Gbolaga Famodun

Full Stack Development case in point
Single Page Frameworks and Cloud Technology

Metropolia University of Applied Sciences
Bachelor of Engineering
Information Technology
Bachelor’s Thesis
7 December 2018
<table>
<thead>
<tr>
<th>Author</th>
<th>Gbolaga Famodun</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Full Stack development case in point Single Page Frameworks and Cloud Technology</td>
</tr>
<tr>
<td>Number of Pages</td>
<td>53 pages + 0 appendices</td>
</tr>
<tr>
<td>Date</td>
<td>7 December 2018</td>
</tr>
<tr>
<td>Degree</td>
<td>Bachelor of Engineering</td>
</tr>
<tr>
<td>Degree Programme</td>
<td>Information Technology</td>
</tr>
<tr>
<td>Professional Major</td>
<td>Smart Systems</td>
</tr>
<tr>
<td>Instructors</td>
<td>Janne Salonen, Lecturer (Project Supervisor)</td>
</tr>
<tr>
<td></td>
<td>Antii Piironen, Lecturer (Studies Coordinator)</td>
</tr>
</tbody>
</table>

Keywords
# Contents

List of Abbreviations

1 Introduction  

2 Web Development  
   2.1 Introduction  
   2.2 History  
   2.3 Forms of Web Development  
      2.3.1 Front end (Client Side) Web Development  
      2.3.2 Back end (Server Side) Web Development  
      2.3.3 Full Stack Web Development  
   2.4 Cloud Computing  

3 JavaScript  
   3.1 Introduction  
   3.2 History  
   3.3 Frameworks and Libraries  
      3.3.1 Frameworks  
      3.3.2 Libraries  

4 Cloud Services  
   4.1 Introduction  
   4.2 Type of Cloud Computing Service  
      4.2.1 Infrastructure-as-a-Service (IaaS)  
      4.2.2 Platform-as-a-Service (PaaS)  
      4.2.3 Software-as-a-Service (SaaS)  
      4.2.4 Serverless Computing  

5 Angular  
   5.1 Introduction  
   5.2 Typescript  
   5.3 RxJS  
   5.4 Zone.js  
   5.5 Angular Architecture
5.5.1 Modules 21
5.5.2 Components 21
5.5.3 Templates, Directives and Data-Binding 21
5.5.4 Services and dependency Injection 22
5.5.5 Routing 23

5.6 Forms 24
5.6.1 Template Driven Forms 24
5.6.2 Reactive Forms 25

6 Google Firebase 27
6.1 Google Cloud Platform 27
6.2 Firebase 28
6.2.1 Cloud Firestore 28
6.2.2 Cloud Functions 28
6.2.3 Hosting 29
6.2.4 Realtime Database 29
6.2.5 ML Kit 29
6.2.6 Authentication 30
6.2.7 Firebase Cloud Storage 30
6.2.8 In-App Messaging 30
6.2.9 Others 31

7 My Project - NewCityShop 32
7.1 Project Fundamentals. 34
7.2 Development Process 34
7.3 Future Implementation 42

8 Conclusion 43

References 44
List of Abbreviations

Models Data object mapping. The set of rules for mapping objects in a programming language to records in a database, and vice versa.

DBMS Database management system. Software for maintaining, querying and updating data and metadata in a database.

NoSQL A database (originally referring to "non SQL" or "non-relational") that provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.

SPA Single Page Applications. Web Application developing Frameworks, platforms and libraries that interacts with the user by dynamically rewriting the current page rather than loading entire new pages from a server.

API Application Programming Interface is a set of subroutine definitions, communication protocols, and tools for building software.

TCP/IP The Internet protocol suite is the conceptual model and set of communications protocols used on the Internet and similar computer networks.

WEB The World Wide Web (WWW), also called the Web, is an information space where documents and other web resources are identified by Uniform Resource Locators (URLs), interlinked by hypertext links, and accessible via the Internet.

URI A Uniform Resource Identifier is a string of characters that understandably identifies a particular network resource.

HTTP Acronym for Hypertext Transfer Protocol also called hypertext transfer protocol and can be defined as a protocol that web browsers and web servers to use for data transmission.

HTML Hypertext Markup Language is the standard markup language for creating web pages and web applications.
CERN European Organization for Nuclear Research.

MIT Massachusetts Institute of Technology.

STATE The latest and current data, contents and design of an entire application.
1 Introduction

The Internet is defined as the global system of interconnected computer networks that uses the TCP/IP to link devices all over the world. The idea of internet dated as far back as the 1960s based on the works of Paul Baran and Donald Davies who worked on various packet switches. Between 1960s and 1970s several other different network protocols were introduced, however during this period the internet was mainly used for academic and military purposes and provide communication and data transfer services. A lot of people today erroneously use the term internet and the World Wide Web interchangeably, but the two terms are obviously not synonymous.

The term web development can be said to be all necessary works involved in the development of a website for accessibility either on the Internet, the World Wide Web or even an intranet which limits accessibility to just within a restricted network. Web development range from development of just a simple single static page to a very complex multi-page web-based internet applications, shopping carts, social network services and so on. Also activities such as web design, web content development, web engineering, client-side and server-side programming, web server and network security configuration and e-commerce development are parts of the general term web development.

It was not until the late 1980s that the World Wide Web, which is the third main service provided by the internet apart from communication and data transfer became more apparent. The web is a global set of information which is available in various format ranging from documents, images, videos, audio files and so on and can be accessed using a special URIs. The URIs are symbolic links to servers, services, databases and other resources. The web is configured to be accessible by using preset or predefined protocols, the main access protocol to the web is the HTTP, and using the HTTP, information, data and business logic can be shared and exchanged by web services.

As a service the idea of web development came about as information need increases. The need to share information lead to the initial growth of the web, however, as the web grows it becomes very apparent that businesses could take advantage of this service and leverage on it to reduce cost, increase productivity and reduce waste. The development of various technologies chief among which is HTML the most basic web language which is responsible for displaying content to the user, Cascading Style Sheets which is responsible for adding allure and artistry to the contents displayed on the web pages and JavaScript which make web contents interactive, form the foundation of the technologies for accessing the web.

It is also suffice to say that at the back bone of every business is data. Be it clients data or proprietary business data, every good web development project always have some form of database behind it. In most cases it is important to ensure the security and safety of the data, thus development of various data technology solutions which provides security and authorization for data access. And we can either provision our database management software and server for our local computing infrastructure or even use the cloud computing database option which provides same service over some network connection.

metropolia.fi/en
and has recently become more popular in the industry, an example is the Google’s Firebase cloud service, these type of server are referred to as Serverless computing.

