**An investigation into the graphical representation of smart cities data**

**Literature Review CPU6001**

**Contents**

[1. Introduction 2](#_Toc471218826)

[2. Approach and Methods 4](#_Toc471218827)

[3.0 Findings 5](#_Toc471218828)

[3.1 MongoDB Database 5](#_Toc471218829)

[3.2 Data Visualisation 6](#_Toc471218830)

[3.3 Data-Driven Documents (D3.JS) 9](#_Toc471218831)

[3.4 Dygraphs 11](#_Toc471218832)

[3.5 Chart.js 12](#_Toc471218833)

[4.0 Evaluation and Reflection 14](#_Toc471218834)

[4.1 Database 14](#_Toc471218835)

[4.2 D3.js 14](#_Toc471218836)

[4.3 Chart.js 15](#_Toc471218837)

[4.4 Dygraphs 16](#_Toc471218838)

[5.0 Bibliography 16](#_Toc471218839)

[6.0 Appendices 20](#_Toc471218840)

[Appendix 1. Tabulated Data (Lambert,2015) 20](#_Toc471218841)

[Appendix 2. Simple D3 graph (Lambert, 2016) 21](#_Toc471218842)

[Appendix 3. Concentric circles emanating from glowing red dot (Bostock, 2016) 21](#_Toc471218843)

[Appendix 4. Fig4. Dygraphs Line Graph Example. (Dygraphs.com, 2016) 22](#_Toc471218844)

[Appendix 5. Chart.js Bar Chart. (Chartjs.org, 2016) 22](#_Toc471218845)

# 1. Introduction

The introduction and progressive implementation of Internet Protocol version 6 (IPv6), the replacement of Internet Protocol version 4 (IPv4) is transforming the available number of Internet Protocol (IP) addresses. By moving from a 32bit 128bit binary numbering system.

The world was quickly using up the 4,290,000,000 (4.29 billion) IPv4 addresses that were available, limiting the number of devices able to have internet connectivity.

IPv6 supports up to 2^128 IP addresses, 1 address for each device connected to the internet and over 340 trillion, trillion, trillion individual IP addresses, which is enough for every person on earth to have many trillions of IP addresses. This change means that there are now sufficient individual internet addresses for a multitude of applications, from domestic Refrigerators, which are starting to emerge with internet connected cameras, voice-activated home control systems such as the Amazon echo or Google Home devices, smart sensors in car park spaces. The list of possible applications is only limited by the imagination of the device builder.

This technological development has facilitated what has become known as the Internet of Things (IoT). (Parr, 2016)

The Manchester-based technology company Biodata Ltd operate in the Smart Cities sector, a sector that employs smart solutions to manage a city’s assets. In a smart city, everything from traffic flow to water supply can be monitored and managed with smart, technological and internet connected devices.

Biodata specialise in intelligent people counting systems and have developed a software package called Video Turnstile. This software utilises a Closed-Circuit Television Camera (CCTV) to automatically detect and counts people moving through controlled points. (Retail Sensing, 2016).

Biodata have installed data logger devices using the Raspberry Pi single board computer, a CCTV camera and their Turnstile people counting software package on public buses in several places, including San Francisco, USA and Jaipur, Northern India.

This data logger passes information captured at one-minute intervals – passenger number movements, geographical coordinates using the Message Queuing Telemetry Transport (MQTT) protocol back to a company-based MQTT broker that has been designed for unreliable and delay prone networks and is particularly suitable for IoT applications. (mqtt.org, 2016) This data is then stored as individual documents in a MongoDB database.

Discussions were held with Dr Collins, the Managing Director of Biodata in October and December 2016 to discover the scope of the project, which produced a valuable insight to the problems being addressed and the reasoning behind them.

According to Dr Collins, the bus companies in certain underdeveloped countries operate a fixed fee system where bus journeys are charged at a flat rate and paid for using hard cash. Corruption can in some cultures be widespread, and problems occur where the full takings from buses are not passed on to company officials. In 2014, IOL, the independent South African media organisation reported that 52 bus drivers based in Durban, South Africa, were dismissed for corruption in one day alone (Zulu, 2016).

Dr Collins reports that the problem in Brazil has become so widespread that after a move to using ticket machines to record passenger numbers, bus drivers had been caught with their own personal machines, which are apparently used to fool bus inspectors during random ticket inspections, while providing employers with an automated count of passengers, albeit a fraudulent number.

