



FACULTY OF SCIENCE & TECHNOLOGY

BSc (Hons) Software Engineering  
May 2018

An Investigation into Optimized Travel Routes using  
Evolutionary Algorithms

by

Joel Holmes

Faculty of Science and Technology  
Department of Computing and Informatics  
Final year project

# Abstract

Evolutionary algorithms are a set of computing principles based on Darwin's evolution. They help form a part of the field known as machine learning or artificial intelligence. The algorithms are excellent at providing solutions to optimisation problems that are considered NP hard, and difficult to solve definitively. These include but are not limited to scheduling issues, Travelling Salesman Problem etc.

This work explores the idea of an artefact that combines an evolutionary algorithm with a web interface. This Interface allows for travellers, mostly backpackers to optimise a route through their desired destinations based on constraining factors. The system has been developed with server-side C# for the algorithm implementation, that is fronted by a bootstrap/Asp.net interface made interactive using JavaScript.

The following dissertation documents the projects journey from the start of the research to the requirements engineering and design, all the way through to the testing and evaluation process.

## Dissertation Declaration

I agree that, should the University wish to retain it for reference purposes, a copy of my dissertation may be held by Bournemouth University normally for a period of 3 academic years. I understand that once the retention period has expired my dissertation will be destroyed.

### Confidentiality

I confirm that this dissertation does not contain information of a commercial or confidential nature or include personal information other than that which would normally be in the public domain unless the relevant permissions have been obtained. In particular any information which identifies a particular individual's religious or political beliefs, information relating to their health, ethnicity, criminal history or sex life has been anonymised unless permission has been granted for its publication from the person to whom it relates.

### Copyright

The copyright for this dissertation remains with me.

### Requests for Information

I agree that this dissertation may be made available as the result of a request for information under the Freedom of Information Act.

Signed:

Name: Joel Holmes

Date:

Programme: BSc (Hons) Software Engineering

### Original Work Declaration

This dissertation and the project that it is based on are my own work, except where stated, in accordance with University regulations.

Signed:

## Acknowledgements

Firstly, I would like to thank Dr Shahin Rostami. For his dedicated supervision which reached far beyond the availability of our weekly scheduled meetings. Due to his constant mentorship and advice I was able to exceed my own expectations for this project, and I am extremely grateful and proud because of this.

During the project none of it would have been possible without the support of my family, specifically my Mother Debbie, Sisters Alayna and Isabella and, Grandparents Betty and Keith Billington. Their patience through my cranky project madness grind, was essential at keeping me grounded. I am especially grateful to Lauren Lett who helped make sure that throughout the process I saw some sunlight and had fun breaks along the way due to her love and companionship. Also, to the Lett family who have given me something to look forward to after all this has finished!

I would also like to mention those specifically who took the time to proof read my work which in the context of a dissertation is no small task. Therefore, massive thanks must go to Lauren Lett and Andrew Shannon.

# Table of contents

List of Figures .....	7
List of Acronyms.....	8
Chapter 1 – Introduction .....	9
1.1 Background and context .....	9
1.2 Problem Definition.....	10
1.3 Proposed Solution.....	11
1.4 Project Aims and Objectives .....	12
1.4.1 Deliverables.....	12
1.4.2 Success Criteria .....	12
1.5 Overview of Remaining Chapters .....	13
Chapter 2 – Literature Review .....	15
2.1 Evolutionary Algorithm Concepts .....	15
2.2 Biological Concepts .....	15
2.2.1 Darwinism .....	15
2.2.2 Natural Selection.....	16
2.2.3 Genes and Characteristics.....	16
2.3 Evolutionary algorithm mechanics .....	16
2.3.1 Optimisation.....	17
2.3.2 Basic Format of an Evolutionary algorithm.....	17
2.3.3 Multi Objective Optimisation.....	19
2.3.4 Objective space plotting .....	20
2.4 Tourism .....	21
Chapter 3 – Requirements Analysis .....	23
3.1 Overview of Requirements .....	23
3.2 Elicitation of Requirements.....	23
3.3 Existing Solutions .....	24
3.4 MoSCoW .....	25
3.5 Risk Analysis .....	26
Chapter 4 – Design and Methodology .....	27
4.1 Development methodology .....	27
4.2 Comparison of Methodologies .....	27
4.2.1 Extreme programming .....	27
4.2.2 Waterfall .....	28

4.2.3 Agile.....	29
4.2.4 Adapting methodologies for solo development .....	30
4.3 GUI Designs .....	31
Chapter 5 – Artefact Implementation .....	33
5.1 Algorithm Design.....	33
5.1.1 Weighted Sum Method.....	33
5.1.2 User Preference Articulation .....	35
5.1.3 Combinatorial Optimisation.....	36
5.1.4 Permutation Encoding .....	36
5.1.5 Double Point Ordered Crossover .....	37
5.1.6 Elitism.....	38
5.2 Artefact Architecture .....	38
5.2.1 Language choices .....	39
5.2.1.1 JavaScript .....	39
5.2.1.2 C# Server Side .....	40
5.2.2 IDE: Visual Studio .....	40
5.2.3 JSON format for API data .....	41
Chapter 6 – Software Quality and Testing.....	42
6.1 Testing approaches .....	42
6.1.2 Black box testing .....	42
6.1.2.2 Equivalence Partitioning and Boundary Value Analysis.....	42
6.1.2.3 Combinatorial testing – Category partition method.....	43
6.1.3 White box testing.....	44
6.1.3.2 Branch Testing.....	44
6.1.3.3 Coverage .....	44
6.2 Static Code Analysis .....	45
Chapter 7 – Evaluation and Conclusion .....	46
7.1 Success Criteria .....	46
7.2 Personal Evaluation.....	47
7.3 Conclusion.....	47
7.4 Future Work.....	48
7.4.2 Deliverables 1 & 2 .....	48
7.4.3 Deliverable 3 .....	50
References .....	51

## List of Figures

Figure 1: Graph to show global youth expenditure on travel (UNWTO, WYSE, 2016).....	9
Figure 2: Visual representation of a selection wheel (Obitko.com, 2018).....	18
Figure 3: Visual representation of simple binary mutation (Tech.io, 2018).....	18
Figure 4: Plot of objective space to aid in selection of solutions (Deb, 2008). ....	20
Figure 5: Graph to show the percentile usage of transportation methods in the context of tourism (Statista, 2016). ....	21
Figure 6: A look at the interface of a solution that does a different role but similar intended GUI (RouteYou, 2018). ....	24
Figure 7: Diagram showing the sequential flow of waterfall SDLC (Mahalakshmi and Sundararajan, 2013).....	28
Figure 9: Early Interface design for solutions home page (Mock up made using balsamiq)...	32
Figure 10: Early Interface design for solutions home page showing how colour will be used (Mock up made using balsamiq).....	32
Figure 11: Weighted sum formula (Deb, 2008). ....	34
Figure 12: Pseudocode for fitness function. ....	35
Figure 13: Visualisation depicting permutation encoding (Obitko.com, 2018). ....	37
Figure 14: Visualisation depicting single point order crossover (Rubicite.com, 2018). ....	37
Figure 15: Depiction of JavaScript map pulled into website. ....	39

## List of Appendices

Appendix A – Project Proposal.....	56
Appendix B – Ethics.....	64
Appendix C – Gantt Chart .....	71
Appendix D – Code libraries referenced .....	72
Appendix E – MoSCoW Prioritisation of Requirements .....	73
Appendix F – Risk Analysis .....	74
Appendix G – Equivalence Classes .....	75
Appendix H – Test Cases .....	76
Appendix I – List of supported countries .....	88
Appendix J – Static Code Analysis .....	89
Appendix K – Screenshots of system .....	90



## List of Acronyms

GUI – Graphical User Interface

STA – Student Travel Association

UNWTO – World Tourism Organisation

WYSE – World Youth Student Education Travel Confederation

NP – Non-deterministic Polynomial time

ICSAI - International Conference on Systems and Informatics

API – Application Programming Interface

EP – Equivalence Partition

BVA – Boundary Value Analysis

SDLC – Software Development Life Cycle

## Chapter 1 – Introduction

This chapter aims to provide a concise and in-depth analysis to the problem identified for this project, and the steps that will be taken to solve it.

### 1.1 Background and context

The travel industry is one of the largest in the world, over 1 billion tourists travel worldwide each year. The youth populace surmised of the ages 15 – 29, accounts for 23% of these travellers and around 280 billion a year is spent by them (UNWTO, WYSE, 2016).

Figure 1 shows a steady increase in youth travel expenditure. It is consistently growing due to the changing attitude of millennials towards backpacking, gap year travelling, and studying abroad (Gov Department of Education, 2012). This means that an already gigantic industry has the potential to grow continuously, however travel has many complications that are amplified when in the context of the young.

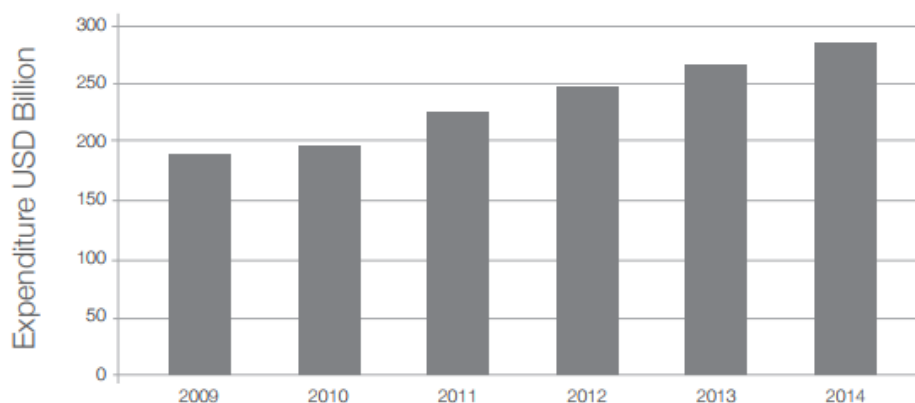


Figure 1: Graph to show global youth expenditure on travel (UNWTO, WYSE, 2016)

Travel for young people can be a daunting prospect as it can be the first true phase of independence. This can make organising travel complex and intimidating for first timers, who before now have had tasks like visa documentation, transportation and accommodation taken care of by senior family members.

All these factors can become rapidly expensive for young people and students, making it very difficult for them to be able to travel. The complication increases further when it is considered how constrained the budget most young travellers deal with is, the average spend of a young traveller is \$3000 (UNWTO, WYSE, 2016).

Due to the limited budget of young people it becomes necessary to try and optimise all these factors mentioned through limiting unnecessary expenditure by finding the best deals. Most young travellers having never planned a travel route before, find it difficult linking all the intended destinations together via transport links. This alone can provide a lot of pressure due to the potential legal ramifications or premature end of a trip due to incorrect legal documentation. Due to this optimal use of travel budget and resources often get pushed to the back of the priorities in favour of ease and convenience.

## 1.2 Problem Definition

The difficulty in planning travel routes often leads backpackers to go with an operated tour company, so that all the planning of flights, accommodation and visas are handled for them. One such company that does this is STA, they make travel planning significantly easier and more convenient, however it is limiting in many ways as a sense of freedom in terms of itinerary and customization of the trip is lost. This is because they must adhere to their specific tour and scheduling. Tour operators tend to choose accommodation outside of city centres, this is due to their prioritisation on dealing with the cheapest accommodation and only specific companies (Medina-Muñoz et al, 2003) (Ling, 2014). This means that not only will the itinerary potentially be filled with things that are uninteresting to the traveller, but there will also be many unnecessary early mornings of transportation for them that negatively affect the trip. Another disadvantage of tour companies is the mark-up cost placed on the price by them, adding to the already significant cost for the service of convenience and providing a tour. This ultimately means

that more of a traveller's money is paid to a tour company, where as if these deals were found by the travellers themselves they would be able to reduce some of the unnecessary expenditures, and then put that money towards further travelling, nicer food or accommodation.

Going it alone and planning a travel route without help is no easy task, although the rewards are there financially in terms of being able to do more with the money available, it can be a massive time sink to plan routes (Patch, 2018).

### 1.3 Proposed Solution

The proposal is for a Web solution, which allows the user to select intended destinations by pinning markers on an interactive world map. A simple clean GUI (Graphical User Interface) will produce a path through the destinations. This path would be ordered and optimised in such a way so that the destinations will be traversed to achieve the cheapest, shortest and most convenient route. The path will be optimised using a field of artificial intelligence known as evolutionary algorithms.

The problem is an advanced version of the travelling sales person issue, this is known to be an optimisation problem that evolutionary algorithms have been excellent at solving due to their NP nature (Potvin, 1996), Evolutionary algorithms have the potential to require a large amount of compute resources to operate (Yu, Qian and Zhou, 2015). Large volumes of data need to be processed such as flight information, distance, cost etc could potentially take a long time for the program to run for the user.

The application of genetic algorithms in the context of the proposed solution has not been done before, therefore has novelty in its originality. This will greatly benefit those trying to plan their trip as it will provide a convenient easy to use planning system without the large costs and loss of customization.

## 1.4 Project Aims and Objectives

To investigate how evolutionary algorithms can be applied to a travel planning solution to better the experience of travelling.

### 1.4.1 Deliverables

In this project there will be deliverables implemented according to achieving the project main aim:

- Evolutionary algorithm processing for optimisation of travel route.
- A web GUI that allows a user with little to no training, to interact with the algorithm parameters and show them results.
- Software Documentation in terms of Requirements elicitation/analysis, design, methodologies and a test plan to provide confidence in code quality, along with test suite of actual run tests and test data.

### 1.4.2 Success Criteria

Achieving and maintaining software quality in a system using methodologies and processes can be a very expensive procedure and has the potential to delay the development cycle and deployment significantly (Pryor, Alan N, 1999). To ensure that software tiptoes the line of being able to balance low costs and on time delivery is difficult enough, however it also must ensure that not only is it ready for delivery, but is it ready to be used? Quality can be understood through characteristics such as Functionality, Security, Usability and Portability etc. As well as prioritising these characteristics, fitness for purpose is very important in terms of software does what it should do, and doesn't do what it shouldn't (Myers, 1979).

A good indicator for software release can be providing the project with a success criterion, this criterion is essential at informing key decisions during the development

cycle of how to shape the software. It is worth noting that requirements can potentially change during development because of user needs changing. The success criteria therefore can also be reviewed to reflect such changes. Despite all this there is a chance for the user to be dissatisfied with the product due to unforeseen changes. It is ambiguous as to where the fault lies in such circumstances. Perspective dictates it could be the fault of the user for being unable to clearly articulate their thoughts, on the other hand it could be the fault of the development team to ineffectively elicit requirements from the user. Either way it is difficult to deduce the maximum requirements available from the user (ICSAI, 2017).

A measure of success for this project can be obtained by how well the project adheres to the must have MoSCoW requirements that have been prioritised in Appendix E.

## 1.5 Overview of Remaining Chapters

The rest of document will consist of 6 more chapters, the way these chapters will be implemented, and their contents are described below:

- Chapter 2 – This chapter will consist of a Background Study and research of the project, to provide the detail required to fully understand the dissertation. This chapter will serve to inform the reader of the research undertaken as well as demonstrate understanding of the problem domain, and what is required to solve it.
- Chapter 3 – This chapter will describe and provide the requirements analysis in a clear, concise manner so that the plan to tackle the problem directly relates back to the specifications and project requirements that have been prioritised.
- Chapter 4 – Design and Implementation methodology, this chapter will provide the justification for the chosen design and implementation methodologies.