One of the main benefits of the web is that much recently various hand held devices have flood the markets, all with state of the art technologies to access the web and take any action or execute transactions that would have normally been possible only on a computer. Thus we can develop our web solutions both for devices and computers and reach more users.

The aim of this project is to develop a web application using a sample framework called Angular and a serverless cloud system to develop a full featured shopping cart web application that will be deployed in Google’s firebase hosting environment.
2 Web Development

2.1 Introduction

In the last few years, there have been major changes to the way web applications are developed and subsequently used. Nowadays the distinction is a single-page application and a multi-page application. Also cloud services has become very popular and are now becoming an integral part of application development, providing various services from database provision, storage, to hosting services. This document showcase the latest methodologies in the development of applications.

These are happy times for application developers, new and more advanced technologies which makes application development faster, easier and cheaper are widespread and these have open up more possibilities than ever before, the preference these days are Single Page Applications (SPAs) which dynamically rewrite its pages and updates only necessary segments rather than loading new pages from server. This singular functionality greatly limit unnecessary interruption of user experiences making the application acts as a desktop application. To achieve this, all needed code – HTML, JavaScript, and CSS – are either fetched with a single page load, or the relevant resources are dynamically loaded and the page is updated with the resources as needed, as the application respond to user actions. (1)

This means the line of separation of front end programming languages of HTML, CSS and JavaScript are becoming more and more blurry, as these new Technologies and Frameworks seek to extend the power of each and to ensure the developed applications are very intelligent, fast and secure.

It must also be noted that the need for businesses to reduce cost, especially technology related cost has led to the emerging prominence of cloud service providers, who provide these services for specify price. And what at first started as a service for corporation and big businesses alike has now become a mainstay in web and application development for small organizations and individuals and even now determines how applications are being developed. (2)

The importance of this cloud service platforms cannot be over emphasized for the provision of technological services needed for the successful running of businesses. As this cloud platforms come with a lot of extra functionalities all available for token prices.

This thesis seek to use all these latest trends and technologies to deliver a full stack shopping application that adhere to the best qualities of a good web application ranging from simplicity, elegance, usability, security, reliability, Good performance, Scalability and so on.
2.2 History

The idea of a global interlinked information system started with Tim Berners, a British scientist, when he submitted a paper titled “Information Management: A Proposal”, to his superiors at CERN and after his idea was accepted, he was able to invent the Web in year 1989. He designed it to enable researchers at various universities and institutions around the world to share information.

By 1990, he had written the HTML, URI and HTTP which are the three cornerstone of today’s web. He also developed the first browser called “WorldWideWeb.app” and the first web server referred to as “http”. He published the first web page on the internet and invited people to join the web community. Tim left CERN and moved to MIT in 1994 where he later founded the World Wide Web Consortium (W3C).

The rapid growth of the Web means several people have access to it and thus more advanced technologies were built to increase the power and the productivity of the Web. Examples are more powerful browsers like Mosaic Netscape browser, CSS (Cascade Style Sheet), and a programming language for the web PHP was introduced. All these lead to the creation of several services on the Web some of which are Yahoo, Match.com, Netflix, Google Search engine which were all introduced before the end of 1998.
2.3 Forms of Web Development

Web development is vast, several sections are involved in the creation of a full featured Web site. Web development entails all activities from Information gathering, planning, layout design, content writing and assembling, coding, testing, database connection and CRUD operations, to server loading. (5) All these steps are mainly divided into two major parts – front end also called client side development and back end development called server side development.

2.3.1 Front end (Client Side) Web Development

The front end of a web site includes all the parts of a web site that the user interact with directly, all activities that produce the interface that users interact with from graphics design using software like fireworks and Photoshop to using technologies like HTML, CSS and JavaScript. It includes several components like web layouts, buttons, colors, pictures, forms and much more.

The three most important parts of client side development are HTML, CSS and JavaScript, and together the three form the building piece of any front end web development project. Most client sides are developed using these three languages or using libraries and frameworks which are basically combination of the three. Examples of such libraries or framework are Angular, React, JQuery, and Bootstrap. (6)

Using all of these technologies client side programming helps to fulfil two major objectives: which is to provide a rich user interface (UI) and a good user experience (UX). A rich user interface has to do with a good and optimal general design of the web site, while the user experience has to do enhancing user experience on the web platform.

It must be noted that much recently the client side development has fallen under two broad design patterns: Single page applications and multiple page applications. Single page design patterns are much recent and have change the way application design are thought of and are implemented. Below we briefly describe the two.
2.3.1.1 Single Page Applications (SPA)

A much recent technology, this front-end application development platform re-display its contents in response to user actions autonomously without having to make any requests to the server to fetch new HTML. Meaning user click and mouse actions get their corresponding response right in the front end.

While there are different types of SPAs based on the fact that they function slightly differently one from the other, most of them rely on browser behavior and native API to enable their main functionality. The main thrust here is how the technology maintain the “state” of the application throughout the platform, such that new users are able to see all the most recent published changes on the web site automatically.

There are basically two ways of maintaining states either externally (location based) or internally. Internal state SPAs are very much limited in that they have only one “entry” into the application, so all users have to start from the root of the application, and there is no external representation. However external state, which are also called location-
based SPAs are able to update users views as they navigate hence no need to start from the root.

Other important things to consider in SPAs are Location Primer which showcased how the SPAs used the “window.location” for navigation. There is also Router Matching: which use routes that define the location the router should match. The Document Object, Session History and the History API are other very vital components of the SPA technology. (7)

2.3.1.2 Multi Page Application

The older and traditional web technology, typically every requests like to display new data or send a set of data back to the server lead to a request to display new sets of HTML from the server in the client browser. It is usually larger and bigger than SPAs and usually have many levels of user interface, thus it is mostly more complex to implement though more recently, using AJAX it is now a bit easier to implement.

It is worth mentioning that using this approach lead to both the front end and back end of the application being more tightly coupled thus making development more complex and time consuming. Maintenance and update are also often quite cumbersome, and it is also not possible to use back end application for mobile application development. (8)
2.3.2 Back end (Server Side) Web Development

Every activity that involves development on a web server that produces corresponding response to user or client requests are referred to as back end development. It is the
part of the web site that client or users do not see. It is used to implement various functionality like Authentication, authorization, file uploads, data connection and database management.

