Olayinka Gold

CUSTOM CLOTHING WEB APPLICATION

Technology and Communication
2018

ACKNOWLEDGEMENTS

My utmost gratitude to God who has sustained me thus far to accomplish this and my deep appreciation to my supervisor Timo Kankaanpää who has taken time to give every support and attention needed to make this thesis a success. I am saying thank you to all the staff of University of Applied Sciences, Vaasa, who has contributed to my knowledge in my period of study and above all my brother Ayoola Gold who has been an inspiration to me throughout my period of study

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Degree Program in Information Technology

ABSTRACT

Author	Olayinka Gold
Title	Custom Clothing Web Application
Year	2018
Language	English
Pages	58
Name of Supervisor	Timo Kankaanpää

The thesis work was done for a small clothing company in Nigeria, Goldy Topman.

This thesis studied the online custom clothing solution. The idea of the topic is getting a person to fit in the most desirable outfit which has been a challenge trending for a while now. Often times, cloths are made in mass production and this gives consumers little or no chance to make and tailor their own clothing. Most outfits comes in standard sizes and if you do not fit in it, you have to accept whatever is available to you. Custom clothing application has not only helped to combat that problem but also avail other functionalities you can explore such as choosing your design from the options and giving your cloth a description. Also the restriction on designing your clothings based on what the manufaturer specifies has been eliminated.

Online custom clothing design applications have been in use over a period of time now on several platforms with different implementation and functionalities. Majority have provided similar functions and some of them have in a way streamlined user experience in making their own designs.

The application was developed with the aim of breaking the norm and bringing diversity and new functionalities for users in a simple way. Creating a different user experience, improving the company sales and flexibility, gives users remote access to designs and facilitate the ordering process.

Keywords Angular, Web Application, TypeScript, Single-Page Application

CONTENTS

LIST OF FIGURES, TABLES AND CODE SNIPPETS

LIST OF ABBREVIATIONS

API	Application Programming Interface
CLI	Command Line Interface
DOM	Document Object Model
ECMA	European Computer Manufacturers Association
GUI	Graphical User Interface
HHTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
HTML	Hyper Text Markup Language
I/O	Input/output
JS	JavaScript
JSON	JavaScript Object Notation
NPM	Node Package Manager
REST	Representation State Transfer
SOAP	Simple Object Access Protocol
SPA	Single-Page Application
TS	TypeScript
UI	User Interface
UML	Unified Modelling Language
URL	Uniform Resource Locator
WSDN	Web Services Developer Network
XML	Xtensible Markup Language

1 INTRODUCTION

1.1 Single Page Application (SPA)

Single-Page Application is an application with one HTML file and several other JavaScript files obtained from the sever and everything is rendered in the browser giving the user a very reactive experience. This makes user experience very smooth and dynamic while using the application. It renders also for mobile application, very fast and responsive. With a SPA there is no need to reload the entire page, most resources are retrieved with a single page load and redraw parts of the page when needed without requiring a full server roundtrip.

1.2 The customer of the solution

"Goldy Topman is a specialist in making t-shirts, hoods, polos, varsity jackets, sweat shirts, crop tops, sport wears, joggers, corporate wears, native attires and many more"

Goldy Topman is a small-scale company growing fast to meet clients with average or minimal demands, the company wants to interact with users on the very basic level and tailor their designs just as they want it. They offer one on one services to get user specification and create designs for clients that have minimal knowledge of what they want.

Currently, the company is operating in one city and just in one location, with a small staff who work to meet the demands. The designs are mostly done by the CDO (Chief Design Officer), who takes the customer requirement and create a design for them. In most cases, they refer to their catalogue but in the absence of an application such as this, they cannot explore their creativity and make their own design.

One of the best ways to implement a solution to what is offered by Goldy Topman is using a SPA which offers very good functionalities over the web and built responsive to suit any type of device regardless of its size. The SPA also allows rendering on almost all the latest browsers with every feature embedded displaying.

1.3 Objective and topic of choice

The objective of this document is the demonstration of using a SPA to achieve a very fast and responsive web application with diverse functionalities. It aims to solve problems associated with people buying the available against the preferable. The application avails users the opportunity to create their own design either with an existing template from the collections or a new one, it also allows users to store their designs and reedit their designs after it has been created.

In addition, users can use the saved data to make reorders with options such as size selection, print location, print type, color, font style and many more as described in the description box.

The current topic is a preferred study subject because it incorporates the full stack web development with modern technologies that allows users interact with the web page and get desired result which is an improvement to the traditional HTML pages. Before now, pages are often static with minimal or no user interaction involved. This study includes both the frontend (what the user sees and interacts with) and the backend (the enabler of the frontend experience).

1.4 Problem definition

Today, orders are received with the staff visiting every client with a catalogue of designs made and suggested designs for clients. Getting the customer requirement in most cases seems to be a challenge. With the implementation of this custom web application, clients will have the chance to visit the web page, explore the available collections and customize and tailor their designs without having a physical meeting with the company.

In addition, the application will facilitate orders across the country as designs and orders can be done online and customer specification can be adequately filled and interpreted by the company to proceed to design. It will also facilitate more orders and improve the company's turn over. The company's awareness will be enhanced, as many can see what they do remotely without visiting the company.

1.5 Implementation plan

The structure of the thesis as well as its design is shown in the figure 1 below. The work starts with a problem to solve, which was carefully thought of having seen the challenges faced to get user satisfaction. A concise study was carried out after getting the requirement specification. The problem involved building an application from scratch, as there is no existing template to work on or improve. This has given limitless opportunities to explore every available technology and use the best ones suitable. Selecting the suitable technology not only considers the frontend of the application but also the backend, thereby creating a sync and seamless work flow between both sides of the application.

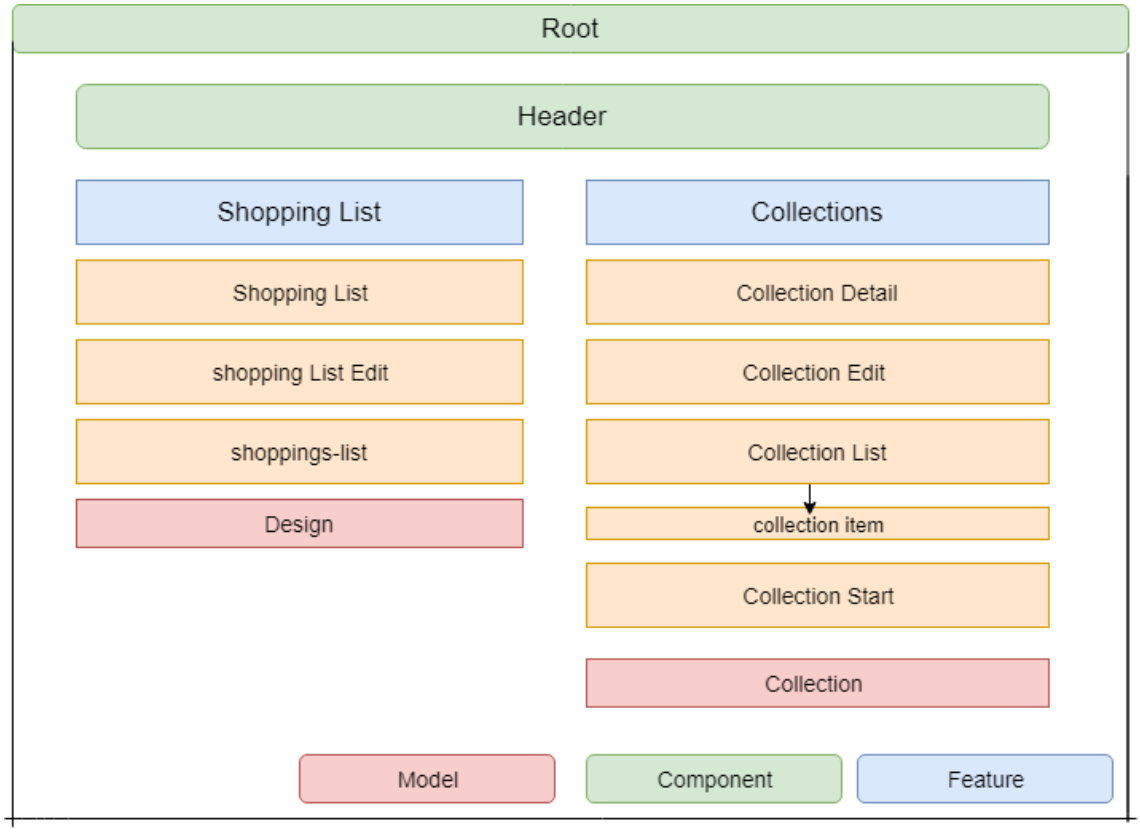

Figure 1: Implementation structure of the application

2 THEORY AND BACKGROUND PROCESS

2.1 Feasibility study

Further in the study of this thesis, the concept of the application has been in use by different manufacturers on different platforms with different approaches and designs but with similar functionalities.

Below is comparison among a few online web applications that offers similar services. Also how their web application has been designed and their key features is discussed.

2.1.1 Customin.com

This website avails the opportunities for diverse functionalities and dynamic but has limitation in several other phases that has caused restrictions to design for the user/customer who wants to create a design. This application has a section where cloth type (such as Polo) can be selected, you can also select features such as Material type, collar type, texture, pocket or not, style type (ladies, Unisex or Men), colour, sleeve length (short or long), this site makes use of readymade brands. Most of the features are proportional to the price.

Design section allows you to add text, art, colour, names, note, print on sleeve (sentence up, sentence down)

Limitations of the application include:

- You can only have a maximum of two colours depending on the product type selected
- You can't create your own colour mix and have something unique
- By using readymade brands, the cloths cannot be tailored to your taste, but the visitors are compelled to find their fittings among the standards offered.

For designs made on the cloth, price is not generated automatically but you should call to negotiate and place an order. There are chat and call platforms for getting further details. /1/

2.1.2 Rush Oder Tees

This website has models for almost all their designs. The site visitor selects a desired product to start the design. There are features like; add text, colour, add clipart, upload art, add name and numbers. The application also allows features like rotation, which can allow design on both sides. It is possible to add size; this feature allows you to include the quantity and with the colour selected the price is generated. There are various styles of cloths for various categories.

The application also has browser cookies that automatically save your previous work and allow site users to start from where they left.

Limitations of the application include:

- Only single colour can be added at a time, there is no opportunity for mixing colours on the readymade designs
- To personalize, the font sizes are just big and small, several text colours are available, just two font styles, and this does not give users the opportunity to do more.
- There is no place to give extra description on what you want to see nor add anything that is not in the listed features or not on the cloth design. /2 /

2.1.3 Spread shirt

This company web application offers diverse designs for different categories and products. They offer features such as colour change, adding text, art upload, size guide for different sizes including their dimensions in inches. They also offer predefined arts in different categories such as animals, emojis, funny, love, shapes, sports and many more. Some are free and some attract some cost which will be added up to the total price of the shirt when the design is completed.

The application offers special features such as selling your designs on the web page depending on how much there is demand by other visitors.

Limitations of the application include.

- They offer readymade cloth, where you cannot adjust the size or upload your desired cloth dimensions.
- The colour offered is just one and users cannot mix colours as they wish
- They make designs on readymade cloths and there is no opportunity to choose your preferred designs. /3/

2.1.4 Vistaprint

They have products that range from different categories, not just clothing but also almost anything that involves printing. For the clothing, they offer features for various categories one of which is industry, which has sub categories such as food & beverage, retail & sales etc. They also have categories like personal and family which offer features like baby, birthday and much

more. In addition, they offer features like add text, image, colour choice. The “add text” feature also has the feature to change colour. The more features are selected the higher the price.

Limitations of the application include:

- The text does not offer the option of choosing font style or font size and the cloth colours cannot be mixed.
- The sizes are fixed and preferred dimensions cannot be used.
- The cloth description cannot be given as to make your own preferred designs aside what is offered on their web page /4 /

Above there is the overview of some web applications that are doing something related to the topic of this thesis. However, what is to be done is in some ways similar but with a different approach. With the design implemented in this application, user experience and interaction with the application will differ from those listed above.

