**A logo for a college

AI-generated content may be incorrect.**

**BSc. (Hons) in Software Engineering**

**Department of Information Technology**

**Faculty of Engineering**

**and Technology**

**K. M. Amandi De Silva**

**F20030112002**

**Module: IT3113**

**Submission Date: 07/05/2025**

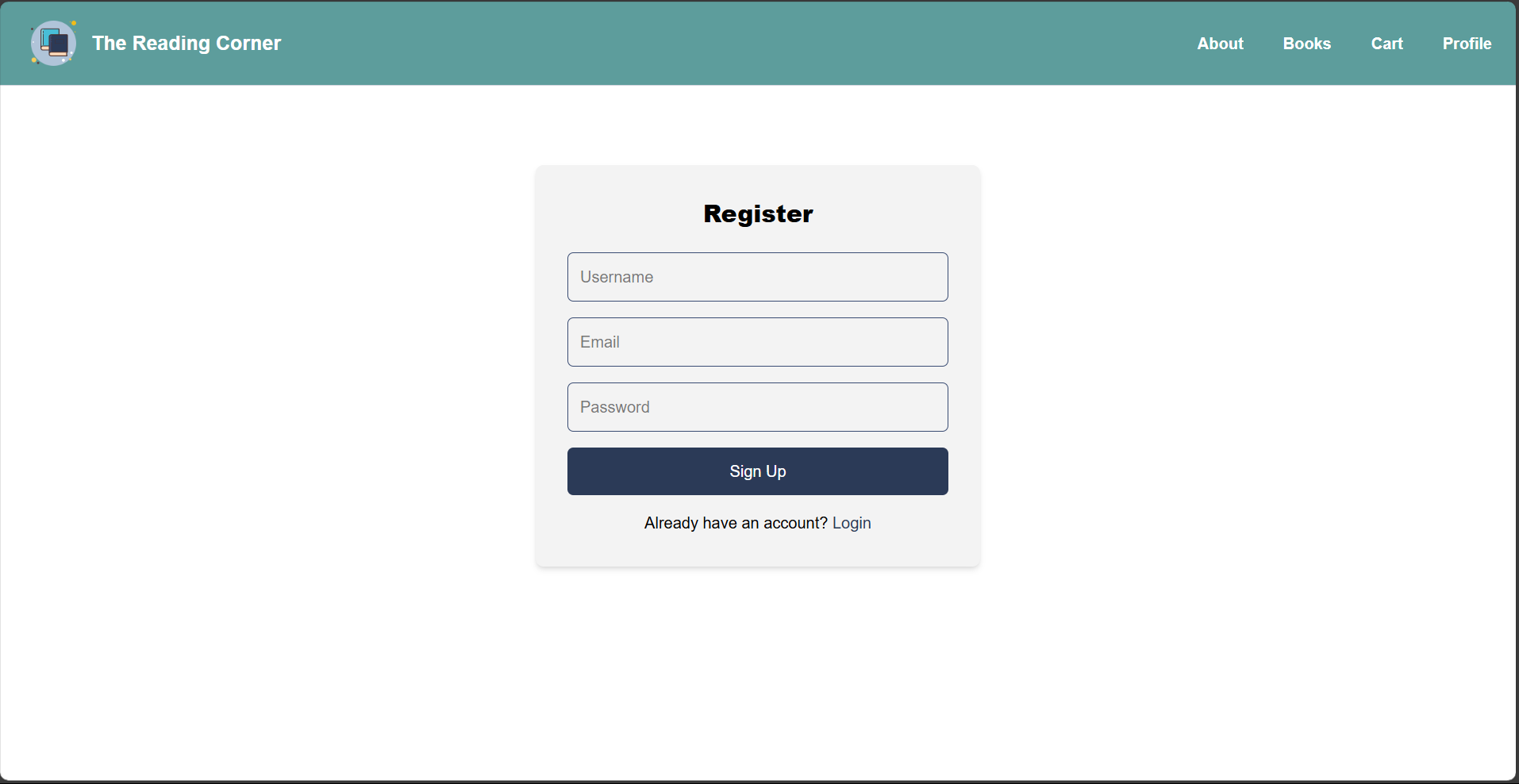
**A blue and black book

AI-generated content may be incorrect.The Reading Corner**

**1. User Interface Documentation Requirements**

**1.1 User Authentication Screens**

**Registration Page**

* **Screenshot:**
* **Code:**

import React, { useState } from 'react';

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

const RegisterPage = () => {

  const navigate = useNavigate();

  const [formData, setFormData] = useState({

    username: '',

    email: '',

    password: ''