Several different programming languages and technologies can are used in the development of the server side, ranging from ASP.Net, PHP, Python, Java, JavaScript and Perl and so on. These languages are used to create what is mostly referred to as web services or APIs (Application Programming Interface) which produces customizable response depending on the requirement of the application.

Below we try to expound on the subject of servers and databases as integral parts of back end web development.

2.3.2.1 Servers

Servers are super computers, machines or programs responsible for serving files as response to users when they made request for these files. There are several types of servers ranging from proxy, mail, virtual, web, network and file servers. Here though we are referring to web servers.

Web servers uses HTTP protocol to serve response that become web pages to users when they are interpreted by the users HTTP clients. Web servers can be located locally on a developer machine, this is especially true during development. Although while publishing the application it is usual for the web server to be changed to a global server so that it can serve web pages or responses to all requesting clients all over the internet. It is worth mentioning that these global servers are always up and running without interruption. Some examples of these servers are Google Firebase, Heroku and AWS S3.

2.3.2.2 Databases

Every good web application always have some form of database management associated with it. Database is a collection of organized information that is stored but can be accessed, managed and updated as necessary. Most database have some form of built-
in index technology which could be used on data to ensure they are easily found and accessed.

There are several types and forms of database management systems based upon their design and underlining technology. The most common ones are Hierarchical, Network, Relational, Non-relational and Object Oriented databases. In Hierarchical and Network types of databases the data is more structured in a parent/child relationships, Object Oriented databases on the other hand add object oriented programming capabilities to database functionalities while Relational databases have data stored in different access control tables, with each of them having a key field that identifies each rows or records. Relational databases also use the SQL (Structured Query Language) as their primary tool of managing and administering the database. Examples of relational database are MySQL, Oracle, SQL Server and so on. (9)

For Non-relational databases, also commonly known as No-SQL Database, the data can either be structured or non-structured and can be stored in either single or several virtual servers, it is more suited for the purpose of storing very large chunks of unstructured, distributed data.

There are four main types of No-SQL databases, the key-value type is a big hash table of keys-values pairs, example is Amazon S3, there is the most popular Document-based type examples are CouchDB and MongoDB, also the Column-based type in which each storage block contains data from just a column example is the HBase, Cassandra and the Graph-based data store a type of network database that use nodes and edges to represent data, example is the Neo4J.

Depending on the manner in which a database service is provided they could also be divided into cloud and non-cloud database service. Non-cloud are database infrastructures that are provisioned locally on the server of the recipient organization, while Cloud databases are provided as a service since they are running on various cloud computing platforms either in public, private or hybrid forms while providing high availability, scalability and fault tolerance to the recipient organization without the need for physical database servers. There are several other advantages to this kind of model for organization chief among which is reduced cost of maintenance. (10)
2.3.3 Full Stack Web Development.

This involves both the front end and back end development process, and to be regarded as a full stack developer an individual must have an adequate knowledge of HTML, CSS and JavaScript and one or more back end languages. More recently organizations are more interested in recruiting those with these kinds of skill set as they are able to build more robust and meet the requirement of the other parts of the applications they are building since they have the requisite knowledge of the other parts of the application and can dive into any parts of the entire application if needed.

More recently it is now possible to use JavaScript or its variants in both the front end and back end of a web application. By using various JavaScript libraries like React or Framework like Angular on the front end and while using the NodeJs on the backend, and this is possible with most common database systems.

2.4 Cloud Computing

Cloud computing entails the delivery of computing and server resources such as database, server, storage, applications and several other IT resources through the internet. This means instead of acquiring various physical massive computing resources individuals and businesses can now concentrate on just delivering their services or computer programs by leveraging on the cloud.

Programmers can also now concentrate their efforts in writing and developing quality applications and deploying them on the cloud. The applications will be automatically initialized by various cloud events which are already developed into the cloud infrastructure. Other advantages of cloud computing apart from reducing development time is that application development becomes easier and takes less effort. The main disadvantage of the technology is that application cannot be run offline and also it cost money depending on usage of computing resource and band width. Microsoft Azure, Amazon web services and Google Cloud are some of examples of cloud services.
3 JavaScript.

3.1 Introduction

Recently JavaScript has emerged as the most prominent programming language in web development. It is the most widely supported programming language by all modern web browsers and devices. It is a high level language that is easy to understand and useful for implementing Object oriented programming techniques. With HTML and CSS, JavaScript has conquered the world of front-end web development.

With only HTML and CSS, web pages are very static and lack user interactivity, it is nothing more than a sign board, JavaScript as a programming language was introduced to solve this problem and make web pages more interactive. JavaScript has very small ‘imprint’ meaning it consumes little memory and executes very fast. It is also multi-paradigm which means it supports procedural or event-driven, Object oriented and functional programming. It’s so widespread that most browsers have a separate dedicated JavaScript engine to execute it.

Recently, JavaScript engines are now embedded in several other host software like web servers, database, word processors and PDF software. In fact JavaScript are now available in runtime environments and can now be used to write mobile and desktop applications.

3.2 History

The history of JavaScript goes hand in hand with that of web browsers, since it was created primarily to make web pages more interactive. In 1993, the first graphical web browser called the NCSA Mosaic was released, the browser was very important in the growth and expansion of the web, Netscape in other to challenge Microsoft for user adoption of web technologies, collaborated with Sun Microsystems to incorporate a more static Java into their browsers went ahead to recruit Brendan Eich with the goal of creating a scripting language to complement Java and increase user interactivity on their browser.
Originally developed under the Mocha, the language was originally called LiveScript but was later renamed JavaScript by December 1995. By 1996 Microsoft Corporation also released their own script called Jscript which was basically a reverse-engineered version of JavaScript and was part of Internet Explorer 3, though it has the additional functionality in that it was available for server-side scripting, however its implementation was noticeably different from the JavaScript has released by Netscape. Thus programmers find it difficult to create a single web site that can function across both browsers.

In other to resolve the issues Netscape submit their version of JavaScript to ECMA (European Computers Manufacturer Association) International to shape out a standard specification which other browsers could implement based on the work done by Netscape, however for the next ten years ECMA could not accomplish this aim since Microsoft refused to submit their work. However in July 2008, both parties eventually met at Oslo and an agreement was eventually reached in early 2009, leading to the release of ES5. After six years ES6 or ES2015 was released with a completely different structure and syntax of JavaScript. ES9 is released in 2018.

