was and was not achieved. We can achieve this by comparing the requirements against the actual product. After this, there will be a discussion about what I had learnt and enjoyed while completing this project.

7.2 Requirements Review

ID	Requirement	Status
1	A General User and a Business User should be able to create separate accounts, the system should record personal info during this time	Complete
2	A trace manager should be able to create and delete trace operative accounts	Complete
3	All users should be able to log into the system and access their respected areas	Complete
4	A User should be able to log out of their account once they are done	Complete
5	A user should be generated a QR code whenever they signup, they should also be able to view it from their home screen	Complete
6	A user should be able to get their barcode scanned by a shop employee	Complete
7	The user should be able to report their current symptoms	Complete
8	Users that show symptoms should be able to order a test kit	Complete
9	A user should be able to upload their COVID test results	Complete
10	A trace manager should be able to assign positive COVID cases to specific trace operatives	Complete
11	A trace operative should be able to view and manage all of their cases	Complete

12	A trace operative should be able search for specific users	Complete
13	A trace operative should be able to add members to a COVID case	Complete
14	A trace operative should be able to contact users by email	Complete
15	A trace operative should be able to contact users by texting their phone number	Not Complete
16	A trace operative should be able to update the status of a case to complete	Complete
17	A trace manager should be able to create case reports	Not Complete

Table 7.1 – Requirements Review

As shown in Table 7.1, almost all our set requirements were implemented. There were only two outstanding requirements that were not implemented. These were requirements with ID 15 (Contact User by Phone Number) and ID 17 (Create Case Reports). The reason for requirement 15 not being implemented was due to the abundance of APIs that would support texting phone numbers. Requirement 17 was not completed due to time constraints. It was deemed necessary that other core functionality was to be completed before this requirement.

7.3 Achievements

The adoption of Agile software development methodology meant that the software was developed iteratively while following a work schedule. This means that all aspects of requirements, design and implementation were completed on time. The quality of the software was prioritised over the number of features. However, most of the requirements were implemented, meaning that that end product was of high quality.

The research that was taken for the literature review explored different Web Technologies. This research helped tremendously with the decisions of what technologies should be used for the project. As a result of the findings in the literature review, a decision was made to use HTML and PHP for the development of the project. As well as undertaking the MVC (Model-View-Control) development approach.

The requirements analysis process gave a better understanding of what the client desired in the system. The stakeholder analysis identified the main users. Once the users were identified, interviews took place in order to build upon the user stories. This gave a good understanding of the system requirements and what needed to be implemented.

The design stage helped identify whether the system can support the requirement set during the requirements chapter. For the backend database, an ER(Entity-Relationship) diagram was created, not only does it help with visualizing what the database will look like, but also helps identify what tables are joined together with relationships. The activity and sequence diagrams identified the workflow of the system and what roadmaps the user would have to undertake. It is also in the design chapter where the MVC design pattern, recommended by the literature review, was chosen. After the back end was designed the sketches and wireframes were made to help visualise the screens that the users will visit. These wireframes were shown to the clients and some users to gain feedback and suggested improvements.

The implementation stage was successful because most features/requirements were implemented in time. The MVC design pattern meant that all of the code was produced to a high standard and was organised efficiently. This resulted in the system being easy to maintain and debug.

The last chapter was the testing chapter. This chapter was validating that the requirements had been implemented correctly. The system was also tested to check if it is fit for purpose and whether it would cope in real scenarios. The application passed all black box and white box testing apart from the few requirements that were not implemented. The backend database was also tested for its functionality, as well as how it handles certain data. The database testing all passed including its checks against SQL injection attacks.

7.4 Improvements

There were a few improvements that could have been done to the system that could be implemented next time. The first improvement would be the website design. The features and functionality were prioritised over the design and look of the system. Some very basic styling was implemented; however a cleaner and more friendly look could improve the system dramatically. This could have been done by implementing a third party library called Bootstrap that helps by providing some template styling.

Another feature that would have been nice to implement is the ability for trace operatives to add businesses to their assigned COVID cases. This way the business could be included in the process of contacting users that may be COVID-19 positive.

Another basic improvement is the formatting of emails. Currently all emails (password reset email, contacting users email) are basic text. Some formatting could be done to these emails to make them more presentable and user friendly. Also, trace operatives have to type out the email they want to send to users, a default email could be placed there with the ability to edit it.

