ONLINE FURNITURE SHOP

A Project Report for Industrial Training and Internship

submitted by

- **MILTON ALAM SK**
- **TANISHA SAHA**
- **KAIF AJMI**
- **PRITAM KAR**
- **VASKAR GHOSH**
- **KOYEL DEBNATH**

In the partial fulfillment of the award of the degree of

BCA

In the

Department of Computer Science

Of

Gitaram Institute of Management

GITARAM
INSTITUTE OF MANAGEMENT

At

Ardent Computech Pvt. Ltd.

ARDENT
COMPUTECH PVT. LTD.
High-End Technology Training and Project

ARDENT®
GROUP OF COMPANIES
INTERNSHIP | DEVELOPMENT | PROJECT
An ISO 9001 : 2015 Certified Company

Ardent Computech Pvt. Ltd.
Drives you to the Industry
Module 132, SDF Building, Sector V, Salt Lake, Pin - 700091
www.ardentcollaborations.com

CERTIFICATEFROMSUPERVISOR

This is to certify that ***"Milton Alam Sk , 231341010021" ,"Tanisha Saha ,231341010057", "Kaif Ajmi, 231341010016", "Pritam Kar, 231341010026", "Vaskar Ghosh, 231341010059", "Koyel Debnath, 231341010019"****have completed the project titled "<****ONLINE FURNITURE SHOP****>" under my supervision during the period from "<01-07-2025>" to "<31-07-2025>" which is in partial fulfilment of requirements for the award of the* ***BCA*** *degree and submitted to the Department of "****<Computer Science>****" of "****<Gitaram Institute of Management>"****.*

_____

SignatureoftheSupervisor

Date:

NameoftheProjectSupervisor:

BONAFIDE CERTIFICATE

Certified that this project work was carried out under my supervision

"Online Furniture Shop" is the Bonafideworkof

Name of the student: MILTON ALAM SK ***Signature:***

Name of the student: TANISHA SAHA ***Signature:***

Name of the student: KAIF AJMI ***Signature:***

Name of the student: PRITAM KAR ***Signature:***

Name of the student: VASKAR GHOSH ***Signature:***

Name of the student: KOYEL DEBNATH ***Signature:***

SIGNATURE

Name:

PROJECTMENTOR

SIGNATURE

Name:

EXAMINERS

Ardent Original Seal

ACKNOWLEDGEMENT

The achievement that is associated with the successful completion of any task would be incomplete without mentioning the names of those people whose endless cooperation made it possible. Their constant guidance and encouragement made all our efforts successful.

We take this opportunity to express our deep gratitude towards our project mentor, ***Subhojit Santra*** for giving such valuable suggestions, guidance and encouragement during the development of this project work.

Last but not the least we are grateful to all the faculty members of **Ardent Computech Pvt. Ltd.** for their support.

ContentPage

COMPANY PROFILE

ARDENT (Ardent Computech Pvt. Ltd.), formerly known as Ardent Computech Private Limited, isanISO9001:2015certifiedSoftwareDevelopment andTrainingCompanybasedin India. Operating independentlysince 2003,the organization has recentlyundergone a strategic merger with ARDENT Technologies, enhancing its global outreach and service offerings.

ARDENT Technologies

ARDENTTechnologiesdelivershigh-endITservicesacrosstheUK,USA,Canada,andIndia. Itscorecompetencieslieinthedevelopment ofcustomizedapplicationsoftware,encompassing end-to-end solutions including system analysis, design, development, implementation, and training. The company also provides expert consultancy and electronic security solutions. Its clientele spans educational institutions, entertainment companies, resorts, theme parks, the service industry, telecom operators, media, and diverse business sectors.

ARDENT Collaborations

ARDENT Collaborations, the Research, Training, and Development division of ARDENT (Ardent Computech Pvt. Ltd.), offers professional IT-enabled services and industrial training programs.These are tailored for freshers and professionals from B.Tech,M.Tech,MBA,MCA, BCA, and MSc backgrounds. ARDENT (Ardent Computech Pvt. Ltd.) provides Summer Training, Winter Training, and Industrial Training to eligible candidates. High-performing students may qualify for stipends, scholarships, and additional benefits based on performance and mentor recommendations.

Associations and Accreditations

ARDENT (Ardent Computech Pvt. Ltd.**)** is affiliated with the National Council of Vocational Training (NCVT) under the Directorate General of Employment &Training (DGET), Ministry of Labour & Employment,Government of India.The institutionupholdsstrict quality standards under ISO 9001:2015 certification and is dedicated to bridging the gap between academic knowledge and industry skills through innovative training programs.

INTRODUCTION

The **Online Furniture Shop** is a web-based e-commerce platform designed to provide customers with a seamless experience for browsing, selecting, and purchasing furniture online. With the growing demand for digital shopping experiences, this project aims to replicate the core functionality of popular online furniture retailers in a simplified and user-friendly format.

Built using modern web technologies like **React.js**, **Bootstrap**, and **Node.js**, the system allows users to explore a catalog of furniture items, view product details, add items to a shopping cart, and proceed to checkout. Registered users can also manage their accounts and view past orders, ensuring a personalized shopping experience

OBJECTIVE

The main objective of the **Online Furniture Shop** project is to develop a fully functional, user-friendly e-commerce platform that enables customers to:

- **Browse** a wide range of furniture products categorized for easy access.
- **Search** and view detailed information about each product.
- **Register and login** securely using authentication mechanisms.
- **Add items to a cart** and manage quantities before purchasing.
- **Place orders** and receive confirmation of successful transactions.

This project also aims to:

- Implement a clean and responsive **frontend interface** using **React.js** and **Bootstrap**.
- Develop a scalable and secure **backend system** using **Node.js** and **Express.js**.
- Provide a structured database solution using **MongoDB** to store user, product, cart, and order data.
- Demonstrate practical application of **full-stack development** principles.

SCOPE

The **Online Furniture Shop** project is designed to simulate a real-world furniture e-commerce application with essential features for both customers and administrators. The scope of the project is divided into **functional** and **technical** areas.

Functional Scope

1. **User Management:**
    - New users can register and create an account.
    - Existing users can log in securely.
    - Users can update their profiles and view order history.
2. **Product Browsing:**
    - Users can browse furniture items across different categories (e.g., sofa, bed, table).
    - Each product includes a name, description, price, image, and availability.
3. **Search and Filtering:**
    - Products can be searched by keyword.
    - Optional: Filter by category, price range, or popularity.
4. **Shopping Cart:**
    - Add or remove items from the cart.
    - Update quantities.
    - View cart summary and total price.
5. **Checkout and Order:**
    - Checkout form for shipping details.
    - Place order and receive confirmation.
    - Orders saved to the database for tracking.
6. **(Optional) Admin Panel:**
    - Add, update, or delete furniture products.
    - View and manage customer orders.

