**A**

**Internship-II Report**

**On**

**“ReactJs“**

**Submitted to**

**V. P. M. P. POLYTECHNIC, GANDHINAGAR**

**DEPARTMENT OF COMPUTER ENGINEERING**

****

**Mentor's Name: Prepared By:  
Darshana mam Kruteen Bharadia(226540307064)  
 Jaimil Modi(226540307080)  
 Kartik Suther(226540307188)**

**V.P.M.P. POLYTECHNIC**

**DEPARTMENT OF COMPUTER ENGINEERING**

****

***CERTIFICATE***

**This is to certify that Mr. / Miss \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ of \_\_\_\_\_\_\_\_\_\_\_class, Enrollment No. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ has satisfactory completed his / her term work in INTERNSHIP PROJECT-II (4350704) for the term ending in \_\_\_\_\_\_\_\_\_\_\_\_\_\_2024.**

**Date:**

**Sign of Internal Guide Sign of H.O.D**

**INDEX**

|  |  |
| --- | --- |
| **SR.NO** | **TOPICS** |
| **1** | Internship Project Registration Form |
| **2** | Joining Letter |
| **3** | 15 DAYS WORK SHEET REPORT-1 |
| **4** | 15 DAYS WORK SHEET REPORT-2 |
| **5** | 15 DAYS WORK SHEET REPORT-3 |
| **6** | STUDENT ATTENDANCE SHEET REPORT |
| **7** | Letter for Completion |
| **8** | Report |

**Internship Project Registration Form:**

## **Joining Letter:**

## **15 DAYS WORK SHEET REPORT-1**

## **15 DAYS WORK SHEET REPORT-2**

**15 DAYS WORK SHEET REPORT-3**

## **STUDENT ATTENDANCE SHEET REPORT**

## **Letter for Completion**

## **REPORT**

## Introduction

ReactJS is a popular JavaScript library for building user interfaces, particularly single-page applications where reactivity and performance are crucial. Developed and maintained by Facebook, React is widely used in both web and mobile app development. This report delves into the core concepts, architecture, advantages, and use cases of ReactJS, providing a thorough understanding of why it has become a cornerstone in modern web development.

We made a travel webpage which lets you book tickets for bus, train, flight, and hotel. It lets you decide from a selection of places or companies that are available on the webpage. After selecting your preferred choice you are directed to a form where you have to fill in details about the travel dates and number of passengers, the start location from where you will begin your journey.

We have used mainly used react and css to make this webpage. React to put functionality into the webpage and css to edit the webpage and put styles into it. There are multiple components that have different functionality and do different things. Most of the code uses class components with importing external css.

Installation

## Getting Started with React: Installation and Environment Setup

Before embarking on your React journey, ensure you have a suitable development environment set up. Here’s a roadmap to get you started:

1. **Node.js and npm:** Download and install Node.js (<https://nodejs.org/en/download/package-manager/current>) as it provides the runtime environment for JavaScript code execution. npm (Node Package Manager) is bundled with Node.js and is used to manage project dependencies.
2. **Create React App (CRA):** Using Create React App (CRA) to establish a new React project with minimal configuration. Run**npx create-react-app my-react-app**in your terminal, replacing my-react-app with your desired project name.

## React requires knowledge in three areas:

**HTML:**

**HTML**is the language of the web, used by **billions of websites** to create the pages you see every day. Want to **learn HTML** from **scratch**and make your web pages? This tutorial is for you!

In this **HTML tutorial**, you will learn how to use HTML to create and style web pages. You will start with **HTML fundamentals**, such as **basic HTML tags** and their **attributes**, **classes**, **layout**, and **responsiveness**. Then, you will move on to **advance HTML5** topics, such as **HTML forms**, **media**, **APIs**, and more. **By the end** of this tutorial, you will have a solid foundation in HTML.

**What is HTML?**

**HTML** stands for **HyperText Markup Language.**It is the standard language used to create and design web pages on the internet. It was introduced by **Tim Berners-Lee** in **1991**at **CERN**as a simple markup language. Since then, it has evolved through versions from **HTML 2.0** to **HTML5** (the latest 2024 version).

HTML is a **combination of Hypertext and Markup language**. Hypertext defines the link between the web pages and Markup language defines the text document within the tag.

**CSS:**

CSS, or Cascading Style Sheets, is the language used to style and enhance HTML documents. It defines the presentation of HTML elements on a web page, enabling changes to fonts, colors, sizes, spacing, column layouts, and animations.

In this **CSS tutorial**, we will cover both **basic and advanced concepts**, including CSS properties, selectors, functions, media queries, and more.

## **What is CSS?**

CSS, or **Cascading Style Sheets**, is a language used to style and enhance websites. It controls how **HTML elements**—such as text, images, and buttons—are displayed on a webpage. **With CSS**, you can adjust font sizes and colors, add backgrounds, and manage the layout, transforming a basic webpage into a visually appealing and user-friendly experience. CSS also simplifies **layout management** across multiple web pages by **using external stylesheets** stored in CSS files.

## Different Ways to Use CSS

CSS has three ways to style the HTML:

* Inline: Add styles directly to HTML elements using the style attribute (limited use).
* [Internal](https://www.geeksforgeeks.org/internal-css): Place styles within a <style> tag inside the HTML file, usually within the <head> section
* [External](https://www.geeksforgeeks.org/external-css): Create a separate CSS file with a .css extension and link it to your HTML file using the <link> tag.

**JavaScript:**

**JavaScript** is the most powerful and versatile web programming language. It is used for making the websites interactive. JavaScript helps us add features like animations, interactive forms and dynamic content to web pages.

In this **JavaScript Tutorial**, we’ll learn all the **basics to advanced**topics and concepts of **JavaScript**. This JavaScript tutorial includes **operators, control flow, functions, objects, Asynchronous JavaScript, Events, DOM manipulation, OOPs, Closures, Event loops etc.**

## **What is JavaScript?**

JavaScript is a **programming language used for creating dynamic content on websites**. It is a **lightweight**, **cross-platform** and **single-threaded** programming language. JavaScript is an **interpreted**language that executes code line by line providing more flexibility. It is a commonly used programming language to**create dynamic and interactive elements in web applications**. It is easy to learn.

## **Why Learn React JS?**

## **React, the popular JavaScript library, offers several exciting reasons for developers to learn it.**

## **First, React is flexible – once you learn its concepts, you can use it across various platforms to build quality user interfaces. Unlike a framework, React’s library approach allows it to evolve into a remarkable tool.**

## **Second, React has a great developer experience, making it easier to understand and write code. Third, it benefits from Facebook’s support and resources, ensuring regular bug fixes, enhancements, and documentation updates. Additionally, React’s broader community support, excellent performance, and ease of testing make it an ideal choice for web development.**

## **Features of React**

## **JSX (JavaScript Syntax Extension):**

JSX combines HTML and JavaScript, allowing you to embed JavaScript objects within HTML elements. It enhances code readability and simplifies UI development.

Example:

const name = "React";

const ele = <h1>Welcome to {name}</h1>;

1. **Virtual DOM (Document Object Model):**

React uses a virtual DOM, which is an exact copy of the real DOM.When there are modifications in the web application, React updates the virtual DOM first and then computes the differences between the real DOM and the virtual DOM.This approach minimizes unnecessary re-rendering and improves performance.

1. **One-way Data Binding:**

React follows one-way data binding, where data flows from parent components to child components. Child components cannot directly return data to their parent components, but they can communicate with parents to modify states based on provided inputs.

1. **Performance:**

React’s virtual DOM and component-based architecture contribute to better performance. Separate components allow efficient rendering and faster execution

1. **Extension:**

React has a rich ecosystem and supports various extensions. Explore tools like Flux, Redux, and React Native for mobile app development and server-side rendering.

1. **Conditional Statements in JSX:**

JSX allows writing conditional statements directly. Display data in the browser based on provided conditions.

Example:

**const age = 12;**

**if (age >= 10)**

**{ return <p>Greater than {age}</p>;}**

**else {return <p>{age}</p>;}**

## 7**. Components:**

## React divides web pages into reusable and immutable components.

## Component-based development simplifies code organization and maintenance.

## **Core React Concepts:**

Here are some essential concepts to learn:

* **Props:** Components can receive data from parent components through props, enabling you to pass information and customize component behavior.
* **State:** Components can manage their internal state using the useState hook. This state dictates the component’s appearance and behavior, and updates trigger re-renders.
* **Lifecycle Methods:** React provides lifecycle methods like componentDidMount and componentDidUpdate that allow you to perform actions at specific stages of a component’s lifecycle.
* **Conditional Rendering:** Control what gets displayed on the screen based on certain conditions using conditional statements within JSX.

## **React Advantages:**

* **Composable:** We can divide these codes and put them in custom components. Then we can utilize those components and integrate them into one place.
* **Declarative:** In ReactJS, the DOM is declarative. We can make interactive UIs by changing the state of the component and ReactJS takes care of updating the DOM according to it.
* **SEO Friendly:** ReactJS affects the SEO by giving you a SPA (Single Page Application) which requires Javascript to show the content on the page which can be rendered and indexed.
* **Community:** ReactJS has a huge community because of its demand each company wants to work with ReactJS. Companies like Meta, Netflix, etc built on ReactJS.
* **Easy to learn:** HTML-like syntax of JSX makes you comfortable with the codes of React, it only requires a basic knowledge of HTML, CSS, and JS fundamentals to start working with it.
* If you want to learn more refer to this article [React JS Advantages](https://www.geeksforgeeks.org/what-are-the-advantages-of-react-js)
* **Debugging is Easy:** The debugging of ReactJS is unidirectional which means while designing any app using ReactJS the child components are nested within parent components. So, the data flow is in a single direction it gets more easier to debug errors.

## **History of React:**

* React was invented by Facebook developers who found the traditional DOM slow. By implementing a virtual DOM, React addressed this issue and gained popularity rapidly.
* The current stable version of ReactJS is 18.2.0, released on June 14, 2022. The library continues to evolve, introducing new features with each update.

## **How does React work?**

React operates by creating an in-memory virtual DOM rather than directly manipulating the browser’s DOM. It performs necessary manipulations within this virtual representation before applying changes to the actual browser DOM. React is efficient, altering only what requires modification.

## **ReactJS Lifecycle:**

Every React Component has a lifecycle of its own, lifecycle of a component can be defined as the series of methods that are invoked in different stages of the component’s existence. React automatically calls these methods at different points in a component’s life cycle. Understanding these phases helps manage state, perform side effects, and optimize components effectively.