  });

  const [error, setError] = useState('');

  const [loading, setLoading] = useState(false);

  const handleChange = (e) => {

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

  };

  const handleRegister = async (e) => {

    e.preventDefault();

    const { username, email, password} = formData;

    if (!username || !email || !password) {

      return setError('All fields are required.');

    }

    try {

      setLoading(true);

      const res = await fetch('http://localhost:4000/api/auth/register', {

        method: 'POST',

        headers: { 'Content-Type': 'application/json' },

        credentials: 'include',

        body: JSON.stringify({ username, email, password }),

      });

      const data = await res.json();

      setLoading(false);

      if (data.success) {

        navigate('/login');

      } else {

        setError(data.message || 'Registration failed.');

      }

    } catch (err) {

      setLoading(false);

      setError('Server error. Please try again later.');

    }

  };

  return (

    <div className="max-w-md mx-auto p-8 bg-[#f3f3f3] rounded-lg shadow-md m-20">

      <h2 className="text-2xl font-extrabold text-center mb-6 text-[#000]">Register</h2>

      {error && <p className="text-red-500 text-center">{error}</p>}

      <form onSubmit={handleRegister}>

        <input

          type="text"

          name="username"

          placeholder="Username"

          onChange={handleChange}

          className="w-full p-3 mb-4 border border-[#3A4C73] rounded-md focus:outline-none focus:ring-2 focus:ring-[#3A4C73]"

        />

        <input

          type="email"

          name="email"

          placeholder="Email"

          onChange={handleChange}

          className="w-full p-3 mb-4 border border-[#3A4C73] rounded-md focus:outline-none focus:ring-2 focus:ring-[#3A4C73]"

        />

        <input

          type="password"

          name="password"

          placeholder="Password"

          onChange={handleChange}

          className="w-full p-3 mb-4 border border-[#3A4C73] rounded-md focus:outline-none focus:ring-2 focus:ring-[#3A4C73]"

        />

        <button

          type="submit"

          disabled={loading}

          className="w-full p-3 bg-[#2B3A57] text-white rounded-md hover:bg-[#3A4C73] focus:outline-none focus:ring-2 focus:ring-[#3A4C73]"

        >

          {loading ? 'Signing up...' : 'Sign Up'}

        </button>

        <p className="mt-4 text-center text-[#000]">

          Already have an account?{" "}

          <Link to="/login" className="text-[#2B3A57]">

            Login

          </Link>

        </p>

      </form>

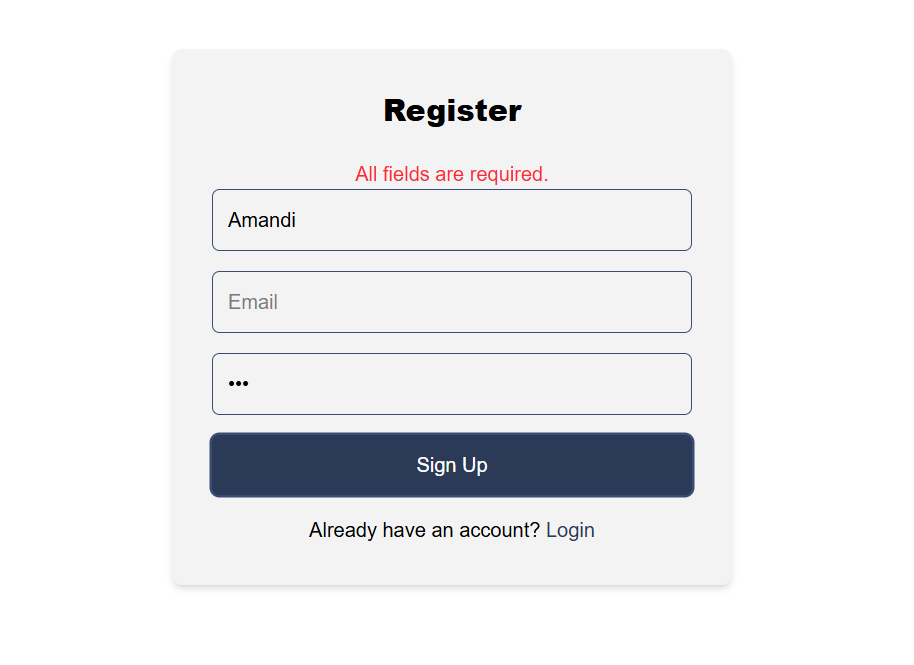
    </div>

  );

};

export default RegisterPage;

* **Validation & Error Messages:**



**Login Page**

* **Screenshot:**

A screenshot of a login form

AI-generated content may be incorrect.