7.5 Conclusion

Overall, this project has been a lot of fun to complete. It has also broadened my development knowledge because of how much research had to take place. I can now take this knowledge and use it in my next project and hopefully increase my knowledge with every project. The obstacles with completing this project were tough but were mostly all accomplished. Based on the fact that the system passed the functional and usability testing, the project has been a success.

References

- Code, V., 2021. Why Visual Studio Code?. [online] Code.visualstudio.com. Available at: <https://code.visualstudio.com/docs/editor/whyvscode> [Accessed 25 March 2021].
- Winscp.net. 2021. File Encryption :: WinSCP. [online] Available at: <https://winscp.net/eng/docs/file_encryption> [Accessed 25 March 2021].
- Siyainfo.com. 2021. Top 6 Most Important Benefits of MVC Architecture for Web Application Development Process - Siya Infotech. [online] Available at: <https://siyainfo.com/2017/01/16/top-6-important-benefits-mvc-architecture-web-application-development-process/> [Accessed 25 March 2021].
- StatCounter Global Stats. 2021. StatCounter Global Stats - Browser, OS, Search Engine including Mobile Usage Share. [online] Available at: <https://gs.statcounter.com/> [Accessed 25 March 2021].
- Statista. 2021. Most popular *database technologies for developers 2020 | Statista*. [online] Available at: <https://www.statista.com/statistics/794187/united-states-developer-survey-most-wanted-used-database-technologies/> [Accessed 25 March 2021].
- AltexSoft. 2021. *Comparing Database Management Systems: MySQL, PostgreSQL, MSSQL Server, MongoDB, Elasticsearch and others*. [online] Available at: <https://www.altexsoft.com/blog/business/comparing-database-management-systems-mysql-postgresql-mssql-server-mongodb-elasticsearch-and-others/> [Accessed 25 March 2021].
- Clockwise.software. 2021. *Relational vs non-relational databases: advantages and disadvantages - Clockwise Software*. [online] Available at: <https://clockwise.software/blog/relational-vs-non-relational-databases-advantages-and-disadvantages/> [Accessed 25 March 2021].
- Mistry, J., 2021. *Top 10 Web Development Framework With Detailed Comparison*. [online] Monocubed. Available at: <https://www.monocubed.com/web-development-framework-comparison/> [Accessed 25 March 2021].
- W3techs.com. 2021. Usage Statistics and Market Share of HTML5 for Websites, April 2021. [online] Available at: <https://w3techs.com/technologies/details/ml-html5> [Accessed 7 April 2021].
- QRA Corp. 2021. Functional vs Non-Functional Requirements: The Definitive Guide - QRA Corp. [online] Available at: <https://qracorp.com/functional-vs-non-functional-requirements> [Accessed 16 April 2021].

Appendix A

Use Cases

ID	**02**
Name	Get Scanned
Actor	General Public
Description	Get your QR code scanned by an employee
Pre-Condition	Person is not logged in Employee is ready to scan
Flow of Events	1. Click on “Log In” 2. Enter details in login 3. QR code should be visible straight after logging in 4. Hold QR code under camera 5. Get confirmation of scanning
Post-Condition	N/A

Table A.1 Use Case

ID	**03**
Name	Register Account
Actor	Shop Employee
Description	Register for a business account
Pre-Condition	Business does not already have an account
Flow of Events	1. Click on “Register” 2. Click on “Business Account” 3. Enter business details. E.G. Name, Address and Email 4. Click on “Register”
Post-Condition	User would then log in with same details

Table A.2 Use Case

ID	**04**
Name	Register Account
Actor	General Public
Description	Register for a personal Account
Pre-Condition	Person does not already have an account
Flow of Events	1. Click on “Register” 2. Click on “Personal Account” 3. Enter personal details 4. Click on “Register”
Post-Condition	User would then log in with same details

Table A.3 Use Case

ID	**05**
Name	Upload Symptoms
Actor	General Public
Description	Upload Symptoms to system and get feedback
Pre-Condition	User has logged in
Flow of Events	1. Click on “Declare Symptoms” 2. Answer each question honestly 3. Click on “Finish” 4. Receive feedback based on your symptoms
Post-Condition	N/A