- Chapter 5 – Artefact, this chapter will provide an overview of the solution using high level technical description of the algorithm mechanics. As well as significant breakdowns of how these mechanics were implemented.
- Chapter 6 - This chapter will include the testing. From testing techniques used and justification for these selections. The chapter will also refer to test cases derived using these techniques.
- Chapter 7 – This final chapter will consist of the Evaluation and how well the objectives have been met according to the success criteria. Finally, being round off by a conclusion and any future work improvements to be made.

## Chapter 2 – Literature Review

This chapter aims to provide the author with the tools and know how, to be able to complete this project to the best of their ability. As well as the author, this chapter serves the reader in terms of providing background knowledge for concepts this dissertation revolves around, the reader may not have previously been exposed to.

### 2.1 Evolutionary Algorithm Concepts

Evolutionary algorithms are the union of computer science and biological concepts. They attempt to solve computing problems in the most optimal manner, mostly through the application of Darwinism theory to a problem using computational power (Back, 1996). When these optimisation problems contain only one objective, it can be referred to as a single objective optimisation. Similarly, where an optimisation problem contains multiple objectives it is known as a multi objective optimisation (Deb, 2008).

### 2.2 Biological Concepts

Firstly, the report will break down the biological concepts that evolutionary algorithms apply to a computing environment.

#### 2.2.1 Darwinism

Evolutionary algorithms are based on a process observed in nature (Back, 1996). This process as observed by (Darwin, 1859) consists of reproduction, mutation, competition and selection. Mutation becomes guaranteed of any population that reproduces itself. Reproduction within a finite area causes natural selection to become a consequential factor (Fogel, 2006). All these factors form a basis of evolution. Evolution was defined as descent with modification (Darwin, 1859). The modification occurs due to Natural Selection. It is the descent with modification that is focused upon and accelerated during evolutionary algorithms.



### 2.2.2 Natural Selection

Natural Selection is the mechanism of evolution. This mechanism allows for the selection of more advantageous characteristics of a species. Individuals that are more easily able to survive due to these characteristics (Williams, 1996), will survive to pass on these attributes through reproduction (Endler, 1986). Because, only the organisms that have characteristics that allow them to survive, live long enough to reproduce. Only these advantageous genes are passed on to the next generation of offspring therefore giving us survival of the fittest.

Fitness level is a key concept applied to evolutionary algorithms in computing as it is used as a selection tool (Jacob, 2001), to determine what chromosomes will be used to reproduce. This is so that the next generation will be most suitable for the task.

### 2.2.3 Genes and Characteristics

Every Biological cell contains a set of chromosomes; these chromosomes are composed of genes. The complete collection of genetic information within an organism is called the genome (Jain et al., 2000). The expression of the genome into visible or noticeable characteristics can be described as the phenotype of an individual (Venter, 2001). Overall genes can be considered responsible for the attributes associated with a species or individual within a species.

Therefore, when programming evolutionary algorithms, chromosomes are recreated as data structures, these data structures will then contain genes or in the case of evolutionary algorithms the attributes (Jacob, 2001).

## 2.3 Evolutionary algorithm mechanics

Now the report will explain how the biological concepts are applied using computational power to give rise to evolutionary algorithms.

### 2.3.1 Optimisation

Evolutionary algorithms are excellent at providing solutions to problems associated with searching or optimisation (Mitchell, 2002). Optimisation is the process of finding the highest value yield for a situation, whilst using the least amount of resources possible (Spall, 2005). However, the yield value can vary from situation to situation based on priorities. In the case of this project highest yield will lead to the lowest fitness value mentioned in Natural selection 2.2.2. In the context of this project the fitness value will be based upon the following attributes of a travel route:

- The cheapest price
- Shortest distance flown between destinations
- Shortest flight time (So as well as distance connecting flights and waits are taken into consideration)

### 2.3.2 Basic Format of an Evolutionary algorithm

The simplest evolutionary algorithm contains 6 steps to run, these steps are as follows (Jacob, 2001).:

1. Initialisation – In order to apply evolution in an artificial manner, there must first be a population to do so. Therefore, the first step of an evolutionary algorithm is to initialise a set of individuals.
2. Initial evaluation – Evaluate the populations' individual fitness's based upon pre-defined criteria.
3. Selection – Now select the individuals with the best fitness levels so that they can be used in next step of the artificial evolution. The image below

shows how a chromosome with a higher fitness level has a higher rate of selection.

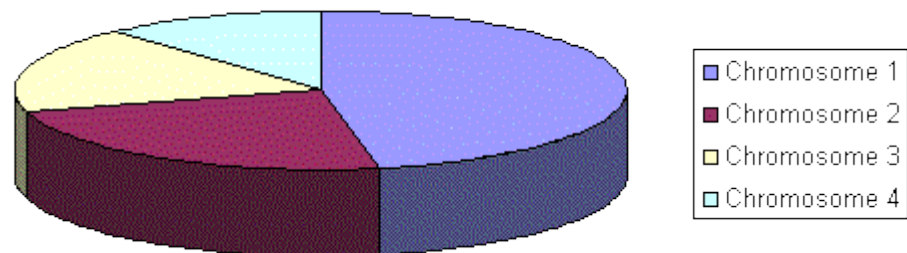


Figure 2: Visual representation of a selection wheel (Obitko.com, 2018).

4. Mutation – The individuals with the highest fitness levels will be chosen and half of each chromosome will be taken to form offspring. This is known as crossover (Fogel, 2006). After this the pair of individuals with the next highest fitness levels will be used as parents to form another offspring and so forth so that a new generation is formed through reproduction. In a simple binary encoded scenario of the chromosome an example of mutation can be seen below.

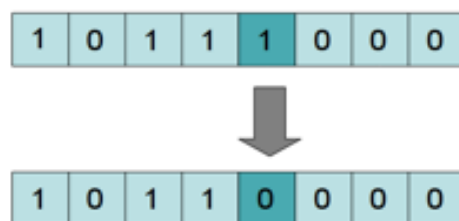


Figure 3: Visual representation of simple binary mutation (Tech.io, 2018)

5. Evaluation – Evaluate all mutants and calculate their fitness.
6. Termination check – Also known as the generational loop, it must be determined whether to go through another iteration of the algorithm and when to terminate so the program doesn't get stuck in an infinite loop. This can be decided based on the following termination criteria (Goldberg, 2012):

- a. Has the goal been achieved? (Maximum predefined fitness level reached therefore can be returned?)
- b. Performance stagnating
- c. Has the algorithm been set to run for a specific number of generations only?

7. If no maximum fitness reached, then continue with step 3.

### 2.3.3 Multi Objective Optimisation

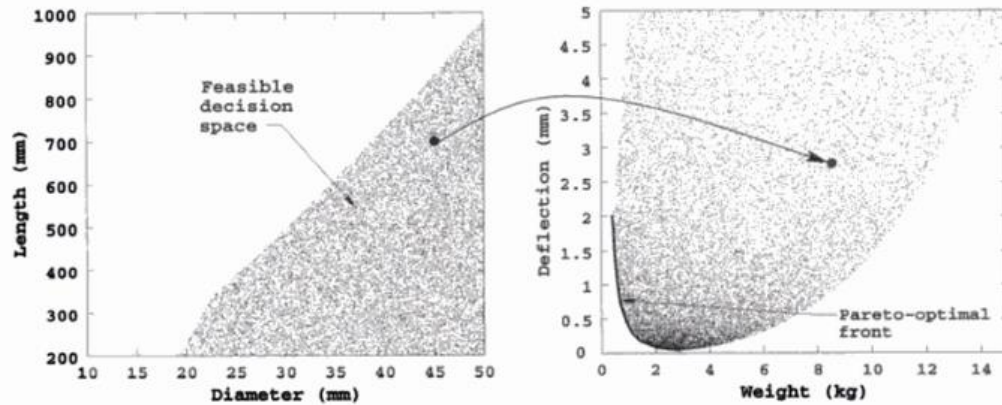
The problem this report focuses around has many contextual factors associated with it. Because of this the problem can be referred to as a multi objective optimisation problem, more specifically a 3-objective problem. These three objectives are:

- Objective 1 – Cheapest price
- Objective 2 – Shortest flight distance
- Objective 3 – Least number of connecting flights

Due to the multi objective nature of this optimisation problem it becomes more difficult to assign a singular fitness level to a solution, therefore an objective value must be assigned instead of a fitness level that is instead an accumulation of all the objectives involved (Deb, 2008). Multiple objectives can have differing worth's towards the objective value depending on solution priority. This makes the selection stage of the algorithm difficult when trying to compare solutions to figure out which one is more optimal. To limit the effect multi objective optimisation can have on selection, (Zitzler, Deb and Thiele, 1999) suggests a plot of objective space.

### 2.3.4 Objective space plotting

Solutions are plotted against one another, to find a region of solutions that have dominance over others.



Figure

4: Plot of objective space to aid in selection of solutions (Deb, 2008).

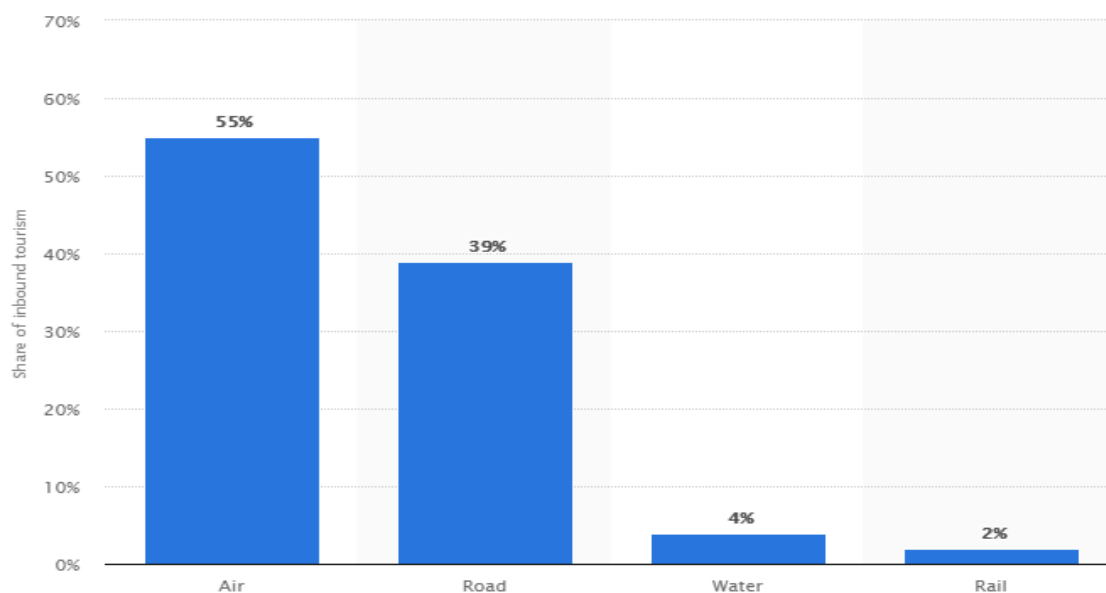
As you can see in figure 4 solutions that happen to lay on the curve are called Pareto-optimal solutions, the curve formed from joining these plots is known as the Pareto-optimal front (Pareto, 1896). The entire space of solutions can be divided into a Pareto-optimal set or a non-Pareto-optimal set, when one solution on an objective plot space is clearly identifiable as better than another, the solution is said to dominate the other. If between the solutions no clear dominant region is apparent, it can be said that these are non-dominated solutions and that they're objective values maybe weighted differently in terms of the multiple objectives prioritised (Deb, 2008). They are both equally important and so it cannot be stated one is definitively better than the other. In some cases where solutions are only marginally better than others the term Weak Pareto-Optimality is used. In the case of a solution being the overwhelmingly obvious optimal solution, the term strong dominance is used (Coello, 2014).

Plot Objective space scatter graphs are suitable for aiding dual objective selection. To aid a 3-objective selection process it was suggested by (Meisel and Michalopoulos, 1973) to use a scatter plot matrix. This can be effective at comparing the two algorithms. In this report a 3-d scatter plot will be used to visualise solutions. Although harder to visualise

than the 2-D scatter plot, the 3-D scatter plot is necessary to cover the three objective points concerned with the problem solution, and fully evaluate the dominant regions to identify the most optimal solution.

## 2.4 Tourism

The definition of tourism varies from institution to institution, however in its basics it can be narrowed down to a person travelling away from their usual environment for leisure (Dictionary, 2018). In 2001 a study was performed to find out the most used method of transportation for tourism purposes. This study showed that the most used method of transportation was by car 61% and air travel the second highest of 21% (European Environment Agency, 2001). Data from 2016 shows how the tables have turned in terms of methods of transportation for tourism purposes. As you can see below from the graph air travel has risen to 55% and road transportation has dropped to 39% (Statista, 2016).



*Figure 5: Graph to show the percentile usage of transportation methods in the context of tourism (Statista, 2016).*

From this data it is easier to understand why there has been a massive increase in the number of websites that offer flight comparison services. As the percentage of people using air transportation as their main method for travelling has risen by 34%, it is only natural that more of these websites would appear to take advantage of this. In the next

Chapter an analysis of some of these solutions will be performed to solidify the proposed systems niche, in a market of over saturated flight comparison websites that all hold similar roles and functions.

## Chapter 3 – Requirements Analysis

Chapter 3 gives detailed depiction of one of software engineering's most important tools/processes, Requirements. This chapter aims to show the reader: What requirements are, how to elicit requirements, how and why they were prioritised and finally a risk analysis.

### 3.1 Overview of Requirements

Requirements analysis are a set of guidelines. These guidelines are a place for people involved with a project such as developers, clients, or any other stakeholders of the project to document their desired properties of the solution, and user's needs. These properties can be varied, and include non-functional requirements, but most commonly are functional (Kotonya and Sommerville, 2004).

Requirements analysis are not a static set of guidelines, they reflect user's needs. The user may not always know what these needs are or how to correctly articulate them to a software engineer. This can be a big challenge when procuring a requirements specification document. Furthermore, user's needs are subject to change throughout development and use of a solution (Van Lamsweerde, 2013). Therefore, it becomes necessary to review the requirements specification at certain points during the development lifecycle to ensure the solution is on track to meet the user's expectations (Kotonya and Sommerville, 2004). The requirements specification is a piece of documentation that acts as a median between the developers and users/clients.

### 3.2 Elicitation of Requirements

Due to a lack of client for the proposed solution, it is necessary for a background study of other similar solutions to be performed. This will be done to analyse present solutions functionality, and by doing this see what functions from other solutions can benefit the proposed solution. Additionally, an analysis of other solutions will allow me to better see



what the proposed solution should do that the others are not to further cement its niche in the travel and tourism industry.

### 3.3 Existing Solutions

Currently there are websites that act as planners for you to organise a trip abroad (RoutePerfect, 2018). However, these sites lean more towards shorter holidays focused around one singular destination. The solution proposed in chapter 1 focuses its priorities on backpackers intending on navigating multiple different countries and even continents. Another key defining factor in the originality of the proposed solution in comparison to existing solutions, is that the proposed solution intends to choose your route for you based on your destinations and limiting factors. This is something no current solution offers.