The main purpose of the Biodata equipment in these circumstances will be simply to protect the revenue of the bus company by counting passenger movements onto the bus, allowing officials to have an accurate value for the fares collected by their employees.

The decision by Biodata to use the non-relational MongoDB database rather than the more traditional, simpler to construct and query SQL database system was explained by way of two reasons: The reported speed and scalability advantage of the MongoDB database was required as a large fleet could easily make 300,000 record entries (or documents) each day, which is a sizeable amount of data, and that MongoDB was becoming mainstream in the Smart Cities sector. It has proved difficult to definitively substantiate Dr Collins’ claims to either the speed or the utilisation within the smart cities sector, as there are many opposing views and academic comparison is difficult to find, but as this is the technology Dr Collins has chosen, the software will be constructed with this constraint in mind.

Biodata work predominantly with JavaScript and have specified that the JavaScript library D3.js (Data-Driven Documents) be used to read data from the MongoDB database and display it graphically in a web page. With these stipulations in mind, the suitability of these technologies will be considered and alternative methods investigated.

# 2. Approach and Methods

Ordinarily, the initial phases of an assignment of this nature would be to formulate a method of selecting the most appropriate tools to complete the job. In this assignment, that would entail a detailed examination of appropriate database technologies by examination of academic documents such as journals, case studies and reports. Expert opinion would be evaluated and eventually, an appropriate technology would be identified.

This process would also be followed in order to discover how data is visualised in a web page and then the various visualisation methods would be investigated from an academic standpoint. This process would hopefully result in the emergence of the most suitable technology.

As discussed in the introduction, the tools for this project have already been selected by the client and are most unlikely to change.

The non-relation MongoDB database and the D3.js JavaScript library are clear constraints. Alternatives will be examined, but it is very unlikely that they will be used in the development of the final artefact.

# 3.0 Findings

## 3.1 MongoDB Database

The data that will ultimately be visualised is used to populate documents in a MongoDB non-relational database.

Initially developed in 2007 by the software company 10gen, MongoDB is an open source database designed to be compatible across multiple platforms and is a ‘document-oriented database system’ (Mehrabani, 2014)

MongoDB employs the use of ‘Documents’ in the database rather than the relational ‘Record’ used in a conventional SQL database. Documents are stored in ‘collections’ rather than tables and have no need for the rigid scheme required in a relational database.

The lack of this ‘relational’ requirement allows the data model to be updated without the need to update existing datasets.

In a development environment, such as this, the ability to easily alter datasets without re-building a database is particularly useful as changes will inevitably be required at some time during the development lifecycle.

The database is designed with the ability to be highly scalable by increasing the server capacity with increased processing power, Hard disk and memory upgrades. MongoDB supports horizontal scaling and as such can be deployed across multiple machines using a process called ‘sharding’. The sharding method issues data across several servers and allows large amounts of data to be utilised efficiently, as multiple CPU’s and large amounts of RAM are available as well as the advantage that many hard drives reduces latency during input and output operations.

Sharded data clusters also increase availability as at times of high usage, partial read/write operations can be performed whilst some clusters are unavailable as they are simply performed on an alternative cluster (MongoDB Inc, 2016)

When consideration is given to the millions of data documents the Biodata data logger will generate, the scalability of the MongoDB database confirm that the technology is extremely suitable for this application.

## 3.2 Data Visualisation

Information captured by the data logger is stored in the database and then needs to be presented to the bus company for analysis. There are various ways of presenting this data, some are simple and others extremely complex.

Using a simple Structured Query Language (SQL) query and a little Hyper Text Markup Language (HTML5), a tabulated table can be produced and all selected records can be viewed on a web page, as shown in Fig1.

This method is simple and quick to produce, but the result is dull and will not maintain user attention for very long. Scientific studies have proved that well-visualised data is better understood and retained longer than data presented in a textual format.

The Interaction Design Foundation point out that tabulated data is extremely useful to present and absorb exact figures, but can often fail to identify trends and patterns in datasets. (Few, 2016)

Visualisations increase the accuracy of understanding of data when compared to the presentation of data in a numerical format. Medical test results presented to a group of doctors and patients show large differences of understanding and recollection depending upon methods used. Groups that were shown visualised representations and numeric data gained a 20% more accurate understanding of data when compared to a group shown only numerical data. (Garcia-Retamero and Hoffrage, 2013)

At the FutureStack 13 data analytics conference, the point was made that people are particularly good at recognising visual patterns and that this recognition takes place at a subconscious level (New Relic, 2013). Understanding a data pattern without the need to even think is some recommendation to the usability and usefulness of visualisation.

