

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

MENG INDIVIDUAL PROJECT INTERIM REPORT

---

# Active Delay Warning Transport Application

---

*Author:*  
Yangfan ZHANG

*Supervisor:*  
Dr. Peter MCBRIEN

May 28, 2015

Submitted in part fulfillment of the requirements for the degree of Master  
of Engineering in Computing of Imperial College London

## Abstract

abstract

Abstract here. The abstract is a very brief summary of the report's contents. It should be about half a page long. Somebody unfamiliar with your project should have a good idea of what it's about having read the abstract alone and will know whether it will be of interest to them. Note that the abstract is a summary of the entire project including its conclusions. A common mistake is to provide only introductory elements in the abstract without saying what has been achieved.

# Acknowledgments

I would like to thank the following people for their support on this project:

Peter McBrien. This project would not have been possible without his exceptional supervision. I appreciate the time and effort he invested in guiding me during the project days and removed many of my roadblocks.

My personal tutor, Prof. Susan Eisenbach, for her encouragement and care throughout the year.

My housemates, friends, and family for their support through my ups and downs.

Daniel Wong, for his time to review my code and paper.

# Contents

<b>Acknowledgements</b>	<b>1</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Background</b>	<b>3</b>
2.1 London Bus Network . . . . .	3
2.1.1 Bus Network Performance . . . . .	3
2.2 Buses Status Updates . . . . .	4
2.3 Transport for London Open Data . . . . .	5
2.3.1 Live Bus Arrivals API Stream . . . . .	5
2.3.2 Bus Stop Locations and Routes . . . . .	6
2.3.3 Journey Planner Bus Timetables . . . . .	6
2.3.4 Summary on Available Data . . . . .	7
2.4 Current Apps that use TfL Open Data . . . . .	7
2.4.1 TfL Journey Planner . . . . .	7
2.4.2 Google Maps . . . . .	8
2.4.3 Citymapper . . . . .	8
2.4.4 Summary on current Apps . . . . .	8
2.5 Literature review on similar research done in other areas of the world . . . . .	8
2.6 Our Approach / Suggestion . . . . .	8
<b>3 Concept Design</b>	<b>9</b>
3.1 Overview . . . . .	9
3.2 Historical Timetable . . . . .	9
3.3 Current Timetable . . . . .	9
3.4 Reference Timetable . . . . .	9
3.5 Assumption . . . . .	9
<b>4 Data Generation</b>	<b>11</b>
4.1 Overview . . . . .	11

4.2	Development Environment . . . . .	11
4.2.1	Virtual Machine . . . . .	11
4.2.2	MySQL Databases . . . . .	11
4.3	Generating Historical Timetable . . . . .	11
4.3.1	Collecting Bus Arrival Times . . . . .	12
4.3.2	Neighbouring Stops . . . . .	13
4.3.3	Finding the average travel time between neighbouring stops . . . . .	14
4.3.4	Filtering out the negative travel times . . . . .	15
4.4	Generating Reference Timetable . . . . .	16
4.4.1	Journey Planner Bus Timetables . . . . .	16
<b>5</b>	<b>Delay Data Service</b>	<b>18</b>
5.1	Overview . . . . .	18
5.2	Django Framework . . . . .	18
5.3	API Endpoints . . . . .	19
5.3.1	Historical & Current . . . . .	19
5.3.2	Reference . . . . .	19
5.3.3	Arrival . . . . .	19
5.4	Summary of data service . . . . .	19
<b>6</b>	<b>Mobile Application for Active Delay Warning</b>	<b>20</b>
6.1	Motivation . . . . .	20
6.1.1	AngularJS . . . . .	20
6.2	Contribution . . . . .	20
<b>7</b>	<b>Technical Details</b>	<b>21</b>
7.1	Overview . . . . .	21
7.2	Daemen Script Management . . . . .	22
7.3	Backend Deployment . . . . .	22
7.4	Frontend Deployment . . . . .	22
7.5	Software Project Management . . . . .	22
7.5.1	Jenkins . . . . .	22
7.5.2	Trello . . . . .	22
<b>8</b>	<b>Evaluation</b>	<b>23</b>
8.1	Compare predictions to real data . . . . .	23
8.2	User feedback . . . . .	23
8.3	Accuracy . . . . .	23
8.4	Performance . . . . .	23
8.5	User feedback on UI . . . . .	23

<b>9 Future Work</b>	<b>24</b>
<b>10 Conclusion</b>	<b>25</b>
<b>Acronyms</b>	<b>26</b>
<b>Glossary</b>	<b>27</b>
<b>References</b>	<b>28</b>

# Chapter 1

## Introduction

The London bus network carries 2.4 billion passengers a year [1]. On average, the buses come about 1 minute later than scheduled [6].