### **1. Initialization**

This is the stage where the component is constructed with the given Props and default state. This is done in the constructor of a Component Class.

### 2. **Mounting Phase**

* **Constructor**: The constructor method initializes the component. It’s where you set up initial state and bind event handlers.
* **render():**This method returns the JSX representation of the component. It’s called during initial rendering and subsequent updates.
* **componentDidMount():** After the component is inserted into the DOM, this method is invoked. Use it for side effects like data fetching or setting timers.

### 3. **Updating Phase**

* **componentDidUpdate(prevProps, prevState)**: Called after the component updates due to new props or state changes. Handle side effects here.
* **shouldComponentUpdate(nextProps, nextState):** Determines if the component should re-render. Optimize performance by customizing this method.
* **render():** Again, the render() method reflects changes in state or props during updates.

### 4. **Unmounting Phase**

* **componentWillUnmount()**: Invoked just before the component is removed from the DOM. Clean up resources (e.g., event listeners, timers).

## **Importing and Exporting React Components:**

We have learned so far that React Apps are a collection of interactive Components, and from the article, on [React Components](https://www.geeksforgeeks.org/reactjs-components/). we know how to create components but even with that knowledge, it will not be sufficient to create a full-fledged React App. To do so we need to know two operations broadly known as **Importing and Exporting**.

We may not have told earlier, but we have been using the import operation in every one of our previous articles when we were importing react and react-dom itself. Similarly, we can also import user-defined classes, components, or even a part of the same. Let us shift our discussion over Importing.

## **Importing a React Component**

The world of JavaScript is always moving and one of the latest ES2015 now provides a more advanced module importing/exporting pattern. In previous engines, the developer had to use the **module.exports = { // Define your exports here. }**, but now with ES2015 every module can have a default export or may export several named parameters, and if it is possible to export it will surely be possible to import the same. Thus, with ES2015 every module may import the default export or several named parameters or even a valid combination. 

React uses the same features as mentioned above, and you may treat each React Component as a module itself. Thus, it is possible to import/export React Components, and is one of the basic operations to be performed. In React we use the keyword **import** and **from** to import a particular module or a named parameter.

Let us now see the **different methods of importing ReactJS components**.

### **Importing default export:**

Every module is said to have at most one default export. In order to import the default export from a file, we can use only the address and use the keyword import before it, or we can give a name to the import making the syntax as the following.

import GIVEN\_NAME from ADDRESS

### **Importing named values:**

Every module can have several named parameters and in order to import one we should use the syntax as follows.

import { PARA\_NAME } from ADDRESS

Similarly, for multiple such imports, we can use a comma to separate two-parameter names within the curly braces.

### **Importing a combination of Default Exports and Named Values:**

The title makes it clear what we need to see is that the syntax is the same. In order to import a combination, we should use the following syntax.

import GIVEN\_NAME, { PARA\_NAME, ... } from ADDRESS

**Note:**When importing we sometimes come across files that have .js extensions like Color.js, when importing we can follow any of the below-given syntaxes:

import Color from './Color.js'  
 // OR  
import Color from './Color'

In the above syntax both the codes work correctly because [ES Modules](https://www.geeksforgeeks.org/es6-modules/) working allows importing in both the ways

## **Exporting** **a React Component**

Now, importing is an operation that requires the permission of the module. Importing is possible only if the module or named property to be imported has been exported in its declaration. In React we use the keyword **export** to export a particular module or a named parameter or a combination.

Let us now see the different ways we can use the export operation in React.

### **Exporting default export:**

We have already learned that every module is said to have at most one default export. In order to export the default export from a file, we need to follow the syntax described below.

export default GIVEN\_NAME

### **Exporting named values:**

Every module can have several named parameters and in order to export one we should use the syntax as follows.

export { PARA\_NAME }

Similarly, for multiple such exports, we can use a comma to separate two-parameter names within the curly braces.

## **Components**

## Components let you split the UI into independent, reusable pieces, and think about each piece in isolation

## Component will be render Root div of index.html file.

## 

## **Types of Component**

## In ReactJS, we have mainly two types of components.

## They are

## 1. Functional Components (Stateless Component)

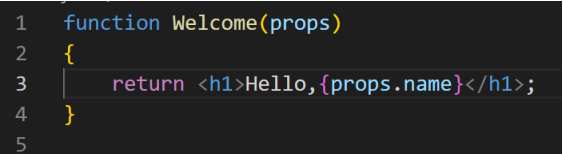
## 2. Class Components (Stateful Component )

## **Functional Components**

## Functional components based on simple or plain JavaScript.

## Functional components can’t maintain their own state that’s why sometimes we can say it stateless components.

## It is just accept the props as an argument and return the react elements.



## Functional because they are basically functions

## Stateless because they do not hold and/or manage state

## Presentational because all they do is output UI elements

## **Class Component**

## A class component requires you to extend from React.Component and create a render function which returns a React element.

## All lifecycle hooks are coming from the React.Component which you extend from in class components.

## So if you need lifecycle hooks you should probably use a class component.

## 

## Class because they are basically classes

## Smart because they can contain logic

## Stateful because they can hold and/or manage local state

## Container because they usually hold/contain numerous other (mostly functional) components

## **LifeCycle Method**

## **Component LifeCycle**

## **>componentDidMount**

## Fired after the component mounted (Called When component will open)

## **>componentWillUnmount**

## Fired before the component will unmount (Called before closing component ) >**getDerivedStateFromProps**

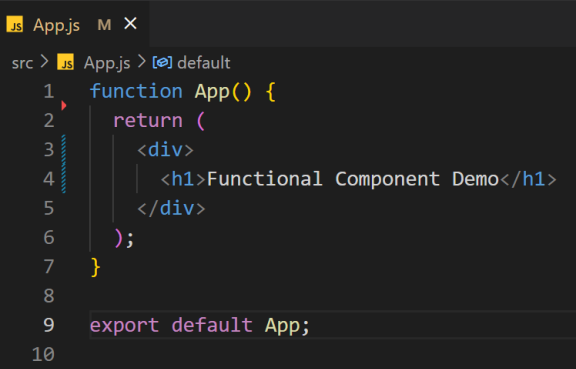
## Fired when the component mounts and whenever the props change. Used to update the state of a component when its props change

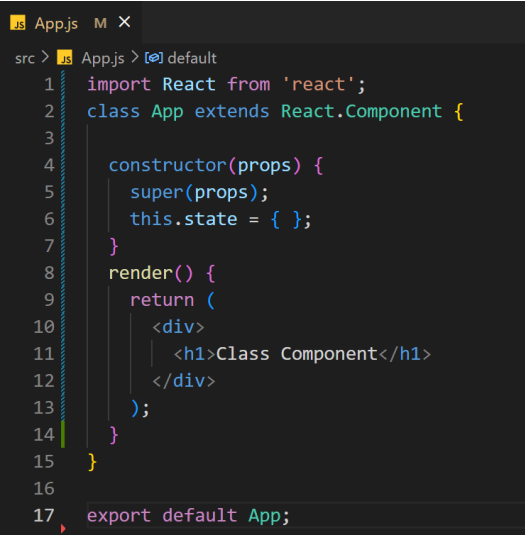
## **>Events**

## onClick

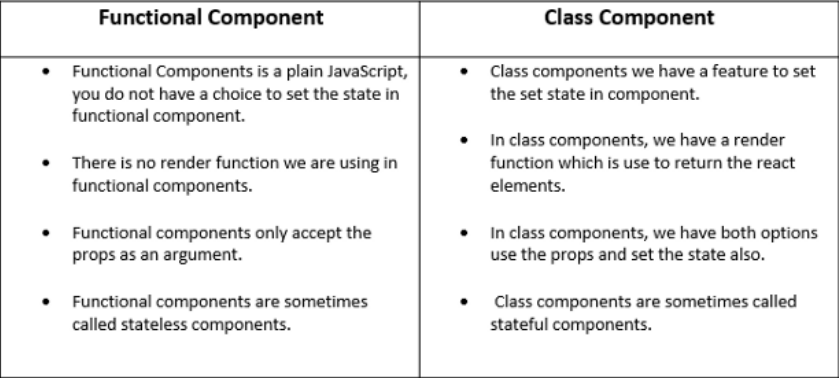
## onSubmit

## onChange

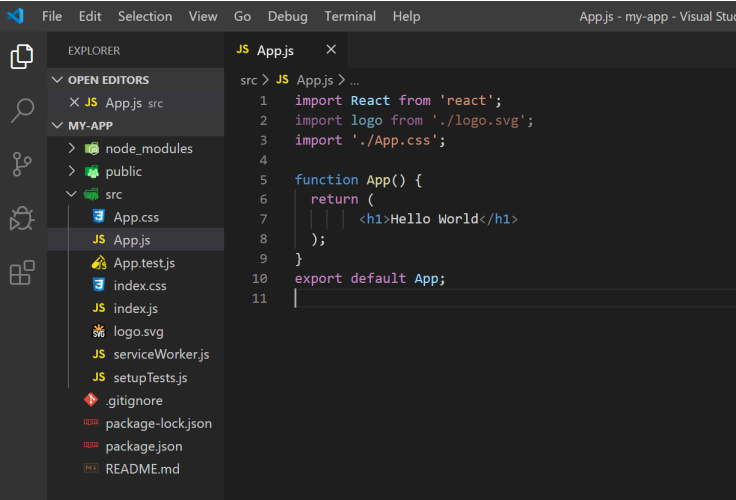
**Functional Component Vs Class Component** 

****

**Functional Vs Class**

****

**Change Content of App Component**

****

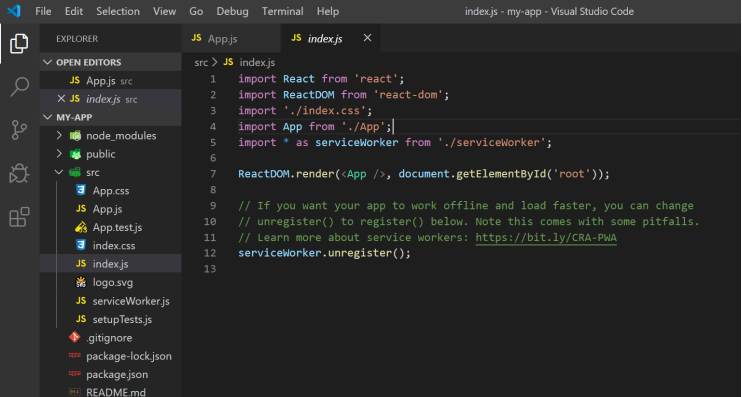
**Render App Component**

We need to Call React-Dom Package to Render App Component in App.

