

SWE 20004 Technical Software Development

Semester 1 2022

Project Report

Project Title : COVID

Project Team : Eamon & Joshua

Year : 2022

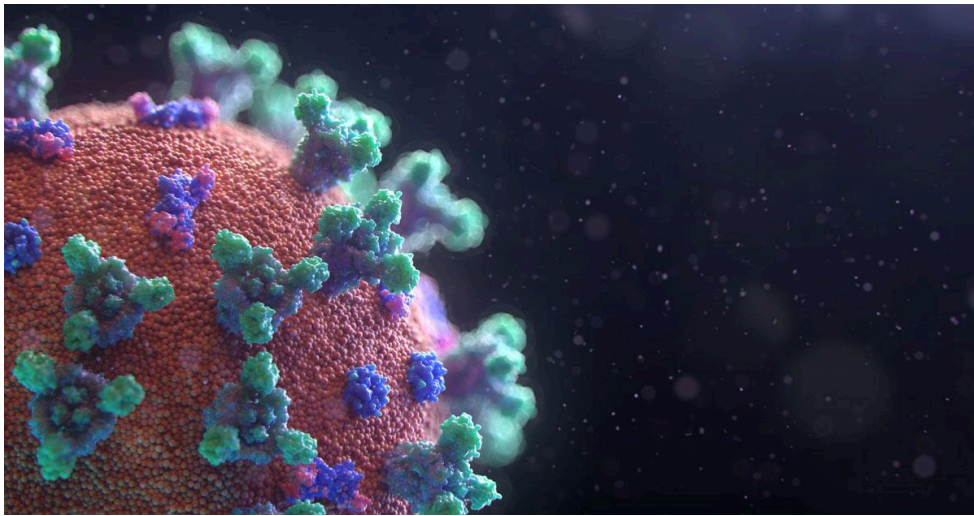


Fig.1: COVID-19 Cell

<https://dizzinessbalancedisorders.com.au/wp-content/uploads/2022/02/Understanding-audio-vestibu.jpg>

File Name : Sem1_Project Report_Eamon & Joshua.pdf

Last Save Date : 11.59pm 31/10/2022

Executive Summary

The following Project Report is the final assessment for Technical Software Development and shows the process behind creating a software solution for the COVID-19 pandemic. It begins with an introduction, describing the project at a basic level as well as explaining the reasons the

program was made and the motivations to do so. The next section is the design concept, starting with the problem statement it goes through the constraints and specifications of the project and then covers the process in creating the final design as well as how it was implemented.

The next section is the design prototype, and shows testing as well as any changes that were made as the program was developed. Finally the report covers the actual project outcomes, this includes a data display, which goes through what it looks like when the code is being run, covering the different menu options, as well as the different scenarios depending on what the user enters, ending with a code explanation, which goes through the important parts of the code explaining how it works.

Next is the individual contributions section where each team member reflects on their work done as well as their technical knowledge of the project, it then covers the 5 learning outcomes in the syllabus, providing 2 examples for each representing how they achieved each one. Finally this report ends with the future recommendations for the software solution as well as a conclusion which links back to the project's objectives and scope.

Table of Content

Executive Summary	1
Table of Content	1
Introduction	3
Background & Motivations	3
Project Description	3
Design Concepts	3
Problem Statement	3
Constraints and Specifications	4
Implementation and Processes	4
Results / Design Prototype	5
Project Outcomes	9
Data Display	9
Code Explanation	16
Individual Contribution #1 (Josh)	18
Reflection	18
Learning Outcomes	18
Technical Learning	20

Individual Contribution #2 (Eamon)	20
Reflection	20
Learning Outcomes	20
Technical Learning	21
Conclusion	21
Future Recommendations	21
Appendix :	21
Coding File	21
Team Work Breakdown	22
References	22

Introduction

Background & Motivations

The COVID-19 pandemic has caused untold disruption to the daily lives of billions of people from 2020 till the present day. It has heavily affected the wellbeing of society as well as the economy, it is the top priority of the government to control the virus before it spirals out of control. Contact tracing has become a difficult task for international governments and digital tools used to assist and automate this process have seen high demand. Automating a portion of patient consultation regarding common symptoms of COVID-19 will reduce the large strain that is currently on the healthcare industry. There has become a need for digital solutions that maintain records of COVID-19 cases, patients and test results to inform the response to the virus at scale as well as to respond to case clusters, outbreaks and high-risk areas.

Project Description

Students have been tasked to use their C++ programming skills and knowledge to create a program that allows users to seamlessly input and update their information and keep track of the status of other patients as well as high risk locations so they can avoid them. During previous pandemics this was not as much of a viable option, but with the innovations made in technology it has allowed us to share and gather large amounts of information on a worldwide scale. The project is based around both a demonstration as well as this final report.. In order to properly convince the government this is an effective idea, students need to demonstrate their knowledge and show the process they have gone through in order for them to be confident that the approach they have chosen is the most viable option.

Design Concepts

Problem Statement

The following COVID Test Recommendation can be broken down into 2 main tasks. The first task is based around creating the primary database, the database includes 3 tables, one for the patients personal details, including their patient ID, name, address etc. Another table will be for the symptoms, broken down into 3 categories, low, medium and high. The symptoms need to be realistic, for example low level symptoms would include coughing or drowsiness, while high level symptoms would include difficulty breathing. The final table is the high risk COVID locations. The challenge being when the user enters the data at certain points the program should know what it needs to store, the government doesn't want to have unnecessary information in the database and it shouldn't ask a patient for redundant data, for example if the user doesn't require a COVID test then the program shouldn't ask for their location either.

The second task involves the covid test recommendation as well as creating a menu display. The menu needs to have a user friendly interface, options should be recognizable, the government does not want to create more confusion, the difficult being it should also detect any errors such as any misinputs as well as not being case sensitive. In order to accurately recommend a covid test to a patient the program needs to be well equipped to deal with as many realistic scenarios as possible, for example a patient who hasn't been to a high risk location but has medium level symptoms should isolate at home. It is vital that only patients who need to be tested do it.