Transport for London (TfL) publishes the live expected bus arrival times. It is currently widely available on digital live bus arrivals signs at more than 2,500 bus stops [2]. Passengers can also access this information via SMS, the web, or mobile applications.

Passengers chose travel mode based on travel time and convenience.  
Find evidence for this assumption. How do people choose travel mode  
/ whether to take bus?

Passengers use the bus arrival times to plan their journey, by factoring in the waiting time when choosing the buses to take. Current London journey planning software takes the journey start time, start location, and destination as input, and recommends routes consisting of a variety of travel mode, with an estimate travel time for each suggested journey. Such popular planners include Google maps [3], Citymapper [4] and TfL Journey Planner [5].

However, the accuracy of the bus arrival times published is affected by many external factors. For example, when there is heavy traffic, the buses are likely to be delayed by a difference significant enough for a change in passengers' route picking. Yet, this delay information is not reflected in the arrival time data or the estimated journey time early enough for the passengers to make a decision to choose an alternate route. As a result, passengers waste time waiting for buses that come much later than expected, or choose to board a bus that will take much longer than the estimated journey time to reach the destination. Although the average bus delay is 1 minute, there was a 16.6 chance of waiting for more than 10 minutes [15].

Scenario, a passenger chose to take a bus, then only realises delay at the end of the journey. Would have chosen another mode of travel should he know of the delay earlier.

What is the problem? what are the objectives? what are the data?  
What is my approach? What are my contributions and results?

This can be avoided if passengers are informed of the delays in bus arrival times and estimated travel time in advance. Currently, there are no available data services or applications that generate predictions potential delays.

For exemple, a passenger chose to take a bus from point A to point b Such delays can predicted by analysing the historical delays. This is achieved by collecting data from the TfL live bus arrivals Application Program Interface (API) stream feed[2], and estimate the average journey time required between two locations on a given time of the day. Next, a bus arrival time table with delays during various time windows over a week can be crafted and fine tuned incrementally. We compared the travel times in this timetable with the official travel times published by the TfL Open Data to find out the delays in bus arrival times.



# Chapter 2

## Background

### 2.1 London Bus Network

The bus network in London is one of the largest and most accessible in the world. It is carrying a staggering number of passengers, with more than 2.4 billion journeys in 2013/14, which was more than any year since 1959 [1].

On an average day between 2005 and 2010, about 14% of the trips made by London residents were by bus [9]. They spent on average 14 minutes per day on these bus trips.

There are currently 19,345 bus stops, and 680 routes served by 8,765 buses daily in London[10].

produce graph for this, number of stops, routes and buses

#### 2.1.1 Bus Network Performance

TfL published the following figures in the second quarter 2014/2015 buses performance data [6].

For the high frequency services, the average scheduled wait was 4.86 minutes, the average excess wait was 0.94 minutes, and the average actual wait was 5.80 minutes. While passengers could expect the buses to come within 10 minutes 83.4% of the time, there was 15.1% chance of waiting for 10-20 minutes, 1.3% chance of waiting for 20-30 minutes, and 0.2% chance of waiting for more than 30 minutes.

For the low frequency services, 87% of the buses services were on time, and 11.4% were 5-15 minutes late.

For the night buses, 84.5% of the services were on time. The average excess wait was 0.68 minutes.

produce graph here

# STATUS UPDATES

The screenshot shows the TfL Buses Status Update Service interface. At the top, there is a search bar with the placeholder text "Find a bus stop or route". The search bar contains the number "14" and a blue "Go" button. Below the search bar, the text "Showing disruptions on route 14 towards Putney Heath / Green Man" is displayed, along with a link "Clear route". Below this, there are two yellow boxes with a warning icon and a plus sign. The first box contains the text "Status alert for route 14" and the second box contains the text "Additional information".