Import App Component and Render in Root ID.

To call Component use

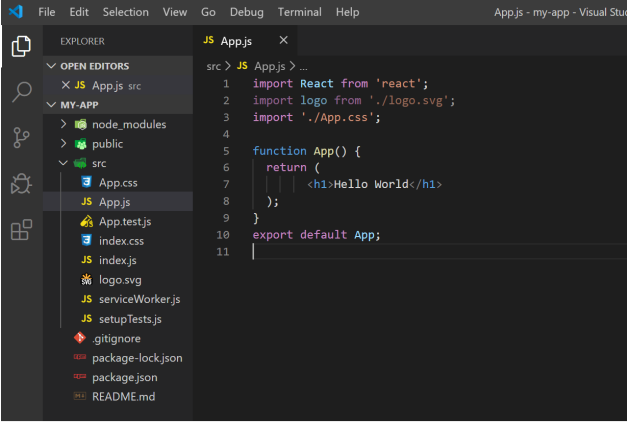
<App/>

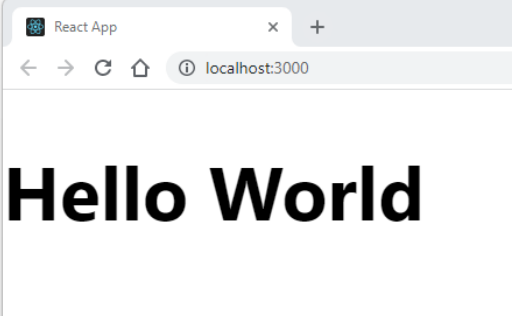
****

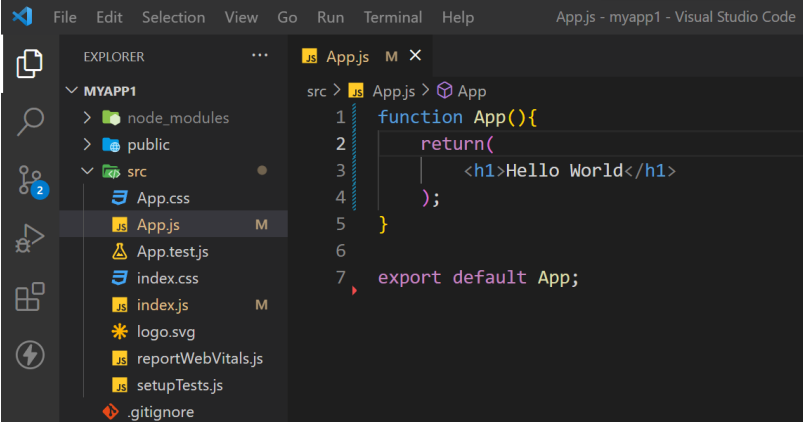
App.JS Component Will Import in Index.JS

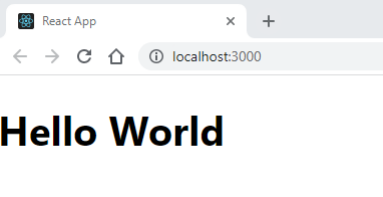
In Index JS we will render App Component in ROOT id.

Index.html will render (Print ) Output

****

****

**Simple Function Component** 

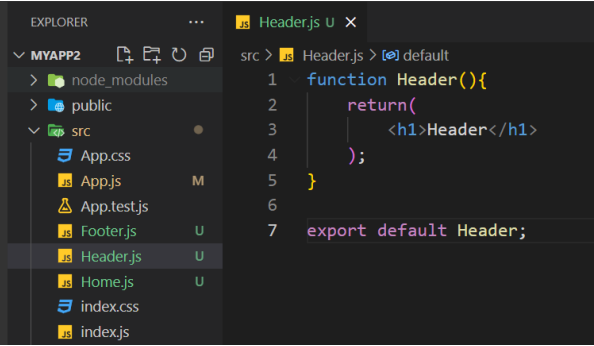


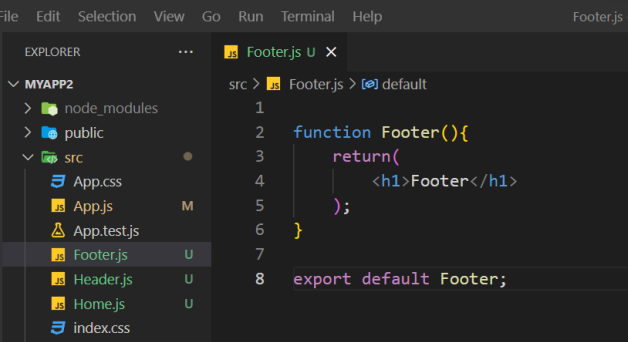
**Reuse Component**

**Create 2 Component**

Header

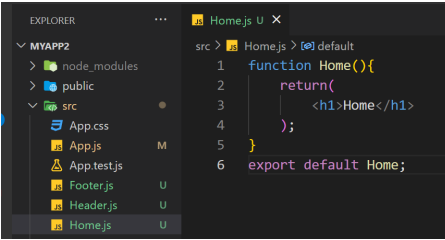
Footer

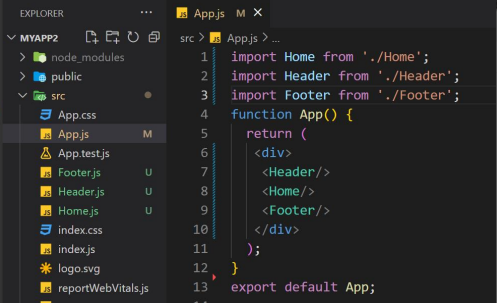


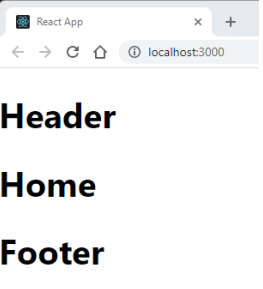


**Create Home Component and Load in App**

We can Create Home Component and then we can load same in App.js Component file.







**Basic Component**

**Imports**

Import React from 'react’;

For Loading React Methods

**Import ReactDOM from 'react-dom’;**

For Loading and Render Data into React DOM.

Using the ES6 syntax with extends React.Component, which extends the Component class

**React Fundamental**

**JSX**

Allows us to write HTML like syntax which gets transformed to lightweightJavaScript objects.

**Virtual DOM**

A JavaScript representation of the actual DOM.

**React.Component**

The way in which you create a new component.

**render (method)**

Describes what the UI will look like for the particular component.

**ReactDOM.render**

Renders a React component to a DOM node.

**state**

The internal data store (object) of a component

**constructor (this.state)**

The way in which you establish the initial state of a component.

**setState**

A helper method used for updating the state of a component and re-rendering the UI

**props**

The data which is passed to the child component from the parent component.

**propTypes**

Allows you to control the presence, or types of certain props passed to the child component.

**defaultProps**

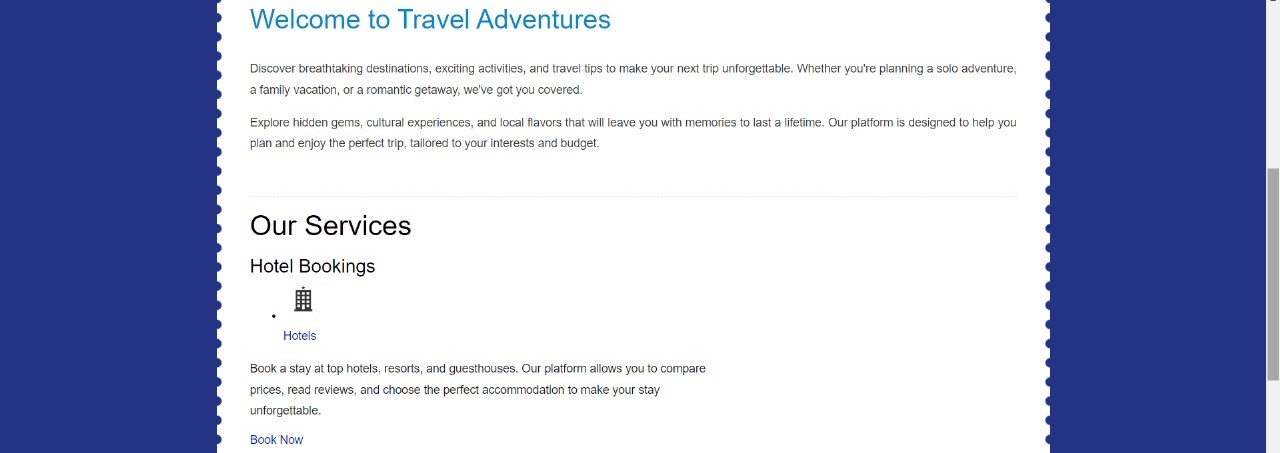
Allows you to set default props for your component.

