WEB701 FINAL PROJECT DEVELOPMENT

Hamish Drogemuller 13515109

Contents

Introduction

This document aims to investigate in detail the implementation of a Svelte based web development application for the fictitious Quill Training Charity.

Database:

Utilising a database was essential for the creation of a working application for Quill Training Charity. In this project we utilise the database to store information on users, products and orders. The way the database was utilised within this application was through the .env file that stores our sensitive credentials that are then called in our db.js file in the backend of our system and then returned using axios throughout the application.

This is the same approach that would be utilised if we were to use a MERN or MEAN stack as this is a pretty standard procedure for backend database applications.

.env

```
PORT=5001
MONGO_URI=mongodb://localhost:27017/QuillTraining
```

backend > JS db.js > ...

```
//Utilise .env to secure our sensitive connection data
require("dotenv").config();
//Using mongoose
const mongoose = require("mongoose");

//Function utilising a try catch to connect to database
const connectDB = async () => {
    try {
        mongoose.connect(process.env.MONGO_URI, {
            useNewUrlParser: true
        });
        console.log("MongoDB connect SUCCESS");
    } catch (error) {
        console.log("MongoDB connect FAIL");
        process.exit();
    }
};
//Export connect function
module.exports = connectDB;
```

```
const fetchData = async () => {
  try {
    const config = {
      headers: { "Access-Control-Allow-Origin": "*" },
    };
    const res = await axios.get(
      "http://localhost:5001/product/products",
      config
    );
    products = res.data.data;
    console.log(products);
  } catch (error) {
    console.log(error);
  }
};
```

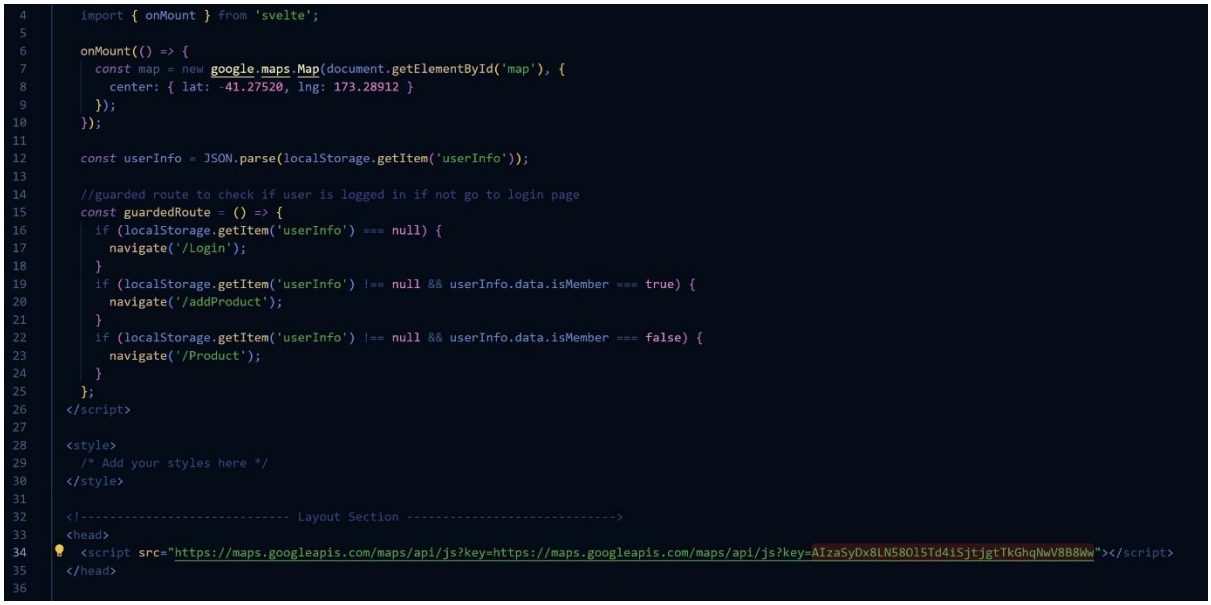
User Controls:

User controls are essential in providing a good experience for website users. To this end I wanted to ensure that the navigation and control for a user was as intuitive as possible. This was achieved with a simplified navigation bar and putting user controls directly under a username when said user is logged in.