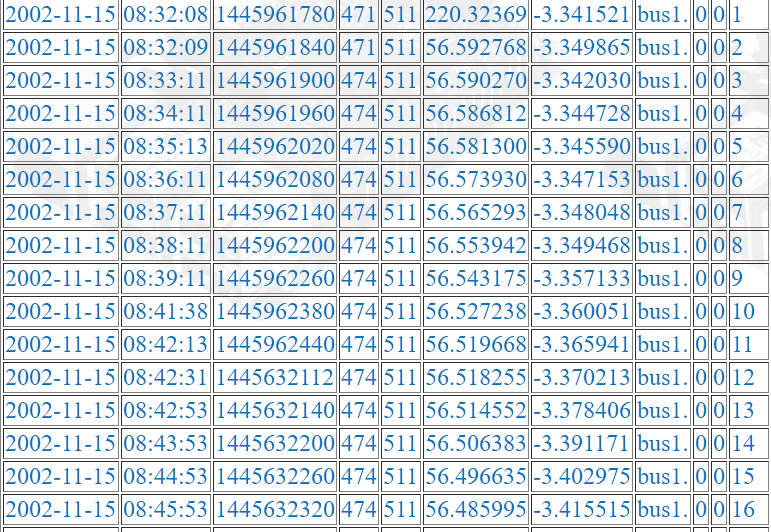


Fig1 Tabulated Data (Lambert,2015)

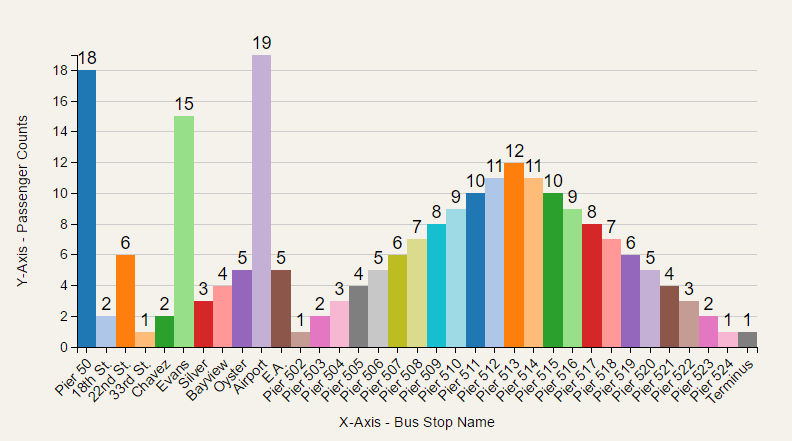


Fig2. Simple D3 graph (Lambert, 2016)

The idiom ‘A picture is worth a thousand words’ is most appropriate as Fig2, ‘Simple D3 graph’ demonstrates very clearly.

The images, Fig1 & Fig2 are presented from similar data in very different ways. Fig1 is simply the contents of a database presented using 16 lines PHP: Hypertext Preprocessor script. Fig2 contains over a hundred lines of script and numerous calls to functions from the D3.js library. The effort required to produce a graphically pleasing image is many times that of a simple tabulation, but the improved usability appears to justify the workload.

## 3.3 Data-Driven Documents (D3.JS)

It has been noted that there is no support for Scalable Vector Graphics (SVG) in older versions of Internet Explorer, specifically, versions pre- Internet Explorer 11, and this should be considered when the target audience is identified. (caniuse.com, 2016)

This restriction has been discussed with Dr Collins and he is happy that his customers will have access to computers using an up to date version of Google Chrome, so in this instance, compatibility will not be an issue.

D3.js is a JavaScript library of functions used to manipulate and visualise data and was developed in 2011 by Mike Bostock, Jeffery Heer and Vadim Ogievetsky.

