**Simplified ERP System with React**

Project Overview

The Simplified ERP System is a web-based application built using React, designed to efficiently manage basic business operations. This system comprises three main pages: Dashboard, Products Management, and Orders Management. It serves as a user-friendly interface to help businesses organize and streamline their key processes.

Table of Contents

1. Project Structure

2. Components

- Dashboard

- Products Management

- Orders Management

3.Technologies Used

4. Routing

5. Styling

6.Running the Application

7. User Interface

8. Future Enhancements

9. Conclusion

Project Structure

The project structure follows a modular approach, with components organized into separate files for clarity. Key directories include:

simplified-erp-system/

|-- src/

| |-- components/

| |-- Dashboard.js

| |-- Products.js

| |-- Orders.js

| |-- App.js

| |-- index.js

| |-- index.css

| |-- ...

|-- public/

|-- package.json

|-- README.md

|-- ...

**. src/components/:** Contains React components for Dashboard, Products Management, and Orders Management.

* **src/App.js:** The main App component responsible for routing.
* **src/index.js:** Entry point of the application.
* **public/:** Public folder containing the HTML template.
* **package.json:** Configuration file for Node.js dependencies.

Components

Dashboard

The Dashboard provides an overview of the system's features. It includes key metrics or features and quick navigation links to the Products and Orders Management pages.

Implementation (`Dashboard.jsx`):

import React from 'react';

import { Link } from 'react-router-dom';

const Dashboard = () => {

  return (

    <div>

      <h1>Welcome to the ERP System</h1>

      <p>Total number of products: 10</p>

      <p>Total number of orders: 5</p>

      <Link to="/products">Manage Products</Link>

      <br />

      <Link to="/orders">Manage Orders</Link>

      <br />

    </div>

  );

};

export default Dashboard;

Products Management

The Products Management page allows users to view, add, edit, and delete product listings. It displays a list of products with details such as name, category, price, and stock quantity.

Implementation (`ProductsManagement.jsx`):

import React, { useState, useEffect } from 'react';

import axios from 'axios';

const ProductsManagement = () => {

  const [products, setProducts] = useState([]);

  useEffect(() => {

    axios.get('https://api.example.com/products')

      .then(response => {

        setProducts(response.data);

      })

      .catch(error => {

        console.log('Error fetching products:', error);

      });

  }, []);

  return (

    <div>

      <h1>Products Management</h1>

      <table>

        <thead>

          <tr>

            <th>Product ID</th>

            <th>Name</th>

            <th>Category</th>

            <th>Price</th>

            <th>Stock Quantity</th>

          </tr>

        </thead>

        <tbody>

          {products.map((product) => (

            <tr key={product.id}>

              <td>{product.id}</td>

              <td>{product.name}</td>

              <td>{product.category}</td>

              <td>{product.price}</td>

              <td>{product.stockQuantity}</td>

            </tr>

          ))}

        </tbody>

      </table>

    </div>

  );

};

export default ProductsManagement;

Orders Management

The Orders Management page facilitates viewing and handling orders. It shows a list of orders, including details like order ID, customer name, order date, and status.

Implementation (`OrdersManagement.jsx`):

import React, { useState, useEffect } from 'react';

import axios from 'axios';

const OrdersManagement = () => {

  const [orders, setOrders] = useState([]);

  useEffect(() => {

    axios.get('https://api.example.com/orders')

      .then(response => {

        setOrders(response.data);

      })

      .catch(error => {

        console.log('Error fetching orders:', error);

      });

  }, []);

  return (

    <div>

      <h1>Orders Management</h1>

      <table>

        <thead>

          <tr>

            <th>Order ID</th>

            <th>Customer Name</th>

            <th>Order Date</th>

            <th>Status</th>

          </tr>

        </thead>

        <tbody>

          {orders.map((order) => (

            <tr key={order.id}>

            <td>{order.id}</td>

            <td>{order.customerName}</td>

            <td>{order.date}</td>

            <td>{order.status}</td>

          </tr>

        ))}

      </tbody>

    </table>

    </div>

  );

};

export default OrdersManagement;

Technologies Used

-React: A JavaScript library for building user interfaces.