Constraints and Specifications

When completing the project there is a set guideline, it needs to be written in C++ and the database needs to be in a text file. The final report must be submitted by the 31st of October to avoid penalties, it must cover specific sections and for 2 member groups have a 2400 word count minimum. Comments must explain all key parts of the code that aren't obvious to a programmer. There should be two block comments, one explaining the equations and test data and another summarizing the inputs, outputs and local variables used. All variables and constant identifiers must have appropriate names. Once the database has been created, students need to input at least 10-15 different sample datasets, when entering data there needs to be at least 5 scenarios on whether to recommend a patient to get a test or not. The program also requires a menu with 6 options that the user can navigate.

Implementation and Processes

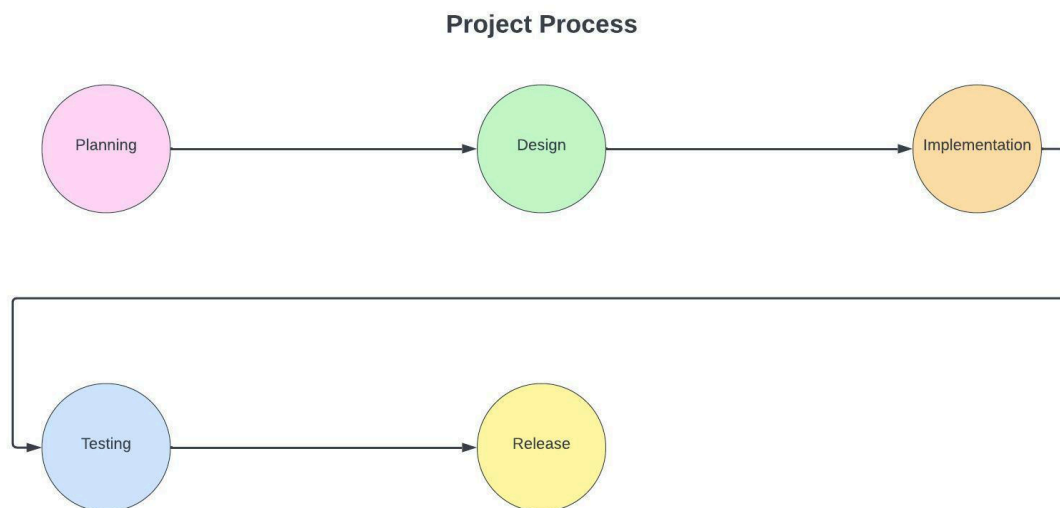


Fig.2: Project Process

The simple process above was followed. The first step was planning, this involved going through all of the available resources and detailing the project's description, objectives, scope and restrictions. Using the information gathered the next step was to design the software solution,

using a flowchart to identify any obvious variables, functions or loops as well as the main structure of the program. The next step was the implementation, this is a process in itself and involved identifying the hardware and software to be used, due to the scope most of this was already stated (C++, text files), however to share code with each other we shared files with each other over media, using both laptops and PC to do the coding. It was important to break down the parts of the code for each team member to do, in general this was creating the database, menu, the recommendation loop, any user inputs and reading and writing into the files. Throughout this process testing was done, trying to find any errors before the last step where it is finally released to the public, which also includes this report and its documentation of the process.

Results / Design Prototype

The first thing done was to create a flowchart of all the basic processes, with the hope that it helps identify the main processes, loops or variables that will be needed. This flowchart for the menu is shown below in Figure 1.