Figure 2.1: TfL Buses Status Update Service

The bus arrivals might be affected by traffic congestion, staff availability, engineering problems, or mechanical breakdown [15].

## 2.2 Buses Status Updates

To inform passengers of the bus service disruptions or diversions, TfL provides a bus status updates service online [7]. Passengers can retrieve relevant bus service status for a given bus stop or route (Figure 2.1).

The textual information include service disruptions, diversion, suspensions, and delays due to heavy traffic.

While this service informs passengers of current abnormal bus schedules, it requires passengers to visit the site to check for specific information, and does not provide an estimation on the travel time under the special conditions.

TfL also publishes live status news and updates on Twitter[8], but it is not convenient for passengers to search specific information relevant to their journeys.

Many current popular journey planners such as Google Maps[3], Citymapper London[4], and TfL Journey Planner incorporate the bus delay information in the suggested journeys as a textual alert. However, these apps do not recompute the estimated journey time for passengers to make an informed

decision.

As a result, the lack of data service on bus delay predictions and travel times estimations presents a significant barrier to bus passengers who wish to plan their bus journeys for specific appointments.

## 2.3 Transport for London Open Data

Open data is defined as data which can be used, re-used and re-distributed freely by anyone - subject only at most to the requirement to attribute and share-alike. There may be some charge, usually no more than the cost of reproduction [11].

It is TfL's strategy to provide free and open data. It began in 2007 using embedded widgets[12]. In 2015, there are around 30 feeds and APIs available [13].

There are 3 forms of data:

- static data files which rarely change,
- feeds that refreshed at regular intervals,
- and APIs that enable a query to receive a bespoke response, depending on the parameters supplied.

Over 5,000 developers have registered for the open data[13], and around 200 travel apps are powered by it [1].

### 2.3.1 Live Bus Arrivals API Stream

The Live Bus Arrivals API Stream provides the predicted time until a bus is expected to arrive at a stop. This API is designed to enable application developers to subscribe to live bus information and use this data to develop innovative services[14].

These arrival predictions are available for the next 30 minutes at any point in time. For example, at 9am, the stream will provide predicted bus arrivals up to 9.30am on the same day. This data is refreshed every 30 seconds [14].

This API is controlled via a number of different HTTP requests and parameters. A request is structured as follows:

`http://server/virtualDirectory/type/version?HTTPparameters`

## Data Types

This API service provides two types of request to users:

- **instant** The instant requests are made by the client and the server will respond with a single message.
- **stream** The streaming requests are made by the client, in response the server will continually serve data to satisfy the request until the connection is terminated.

The data source for both instant and streaming requests is consistent, ensuring that the data provided to the public remains the same.

We used this API as a source for bus arrival times. See details in Section 4.3.1.

### 2.3.2 Bus Stop Locations and Routes

The TfL Open Data provides network information on the location of all bus stops in London, and the sequence of bus stops in every bus route.

This data is in the Comma Separated Values (CSV) format. Every entry in the bus sequence file consists of information on the route number, the route direction, the sequence at which the given bus stop is positioned in the route. There is also information on the bus stop name, and bus stop codes for identification purposes.

We used the bus sequences as a reference when calculating the bus journey time. See Section 4.3.2 for details.

### 2.3.3 Journey Planner Bus Timetables

The Journey Planner Bus Timetables[21] contains information on official bus schedules including stops, routes, departure times, departure frequencies, operational notes, as well as the days on which the services run.

The timetables uses the Extensible Markup Language (XML) [17] format, with the schema defined in TransXChange [16], the UK nationwide standard for exchanging bus schedules and related data. For this project, we used the General schema version 2.1[18][19], the latest available version for download. Each XML file contains the bus timetables for one route.

## Data Structure

The TransXChange timetable model has the following seven basic concepts[20]:

1. **Service** contains one or more **JourneyPattern** elements and one or more **VehicleJourney** elements. This is the basic concept that brings together the information about a registered bus service.
2. **Registration** specifies the registration details for a service.
3. **Operator** indicates the entity who runs the service.
4. **Route** describes the physical path taken by buses on the service as an ordered list of **StopPoints**.
5. **StopPoint** contains reusable declarations of the stops used by the routes and journey patterns of the schedule. All **StopPointRef** instances elsewhere in a document are resolved against the contents of the **StopPoints** element. All stops are defined as being NaPTAN points.
6. **JourneyPattern** specifies an ordered list of links between the **StopPoints**, giving the *relative travel times* between each pair of neighbouring stops.
7. **VehicleJourney** specifies the individual scheduled journey *at a specific absolute time*.