-React Router: A library for handling navigation in a React application.

-HTML/CSS: Markup and styling for the user interface.

Routing

React Router is utilized for navigation between different sections of the ERP system.

Implementation (`App.jsx`):

import React from 'react';

import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';

import Dashboard from './components/Dashboard';

import ProductsManagement from './components/ProductsManagement';

import OrdersManagement from './components/OrdersManagement';

function App() {

  return (

    <Router>

      <Routes>

        <Route path="/" element={<Dashboard />} />

        <Route path="/products" element={<ProductsManagement />} />

        <Route path="/orders" element={<OrdersManagement />} />

      </Routes>

    </Router>

  );

}

export default App;

Styling

Styling is crucial for a pleasant user experience. You can use CSS or styling libraries (e.g., Bootstrap) to make the components responsive and visually appealing.

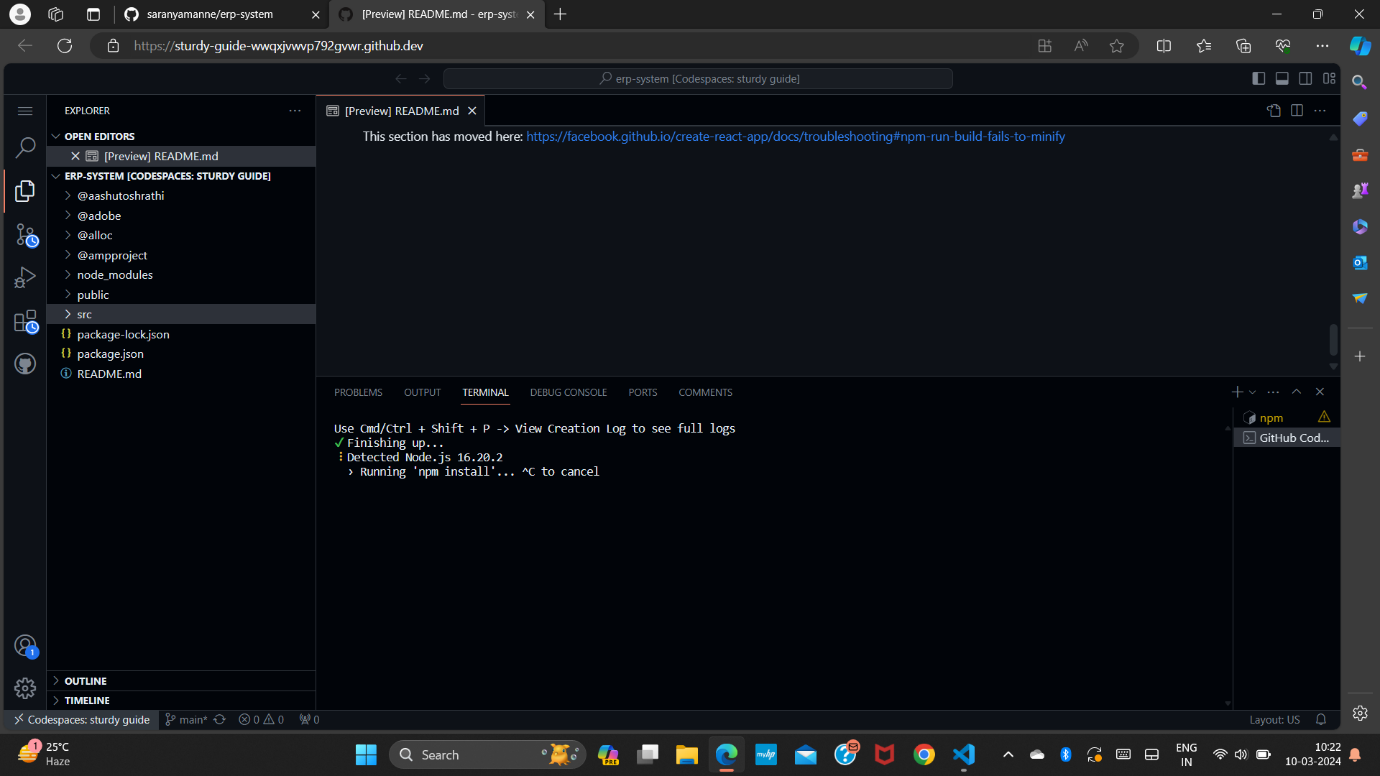
Running the Application

To run the application, make sure Node.js and npm are installed on your machine. Then, follow these steps:

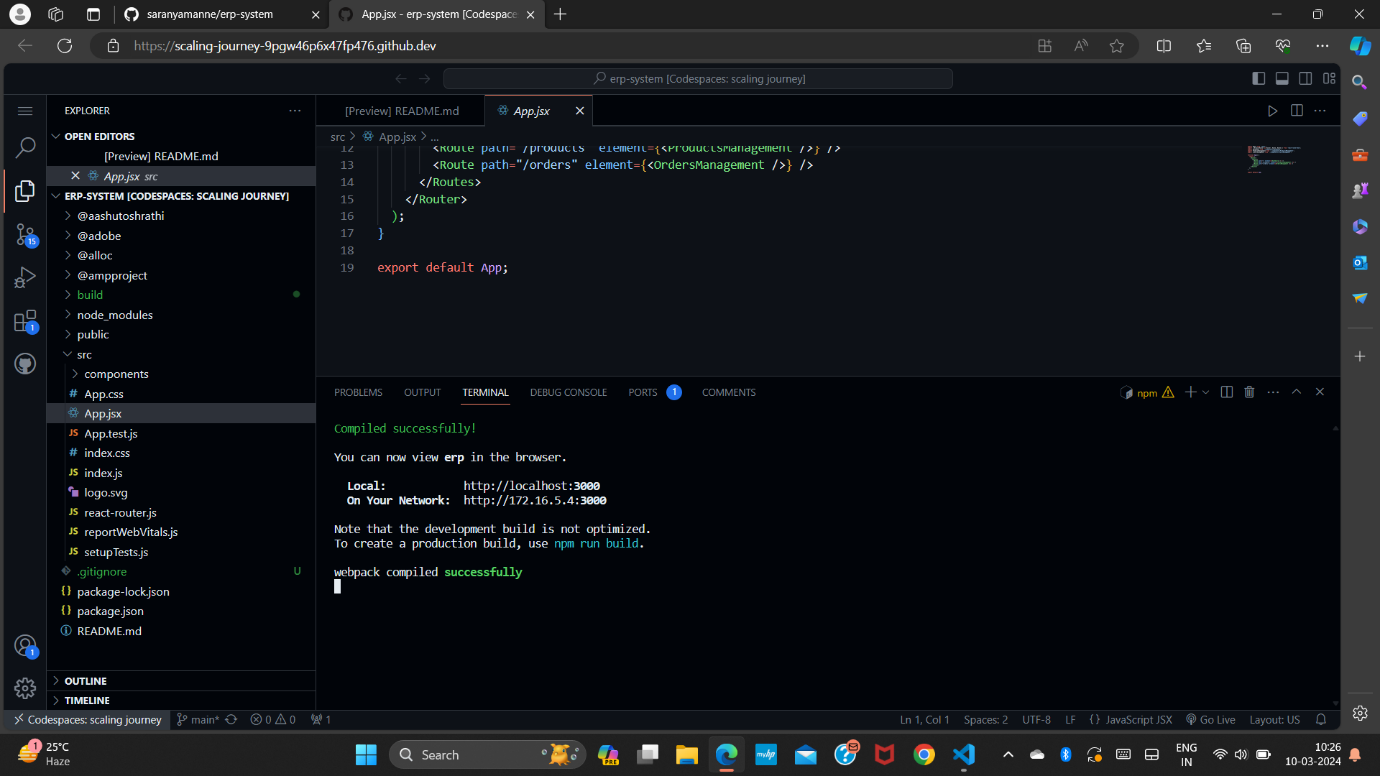
1. Clone the repository: `https://github.com/saranyamanne/erp-system.git `

