**Background research**

Potential Frameworks

* Angular (Google)
* React (Facebook)
* VueJs
* Plain HTML, CSS, and JavaScript (don’t use a framework)
* Django

Potential Languages:

* JavaScript
* TypeScript
* Python

Potential Styling libraries:

* Bootstrap
* Foundation
* Bulma

**What is Angular?**

It is a Typescript based framework for building “single-page client applications using HTML and TypeScript” [4]. It is an open-source project started by a google team.

**What is React?**

It is a Component based JavaScript framework for creating complex UI’s. “It is maintained by Facebook and a community of individual developers and companies” [5]. And is one of the more popular frameworks. And supports many 3rd party libraries and modules.

**What is VueJs?**

It is a non-monolithic framework, normally used in conjunction with other libraries to build a fully-fledged Modern website it “is a progressive framework for building user interfaces” [6]. And the “The core library is focused on the view layer only” [6].

**What is Django?**

“Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design” [7]. It used python as its language of choice, making it easy to implement.

**What is Bootstrap, Foundation and Bulma?**

**Bootstrap** is a “a collection of HTML, CSS, and JavaScript tools for creating and building web pages and web applications” [8]. It makes it quick and easy to produce a good-looking website and allows the developer to spend more time working on functionality, than worrying about the style of components.

**Foundation is a** “Foundation is a family of responsive front-end frameworks that make it easy to design beautiful responsive websites, apps and emails that look amazing on any device [9]”. And claims to be the most advanced responsive front-end framework. It is also an open-source project hosted on Github. It is composed of HTML CSS and JavaScript.

**Bulma** is a “free, open-source framework that provides ready-to-use frontend components” [10]. It is a relatively new library. However, this technology is easy to use and its implementation and is similar too a vanilla bootstrap implementation.

**What is JavaScript?**

Mozilla defines JavaScript as **“a scripting language that enables you to create dynamically updating content, control multimedia, animate images, and pretty much everything else.”** [12]. It is also the language of choice for the module then general. It is largely used in web development.

**What is Python?**

Python.org describes its language as **“an interpreted, object-oriented, high-level programming language with dynamic semantics”** [13]. It is easy to use with simple syntax and has a large range of 3rd part modules and libraries. Python is generally only used for font-end web development injunction with Python, however, is extensively used for backend processes.

**What is TypeScript?**

In short TypeScript is JavaScript but enforced static typing. The TypeScript website describes its language as **“an open-source language which builds on JavaScript, one of the world’s most used tools, by adding static type definitions.”** [14]. It is rising in popularity within the web-development industry.

**Justification of the selected technology.**

Selected technologies:

* **React** (Framework)
* **Bootstrap** 4 (Styling CSS classes/Components)
* ‘**react-bootstrap**’ (A Combination of the two libraries above)
* **JavaScript**

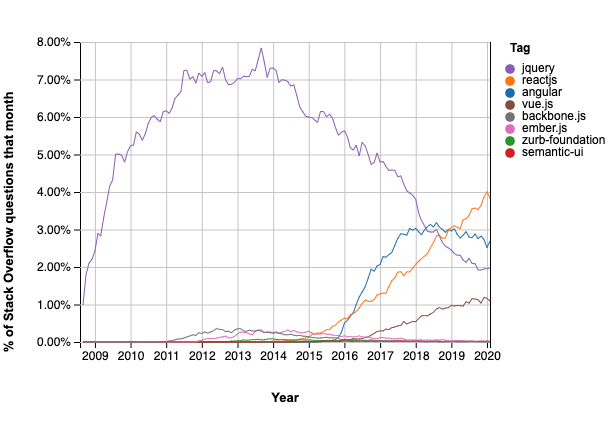
**Reasonings per technology.**

**Why React?**

Firstly, React claims:

**“React makes it painless to create interactive UIs” [1]**

As it did not make sense to create a website using plain html/CSS/JavaScript and we were permitted to use frameworks. As there has been no reason to do so for many years when robust frameworks have been developed to improve quality and security of front-end technologies. React was chosen as it would allow the team to broaden their skills through learning a new front-end framework. We choose ReactJS specifically as it is now one for the most popular frameworks for frontend development as we can see from this graph from stack overflow highlighting the increase of questions related to ReactJS. It also seemed that it would be good opportunity for the team to increase their employability by learning a popular web framework.