2.2 Web Application

A web application is any computer program designed to perform specific functions using the web browser as its client. It encompasses the “client” and the “server” environments. The client-server environment is one in which multiple computers share information such as entering information into a database. The “client” is the application that the user interacts with to enter and share information. / 5/

The backend is a mix of a server, databases, APIs, and operating systems that power an app’s frontend(client), this could involve the use of cloud-based servers and data warehouses, containerization with a service like Docker, Backend-as-a-Service(BaaS) providers. /6 /

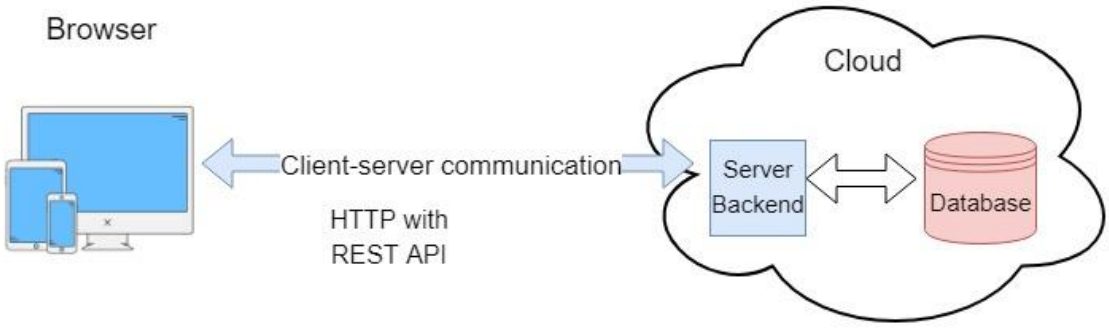

Figure 2: Client and Server

This approach is platform independent as it runs on the web using a browser. It does not consider the user operating system and the application can be accessed over the internet.

2.3 Programming Languages

A programming language is developers' tool which is selected depending on the task to be performed, project type to be executed and more. And in some cases, selection is made based on preference. Programming languages could be broken down into client-side-scripting and server-side-scripting. Examples of client-side languages are JavaScript, ActionScript, VBScript and more while for the server-side languages includes PHP, C#, C++ and more. While some could be used for both client-side and the server-side depending on the framework used.

2.4 Frameworks and Libraries

2.4.1 Frameworks

A framework provides a set of constructs for building your application. It is an implementation of the web application where your code fills in the details, it is in control of how it is to be used and it is called into your code when something app-specific is needed. Angular JS, and Ember are examples of frameworks. Frameworks decide what you can do with the application and afterwards accepts your code for use. Code is supplied into the framework and how it is to function is explained in the code. It basically determines how the application runs and then calls in your user code to brace and explore its functions. For this application, the UI framework is bootstrap 3 and the JS framework is Angular 5

2.4.2 Libraries

Libraries are sets of functions encapsulated in one or more JS files which are useful when writing web apps. In this case, the code controls the behaviour of the function to perform and it is called into the library when it is needed, for example, jQuery, which is in use all over application logic to manipulate the structure of the page or change the style and layout of items in the application. When using a library, you can determine what you can do with your code in the library.

2.5 Document Object Model(DOM)

The Document Object Model (DOM) is a programming interface for HTML and XML documents. It represents the page so that programs can change the document structure, style, and content. The DOM represents the document as nodes and objects. That way, programming languages can connect to the page.

The DOM is not a programming language, but without it, the JavaScript language wouldn't have any model or notion of web pages, HTML documents, XML documents, and their component parts (e.g. elements). Every element in a document—the document, the head, tables

within the document, table headers, text within the table cells—is part of the document object model for that document, so they can all be accessed and manipulated using the DOM and a scripting language like JavaScript. /7 /

In a simple equation, it can be explained thus;

API (HTML or XML page) = DOM + JS (scripting language).

2.6 Database

On a web page or in the context of a website, database is often regarded as the "brains" that make them dynamic. When a search is done from an online store, request is made or post is made from the user, the database is responsible for executing this task. Database accepts new and edited data when users interact with the website or application. Users can change the content of the database depending on the services users are opened to use. Users can upload files, post pictures, retrieve files and do many others with the help of a database working. Database consist of two main types; Structured Query Language(SQL) and No Structured Query Language(NoSQL) or relational database and non-relational database respectively. Both are used for different purposes depending on the requirement of the application design and implementation. Relational database like phone books that store phone numbers and addresses. Data stored in it are structured in tables with columns and rows. Examples of SQL databases are MySQL, Oracle, IMB DB2, Sybase, Microsoft Azure.

Non-relational database on the other hand, are document-oriented and they are distributed like folders. A file could store data which contains different sub files such as photos, texts, links and more. They offer easy access to developers. Examples include, MongoDB, Firebase.

2.7 Servers

Servers are high powered computers that provides shared resources which includes file storage, security and encryption that networks need to run on and act as the lifeblood of the network. Servers contain website files, database and server-side software. Modern server architecture is not restricted to physical machines as explained above but could also include the cloud. Server architecture comes in various forms which includes dedicated servers, virtualized servers, application servers and the cloud. Servers involves storing and communicating which is possible over standard called protocol.

Servers, just like computers has an operating system and server software which makes them different and they have special configurations that makes it possible for them to run special tasks. One of the common types of server is the web(HTTP) server.

The web servers are computers though which a website can be accessible by a user via HTTP and HTTPS requests over the internet. Web servers make it possible to access and communicate via Internet Protocol (IP) address with browsers.

3 USED TECHNOLOGIES

3.1 Angular

Angular is a JavaScript framework which allows creation of reactive Single-Page Application. It provides set of functionalities to greatly reduce the amount of code written and effort involved in making an application fully functional.

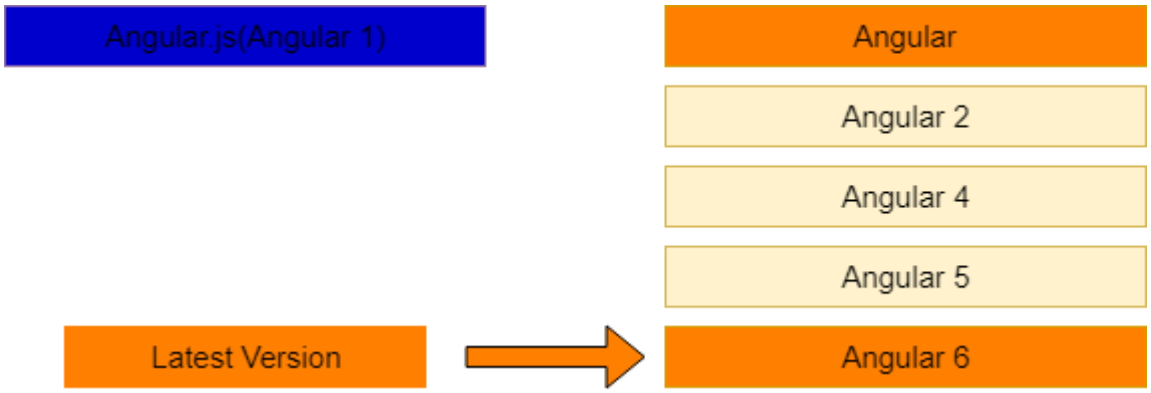

Figure 3: Angular Versions

SPA is an application with one HTML file and several other JavaScript/TypeScript files which is obtained from the server and everything is rendered in the browser giving the user a very reactive experience. It allows users to interact with the DOM and change contents of the web page dynamically in run time. Angular is a futuristic language that uses features like TS, which is a superset of JS and it needs to be compiled before it runs in the browser. This application has been built with the latest version of Angular which is Angular 5.

3.1.1 TypeScript

TypeScript uses existing JS code and incorporates popular JS libraries. It compiles clean JS code which runs on any browser, in Node.js or any JS engine that supports ECMAScript 3(or newer)

The TS offers more features than Vanilla JS such as Types, Classes, Interfaces and more which allows a few annotations to make a big difference to the static verifications of the code. It does not run in the browser but is compiled to JS. It is an addition to JS, not a complete replacement. /8 /

The application has been designed to give users very fast and responsive experience, downloading all pages all at once and the navigation between pages becomes very responsive without having to load each page from the server every time it is clicked.

3.2 Node Package Manager(npm)

Node Package Manager is a package manager for JavaScript with three distinct components such as:

- website
- the Command Line interface(CLI)
- registry

The website is used to discover packages, set up user profiles and manage other aspects of the npm, CLI, runs from a terminal and it is the tool developers use to interact with npm. The registry serves as public database of JavaScript software and meta-information surrounding it. With the npm, packages can be adapted to applications, download standalone tools that can be used right away, share code with other npm users, manage multiple versions of the code and code dependencies.

3.3 Node JS

NodeJS is a JS runtime built on Chrome's V8 JS engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world. /9 /

The Node.js is first installed before building the application. Node.js is the server with which JS frameworks which includes Angular.js runs on. With Node.js installed, project configuration was possible through the CLI, which is a tool set which makes creating, managing and building Angular applications in simple steps possible. Once the project is built from the CLI, ng server will build all the source code, as Angular TS code needs to be compiled to JS code before it displays in the browser. A server is needed as against running the "index.html" file directly because it loads huge resources dynamically.

3.4 Firebase

3.4.1 Firebase

Firebase is a real-time NoSQL cloud database (but also a cloud storage and messaging service and authentication service) that helps to build apps without building the backend. With it, saving and retrieving of JSON objects is possible, user authentication can be built, and data updates can be accessed in real-time across connected devices in milliseconds: data remains available if the app goes offline, providing a great user experience regardless of network connectivity. /10/

3.4.2 Reason for Firebase Over MongoDB

Firebase is a real-time engine with backward connectivity i.e. a cross-platform app can be built where clients subscribe to events on specific data and the server actively inform clients about

changes. Firebase now offers various features that could be a struggle to build on your own, which are;

For development

- Cloud Messaging: Deliver and receive messages across platforms reliable
- File Storage; Easy file storage (including iOS)
- Hosting: Deliver static files from Firebase's servers (included in free plan)

For growth

- Remote Config: Customize your app on the fly: suitable for A/B testing
- Dynamic Links: Send users to the right place inside in the app
- Notifications: Engage with users at the right moment.

Firebase is a Backend-as-a-Service(BaaS) containing identity management, real-time data views and a document database which runs in the cloud. Firebase is designed for real-time updates, it is easy to integrate with Angular.

MongoDB is a full-fledged database with a rich query language. It runs on your own machine, but there are also cloud providers for it. MongoDB is ideal for more robust and larger apps with the possibility of real-time integration but would require extra work to achieve. /11 /

3.5 Visual studio Code

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes (such as .NET and Unity). /12 /

Among many other text editors, VS code has been selected because it is very handy and the interface is easy to use. It has built-in support for Node.js development with JavaScript and TypeScript. It has an integrated terminal called the power shell which act as the Angular CLI where the Node.js server can run and update is seen in real time. It also supports other web technologies such as HTML, CSS, JSON. The interface colour makes code easier to read, error easy to detect and identification of different files easy.

3.6 GitHub

Git is a free and open source distributed version control system used for the versioning of this application. With this tool, it is possible to create an online repository, branch, make commits,

pull requests and merge requests. With GitHub, application code can be managed, commit can be done to every significant change and allows remote access over the internet. It makes possible accessing other helpful resources and committed files to the online repository helps to prevent losing files if local repository or system is damaged.

3.7 REST

Representational State Transfer is used for web based architectures for data communication. It uses web standards, exchange of data is done using either XML or JSON and it is simpler compared to for example SOAP, WSDN. It uses HTTP to make calls between machines. REST is a language-independent architectural style, all needed is for the programming language to have the ability to make web-based request using HTTP. RESTful API uses HTTP requests to do the following;

- GET: to retrieve a resource
- PUT: update a resource
- POST: to create a resource
- DELETE: to remove a resource

3.8 Hyper Text Markup Language(HTML) and Cascading Style Sheet(CSS)

The HTML defines the webpage's structure and its content. It is used to create web pages and make them functional. HTML consists of tags attributes with which defines its structure on the web page. The tags mark up the start of an HTML element and they enclosed in angle brackets, the attributes contain useful information which are enclosed in the opening and closing tags.

With CSS, appearance such as font size, colour, style and many more can be designed. It sets the formatting and the appearance of the web page.

4 ANALYSIS AND DESIGN

4.1 Requirement Specification

4.1.1 Preword

This application is made for users (buyers) to use either for the readymade designs offered for making their preference from scratch with the template offered in the application. Non-registered users can only visit the website and view collections, browse through the web page menus but not place order or customize their designs. Registered and signed in users will have the opportunity to choose their preferred designs and specify descriptions of what they want on the cloth and place orders.

4.1.2 General view

The application will be implemented as a web application where users can access it via any device of their choice using even the earlier versions of browsers such as Google chrome, Mozilla Firefox and Internet explorer over the Internet. The users will need to register before they can access the customize functionality of the application. The Table1 shows the functional characteristics of the application.