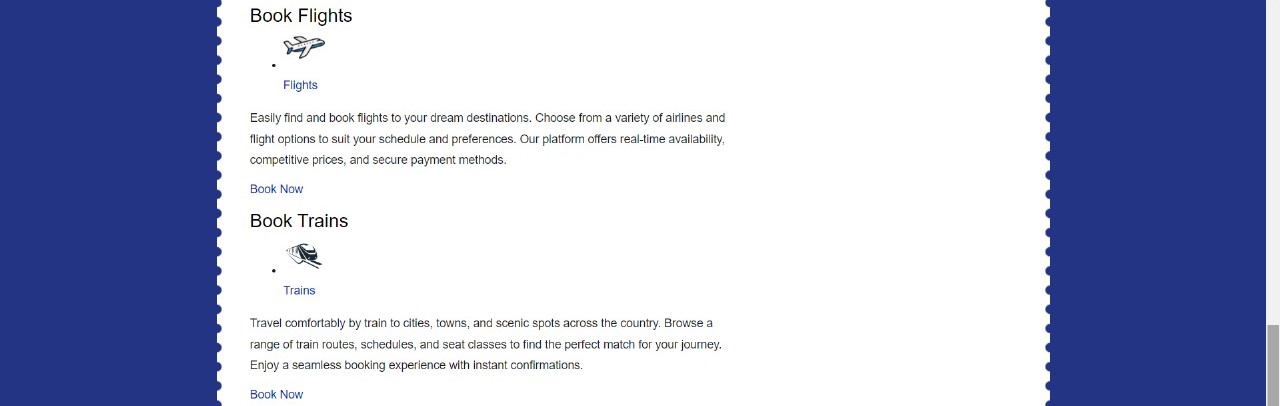
Project

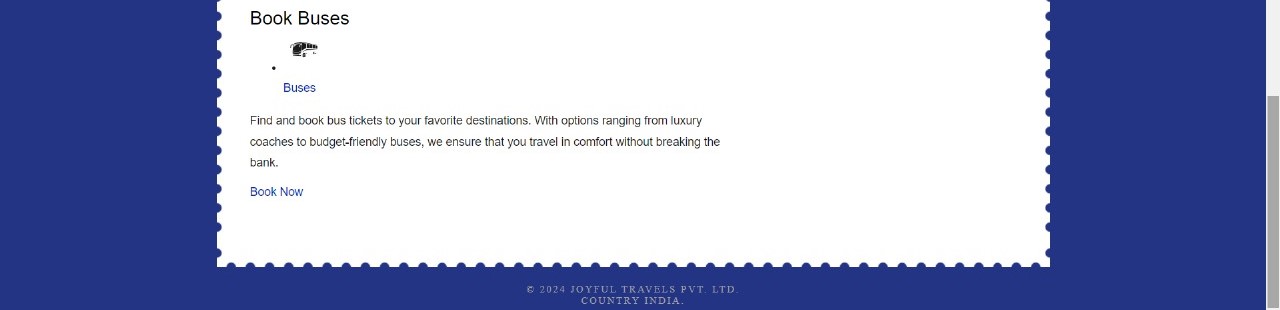
**Project photos**

**Homepage**



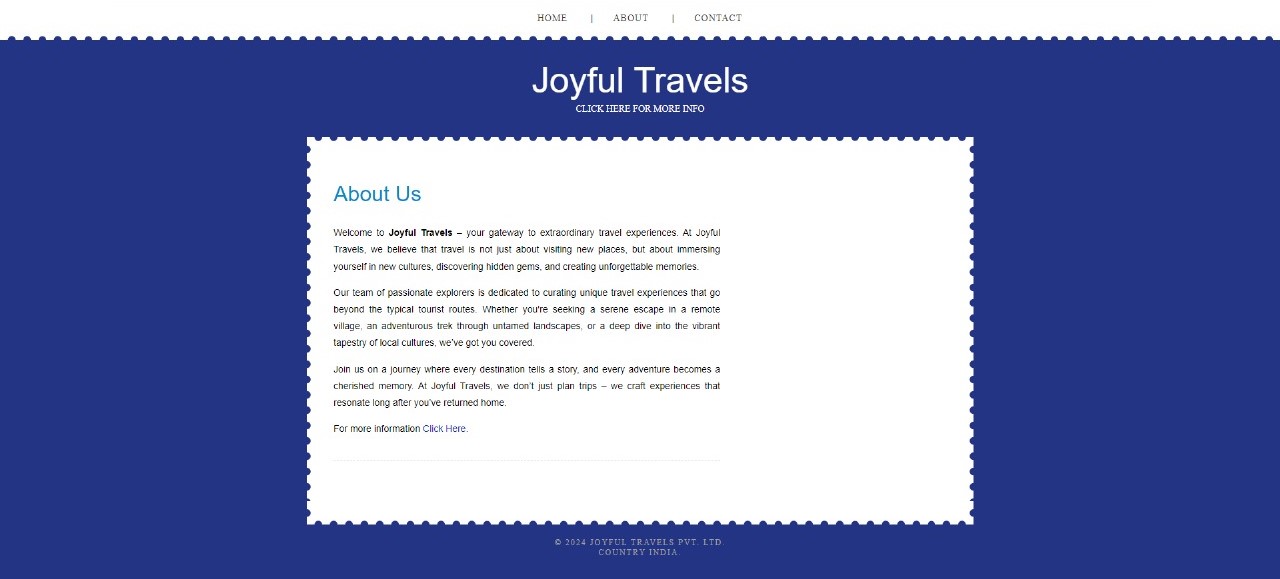






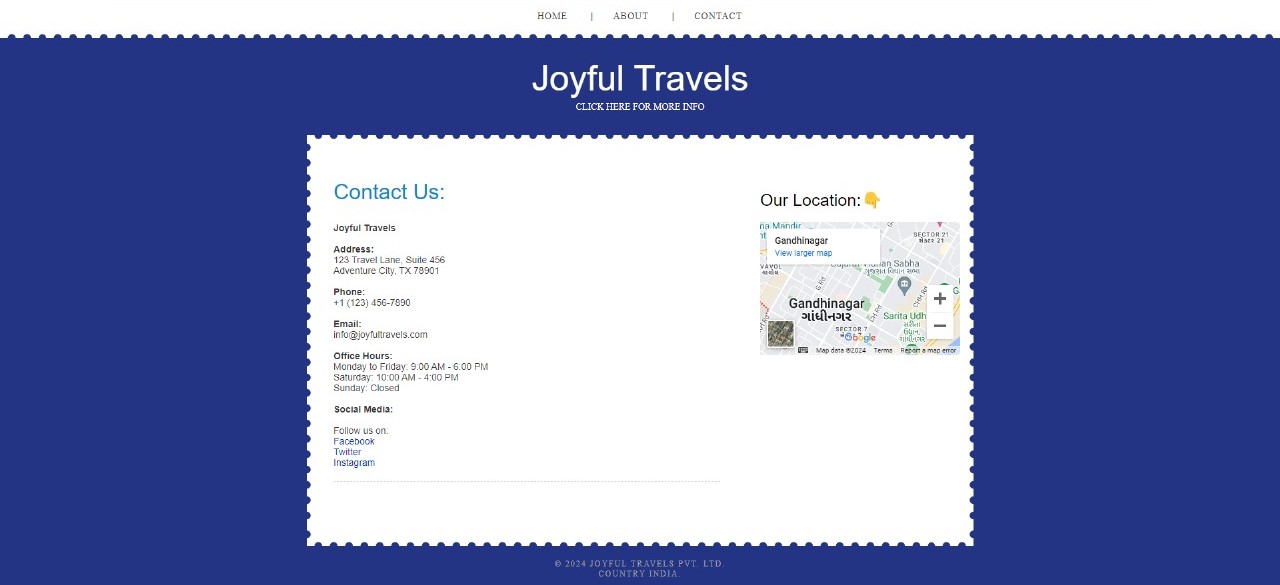
The above pictures show our websites home page from where you can access services such as hotel, flights, trains, and bus booking as well as know more about the company and contact

**About us page**



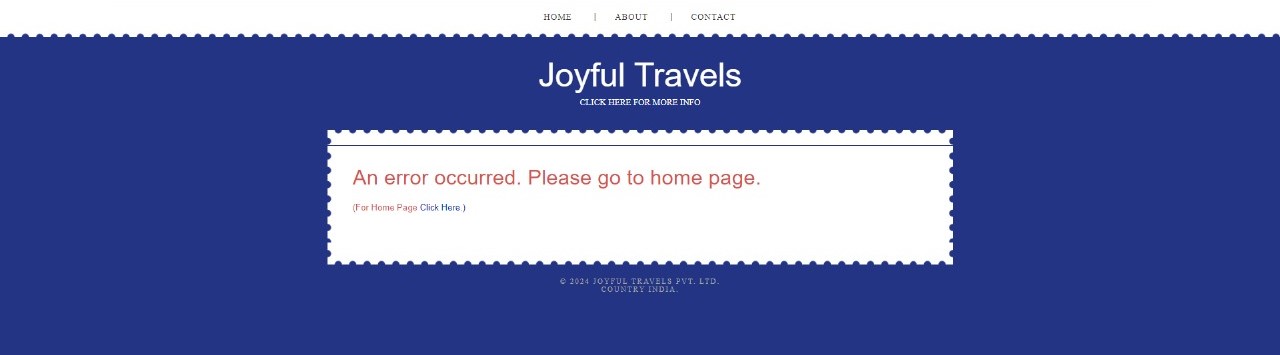
This page shows the details and info about the company

**Contact page**



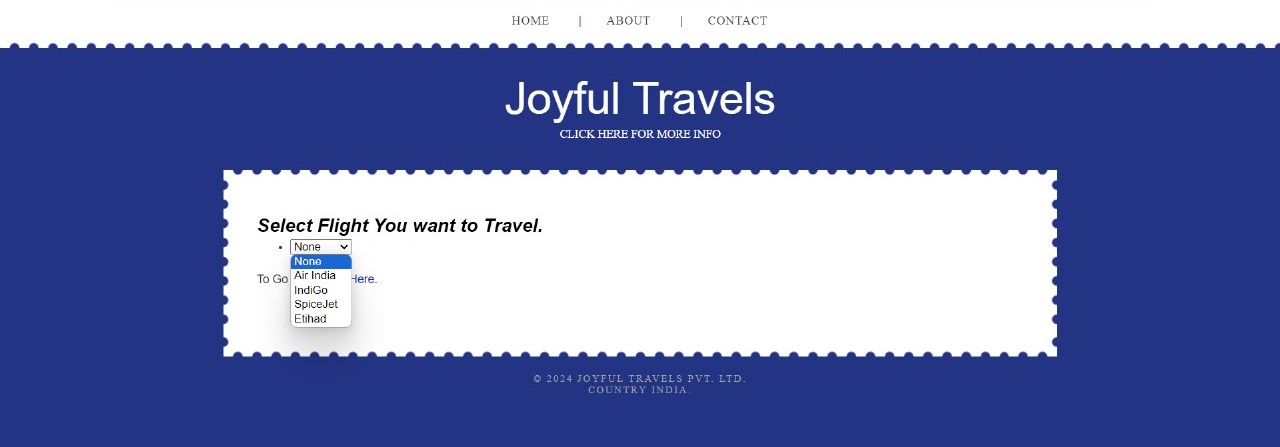
This page shows the contact, address, location of the company