There exists an outdoor travel planner that is intended more for long haul cycle and leisure trips (RouteYou, 2018). This site has provided inspiration such as the GUI, and some of this will be applied in terms of functionality of the site to the backpacking world travel context for the solution in this project. Site interface of RouteYou can be seen in

Figure 6.

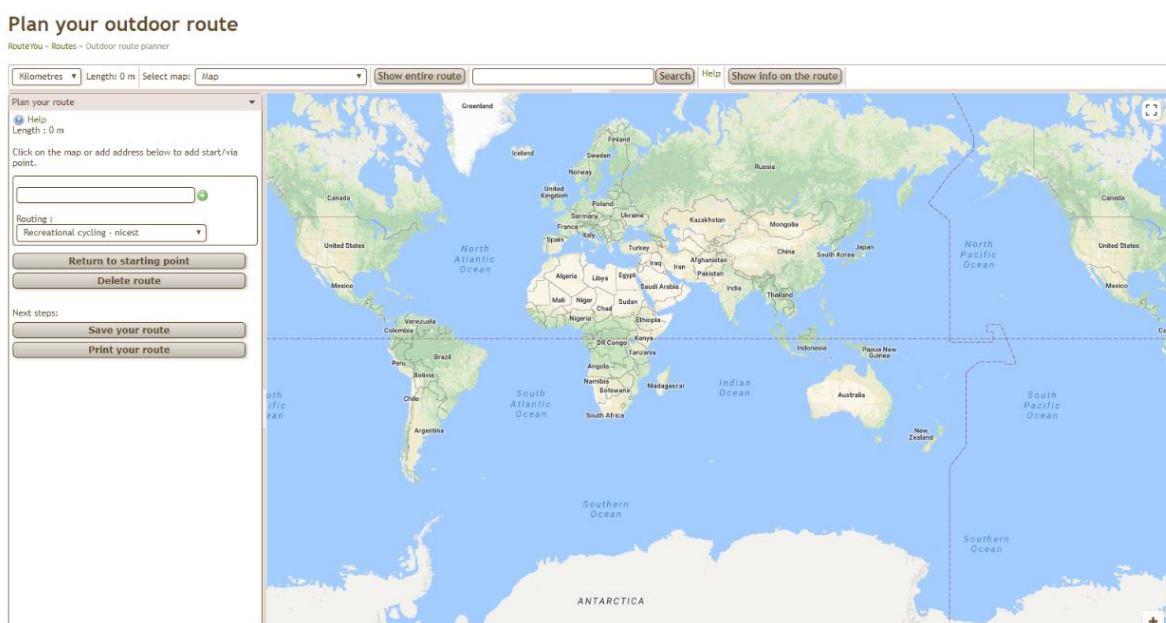


Figure 6: A look at the interface of a solution that does a different role but similar intended GUI (RouteYou, 2018).

### 3.4 MoSCoW

Requirements need to be prioritised to work efficiently towards more significant goals with the time available. A MoSCoW approach was chosen as it is possible to prioritise requirements effectively by a single person. These requirements can be seen in Appendix E.

MoSCoW principles (Craddock et al., 2008) were originally developed by Dai Clegg. The principles were divided into four identifiers of requirements, as can be seen below.

1. Must have's – These are requirements that are essential to the solution. For any sort of satisfactory user and developer interaction must be contained within the system. Without the Must have requirements the solution could potentially be; Unsafe, Illegal, project should be cancelled without this functionality.
2. Should have's – A should have requirement can be defined as being important to the systems functionality, but not vital. The solution is still viable without such requirements; however, it may impact systems quality such as performance, or usability without.
3. Could have's – Could have's are desirable functions that can be added if the time scale and planning allow for it but are not as severely hurtful to system quality if left out in comparison to should have's.
4. Wont have's - These are requirements that are a part of the list to provide clarity on the projects aims but due to the constraints such as time and resources will not be added during this development life cycle.

### 3.5 Risk Analysis

Risk management is a process that is best described as identifying the potential for a problem to occur (Charette, 1989). By doing this it can be understood how likely this problem is to occur, how severe will this problem be if/when it occurs and finally what steps can be taken during this project to minimize the potential undesirable outcome. Risk management is essential to any project because it not only allows anticipation of errors and preplanning for their eventuality, but errors found earlier in the SDLC have also been found to be much cheaper to fix, whereas the later they are found in the cycle the more expensive they become by factors of between 50-200(Boehm and Papaccio, 1988). A risk analysis has been performed for the project and can be seen in Appendix F.

## Chapter 4 – Design and Methodology

This chapter justifies and explains the selected development methodology used during this process. To explain why these methodologies were used over others they are also critically evaluated against some alternatives.

### 4.1 Development methodology

Selecting a development methodology can be a very impactful part of producing a piece of software. A lack of a structured methodology sometimes known as chaos coding, can have a negative impact on the quality of the software produced. It is important to note that no single development methodology is perfect for every proposed system. Software engineering is all about taking into consideration contextual factors of the specific situation. Through the identification of these contextual factors you can choose a development methodology based on the requirements of the project, and which method is best suited to produce those requirements (Pressman and Maxim, 2015).

### 4.2 Comparison of Methodologies

In this section of the dissertation the aim will be to provide a brief evaluation of the more common software development methodologies.

#### 4.2.1 Extreme programming

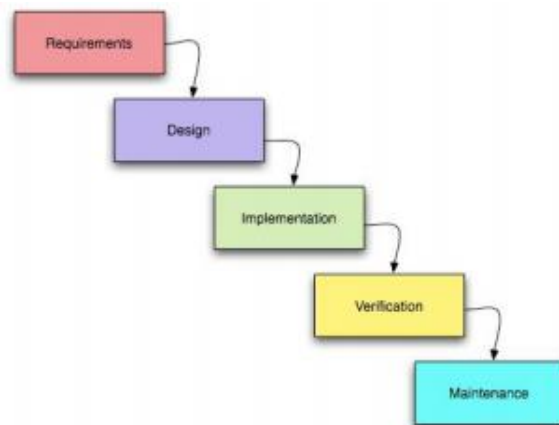
In the short term extreme programming may increase the speed at which features can be up and running through heavily programming focused philosophy. Due to this the documentation of software can be heavily neglected (Beck, 2000). A lack of clear and concise documentation can lead to an increase in maintainability issues and cost more money further down the line. All software ages to a certain degree and cannot be forever maintained by the same individuals who created it. Therefore, software documentation is essential to communicate and evidence key pieces of information for present and

future developers (Parnas, 1994). Extreme programming uses pair programming as a core component of its methodology.

Due to this core component of pair programming, and the negligence of software documentation which has proven to be a key determining factor in software quality, extreme programming is not a viable methodology for this project.

#### 4.2.2 Waterfall

Waterfall methodology is one of the most traditional SDLC's there is. It has a sequential pattern and moves through each stage one at a time.



*Figure 7: Diagram showing the sequential flow of waterfall SDLC (Mahalakshmi and Sundararajan, 2013).*

Waterfalls advantages and disadvantages can be viewed as the opposite of extreme programming. The sequential flow means that the design and documentation sections of a project is done very thoroughly. This is because one section of life cycle is not ended until completed in its entirety (Mahalakshmi and Sundararajan, 2013).

This is beneficial for some projects where corners cannot be allowed to be cut, for example critical software where lives are potential at stake in the event of software failure. The sequential flow can mean that the early stages of development are performed excellently, but to complete the latter stages such as implementation and testing

thoroughly, either features must be cut to fit the project within a time frame or the project will not be delivered on time (Langer, 2012).

Another issue with waterfall is that once a section has been completed it is difficult to revisit without rendering a lot of other work obsolete. This makes mid project pivoting, and flexibility very difficult during development. This is a problem because users' needs are known to be constantly changing and evolving, and if the developers are stuck adhering to requirements that no longer reflect the users present desires then resources are just being wasted completely (Van Lamsweerde, 2013).

#### 4.2.3 Agile

Agile is a methodology that divides up the project timeline into cycles called sprints (Shore and Warden, 2008). It is common practice for these sprints to focus on a single user story or requirement that will define that iteration of the software. The sprints are to include daily meetings to ensure the whole team is on track and emphasise collaboration and communication throughout the development process (Cockburn, 2009). At the end of each sprint/iteration it is expected that a version of software will have been created that is releasable.

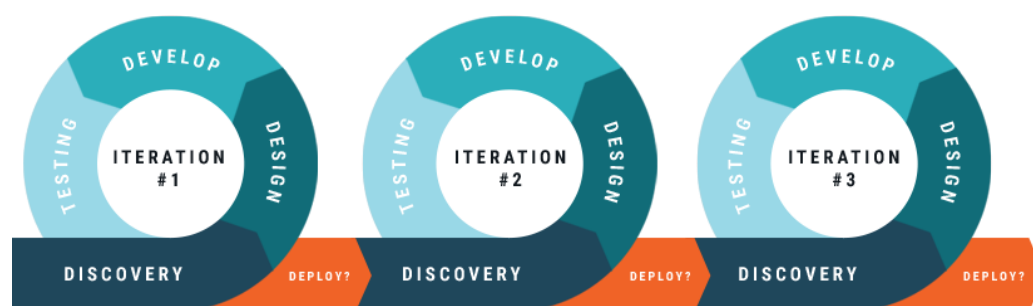


Figure 8: Agile development visualisation (App Developers UK | Mobile App Development | The Distance, 2018)

As well as the daily scrum meetings for developers, weekly meetings are held with clients to provide a feedback loop. The collaboration element of agile is one of the main focuses. Due to this project being a solo developer task it can be very difficult to fully implement agile.

Agile principles will be adopted rather than agile itself, instead of committing the project to one methodology it will try to take a hybrid approach to development. This will be done by utilising some of the fast-paced programming approaches to software of agile, and extreme programming. However, at the same time to focus more on documentation, than some of these chaos coding techniques. Documentation has been proven to affect many quality attributes such as maintainability, and so if it is neglected the quality will be negatively affected (Mantyla and Lassenius, 2009).

#### 4.2.4 Adapting methodologies for solo development

As explored in the above sections, there is no single methodology that conquers all scenarios, and the selection process only becomes more complicated by having a solo development team, as they are mostly suited for larger teams and organisations. To follow a methodology that suits this project, the following things will be done:

- Fixed timebox sprints (7 days) – Week long sprints have been adopted as this coincides well with the weekly supervisor meetings that have been planned throughout the project. This in turn means that the two feedback loops can be combined without expending any extra resources on these tasks.
- Gantt chart – It has been widely debated whether traditional planning methods such as Gantt charts have a place in agile. This is due to the rigid structuring they are perceived to have (Karlesky and Voord, 2008). The value of a traditional planning method such as a Gantt chart in this context is that not a full agile approach is being taken, and Gantt charts have value with visualising time

frames. This is not only important for the developing, but the documentation. Furthermore, the Gantt chart does not have to be reviewed as an immovable structure as they can be revisited and revised later if necessary. The Gantt chart for this project was originally very different as can be seen by the project proposal Appendix A. This Gantt chart was eventually revised as the full scope of the project was revealed found in Appendix C.

### 4.3 GUI Designs

When Considering the design of the applications front end, first and foremost it is key to consider the intended userbase of the solution. The solution must be designed so that it is as inclusive as possible for the intended userbase. However, this doesn't not mean it should exclude unintended users (O. Galitz, 2007). For example, in Chapter 1 of the report the target userbase was identified as youth backpackers. Generally young people are very hands on with technology and more easily able to use it compared to other demographics. Just because our intended audience may be more familiar with technology doesn't mean usability should be neglected. Unnecessarily making it difficult for users to use the system out of ignorance of other potential users could cause some young travellers for example, that potentially have learning disabilities or some elderly travellers to have trouble using the system and be excluded from the userbase due to poor design.

A well-made interface with good usability can be evaluated by how much training is required to use a solution (O. Galitz, 2007). In the case of the proposed solution the aim is for little to non-training required. To ensure the solution is inclusive as well as not exclusive, as part of the testing process later in the report Usability testing will be performed and any necessary changes identified from this can be acted upon.



Figure 9 shows an initial design of the home page, which contains a world map which will be clickable to allow the user to select countries as intended destinations. These destinations that are selected will then be listed and the algorithm will be able to be run by the user on them. After which, the route can be saved for later viewing or editing.

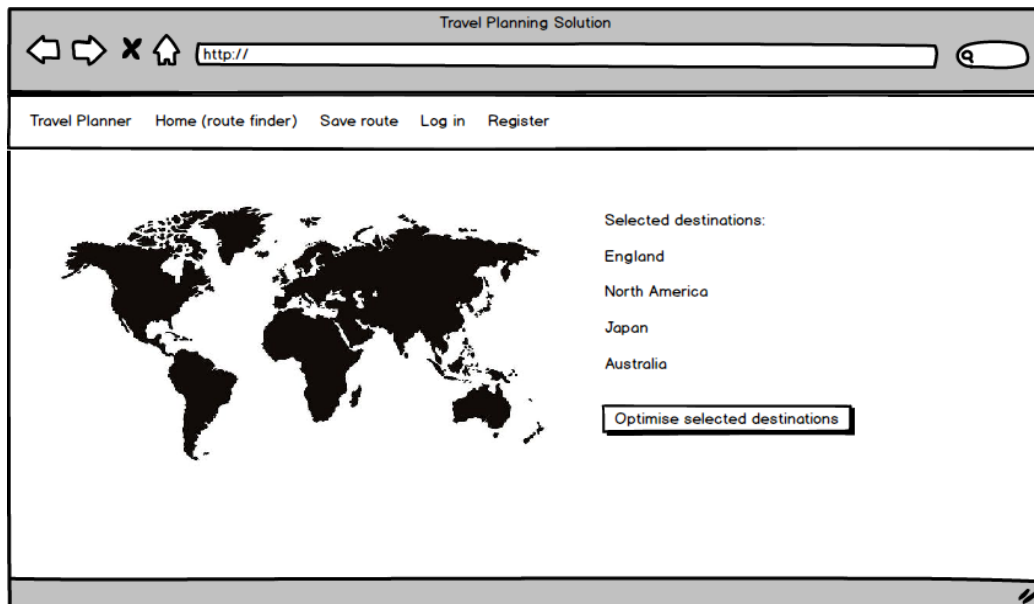


Figure 9: Early Interface design for solutions home page (Mock up made using balsamiq).

Figure 10 Shows the same initial design, but with colour introduced to show how when a country is selected it turns blue to make it stand out and more easily visible for the user.



Figure 10: Early Interface design for solutions home page showing how colour will be used (Mock up made using balsamiq).

## Chapter 5 – Artefact Implementation

This chapter's purpose is to set out the methods, tools and techniques used to implement my artefacts. To justify these selections, they will also be critically evaluated against alternatives were able for example, in 5.2.3 JSON vs XML.

### 5.1 Algorithm Design

The algorithm was designed using a whole range of operators and components e.g. user preference articulation, elitism etc. These will be explained in the section 5.1.

#### 5.1.1 Weighted Sum Method

Due to the multi objective nature of the optimisation problem proposed in this report, comparison of potential solutions can become increasingly complex. One method to reduce this complexity is known as the weighted sum method. The idea is that Multi objective/criterion selection can be reduced to a single objective fitness level, by applying weights to the objectives based on the problems prioritisation (Fisher, 1958).

