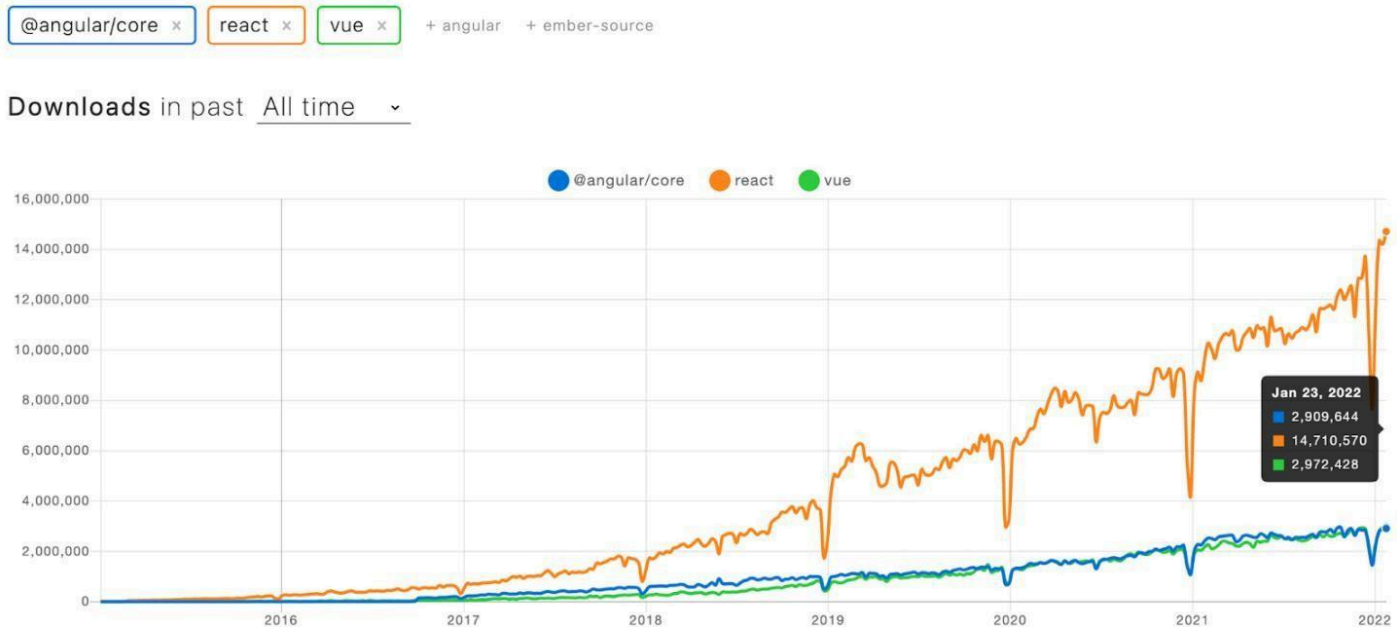
Lab-3: Frontend Development using React JS

Table of Contents:

Introduction to React JS..... 1
Why React?.....2
Single Page Application..... 3
Component Based Architecture.....3
File Structure..... 8
Lab Flow & Resources..... 11
Tutorial..... 12

Introduction to React JS

React, often referred to as React JS or simply React, is an open-source JavaScript library for building user interfaces used to build single page applications. Developed and maintained by Facebook and a community of developers, React has become one of the most popular and influential front-end libraries in the world of web development.



Why React?

React was designed to solve some key challenges in building modern web applications:

1. **Component-Based Architecture:**

    a. React encourages a modular approach to building User Interfaces. In React, the user interface is broken down into reusable components, which makes code organization and maintenance easier.

    b. A component is a piece of the UI (user interface) that has its own logic and appearance. A component can be as small as a button, or as large as an entire page.

2. **Virtual DOM:** React introduces a Virtual DOM, a lightweight copy of the actual DOM (Document Object Model). This allows React to efficiently update and render only the parts of the UI that have changed, resulting in better performance.