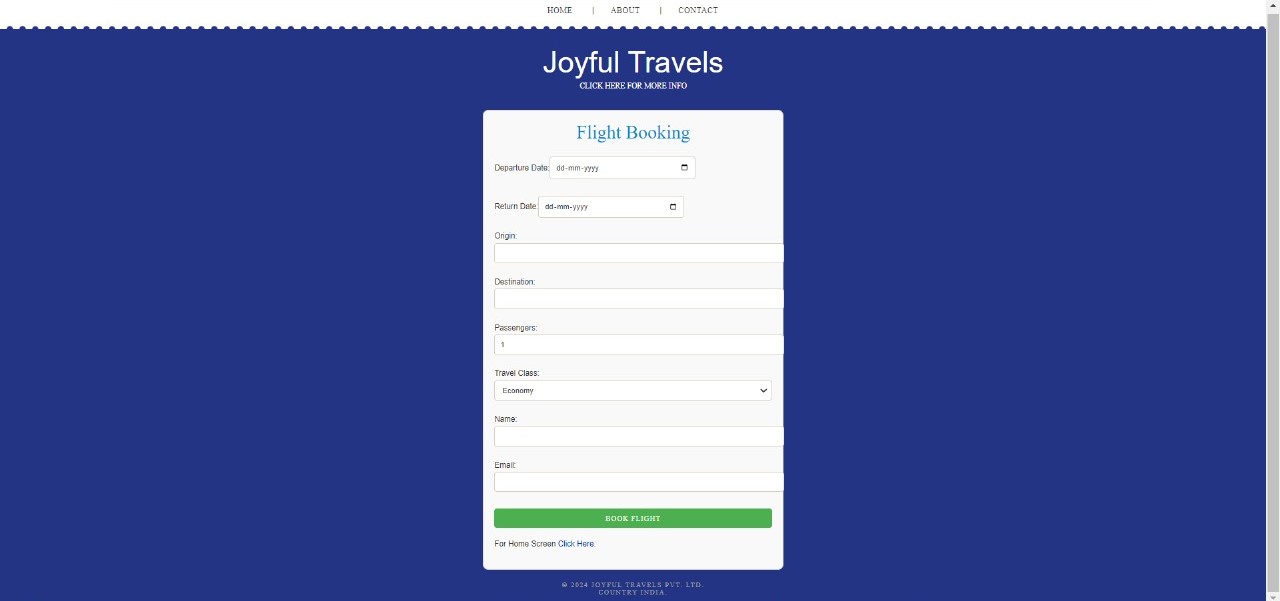
**Error page**



This page appears when there some sort of error while exploring the website

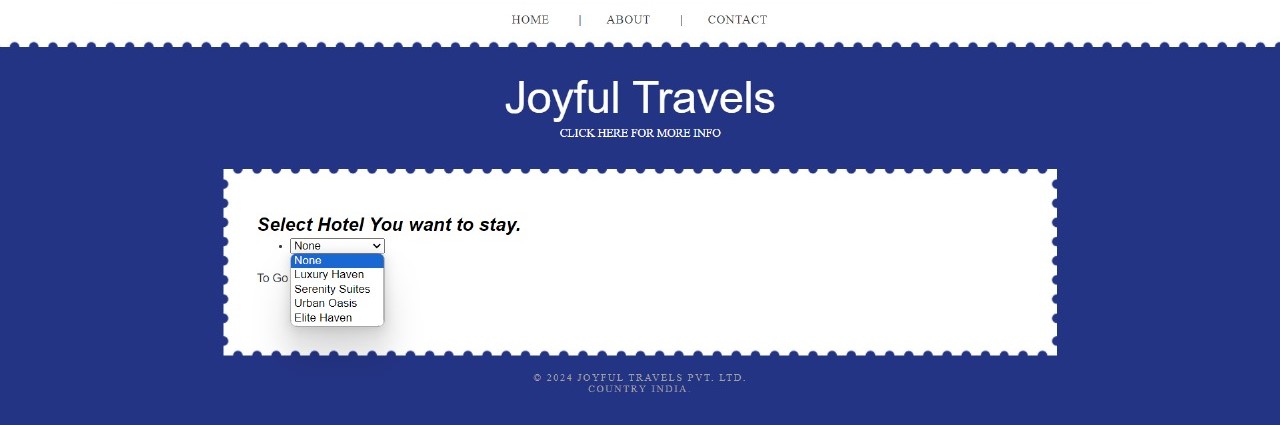
**Flight Booking**

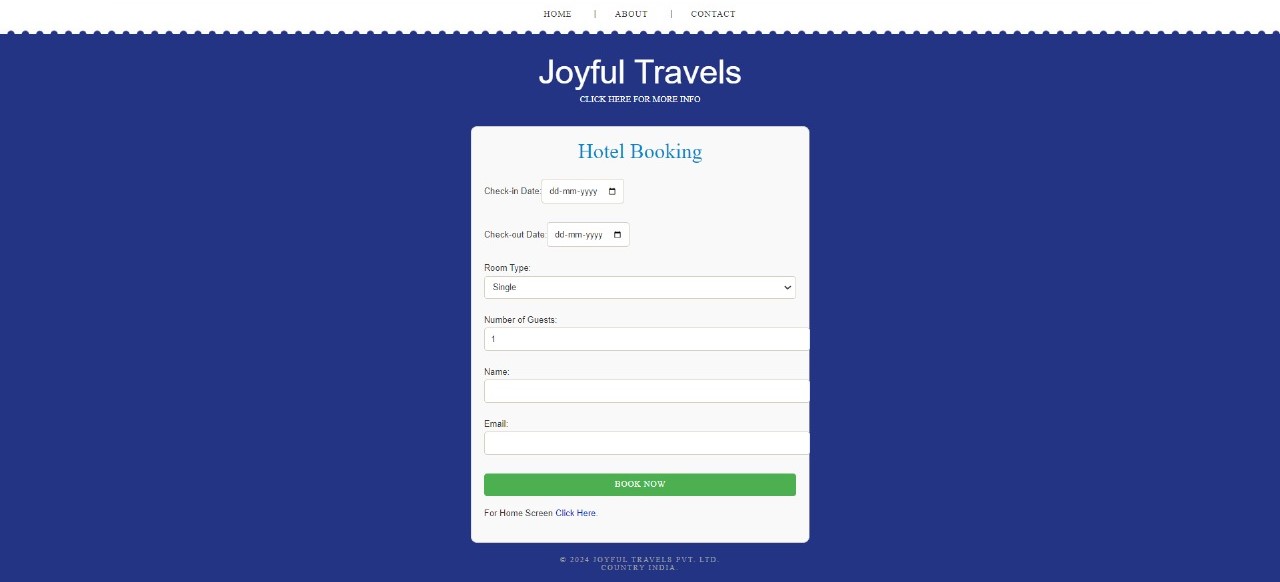




The above pages show the process of flight booking. The website provides a select flight companies and then lets you choose details such as departure date, return date, to and from, number of passengers and more.

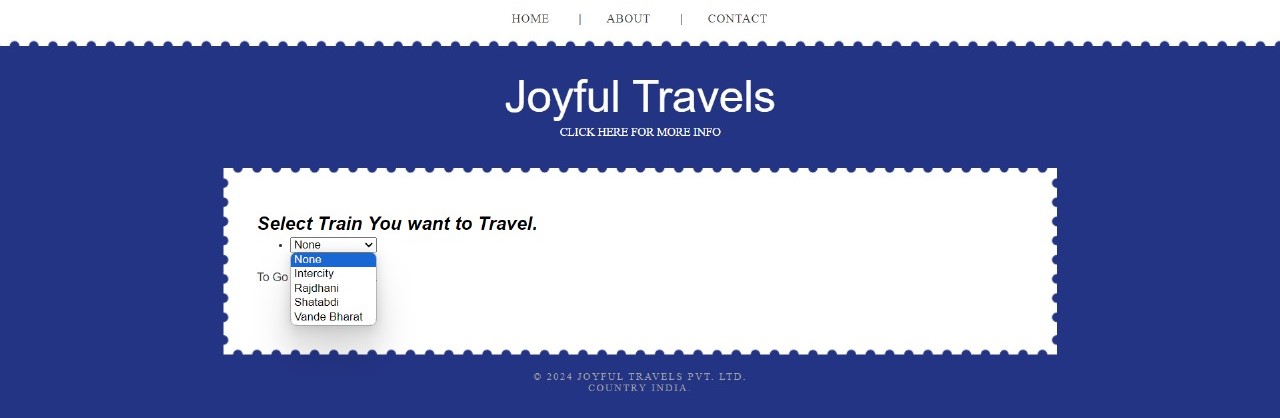
**Hotel Booking**

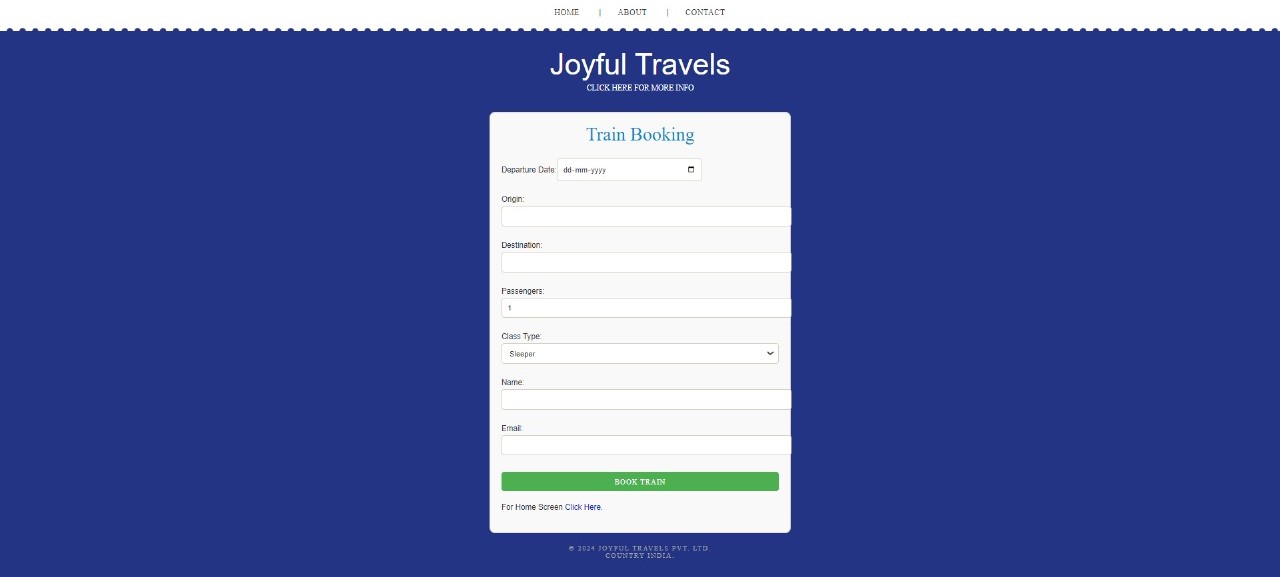




The above pages show the process of hotel booking. The website provides a select hotels and then lets you choose details such as check in date, check out date, room type, number of guests and more.

