MIT-ADT UNIVERSITY
PUNE, INDIA
A Leap Towards World Class Education

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

## Department of Information Technology

# Lab Manual

## Practical -CPAD

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

## Batch - SAMD

### Name of the Student

**Mr.-Sujit Bhagwat**
**MITU23BTITD013**

**A.Y. 2024 – 2025 (SEM-VI)**

| Assignment No 1: Project Assignment based on Unit-I and Unit-II | |
|---|---|
| Project-I | **🎮 Project Title: GameZone**<br><br>**Tagline: A Gamer's One-Stop Destination to Buy the Latest Games**<br><br>---<br><br>**📝 Project Statement:**<br><br>**GameZone is an e-commerce web application built using ReactJS, specifically tailored for gaming enthusiasts. It offers a seamless experience for browsing, purchasing, and managing game products, ensuring a modern, fast, and responsive UI with intuitive navigation and state management.**<br><br>---<br><br>**🎯 Objective:**<br><br>**To develop a fully functional and responsive gaming-focused e-commerce platform using ReactJS that includes page routing, product management, shopping cart functionality, checkout flow, and user profile features, all powered by Context API or Redux for state management.**<br><br>---<br><br>**🚀 Core Features:**<br><br>**🧭 Navigation Bar with Routing:**<br><br>**A sticky top navigation bar using React Router with links to:**<br><br>• **Home: Highlights new & featured games.**<br>• **Products: Displays a listing of all available games.**<br>• **Cart: Shows items added to the cart with options to modify.**<br>• **Checkout: Users confirm purchase and enter shipping info.** |

- **Profile: User account page with profile editing and order history.**

---

🏠 **Home Page:**

- **Product carousel/banner showcasing best-selling or newly launched games.**
- **Search bar for quickly finding games.**
- **Highlight cards for categories like Action, RPG, Indie, etc.**
- **GameZone welcome message with a gamer-oriented UI aesthetic.**

---

🛍️ **Product Listing Page:**

- **Fetch games from a static JSON or mock API.**
- **Display each product with:**
  - **Game image.**
  - **Title.**
  - **Price.**
  - **⭐ Ratings (optional).**
  - **"Add to Cart" button.**
- **Use React Router links to navigate to the individual Product Details.**

---

📄 **Product Details Page:**

- **Large view of the selected game.**
- **Full product description, genre, platform compatibility.**
- **Price and quantity selector.**
- **"Add to Cart" functionality.**
- **Optional: Reviews & ratings section.**

---

🛒 **Cart Page:**

- **List of games added to the cart.**
- **Each item includes:**
  - **Thumbnail + game title.**
  - **Selected quantity (with ability to increase/decrease).**
  - **Price per unit and total cost.**
  - **"Remove" option.**
- **Display:**
  - **Subtotal and grand total.**
  - **"Continue Shopping" and "Proceed to Checkout" buttons.**

---

💳 **Checkout Page:**

- **User fills in:**
  - **Shipping address.**
  - **Contact details.**

- (Optional) Payment method.

- Review of order items and total price.
- "Place Order" button to confirm purchase.
- Order confirmation message or modal.

---

👤 **User Profile Page:**

- View user details:
  - Name, email, shipping address.
- Form to update profile.
- View past orders with game names, prices, and order status.

---

## 🧠 State Management (Context API or Redux):

- Global state includes:
  - Cart items.
  - Product data (if fetched).
  - User info (optional).
- Allows consistent access to cart data across all pages.
- Preserves cart data between routes and reloads.

---

## 📱 Responsive Design:

- Designed using CSS Flexbox and Grid to support:

| | |
|---|---|
| | ○ **Desktop: Full product details and grid views.**<br>○ **Mobile: Stacked UI, collapsible menus, and scrollable product cards.**<br>●<br>Expected Outcome:<br>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. |

## Experiment No.1

**Problem Statement:** GameZone

**Objective:**

To understand the fundamental building blocks of React.js and how to develop interactive, modular, and responsive UI components for a game-related application using React concepts such as components, state management, hooks, props, event handling, form controls, routing, and more.

**Theory:**

# 1. Functional Component

**Theory**:
Functional components are JavaScript functions that return React elements. They are simpler and preferred over class components due to the introduction of React Hooks.

**Syntax**:

jsx CopyEdit

```
function GameCard(props) {   return
<div>{props.title}</div>; }
```

Or using arrow function:

jsx CopyEdit

```
const GameCard = ({ title }) => <div>{title}</div>;
```

# 2. Class Component (Legacy)

**Theory**:
Class components were the original way to manage state and lifecycle methods in React. With Hooks, they are used less frequently.

**Syntax**:

jsx CopyEdit

```
class GameList extends React.Component {   render() {
            return <h1>Game List</h1>;
 } }
```

# 3. JSX (JavaScript XML)

**Theory**:
JSX is a syntax extension that looks similar to HTML and is used to describe what the UI should look like in React.

**Syntax**:

jsx CopyEdit

```
const element = <h1>Welcome to Game Zone</h1>;
```

# 4. Props (Properties)

**Theory**:

Props are used to pass data from one component to another. They are read-only and immutable in the child component.

**Syntax**:

jsx CopyEdit

```
function GameCard(props) {   return
<div>{props.name}</div>;
}
```

# 5. useState (React Hook)

**Theory**:
useState is a Hook that allows you to add state to functional components.

**Syntax**:

jsx CopyEdit

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

function GameCounter() {
  const [count, setCount] = useState(0);
  return <button onClick={() => setCount(count + 1)}>Play {count}</button>;
}
```

# 6. useEffect (React Hook)

**Theory**:
useEffect lets you perform side effects in functional components, such as data fetching or subscriptions.

**Syntax**:

jsx CopyEdit

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

useEffect(() => {
  console.log("Component mounted or updated");
}, []);
```

# 7. State

**Theory**:
State is a built-in object that stores property values that belong to a component. When the state changes, the component re-renders.