3.3 Frameworks and Libraries

3.3.1 Frameworks

A software framework can be defined as a detailed or conceptual environment where universal code with general functionality can be carefully selected to be overridden or made even more specialized by developers in order to provide application-specific software. Thus a framework provides standardization in the development and deployment of software applications, products and solutions. They many times contain other support programs, code libraries, APIs, compilers and tools which serve as a link to the different components in the framework.

It should be noted though that Frameworks are quite different from normal libraries, in a few different ways. For example the general program flow is not dictated by the developer but by the framework, also a developer can extend the framework to provide specific functionality and also the fundamental framework code should not be modifiable but can only be extended. Examples of JavaScript framework are Angular, EmberJs, and Node.
3.3.2 Libraries

A software library is a piece of programming code that can be used in the development of software programs. Libraries may include documentation, configuration information, classes, and written code samples among other things. Often libraries have a well-defined interface through which certain specific behavior could be invoked thus reducing the need to re-write specific sets of codes. Thus they usually come pre-packaged with a set of pre-implemented functionalities which are designed to help programmers carry out frequently executed tasks. The difference between a function of a library and a function local to a program depends mainly on how the function is used or called.

Usually libraries can be used by several programs that have no connection to each other. It must be mentioned that most compiled programming languages like Java and Python have their own standard libraries and programmers can also create additional custom library stack. Example of JavaScript library are React and jQuery.

Figure 4: Framework Vs Library.
4 Cloud Services

4.1 Introduction

Cloud services are computing resources provided for a price to individuals or organizations by a cloud computing provider. Amazon was responsible for making the term ‘Cloud’ popular when they released the Elastic Compute Cloud product in 2006. However the word cloud has been used as early as 1982 incidentally as a metaphor for the internet. Telecommunications companies in the 1990’s also offered and shared virtual private network (VPN) services at lower costs as a precursor of the cloud computing technology.

The goal of cloud computing is to enable users benefit from several technologies without the need to have a thorough understanding of any of them. The cloud helps users to increase productivity, efficiency and cut cost by eliminating the time it would have taken to provision several IT obstacles, and allow users focus on core their business.

4.2 Type of Cloud Computing Service

Cloud services can be described in two different ways, either in terms of its deployment model or the kind of service on offer by the service. In terms of deployment cloud service can be classified as public, private, hybrid or community cloud. These forms of cloud services have been discussed previously.

In terms of offered service, we have Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), Software-as-a-Service (SaaS), and also Storage, Database, Information, Process, Application, Integration, Security, Management and Testing-as-a-Service.

4.2.1 Infrastructure-as-a-Service (IaaS)

The common type of cloud service, it offers data storage disks and virtual servers as a service. It usually provide several high-level APIs to mask the underlying physical computing hardware. Hypervisors are used to run the virtual machines and are also able to
scale services up or down depending on the client’s needs. Containerization is also possible and offers higher performance than virtualization, since the technology require no hypervisors and can also auto-scale computing resources depending on user’s needs.

All these resources are provisioned by providers from their extensively large pools of equipment located in data centers. And users can installed operating systems images and applications software on them in order to use them to deploy their applications.

4.2.2 Platform-as-a-Service (PaaS)

This entails the provision of an application hosting environment with the capacity to deploy applications and configure the general environmental settings of the host. The vendors are able to provision a toolkit, the needed standards, channels for distribution and payment if needed, in all these entails everything needed in a development environment by application developers.

In a typical PaaS model several resource like operating systems, programming-language execution environment, database, and a web server are among the underlying solutions provisioned upon which the user then run their application without having to acquire and set up those servers themselves.

Common examples like Microsoft Azure, Google App engine and Oracle Cloud Platform are available for a price, with the additional capacity to extend available computing resources to match the hosted application demand.

4.2.3 Software-as-a-Service (SaaS)

This entails the access granted to a consumer to use a provider software applications and corresponding database running on cloud infrastructure. This application can be accessed over the internet with another client service like a web browser or a program interface. SaaS is often called ‘on-demand’ software and is mostly priced on pay per use basis. The consumer does not have any access, and no management privileges to the underlying cloud infrastructures.
One major advantage of SaaS is that users get to use software without having to install them on their own computers, also updates to the applications will not have to force users to install new software as software are updated centrally. The noticeable drawback however is that user’s data will be stored on the provider’s server which can allow unauthorized access to such data.

4.2.4 Serverless Computing

A rather newer technology, this is a code execution model where user request are serviced by virtual machines which are managed by the provider. The user’s code are deployed directly to the cloud provider instead of any server, and all deployments are handled by the cloud itself. Using the necessary events on the cloud, the user’s code are deployed and run as necessary.

This is very advantageous to the programmer since they only have to worry about their codes, however this solution cannot be used offline.

Figure 5: Cloud Services Types and Examples.
5 Angular

5.1 Introduction

Formerly known as AngularJs, this technology is a structural software framework for dynamic web applications. This framework is open-sourced and designed by engineers at Google. Angular is front-end web application platform which provides lots of features out of the box, like data-fetching, state management, development language, build chain and others. One of the major selling point of angular is Typescript which is the proprietary programming language dedicated to the development of applications using the angular platform. When Angular was first released in year 2009, it was called AngularJs and like the first release of most Frameworks, it was not perfect, the bundle size was quite huge and its architecture and digest loop system was chaotic and this set limitations on its performance. To address this glaring setback Google embark upon a complete re-write of the Framework and by September 2016 they roll out the brand new and much improved Angular2+ which is commonly referred to as just Angular. Ever since they are released several updates on the version and as at the time of this writing we have version 7.0 as the stable release of angular.

Angular is very useful for enterprise level applications since it is full featured and uses Typescript which is Object oriented programming language. It has a strong co-operate support from Google. Angular version seven has just been released in the month of October 18, 2018. There are three important pillars of angular, which are the main building blocks of the framework, they are: Typescript, RxJS and Zone.js

5.2 Typescript

JavaScript have lots of short comings when used in the development of large-scale applications, lots of challenges emerges when dealing with complex JavaScript code this lead to a high demand by programmers for a custom tool that can help the development of components in the language. The tool they believe should not break compatibility with the standard JavaScript language and must have cross platform support. Thus lead to a
series of syntactical language extensions for the JavaScript compiler that interprets this extensions into regular JavaScript.

Though JavaScript have a lot of short coming, it continue to get more popular and attract more users, to the extent the developers of other programming languages like C++, C# and Java were attracted to it, but the syntax of the programming language is very different from the object oriented programming language that these users are very much used to.