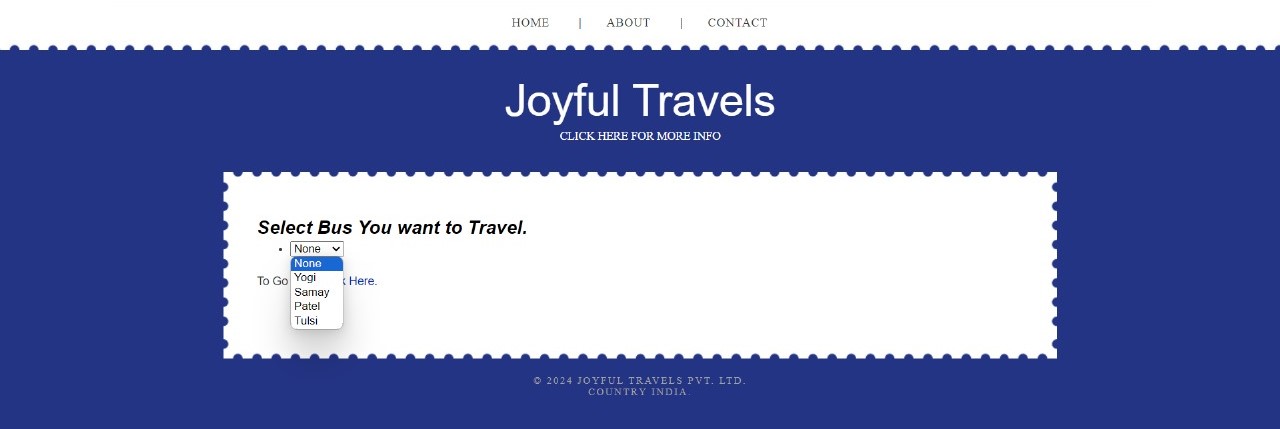
**Train Booking**

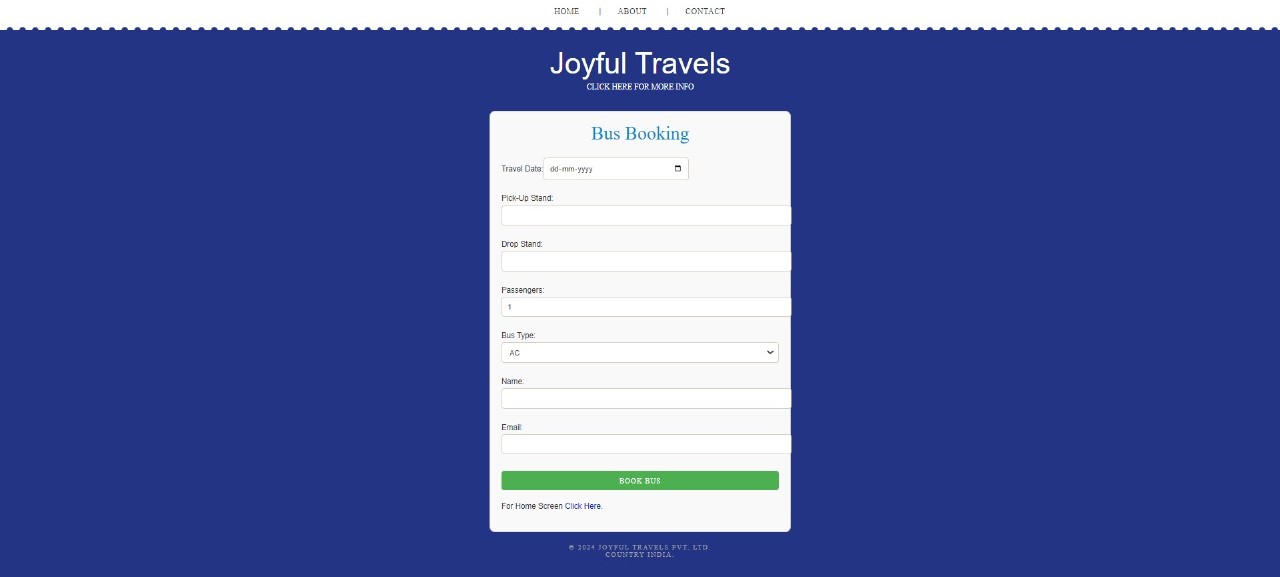




The above pages show the process of train booking. The website provides a select train companies and then lets you choose details such as departure date, to and from, number of passengers and more.

**Bus Booking**



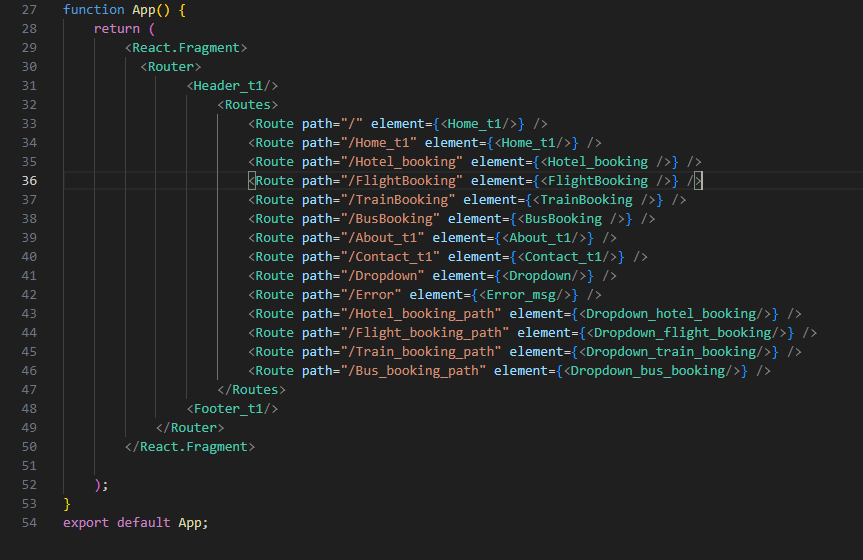


The above pages show the process of bus booking. The website provides a select bus companies and then lets you choose details such as departure date, to and from, number of passengers and more.