2. Navigate to the project directory: `cd erp-system`

3. Install dependencies: `npm install`



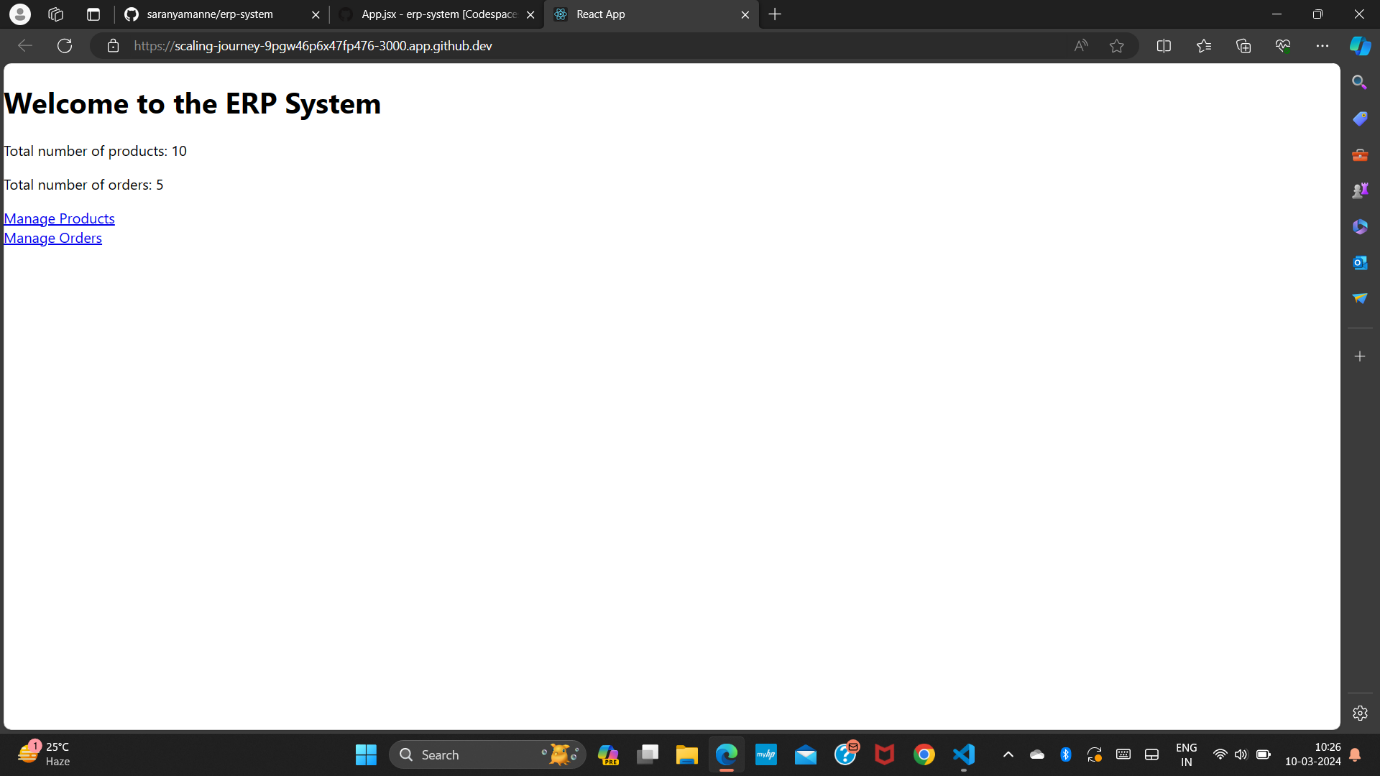
4. Start the development server: `npm start`