The weights I have decided to apply to each objective are as follows:

- Price – 0.5
- Distance – 0.3
- No of Connecting flights 0.2

After assigning weights to the objectives based on priorities, it then becomes key to normalise the data. For example, price and number of connecting flights have different units and scales. The data must have the same scale to give the weights meaning. To do this the values of Price and Distance are multiplied by  $10^{-2}$  (Messac, Ismail-Yahaya and Mattson, 2003). This is based on the longest possible flight from Dubai to New Zealand is approximately 9032 miles, and 1,184.19 GBP with between 1-2 connecting flights (Emirates, 2018).

The weight sum formula (figure 11), has been used to devise some pseudo code (figure 12). The pseudocode is a function that intends to append the multiple objectives into one fitness level whilst taking constraints such as flight dates, and minimum number of destinations into account.

$$\left. \begin{array}{ll} \text{Minimize} & F(\mathbf{x}) = \sum_{m=1}^M w_m f_m(\mathbf{x}), \\ \text{subject to} & g_j(\mathbf{x}) \geq 0, \quad j = 1, 2, \dots, J; \\ & h_k(\mathbf{x}) = 0, \quad k = 1, 2, \dots, K; \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \dots, n. \end{array} \right\}$$

Figure 11: Weighted sum formula (Deb, 2008).

There can be disadvantages to the weighted sum approach. If some of the objectives are intended to be minimised, but others are to be maximised they must be converted into one type. This adds further complexity to an already difficult scenario (Rao, 1994). Also, if any of the weights of the objective  $\leq 0$  then a Pareto-optimal solution cannot be found. It is appropriate to use a weighted sum approach in this scenario because all the objectives share the same type of minimization. In addition to sharing the same type all the weighted objectives are equalled to 1 so a Pareto-optimal solution will always be found in convex circumstance.

```

Function Fitness (price,distance,noOfConnectingFlights,noOfDestinations,outboundFlightDate,inBoundFlightDate,flightDate)

    weight = [0.5,0.3,0.2]
    minDestinations = 3

    price = (price * 10^-2) * weight[1]
    distance = (distance * 10^-2) * weight[2]
    noOfConnectingFlights = noOfConnectingFlights * weight[3]

    objValue = Price + Distance + noOfConnectingFlights

    if(flightDate.after(outboundFlightDate) && flightDate.before(inBoundFlightDate)) {

        if (noOfDestinations ≥ minDestinations) {

            return objValue

        }

        Else {

            return = null
            errorMessage.show(invalid number of destinations selected error message)

        }

    }

    Else {

        return = null
        errorMessage.show(invalid date error message)

    }

}

```

Figure 12: Pseudocode for fitness function.

### 5.1.2 User Preference Articulation

The algorithm designed for this project has the main aim of helping humans make decisions. The decisions in the context of the defined problem are concerned with backpackers and helping them make decisions that more efficiently utilise their resources, such as money. The aim of the evolutionary algorithm is to find a single or set of pareto optimal solutions that the user can apply to help plan their travel route. The method being used to compare the solutions effectiveness, with objective values as explained in the previous section is weighted sum. Usually an evolutionary algorithm iterates through populations until a satisfactory solution has been found according to the rules set in place.

In this case the rules can be considered as the weights assigned to each of three objectives of the solution. User preference articulation will be performed by allowing the user to change these weights through an input, for example a text box. This project will incorporate a priori method of preference articulation, this is where the user defines the

preferences before the algorithm is run and so the decision maker is predefined (Rostami et al., 2015).

By default, the weights have been set with the price having the highest impact on the objective value because, as mentioned in chapter 1 the main target audience is for low budget youth travellers who want to optimise their travel experience. However, through user preference articulation the program can be more customizable, and appropriate for a wider range of audiences. This in turn allows for users to a certain degree bend the rules of the algorithm, so that satisfactory pareto optimal solutions are more aligned with their own needs (Thiele et al., 2009).

### 5.1.3 Combinatorial Optimisation

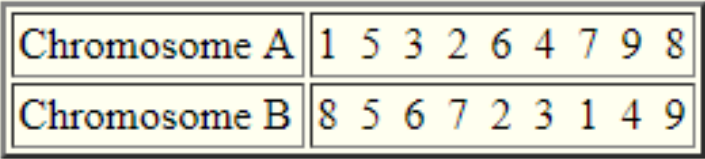
To explore the varied travel routes available according to the parameters set by the user such as, destinations and date. The problem variable that will be changed is the sequence that the destinations will be visited in. This is known as a combinatorial optimisation problem.

Although real numbers are involved in the problem, such as the flight prices, these prices are dependent upon the order of which the destinations are selected. Due to this the number of destinations can be viewed as discrete and finite, which leads the search to find the most optimal permutation of them (Papadimitriou and Steiglitz, 1998).

### 5.1.4 Permutation Encoding

As alluded to in the literature review of the project, one of the most important stages of producing an evolutionary algorithm is to decide how the chromosomes will be encoded. In this project permutation encoding will be used.

A permutation is the number of possible ways to order a set of elements (Dictionary.com, 2018). Permutation encoding is most commonly associated with the TSP problem, which this project is loosely related too.



Chromosome A	1	5	3	2	6	4	7	9	8
Chromosome B	8	5	6	7	2	3	1	4	9

Figure 13: Visualisation depicting permutation encoding (Obitko.com, 2018).

In this example the data is encoded so that each number in the chromosome represents a destination and the order they are visited. Due to the nature of the problem the number of ways crossover can be applied to the chromosomes becomes very limited. This is because every destination must remain present in every solution identified during the evolutionary process, it is imperative that only the order changes. Therefore, the type of crossover that has been used in this algorithm is double point ordered (Gen and Cheng, 1997).

#### 5.1.5 Double Point Ordered Crossover

In single point ordered crossover a collection of consecutive alleles is selected from parent 1, and remaining values are put into the offspring in the order they appear in parent 2. If a second offspring is desired from these parents, the parents can be flipped, and the process starts again. Order crossover can be seen from figure 14 below.

```

Parent 1: 8 4 7 3 6 2 5 1 9 0
Parent 2: 0 1 2 3 4 5 6 7 8 9
Child 1:  0 4 7 3 6 2 5 1 8 9
  
```

Figure 14: Visualisation depicting single point order crossover (Rubicite.com, 2018).

Double point ordered crossover has the exact same mechanisms except from the fact, that the selection of consecutive alleles occurs twice. Order crossover is one of the fastest crossover operators due to the lack of overhead operations required. This means that more generations can be processed in given time as opposed to other types of crossover (Fogel, Bäck and Michalewicz, 2000).

#### 5.1.6 Elitism

An elitist model for an evolutionary algorithm works as a counter balance to the mutation operator. Elitism is the process of preserving a percentage of the highest fitness solutions, and not allowing them to be mutated. This is so that they pass through the selection process and enter the next generation unaltered (Goldberg, 2012). This can be very beneficial for performance purposes because it means the algorithm does not waste time discovering, previously discarded solutions. These elite solutions remain eligible for selection during the crossover operator. Overall it has been proven that an elitism operator being present converges algorithms towards a global optimal solution (Rudolph, 1996).

It must be noted that although elitism can be a very beneficial tool to aid an evolutionary algorithm, the elitism operator should not be overdone. If too high of a percentage of the population are considered elites then this limits the diversity of the pool of solutions decreasing the chance for potentially more dominant solutions to arise (Deb, 2008). Due to this the elitism percentage for my algorithm will be set between 5-10% of the population.

#### 5.2 Artefact Architecture

During the production of this artefact, code libraries were used to aid functionality and flexibility of the project, for a full list of the libraries used see Appendix D.

### 5.2.1 Language choices

During this section an evaluation of the languages used within the project will be presented, along with justification for the selected choices.

#### 5.2.1.1 JavaScript

JavaScript was used for the website so that it could be made interactive. It is mainly for the world map that is pulled in to the solution. The world map provides clickable countries so that it is easier for the user to select their destinations, and more clearly visible which destinations have been selected. As can be seen via figure 15 below.

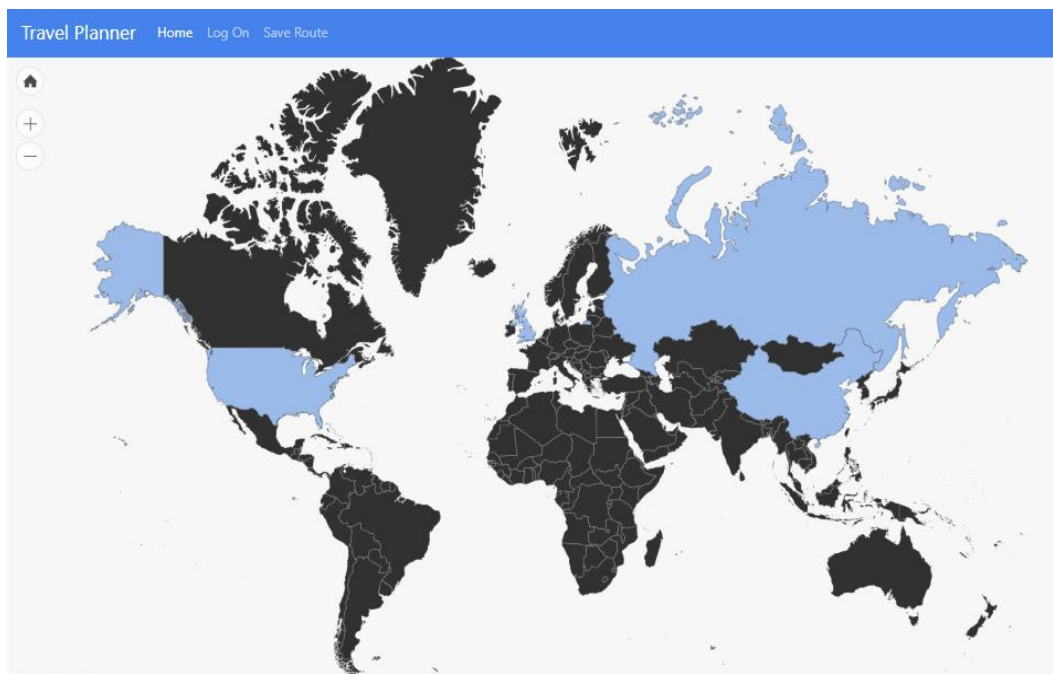


Figure 15: Depiction of JavaScript map pulled into website.

The reason JavaScript was chosen to implement the map was due to how fast it can be rendered to the end user. This is because the script is executed client side, and therefore processing occurs on the user's processor rather than the server which means less bandwidth is taken up. Also, due to being one of the most common DOM manipulator languages a lot of prewritten customisable functionality to implement the map was already readily available to me through a map library (amCharts, 2018).



Even though JavaScript is one of the most common client-side scripts, a lot of rendering inconsistencies persist. This is due to different layout engines changing how the functionality and interface is managed. This has the potential to negatively impact the websites portability across multiple browsers, and devices such as portable vs desktops (Flanagan, 2011). To circumvent this a bootstrap template that is dynamic across multiple devices, and browsers will be used as to aid consistency. Along with this extra testing will be performed to ensure portability on popular browsers and devices. Full screenshots of the end system can be seen in Appendix K.

#### 5.2.1.2 C# Server Side

C#.net was chosen as the implementation language for the evolutionary algorithm because, it is easy integrate the code with html using the Asp.net framework. The .net class library allows for rapid prototype development, this was very beneficial due to the tight time constraints that were afforded to developing the artefact. The extensive class libraries allowed me to use GAF a genetic algorithm C# library. This library provides support for implementing an evolutionary algorithm and, can help shed any unnecessary code complexity through its predefined data structures usable on for example chromosomes.

Evolutionary algorithms can take on a lot of overheads during most methods of crossover, in C# these overheads can take a long time to process unless a specific memory clean up function is designed. The selected crossover double point ordered has barely any overheads and so this lessens the burden on real time processing negating one of C# flaws.

#### 5.2.2 IDE: Visual Studio

The use of C# and Asp.net meant that it was ideal to develop all the project including the JavaScript files in Visual studio. Visual studio is an excellent IDE because it is language

independent therefore, you are not constrained to using any language other than the one that is best fit for the requirements of the project.

As well as being language independent, Visual Studio has a lot of built in features that make developing a project seamlessly easier. The integrated Git repository feature allows for easy tracking of different work versions along with the union of branches into the master branch when a file is ready. Not only does this help avoid risks such as data failure and loss of the project, but it also allows for the development of new features and changes without compromising pre-existing code that works (Visual Studio, 2018).

The lack of language constraint could be viewed as a negative aspect for some developers, as it does contain a lot of bloat on the installation size that is not necessary or used by all developers.

### 5.2.3 JSON format for API data

JSON was used as the format for data received from the API TravelPayouts. JSON is very beneficial as it is easily integrated into Visual Studio projects thanks to the JSON.net library, and parsing JSON into an object that can be referenced requires minimal effort. This is important because, a lot of data needs to be imported for the algorithm to run. Shortening the time, it takes to turn the string received from the API into a referenceable object, means that the time taken to output to the user is greatly decreased. JSON was chosen over something like XML since XML requires data to be read at the client side (PENG, CAO and XU, 2011). The algorithm is being implemented using C#.net which is a server-side scripting language, therefore the data cannot exist in the same space without the use of extra unnecessary code such as ajax calls.

## Chapter 6 – Software Quality and Testing

This chapter will give the reader an understanding of the test techniques used by the author to generate the test cases. In this case EP and BVA was used with branch testing. As well as test cases this chapter explores code confidence in terms of coverage.

### 6.1 Testing approaches

Testing a program is all about approaching software with a mindset. In most cases an exhaustive approach to testing is either one of two things, impractical or impossible. This is due to constraining factors that must be considered such as time and financial resources (Myers, 1979). If a program cannot be exhaustively tested then the program needs to have test case derivation techniques applied to it, so that we can acquire code confidence. When a program is being tested it is important to make sure it does what it should do, but also equally important that it doesn't do what it shouldn't (Myers, 1979). All the test cases derived using the below techniques can be found in Appendix H.

#### 6.1.2 Black box testing

Black box testing is a test case derivation approach where you assume the position of someone who knows nothing about the programs code or inner workings. This is done by providing inputs and then verifying the outputs against an expected outcome (Beizer, 1995).

##### 6.1.2.2 Equivalence Partitioning and Boundary Value Analysis

EP (Equivalence Partitioning) is a way of placing inputs into classes based upon how the tester thinks the program will handle the inputs. If the program is expected to handle the inputs in the same way they are all put in the same class. If they are not a new class is made (BSI, 1998). This way in theory test cases are not redundant, as we can expect if one input in a class will find an error then all the inputs in that class will find an error

(Ostrand and Balcer, 1988). A list of how this programs inputs has been separated into classes can be seen in Appendix G.

BVA (Boundary Value Analysis) can be considered as an extension of EP as it is about selecting inputs on the very edges of input classes. The reason for this is that there is evidence to show that errors have tendencies to form clusters that appear most at the boundaries (Hamill and Goseva-Popstojanova, 2009).

EP and BVA were chosen to derive test cases for this project due to the immense number of possible inputs for the program such as Date of travel, destinations and algorithm weights. As testing is all about using limited resources efficiently. As explained above EP and BVA allows the author to test this artefact and ensure key features work as intended. This is key as we must obtain maximum coverage from minimal test cases, which equivalence classes help to achieve.