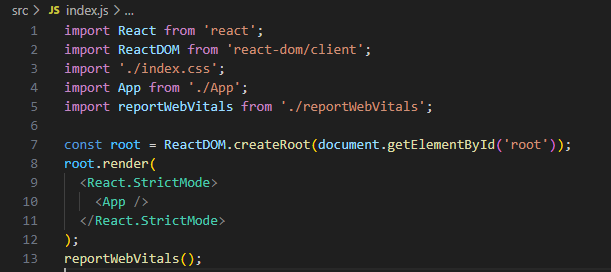
**Code**

**App.js**

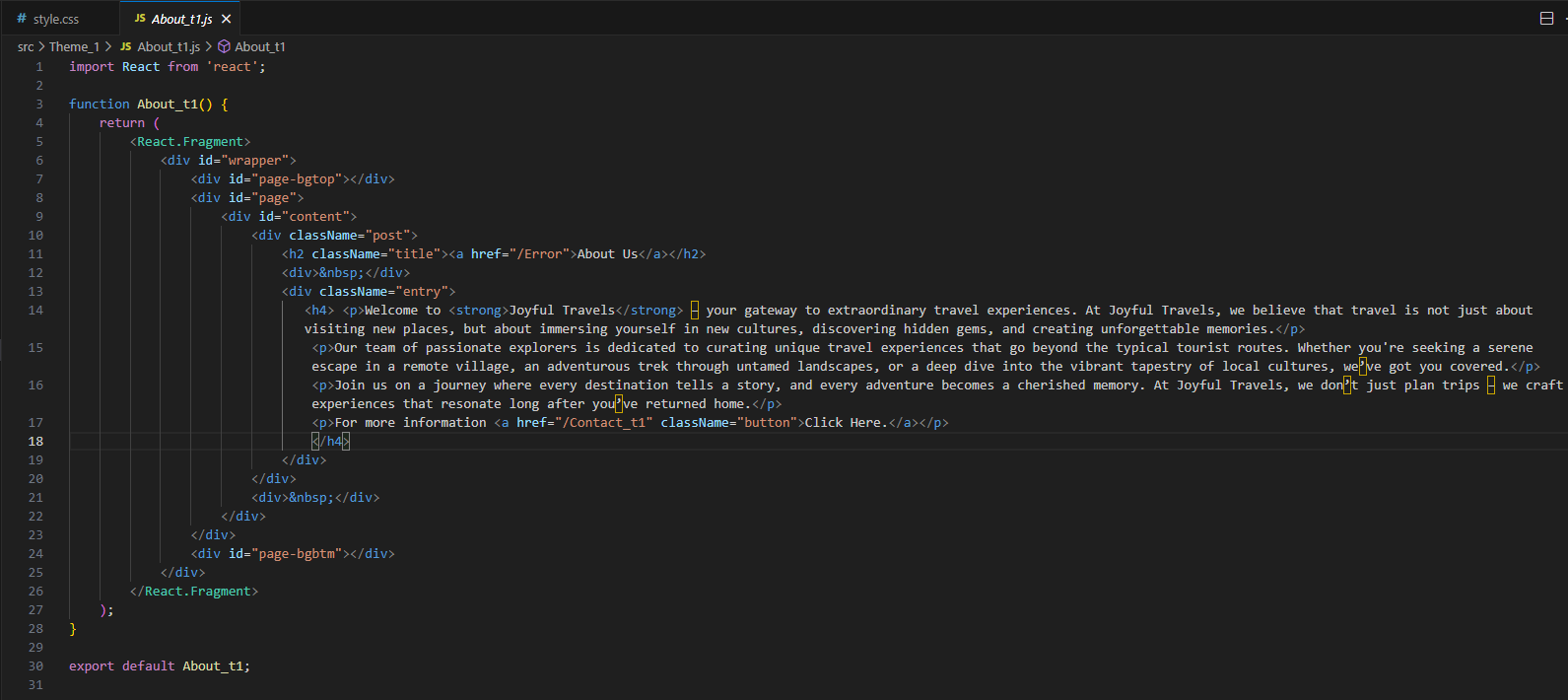
****

****

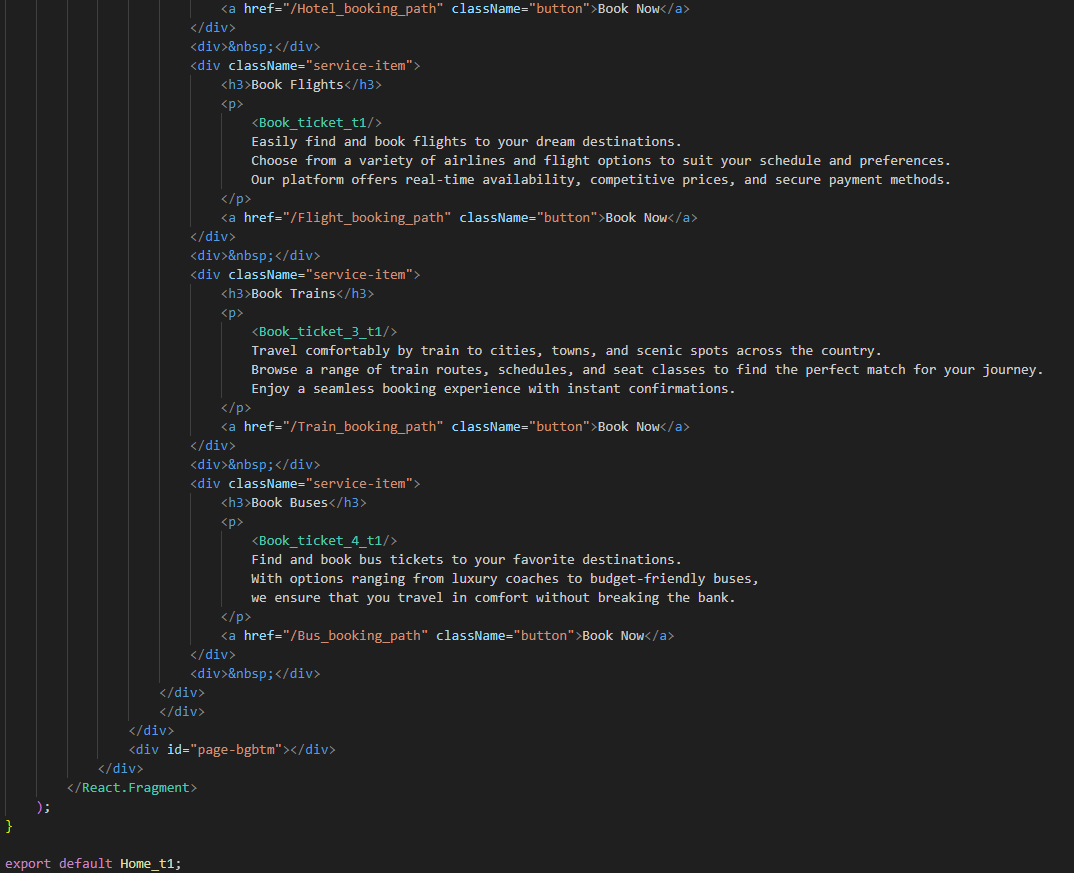
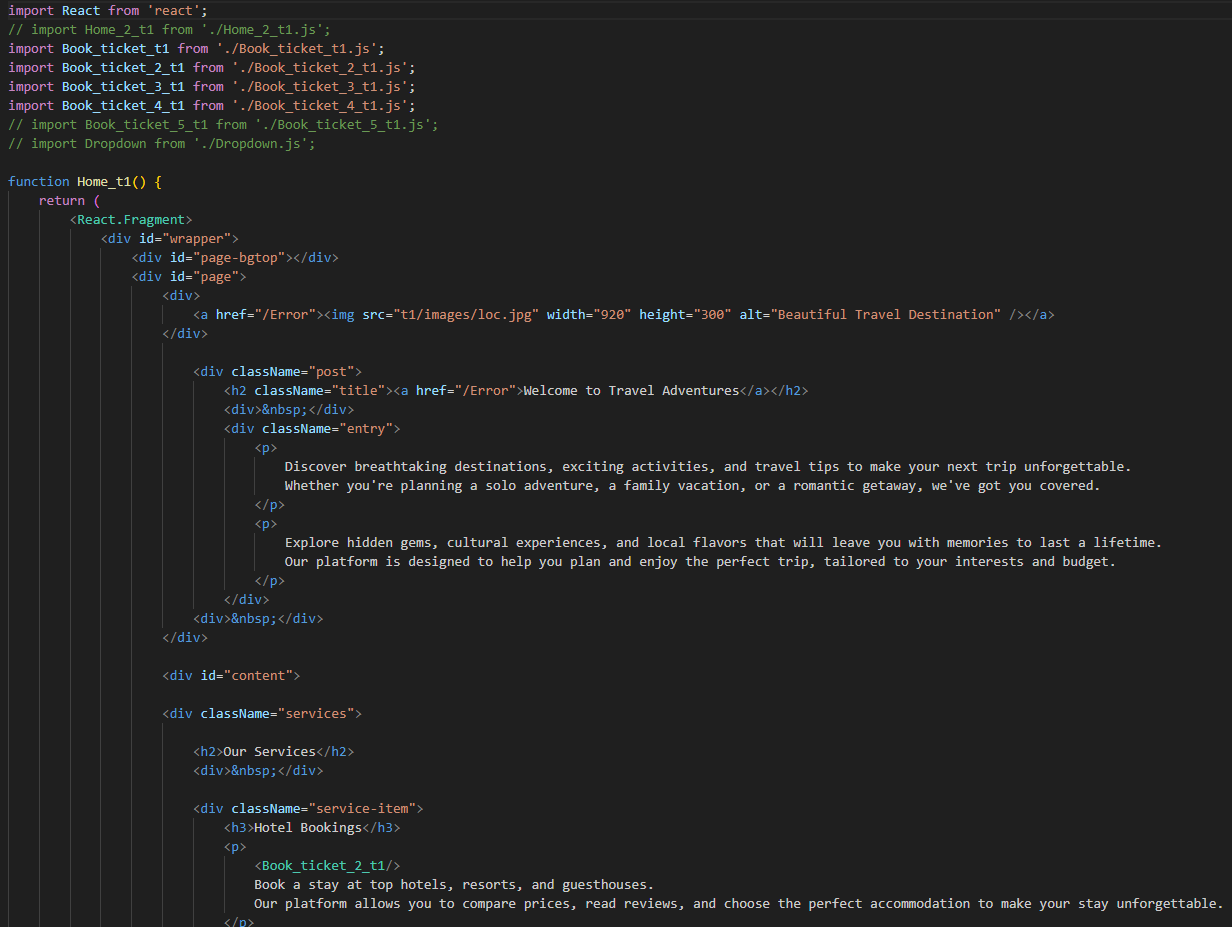
**Index.js**

