# MIT Art, Design and Technology University
# MIT School of Computing, Pune

## Department of Information Technology

# Lab Manual

## Practical - Web Programming

## Class - T.Y. (SEM-VI), SMAD

## Batch - SMAD

### Name of the Student

**Mr. ANIKET SULANE**

**A.Y. 2024 – 2025 (SEM-VI)**

| Assignment No 1: Project Assignment based on Unit-I and Unit-II | |
|---|---|
| **Project-I** | Project Title: ReactJS E-Commerce Website with Page Routing Project<br><br>Statement:<br><br>Objective:<br>Develop a fully functional E-commerce website using ReactJS that includes page routing functionality for smooth navigation between different sections like Home, Product Listing, Product Details, Cart, and Checkout.<br><br>Project Features:<br><br>1. Navigation Bar with Routing:<br>  ○ Create a Navigation Bar at the top of the website with links to:<br>  ▪ Home: Displays featured products and general information about the store.<br>  ▪ Products: Lists all available products for purchase.<br>  ▪ Cart: Displays the products added to the shopping cart.<br>  ▪ Checkout: Allows the user to review and complete the purchase.<br>  ▪ Profile: Displays user account information and order history.<br>  ○ Implement React Router for routing between these pages.<br>2. Home Page:<br>  ○ The Home page should showcase highlighted products or categories.<br>  ○ Implement a product carousel or featured section.<br>  ○ Include a search bar to allow users to search for products directly.<br>3. Product Listing Page:<br>  ○ Create a Product Listing page that fetches products dynamically from an API or static JSON file.<br>  ○ Each product in the list should show:<br>  ▪ Product image.<br>  ▪ Product name.<br>  ▪ Price.<br>  ▪ A button to add the product to the cart.<br>  ○ Implement React Router links on product items to navigate to the Product Details page.<br>4. Product Details Page:<br>  ○ When a user clicks on a product, navigate them to the Product Details page.<br>  ○ The Product Details page should display:<br>  ▪ Full-size product image.<br>  ▪ Detailed product description.<br>  ▪ Price. |

        - Add to Cart button.
    - Allow users to select product quantity before adding it to the cart.
5. Cart Page:
    - Implement a Cart page that displays products added to the cart.
    - Each cart item should include:
        - Product name.
        - Product quantity.
        - Price per item and total cost for the item.
        - A button to remove items from the cart.
    - Display a total price for all items in the cart.
    - Include buttons for Continue Shopping and Proceed to Checkout.
6. Checkout Page:
    - Create a Checkout page where users can review their cart, enter shipping information, and confirm the order.
    - Display an order summary with total cost, shipping details, and payment options.
    - Provide a Place Order button to finalize the transaction.
7. User Profile Page:
    - Create a Profile page where users can view their account information and order history.
    - Include a form to update the profile details (name, email, shipping address).
8. State Management with Context API/Redux:
    - Use React Context API or Redux to manage the global state of the cart, allowing users to add, remove, and update quantities of products across different pages.
    - Ensure that the cart data is preserved during navigation between pages.
9. Responsive Design:
    - Ensure the website is responsive and works well on both desktop and mobile devices using CSS Flexbox or CSS Grid.

Technical Requirements:

- ReactJS: For building the user interface.
- React Router: For implementing page routing functionality.
- State Management: Use React Context API or Redux to manage cart state and other global data.
- API Integration: Fetch products from a static JSON file or a mock API (e.g., JSONPlaceholder or a custom API).
- Responsive Styling: Use CSS/SCSS to ensure the website is mobile-friendly.
- Optional: Add localStorage or sessionStorage to persist cart data between page reloads.

Expected Outcome:

| | |
|---|---|
| | By completing this project, you will:<br>● Learn how to implement page routing in a ReactJS application using React Router.<br>● Build a dynamic E-commerce website with features like product listings, cart management, and checkout functionality.<br>● Gain experience with state management and ensuring the cart data persists across different pages.<br>● Develop a responsive, user-friendly interface that functions well on both mobile and desktop screens. |
| **Assignment No 2: Project Assignment based on Unit-III, IV and V** | |

| | |
|---|---|
| **Project-II** | Project Title: React Native E-Commerce Application<br><br>Project Statement:<br><br>Objective:<br>Build a fully functional E-commerce mobile application using React Native that integrates navigation, core components, user interaction, dynamic list rendering, and state management.<br><br>Project Features:<br><br>1. User Authentication:<br>  ○ Implement a simple login screen where users can enter their credentials (username and password).<br>  ○ After successful login, navigate to the Home Screen.<br>2. Navigation:<br>  ○ Set up react-navigation to allow users to navigate between the following screens using StackNavigator:<br>  ▪ Home Screen: Displays a list of products.<br>  ▪ Product Details Screen: Shows details of the selected product.<br>  ▪ Cart Screen: Displays products added to the shopping cart.<br>  ▪ Profile Screen: Displays user profile details.<br>  ▪ Settings Screen: Allows the user to modify their account settings.<br>3. Core Components & UI:<br>  ○ Use core React Native components such as View, Text, Image, StyleSheet, and Button to build a user-friendly interface.<br>  ○ Apply Flexbox for layout, ensuring responsiveness across different screen sizes.<br>4. Product List and Cart:<br>  ○ On the Home Screen, render the list of products using FlatList. Each item should display a product image, name, and price. |