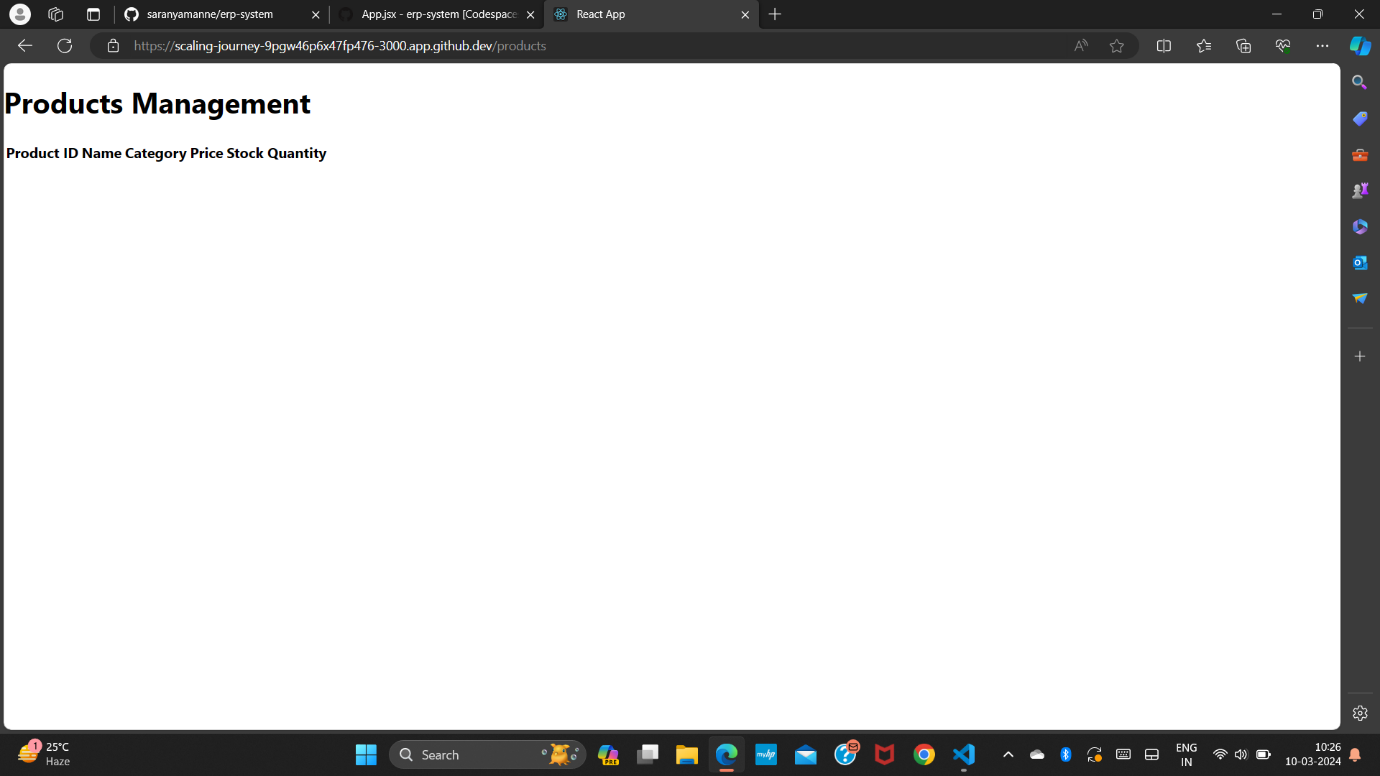


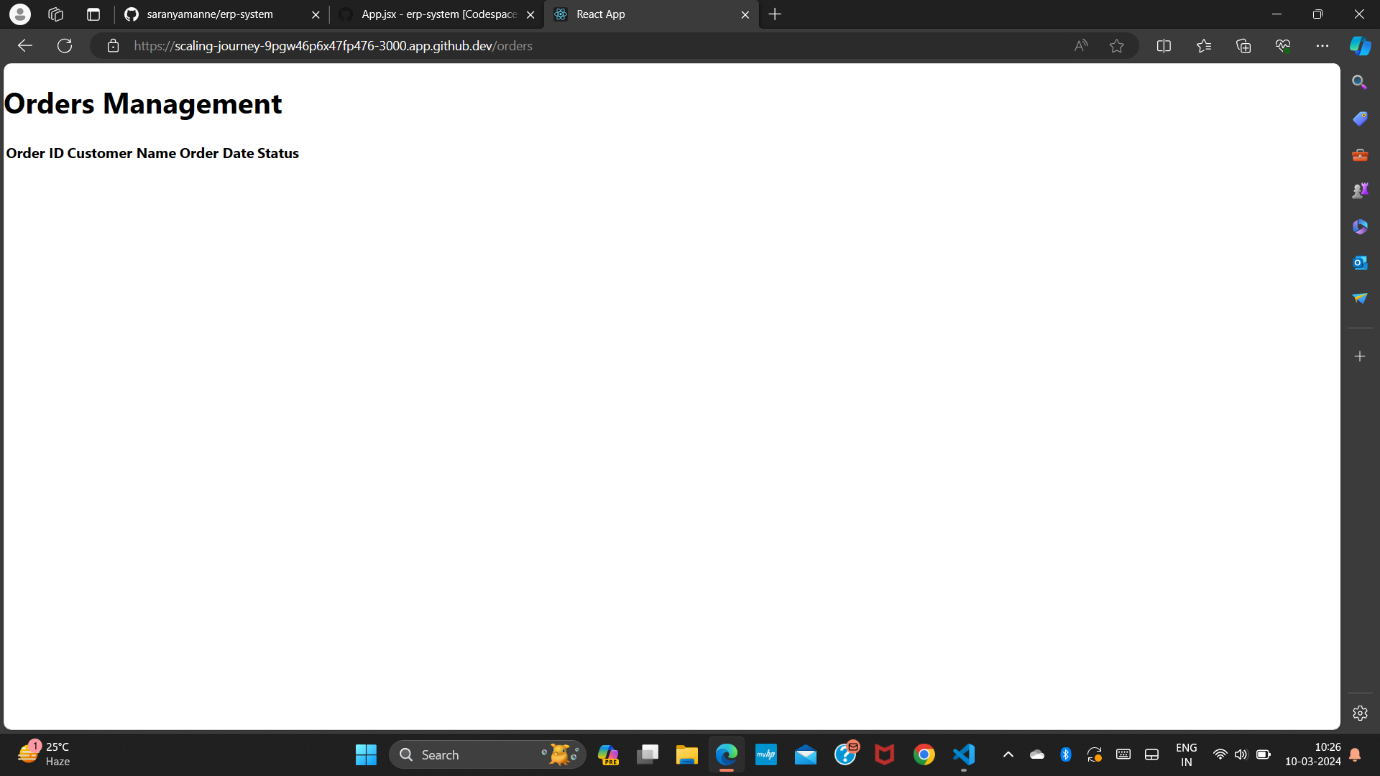
5. Open your browser and go to http://localhost:3000 to view the application.

User Interface

After opening the browser it looks like:







Future Enhancements

While this version of the Simplified ERP System focuses on basic functionalities, future enhancements could include:

- Integration with a backend to handle real data and user authentication.

- Enhanced user interfaces for product and order management with interactive features.

- Implementation of additional modules, such as customer management, inventory tracking, etc.

-User authentication and authorization for secure access.

-Advanced features like search, filters, and sorting in Products and Orders Management.

Conclusion

The Simplified ERP System with React provides a foundation for managing basic business operations. By extending its features and integrating a backend, the system can be expanded to meet the evolving needs of businesses. It provides a modular and responsive interface, making it adaptable to various screen sizes. This documentation serves as a guide to understanding the project structure and components for further development and customization.