Table A.4 Use Case

ID	**06**
Name	Order Test
Actor	General Public

Description	Order a test to complete at home
Pre-Condition	User has logged in User has received feedback saying they should order a kit
Flow of Events	1. Click on “Order Test” 2. Insert Unique code 3. Confirm personal shipping details 4. Click on “Order”
Post-Condition	N/A

Table A.5 Use Case

ID	**07**
Name	Upload Test Results
Actor	General Public
Description	Upload Test Results to the system
Pre-Condition	User has logged in User has completed the test kit
Flow of Events	1. Click on “Upload Results” 2. Upload test results 3. Click on “Finish”
Post-Condition	N/A

Table A.6 Use Case

ID	**08**
Name	Query Database
Actor	Trace operative
Description	Search database for people who were in contact with someone positive
Pre-Condition	User has logged in
Flow of Events	1. Click on “Trace Users” 2. Insert location, time and date 3. Click on “Search”

	4. View list of people
Post-Condition	N/A

Table A.7 Use Case

ID	**09**
Name	Contact Users
Actor	Trace operative
Description	Contact users that have been in contact with someone positive
Pre-Condition	User has logged in
Flow of Events	1. Click on chosen case to contact 2. Insert message to be sent 3. Click “Send”
Post-Condition	N/A

Table A.8 Use Case

Appendix B

Activity and Sequence Diagrams

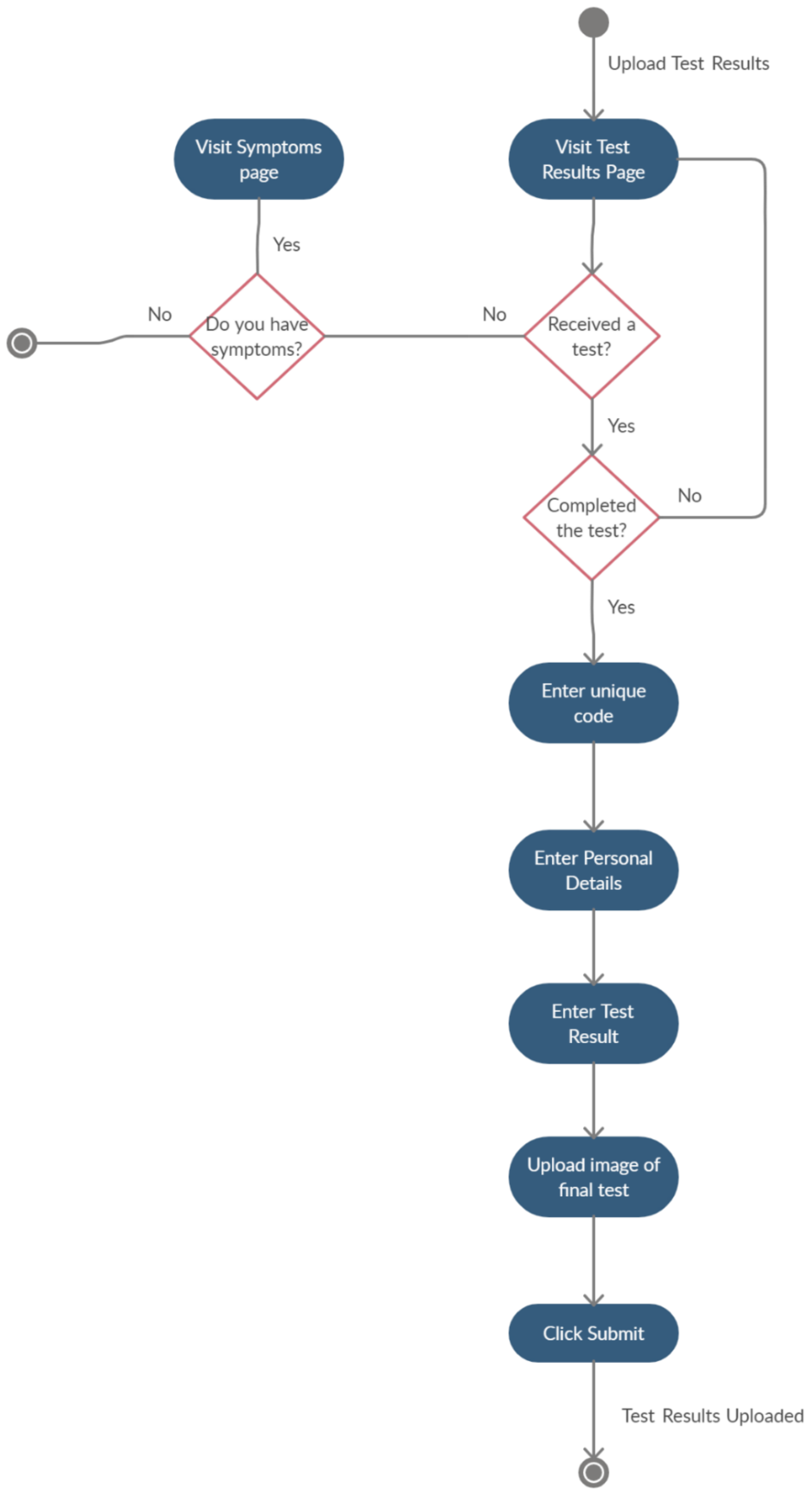


Figure B.1.1 – Uploading Test Results

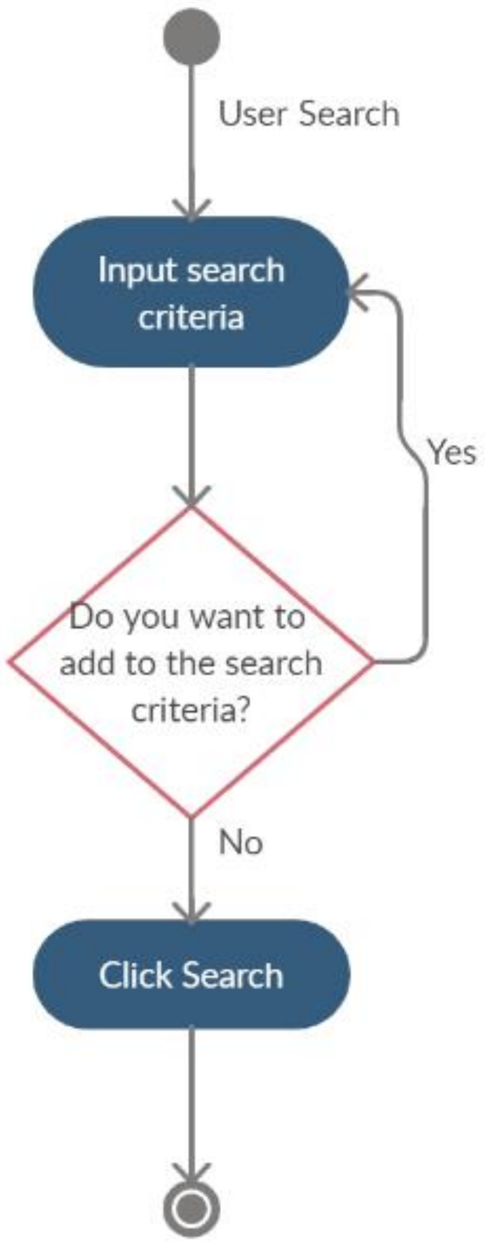


Figure B.1.2 – User Search

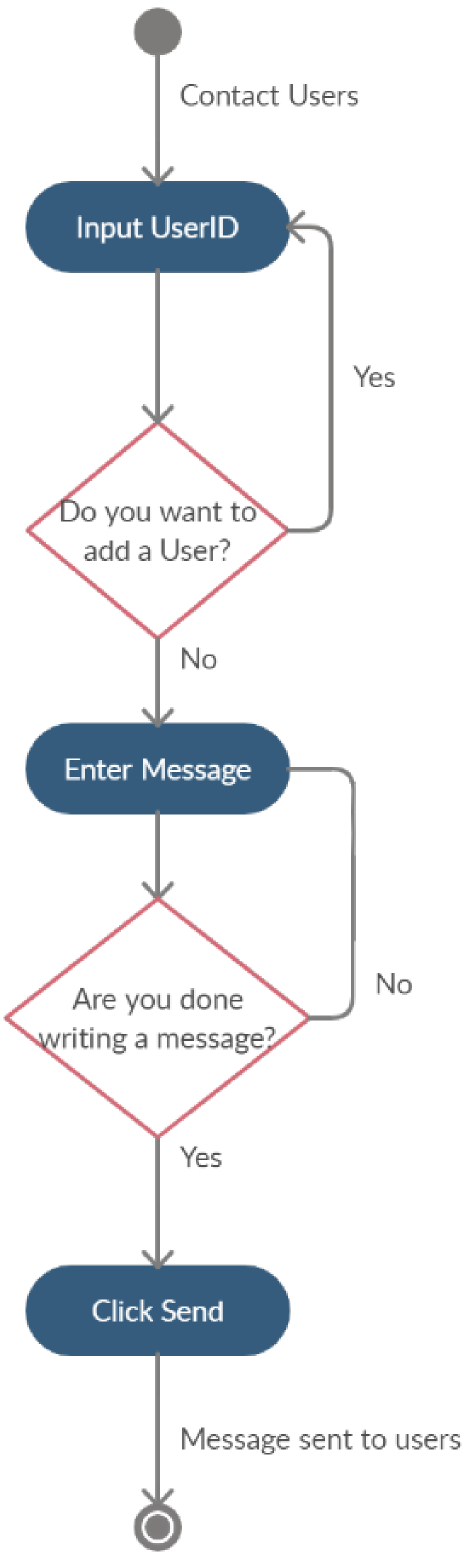


Figure B.1.3 – Contact Users

Appendix C

Data Dictionary

Column	Type	Null	Default	Comments
businessID	Int(10)	No	-	Primary Key
businessName	Varchar(50)	No	-	
businessAddress1	Varchar(200)	No	-	
businessAddress2	Varchar(200)	Yes	NULL	
businessCity	Varchar(50)	No	-	
businessPostCode	Varchar(8)	No	-	
businessPhoneNumber	int(11)	No	-	
businessEmail	Varchar(150)	No	-	
businessPassword	Varchar(50)	No	-	

Table B.1 – Business Table

Column	Type	Null	Default	Comments
symptomID	Int(10)	No	-	Primary Key
userID	Int(10)	No	-	Foreign Key
Fever	Varchar(10)	Yes	-	
Cough	Varchar(10)	Yes	-	
Breathing	Varchar(10)	Yes	-	
Aches	Varchar(10)	Yes	-	
headache	Varchar(10)	Yes	-	
tasteSmell	Varchar(10)	Yes	-	
soreThroat	Varchar(10)	Yes	-	
Congestion	Varchar(10)	Yes	-	

Nausea	Varchar(10)	Yes	-	
diarrehea	Varchar(10)	Yes	-	

Table B.2 – Symptoms Table

Column	Type	Null	Default	Comments
orderID	Int(10)	No	-	Primary Key