**Why React over Angular, Vuejs, Django?**

We decided not to use Angular as it is typescript based, as the module we are doing teaches JavaScript, we decided it would be more familiar to the team to use a framework that uses the JavaScript Language.

We decided not to use Vuejs either despite it also using JavaScript, because of the vast amount of help and support on the internet targeting ReactJS. Due to its popularity and the number of 3rd party libraries it supports.

Django was immediately disregarded as it lies so far out of the scope of the module that we decided not to use it.

**Why Bootstrap 4?**

Bootstrap 4 was chosen as it enables an inexperienced web-development team to produce a visually appealing websites, in the case where the team processes little CSS experience or knowledge, as only 1 member of the team had ever done any frontend web development before. It allowed us to spend more time working on the dynamic elements of a Dynamically Authored Website, than spending 90% of our time trying to fix CSS issues. Importantly Bootstrap is well developed and learning resources are in high supply. And it is implemented in react-bootstrap which is discussed below.

**Why ‘react-bootstrap’ a special combination of ReactJS and Bootstrap 4?**

Firstly ‘react-bootstrap’ claims to be:

**“The most popular front-end framework” (3)**

This again will allow us to expand our employability, as many companies use this framework that we now have experience with.

Main reason to use this library was, that it simplified the relationship between React and Bootstrap, but also still left open the option to use Bootstrap as it was originally designed or too even write totally custom CSS. This is an extremely strong tool for the in-experienced web developer. And again, provided us more time to work on key dynamic elements rather than getting bogged down on CSS issues. The wide range of documentation and guides make this a good choice for a University Project, as we have not explicitly been taught these frameworks so strong resources and guides are key.

**Why JavaScript?**

JavaScript was chosen as the module content was taught in JavaScript. And most web development is using JavaScript. Despite the rise in popularity of Typescript for example. We decided to use JavaScript as React uses it by default and the module was taught in JavaScript, meaning team members we equipped to use the language within the coursework.

**Implementation**

**Usage of 3rd Part libraries and Resources**

We implement the following libraries and do not take credit for the code within them:

**Awesome Fonts**

    "@fortawesome/fontawesome-svg-core": "^1.2.34",

    "@fortawesome/free-brands-svg-icons": "^5.15.2",

    "@fortawesome/free-regular-svg-icons": "^5.15.2",

    "@fortawesome/free-solid-svg-icons": "^5.15.2",

    "@fortawesome/react-fontawesome": "^0.1.14",

Awesome fonts were used to implement some icons and fonts.

**React Libraries**

    "react": "^17.0.1",

    "react-bootstrap": "^1.5.0",

    "react-cookie": "^4.0.3",

    "react-dom": "^17.0.1",

    "react-google-maps": "^9.4.5",

    "react-media": "^1.10.0",

    "react-router-dom": "^5.2.0",

    "react-scripts": "^4.0.3",

    "react-social-embed": "0.0.3",

    "react-twitter-embed": "^3.0.3",

    "@react-google-maps/api": "^2.1.1",

The most import libraries in the react section are react 17.0.1 and react-bootstrap 1.5.0

**React**: was the main framework used to build the website.

**React-Bootstrap** was a library that combined react and bootstrap that help use build a visually appealing website, with limited experience in web development.

**React-router-dom + react-dom:** was used to implement dynamic page generation/navigation.

**React-twitter-embed**: was used to embed a twitter post within the website.

**React-social-embed**: was used to embed a Facebook post into the website.

**React-google-maps** was used to implement google API integrations.

**React-scripts**: provided local development helper scripts.

**React-cookie**: provided functionality to create and read cookies from the user’s browser.

Note other libraries are present in package.json but where either created by the default react app or are not being used or where for local development.

**Changes to original design.**

**Header bar**

The original header bars shape was slightly changed, this was done as the new design with much neater and pleasing to the eye once it was developed than that design presented in the wireframe.

**Banner**

We added a web banner as it was mentioned in the specification, this was not in the original design however.

**Addition Pages/Functionality**

We add several additional pages/functionalities.

