2018

PROGRAMMING DISTRIBUTED COMPONENTS REPORT

000874782

USMAN BASHARAT

Table of Contents

Chapter 1: Introduction

This report discusses the design documentation, process and evaluation of Journey Planning System for Live Journeys. The application will be consisted of desktop application and the website. The structure of the report will be as following:

Chapter 2 will contain all the design documentation such as some UML Designs alongside Entity Relationship Diagram.

Chapter 3 will contain the screenshot of the finished desktop application and website. This will demonstrate how both applications work.

Chapter 4 is a critical evaluation of any issues that I came across with the finished product. This will also include any improvements that can be made with the current system. In addition, any issues that were faced during the implementation of the product.

Chapter 5 will lastly include how the algorithms behind each of the finished products for the website and desktop application work. This will also include any pseudo code to explain the algorithm alongside any screenshots to prove it.

Chapter 6 shows the changes of what is added after the demonstration.

Chapter 2: Design Documentation

Entity Relationship Diagram

Figure 1 shows the Entity Relationship Diagram (ERD) of the Live Journeys database. This denotes the tables, columns, keys, data type alongside the relationship they have with each other.

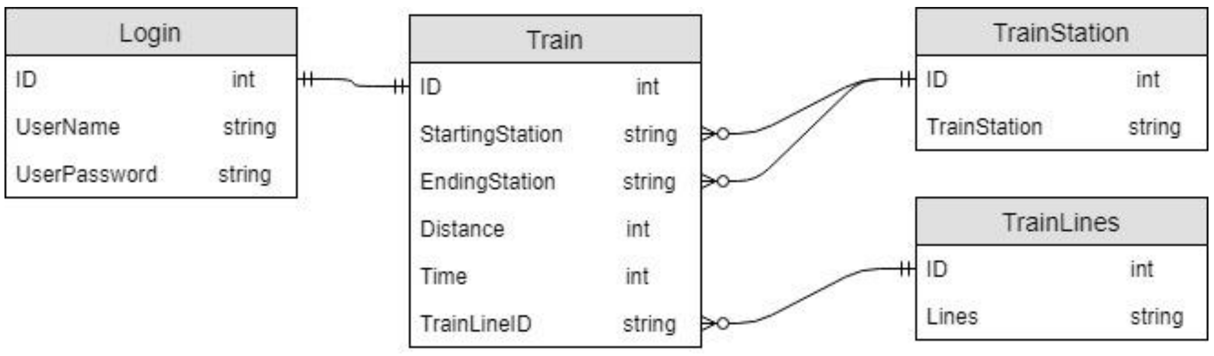


Figure 1 shows the Entity Relationship Diagram for Live Journeys.

UML Class Diagram

Figure 2 and Figure 3 shows the different structure for the class diagram for Live Journeys. The diagram shows the classes, relationship, variable and methods used within the diagrams shown below.