trackID	Varchar(20)	Yes	NULL	
orderDate	datetime	No	CURRENT_TIMESTAMP	

Table B.3 – Order Test Table

Column	Type	Null	Default	Comments
scanID	Int(10)	No	-	Primary Key
userID	Int(10)	No	-	Foreign Key
businessID	Int(10)	No	-	Foreign Key
time	datetime	No	CURRENT_TIME	
Date	date	No	CURRENT_DATE	

Table B.4 – QR Scan Table

Column	Type	Null	Default	Comments
caseID	Int(10)	No	-	Primary Key
scanID	Int(10)	No	-	Foreign Key
dateReorted	date	No	-	
caseStatus	Varchar(20)	No	“OPEN”	

Table B.5 – COVID Case Table

Column	Type	Null	Default	Comments
caseMemberID	Int(10)	No	-	Primary Key
caseID	Int(10)	No	-	Foreign Key
scanID	Int(10)	No	-	Foreign Key

Table B.6 – Case Members Table

Column	Type	Null	Default	Comments
assignmentID	Int(10)	No	-	Primary Key
operativeID	Int(10)	No	-	Foreign Key
caseID	Int(10)	No	-	Foreign Key

Table B.7 – Case Assignment Table

Column	Type	Null	Default	Comments
operativeID	Int(10)	No	-	Primary Key

Table B.8 – Trace Oerative Table

Appendix D

Test Tables

Test ID	**2**	**User Type**	Business
Test Goal	Register Account		
Test Description	A business should be able to register an account		
Instruction	1. At login screen click "Register Business Account" 2. Input details requested 3. Click "Register"		

