Hitch a Ride - Online Carpooling Application

Final Report for CS39440 Major Project

Author: Alex Roan (alr16@aber.ac.uk)

Supervisor: Fred Labrosse (ffl@aber.ac.uk)

10th April 2012

Version: 1.0 (Draft)

This report was submitted as partial fulfilment of a MEng degree in Software Engingeering (G601)

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Wales, UK

Declaration of originality

In signing below, I confirm that:

- This submission is my own work, except where clearly indicated.

- I understand that there are severe penalties for plagiarism and other unfair practice, which can lead to loss of marks or even the withholding of a degree.

- I have read the sections on unfair practice in the Students' Examinations Handbook and the relevant sections of the current Student Handbook of the Department of Computer Science.

- I understand and agree to abide by the University's regulations governing these issues.

Signature ....

Date ....

Consent to share this work

In signing below, I hereby agree to this dissertation being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Signature ....

Date ....

Acknowledgements

I am grateful to...

I'd like to thank...

Abstract

Include an abstract for your project. This should be no more than 300 words.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Background & Objectives

This section should discuss your preparation for the project, including background reading, your analysis of the problem and the process or method you have followed to help structure your work. It is likely that you will reuse part of your outline project specification, but at this point in the project you should have more to talk about.

Note:

- All of the sections and text in this example are for illustration purposes. The main Chapters are a good starting point, but the content and actual sections that you include are likely to be different.

- Look at the document on the Structure of the Final Report for additional guidance.

1.1 Background

1.1.1 Overview

As a student studying in a university away from home, I have often needed to travel back home during the holidays or on weekends. My motivation for this project comes from the requirement that many people have to travel frequently from one place to another.

I have always driven to Cardiff from Aberystwyth and back during the holidays in a car with 4 spare seats. If there was some way I could offer those spare seats to other students or people travelling in the same or similar direction the cost of fuel could be split, meaning a cheaper mode of transport.

1.1.2 Example Use Case

Upon opening the home page, the user will be confronted with the option of either registering, or logging in using existing details. Either action done successfully will log the user in.

Once logged in, the user is taken to their home page, which displays predicted journeys that the site thinks the user might be interested in using their preferences. From this page, they can access all features within the site, which include:

- 'My Activity' - Information about upcoming rides and hitches related to the user. The ability to accept or decline hitch requests for their journeys.

- 'Messages' - Messages to and from other users.

- 'My Profile' - Personal details about the user, available for editing.

- 'Find a Journey' - A search page used to search for journeys from one location to another.

- 'Post Journey' - The page used to post new journeys that the user will be partaking in.

When a user posts a journey from location A to location B using the 'Post Journey' page, it will display a map route which needs to be accepted by the user. Once this is accepted, the journey will be entered into the database, where it will be returned in other users searches if the parameters match the details of the journey.

If another user requests to hitch a ride on the journey, the driver will be prompted in the 'My Activity' page, and will have the option of accepting or declining the hitch request.

The hitch request could be from location A to location B, but it could also be from location A to a new location C. Similarly it could be from location C to location B, or even location C to location D. It is up to the driver to accept or decline the hitch request, depending on how it changes the route that the driver takes.

If the driver accepts this new hitcher, the route will be altered if necessary to include the new pickup / drop off points and saved into the database with one of the spare spaces now filled.

1.1.3 Existing Services

There are existing websites that offer the same sort of carpooling service. The leading services are Carpooling.com [2] and BlaBlaCar [**?**]. These sites provide a platform for people to offer rides to other people needing to travel in similar directions.

1.2 Analysis

Taking into account the problem and what you learned from the background work, what was your analysis of the problem? How did your analysis help to decompose the problem into the main tasks that you would undertake? Were there alternative approaches? Why did you choose one approach compared to the alternatives?

There should be a clear statement of the objectives of the work, which you will evaluate at the end of the work.

In most cases, the agreed objectives or requirements will be the result of a compromise between what would ideally have been produced and what was felt to be possible in the time available. A discussion of the process of arriving at the final list is usually appropriate.

1.2.1 Original Goals

The original goals of the project were to produce an online service that enables people to share journeys with people who are looking to travel to and from similar locations or locations along the route of the driver.

This would enable the users to reduce the cost of the journeys by sharing the cost of fuel.

1.2.2 Requirements

Enable users to:

- FR1 - Register to the site

- FR2 - Log in if an existing member

- FR3 - Post new journeys to the site, offering spare spaces

- FR4 - Search for a journey between two locations

- FR5 - Request to join a journey offered by another user

- FR6 - Accept a hitcher requesting to join a journey

- FR7 - Decline a hitcher requesting to join a journey

- FR8 - Suggest journeys depending on preferences

- FR9 - Cancel a journey

- FR10 - Send messages to other users

1.3 Process

You need to describe briefly the life cycle model or research method that you used. You do not need to write about all of the different process models that you are aware of. Focus on the process model that you have used. It is possible that you needed to adapt an existing process model to suit your project; clearly identify what you used and how you adapted it for your needs.

