Proposed Solution

-Provide a dynamic web application that allows the creation of rota’s, employee and pay roll management

What is a web application?

In order to be able to create a web application or web-app it is important to define what a web application is as this will aim the selecting of what technologies will be of benefit to use. Lotfy (2018), describes a web application as a client-server software program where the backend runs on a web server. The ‘backend’ is what handles the ‘behind-the-scenes’ functionality of the web application, such as storing and retrieving information, and the client-side or ‘front-end’, displays the information provided to a user. Furthermore, the front-end also allows the user to interact and in some cases change the information provided by the backend. There is usually a typical flow when a user interacts with a web application and is as follows, Gibb (2016):

1. A user makes a request to a web server through the front-end interface.
2. A web server forwards this request to the appropriate web application server.
3. The web application server performs the requested task such querying a database or processing the data and generates the results of the requested data.
4. The web application server sends the results to the web server with the requested information or processed data.
5. Web server responds back to the client with the requested information that then appears on the user’s display.

This typical flow of requests from the frontend and backend is what makes up a web application and this idea was used in the creation of the solution. Figure 5 shows this has idea has been used within the application.

Technologies used

There are many different frameworks and ecosystems that are used to create web-applications and they are often referred to as a stack. Stacks are made up of technologies used to create the frontend, a backend and a database, some examples include:

* LAMP stack: JavaScript - Linux - Apache - MySQL – PHP
* Django stack: JavaScript - Python - Django - MySQL.

While each so-called stack has its advantages and disadvantages, for this project a M.E.A.N stack was used. The M.E.A.N stack which is often referred to as ‘mean stack’, or just mean, is a collective of four pieces of technology: MongoDB, Express.JS, Angular, and Node.JS. The strength of using the Mean stack is that it relies on the centralised use of JavaScript as the basic programming component (Elrom, 2016). This allows for robust, easily maintainable web applications with a relatively quick building process (IBM, 2019). While, in a traditional mean stack a non-relational database of MongoDB would be used. However, due to having no experience in working with non-relational databases, a relational database of MySQL was used within this web application.

Node.js

Node.js is an open-source and cross-platform JavaScript runtime environment that uses Google Chrome’s V8 execution engine and enables the development of all kinds of server-side tools and applications. As a result, the runtime is intended to be used outside the browser and as such the environment omits browser specific JavaScript API’s and other traditional API’s including HTTP. This has a number of benefits, one of which being the speed at which it can do this at. This speed comes from it running in a single process and not creating a new thread for every request. Instead node uses a set of asynchronous I/O (input/output) primitives that prevent JavaScript code from blocking (Nodejs. 2019). Other webservers such as Apache use the blocking I/O model to handle requests and use multi threads that increases complexity and overhead for the server as it causes threads to wait for I/O while the server processes and retrieves the data and as such diminish speed and performance (D. Dunka, A. Emmanuel and O. Oyerinde, 2018). Instead when Node.js performs an I/O operation, like reading from the network or accessing a database, it won’t block the thread and resumes the desired operations when the response comes back.

When discussing Node, its support for its built-in package manager using NPM (Node package manager), should not be left out. This is a tool that comes by default with every Node.js installation and allows the access to a set of publicly available, reusable components, that are available and installed using the NPM CLI.

Express.js

In order to add specific handling for different HTTP verbs (e.g. GET, POST, DELETE, UPDATE, etc), separately handle requests at different URL paths ("routes"), serve static files, or use templates to dynamically create the responses a web framework can be used (Mozilla, 2018). While it is possible to do this via all this Node alone, it is time consuming and this is were Express.js, or more simply Express is used. It is described as a minimal and flexible Node.js web application that provides a robust set of features for web and even mobile applications (Express.js, 2020). It manages everything from routing to handling requests and views. Express, allows the creation of a REST API server and makes it easier to write secure, modular and fast Node.js applications with higher efficiency, reliability and less duplications.

Within this application Express has been used within Node to provide responses depending on client requests. Figure 1 shows a simple example of how express responses when a specific URL is requested by the client.

A screenshot of a cell phone

Description automatically generated

Figure 1 - Simple Express response

Sequelize

Instead of writing raw SQL queries to insert, update and delete data from the database an ORM (Object Relational Mapping), called Sequelize was used. Object Relational Mapping is a technique that maps software objects to database schema. By interacting with these objects, it meant that no raw SQL database queries had to written and management of the database was easier. When an object is read, created, updated or deleted, Sequelize creates a SQL string that queries the database and returns the result as a JavaScript object. The benefit of using Sequelize is that allows the use of JavaScript instead of SQL for database querying, which as a result simplifies the code as well as making it more readable.

Sequelize models are an abstraction that represents a table in the database. The model is used to tell Sequelize about the entity it represents, such as the name of the table in the database and which columns it has (and their data types) (Sequelize, 2020). Figure 2 shows an example of a model used within the application.

To query the database the models are used along with a number of Sequelize methods that are converted into SQL statements and an example of such can be seen in Figure 3. As mentioned before, by using Sequelize it makes querying the database a lot easier, especially when more complex SQL statements are required.

