Josh Munro

[Company name]  [Company address]

web701 Assessment three

Table of Contents

[Repository 1](#_Toc75551976)

[Other notes 1](#_Toc75551977)

[Part Two Implementation and testing 2](#_Toc75551978)

[Exploring and Describing web technology used in the web app. 4](#_Toc75551979)

[Introduction of web technology 4](#_Toc75551980)

[How I implemented the web technology 4](#_Toc75551981)

[Problems I have faced. 4](#_Toc75551982)

[Describe the future of Angular web framework. 5](#_Toc75551983)

[References 6](#_Toc75551984)

# Repository

<https://github.com/Josh-Munro/FinalAssessment>

# Other notes

I have used Mongo Compass for my database. However, the schema and collections should automatically be created inside local host when the server is run.

# Part Two Implementation and testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Functionality | Should Pass Values | Expected Result | Result | Should Fail Value | Expected Result | Result |
| Register | [Josh-munro@live.nmit.ac.nz](mailto:Josh-munro@live.nmit.ac.nz) | Valid email | Pass | bfwufbufw | Invalid email | Fail |
| Register | 12345 | Pass | Pass | Null | null | null |
| Login | [Josh-munro@live.nmit.ac.nz](mailto:Josh-munro@live.nmit.ac.nz) | Valid email | Pass | [Not-joshmunro@live.nmit.ac.nz](mailto:Not-joshmunro@live.nmit.ac.nz) | Invlaid email | Pass |
| Login | 12345 | Correct Password | Pass | Edbuefbuf | Invalid password | Pass |
| Admin dashboard | Email: Admin  Password: Admin  Login | Admin specific components | Pass | Email: Beneficiary  Password: Bene | Beneficiary components | Pass |
| Testing is admin textbox when registering | Is Admin Checkbox checked | The user role is an Admin | Pass | Is Admin checkbox is un-checked | The user is not an admin | Pass |
| Item being redeemed | Item Name: Potato Soup  Item Promocode: ILOVESOUP | Item Redeemed successfully | Pass | Item Name: Potato Soup  Item Promocode: ILOVESOUP | Item could not be redeemed | Pass |
| Creating an Item | Name: Onion Soup  Promocode: ILOVESOUP | Item created successfully | Pass | Null values | Item could not be created successfully | Pass |
| Displaying Items correctly | Display list of items Name and Promocodes from database | Display correct values from database | Pass | Null, data is from incorrect collections, missing information | Null | Pass |
| Displaying Beneficiaries correctly | Display list of Beneficiaries Name and Promocodes from database | Display correct values from database | Pass | Null, data is from incorrect collections, missing information | Null | Pass |
| Home route | Home route when clicked takes user back to home page | Home route directs user to home page | Pass | Null | Null | null |
|  |  |  |  |  |  |  |
| About route | About route when clicked takes user back to home page | About route directs user to about page | Pass | Null | Null | null |
| dashboard route | Dashboard route when clicked takes user back to home page | Dashboard route directs user to dashboard page | Pass | Null | Null | null |
| Login route | Login route when clicked takes user back to login page | Login route directs user to login page | Pass | Null | Null | null |
| Register route | Register route when clicked takes user back to home page | Register route directs user to home page | Pass | Null | Null | null |

# Exploring and Describing web technology used in the web app.

## Introduction of web technology

Quoted from the official Angular website, “Angular is a development platform, built on TypeScript” (*Angular - Introduction to Angular Concepts*, n.d.). Angular can be a component focused framework for developing scalable and reusable code (*Angular - Introduction to Angular Concepts*, n.d.). There is an excellent range of library features that allow for routing, form management, two-way data binding and many other useful features (*Angular - Introduction to Angular Concepts*, n.d.).

A major benefit of Angular is it ability to scale from single-developer applications to multi-user enterprise application (*Angular - Introduction to Angular Concepts*, n.d.). Whereas other frameworks struggle to do so.

The structure of Angular relies on components designed and organised through NgModules (*Angular - Introduction to Angular Concepts*, n.d.).

The components of Angular uses Services, to bring specific login that are included directly to the component (*Angular - Introduction to Angular Concepts*, n.d.).

Angular has a neat way of sharing data across different components using Services. This is allowed through the use if dependency injection (*Angular - Introduction to Angular Concepts*, n.d.).

With all these features together Angular has a unique approach to how a component-based web framework should work (*Angular - Introduction to Angular Concepts*, n.d.). However, these rules are set in place to make the development rigid and simpler to use.

## How I implemented the web technology

I implemented the web technology by installing node js on my desktop computer. Node js is a javascript engine that allows web frameworks to be written without dependencies on the web browsers. After the installation of Node js, I opened Visual Studio Code. Next, I installed angular cli using this command npm install -g @angular/cli. After installing Angular, I created a new Angular application. I did this by typing ng new CanGive – the app was named CanGive. In doing all of this I had successfully implemented the Angular web technology.