Technical Scope

- **Frontend**: Developed using **React.js** with **Bootstrap** for responsive design.
- **Backend**: Developed using **Node.js** and **Express.js** for RESTful APIs.
- **Database**: Uses **MongoDB** for storing users, products, cart items, and orders.
- **Security**: Password hashing, token-based authentication (e.g., JWT).
- **Deployment**: Can be hosted on platforms like Vencel (frontend) and Render/Heroku (backend).

Limitations (Out of Scope for Now)

- No online payment gateway integration (e.g., Stripe, Razor pay).
- No real-time order tracking.
- No AI-based recommendations or analytics.
- No multi-vendor or seller dashboard features.

SYSTEM ANALYSIS

System analysis involves examining the **functional**, **technical**, and **user-based** requirements of the Online Furniture Shop project. It helps define how the system will operate, what it must achieve, and how different components interact.

1. Problem Definition

In the traditional furniture retail system:

- Customers have limited access to product choices and store timings.
- No real-time inventory tracking or price updates.
- Physical stores often result in time-consuming shopping experiences.

Objective:

To solve these issues by developing a **web-based furniture shopping platform** that allows users to conveniently browse and purchase products online anytime and from anywhere.

2. Feasibility Study

a. Technical Feasibility

- The application uses widely supported technologies: React.js, Node.js, Express, and MongoDB.
- Hosting platforms (e.g., Vercel, Render) support deployment of both frontend and backend components.
- Easily scalable architecture for handling more users and products in the future.

b. Operational Feasibility

- The system is designed for ease of use by customers with minimal technical knowledge.
- Admin features (if implemented) can be handled with basic training.
- Cart, checkout, and login processes are simple and intuitive.

3. System Requirements Analysis

User Requirements

- Users should be able to register/login.
- View furniture items with images and details.
- Add/remove items to/from the cart.
- Place orders and receive confirmation.

Admin Requirements *(Optional)*

- Add/update/delete furniture products.
- View customer orders and manage stock.

System Requirements

Frontend:

- Framework: React.js
- Styling: Bootstrap, CSS
- Routing: React Router

Backend:

- Server: Node.js + Express
- Database: MongoDB
- API: RESTful endpoints

Security:

- JWT-based authentication
- Password encryption with crypt

Tools:

- Postman (API testing), GitHub (version control), VS Code (IDE)

he system analysis?

Identification of Need

In today's digital age, consumers increasingly prefer the convenience of shopping from home rather than visiting physical stores. Traditional furniture retailing methods often suffer from various limitations, including:

- **Limited accessibility** – Customers must visit stores during business hours.
- **Geographical restrictions** – Users outside urban areas may have limited options.
- **Lack of real-time availability** – Inventory updates and product availability are not always accurate.
- **Time-consuming process** – Comparing products, prices, and designs can take hours in-store.

These challenges highlight the **need for an online furniture shopping platform** that simplifies and modernizes the buying experience.

🔍 Why the Online Furniture Shop is Needed:

1. **Customer Convenience:**
    - Allows users to browse and purchase furniture 24/7 from any location.
    - Provides a detailed catalog with images, prices, and specifications.
2. **Efficient Product Management:**
    - Admins can easily update products, prices, and availability through a web interface.
3. **Scalability & Reach:**
    - A web-based platform can serve a much larger audience than a single physical store.
4. **Cost-Effectiveness:**
    - Reduces operational costs related to physical space, staffing, and printed catalogs.
5. **Digital Transformation:**
    - Supports the shift of traditional furniture businesses to digital platforms, increasing competitiveness.

🎯 Goals Addressed by the Project

- Improve user experience in furniture selection and purchase.
- Provide a modern, secure, and scalable online shopping environment.
- Enable furniture sellers to manage inventory and customer orders easily.

Feasibility Study (Summary)

✅ Technical Feasibility

- Uses modern, open-source tools: React.js, Node.js, MongoDB.
- Easy to develop and deploy on free hosting platforms.

- Fully supported by current developer tools.

💼 Operational Feasibility

- Simple, user-friendly interface for customers and admins.
- Works on mobile and desktop (responsive design).
- Easy to manage products and orders.

💰 Economic Feasibility

- No software licensing costs.
- Development and hosting are low-cost.
- Suitable for small businesses or academic projects.

⏱️ Time Feasibility

- MVP can be developed in 2–4 weeks.
- Can be built and tested within academic or project deadlines.

Work flow

🖥️ 1. User Interaction Layer (Frontend - React + Bootstrap)

Steps:

1. **Homepage Loads**
    - React renders product listing UI.
    - GET /api/products API call is triggered.
2. **User Browses Products**
    - React fetches and displays product details using Axios/Fetch.
    - Bootstrap handles responsive UI styling.
3. **User Registers or Logs In**
    - React form posts data to POST /api/auth/register or POST /api/auth/login.
    - On successful login, token is stored (usually in localStorage).
4. **Add to Cart**
    - User clicks "Add to Cart" → React dispatches cart update in state.
    - Optionally synced with backend via POST /api/cart.
5. **Checkout**
    - React form captures address and order info.
    - Sends POST /api/orders with cart and user details.

⚙️ 2. Backend Processing (Node.js + Express)

Key Functions:

- **Routes Handle Requests**
    - e.g., /api/products, /api/auth/login, /api/orders
- **Controllers Process Logic**
    - Validate user inputs, authenticate tokens, CRUD operations.
- **Database Operations**
    - Uses Mongoose to read/write to MongoDB.
- **Responses Sent Back**
    - JSON data returned to frontend for UI updates.

🗄️ 3. Database Layer (MongoDB)

Collections:

- users → Stores user credentials and profile data
- products → Stores furniture items
- cart → Temporary cart data (optional)
- orders → Final placed orders

🔁 Workflow Cycle Example

[React Component]

↓ Axios Fetch
[Node.js API (Express)]
↓ Mongoose Query
[MongoDB Database]
↑ Result (JSON)
[Node.js Sends Response]
↑ React Displays Data
[Bootstrap Renders UI]

📋 **Example: "Add to Cart" Workflow**

1. User clicks **Add to Cart** in React component
2. React sends POST /api/cart with product and user ID
3. Node.js controller receives and stores in MongoDB
4. Backend sends response → React updates cart state
5. Bootstrap updates UI to show cart icon with item count

Advantages

1. **24/7 Access – Shop anytime from anywhere.**
2. **Responsive UI – Mobile-friendly with Bootstrap.**
3. **Fast Performance – React ensures smooth user experience.**
4. **Secure Login – JWT & encrypted passwords.**
5. **Real-Time Updates – Instant product & cart changes.**
6. **Easy Management – Admin can handle products/orders.**
7. **Low Cost – Built with free, open-source tools.**
8. **Scalable – Easily expandable for future growth.**
9. **Full-Stack Learning – Great practice for MERN stack developers.**

❌ Disadvantages

1. **No Payment Integration** – Lacks online payment options.
2. **No Real-Time Chat** – No customer support or live help.
3. **Basic Admin Panel** – Limited features for inventory control.
4. **No Delivery Tracking** – Users can't track order status.
5. **Security Risks** – Needs strong protection against cyberattacks.
6. **Limited SEO** – React SPAs may need extra setup for search engines.
7. **Internet Required** – Can't be used offline.