A close up of text on a black background

Description automatically generated

Figure 2 - Simple Sequelize Model

A screenshot of a computer screen

Description automatically generated

Figure 3 - Simple Sequelize Query

JavaScript Object Notation (JSON)

JSON is a way to store information in an organised, easy-to-access manner. It gives a human-readable collection of data that can be accessed in a logical manner. It is also used to transmit serialized data over a network connection (D. Dunka, A. Emmanuel and O. Oyerinde, 2018). In this application it is used to transfer data from the web server (Node) to the front-end (Angular) and vice-versa.

Angular

Angular is a JavaScript framework that allows the creation of client-side or front-end applications (Angular.io, n.d.). It provides many features such as components, directives, data binding, form processing, services, and dependency injection (Joshi, n.d.). Angular applications are written using TypeScript, a superset of JavaScript and provides many object-oriented features such as classes, interfaces, and data types. It is then complied into JavaScript so that it can be used on any browser.

Angular uses the Model-View-ViewModel (MVVM) pattern that is derived from the MVC model. The MVVM separates the development of the user interface from the development of the back-end logic (The MVVM Pattern, 2012). There are three components of the MVVC pattern:

1. *The Model*. It refers to the to the domain model that includes a data model along with business logic and validation logic.
2. *The View*. It is responsible for defining the structure, layout and appearance of what the user sees on the screen.
3. *The ViewModel*. The ViewModel is responsible for handling the view logic. The view model interacts with the model by calling methods in the model class. It then provides data from the model in a form that the view can easily use. (The MVVM Pattern, 2012).

A picture containing screenshot

Description automatically generated

Figure 4 - MVVM Design Pattern

Angular supports this MVVM design pattern by using a service to implement the Model. The View is implemented using Angular’s template view (HTML with data bindings). As the View is bound to the ViewModel when a property is changed in the ViewModel it is instantly reflected in the View. The ViewModel is implemented using the Component decorator.

As Angular follows the MVVM design pattern it cleanly separates the view from the models and allows the creation of HTML templates that are dynamically filled with data and automatically update when the data changes. This was one of the main reasons why Angular was chosen to develop the client side of the application.

Bring all the technologies together.

All the described technologies were brought together to develop a dynamic web application. On the front-end Angular has been used to provide an interface for the user, allowing them to make requests and view requests. Node has been used as web server to parse and return user requests. Express has been used to make HTTP requests to the database. Information and is passed between each technology using JSON data (Figure 5).

A screenshot of a cell phone

Description automatically generated

Figure 5 - Application Flow

Why this technology stack was picked.

Due to the time constrains associated with such a project the main reason as why this stack was picked was its wide use of JavaScript. This allowed for a relatively quick build time and meant that it was it was easier to move between writing client side and server-side code, as they are all written in the one language. Furthermore, with Angular using the MVVM architecture it can provide high quality user interfaces. With Node’s unblocking architecture it means that the server-side is ‘quicker’ when compared to other server-side languages. As well as these factors there is a large community of developers making find solutions to problems that were ran into easier and quicker to solve.

Angular.io. 2020. Angular. [online] Available at: <https://angular.io/docs> [Accessed 17 August 2020].

D. Dunka, B., A. Emmanuel, E. and O. Oyerinde, D. (2018). Simplifying Web Application Development Using - Mean Stack Technologies. International Journal of Latest Research in Engineering and Technology (IJLRET), 04(01), pp.62–76.

Dickey, J. (2015). Write modern web apps with the MEAN stack: Mongo, Express, AngularJS, and Node.js. San Francisco, CA: Peachpit Press

Docs.microsoft.com. 2012. The MVVM Pattern. [online] Available at: <https://docs.microsoft.com/en-us/previous-versions/msp-n-p/hh848246(v=pandp.10)?redirectedfrom=MSDN> [Accessed 17 August 2020].

Elrom, E. (2016). AngularJS SEO. Pro MEAN Stack Development, 197-219. doi:10.1007/978-1-4842-2044-3\_8

Gibb, R., 2016. What Is A Web Application? | How A Web Application Works. [online]

Articles for Developers Building High Performance Systems. Available at: <https://blog.stackpath.com/web-application/> [Accessed 14 August 2020].

IBM Cloud Education, 2019. Mean-Stack-Explained. [online] Ibm.com. Available at: <https://www.ibm.com/uk-en/cloud/learn/mean-stack-explained> [Accessed 14 August 2020].

Joshi, B., n.d. Beginning Database Programming Using ASP.NET Core 3.

Lotfy, M. and Pyatt, K. (2008). The mean stack web application development platform: tutorial presentation. Journal of Computing Sciences in Colleges, [online] Volume 34(Issue 2). Available at: <https://dl.acm.org/doi/10.5555/3282588.3282602> [Accessed 14 Aug. 2020].

Nodejs. [2019]. Introduction to Nodejs. [Online]. [17 August 2020]. Available from: http://www.nodejs.dev/learn

Perrenourd, M. (2015). Learning web development with the MEAN stack

Sequelize.org. 2020. Manual | Sequelize. [online] Available at: <https://sequelize.org/master/manual/model-basics.html> [Accessed 17 August 2020].