#### 6.1.2.3 Combinatorial testing – Category partition method

EP/BVA combination is weak at testing combinations of inputs. This is a significant downfall for this project specifically as there is such a range of combinations available for the user to input. From selecting countries and each country having their own specific data in the algorithm and outcome. To overcome this shortfall in the testing strategy combinatorial testing will be used to generate testcases as well (Myers, 1979). Pairwise testing is a combinatorial testing method excellent at covering a lot of conditions with minimal cases however, due to the large number of supported countries, category partition method will be used to generate more of the test cases in Appendix H. Category partition method allows you to specifically identify areas most likely to reveal faults, by excluding combinations that are impossible or unlikely (Ostrand and Balcer, 1988). So, for this instance a combination of 12 destinations was not tested, because that many places would be unnecessary for the user.

### 6.1.3 White box testing

White box testing techniques are based upon someone having working knowledge of the code and using this knowledge to their advantage in writing test cases. White box test techniques allow us to lessen reliance on requirements specification by focusing in on the structure and technical side of the program. This is very useful for this project as there is no client provided specification.

Another benefit of white box testing is knowing when to finish testing. Testers are not afforded unlimited resources and so a termination factor must be set to ensure time is not being wasted on testing. One such method for this is test coverage. 100% test coverage has been reached when every line of code has been executed at least once, this can help develop confidence in the code (Myers, 1979).

#### 6.1.3.2 Branch Testing

Branch testing is the process of executing each conditional statement for the conditions of both true and false, at least once. This is much more effective than statement testing which would not account for both conditions and only test once. Additionally, Branch testing implies statement testing therefore if every branch has been covered then 100% test coverage can be achieved (Kaner et al, 1993). Branch and statement testing are weak in some ways as certain combinations can be unidentifiable (Myers, 1979). This weakness is limited by the projects use of combinatorial testing techniques. Branch testing was also ideal for this project due to a very large switch case statement contained in the JavaScript file, making it necessary to ensure that all the branches in the switch statement have been covered.

#### 6.1.3.3 Coverage

661 statements in solution from 3 files / 661 statements covered \* 100 = 100% coverage

31 branch statements in solution / 31 branches covered \* 100 = 100% coverage

## 6.2 Static Code Analysis

Static code analysis has been viewed by many as a poor method for finding errors and proving quality of software. This is because the code is not analysed with inputs present. While there may be truth to not being able to attain confidence in software quality through static code analysis alone, it can be still a useful tool for quickly finding many types of errors that can be easily missed with the human eye. These errors include but are not limited to variables that haven't been initialised and recursive loop functions etc (Ayewah et al., 2008). As evolutionary algorithms perform recursion until the termination condition has been met this is a very valuable process in the context of this dissertation. Visual Studio now has a built in static code analyser in the form of FxCop, therefore it was beneficial to run it on my code to make sure there were no simple errors missed that might have caused confusion before testing began in terms of locating errors. The results of this static code analysis can be seen in Appendix J.

## Chapter 7 – Evaluation and Conclusion

This chapter will serve as a final summary for the author and reader to understand and evaluate the projects strengths and weaknesses. This chapter will also conclude the main body of this dissertation by providing a closing statement on findings and future improvements.

### 7.1 Success Criteria

The success criteria defined in chapter 1, determined that for this to be a success all the must have requirements shall be met for the web application. Must have requirements are shown in Appendix E and were prioritised using MoSCoW. The prioritisation process was excellent at distinguishing features that were essential to the artefact from those that could be sacrificed so that the project could meet the deadline. Overall this project can be considered a success since all must have requirements were met in this process with the addition of one should have, this is proven by the testing performed in Appendix H as it shows the must have features are functionable.

Other than the success criteria it was decided that to meet the main aim of the project, that three deliverables would need to be produced.

1. Deliverable 1 - A web application that allows the user to set and alter parameters of the algorithm interactively, whilst also displaying the results of the algorithm.
2. Deliverable 2 – An evolutionary algorithm backend that supports the GUI and optimises the user's parameters.
3. Deliverable 3 - Software documentation that clearly lays out a logical description of the process used to develop the project and artefact throughout the whole SDLC from research to Requirements and finally all the way through to testing and evaluation.

## 7.2 Personal Evaluation

Though this project has been very challenging it has provided an excellent platform for learning and understanding a more niche field of artificial intelligence. Through the undertaking of this research the author has also gained valuable skills that are translatable into a working environment, such as how to conduct a formal literature review.

The research was aided by the fact that the concept was the authors original idea. This meant the dissertation was not as constrained or limited like a client specific project. It is worth noting that a client specific project also has merits as it would have been beneficial for adapting to an industry development role. Having said this, the project still traverses through the whole SDLC and shows an in depth understanding of each stage of the SDLC. Furthermore, beyond the SDLC the author has shown the ability to take an idea all the way from conception to a releasable product. This has been crucial to showing me how various Software Engineering skills must be applied to the situations and not just memorised in an academic environment.

## 7.3 Conclusion

Overall the main aim as stated in chapter 1 was to investigate how evolutionary algorithms can be applied to a travel planning solution, to better the experience of travelling. The solution overall saves a huge amount of time when compared to the user planning a route for themselves. In the test plan in Appendix H cases 16 – 25 were used to make an average time for the algorithm to run, with 3 destinations and 4. Across 5 test cases the average time for the algorithm to run for 3 destinations came to 2 minutes and 14 seconds, and the average time for 4 destinations came to 4 minutes 37 seconds. A study has shown that on average 38 comparison websites are used by travel planners to find the cheapest prices (Expedia group media solutions, 2018).



Currently around 37% of travellers end up cancelling their trip and not completing the booking due to information overload (Wyndham Vacation Rentals, 2018). The short algorithm run time can serve a huge purpose in the future in terms of lowering the number of websites needed to find the cheapest prices, by replacing all the other flight comparison websites. This solution therefore, makes travel planning much more convenient by consolidating the amount of pre-travel planning that is necessary down from 38 websites. This will lead to many less people cancelling their trips and improve the overall experience of travel planning.

## 7.4 Future Work

At the beginning of the project there were many ideas and concepts I wanted to bring to the deliverables. The ideas and concepts that didn't make it into this iteration of the software will now be presented to the reader.

### 7.4.2 Deliverables 1 & 2

One of the largest drawbacks of the first two deliverables, particularly deliverable two, is the API used to gain flight data. Commercial flight data is very expensive, finding a free source of this proved to be very challenging. A potential lack of data was identified in my Risk Analysis which can be found in Appendix F. To alleviate this risk the author applied to Skyscanner for access to their API before the beginning of the project.

Skyscanner would have been an ideal API for the implementation as it has a wealth of flight data pulled in from major sources (Skyscanner.github.io, 2018). Also, it is free which is important for this project as no external funding or budget was applied for or received during the development. Unfortunately, due to the vast volume of requests Skyscanner receives for access to their resources I couldn't obtain an API developer key.

Due to the lack of response from Skyscanner the project used a free commercial flight data source called travel pay-outs. This API was excellent at providing free easily accessible data to develop a base solution. However, if the solution was to be improved beyond this project's deadline, it would be necessary for multiple API's to be used for flight data. Currently the program has an issue that for some destinations the data does not simply exist in travel pay-outs database.

On top of this if many destinations are selected the algorithm can have a very long response time. While it is still far quicker than a manual user search, the lack of progress indicators could lead to user frustration. In the future to reduce this something like a progress bar would be very beneficial, so that the user is aware the solution has not crashed and that the processing is not currently completed. Furthermore, towards the end of the project shortcomings have been identified that could potentially improve algorithm run time.

A potential shortcoming is the chosen implementation language. The reason C# was chosen was due to the excellent integration with asp.net and the genetic algorithm libraries such as GAF that were discovered during research. Additionally, C# is a compiled language which is beneficial because the algorithms are CPU based and have a longer running time. However, the object orientated approach to the algorithm may have negatively affected run time as some data was forced to be programmed as global variables as opposed to being passed as parameters as is the traditional OOP method. This is due to the interaction between server-side C# and client-side JavaScript. This has a higher probability of leading to memory leaks which can cause the run time to increase further. A non-object orientated language such as python still has numerous libraries supporting genetic algorithm development but does not suffer as badly in terms of memory allocation.

Another way to reduce the algorithm run time would have been to change the algorithms termination conditions to run less generations, however this would be at the cost of optimality and so is a user trade off.

### 7.4.3 Deliverable 3

The documentation of this project has been implemented to a clear and professional standard. The documentation was able to keep to key timings throughout the timeframe of the project due to the Gantt chart in Appendix C. The chart was very beneficial at maintaining a continuous feedback loop between author and supervisor. The software documentation will prove beneficial for bringing other potential developers up to speed on the project, when future work is being carried out on completion of the dissertation. As well as this the documentation can also demonstrate the progress made with the solution and may be able to support an application for certain companies like Skyscanner to endorse this type of project with either free database usage or funding.

Word count in the main body: 9980

## References

- 2017 4<sup>th</sup> International Conference on Systems and Informatics(ICSAI). (2017). Selection of requirement elicitation techniques using laddering 2017. IEEE Xplore Digital Library, EBSCOhost,[Accessed February 2, 2018]
- amCharts (2018). *JavaScript Maps - amCharts*. [online] Available at: <https://www.amcharts.com/javascript-maps/> [Accessed 13 Apr. 2018].
- App Developers UK | Mobile App Development | The Distance*. (2018). *The Agile App Development Process | The Distance*, York. [online] Available at: <https://thedistance.co.uk/agile-app-development-process/> [Accessed 6 Apr. 2018].
- Back, T. (1996). *Evolutionary algorithms in theory and practice*. New York [u.a.]: Oxford Univ. Press.
- Beck, K. (2000). *Extreme programming explained*. Boston [etc.]: Addison-Wesley.
- Beizer, B. (1995). *Black-box testing*. New York: J. Wiley.
- Boehm, B. and Papaccio, P. (1988). Understanding and controlling software costs. *IEEE Transactions on Software Engineering*, 14(10), pp.1462-1477.
- Burkhard, S., Kow, N. and Fuggle, L. (2017). Travel trend report. [online] Available at: <https://www.treksoft.com/en/blog/7-travel-trends-for-2017-that-will-drive-the-global-tourism-industry> [Accessed 2 May 2018].
- Charette, R. (1989). *Software engineering risk analysis and management*. New York: McGraw-Hill Book Company.
- Cockburn, A. (2009). *Agile software development*. Upper Saddle River, NJ: Addison-Wesley.
- Coello Coello, C. (2014). Evolutionary algorithms for solving multi -objective problems. [Place of publication not identified]: Springer.
- Craddock, A., Fazackerley, B., Messenger, S., Roberts, B. and Stapleton, J. (2008). DSDM Atern.
- Darwin, C. (1859). *The Origin of Species*.
- Deb, K. (2008). *Multi-objective optimization using evolutionary algorithms*. Chichester: John Wiley & Sons.
- Dictionary,t.(2018). Tourism Meaning in the Cambridge English Dictionary. [online] Dictionary.cambridge.org. Available at: <https://dictionary.cambridge.org/dictionary/english/tourism> [Accessed 23 Mar. 2018].
- Dictionary.com. (2018). *the definition of permutation*. [online] Available at: <http://www.dictionary.com/browse/permutation> [Accessed 9 Apr. 2018].
- Emirates. (2018). Emirates flights – Book a flight, browse our flight offers and explore the Emirates Experience. [online] Available at: <https://www.emirates.com/uk/english/> [Accessed 13 Mar. 2018].

- Endler, J. (1986). *Natural selection in the wild*. Princeton: Princeton University Press.
- European Environment Agency. (2001). *Tourism travel by transport modes*. [online] Available at: <https://www.eea.europa.eu/data-and-maps/indicators/tourism-travel-by-transport-modes> [Accessed 23 Mar. 2018].
- Expedia group media solutions (2018). *Travellers Attribution*. [online] Expedia groupd. Available at: [https://info.advertising.expedia.com/hubfs/Content\\_Docs/Premium\\_Content/pdf/Attribution\\_Study\\_Final\\_Media\\_Kit-2017-09.pdf?t=1524950664167](https://info.advertising.expedia.com/hubfs/Content_Docs/Premium_Content/pdf/Attribution_Study_Final_Media_Kit-2017-09.pdf?t=1524950664167) [Accessed 1 May 2018].
- Fisher, W. (1958). On Grouping for Maximum Homogeneity. *Journal of the American Statistical Association*, 53(284), p.789.
- Flanagan, D. (2011). *JavaScript*. Beijing: O'Reilly.
- Fogel, D. (2006). *Evolutionary computation*. Hoboken, N.J: Wiley.
- Fogel, D., Bäck, T. and Michalewicz, Z. (2000). *Evolutionary computation*. Bristol: Institute of Physics Publishing.
- Gen, M. and Cheng, R. (1997). *Genetic algorithms and engineering design*. New York: Wiley.
- Goldberg, D. (2012). *Genetic algorithms in search, optimization, and machine learning*. Boston [u.a.]: Addison-Wesley.
- Gov Department of Education (2012). *Gap year takers: uptake, trends and long term outcomes*. [online], p.28. Available at: [https://www.gov.uk/government/uploads/system/uploads/attachment\\_data/file/219637/DFE-RR252.pdf](https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/219637/DFE-RR252.pdf) [Accessed 29 Jan. 2018].
- Hamill, M. and Goseva-Popstojanova, K. (2009). *Common Trends in Software Fault and Failure Data*. *IEEE Transactions on Software Engineering*, 35(4), pp.484-496.
- Jacob, C. (2001). *Illustrating evolutionary computation with Mathematica*. San Francisco: Morgan Kaufmann.
- Jain, L., Dumitrescu, D., Dumitrescu, A. and Lazzerini, B. (2000). *Evolutionary Computation*. Florida: CRC PRESS.
- Kaner, C.K, Falk, J.F, Nguyen, H.Q.N, 1993. *Testing computer software (Second edition)*. Place of publication: VAN NOSTRAND REINHOLD
- Karlesky, M. and Voord, M. (2008). *Agile Project Management (or, Burning Your Gantt Charts)*. *Embedded Systems Conference Boston (Boston, Massachusetts)*.
- Kotonya, G. and Sommerville, I. (2004). *Requirements engineering*. Chichester: John Wiley & Sons.
- Langer, A. (2012). *Guide to software development*. London: Springer.
- Ling, C. (2014). A Study on Reasons and Solutions to Tour Guides' Ripping Off Tourist. *American Journal of Industrial and Business Management*, 04(02), pp.90-93.