D3.js binds data to the Document Object Model (DOM) and employs HTML5 and Cascading Style Sheets (CSS) to create visualisations as an SVG object drawn upon a HTML Canvas. (Bostock et al, 2016)

Data is most commonly supplied in the popular Comma Separated Values (CSV) or JavaScript Object Notation (JSON) formats. D3.js has the ability to process the GeoJSON (Geographical JavaScript Object Notation) which could possibly be implemented in future versions to integrate visualisations into geographical mapping visualisations. (DashingD3js.com,2016)

D3.js has mathematical functions and can arrange data visualisations based upon calculations of data sets, functions can select HTML elements such as <p> (paragraphs) and manipulate these to change colour, text size and transitions

Building a visualisation with D3.js is a systematic process where functions are invoked sequentially to steadily build a visualisation one piece at a time. One does not merely pass an array of data to a function that will process the data and produce a histogram as it is first necessary to construct labelled axes and gridlines.

It is necessary to append labels and to add CSS styling rules to various elements and this ensures that visualisations can be as individual as the developer, rather than selecting a standard pre-programmed chart.

D3.js becomes quite complex as functions that call functions as part of their own method are constructed. This complexity can produce some quite impressive and infinitely useful visualisations such as integrating data into animated graphics.

Spoof internet news site The Onion has used a graphic created by D3 creator Mike Bostock (Fig3 Concentric circles emanating from glowing red dot). This animation depicts the are affected by a supposed earthquake and contributes greatly to the visual explanation of this humorous effect. This animation does demonstrate the capabilities of D3.js and in a serious situation is obviously a most effective and useful tool.

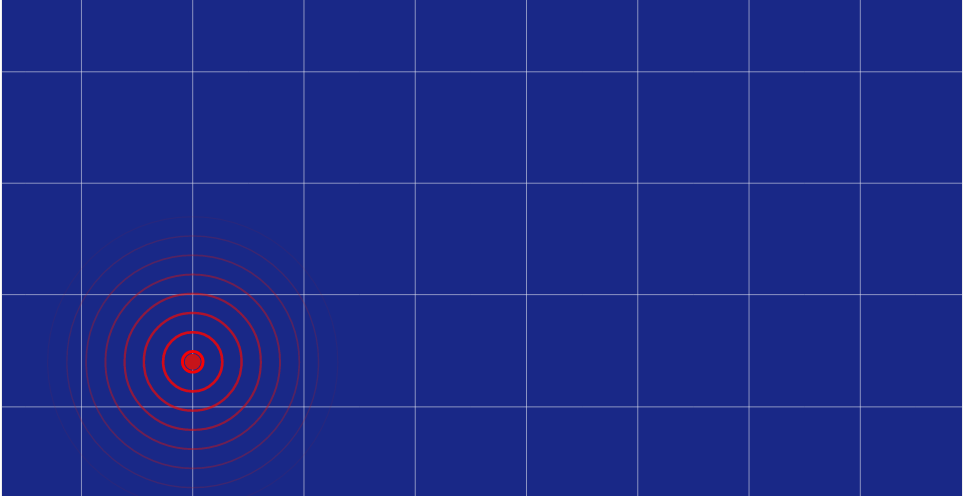


Fig3. Concentric circles emanating from glowing red dot (Bostock, 2016)

## 3.4 Dygraphs

In common with D3.js, Dygraphs is an open source JavaScript library that uses the HTML5 Canvas element. Problems with backwards compatibility exist with Canvas as they do with SVG, but as the Dygraphs documentation points out, “*Older versions of IE have a small market share that's decreasing by the day. Major websites, like the New York Times, have already dropped support for IE8. Future versions of dygraphs will do the same”.* (Dygraphs.com. 2016) Dygraphs was released in 2009 by Google Engineer Dan Vanderkam and is based upon an internal dashboard used by his team at Google. (Konigsberg, 2016)

Dygraphs is an open-source visualisation library and unfortunately only accepts a limited number of data types and does not support data in the JSON format currently used by Biodata. Many free tools for converting JSON files to CSV are available and conversion using JavaScript is possible (Praney, 2017), so it is reasonable to suppose that this obstacle could be crossed should it be necessary.

Visualisations built with Dygraphs are dynamic, reactive and when compared to d3.js, initially appears to be very simple as shown in Fig4.