Reference	**Description**	**Priority**
F1	Allow registered users to access all web page functions	1
F2	Edit an existing collection to your specification	1
F3	Make new collections with predefined template	1
F4	Show image preview	2
F5	Edit / add your own cloth description.	2
F6	Save design to user profile	3
F7	Add to shopping list	3
F8	Edit/ add to shopping list	2

Table 1: Functional Characteristics of the Application

The Table 2 shows the other characteristics of the application and includes their priority levels.

Characteristic	Describe	Priority
Responsive web design	Application is responsive on all devices	1
Cross-browser compatibility	Works in latest version of Chrome, Firefox and Internet Explorer	2
User friendly	Application interface is simple and easy to navigate	2
Simultaneous users		
Responsive time	<0,3s between request and response	2
Increased productivity	Easy to get started	2

Table 2: Other Characteristic of the Application

4.2 Architectural design

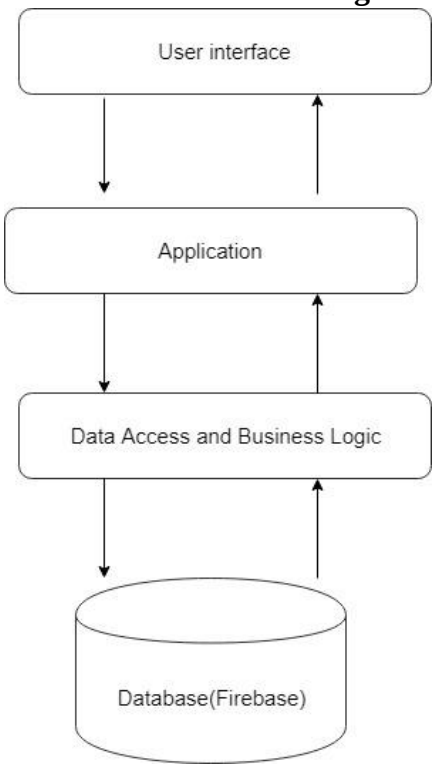

Figure 4: Architectural design of the application

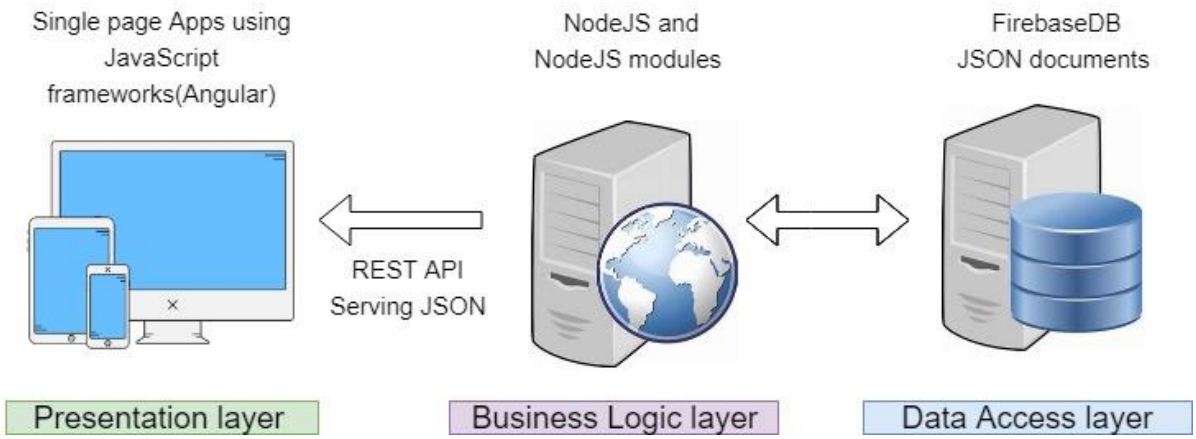

Figure 5: Full Stack Development from client to server

4.3 Unified Modeling Language (UML)

Unified Modeling language (UML) is a standardized modelling language enabling developers to specify, visualize, construct and document artefacts of a software system. Thus, UML makes these artefacts scalable, secure and robust in execution. UML is designed to enable users to develop an expressive, ready to use visual modelling language. In addition, it supports high level development concepts such as frameworks, patterns and collaborations. In a UML diagram there are different structure diagrams which are, class diagrams, component diagrams deployment diagrams, object diagrams and package diagrams /13 /

4.4 Angular Framework

Angular is a framework with the following features;

- Complete rewrite of the framework
- Component-based
- Mobile Support
- Server-side rendering
- Powerful templates
- Supports dynamic applications
- Components

Angular components have their own template (html code), style and business logic which allows us to split up the complex web page into reusable parts. The created components are added to the application component which is the root component and not the index.html.

4.4. 1 Components

It is a special kind of a directive that uses a simpler configuration and is suitable for a component-based application structure. This makes it easier to write an app in a way that is like using Web Components or using the new Angular's style of application architecture.

Advantages of Components:

- simpler configuration than plain directives
- promotes sane defaults and best practices
- optimized for component-based architecture
- writing component directives will make it easier to upgrade to Angular

Components only control their own view and data and they have a well-defined public API-inputs and outputs /14 /

4.4.2 Decorators

Decorators are TS features that allows enhancing classes, they are design patterns that are used to separate modification without modifying the original source code. Decorators in Angular JS are functions that allow a service, directive or filter to be modified prior to its usage.

4.4.3 Directives

Directives are markers on a DOM element (such as an attribute, element name, comment or CSS class) that tells the HTML compiler of Angular to attach a specified behaviour to that DOM (such as event listeners). There are in built directives in Angular JS like ngBind, ngModel and ngClass

4.4. 4 Services

Service help keep a component as lean as possible, providing features such as:

- Fetching data from the server, user input validation.
- Mediating between the view and domain logic
- They help in dependency injection

Service name and service function factory functions are registered into an Angular JS module.

The **service factory function** generates the single object or function that represents the service to the rest of the application. The object or function returned by the service is injected into any component (controller, service, filter or directive) that specifies a dependency on the service.

4.4.5 Shared

The shared folder contains TS files used in the services and Components file. The files contained in it define the structure of how the component it is being injected in behaves, providing extra functions for the existing component. It defines the attributes and their types.

4.8.6 Store

It is a JS object that holds the application state, it acts like the database in this, from where data is fetched.

4.5 Use case Diagram

A use case diagram consists of actors or users, uses cases and how they relate. This diagram helps model the system of the application and captures the functionality of the system. It gathers the requirement of the system, gets an exterior view of the system, identifies the external and internal factors that make up of the system and show interactions with the actors or users

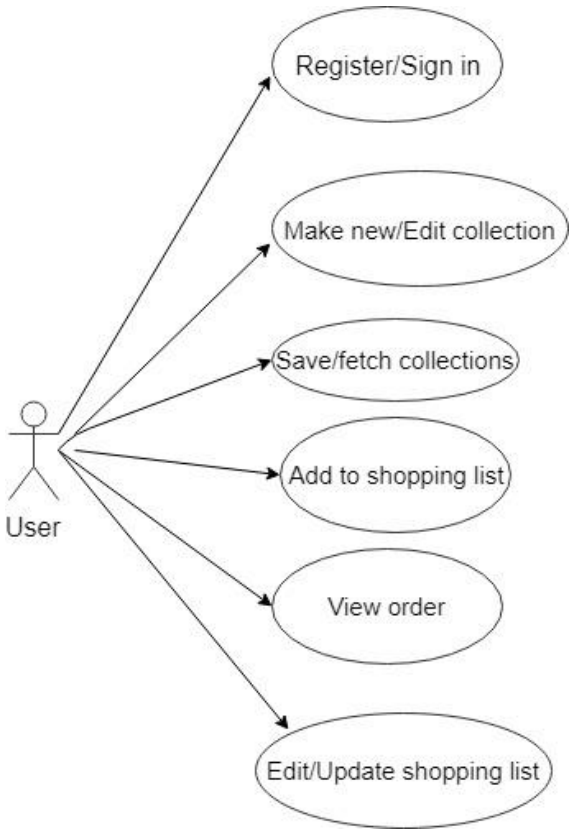

Figure 6: Use case diagram

4.6 Class Diagram

Class diagram represents the system class, the attributes and the relationships among the classes. They represent the static view of the application, they describe the attributes and operations of a class in their systematic flow as shown in Figure 7.

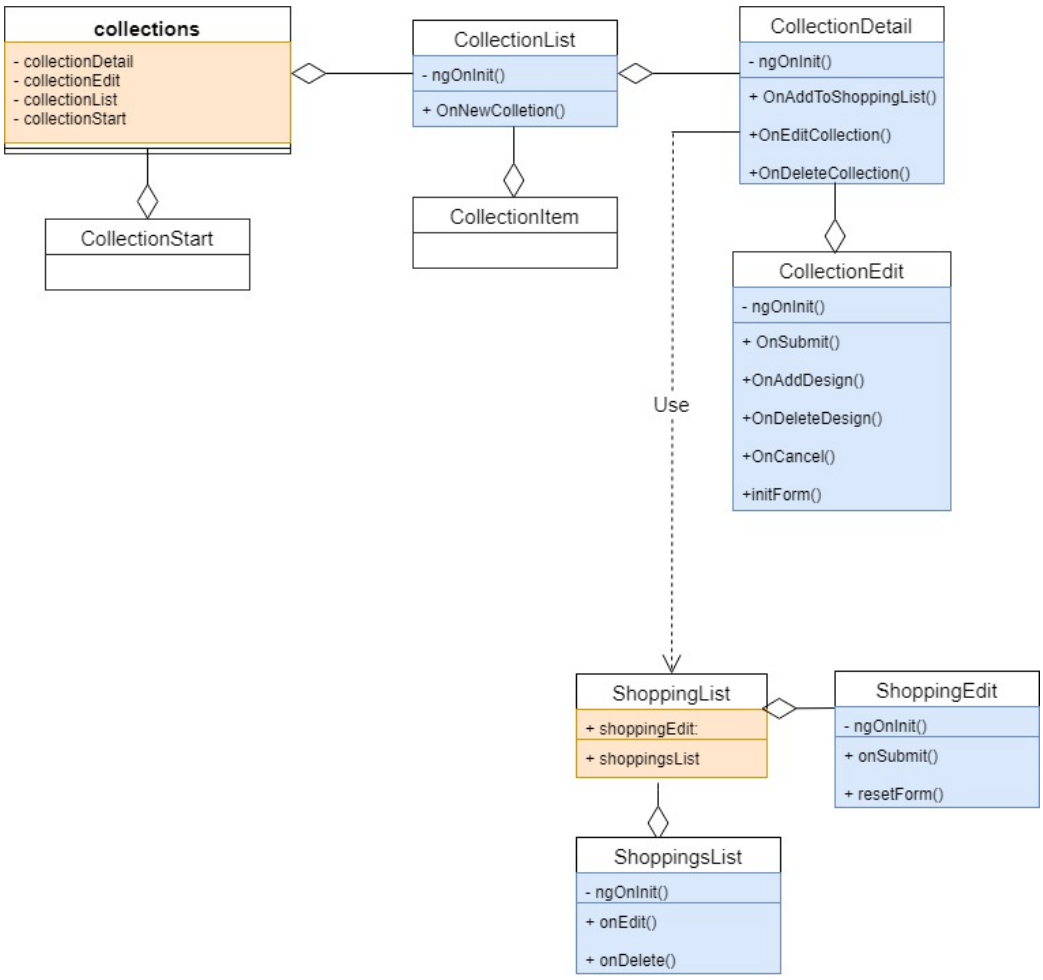

Figure 7: Class diagram

4.7 Sequence Diagram

Sequence diagram show how objects interact in a single use case. They illustrate how functions is carried out and in the order in which the interactions occur.

4.7.1 User Registration

User registration sequence diagram shows the various steps involved in user registration. First, the use starts the application via a web browser, navigates to the registration page, registers using email and password, which must follow the form validation criteria before registration is

successful. The onSignup method is called and new authentication is dispatched with the user email and password. This creates a post to the database and user data is saved to user profile. If details are wrong or have been used, the application returns display error to the user and if successful it redirects the user to the home page of the application. The user registration sequence diagram is shown in Figure 8 below.