These elements give a complete official bus schedule for each route with the departure time for each bus journey, the relative bus travel time between stops, and the days on which the service operates on. We extracted the official bus timetables from these XML files, as discussed in Section 4.4.1.

### 2.3.4 Summary on Available Data

While there is sufficient data on London bus network, official bus timetables, and live arrival times, there is no data service offering real-time predictions on bus delays. We used these available data to create predictions for bus travel timetables, as discussed in Chapter 4.

## 2.4 Current Apps that use TfL Open Data

### 2.4.1 TfL Journey Planner

- name of app
- company/app logo

- year released
- version
- platform
- main features

#### **2.4.2 Google Maps**

#### **2.4.3 Citymapper**

#### **2.4.4 Summary on current Apps**

No app that gives predictions and delay warnings on travel time

### **2.5 Literature review on similar research done in other areas of the world**

### **2.6 Our Approach / Suggestion**

We try to find a solution for these two problems by providing a service of bus travel time predictions, and a demo application to show case the use of such a data service.

# Chapter 3

## Concept Design

### 3.1 Overview

TfL releases average travel time between stops. However, this data is too general and sometimes not accurate.

The general idea to predict delays is that the average travel time between 2 stops depends mainly on the day of the week and the hour.

### 3.2 Historical Timetable

We collect historical bus arrival times, and calculate the average travel time between 2 neighbouring bus stops by all routes, on given day and hour

### 3.3 Current Timetable

We collect the past 1 hour of bus arrival times, and calculate the average travel time between neighbouring bus stops.

### 3.4 Reference Timetable

Get the official average travel time from journey planner timetable

### 3.5 Assumption

The historical timetable should converge to reference timetable.

When the current timetable is very different from reference timetable, there's delay.



# Chapter 4

## Data Generation

### 4.1 Overview

UML diagrams

### 4.2 Development Environment

#### 4.2.1 Virtual Machine

#### 4.2.2 MySQL Databases

We chose to use MySQL to store the data for the following benefits it offers:

- **User Interface** MySQL has convenient database management tools such as phpMyAdmin[29] and Sequel Pro[30] for easy data browsing.
- **Scalability** MySQL can handle memory heavy computation efficiently, given the correct configuration.

### 4.3 Generating Historical Timetable

- data collection
- data storing
- data processing
- DB optimisation

- python scripts and SQLs
- resulting table name and schema

### 4.3.1 Collecting Bus Arrival Times

We collect bus arrival data for analysis from the live bus arrivals API. The base URL used in this project was [http://countdown.api.tfl.gov.uk/interfaces/ura/stream\\_V1](http://countdown.api.tfl.gov.uk/interfaces/ura/stream_V1).

We supplied the following parameters which specify the fields returned by the API.

- *StopID* This is the alphanumeric identifier of a bus stop. It is also known as `stop_code_lbsl`.
- *LineName* This is the route number that is displayed on the front of the bus on any publicity advertising the route.
- *VehicleID* The unique identifier of the vehicle.
- *TripID* The identifier of the specific trip that the prediction is for.
- *EstimatedTime* This is the predicted time of arrival for the vehicle at a specific stop.
- *ExpireTime* This is the time at which the corresponding prediction is no longer valid and should stop being displayed.

The resulting query URL is [http://countdown.api.tfl.gov.uk/interfaces/ura/stream\\_V1?ReturnList=StopID,LineName,VehicleID,TripID,EstimatedTime,ExpireTime](http://countdown.api.tfl.gov.uk/interfaces/ura/stream_V1?ReturnList=StopID,LineName,VehicleID,TripID,EstimatedTime,ExpireTime).

Each data entry contains an estimated arrival time for each bus journey at a given bus stop. This estimated arrival time is stored in the database via an UPDATE statement, which ensures that only the latest estimated arrival times per journey per bus stop are stored. To avoid having data being overwritten, only the entries that have been changed in the recent 10 minutes can be updated. This has been achieved through running a daemon process written in python on a virtual host. Table 4.1 shows the schema for arrivals table.