**Syntax** (using `useState`):

```
jsx CopyEdit

const [score, setScore] = useState(0);
```

# 8. Event Handling

**Theory**:
React handles events using camelCase syntax and functions.

**Syntax**:

```
jsx CopyEdit function StartGame() {   function
handleClick() {              alert("Game started!");
 }
 return <button onClick={handleClick}>Start</button>;
}
```

# 9. Conditional Rendering

**Theory**:
You can use JavaScript conditional operators to render elements dynamically.

**Syntax**:

```
jsx CopyEdit

function GameStatus({ isActive }) {
 return <div>{isActive ? "Game is On" : "Game is Off"}</div>;
}
```

# 10. Lists and Keys

**Theory**:

Rendering lists in React involves using `.map()` and providing a unique key for each item.

**Syntax**:

jsx CopyEdit

```
const games = ['Chess', 'Football', 'Basketball'];

const GameList = () => (
 <ul>
        {games.map((game, index) => <li key={index}>{game}</li>)}
 </ul> );
```

# 11. Component Composition

**Theory**:
React allows you to build components from other components, making UI modular and reusable.

**Syntax**:

```
jsx CopyEdit function Dashboard() {
return (
        <div>
        <Header />
        <GameList />
        <Footer />
        </div>
 ); }
```

# 12. Fetch API (Data Fetching)

**Theory**:

You can fetch data from an API in useEffect and store it in state.

**Syntax**:

```
jsx CopyEdit useEffect(() => {
 fetch('https://api.example.com/games')
        .then(res => res.json())
        .then(data => setGames(data));
}, []);
```

## 13. Routing (react-router-dom)

**Theory**:
React Router is used for navigating between different views or pages in a React application.

**Syntax**:

jsx CopyEdit

```
import { BrowserRouter, Routes, Route } from 'react-router-dom';
 function App() {   return (
<BrowserRouter>
            <Routes>
            <Route path="/" element={<Home />} />
            <Route path="/game/:id" element={<GameDetail />} />
            </Routes>
  </BrowserRouter>
 ); }
```

# 14. Context API (State Sharing)

**Theory**:
The Context API allows you to share state across many components without prop drilling.

**Syntax**:

jsx CopyEdit

```
const GameContext = React.createContext();

function App() {
 const [games, setGames] = useState([]);   return (
  <GameContext.Provider value={{ games, setGames }}>
            <GameList />
  </GameContext.Provider>
 ); }
```

# 15. Form Handling

**Theory**:
React provides controlled components to handle form inputs using state.

**Syntax**:
jsx CopyEdit function GameForm() {

```
 const [game, setGame] = useState("");
  const handleSubmit = (e) => {
   e.preventDefault();    console.log(game);
 };          return (
            <form onSubmit={handleSubmit}>
            <input value={game} onChange={(e) => setGame(e.target.value)} />
            <button type="submit">Add Game</button>
            </form>
 );
```

```
}
```

# 16. Functional Component

**Theory**: The building block of modern React apps. Functions that return JSX elements.

jsx CopyEdit

```
function GameCard(props) {   return
<div>{props.title}</div>;
}
```

# 17. Image Component

**Theory**: JSX allows rendering images using the `<img>` tag. Images can be local or remote.

jsx
CopyEdit

```
<img src="https://example.com/game.jpg" alt="Game Thumbnail" width="200" />
```

Or local:

jsx CopyEdit

```
import gameImage from './assets/game.jpg';
<img src={gameImage} alt="Game" />
```

# 18. Button Component

**Theory**: Buttons trigger actions or events, typically using event handlers like `onClick`.

jsx CopyEdit

```
<button onClick={() => alert('Game Started!')}>Start Game</button>
```

# 19. Text Field (Input)

**Theory**: Inputs are controlled components in React. Their values are synced with the component state.

jsx CopyEdit

```
const [username, setUsername] = useState('');
 <input   type="text"
value={username}
 onChange={(e) => setUsername(e.target.value)}    placeholder="Enter your name"
/>
```

# 20. Checkbox

**Theory**: Used for true/false values. React keeps the checkbox state in sync.

jsx CopyEdit

```
const [checked, setChecked] = useState(false);
 <input   type="checkbox"
checked={checked}
 onChange={() => setChecked(!checked)}
/> Enable Sound
```

# 21. Select Dropdown

**Theory**: Allows users to choose one value from a list of options.

jsx CopyEdit

```
const [difficulty, setDifficulty] = useState("easy");
<select value={difficulty} onChange={(e) => setDifficulty(e.target.value)}>
 <option value="easy">Easy</option>
 <option value="medium">Medium</option>
 <option value="hard">Hard</option>
</select>
```

# 22. TextArea

**Theory**: Multiline input box for longer text like feedback or comments.

jsx CopyEdit

```
<textarea
 rows="4"    cols="50"
value={feedback}
 onChange={(e) => setFeedback(e.target.value)}   placeholder="Leave your review..."
></textarea>
```

# 23. Card Component (Custom Layout)

**Theory**: Cards are used to group related content. Often styled with CSS.

jsx CopyEdit

```
function GameCard({ title, image }) {   return (
            <div className="card">
            <img src={image} alt={title} />
    <h3>{title}</h3>      <button>Play</button>
            </div>
 ); }
```

# 24. Container / Wrapper Component

**Theory**: Used to layout or wrap other components, often used for styling or structure.

jsx CopyEdit

```
function PageWrapper({ children }) {
  return <div className="page-wrapper">{children}</div>;
}
```

# 25. Navigation Bar

**Theory**: A component that holds navigation links using React Router or anchors.

jsx CopyEdit

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

function NavBar() {   return (
            <nav>
            <Link to="/">Home</Link> |
            <Link to="/games">Games</Link> |
            <Link to="/about">About</Link>
            </nav>
 ); }
```

# 26. Status Indicator (Badge or Label)

**Theory**: A simple visual indicator showing status or information.

jsx CopyEdit

```
function StatusBadge({ online }) {   return <span style={{ color: online ? 'green' : 'red' }}>{online ? 'Online' : 'Offline'}</span>;
}
```

# 27. Submit Form Button

**Theory**: Buttons inside forms to trigger submission. Paired with `onSubmit`.