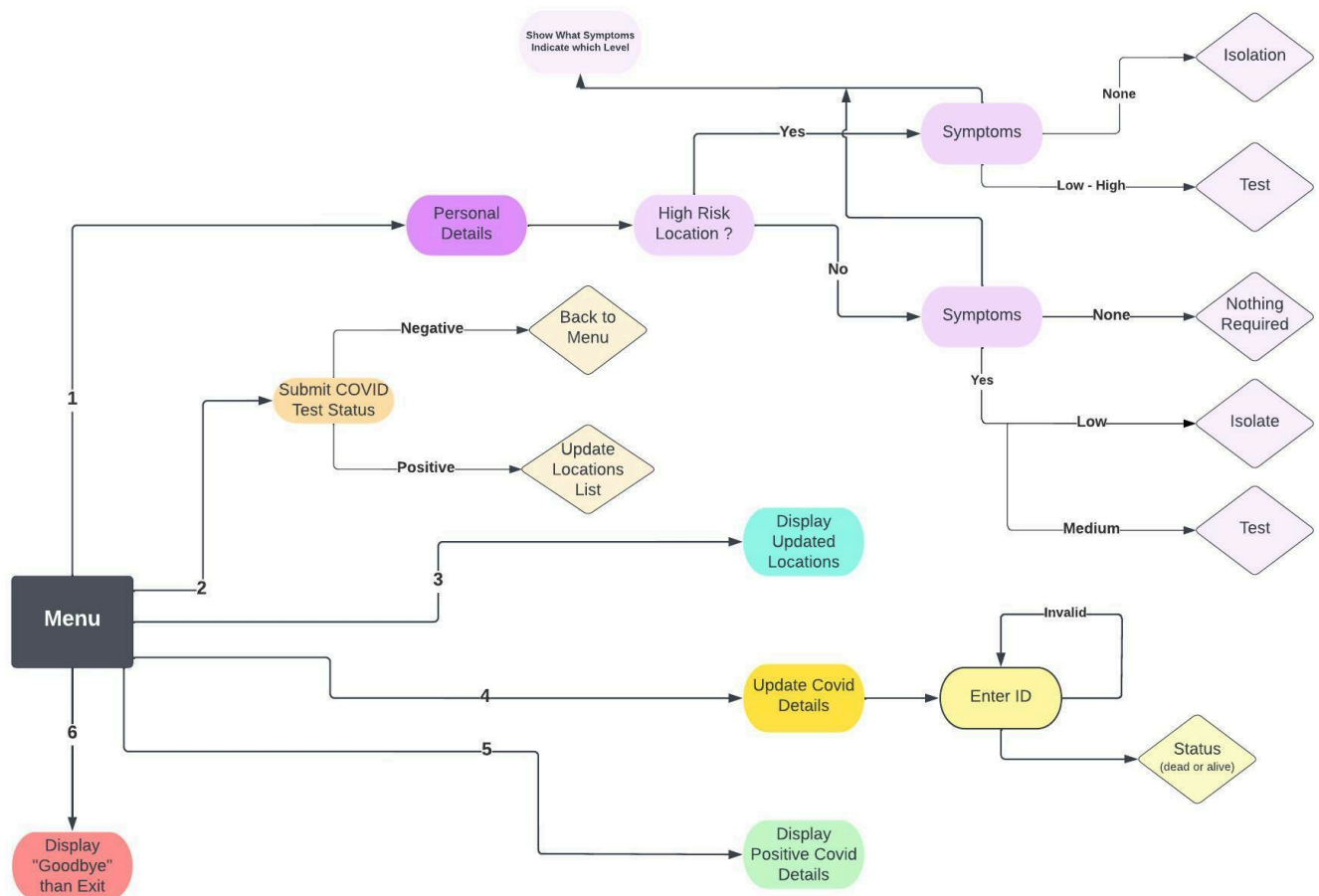
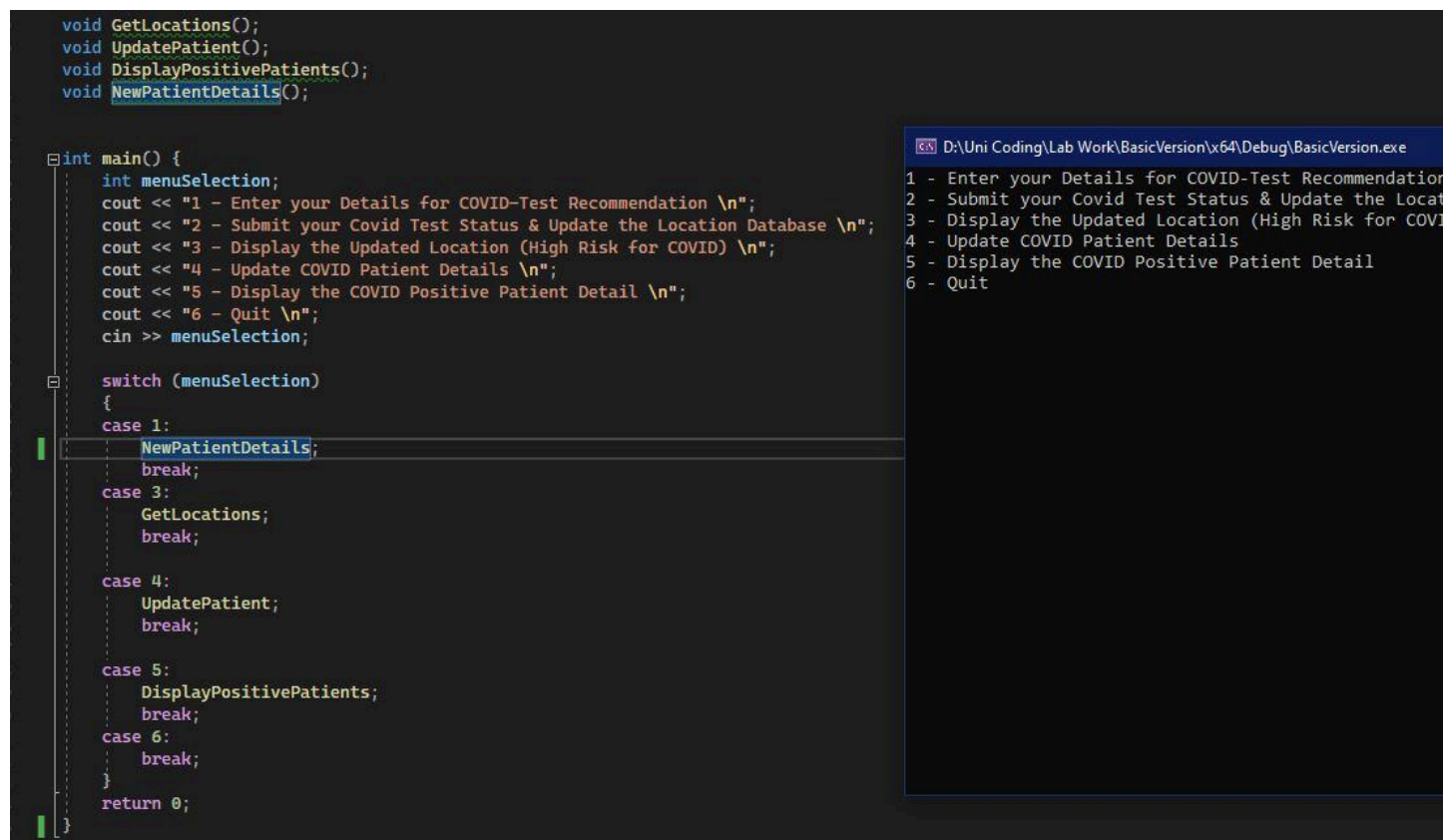