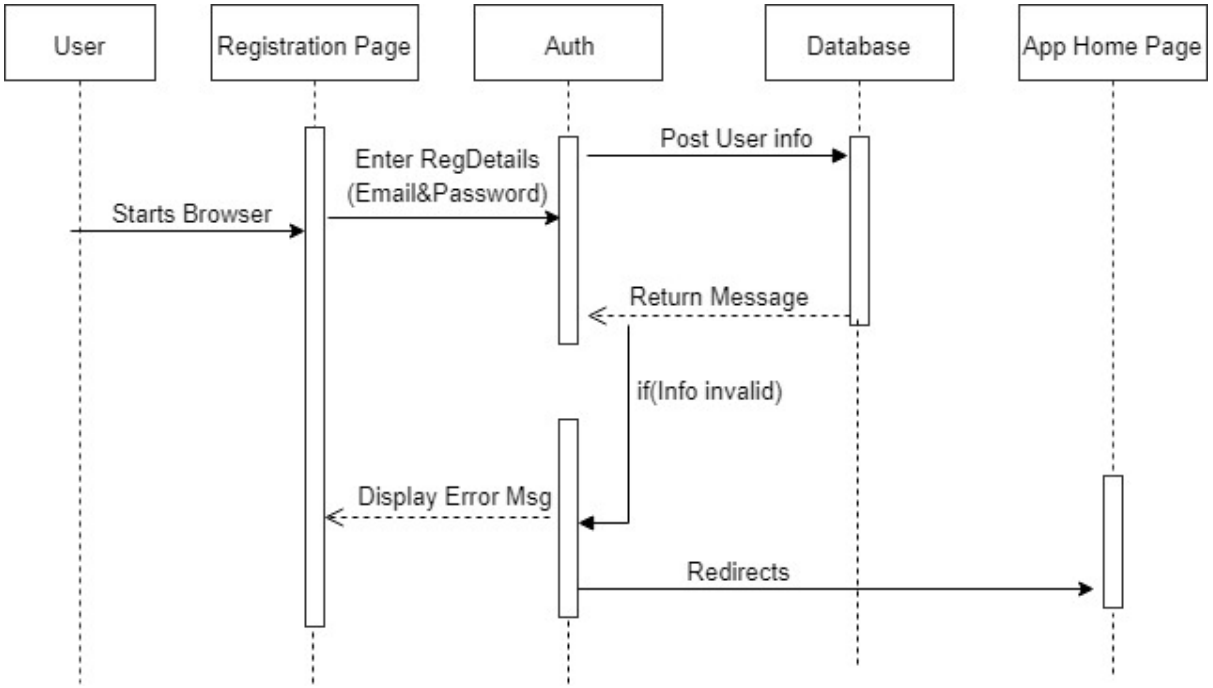

Figure 8: User Registration

4.7.2 User Login

User login sequence diagram shows the steps involved on how a user login takes place. The registered user navigates to the login page, enters authentication details, the details are authenticated and a unique login token is generated which identifies the user over the browser at that time. The token is destroyed each time the user logs out and a new one is generated at login. If not authenticated, an error message is generated, otherwise the program redirects the user to the home page of the app, giving the user access to full functionality of the collection page and the shopping list page. The user login sequence diagram is shown in Figure 9.

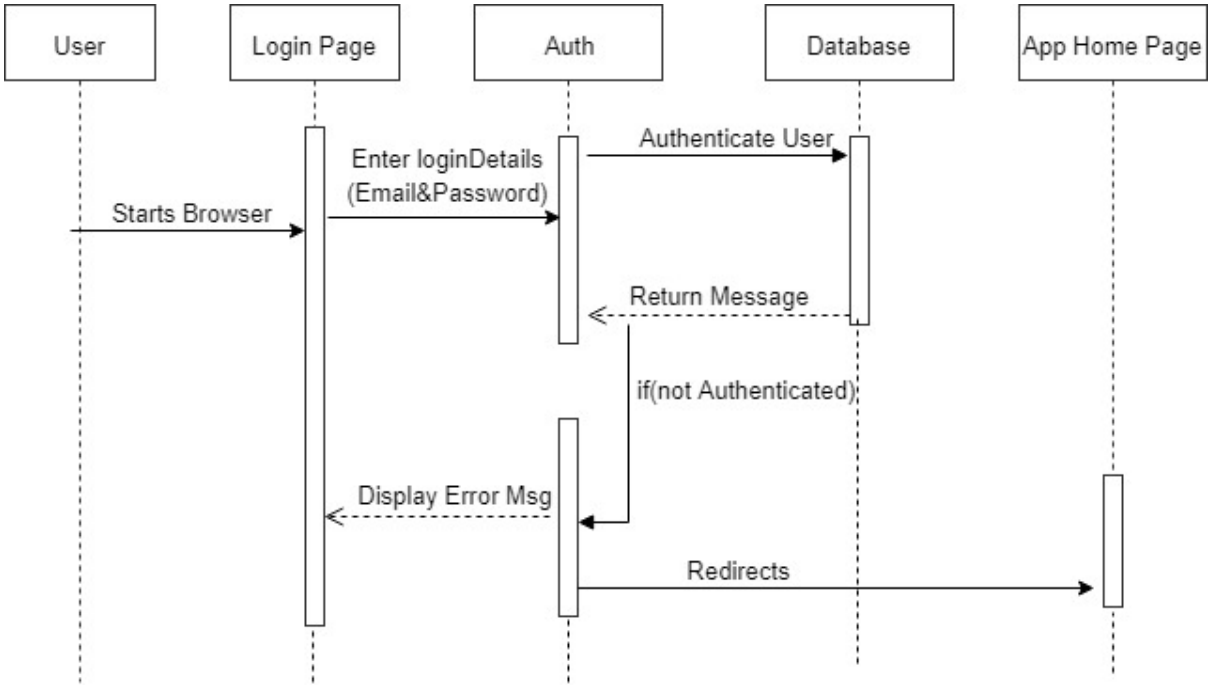

Figure 9: User login

4.7.3 Collection

Figure 10 shows how the collections page works and how designs are made by registered users. First, the user navigates to the collection page, views collections already available, clicks on each collection to get full details with the option to make a new collection or edit existing collection. The user needs to complete every collection detail before saving can be done. The information is saved to user profile and a post is created, if successful, the newly or edited saved collection is added to the collections. The user can login at another time, fetch collections and edit the previously saved collection.

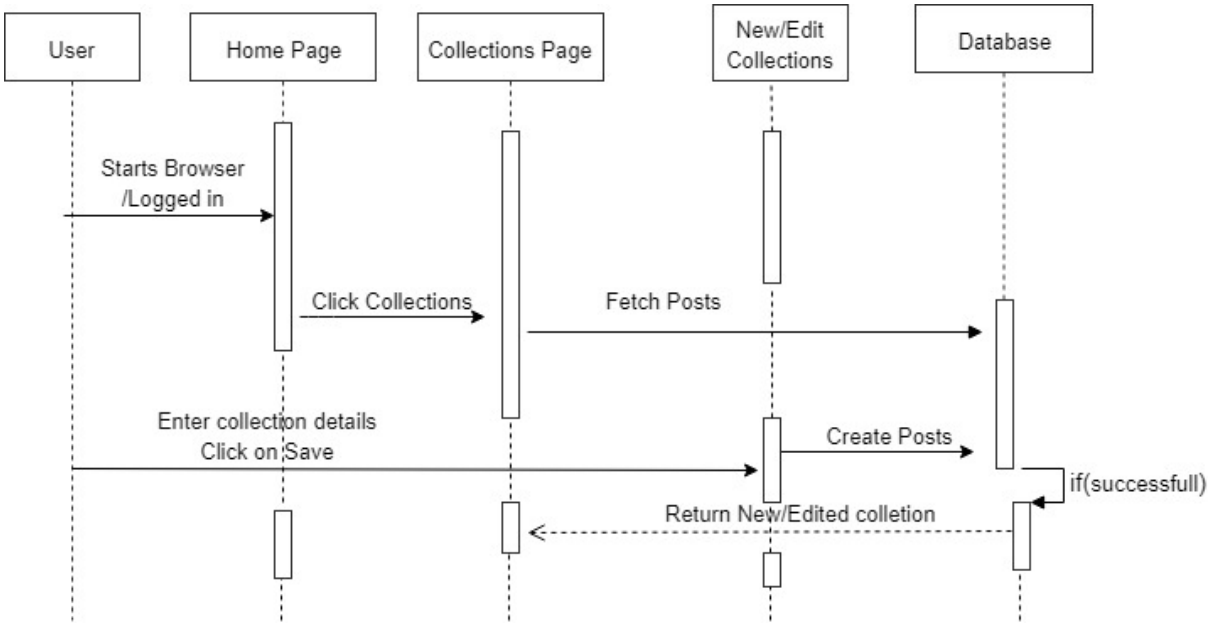

Figure 10: Collection Sequence Diagram

4.7.4 Shopping List

In this sequence, the steps to making the shopping list is shown in the Figure 11. The registered and logged in user can select a collection and add items to shopping list, and on each collection, specifications such as size, amount and collection name must be filled before it can be sent to the shopping list, the users can also edit the shopping list before submitting the order, and after submitting, a user can see a list of orders placed on the right-hand side of the page, the list can be edited or deleted depending on the action the user wants to execute.

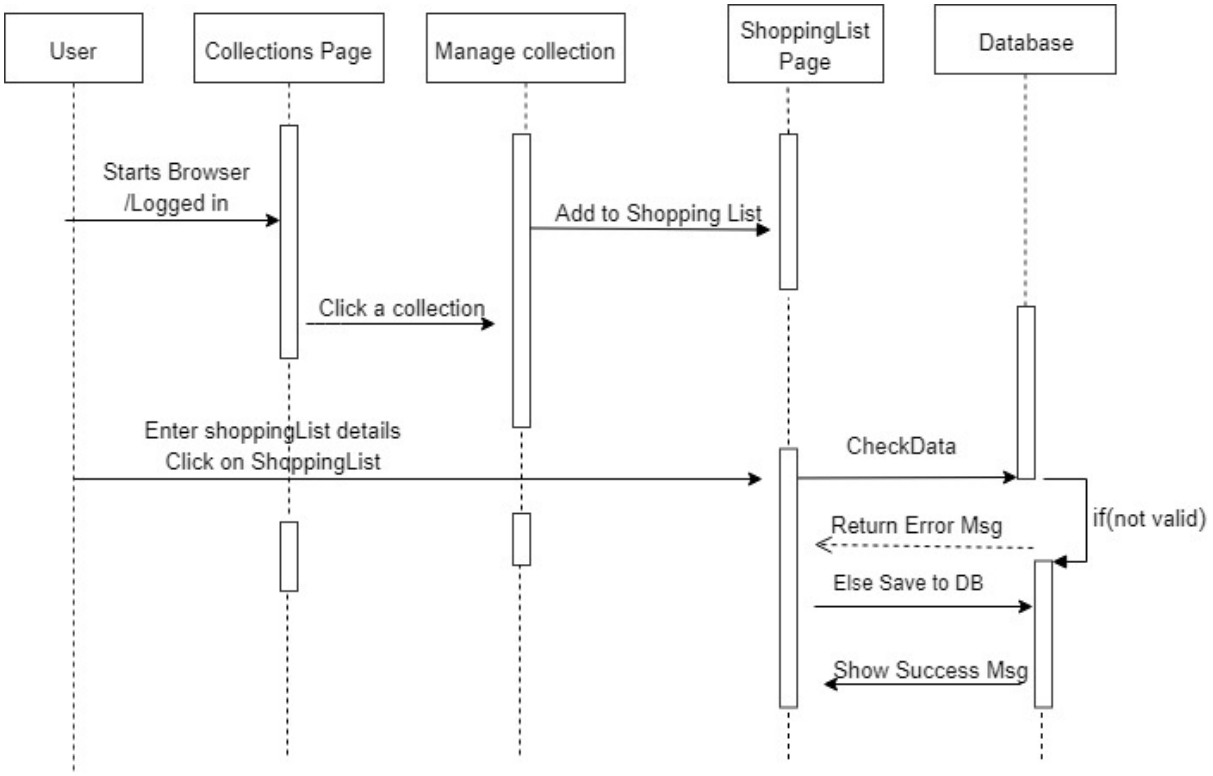

Figure 11: Shopping List Sequence Diagram

4.8 Package diagram

Package diagram represents splitting the system(application) into logical groupings and dependency among the groupings Elements are organized into groups to provide better structure for system model as shown in Figure 12.