```
jsx
CopyEdit
<form onSubmit={(e) => { e.preventDefault(); alert("Submitted"); }}>
  <input type="text" />
  <button type="submit">Submit</button>
</form>
```

# 28. Styled Component (Inline CSS Example)

**Theory**: Styling in React can be done inline using the `style` prop.

```
jsx CopyEdit

<button style={{ backgroundColor: 'blue', color: 'white' }}>
  Styled Button
</button>
```

# 29. Footer Component

**Theory**: A simple reusable UI piece placed at the bottom of a page.

```
jsx CopyEdit function Footer() {
  return <footer>&copy; 2025 Game Zone. All rights reserved.</footer>; }
```

# 30. Loading Spinner / Loader Component

**Theory**:
Loaders or spinners are used to show that content is being loaded asynchronously (e.g. fetching game data or images). This improves user experience by visually indicating that something is in progress.

**Syntax**:

```
jsx CopyEdit function Loader() {
```

return Loading... ; }

## Code:

A. Home page:

code:

```
import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';
import './Home.css';


const games = [
  {
    id: 1,
    title: 'Cyberpunk 2077',
    description: 'Futuristic open-world RPG.',
    price: '$59.99',
    image: 'https://images.igdb.com/igdb/image/upload/t_cover_big/co7497.webp'
  },
  {
    id: 2,
    title: 'Elden Ring',
    description: 'Dark fantasy action RPG.',
    price: '$69.99',
    image:
'https://th.bing.com/th/id/R.6eb59976f763b05a21fb206ad4a6ebe2?rik=RJHdVxmdgmwQKg&
pid=ImgRaw&r=0'
  },
  {
    id: 3,
    title: 'FIFA 24',
```

```
    description: 'The latest football simulation.',
    price: '$49.99',
    image:
'https://th.bing.com/th/id/OIP.Y7ddiXBTzshmoU41dM03RgHaLH?w=115&h=180&c=7&r=0&o=5
&cb=iwc2&dpr=1.1&pid=1.7'
  },
  {
    id: 4,
    title: 'GTA V',
    description: 'Crime, chaos, and an open world.',
    price: '$29.99',
    image:
'https://th.bing.com/th/id/R.173966c30ba346e61bf8d987d9cbebd2?rik=WRZsCdggRrhNKg&
riu=http%3a%2f%2f2.bp.blogspot.com%2f-XR0Rh3ZTni0%2fUKnAtCXxArI%2fAAAAAAAAAH8%2f7
vafmzl2Iss%2fs1600%2fgta5poster.jpg&ehk=GHYUtAzuyS%2f%2bCKSBCyICxPgbH0C%2bFbghzjU
i5NSq2kg%3d&risl=&pid=ImgRaw&r=0'
  },
  {
    id: 5,
    title: 'Minecraft',
    description: 'Create and explore your blocky world.',
    price: '$26.95',
    image:  'https://images.igdb.com/igdb/image/upload/t_cover_big/co8fu6.webp'
  },
  {
    id: 6,
    title: 'Call of Duty: MW2',
    description: 'Fast-paced military shooter.',
    price: '$59.99',
    image:  'https://wallpaperaccess.com/full/2204595.jpg'
  },
```

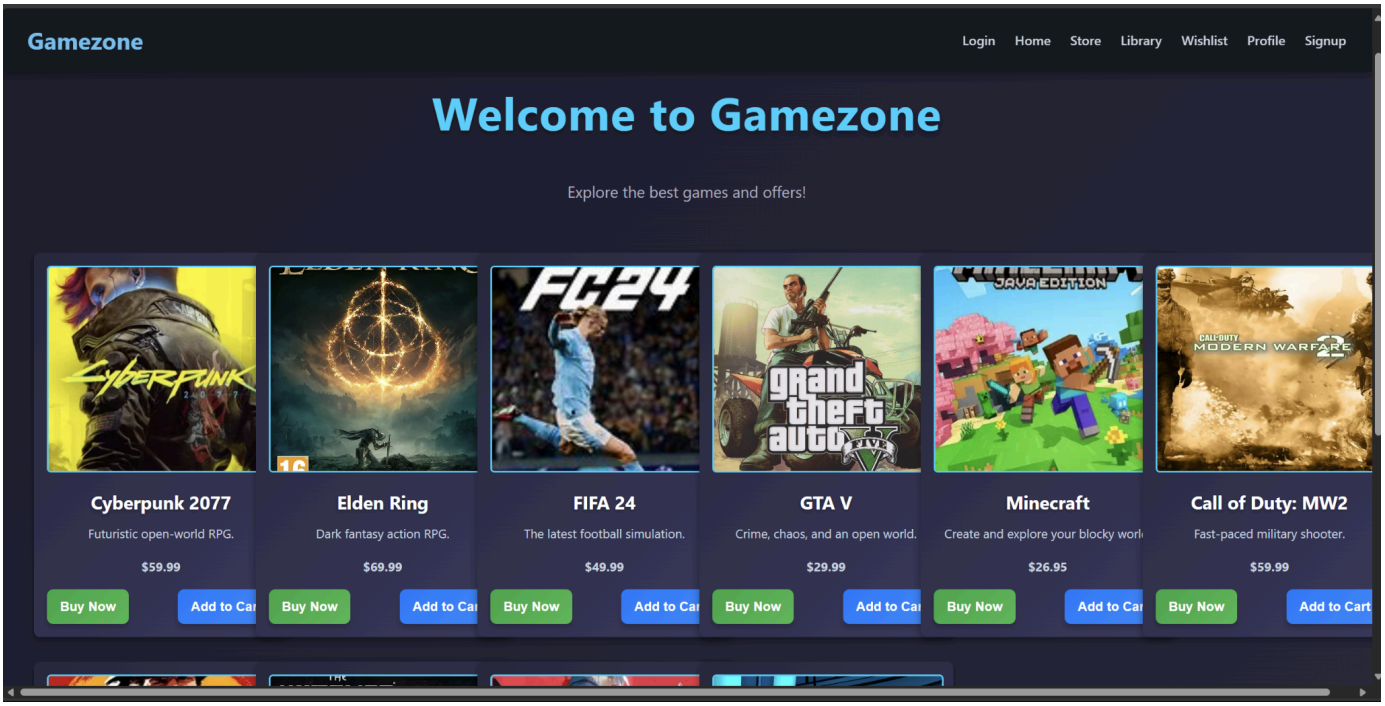
```
  {
    id: 7,
    title: 'Red Dead Redemption 2',
    description: 'Epic wild west adventure.',
    price: '$39.99',
    image: 'https://images.igdb.com/igdb/image/upload/t_cover_big/co1q1f.webp'
  },
  {
    id: 8,
    title: 'The Witcher 3',
    description: 'Monster hunting fantasy RPG.',
    price: '$29.99',
    image:
'https://cdn.mobygames.com/covers/907872-the-witcher-3-wild-hunt-new-quest-fools-gold-playstation-4-front.jpg'
  },
  {
    id: 9,
    title: 'Valorant',
    description: 'Team-based tactical shooter.',
    price: 'Free',
    image:  'https://images.igdb.com/igdb/image/upload/t_cover_big/co8ok7.webp'
  },
  {
    id: 10,
    title: 'Among Us',
    description: 'Multiplayer betrayal game.',
    price: '$4.99',
```