Fig.3: Menu FlowChart

Menu options 3,5 and 6 all require a simple display of certain data from the primary database, the other 3 options require multiple steps. The most important menu option is 1, which is when a patient enters new details and is then given a recommendation. Important to identify what order they should be asked something to avoid redundant information. To create the menu a simple switch case can be used, recalling the prototype functions for each menu option. Below was the first menu ;



```

void GetLocations();
void UpdatePatient();
void DisplayPositivePatients();
void NewPatientDetails();

int main() {
    int menuSelection;
    cout << "1 - Enter your Details for COVID-Test Recommendation \n";
    cout << "2 - Submit your Covid Test Status & Update the Location Database \n";
    cout << "3 - Display the Updated Location (High Risk for COVID) \n";
    cout << "4 - Update COVID Patient Details \n";
    cout << "5 - Display the COVID Positive Patient Detail \n";
    cout << "6 - Quit \n";
    cin >> menuSelection;

    switch (menuSelection)
    {
    case 1:
        NewPatientDetails;
        break;
    case 3:
        GetLocations;
        break;
    case 4:
        UpdatePatient;
        break;
    case 5:
        DisplayPositivePatients;
        break;
    case 6:
        break;
    }
    return 0;
}

```

```

D:\Uni Coding\Lab Work\BasicVersion\x64\Debug\BasicVersion.exe
1 - Enter your Details for COVID-Test Recommendation
2 - Submit your Covid Test Status & Update the Location Database
3 - Display the Updated Location (High Risk for COVID)
4 - Update COVID Patient Details
5 - Display the COVID Positive Patient Detail
6 - Quit

```

Fig.4: Basic Menu Code and Output

The final menu is shown below in the data display. The most challenging part was to create the primary database by reading and writing files, in the figure below is the code for reading and writing the file in order to update a patient's details without the user input yet ;

```

string GetNextCSVValue(string* csvLine) {
    string returnValue = "";
    int commaIndex = csvLine->find(",");
    if (commaIndex == -1) {
        returnValue = csvLine->substr(0, csvLine->length());
        csvLine->erase(0, csvLine->length());
    }
    else {
        returnValue = csvLine->substr(0, commaIndex);
        csvLine->erase(0, commaIndex + 1);
    }
    return returnValue;
}

class PatientDetails {
public:
    void FromCSV(string csv) {
        patientId = stoi(GetNextCSVValue(&csv));
        name = GetNextCSVValue(&csv);
        dob = stoi(GetNextCSVValue(&csv));
        address = GetNextCSVValue(&csv);
        visitedLocation = GetNextCSVValue(&csv);
        dateTime = stoi(GetNextCSVValue(&csv));
        lastOverseasTravel = GetNextCSVValue(&csv);
        covidTest = GetNextCSVValue(&csv);
        status = GetNextCSVValue(&csv);
    }

    string AsCSVString() {
        return to_string(patientId) + "," + name + "," + to_string(dob) + "," + address + "," + visitedLocation + "," +
            to_string(dateTime) + "," +
            lastOverseasTravel + "," + covidTest + "," + status;
    }

    int patientId;
    string name;
    int dob;
    string address;
    string visitedLocation;
};

```

Fig.5: Reading and Writing Files, for Menu Option 2

The final step was the user input, below is menu option 1 however there is not a symptom database as of yet,


```

1- Enter your detail for COVID-Test Recommendation
2- Submit Your Covid test status & Update the Location database
3- Display the Updated Location (High Risk for COVID)
4- Update COVID Patient Details
5- Display the COVID Positive Patient Detail
6- Quit
Please enter a value between 1 and 6:
1
Please enter your name:
Joshua Lillington
Please enter your date of birth:
20/12/2002
Please enter your address:
Lilydale
Have you traveled overseas in the past 2 weeks:
No
No locations found
Your patient id is: 2
Symptom database not found.
Press any key to continue . . .

```

Fig.6: Display of Menu Option 1

Then the patient could select option 5 and choose their ID and it would display their details, this is shown below.

```

Please enter a value between 1 and 6:
5
Enter the Id of the patient you wish to view:
2
Patient Details:
Id: 2
Name: Joshua Lillington
Date of Birth: 20/12/2002
Address: Lilydale
Recent Overseas Travel: No
Press any key to continue . . .

```

Fig.7: Display of Menu Option 5

In the final design the symptom table is implemented and it will recommend what the patient should do depending on their answers. This is shown below ;

```

Please enter a value between 1 and 6:
1
Please enter your name:
Joshua
Please enter your date of birth:
20/12/2002
Please enter your address:
Lilydale
Have you traveled overseas in the past 2 weeks:
No

Please enter the number of the location you have visited recently:
1- Swinburne University
2- Glenferrie Train Station
3- MCG
4- None of the above
Please enter a value between 1 and 4:
1
Please enter when you visited:
26th November
Your patient id is: 2
What are your current symptoms? Enter 'Done' to finish entering symptoms:
Coughing, Sore Throat
Done
Please isolate at home, if you have any symptoms, please immediately test for COVID.
Press any key to continue . . .

```

Fig.8 : Display of COVID Test Recommendation

Project Outcomes

When considering the final design it's important to look at the project's compatibility and how it aligns with set objectives and outcomes. As will be shown below it fulfills all of these, it has a total of 5 different realistic covid situation recommendations and is able to save over 10 datasets in the primary database text file. Overall the final program both efficiently and effectively fulfills the objectives, it has an easy to use menu, it gives realistic advice and saves only the necessary patient data, all 6 menu options work as they should, and there is also error handling where required so patients don't have to worry if they misinput their data.

Data Display

To begin with the patient is presented with a menu, if they select option 1, it will ask for their user details. The patient details will be saved to a csv file with one row per patient. The code will ask the patient different questions depending on their previous inputs. If a patient indicates that they have been to a high-risk location, they are asked when they did. The patient is also asked to input the symptoms they are experiencing, in order to determine the necessity for a COVID test. The necessity for a test is determined upon whether the patient has been to a high risk location and their most severe level of symptoms.