For all the reasons mentioned earlier Typescript was created and released to the general public as open source programming language first in October 2012 after two years internal development at Microsoft, it has supports for several features like classes, modules and the arrow function syntax. Typescript is a superset of JavaScript which can be used both in the client-side and the server-side. A Typescript program in a file has a ‘.ts’ or ‘.tsx’ file extension and complies into a JavaScript with ‘.js’ extension. (11)

By default angular framework uses the Typescript programming language, an example code can be seen below, that illustrate the process:

class Employee {
  public EmployeeNumber: string = "";
  constructor() {
    this.EmployeeNumber = "1";
  }
}

// This creates a file called ‘employee.js’ and converts the above codes into the following:
var Employee = (function () {
  function Employee() {
    this.EmployeeNumber = "";
    this.EmployeeNumber = "1";
  }
  return Employee;
})();

Listing 1. The Conversion of Typescript codes to JavaScript codes
5.3 RxJS

This is a very important library used mainly for reactive programming by taking advantage of Observables. Observables are asynchronous stream of data which programmers can subscribe to. This library has several built-in operators that allows observation, transformation and filtering of streams of data, even combining multiple streams of data to create more advanced and powerful dataflow in a single operation. Angular use this library to be handle all data and information passed to it as observable stream from routing parameters to HTTP responses. (11)

5.4 Zone.js

Zone is a concept that provides multiple catch on asynchronous operation so that programmers can get feedback or notification of their states, cancel tasks, handle errors or extend execution with additional operations. Thus Zone give the execution context of asynchronous operations, it does this by patching the standard APIs like DOM event listeners into a Zone. (11)

5.5 Angular Architecture

Angular implements its various core and optional functionalities as a group of Typescript libraries that developers can import into their application. The most basic of these groups are the NgModules which aggregate related codes into functional groups. In additional to the root module which is always present in every angular application and allows for bootstrapping of the application, the angular framework has components which is used to define various kinds of ‘views’ that a programmer can extend to implement his program logic and also ‘services’ which provides specific functionality and could be injected as dependencies making the program more modular, efficient and reusable. Below we attempt to describe these major building block of an angular application.
5.5.1 Modules

Angular modules are referred to as NgModules, they are different but also complement JavaScript modules, and they define the guidelines for the aggregation of closely related components that are used in an application domain. Thus a particular NgModule can ally its components with other related code block to form functional units.

One of the very important capabilities that NgModule inherit from JavaScript modules is its ability to import functionality from other NgModules and also export their own functionalities to other NgModules. A very explicit example of this is in the use of the router service in an angular application, this is accomplished by importing the Router NgModule. This importing is accomplished easily by taking advantage of a concept called lazy-loading which is an act of loading modules only on demand. This concept have an added advantage of minimizing the amount of code that needs to be loaded at startup.

5.5.2 Components

The components are classes with metadata information about their type and instruct angular about how to use them. They contain application data and logic that is connected to an HTML template view and could be manipulated to determine how an element of the view should look and behave on the screen.

Components can be created, updated and destroy using the various ‘life cycle hooks’ provided by the angular framework. And a component class can be recognize by using the @Component() decorator which provides the components template and the related component-specific metadata.

5.5.3 Templates, Directives and Data-Binding

Templates are the defining views of components, they are the HTML we use to show our pages. They also can help in improving user experience by making use of a special property called ‘pipes’ which can transform data into a more user friendly values for display. Templates also combines with special angular markup that have the capability to modify the HTML elements before they are displayed, these angular markup are known
as Directives. Thus directives adds behavior to the DOM elements, and they are define by the ‘@decorative’ decorator. It is also possible to add many directives to the DOM elements.

Angular is known to have three types of directive/decorative which are Directive-with-a-template an example of this is a Component. There is Structural Directives which can add, delete and replace DOM elements and there is the Attribute directives which can change the appearance of the DOM elements.

Data-binding is a major feature of any JavaScript framework and it is the concept behind the connection between the application data and the DOM. There are three major types of data binding, the first is Event binding which allows the application to respond to user actions and user inputs in the target environment by updating the application data. Another is the Property binding which allows a programmer to introduce values that are computed from the application data into the HTML. It is worth mentioning that before a view is displayed to a user angular check out all the annotated directives and process the binding syntax to modify the HTML elements and the DOM according to the implemented program data and logic. The third is the two-way binding which indicates that the changes as reflected in the DOM when implemented are also reflected in the program data.

5.5.4 Services and dependency Injection

A service is a class that consist of any function or feature that helps serve a specific purpose. This means that for any data or program logic in our application that is not linked to a specific view and that we would like to share across several different components we can use a service class. Like in other cases use a decorator ‘@Injectable’ in the naming of the service class to provide the necessary metadata information needed by angular to inject the service into various components as a dependency.

Dependency injection allows programmers to add several functionalities to components without having to re-write the necessary code, it is so important in keeping our application lean and efficient, as several common tasks are delegated to services and the needed services can be injected as a dependency service throughout the web application.
5.5.5 Routing

The routing in Angular is provided by the Router NgModule, this service enable a programmer to define a navigation path in the various application states and view levels in the application. The router perform the required function when a user enter a URL into the browser, when user clicks on a link on a page or when the browser back and forward button are pressed.

In Angular the router maps URL links to views instead of pages, this is achieved by the fact that the router intercept the behavior of the normal behavior of the browser when a user click on links and show or hides views. The router interprets links in accordance to the programmer stated navigation rules and data state, and also logs the browser navigation history to ensure the proper functioning of the forward and backward history button.

The navigation rules are defined by linking navigation paths with the corresponding components. (12)

Figure 6: Angular overview and Architecture
5.6 Forms

There is hardly a web application without some kind of forms, forms are used for several functionalities in an application ranging from user registration, log in, order placement, different kinds of data-entry tasks, schedule a meeting, and so on. In Angular creating and using forms is quite easy, though is a bit different from the regular HTML. In all, it is important that forms are easy to use for users and efficiently get their jobs done.

In Angular there are two distinct ways of working with forms namely template forms and reactive forms.

5.6.1 Template Driven Forms

In Template forms, most of the development work is done in the views/template. Angular provide a FormsModule from @angular/core library which should be imported into the app.module.ts file in order to work with template forms.