| | |
|---|---|
| | ○ Implement a button for each product to add it to the Cart.<br>5. Handling User Interactions:<br>○ Implement touch events to handle interactions such as adding products to the cart, navigating to the product details, and updating the cart.<br>○ Use onPress to trigger actions like adding a product to the cart or navigating between screens.<br>6. Cart Management:<br>○ Use React Context API for global state management to handle the cart's data (adding/removing products).<br>○ Show the cart's content and total price on the Cart Screen.<br>7. Product Details:<br>○ When a user clicks on a product in the Home Screen, navigate to the Product Details Screen and display detailed information about the product, such as the description, image, and price.<br>8. Responsive Design:<br>○ Ensure that the app is responsive, and elements adjust properly to different screen sizes using Flexbox.<br>9. Rendering Lists with SectionList:<br>○ On the Home Screen, group products by categories (e.g., electronics, clothing) and use SectionList to render them in separate sections with headers.<br><br>Technical Requirements:<br><br>• React Native with Expo CLI for development and testing.<br>• React Navigation for screen transitions.<br>• State Management using React Context API or useState, useReducer.<br>• Dynamic data rendering using FlatList and SectionList.<br>• Styling using StyleSheet and Flexbox for layout.<br><br>Expected Outcome:<br><br>By completing this project, students will gain hands-on experience with:<br><br>• Setting up and using React Native and React Navigation.<br>• Building a fully interactive mobile application with dynamic data rendering and state management.<br>• Applying Flexbox for responsive layouts and managing user interactions.<br>• Creating a simple e-commerce app with multi-screen navigation, product lists, and state-driven cart functionality. |

# Experiment No.1

## Problem Statement:

In today's digital era, users expect fast, responsive, and easy-to-navigate e-commerce platforms. Many existing platforms are cluttered or lack smooth navigation, real-time cart updates, and mobile responsiveness. There is a need for a modern, user-friendly web application that allows users to browse products, view detailed information, manage their cart, and complete purchases with ease.

This project aims to **develop a dynamic E-Commerce website using ReactJS**, featuring **page routing**, **state management**, and **responsive design**, which addresses common usability issues and provides a seamless online shopping experience.

## Objective:

The objective of this project is to develop a fully functional and responsive **E-Commerce website using ReactJS** that provides a seamless user experience through **page routing**, **state management**, and **dynamic product display**. The website aims to allow users to browse products, view detailed information, manage their shopping cart, and complete the checkout process efficiently. It also focuses on building a real-world application that demonstrates skills in **React development**, **routing**, **global state handling**, and **responsive UI design**.

## Theory:

This project is a **responsive E-Commerce website** developed using **ReactJS**. It includes multiple pages such as **Home**, **Products**, **Product Details**, **Cart**, **Checkout**, and **Profile**, all connected through **React Router** for smooth and fast navigation. The Home page features highlighted products and a search bar for easy access.

Products are dynamically loaded from a static JSON file or API and displayed with name, price, and image. Users can view detailed information on the **Product Details Page**, select quantity, and add items to the cart. The **Cart Page** displays all added items, allows quantity updates or removal, and shows the total cost.

In the **Checkout Page**, users review their cart, enter shipping details, and confirm their order. The **Profile Page** displays user details and order history, with an option to update profile info.

The application uses **React Context API or Redux** for managing the global cart state, ensuring data consistency across pages. It also supports **localStorage** to preserve cart data after page reloads. Styling with **CSS Flexbox or Grid** ensures the design is fully **responsive** and mobile-friendly. This project demonstrates core frontend development skills with real-world application.