### Assumption

We assume that the actual bus arrival time is the midpoint between the last estimated arrival time, and the system time when the clear signal (*ExpireTime* = 0) is received.

Column Name	Type	Default
id(Primary)	int(11)	Auto Increment
stop_code_lbsl	varchar(64)	
route	varchar(64)	
vehicle_id	varchar(64)	
trip_id	varchar(64)	
arrival_date	date	
arrival_time	timestamp	NULL
expire_time	timestamp	NULL
recorded_time	timestamp	Current Timestamp

Table 4.1: delay\_arrivals Table Schema

Column Name	Type	Comments
id(Primary Key)	int(11)	Auto Increment
route	varchar(64)	The route name
run	int(11)	The route direction
sequence	int(11)	The sequence of the bus stop in the route
stop_code_lbsl	varchar(64)	The internal bus stop identifier
bus_stop_code	varchar(64)	The public code for the bus stop
naptan_atco	varchar(64)	The national identifier of the bus stop
stop_name	varchar(64)	The name of the bus stop

Table 4.2: delay\_bus\_sequences Table Schema

### 4.3.2 Neighbouring Stops

We imported the bus sequences into the delay\_bus\_sequences table (Table 4.2)

**Question:** Can I skip some columns of the table schema, as those columns have not been used in the project, but just storing as reference for now?

Additionally, we extracted information on all pairs of neighbouring bus stops and the routes that serve between them. We save this information in the delay\_neighbours table (Table 4.3). See sample data in Table 4.4.

Column Name	Type	Comments
id(Primary Key)	int(11)	Auto Increment
route	varchar(64)	The bus route
start_stop	varchar(64)	The stop_code_lbsl for the start stop
end_stop	varchar(64)	The stop_code_lbsl for the end stop

Table 4.3: delay\_neighbours Table Schema

id	route	start_stop	end_stop
18433	30	10002	11469
44878	N19	10002	11469
47128	N41	10002	29772
8653	19	10002	11469

Table 4.4: Sample data in delay\_neighbours Table

### 4.3.3 Finding the average travel time between neighbouring stops

update this part

To experiment with the queries, we selected one pair of the neighbouring stops (10002, 11469), and listed the time required to travel from stop 10002 to stop 11469 by finding the difference in arrival times for each journey. Sample entries of this list is shown in Figure 4.1.

We then calculated the average journey time required to travel from 10002 to 11469 for each hour in each week of the day. This information is stored as a timetable, which would be used for further analysis.

Figure4.2 shows the timetable generated. Each cell indicates the average journey time required to travel from stop 10002 to stop 11469 at a give hour of a give week of day. The **NULL** values are due to a current databases performance issue. This will be resolved later.

start_stop	end_stop	route	trip_id	day	start_time	end_time	length_in_secs
10002	11469	19	346930	Monday	1/12/2015 12:34:16	1/12/2015 12:35:28	72
10002	11469	4	547334	Monday	1/12/2015 12:39:55	1/12/2015 12:41:14	79
10002	11469	19	346961	Monday	1/12/2015 12:40:04	1/12/2015 12:41:53	109
10002	11469	30	226765	Monday	1/12/2015 12:45:07	1/12/2015 12:47:11	124
10002	11469	4	547676	Monday	1/12/2015 12:47:17	1/12/2015 12:49:12	115
10002	11469	19	346976	Monday	1/12/2015 12:48:21	1/12/2015 12:50:09	108
10002	11469	30	276872	Monday	1/12/2015 12:54:05	1/12/2015 12:55:08	63

