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

Department of Information Technology

SUBJECT – FULL STACK FRONT END DEVELOPMENT

COURSE CODE – 23IT3001

Class – T.Y. SMAD

Name of Student

Prathmesh Waghmare – ADT23SOCB1545

Under the Guidance of

Prof. Rohini Bhosale

A.Y. 2025-2026 (SEM – I)

Assignment 1 : React Project Setup and Navigation

Aim :

To create a React project using create-react-app, set up routing using react-router-dom, and build a top navigation bar with links to different pages such as Home, Products, Cart, Checkout, and Profile.

Theory :

- **Introduction to React**

  React is an open-source JavaScript library developed by Facebook for building user interfaces, especially for single-page applications. It allows developers to create reusable UI components, manage data efficiently, and render the user interface dynamically without reloading the entire page. It follows a component-based architecture, meaning that every part of the UI is broken down into small, independent, and reusable components.
  React uses a virtual DOM (Document Object Model) which enhances performance by updating only the parts of the web page that have changed, rather than re-rendering the whole page.

- **React Project Setup using create-react-app**

  To begin working with React, developers use the create-react-app tool. It is an officially supported way to set up a new React project quickly without configuring build tools like Webpack or Babel manually.

  Key Features of create-react-app:

  1. Automatically configures the project structure.
  2. Includes a development server with live reloading.
  3. Simplifies dependency management.
  4. Provides built-in support for JSX and ES6 syntax.

  After running the setup command, the tool generates all the necessary files and dependencies required for a React environment.

- **React Router and Single Page Applications (SPA)**

  In traditional websites, each link click results in a new page request to the server. React, however, uses Single Page Application (SPA) architecture, where only one HTML page is loaded, and the content changes dynamically using JavaScript.
  To handle multiple views or pages within an SPA, React developers use a library called react-router-dom.
  React Router enables navigation between different components in the application without refreshing the page. It provides components like:

1. BrowserRouter: Wraps the whole application and keeps the UI in sync with the URL.
2. Routes and Route: Define which component should render for a specific path.
3. Link and NavLink: Used to navigate between pages without reloading.

- **Navigation Bar (Navbar)**

  A Navigation Bar is an essential user interface element that allows users to move between different sections of the application easily.

  In a React application, the navigation bar is typically created as a separate reusable component. It includes several navigation links (using the Link component from React Router) that point to different routes defined in the application.

  Features of a Good Navigation Bar:

  1. It is consistent across all pages.

  2. It provides clear and visible links to major sections.

  3. It is responsive, meaning it adjusts to different screen sizes.

  4. It often includes a brand logo or title on the left and menu options on the right.

- **Components and Routing Structure**

  In a React project, each webpage or screen is generally represented as a component. Examples of such components include Home, Products, Cart, Checkout, and Profile. Each of these components is connected using React Router routes.

  When the user clicks a link in the navigation bar, the URL changes and the corresponding component is rendered — all without reloading the page.

  This creates a smooth and seamless navigation experience for the user.

- **Working of Navigation in React**

  1. The application is wrapped inside a **Router** (BrowserRouter).
  2. Each page or screen is defined as a **Route**.
  3. The **Navbar component** contains clickable links to these routes.
  4. When a link is clicked, React Router intercepts the navigation event and updates the view accordingly.
  5. The **current component** associated with the selected route is displayed dynamically.

  This approach provides faster navigation, better user experience, and efficient content loading.

- **Advantages of Using React Router and Navigation**

  1. Improved User Experience: Page transitions are instant without full reloads.

2. SEO Friendly: Routes can be defined clearly with specific URLs.
3. Component-Based Organization: Each route corresponds to a specific component, making maintenance easier.
4. Scalability: New pages and routes can be added easily as the application grows.
5. Performance: Navigation happens on the client-side, reducing server load.

Code :

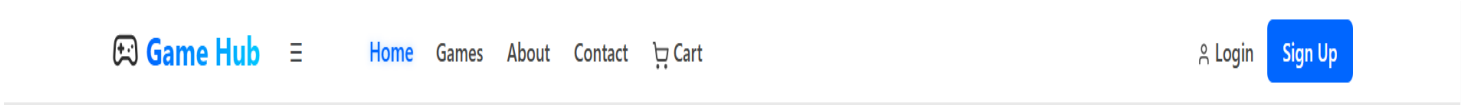
```
npm create vite@latest

create-app npm install

cd

cre

ate-

app

np

m

run

dev
```

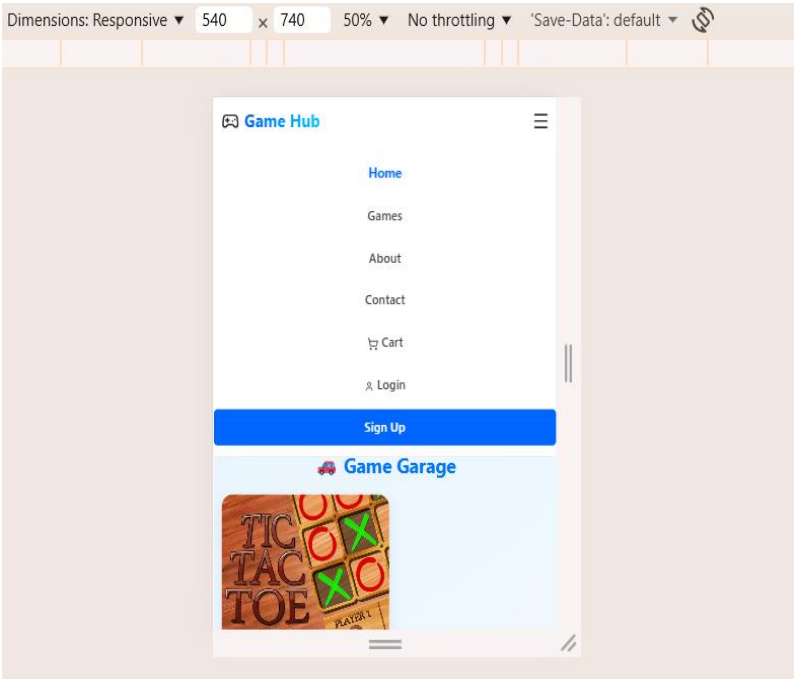
```
import React, { useState } from "react";
import { Link, NavLink } from "react-
router-dom";
import { Gamepad2, ShoppingCart, User }
from "lucide-react";

export default function Navbar() {
  const [open, setOpen] = useState(false);

  return (
    <nav className="navbar-container">
      <div className="nav-inner
container">

        {/* Logo */}
        <Link to="/" className="nav-logo">
          <Gamepad2 size={28} />
<span>Game Hub</span>
        </Link>

        {/* ✅ Mobile Menu Button */}
        <div className="menu-toggle" onClick={() =>
setOpen(!open)}>
          ☰
        </div>

        {/* Menu */}
        <div className={`nav-links ${open ? "active" :
""}`}>
          <NavLink to="/" className="nav-
item">Home</NavLink>
          <NavLink to="/games" className="nav-
item">Games</NavLink>
          <NavLink to="/about" className="nav-
item">About</NavLink>
          <NavLink to="/contact" className="nav-
item">Contact</NavLink>
          <NavLink to="/cart" className="nav-item
cart-btn">
            <ShoppingCart size={18} /> Cart
          </NavLink>
        </div>

        {/* Auth Buttons */}
        <div className={`auth-section ${open ?
"active" : ""}`}>
          <Link to="/login" className="nav-login">
            <User size={15}/> Login
          </Link>
          <Link to="/register" className="nav-
register">
            Sign Up
          </Link>
        </div>
        {/* Mobile Dropdown */}
       <div className={`mobile-menu ${open ?
"active" : ""}`}>
        <NavLink to="/" className="nav-
item">Home</NavLink>
        <NavLink to="/games" className="nav-
```

```
item">Games</NavLink>
      <NavLink to="/about"
className="nav-item">About</NavLink>
      <NavLink to="/contact"
className="nav-
item">Contact</NavLink>
      <NavLink to="/cart" className="nav-
item">
      <ShoppingCart size={18} /> Cart
      </NavLink>
```

```
      <Link to="/login" className="nav-login"><User
size={15}/> Login</Link>
      <Link to="/register" className="nav-
register">Sign Up</Link>
      </div>

     </div>
   </nav>
  );
}
```

Output :





Conclusion :

We successfully implement Navbar using jsx.

Assignment 2 : Home Page Development

Aim :

To design and develop the Home Page layout of a Game Hub React application that includes featured games and a functional search bar that filters games by name or level category.

Theory :

- **Introduction**

  The Home Page is the first and most important part of any web application. It provides an overview of the application's purpose, displays key features or game categories, and helps users easily navigate to other sections.

  In this assignment, the Home Page is developed using React components, Bootstrap, and custom CSS, and includes dynamic rendering of featured games.

- **Concept of Components in React**

  React follows a component-based architecture, meaning the UI is built using reusable and independent pieces called components.

  In this assignment:

  Home.jsx is the main page component displaying all games.

  Instead of using a separate sub-component (like GameCard), the game card design, game data, and search logic are all implemented within the Home.jsx file.

  Each component has its own logic and UI, allowing separation of concerns and easier maintenance.

- **Designing the Home Page Layout**

  A Game Hub Home Page layout typically includes:

  1. A banner or header section to welcome the user.
  2. A section displaying all featured and available games.
  3. A search bar to help users quickly find games by name or level.
  4. Proper alignment and spacing using Bootstrap classes such as container, row, and col.

  This helps in creating a responsive and clean layout suitable for all devices.

- **Featured Games or Levels**

  The featured games section highlights key games and difficulty levels to attract user interest.

  In React, this is implemented by maintaining a list (array) of games inside the component's state.

  Each game contains attributes such as:

  - id

  - name

  - level (Beginner / Intermediate / Advanced)

  - rating

  - image
- These games are dynamically displayed using the `.map()` function, ensuring scalability — new games can be added easily without changing the UI code.

- **Search Bar Functionality**

  A search bar is an essential feature that allows users to filter products based on their input.

  In React, this is implemented using:

  - State variables to store the search term.

  - The onChange event handler to update the state whenever the user types.

  - he filter() method to match user input with game name or level.
- This enables real-time filtering — search results update instantly as the user types, improving interactivity and user experience.

- **Styling and Responsiveness**

  Bootstrap and CSS are used to style the page:

  - Containers and grid layouts organize products in rows and columns.

  - The navigation bar remains fixed.

  - Margins and padding ensure spacing between elements.

  - Responsive design ensures the layout adapts to different screen sizes.

Code :

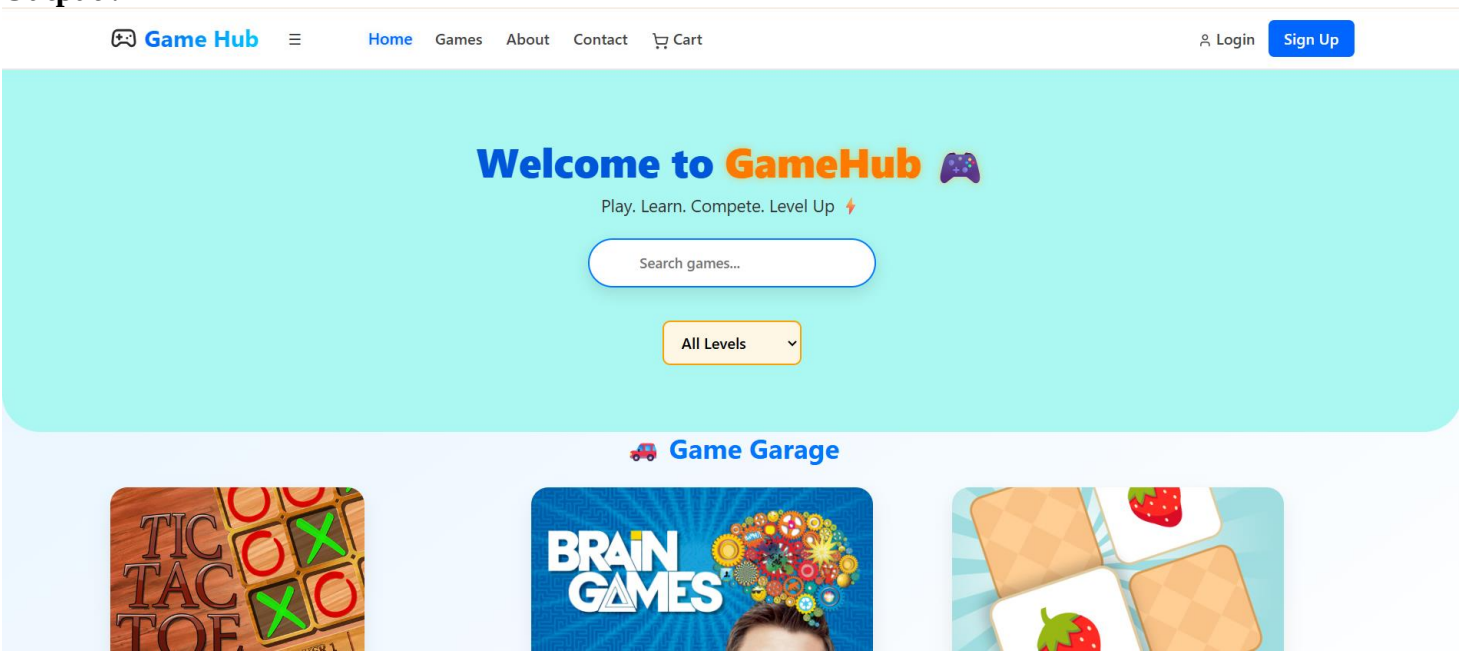
```
import React, { useState } from "react";
import { Link } from "react-router-dom";
import { motion } from "framer-motion";
import { Search } from "lucide-react";

// ✅ Game list with category(level)
const games = [
  { id: 1, name: "Tic Tac Toe", level: "Beginner", price: 0, description: "Classic 3x3 strategy board game.", img: "https://img.gamedistribution.com/abebecfa89b646448c834963627b325d-512x512.jpeg" },
  { id: 2, name: "Chess", level: "Advanced", price: 99, description: "Strategic board game to test intelligence.", img: "https://static.vecteezy.com/system/resources/previews/018/871/720/original/king-and-soldier-chess-pieces-on-transparent-background-leadership-concept-png.png" },
  { id: 3, name: "Number Puzzle", level: "Intermediate", price: 49, description: "Brain-boosting number puzzle challenge.", img: "https://tse4.mm.bing.net/th/id/OIP.34XhO9kOhDaGy3ah4n5K0wHaHa?pid=Api&P=0&h=180" },
  { id: 4, name: "Brain Games", level: "Intermediate", price: 79, description: "Fun mind training & reasoning puzzles.", img: "https://www.besttechie.com/content/images/wordpress/2020/05/brain-games.jpeg" },
  { id: 5, name: "Cricket Game", level: "Beginner", price: 0, description: "Virtual cricket match challenge.", img: "https://tse4.mm.bing.net/th/id/OIP.e2P3ReDBMC5G87UnoHrt3wHaEM?pid=Api&P=0&h=180" },
  { id: 6, name: "Car Racing", level: "Advanced", price: 149, description: "High-speed racing game.", img: "https://lh4.ggpht.com/DnegeWoA2nbLodwAI942HiEomkdHHWTG6pRBpV3CjnQkAE1LODW5-aJ93xWa8fDEk6c=h900" },
  { id: 7, name: "Memory Match", level: "Beginner", price: 39, description: "Enhance your memory with fun card matches.", img: "https://img.gamedistribution.com/854fd8048c814acaa86b0faffe364290-512x512.jpeg" },
  { id: 8, name: "Sudoku", level: "Advanced", price: 89, description: "Classic number placement logic puzzle.", img: "https://wallpaperaccess.com/full/9692889.png" },
];

export default function Home() {
  const [search, setSearch] = useState("");
  const [filterLevel, setFilterLevel] = useState("All");

  // ✅ Add to MyGames (localStorage cart simulation)
  const addToMyGames = (game) => {
    const stored = JSON.parse(localStorage.getItem("myGames")) || [];
    const exists = stored.find(g => g.id === game.id);
    if (!exists) {
      stored.push(game);
      localStorage.setItem("myGames", JSON.stringify(stored));
      alert(`${game.name} added to My Games!`);
    } else {
      alert("Already added!");
    }
  };

  // ✅ Filter + Search logic
  const filteredGames = games.filter(g =>
    g.name.toLowerCase().includes(search.toLowerCase()) &&
    (filterLevel === "All" || g.level === filterLevel)
  );

  return (
    <div className="home-container">
      {/* Hero Section */}
      <motion.section
        className="hero hero-orange"
        initial={{ opacity: 0, y: -20 }}
        animate={{ opacity: 1, y: 0 }}
      >
        <h1 className="hero-title">
          Welcome to <span className="hub-title">GameHub 🎮</span>
        </h1>
        <p className="hero-sub">Play. Learn. Compete. Level Up ⚡</p>

        {/* Search */}
        <div className="search-box">
          <Search size={20} />
          <input
            type="text"
            placeholder="Search games..."
            onChange={(e) => setSearch(e.target.value)}
          />
        </div>

        {/* Category Filter */}
        <select className="filter-select" onChange={(e)
```

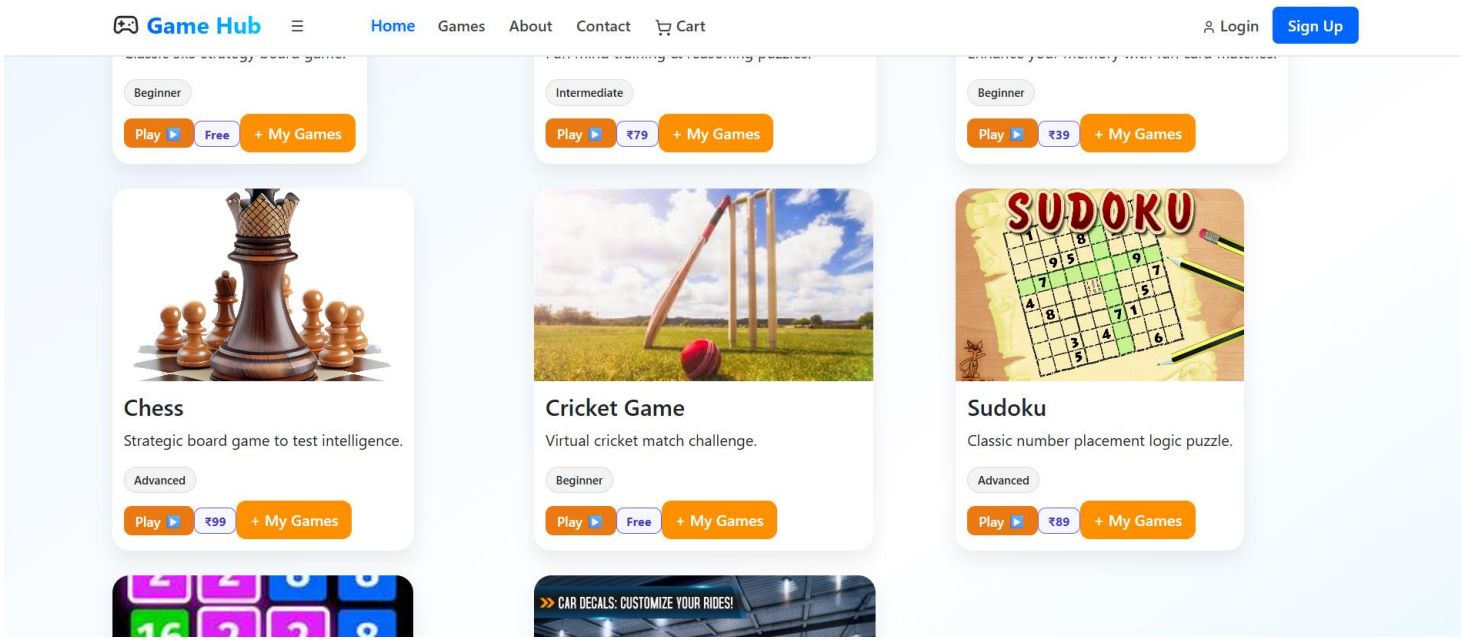
```
=> setFilterLevel(e.target.value)}>
        <option value="All">All Levels</option>
        <option
value="Beginner">Beginner</option>
        <option
value="Intermediate">Intermediate</option>
        <option
value="Advanced">Advanced</option>
       </select>
     </motion.section>

     {/* Game Display */}
     <section className="game-section container">
       <h2 className="section-heading text-center
fw-bold mb-4">🚗 Game Garage</h2>

       <div className="masonry-grid">
         {filteredGames.map((g, i) => (
          <motion.article
           key={g.id}
           className={`mag-card mag-${i % 3}`}
           initial={{ opacity: 0, y: 12 }}
           animate={{ opacity: 1, y: 0 }}
           transition={{ duration: 0.3, delay: i * 0.04
}}
          >
           <img src={g.img} alt={g.name} />
           <div className="mag-body">
            <h4>{g.name}</h4>
            <p
className="excerpt">{g.description}</p>
             <span className="level-
tag">{g.level}</span>

             <div className="mag-actions">
<Link
 to={`/game/${g.id}`}
 state={g}
 className="mag-play"
>
 Play ▸
</Link>

<span className="price-tag">
 {g.price === 0 ? "Free" : `₹${g.price}`}
</span>
              <button className="mag-add-btn"
onClick={() => addToMyGames(g)}>
               + My Games
              </button>
             </div>
            </div>
           </motion.article>
          ))}
         </div>
        </section>

        {/* Bottom Banner */}
        <section className="hub-banner">
         🎯 Level-Up Your Mind With Fun Learning
Games 🚀
        </section>
       </div>
     );
}
```