**Code:**

A. Home page:

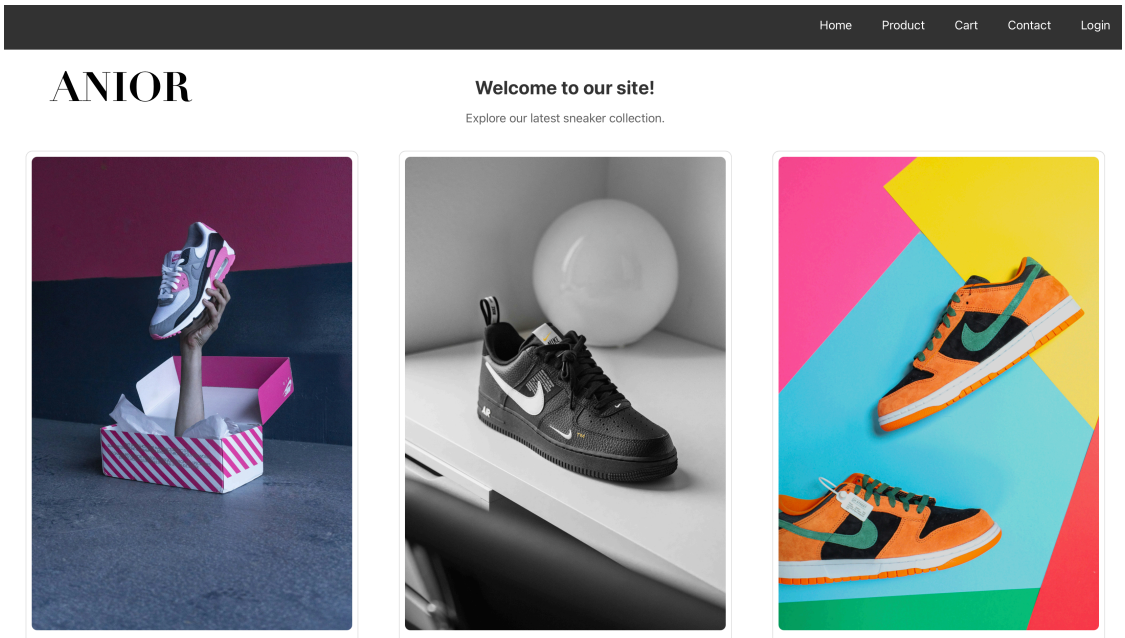
code: import React, { useState } from "react";

```
import { useNavigate } from "react-router-dom";
import A1 from "../assets/A1.jpg";
import A2 from "../assets/A2.jpg";
import A3 from "../assets/A3.jpg";
import A4 from "../assets/A4.jpg";
import A5 from "../assets/A5.jpg";
import A6 from "../assets/A6.jpg";
import A7 from "../assets/A7.jpg";
import A8 from "../assets/A8.jpg";
import A9 from "../assets/A9.jpg";
import "./Home.css";

const products = [
  { id: 1, img: A1, name: "Sneaker 1", price: 4999 },
  { id: 2, img: A2, name: "Sneaker 2", price: 5499 },
  { id: 3, img: A3, name: "Sneaker 3", price: 6299 },
  { id: 4, img: A4, name: "Sneaker 4", price: 4799 },
  { id: 5, img: A5, name: "Sneaker 5", price: 5999 },
  { id: 6, img: A6, name: "Sneaker 6", price: 6499 },
  { id: 7, img: A7, name: "Sneaker 7", price: 10899 },
  { id: 8, img: A8, name: "Sneaker 8", price: 9999 },
  { id: 9, img: A9, name: "Sneaker 9", price: 8799 },
];

const Home = ({ setCart }) => {
  const [selectedQuantities, setSelectedQuantities] = useState({});
  const [message, setMessage] = useState("");
  const navigate = useNavigate();

  const handleAddToCart = (product) => {
    const quantity = selectedQuantities[product.id] || 1;
    setCart((prevCart) => [...prevCart, { ...product, quantity }]);
    setMessage(`${product.name} (x${quantity}) added to cart!`);
    setTimeout(() => setMessage(""), 2000);
  };

  const handleQuantityChange = (productId, quantity) => {
    setSelectedQuantities((prev) => ({
```

```
    ...prev,
    [productId]: quantity,
  }));
};

return (
  <div className="home-container">
    <h1 className="title">Welcome to our site!</h1>
    <h1 className="brand-name">ANIOR</h1>
    <p className="subtitle">Explore our latest sneaker collection.</p>

    {message && <p className="cart-message">{message}</p>}

    <div className="product-grid">
      {products.map((product) => (
        <div className="product-box" key={product.id}>
          <img src={product.img} alt={product.name} className="product-image" />
          <h3>{product.name}</h3>
          <p>₹{product.price}</p>

          <select
            className="quantity-selector"
            onChange={(e) => handleQuantityChange(product.id, parseInt(e.target.value, 10))}
          >
            {[...Array(10)].map((_, i) => (
              <option key={i + 1} value={i + 1}>
                {i + 1}
              </option>
            ))}
          </select>

          <div className="button-container">
            <button className="add-to-cart" onClick={() => handleAddToCart(product)}>
              Add to Cart
            </button>
            <button className="go-to-cart" onClick={() => navigate("/cart")}>
              Go to Cart
            </button>
          </div>
        </div>
      ))}
    </div>
  </div>
);
};

export default Home;
```

## Output:

A. Index/Home page output:




## Code:

B. menu/product page:

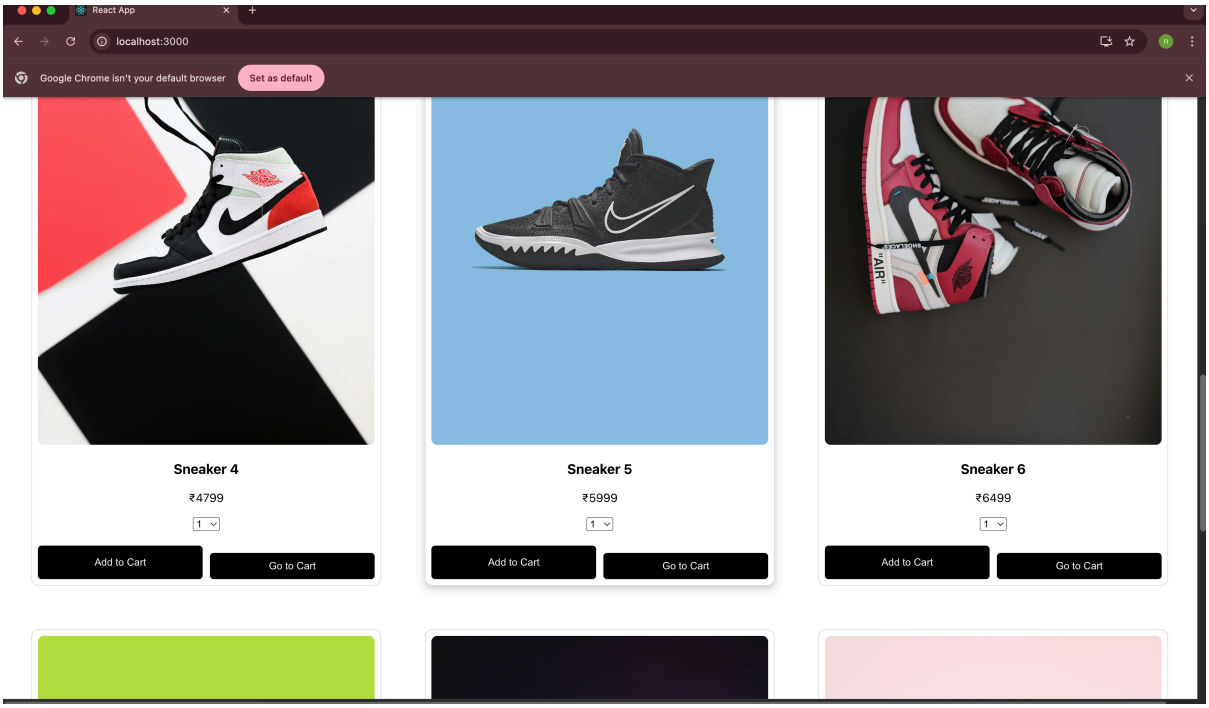
code:

```
import React from "react";
import "./Product.css"; // Make sure you have a Product.css file

const Product = ({ products, addToCart }) => {
  return (
    <div className="product-container">
      <h1>Our Products</h1>
      <p>Explore our latest sneakers collection!</p>
      <div className="product-grid">
        {products.map((product) => (
          <div className="product-box" key={product.id}>
            <img src={product.img} alt={product.name} className="product-image" />
            <h3>{product.name}</h3>
            <p>{product.price}</p>
            <button className="add-to-cart" onClick={() => addToCart(product)}>
```

```
          Add to Cart
        </button>

      </div>
    ))}
  </div>
</div>
);
};

export default Product;
```