🕮 Study of the System

1. **System Type:**
   Web-based e-commerce platform for selling furniture online.
2. **Users:**
   - **Customers** – Browse, register, shop, and order.
   - **Admin** – Manage products and view orders (optional).
3. **Frontend (React + Bootstrap):**
   - Displays products, forms, cart, and user interface.
   - Communicates with backend via REST APIs.
4. **Backend (Node.js + Express):**
   - Handles user authentication, product data, orders, and cart logic.
   - Connects to the database and manages routes.
5. **Database (MongoDB):**
   - Stores users, products, cart, and order information.
6. **Working Flow:**
   - User → React UI → API → Node.js → MongoDB → response to user.

⌨Input and Output (In Short)

▼Input:

- **User Registration Data** – Name, email, password
- **Login Credentials** – Email, password
- **Product Details** – Name, price, description (admin)
- **Cart Actions** – Product selection, quantity
- **Order Info** – Shipping address, cart items

▲Output:

- **Success/Error Messages** – Login, registration, checkout
- **Product Listings** – Names, prices, images
- **Cart Summary** – Items, total price
- **Order Confirmation** – Order placed message, details
- **Admin Notifications** – Product added/updated/deleted

Software Requirement Specifications

1. Frontend Requirements

- **Language/Framework:** React.js
- **Styling:** Bootstrap, CSS
- **Browser Support:** Chrome, Firefox, Edge
- **Functionality:** Product display, user forms, cart, responsive UI

2. Backend Requirements

- **Language/Framework:** Node.js with Express.js
- **APIs:** RESTful services for products, auth, cart, and orders
- **Security:** JWT for authentication, bcrypt for password hashing

3. Database Requirements

- **Database:** MongoDB
- **Collections:** Users, Products, Cart, Orders

4. Tools & Platforms

- **IDE:** VS Code
- **Version Control:** Git + GitHub
- **API Testing:** Postman
- **Deployment:**Vencel (Frontend), Render/Heroku (Backend)

Software Engineering Paradigm Applied

📌 Paradigm Used:

Incremental Model (with elements of the **Waterfall Model**)

☑ Reason for Use:

- Project was developed **in phases**:
    1. User Interface
    2. Authentication
    3. Product Management
    4. Cart & Checkout
- Each module was **built, tested, and improved** before adding the next.
- Helps manage time and functionality for small to medium projects.

⚙ Key Benefits:

- Easier to test and debug in small parts
- Allows early partial deployment
- Reduces risk by integrating features step-by-step

System Design

1. Frontend Design (React + Bootstrap)

- **Pages:** Home, Product Details, Login/Register, Cart, Checkout
- **Components:** Navbar, Product Card, Search Bar, Cart Icon
- **UI:** Responsive layout with Bootstrap classes

2. Backend Design (Node.js + Express)

- **APIs:**
    - `/api/auth` – Register/Login
    - `/api/products` – Get/Add products
    - `/api/cart` – Manage user cart
    - `/api/orders` – Place and retrieve orders
- **Middleware:** Authentication, error handling

3. Database Design (MongoDB)

- **Collections:**
  - `users`: name, email, password
  - `products`: title, description, price, image
  - `cart`: userID, productID, quantity
  - `orders`: userID, items, address, status

HOW TO DRAW USE CASEDIAGRAM

What is a Use Case Diagram?

A **Use Case Diagram** shows **what the system does** from a **user's perspective**. It defines:

- **Actors** (users or other systems)
- **Use Cases** (features/actions)
- **Relationships** between actors and use cases

Steps to Draw a Use Case Diagram

1. Identify Actors

Actors are users or systems that interact with your application.

For example:

- **Customer**
- **Admin**

2. Identify Use Cases (Functions)

What actions can each actor perform?

For Customer:

- Register/Login
- Browse Products
- View Product Details
- Add to Cart
- Place Order
- View Order History

For Admin:

- Login
- Add/Update/Delete Products
- View Orders

Symbols to Use:

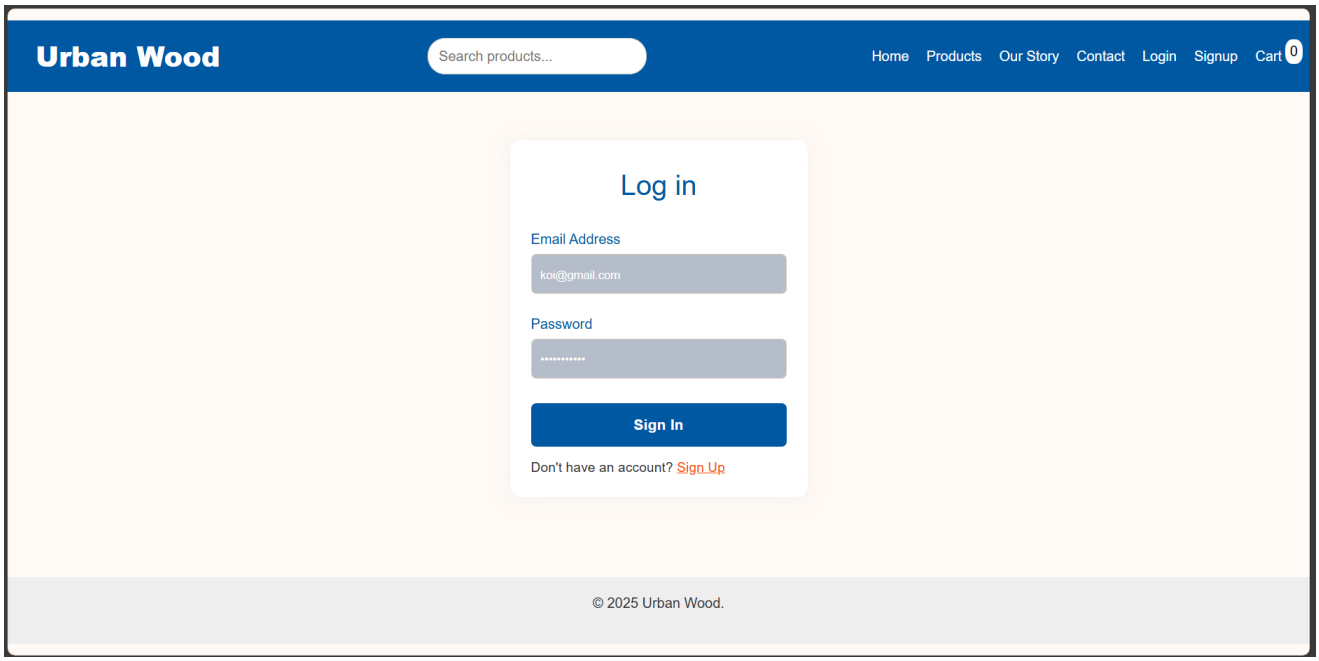
Symbol	Meaning
Stick figure	Actor (e.g., Customer)
Oval	Use Case (action)
— Line	Relationship
↷ Include	One use case always includes another
Extend	Optional use case depending on condition

