# UYR Events

Contents

[UYR Events 1](#_Toc179384687)

[Aim of the Project 1](#_Toc179384688)

[Analysis and Design 1](#_Toc179384689)

[Implementation 4](#_Toc179384690)

[Using Test Data 6](#_Toc179384691)

[Displaying Data 7](#_Toc179384692)

# Aim of the Project

The aim of this project is to create an events page for my company. This task was given to me by my technical director and CEO, with the hopes of expanding into a different market and increasing sales. This will be a full stack application that will allow users to create and organise their events. Along with this, we will display our products that can then be purchased for the events. The website will require login information and a basket for checking out. For this project, I will be using a React frontend, including JavaScript and HTML, and a C# backend API. I will also be using the Bootstrap framework for styling purposes.

# Analysis and Design

First, I created wireframes for my design. The designs had to be clear and intuitive to enhance the user experience. Below are some examples of the wireframes I created:

A screenshot of a computer

Description automatically generated

Figure : Events Page

This will be the page that displays all the events a user has. Users will quickly be able to see information as each event card, seen in the middle, will show the event title, hosts, date, time, location, how many are invited and how many have accepted invitation. At the bottom of the card will be the buttons to edit the information of an event or to delete the event. Above the events is button to add a new event, which will trigger a modal to fill in the new information. To the left of the page will be the filters, which will help users find information quickly, leading to a positive user experience. To the right will be suggested products that when clicked will take you to that product page. This will hopefully improve conversion rates.

A screenshot of a computer

Description automatically generated

Figure : Product Page

Above is the product page where a user will be able to choose the type, size and quantity of a product, along with their associated prices. On the left, users will be able to review a preview of the product before pressing the ‘Add to Basket’ button.

The header of all my pages will be a navbar, where users can swiftly navigate through each section of the website. Within the navbar will be buttons to navigate to the basket and the login page for a user. Also, for pages that it applies to, there will be breadcrumbs that indicate the page’s location within its navigational hierarchy, resulting in a greater user experience. Here I show a footer section which may show support information such as contact numbers and email addresses.

A screenshot of a computer

Description automatically generated

Figure 3: Basket

Lastly, here I am showing the users basket with all the products they have added and their associated costs. At the bottom will be the subtotal, delivery costs and final total of all goods in the basket. On the right-hand side is where the user will add their address and finally checkout to pay. The basket will need global state as users may be adding products to it from multiple different pages.

I plan to do this project with the support of my line manager and fellow colleagues who have more experience in aspects such as databases and C# backend API’s.

# Implementation

The first thing I implemented into my application was the routing paths that I will require. This gave me a basic idea of how users would navigate through my multi page app. I did this by importing in the JavaScript framework, React Router.

A screen shot of a computer program

Description automatically generated

Figure : Use of React Router

In figure 4 you can see how I have separated out each individual page, resulting in cleaner, more organised code. I have also determined the URL of each page with a relevantly labelled path.

On lines 23 and 34, you can see I have created a Header and Footer component. These have been purposely placed there, so that they are persistent throughout my website.

A screenshot of a computer

Description automatically generated

Figure 5: Header & Footer Design

In figure 5 above, you can see the design I have for the Header and Footer components. The layout of the elements, has taken inspiration from one of our existing sites for brand consistency.

## Using Test Data

In order to progress and test my application I realised I would need some test data. To do this I decided on using JSON Server, which would allow me to create a mock REST API for my application.

A screen shot of a computer

Description automatically generated

Figure 6: JSON Server Data

In figure 6 you can see the data that I have inputted to create some events. I have also created properties for messages, users, products and a basket, for me to use as endpoints.

The live data can be seen below in figure 7, whereby I am previewing the data on the events endpoint.

A screenshot of a computer program

Description automatically generated

Figure 7: JSON Data from Events Endpoint

## Displaying Data

My next task was trying to display the data on my webpage. I decided on using Axios, a promised-based HTTP Client, to make my API calls.

I made a utils.js folder to separate out my API calls, allowing for reusability, and added my GET request. Seen below (figure 8). This makes a call to my events endpoint and receives a promise which is then returned.

A screen shot of a computer

Description automatically generated

Figure 8: Axios GET Request

In order to display the returned data, I used the React Hook, useEffect (see figure 9). This runs on first render and triggers my API call. Once this runs, I used the React Hook, useState, to set my events data to the returned value.

A screen shot of a computer program

Description automatically generated

Figure 9: useEffect and useState Hook

Finally, in the body of my Events component, I can now map over the data and provide it as props to my EventCard component for formatting. The result can be seen below in figure 10.

A screenshot of a computer

Description automatically generated

Figure : Displaying my Events Data

Using Bootstrap, I have styled each event within a card, applying suitable padding and margin between each element. I have also created a button group which have the edit and delete buttons. The layout has been set using grid and changes dynamically to smaller or larger screen sizes.