**Output:**

B. menu/product page output:



**Code:**

C. cart page:

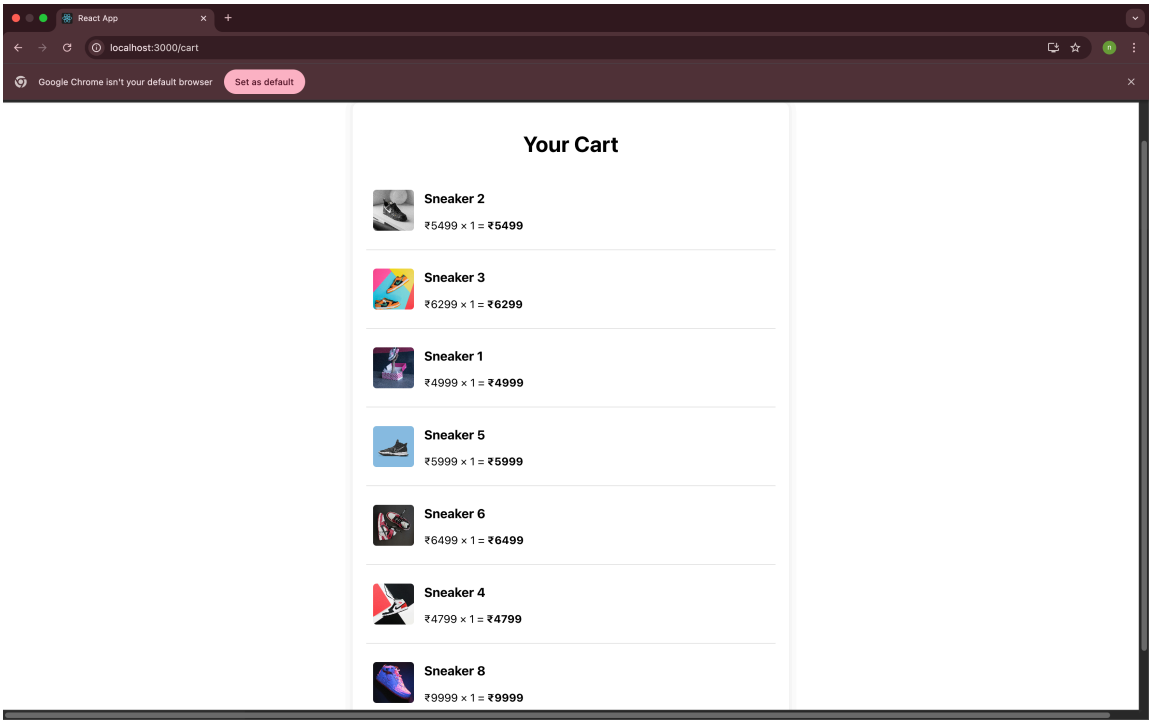
code:

```
import "./Cart.css";

const Cart = ({ cart }) => {
  const calculateTotal = () => {
    return cart.reduce((total, item) => {
      const priceStr = item.price ? String(item.price) : "0"; // Ensure it's a string
      const priceNumber = parseInt(priceStr.replace(/[₹,]/g, ""), 10) || 0; // Convert safely
      return total + priceNumber * (item.quantity || 1);
    }, 0);
  };

  return (
    <div className="cart-container">
      <h1>Your Cart</h1>
      {cart.length === 0 ? (
        <p className="empty-cart">Your cart is empty.</p>
      ) : (
        <ul className="cart-list">
          {cart.map((item) => {
            const priceStr = item.price ? String(item.price) : "0";
            const priceNumber = parseInt(priceStr.replace(/[₹,]/g, ""), 10) || 0;

            return (
              <li key={item.id} className="cart-item">
                <img src={item.img} alt={item.name} className="cart-image" />
                <div className="cart-details">
                  <h3>{item.name}</h3>
                  <p>
                    ₹{priceNumber} × {item.quantity || 1} = <strong>₹{priceNumber * (item.quantity || 1)}</
strong>
                  </p>
                </div>
              </li>
            );
          })}
        </ul>
      )}
      <h2 className="cart-total">Total: ₹{calculateTotal()}</h2>
    </div>
  );
```

```
};

export default Cart;
```