1.3.1 Overview

1.3.2 Methodology

I have chosen to use the Waterfall Model as the methodology for this project. The reason for this is that this project was of my own design, meaning that the requirements of it were known from the very beginning and were very unlikely to change.

1.3.3 Planning

1.3.4 Research

The vast majority of this service depends on its ability to recognise, locate and route directions from an origin to a destination. The APIs that were looked into were the Google Directions API [4] and the Google Geocoding API.

Chapter 2

Design

You should concentrate on the more important aspects of the design. It is essential that an overview is presented before going into detail. As well as describing the design adopted it must also explain what other designs were considered and why they were rejected.

The design should describe what you expected to do, and might also explain areas that you had to revise after some investigation.

Typically, for an object-oriented design, the discussion will focus on the choice of objects and classes and the allocation of methods to classes. The use made of reusable components should be described and their source referenced. Particularly important decisions concerning data structures usually affect the architecture of a system and so should be described here.

How much material you include on detailed design and implementation will depend very much on the nature of the project. It should not be padded out. Think about the significant aspects of your system. For example, describe the design of the user interface if it is a critical aspect of your system, or provide detail about methods and data structures that are not trivial. Do not spend time on long lists of trivial items and repetitive descriptions. If in doubt about what is appropriate, speak to your supervisor.

You should also identify any support tools that you used. You should discuss your choice of implementation tools - programming language, compilers, database management system, program development environment, etc.

Some example sub-sections may be as follows, but the specific sections are for you to define.

2.1 Overview

2.2 Technologies

2.2.1 PHP

A few languages were considered for this project, these include Ruby, Python and Perl.

Perl was the first to be discarded. This was mainly due to its performance and usability when used in an Object Oriented fashion.

Ruby with Ruby of Rails provides a very stable platform to develop upon, but does not quite have the flexibility that I intended to yield in the development.

The final decision came between Python and PHP. Python is a clean language with very good performance and if easy to use. However, the flexibility that PHP provides when developing web pages and its good Object Oriented capabilities meant that I sided with it even though its performance may not be as good as Python.

2.2.2   JQuery

JQuery is featured on the website as a means of displaying certain features. Graphical maps generated by Google Directions [4] service are retrieved and displayed using JQuery. Some use of it is also used on the messages page of the site.

2.2.3   PSQL Database

2.2.4   Github

Github [?] version control was the control of choice for the project..

2.3   Service Infrastructure

2.3.1   University Hosting

2.3.2   LAMP Server

(PSQL instead of MySQL)

2.4   Service Design

2.4.1   Overview

2.4.2   Method

2.4.3   Structure of Database API

-Controllers -Models

2.5 Website Design

2.5.1 Overview

2.5.2 PHP

2.5.3 Bootstrap

2.5.4 JavaScript and JQuery Library

Google maps embedding

Chapter 3

Implementation

The implementation should look at any issues you encountered as you tried to implement your design. During the work, you might have found that elements of your design were unnecessary or overly complex; perhaps third party libraries were available that simplified some of the functions that you intended to implement. If things were easier in some areas, then how did you adapt your project to take account of your findings?

It is more likely that things were more complex than you first thought. In particular, were there any problems or difficulties that you found during implementation that you had to address? Did such problems simply delay you or were they more significant?

You can conclude this section by reviewing the end of the implementation stage against the planned requirements.

3.1 Overview

3.2 PHP

3.3 Google Directions API

-talk about long motorway journeys and how steps can be very far apart

3.4 Database

3.5 Database API

-procedural v OO

3.6   Website

-communication with API.

Chapter 4

Testing

Detailed descriptions of every test case are definitely not what is required here. What is important is to show that you adopted a sensible strategy that was, in principle, capable of testing the system adequately even if you did not have the time to test the system fully.

Have you tested your system on real users? For example, if your system is supposed to solve a problem for a business, then it would be appropriate to present your approach to involve the users in the testing process and to record the results that you obtained. Depending on the level of detail, it is likely that you would put any detailed results in an appendix.

The following sections indicate some areas you might include. Other sections may be more appropriate to your project.

4.1   Overview

4.2   Database API Testing

4.2.1   Overview?

4.2.2   Unit Tests

4.2.3   Functional Tests

4.3   Website Testing

4.3.1   Overview

4.3.2   Functional Tests

Chapter 5

Evaluation

Examiners expect to find in your dissertation a section addressing such questions as:

- Were the requirements correctly identified?

- Were the design decisions correct?

- Could a more suitable set of tools have been chosen?

- How well did the software meet the needs of those who were expecting to use it?

- How well were any other project aims achieved?

- If you were starting again, what would you do differently?

Such material is regarded as an important part of the dissertation; it should demonstrate that you are capable not only of carrying out a piece of work but also of thinking critically about how you did it and how you might have done it better. This is seen as an important part of an honours degree.