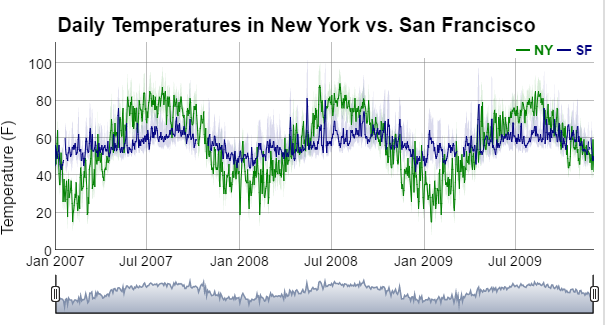


Fig4. Dygraphs Line Graph Example. (Dygraphs.com, 2016)

Indeed it appears that the only visualisations available are line charts of one type or another. The Dygraphs library of functions can be downloaded and addressed in the HTML document and accessed through a custom JavaScript file.

A Dygraph object is constructed using 3 arguments - An HTML Div element, a data series in the CSV format and some CSS style rules. (Mills, 2013)

Examples of Dygraphs charts suggest that they are more mathematical in presentation than examples built using D3.js, but the apparent simplicity of the library may well have some bearing on this.

## 3.5 Chart.js

Chart.js is another popular open source JavaScript visualisation library, albeit somewhat simpler than D3.js. Again, Chart.js utilises the HTML Canvas element, so Internet Explorer versions prior to Internet Explorer 11 will not be supported, but there is a pattern emerging where some of the more dynamic technologies are abandoning support for the more primitive browser.

Chart.js is capable of producing six different visualisations including Pie Charts, Line Graphs and Bar Charts (Murphy and Dondido, 2016). This gives more choice than the line graphs available in the Dygraphs library, and the finished articles are more visually pleasing than the utilitarian line chart, as shown in Fig5 below.

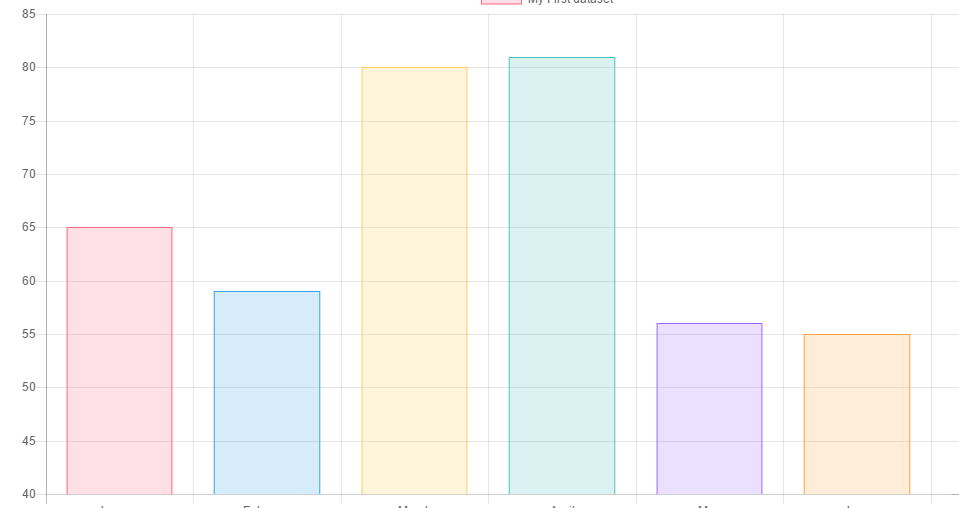


Fig5 Chart.js Bar Chart. (Chartjs.org, 2016)

Chart.js is similar in complexity to Dygraph as visualisations are selected from a library, rather than constructed from a collection of functions and it is necessary to present data formatted as CSV rather than JSON format.

# 4.0 Evaluation and Reflection

## 4.1 Database

There are always compromises to be made when selecting the technologies and software to be used on a new project, and in this particular project, the decisions affecting these technologies have already been made by others.

It is not the place of the developer to question these decisions too closely, as he will be unlikely be in possession of the reasoning used in that process. Despite this, the technologies have been reflected upon and opinions formed as to the suitability of these technologies.

If we firstly take the chosen database technology, MongoDB and the suitability for this project. It is without doubt that an everyday relational SQL database would have made the whole project somewhat simpler, indeed removed a whole layer of complexity.

If one large bus company made 300,000 database entries in one day, widespread integration of the system could quite conceivably involve several million entries each day. With this one fact in mind, MongoDB appears to be an extremely well-suited database system for this project and the complexities involved become insignificant.

There are a great many methods of visualising data. We have looked at only three, but there are dozens more one could evaluate and consider. Firstly, we shall evaluate the suitability of D3.js.