**Output:**

C. cart page output:



**Code:**

E. contact us page:

code:

```
const Contact = () => {
  return (
    <div>
      <h1>Contact Us</h1>
      <p>Email: support@sneakers.com</p>
      <p>Phone: +1 234 567 890</p>
    </div>
  );
};

export default Contact;
```

**Output:**

E. contact us page output:

**Code:**

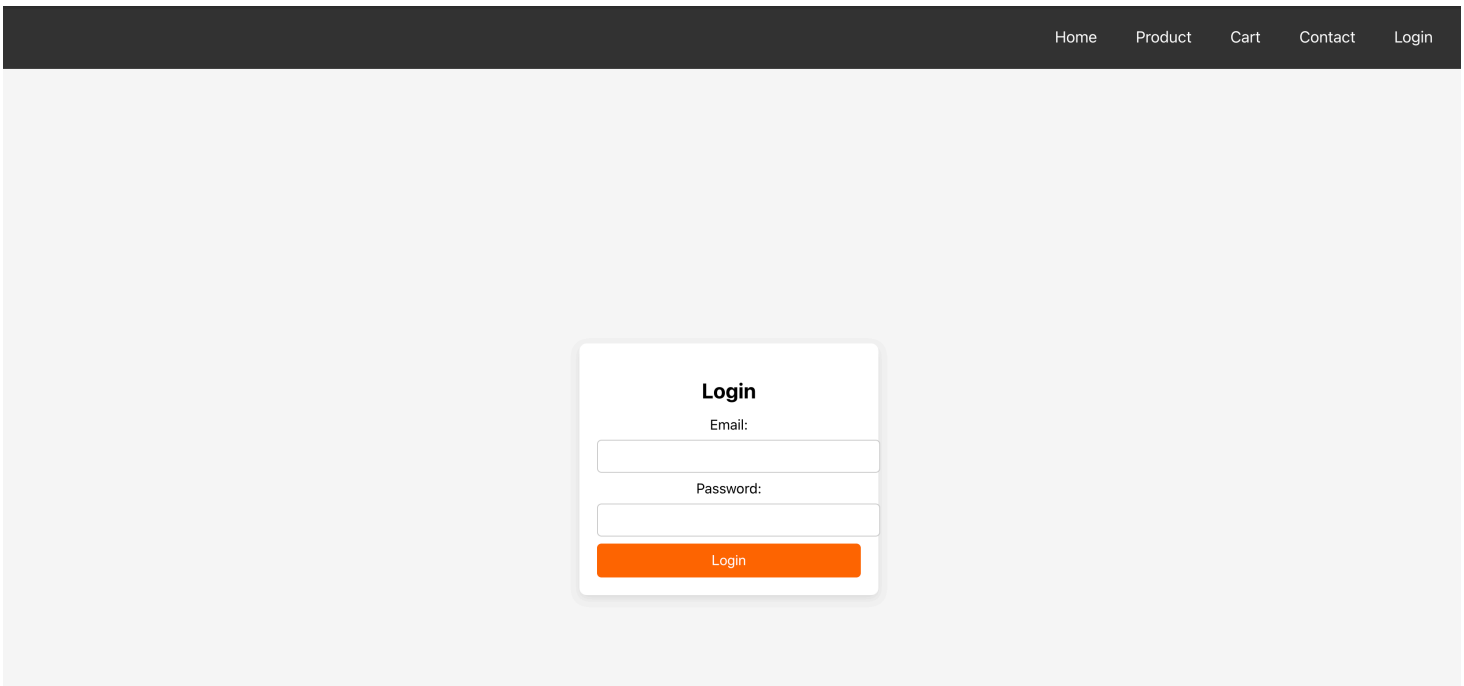
G. login page:

code:

```
import { useState } from "react";
import "./Login.css";

const Login = ({ setIsAuthenticated }) => { // Accept the prop
  const [formData, setFormData] = useState({ email: "", password: "" });

  const handleChange = (e) => {
    setFormData({ ...formData, [e.target.name]: e.target.value });
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    setIsAuthenticated(true); // Set authentication on login
    alert("Login Successful!");
  };

  return (
    <div className="login-container">
      <form className="login-form" onSubmit={handleSubmit}>
        <h2>Login</h2>
        <label>Email:</label>
        <input type="email" name="email" value={formData.email} onChange={handleChange} required />

        <label>Password:</label>
        <input type="password" name="password" value={formData.password}
onChange={handleChange} required />

        <button type="submit">Login</button>
      </form>
    </div>
  );
};

export default Login;
```

## Output:

G. login page output:




## Conclusion:

The **ReactJS E-Commerce Website with Page Routing** project successfully demonstrates the development of a real-world web application using modern frontend technologies. It provides a smooth user experience through **efficient routing**, **dynamic product display**, **cart management**, and **responsive design**. By implementing **React Router** and **Context API/ Redux**, the project ensures seamless navigation and state consistency across all pages. Overall, this project enhances understanding of component-based architecture, global state handling, and responsive UI development, preparing learners for advanced frontend development tasks.