There will be good things and room for improvement with any project. As you write this section, identify and discuss the parts of the work that went well and also consider ways in which the work could be improved.

Review the discussion on the Evaluation section from the lectures. A recording is available on Blackboard.

5.1   Original Goals

5.2   Accomplishments

5.3   Future Improvements

5.4   Future Development

5.5   Design Choices

5.6   Approach

Appendices

Appendix A

Third-Party Code and Libraries

If you have made use of any third party code or software libraries, i.e. any code that you have not designed and written yourself, then you must include this appendix.

As has been said in lectures, it is acceptable and likely that you will make use of third-party code and software libraries. The key requirement is that we understand what is your original work and what work is based on that of other people.

Therefore, you need to clearly state what you have used and where the original material can be found. Also, if you have made any changes to the original versions, you must explain what you have changed.

As an example, you might include a definition such as:

Apache POI library The project has been used to read and write Microsoft Excel files (XLS) as part of the interaction with the clients existing system for processing data. Version 3.10-FINAL was used. The library is open source and it is available from the Apache Software Foundation [1]. The library is released using the Apache License [**?**]. This library was used without modification.

Appendix B

Code samples

2.1   Random Number Generator

The Bayes Durham Shuffle ensures that the psuedo random numbers used in the simulation are further shuffled, ensuring minimal correlation between subsequent random outputs [**?**].

```
#define IM1 2147483563
#define IM2 2147483399
#define AM (1.0/IM1)
#define IMM1 (IM1-1)
#define IA1 40014
#define IA2 40692
#define IQ1 53668
#define IQ2 52774
#define IR1 12211
#define IR2 3791
#define NTAB 32
#define NDIV (1+IMM1/NTAB)
#define EPS 1.2e-7
#define RNMX (1.0 - EPS)

double ran2(long *idum)
{
  /*---------------------------------------------------*/
  /* Minimum Standard Random Number Generator          */
  /* Taken from Numerical recipies in C                */
  /* Based on Park and Miller with Bays Durham Shuffle */
  /* Coupled Schrage methods for extra periodicity     */
  /* Always call with negative number to initialise    */
  /*---------------------------------------------------*/

  int j;
  long k;
  static long idum2=123456789;
```

```
static long iy=0;
static long iv[NTAB];
double temp;

if (*idum <=0)
{
  if (-(*idum) < 1)
  {
    *idum = 1;
  }else
  {
    *idum = -(*idum);
  }
  idum2=(*idum);
  for (j=NTAB+7;j>=0;j--)
  {
    k = (*idum)/IQ1;
    *idum = IA1 *(*idum-k*IQ1) - IR1*k;
    if (*idum < 0)
    {
      *idum += IM1;
    }
    if (j < NTAB)
    {
      iv[j] = *idum;
    }
  }
  iy = iv[0];
}
k = (*idum)/IQ1;
*idum = IA1*(*idum-k*IQ1) - IR1*k;
if (*idum < 0)
{
  *idum += IM1;
}
k = (idum2)/IQ2;
idum2 = IA2*(idum2-k*IQ2) - IR2*k;
if (idum2 < 0)
{
  idum2 += IM2;
}
j = iy/NDIV;
iy=iv[j] - idum2;
iv[j] = *idum;
if (iy < 1)
{
  iy += IMM1;
}
```

```
   if ((temp=AM*iy) > RNMX)
   {
     return RNMX;
   }else
   {
     return temp;
   }
}
```

Annotated Bibliography

[1] Apache Software Foundation, "Apache POI - the Java API for Microsoft Documents," http://poi.apache.org, 2014.

     This is my annotation. I should add in a description here.

[2] Carpooling.com, "Carpooling.com - existing online carpooling service," http://www.carpooling.co.uk/, 2001.

     Existing carpooling website

[3] H. M. Dee and D. C. Hogg, "Navigational strategies in behaviour modelling," *Artificial Intelligence*, vol. 173(2), pp. 329–342, 2009.

     This is my annotation. I should add in a description here.

[4] developers.google.com, "Google directions api," https://developers.google.com/maps/documentation/directions/, 2014.

     API used to retrieve routing data

[5] S. Duckworth, "A picture of a kitten at Hellifield Peel," http://www.geograph.org.uk/photo/640959, 2007, copyright Sylvia Duckworth and licensed for reuse under a Creative Commons Attribution-Share Alike 2.0 Generic Licence. Accessed August 2011.

     This is my annotation. I should add in a description here.

[6] M. Neal, J. Feyereisl, R. Rascunà, and X. Wang, "Don't touch me, I'm fine: Robot autonomy using an artificial innate immune system," in *Proceedings of the 5th International Conference on Artificial Immune Systems*. Springer, 2006, pp. 349–361.

     This paper...

[7] Various, "Fail blog," http://www.failblog.org/, Aug. 2011, accessed August 2011.

     This is my annotation. I should add in a description here.