Example (Text-Based View)

```
Actor: Customer
↳ Login/Register
↳ Browse Products
↳ View Product Details
↳ Add to Cart
↳ Place Order
↳ View Order History

Actor: Admin
↳ Login
↳ Manage Products
↳ View Orders
```

UI SNAPSHOTS

LOGINPAGE:

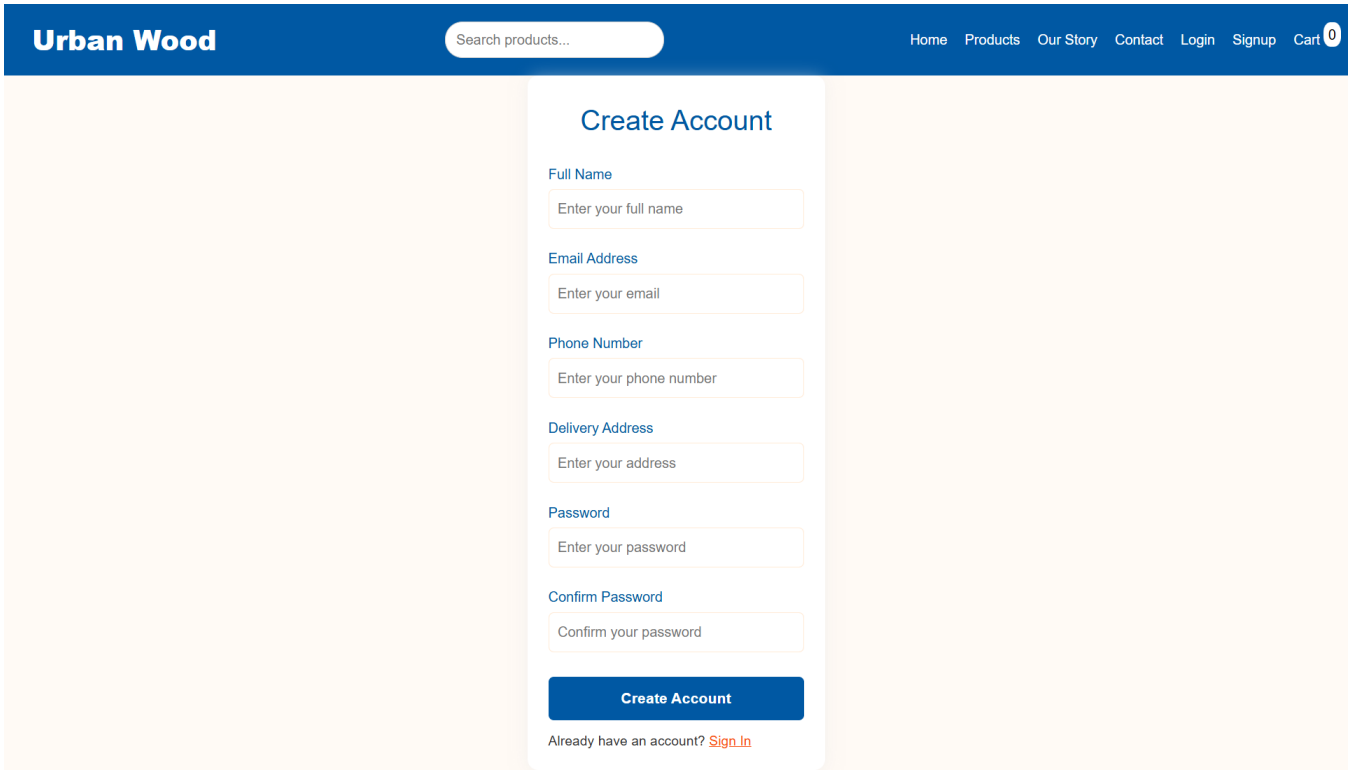


Code:

```jsx
import Footer from "../component/Footer/Footer";

const Login = () => {
    return (
        <>
            <div className="login-signup-wrapper">
                <div className="login-signup-card">
                    <h2>Log in</h2>
                    <form>
                        <label htmlFor="email">Email Address</label>
                        <input type="email" id="email" required placeholder="Enter your email" />

                        <label htmlFor="password">Password</label>
                        <input type="password" id="password" required placeholder="Enter your password" />

                        <button type="submit">Sign In</button>

                        <div className="form-alt">
                            Don't have an account? <a href="/signup">Sign Up</a>
                        </div>
                    </form>
                </div>
            </div>
            <Footer />
        </>
    );
};
```

```
export default Login;
```

SIGN UP:

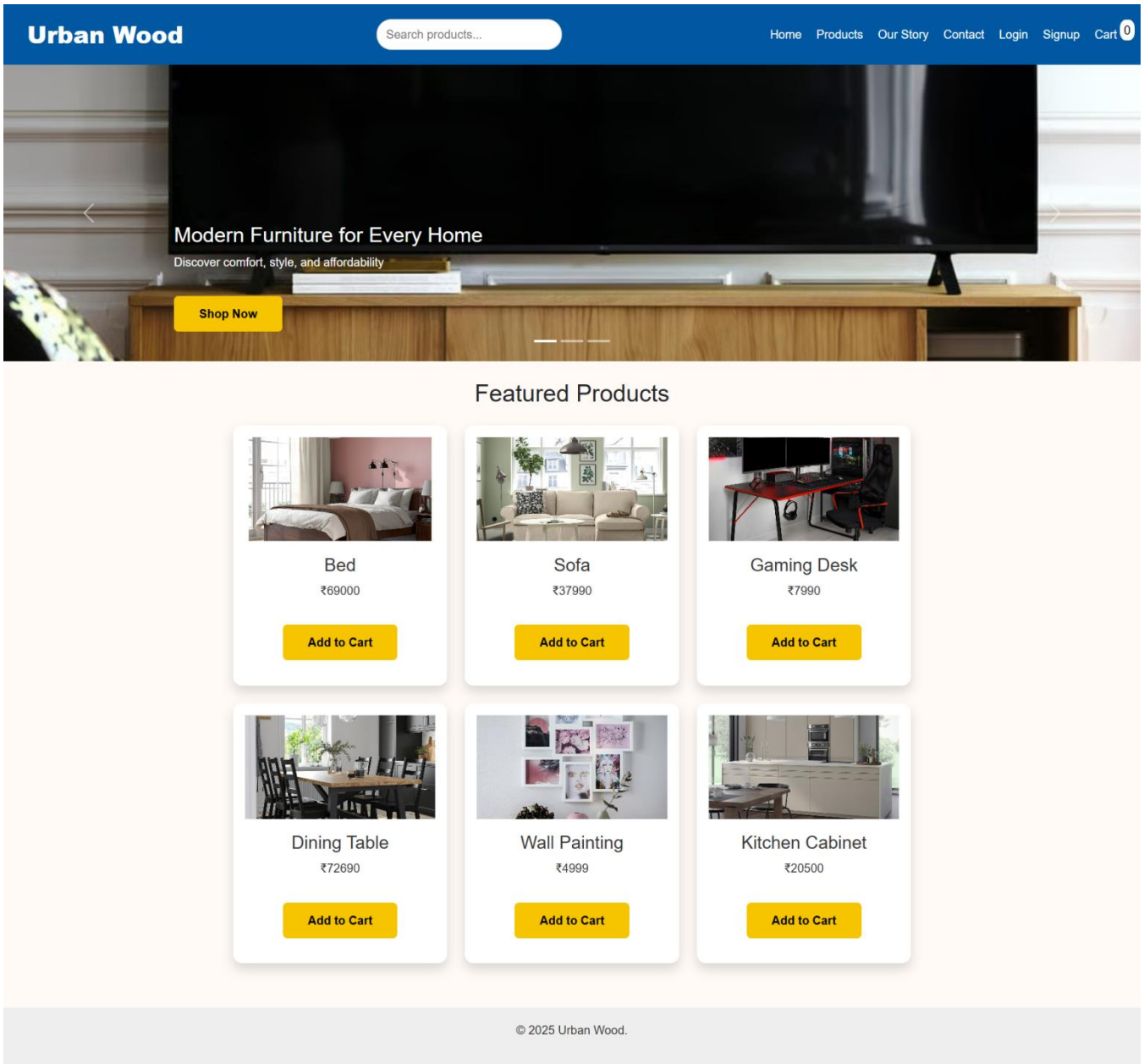


Code:

```
const Signup = () => {
    return (
        <>
            <div className="login-signup-wrapper">
                <div className="login-signup-card">
                    <h2>Create Account</h2>
                    <form>
                        <label htmlFor="fullname">Full Name</label>
                        <input type="text" id="fullname" required placeholder="Enter your full name" />

                        <label htmlFor="email">Email Address</label>
                        <input type="email" id="email" required placeholder="Enter your email" />

                        <label htmlFor="phone">Phone Number</label>
                        <input type="tel" id="phone" required placeholder="Enter your phone number" />

                        <label htmlFor="address">Delivery Address</label>
                        <input type="text" id="address" required placeholder="Enter your address" />

                        <label htmlFor="password">Password</label>
                        <input type="password" id="password" required placeholder="Enter your password" />

                        <label htmlFor="confirmpassword">Confirm Password</label>
                        <input type="password" id="confirmpassword" required placeholder="Confirm your password" />

                        <button type="submit">Create Account</button>

                        <div className="form-alt">
                            Already have an account? <a href="/login">Sign In</a>
                        </div>
                    </form>
```

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

export default Signup;
```

HOMEPAGE:

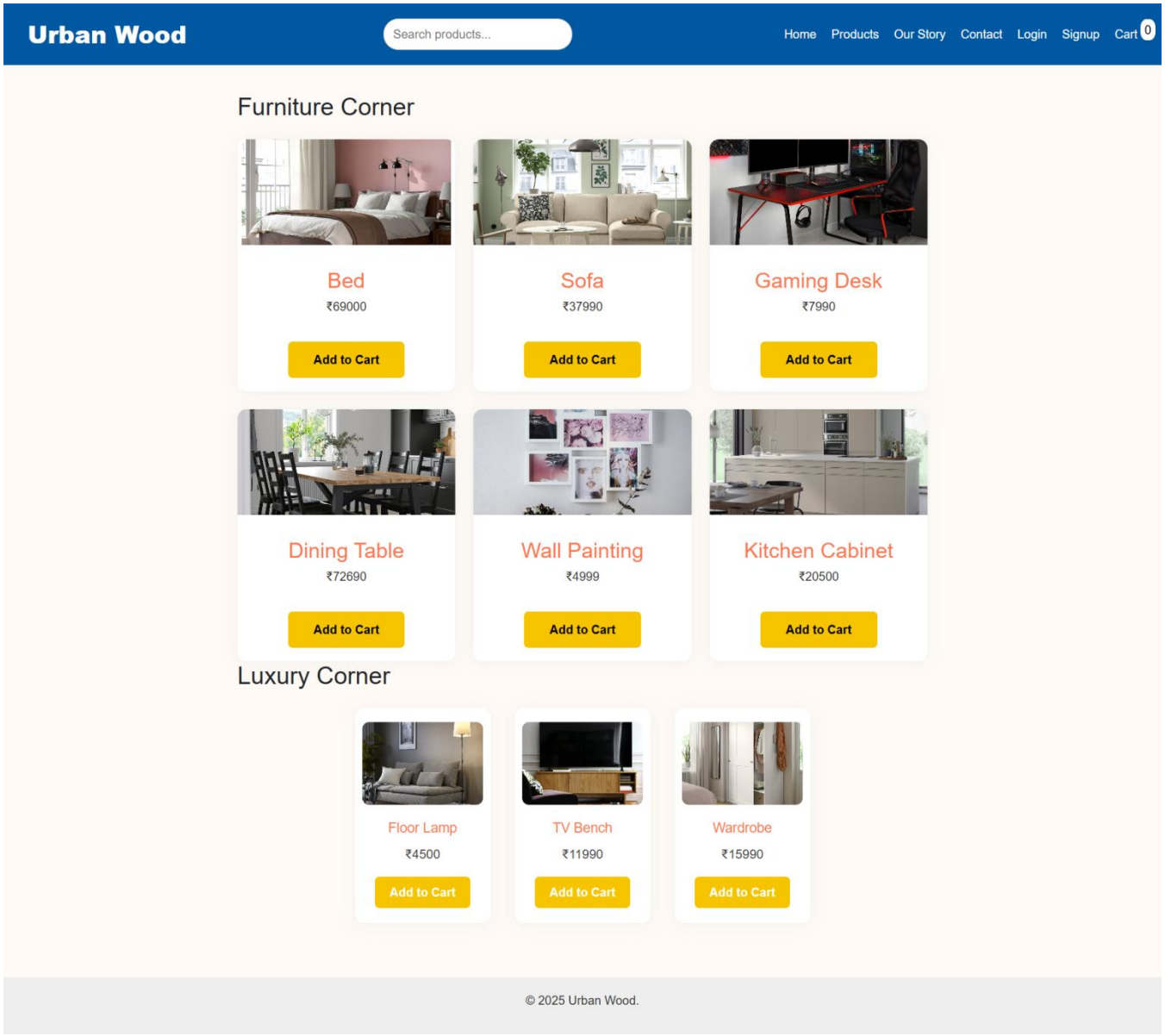


Code:

```
import { foodCard } from "../component/data";
import Footer from "../component/Footer/Footer";
import HeroCarousel from "./HeroCarousel";
import 'bootstrap/dist/css/bootstrap.min.css';
import 'bootstrap/dist/js/bootstrap.bundle.min.js';
import '../App.css';

const Home = ({ cart, setCart }) => {
  const featured = foodCard.filter(item => item.type === "furniture" || item.type === "home");

  const addToCart = id => {
```

```
    if (!cart.includes(id)) setCart([...cart, id]);
  };

  return (
    <>
      <main>
        <section className="container-fluid px-0"><HeroCarousel /></section>
        <section className="specials py-4">
          <h2 className="text-center">Featured Products</h2>
          <div className="card-container d-flex justify-content-center gap-4 flex-wrap mt-3">
            {featured.map(item => (
              <div className="dish-card text-center shadow p-3" key={item.id}>
                <img src={item.url} alt={item.title} className="img-fluid" />
                <div className="card-body">
                  <h4>{item.title}</h4>
                  <p>₹{item.price}</p>
                  <button
                    className="btn btn-yellow-black"
                    onClick={() => addToCart(item.id)}
                  >
                    {cart.includes(item.id) ? "Added" : "Add to Cart"}
                  </button>
                </div>
              </div>
            ))}
          </div>
        </section>
      </main>
      <Footer />
    </>
  );
};

export default Home;
```

PRODUCTS:

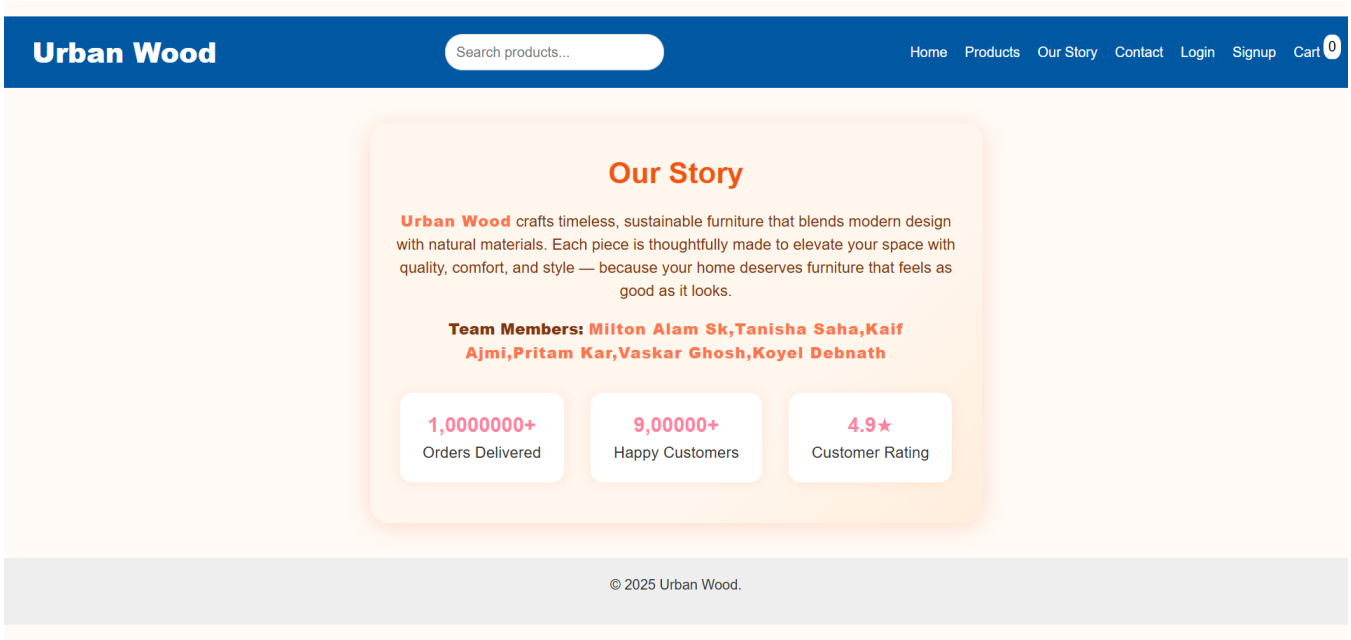


Code:

```
import { foodCard } from "../component/data";
import Footer from "../component/Footer/Footer";

const Menu = ({ searchTerm, cart, setCart }) => {
  const showFurniture = foodCard.filter(ele => ele.type === "furniture");
  const showDesserts = foodCard.filter(ele => ele.type === "dessert");

  const handleAdd = id => {
    if (!cart.includes(id)) setCart([...cart, id]);
  };

  const filteredFurniture = showFurniture.filter(item =>
    item.title.toLowerCase().includes(searchTerm.toLowerCase())
  );
```