# Experiment No.2

## Problem Statement:

In today's digital age, mobile shopping has become increasingly popular. However, many small businesses and users lack access to a smooth and responsive mobile shopping platform. There is a need for a **mobile application** that offers **easy product browsing**, **cart management**, and **secure checkout**, all while being fast, intuitive, and user-friendly. This project aims to solve that by developing a **React Native E-Commerce app using Expo** that replicates the core features of a web-based store in a mobile environment.

## Objective:

The objective of this project is to develop a **mobile E-Commerce application using React Native and Expo** that allows users to **browse products**, **view product details**, **manage their shopping cart**, **perform checkout**, and **view/update profile information**. The app aims to deliver a smooth, responsive, and real-time experience similar to modern e-commerce platforms, while utilizing **React Navigation** for routing and **Context API or Redux** for global state management.

## Theory:

This project is a **mobile E-Commerce application** developed using **React Native with Expo**. It includes essential screens such as **Home**, **Products**, **Product Details**, **Cart**, **Checkout**, and **Profile**, allowing users to interact with products in a mobile environment. The app uses **React Navigation** for seamless screen transitions, providing a native-like experience.

Products are fetched dynamically from a static JSON file or API and displayed with details like image, name, and price. Users can add items to the cart, view and update quantities, and proceed to checkout. The **Cart screen** summarizes items with total pricing and offers a smooth flow toward purchase.

The app also includes a **Profile screen**, enabling users to view and update their personal details and review past orders. To maintain consistency across screens, the project uses **React Context API or Redux** for state management, ensuring the cart state is persistent throughout the app. With **Expo**, the development and testing process is simplified, allowing faster iteration and deployment. The app is styled using **responsive UI design principles**, ensuring compatibility across various mobile devices and screen sizes.

**Code:**

A. Home page:

code:

```
import React from 'react';
import { StyleSheet, Text, View, Platform } from 'react-native';

export default function HomeScreen() {
  return (
    <View style={styles.container}>
      <Text style={styles.welcomeText}>Welcome to Sneakers-Shop</Text>
    </View>
  );
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#777',  // soft grey background
    justifyContent: 'center',
    alignItems: 'center',
    padding: 20,
  },
  welcomeText: {
    color: '#fff',
    fontSize: 36,
    fontWeight: 'bold',
    textAlign: 'center',
    fontFamily: Platform.OS === 'ios' ? 'Helvetica Neue' : 'sans-serif',
    // If you want a more aesthetic font, you can add custom fonts here
  },
});
```

## **Output**:

A. Index/Home page output:

**Code:**

B. menu/product page:

code:

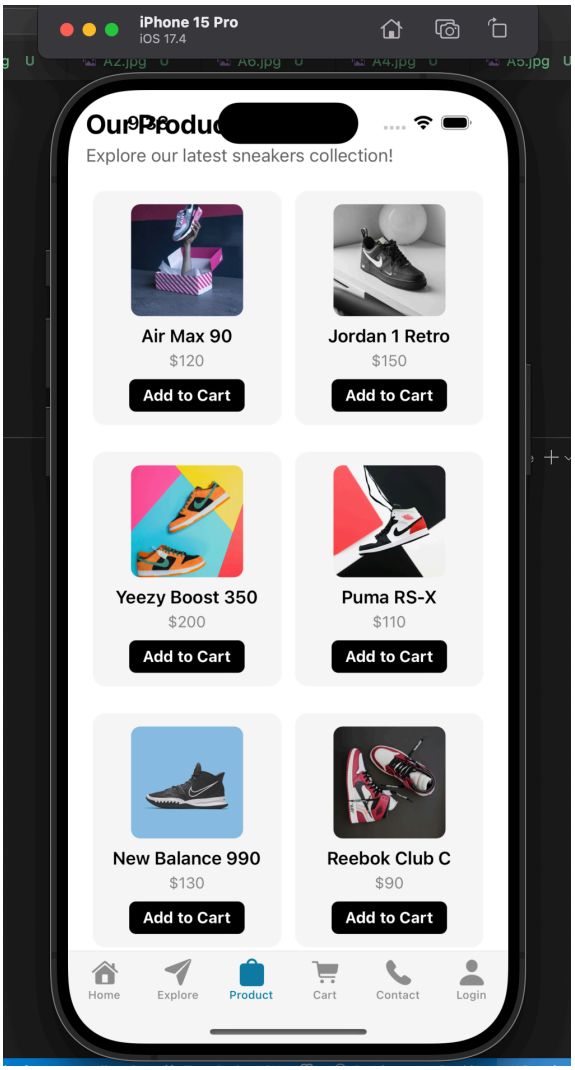
```
import React from 'react';
import { FlatList, Image, Pressable, StyleSheet, Text, View } from 'react-native';
import { useCart } from '../../context/CartContext';

const sampleProducts = [
  {
    id: '1',
    name: 'Air Max 90',
    price: '$120',
    img: require('../../assets/images/A1.jpg'),
  },
  {
    id: '2',
    name: 'Jordan 1 Retro',
    price: '$150',
    img: require('../../assets/images/A2.jpg'),
  },
  {
    id: '3',
    name: 'Yeezy Boost 350',
    price: '$200',
    img: require('../../assets/images/A3.jpg'),
  },
  {
    id: '4',
    name: 'Puma RS-X',
    price: '$110',
    img: require('../../assets/images/A4.jpg'),
  },
  {
    id: '5',
    name: 'New Balance 990',
    price: '$130',
    img: require('../../assets/images/A5.jpg'),
  },
  {
    id: '6',
    name: 'Reebok Club C',
    price: '$90',
    img: require('../../assets/images/A6.jpg'),
  },
];

const Product = () => {
```

```
  const { addToCart } = useCart();

  return (
    <View style={styles.container}>
      <Text style={styles.title}>Our Products</Text>
      <Text style={styles.subtitle}>Explore our latest sneakers collection!</Text>

      <FlatList
        data={sampleProducts}
        keyExtractor={(item) => item.id}
        numColumns={2}
        contentContainerStyle={styles.grid}
        renderItem={({ item }) => (
          <View style={styles.productBox}>
            <Image source={item.img} style={styles.productImage} />
            <Text style={styles.name}>{item.name}</Text>
            <Text style={styles.price}>{item.price}</Text>
            <Pressable
              style={styles.button}
              onPress={() =>
                addToCart({
                  ...item,
                  quantity: 1,
                })
              }
            >
              <Text style={styles.buttonText}>Add to Cart</Text>
            </Pressable>
          </View>
        )}
      />
    </View>
  );
};

export default Product;

const styles = StyleSheet.create({
  container: {
    padding: 16,
    flex: 1,
    backgroundColor: '#fff',
  },
  title: {
    fontSize: 24,
    fontWeight: 'bold',
    marginBottom: 4,
  },
  subtitle: {
    fontSize: 16,
```

```
  color: '#666',
  marginBottom: 16,
},
grid: {
  gap: 12,
},
productBox: {
  flex: 1,
  backgroundColor: '#f9f9f9',
  borderRadius: 12,
  padding: 12,
  margin: 6,
  alignItems: 'center',
},
productImage: {
  width: 100,
  height: 100,
  marginBottom: 8,
  borderRadius: 8,
},
name: {
  fontWeight: '600',
  fontSize: 16,
  marginBottom: 4,
},
price: {
  fontSize: 14,
  color: '#888',
  marginBottom: 8,
},
button: {
  backgroundColor: '#000',
  paddingVertical: 6,
  paddingHorizontal: 12,
  borderRadius: 6,
},
buttonText: {
  color: '#fff',
  fontWeight: 'bold',
},
});
```