The inputs and responses are as follows:

Visited a high risk location	Symptom Level	Output
Yes	None	Please isolate at home, if you have any symptoms, please immediately test for COVID.
Yes	Low-Severe	Please immediately isolate and test for COVID as soon as possible.
No	None	We do not recommend you isolate at the moment.
No	Low-Medium	We do not recommend you get tested, however please isolate at home and get tested if your symptoms worsen.
No	Severe	Please immediately isolate and test for COVID as soon as possible.

```

Please enter your name:
Joe Bloggs
Please enter your date of birth:
10/3/1985
Please enter your address:
8 Sundown Street
Have you traveled overseas in the past 2 weeks:
No

Please enter the number of the location you have visited recently:
1- Glenferrie Station
2- MCG
3- Swinburne University
4- Melbourne Central
5- None of the above
Please enter a value between 1 and 5:
2
Please enter when you visited:
19/4/2022
Your patient id is: 11
What are your current symptoms? Enter 'Done' to finish entering symptoms:
Dry Cough
Headache
Done
Please immediately isolate and test for COVID as soon as possible.
Press any key to continue . . .

```

Fig.8: Menu Option 1

The patient data is saved with the following format in a CSV file:

Id,Name,Date of Birth,Address,Location Visited,Time of Visit,Traveled Overseas
Recently,COVID Test Result

The test data set is as follows:

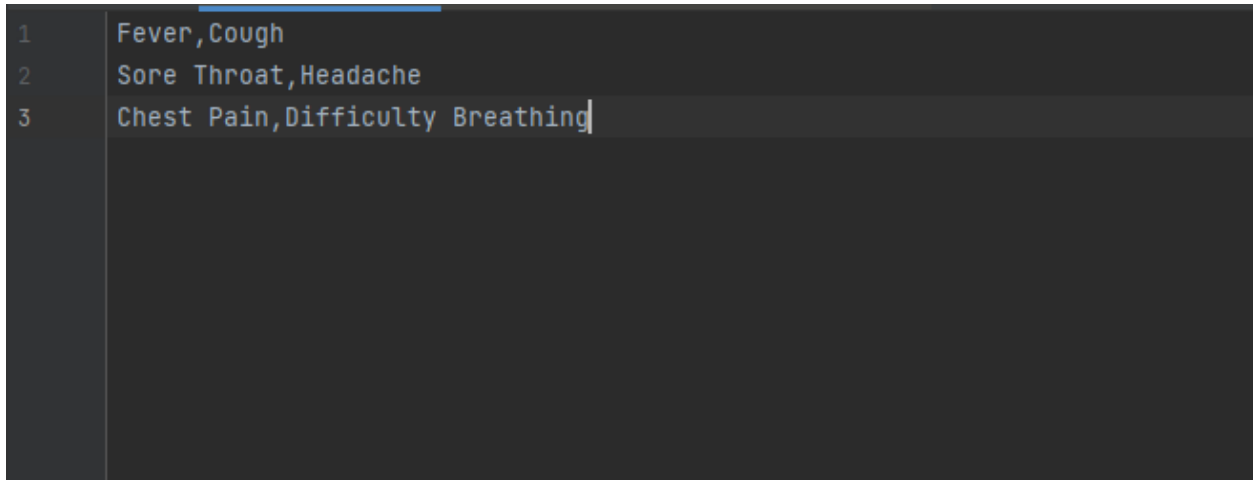
```

0,Eamon Heffernan,09/09/2003,482 Paradise Lane,Swinsburne University,20/03/2022,No,Positive,
1,Freyja Gilbert,13/10/1966,3078 Stratford Park,,,No,,
2,Avani Sparks,18/12/1956,2434 Don Jackson Lane,Glenferrie Station,04/08/2022,No,,
3,Arjan Vu,18/12/1956,4441 Gateway Road,,,No,Negative,
4,Millicent Madden,03/05/1988,4085 Camden Place,,,No,Positive,
5,Shah Sims,02/03/1995,231 Hickman Street,,,No,Positive,
6,Asad Clay,03/09/1984,1995 Drummond Street,MCG,20/10/2022,No,Negative,
7,Amir Rangel,19/09/2004,2747 Bloomfield Way,,,No,,
8,Harvie Kumar,18/11/1950,3218 Rodney Street,,,No,Negative,
9,Romario Finley,28/06/1980,3025 Spirit Drive,,,No,,
11,Joe Bloggs,10/3/1985,8 Sundown Street,MCG,19/4/2022,No,,

```

Fig.9: Patient Data Saved in CSV File

The symptoms are stored in a csv file with each row being dedicated to a new tier of symptoms, with each symptom separated by commas.

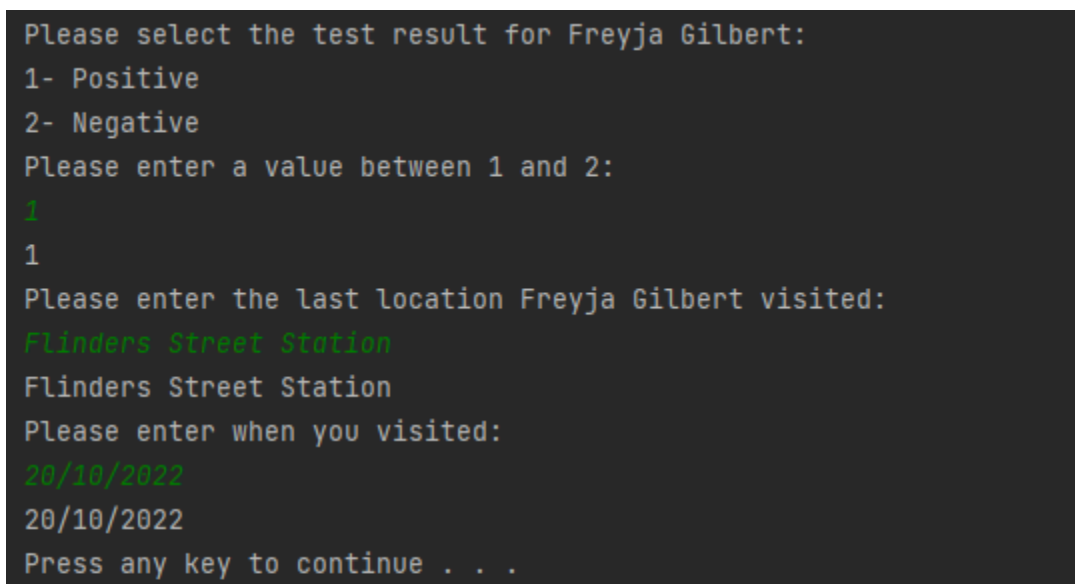


A screenshot of a CSV file with a dark background. It shows three rows of data. The first row contains 'Fever,Cough'. The second row contains 'Sore Throat,Headache'. The third row contains 'Chest Pain,Difficulty Breathing'. Each row is numbered on the left side of the image: 1, 2, and 3 respectively.