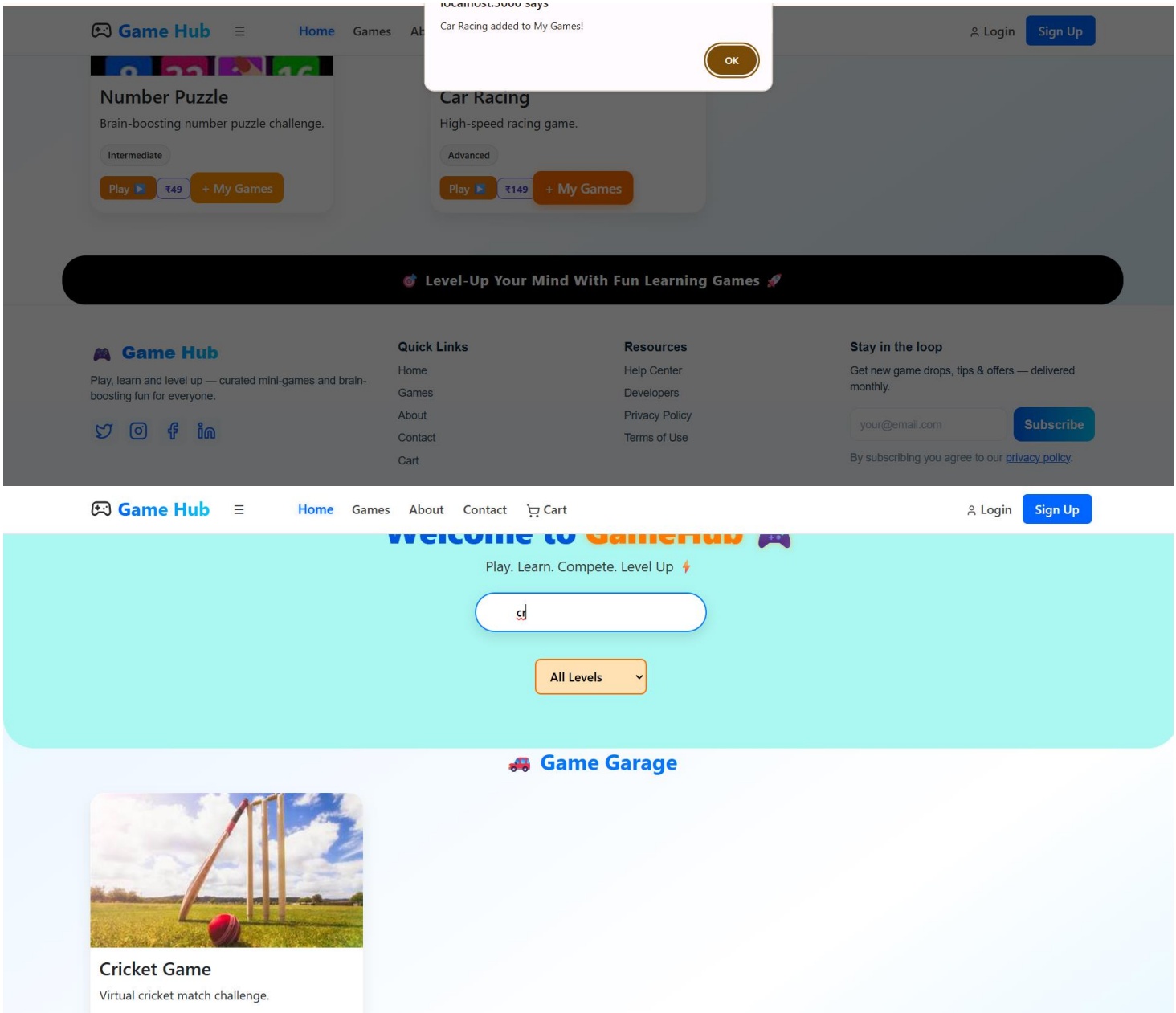
```
import React, { useState } from "react";
import { Link } from "react-router-dom";
import { Twitter, Instagram, Facebook, Linkedin }
from "lucide-react";

export default function Footer() {
 const [email, setEmail] = useState("");

 const submitNewsletter = (e) => {
  e.preventDefault();
  if (!email) {
   alert("Please enter an email.");
   return;
  }
  alert(`Thanks — ${email} subscribed to Game Hub
newsletter!`);
  setEmail("");
 };

 return (
  <footer className="gv-footer">
   <div className="gv-footer-top container">
    {/* Column 1: Brand + desc */}
    <div className="gv-col gv-brand">
     <div className="gv-logo">🎮 <span
className="gv-logo-text">Game Hub</span></div>
     <p className="gv-desc">
      Play, learn and level up — curated mini-games
and brain-boosting fun for everyone.
```

```
        </p>
        <div className="gv-socials" aria-
label="Follow Game Hub">
          <a href="https://twitter.com"
target="_blank" rel="noopener noreferrer" aria-
label="Twitter">
            <Twitter />
          </a>
          <a href="https://instagram.com"
target="_blank" rel="noopener noreferrer" aria-
label="Instagram">
            <Instagram />
          </a>
          <a href="https://facebook.com"
target="_blank" rel="noopener noreferrer" aria-
label="Facebook">
            <Facebook />
          </a>
          <a href="https://linkedin.com"
target="_blank" rel="noopener noreferrer" aria-
label="LinkedIn">
            <Linkedin />
          </a>
        </div>
      </div>

      {/* Column 2: Quick Links */}
      <div className="gv-col">
        <h4>Quick Links</h4>
        <nav className="gv-links" aria-
label="Quick links">
          <Link to="/">Home</Link>
          <Link to="/games">Games</Link>
          <Link to="/about">About</Link>
          <Link to="/contact">Contact</Link>
          <Link to="/cart">Cart</Link>
        </nav>
      </div>

      {/* Column 3: Resources */}
      <div className="gv-col">
        <h4>Resources</h4>
        <nav className="gv-links" aria-
label="Resources links">
          <Link to="/contact">Help Center</Link>
          <Link to="/about">Developers</Link>
          <Link to="/privacy">Privacy Policy</Link>
          <Link to="/terms">Terms of Use</Link>
        </nav>
      </div>

      {/* Column 4: Newsletter */}
      <div className="gv-col">
        <h4>Stay in the loop</h4>
        <p className="small">Get new game drops,
tips & offers — delivered monthly.</p>

        <form className="gv-newsletter"
onSubmit={submitNewsletter}>
          <input
            type="email"
            placeholder="your@email.com"
            value={email}
            onChange={(e) => setEmail(e.target.value)}
            aria-label="Email for newsletter"
          />
          <button type="submit" className="btn-
sub">Subscribe</button>
        </form>

        <p className="gv-minor small">
          By subscribing you agree to our <Link
to="/privacy">privacy policy</Link>.
        </p>
      </div>
    </div>

    <div className="gv-footer-bottom">
      <div className="container d-flex justify-
content-between align-items-center">
        <div className="copyright">
          © {new Date().getFullYear()} Game Hub •
Built by <strong>Prathmesh Waghmare</strong>
        </div>

        <div className="gv-made small">
          Designed with ♥ — bright, friendly and
accessible UI
        </div>
      </div>
    </div>
  </footer>
 );
}
```

Output :


Game Hub
Home
Games
About
Contact
Cart
Login
Sign Up
Welcome to GameHub
Play. Learn. Compete. Level Up
Search games...
All Levels
Game Garage
TIC TAC TOE
BRAIN GAMES


Game Hub
Home
Games
About
Contact
Cart
Login
Sign Up
Beginner
Play
Free
+ My Games
Intermediate
Play
₹79
+ My Games
Beginner
Play
₹39
+ My Games
Chess
Strategic board game to test intelligence.
Advanced
Play
₹99
+ My Games
Cricket Game
Virtual cricket match challenge.
Beginner
Play
Free
+ My Games
SUDOKU
Sudoku
Classic number placement logic puzzle.
Advanced
Play
₹89
+ My Games
CAR DECALS: CUSTOMIZE YOUR RIDES!


Game Hub
Home
Games
Login
Sign Up
Car Racing added to My Games!
OK
Number Puzzle
Brain-boosting number puzzle challenge.
Intermediate
Play
₹49
+ My Games
Car Racing
High-speed racing game.
Advanced
Play
₹149
+ My Games
Level-Up Your Mind With Fun Learning Games
Game Hub
Play, learn and level up — curated mini-games and brain-boosting fun for everyone.
Quick Links
Home
Games
About
Contact
Cart
Resources
Help Center
Developers
Privacy Policy
Terms of Use
Stay in the loop
Get new game drops, tips & offers — delivered monthly.
your@email.com
Subscribe
By subscribing you agree to our privacy policy.

Game Hub
Home
Games
About
Contact
Cart
Login
Sign Up
Play. Learn. Compete. Level Up
cr
All Levels
Game Garage
Cricket Game
Virtual cricket match challenge.

Conclusion :

We successfully implemented Home page, Footer and Search Functionality.

Assignment 3 : Fetch Game data from array or static JSON file

---

Aim :

To understand and implement Routing in React using the react-router-dom library, and to design a Games **page** that displays a list of 12 games with their details and options such as Play and Add to My Games (cart/favorites).

Theory :

- **Introduction to Routing**

  Routing in React refers to the process of navigating between different views or pages in a single-page application (SPA).

  In traditional websites, navigation between pages requires full-page reloads, but React applications handle navigation on the client-side using JavaScript — resulting in a faster, smoother user experience.

  Routing in React is managed through the React Router library (react-router-dom).

- **What is React Router ?**

  React Router is a powerful and widely used library for managing routes in React applications. It allows you to define different paths (URLs) and map them to specific components that should render when those paths are accessed.

- **Components of React Router**

Component / Hook	Description
BrowserRouter	The main container that keeps the UI in sync with the URL and wraps the entire Game Hub application.
Routes	Acts as a container for all route definitions.
Route	Defines a specific path (URL) and the component that should render when that path is accessed.
Link	Replaces <a> tag for navigation. Prevents full page reload and provides smooth transitions.
NavLink	Similar to Link, but adds active styling to indicate the current route.
useNavigate	A hook used to programmatically navigate between routes (e.g., redirect to /games).

- **How Routing Works ?**

1. The entire React application is wrapped inside <BrowserRouter>.
2. The <Routes> component defines multiple <Route> components.
3. Each <Route> specifies a URL path and the corresponding component to render.
4. When the user clicks a Link (or navigates to a URL), React Router dynamically renders the corresponding component without reloading the page.

This approach gives the illusion of multiple pages while actually rendering everything within one HTML document — hence the term Single Page Application.

- **Advantages of Routing**

- Fast Navigation: No page reloads; routing handled client-side.

- Better User Experience: Seamless transitions between pages.

- Maintainability: Each route is mapped to a separate, modular component.

- Dynamic Routes: Pages can be rendered based on dynamic data (e.g., /games/:id).

- Reusable Components: Navigation structure remains the same while content changes dynamically.

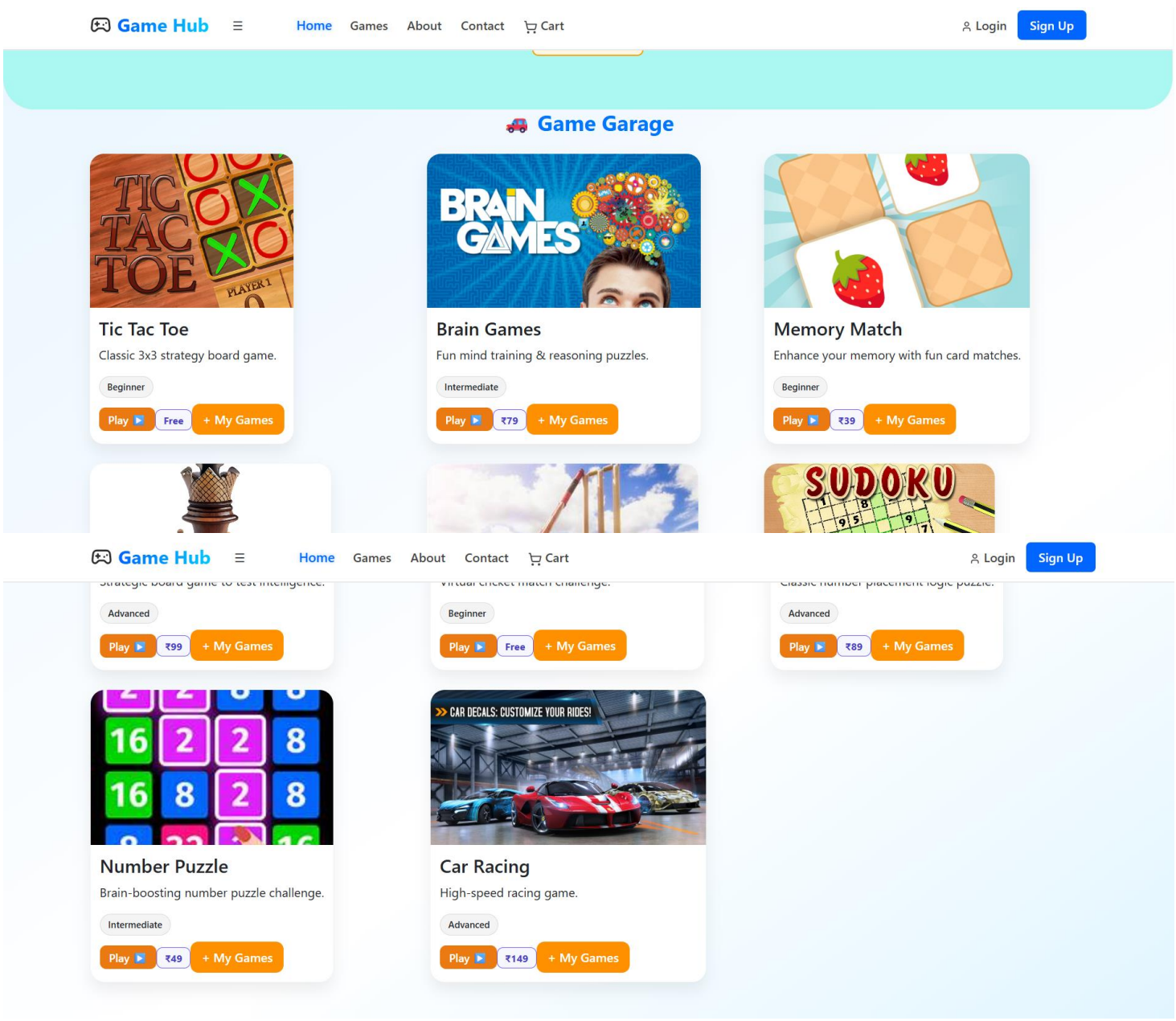
Code :

```
import React, { useState } from "react";
import { Link } from "react-router-dom";
import { motion } from "framer-motion";
import { Search } from "lucide-react";

// ✅ Game list with category(level)
const games = [
  { id: 1, name: "Tic Tac Toe", level: "Beginner", price: 0, description: "Classic 3x3 strategy board game.", img: "https://img.gamedistribution.com/abebecfa89b646448c834963627b325d-512x512.jpeg" },
  { id: 2, name: "Chess", level: "Advanced", price: 99, description: "Strategic board game to test intelligence.", img: "https://static.vecteezy.com/system/resources/previews/018/871/720/original/king-and-soldier-chess-pieces-on-transparent-background-leadership-concept-png.png" },
  { id: 3, name: "Number Puzzle", level: "Intermediate", price: 49, description: "Brain-boosting number puzzle challenge.", img: "https://tse4.mm.bing.net/th/id/OIP.34XhO9kOhDaGy3ah4n5K0wHaHa?pid=Api&P=0&h=180" },
  { id: 4, name: "Brain Games", level: "Intermediate", price: 79, description: "Fun mind training & reasoning puzzles.", img: "https://www.besttechie.com/content/images/wordpress/2020/05/brain-games.jpeg" },
  { id: 5, name: "Cricket Game", level: "Beginner", price: 0, description: "Virtual cricket match challenge.", img: "https://tse4.mm.bing.net/th/id/OIP.e2P3ReDBMC5G87UnoHrt3wHaEM?pid=Api&P=0&h=180" },
  { id: 6, name: "Car Racing", level: "Advanced", price: 149, description: "High-speed racing game.", img: "https://lh4.ggpht.com/DnegeWoA2nbLodwAI942HiEomkdHHWTG6pRBpV3CjnQkAE1LODW5-aJ93xWa8fDEk6c=h900" },
  { id: 7, name: "Memory Match", level: "Beginner", price: 39, description: "Enhance your memory with fun card matches.", img: "https://img.gamedistribution.com/854fd8048c814acaa86b0faffe364290-512x512.jpeg" },
```

```
  { id: 8, name: "Sudoku", level: "Advanced", price: 89, description: "Classic number placement logic puzzle.", img: "https://wallpaperaccess.com/full/9692889.png" },
];

export default function Home() {
  const [search, setSearch] = useState("");
  const [filterLevel, setFilterLevel] = useState("All");

  // ✅ Add to MyGames (localStorage cart simulation)
  const addToMyGames = (game) => {
    const stored = JSON.parse(localStorage.getItem("myGames")) || [];
    const exists = stored.find(g => g.id === game.id);
    if (!exists) {
      stored.push(game);
      localStorage.setItem("myGames", JSON.stringify(stored));
      alert(`${game.name} added to My Games!`);
    } else {
      alert("Already added!");
    }
  };

  // ✅ Filter + Search logic
  const filteredGames = games.filter(g =>
    g.name.toLowerCase().includes(search.toLowerCase()) &&
    (filterLevel === "All" || g.level === filterLevel)
  );

  return (
    <div className="home-container">
      {/* Hero Section */}
      <motion.section
        className="hero hero-orange"
        initial={{ opacity: 0, y: -20 }}
        animate={{ opacity: 1, y: 0 }}
      >
        <h1 className="hero-title">
          Welcome to <span className="hub-title">GameHub 🎮</span>
        </h1>
        <p className="hero-sub">Play. Learn. Compete. Level Up ⚡</p>

        {/* Search */}
        <div className="search-box">
          <Search size={20} />
          <input
            type="text"
            placeholder="Search games..."
            onChange={(e) => setSearch(e.target.value)}
          />
        </div>

        {/* Category Filter */}
        <select className="filter-select" onChange={(e) => setFilterLevel(e.target.value)}>
          <option value="All">All Levels</option>
          <option value="Beginner">Beginner</option>
          <option value="Intermediate">Intermediate</option>
          <option value="Advanced">Advanced</option>
        </select>
      </motion.section>

      {/* Game Display */}
      <section className="game-section container">
        <h2 className="section-heading text-center fw-bold mb-4">🚗 Game Garage</h2>

        <div className="masonry-grid">
          {filteredGames.map((g, i) => (
            <motion.article
              key={g.id}
              className={`mag-card mag-${i % 3}`}
              initial={{ opacity: 0, y: 12 }}
              animate={{ opacity: 1, y: 0 }}
              transition={{ duration: 0.3, delay: i * 0.04 }}
            >
              <img src={g.img} alt={g.name} />
              <div className="mag-body">
                <h4>{g.name}</h4>
                <p className="excerpt">{g.description}</p>
                <span className="level-tag">{g.level}</span>

                <div className="mag-actions">
```

```
<Link
  to={`/game/${g.id}`}
  state={g}
  className="mag-play"
>
  Play ▶
</Link>

<span className="price-tag">
  {g.price === 0 ? "Free" : `₹${g.price}`}
</span>
                <button className="mag-add-btn"
onClick={() => addToMyGames(g)}>
                  + My Games
                </button>
              </div>
            </div>
          </motion.article>
        ))}
      </div>
    </section>

    {/* Bottom Banner */}
    <section className="hub-banner">
      🎯 Level-Up Your Mind With Fun
Learning Games 🚀
    </section>
   </div>
  );
}
```

Output :



Conclusion :

We successfully implemented JSON handling and routing.

Assignment 4 : Game Details Page

Aim :

To design and develop a Game Details Page in React that displays complete information of a selected game, including its image, name, level, description, and price, along with options to Play the Game and Add it to My Games. This page helps users explore details of a game before playing or saving it to their personal library.

Theory :

- **Introduction**

  The **Game Details Page** serves as a key section of the Game Hub application.

  It displays full information about a selected game and allows users to view details, play the game, or add it to their personal library.

  In React, this page is implemented as a functional component that uses routing state to receive game data and follows the component-based architecture for reusability and clean code structure.

- **Component-Based Architecture**

  React uses modular, reusable components to build the UI.
  In this project:

  - Home.jsx displays the list of all games.
  - GameDetails.jsx functions as the page-level component, responsible for showing complete game information and handling user actions.

  This structure ensures scalability and simplifies future updates — any change to game details is handled in one component only.

- **Game Information Rendering**

  The featured products section highlights key products or categories to attract user attention.

  In React, this is implemented by maintaining a list (array) of products inside the component's state. Each product has attributes such as:

  - id
  - name
  - level

- price

- description

- image

- These values are passed to the Game Details Page and displayed dynamically.
  This approach allows new games to be added easily by updating the game data array without modifying the UI code.

- **Add to My Games Functionality**

  Instead of a typical “Add to Cart” feature, this project uses Add to My Games, where selected games are stored in localStorage.
  This allows users to save games and access them later, enhancing user experience and mimicking game libraries seen in gaming platforms.