Figure 4.1: List of journey time from stop 10002 to stop 11469

hour	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
0	NULL	NULL	75.1	56	63.5	62.5	NULL
1	NULL	NULL	68.4	60.6	69.6	54	NULL
2	NULL	NULL	50.5	32.5	58.5	86.5	NULL
3	NULL	NULL	NULL	64	69	51.5	NULL
4	NULL	NULL	NULL	88.5	62	46	NULL
5	NULL	NULL	NULL	61	73	59.3	NULL
6	NULL	NULL	NULL	69	79.4	75.1	NULL
7	NULL	NULL	NULL	88.1	81.7	84.3	NULL
8	NULL	NULL	NULL	90.2	118.8	116.7	NULL
9	NULL	NULL	NULL	82	90.9	101.7	NULL
10	NULL	NULL	NULL	74.4	90.8	145.6	NULL
11	NULL	NULL	NULL	111.2	97.9	124.2	NULL
12	NULL	95.5	267.3	169.9	146.5	114.7	NULL
13	NULL	125.4	245.3	168.2	104.7	NULL	NULL
14	NULL	90.4	NULL	123.8	85.7	NULL	NULL
15	NULL	120.7	113	153.1	87.2	NULL	NULL
16	NULL	168.1	125.1	134.1	100.5	NULL	NULL
17	NULL	113	110.4	124.6	126.5	NULL	NULL
18	NULL	127.1	110.9	179.8	148.6	NULL	NULL
19	NULL	115.5	149.3	243.7	160	NULL	NULL
20	NULL	87.7	96	140.5	85.2	NULL	NULL
21	NULL	80.1	94	94.5	85	NULL	NULL
22	NULL	75	93.3	80.4	100.7	NULL	NULL
23	NULL	54.6	75.6	78	73.7	NULL	NULL

Figure 4.2: Average journey time in seconds from stop 10002 to stop 11469 for each hour of each day of week

Start Stop	End Stop	Route	Start Time	End Time	Travel Time(sec)
9326	15552	W13	14:53:18	14:52:14	-64

Table 4.5: Travel Time Log Entry with Negative Travel Time

We plan to construct a timetable this way for each pair of the neighbouring bus stop.

#### 4.3.4 Filtering out the negative travel times

In the travel\_time\_log generated, there were trips between two neighbouring stops with negative travel times, such as the entry shown in Table 4.5.

This was because when we performed the join of the arrivals table, there was a more recent update on the arrival times for the start stop, whereas the arrival times of end stop had not been updated. In Table 4.6, we observed that the Recorded Time for the end stop 9326 was more recent than that of the end stop. As a result, the arrival time for the end stop was earlier than the start stop, causing the travel time to be negative.

We filtered out these negative values before calculating the average travel

Stop Code	Route	Vehicle ID	Trip ID	Arrival Time	Recorded Time
9326	W13	18685	135229	14:53:18	14:48:25
15552	W13	18685	135229	14:52:14	14:44:02

Table 4.6: Arrivals Entries to Explain Negative Travel Time

time between neighbouring stops.

## 4.4 Generating Reference Timetable

### 4.4.1 Journey Planner Bus Timetables

Need more detailed explanation. Walk through an example with xml code

In order to calculate the delays in bus arrival times, we need the official bus travel times between stops for reference. This data was extracted from the Journey Planner Bus Timetables[21] as part of the TfL Open Data.

Each xml file contains bus schedule information for a route. For each day of the week, there are predefined number of VehicleJourneys running the give route throughout the day. We used the VehicleJourneys as a starting point to retrieve the departure time of the actual vehicle from the terminal. We then retrieve the corresponding JourneyPattern, and obtained the travel time between each neighbouring stops on the route for the given vehicle journey, to compute the cumulative travel times throughout the route.

The above computation was performed on each xml file to generate the actual arrival time and travel time for each vehicle trip at each stop in the route throughout the day. The results of this computation was stored in the `delay_tfl_timetable` table(Table 4.7).

Column Name	Type	Comments
id(Primary Key)	int(11)	Auto Increment
route	varchar(64)	The bus route
day	varchar(32)	The day of week for the vehicle journey
run	int(11)	The route direction
sequence	int (11)	The sequence of the bus stop in the route
stop_name	varchar(64)	The name of the bus stop
naptan_atco	varchar(64)	The national identifier of the bus stop
arrival_time	datetime(6)	The expected arrival time for the given vehicle at the current bus stop
travel_time	int(11)	The travel time in seconds from the previous stop in the route to the current stop
cumulative_travel_time	int(11)	The travel time in seconds from the terminal to the current stop
departure_time_from_origin	datetime(6)	The departure time of the given vehicle from the terminal

Table 4.7: delay\_tfl\_timetable Table Schema

# Chapter 5

## Delay Data Service

### 5.1 Overview

To enable structured query of the predictions generated, we created a REST API service for the prediction data. We hope to make these endpoints freely available for other developers to use.