i. Windows Form Application

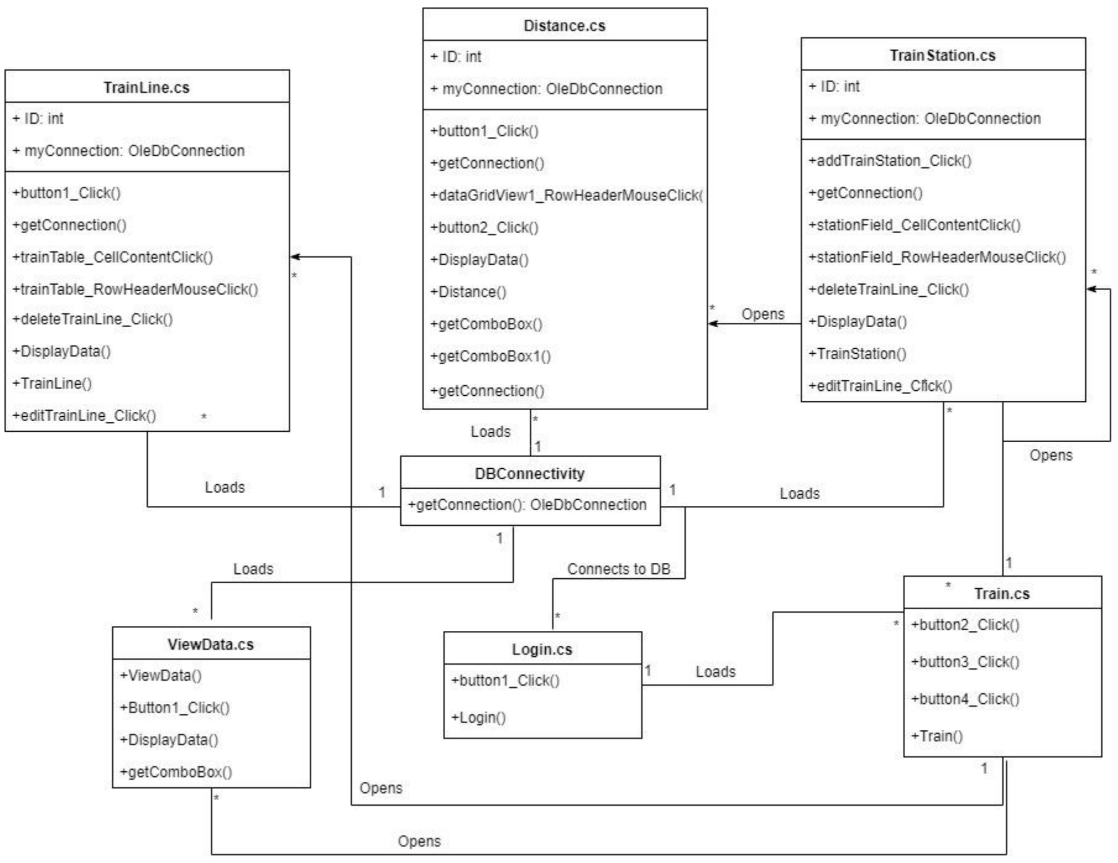


Figure 2 shows the class diagram for Live Journey

ii. SOAP Website

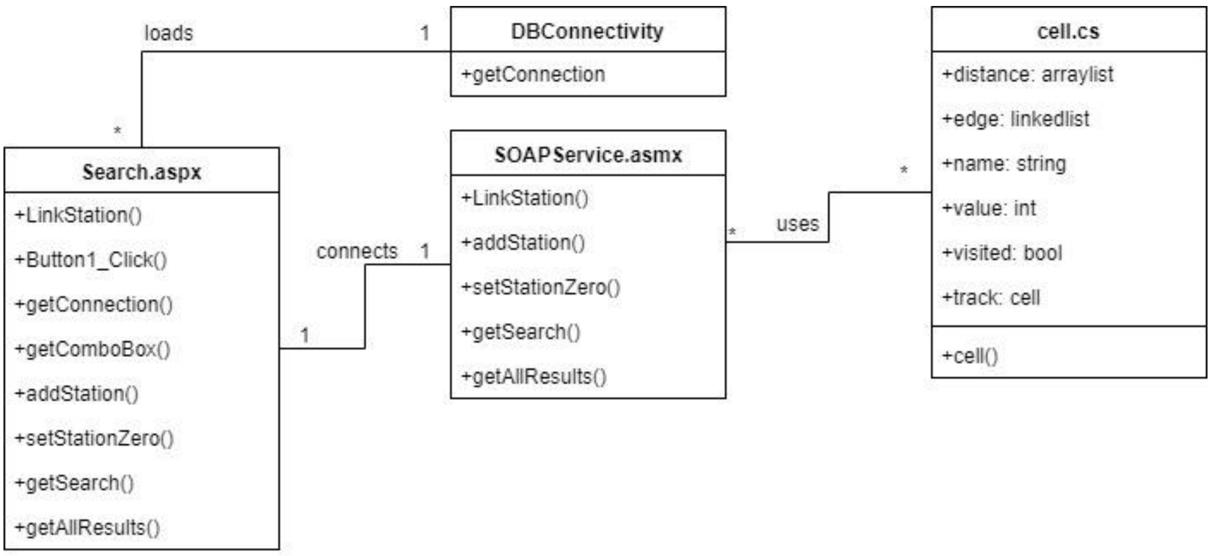


Figure 3 shows the class diagram for SOAP Website

UML Use Case

Figure 4 shows all the actors within the system, what they do within the system and the overall concepts using the use case to complete it.