We use the model form controls by adding the ngModel directive and the name attribute which creates a two-way data bindings for reading and writing input-control values. Thus any form data that we want angular to have access to we add the ngModel to such form element. Another directive ngForm is also added to the form template and an onClickSubmit function is added with the form value passed to it. All these code are in the views, however in the component class we define the onClickSubmit method which is called when the user submit the form. (13)

```
<form #uLogin = "ngForm" (ngSubmit) = "onClickSubmit(uLogin.value)" >
  <input type = "text" name = "emailid" placeholder = "emailid" ngModel>
  <br/>
  <input type = "password" name = "passwd" placeholder = "passwd" ngModel>
  <br/>
  <input type = "submit" value = "submit">
</form>
```

```
import { Component } from '@angular/core';
import { MyserviceService } from './myservice.service';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
```
export class AppComponent {
  title = 'Angular Project!';
  todaydate;
  componentproperty;
  constructor(private mybigservice: MyserviceService) { }
  ngOnInit() {
    this.todaydate = this.mybigservice.showTodayDate();
  }
  onClickSubmit(data) {
    alert("Entered Email id : " + data.emailid);
  }
}

Listing 2. Sample Template driven Form and its corresponding component

5.6.2 Reactive Forms

To use the Model driven forms we have to import the ReactiveFormsModule from @angular/forms into the app.module.ts file. Then we also need to import a few other modules from the same @angular/forms into the component in order to use the model driven form.

The various formdata variables are also initialized at the start of the class in the components and are also initialized with default values using the FormGroup in the template/view.

It is worth mentioning that form validation is possible and very similar in both template and reactive forms. Programmers can use the built-in form validation features or create their own custom validation features. Built-in features like mandatory field, minlength, maxlength and pattern are already available out of the box and can be accessed using the Validators module. Submit buttons can be toggled for enable or disabled state depending on if there are errors in the form, the error value which can be obtained by using the ‘formdata.valid’ function. For custom built validation features we define custom functions and add necessary details in the FormControl object. (14)

```html
<div>
  <form [formGroup] = "formdata" (ngSubmit)="onClickSubmit(formdata.value)" >
    <input type = "text" class = "fortextbox" name = "emailid" placeholder = "emailid" formControlName = "emailid">
    <br/>
    <input type = "password" class="formbox" name = "password" placeholder = "passwordd" formControlName = "passwordd">
    <br/>
    <input type="submit" [disabled]="!formdata.valid" class="forsubmit"/>
  </form>
</div>
```
Listing 3. Reactive Forms and Validation Example
6 Google Firebase

6.1 Google Cloud Platform

Like many other big tech organization, google have a cloud platform which basically is a collection of cloud services that runs the same infrastructure that google use internally for their end-user products like Google Search and Youtube. The Google cloud platform provides different services that range from computing, data storage, data analysis and machine learning. All of these products are available in free tier which can be used for development purpose and paid subscriptions. The platforms are very secure and easy to use for customers. (15)

Released April 2008, Google cloud platform offered over 90 services in the Infrastructure as a service, Platform as a service, and Software as a service in the categories of Networking, Compute, Storage and Databases, Big data, Machine Learning, Identity and Security and Management and Developer tools. The platform is available in 17 regions and 52 zones. Each regions is an independent geographic location that consists of many zones.

Some of the cloud services by Google are App Engine, Compute Engine, Kubernetes engine, and Cloud Functions all under the compute cloud infrastructure, also there are the Cloud Storage, Cloud SQL, Cloud Datastore, Cloud BigTable and a few others, under the Storage and Databases cloud infrastructure. Then there is Virtual Private Cloud, Cloud Load Balancing and others under the Networking Cloud Infrastructure. The BigQuery, Cloud Dataproc, Cloud Data Studio and others are also available under the Big Data infrastructure. The Cloud Speech-to-Text, Cloud Text-to-Speech, Cloud Translation API, Vision API, Video Intelligence are some of the services under the Cloud AI infrastructure. The Cloud Console, Stackdriver, Cloud Shell, Cloud APIs and some others are under the Management Tools cloud infrastructure, there are also Identity and security cloud infrastructure, the IoT infrastructure, API platform infrastructure each having their own cloud based services.

Though all of these services are available individually, Google also from time to time create different products by groping business related services together and delivery them
as a single suite of solutions for business enterprise, some example of these products are: Apigee and Firebase which are cloud computing related products, G Suite, Drive Enterprise and Cloud Search are also other cloud products on the Google platform. (16)

6.2 Firebase

Firebase is a google cloud platform that makes developing mobile and web applications very easy for developers. Originally developed by Firebase Inc. in 2011 it was later acquired by Google in 2014. Google’s Firebase is an all-in-one back-end infrastructure often referred to as Backend-as-a-Service (BaaS) cloud platform. The platform currently has about 18 products which are used by over 1.5 million applications. These 18 products covers most of the important features needed to run an enterprise level web and mobile application. Chief among these services are Cloud Firestore, Cloud Functions, Hosting, Realtime Database, ML Kit, Authentication and Cloud Storage and are discussed below.

6.2.1 Cloud Firestore

The Google Cloud Firestore service is a very malleable, scalable NoSQL database for mobile, web and server application development and is used to store and sync data for client and server side development. It ensures that data is in sync across all associated client apps through realtime listeners and also offers offline support for connected mobile and web application irrespective of network latency or internet connectivity. It also seamless integration with other Firebase and Google Cloud Platform products and services.

6.2.2 Cloud Functions

The Google Cloud Functions is an example of a Function-as-a-Service platform, it is a serverless execution environment used for building and connecting cloud services. Cloud Functions make it very possible for programmers that develop various code functions into the cloud that they can use to control their other cloud services, they do this by attaching events emitted from their other cloud services and infrastructure to the cloud functions whose functionalities are then fired. The code executes in a fully managed
environment and the programmer does not have to provide any infrastructure or provision any server. This cloud functions can be written with NodeJs 6.14.0, NodeJs 8.11.1 or Python. (17)

6.2.3 Hosting

The Google Cloud Hosting is a platform providing both static and dynamic web hosting capabilities. It was launched in May, 2014, and it’s helpful in providing services and infrastructure needed to deploy scalable, globally-available multi-tiered web application either built from scratch or just a static web site. This means developers are able to focus just on their application while the Google Cloud Hosting platform handles the rest. The infrastructure is so powerful it can handle billions of request per day and also automatically scale down when the need arises. Files are delivered through HTTPSecure and Secure Sockets Layer encryption (SSL) over content delivery Network (CDN).

Programmers can deploy different applications written with programming languages ranging from Python, Java, PHP and several others, also software Frameworks can be deployed easily. And it even allows own-runtime with App Engine Flexible environment.