## 4.2 D3.js

D3.js is very well documented and has quite a following, with many tutorial videos and several e-books freely available online. In common with all three tools, D3.js is an open source technology, and as such, is free of charge to use.

This library occupies the top of the pack. The limit of visualisation with this library rests with the developer’s creativity (or skill) as visualisations are steadily built using many functions, rather than selecting one of a pre-built set and passing it the data to visualise.

Great understanding is required to successfully use this tool. One must understand what the client expects to get from the visualisation, one must understand the library itself in some detail and an understanding of the psychology of visualisation is necessary.

D3.js is very different from other technologies in the way visualisations are produced and is quite clearly much more labour intensive, but the trade off is the possibility to produce some quite stunning graphical elements.

## 4.3 Chart.js

Chart.js is a popular and well-documented tool. It is open source, reasonably easy to implement and provides an interesting and aesthetically pleasing, albeit very limited, set of visualisations.

Chart.js would be simple to master and would most likely solve the task in hand without too many complications.

The resulting visualisations would have a soft and minimalist look and present the information required of them. These visualisations would also be quite ordinary and adequate, but most probably, not be astounding.

## 4.4 Dygraphs

Dygraphs appear to sit technically in the middle of Chart.js and D3.js. The most basic of visualisations such as a simple line chart are extremely simple to construct and become steadily more complex.

The interactivity is a useful and interesting feature and ‘mouseover’ events displaying values for points on charts would be extremely useful when visualising a large data set.

Dygraphs visualisations appear to be quite flat and mathematical in appearance and do not carry the aesthetically and presumably psychological benefits of Chart.js or D3.js, but in many applications, this would be suitable.

With these points in mind, this project would work well with any combination of the discussed technologies, but with the application of a lot of effort and thought, the chosen tools of MongoDB and D3.js are most likely the best for this particular task.

# 5.0 Bibliography

Anon, (2016). *Can I use... Support tables for HTML5, CSS3, etc*. [online] Caniuse.com. Available at: http://caniuse.com/#feat=svg [Accessed 27 Dec. 2016].

Anon, (2016). *Cite a Website - Cite This For Me*. [online] Dygraphs.com. Available at: http://dygraphs.com/ie.html [Accessed 30 Dec. 2016].

Anon, (2016). *FAQ - Frequently Asked Questions | MQTT*. [online] Mqtt.org. Available at: http://mqtt.org/faq [Accessed 22 Dec. 2016].

Anon, (2016). *GeoJSON | DashingD3js.com*. [online] DashingD3js.com. Available at: https://www.dashingd3js.com/lessons/geojson [Accessed 29 Dec. 2016].

Anon, (2016). *Sharding — MongoDB Manual 3.4*. [online] Docs.mongodb.com. Available at: https://docs.mongodb.com/manual/sharding/ [Accessed 27 Dec. 2016].

Anon, (2016). *Video People Counting Systems | Video Turnstile*. [online] Videoturnstile.com. Available at: http://videoturnstile.com/ [Accessed 21 Dec. 2016].

Bostock, M. (2016). *Concentric circles emanating from glowing red dot*. [image] Available at: http://bl.ocks.org/mbostock/4503672 [Accessed 30 Dec. 2016].

Bostock, M. and et al, (2016). *D3: Data-Driven Documents*. Stanford CA.

Bostock, M., Ogievetsky, V. and Heer, J. (2011). D&#x0B3; Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12), pp.2301-2309.

*Breaking News concentric circles*. (2016). [video] http://www.theonion.com/video/breaking-news-series-of-concentric-circles-emanati-14204: Mike Bostock.

Bryant, N. and Wildfire, J. (2016). Webcharts – A Web-based Charting Library for Custom Interactive Data Visualization. *Journal of Open Research Software*, 4(1).

Chartjs.org, (2016). *Chart.js Bar Chart*. [image] Available at: http://www.chartjs.org/docs/#bar-chart [Accessed 31 Dec. 2016].

Dygraphs.com, (2016). *Dygraphs Line Graph*. [image] Available at: http://dygraphs.com/gallery/#g/range-selector [Accessed 31 Dec. 2016].

Few, S. (2016). *Data Visualization for Human Perception: The Encyclopedia of Human-Computer Interaction, 2nd Ed.*. [online] The Interaction Design Foundation. Available at: https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-computer-interaction-2nd-ed/data-visualization-for-human-perception [Accessed 31 Dec. 2016].