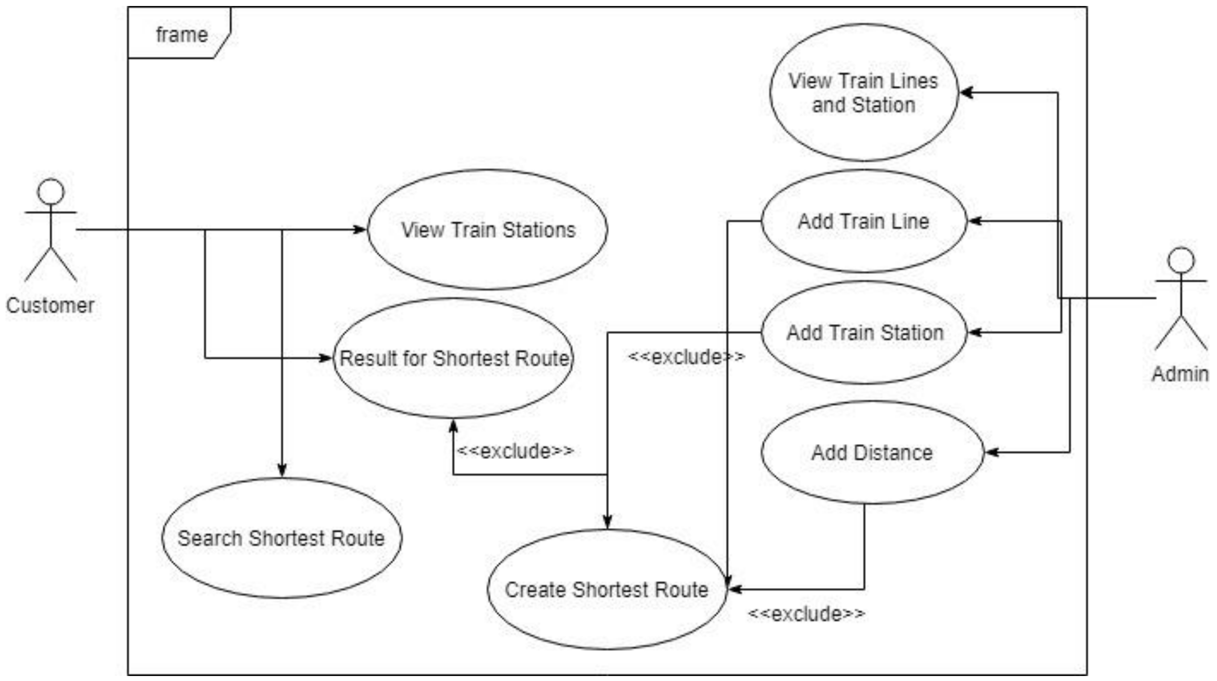


Figure 4 shows the Use Case for Live Journeys

Chapter 3: Screenshot of Features

a. Windows Form Application

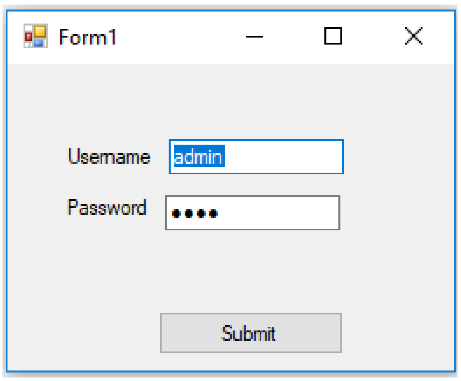


Figure 5 shows the login page.

Referring to Figure 5, this is the login for the desktop application where the administrator can login and able to enter their details. These details are connected to the database server. If the username and password are entered wrong, validation is set out for the correct username and password. Referring to Figure 6, once the user has entered the correct details; the user can have a choice between what the user would like to do.