6.2.4 Realtime Database

This is a Google cloud-hosted NoSQL database, where data is store in a JSON format and is synced with all connected clients and remains available even when applications goes offline, thus all cross platform apps share the realtime database and receive instant updates with the latest data.

The database could be accessed with a rest API and has bindings for may JavaScript frameworks and libraries like Angular and React.

6.2.5 ML Kit

Still at beta stage as at the time of this writing, the Google Cloud Machine Learning Kit is mobile learning system, it was launched in May 8th, 2018. And offers different sets of
features ranging from text recognition, face detection, barcode scanning, image labelling and landmark recognition. It is available for iOS and Android developers.

6.2.6 Authentication

Applications need to know the identity of its users, and Google Firebase Authentication service provides the feature to authenticate users using client-side code. This service is important to ensure that users’ data are securely saved in the cloud and provide personalized experience across all of users’ devices.

This service is provisioned through backend services, ready-made user interface libraries and intuitive SDKs which support various forms of authentication from passwords, phone numbers and other providers like Google, Facebook, Twitter and more.

6.2.7 Firebase Cloud Storage

This cloud solution is developed for the need to store and serve user-generated contents like videos, audios, photos, and other files. It is a stand-alone solution that require no server and could be used in both mobile and web applications.

It is very scalable hence developers do not need to worry about available space, it is secured, as files can be linked specifically to a user or group of users using storage security rules and the network is very resilient as uploads and downloads are automatically retried in case of failures due to poor network connections.

6.2.8 In-App Messaging

A very important aspect of mobile applications development is the ability of engage app users with targeted messages that encourage them to complete vital in-app actions like buying an item, subscribing to a content and so on. This service is provided by Firebase Cloud In-App Messaging Service. This service also integrate well with Analytics and Predictions to give a very advanced targeting capabilities.
6.2.9 Others

Other services provided by Firebase worth mentioning are Firebase Crashlytics; which is a crash reporting cloud service of Firebase. Firebase Performance: which provides an application performance information with user’s experiences. Firebase App Indexing: which is responsible for getting an app into Google search engine. Firebase Dynamic Links: which are responsible in provision of smart links that dynamically change behavior to provide best user experience across platforms.
7  My Project - NewCityShop

NewCityShop is an online shopping cart which is designed to bring a very efficient and customer satisfactory shopping experience to users. I designed the platform using JavaScript Framework Angular version 4 for the Front-end development and I use the Google’s Firebase Cloud Platform for the back-end. My future plan is to create a fully cloud based solution that business owners who are desirous of having their own online store can advantage of. The nature of the development is that in just a few hours a fully functional online shopping cart can be made available to a prospective customer and the customer can then sign up as the administrator and can load his own products and services on to the platform.

The website will have a good, clean UI which will attract prospective buyers to spend time on the platform and it will also display some of the goods and services the business owner wants to sell on the home page, which will help in getting customers to quickly identify available goods and services they are interested in buying. There will also be a products and services category menu options very visible on the home page to help users identify the series of products they can buy on the website. The color of the UI will be very friendly so that the users are not distracted or irked. The site will be fully responsive so that all used devices irrespective of the screen size can have same great experience on the platform. Since Angular is mainly for creating web applications users from all OS and platforms can use the NewCityShop from their web browser. The application will also be a Single Page Application which means user can find all information on the site without navigating to another page in a single session.

At the top of the homepage is the menu displayed with links, first is the customized logo of the business, which has a link to the homepage of the website, next is the shopping cart menu which also have a badge designed to show the number of items in the shopping cart, the cart has a link which display the shopping cart view with the details of all item in the cart and their total value. Next is the About Us link which display the any information the business owner wish to share with the prospective customers about the business. There is also the Contact Us menu with a link which show any contact information the business owner have to share with the prospective clients. Next menu item is the Sign Up menu with the link that display the signup page view, which allow the user
to sign up into the platform, the signup page collects very minimal information from the user making it very fast for the user to get started with shopping. The last menu is the signin/login menu item, this menu item has a link which opens the login view. The login view take advantages of several Firebase Authentication options to include the Firebase ‘loginEmail’ function which uses the user email and password which the user used in Signup for login, It also use the social media login option available in Firebase Authentication using two options in LoginGoogle and LoginFacebook functionality. Also there is a link to create a new account in case the user want to create a new user account. The Authorization is very secure and it is impossible to complete a transaction without authorization on the platform. This is based on available best practices users should be able browse through a shopping website and even select items they are interested in buying into the cart before they are logged-in. Login should for enforced at the point of payment for selected products. Also there is a footer which has the copyright details, terms and conditions and the privacy policy of the website.

Figure 7: Home Page of Project
7.1 Project Fundamentals.

The NewCityShop is developed using the latest techniques and trends in web development process, which is the Serverless application development using a framework along with many other libraries like bootstrap for Angular while Google Firebase cloud platform products are used for the backend process. Firebase realtime database is used for the database and Firebase Hosting is the deployment platform.

7.2 Development Process

The development process of the platform was a bit longer than it was initially anticipated. I started my development by installing the Node Version Manager which by far is the best way to work with the Node Package Manager the platform for running all Node related applications. NVM makes it possible to have multiple versions of NPM on a machine thus multiple instance of Node, which often have different out of the box functionalities. By this architecture I am able select a default version of NPM and Node to run as my development server. For this project I use node version 4.9.1 and I first created angular CLI by using the code: npm install –g @angular/cli. Then I created a boilerplate package for angular inside the project folder using the code in CLI: ng new newcityshop. Then navigating inside the project folder, I start the npm server with the command: ng serve. Issuing this command start the npm server and automatically open up the homepage in the web browser, I add favicons using my customized logo and custom fonts. I then added bootstrap library to my project in order to ensure that I have my desired look and feel on the website by issuing the command: npm i bootstrap –save. I also added the angular moment library which is very useful in format time and date related values using the command: npm i angular2-moment. I also added the font-awesome and primeng library which is very useful in generating some aesthetic signs and signals for my web interface using the same format of commands: npm i font-awesome –save and npm i priming –save. All these commands install the mentioned library and add them and their version details as dependency to the ‘package.json’ file.
After all of the above necessary environmental set up, I proceed with the integration of my application with the Firebase Cloud service. First I open a Gmail account, since it is necessary in accessing all google services, then logged-in into firebase console from my browser and create a new project which I named “NewCityShopping”, on creation of the project, firebase create three different integration scripts one each for iOS, Android and web application.