The next step was to use the build components through the CLI, as Angular is a framework that consists of building blocks called components. Components are made up of a html file, CSS file and a Typescript file. I implemented my components in my web application by repopulating information into this file accordingly.

Since Angular is a Single Page Application framework and is based upon the MVC pattern (model, view controller), I had to install and import Router and other features to properly implement a single page application. By implementing the Router, the framework was able to render different components/views depending on what component the user wants to see.

Services also played an important role in the development of my Angular website. Angular Typescript supports dependency injections; thus I was able to inject services into the constructor of my chosen components, making it easy to access and pass data through the service and components.

With all these features and tools, I was able to fully implement Angular and build a client side application.

## Problems I have faced.

There were many problems I had faced and overcome through lots of trial and error.

A simple problem I had to begin with, was how to add images to the website. I spent a while figuring out why my image will not work or display. However, after research I was able to conclude that the images are best put into the public folder, where assets and images are easily accessible through the Angular application.

Another problem I had to solve was how to move information from the Angular forms to the API. I had initially created a login component, however I wanted to recreate a similar component for the purchasing of a token. In the API mixed the component information variables from the login component with the names of the token component variables, therefore the information was not correctly passed to the API from the service.

A skill I wanted to learn was how to find a way to store temporary data of the current user logged into the website. After many trials and failures in trying to implement this feature on my own, I had to get help from the tutor to show me some direction. The tutor taught how there is a get and set method that stores temporary data when the login API returns the users information once the login was successful. Once I had solved this problem, it was really relieving as it meant that I could store temporary data to enable more dynamic and relevant content for the user types.

Another problem I faced, was creating the routes for the website. Initially I had created the routes paths and installed Angular-router and the necessary packages, however, when implemented a navbar in the root app component the routes were not working. I then discovered that the problem was that I did not incorporate router-outlet tag in the app component which is essential for routing in Angular.

## Describe the future of Angular web framework.

Angular is a mature framework compared to other newer frameworks that have come into the scene over the past decade (*Angular*, n.d.). The future of Angular looks bright and exciting. Whilst adopting Typescript, Angular strives to bring better coding and scripting experience for developers (Sebek, 2021). There are new updates coming to Angular this year that hopes to incorporate stricter type checking for Reactive forms. In doing this the Angular application allows for better bug detection and live errors (Sebek, 2021).

Angular is pushing the design and simplicity of its framework, therefore Angular will be making Zone.js optional. In return the Angular framework will smaller in size and improves debugging (Sebek, 2021).

From my research around Angular, as I analysed their pros, cons, and differences between other frameworks. I am glad to say that Angular continues to be a popular and in demand framework for modern web developers (*Is Angular Dying Because of React?*, n.d.). I foresee Angular to continue building upon its framework, making it more friendly, powerful, and simpler for Angular developers.

For the impact of internet users and developers Angular is known to be the most powerful and scalable framework out there (*8 Reasons Why You Need Angular Today | Grazitti Interactive*, n.d.). Therefore, Angular has a place in future business and applications that allow for large scalability and popular use of internet users. Compared to React and Vue, Angular can scale and have a larger scope when developers want the comfort of knowing that the application is future proof (*8 Proven Reasons You Need Angular for Your Next Development Project*, n.d.). This also means that future users of the Angular framework can have a great user experience as Angular is able to bring power and speed no matter the size of the application.

Angular will continue to be a popular framework, and will continue to do things its own ways, which will have an impact of developers of Angular. Single Page Applications are becoming more popular and hasn’t reached its peak in the internet world.

# References

*8 Proven Reasons You Need Angular for Your Next Development Project*. (n.d.). Grazitti Interactive. Retrieved June 25, 2021, from https://www.grazitti.com/blog/8-proven-reasons-you-need-angular-for-your-next-development-project

*8 Reasons Why You Need Angular Today | Grazitti Interactive*. (n.d.). Retrieved June 25, 2021, from https://www.grazitti.com/blog/8-proven-reasons-you-need-angular-for-your-next-development-project/

*Angular*. (n.d.). Retrieved June 5, 2021, from https://angular.io/

*Angular—Introduction to Angular concepts*. (n.d.). Retrieved May 16, 2021, from https://angular.io/guide/architecture

*Is Angular dying because of React?* (n.d.). Retrieved June 25, 2021, from https://www.dottedsquirrel.com/is-angular-dying-because-of-react/

Sebek, M. (2021, May 13). *The Future of Angular 12 and Beyond*. Medium. https://michelle-sebek.medium.com/the-future-of-angular-12-and-beyond-524e0d38dc64