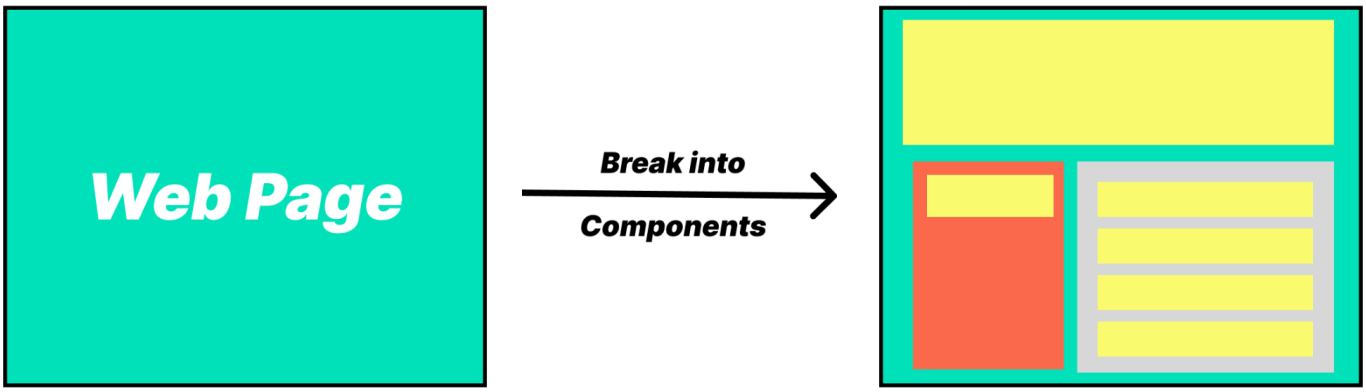
3. **Declarative Syntax:** React uses a declarative syntax, which means you describe the desired UI state, and React takes care of updating the actual UI to match that state. This makes code more predictable and easier to understand.

4. **Community and Ecosystem:** React has a vast and active community, which means abundant resources, libraries, and tools are available for developers. You'll find a wealth of third-party packages, extensions, and best practices to leverage.

Single Page Application

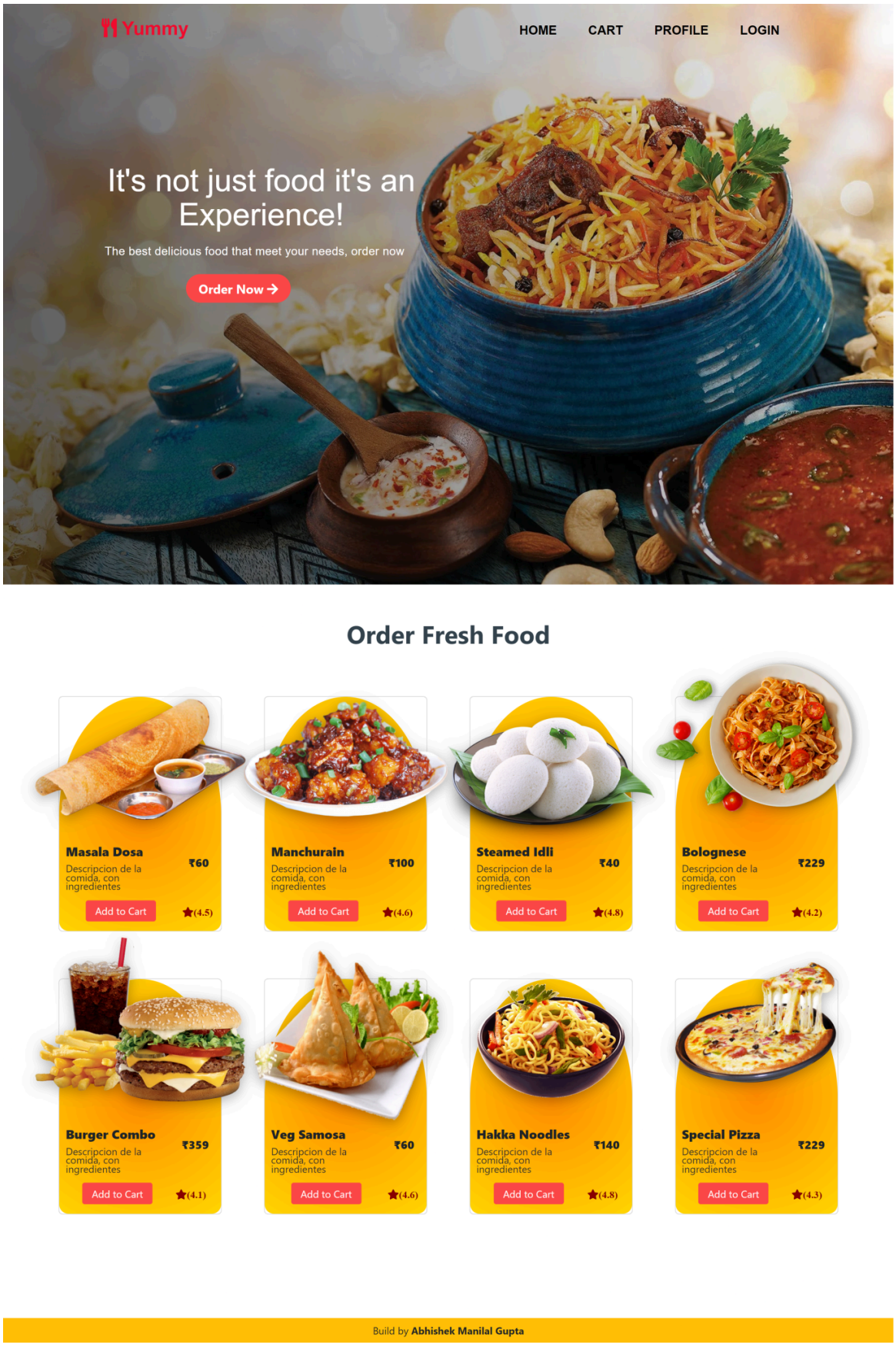
It means you load once, and rest everything is done by Javascript, like routing, loading css, images, videos, etc.