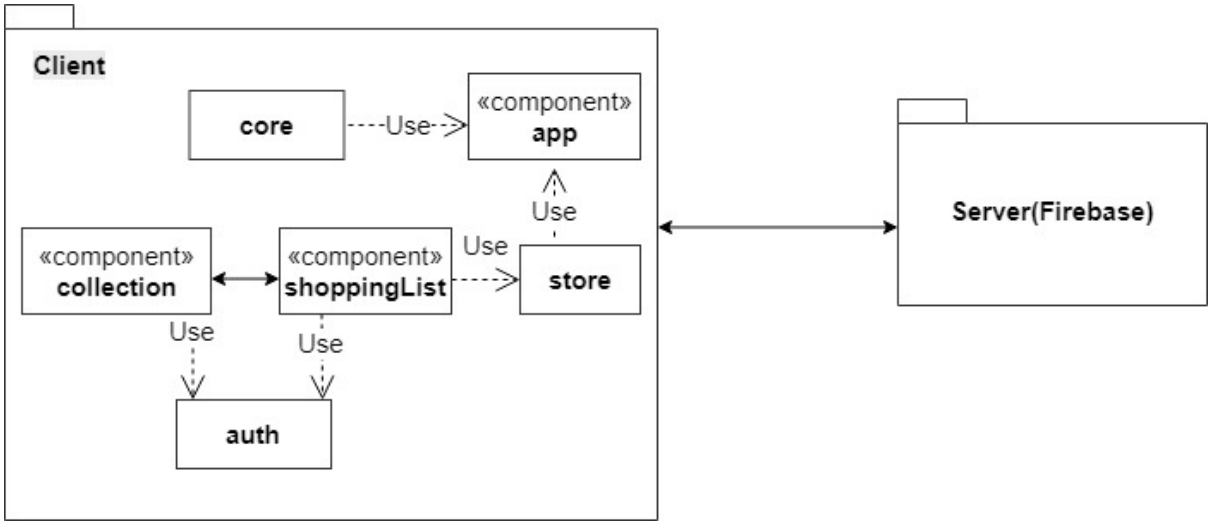

Figure 12: Package Diagram

4.9 Model-View-Controller(MVC) Framework

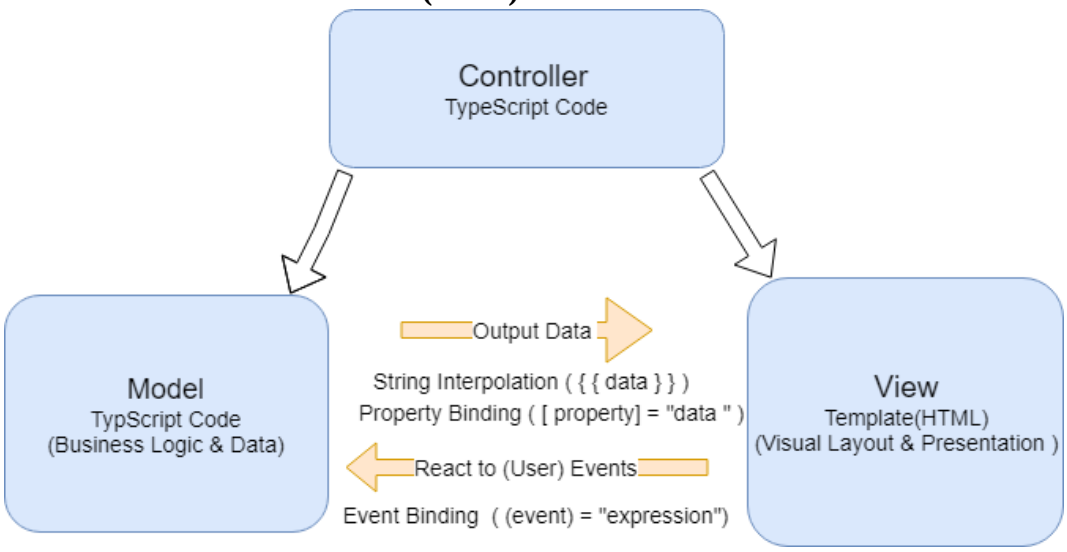

Figure 13: MVC Architecture of Angular

In the MVC architecture pattern, different aspects of the application are broken into components to separate their functions and create independence. This framework allows isolation of domain logic from user interface, permits independent development, testing and maintenance. The Model contains the data and business logic, the View contains the visual layout and the presentation of the entire application.

In using this pattern, relationship between components is established via interactions. The Model is only aware of itself, View is aware of the Model, as it is responsible for rendering data contained in the Model and invoking actions(methods/functions) on the Model. The Controller on the other hand, creates and populate the Model and present it to the View. The Controller must be aware of the Model and how to resolve the View and make it available to the Model.

4.9.1 Views

View renders the model to a correct form suitable for interaction, typically a user interface element and allows the existence of multiple views for a single model for different purposes. A viewport typically has a one to one correspondence with a display surface and knows how to render it. The view in Angular JS application is created by Template(HTML), because Angular JS supports SPA multiple views which can be on a single page. To achieve having multiple view on a single page, directives like ng-view and ng-template have been provided.

4.9.2 Controllers

Controller receives user input and initiate responses by making calls on the model objects. A controller accepts input from the user and instructs the model and viewport to perform action based on that input. The controller in Angular JS is defined by using the ng-controller directive. With the controller function in the application, this serves as the main entry point for the controller at runtime and will be called by the Angular JS runtime. In this function, the controller sets up the model and makes it ready in the view. To make this happen, the controller may require various services. Angular JS uses Dependency injection, a technique whereby one object (or static method) supplies the dependencies of another object. A dependency is an object that can be used (a service) /15/, to provide controllers and other objects with the required dependencies. One of the services used is the shopping.services.ts which provides functions for the model used by the view

4.9.3 Model

Model manages the behaviour and data of the application domain, responds to requests for information about its state (usually from the view) and responds to instructions to change state (usually from the controller). In event-driven systems, the model notifies observers (usually views) when the information changes so they can react.

In Figure 13, the interaction between the view and model can be seen. The data from the model can be used on in the view. Angular provides smooth communication which is often referred to as Databinding and which can be either though property binding or event binding or the combination of both, this is Two-Way-Binding, which allows changing the value in the view from a text box.

5 IMPLEMENTATION

This section describes how the code of the application is implemented, where they are used and how they are used to achieve the goal in this application. Angular JS, Angular 5 is used for the frontend of the application and Firebase is used for the database while NodeJS is used to run the backend.

5.1 User Sign Up Component

The user signUp component imports the AuthActions from the store where the rules for authenticating is set. The onSignUp method is implemented with the authentication rules which is the username: email and password: password. The rules are set according to the permission granted in Firebase from which each user registry details are stored as shown in Code Snippet 1

```
onSignup(form: NgForm) {
  const email = form.value.email;
  const password = form.value.password;
  this.store.dispatch(new AuthActions.TrySignup({username: email, password: password}));
}
```

Code Snippet 1: Sign Up Function

5.2 User Sign Up Template

The template (HTML) represents the visual layout of the class on the web page. A form is created and a disabled function is added to render the submit button invalid until every sign-up detail is filled as shown in Code Snippet 2.

```
<div class="row">
  <div class="col-xs-12 col-sm-10 col-md-8 col-sm-offset-1 col-md-offset-2">
    <form (ngSubmit)="onSignup(f)" #f="ngForm">
      <div class="form-group">
        <label for="email">Mail</label>
        <input type="email" id="email" name="email" ngModel class="form-control">
      </div>
      <div class="form-group">
        <label for="password">Password</label>
        <input
```

```
        type="password"
        id="password"
        name="password"
        ngModel
        class="form-control">
      </div>
      <button class="btn btn-primary" type="submit" [disabled]="!f.valid">Sign Up</button>
    </form>
  </div>
</div>
```

Code Snippet 2: Sign Up Template

5.3 User Sign in Component

The registered user can signin/login to access the functionalities (make new or edit collections and add or edit shopping list) of the application. The Signin class imports the AuthActions which sets the rules to authenticate the user. A token is generated every time the sign in function is called. This identifies the user on the page and gets destroyed when the user logs out. This is shown in Code Snippet 3.

```
onSignin(form: NgForm) {
  const email = form.value.email;
  const password = form.value.password;
  this.store.dispatch(new AuthActions.TrySignin({username: email, password: password}) );
 }
```

Code Snippet 3: Sign Up Function

5.4 User Sign in Template

The sign in template shows the visual layout of the sign in page, only the user email and password is required to authenticate the user. Once the sign in details is invalid, the sign in button remain invalid and sends an error message to the user if details are incorrect. If sign details are correct, it redirects the user to the home page.

```
<div class="row">
  <div class="col-xs-12 col-sm-10 col-md-8 col-sm-offset-1 col-md-offset-2">
    <form (ngSubmit)="onSignin(f)" #f="ngForm">
      <div class="form-group">
        <label for="email">Mail</label>
```

```
      <input type="email" id="email" name="email" ngModel class="form-control">
    </div>
    <div class="form-group">
      <label for="password">Password</label>
      <input
        type="password"
        id="password"
        name="password"
        ngModel
        class="form-control">
    </div>
    <button class="btn btn-primary" type="submit" [disabled]="!f.valid">Sign In</button>
  </form>
 </div>
</div>
```

Code snippet 4: Sign in Template

5.5 Collection Details Component

This component handles the function available to each collection in the list. It allows us to manage the collection selected which is only possible via authentication. The registered and logged in user can use the managed collection naviagtion. The functions here allow adding items to shopping list, edit collection and delete collection. The ngOnInit method is the life cycle hook which runs continuously the moment a collection is selected.

```
ngOnInit() {
 this.route.params
  .subscribe(
   (params: Params) => {
    this.id = +params['id'];
    this.collectionState = this.store.select('collections');
   }
  );
}

onAddToShoppingList() {
```

```
  this.store.select('collections')
    .take(1)
    .subscribe((collectionState: fromCollection.State) => {
      this.store.dispatch(new ShoppingListActions.AddDesigns(
        collectionState.collections[this.id].designs)
      );
    });
}

onEditCollection() {
  this.router.navigate(['edit'], {relativeTo: this.route});
 }

onDeleteCollection() {
  this.store.dispatch(new CollectionActions.DeleteCollection(this.id));
  this.router.navigate(['/collections']);
}
```

Code Snippet 5: Collection Detail Component

5.6 Collection Edit Component

In this component there are several methods that all work together to make up the edit collection. The first method implemented is the ngOnInit which is the life cycle hook of the component

```
ngOnInit() {
  this.route.params
    .subscribe(
      (params: Params) => {
        this.id = +params['id'];
        this.editMode = params['id'] != null;
        this.initForm();
      }
    );
}
```

Code Snippet 6: ngOnInit Method

5.6.1 onSubmit Method

onSubmit method checks if in edit mode and dispatches to the UpdateCollection method call from the collections.actions in the store. If not in the edit mode, the AddCollection method is called and a new collection is created as shown in Code Snippet 7.

```
onSubmit() {
  if (this.editMode) {
    this.store.dispatch(new CollectionActions.UpdateCollection({
      index: this.id,
      updatedCollection: this.collectionForm.value
    }));
  } else {
    this.store.dispatch(new CollectionActions.AddCollection(this.collectionForm.value));
  }
  this.onCancel();
 }
```

Code Snippet 7: onSubmit Method

5.6.2 InitForm Method

InitForm method creates the form on which each collection detail is filled. If in the edit mode, the details of the collection are displayed in the box and the detail can be edited. If not in the edit mode, an empty form is created with form validation in place to ensure no input is left empty by the user before submission. A pattern is included for the amount to ensure users cannot enter string or negative numbers. The method is made private to ensure it is only called within this component.

```
private initForm() {
  let collectionName = '';
  let collectionImagePath = '';
  let collectionDescription = '';
  let collectionDesigns = new FormArray([]);

  if (this.editMode) {
    this.store.select('collections')
      .take(1)
      .subscribe((collectionState: fromCollection.State) => {
```

```
      const collection = collectionState.collections[this.id];
      collectionName = collection.name;
      collectionImagePath = collection.imagePath;
      collectionDescription = collection.description;
      if (collection['designs']) {
        for (let design of collection.designs) {
          collectionDesigns.push(
            new FormGroup({
              'name': new FormControl(design.name, Validators.required),
              'size': new FormControl(design.size, Validators.required),
              'amount': new FormControl(design.amount, [
                Validators.required,
                Validators.pattern(/^[1-9]+[0-9]*$/)
              ])
            })
          );
        }
      }
    });
  }
```

Code Snippet 8: initForm Method

5.7 ShoppingList Edit Component

This component is responsible for pushing the shopping list items into the database, the submit method and the reset form method are called in this component. The reset form method is called from the ngOnInit method which is the life cycle hook of the component. The insertDesign and updateDesign method will be called from the shoppingService and this service is imported into this component.

```
ngOnInit() {

  this.resetForm();
  }
```

```
  onSubmit(shoppingForm: NgForm){
    if(shoppingForm.value.$key == null)
     this.shoppingService.insertDesign(shoppingForm.value);
    else
     this.shoppingService.updateDesign(shoppingForm.value);
    this.resetForm(shoppingForm);
    this.toastr.success("Submitted Successfully", 'Shopping List');
  }

  resetForm(shoppingForm?: NgForm){
    if(shoppingForm != null)
    shoppingForm.reset();
    this.shoppingService.selectedDesign = {
      $key: null,
      name: '',
      size: '',
      amount: 0
    }
  }
```

Code Snippet 9: shoppingEdit component

5.8 Shoppings List Component

In this component, item values are edited and deleted in real time from the database. Users can edit (update and delete) their submitted and saved item. The shoppingService is imported to allow the use of some methods already implemented in it.