1	Fever,Cough
2	Sore Throat,Headache
3	Chest Pain,Difficulty Breathing

Fig.10: Symptom File

The second menu option allows the patient to input the results of a COVID test they have taken. If the test was positive, they are asked for a location they have visited, as well as when they visited which will be saved in the high risk locations store. The result of the test will be saved into the existing patient's data within the user database.



A screenshot of a terminal window with a dark background. It shows a menu for Freyja Gilbert. The prompt is 'Please select the test result for Freyja Gilbert:'. The options are '1- Positive' and '2- Negative'. The user has entered '1'. The prompt is 'Please enter a value between 1 and 2:'. The user has entered '1'. The prompt is 'Please enter the last location Freyja Gilbert visited:'. The user has entered 'Flinders Street Station'. The prompt is 'Please enter when you visited:'. The user has entered '20/10/2022'. The prompt is 'Press any key to continue . . .'. The user has entered '20/10/2022'.

```
Please select the test result for Freyja Gilbert:
1- Positive
2- Negative
Please enter a value between 1 and 2:
1
Please enter the last location Freyja Gilbert visited:
Flinders Street Station
Flinders Street Station
Please enter when you visited:
20/10/2022
20/10/2022
Press any key to continue . . .
```

Fig.11: Menu Option 2

The high risk locations are stored in a text file with one location per line.

```
1  Glenferrie Station
2  MCG
3  Swinburne University
4  Melbourne Central
5  |
```

Fig.12: High Risk Location Text File

The third menu option will display the current high risk locations to the patient in a list from the database file.

```
The current High Risk Locations for COVID are:
Glenferrie Station
MCG
Swinburne University
Melbourne Central
Flinders Street Station
Press any key to continue . . .|
```

Fig.13: Menu Option 3

The fourth menu option will allow updating a patient's information.

The patient is prompted to input their patient id, once matched they are prompted to choose the field they want to update. The user is then prompted for the value they wish to set it to. Which is then saved to the database.

```
Enter the Id of the patient you wish to update:
4
4
Please select The field you wish to update:
1- Patient Id
2- Name
3- Date of Birth
4- Address
5- Visited Location
6- Time of Visit
7- Last Overseas Travel
8- Status
9- Quit
Please enter a value between 1 and 9:
2
2
Please enter the new value:|
Joe Bloggs
Joe Bloggs
Updated.
```

Fig.14: Menu Option 4

The fifth menu option will display an output of all the patients and their details. It reads all the users from the database and formats it into readable text, and outputs it to the terminal.

```
Patient Details:
Id: 0
Name: Eamon Heffernan
Date of Birth: 09/09/2003
Address: 482 Paradise Lane
Visited Location: Swinburne University at 20/03/2022.
Recent Overseas Travel: No
Covid Test Result: Positive
```

Patient Details:

Id: 1

Name: Freyja Gilbert

Date of Birth: 13/10/1966

Address: 3078 Stratford Park

Visited Location: Flinders Street Station at 20/10/2022.

Recent Overseas Travel: No

Covid Test Result: Positive

Patient Details:

Id: 2

Name: Avani Sparks

Date of Birth: 18/12/1956

Address: 2434 Don Jackson Lane

Visited Location: Glenferrie Station at 04/08/2022.

Recent Overseas Travel: No

Patient Details:

Id: 3

Name: Arjan Vu

Date of Birth: 18/12/1956

Address: 4441 Gateway Road

Recent Overseas Travel: No

Covid Test Result: Negative

Patient Details:

Id: 4

Name: Joe Bloggs

Date of Birth: 03/05/1988

Address: 4085 Camden Place

Recent Overseas Travel: No

Covid Test Result: Positive

Patient Details:

Id: 5

Name: Shah Sims

Date of Birth: 02/03/1995

Address: 231 Hickman Street

Recent Overseas Travel: No

Covid Test Result: Positive

Patient Details:

Id: 6

Name: Asad Clay

Date of Birth: 03/09/1984

Address: 1995 Drummond Street

Visited Location: MCG at 20/10/2022.

Recent Overseas Travel: No

Covid Test Result: Negative

Patient Details:

Id: 7

Name: Amir Rangel

Date of Birth: 19/09/2004

Address: 2747 Bloomfield Way

Recent Overseas Travel: No

Patient Details:

Id: 8

Name: Harvie Kumar

Date of Birth: 18/11/1950

Address: 3218 Rodney Street

Recent Overseas Travel: No

Covid Test Result: Negative

Patient Details:

Id: 9

Name: Romario Finley

Date of Birth: 28/06/1980

Address: 3025 Spirit Drive

Recent Overseas Travel: No

Patient Details:

Id: 11

Name: Joe Bloggs

Date of Birth: 10/3/1985

Address: 8 Sundown Street

Visited Location: MCG at 19/4/2022.

Recent Overseas Travel: No

The final menu option prints out goodbye and exits the program.

Code Explanation

The program starts with the main function which includes an infinite loop that will output the menu options repeatedly, as to display the menu each menu interaction. The user's selection uses the `GetNumericInput` function which will repeatedly attempt to prompt the user to enter a value, check if that value can be assigned to an `int` type until a valid input is entered. This value is returned to the main function and used in a switch case to determine the function to run.