Map:

To assist in users choosing training items that are in a similar locale to themselves it is intended to integrate google maps into the product cards. This is achieved through the utilisation of a google maps api. Utilising the google api can assist in a wide variety of ways, the main benefit we wish to focus on for Quill Training is Geocoding[1].

This can be achieved by utilising the onMount method provided by svelte to target a map element within the page. This combined with a src script containing a the google API with a personal token allows us to populate a map with longitude and latitude coordinates.



```
import { onMount } from 'svelte';

onMount(() => {
  const map = new google.maps.Map(document.getElementById('map'), {
    center: { lat: -41.27520, lng: 173.28912 }
  });
});

const userInfo = JSON.parse(localStorage.getItem('userInfo'));

//guarded route to check if user is logged in if not go to login page
const guardedRoute = () => {
  if (localStorage.getItem('userInfo') === null) {
    navigate('/Login');
  }
  if (localStorage.getItem('userInfo') !== null && userInfo.data.isMember === true) {
    navigate('/addProduct');
  }
  if (localStorage.getItem('userInfo') !== null && userInfo.data.isMember === false) {
    navigate('/Product');
  }
};
</script>

<style>
  /* Add your styles here */
</style>

<!---------------------------- Layout Section ---------------------------->
<head>
  <script src="https://maps.googleapis.com/maps/api/js?key=https://maps.googleapis.com/maps/api/js?key=AIzaSyDx8LN58Ol5Td4iSjtjgtTkGhqNwV8B8Ww"></script>
</head>
```

After extensive testing this approach was unable to be implemented so we have moved to attempting to implement a single map that flags each product depending on what location a member puts in the newly added location field when creating a product.

If we wanted to implement this in another framework such as angular we could use the below code instead:

```
const mapContainerStyle = {

  width: '100%',

  height: '200px',

};


const mapUrl = `https://www.google.com/maps/embed/v1/place?key=”Your_API_Key”=Nelson, New Zealand

)}`;


return (

  <div className="card" style={advertisementStyle} onClick={toggleExpansion}>
```

```
    <img src={image} className="card-img-top" alt={title} />
    <div className="card-body">
     <h5 className="card-title">{title}</h5>
     {expanded && (
      <div>
       <p>{description}</p>
       <div style={mapContainerStyle}>
        <iframe
         title="Google Map"
         width="100%"
         height="100%"
         style={{ border: 0 }}
         src={mapUrl}
         allowFullScreen
        ></iframe>
       </div>
       <p>Amount: {amount}</p>
      </div>
     )}
    </div>
   </div>
  );
}
```

The Google Maps API is well developed and documented, making it incredibly easy to research and implement into an application.

The issues I encountered when trying to implement the google maps API were around implementing the map within a svelte component and an inability to locate a suitable fix within the svelte forums.

It was intended to apply google maps to the products themselves with a primary map that marks all trainings for easier viewing. A wide range of supporting libraries can be found on the googlemaps github, these range from info bubbles to markers to adding canvaslayers[(2)].

Final Notes:

Overall this project personally felt disappointing as I struggled to implement the google maps into the frontend. I feel that going with the React Frontend would have been a better decision for the project as previous experience would have greatly assisted in developing an application for Quill Training Charity.

References:

1. Ancoris. (2021, March 3). Do more with Google Maps APIs. Ancoris. https://www.ancoris.com/blog/do-more-with-google-maps-apis
2. Google Maps. (n.d.). Libraries | Google Maps JavaScript API | Google Developers. Retrieved June 22, 2023, from https://googlemaps.github.io/libraries.html