Garcia-Retamero, R. and Hoffrage, U. (2013). Visual representation of statistical information improves diagnostic inferences in doctors and their patients. *Social Science & Medicine*, 83, pp.27-33.

json-csv.com, (2017). *JSON to CSV - Convert your JSON to CSV for Free Online*. [online] Json-csv.com. Available at: https://json-csv.com/ [Accessed 3 Jan. 2017].

Konigsberg, R. (2016). [Blog] *the Dygraphs Blog*. Available at: http://blog.dygraphs.com/2013/08/announcing-dygraphs-100.html [Accessed 31 Dec. 2016].

Lambert, S. (2016). *Current File*. [online] Scottlambert.uk. Available at: http://scottlambert.uk/php/allData.php [Accessed 27 Dec. 2016].

Lambert, S. (2016). *Simple D3 graph*. [image] Available at: http://thewebsitebloke.co.uk/html/busStop1.html [Accessed 29 Dec. 2016].

Maclean, M. (2014). 1st ed. [ebook] D3 Tips and Tricks: Interactive Data Visualization in a Web Browser: Malcolm Maclean. Available at: https://www.amazon.co.uk/D3-Tips-Tricks-Interactive-Visualization-ebook/dp/B00IZ6P1KU/ref=sr\_1\_fkmr0\_1?ie=UTF8&qid=1483096269&sr=8-1-fkmr0&keywords=D3+Tips+and+Tricks+v3.x+Malcolm+Maclean [Accessed 30 Dec. 2016].

Mehrabani, A. (2014). *MongoDB high availabiltiy*. 1st ed. Birmingham, UK: Packt Pub.

Mills, B. (2013). *Short tutorial on deploying your first Dygraphs chart*. [image] Available at: https://www.youtube.com/watch?v=W-1XN8GFvus [Accessed 31 Dec. 2016].

Murphy, R. and Dondido, P. (2016). *An Employee Performance Simulation to Aide in Managerial Decision Making in a Target Driven Work Environment*. Dublin: Dublin Institute of Technology, p.9.

New Relic, (2013). *Data Visualization Design*. [image] Available at: https://www.youtube.com/watch?v=VG6fFjIFxbo [Accessed 31 Dec. 2016].

Parr, B. (2016). *IPv4 & IPv6: A Short Guide*. [online] Mashable. Available at: http://mashable.com/2011/02/03/ipv4-ipv6-guide/#zk.TJSPGZOqs [Accessed 31 Dec. 2016].

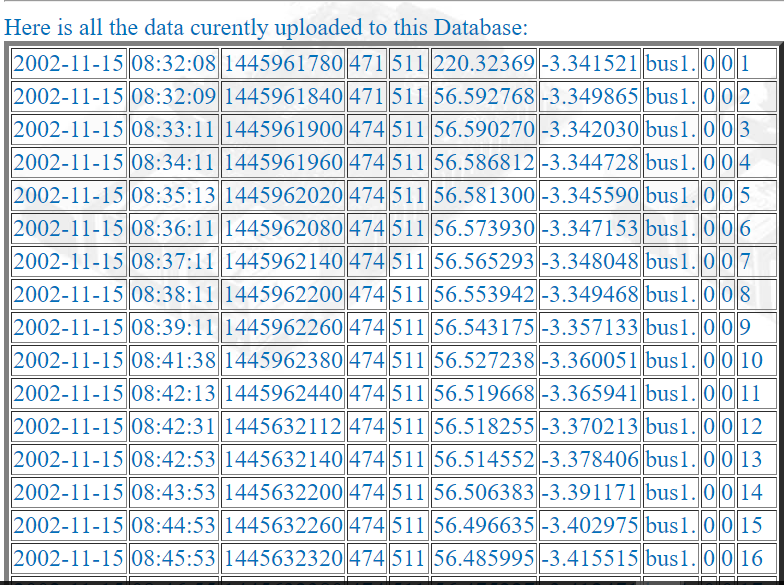
Praney, H. (2017). *How to convert JSON to CSV format and store in a variable*. [online] Stackoverflow.com. Available at: http://stackoverflow.com/questions/8847766/how-to-convert-json-to-csv-format-and-store-in-a-variable [Accessed 3 Jan. 2017].