Expected Results	Account should be created and input into database

Table C.1

Test ID	**3**	**User Type**	Operative Manager
Test Goal	Create Trace Operative Account		
Test Description	An operative manager should be able to create a trace operative account		
Instruction	1. Navigate to "Manage Operatives" 2. Fill in details 3. Click "Create"		
Expected Results	A trace operative account will have been made in input into the database		

Table C.2

Test ID	**4**	**User Type**	General Public
Test Goal	User Login		
Test Description	A user should be able to login to their account		
Instruction	1. At login screen input email and password 2. Click "Login" 3. User should be taken to their home page		
Expected Results	User should be able to see their QR code on their home page		

Table C.3

Test ID	**5**	**User Type**	Business
Test Goal	Business Login		
Test Description	A business should be able to login to their account		
Instruction	1. At login screen input email and password 2. Click "Login" 3. User should be taken to their home page		
Expected Results	Business should be directed to their home page		

Table C.4

Test ID	6	User Type	Trace Operative
Test Goal	Trace Operative Login		
Test Description	A Trace operative should be able to login to their account		
Instruction	1. At login screen input email and password 2. Click “Login” 3. User should be taken to their home page		
Expected Results	A user should be able to only order a test if they have a unique code		

Table C.5

Test ID	7	User Type	General Public
Test Goal	Access QR code		
Test Description	User should be able to access their QR code quickly		
Instruction	1. User logs in 2. User should be able to see their QR code on their home screen		
Expected Results	Trace Operative should be taken to their home page and see their assigned cases		

Table C.6

Test ID	8	User Type	Business
Test Goal	Scan QR code		
Test Description	A business should be able to scan a QR code		
Instruction	1. Business logs in 2. Navigate to camera application on phone 3. Hover camera over QR code 4. Click popup to navigate to website		
Expected Results	Business should be scan a QR code and see it in their list of scans		

Table C.6

Test ID	**9**	**User Type**	General Public
Test Goal	Report Symptoms		
Test Description	A user should be able to report their symptoms and receive feedback		
Instruction	1. Navigate to "Report Symptoms" page 2. Fill in information about symptoms 3. Click "Submit" 4. Feedback page should be visible		
Expected Results	A user should be able to see feedback based on the symptoms they input		

Table C.7

Test ID	**10**	**User Type**	General Public
Test Goal	Order Test		
Test Description	After getting a unique code a user should be able to order a test kit		
Instruction	1. Copy unique code after receiving feedback about symptoms 2. Navigate to "Order Test" page 3. Paste unique code in field 4. Fill in personal details 5. Click "Order" 6. Receive feedback if order was successful		
Expected Results	A user should be able to only order a test if they have a unique code		

Table C.8

Test ID	**11**	**User Type**	General Public
Test Goal	Upload Test result		
Test Description	After completing a test the user should be able to upload their results		
Instruction	1. Navigate to "Upload Results" page 2. Fill in unique code 3. Fill in details of test 4. Click "Submit" 5. Receive feedback based on test results		

Expected Results	A user should receive relevant feedback based on their test result

Table C.9

Test ID	**12**	**User Type**	Operative Manager
Test Goal		Assign Cases	
Test Description		An Operative manager should be able to assign COVID cases to operatives	
Instruction		1. Navigate to "Assign Cases" 2. Select an operative from the dropdown next to each case	
Expected Results		The trace operative should now be able to view the case from their home screen	

Table C.10

Test ID	**13**	**User Type**	Trace Operative
Test Goal		User Search	
Test Description		An Operative should be able to search for users	
Instruction		1. Navigate to "User Search" 2. Input fields to search by 3. Click "Search"	
Expected Results		The trace operative should be able to see all of their results	

Table C.11

Test ID	**14**	**User Type**	Trace Operative
Test Goal		Add User to Case	
Test Description		An Operative should be able to add a user to a specific case once searching for a user	
Instruction		1. Search for specific User 2. Click on the User 3. Select the case to add the user to	
Expected Results		The case should now be updated with the new users added	

Table C.12

Test ID	**15**	**User Type**	Trace Operative
Test Goal	Update Cases		
Test Description	An Operative should be able to update their cases		
Instruction	4. Navigate to “Cases” 5. Select a case to update 6. Update case with new information 7. Click “Update”		
Expected Results	The COVID case should now be updated		

Table C.13