Component Based Architecture

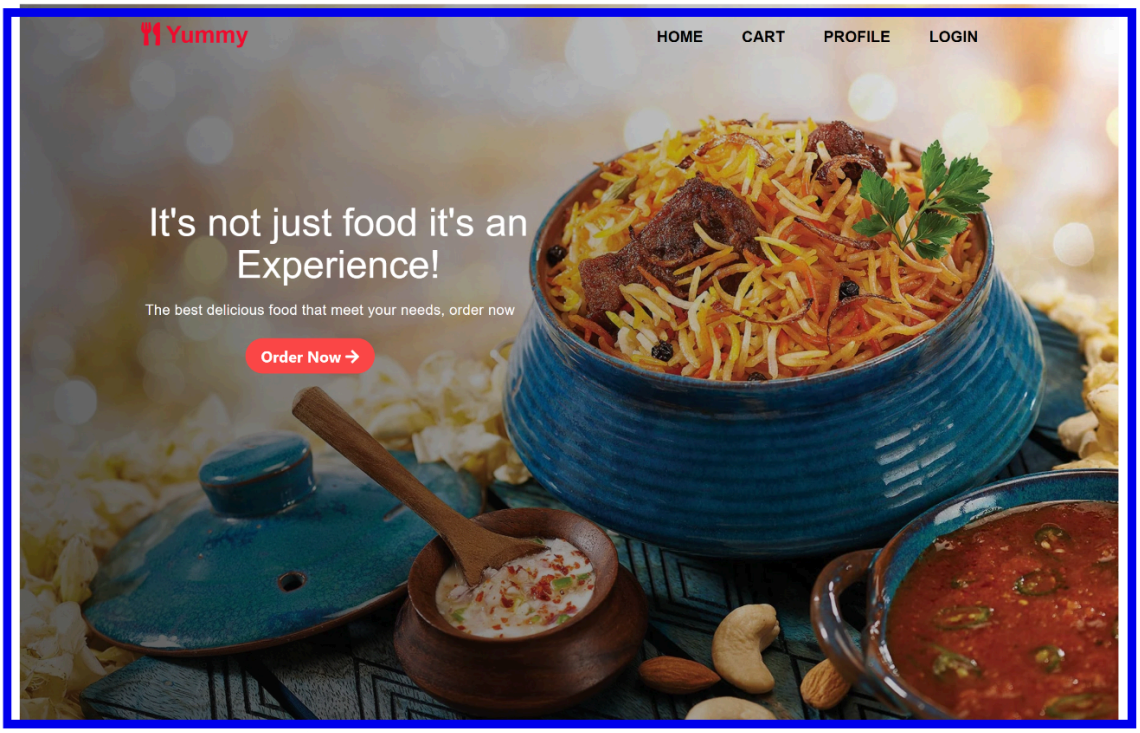


1. React encourages a modular approach to building User Interfaces. In React, the **user interface is broken down into reusable components**, which makes code **organization** and **maintenance easier.**
2. A component is a piece of the UI (user interface) that has its own logic and appearance. A component can be as small as a button, or as large as an entire page.

Question: So, in our “Yummy” website what are the reusable components?



Solution:



Header

Products

Footer

Navbar

Header

Sub heading

Products Section

Product Card

Footer

Identifying reusable components here:

1. First we divide our web pages into sections, irrespective of the fact whether they are reusable or not, just divide it into sections/components.
    a. Header
    b. Products Section
    c. Footer
2. Now, further we dig deep, and break it down further into components which are reusable across many pages in our application.
    a. Navbar
    b. Sub Header
    c. Product Card

After Identifying all the components, we can code them individually or a group of developers can code it.

Then to make pages, we create large components which will act like a single page. And will contain multiple components in it.

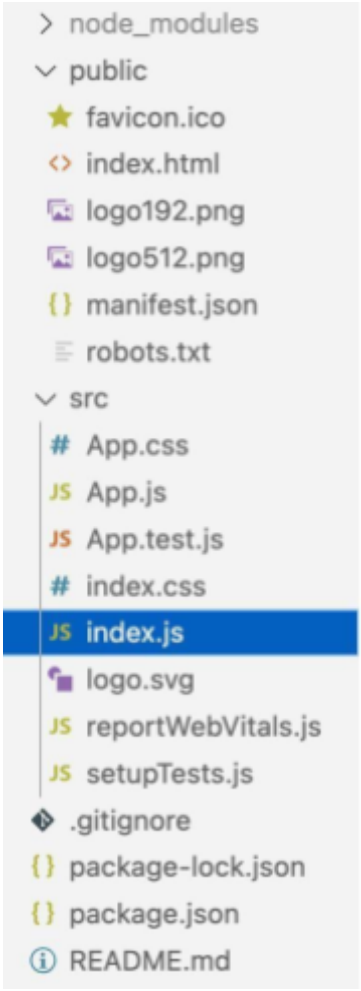
File Structure

(Prerequisite: Lab-1 Environment Setup)

1. **Create a react application** naming ‘react-app’ by running following commands on terminal.

    a. Steps:
        i. $ npx create-react-app react-app
        ii. $ cd react-app
        iii. $ code . (Open VS Code)
        iv. (Now, Return to / Open Terminal)
        v. $ npm start (It will start our react app typically on localhost:3000).

2. **Project Folder Structure:**



Sr. No.	Folder/ Folder Item	Description
1.	node_modules	This directory contains all the dependencies (third-party packages and libraries) used in your React application. You don't need to manually manage these packages; tools like npm or yarn handle the installation and management of these dependencies for you.
2.	public	The public directory is where you place static assets and the main HTML file for your application.
3.	public/index.html	This is the main HTML file that serves as the entry point for your React application. It typically includes a root element with an id="root" where the React app will be rendered. You can also include other static assets like CSS, images, and fonts in this directory.
4.	src	The src directory is where you put your application's source code, including React components, styles, and other JavaScript files.
5.	src/index.js	This is the entry point for your React application. It usually contains the code that renders your React application and attaches it to the root element specified in public/index.html.
6.	src/App.js	The main component that represents your application. It may contain the high-level structure of your app, routing, and other global components.
7.	package.json	This file contains metadata about your project, including its name, version, dependencies, and scripts. It's also where you specify project- specific configurations.
8.	package-lock.json	These files contain detailed information about the specific versions of dependencies installed in your project. They help ensure that the same versions are used when you or other developers work on the project.