Retail Sensing, (2016). *Automatic Passenger Counting for Bus and Train*. [online] Retail Sensing. Available at: http://www.retailsensing.com/automated-passenger-counting.html [Accessed 31 Dec. 2016].

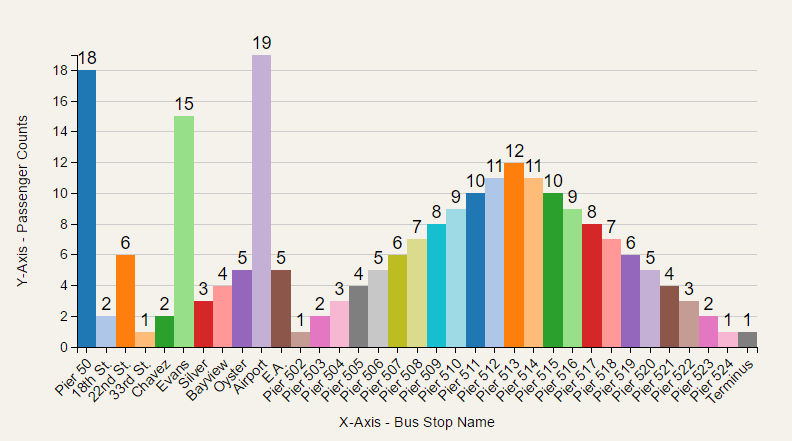
Zulu, Z. (2016). *Cheating bus drivers exposed | IOL*. [online] Iol.co.za. Available at: http://www.iol.co.za/news/crime-courts/cheating-bus-drivers-exposed-1653190 [Accessed 21 Dec. 2016].

# 6.0 Appendices

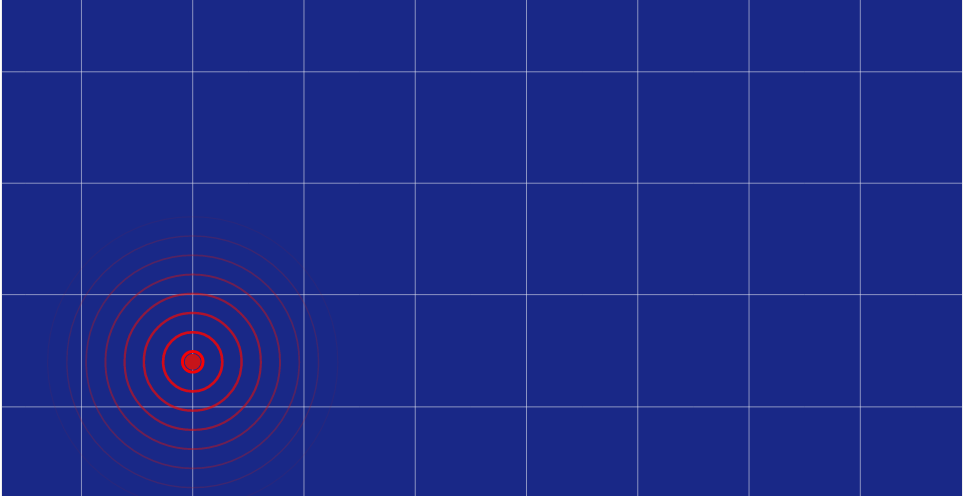
## Appendix 1. Tabulated Data (Lambert,2015)



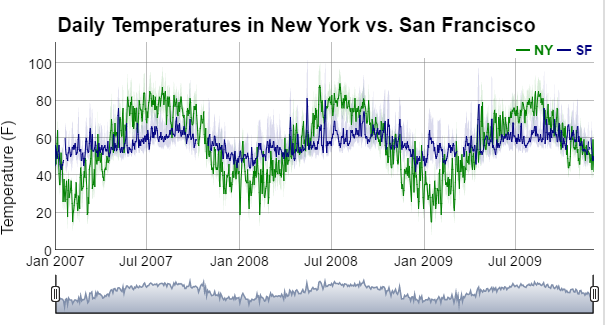
## Appendix 2. Simple D3 graph (Lambert, 2016)



## Appendix 3. Concentric circles emanating from glowing red dot (Bostock, 2016)



## Appendix 4. Fig4. Dygraphs Line Graph Example. (Dygraphs.com, 2016)



## Appendix 5. Chart.js Bar Chart. (Chartjs.org, 2016)