- **Routing and Navigation**

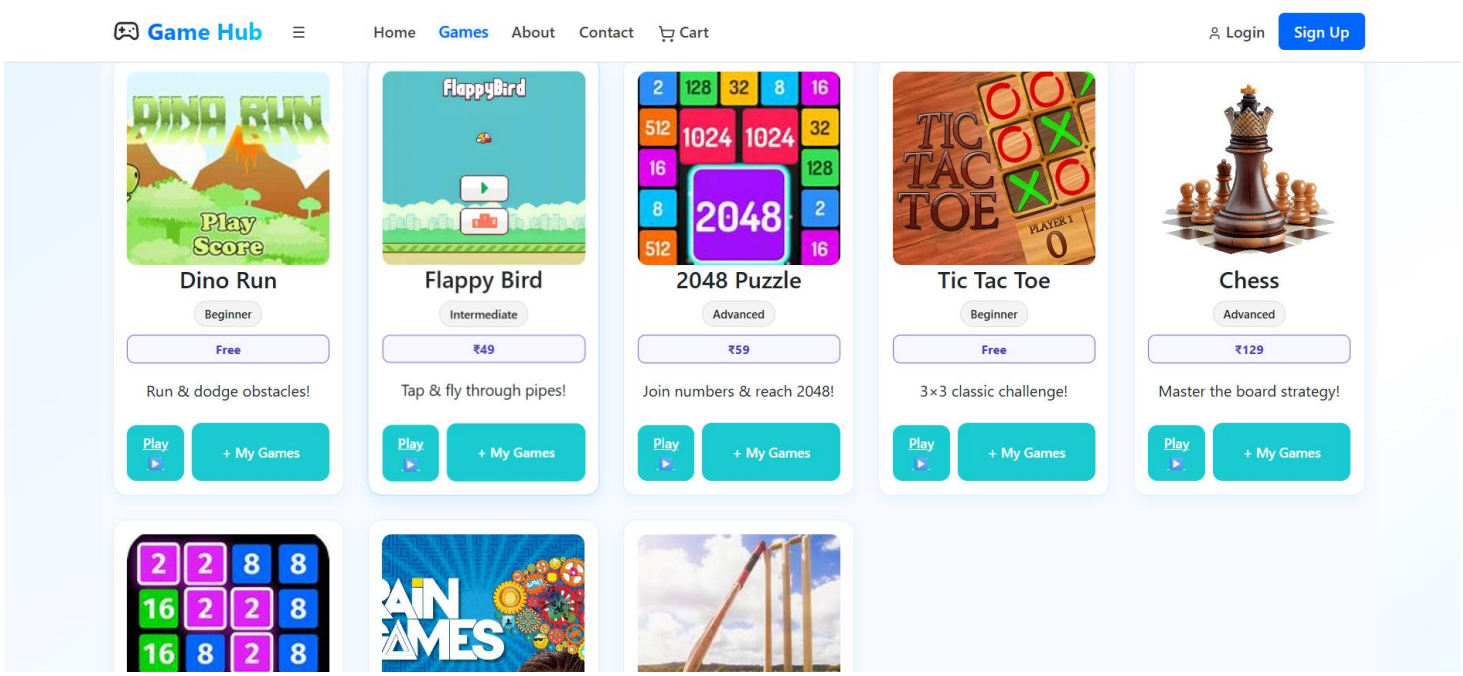
  Navigation to the Game Details Page happens through React Router.
  When a user clicks a game on the Home Page, they are routed to /game/:id using the <Link> component.
  The game data is accessed using useLocation(), and users can return using useNavigate().
  This client-side routing provides smooth navigation similar to modern Single Page Applications (SPA).

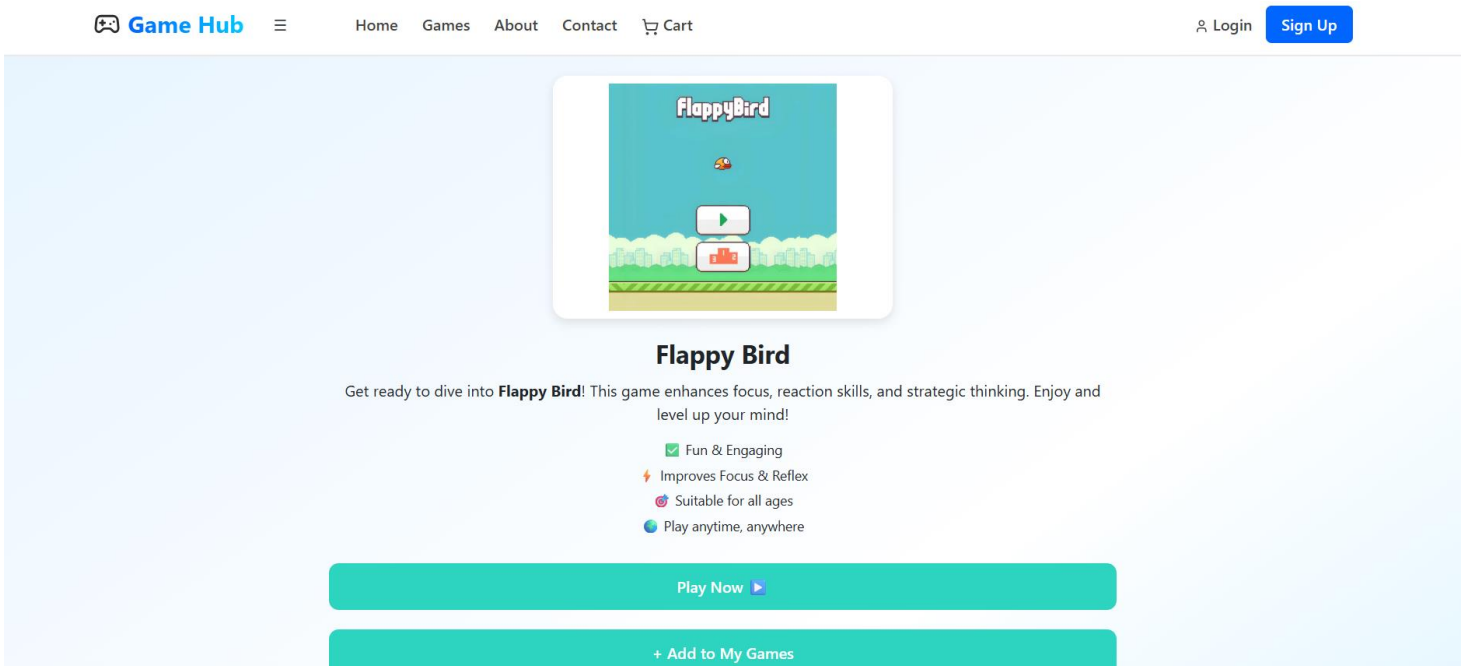
Code :

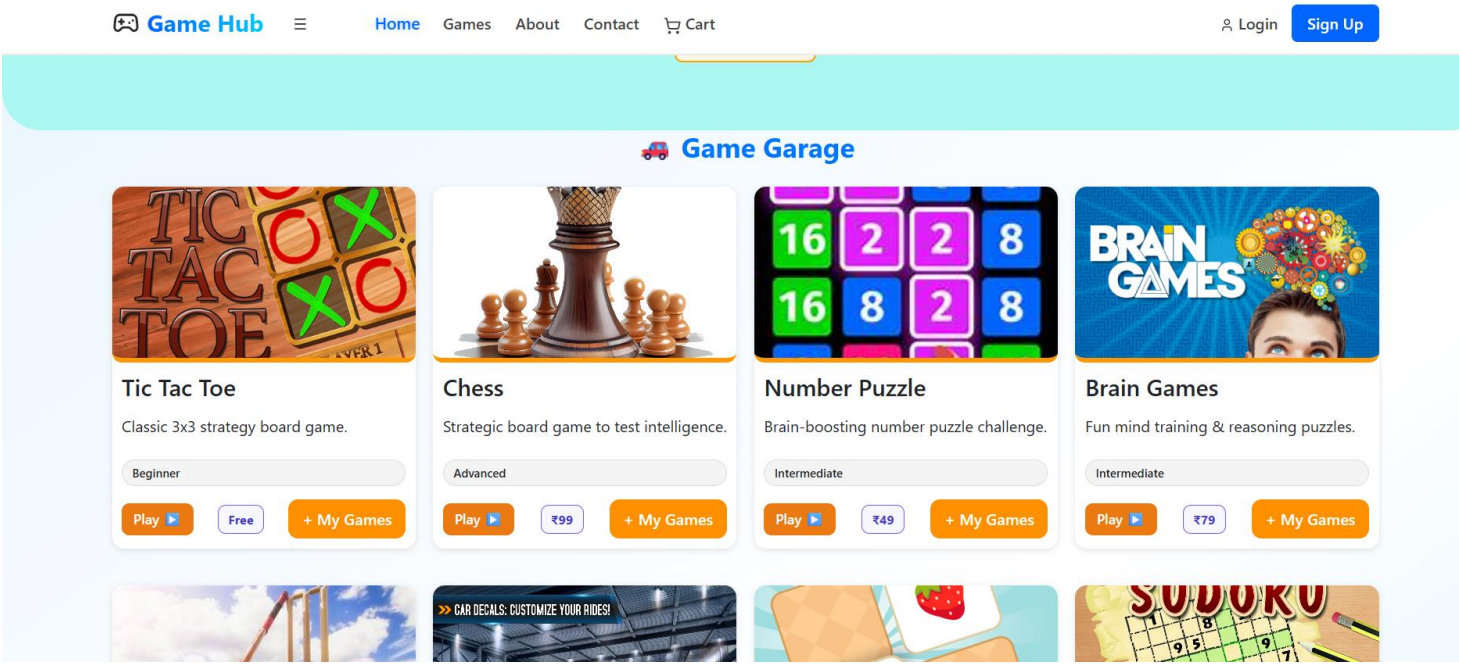
```
import React from "react";
import { useLocation, useNavigate } from "react-router-dom";
import { motion } from "framer-motion";

export default function GameDetails() {
  const { state: game } = useLocation();
  const navigate = useNavigate();

  const handleAddToGames = () => {
    let library = JSON.parse(localStorage.getItem("myGames")) || [];

    // prevent duplicate
    if (!library.find(g => g.id === game.id)) {
      library.push(game);
      localStorage.setItem("myGames", JSON.stringify(library));
      alert(`${game.name} added to My Games ✅`);
    } else {
      alert("Already added ✓");
    }
  };

  return (
    <motion.div
      className="game-details-container"
      initial={{ opacity: 0, scale: 0.95 }}
      animate={{ opacity: 1, scale: 1 }}
      transition={{ duration: 0.4 }}
    >
      <img src={game.img} alt={game.name} className="details-img" />

      <h1 className="details-title">{game.name}</h1>
      <p className="details-desc">
```

```
      Get ready to dive into <b>{game.name}</b!
This game enhances focus,
      reaction skills, and strategic thinking. Enjoy
and level up your mind!
    </p>

    <ul className="details-features">
      <li>✅ Fun & Engaging</li>
      <li>⚡ Improves Focus & Reflex</li>
      <li>🎯 Suitable for all ages</li>
      <li>🌍 Play anytime, anywhere</li>
    </ul>

    <div className="details-btn-group">
      <button className="play-now-btn">
        Play Now ▶
      </button>

      <button onClick={handleAddToGames}
className="play-now-btn">
        + Add to My Games
      </button>

      <button onClick={() => navigate(-1)}
className="back-btn">
        ⬅ Back
      </button>
    </div>
  </motion.div>
 );
}
```

Output :


Game Hub
Home
Games
About
Contact
Cart
Login
Sign Up
Dino Run
Beginner
Free
Run & dodge obstacles!
Play
+ My Games
Flappy Bird
Intermediate
₹49
Tap & fly through pipes!
Play
+ My Games
2048 Puzzle
Advanced
₹59
Join numbers & reach 2048!
Play
+ My Games
Tic Tac Toe
Beginner
Free
3×3 classic challenge!
Play
+ My Games
Chess
Advanced
₹129
Master the board strategy!
Play
+ My Games


Game Hub
Home
Games
About
Contact
Cart
Login
Sign Up
Flappy Bird
Get ready to dive into Flappy Bird! This game enhances focus, reaction skills, and strategic thinking. Enjoy and level up your mind!
Fun & Engaging
Improves Focus & Reflex
Suitable for all ages
Play anytime, anywhere
Play Now
+ Add to My Games



Conclusion :

We successfully implemented Products Page.

Assignment 5 : Cart Page

Aim :

To design and develop a Game Cart Page in React that allows users to view, manage, and remove selected games added from the Home and Games pages.

Theory :

- **Introduction**

  The Cart Page in GameHub acts as a temporary game collection area where users store games they wish to play later or purchase.
  It displays selected games along with their image, name, and price, and maintains this data even after refresh using localStorage.
  This page improves user engagement by helping users bookmark and manage their favorite games easily.

- **Data Flow in React Cart Page**

  When a user clicks "**+ My Games**" on the Home or Games page, the selected game object is saved in localStorage.
  The Cart Page retrieves this data using localStorage.getItem() and displays the list of selected games.
  Thus, the flow is:
  Home/Games Page → Add to My Games → localStorage → Cart Page display.

- **State Management**

  React's useState() hook is used to store and manage all cart items.

- **Component Structure**

  1. Home.js / Games.js – Contains "+ My Games" button to add games to cart.
  2. Cart.js: Displays all selected products with options to update or remove.
  3. App.js: Handles routing between /products and /cart.

  This modular design ensures code reusability, scalability, and separation of concerns.

- **Quantity Handling**

  Each product's quantity is controlled by an input field.

- **Deletion of Products**

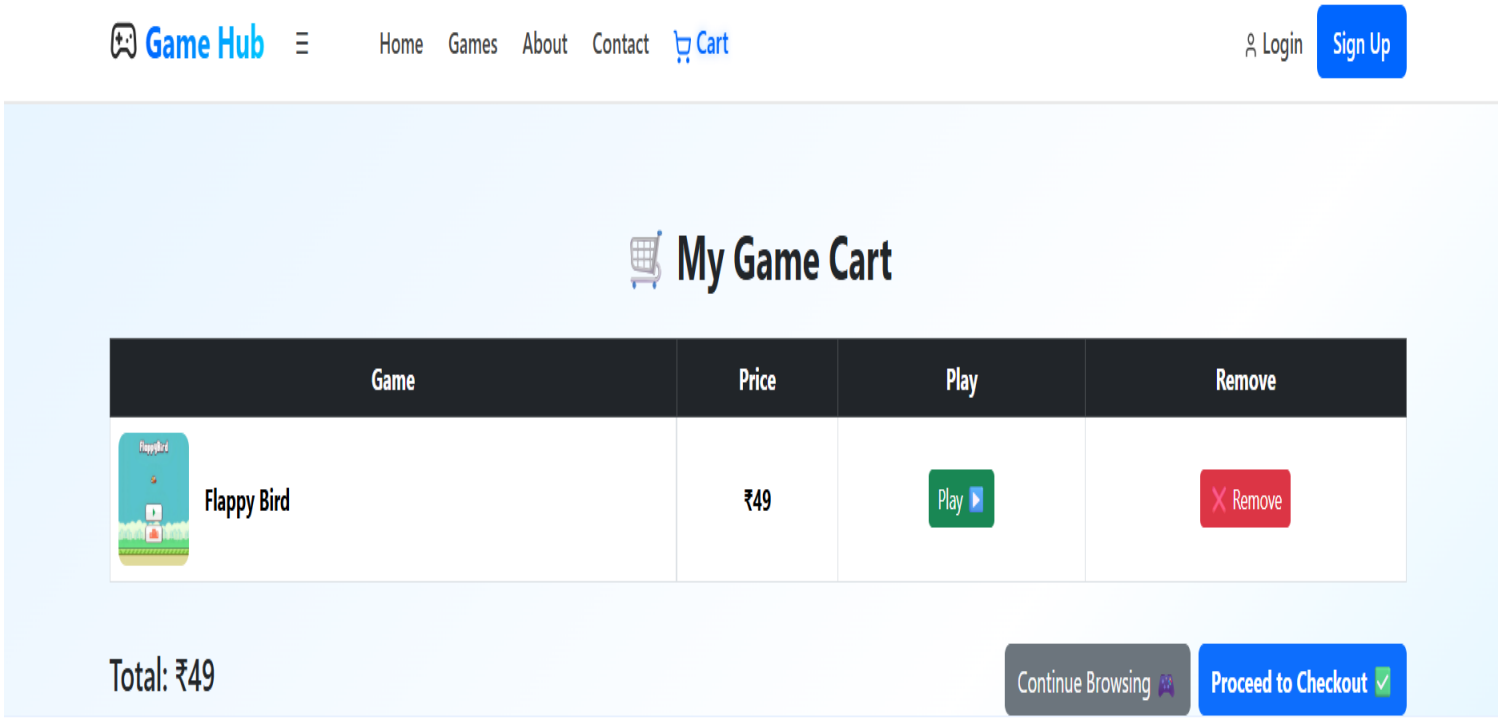
  The delete functionality uses the filter() method.\

Code :

```
import React, { useEffect, useState } from "react";
import { Link, useNavigate } from "react-router-dom";

export default function Cart() {
  const [cartItems, setCartItems] = useState([]);
  const navigate = useNavigate();

  useEffect(() => {
    const savedCart = JSON.parse(localStorage.getItem("cart")) || [];
    setCartItems(savedCart);
  }, []);

  const removeItem = (id) => {
    const updated = cartItems.filter(item => item.id !== id);
    setCartItems(updated);
    localStorage.setItem("cart", JSON.stringify(updated));
  };

  const totalPrice = cartItems.reduce((sum, game) => sum + game.price, 0);

  const handleCheckout = () => {
    navigate("/checkout", { state: { cartItems, total: totalPrice } });
  };

  return (
    <div className="container" style={{ marginTop: "80px" }}>
      <h2 className="fw-bold text-center mb-4">🛒 My Game Cart</h2>

      {cartItems.length === 0 ? (
        <p className="text-center fs-5">
          Your game cart is empty 😢 <br /><br />
          <Link to="/" className="btn btn-primary">Browse Games</Link>
        </p>
      ) : (
        <>
          <div className="table-responsive">
            <table className="table table-bordered align-middle">
              <thead className="table-dark text-center">
                <tr>
                  <th>Game</th>
                  <th>Price</th>
                  <th>Play</th>
                  <th>Remove</th>
                </tr>
              </thead>
              <tbody>
                {cartItems.map((game) => (
                  <tr key={game.id} className="text-center">
                    <td className="d-flex align-items-center gap-3">
                      <img
                        src={game.img}
                        alt={game.name}
                        style={{ width: "70px", height: "70px", objectFit: "cover", borderRadius: "8px" }}
                      />
                      <strong>{game.name}</strong>
                    </td>

                    <td className="fw-bold">{game.price === 0 ? "Free" : `₹${game.price}`}</td>

                    <td>
                      <a href={game.link} target="_blank" rel="noopener noreferrer" className="btn btn-success btn-sm">
                        Play ▶
                      </a>
                    </td>
```

```
                <td>
                  <button className="btn
btn-danger btn-sm" onClick={() =>
removeItem(game.id)}>
                    ❌ Remove
                  </button>
                </td>
              </tr>
            ))}
          </tbody>
        </table>
      </div>

      {/* Total & Buttons */}
      <div className="d-flex justify-
content-between align-items-center mt-
3">
        <h4>Total: {totalPrice === 0 ?
"Free" : `₹${totalPrice}`}</h4>

        <div className="d-flex gap-
2">
          <Link to="/" className="btn
btn-secondary">Continue Browsing
🎮</Link>
          <button className="btn btn-
primary fw-bold"
onClick={handleCheckout}>
            Proceed to Checkout ✅
          </button>
        </div>
      </div>
    </>
  )}
 </div>
);
}

const styles = {
 container: {
  maxWidth: "700px",
  margin: "50px auto",
  padding: "20px",
 },
 title: {
  fontSize: "28px",
  fontWeight: "bold",
  marginBottom: "20px",
  textAlign: "center",
 },
 empty: {
  fontSize: "18px",
  textAlign: "center",
 },
 card: {
  display: "flex",
  alignItems: "center",
  gap: "15px",
  padding: "15px",
  background: "#fff",
  borderRadius: "12px",
  marginBottom: "15px",
  boxShadow: "0 4px 10px rgba(0,0,0,0.1)",
 },
 image: {
  width: "80px",
  height: "80px",
  borderRadius: "10px",
  objectFit: "cover",
 },
 details: {
  flex: 1,
 },
 name: {
  fontSize: "20px",
  marginBottom: "10px",
 },
 removeBtn: {
  background: "#ff4d4d",
  color: "#fff",
  border: "none",
  padding: "8px 14px",
  borderRadius: "8px",
  cursor: "pointer",
  fontSize: "14px",
  transition: "0.2s",
 }
}
```

Output :



Conclusion :

We successfully implemented Cart Page.

Assignment 6 : Checkout Page

Aim :

To create a Checkout Page in the GameHub application that collects player details and displays a summary of selected games, allowing the user to confirm the order and complete the purchase process.

Theory :

- **Introduction**

  The Checkout Page is the final step in the GameHub purchase flow.
  It allows users to enter necessary details, review their selected games, and confirm the order.
  This page combines form handling, data validation, and cart data display to create a smooth user experience similar to modern gaming platforms and app stores.

- **Role of Checkout Page**

  The Checkout Page in GameHub serves three main functions:

  1. Collect Player Information– It captures customer information such as name, address, city, postal code, and contact number.
  2. Show Game Order Summary – It provides a detailed summary of all products added to the cart, including their quantities and total price.
  3. Confirming the Purchase – It allows the user to finalize the order through a confirmation button, typically labeled “Place Order”.

  This page combines user input, data display, and interaction handling within a single interface.

- **Data Flow from Cart Page**

  The Checkout Page receives order details from the Cart Page, such as:

  1. The list of selected products
  2. Their quantities and prices
  3. The total amount payable

  This information is passed from the cart to the checkout component using React Router navigation and state transfer.
  This approach ensures smooth data flow within the single-page application architecture of React without the need for backend integration or page refresh.

- **Form Handling and User Input**

  The Checkout Page contains a structured form to collect player information.

  Each input field — such as Full Name, Address, City, Pincode, and Phone Number — is linked to the internal state of the component.

  React's state-driven mechanism ensures that:

  1. Real-time form input tracking.
  2. Validation before order confirmation.
  3. Consistent data flow until order completion.

  This approach makes the checkout interactive and error-free.

- **Order Summary Section**

  The right-hand section of the Checkout Page displays the Order Summary, allowing the user to verify their purchase before placing the order.

  It typically includes:

- Game name
- Price
- Quantity
- Sub-total for each game
- Final total cost

  This helps users verify all selected games before placing the order, ensuring transparency and accuracy.

- **Place Order Button and Confirmation**

  Once all details are filled and verified, the user can click the "Place Order" button to finalize the purchase.

  The system then:
  - Validates the entered information to ensure no required field is empty.
  - Displays a confirmation message indicating successful order placement.
  - Redirects the user back to the Home or Thank You page for completion of the process.

  This simulates a real-world gaming purchase flow similar to Steam, Play Store, and

  gaming portals.