Index.js under the src folder is the main entry point of the React application. In this file, the main DOM is initiated with the name ‘root’ and renders the first module(i.e. component). Following is the demo index.js file that calls the App component.

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Following is the demo App.js that generates the main page which will be shown on the browser when the application starts.

```
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default App;
```

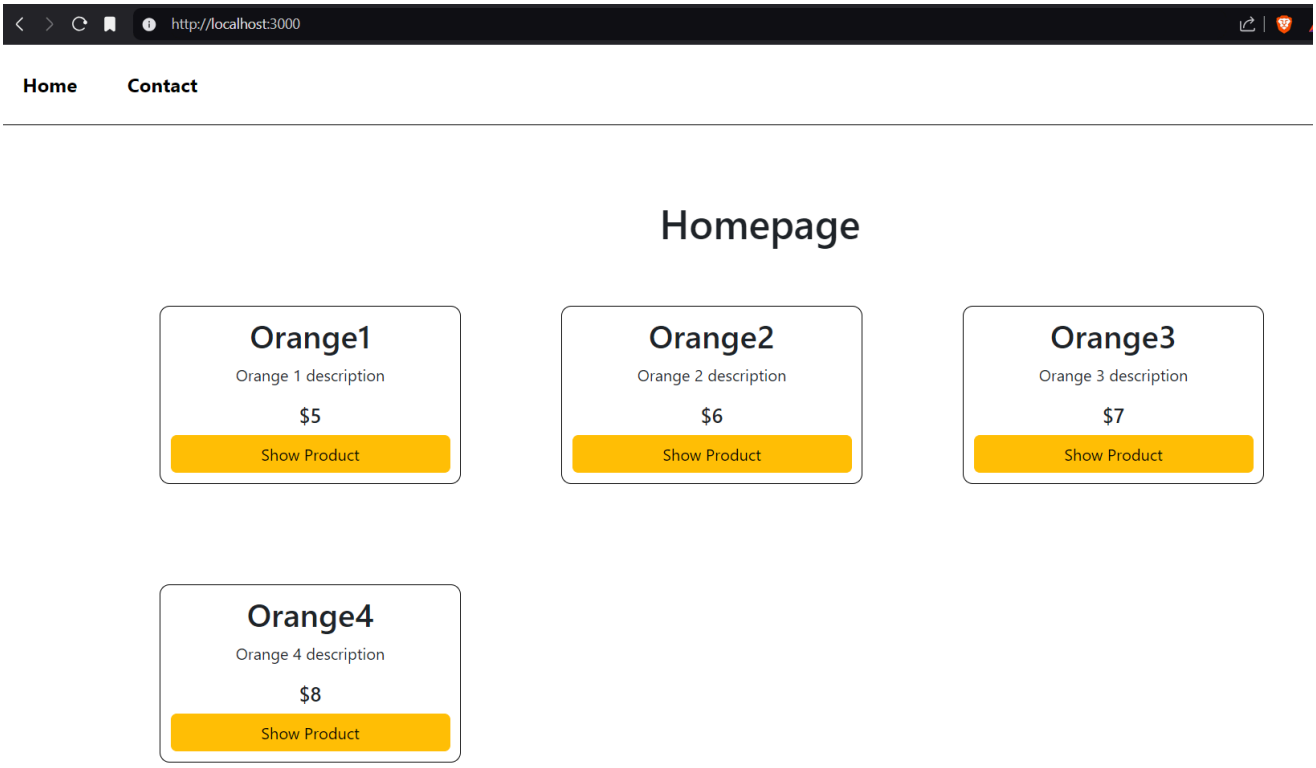
Lab Flow & Resources

So, During lab we will cover, following topics

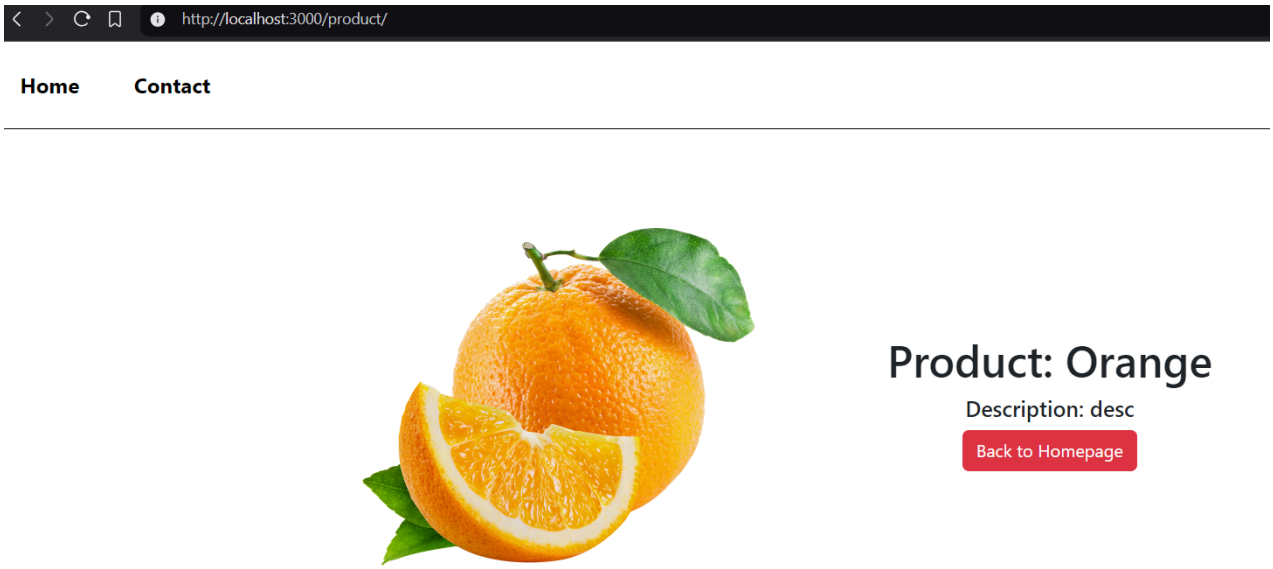
1. Creating React Application
    a. https://create-react-app.dev/docs/getting-started
2. Including bootstrap in react application
    a. https://react-bootstrap.netlify.app/docs/getting-started/introduction
    b. Inside you app.js file => import 'bootstrap/dist/css/bootstrap.min.css';
3. Explaining File structure
    a. Creating pages (large component/section) (JSX files)
    b. Creating components (smaller components/section) which will be used by those pages.
4. Introduce JSX
5. Hello World program
6. Creating React Functional Components
7. Passing Props
8. Re-rendering of Components
9. Component Tree
10. React Router (https://reactrouter.com/en/main/start/tutorial)
    a. Show why anchor tags are bad => Causes re-rendering of app
    b. Use of Link tags & Passing parameters via URL (useParams Hook)