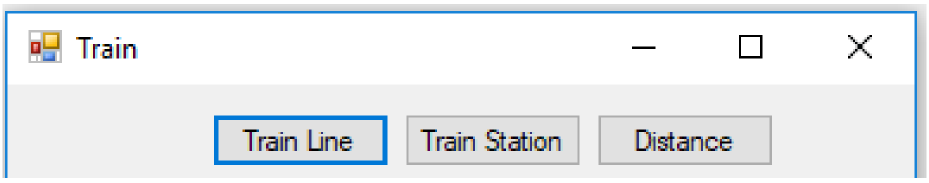
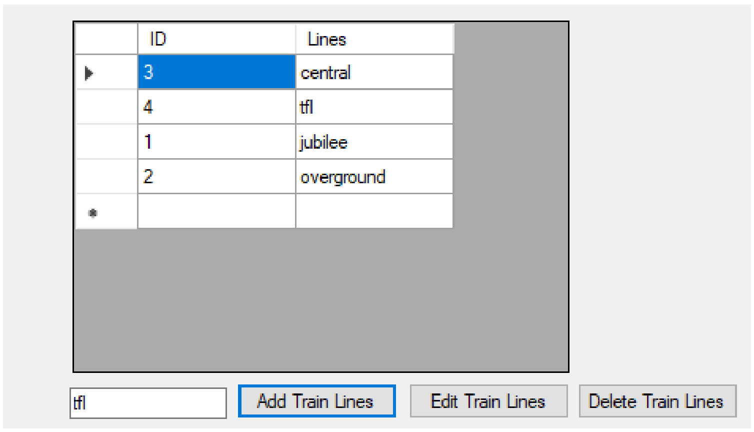
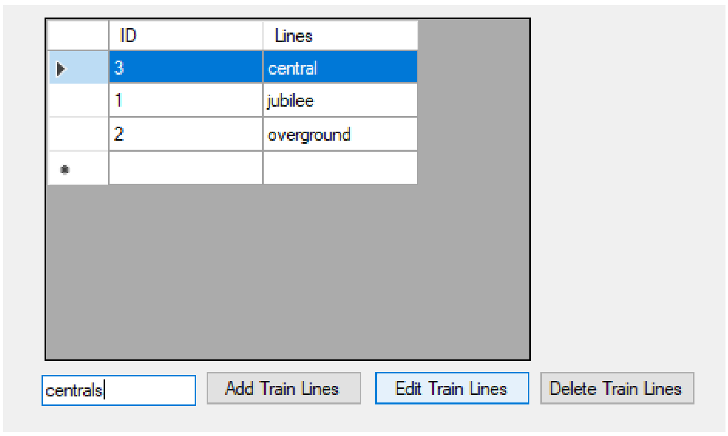


Figure 6 shows the choices for the administration.



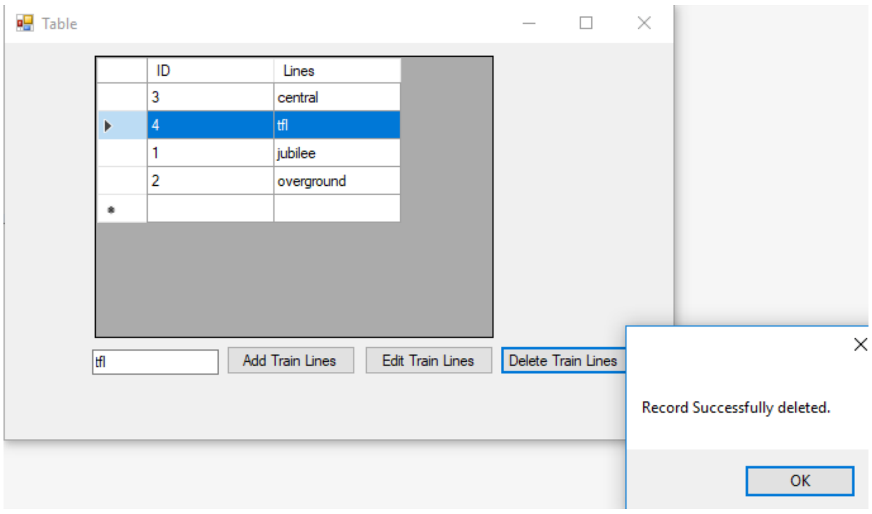
ID	Lines
3	central
4	tfl
1	jubilee
2	overground



ID	Lines
3	central
1	jubilee
2	overground

Figure 7 shows the options of add, edit and delete train line

Referring to Figure 7, this shows adding a new train line as an administrator on the desktop application. The process of adding a new train line is to type in the new train line and click the add button. This would be successful and be completely added to the system. To edit the train line, you click the line that you want to edit. Once you clicked the train line, you can edit it and complete it by clicking the edit button.



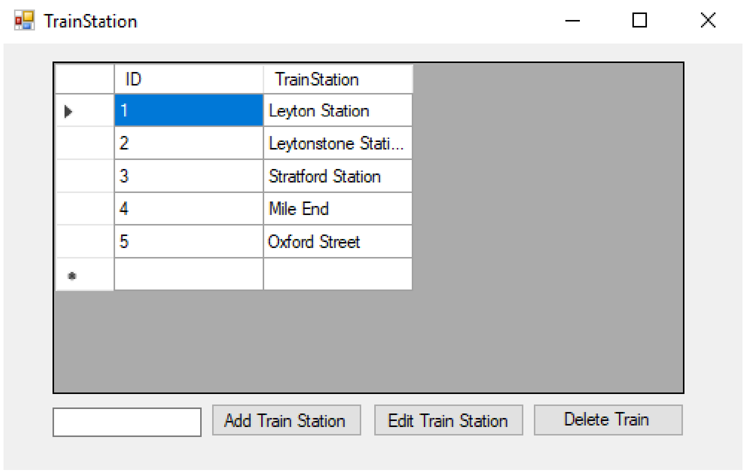
	ID	Lines
	3	central
▶	4	tfl
	1	jubilee
	2	overground
*		

tfl

Record Successfully deleted.