If the enter patient details menu option is selected, the code clears the "cin" to avoid unwanted inputs from being saved, a patient object is then created. The patient is prompted for their basic details with a series of prompts using "cout" and taking the input via `getline` as to get values including spaces, which is then saved to the patient details object.

The list of high risk locations is then read out from the text file. A vector is used to store the locations as the size of vectors can be modified during runtime. The file is opened and then each line is looped through, adding each to the location vector before returning the vector.

There is a special case for 0 locations as it is not worth prompting the patient for locations they have attended if there are no high risk locations. If there are locations, they are printed out one by one with a number next to them. The user is then asked to select if they have been to any of the locations, or none of the above. The user's input is processed using the same `GetNumericUserInput` as the main menu. If the user selected another value, they are then prompted to input the time and date they visited the location, the location and time are then saved to the patient object.

All existing patient details are then retrieved from the local `patientsDetails.csv` file. This is done via a loop that goes through each line of the file, each representing a patient, into a function that creates a patient object from each line of csv. This is done by reading the line, finding the next instance of a comma and removing the text before it, repeating it until there are no more commas. These parts of the line can then be assigned to the variables in the patient class. All the existing patients are then returned to the function that is taking the patient details.

Each patient is then iterated over to find the highest existing patient id. Once found, the new patient's id is set to one higher than the existing highest patient id.

The symptoms are then loaded from their csv file using the same function as was used while loading the patients. In the symptoms file, each level of symptom is given their own line, and each line contains multiple symptoms. The symptoms are read into a pre-initialized 2 dimensional vector which is returned.

The patient is asked to list their symptoms, if any of their entered symptoms match the symptoms in the vectors, the maximum level is incremented to the highest level of symptoms found, as only the most severe level of symptoms is of note. This was optimized by only checking the levels of the symptoms higher than the current level for matches, as any others are unnecessary. The combination of symptom level and whether the patient has visited a high risk location is then used in if statements to determine the output to “cout”.

The patient data is then saved by rewriting the entire patient file, using a function on each patient that takes each value in the patient classes and adds them to a string, separated by commas.

The second menu option prompts the user to enter the id of a user. A numeric input is then taken, and all the patient details are searched for a matching id using the GetPatientFromId function, which loops over all the patient details until one with a matching id is found, returning that object. If a patient isn't found, the user is given an error message and returned to the main menu.

If the patient is found, they are prompted to enter their covid result using another numerical input. This result is saved to their patient details object. If they are positive, they are asked to input the location they have been to recently, and when they went. This information is saved to their patient details. The location vector is then looped over to find if it already contains the location, if it doesn't the new location is added. Both the locations and all the patient details are then saved to their files. The locations are saved by opening the file in write mode, and outputting one location per line.

The third menu option simply gets all the locations, iterates over them and outputs them to “cout”.

The fourth menu option will prompt the user for a patient id using the same code as the COVID test result submission, and then use a function that will output all the available fields that can be updated. The patient is then prompted to select one with a numerical user input. One one is selected, they are prompted to enter a new value. The result of the field selection is then used in a switch case to determine which variable to save the input under. All the patients are then written to the file.

The fifth menu option displays all the patients and their details. This is done by getting the patient details from the file and then iterating over them and running the function in their class that outputs their information to the terminal. This function consists of many outputs to cout, with prefixes explaining what the field is and the value. The user is then returned to the main menu.

The sixth menu item simply outputs “Goodbye” and exits the program.

Individual Contribution #1 (Josh)

Reflection

Overall I feel like I have contributed a good amount to this project, I have been heavily involved with writing the reports and the presentation and have done as much as I am capable of with my knowledge of programming. My partner is much more skilled than me, however I have taken the time to learn the code, writing lots of notes, on things such as any new techniques used, as well as writing my own code that does similar things however in a less efficient and effective way thus it has not been used. I have learnt that although I am good at deciphering code and breaking down the project to design it, I have major issues translating it into an actual working program, especially trying to get all of the code structured correctly, as I often get stuck on errors and can get lost easily. I definitely need to continue practicing my coding, a different approach to learning / thinking is most likely required. This was my 2nd unit of coding and my first time learning C++.

Learning Outcomes

1. Describe and apply the basics of programming for an engineering application

When creating the COVID-19 Test Recommendation program it was required to apply the basics of programming for this to work, examples include the use of files to create the database, and loops which are used throughout the entire program, and include if else, if, for etc, the program would not be viable without the use of them, for example the test recommendation had required 5 different scenarios, so using a loop or switch case is obviously the best choice.

Throughout this final report and the comments included in the code or individual notes it shows an ability to describe the basics of programming, which for this project is directly applicable to engineering. This meant having the ability to take the problem of creating a software solution to aid with the COVID-19 pandemic and adapting it and breaking it down to fulfill the objectives to help both the government and society as a whole.

2. Use a programming language, and associated class libraries, to develop and implement a technical software application

This project report and the code with it speaks for itself, this whole project requires the ability to use C++ which is a programming language, and using the different class libraries to access new functions etc. For example `<algorithm>` allows access to new functions such as count, and `<fstream>` allows for reading and writing files. By applying the knowledge taught in class it allowed us to look at the task of a COVID Test Recommendation and develop a technical software application that fulfills the criteria.

3. Design and implement a sustainable code base for scientific and engineering problems

By using code comment, especially things that non programmers wouldn't understand it means if anyone else would like to adapt the code it makes it a lot easier.

Something vital for this program was to only ask for necessary information, for example, if someone doesn't have covid, it shouldn't ask for their location as it is not needed, by only taking in vital data it makes for a code base that is more sustainable as it doesn't use up unnecessary storage.

4. Design, implement, evaluate, and apply unit testing and documentation strategies in developing technical software

When analyzing the code, I went through each block of code with the intent of understanding what each line achieved. Focusing on the variables and functions and their specific use in each instance.