- Mahalakshmi, M. and Sundararajan, D. (2013). *Traditional SDLC Vs Scrum Methodology – A Comparative Study. International Journal of Emerging Technology and Advanced Engineering*, 3(6).
- Mantyla, M. and Lassenius, C. (2009). *What Types of Defects Are Really Discovered in Code Reviews?. IEEE Transactions on Software Engineering*, 35(3), pp.430-448.
- Medina-Muñoz, R., Medina-Muñoz, D. and García-Falcón, J. (2003). Understanding European tour operators' control on accommodation companies: an empirical evidence. *Tourism Management*, 24(2), pp.135-147.
- Meisel, W. and Michalopoulos, D. (1973). A Partitioning Algorithm with Application in Pattern Classification and the Optimization of Decision Trees. *IEEE Transactions on Computers*, C-22(1), pp.93-103.
- Messac, A., Ismail-Yahaya, A. and Mattson, C. (2003). The normalized normal constraint method for generating the Pareto frontier. *Structural and Multidisciplinary Optimization*, 25(2), pp.86-98.
- Mitchell, M. (2002). *An introduction to genetic algorithms*. New Delhi: Prentice-Hall of India.
- Myers, G.J.M., 2 May 1979. *The art of software testing*. Place of publication: A WILEY-INTERSCIENCE PUBLICATION.
- N. Mudaliar, D. and K. Modi, D. (2013). *Unravelling Travelling Salesman Problem by Genetic Algorithm Using M-Crossover Operator*. [online] Available at: (<http://ieeexplore.ieee.org/document/6497974/?reload=true>) [Accessed 31 Jan. 2018].
- O. Galitz, W. (2007). *The Essential Guide to User Interface Design*. Wiley.
- Obitko.com. (2018). *Selection - Introduction to Genetic Algorithms - Tutorial with Interactive Java Applets*. [online] Available at: <http://www.obitko.com/tutorials/genetic-algorithms/selection.php> [Accessed 23 Mar. 2018].
- Ostrand T.J. & Balcer, M.J., 1988. *The Category-Partition Method for Specifying and Generating Functional Tests. Comms. ACM [online]*, 31 (6), June 1988, pp676-686.
- Papadimitriou, C. and Steiglitz, K. (1998). *Combinatorial optimization*. Mineola, N.Y.: Dover Publications.
- Pareto, V. (1896). *Cours d'économie politique professé à l'Université de Lausanne*. 2 vol. Lausanne: Rouge.
- Parnas, D. (1994). *Software aging. Proceedings. ICSE-16., 16th International Conference on*.
- Patch, N. (2018). *Travel agent vs do it yourself bookings - CHOICE*. [online] CHOICE. Available at: <https://www.choice.com.au/travel/on-holidays/advice/articles/travel-agent-vs-do-it-yourself> [Accessed 5 Mar. 2018].
- PENG, D., CAO, L. and XU, W. (2011). Using JSON for Data Exchanging in Web Service Applications. *Journal of Computational Information Systems* 7: 16 (2011) 5883-5890.
- Potvin, J. (1996). Genetic algorithms for the traveling salesman problem. *Annals of Operations Research*, 63(3), pp.337-370.
- Pressman, R. and Maxim, B. (2015). *Software engineering*. New York: McGraw-Hill.

- Pryor, Alan N. A Discrimination of Software Implementation Success Criteria,(August 1999). Denton, Texas. University of North Texas Libraries, Digital Library, [digital.library.unt.edu](http://digital.library.unt.edu); [Accessed February 2, 2018]
- Rao, S. (1984). Engineering optimization. Wiley.
- Rostami, S., O'Reilly, D., Shenfield, A. and Bowring, N. (2015). A novel preference articulation operator for the Evolutionary Multi-Objective Optimisation of classifiers in concealed weapons detection. *Information Sciences*, 295, pp.494-520.
- RoutePerfect. (2018). [online] Available at: <https://www.routeperfect.com/trip-planner> [Accessed 3 Mar. 2018].
- RouteYou.(2018). Outdoorrouteplanner.[online]Availableat:  
<https://www.routeyou.com/route/planner/0/outdoor-route-planner> [Accessed 3 Mar. 2018].
- Rubicite.com. (2018). Order 1 Crossover Operator Tutorial. [online] Available at: <http://www.rubicite.com/Tutorials/GeneticAlgorithms/CrossoverOperators/Order1CrossoverOperator.aspx> [Accessed 10 Apr. 2018].
- Rudolph, G. (1996). Convergence of evolutionary algorithms in general search spaces. *Proceedings of the Third IEEE Conference on Evolutionary Computation*, pp.50-54.
- S. Esquivel, R. Gallard, Z. Michalewicz, (1995). "Another Approach to Crossover in Genetic Algorithms", *Proceeding of Primer Congreso Argentino de Ciencias de la Computación*, pp. 141-150.
- Schwaber, K. and Beedle, M. (2002). *Agile software development with Scrum*. Upper Saddle River (New Jersey): Prentice Hall.
- Shore, J. and Warden, S. (2008). *The Art of Agile Development*. Sebastopol: O'Reilly Media, Inc.
- Skyscanner.github.io. (2018). API Reference. [online] Available at: <https://skyscanner.github.io/slate/> [Accessed 25 Apr. 2018].
- Software testing. Vocabulary. (1998). BSI.*
- Spall, J. (2005). Introduction to Stochastic Search and Optimization. Hoboken: Wiley.
- Statista. (2016). International inbound tourism by mode of transport 2016 | Statistic. [online] Statista. Available at: <https://www.statista.com/statistics/305515/international-inbound-tourism-by-mode-of-transport/> [Accessed 23 Mar. 2018].
- Tech.io. (2018). The free knowledge-sharing platform for technology. [online] Available at: <https://tech.io/playgrounds/334/genetic-algorithms/tools> [Accessed 23 Mar. 2018].
- Thiele, L., Miettinen, K., Korhonen, P. and Molina, J. (2009). A Preference-Based Evolutionary Algorithm for Multi-Objective Optimization. *Evolutionary Computation*, 17(3), pp.411-436.
- UNWTO, WYSE (2016). The global report on the power of youth travel. [online] Madrid, Spain:UNWTO,p.10.Available at:[https://www.wysetc.org/wp-content/uploads/2016/03/Global-Report\\_Power-of-Youth-Travel\\_2016.pdf](https://www.wysetc.org/wp-content/uploads/2016/03/Global-Report_Power-of-Youth-Travel_2016.pdf) [Accessed 29 Jan. 2018].

Van Lamsweerde, A. (2013). Requirements engineering. Chichester [u.a.]: Wiley.

Venter, J. (2001). The sequence of the human genome. Washington, DC: American Association for the Advancement of Science.

Visual Studio. (2018). [online] Available at: <https://docs.microsoft.com/en-us/vsts/git/tutorial/creatingrepo?view=vsts&tabs=visual-studio> [Accessed 13 Apr. 2018].

Williams, G. (1996). Adaptation and Natural Selection: A Critique of Some Current Evolutionary Thought (Princeton science library). Princeton University Press.

Wyndham Vacation Rentals. (2018). *Why Stress About Your Next Trip?*. [online] Available at: <https://www.wyndhamvacationrentals.com/vacation-burn-out> [Accessed 2 May 2018].

Yu, Y., Qian, C. and Zhou, Z. (2015). Switch Analysis for Running Time Analysis of Evolutionary Algorithms. IEEE Transactions on Evolutionary Computation, 19(6), pp.777-792.

Zitzler, E., Deb, K. and Thiele, L. (1999). Comparison of multiobjective evolutionary algorithms: empirical results. Zürich: Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology Zürich (ETH).



## Appendix A – Project Proposal

**Undergraduate Project Proposal Form**

Please refer to the **Project Handbook Section 4** when completing this form

<b>Degree Title:</b>  Software Engineering	<b>Student's Name:</b>  Joel Holmes
	<b>Supervisor's Name:</b>  Dr Shahin Rostami
	<b>Project Title/Area:</b>  Using evolutionary algorithms to best optimise travel planning.

**Section 1: Project Overview****1.1 Problem definition - use one sentence to summarise the problem:**

In 2016 the US generated 7.6 trillion dollars, and 292 jobs from travel and tourism (World Travel and Tourism Council). Travelling is very popular amongst the younger members of the world population between the ages of 18- 25, however they mostly travel on limited budgets. Deciding upon and optimising a travel route can become tiresome, with so many factors to consider after deciding upon destinations it can be difficult to know where to start.

**1.2 Background - please provide brief background information, e.g., client:**

Once people have decided upon their destinations, choosing a route through them can be an entirely new challenge. Many factors must be considered such as flight duration, price of tickets and much more. Currently the main way of getting the best optimised route for travelling is either, paying expensive fees for a company like STA to give you a predefined route and set of activities which may not all be to your liking, or going on websites like sky scanner, and blindly selecting dates to get the best deals.

Evolutionary algorithms are excellent at providing solutions for optimisation problems, and could really provide an impact to get people out of the STA pit fall, allowing you to personalise your travel experience without having to compromise with poor price decisions. My solution is a web application that allows you to select x amount of destinations, and a certain length of time you wish to travel for. Then the application will plot the cheapest and shortest route based upon total flight duration, prices and how long you want to spend in each destination.

### **1.3 Aims and objectives – what are the aims and objectives of your project?**

The aim of my project, and artefact will be to use genetic algorithms to best optimise a route of travel for a person, based upon limiting factors mentioned above such as budget. I will be doing this so people planning to travel can have the best experience without having to waste funds. I personally know how difficult it can be to figure out what route to take through your desired destinations as I went travelling last summer myself, therefore I have a first-hand perspective of the issue.

#### **Research objectives**

My research aims will be achieved by fulfilling the following objective criteria:

1. The users will be able to select destinations, and how long they wish to spend in each destination on a GUI.
2. A genetic algorithm of my own design will use data of flight prices, and duration pulled in from API's and data stores, along with user entered parameters, such as length of trip, to configure an optimal route.
3. The optimal travel route will be displayed to the user with a clean and easy to use GUI. Although targeted at younger generations usually familiar with most forms of technology, I do not want to exclude people who are not skilled with technology from the use of this solution.

4. The solution will allow the user to save their Trip for reviewing and editing later.

## Section 2: Artefact

### **2.1: What is the artefact that you intend to produce?**

The artefact for this project will be an evolutionary algorithm I will write, that will have its fitness based upon flight duration, and flight cost. For example, the shorter the flight duration or the lower the flight cost then the higher the fitness level, and the more optimal the solution for the algorithm to select to “Reproduce”. Furthermore, I will be implementing a web solution for the users to pick their destinations on a GUI and then display the optimal route for them on this GUI.

### **2.2 How is your artefact actionable (i.e., routes to exploitation in the technology domain)?**

My artefact will have use because, currently travel websites such as (TripHobo) allow you to select routes and choose the cheapest flights based upon dates preselected by the user. However, nothing currently exists that will allow you to pick for example 4 destinations: South Korea, Japan, Thailand and Australia, and then suggest it is more efficient, and cost effective based upon flight time and flight prices to in fact go to Australia, Japan, South Korea and then Thailand before coming home, as well as picking the cheapest dates for you automatically all in one. A lot of the current process is simply trying to decide the best route yourself where certain dates, and prices are concerned. From my own research of finding nothing like this on the market I feel like there is a real niche for this type of solution powered by genetic algorithms which are excellent at solving optimisation problems which humans either cannot or don't have the time too due to a ridiculous large number of variables and factors to consider and control.

## Section 3: Evaluation

### **3.1 How are you going to evaluate your work?**

I will be able to evaluate my work by testing with methods, such as functional, structural, and inviting people to beta test my solution. Testing will help to evaluate whether the solution performs these tasks for them while doing an efficient and effective job. Evolutionary algorithms can take a lot of time to process due to being complex and requiring high computing power, however the time saved by not having to perform these actions themselves will be worth the time taken to do the search itself, and the times will be compared to studies on time spent booking vacations.

### **3.2 Why is this project honours worthy?**

My project proposal is honours worthy because it involves the use, and showcase of the many skills I've gained since the start of my degree. From programming the GUI of the solution to implementing my own algorithm and even using BU's Harvard referencing system, these skills have a range of depth and difficulty to them. The project I have chosen will allow me to prove that not only have I nurtured the skills I have attained in my three years at BU, but also that I can apply them to a scenario not given to me by an academic teacher at the university, and instead a task I want to achieve myself.

### **3.3 How does this project relate to your degree title outcomes?**

This project relates to my degree title of software engineering, because it will allow me to demonstrate a logical, and unambiguous approach when it comes to writing a specification/criterion for my solution that I can adhere to when developing this project. I will also be performing testing on my solution, along with the algorithms. The testing will bring in attributes gained from my software quality and testing module with Gernot, such as Beta testing, functional testing, and structural testing. My software quality and testing module will also allow myself to be able to approach my own work in two different states of mind, one state constructive for developing, and a second state destructive for testing my program with an unbiased approach. My artefact for this project will be the producing of the front-end solution, as well as the implementation of my own algorithm which both classify in the project handbook appendix A as build artefacts.

### **3.4 How does your project meet the BCS Undergraduate Project Requirements?**

My project meets the BCS undergraduate project requirements because as stated within appendix C in the project handbook under BCS criteria I must show ability to apply practical and analytical skills learnt throughout the program. Designing and implementing my own algorithm, as well as solution, will demonstrate my practical skills. Using testing to rigorously prove my program does what it should

do, while also not doing what it shouldn't will display my analytical skills. Balancing both the algorithm and GUI while also leaving enough time to adequately test the solution will demonstrate effectively my planning and time management skills.

### 3.5 What are the risks in this project and how are you going to manage them?

Risk	Probability	Severity	Effect on project	Reduction Actions
May be difficult to obtain data from some companies such as flight prices and so forth from other countries or languages.	Medium	Medium	Lack of data to test my algorithms with could result in poor proof of working concept and could leave potentially more unknown errors within the algorithm or solution.	Start early to search out lots of data stores and API's to connect with and pull this data from.
Coding errors in my algorithm design or biased logic	Medium	High	The algorithms will product incorrect output data with may cause us to incorrectly infer something to do with the performance of the algorithm	Adhering to simple coding standard practices such as indentation and other such methods will allow code to be more organised and errors therefore will be far easier to identify
There may be biases in the input data used for training for example the training data could be insufficiently large or incomplete.	Low	High	If there is mismatch in the training data used it could lead to more incorrect interpretations of the output data and be less useful in providing a performance indicator	To minimise this each data set will have a minimum limit set and will have a verified source

## Section 4: References

### 4.1 Please provide references if you have used any.

1- World Travel and tourism council, 1991-2017, wtcc[online], Available from <https://www.wttc.org/-/media/files/reports/economic-impact-research/regions-2017/world2017.pdf>  
[Accessed on 19/11/2017]

2 – TripHobo, 2017, triphobo[online], Available from <https://www.triphobo.com/tripplanner>  
[Accessed on 19/11/2017]

3 – Clever Algorithms, 2015-2017, cleveralgorithms[online], Available from <http://www.cleveralgorithms.com/nature-inspired/evolution.html>  
[Accessed on 19/11/2017]

## Section 5: Ethics (please delete as appropriate)

### 5.1 Have you submitted the ethics checklist to your supervisor?

Yes

### 5.2 Has the checklist been approved by your supervisor?

Yes / No

## Section 6: Proposed Plan (please attach your Gantt chart below)

Task	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Week 15	Week 16
Research Algorithm differences in developing																
Implement Algorithms																
Design GUI																
Test GUI																
Test algorithms efficiency of providing optimal travel route																
Beta test solution																
Review beta test results																
Make adjustments based upon beta test comments																
Evaluation																
Project defence																



## Appendix B – Ethics

**1. Student Details**

<b>Name</b>	Joel Holmes
<b>School</b>	Faculty of Science & Technology
<b>Course</b>	BSc Software Engineering
<b>Have you received external funding to support this research project?</b>	No
<b>Please list any persons or institutions that you will be conducting joint research with, both internal to BU as well as external collaborators.</b>	

**2. Project Details**

<b>Title</b>	Using evolutionary algorithms to best optimise travel planning.
<b>Proposed Start Date</b>	29-January-2018
<b>Proposed End Date</b>	31-August-2018 – <b>Note this is not the submission deadline!</b>
<b>Supervisor</b>	Shahin Rostami

<b>Summary (including detail on background methodology, sample, outcomes, etc.)</b>
<p>The outcome of this project is to provide a solution that is efficient, and effective at providing necessary help where travel planning is concerned using evolutionary algorithms to give people a better understanding of the trip they wish to embark on. During the project I have planned involvement of humans with concerns to beta testing my solution, however no sensitive information about these persons will be required to be stored throughout the process, and only information concerned with my project will be used such as possible changes, comments or errors.</p>

### 3. External Ethics Review (Answer “Yes” go to 4, “No” go to 5)

<b>Does your research require external review through the NHS National Research Ethics Service (NRES) or through another external Ethics Committee?</b>	No
---	----

### 4. External Ethics Review Continued

<p>Answered “Yes” to question 3 will conclude the BU Ethics Review so you do not need to answer the following questions. Note you will need to obtain external ethical approval before commencing your research.</p>
--

### 5. Research Literature (Answer “Yes” go to 6, “No” go to 7)

<b>Is your research solely literature based?</b>	No
--	----

## 6. Research Literature Continued (Either answer will conclude the review)

Will you have access to personal data that allows you to identify individuals OR access to confidential corporate or company data (that is not covered by confidentiality terms within an agreement or by a separate confidentiality agreement)?	No
Describe how you will collect, manage and store the personal data (taking into consideration the Data Protection Act and the Data Protection Principles).	