* **Code:**

import React, { useState } from 'react';

import { useNavigate, useLocation, Link } from 'react-router-dom';

const LoginPage = () => {

  const navigate = useNavigate();

  const location = useLocation();

  const redirectTo = location.state?.redirectTo || '/profile';

  const [email, setEmail] = useState('');

  const [password, setPassword] = useState('');

  const [error, setError] = useState('');

  const [loading, setLoading] = useState(false);

  const [rememberMe, setRememberMe] = useState(false);

  const handleLogin = async (e) => {

    e.preventDefault();

    try {

      setLoading(true);

      const res = await fetch('http://localhost:4000/api/auth/login', {

        method: 'POST',

        headers: { 'Content-Type': 'application/json' },

        credentials: 'include',

        body: JSON.stringify({ email, password })

      });

      const data = await res.json();

      setLoading(false);

      if (data.success) {

        navigate(redirectTo);

      } else {

        setError(data.message || 'Login failed.');

      }

    } catch (err) {

      setLoading(false);

      setError('Server error. Please try again later.');

    }

  };

  return (

    <div className="max-w-md mx-auto p-8 bg-[#f3f3f3] rounded-lg shadow-md m-20">

      <h2 className="text-2xl font-extrabold text-black mb-5 text-center">Login</h2>

      {error && <p className="text-red-500 text-center">{error}</p>}

      <form onSubmit={handleLogin}>

        <input

          type="email"

          placeholder="Email"

          value={email}

          onChange={(e) => setEmail(e.target.value)}

          className="w-full p-3 mb-4 border rounded-md focus:outline-none focus:ring-2 focus:ring-[#3A4C73]"

        />

        <input

          type="password"

          placeholder="Password"

          value={password}

          onChange={(e) => setPassword(e.target.value)}

          className="w-full p-3 mb-4 border rounded-md focus:outline-none focus:ring-2 focus:ring-[#3A4C73]"

        />

        <label className="mb-5 flex items-center">

          <input

            type="checkbox"

            checked={rememberMe}

            onChange={(e) => setRememberMe(e.target.checked)}

            className="mr-2"

          />

          Remember me

        </label>

        <button

          type="submit"

          disabled={loading}

          className="w-full p-3 bg-[#2B3A57] text-white rounded-md hover:bg-[#3A4C73] focus:outline-none focus:ring-2 focus:ring-[#3A4C73]"

        >

          {loading ? 'Logging in...' : 'Login'}

        </button>

        <p className="mt-4 text-center text-[#000]">

          Don't have an account? <Link to="/register" className="text-[#2B3A57]">Register</Link>

        </p>

      </form>

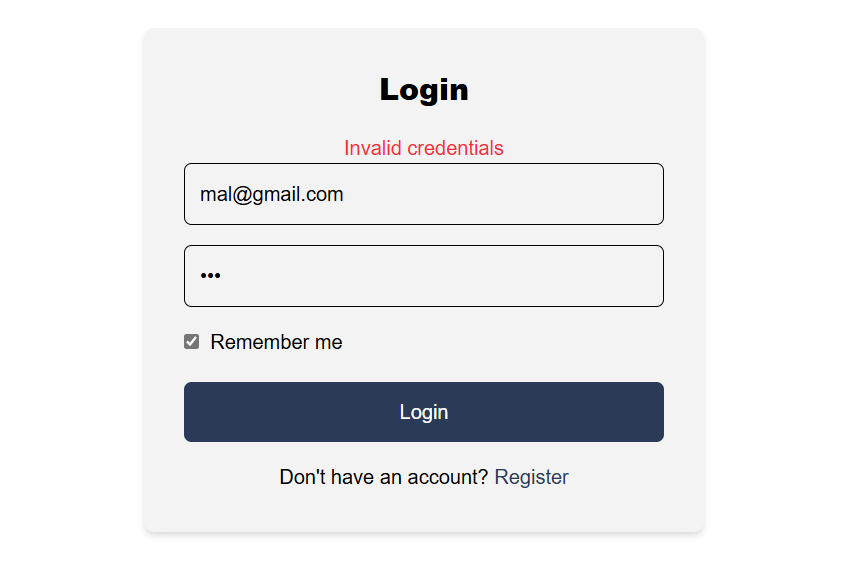
    </div>

  );

};