### 5.2 Django Framework

We used Django Framework[22] for the backend of this project. It is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. We chose it for the following advantages:

- It stores databases schema as data models[23], and allows for quick database migration. This makes deployment to the production server easy by using the given migration command [24].
- Django handles user authentication securely [25], which saves work for development on this part.
- It has a nice compatible REST framework[26], which supports REST API routing[27] and query parameter parsing. Additionally, it offers a built-in user interface that displays the query result with pagination[28].
- Django has a Debug mode that display constructive error messages, make the development much easier.
- It is written in Python, which is a preferred language for its simple and short syntax.



## **5.3 API Endpoints**

- address of the endpoint
- parameters supplied
- return result format

### **5.3.1 Historical & Current**

### **5.3.2 Reference**

### **5.3.3 Arrival**

## **5.4 Summary of data service**

Met the first objective by supplying data on bus travel time predictions

# Chapter 6

## Mobile Application for Active Delay Warning

### 6.1 Motivation

Demo App to use the data for effective delay warning

- Write from the user point of view

- to demonstrate what the predictions can be used for non-technical users

- easy to use interface

- easy to understand data presentation

- personalised warning feature

#### 6.1.1 AngularJS

### 6.2 Contribution

save users time make informed decisions when choosing travel mode

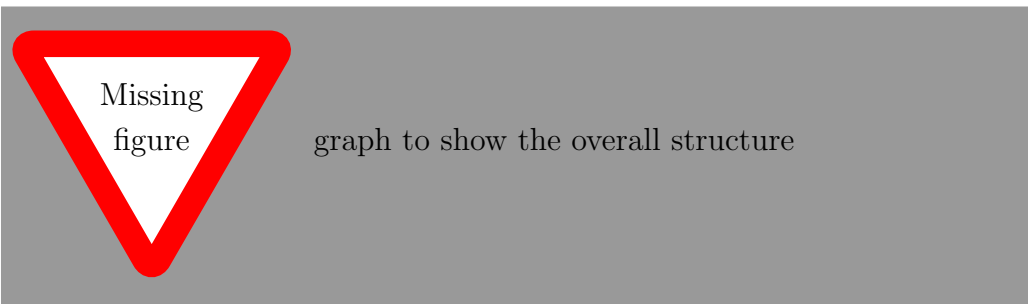
# Chapter 7

## Technical Details

Missing Testing section

### 7.1 Overview

- Used Django Framework to build the backend that retrieves data from a MySQL database.
- Used AngularJS with Twitter Bootstrap for frontend development



## **7.2 Daemen Script Management**

## **7.3 Backend Deployment**

## **7.4 Frontend Deployment**

## **7.5 Software Project Management**

### **7.5.1 Jenkins**

### **7.5.2 Trello**

# Chapter 8

## Evaluation

### 8.1 Compare predictions to real data

We can compare the predictions generated to the live bus arrivals data stored in the arrivals table. We can calculate the standard deviation of the difference in the predicted time and actual arrival time. This value will be the direct indicator of the accuracy of the predictions.

### 8.2 User feedback

We can get students that take buses regularly to use the mobile application to test the accuracy of the delay predictions. The user feedback gathered will be the second input for evaluation.

### 8.3 Accuracy

How accurate are the predictions?

### 8.4 Performance

How reliable is the service?

### 8.5 User feedback on UI

How effective is it at warning delay? ~~~~~ Stashed changes

# Chapter 9

## Future Work

¡Future work here!

# Chapter 10

## Conclusion

¡Conclusion here!