Figure 8 shows the process of deleting a train line.

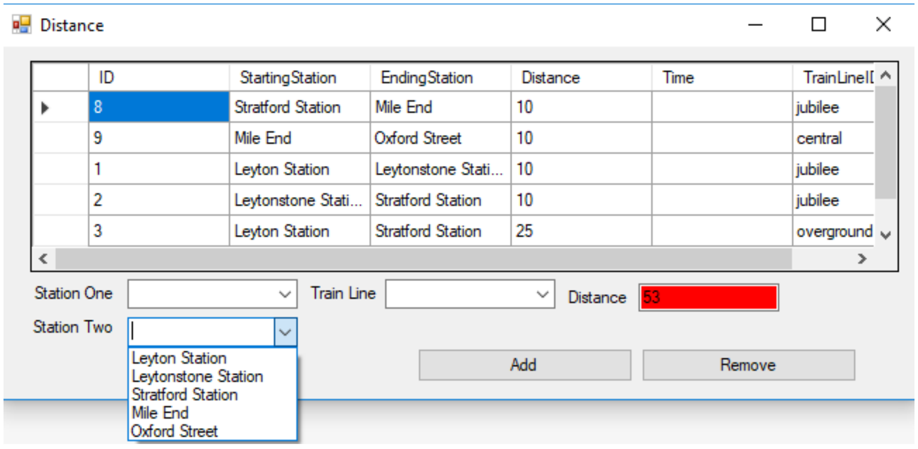
Referring to Figure 8, this shows the process of deleting a train line. This is the same process of editing by clicking the one you want to delete and click the delete button.



	ID	TrainStation
▶	1	Leyton Station
	2	Leytonstone Stati...
	3	Stratford Station
	4	Mile End
	5	Oxford Street
*		

Referring to Figure *, this shows the same options as train line. Please refer over to Figure 7 and 8 as this works the same as train station.

Figure 9 shows the options of add, edit and delete for train station



	ID	StartingStation	EndingStation	Distance	Time	TrainLineI[
▶	8	Stratford Station	Mile End	10		jubilee
	9	Mile End	Oxford Street	10		central
	1	Leyton Station	Leytonstone Stati...	10		jubilee
	2	Leytonstone Stati...	Stratford Station	10		jubilee
	3	Leyton Station	Stratford Station	25		overground

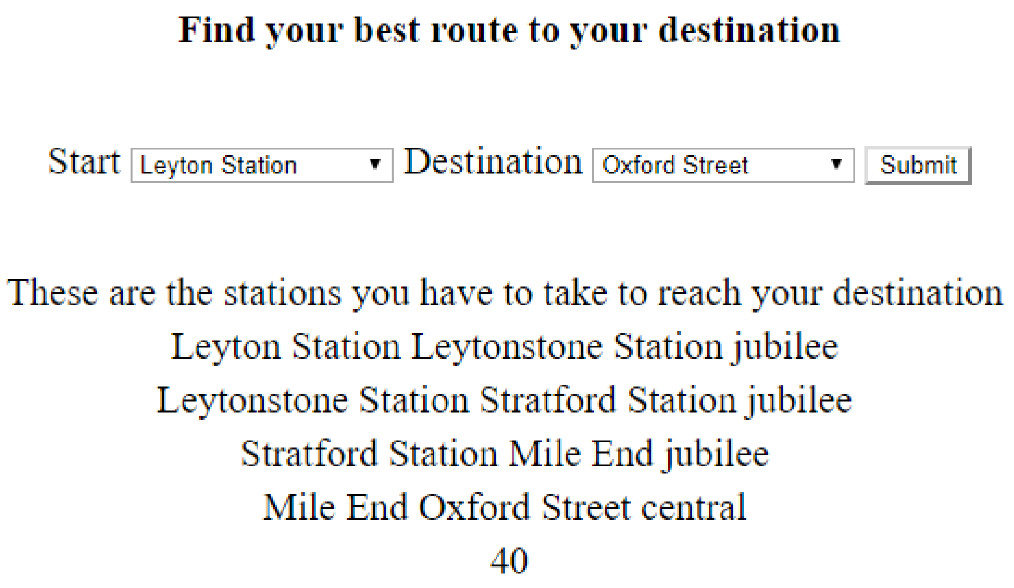
Station One
Train Line
Distance 53
Station Two