```
    image:
'https://assets.arpost.co/wp-content/uploads/2022/12/28181834/Among-US-VR-game.pn
g'

  }

];


function Home() {

  const [cart, setCart] = useState([]);

  const navigate = useNavigate();


  const addToCart = (game) => {

    setCart((prevCart) => [...prevCart, game]);

  };


  const goToCheckout = (game) => {

    navigate('/checkout', { state: { game } });

  };


  return (

    <div className="home">

      <h1>Welcome to Gamezone</h1>

      <p>Explore the best games and offers!</p>

      <div className="game-list">

        {games.map(game => (

          <div className="game-card" key={game.id}>

            <img src={game.image} alt={game.title} />

            <h3>{game.title}</h3>

            <p>{game.description}</p>

            <p className="price">{game.price}</p>
```

```
            <div className="game-buttons">

              <button className="buy-btn" onClick={() => goToCheckout(game)}>Buy
Now</button>

              <button className="cart-btn" onClick={() => addToCart(game)}>Add to
Cart</button>

            </div>

          </div>

        ))}

      </div>

    </div>

  );

}


export default Home;
```

## Output:

A. Index/Home page output:



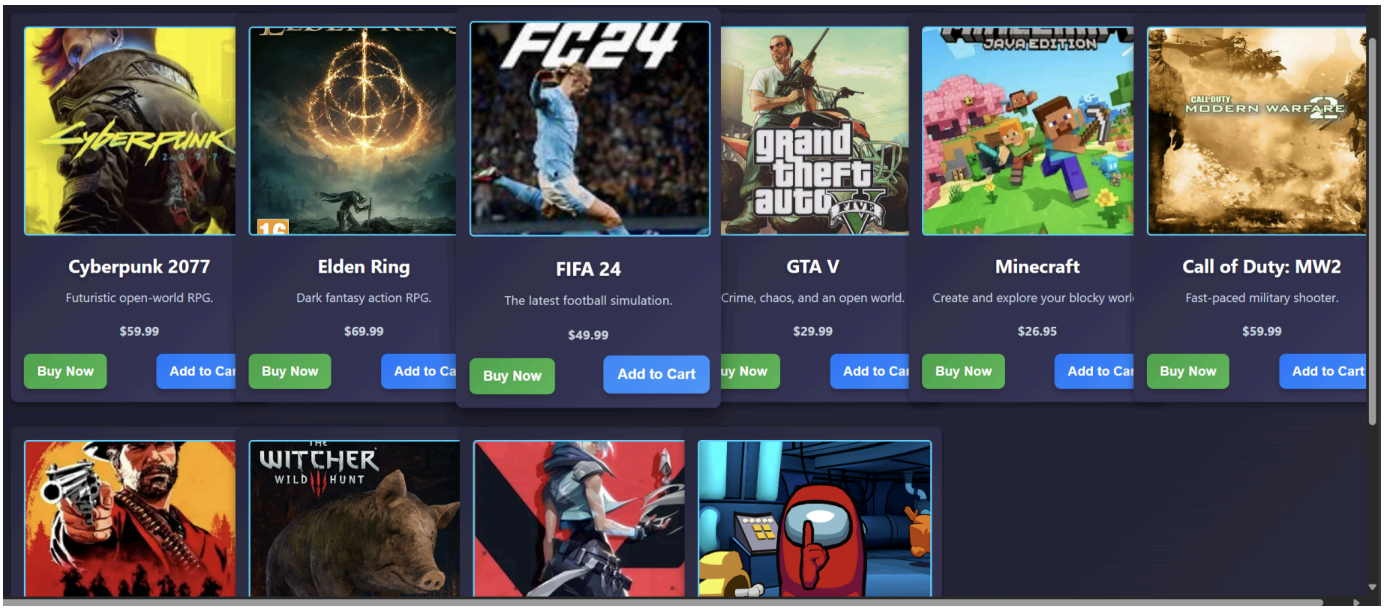

## Code:

B. menu/product page:

code:

## Output:

B. menu/product page output:



## Code:

C. Whishlist:-

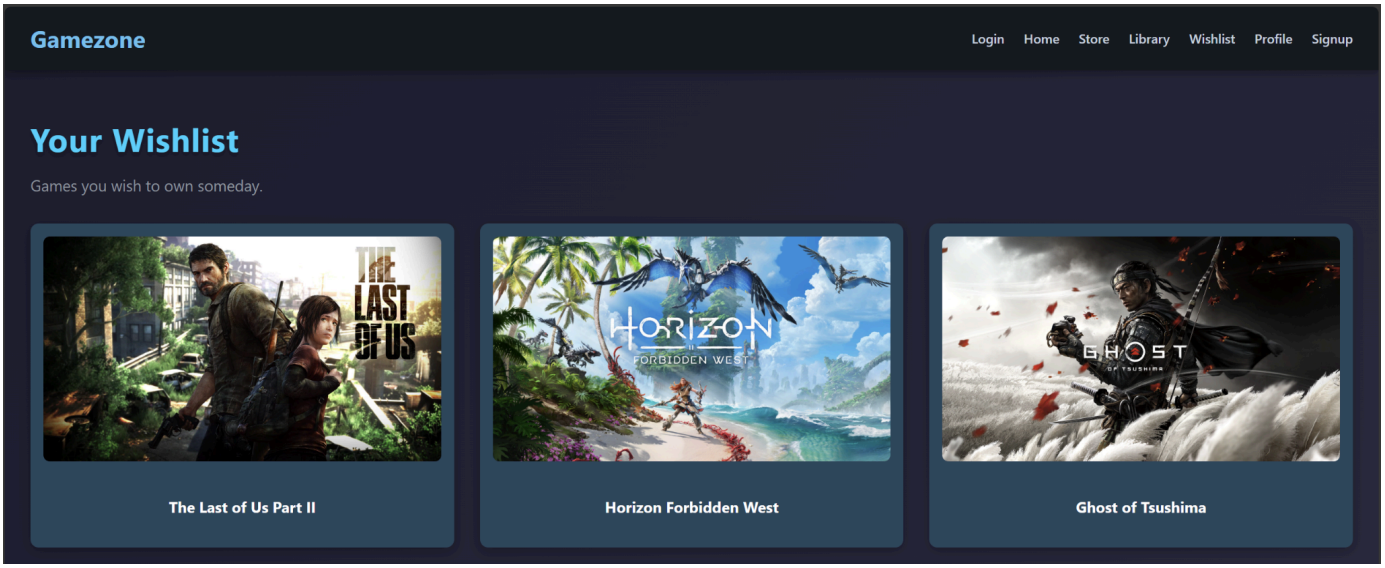
code:

```
import React from 'react';
import './Wishlist.css';


const Wishlist = () => {
  const wishlistGames = [
    {
      id: 1,
      title: 'The Last of Us Part II',
      image: 'https://eskipaper.com/images/the-last-of-us-wallpaper-7.jpg'
    },
    {
      id: 2,
      title: 'Horizon Forbidden West',
      image:
'https://gameranx.com/wp-content/uploads/2022/01/horizon-forbidden-west-1-scaled.
jpg'
    },
    {
      id: 3,
      title: 'Ghost of Tsushima',
      image:
'https://gmedia.playstation.com/is/image/SIEPDC/ghost-of-tsushima-master-image-en
-24jun21?$native$'
    }
  ];


  return (
```

```
    <div className="wishlist-page">
      <h1>Your Wishlist</h1>
      <p>Games you wish to own someday.</p>
      <div className="wishlist-grid">
        {wishlistGames.map(game => (
          <div className="wishlist-card" key={game.id}>
            <img src={game.image} alt={game.title} />
            <h3>{game.title}</h3>
          </div>
        ))}
      </div>
    </div>
  );
};

export default Wishlist;
```

## Output:

C. Wishlist output:




## Code:

D. Profile:

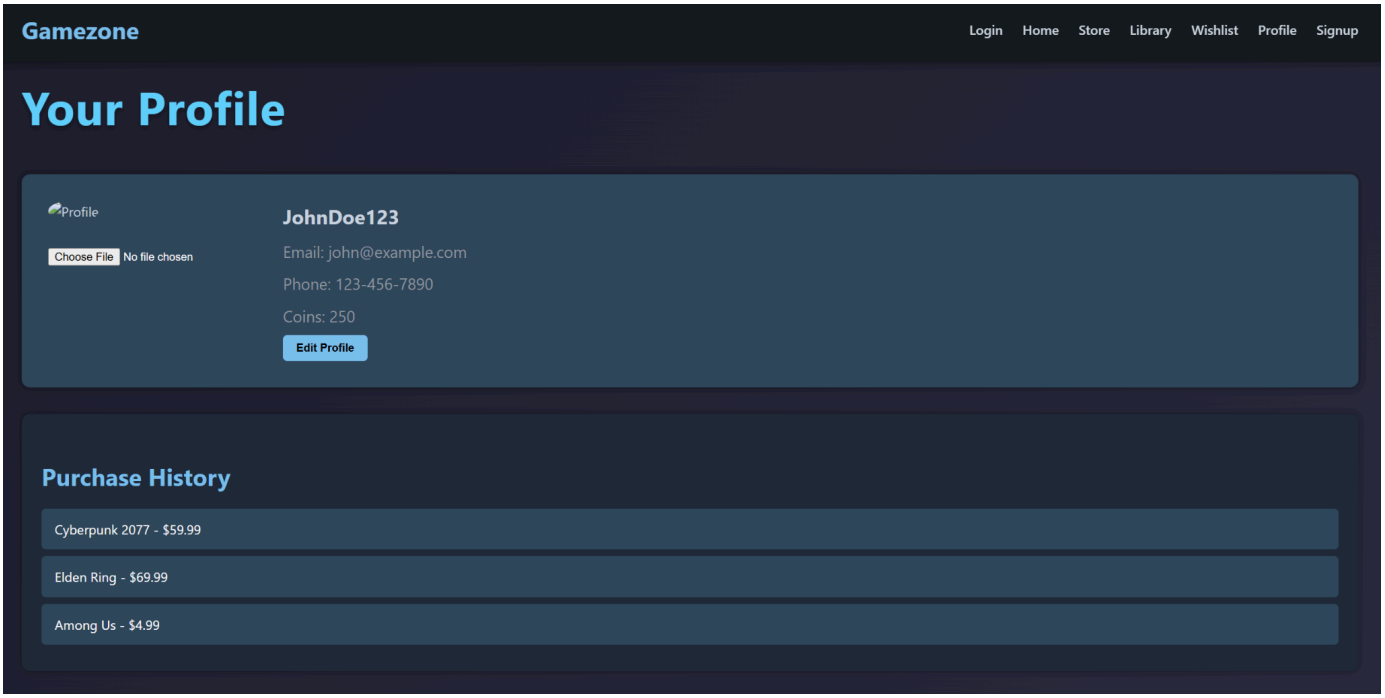
code:

```
import React, { useState } from 'react';
import './Profile.css';


function Profile() {
  const [profilePic, setProfilePic] = useState(null);
  const [username, setUsername] = useState('JohnDoe123');
  const [email, setEmail] = useState('john@example.com');
  const [phone, setPhone] = useState('123-456-7890');
  const [coins, setCoins] = useState(250);
  const [editing, setEditing] = useState(false);


  const purchaseHistory = [
    'Cyberpunk 2077 - $59.99',
    'Elden Ring - $69.99',
    'Among Us - $4.99',
  ];
```

```
  const handleImageUpload = (e) => {
    const file = e.target.files[0];
    if (file) {
      setProfilePic(URL.createObjectURL(file));
    }
  };

  return (
    <div className="profile-page">
      <h1>Your Profile</h1>
      <div className="profile-card">
        <div className="profile-image">
          <img src={profilePic || 'https://via.placeholder.com/150'}
alt="Profile" />
          <input type="file" onChange={handleImageUpload} />
        </div>
        <div className="profile-details">
          {editing ? (
            <>
              <input type="text" value={username} onChange={(e) =>
setUsername(e.target.value)} placeholder="Username" />
              <input type="email" value={email} onChange={(e) =>
setEmail(e.target.value)} placeholder="Email" />
              <input type="text" value={phone} onChange={(e) =>
setPhone(e.target.value)} placeholder="Phone Number" />
              <button onClick={() => setEditing(false)}>Save</button>
            </>
          ) : (
            <>
              <h2>{username}</h2>
```

```
              <p>Email: {email}</p>
              <p>Phone: {phone}</p>
              <p>Coins: {coins}</p>
              <button onClick={() => setEditing(true)}>Edit Profile</button>
            </>
          )}
        </div>
      </div>

      <div className="purchase-history">
        <h3>Purchase History</h3>
        <ul>
          {purchaseHistory.map((item, index) => (
            <li key={index}>{item}</li>
          ))}
        </ul>
      </div>
    </div>
  );
}

export default Profile;
```

**Output:**




## Code:

E. Login-:

code:

```
import { useState } from 'react';
import { useNavigate } from 'react-router-dom';
import './Login.css';


const Login = () => {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const navigate = useNavigate();


  // Login function
  const login = (email, password) => {
    const users = JSON.parse(localStorage.getItem('users')) || [];
```

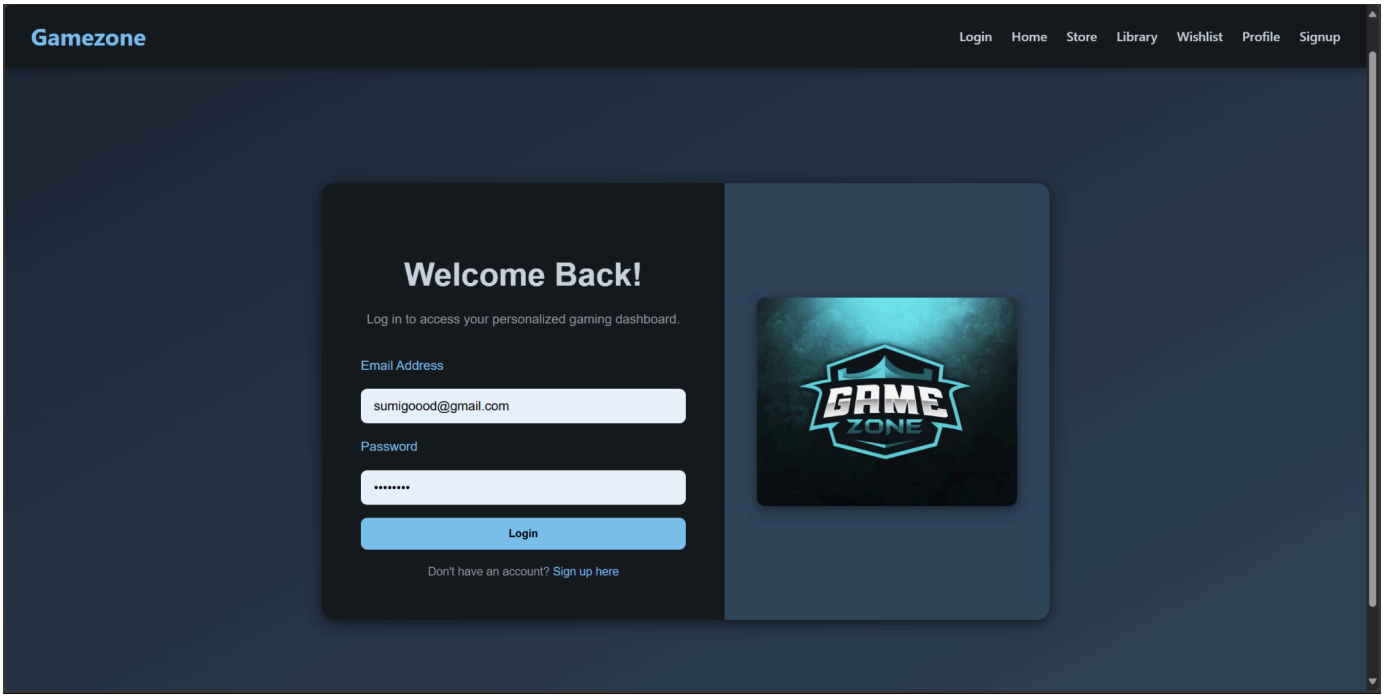
```
    const user = users.find((u) => u.email === email && u.password === password);

    if (user) {
      localStorage.setItem('authUser', JSON.stringify(user));
      return true;
    } else {
      alert('Invalid email or password');
      return false;
    }
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    if (login(email, password)) {
      navigate('/'); // Redirect to home after successful login
    }
  };

  return (
    <div className="login-container">
      <div className="login-card">
        <div className="login-content">
          <h2 className="login-title">Welcome Back!</h2>
          <p className="login-subtext">Log in to access your personalized gaming
dashboard.</p>