```
  const filteredDesserts = showDesserts.filter(item =>
    item.title.toLowerCase().includes(searchTerm.toLowerCase())
  );

  return (
    <>
      <section className="specials">
        <h2>Furniture Corner</h2>
        <div className="card-container">
          {filteredFurniture.map(item => (
            <div key={item.id} className="dish-card">
              <img src={item.url} alt={item.title} />
              <div className="card-body">
                <h3>{item.title}</h3>
                <p>₹{item.price}</p>
                <button
                  className="btn btn-yellow-black"
                  onClick={() => handleAdd(item.id)}
                >
                  {cart.includes(item.id) ? "Added" : "Add to Cart"}
                </button>
              </div>
            </div>
          ))}
        </div>
        <h2>Luxury Corner</h2>
        <div className="dessert-row">
          {filteredDesserts.map(item => (
            <div key={item.id} className="dessert-card">
              <img src={item.url} alt={item.title} />
              <h3>{item.title}</h3>
              <p>₹{item.price}</p>
              <button onClick={() => handleAdd(item.id)}>
                {cart.includes(item.id) ? "Added" : "Add to Cart"}
              </button>
            </div>
          ))}
        </div>
      </section>
      <Footer />
    </>
  );
};

export default Menu;
```

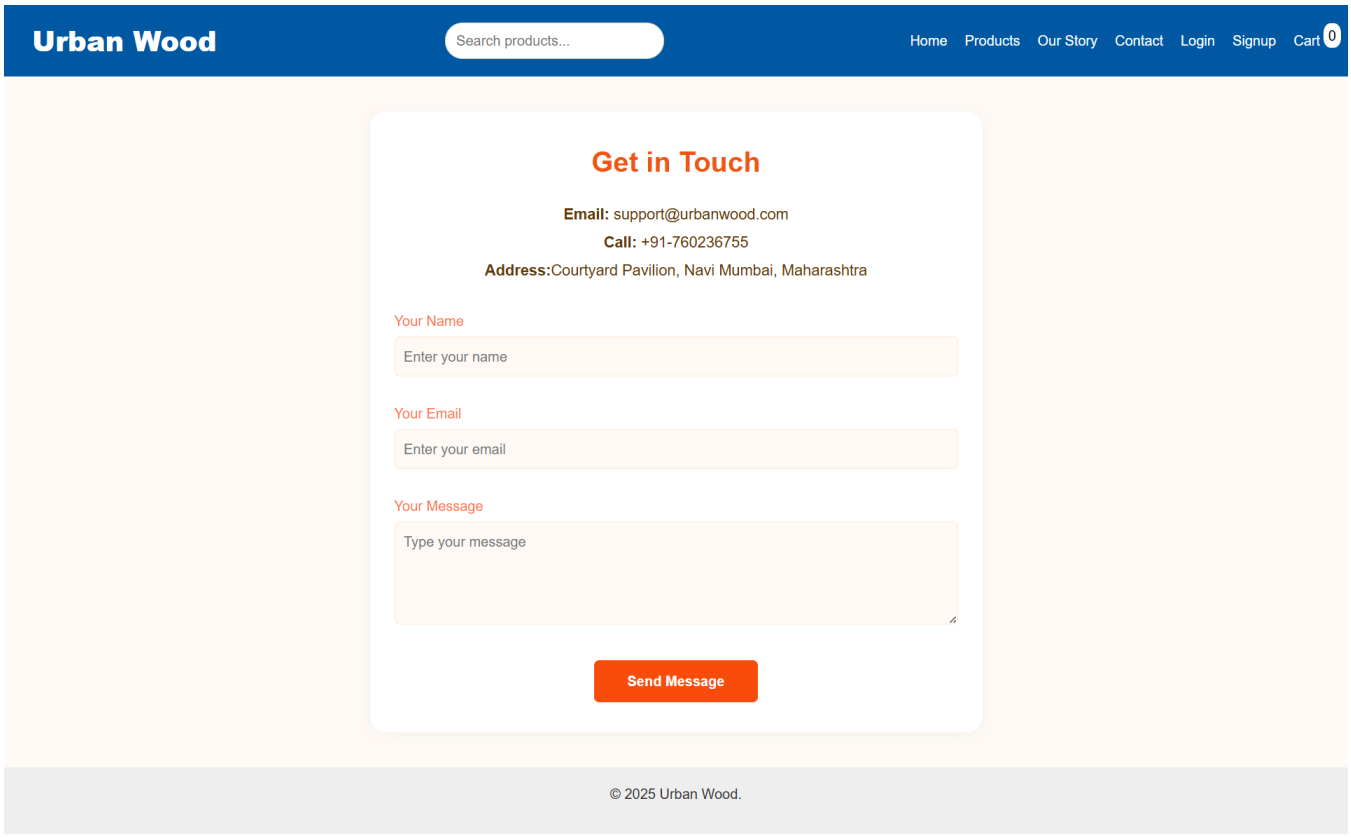
OUR STORY:



Code:

```
import Footer from "../component/Footer/Footer";

const About = () => {
    return (
        <>
            <div className="about-card">
                <h2>Our Story</h2>
                <p>
                   <span class="UrbanWood-accent"><b>Urban Wood</b></span>  crafts timeless, sustainable furniture that blends modern design with natural materials. Each piece is thoughtfully made to elevate your space with quality, comfort, and style — because your home deserves furniture that feels as good as it looks.
                </p>
                <p className="team-list">
                    <b>Team Members: </b>
                    <span class="UrbanWood-accent"><b>Milton Alam Sk,Tanisha Saha,Kaif Ajmi,Pritam Kar,Vaskar Ghosh,Koyel Debnath</b></span>
                </p>
                <div className="about-stats-row">
                    <div>
                        <span className="about-highlight">1,0000000+</span><br />Orders Delivered
                    </div>
                    <div>
                        <span className="about-highlight">9,00000+</span><br />Happy Customers
                    </div>
                    <div>
                        <span className="about-highlight">4.9★</span><br />Customer Rating
                    </div>
                </div>
            </div>
            <Footer />
        </>
    );
};