Leyton Station
Leytonstone Station
Stratford Station
Mile End
Oxford Street

Figure 10 shows the admin can set a distance between two stations.

Referring to Figure 10, this shows the distance between two stations that the admin can be able to add. The stations are viewed from the stations shown in Figure 9. Once the user has selected both stations, they can select the train line alongside its distance. The distance component is also added to this by adding a c-sharp class to it. This class enables the JTextField to be able to go red once the distance is above 50. The c-sharp class enables this to go red by inheriting.

b. SOAP Website


Find your best route to your destination

Start Leyton Station Destination Oxford Street Submit

These are the stations you have to take to reach your destination
Leyton Station Leytonstone Station jubilee
Leytonstone Station Stratford Station jubilee
Stratford Station Mile End jubilee
Mile End Oxford Street central
40

Figure 11 shows the intermediate level to calculate the shortest route.

Referring to Figure 11, this is the shortest distance that needs to be calculated throughout all the stations. This shows the result using the Dijkstra Algorithm to calculate the shortest route from Leyton Station to Oxford Street. This also shows from each station to take what train line from each. Please refer over to Chapter 4 where the Dijkstra Algorithm would be explained thoroughly.

Chapter 4: Evaluation

The application that is created for this project only meets the basic and intermediate level from the coursework specification. Both handle the inputs and validations with well commented code for other users to understand what has been coded. However, the implementation could be improved by the advanced level being complete. This could improve the system outcome by it being more flexible for the user to understand and alongside more features. Some of the features that would have been beneficial is visualisation and making the shortest distance more advanced by adding delays and times to this.

At present, the SOAP Web Service does not have a separate project as this is in the same project as the website application. This could be improved by adding all of this to another project as this would make sense and easier. At this point, it is all cramped and works within one project for the website. The SOAP Web Service contains the shortest route possible component. This could be improved by identifying the database to be connected with this too. The component is only added in the SOAP Web Service. However, this would add more benefit to other applications if the database is added alongside this to make it flexible and easier to understand.

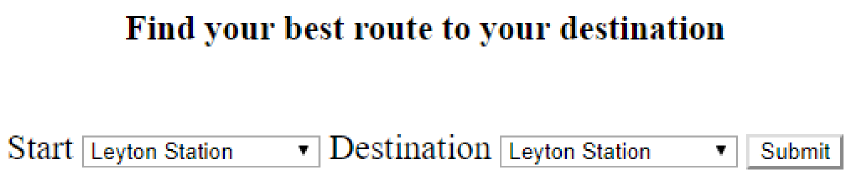
One flexibility the algorithm that has shown is the algorithm considers other train lines to be able to find the correct route. For example, if I wanted to go from Leyton Station and Oxford Street is on another line, it would consider the right route to be able to provide the correct one for the customer.

One positive is the distance component for the basic level is complete and is able to work with other textboxes without being able to change anything. This is complete by using another c sharp class. Once the class has been implemented, this works straight away by the textbox turning red as soon as it goes above 50.

One feature that is necessary that I would like to add is more user interface to make it clear and understand. At this point, it is very basic and does not have the necessary tools for the desktop and website application to be at a good standard.

Chapter 5: Algorithms Explanation

The algorithm that is used for the intermediate level is Dijkstra. Dijkstra is known to find the shortest paths between nodes in a graph representation. This can be altered and switched to the scenario for train station and train line. This is what the user is able to see and search for the shortest distance available.



Find your best route to your destination

Start Leyton Station Destination Leyton Station Submit

The Dijkstra Algorithm that has been implemented works in four sections that all come together. The first aim is to add the stations that is available. Once these stations that are available. These need to be linked by this being the second stage of this algorithm. Linking each station by the start and end would be necessary as each station would need to be linked with each other. During the linking of each station, the distance is added at this stage too. After this is complete, the first station that is added has to be set to 0 as once I was completing the algorithm, the result I got without using this was the other way around. Therefore, by putting this to 0, this would enable the result to be the correct way. The last stage is to search for the best route and output the result for the user to see.