          <form onSubmit={handleSubmit} className="login-form">
            <label htmlFor="email">Email Address</label>
            <input
```

```
            id="email"
            type="email"
            placeholder="Enter your email"
            value={email}
            onChange={(e) => setEmail(e.target.value)}
            required
          />

          <label htmlFor="password">Password</label>
          <input
            id="password"
            type="password"
            placeholder="Enter your password"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
            required
          />

          <button type="submit" className="login-button">Login</button>
        </form>

        <div className="login-footer">
          Don't have an account?{' '}
          <a href="/signup">Sign up here</a> {/* Link to signup page */}
        </div>
      </div>

      <div className="login-image">
```

```
          <img

src="https://fiverr-res.cloudinary.com/images/t_main1,q_auto,f_auto,q_auto,f_auto
/gigs/116965796/original/0d1b0d77c52830452fdf3ba75a74fa24e31520a0/do-minimalist-l
ogo-design-for-your-business.jpg"

            alt="Gaming"

          />

        </div>

      </div>

    </div>

  );

};


export default Login;
```

## Output:

E. Login output:



## Code:

F. registration page:

code:

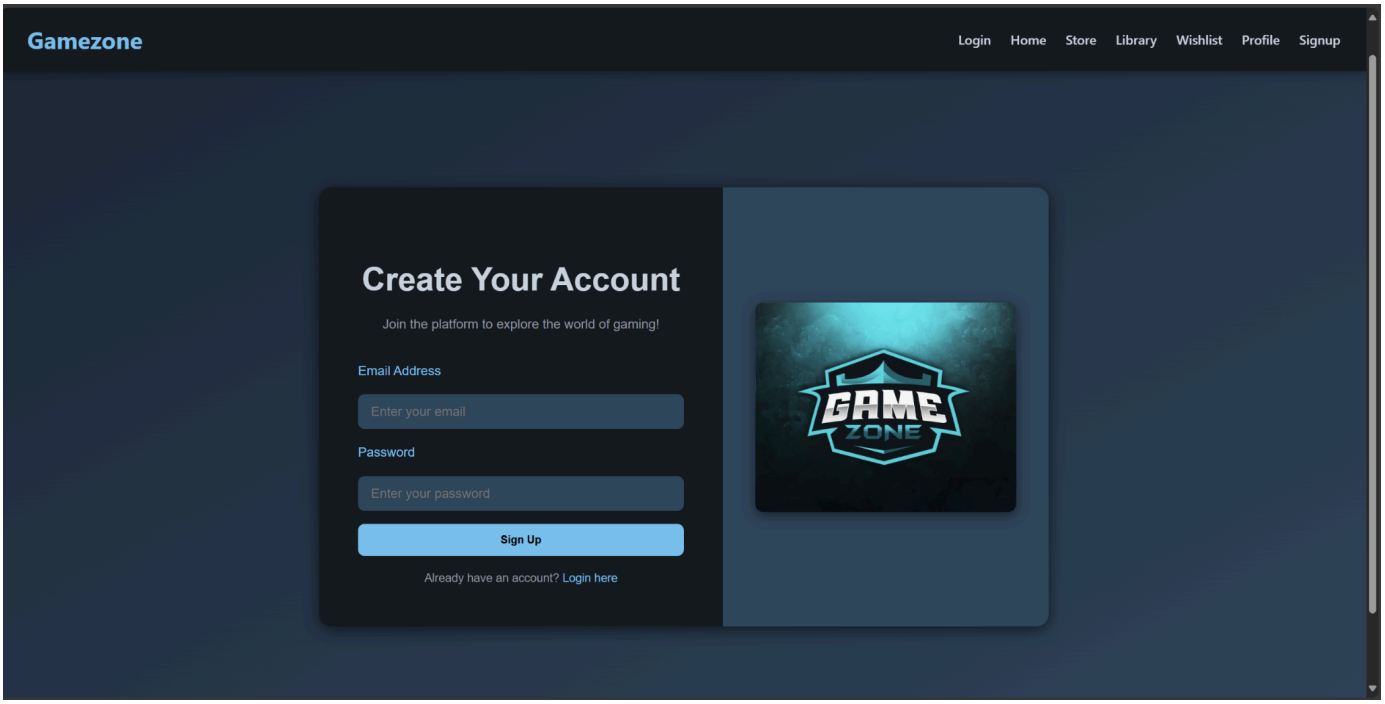
```
import { useState } from 'react';
import { useNavigate } from 'react-router-dom';
import './Signup.css';


const Signup = () => {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const navigate = useNavigate();


  // Signup function directly in the component
  const signup = (email, password) => {
    const users = JSON.parse(localStorage.getItem('users')) || [];
    const userExists = users.some((u) => u.email === email);


    if (userExists) {
      alert('Email already exists');
      return false;
    }


    const newUser = { email, password };
    users.push(newUser);
    localStorage.setItem('users', JSON.stringify(users));
    localStorage.setItem('authUser', JSON.stringify(newUser));
    return true;
  };
```

```
  const handleSubmit = (e) => {
    e.preventDefault();
    if (signup(email, password)) {
      navigate('/'); // Redirect to home after successful signup
    }
  };

  return (
    <div className="signup-container">
      <div className="signup-card">
        <div className="signup-content">
          <h2 className="signup-title">Create Your Account</h2>
          <p className="signup-subtext">Join the platform to explore the world of gaming!</p>