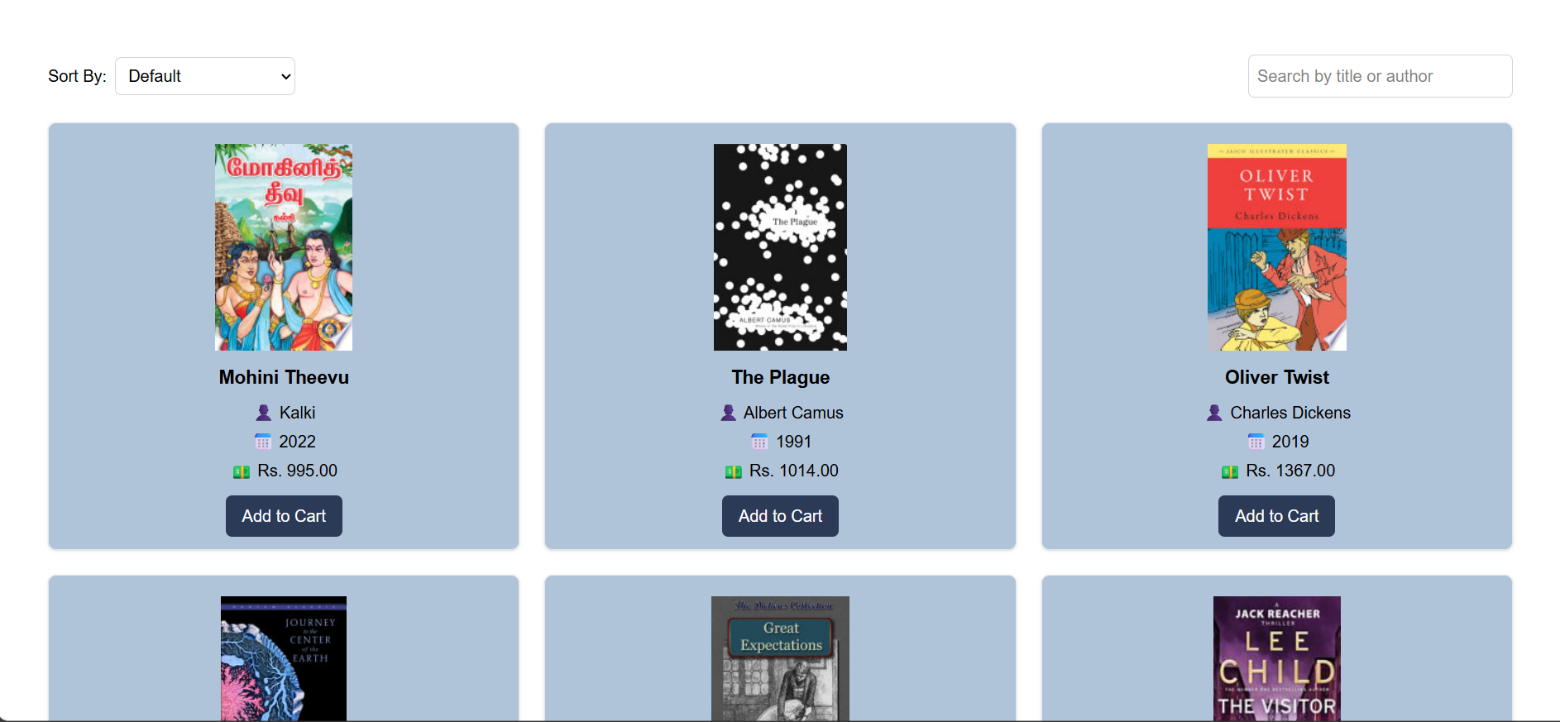
export default LoginPage;

* **Authentication Messages:**
* Verify the email address
* Verify the Password



**1.2 Book Listing and Search**

**Main Book Listing Page**

* **Screenshot:**
* **Code:**
  + API Integration
  + Search Implementation

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

import BookCard from "../components/BookCard";

const BookPage = () => {

  const [books, setBooks] = useState([]);

  const [searchTerm, setSearchTerm] = useState("");

  const [filter, setFilter] = useState("default");

  useEffect(() => {

    const query = searchTerm.trim() !== ""

      ? encodeURIComponent(searchTerm)

      : "subject:fiction";

    fetch(

      `https://www.googleapis.com/books/v1/volumes?q=${query}&maxResults=30&orderBy=relevance`

    )

      .then((res) => res.json())