Pseudo Code

This is the pseudo code to explain the algorithm behind the website application.

Link Stations

```
If(cell start == starting station {
If(cell end == endingstation {
add the link between the station
add the distance
        }
}
```

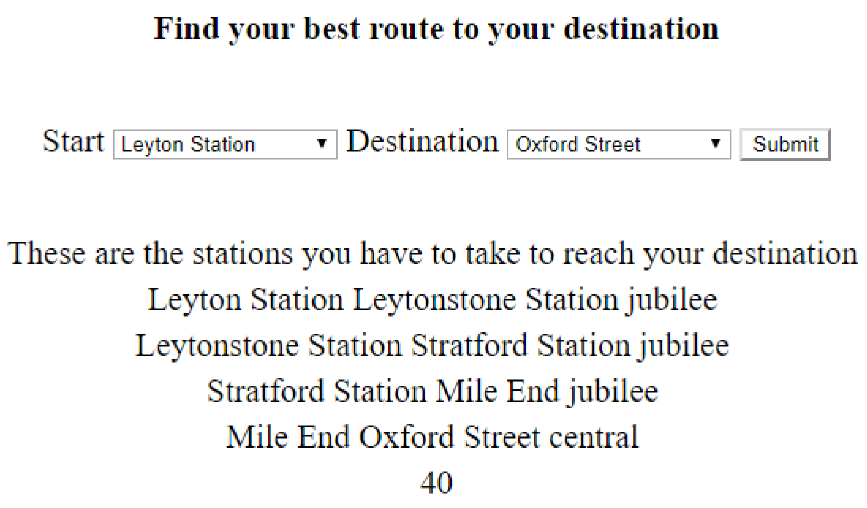
Search

```
If(value selected > value overall) {
Add to temporary
}
If(cell in temporary) {
```

```
Temp.remove();
}
```

Example

This is an example shown of how the algorithm shows the result. The result is shown below of the shortest route in the algorithm.



Find your best route to your destination

Start Leyton Station Destination Oxford Street Submit

```
These are the stations you have to take to reach your destination
Leyton Station Leytonstone Station jubilee
Leytonstone Station Stratford Station jubilee
Stratford Station Mile End jubilee
Mile End Oxford Street central
40
```

Improvements

The algorithm that is shown above is for the intermediate level. The advanced level is not complete and the improvements that could be made is the following:

Amend the distance calculation component you implemented for the intermediate functionality so that it also includes the time it takes to make a journey, not only the distance.

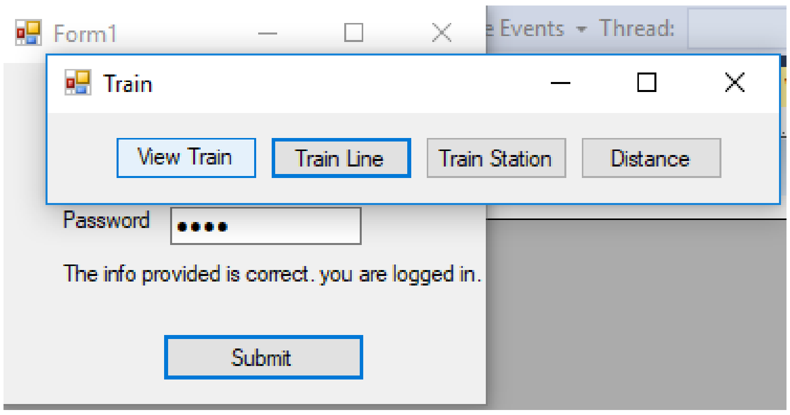
To simplify the calculation, you can make an assumption that trains on all lines go every 10 minutes and that changing a train just adds 15 minutes to a journey. A user should be able to see an estimated time of arrival and a breakdown of when one arrives and departs when changing trains.

Create a visualisation component which makes it possible for a user to visualise the train network. You can decide whether you want to develop the visualisation component to be integrated into basic desktop application or the intermediate web application.

Finally, make the implementation of the distance calculation component more robust by adding delays to particular sections of the network. A delay should be added to the link between two stations and the component should take delays into consideration when calculating the fastest route.