5.8.1 ngOnInit Method

This method is always called and it gets data from the database. It fetches each user's data from the database and displays on the shopping list page.

```
ngOnInit() {
  this.shoppingService.getData();
  var x = this.shoppingService.getData();
  x.snapshotChanges().subscribe(item => {
```

```
    this.shoppingsList = [];
    item.forEach(element => {
      var y = element.payload.toJSON();
      y["$key"] = element.key;
      this.shoppingsList.push(y as Design);
    });
  });
 }
```

Code Snippet 10: ngOnInit Method

5.8.2 onEdit and onDelete Methods

The onEdit and onDelete methods are responsible for manipulating the data and respond when they are called

```
onEdit(dsg : Design){
   this.shoppingService.selectedDesign = Object.assign({}, dsg);
 }


 onDelete(key : string){
   if(confirm('Are you sure you want to delete this item ?') == true){
   this.shoppingService.deleteDesign(key);
   this.toastr.warning("Deleted Successfully", "Shopping List");
    }
 }
```

Code Snippet 11: onEdit and onDelete Methods

5.9 App Routing Module

The routing module shows how the routing of the application is carried out, it imports components to be available in the navigation of the application and their paths are set as seen in the Code Snippet 11. The AuthGuard is imported and used to access the page that needs authentication. The collections is loaded with children and this is called from the collections modules, which loads every child collection component.

```
const appRoutes: Routes = [
 { path: '', component: HomeComponent },
 { path: 'contact', component: ContactComponent },
```

```
  { path: 'about', component: AboutComponent },
  { path: 'collections', loadChildren: './collections/collections.module#CollectionsModule'},
  { path: 'shopping-list', component: ShoppingListComponent , canActivate: [AuthGuard]},
];
```

Code Snippet 12: App routing modules

5.10 Firebase setup

To set up Firebase, create a new Firebase app on your Firebase console with your Google Account, add project and give a desired name. Afterwards, "Add Firebase to your web app" is clicked from the project overview. The config key is copied into *src/environments/environment.ts*

```
export const environment = {
  production: false,
  firebaseConfig : {
    apiKey: "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",
    authDomain: "shoppinglist-fe6ae.firebaseapp.com",
    databaseURL: "xxxxxxxxxxxxxxxxxxxxxxxxxx",
    projectId: "shoppinglist-fe6ae",
    storageBucket: "shoppinglist-fe6ae.appspot.com",
    messagingSenderId: "61160140921"
  }
};
```

Code Snippet 13: Adding firebase to the web app

From the Database tab, the Rules are edited depending on how it should respond on the web page, and then published. /8/

5.11 Index.html

This is main view of the application from which other views are injected and displayed as a Single Page. Every other view in the app is initiated from the app-root. The app components are injected through the <app-root></app-root>.