export default About;
```

CONTACT :

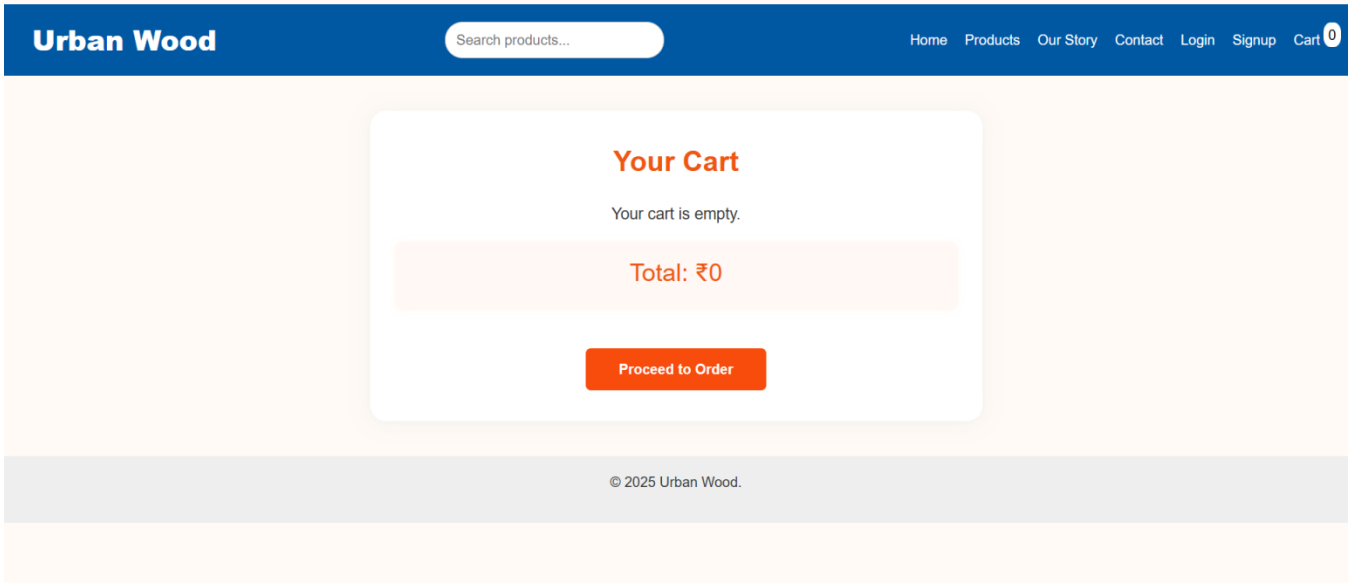


Code:

```
import Footer from "../component/Footer/Footer";

const Contact = () => {
    return (
        <>
            <section>
                <div className="main-card">
                    <h2>Get in Touch</h2>
                    <div className="contact-info">
                        <div><b>Email:</b> support@urbanwood.com</div>
                        <div><b>Call:</b> +91-760236755</div>
                        <div><b>Address:</b>Courtyard Pavilion, Navi Mumbai, Maharashtra</div>
                    </div>
                    <form>
                        <label htmlFor="name">Your Name</label>
                        <input type="text" id="name" required placeholder="Enter your name" />
                        <label htmlFor="email">Your Email</label>
                        <input type="email" id="email" required placeholder="Enter your email" />
                        <label htmlFor="message">Your Message</label>
                        <textarea id="message" rows="4" placeholder="Type your message"></textarea>
                        <button type="submit">Send Message</button>
                    </form>
                </div>
            </section>
            <Footer />
        </>
    );
}
```

```
export default Contact;
```

CART :



Code:-

```
import { useEffect, useState } from "react";
import { foodCard } from "../component/data";
import Footer from "../component/Footer/Footer";

const Cart = ({ cart, setCart }) => {
  const [total, setTotal] = useState(0);

  useEffect(() => {
    let sum = 0;
    for (const item of foodCard) {
      if (cart.includes(item.id)) sum += item.price;
    }
    setTotal(sum);
  }, [cart]);

  const handleDelete = id => {
    setCart(cart.filter(i => i !== id));
  };

  return (
    <>
      <section>
        <div className="main-card">
          <h2>Your Cart</h2>
          <div className="cart-items">
            {cart.length === 0 ? (
              <p>Your cart is empty.</p>
            ) : (
              foodCard
                .filter(item => cart.includes(item.id))
                .map(item => (
                  <div key={item.id} className="cart-card">
                    <img src={item.url} alt={item.title} />
                    <div className="cart-details">
                      <h3>{item.title}</h3>
                      <span>₹{item.price}</span>
```

```
                  </div>
                  <button className="delete-btn" onClick={() => handleDelete(item.id)}>
                    Delete
                  </button>
                </div>
              ))
          )}
        </div>
        <div className="cart-total">
          <h3>Total: ₹{total}</h3>
        </div>
        <a href="/order" className="btn">Proceed to Order</a>
      </div>
    </section>
    <Footer />
  </>
 );
};

export default Cart;
```

1. CONCLUSION

As the furniture retail industry in India continues to evolve, your furniture shop is well-positioned to capitalize on growing urban demand, rising consumer purchasing power, and expanding trends—especially in modular, customizable, and eco-conscious offerings. With a focused product range featuring high-quality designs and reliable logistics and pricing strategies, your shop can differentiate itself in a fragmented market dominated by both organized and unorganized players.

The store's key strengths include a carefully curated selection that balances form and function, competitive pricing, and excellent customer service—factors known to foster customer loyalty and repeat sales. While consumer appetite leans toward multifunctional and space-saving pieces, a growing affinity for sustainable and personalized furniture further shapes the market landscape.

However, the competitive landscape brings operational challenges, including navigating supply chain fluctuations, managing inventory effectively, and maintaining margins amid raw material price swings. Moreover, building visibility and trust in a largely unorganized sector requires strategic investments in branding and customer education

2. FUTURE SCOPE & FURTHER ENHANCEMENTS

Futurescope:-

- Development ofmobileappsforseamless accessbymembersand staff.
- Expansiontosupportmultiplegymbranchesorfranchises.
- IntegrationofAIforpersonalizedtrainingand nutritionsuggestions.
- Additionofvirtualfitnesssessionsand videolibraries.
- Advanceddataanalyticsforperformance,trends,anduser behavior.
- Real-timememberprogresstrackingandgoalsetting.
- BiometricorRFID-basedattendancesystemforsecurityandautomation.
- Corporatewellnessandgroupmembershipmanagement.
- Integrationwithwearablefitnessdevices.
- Cloud-basedscalabilitytosupportincreasingusers anddata load.

FURTHER ENHANCEMENTS:-

- **MobileAppSupport:-**EnablefullfunctionalityonAndroidandiOSplatforms.
- **AI-PoweredSuggestions:-**Personalizedworkoutanddiet plansbasedonuser data.
- **VirtualTraining:-**On-demandandlive-streamedfitnessclasses.
- **SmartChatbot:-**Instantcustomersupportforcommonqueriesandhelp.
- **Loyalty&RewardProgram:-**Pointsordiscountsforregularattendanceor referrals.
- **DetailedReports&Dashboards:-**Visualperformanceandusageanalyticsfor admin.
- **Biometric&RFIDIntegration:-**Secureandautomatedcheck-insystems.
- **Multi-LanguageSupport:-**Forgymsoperating indifferent regions.
- **CorporateAccountModule:-**Groupmanagementforcompanyemployees.
- **OfflineMode:-**Limitedaccessto featureswithoutinternet,syncingwhen reconnected.

3. BIBLIOGRAPHY

1) **https://www.youtube.com/**

2) **https://www.google.com/**

3) **https://github.com/**

4) https://nodejs.org/en

5) https://react.dev/