**Output:**

B. menu/product page output:




C. cart page:

code:

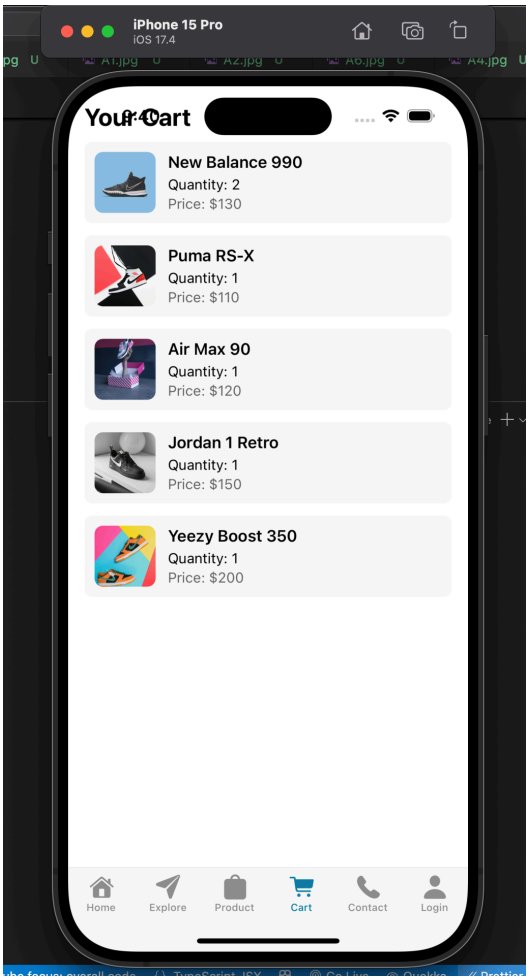
```
import React from 'react';
import { FlatList, Image, StyleSheet, Text, View } from 'react-native';
import { useCart } from '../../context/CartContext';

export default function CartScreen() {
  const { cart } = useCart();

  if (cart.length === 0) {
```

```
  return (
    <View style={styles.container}>
      <Text style={styles.title}>Your Cart</Text>
      <Text style={styles.subtitle}>You have no items in your cart yet.</Text>
    </View>
  );
}

return (
  <View style={styles.container}>
    <Text style={styles.title}>Your Cart</Text>
    <FlatList
      data={cart}
      keyExtractor={(item) => item.id}
      renderItem={({ item }) => (
        <View style={styles.cartItem}>
          <Image
            source={typeof item.img === 'string' ? { uri: item.img } : item.img}
            style={styles.cartImage}
            resizeMode="cover"
          />
          <View style={styles.cartDetails}>
            <Text style={styles.cartName}>{item.name}</Text>
            <Text style={styles.cartQuantity}>Quantity: {item.quantity}</Text>
            <Text style={styles.cartPrice}>Price: {item.price}</Text>
          </View>
        </View>
      )}
    />
  </View>
);
}



const styles = StyleSheet.create({
  container: { flex: 1, padding: 16, backgroundColor: '#fff' },
  title: { fontSize: 24, fontWeight: 'bold', marginBottom: 10 },
  subtitle: { fontSize: 16, color: 'gray' },
  cartItem: {
    flexDirection: 'row',
    marginBottom: 12,
    backgroundColor: '#f9f9f9',
    borderRadius: 8,
    padding: 10,
    alignItems: 'center',
  },
  cartImage: {
    width: 60,
    height: 60,
```

```
  borderRadius: 8,
  marginRight: 12,
 },
 cartDetails: {
  flex: 1,
 },
 cartName: {
  fontWeight: '600',
  fontSize: 16,
 },
 cartQuantity: {
  marginTop: 4,
  fontSize: 14,
 },
 cartPrice: {
  marginTop: 2,
  color: '#666',
 },
});
```