* Addition of a Registration and Login Page.
* Addition of a Profile Page.
* Addition of a Find us Page.
* Addition of a Contact Page.
* Registration Success Page.

**Standards Compliance**

To adhere to standards compliance regulations, we….

**Gathering and Conditioning data**

**Cookies/Local Storage**

Our website implements cookies. The first implementation was that the website will ask the user to accept/or deny the use of cookies on the site. If the cookie does not exist a prompt, we be displayed to the user. Once the user makes their choice, subsequent returns the website will not display the prompt.

Second for those users who have logged in before the site uses cookies to remember the username, so that a welcome message can be displayed to user along with the date that they last logged in.

**Filters.**

Our website performs both the gathering and conditioning of data on many of the Pages. Both the Car and Parts Listing Page have several input filters that the user can select, to narrow down the displayed items on the page. Ranging from price filters to milage filters and many more. These filters will dynamically update the page as the user interactions with each filter component.

**User Account and login access.**

There is a registration page that allows the user to create an account and a follow up login page allowing the user to login with the created account, each account will also have access to the profile page where the user’s key information will be displayed. These combinations of pages provide significant gathering and conditioning of data, as data must be input, stored, and retrieved on other pages of the site.

**Dynamic Content**

All the cars and part within the website are loaded from a variable, this makes it easy to add cars or parts to the garage website. No further work is required, than adding the item data. This make is easy to add new content to the website and that content will then be immediately render in the appropriate places within the website.

As filters previously mentioned are interacted with the content will dynamically change to reflect the user selected filters.

An improvement to our website would be to add an Admin section with and add the ability to add or remove cars or parts, however as databases were not allowed within this module coursework, we decided not to Mock this feature, as the work would be significantly more to mock this feature than using a database itself.

**Banner**

The home page contains the websites banner, it is a simple design stating the websites name, highlighting that a sale is currently on and informs the user than the garage us open 24/7.

**Image effects** / **Animation**

The website contains many carousels, that will dynamically cycle through each item in the carousel, image effect buttons overlap the image to allow the user to cycle manually also if desired. Using animations from react-boostrap.

**Dynamic Validation**

We have implemented dynamic validation in the registration page, it will list all the invalid inputs and ask the user too to correct the validation errors. The error prompt alert will dynamically change in real time to reflect the remaining errors that user must correct.

**Dynamic interactive design and event handling techniques**

The Website is fully responsive on every page and implements many even handling techniques.

These include the previously mentioned Login Registration data collection and the Listing pages filters. Each “button”, “slider” or other component utilizes React Hooks to update state and re-render components based on the state of the altered variables, without the page refreshing. For Example, this means that when a user types a password into the Registration Page, there is a progress bar that dynamically updates the strength of the password as the user types without refreshing or navigating to other pages.

This is also the case when a user re-sizes the webpage the content will be dynamically formatted in a responsive manner.

**Our Accessibility**

One of our key goals in address accessibility was that the website should maintain a clear and neat structure regardless of the users selected browser zoom, this allows near sited user to scale the web-browser as they see fit and still consumer the pages content efficiently and without struggle.

**Consideration of performance and optimization.**

To compare our websites performance, we ran it thought the following tool:

<https://www.websitecarbon.com/>

Which allows us to measure the efficiency of our website as compared to all websites globally. Basically, calculating is carbon output.

“An average website produces 4.61 grams of CO2 for every page view. For websites that have an average of 10,000 page views per month, that makes 553 kilograms of CO2 per year.” [11].