- **Hooks and React Concepts Involved**

  The GameHub Checkout Page uses key React concepts:

  1. State Management: Stores and updates user shipping details during form entry.
  2. Routing and Data Transfer: Game order details are passed from the Cart to Checkout using useNavigate and useLocation without page reload.
  3. Form Validation: Ensures all required fields are completed before placing the game order.

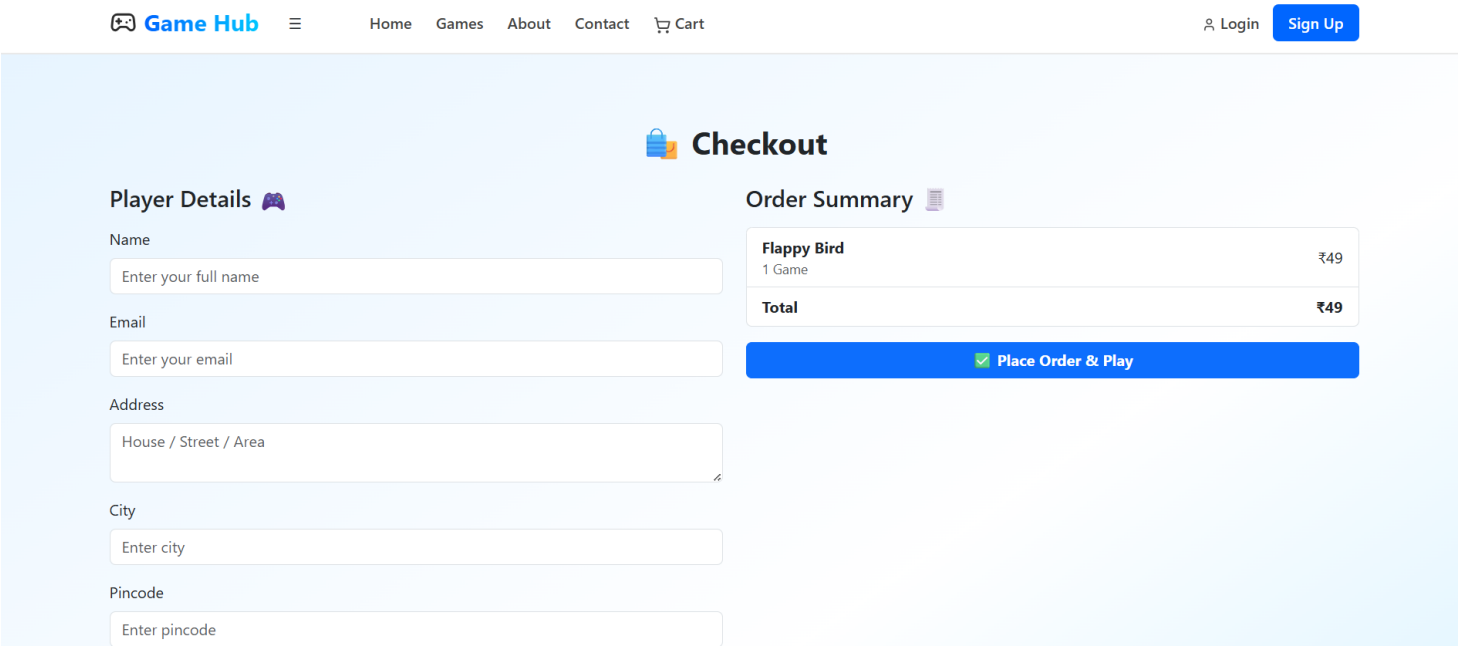
Together, these mechanisms enable GameHub's checkout system to smoothly handle player inputs, transfer selected game data, and validate details — showcasing how React's component-based structure efficiently manages user actions and dynamic game data.

Code :

```
import React, { useState } from "react";
import { useLocation, useNavigate } from "react-router-dom";

export default function Checkout() {
  const location = useLocation();
  const navigate = useNavigate();

  const { cartItems = [], total = 0 } = location.state || {};

  const [formData, setFormData] = useState({
    username: "",
    email: "",
    address: "",
    city: "",
    pincode: "",
  });

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

  const handlePlaceOrder = () => {
    if (
      !formData.username ||
      !formData.email ||
      !formData.address ||
      !formData.city ||
      !formData.pincode
    ) {
      alert("⚠ Please fill all details to continue.");
      return;
    }

    alert("🎉 Order Successful! Enjoy your games!");

    // Clear cart from storage
    localStorage.removeItem("cart");

    navigate("/");
  };

  return (
    <div className="container mt-5 pt-4">
      <h2 className="fw-bold text-center mb-4">🛍 Checkout</h2>

      <div className="row">

        {/* ✅ User Form */}
        <div className="col-md-6 mb-4">
          <h4 className="mb-3">Player Details 🎮</h4>
          <form>

            <div className="mb-3">
              <label className="form-label">Name</label>
              <input
                type="text"
                name="username"
                className="form-control"
                value={formData.username}
                onChange={handleChange}
                placeholder="Enter your full name"
              />
            </div>

            <div className="mb-3">
              <label className="form-label">Email</label>
              <input
                type="email"
                name="email"
                className="form-control"
                value={formData.email}
                onChange={handleChange}
                placeholder="Enter your email"
              />
            </div>

            <div className="mb-3">
              <label className="form-label">Address</label>
              <textarea
                name="address"
                className="form-control"
                value={formData.address}
                onChange={handleChange}
                placeholder="House / Street / Area"
              ></textarea>
            </div>
```

```
          <div className="mb-3">
            <label className="form-
label">City</label>
            <input
              type="text"
              name="city"
              className="form-control"
              value={formData.city}
              onChange={handleChange}
              placeholder="Enter city"
            />
          </div>

          <div className="mb-3">
            <label className="form-
label">Pincode</label>
            <input
              type="text"
              name="pincode"
              className="form-control"
              value={formData.pincode}
              onChange={handleChange}
              placeholder="Enter pincode"
            />
          </div>

        </form>
      </div>

      {/* ✅ Order Summary */}
      <div className="col-md-6">
        <h4 className="mb-3">Order Summary
</h4>

        {cartItems.length === 0 ? (
          <p>No games selected.</p>
        ) : (
          <ul className="list-group mb-3">
            {cartItems.map((game) => (
              <li
                key={game.id}
                className="list-group-item d-flex
justify-content-between align-items-center"
              >
                <div>
                  <strong>{game.name}</strong>
                  <p className="text-muted small mb-
0">1 Game</p>
                </div>
                <span>
                  {game.price === 0 ? "Free" :
`₹${game.price}`}
                </span>
              </li>
            ))}

            <li className="list-group-item d-flex
justify-content-between">
              <strong>Total</strong>
              <strong>{total === 0 ? "Free" :
`₹${total}`}</strong>
            </li>
          </ul>
        )}

        <button
          className="btn btn-primary w-100 fw-bold"
          onClick={handlePlaceOrder}
        >
          ✅ Place Order & Play
        </button>
      </div>
    </div>
  );
}
```

Output:



Conclusion :

We successfully implemented Checkout Page.

Assignment 7 : Profile Page

Aim :

To design and implement a Profile Page in the GameHub web application that displays all user details collected during registration.

Theory :

- **Introduction**

  The Profile Page is a key feature in GameHub that provides each gamer with a personalized dashboard showing their stored account information.
  It retrieves user details saved at the time of registration and displays them in a structured and interactive UI. This enhances personalization and improves user engagement inside the platform.

- **Purpose of the Profile Page**

  The main objective of the Profile Page is to:

  1. Show the registered user's complete details.
  2. Provide quick access to personal and gaming-related information.
  3. Enhance user identity and personalization.
  4. Serve as a foundation for future features like edit profile, badges, game history, and achievements.

- **Data Handling Mechanism**

  In this application, user information is initially captured during registration and stored locally using LocalStorage — a built-in browser feature that allows data persistence even after the page reloads.

  When the Profile Page loads:

  1. App checks if user is logged in.
  2. User data is fetched from LocalStorage.
  3. The retrieved data is then displayed on the Profile Page using React’s dynamic rendering.
  4. This ensures that each logged-in user sees their own data without requiring a backend database.

- **Component Design and Structure**

  Personal Information:
  Displays Name ,Username,Date of Birth,Gender & Country.

  Contact Information:
  Includes email & phone number.

  Login Status:
  Logout option

  .

Code :

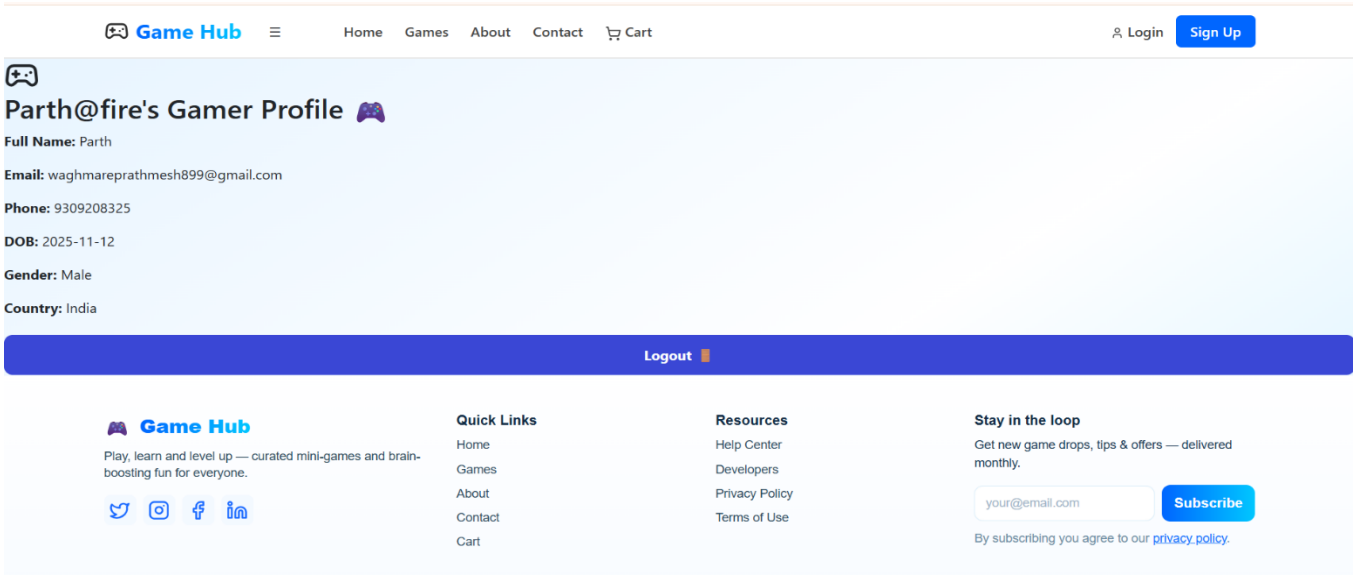
```
import loginGif from
"../assets/gifs/login.gif";

import React, { useState } from "react";
import { Link } from "react-router-dom";
import { motion } from "framer-motion";
import { Gamepad2 } from "lucide-react";

export default function Login() {
  const [form, setForm] = useState({ email:
"", password: "" });

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

  const handleSubmit = (e) => {
  e.preventDefault();

  const storedUser =
JSON.parse(localStorage.getItem("gameHub
User"));

  if (!storedUser) {
    alert("⚠ No account found. Please
register first!");
    return;
  }

  if (storedUser.email === form.email &&
storedUser.password === form.password) {

localStorage.setItem("gameHubLoggedIn",
"true");
    alert("✔ Login Successful!");
    window.location.href = "/profile";
  } else {
    alert("✕ Incorrect Email or Password!");
  }
};


  return (
    <div className="auth-page">
      {/* Left Content */}
      <motion.div
        className="auth-info"
        initial={{ opacity: 0, x: -40 }}
        animate={{ opacity: 1, x: 0 }}
      >
        <h1>Welcome Back 👋</h1>
        <p>Your gaming world is waiting. Log in &
continue your journey!</p>

        <img src={loginGif} alt="Login Animation"
className="auth-illustration" />
      </motion.div>

      {/* Login Form */}
      <motion.div
        className="auth-box shadow-lg"
        initial={{ opacity: 0, x: 40 }}
        animate={{ opacity: 1, x: 0 }}
      >
        <div className="auth-logo mb-3">
          <Gamepad2 size={35} className="me-2" />
          Game Hub
        </div>

        <h2 className="auth-title">Login</h2>
        <p className="auth-sub">Enter your details to
```

```
continue</p>

      <form onSubmit={handleSubmit}>
        <input
          type="email"
          name="email"
          className="auth-input"
          placeholder="Email address"
          value={form.email}
          onChange={handleChange}
          required
        />

        <input
          type="password"
          name="password"
          className="auth-input"
          placeholder="Password"
          value={form.password}
          onChange={handleChange}
          required
        />

        <button type="submit"
className="auth-btn">
          Login 🚀
        </button>
      </form>

      <p className="auth-footer">
        New here? <Link to="/register"
className="link-text">Create
Account</Link>
      </p>
    </motion.div>
  </div>
 );
}
export default Signup;
import loginGif from
"../assets/gifs/login.gif";

import React, { useState } from "react";
import { Link } from "react-router-dom";
import { motion } from "framer-motion";
import { Gamepad2 } from "lucide-react";

export default function Login() {
 const [form, setForm] = useState({ email:
"", password: "" });

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

 const handleSubmit = (e) => {
 e.preventDefault();

 const storedUser =
JSON.parse(localStorage.getItem("gameHubUser"));

 if (!storedUser) {
  alert("⚠️ No account found. Please register first!");
  return;
 }

 if (storedUser.email === form.email &&
storedUser.password === form.password) {
  localStorage.setItem("gameHubLoggedIn", "true");
  alert("✅ Login Successful!");
  window.location.href = "/profile";
 } else {
  alert("❌ Incorrect Email or Password!");
 }
};


 return (
  <div className="auth-page">
   {/* Left Content */}
   <motion.div
    className="auth-info"
    initial={{ opacity: 0, x: -40 }}
    animate={{ opacity: 1, x: 0 }}
   >
    <h1>Welcome Back 👋</h1>
    <p>Your gaming world is waiting. Log in &
continue your journey!</p>

    <img src={loginGif} alt="Login Animation"
className="auth-illustration" />
   </motion.div>

   {/* Login Form */}
   <motion.div
    className="auth-box shadow-lg"
    initial={{ opacity: 0, x: 40 }}
    animate={{ opacity: 1, x: 0 }}
   >
```

```
      <div className="auth-logo mb-3">
        <Gamepad2 size={35}
className="me-2" />
        Game Hub
      </div>

      <h2 className="auth-
title">Login</h2>
      <p className="auth-sub">Enter your
details to continue</p>

      <form onSubmit={handleSubmit}>
        <input
          type="email"
          name="email"
          className="auth-input"
          placeholder="Email address"
          value={form.email}
          onChange={handleChange}
          required
        />

        <input
          type="password"
          name="password"
          className="auth-input"
          placeholder="Password"
          value={form.password}
          onChange={handleChange}
          required
        />

        <button type="submit"
className="auth-btn">
          Login 🚀
        </button>
      </form>

      <p className="auth-footer">
        New here? <Link to="/register"
className="link-text">Create
Account</Link>
      </p>
    </motion.div>
  </div>
 );
}
export default Login;
import React, { useEffect, useState } from
"react";
import { Gamepad2 } from "lucide-react";
import { motion } from "framer-motion";

export default function Profile() {
 const [user, setUser] = useState(null);

 useEffect(() => {
  const isLogged =
localStorage.getItem("gameHubLoggedIn");
  const storedUser =
JSON.parse(localStorage.getItem("gameHubUser"));

  if (!isLogged || !storedUser) {
   window.location.href = "/login";
   return;
  }

  setUser(storedUser);
 }, []);

 const logout = () => {
  localStorage.removeItem("gameHubLoggedIn");
  alert("👋 Logged out successfully!");
  window.location.href = "/";
 };

 if (!user) return null;

 return (
  <div className="profile-page">
   <motion.div
    className="profile-card shadow-lg"
    initial={{ opacity: 0, scale: 0.8 }}
    animate={{ opacity: 1, scale: 1 }}
   >
    <div className="profile-header">
     <Gamepad2 size={40} />
     <h2>{user.username}'s Gamer Profile
🎮</h2>
    </div>

    <div className="profile-info">
     <p><b>Full Name:</b> {user.name}</p>
     <p><b>Email:</b> {user.email}</p>
     <p><b>Phone:</b> {user.phone}</p>
     <p><b>DOB:</b> {user.dob}</p>
     <p><b>Gender:</b> {user.gender}</p>
     <p><b>Country:</b> {user.country}</p>
    </div>
```

```
        <button className="auth-btn logout-
btn" onClick={logout}>
          Logout 
        </button>
      </motion.div>
    </div>
  );
}
export default Profile ;
```

Output :


Game Hub
Home
Games
About
Contact
Cart
Login
Sign Up
Parth@fire's Gamer Profile
Full Name: Parth
Email: waghmareprathmesh899@gmail.com
Phone: 9309208325
DOB: 2025-11-12
Gender: Male
Country: India
Logout
Game Hub
Play, learn and level up — curated mini-games and brain-boosting fun for everyone.
Quick Links
Home
Games
About
Contact
Cart
Resources
Help Center
Developers
Privacy Policy
Terms of Use
Stay in the loop
Get new game drops, tips & offers — delivered monthly.
your@email.com
Subscribe
By subscribing you agree to our privacy policy.

Conclusion:

We successfully implemented Profile Page.

Assignment 8 : State Management

Aim :

The aim of this assignment is to implement centralized state management in a React-based gaming application (GameHub) using localStorage simulating Context-API behavior. The primary objective is to maintain a global Game Cart system that can be accessed and modified across all pages, ensuring that selected games remain saved even after navigation or refresh.

Additionally, this assignment implements login-based restrictions, ensuring players must be logged in before proceeding to checkout. This adds real-world authentication behavior similar to gaming platforms like Steam / Epic Games / Play Store.

Theory :

State management is the backbone of any modern React application that handles dynamic data, user interactions, and complex workflows. When an application grows beyond a few components, maintaining data consistency across multiple screens using individual component states (useState) becomes inefficient and difficult to scale. In this GameHub-based gaming application, the need for a centralized way to store and manage selected game data led to the use of shared global state logic. This ensures that any game added to the cart from one page is instantly reflected across other components that use the same cart data.

In a typical digital gaming platform like this one, users can browse available games, add them to their gaming cart, preview details, or remove selected titles. Without global state management, this cart data would exist only within the local scope of the component and would get lost when navigating away or refreshing the page. By implementing centralized state logic along with localStorage, the gaming cart becomes persistent across navigation and refreshes, providing a seamless and professional-grade user experience similar to platforms like Steam or Epic Games Store.

The implementation begins with storing and managing the cart data in a single shared store rather than individual components. React's useState hook manages the cart array, which stores all selected games. Every modification — such as adding a game, removing a game, or updating state — is handled through dedicated functions that maintain immutability and avoid accidental overwrites. The useEffect hook ensures that cart data is continuously synced with the browser's localStorage, guaranteeing persistence even after a page reload or browser close. In this way, the system functions similarly to a Context API-powered global state, but remains lightweight and efficient for this scale of application.

After establishing a persistent cart system, authentication control was integrated into the checkout process. In real gaming platforms, users must be logged in before they can proceed to claim, download, or play games. Similarly, this app checks whether the user is logged in by verifying a saved flag (isLoggedIn) in localStorage. If the user is not authenticated, they are prevented from accessing the checkout page and redirected to the login screen. Before redirecting, the app temporarily saves the selected games under a pendingCheckout key so that once the user logs in, their game data is restored automatically. This ensures a smooth experience where users never lose their selected games even if login is required midway.

From an architectural perspective, this setup ensures consistent data flow across the GameHub app. The cart data becomes globally available throughout the platform without passing props manually between components. Pages like Game Listing, Cart, and Checkout all share the same synchronized state. This eliminates prop drilling and minimizes errors caused by isolated local states. Whenever a user adds or removes a game, the UI updates automatically across all relevant pages due to React's reactivity system.

This centralized design gives the Cart page the ability to always display up-to-date selected games and dynamically compute the total cost. The Checkout page also automatically pulls the game list and prevents finalizing an order unless the player is verified. The result is a modern, user-friendly shopping-and-playing flow similar to real digital gaming stores.

One of the most significant advantages of this method is that game data is persistent across navigation and refreshes. Traditional local component state would clear the cart during a page refresh, but by syncing the global state to localStorage, all selections remain intact. Additionally, combining authentication checks with global cart persistence ensures a real-world, secure, and practical system for managing user sessions and purchase-like actions in a gaming environment.

In essence, this implementation demonstrates how centralized state logic and persistent storage can efficiently emulate Context API benefits without the complexity of Redux. It provides a clean, scalable, and secure architecture for managing interactive gaming features and user-level restrictions. The result is a robust foundation for modern React applications designed for user interaction, product-like selections, and controlled access flows.

Result:

A fully functional persistent game cart system was successfully implemented in the GameHub application. The cart data is shared across all components and retained even after page refresh or navigation. Users can add, view, and remove games in real time, and the total value updates automatically. If a user attempts to proceed to checkout without logging in, the system restricts access, redirects them to the login page, and restores their cart afterward. This ensures data consistency, secure user access, and a smooth, real-world gaming experience throughout the entire process.

.