## 7. Human Participants Part 1 (Answer “Yes” go to 8, “No” go to 12)

Will your research project involve interaction with human participants as primary sources of data (e.g. interview, observation, original survey)?	No
---	----

## 8. Human Participants Part 2 (Answer any “Yes” go to 9)

Does your research specifically involve participants who are considered vulnerable (i.e. children, those with cognitive impairment, those in unequal	No
--	----

relationships—such as your own students, prison inmates, etc.)?	
Does the study involve participants age 16 or over who are unable to give informed consent (i.e. people with learning disabilities)? NOTE: All research that falls under the auspices of the Mental Capacity Act 2005 must be reviewed by NHS NRES.	No
Will the study require the co-operation of a gatekeeper for initial access to the groups or individuals to be recruited? (i.e. students at school, members of self-help group, residents of Nursing home?)	No
Will it be necessary for participants to take part in your study without their knowledge and consent at the time (i.e. covert observation of people in non-public places)?	No
Will the study involve discussion of sensitive topics (i.e. sexual activity, drug use, criminal activity)?	No

## 9. Human Participants Part 2 Continued

Describe how you will deal with the ethical issues with human participants?

## 10. Human Participants Part 3 (Answer any “Yes” go to 11, all “No” go to 12)

Could your research induce psychological stress or anxiety, cause harm or have negative consequences for the participant	No
--	----

or researcher (beyond the risks encountered in normal life)?	
Will your research involve prolonged or repetitive testing?	No
Will the research involve the collection of audio materials?	No
Will your research involve the collection of photographic or video materials?	No
Will financial or other inducements (other than reasonable expenses and compensation for time) be offered to participants?	No

## 11. Human Participants Part 3 Continued

<p>Please explain below why your research project involves the above mentioned criteria (be sure to explain why the sensitive criterion is essential to your project's success). Give a summary of the ethical issues and any action that will be taken to address these. Explain how you will obtain informed consent (and from whom) and how you will inform the participant(s) about the research project (i.e. participant information sheet). A sample consent form and participant information sheet can be found on the Research Ethics website.</p>

## 12. Final Review

Will you have access to personal data that allows you to identify individuals OR access to confidential corporate or company data (that is not covered by confidentiality terms within an agreement or by a separate confidentiality agreement)?	No
Will your research take place outside the UK (including any and all stages of research: collection, storage, analysis, etc.)?	No
Please use the below text box to highlight any other ethical concerns or risks that may arise during your research that have not been covered in this form.	

**Review Completion Date: 10-November-2017 – Double click to change it!**

***The following section is to be filled by the supervisor only***

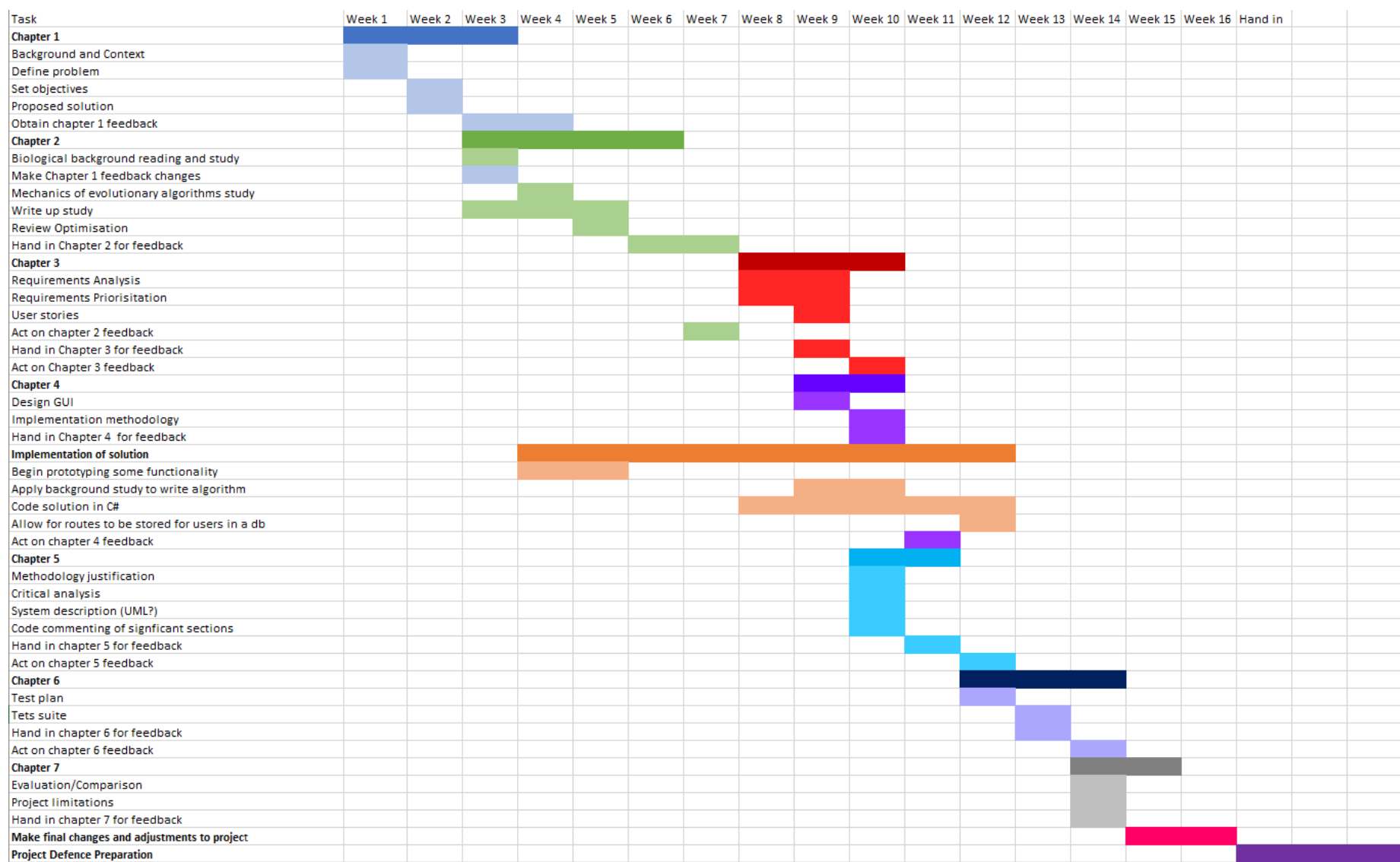
---

**Supervisor's Review:**

Choose an item.

**Please leave your comments:**

## Appendix C – Gantt Chart





## Appendix D – Code libraries referenced

Name	Description	Source
Newtonsoft Json	Extensive C# library that was used in this project to convert JSON strings received from the API into easily accessible objects.	<a href="https://www.newtonsoft.com/json">https://www.newtonsoft.com/json</a>
GAF	C# library package available by NuGet which acts as an assembly for genetic algorithms.	<a href="https://www.codeproject.com/Articles/873559/Implementing-Genetic-Algorithms-in-Csharp">https://www.codeproject.com/Articles/873559/Implementing-Genetic-Algorithms-in-Csharp</a>
AmCharts	Interactive chart and mapping source used in JavaScript	<a href="https://www.amcharts.com/javascript-maps/">https://www.amcharts.com/javascript-maps/</a>
Bootstrap	Html, CSS library that helps with quality factors such as portability and maintainability.	<a href="https://getbootstrap.com/">https://getbootstrap.com/</a>
Asp.net	Microsoft framework for web interface building	<a href="https://www.asp.net/">https://www.asp.net/</a>
TravelPayouts	API accessed for flight data	<a href="https://www.travelpayouts.com/dashboard">https://www.travelpayouts.com/dashboard</a>

## Appendix E – MoSCoW Prioritisation of Requirements

	Functionality	Requirements No
Must	Select destinations to generate a route (min 3)	RQ1
	Generate a route from desired destinations based on minimized constraints	RQ2
	Route adheres to dates selected by user	RQ3
	Portability – The solution can be used on multiple platforms such as; A variety of browsers, handheld devices etc.	RQ4
	Allows users to customize algorithm through user preference articulation	RQ5
Should	User login	RQ6
	User saving of routes to database	RQ7
	User loading of routes from the database	RQ8
	The System is very easy to use with very little to no guidance or training. This should be amplified through the design of the GUI which shall be simple.	RQ9
Could	Editing of saved routes	RQ10
	Export route information to excel or be printed	RQ11
	Linking to a booking website so that the generated route can be applied to the users travelling experience.	RQ12
Wont	Draw route on the map	RQ13
	Functionality for all countries shown on the map (will be done for most popular routes due to time considerations)	RQ14

## Appendix F – Risk Analysis

Risk	Probability (1-5)	Severity (1-5)	Effect on project	Actions	Action type
Hard drive failure	1	5	Loss of dissertation deliverables and documentation	All work will be saved to external USB and external cloud storage such as drop box	Prevention
M crossover proves to be ineffective and take too long to process data	3	2	Data will not be able to be processed to find optimized route	Can adapt and alter what type of crossover is used in the algorithm so that the data can still be processed effectively but also quickly	Acceptance
May be unable to create an algorithm that can process the data	2	4	Data will not be able to be processed to find optimized route	Can use pre-existing algorithm to process the data as will already have the software documentation and build as artefacts. So, the loss of an original algorithm will not affect my degree discipline artefact type according to the project handbook so long as I have a build	Acceptance
May be difficult to obtain data from some companies for flight prices and so forth	2	3	Lack of data to test the algorithms with could result in poor proof of working concept and potentially leave me with false positives with how the program behaves	Start early to find datastores and API's to connect with and pull this data from	Prevention
Programming proves to take longer than a third of the project (based on the fact the build is worth a third)	3	3	This could push back other important task such as documentation and testing, which could not only lead to some tasks being poor quality or not done at all	By adopting an agile methodology, I will be able to work more flexibly than opposed to a more iterative waterfall process which can be quite rigid. Due to the flexibility of the approach to the project I will be able to jump between tasks based on the level of prioritization I assign to them.	Mitigation
Computer power available to test algorithm is insufficient	2	4	If the algorithm cannot be tested and, working concept be proven then the artefact cannot be successfully evaluated against objectives and success criteria.	Multiple machines are available to me other than the machine used for development. If the algorithm proves to be too heavy for the development machine to process effectively then the code will be ported to a different computer to test the algorithm.	Acceptance

## Appendix G – Equivalence Classes

1. Number of Destinations is below the minimum (invalid)
2. Number of Destinations is a correct amount from 3 – max number of supported (valid)
3. Not supported Country is selected (invalid)
4. Supported Country is selected (valid)
5. Travel date specifying the start of a month is selected (valid)
6. Incorrect travel date specified such as a day in the month not the first chosen or incorrect format (invalid)
7. User input Weights for the algorithm are incorrect or missing (invalid)
8. User input Weights for the algorithm are correct and equal to 1 (valid)
9. After algorithm runs user is shown destinations entered and in the correct order (valid)
10. After algorithm is run the user is shown the total cost, distance and number of changes for their flight (valid)

## Appendix H – Test Cases

ID	Description	Input	Expected Outcome	Actual Outcome	Pass/Fail P/F	Comments
1	Test case to show valid EP classes to check the algorithm works under ideal circumstances.	<p>3 destinations UK, France, Germany. Date input – 01/05/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>	Algorithm will run and after a period the algorithm will display on the website the optimal permutation of destinations to visit and the total price, distance and number of changes associated with this trip.	<p>Order of travel: LON, PAR, BER</p> <p>Total Distance: 1217</p> <p>Total Changes: 3</p> <p>Total Price: 240</p> <p>Outcome was as expected</p>	P	
2	Follow on from test case 1 but this time using the opposite boundary of number of destinations and selecting the maximum number of possible destinations.	<p>Select every supported destination found in Appendix I.</p> <p>Date input – 01/05/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p>	It is likely that due to the large number of destinations selected and using only one API that not enough data for all the destinations will be able to be found.	Error thrown null value for data from Washington to Canada	F	Due to limited data availability from API unable to run, now added error validation so that if no flight data is available a message is shown to the user asking them to try and change the dates.

