Community Events Map

SUMMARY

The aim of this project is to create an interactive map application that will allow users to view and create events in their local community in order to encourage community participation.

ASSUMPTIONS

The Background information states the following:

"*Community Events Ltd would like you to create an Interactive Community Events Map that enables members to create and view upcoming community events.*"

I assume "*members*" means that the user *must* be a member in order to view and interact with the map. With this in mind, I will be adding in login functionality as well as register functionality so that users can become a *member*. I will also be adding protected routes so that only members can access the Map page.

Here, I will list my assumptions about the project based on the specifications/overview provided.

1. *"Provide basic user controls to navigate the interactive map."*

I assume this will refer to controls that will allow the user to zoom in and out of the map as well as move across the map freely via click and drag, as an example.

2. *" Enable users to create a Community event which will be stored and persist between sessions. New events will behave as all other events."*

I assume this to mean that the session state must persist and that any new events will behave the same as existing events.

3. Allow users to view full details of an event marked on the map through an interaction e.g.: click or hover.

I assume the full details of the event will include a title, description of what the event entails, the date/time of event, and the location. I also assume that when an event is clicked/hovered that this will bring up a window that lists all the above information.

As there is no mention of deleting or updating an event, I assume I will not need to add delete and update functionality. I will only need a get and post http request.

I assume I will only need to provide design documentation such as my proposed architecture, wireframe of the user interface, use case diagram and an entity relationship diagram for my database schema.

As there is no mention of deployment, I assume the testing will be the final phase of the project.

TEXTUAL DESIGN OVERVIEW

Based upon the requirements and my assumptions above, this is how the website will be laid out and this is how it will function. All the design diagrams and wireframes can be found in the *Design Documentation* folder.

Upon entering the website, the user will be introduced to a landing page that will have information about the interactive map along with two buttons:

- Login Button - This will redirect existing users to the login page where they'll be asked for their username and password

- Register Button - This will redirect new users to the register page where they can become a member by creating a unique username and entering their email and a password.

Once a user is logged in, they'll then be redirected to the dashboard page and the header will show a 'Logout' button. The dashboard will display the map and there will be a table underneath displaying the events along with the date, and username of who created the event.

To add an event, the user will double click the location on the map and an info window will appear with a form to create an event, once the event is created, this is shown on

the map and will also be added to the table below. This event will also persist between sessions.

There will be a filter located within the table, this will allow the user to filter events by date and the type of event they're looking for, this will then remove any markers from the map that don't match the filtered criteria.

Should the user log out, this will then redirect to the landing page where the 'Login' and 'Register' button will reappear in the header.

APPROACH

As I'm aware of the 40 hour time constraint, everything stated within this approach is mindful of this and will assist in speeding up the process of creating the app.

This is the architecture for the app based on the requirements provided and my assumptions above:

Client:

- ***React*** - I'll also be installing the ***react-router-dom*** package that will assist in creating the URL routes for each page.

- ***Google Maps API*** - Provides the Map to the application as well as all the other available features Google Maps offers such as streetview, info windows etc.

- ***Axios*** - Handles communication between my API endpoints.

- ***Material-UI*** - MUI will assist in building the UI quickly with the readily available assets and components. This also assists in making the app more responsive across multiple devices.

Server:

- ***MongoDB*** - my database.

- ***Mongoose -*** provides object modelling for *MongoDB* via schemas.

- ***Node.js*** - my development server.

- ***Express*** - a framework for Node, this will speed up creating the backend server.

- ***Express-async-handler*** - middleware to handle exceptions inside of my async express routes.

- ***Dotenv*** - handles setting and accessing local environment variables.

- ***Bcrypt*** - hashes user passwords to safely store in the database.

- ***Nodemon*** - a dev dependency that will watch my server for any changes. This saves me from having to manually restart my server on each change.

- ***Colors*** - a dev dependency that allows me to style my console logs. This allows me to easily identify connections and errors when debugging.

Misc:

- ***Postman*** - to check and test my API endpoints.

- ***Git*** - version control system that will allow me to create branches and work on different components in isolation before releasing on to the main branch. This will also help me to roll back to previous versions should an issue arise.

- ***Trello*** - kanban board that will allow me to create tickets to break down the project into manageable tasks and help me keep track of my progress.

PROPOSED TIMELINE

The timeframe for completion is 40 hours, this commences at 8:30am, ***Monday 9th May 2022*** and ends ***Friday 13th May 2022***.

This is a rough timeline as to how I plan to approach my project each day.

Day 1

Hours 1-5

- ☐ Complete design documentation such as:
  - ☐ Wireframes
  - ☐ Entity Relationship Diagram(ERD)
  - ☐ Use Case Diagrams(UCD)
- ☐ Create ticket backlog on kanban board.

Hours 5-8

Create starter files:

Client:

- ☐ Create React App .
- ☐ Remove Boilerplate.
- ☐ Create file structure (components, routes, etc)

Server:

- ☐ Create Database (MongoDB)
- ☐ Initialise node package manager(npm)
- ☐ Create file structure (models, config, routes, controllers, etc)
- ☐ Server setup.
- ☐ Connect database.

Day 2

Hours 8-16

For the duration of this day, I will solely be focusing on completing the server side.

- ☐ Create API routes and test with Postman.
- ☐ Create User and Event models.
- ☐ Create the Register/Login functions.
- ☐ Hash Passwords.
- ☐ Test Register/Login with Postman.
- ☐ Create Event functionality (Add Event, Get All Events).
- ☐ Test Events with Postman.

Day 3

Hours 16-20

For the duration of this day I will focus on creating the front-end layout/styling and integrating the Map from GoogleAPI.

- ☐ Create components (Header, Login, Event Table etc).
- ☐ Flesh out each component with MUI assets (Toolbar, DataGrid, Grid etc).
- ☐ Make each component responsive across mobile and desktop (more devices, if time allows) .
- ☐ Create routes to each page (Dashboard, Login etc).

Hours 20-24

- ☐ Get API key for the map.
- ☐ Add User controls.
- ☐ Add central point to the map.

Day 4
Hours 24-32

This day will be focused on adding functionality to the project.

- ☐ Receive data from the backend via Axios.
- ☐ Add existing events to the map.
- ☐ Create event functionality
- ☐ and send this to the backend/database,
- ☐ Create login/register User functionality and send it to the backend/database.
- ☐ Filter events.
- ☐ Display events in a table.

Day 5
Hours 32-40

This day will be focused on the following:

- ☐ Testing
- ☐ User Guide
- ☐ Documentation (Implementation, Improvements, Retrospective)

CONSTRAINTS

The timeline above is very much the desirable outcome with the 40 hours given.

40 hours is quite a tight deadline even if things were going smoothly without bugs due to my experience mainly being frontend. This is why I will be focusing on backend prior to frontend to hopefully gain an advantage.

If any bugs or issues are to arise, then this will impact the time I have to complete the

project which will result in bugs and/or unfinished components. To combat this, I’m going to be debugging and testing as I go along to reduce the impact.