I copy the integration code for web application and paste in the environment.ts file which is in the environment folder inside the main project folder like below:

```javascript
production: false,
firebase: { 
  // Initialize Firebase
  apiKey: "_",
  authDomain: "",
  databaseURL: "",
  projectId: "",
  storageBucket: "",
  messagingSenderId: ""
}
```

Listing 4. Firebase Web Application Configuration Script

After the above, I set up my angular application to use firebase by adding the needed libraries issuing the following code:
i.)  `npm install firebase angularfire2 -save.`

The above command installs firebase library and angularfire2 library which is a special library for working with angular2+ applications. This command also updates the dependency information of 'package.json' file with these two libraries. Then in the app.module.ts file, I use the angularFireModule, angularFireDatabaseModule, angularFireAuthModule to import the environment firebase object that was declared. I later re-factor all these code into the shared modules file as I continue to create a more modular and extensible application.

I created four major modules for the application to improve modularity and make it easy to manage and extend the application as the need arise, I break down each functionality in the application into components and some components are also passed to other components like dependency injection thus increasing the robustness of the application. The four major modules are: admin, core, shared, and shopping.

![Admin's Product-Form Component](image)

Figure 9: Admin's Product-Form Component

The admin module encompass all major administrative functionalities each broken down into a separate component. I create the admin-products component which is a listing of...
all available products in the platform, each with link to edit the product, there is also a link to create a new product. Then I developed a product-form component which is used for product preview, creation and update. There is an admin-category which is a listing of all available products category on the platform, there is a link to add a new category and edit each of the already existing categories on the admin-category component view, then there is the category-form component which is the form that is actually used to create, update and preview each category. And the admin-order component which is a listing of all order placed on the platform by customers with navigational functionality through the list. Below is a sample component code.

```typescript
import { CategoryService } from '../../../../shared/helpers/category.service';
import { Component, OnInit, Input } from '@angular/core';

@Component({
  selector: 'product-filter',
  templateUrl: './product-filter.component.html',
  styleUrls: ['./product-filter.component.css']
})
export class ProductFilterComponent implements OnInit {

  categories$;
  @Input('category') category;

  constructor(private categoryService: CategoryService) {
  }

  ngOnInit() {
    this.categories$ = this.categoryService.getAll();
  }
}

Listing 5. Product-Filter Component
```
I also have the core modules which contains all the core functionalities and features of the application, the login component, the ‘Sign Up’ component, the navbar component which contains the navigation bar, the ‘About Us’ component, the footer component, mysetting component which allow user update their profile information, address and password. And the product-details components which is the main detailed view of every single product on the platform.

Figure 10: Product details Component of the Shared Module

The shared module contains a whole lot of views that are needed and used in other parts of the application. The product-card component is useful in displaying a product with some minimalistic information in form of a bootstrap card design, the product-quantity component is also very vital in displaying the realtime information about the quantity of products in the shopping cart of the customers making use of the shoppingCart service. Another component in this module is the gallery component which is used in displaying the full product information page, the products image files are displayed using CSS modal class in a gallery form. Then there is the notification component which is used in
displaying success or error response of various user events as they happen on the platform.

Figure 11: Shopping Cart Component

The shopping module has a number of components ranging from my-orders component which display a list of all orders the user whose session is active has made since they have started using the platform till date, the order are arranged according to the date it was completed in decreasing order, then there is the order-details which displays the full information in a single completed order. I also create the check-out component which I passed a ‘shoppingcart’ observable object to and it show the details of the current ‘shoppingcart’ order. Then I created the products components which I used to display several products-cards components created earlier in the shared components at once, inside the products components, I added another components called products-filter which I used to display products based on the user selected products category. I created the shipping-form component which a user must use to fill-in their information to complete the details of their transaction. I also created the shopping-cart component which display the details of the current session shopping details, this shopping cart is a special component which can be passed to other components and consumed by them. I also have the shopping-cart-summary which is the component used in the shopping module to display the details.
of the shopping-cart component. The last component I created in this module is the order-success component which I used to notify the customer about the success of their placed order and its details.

Figure 12: Order Details Component

Several database models were defined and used in application implementation ranging from product, product category, application-user, order, shopping-cart item, and shopping-cart.

```typescript
export class Product {
  $key: string;
  title: string;
  price: number;
  category: string;
  description: string;
  file: File;
  filename: Array<string>;
  imageUrl: Array<string>;
}```
Listing 6. The Product Model Class.

Also several helper services which are responsible with fetching and saving data are used to keep my components lean and concern primarily with data presentation. From alert service, user-service, auth-service, auth-guard service which is used to implement Authorization on the platform among several others are used.

```typescript
import { AppUser } from 'shared/models/app-user';
import { AngularFireDatabase, FirebaseObjectObservable } from 'angularfire2/database';
import { Injectable } from '@angular/core';
import * as firebase from 'firebase';

@Injectable()
export class UserService {
  constructor(private db: AngularFireDatabase) { }

  save(user: firebase.User) {
    this.db.object('/users/' + user.uid).update({
      uid: user.uid,
      name: user.displayName,
      email: user.email,
      photoURL: user.photoURL
    });
  }

  updateUserInfo(user: firebase.User, name?, phoneNumber?, address?) {
    this.db.object('/users/' + user.uid).update({
      //name: objectValue.firstname + objectValue.lastname,
      name: name,
      phoneNumber: phoneNumber,
      address: address
    });
  }

  get(uid: string): FirebaseObjectObservable<AppUser> {
    return this.db.object('/users/' + uid);
  }
}
```
Listing 7. The UserService Class.

7.3 Future Implementation

It is my hope that the application can be improved upon and several more functionalities can be added or improved upon in the future, from improving cookies which can be very useful to provide customized shopping experience for each user based on their usage of the web application. Also more content are desperately needed on the website in order to create enough traction and pull users to the platform. I already push the source code to GitHub with this link: https://github.com/hiroyalty/cityshop. So achieving these goals will be quite easy.
8 Conclusion

Web and general application development have gone through lots of changes in the last few years, and JavaScript has surely been at the forefront of these changes. Through the release of frameworks like Angular, React and Vue, we can now rapidly develop applications that is easy to maintain, very secure and scalable. Combine this with cloud computing we can now even focus mainly on mainly the core objectives of our application. There are myriads of shopping cart solutions out there, however most of them are cumbersome, slow and lack the data intelligence that type of architecture offers. My hope is that the reader of this thesis will be encouraged to switch to implementing this form of development model for future projects.
References


Figure References