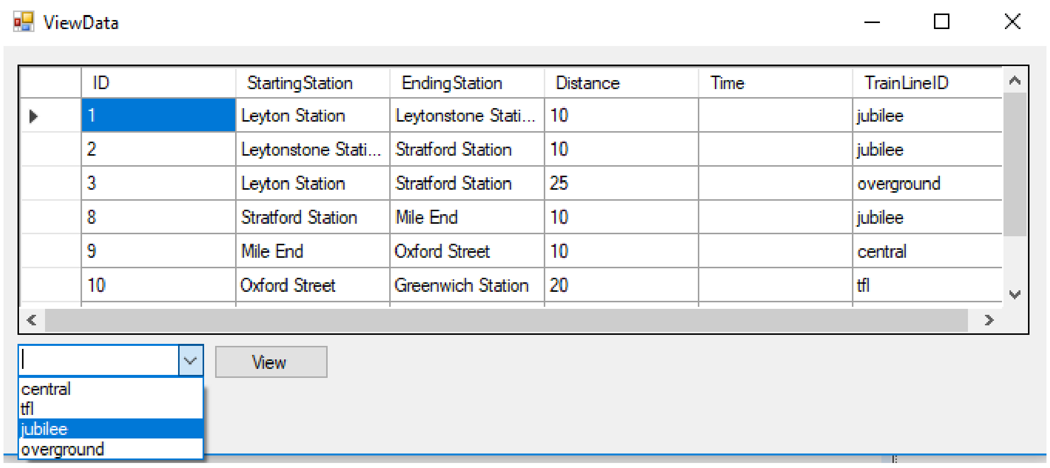
Chapter 6: Changes made after Demonstration

After the demonstration, I realised by speaking with Elena Popa at the demonstration for Programming Distributed Component; I need to add some changes to the current application. Elena mentioned that I needed the administrator to be able to view the train line in a separate page. This is explained below. Referring to Figure 12, 13 and 14, these images are the process in which the administrator is supposed to take to view what train line is on what station. Referring to Figure 14, this is the result of what is expected of this.



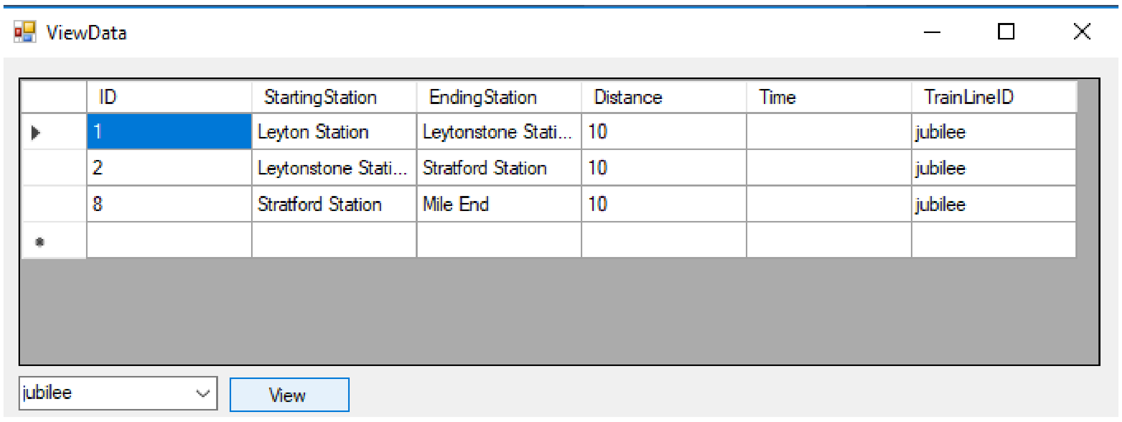
The info provided is correct. you are logged in.

Figure 12 shows how to access the new added feature to the Live Journeys.



ID	StartingStation	EndingStation	Distance	Time	TrainLineID
1	Leyton Station	Leytonstone Stati...	10		jubilee
2	Leytonstone Stati...	Stratford Station	10		jubilee
3	Leyton Station	Stratford Station	25		overground
8	Stratford Station	Mile End	10		jubilee
9	Mile End	Oxford Street	10		central
10	Oxford Street	Greenwich Station	20		tfl

Figure 13 shows the choice in which the train line can be selected.



ID	StartingStation	EndingStation	Distance	Time	TrainLineID
1	Leyton Station	Leytonstone Stati...	10		jubilee
2	Leytonstone Stati...	Stratford Station	10		jubilee
8	Stratford Station	Mile End	10		jubilee

Figure 14 shows the result of the chosen train line.