First thing I did was to create the flowchart, it allowed me to understand the basic processes of the code, breaking down the large task into its different sections. I wrote down anything I found useful, or ideas that I had about the code to use later in development.

5. Demonstrate teamwork skills to produce high-quality project reports and deliver professional presentations

For the project report as well as the brief it was important to work together in giving each other tasks to do, reading over each other's work to make sure the report flowed well. Making sure to ask questions if help was needed especially about the structure of the project, to make sure no criteria was missing.

When working towards the presentation, me and my partner stayed in contact with each other, making sure we understood our different parts, focusing on each other's strengths which allowed us to do a professional presentation as best as we could.

Technical Learning

I was not involved with most of the programming of the final design, I took the time however to analyze the written code, going over each block and understanding what everything did and how it connected, learning new techniques or shortcuts for the code such as removing trailing lines and also classes, before we eventually covered it in actual class. Above is the code explanation.

Individual Contribution #2 (Eamon)

Reflection

During the course of this project I believe that I have contributed significantly to our team's output during the duration of this project. I took the lead during the programming tasks to develop a majority of the code. I learned a large amount about C++ and its functions and libraries throughout development and used them to build a functioning application with proper saving and reading to files. I believe that I could have done more during writing reports, however I believe that overall we shared the workload reasonably evenly. Going into this project I had significant experience with programming, and this allowed me to take the lead during the programming sections of the task.

Learning Outcomes

1. Describe and apply the basics of programming for an engineering application

During this project I took the lead to develop the program. I used programming principles such as selection to determine which menu item that the user selected, as well as whether they should be displayed certain prompts such as only asking the patient what time they visited a location if they indicated that they did visit a location.

I also used classes and file i/o to make an easy way to interface with the patient information within the code as well as full functionality with saving and loading patient details from local files.

2. Use a programming language, and associated class libraries, to develop and implement a technical software application

As I developed most of the project, I used C++ to create much of the program. I used basic C++ features such as selection and iteration as well as more complex features such as pointers and libraries to enhance the program. I used class libraries to interface with files for saving and loading information, use and modify strings and use and implement vectors to hold data. The program is complete and without major issues, and fulfills the requirements laid out in the design brief.

3. Design and implement a sustainable code base for scientific and engineering problems

To make the code base easy to use and build upon, it contains comments and documentation that should make improving on the code significantly easier. The code should also be easy to follow and follow the expected C++ styling guide with correct casing, indentation and whitespace, as well as well named functions and variables.

4. Design, implement, evaluate, and apply unit testing and documentation strategies in developing technical software

To test the code and program, I used manual unit testing to determine that individual functions were producing the correct outcome. This was conducted on the original development of a function, as well as after changes. I also implemented end to end testing to ensure that given the correct inputs the program would output the correct results. To document the code, I added a header explaining the input and outputs of the program as a whole as well as comments explaining each function and their parameters and return values. I also added inline comments to explain the more difficult to understand lines of code where needed.

5. Demonstrate teamwork skills to produce high-quality project reports and deliver professional presentations

Throughout this project, me and my partner have worked well to share the workload, working to our advantages. We have not had any disagreements over workload or the direction of the project, and have both been willing and happy to work to our strengths and do more of different parts of the project to most efficiently deliver our project reports and presentation.

Technical Learning

This project has allowed me to learn C++ and in turn to learn more about lower level computing information such as the stack and heap, memory management and manual garbage collection. Since I already had significant experience with programming, this project did not teach me a lot in terms of basic programming skills, however, I was enthusiastic to learn about the depth of control that C++ gives you over its features. I also learned byte array management, pointers, as well as using the class libraries that C++

Conclusion

When considering the project objectives and scope it is evident that the final project's design fulfills this. Scope wise, the program is written in C++, the database is in a text file, all of the required comments have been written etc. Objectively it has fulfilled its need as an effortless user experience with realistic patient recommendations and a database that could be easily used by the government to assist with contact tracing or trends. There are obviously ways for this project to be upgraded and changed to make it a more user positive experience but for the requirements given this software solution is more than satisfactory.

Future Recommendations

Despite fulfilling the COVID Test Recommendation requirements the program can still be heavily upgraded, especially on the users side. This includes adding an option menus, such as a way to go back to the menu or exit the program no matter where they are in the program, an option to look at contact information would also be good such as if emergency help is required. There are also more specific things such as options for users to change settings such as having multiple languages, accessibility options for impaired people, or the ability to adjust the

interface. An upgraded interface would be required for real use, such as making it usable on phones and compatible with touch screens. Another future recommendations would be a way to have updated graphs for cases or deaths which could be adjusted by the user, such as deaths or cases per day, it would allow the public to more easily view information on covid such as when it is spiking or even how many cases are in specific regions or areas.

Appendix :

Code File

https://drive.google.com/file/d/1lpWSiReTCiHm6s7bTdSjkJ35k0h09VZw/view?usp=share_link

Patient Details File

https://drive.google.com/file/d/1ikmm7ZFynwjcYqs0TKiQGVKuQYHVCm_V/view?usp=share_link

Locations File

https://drive.google.com/file/d/1K6W7i3yrseuOl7aXjymXW7TTDcFZEEwd/view?usp=share_link

Symptoms File

https://drive.google.com/file/d/1IamOXU7tZwyLdZImfyLYJ6ozZEEQHR8i/view?usp=share_link

Team Work Breakdown

Task (/5)	Eamon	Josh
Code	5	3.5
Report	4	5

Word Count ; 4390

References

Sharpe, P.D.M. (2022) *Covid-19 inner ear infections affecting balance and hearing, Dizziness & Balance Disorders Centre*. Available at:

<https://dizzinessbalancedisorders.com.au/covid-19-inner-ear-infections-affecting-balance-and-hearing/> (Accessed: October 31, 2022).