Code :

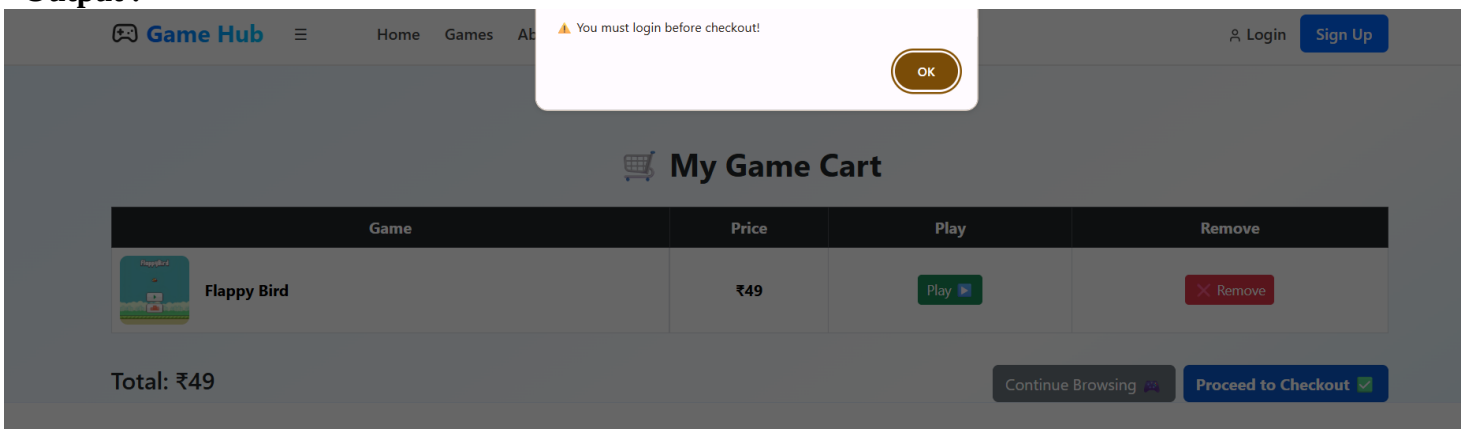
```
import React, { useEffect, useState } from "react";
import { Link, useNavigate } from "react-router-dom";

export default function Cart() {
  const [cartItems, setCartItems] = useState([]);
  const navigate = useNavigate();

  useEffect(() => {
    const savedCart = JSON.parse(localStorage.getItem("cart")) || [];
    setCartItems(savedCart);

    // ✅ Restore cart if user logged in after being redirected
    const isLoggedIn = localStorage.getItem("isLoggedIn");
    const pendingCheckout = JSON.parse(localStorage.getItem("pendingCheckout"));

    if (isLoggedIn && pendingCheckout && savedCart.length === 0) {
      setCartItems(pendingCheckout);
      localStorage.setItem("cart", JSON.stringify(pendingCheckout));
      localStorage.removeItem("pendingCheckout");
    }
  }, []);

  const removeItem = (id) => {
    const updated = cartItems.filter(item => item.id !== id);
    setCartItems(updated);
    localStorage.setItem("cart", JSON.stringify(updated));

    // ✅ Clear pending checkout when cart becomes empty
    if (updated.length === 0) {
      localStorage.removeItem("pendingCheckout");
    }
  };

  const totalPrice = cartItems.reduce((sum, game) => sum + game.price, 0);

  const handleCheckout = () => {
    const isLoggedIn = localStorage.getItem("isLoggedIn");

    if (!isLoggedIn) {
      alert("⚠ You must login before checkout!");
      // ✅ Save cart for restoring after login
      localStorage.setItem("pendingCheckout", JSON.stringify(cartItems));

      navigate("/login");
      return;
    }

    // ✅ Redirect to Checkout with cart & total
    navigate("/checkout", { state: { cartItems, total: totalPrice } });
  };

  return (
    <div className="container" style={{ marginTop: "80px" }}>
      <h2 className="fw-bold text-center mb-4">🛒 My Game Cart</h2>

      {cartItems.length === 0 ? (
        <p className="text-center fs-5">
          Your game cart is empty 😢 <br /><br />
          <Link to="/" className="btn btn-primary">Browse Games</Link>
        </p>
      ) : (
        <>
          <div className="table-responsive">
            <table className="table table-bordered align-middle">
              <thead className="table-dark text-center">
                <tr>
                  <th>Game</th>
                  <th>Price</th>
                  <th>Play</th>
                  <th>Remove</th>
                </tr>
              </thead>
              <tbody>
                {cartItems.map((game) => (
                  <tr key={game.id} className="text-center">
                    <td className="d-flex align-items-center gap-3">
                      <img
                        src={game.img}
                        alt={game.name}
                        style={{ width: "70px", height: "70px", objectFit: "cover", borderRadius: "8px" }}
                      />
                      <strong>{game.name}</strong>
                    </td>
```

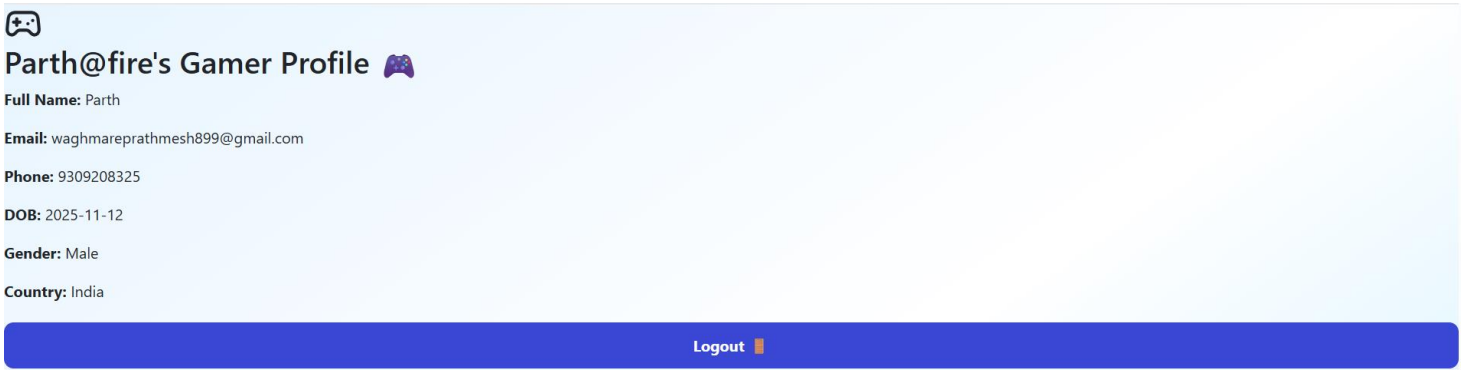
```
                <td className="fw-bold">
                  {game.price === 0 ? "Free" :
`₹${game.price}`}
                </td>

                <td>
                  <a href={game.link}
target="_blank" rel="noopener noreferrer"
className="btn btn-success btn-sm">
                    Play ▶
                  </a>
                </td>

                <td>
                  <button className="btn btn-danger
btn-sm" onClick={() => removeItem(game.id)}>
                    ❌ Remove
                  </button>
                </td>
              </tr>
            ))}
          </tbody>
        </table>
      </div>

      {/* Total & Checkout */}
      <div className="d-flex justify-content-
between align-items-center mt-3">
        <h4>Total: {totalPrice === 0 ? "Free" :
`₹${totalPrice}`}</h4>

        <div className="d-flex gap-2">
          <Link to="/" className="btn btn-
secondary">Continue Browsing 🎮</Link>
          <button className="btn btn-primary fw-
bold" onClick={handleCheckout}>
            Proceed to Checkout ✅
          </button>
        </div>
      </div>
    </>
  )}
 </div>
 );
}

export default Cart;
import React, { useState, useEffect } from "react";
import { useLocation, useNavigate } from "react-
router-dom";

export default function Checkout() {
 const location = useLocation();
 const navigate = useNavigate();

 const { cartItems = [], total = 0 } = location.state ||
{};

 const [restoredCart, setRestoredCart] =
useState([]);
 const displayCart = cartItems.length > 0 ? cartItems
: restoredCart;
 const totalAmount = displayCart.reduce((sum,
game) => sum + game.price, 0);

 const [formData, setFormData] = useState({
  username: "",
  email: "",
  address: "",
  city: "",
  pincode: "",
 });

 // ✅ Check login + restore saved cart if redirected
from login
 useEffect(() => {
  const isLoggedIn =
localStorage.getItem("isLoggedIn");
  if (!isLoggedIn) {
   alert("⚠ Please login to continue checkout.");
   navigate("/login");
   return;
  }

  // ✅ Restore cart if saved as pending checkout
  const pendingData =
JSON.parse(localStorage.getItem("pendingCheckout
"));
  if (pendingData && cartItems.length === 0) {
   setRestoredCart(pendingData);
   localStorage.removeItem("pendingCheckout");
  }
 }, []);

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

 const handlePlaceOrder = () => {
  if (
   !formData.username ||
   !formData.email ||
   !formData.address ||
   !formData.city ||
   !formData.pincode
  ) {
```

```
    alert("⚠ Please fill all details to continue.");
    return;
  }

  alert("🎉 Order Successful! Enjoy your games!");

  // ✅ Clear cart fully
  localStorage.removeItem("cart");
  localStorage.removeItem("pendingCheckout");

  navigate("/");
 };

 return (
  <div className="container mt-5 pt-4">
   <h2 className="fw-bold text-center mb-4">🛍 Checkout</h2>

   <div className="row">

    {/* ✅ User Form */}
    <div className="col-md-6 mb-4">
     <h4 className="mb-3">Player Details 🎮</h4>
     <form>
      <div className="mb-3">
       <label className="form-label">Name</label>
       <input
        type="text"
        name="username"
        className="form-control"
        value={formData.username}
        onChange={handleChange}
        placeholder="Enter your full name"
       />
      </div>

      <div className="mb-3">
       <label className="form-label">Email</label>
       <input
        type="email"
        name="email"
        className="form-control"
        value={formData.email}
        onChange={handleChange}
        placeholder="Enter your email"
       />
      </div>

      <div className="mb-3">
       <label className="form-label">Address</label>
       <textarea
        name="address"
        className="form-control"
        value={formData.address}
        onChange={handleChange}
        placeholder="House / Street / Area"
       ></textarea>
      </div>

      <div className="mb-3">
       <label className="form-label">City</label>
       <input
        type="text"
        name="city"
        className="form-control"
        value={formData.city}
        onChange={handleChange}
        placeholder="Enter city"
       />
      </div>

      <div className="mb-3">
       <label className="form-label">Pincode</label>
       <input
        type="text"
        name="pincode"
        className="form-control"
        value={formData.pincode}
        onChange={handleChange}
        placeholder="Enter pincode"
       />
      </div>
     </form>
    </div>

    {/* ✅ Order Summary */}
    <div className="col-md-6">
     <h4 className="mb-3">Order Summary □</h4>

     {displayCart.length === 0 ? (
      <p>No games selected.</p>
     ) : (
      <ul className="list-group mb-3">
       {displayCart.map((game) => (
        <li
         key={game.id}
         className="list-group-item d-flex justify-content-between align-items-center"
        >
```

```
              <div>
                <strong>{game.name}</strong>
                <p className="text-muted small mb-
0">1 Game</p>
              </div>
              <span>
                {game.price === 0 ? "Free" :
`₹${game.price}`}
              </span>
            </li>
          ))}

          <li className="list-group-item d-flex
justify-content-between">
            <strong>Total</strong>
            <strong>{totalAmount === 0 ? "Free" :
`₹${totalAmount}`}</strong>
          </li>
        </ul>
      )}

      <button
        className="btn btn-primary w-100 fw-
bold"
        onClick={handlePlaceOrder}
      >
        ✅ Place Order & Play
      </button>
    </div>
  </div>
 </div>
 );
}
export default Checkout;;
```

Output :





Conclusion:

This assignment demonstrates how React's localStorage-based global cart logic and authentication checks effectively manage user state in a gaming web application. By centralizing the game cart handling, the GameHub platform maintains a clean and modular structure where selected games remain accessible across pages without data loss. The login-based restriction ensures that only authenticated players can proceed to checkout, simulating real-world access control used in membership-based gaming platforms.

This implementation highlights how a lightweight state-sharing approach can be equally powerful as advanced state management tools for small to medium applications. With persistent cart data, seamless cross-page access, and secure checkout flow, the GameHub project reflects a polished and professional handling of state and user sessions in modern React environments — without the overhead of external libraries like Redux

.

Assignment 9 : Routing and Component Structure

Aim :

The aim of this assignment is to implement seamless navigation and modular component architecture in a React-based Game Hub application using React Router. The main objective is to build a single-page application structure where users can navigate between pages such as Home, Games, Cart, Checkout, Login, and Profile without reloading the browser. Additionally, the project focuses on breaking the interface into reusable UI components like Navbar, GameCard, and Footer to ensure maintainable, scalable, and structured code. Through this assignment, efficient routing, reusable components, and proper UI flow are applied to create a smooth gaming experience platform.

Theory :

Modern web applications require fast navigation, interactive features, and a smooth user experience. React provides the foundation for developing such single-page applications (SPAs), where components dynamically update without refreshing the entire browser window. However, for structured navigation between multiple pages in a SPA, React Router becomes an essential tool. Instead of loading new HTML pages, React Router renders different components based on URL changes, creating the illusion of multi-page navigation while maintaining high performance. This makes the application feel responsive, scalable, and user-friendly—crucial requirements for an interactive Game Hub platform.

In this Game Hub project, React Router DOM is used to define routes for Home, Games, Cart, Checkout, Login, and Profile pages. The application is wrapped inside <BrowserRouter> allowing global routing functionality. Each route path, such as "/games" or "/cart", is linked to a corresponding component using the <Route> element inside the <Routes> wrapper. This ensures that navigating to different sections of the application instantly switches to the respective component without a page refresh, preserving the single-page architecture. Users can browse games, add them to their personal cart, view selections, and proceed to checkout seamlessly.

The Navbar plays a crucial role in navigation, acting as a persistent component visible across all pages. It uses React Router's <NavLink> and <Link> components instead of traditional anchor tags. This ensures instant client-side routing and prevents full page reloads. The Navbar also conditionally shows Login, Profile, and Cart options based on user authentication status stored in localStorage, providing both usability and real-world logic similar to professional web platforms.

Component modularity is another core part of this assignment. The entire UI is divided into reusable and self-contained components such as Navbar, GameCard, Footer, and Cart Item structures. Each component holds its own UI logic and styling, making the code more organized and maintainable. For example, GameCard displays each game's image, name, price, level tag, gameplay link, and "Add to My Games" button. This component can easily render multiple games by simply passing data through props, eliminating repeated markup and ensuring clean code practices.

This modular structure improves scalability—new games or UI sections can be added with minimal changes. If additional pages like Leaderboards or Wishlist were introduced in the future, they could be integrated simply by creating new components and assigning new routes. The separation of concerns ensures each component handles one responsibility, making debugging and updating the project easier.

Furthermore, protected route behavior is implemented for the Checkout page to enhance user flow and control access. When a user tries to proceed to checkout without being logged in, the application redirects them to the Login page instead of allowing access. This is achieved by checking the login status in localStorage before navigating. Such conditional routing mimics real-life gaming platforms and enhances application security, while also preserving the user's selected games for later retrieval after login.

Overall, this project successfully demonstrates efficient SPA navigation using React Router combined with modular, reusable components that enhance code clarity and development scalability. The application maintains performance and user experience consistency as components re-render only when necessary. The smooth transitions, component-based structure, and authentication-based routing logic reflect modern front-end engineering practices suited for scalable industry-level projects like online game platforms.

A fully functional single-page Game Hub application was developed with smooth navigation across pages like Home, Games, Cart, Checkout, Login, and Profile using React Router. Reusable components such as Navbar and Footer improved UI consistency and code organization. Client-side routing provides instant transitions without page reloads, delivering a fast and interactive user experience. Login-based protected navigation ensures that only authenticated users access the checkout process. The final system demonstrates proper integration of routing, modular component design, and real-world navigation logic in a professional-grade web application interface.

Code :

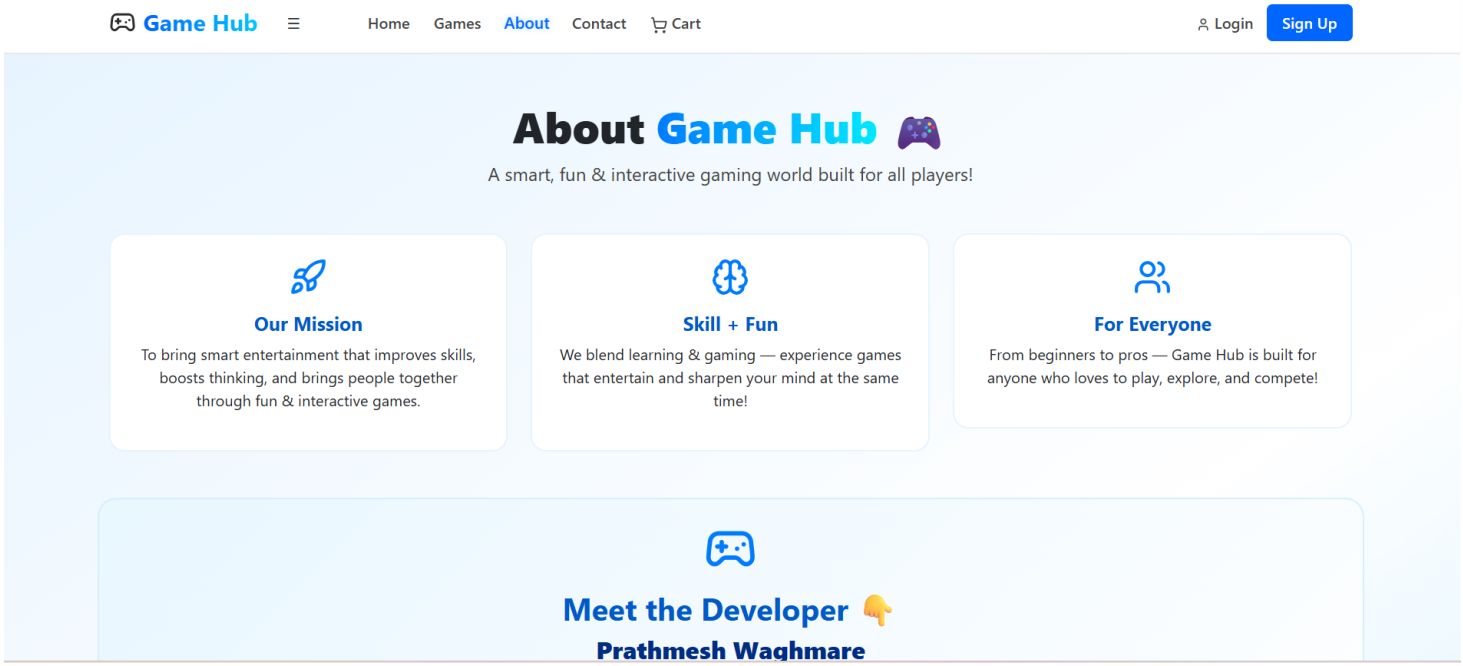
```
import React from "react";
import { useLocation, useNavigate } from "react-
router-dom";
import { motion } from "framer-motion";

export default function GameDetails() {
  const { state: game } = useLocation();
  const navigate = useNavigate();

  const handleAddToGames = () => {
    let library =
JSON.parse(localStorage.getItem("myGames")) || [];

    // prevent duplicate
    if (!library.find(g => g.id === game.id)) {
      library.push(game);
      localStorage.setItem("myGames",
JSON.stringify(library));
      alert(`${game.name} added to My Games ✅`);
    } else {
      alert("Already added ✓");
    }
  };

  return (
    <motion.div
      className="game-details-container"
      initial={{ opacity: 0, scale: 0.95 }}
      animate={{ opacity: 1, scale: 1 }}
      transition={{ duration: 0.4 }}
    >
      <img src={game.img} alt={game.name}
classNAme="details-img" />

      <h1 classNAme="details-
title">{game.name}</h1>
      <p classNAme="details-desc">
        Get ready to dive into <b>{game.name}</b>!
This game enhances focus,
        reaction skills, and strategic thinking. Enjoy and
level up your mind!
      </p>

      <ul classNAme="details-features">
        <li>✅ Fun & Engaging</li>
        <li>⚡ Improves Focus & Reflex</li>
        <li>🎯 Suitable for all ages</li>
        <li>🌍 Play anytime, anywhere</li>
      </ul>

      <div classNAme="details-btn-group">
      <button
    onClick={() => window.open(game.link, "_blank")}
    classNAme="play-now-btn"
>
    Play Now ▶
</button>
```

```
      <button onClick={handleAddToGames}
className="play-now-btn">
        + Add to My Games
      </button>

      <button onClick={() => navigate(-1)}
className="back-btn">
        ⬅ Back
      </button>
    </div>
  </motion.div>
 );
}
export default GameDetails;
import React, { useState } from "react";
import { Link, NavLink } from "react-router-dom";
import { Gamepad2, ShoppingCart, User } from
"lucide-react";

export default function Navbar() {
 const [open, setOpen] = useState(false);

 return (
  <nav className="navbar-container">
   <div className="nav-inner container">

    {/* Logo */}
    <Link to="/" className="nav-logo">
     <Gamepad2 size={28} /> <span>Game
Hub</span>
    </Link>

    {/* ✅ Mobile Menu Button */}
    <div className="menu-toggle" onClick={() =>
setOpen(!open)}>
      ☰
    </div>

    {/* Menu */}
    <div className={`nav-links ${open ? "active" :
""}`}>
      <NavLink to="/" className="nav-
item">Home</NavLink>
      <NavLink to="/games" className="nav-
item">Games</NavLink>
      <NavLink to="/about" className="nav-
item">About</NavLink>
      <NavLink to="/contact" className="nav-
item">Contact</NavLink>
      <NavLink to="/cart" className="nav-item
cart-btn">
       <ShoppingCart size={18} /> Cart
      </NavLink>
    </div>

    {/* Auth Buttons */}
    <div className={`auth-section ${open ?
"active" : ""}`}>
      <Link to="/login" className="nav-login">
       <User size={15}/> Login
      </Link>
      <Link to="/register" className="nav-
register">
       Sign Up
      </Link>
    </div>
    {/* Mobile Dropdown */}
    <div className={`mobile-menu ${open ?
"active" : ""}`}>
     <NavLink to="/" className="nav-
item">Home</NavLink>
     <NavLink to="/games" className="nav-
item">Games</NavLink>
     <NavLink to="/about" className="nav-
item">About</NavLink>
     <NavLink to="/contact" className="nav-
item">Contact</NavLink>
     <NavLink to="/cart" className="nav-item">
     <ShoppingCart size={18} /> Cart
     </NavLink>
     <Link to="/login" className="nav-login"><User
size={15}/> Login</Link>
     <Link to="/register" className="nav-
register">Sign Up</Link>
    </div>

   </div>
  </nav>
 );
}
export default Navbar;
import React, { useState } from "react";
import { Link } from "react-router-dom";
import { Twitter, Instagram, Facebook, Linkedin }
from "lucide-react";

export default function Footer() {
 const [email, setEmail] = useState("");

 const submitNewsletter = (e) => {
  e.preventDefault();
  if (!email) {
   alert("Please enter an email.");
   return;
  }
  alert(`Thanks — ${email} subscribed to Game Hub
newsletter!`);
```

```
  setEmail("");
 };

 return (
  <footer className="gv-footer">
   <div className="gv-footer-top container">
    {/* Column 1: Brand + desc */}
    <div className="gv-col gv-brand">
     <div className="gv-logo">🎮 <span
className="gv-logo-text">Game
Hub</span></div>
     <p className="gv-desc">
      Play, learn and level up — curated mini-
games and brain-boosting fun for everyone.
     </p>
     <div className="gv-socials" aria-
label="Follow Game Hub">
      <a href="https://twitter.com" target="_blank"
rel="noopener noreferrer" aria-label="Twitter">
       <Twitter />
      </a>
      <a href="https://instagram.com"
target="_blank" rel="noopener noreferrer" aria-
label="Instagram">
       <Instagram />
      </a>
      <a href="https://facebook.com"
target="_blank" rel="noopener noreferrer" aria-
label="Facebook">
       <Facebook />
      </a>
      <a href="https://linkedin.com"
target="_blank" rel="noopener noreferrer" aria-
label="LinkedIn">
       <Linkedin />
      </a>
     </div>
    </div>

    {/* Column 2: Quick Links */}
    <div className="gv-col">
     <h4>Quick Links</h4>
     <nav className="gv-links" aria-label="Quick
links">
      <Link to="/">Home</Link>
      <Link to="/games">Games</Link>
      <Link to="/about">About</Link>
      <Link to="/contact">Contact</Link>
      <Link to="/cart">Cart</Link>
     </nav>
    </div>

    {/* Column 3: Resources */}
    <div className="gv-col">
     <h4>Resources</h4>
     <nav className="gv-links" aria-
label="Resources links">
      <Link to="/contact">Help Center</Link>
      <Link to="/about">Developers</Link>
      <Link to="/privacy">Privacy Policy</Link>
      <Link to="/terms">Terms of Use</Link>
     </nav>
    </div>

    {/* Column 4: Newsletter */}
    <div className="gv-col">
     <h4>Stay in the loop</h4>
     <p className="small">Get new game drops,
tips & offers — delivered monthly.</p>

     <form className="gv-newsletter"
onSubmit={submitNewsletter}>
      <input
       type="email"
       placeholder="your@email.com"
       value={email}
       onChange={(e) => setEmail(e.target.value)}
       aria-label="Email for newsletter"
      />
      <button type="submit" className="btn-
sub">Subscribe</button>
     </form>

     <p className="gv-minor small">
      By subscribing you agree to our <Link
to="/privacy">privacy policy</Link>.
     </p>
    </div>
   </div>

   <div className="gv-footer-bottom">
    <div className="container d-flex justify-
content-between align-items-center">
     <div className="copyright">
      © {new Date().getFullYear()} Game Hub •
Built by <strong>Prathmesh Waghmare</strong>
     </div>

     <div className="gv-made small">
      Designed with ♥ — bright, friendly and
accessible UI
     </div>
    </div>
   </div>
  </footer>
 );
}
```

```
export default Footer;
```