****

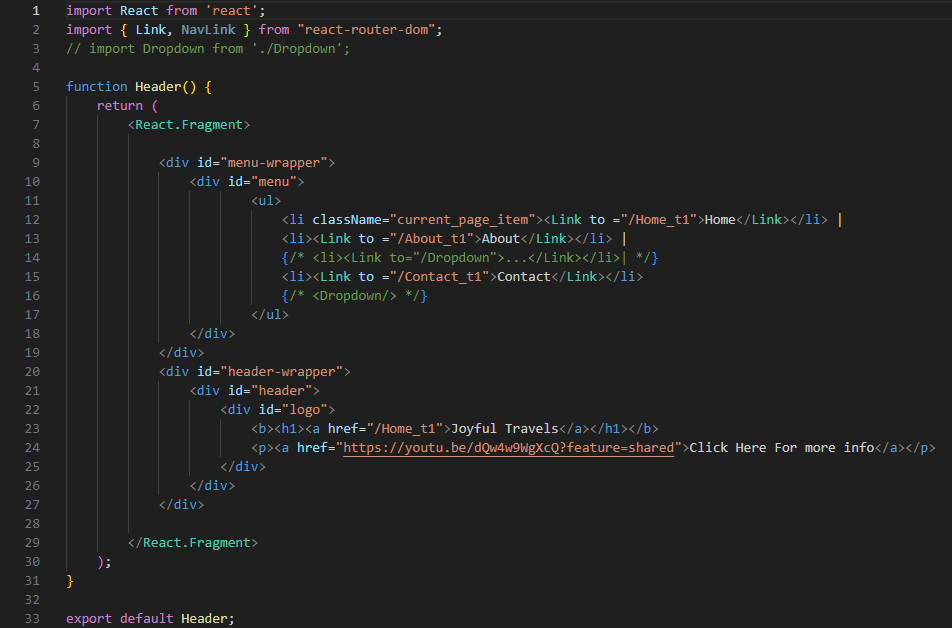
**About page**



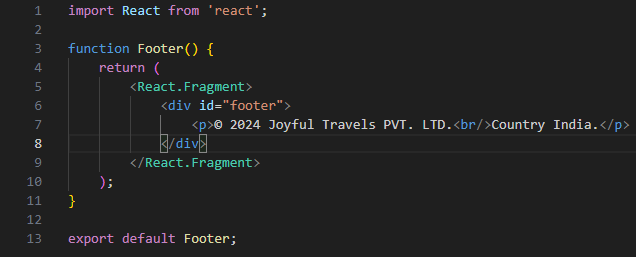
**Home**



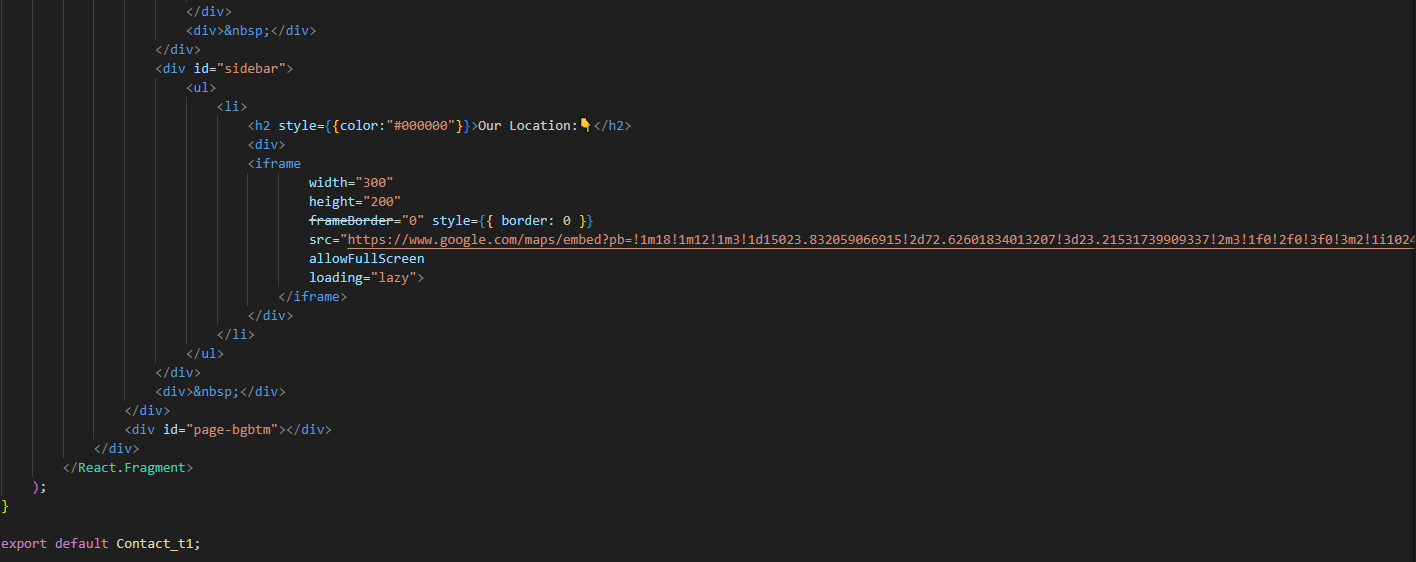
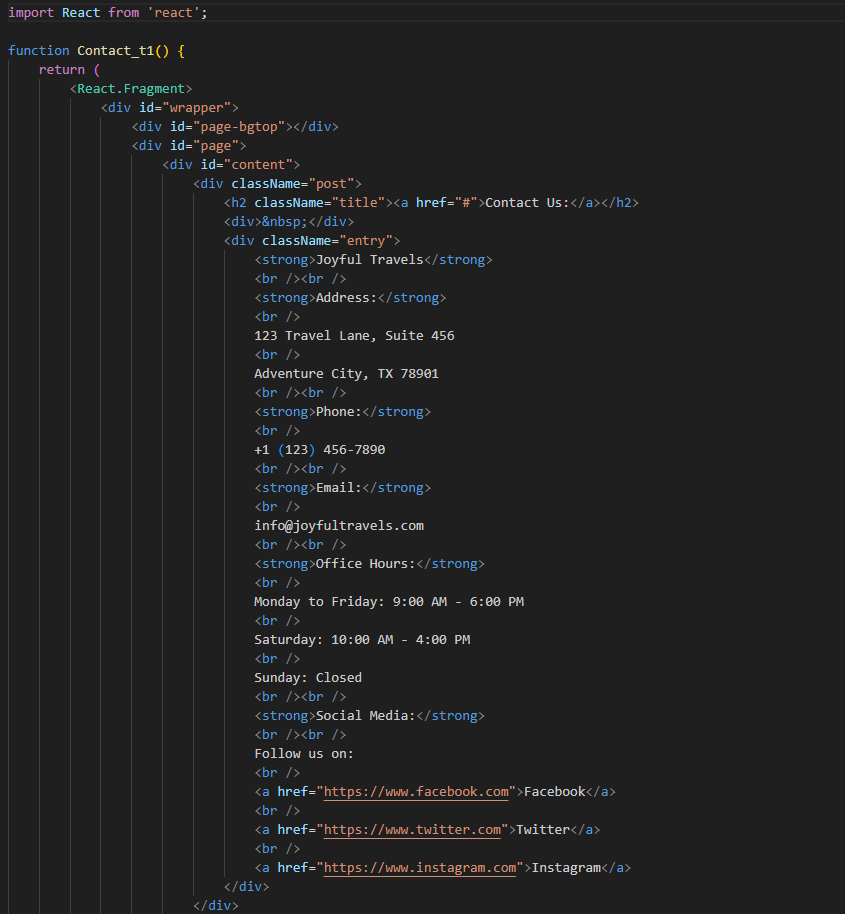
**Header**



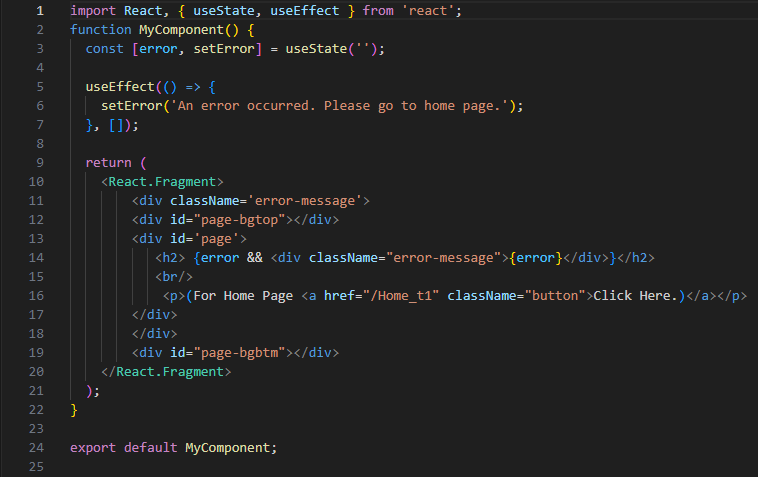
**Footer**



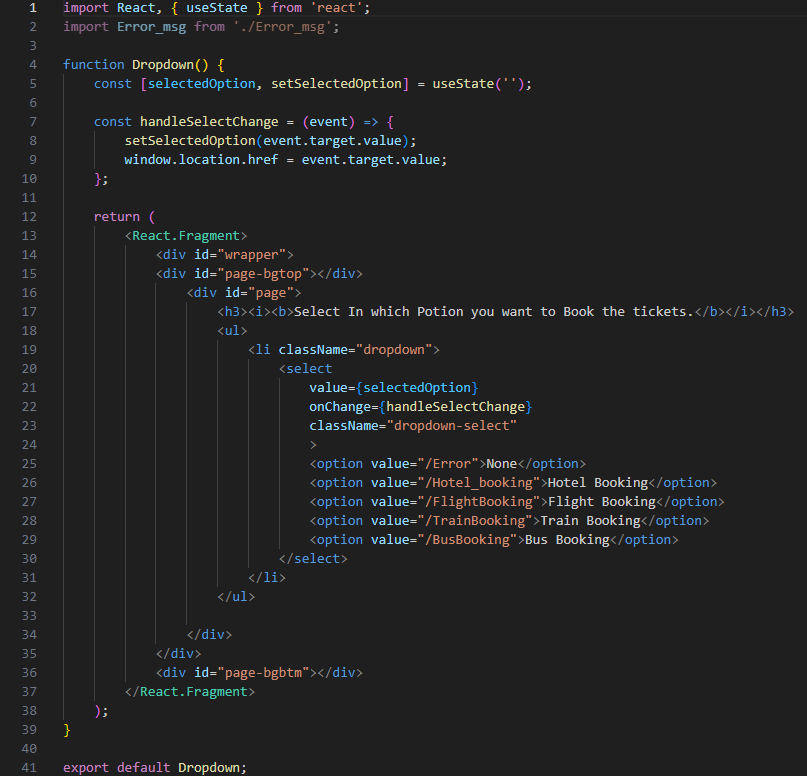
**Contact page**



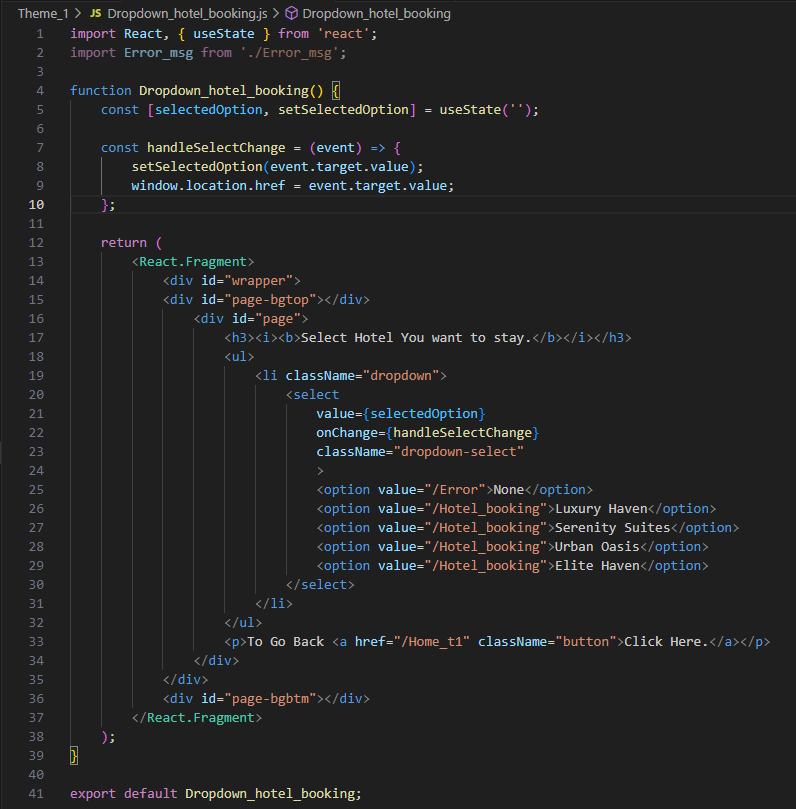
**Error page**

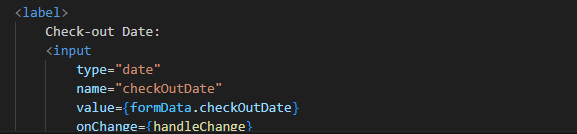


**Dropdown**

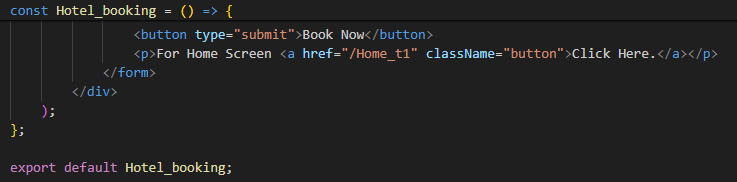


**Hotel Booking and dropdown**



****



****