		Confirm destinations then click optimize				
3	Follow on from test case 2 error added more API strings so there is more json data to choose from and error message	<p>Select every supported destination found in Appendix I.</p> <p>Date input – 01/05/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p>	<p>It is likely that due to the large number of destinations selected and using only one API that not enough data for all the destinations will be able to be found.</p> <p>Error message will be displayed.</p>	<p>Error message correctly displayed for when no flight data is present.</p> <p>Fault from t test case 2 correctly fixed</p>	P	
4	Test case to test invalid EP class with no destinations selected.	<p>Date – 01/05/2018</p> <p>Weights – Price= 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations</p>	Error message should be thrown to the user explaining that an invalid number of destinations has been selected.	Error message was correctly displayed explaining to the user that minimum of 3 destinations must be selected	P	
5	Test Case to test invalid ep class with incorrect date input.	<p>3 destinations UK, France, Germany. Date input –</p> <p>Weights – Price = 0.5</p>	Error message should be thrown to the user explaining that an incorrect date has been selected.	Error message correctly displayed explaining the date is incorrect.	P	

		Distance = 0.3 Changes = 0.2  Confirm destinations then click optimize				
6	Invalid ep class test of unsupported country selected	3 destinations Peru, France, Germany. Date input – 20/05/2018  Weights – Price = 0.5 Distance = 0.3 Changes = 0.2  Confirm destinations then click optimize	Error message should be thrown to the user stating that unfortunately the country is not currently supported and to please check the current list of supported countries.	Country is correctly not loaded into the array of destinations however error message fails to show	F	Potential restructuring of JavaScript switch statement from line 55 required
7	Test for invalid weight ep class	3 destinations UK, France, Germany. Date input – 01/05/2018  Weights – Price = 0 Distance = 0 Changes = 0	Error message should be thrown to the user to explain that the weight parameters for the algorithm must first be set and that the values are currently invalid	Error message correctly display explaining to the user that the sum of the weights must be equal to 1	P	

		Confirm destinations then click optimize				
8	Test for invalid weight ep class follow on from case 6	<p>3 destinations UK, France, Germany. Date input – 20/05/2018</p> <p>Weights – Price = 1 Distance = 1 Changes = 1</p> <p>Confirm destinations then click optimize</p>	Error message should be thrown to the user to explain that the weight parameters for the algorithm must first be set to equal 1 and that the values are currently invalid	Error message didn't display	F	Boundary condition was set to below one only not to trigger on above, boundary condition now changed, and Error message correctly shows.
9	Follow on from test case 8 failure to test changes	<p>3 destinations UK, France, Germany. Date input – 20/05/2018</p> <p>Weights – Price = 1 Distance = 1 Changes = 1</p> <p>Confirm destinations then click optimize</p>	Error message should be thrown to the user to explain that the weight parameters for the algorithm must first be set to equal 1 and that the values are currently invalid	Error message now correctly displays fault from test case 8 is solved	P	



10	Test case to show 4 different combinations supported countries work	<p>4 destinations UK, Spain, Paris, Germany Date input – 01/05/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>	Algorithm will run and after a period the algorithm will display on the website the optimal permutation of destinations to visit and the total price, distance and number of changes associated with this trip.	Expected outcome generated	P	Algorithms run time increased from 2/3 minutes with 3 destinations to around 5 with 4 destinations
11	Test case to show 5 different combinations supported countries work	<p>5 destinations UK, Spain, Paris, Germany, Italy Date input – 01/05/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>	Algorithm will run and after a period the algorithm will display on the website the optimal permutation of destinations to visit and the total price, distance and number of changes associated with this trip.	<p>Output as expected Order: BER, ROM, PAR, LON , MAD Total price: 531</p> <p>Total Distance: 3899</p> <p>Total Changes: 0</p>	P	Algorithms run time increased from 2/3 minutes with 3 destinations to around 12 with 5 destinations
12	Testing different combinations of supported countries	3 destinations Tokyo, Germany, Australia	Algorithm will run and after a period the algorithm will display			

		<p>Date input – 01/05/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>	on the website the optimal permutation of destinations to visit and the total price, distance and number of changes associated with this trip.			
13	Test to see Capabilities in Google chrome	<p>3 destinations UK, France, Germany. Date input – 01/05/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>	Algorithm will run and after a period the algorithm will display on the website the optimal permutation of destinations to visit and the total price, distance and number of changes associated with this trip.	Outcome was as expected	P	
14	Test to see Capabilities in Microsoft Edge	<p>3 destinations UK, France, Germany. Date input – 01/05/2018</p> <p>Weights – Price = 0.5</p>	Algorithm will run and after a period the algorithm will display on the website the optimal permutation of destinations to visit and the total	Outcome was as expected	P	

		Distance = 0.3 Changes = 0.2  Confirm destinations then click optimize	price, distance and number of changes associated with this trip.			
15	Test to see Capabilities in Mobile Safari	3 destinations UK, France, Germany. Date input – 01/05/2018  Weights – Price = 0.5 Distance = 0.3 Changes = 0.2  Confirm destinations then click optimize	Algorithm will run and after a period the algorithm will display on the website the optimal permutation of destinations to visit and the total price, distance and number of changes associated with this trip.	Outcome was as expected	P	
<b>Tests for average time taken for Destinations 3,4,5,6</b>						
Test no	Description	Input	Expected	Time taken	Comments	
16	Test no 1 for average time taken for 3 destinations.	3 destinations UK, France, Germany. Date input – 01/06/2018  Weights – Price = 0.5 Distance = 0.3	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 2 minutes 47 seconds to run in its entirety.		

		Changes = 0.2  Confirm destinations then click optimize				
17	Test no 2 for average time taken for 3 destinations.	3 destinations UK, France, Germany. Date input – 01/06/2018  Weights – Price = 0.5 Distance = 0.3 Changes = 0.2  Confirm destinations then click optimize	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 2 minutes 06 seconds to run in its entirety.		
18	Test no 3 for average time taken for 3 destinations.	3 destinations UK, France, Germany. Date input – 01/06/2018  Weights – Price = 0.5 Distance = 0.3 Changes = 0.2  Confirm destinations then click optimize	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 2 minutes 01 seconds to run in its entirety.		

19	Test no 4 for average time taken for 3 destinations.	<p>3 destinations UK, France, Germany. Date input – 01/06/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 2 minutes 23 seconds to run in its entirety.		
20	Test no 5 for average time taken for 3 destinations.	<p>3 destinations UK, France, Germany. Date input – 01/06/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 1 minutes 54 seconds to run in its entirety.		
21	Test no 1 for average time taken for 4 destinations	<p>4 destinations UK, France, Germany, Madrid</p>	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 4 minutes 44		

		<p>Date input – 01/06/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>		seconds to run in its entirety.		
22	Test no 2 for average time taken for 4 destinations	<p>4 destinations UK, France, Germany, Madrid Date input – 01/06/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 4 minutes 42 seconds to run in its entirety.		
23	Test no 3 for average time taken for 4 destinations	<p>4 destinations UK, France, Germany, Madrid Date input – 01/06/2018</p>	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 4 minutes 53 seconds to run in its entirety.		

		<p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>				
24	Test no 4 for average time taken for 4 destinations	<p>4 destinations UK, France, Germany, Madrid Date input – 01/06/2018</p> <p>Weights – Price = 0.5 Distance = 0.3 Changes = 0.2</p> <p>Confirm destinations then click optimize</p>	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 4 minutes 34 seconds to run in its entirety.		
25	Test no 5 for average time taken for 4 destinations	<p>4 destinations UK, France, Germany, Madrid Date input – 01/06/2018</p> <p>Weights – Price = 0.5 Distance = 0.3</p>	Algorithm will run, and time taken will be recorded.	Algorithm ran and output correctly. It took 4 minutes 12 seconds to run in its entirety.		

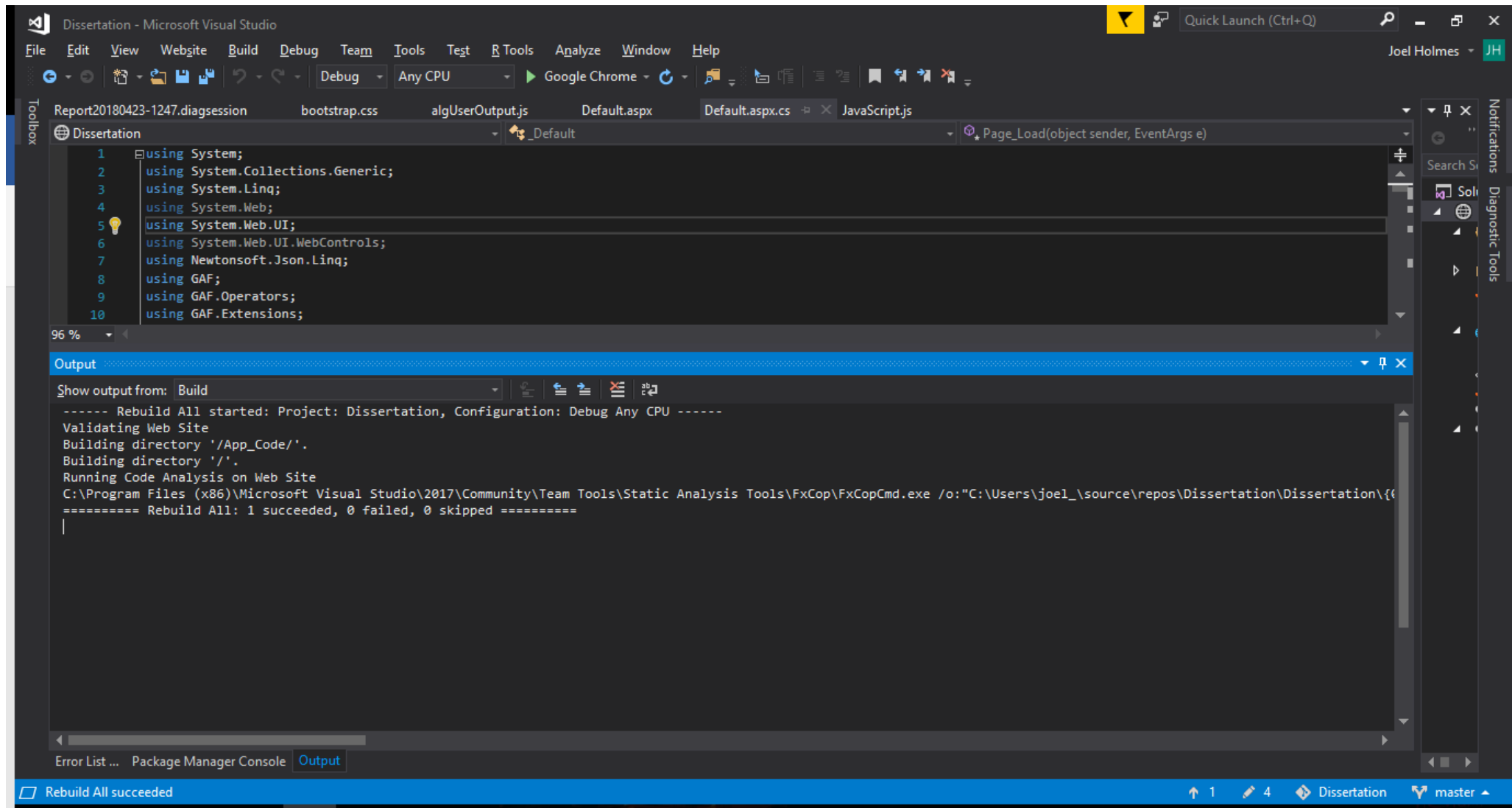
		Changes = 0.2  Confirm destinations then click optimize				
--	--	---	--	--	--	--



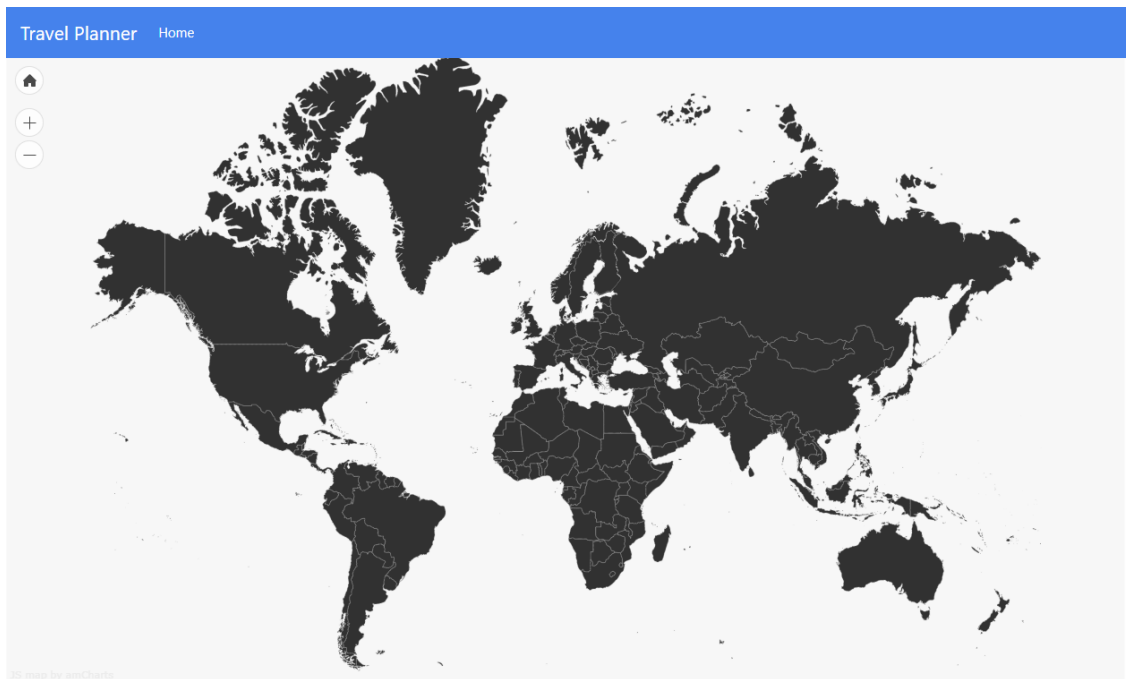
## Appendix I – List of supported countries

- United Kingdom
- France
- Germany
- Japan
- Australia
- Spain
- Portugal
- Ireland
- Italy
- Belgium
- Switzerland
- Netherlands
- Norway
- Sweden
- Denmark

## Appendix J – Static Code Analysis



## Appendix K – Screenshots of system



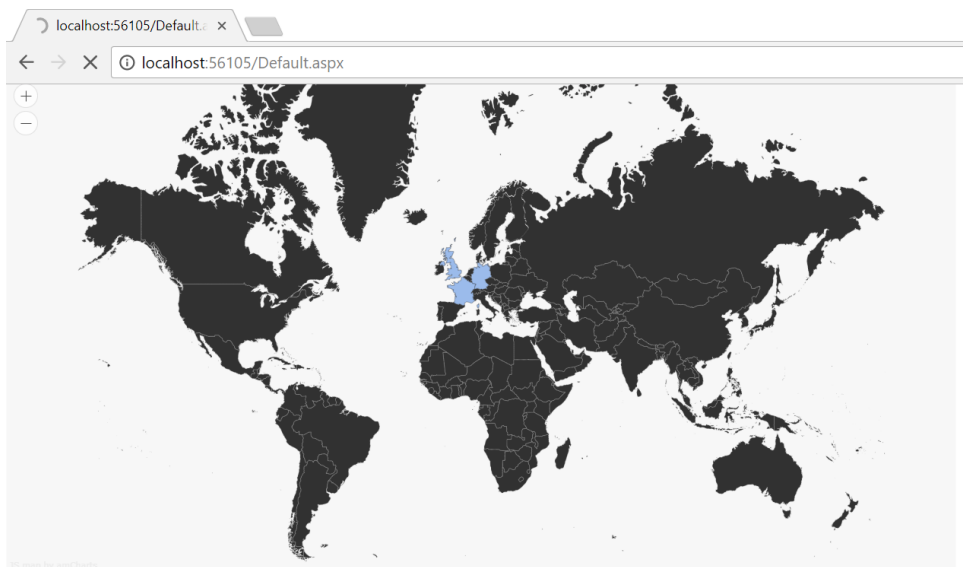
**Currently selected destinations:**

**Travel Start date (Must be the first day of specified month)**

dd/mm/yyyy

The larger the weights the higher prioritization on the variable in the algorithm

Please input 3 decimal values for each of the weights, the sum of the weights must be equal to 1



**Currently selected destinations:**

Germany, France, United Kingdom

**Travel Start date (Must be the first day of specified month)**

01/05/2018

The larger the weights the higher prioritization on the variable in the algorithm

Please input 3 decimal values for each of the weights, the sum of the weights must be equal to 1

Price 0.5

Distance 0.3

Number of changes 0.2


Confirm destinations

Optimise Route

Waiting for localhost...

localhost:56105/Default.aspx

localhost:56105/Default.aspx



Map by openStreetMap

**Currently selected destinations:**

**Travel Start date (Must be the first day of specified month)**

dd/mm/yyyy

The larger the weights the higher prioritization on the variable in the algorithm Please input 3 decimal values for each of the weights, the sum of the weights must be equal to 1

Price Distance Number of changes Confirm destinations Optimise Route

**Total Price of trip:**

152

**Total Distance:**

1258

**Total Changes:**

0

**Order of Travel:**

PAR,LON,BER