Output :



Conclusion :

We successfully implemented Components.

Assignment 10 : Full React Application

Aim :

The aim of this assignment is to set up a new React project using Create React App, configure the initial project structure, and implement basic navigation across multiple pages using React Router DOM. The primary goal is to understand how routing works within single-page applications and to create a top navigation bar that links key sections such as Home, Games, Cart, Checkout, and Profile, allowing smooth transitions without full- page reloads.

Theory :

- **All About React**

  This project begins with the creation of a React application using Create React App, which provides an automatically configured React development environment including Webpack and Babel. The source code structure is organized into components, pages, and context folders, supporting modular development.
  React Router DOM is used to enable seamless client-side navigation. In traditional web applications, changing pages leads to full browser reloads. However, using SPA architecture, only the required component updates while the rest stays intact. BrowserRouter is used to wrap the app, while Routes and Route define individual paths like /, /games, /cart, and /checkout.
  The Navbar, created as a reusable component, uses Link from React Router DOM instead of <a> tags to avoid page reloads. This ensures instant and smooth navigation throughout the application..

- **Home Page**

  The Home page serves as the primary interface between users and the Game Hub application. It is designed to be visually engaging, modular, and user-friendly. React's component-driven architecture allows each section of the Home page to be built as an independent component, such as the welcome banner, featured game showcase, and navigation prompts. Rather than displaying products, the Home page highlights popular and trending games, encouraging users to explore further.

  In this project, the Home page showcases selected featured games taken from a predefined game list. Each game is represented using a **Game Card** component, displaying the game image, title, level, and access button. The UI layout uses modern responsive CSS flex and grid styling to ensure a smooth experience across mobile and desktop screens. Interactive buttons and motion animations enhance the immersive feel of the platform. When users interact, React state management provides immediate UI responses, demonstrating reactivity and smooth rendering.

  The overall design emphasizes user engagement and accessibility, offering intuitive gameplay discovery and instant navigation to the Games page. This illustrates how React components, styling, and event-driven logic combine to build a dynamic, modern landing experience for gaming platforms.

- **Games Page**

The Games page is the core section of the Game Hub platform, showcasing all available games in a structured and interactive format. Each game is displayed using a reusable Game Card component, supporting clean modularity. The page uses .map() to loop over an array of game objects and dynamically render game cards with consistent UI presentation.
A search bar is implemented at the top to allow users to filter games by name in real time. React's useState hook stores user input and updates the visible game list by applying JavaScript's filter() function. Every Game Card contains two main actions: Play Now, which navigates the user to the selected game, and Add to My Games, which stores the game in global saved items using Context API/localStorage. Smooth hover animations, clean typography, and responsive grid layout ensure a visually polished game collection page.
The Games page demonstrates real-time interactivity, reusable component structure, and efficient client-side rendering — clearly reflecting modern React development patterns.

- **Cart Page**

The Cart page represents the user's personalized collection where saved games are displayed. Instead of physical items, it manages and lists digital games that the user has added. Each game card shows the game name, difficulty level, price (if applicable), and access option. React's state and Context API ensure instant updates when items are added or removed.
The page maps saved games dynamically using. map() and displays them in a clean playable list. If the user selects a paid game, it contributes to the total price shown on the page. A **Proceed to Checkout** button is provided for paid selections, redirecting users to the checkout screen via React Router without page reload.
This component highlights real-time UI state updates, persistent data handling, and smooth routing transitions, reinforcing the event-driven nature of React and the advantages of centralized state management.

- **Checkout Page**

The Checkout page simulates an online game-purchase and access experience. While no physical delivery exists, the page collects basic user information such as name, email, and payment details using controlled input fields. Each field is linked to React state to maintain real-time synchronization between the form and UI.
Validation ensures that all mandatory fields are completed before allowing users to finalize their checkout. The selected games list is retrieved from saved state/localStorage and displayed as a summary. After successful form submission, a confirmation message is shown and users are redirected, simulating a real digital-checkout flow.
React Router and state logic are used to control page access, ensuring only authenticated users can reach the checkout page. This demonstrates real-world application of access-protected navigation using React Router and session-based validation.

- **Registration , Login and Profile**

React's form handling and state logic are used to build registration and login pages. The forms capture

user email, password, and other basic data as controlled components. Validation checks ensure proper form input before permitting login or registration. Upon successful login, user data and session flags are stored using localStorage so the login state persists across reloads.
The Profile page retrieves stored user information and displays it in a structured manner, including username and email. Conditional rendering inside the Navbar automatically switches between **Login/Sign Up** and **My Profile/Logout** based on login status, ensuring personalized user experience and session-aware UI behavior.
This implementation effectively demonstrates user authentication simulation, session storage, and conditional UI rendering without external backend services, making it a complete front-end identity handling workflow.

- **State Management Using Context API**

React's Context API is used to manage the global "My Games" storage and user session data without prop drilling. A GameContext (similar to CartContext in e-commerce) maintains game data and provides methods such as addGame(), removeGame(), and session validation helpers. By wrapping the application with a Context Provider, all components can access shared game data instantly.
LocalStorage integration ensures that all game selections and login state persist across browser refreshes. Before accessing the Checkout page, login validation verifies whether the user is authenticated; otherwise, users are redirected to login — implementing route protection behavior.
This centralized state model greatly simplifies management, improves reusability, and ensures scalability for future game additions and advanced features.

- **Component Structure**

React Router DOM is used to enable seamless navigation between pages of the Game Hub platform. Components like BrowserRouter, Routes, and Route control which component is displayed based on the route path. The Navbar and Footer remain constant across all views, while Route dynamically renders pages such as Home, Games, My Games, Checkout, Login, and Profile.
Modular component design ensures reusability and clean project architecture. UI elements including Navbar, Footer, and GameCard are built independently and imported as needed. This separation of concerns improves readability, simplifies maintenance, and enables easy future expansion such as adding leaderboards, tournaments, or user achievements.
The project showcases a fully structured Single Page Application with client-side routing and reusable components, achieving a scalable and efficient React-based gaming portal system.

Code :

```
import React, { useState } from "react";
import { Link, NavLink } from "react-router-dom";
import { Gamepad2, ShoppingCart, User } from "lucide-react";

export default function Navbar() {
  const [open, setOpen] = useState(false);

  return (
    <nav className="navbar-container">
      <div className="nav-inner container">
```

```
      {/* Logo */}
      <Link to="/" className="nav-logo">
        <Gamepad2 size={28} /> <span>Game Hub</span>
      </Link>

      {/* ✅ Mobile Menu Button */}
      <div className="menu-toggle" onClick={() => setOpen(!open)}>
        ☰
      </div>

      {/* Menu */}
      <div className={`nav-links ${open ? "active" : ""}`}>
        <NavLink to="/" className="nav-item">Home</NavLink>
        <NavLink to="/games" className="nav-item">Games</NavLink>
        <NavLink to="/about" className="nav-item">About</NavLink>
        <NavLink to="/contact" className="nav-item">Contact</NavLink>
        <NavLink to="/cart" className="nav-item cart-btn">
          <ShoppingCart size={18} /> Cart
        </NavLink>
      </div>

      {/* Auth Buttons */}
      <div className={`auth-section ${open ? "active" : ""}`}>
        <Link to="/login" className="nav-login">
          <User size={15}/> Login
        </Link>
        <Link to="/register" className="nav-register">
          Sign Up
        </Link>
      </div>
      {/* Mobile Dropdown */}
      <div className={`mobile-menu ${open ? "active" : ""}`}>
      <NavLink to="/" className="nav-item">Home</NavLink>
       <NavLink to="/games" className="nav-item">Games</NavLink>
      <NavLink to="/about" className="nav-item">About</NavLink>
      <NavLink to="/contact" className="nav-item">Contact</NavLink>
      <NavLink to="/cart" className="nav-item">
      <ShoppingCart size={18} /> Cart
      </NavLink>
      <Link to="/login" className="nav-login"><User size={15}/> Login</Link>
      <Link to="/register" className="nav-register">Sign Up</Link>
      </div>

     </div>
    </nav>
  );
}

export default Navbar;
import React from "react";
import { useLocation, useNavigate } from "react-router-dom";
import { motion } from "framer-motion";

export default function GameDetails() {
  const { state: game } = useLocation();
  const navigate = useNavigate();

  const handleAddToGames = () => {
    let library = JSON.parse(localStorage.getItem("myGames")) || [];
```

```
    // prevent duplicate
    if (!library.find(g => g.id === game.id)) {
      library.push(game);
      localStorage.setItem("myGames",
JSON.stringify(library));
      alert(`${game.name} added to My Games
✅`);
    } else {
      alert("Already added ✓");
    }
  };

  return (
    <motion.div
      className="game-details-container"
      initial={{ opacity: 0, scale: 0.95 }}
      animate={{ opacity: 1, scale: 1 }}
      transition={{ duration: 0.4 }}
    >
      <img src={game.img} alt={game.name}
className="details-img" />

      <h1 className="details-
title">{game.name}</h1>
      <p className="details-desc">
        Get ready to dive into
<b>{game.name}</b>! This game enhances
focus,
        reaction skills, and strategic thinking.
Enjoy and level up your mind!
      </p>

      <ul className="details-features">
        <li>✅ Fun & Engaging</li>
        <li>⚡ Improves Focus & Reflex</li>
        <li>🎯 Suitable for all ages</li>
        <li>🌍 Play anytime, anywhere</li>
      </ul>

      <div className="details-btn-group">
      <button
  onClick={() => window.open(game.link,
"_blank")}
  className="play-now-btn"
>
  Play Now ▶
</button>



        <button onClick={handleAddToGames}
className="play-now-btn">
          + Add to My Games
        </button>

        <button onClick={() => navigate(-1)}
className="back-btn">
          ⬅ Back
        </button>
      </div>
    </motion.div>
  );
}

export default GameDetails;

import React, { useState } from "react";
import { Link } from "react-router-dom";
import { Twitter, Instagram, Facebook, Linkedin
} from "lucide-react";

export default function Footer() {
  const [email, setEmail] = useState("");

  const submitNewsletter = (e) => {
    e.preventDefault();
    if (!email) {
      alert("Please enter an email.");
```

```
    return;
  }
  alert(`Thanks — ${email} subscribed to Game Hub newsletter!`);
  setEmail("");
 };

 return (
  <footer className="gv-footer">
   <div className="gv-footer-top container">
    {/* Column 1: Brand + desc */}
    <div className="gv-col gv-brand">
     <div className="gv-logo">🎮 <span className="gv-logo-text">Game Hub</span></div>
     <p className="gv-desc">
      Play, learn and level up — curated mini-games and brain-boosting fun for everyone.
     </p>
     <div className="gv-socials" aria-label="Follow Game Hub">
      <a href="https://twitter.com" target="_blank" rel="noopener noreferrer" aria-label="Twitter">
       <Twitter />
      </a>
      <a href="https://instagram.com" target="_blank" rel="noopener noreferrer" aria-label="Instagram">
       <Instagram />
      </a>
      <a href="https://facebook.com" target="_blank" rel="noopener noreferrer" aria-label="Facebook">
       <Facebook />
      </a>
      <a href="https://linkedin.com" target="_blank" rel="noopener noreferrer" aria-label="LinkedIn">
       <Linkedin />
      </a>
     </div>
    </div>

    {/* Column 2: Quick Links */}
    <div className="gv-col">
     <h4>Quick Links</h4>
     <nav className="gv-links" aria-label="Quick links">
      <Link to="/">Home</Link>
      <Link to="/games">Games</Link>
      <Link to="/about">About</Link>
      <Link to="/contact">Contact</Link>
      <Link to="/cart">Cart</Link>
     </nav>
    </div>

    {/* Column 3: Resources */}
    <div className="gv-col">
     <h4>Resources</h4>
     <nav className="gv-links" aria-label="Resources links">
      <Link to="/contact">Help Center</Link>
      <Link to="/about">Developers</Link>
      <Link to="/privacy">Privacy Policy</Link>
      <Link to="/terms">Terms of Use</Link>
     </nav>
    </div>

    {/* Column 4: Newsletter */}
    <div className="gv-col">
     <h4>Stay in the loop</h4>
     <p className="small">Get new game drops, tips & offers — delivered monthly.</p>

     <form className="gv-newsletter" onSubmit={submitNewsletter}>
      <input
       type="email"
```

```
          placeholder="your@email.com"
          value={email}
          onChange={(e) =>
setEmail(e.target.value)}
          aria-label="Email for newsletter"
         />
         <button type="submit" className="btn-
sub">Subscribe</button>
        </form>

        <p className="gv-minor small">
         By subscribing you agree to our <Link
to="/privacy">privacy policy</Link>.
        </p>
       </div>
     </div>

     <div className="gv-footer-bottom">
      <div className="container d-flex justify-
content-between align-items-center">
        <div className="copyright">
         © {new Date().getFullYear()} Game Hub
• Built by <strong>Prathmesh
Waghmare</strong>
        </div>

        <div className="gv-made small">
         Designed with ♥ — bright, friendly and
accessible UI
        </div>
       </div>
     </div>
   </footer>
  );
}

export default Footer;
import React from "react";
import { useLocation, useNavigate } from
"react-router-dom";
import { motion } from "framer-motion";

export default function GameDetails() {
  const { state: game } = useLocation();
  const navigate = useNavigate();

  const handleAddToGames = () => {
   let library =
JSON.parse(localStorage.getItem("myGames"))
|| [];

   // prevent duplicate
   if (!library.find(g => g.id === game.id)) {
    library.push(game);
    localStorage.setItem("myGames",
JSON.stringify(library));
    alert(`${game.name} added to My Games
✅`);
   } else {
    alert("Already added ✓");
   }
  };

  return (
   <motion.div
    className="game-details-container"
    initial={{ opacity: 0, scale: 0.95 }}
    animate={{ opacity: 1, scale: 1 }}
    transition={{ duration: 0.4 }}
   >
    <img src={game.img} alt={game.name}
className="details-img" />

    <h1 className="details-
title">{game.name}</h1>
    <p className="details-desc">
     Get ready to dive into <b>{game.name}</b>!
This game enhances focus,
     reaction skills, and strategic thinking. Enjoy
and level up your mind!
```

```
    </p>

    <ul className="details-features">
      <li>✅ Fun & Engaging</li>
      <li>⚡ Improves Focus & Reflex</li>
      <li>🎯 Suitable for all ages</li>
      <li>🌍 Play anytime, anywhere</li>
    </ul>

    <div className="details-btn-group">
    <button
  onClick={() => window.open(game.link,
"_blank")}
  className="play-now-btn"
>
  Play Now ▶
</button>



      <button onClick={handleAddToGames}
className="play-now-btn">
        + Add to My Games
      </button>

      <button onClick={() => navigate(-1)}
className="back-btn">
        ⬅ Back
      </button>
    </div>
   </motion.div>
  );
}
import React, { useEffect, useState } from
"react";
import { Link, useNavigate } from "react-
router-dom";

export default function Cart() {
  const [cartItems, setCartItems] = useState([]);
  const navigate = useNavigate();

  useEffect(() => {
    const savedCart =
JSON.parse(localStorage.getItem("cart")) || [];
    setCartItems(savedCart);

    // ✅ Restore cart if user logged in after being
redirected
    const isLoggedIn =
localStorage.getItem("isLoggedIn");
    const pendingCheckout =
JSON.parse(localStorage.getItem("pendingChec
kout"));

    if (isLoggedIn && pendingCheckout &&
savedCart.length === 0) {
      setCartItems(pendingCheckout);
      localStorage.setItem("cart",
JSON.stringify(pendingCheckout));

localStorage.removeItem("pendingCheckout");
    }
  }, []);

  const removeItem = (id) => {
    const updated = cartItems.filter(item =>
item.id !== id);
    setCartItems(updated);
    localStorage.setItem("cart",
JSON.stringify(updated));

    // ✅ Clear pending checkout when cart
becomes empty
    if (updated.length === 0) {

localStorage.removeItem("pendingCheckout");
    }
  };
```

```
  const totalPrice = cartItems.reduce((sum, game) => sum + game.price, 0);

  const handleCheckout = () => {
    const isLoggedIn = localStorage.getItem("isLoggedIn");

    if (!isLoggedIn) {
      alert("⚠️ You must login before checkout!");

      // ✅ Save cart for restoring after login
      localStorage.setItem("pendingCheckout", JSON.stringify(cartItems));

      navigate("/login");
      return;
    }

    // ✅ Redirect to Checkout with cart & total
    navigate("/checkout", { state: { cartItems, total: totalPrice } });
  };

  return (
    <div className="container" style={{ marginTop: "80px" }}>
      <h2 className="fw-bold text-center mb-4">🛒 My Game Cart</h2>

      {cartItems.length === 0 ? (
        <p className="text-center fs-5">
          Your game cart is empty 😢 <br /><br />
          <Link to="/" className="btn btn-primary">Browse Games</Link>
        </p>
      ) : (
        <>
          <div className="table-responsive">
            <table className="table table-bordered align-middle">
              <thead className="table-dark text-center">
                <tr>
                  <th>Game</th>
                  <th>Price</th>
                  <th>Play</th>
                  <th>Remove</th>
                </tr>
              </thead>
              <tbody>
                {cartItems.map((game) => (
                  <tr key={game.id} className="text-center">
                    <td className="d-flex align-items-center gap-3">
                      <img
                        src={game.img}
                        alt={game.name}
                        style={{ width: "70px", height: "70px", objectFit: "cover", borderRadius: "8px" }}
                      />
                      <strong>{game.name}</strong>
                    </td>

                    <td className="fw-bold">
                      {game.price === 0 ? "Free" : `₹${game.price}`}
                    </td>

                    <td>
                      <a href={game.link} target="_blank" rel="noopener noreferrer" className="btn btn-success btn-sm">
                        Play ▶
                      </a>
                    </td>
```

```
            <td>
              <button className="btn btn-danger btn-sm" onClick={() => removeItem(game.id)}>
                ❌ Remove
              </button>
            </td>
          </tr>
        ))}
      </tbody>
    </table>
  </div>

  {/* Total & Checkout */}
  <div className="d-flex justify-content-between align-items-center mt-3">
    <h4>Total: {totalPrice === 0 ? "Free" : `₹${totalPrice}`}</h4>

    <div className="d-flex gap-2">
      <Link to="/" className="btn btn-secondary">Continue Browsing 🎮</Link>
      <button className="btn btn-primary fw-bold" onClick={handleCheckout}>
        Proceed to Checkout ✅
      </button>
    </div>
  </div>
</>
```