**Output:**

C. cart page output:



## Code:

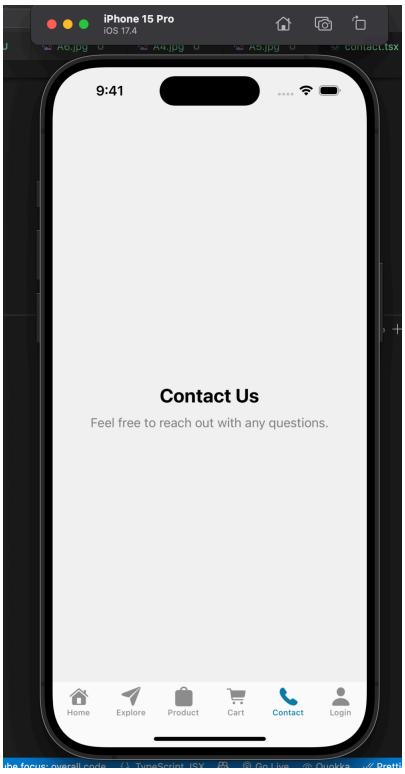
E. contact us page:

code:

```
import React from 'react';
import { StyleSheet, Text, View } from 'react-native';

export default function ContactScreen() {
  return (
    <View style={styles.container}>
      <Text style={styles.title}>Contact Us</Text>
      <Text style={styles.subtitle}>Feel free to reach out with any questions.</Text>
    </View>
  );
}

const styles = StyleSheet.create({
  container: { flex: 1, justifyContent: 'center', alignItems: 'center', padding: 16 },
  title: { fontSize: 24, fontWeight: 'bold', marginBottom: 10 },
  subtitle: { fontSize: 16, color: 'gray' },
});
```

## Output:

E. contact us page output:



**Code:**

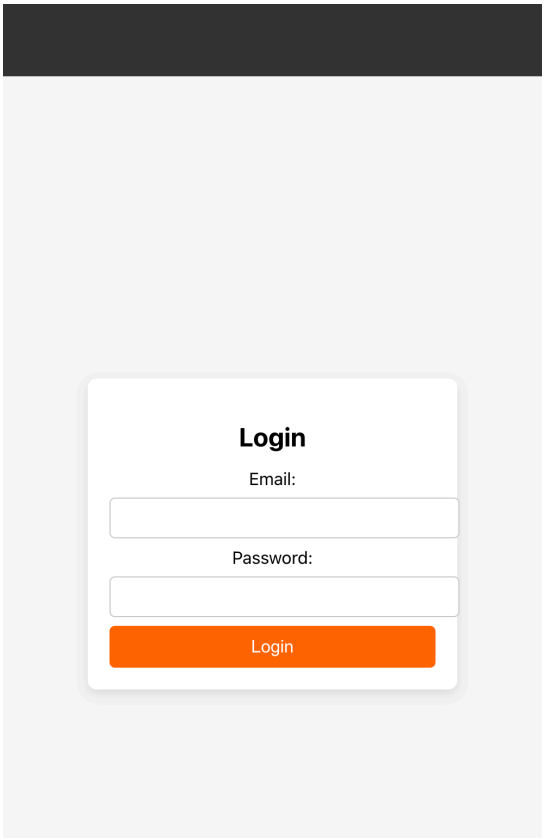
G. login page:

code:

```
import { useState } from "react";
import "./Login.css";

const Login = ({ setIsAuthenticated }) => { // Accept the prop
  const [formData, setFormData] = useState({ email: "", password: "" });

  const handleChange = (e) => {
    setFormData({ ...formData, [e.target.name]: e.target.value });
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    setIsAuthenticated(true); // Set authentication on login
    alert("Login Successful!");
  };

  return (
    <div className="login-container">
      <form className="login-form" onSubmit={handleSubmit}>
        <h2>Login</h2>

        <label>Email:</label>
        <input type="email" name="email" value={formData.email} onChange={handleChange} required />

        <label>Password:</label>
        <input type="password" name="password" value={formData.password}
onChange={handleChange} required />

        <button type="submit">Login</button>
      </form>
    </div>
  );
};

export default Login;
```

**Output:**

G. login page output:




**Conclusion:**

The E-Commerce mobile application built with **React Native and Expo** successfully replicates the key features of a modern shopping platform in a mobile environment. With smooth **navigation**, **dynamic product handling**, **cart management**, and **responsive UI**, it provides users with an intuitive and seamless shopping experience. By using **React Navigation** and **Context API/Redux**, the app maintains consistent state across screens and supports efficient performance. This project enhances skills in mobile development, state management, and UI design, and demonstrates the practical use of React Native in building real-world applications.