    c. Navigating to a page upon an event occurrence.
    d. useLocation Hook (optional) [reference]

Tutorial

Homepage



Product Page



CSS Code:

```css
.App {
  text-align: center;
}
.nav {
  display: flex;
  flex-direction: row;
  margin-bottom: 50px;
  border-bottom: 1px solid #000;
}
.nav-item {
  text-decoration: none;
  color: black;
  text-transform: capitalize;
  font-weight: bold;
  font-size: larger;
  padding: 25px;
}
.product-card {
  display: flex;
  flex-direction: column;
  margin: 50px;
  padding: 10px;
  border-radius: 10px;
  border: 1px solid #000;
  width: 300px;
}
.products {
  display: flex;
  flex-direction: row;
  flex-wrap: wrap;
}
.product-page {
  display: flex;
  flex-direction: row;
  justify-content: center;
  align-items: center;
}
.product-left-side img{
  width: 400px;
}
.product-right-side {
  width: 500px;
  float: left;

}
```

Product API Data:

```
const productsData = [
        {
            productName: "Orange1",
            description: "Orange 1 description",
            price: "$5",
        },
        {
            productName: "Orange2",
            description: "Orange 2 description",
            price: "$6",
        },
        {
            productName: "Orange3",
            description: "Orange 3 description",
            price: "$7",
        },
        {
            productName: "Orange4",
            description: "Orange 4 description",
            price: "$8",
        },
    ];
```