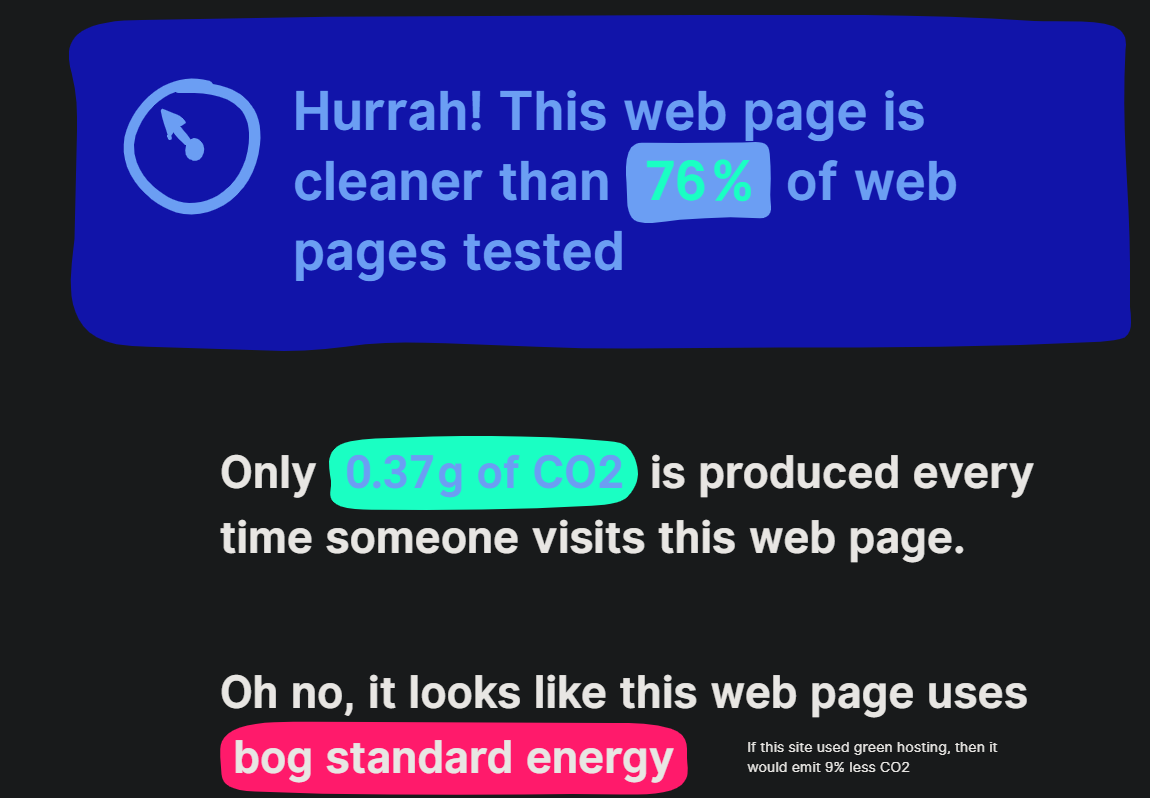
**React Production Build**

React itself has a large amount of built-in optimization features that a user itself does not have be aware of or implement.

React provides the functionality to compile the website into a production build. Production build will significantly speed up site and its efficiency.

**Our energy output/efficiency**

The production build of our website was cleaner that 76% of the web and only produced 0.37g of COD per visit.



**ReactJS and the Virtual DOM**

As we choose to use react as our framework of choice, we would have had to addition work to see the DOM structure, so instead it was suggested that we discuss what a virtual DOM is and the benefits or negatives of using this system for website development.

**Firstly, what is a regular DOM?**

A **DOM** or **Document Object Model** in short is a “tree-like structure that organizes the elements on a web page and allows scripting languages to access them” [15]. This all happens within the browser’s memory.

**How would you define a Virtual DOM?**

**What is Gained from a virtual DOM?**

**What is Lost from a virtual DOM?**

**Testing**

**Manual Testing**

**Automating Testing**

References

1. <https://reactjs.org/>
2. <https://www.simform.com/wp-content/uploads/2020/02/stacktrend.png>
3. <https://react-bootstrap.netlify.app/>
4. <https://angular.io/guide/architecture>
5. <https://en.wikipedia.org/wiki/React_(JavaScript_library)>
6. <https://vuejs.org/v2/guide/>
7. <https://www.djangoproject.com/>
8. <https://www.toptal.com/front-end/what-is-bootstrap-a-short-tutorial-on-the-what-why-and-how>
9. <https://get.foundation/>
10. <https://bulma.io/>
11. <https://en.reset.org/blog/whats-carbon-footprint-your-website-01162020#:~:text=An%20average%20website%20produces%204.61,carbon%20footprint%20of%20their%20websites>.
12. <https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript>
13. <https://www.python.org/doc/essays/blurb/>
14. <https://www.typescriptlang.org/>
15. https://www.codecademy.com/courses/build-interactive-websites/lessons/javascript-dom/exercises/document