# Acronyms

**API** Application Program Interface. 2, 4

**CSV** Comma Separated Values. 4

**TfL** Transport for London. 1, 2, 4

**XML** Extensible Markup Language. 4



# Glossary

**TransXChange** the UK nationwide standard for exchanging bus schedules and related data. 4, 5

# References

- [1] Transport for London Annual Report and Statement of Accounts 2013/14, <http://tfl.gov.uk/cdn/static/cms/documents/annual-report-2013-14.pdf> [visited on 27/01/2015]
- [2] Transport for London Live Bus Arrivals, <http://www.tfl.gov.uk/modes/buses/live-bus-arrivals> [visited on 27/01/2015]
- [3] Google Maps, <https://www.google.co.uk/maps> [visited on 30/01/2015]
- [4] CityMapper London, <https://citymapper.com/london> [visited on 30/01/2015]
- [5] TfL Plan A Journey, <http://tfl.gov.uk/plan-a-journey/> [visited on 30/01/2015]
- [6] Transport for London Buses Network Performance Second Quarter 2014/15, <http://tfl.gov.uk/cdn/static/cms/documents/network-performance-latest-quarter.pdf> [visited on 27/01/2015]
- [7] Transport for London Buses Status Updates, <http://www.tfl.gov.uk/bus/status/> [visited on 22/05/2015]
- [8] TfL Bus Alerts Twitter, <https://twitter.com/TfLBusAlerts/status/601795342049398784> [visited on 22/05/2015]
- [9] Travel in London, Supplementary Report: London Travel Demand Survey (LTDS), <http://www.tfl.gov.uk/cdn/static/cms/documents/london-travel-demand-survey.pdf> [visited on 28/01/2015]
- [10] Transport for London Bus Stops Locations and Routes Open Data, <https://www.tfl.gov.uk/info-for/open-data-users/our-feeds> [visited on 28/01/2015]

- [11] APPSI Glossary, <http://www.nationalarchives.gov.uk/appsi/appsi-glossary-a-z.htm>
- [12] p26. What is the Value of Open Data?,  
url<https://www.nationalarchives.gov.uk/documents/meetings/20140128-appsi-what-is-the-value-of-open-data.pdf>
- [13] Transport for London Open Data, <https://www.tfl.gov.uk/info-for/open-data-users/our-open-data> [visited on 28/01/2015]
- [14] Transport for London Live Bus & River Bus Arrivals API Interface Documentation, <https://www.tfl.gov.uk/cdn/static/cms/documents/tfl-live-bus-river-bus-arrivals-api-documentation-v16.pdf> [visited on 28/01/2015]
- [15] Transport for London Buses Performance Data, <https://www.tfl.gov.uk/corporate/publications-and-reports/buses-performance-data#on-this-page-5> [visited on 28/01/2015]
- [16] TransXChange, <https://www.gov.uk/government/collections/transxchange> [visited on 13/05/2015]
- [17] XML Schema Language, <http://www.w3.org/TR/xmlschema-2/> [visited on 13/05/2015]
- [18] TransXChange Downloads & Schema, [https://www.gov.uk/government/uploads/system/uploads/attachment\\_data/file/391783/Downloads\\_Schema-2015-01-05\\_-\\_Updated.pdf](https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/391783/Downloads_Schema-2015-01-05_-_Updated.pdf) [visited on 13/05/2015]
- [19] TransXChange Schema 2.1 xsd, [http://www.transxchange.org.uk/schema/2.1/TransXChange\\_general.xsd](http://www.transxchange.org.uk/schema/2.1/TransXChange_general.xsd) [visited on 13/05/2015]
- [20] TransXChange Schema Guide 2.1 & 2.2a, <http://81.17.70.199/transxchange/schema/2.1/guide/TransXChangeSchemaGuide-2.1-v-45.pdf> [visited on 13/05/2015]
- [21] Transport for London Open Data Feeds, <https://www.tfl.gov.uk/info-for/open-data-users/our-open-data> [visited on 13/05/2015]
- [22] Django Framework, <https://www.djangoproject.com/> [visited on 13/05/2015]

- [23] Django Framework Documentation - Models, <https://docs.djangoproject.com/en/1.8/topics/db/models/> [visited on 28/05/2015]
- [24] Django Framework Documentation - Migrations, <https://docs.djangoproject.com/en/1.8/topics/migrations/> [visited on 28/05/2015]
- [25] Django Framework Documentation - User Authentication, <https://docs.djangoproject.com/en/1.8/topics/auth/> [visited on 28/05/2015]
- [26] Django REST Framework, <http://www.django-rest-framework.org/> [visited on 28/05/2015]
- [27] Django REST Framework API Guide - Routers, <http://www.django-rest-framework.org/api-guide/routers/> [visited on 28/05/2015]
- [28] Django REST Framework API Guide - Pagination, <http://www.django-rest-framework.org/api-guide/pagination/> [visited on 28/05/2015]
- [29] phpMyAdmin, [http://www.phpmyadmin.net/home\\_page/index.php](http://www.phpmyadmin.net/home_page/index.php) [visited on 28/05/2015]
- [30] Sequel Pro, <http://www.sequelpro.com/> [visited on 28/05/2015]