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

/* Optional inline styles object (unchanged) */
const styles = {
container: { maxWidth: "700px", margin: "50px auto", padding: "20px" },
title: { fontSize: "28px", fontWeight: "bold", marginBottom: "20px", textAlign: "center" },
empty: { fontSize: "18px", textAlign: "center" },
card: {
display: "flex", alignItems: "center", gap: "15px", padding: "15px",
background: "#fff", borderRadius: "12px", marginBottom: "15px",
boxShadow: "0 4px 10px rgba(0,0,0,0.1)"
},
image: { width: "80px", height: "80px", borderRadius: "10px", objectFit: "cover" },
details: { flex: 1 },
name: { fontSize: "20px", marginBottom: "10px" },
removeBtn: {
background: "#ff4d4d", color: "#fff", border: "none",
padding: "8px 14px", borderRadius: "8px", cursor: "pointer",
fontSize: "14px", transition: "0.2s"
}
};
import React, { useState, useEffect } from "react";
import { useLocation, useNavigate } from "react-router-dom";

export default function Checkout() {
const location = useLocation();
const navigate = useNavigate();

const { cartItems = [], total = 0 } = location.state || {};

const [restoredCart, setRestoredCart] = useState([]);
const displayCart = cartItems.length > 0 ? cartItems : restoredCart;
const totalAmount = displayCart.reduce((sum,
```

```
game) => sum + game.price, 0);

  const [formData, setFormData] = useState({
    username: "",
    email: "",
    address: "",
    city: "",
    pincode: "",
  });

  // ✅ Check login + restore saved cart if redirected from login
  useEffect(() => {
    const isLoggedIn = localStorage.getItem("isLoggedIn");
    if (!isLoggedIn) {
      alert("⚠️ Please login to continue checkout.");
      navigate("/login");
      return;
    }

    // ✅ Restore cart if saved as pending checkout
    const pendingData = JSON.parse(localStorage.getItem("pendingCheckout"));
    if (pendingData && cartItems.length === 0) {
      setRestoredCart(pendingData);
      localStorage.removeItem("pendingCheckout");
    }
  }, []);

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

  const handlePlaceOrder = () => {
    if (
      !formData.username ||
      !formData.email ||
      !formData.address ||
      !formData.city ||
      !formData.pincode
    ) {
      alert("⚠️ Please fill all details to continue.");
      return;
    }

    alert("🎉 Order Successful! Enjoy your games!");

    // ✅ Clear cart fully
    localStorage.removeItem("cart");
    localStorage.removeItem("pendingCheckout");

    navigate("/");
  };

  return (
    <div className="container mt-5 pt-4">
      <h2 className="fw-bold text-center mb-4">🛍️ Checkout</h2>

      <div className="row">

        {/* ✅ User Form */}
        <div className="col-md-6 mb-4">
          <h4 className="mb-3">Player Details 🎮</h4>
          <form>
            <div className="mb-3">
              <label className="form-label">Name</label>
```

```
          <input
            type="text"
            name="username"
            className="form-control"
            value={formData.username}
            onChange={handleChange}
            placeholder="Enter your full name"
          />
        </div>

        <div className="mb-3">
          <label className="form-label">Email</label>
          <input
            type="email"
            name="email"
            className="form-control"
            value={formData.email}
            onChange={handleChange}
            placeholder="Enter your email"
          />
        </div>

        <div className="mb-3">
          <label className="form-label">Address</label>
          <textarea
            name="address"
            className="form-control"
            value={formData.address}
            onChange={handleChange}
            placeholder="House / Street / Area"
          ></textarea>
        </div>

        <div className="mb-3">
          <label className="form-label">City</label>
          <input
            type="text"
            name="city"
            className="form-control"
            value={formData.city}
            onChange={handleChange}
            placeholder="Enter city"
          />
        </div>

        <div className="mb-3">
          <label className="form-label">Pincode</label>
          <input
            type="text"
            name="pincode"
            className="form-control"
            value={formData.pincode}
            onChange={handleChange}
            placeholder="Enter pincode"
          />
        </div>
      </form>
    </div>

    {/* ✅ Order Summary */}
    <div className="col-md-6">
      <h4 className="mb-3">Order Summary 🧾</h4>

      {displayCart.length === 0 ? (
        <p>No games selected.</p>
      ) : (
        <ul className="list-group mb-3">
          {displayCart.map((game) => (
            <li
              key={game.id}
              className="list-group-item d-flex justify-content-between align-items-center"
```

```
            >
              <div>
                <strong>{game.name}</strong>
                <p className="text-muted small mb-0">1 Game</p>
              </div>
              <span>
                {game.price === 0 ? "Free" : `₹${game.price}`}
              </span>
            </li>
          ))}

          <li className="list-group-item d-flex justify-content-between">
            <strong>Total</strong>
            <strong>{totalAmount === 0 ? "Free" : `₹${totalAmount}`}</strong>
          </li>
        </ul>
      )}

      <button
        className="btn btn-primary w-100 fw-bold"
        onClick={handlePlaceOrder}
      >
        ✅ Place Order & Play
      </button>
    </div>
  </div>
 );
}
import React from "react";
import { motion } from "framer-motion";
import { Mail, Phone, MessageSquare, MapPin, Send } from "lucide-react";

export default function Contact() {
 return (
  <div className="contact-container">

   {/* Hero Title */}
   <motion.div
    className="contact-header"
    initial={{ opacity: 0, y: -15 }}
    animate={{ opacity: 1, y: 0 }}
   >
    <h1>
     Contact <span>Game Hub</span> 🎮
    </h1>
    <p>We’re here to help you connect, play & grow!</p>
   </motion.div>

   <div className="contact-content">

    {/* Left Box - Info */}
    <motion.div
     className="contact-info"
     initial={{ opacity: 0, x: -20 }}
     animate={{ opacity: 1, x: 0 }}
    >
     <h3>Reach Out</h3>
     <p>Let’s talk gaming, ideas or support!</p>

     <div className="info-line">
      <Phone /> <a href="tel:+91XXXXXXXXXX">+91 9309208325</a>
     </div>
     <div className="info-line">
      <Mail /> <a href="mailto:yourmail@example.com">prathmeshwaghmare289@gmail.com</a>
     </div>
```

```
        <div className="info-line">
          <MapPin /> Pune, Maharashtra, India
        </div>

        <motion.a
          href="tel:+91XXXXXXXXXX"
          className="call-btn"
          whileHover={{ scale: 1.05 }}
        >
          <MessageSquare /> Let's Connect
        </motion.a>
      </motion.div>

      {/* Right Box - Form */}
      <motion.form
        className="contact-form"
        initial={{ opacity: 0, x: 20 }}
        animate={{ opacity: 1, x: 0 }}
      >
        <h3>Drop a Message</h3>

        <input type="text" placeholder="Your
Name" required />
        <input type="email" placeholder="Your
Email" required />
        <textarea placeholder="Write your
message..." rows={4} required />

        <motion.button
          className="send-btn"
          whileHover={{ scale: 1.05 }}
        >
          Send  <Send size={18}/>
        </motion.button>
      </motion.form>
    </div>

  </div>
 );
}
import React, { useState } from "react";
import { motion } from "framer-motion";
import { Search } from "lucide-react";
import { Link, useNavigate } from "react-router-
dom"; // ✅ added navigate

const gamesList = [
  { id: 1, name: "Dino Run", level: "Beginner",
price: 0, description: "Run & dodge obstacles!",
img:
"https://tse4.mm.bing.net/th/id/OIP.vpWkiSO-
Ljn0K0nuilCAFAHaEK?pid=Api&P=0&h=180"
, link: "https://t-rex-runner.com/" },
  { id: 2, name: "Flappy Bird", level:
"Intermediate", price: 49, description: "Tap &
fly through pipes!", img: "https://flappy-
bird.com/wp-content/uploads/2024/02/flappy-
bird-start-screen.jpg", link:
"https://flappybird.io/" },
  { id: 3, name: "2048 Puzzle", level: "Advanced",
price: 59, description: "Join numbers & reach
2048!", img:
"https://tse2.mm.bing.net/th/id/OIP.lo74lDyhfY6
cYA88QiKTfwHaHa?pid=Api&P=0&h=180",
link: "https://play2048.co/" },
  { id: 4, name: "Tic Tac Toe", level: "Beginner",
price: 0, description: "3×3 classic challenge!",
img:
"https://img.gamedistribution.com/abebecfa89b6
46448c834963627b325d-512x512.jpeg", link:
"https://playtictactoe.org/" },
  { id: 5, name: "Chess", level: "Advanced",
price: 129, description: "Master the board
strategy!", img:
"https://static.vecteezy.com/system/resources/pre
views/018/871/720/original/king-and-soldier-
chess-pieces-on-transparent-background-
leadership-concept-png.png", link:
"https://www.chess.com/play/online" },
  { id: 6, name: "Number Puzzle", level:
"Intermediate", price: 39, description: "Train
logic & memory!", img:
"https://tse4.mm.bing.net/th/id/OIP.34XhO9kOh
DaGy3ah4n5K0wHaHa?pid=Api&P=0&h=180",
link: "https://sudoku.com/" },
```

```
  { id: 7, name: "Brain Games", level: "Intermediate", price: 69, description: "Boost thinking power!", img: "https://www.besttechie.com/content/images/wordpress/2020/05/brain-games.jpeg", link: "https://memory.puzzle.games/" },
  { id: 8, name: "Cricket Game", level: "Beginner", price: 0, description: "Hit & score like a pro!", img: "https://tse4.mm.bing.net/th/id/OIP.e2P3ReDBMC5G87UnoHrt3wHaEM?pid=Api&P=0&h=180", link: "https://gamepix.com/t/cricket" },
];

export default function Games() {
  const [search, setSearch] = useState("");
  const navigate = useNavigate(); // ✅

  const filteredGames = gamesList.filter(game =>
    game.name.toLowerCase().includes(search.toLowerCase())
  );

  const handleAddToMyGames = (game) => {
    const isLoggedIn = localStorage.getItem("isLoggedIn");

    // ✅ For paid games, block if not logged in
    if (game.price > 0 && !isLoggedIn) {
      alert("⚠️ Login required to add paid games!");

      // ✅ Save attempted item
      localStorage.setItem("pendingCheckout", JSON.stringify([game]));

      navigate("/login");
      return;
    }

    // ✅ Your original logic (untouched)
    let cart = JSON.parse(localStorage.getItem("cart")) || [];
    const exists = cart.find(item => item.id === game.id);

    if (exists) {
      exists.qty += 1;
    } else {
      cart.push({ ...game, qty: 1 });
    }

    localStorage.setItem("cart", JSON.stringify(cart));
    alert(`${game.name} added to My Games ✅`);
  };

  return (
    <div className="games-page container">
      <h1 className="games-title text-center">🎮 Explore All Games</h1>
      <p className="games-sub text-center">Find your next favorite game!</p>

      {/* Search */}
      <div className="games-search-box">
        <Search size={18} />
        <input
          type="text"
          placeholder="Search games..."
          value={search}
          onChange={(e)=>setSearch(e.target.value)}
        />
      </div>

      <div className="games-grid">
        {filteredGames.map((g, idx) => (
          <motion.div key={idx} className="game-
```

```
card" whileHover={{ scale: 1.06 }}>
          <img src={g.img} alt={g.name}
className="game-card-img" />

          <h4>{g.name}</h4>
          <span className="level-
tag">{g.level}</span>
          <p className="price-tag">{g.price ===
0 ? "Free" : `₹${g.price}`}</p>

          <p className="game-
desc">{g.description}</p>

          <div className="game-btn-group">
            <Link to={`/game/${g.id}`} state={g}
className="play-now-btn">
              Play ▶
            </Link>

            <button className="add-cart-btn"
onClick={() => handleAddToMyGames(g)}>
              + My Games
            </button>
          </div>
        </motion.div>
      ))}

      {filteredGames.length === 0 && <p>No
games found 😢</p>}
    </div>
  </div>
 );
}
import React, { useState } from "react";
import { Link } from "react-router-dom";
import { motion } from "framer-motion";
import { Search } from "lucide-react";

// ✅ Game list with category(level)
const games = [
 {
  id: 1,
  name: "Tic Tac Toe",
  level: "Beginner",
  price: 0,
  description: "Classic 3x3 strategy board
game.",
  img:
"https://img.gamedistribution.com/abebecfa89b6
46448c834963627b325d-512x512.jpeg",
  link: "https://playtictactoe.org/"
 },
 {
  id: 2,
  name: "Chess",
  level: "Advanced",
  price: 99,
  description: "Strategic board game to test
intelligence.",
  img:
"https://static.vecteezy.com/system/resources/pre
views/018/871/720/original/king-and-soldier-
chess-pieces-on-transparent-background-
leadership-concept-png.png",
  link: "https://www.chess.com/play"
 },
 {
  id: 3,
  name: "Number Puzzle",
  level: "Intermediate",
  price: 49,
  description: "Brain-boosting number puzzle
challenge.",
  img:
"https://tse4.mm.bing.net/th/id/OIP.34XhO9kOh
DaGy3ah4n5K0wHaHa?pid=Api&P=0&h=180",
  link:
"https://www.proprofsgames.com/puzzle/number
/"
 },
```

```
  {
    id: 4,
    name: "Brain Games",
    level: "Intermediate",
    price: 79,
    description: "Fun mind training & reasoning puzzles.",
    img: "https://www.besttechie.com/content/images/wordpress/2020/05/brain-games.jpeg",
    link: "https://brain-games.org/"
  },
  {
    id: 5,
    name: "Cricket Game",
    level: "Beginner",
    price: 0,
    description: "Virtual cricket match challenge.",
    img: "https://tse4.mm.bing.net/th/id/OIP.e2P3ReDBMC5G87UnoHrt3wHaEM?pid=Api&P=0&h=180",
    link: "https://www.cricketgames.me/"
  },
  {
    id: 6,
    name: "Car Racing",
    level: "Advanced",
    price: 149,
    description: "High-speed racing game.",
    img: "https://lh4.ggpht.com/DnegeWoA2nbLodwAI942HiEomkdHHWTG6pRBpV3CjnQkAE1LODW5-aJ93xWa8fDEk6c=h900",
    link: "https://www.carsimulator.games/"
  },
  {
    id: 7,
    name: "Memory Match",
    level: "Beginner",
    price: 39,
    description: "Enhance your memory with fun card matches.",
    img: "https://img.gamedistribution.com/854fd8048c814acaa86b0faffe364290-512x512.jpeg",
    link: "https://memory-game-online.freecodecamp.org/"
  },
  {
    id: 8,
    name: "Sudoku",
    level: "Advanced",
    price: 89,
    description: "Classic number placement logic puzzle.",
    img: "https://wallpaperaccess.com/full/9692889.png",
    link: "https://sudoku.com/"
  },
];



export default function Home() {
  const [search, setSearch] = useState("");
  const [filterLevel, setFilterLevel] = useState("All");

  // ✅ Add to My Games (Cart System)
const addToMyGames = (game) => {
  let cart = JSON.parse(localStorage.getItem("cart")) || [];

  const exists = cart.find(item => item.id === game.id);

  if (exists) {
    exists.qty += 1; // Increase quantity
  } else {
```

```
    cart.push({ ...game, qty: 1 });
  }

  localStorage.setItem("cart", JSON.stringify(cart));
  alert(`${game.name} added to My Games ✅`);
};



  // ✅ Filter + Search logic
  const filteredGames = games.filter(g =>
    g.name.toLowerCase().includes(search.toLowerCase()) &&
    (filterLevel === "All" || g.level === filterLevel)
  );

  return (
    <div className="home-container">
      {/* Hero Section */}
      <motion.section
        className="hero hero-orange"
        initial={{ opacity: 0, y: -20 }}
        animate={{ opacity: 1, y: 0 }}
      >
        <h1 className="hero-title">
          Welcome to <span className="hub-title">GameHub 🎮</span>
        </h1>
        <p className="hero-sub">Play. Learn. Compete. Level Up ⚡</p>

        {/* Search */}
        <div className="search-box">
          <Search size={20} />
          <input
            type="text"
            placeholder="Search games..."
            onChange={(e) => setSearch(e.target.value)}
          />
        </div>

        {/* Category Filter */}
        <select className="filter-select" onChange={(e) => setFilterLevel(e.target.value)}>
          <option value="All">All Levels</option>
          <option value="Beginner">Beginner</option>
          <option value="Intermediate">Intermediate</option>
          <option value="Advanced">Advanced</option>
        </select>
      </motion.section>

      {/* Game Display */}
      <section className="game-section container">
        <h2 className="section-heading text-center fw-bold mb-4">🚗 Game Garage</h2>

        <div className="masonry-grid">
          {filteredGames.map((g, i) => (
            <motion.article
              key={g.id}
              className={`mag-card mag-${i % 3}`}
              initial={{ opacity: 0, y: 12 }}
              animate={{ opacity: 1, y: 0 }}
              transition={{ duration: 0.3, delay: i * 0.04 }}
            >
              <img src={g.img} alt={g.name} />
              <div className="mag-body">
                <h4>{g.name}</h4>
                <p className="excerpt">{g.description}</p>
```

```
            <span className="level-
tag">{g.level}</span>

            <div className="mag-actions">
<Link
  to={`/game/${g.id}`}
  state={g}
  className="mag-play"
>
  Play ▶
</Link>

<span className="price-tag">
  {g.price === 0 ? "Free" : `₹${g.price}`}
</span>
              <button className="mag-add-btn"
onClick={() => addToMyGames(g)}>
                + My Games
              </button>
            </div>
          </div>
        </motion.article>
      ))}
    </div>
  </section>

  {/* Bottom Banner */}
  <section className="hub-banner">
    🎯 Level-Up Your Mind With Fun
Learning Games 🚀
  </section>
 </div>
 );
}
import loginGif from "../assets/gifs/login.gif";

import React, { useState } from "react";
import { Link } from "react-router-dom";
import { motion } from "framer-motion";
import { Gamepad2 } from "lucide-react";

export default function Login() {
 const [form, setForm] = useState({ email: "",
password: "" });

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

 const handleSubmit = (e) => {
 e.preventDefault();

 const storedUser =
JSON.parse(localStorage.getItem("gameHubUse
r"));

 if (!storedUser) {
  alert("⚠ No account found. Please register
first!");
  return;
 }

 if (storedUser.email === form.email &&
storedUser.password === form.password) {
  localStorage.setItem("gameHubLoggedIn",
"true");
  alert("✅ Login Successful!");
  window.location.href = "/profile";
 } else {
  alert("❌ Incorrect Email or Password!");
 }
};


 return (
  <div className="auth-page">
```

```
    {/* Left Content */}
    <motion.div
      className="auth-info"
      initial={{ opacity: 0, x: -40 }}
      animate={{ opacity: 1, x: 0 }}
    >
      <h1>Welcome Back 👋</h1>
      <p>Your gaming world is waiting. Log in & continue your journey!</p>

      <img src={loginGif} alt="Login Animation" className="auth-illustration" />
    </motion.div>

    {/* Login Form */}
    <motion.div
      className="auth-box shadow-lg"
      initial={{ opacity: 0, x: 40 }}
      animate={{ opacity: 1, x: 0 }}
    >
      <div className="auth-logo mb-3">
        <Gamepad2 size={35} className="me-2" />
        Game Hub
      </div>

      <h2 className="auth-title">Login</h2>
      <p className="auth-sub">Enter your details to continue</p>

      <form onSubmit={handleSubmit}>
        <input
          type="email"
          name="email"
          className="auth-input"
          placeholder="Email address"
          value={form.email}
          onChange={handleChange}
          required
        />

        <input
          type="password"
          name="password"
          className="auth-input"
          placeholder="Password"
          value={form.password}
          onChange={handleChange}
          required
        />

        <button type="submit" className="auth-btn">
          Login 🚀
        </button>
      </form>

      <p className="auth-footer">
        New here? <Link to="/register" className="link-text">Create Account</Link>
      </p>
    </motion.div>
  </div>
 );
}
import React, { useEffect, useState } from "react";
import { Gamepad2 } from "lucide-react";
import { motion } from "framer-motion";

export default function Profile() {
 const [user, setUser] = useState(null);

 useEffect(() => {
  const isLogged = localStorage.getItem("gameHubLoggedIn");
  const storedUser = JSON.parse(localStorage.getItem("gameHubUse
```

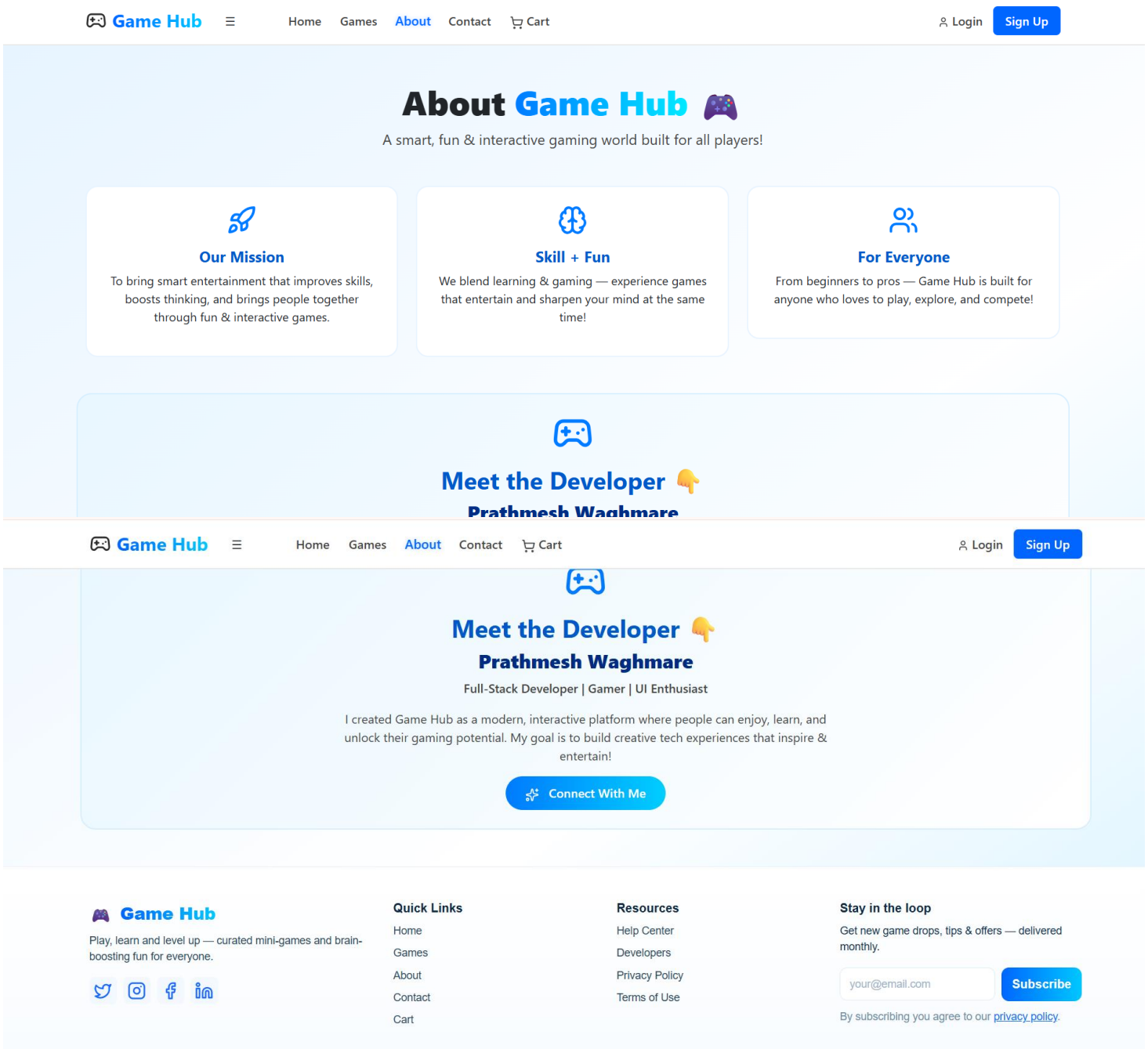
```
r"));

  if (!isLogged || !storedUser) {
    window.location.href = "/login";
    return;
  }

  setUser(storedUser);
 }, []);

 const logout = () => {

localStorage.removeItem("gameHubLoggedIn")
;
  alert("👋 Logged out successfully!");
  window.location.href = "/";
 };

 if (!user) return null;

 return (
  <div className="profile-page">
   <motion.div
    className="profile-card shadow-lg"
    initial={{ opacity: 0, scale: 0.8 }}
    animate={{ opacity: 1, scale: 1 }}
   >
    <div className="profile-header">
     <Gamepad2 size={40} />
     <h2>{user.username}'s Gamer Profile
🎮</h2>
    </div>