          <form onSubmit={handleSubmit} className="signup-form">
            <label htmlFor="email">Email Address</label>
            <input
              id="email"
              type="email"
              placeholder="Enter your email"
              value={email}
              onChange={(e) => setEmail(e.target.value)}
              required
            />

            <label htmlFor="password">Password</label>
            <input
```

```
              id="password"
              type="password"
              placeholder="Enter your password"
              value={password}
              onChange={(e) => setPassword(e.target.value)}
              required
            />

            <button type="submit" className="signup-button">Sign Up</button>
          </form>

          <div className="signup-footer">
            Already have an account? <a href="/login">Login here</a>
          </div>
        </div>

        <div className="signup-image">
          <img
src="https://fiverr-res.cloudinary.com/images/t_main1,q_auto,f_auto,q_auto,f_auto
/gigs/116965796/original/0d1b0d77c52830452fdf3ba75a74fa24e31520a0/do-minimalist-l
ogo-design-for-your-business.jpg" alt="Join Gaming" />
        </div>
      </div>
    </div>
  );
};

export default Signup;
```

## Output:

F. registration page output:




## Code:

G. Library page:

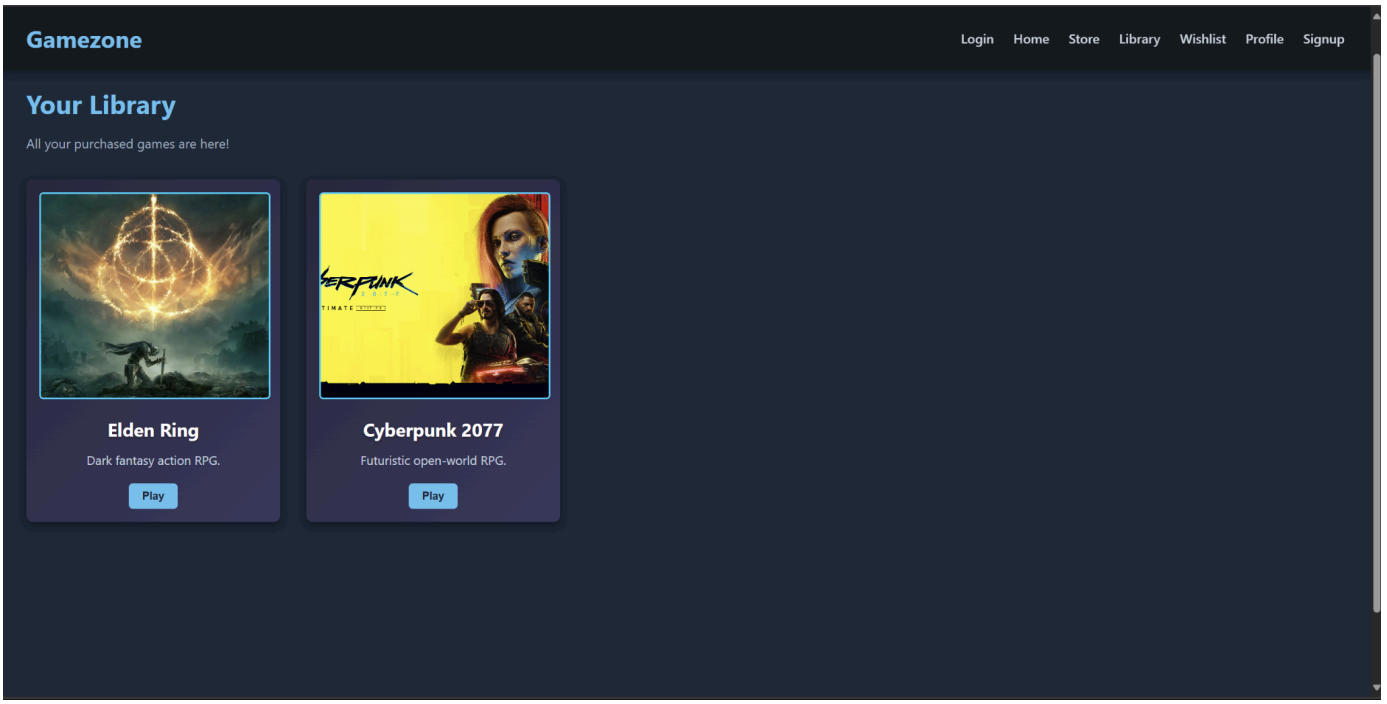
code:

```
// Removed unused React import

import './Library.css';


const purchasedGames = [

  {

    id: 1,

    title: 'Elden Ring',

    description: 'Dark fantasy action RPG.',

    image:
'https://fond-decran.com/wp-content/uploads/2022/04/elden-ring-wallpaper.jpg'

  },
```

```
  {
    id: 2,
    title: 'Cyberpunk 2077',
    description: 'Futuristic open-world RPG.',
    image:
'https://4kwallpapers.com/images/wallpapers/cyberpunk-2077-2880x1800-13543.jpg'
  }
];


function Library() {
  return (
    <div className="library">
      <h2>Your Library</h2>
      <p>All your purchased games are here!</p>
      <div className="library-games">
        {purchasedGames.map(game => (
          <div className="game-card" key={game.id}>
            <img src={game.image} alt={game.title} />
            <h3>{game.title}</h3>
            <p>{game.description}</p>
            <button className="play-button">Play</button>
          </div>
        ))}
      </div>
    </div>
  );
}


export default Library;
```

## Output:

G. Library page output:




## Conclusion:

This comprehensive React guide covers 30 essential topics ranging from core concepts like functional and class components to advanced features like the Context API and routing. By exploring real-world examples such as building a GameCard, implementing form handling, using hooks like useState and useEffect, and creating UI components like navbars, buttons, cards, and loaders, developers are equipped with practical knowledge to build modern, scalable React applications. Emphasis on modular design, user interactivity, and performance ensures a strong foundation for any front-end development project using React.