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
```

```
  <title>MyClothingApp</title>
  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
</head>
<body>
  <app-root></app-root>
</body>
</html
```

Code Snippet 14: Index.html of the app

6 Graphical User Interface (GUI) DESIGN AND DATABASE

The GUI provides the platform the user interacts, which includes the design and implementation of the application. With this interface users can explore the functions of what is being designed and then manage the functions accordingly. Without the GUI, the application will be of little or no use to users as there would be nothing for them interact with.

6.1 Home Page

The home page of the application as shown in Figure 14 is the first page the user is exposed to once it is opened from the web browser. Home page welcomes the user and show a brief information of what the company does, show available navigations at the top of the page to help user manage the application.

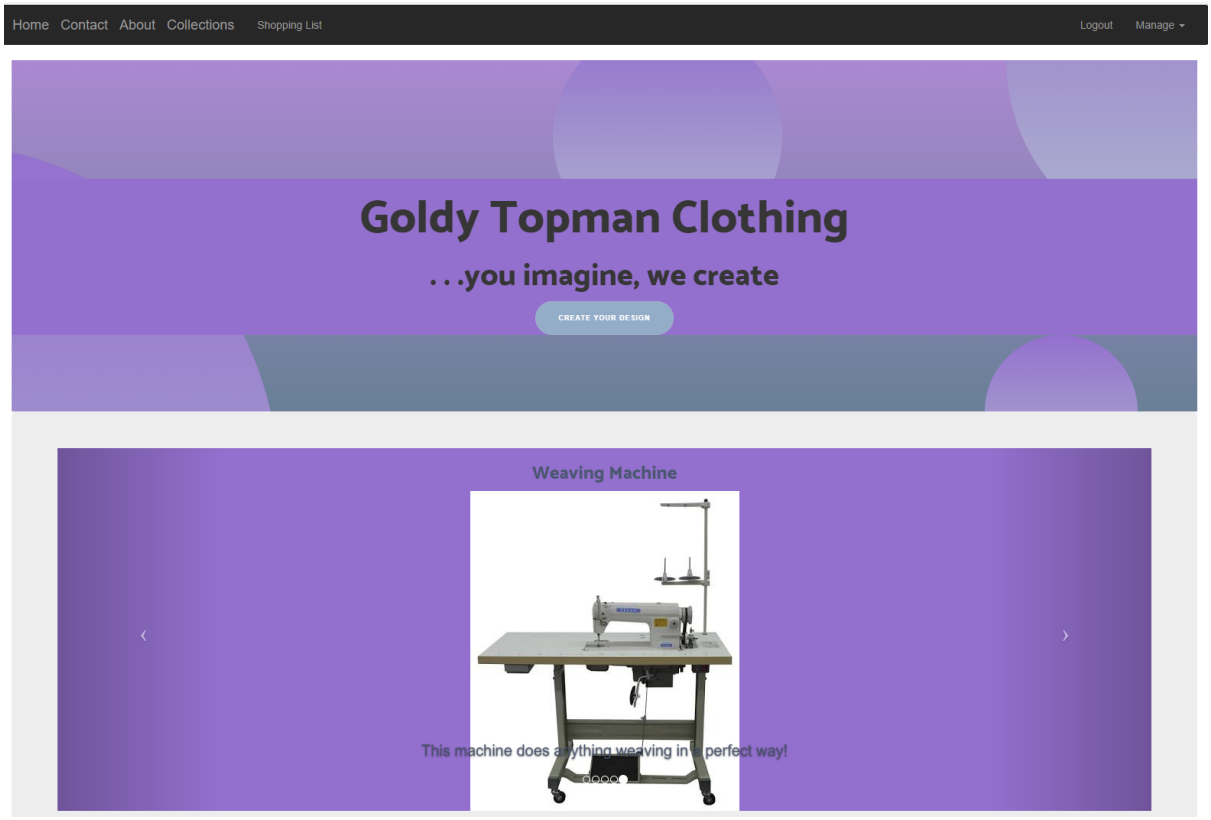

Figure 14: Home Page

6.2 Register Page

In the Register page of the application, the user registers with email and password in which validators has been added in the HTML code to check for valid user input. On successful registration, the user is redirected to the home page of the application. The navigations in the header component of the application has been designed to initially display Register and Login, and on either Register or Login, Register and Login navigations changes to Manage and Logout.

Figure 15: Registration Page

6.3 Login Page

In the Login page of the application, registered users login to access the full functionalities of the web application. On login, users can make their custom design, choose from an existing collection, save new design or designs and fetch saved collection or collections and finally add to shopping list.

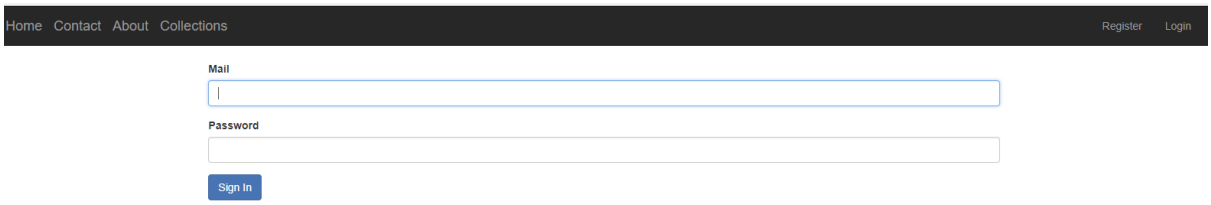

Figure 16: Login Page

6.4 Collection Page

The Collection page of the application consist of components such as collection list, collection item, collection detail and collection edit. From the list of collections, non- registered and logged out users can view collections but not edit the collection and therefore see the full details of the collections, while the registered and logged in users can view collection details and make their custom design either through the edit collection button or new collection button.

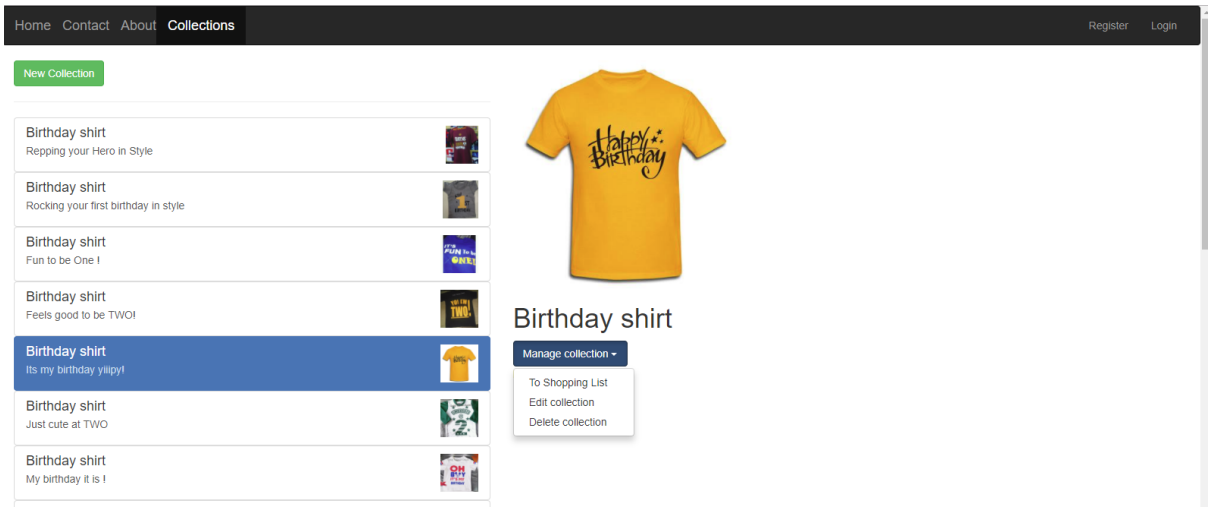

Figure 17: Collection List

In collection detail, the save button is disabled until every field in the form is filled correctly. The user can then add to shopping list or save collection for future use.

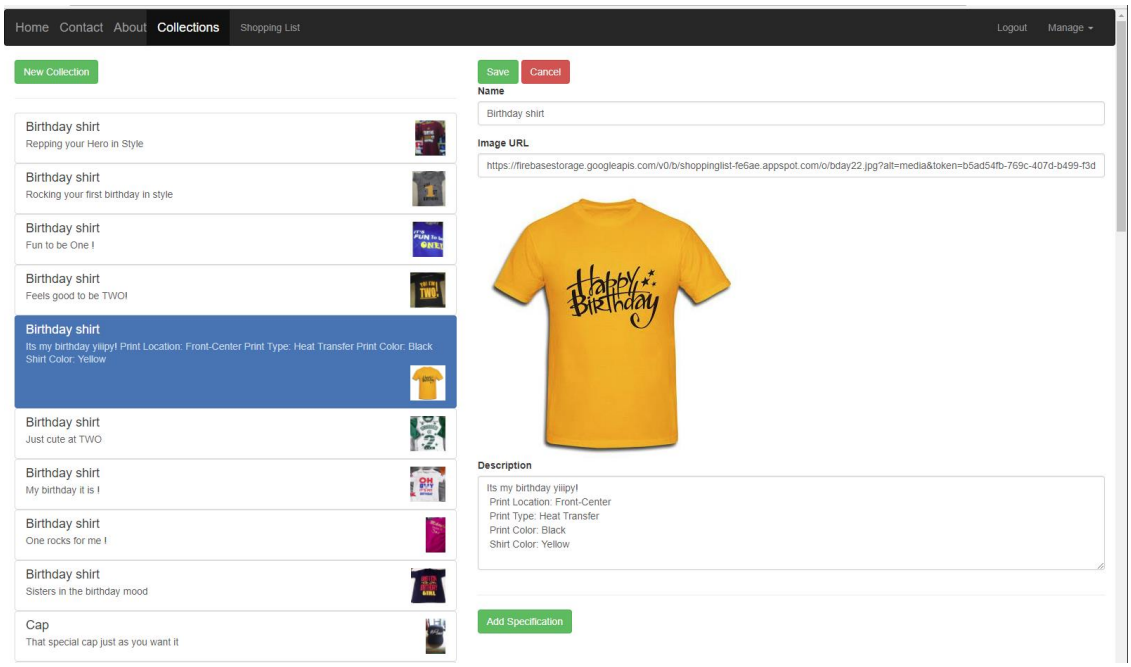

Figure 18: Collection Detail

6.5 Shopping List Page

In the Shopping List page of the application, authenticated logged in users can make the shopping list by adding the items needed which includes item name, quantities and size. Form validators has been included to disable the submit button until valid input is given by the user. The user can edit the submitted shopping list items before notifying the company for further processing.

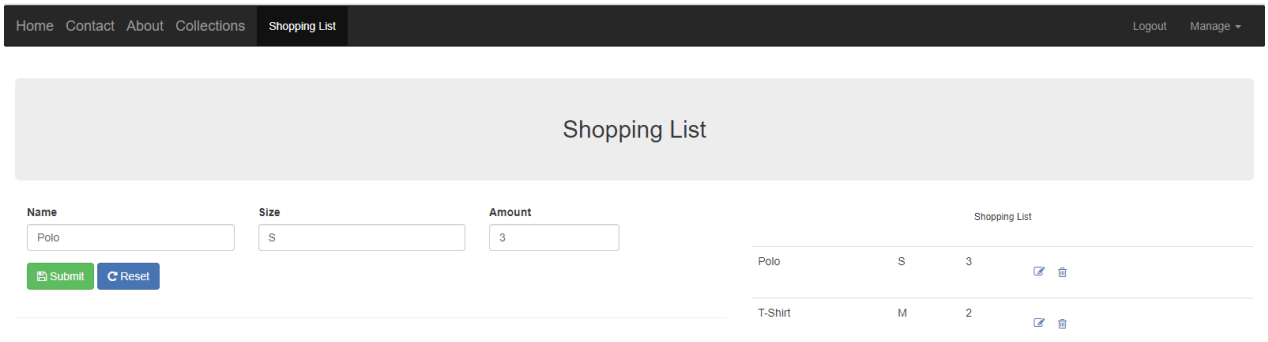

Figure 19: Shopping Edit

In the Shopping edit page of the application, on clicking the edit icon, the user input is displayed in the form fields and with this, user can edit. When user is satisfied and submits data is written to Firebase database in real-time, a successful message is displayed to the user. Suppose the

user wants to delete an item or items from the shopping list items, a pop up message is displayed to confirm if user wants to delete to avoid deleting items accidentally.

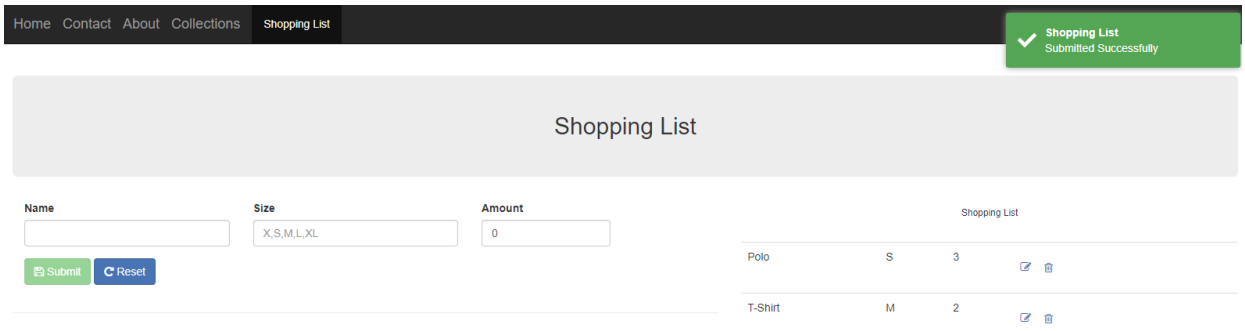

Figure 20: Shopping list with successful submission notification

6.6 Contact Page

The contact page of the application contains the company contact details and a feedback form where both visitors and registered users can drop a feedback message and share their opinion with the company. The contact details of the application have been designed with font-awesome icons to graphically express the details. The form has been designed with validators which ensures every important detail of the form is filled before submission. The submission button is disabled until form validation criteria is fulfilled and form reset function is used to reset form on submission. The successful notification is used to alert users their feedback has been submitted.

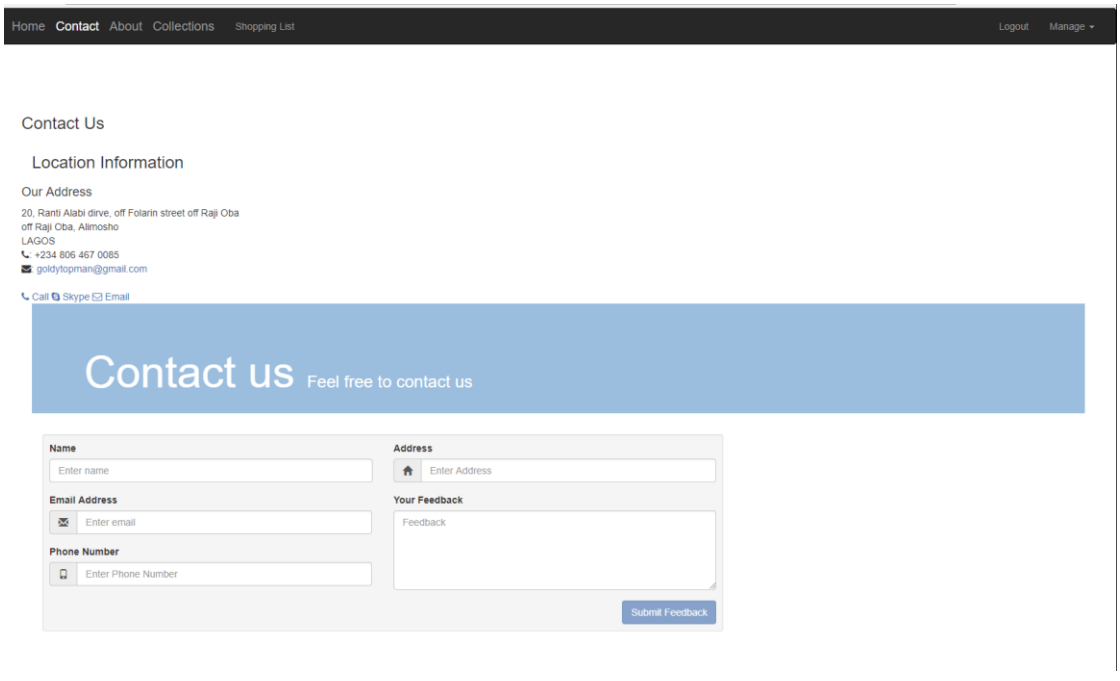

Figure 21: Contact Page

6.7 About Page

The About page of the application contains the information about the company and the founders which are the Chief Executive Officer(CEO) and Chief Design Officer(CDO) of the company. It displays their pictures and their competence. The page has been designed with no links or buttons, just text and images where visitors and registered members can get enough information about the company.

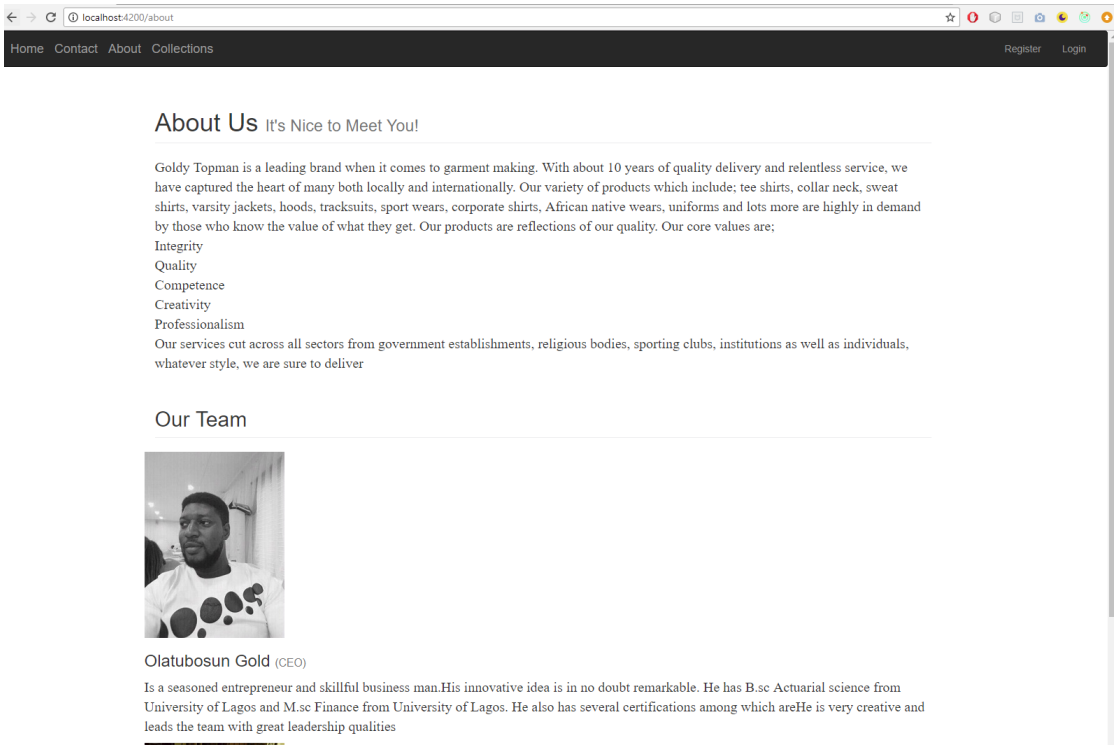

Figure 22: About Page

6.8 Database(Firebase)

The database which has been used in this application is Firebase, it is a real-time cloud based database developed by Google. The dashboard for authentication gives the options to view registered users and sign-in method. From the user's option, users can be added and removed by the admin. The sign-in method allows the admin to determine the sign-in pattern, this could be through social media platforms, personal emails or with email and password as used in the application to avoid restriction of any form. Firebase uses this to authenticate each user on the application and a unique identification for every registered user is created and finally generates a token on successful registration or login to access the all functionalities available in the application. The generated token is destroyed when users logout or closes the application.

Firebase is where the collections are saved and every custom design made by registered users are saved and fetched. The shopping list collections is saved in the database which is visible to

the administrator for management use. Since Firebase is cloud based, data is stored and managed over the cloud. Local machine is not needed to access and manage the data as data can be managed over the internet. Contact information which includes feedback from visitors and registered users are stored in the database and available to the admin for management and data handling.

Figure 23: Contact Data

Figure 23 shows the data stored for each user from the contact page of the application. Each data stored is uniquely identified by its own unique identification generated by Firebase.

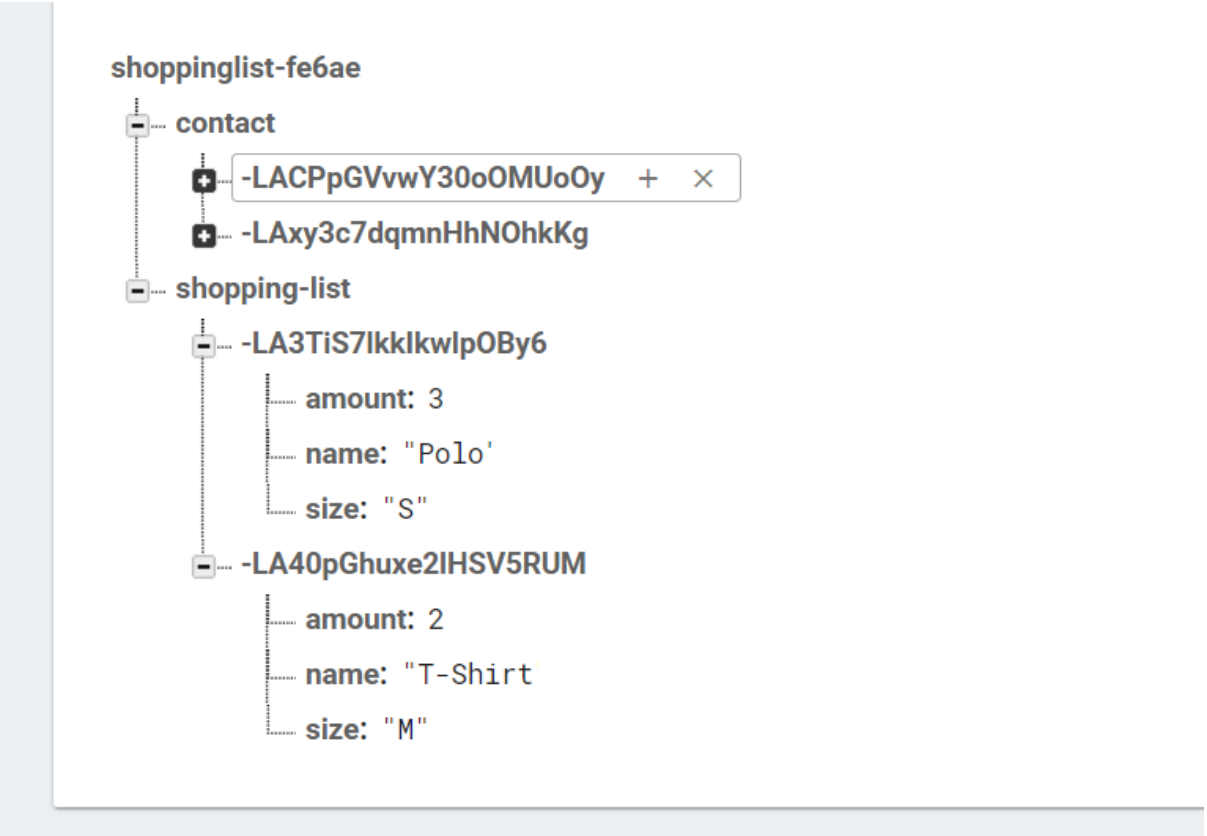

Figure 24: Shopping List Data

Figure 24 shows shopping list items submitted by the users in the shopping list page of the application. In the shopping list page of the application, users can manage their items from the application, which includes, edit, delete and add.

7 TESTING AND RESULT

Angular provides the opportunity to test the application against many odds that could be unnoticed in the development of the application. Testing application involves verifying the behavior of components which have different roles in the application. Some of the testing available in Angular are;

7.1 Unit Tests

Unit test is a test of the component in isolation and this is done without the template code. It is easy to write and fast to execute. Unit test is executed without external resources like database, file, browsers, etc. This test has a single responsibility, it tests only one thing

7.2 Integration Tests

Integration testing in Angular is done with external template and the components are tested with external resources

7.3 End-to-end Tests

The End-to-end test is done on the entire application, it tests the application functionality and gives more confidence on the application. It is sometimes slow and fragile as it involves high level details in testing procedures

7.4 Angular Testing Tools

In this thesis, the testing tools used includes:

6.4.1 Jasmine

Jasmine a behavior-driven development framework for testing JavaScript Code, it does not require any other JavaScript frameworks. It does not require a DOM and it has a clean and obvious syntax to write tests with ease. /16/

7.4.2 Karma

Karma is a test runner for writing and running unit tests while developing Angular applications. It increases productivity and with Karma, multiple browser test is achievable.

To carry out unit test on the application, the following are the fundamentals required;

- test files with the ". spec.ts" extension. These files are automatically generated when a component is generated using the Angular CLI.
- Angular CLI where the running the test is carried out using the "ng test" command

The test file contains some functions which explains what is to be done in the test. Some of the functions used in the test includes;

- describe(), this defines a suit that is a group of related tests

- it(), this defines a spec or test
- expect(), this takes the actual value as the parameter chained with a matcher function
- matcher function, this takes the expected parameter value and reports to Jasmine if the expectation is true or false.

This thesis focusses on unit test which involves running the code and verifying aspects of its behavior which includes function calls or return values. This test considers single functions as well as interaction between a service and dependencies, the unit test is distinct from end-to-end(E2E) tests which involves using browser automation like Selenium to interact with the application's interface and it also verify high level details. Unit test can enhance productivity and give direct information about the nature of failures encountered.

To get started, the Karma is needed which at its core, launches instances any of the web browsers selected, loads the files specified and reports the results of the test from the browsers back to the terminal which is currently opened. With Karma, the test can run in multiple browsers as configured.