    <div className="profile-info">
     <p><b>Full Name:</b> {user.name}</p>
     <p><b>Email:</b> {user.email}</p>
     <p><b>Phone:</b> {user.phone}</p>
     <p><b>DOB:</b> {user.dob}</p>
     <p><b>Gender:</b> {user.gender}</p>
     <p><b>Country:</b> {user.country}</p>
    </div>

    <button className="auth-btn logout-btn"
onClick={logout}>
     Logout 🚪
    </button>
   </motion.div>
  </div>
 );
}
import registerGif from
"../assets/gifs/register.gif";
import React, { useState } from "react";
import { Link } from "react-router-dom";
import { motion } from "framer-motion";
import { Gamepad2 } from "lucide-react";

export default function Register() {
 const [form, setForm] = useState({
  name: "",
  username: "",
  email: "",
  phone: "",
  dob: "",
  gender: "",
  country: "",
  password: "",
  confirmPassword: ""
 });

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

 const handleSubmit = (e) => {
```

```
  e.preventDefault();

  if (form.password !== form.confirmPassword)
{
    alert("❌ Passwords do not match!");
    return;
  }

  // ✅ Save user details locally
  localStorage.setItem("gameHubUser",
JSON.stringify(form));
  alert("✅ Account Created Successfully!");

  // Redirect to login
  window.location.href = "/login";
};

  return (
    <div className="auth-page">
      <motion.div
        className="auth-info"
        initial={{ opacity: 0, x: -40 }}
        animate={{ opacity: 1, x: 0 }}
      >
        <h1>Create Your Gamer Profile
🎮🔥</h1>
        <p>Join <b>GameHub</b> and unlock
your personal game world!</p>
        <img src={registerGif} alt="Register
Animation" className="auth-illustration" />
      </motion.div>

      <motion.div
        className="auth-box shadow-lg"
        initial={{ opacity: 0, x: 40 }}
        animate={{ opacity: 1, x: 0 }}
      >
        <div className="auth-logo mb-3">
          <Gamepad2 size={35} className="me-2"
/>
          <b>GameHub</b>
        </div>

        <h2 className="auth-title">Sign Up</h2>
        <p className="auth-sub">Become part of
the next-gen gaming world 🕹️</p>

        <form onSubmit={handleSubmit}>
          <input
            type="text"
            name="name"
            className="auth-input"
            placeholder="Full Name"
            value={form.name}
            onChange={handleChange}
            required
          />

          <input
            type="text"
            name="username"
            className="auth-input"
            placeholder="Username"
            value={form.username}
            onChange={handleChange}
            required
          />

          <input
            type="email"
            name="email"
            className="auth-input"
            placeholder="Email Address"
            value={form.email}
            onChange={handleChange}
            required
          />
```

```
        <input
          type="tel"
          name="phone"
          className="auth-input"
          placeholder="Phone Number"
          value={form.phone}
          onChange={handleChange}
          required
        />

        <input
          type="date"
          name="dob"
          className="auth-input"
          value={form.dob}
          onChange={handleChange}
          required
        />

        <select
          name="gender"
          className="auth-input"
          value={form.gender}
          onChange={handleChange}
          required
        >
          <option value="">Select Gender</option>
          <option value="Male">Male</option>
          <option value="Female">Female</option>
          <option value="Other">Other</option>
        </select>

        <input
          type="text"
          name="country"
          className="auth-input"
          placeholder="Country"
          value={form.country}
          onChange={handleChange}
          required
        />

        <input
          type="password"
          name="password"
          className="auth-input"
          placeholder="Password"
          value={form.password}
          onChange={handleChange}
          required
        />

        <input
          type="password"
          name="confirmPassword"
          className="auth-input"
          placeholder="Confirm Password"
          value={form.confirmPassword}
          onChange={handleChange}
          required
        />

        <button type="submit" className="auth-btn">
          Create Account ✓
        </button>
      </form>

      <p className="auth-footer">
        Already have an account? <Link to="/login" className="link-text">Login Here</Link>
      </p>
```

```
  </motion.div>                                    );
</div>                                           }
```

Output :



Game Hub ☰ Home Games About Contact Cart Login Sign Up

Dino Run

Beginner

Free

Run & dodge obstacles!

Play + My Games

Flappy Bird

Intermediate

₹49

Tap & fly through pipes!

Play + My Games

2048 Puzzle

Advanced

₹59

Join numbers & reach 2048!

Play + My Games

Tic Tac Toe

Beginner

Free

3×3 classic challenge!

Play + My Games

Chess

Advanced

₹129

Master the board strategy!

Play + My Games

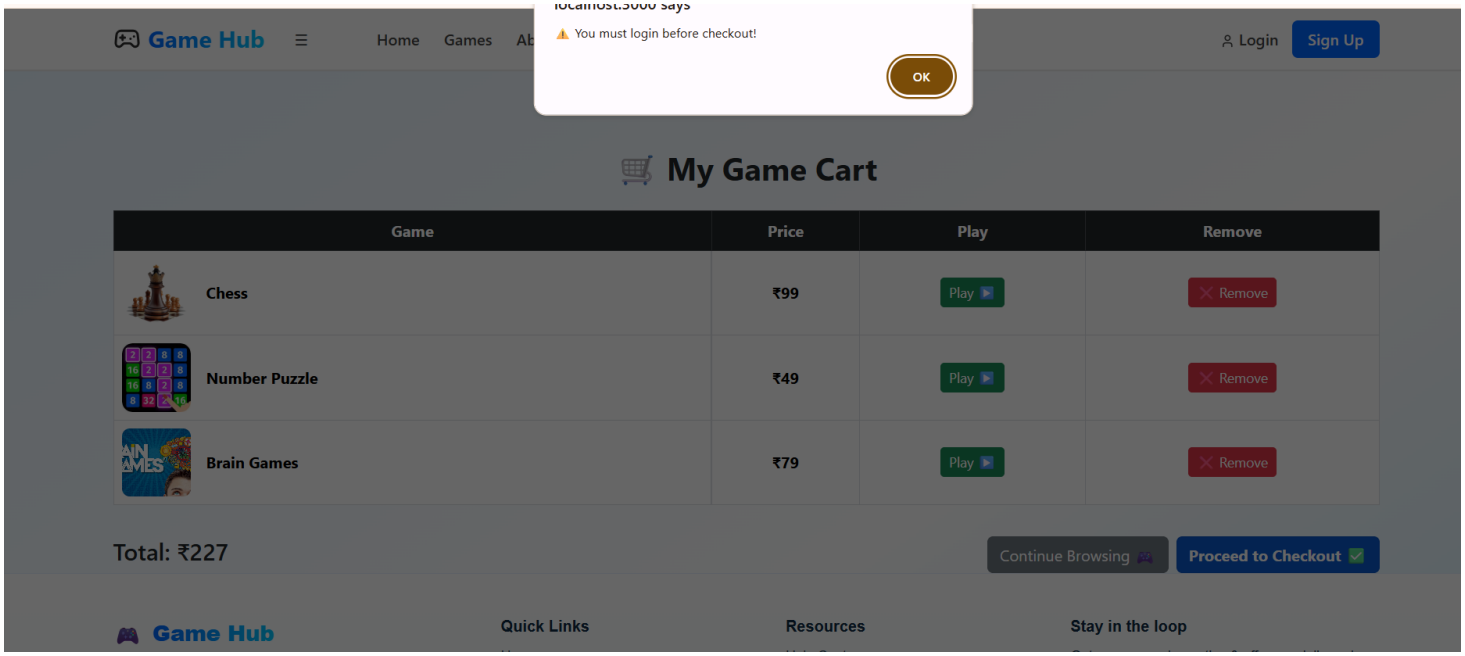
Game Hub ☰ Home Games About Contact Cart Login Sign Up

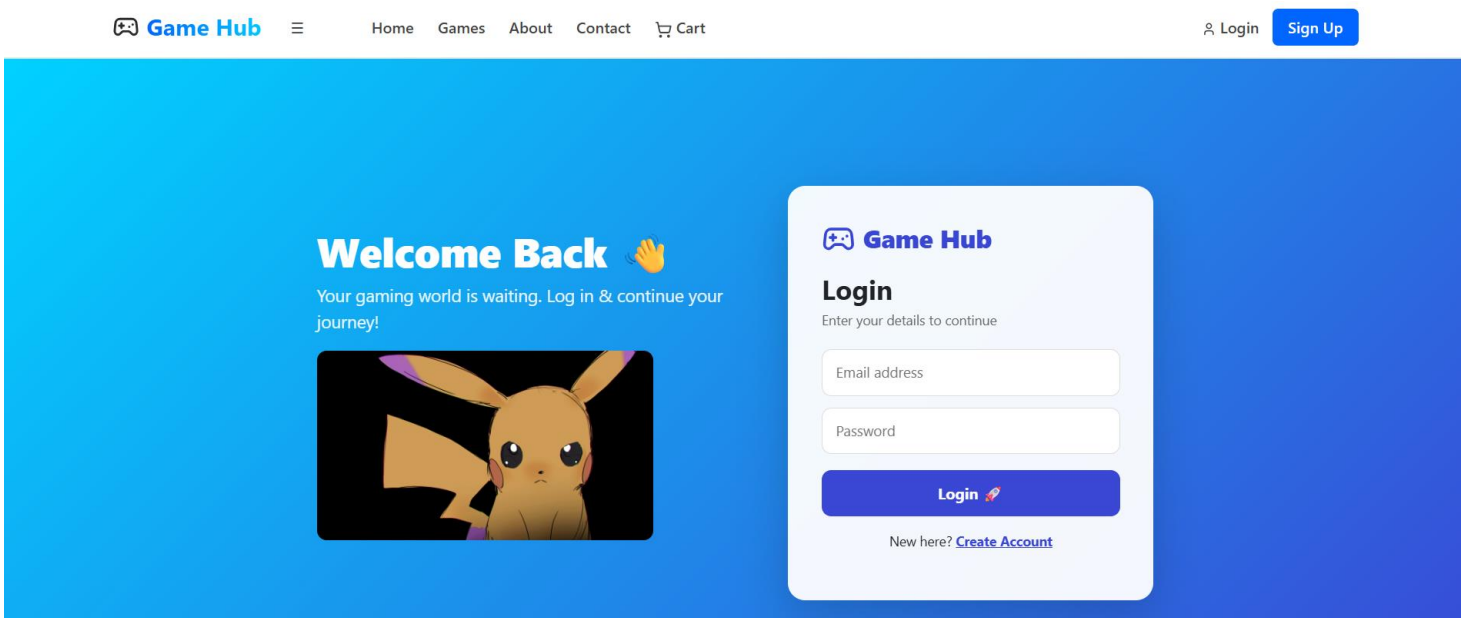
My Game Cart

Game	Price	Play	Remove
 Chess	₹99	Play	Remove
 Number Puzzle	₹49	Play	Remove
 Brain Games	₹79	Play	Remove

Total: ₹227

Continue Browsing Proceed to Checkout

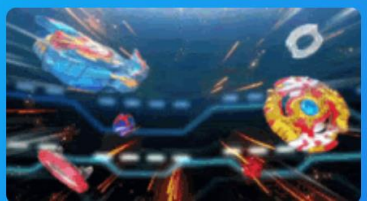

Game Hub
Home
Games
Login
Sign Up
You must login before checkout!
OK
My Game Cart
Game
Price
Play
Remove
Chess
₹99
Play
Remove
Number Puzzle
₹49
Play
Remove
Brain Games
₹79
Play
Remove
Total: ₹227
Continue Browsing
Proceed to Checkout
Game Hub
Quick Links
Resources
Stay in the loop


Game Hub
Home
Games
About
Contact
Cart
Login
Sign Up
Welcome Back
Your gaming world is waiting. Log in & continue your journey!
Game Hub
Login
Enter your details to continue
Email address
Password
Login
New here? Create Account

Game Hub ☰ Home Games About Contact Cart Login Sign Up

Create Your Gamer Profile

Join **GameHub** and unlock your personal game world!



GameHub

Sign Up

Become part of the next-gen gaming world

Full Name

Username

Email Address

Phone Number

dd-mm-yyyy

Select Gender

Country

Game Hub ☰ Home Games About Contact Cart Login Sign Up

parth@fire's Gamer Profile

Full Name: Parth

Email: waghmareprathmesh899@gmail.com

Phone: 9309208325

DOB: 2025-11-05

Gender: Male

Country: India

Logout

Game Hub

Play, learn and level up — curated mini-games and brain-boosting fun for everyone.

Quick Links

Home
Games
About
Contact
Cart

Resources

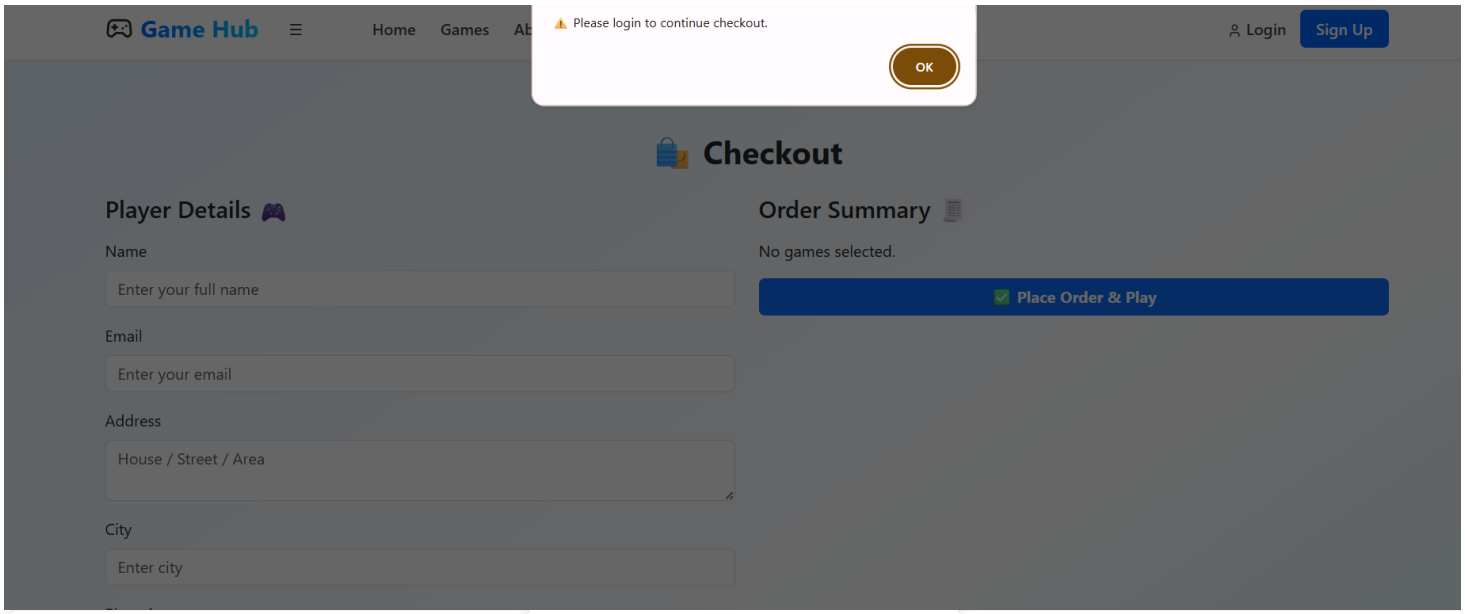
Help Center
Developers
Privacy Policy
Terms of Use

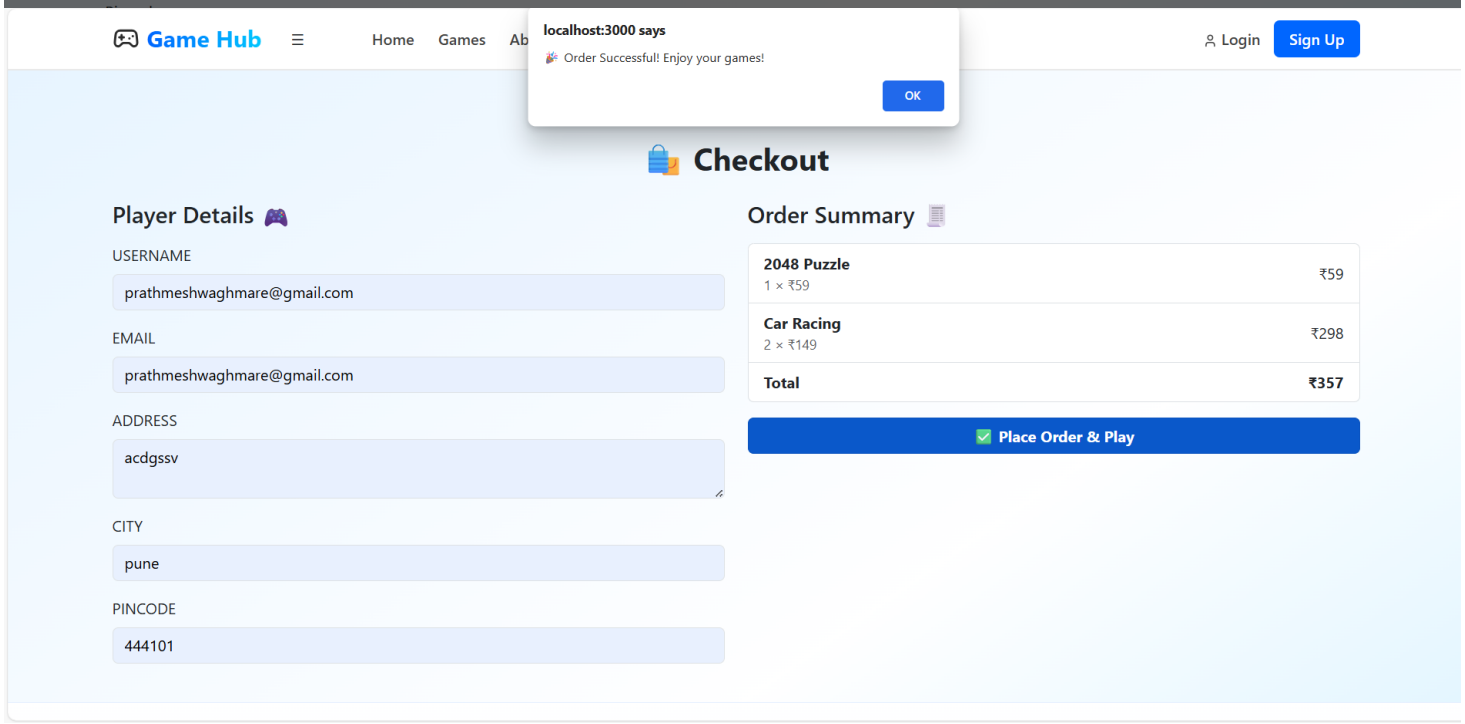
Stay in the loop

Get new game drops, tips & offers — delivered monthly.

your@email.com Subscribe

By subscribing you agree to our privacy policy.


Game Hub
Home
Games
Login
Sign Up
Please login to continue checkout.
OK
Checkout
Player Details
Name
Enter your full name
Email
Enter your email
Address
House / Street / Area
City
Enter city
Order Summary
No games selected.
Place Order & Play


Game Hub
Home
Games
Login
Sign Up
localhost:3000 says
Order Successful! Enjoy your games!
OK
Checkout
Player Details
USERNAME
prathmeshwaghmare@gmail.com
EMAIL
prathmeshwaghmare@gmail.com
ADDRESS
acdgssv
CITY
pune
PINCODE
444101
Order Summary
2048 Puzzle
1 × ₹59
₹59
Car Racing
2 × ₹149
₹298
Total
₹357
Place Order & Play

Conclusion :

We successfully implemented a full react application.