```
plugins: [
      require('karma-jasmine'),
      require('karma-chrome-launcher'),
      require('karma-ie-launcher'),
      require('karma-firefox-launcher'),
      require('karma-jasmine-html-reporter'),
      require('karma-coverage-istanbul-reporter'),
      require('@angular/cli/plugins/karma')
    ],
browsers: ['Chrome', 'IE', 'Firefox'],
```

Code Snippet 15: Karma configuration for multiple browsers test

Services like shoppingService use the Angular Firestore to store and retrieve user entry on authentication. The services have been provided in the spec.ts file and tested accordingly to verify communication between the function calls and the services provided.

```
describe('shoppingService', () => {
  beforeEach(() =>{
    TestBed.configureTestingModule({
      providers: [ShoppingService, {provide: AngularFireDatabase, useValue:firestore}]
    });
  });
    it('should be created', inject([ShoppingService], (service: ShoppingService)=>{
    expect(service).toBeTruthy();
    }));
```

Code Snippet 16: Providers using shoppingService and Angular Firebase

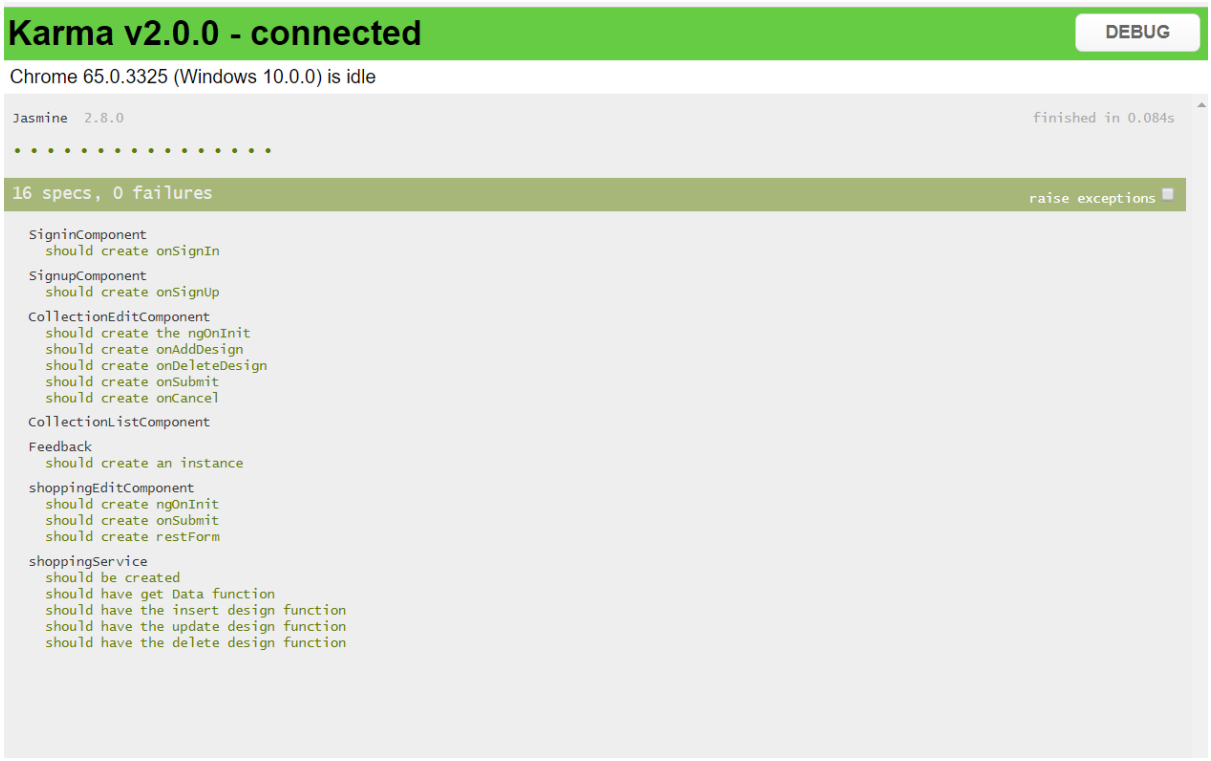

Figure 25: Test results of various component functions

Figure 25 shows the test result in the test runner(Karma) from the web browser. Various components used in the applications with its functions and services has been tested as shown in Figure 25 and the result shows the functions does not contain bugs. Dependency injection which includes services has been tested in function calls and the test result passed.

The application has been tested on several devices to be sure user gets same UI experience.

8 CONCLUSION AND FURTHER IMPROVEMENTS

This documentation includes the necessary information on the structure and building of the Custom Clothing Web Application. It has considered the use of different technologies and it involves communication between the frontend and backend of the application. It has considered feasibility studies of similar applications, concise analysis, design and implementation.

The use of Angular has proven to be one of the best technology for the design and implementation of this application with its ability to provide SPA which offers very good user experience and responsiveness, fast implementation and helpful tutorials to manage difficult tasks. The communication of Angular with Firebase on the backend has helped develop the application and incorporate functionalities with less difficulty. The use of Firebase has helped to achieve functions with less complex code.

Furthermore, the application has been tested against bugs and other factors that could have been unnoticed in period of development. The use of in-built testing environment provided by Angular CLI has made the test easy to write and monitor in real-time.

Creating a unique design from scratch was one of the challenging task faced in the development of this application which involves creativity and structuring.

For further improvements, Angular offers lots of functionalities that were not fully explored in the design and implementation of this application. These functionalities would help improve the interface by adding more components and animations, make the web page more dynamic for users and provide more services for users to make their custom cloths. The view of the application can be further improved, in which user input in the description field can be seen dynamically on every selected collection without just listing them in the description box of the application.

The database can be further designed with a more robust NoSQL database like MongoDB in the future when the data size increases.

Lastly, full test should be done on the application such as the integration and end-2-end(E2E) test which involves browser automation, useful for insurance policy and sanity-check. This test will alert when something goes wrong for deep investigation.

REFERENCES

/1/ Customin.com: Accessed 16.04.2018
https://www.customink.com/ndx/#/

/2/ Rush Order Tees: Accessed 16.04.2018
https://www.rushordertees.com/design-t-shirts/

/3/ Spread Shirt: Accessed 16.04.2018
https://www.spreadshirt.com/custom/t-shirts

/4/ Vistaprint: Accessed 16.04.2018
https://www.vistaprint.in/clothing-bags/mens-t-shirts

/5/ Web Application: Accessed 16.04.2018
https://www.lifewire.com/what-is-a-web-application-3486637

/6 / Backend development: Accessed 16.04.2018
https://www.upwork.com/hiring/development/a-beginners-guide-to-back-end-development/

/7/ Document Object Model: Accessed 16.04.2018
https://developer.mozilla.org/en-US/docs/Web/JavaScript/JavaScript_technologies_overview

/8/ TypeScript: Accessed 16.04.2018
http://www.typescriptlang.org/

/9/ NodeJS: Accessed 16.04.2018
https://nodejs.org/en/

/10/ Firebase: Accessed 16.04.2018
https://medium.com/factory-mind/angular-firebase-typescript-step-by-step-tutorial-2ef887fc7d71

/11/ Firebase vs MongoDB: Accessed 16.04.2018
https://stackoverflow.com/questions/29223835/mongodb-vs-firebase

/12/ Visual Studio Code: Accessed 16.04.2018

https://code.visualstudio.com/docs

/13/ Unified Modelling Language: Accessed 16.04.2018
https://www.techopedia.com/definition/3243/unified-modeling-language-uml

/14/ Components: Accessed 16.04.2018
https://docs.angularjs.org/guide/component

/15/ Dependency Injection: Accessed 16.04.2018
https://en.wikipedia.org/wiki/Dependency_injection

/16/ Jasmine: Accessed 16.04.2018
https://jasmine.github.io/2.4/introduction.html