      .then((data) => {

        const processed = (data.items || []).map((item) => ({

          id: item.id,

          title: item.volumeInfo.title,

          authors: item.volumeInfo.authors || ["Unknown"],

          image:

            item.volumeInfo.imageLinks?.thumbnail ||

            "https://via.placeholder.com/128x193?text=No+Cover",

          year: item.volumeInfo.publishedDate?.slice(0, 4) || "Unknown",

          price: Math.floor(Math.random() \* 1000) + 500,

        }));

        setBooks(processed);

      })

      .catch(() => setBooks([]));

  }, [searchTerm]);

  // Only sorting now—no client‐side filtering by searchTerm

  const sortedBooks = [...books].sort((a, b) => {

    if (filter === "title") return a.title.localeCompare(b.title);

    if (filter === "price-low") return a.price - b.price;

    if (filter === "price-high") return b.price - a.price;

    if (filter === "year") return b.year.localeCompare(a.year);

    return 0;

  });

  return (

    <div className="px-4 py-6 md:px-8 md:py-10 lg:px-12 lg:py-14">

      <div className="flex flex-col md:flex-row justify-between items-center mb-6 gap-4">

        <div className="flex items-center gap-2">

          <label htmlFor="sort" className="font-medium">

            Sort By:

          </label>

          <select

            id="sort"

            onChange={(e) => setFilter(e.target.value)}

            className="p-2 border border-gray-300 rounded-md text-base"

          >

            <option value="default">Default</option>

            <option value="title">Title (A-Z)</option>

            <option value="price-low">Price (Low to High)</option>

            <option value="price-high">Price (High to Low)</option>

            <option value="year">Published Year</option>

          </select>

        </div>

        <div className="flex items-center gap-2 w-full md:w-auto">

          <input

            type="text"

            placeholder="Search by title or author"

            value={searchTerm}

            onChange={(e) => setSearchTerm(e.target.value)}

            className="p-2 border border-gray-300 rounded-md text-base w-full md:w-64"

          />

        </div>

      </div>

      <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-6">

        {sortedBooks.map((book) => (

          <BookCard key={book.id} book={book} />

        ))}

      </div>

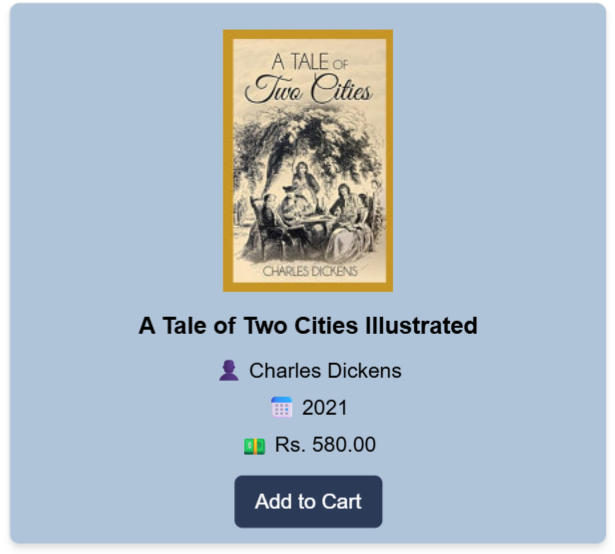
    </div>

  );

};

export default BookPage;

* **Details Displayed:**



* **Code:**

import React from "react";

import { useNavigate } from "react-router-dom";

import { useCart } from "../context/CartContext";

const BookCard = ({ book }) => {

  const navigate = useNavigate();

  const { addToCart } = useCart();

  const goToDetails = () => {

    navigate(`/book/${book.id}`);

  };

  const goToCart = (e) => {

    e.stopPropagation();

    addToCart(book, "/about");

    navigate("/cart");

  };

  return (

    <div

      className="border border-gray-200 rounded-lg p-2 bg-[#B0C4D9] shadow-sm max-h-[600px] pb-3 text-center hover:scale-105 hover:shadow-md transition-transform duration-200"

      onClick={goToDetails}

    >

      <img

        src={book.image}

        alt={book.title}

        className="h-[200px] object-cover mx-auto my-3"

      />

      <h3 className="text-lg font-bold text-black mb-2">{book.title}</h3>

      <div className="space-y-1">

        <p className="font-medium text-black">👤 {book.authors.join(", ")}</p>

        <p className="text-black">📅 {book.year}</p>

        <p className="text-black">💵 Rs. {book.price}.00</p>

      </div>

      <button

        onClick={goToCart}

        className="mt-3 bg-[#2B3A57] text-white px-4 py-2 rounded-md hover:bg-[#3A4C73] transition-colors duration-300"

      >

        Add to Cart

      </button>

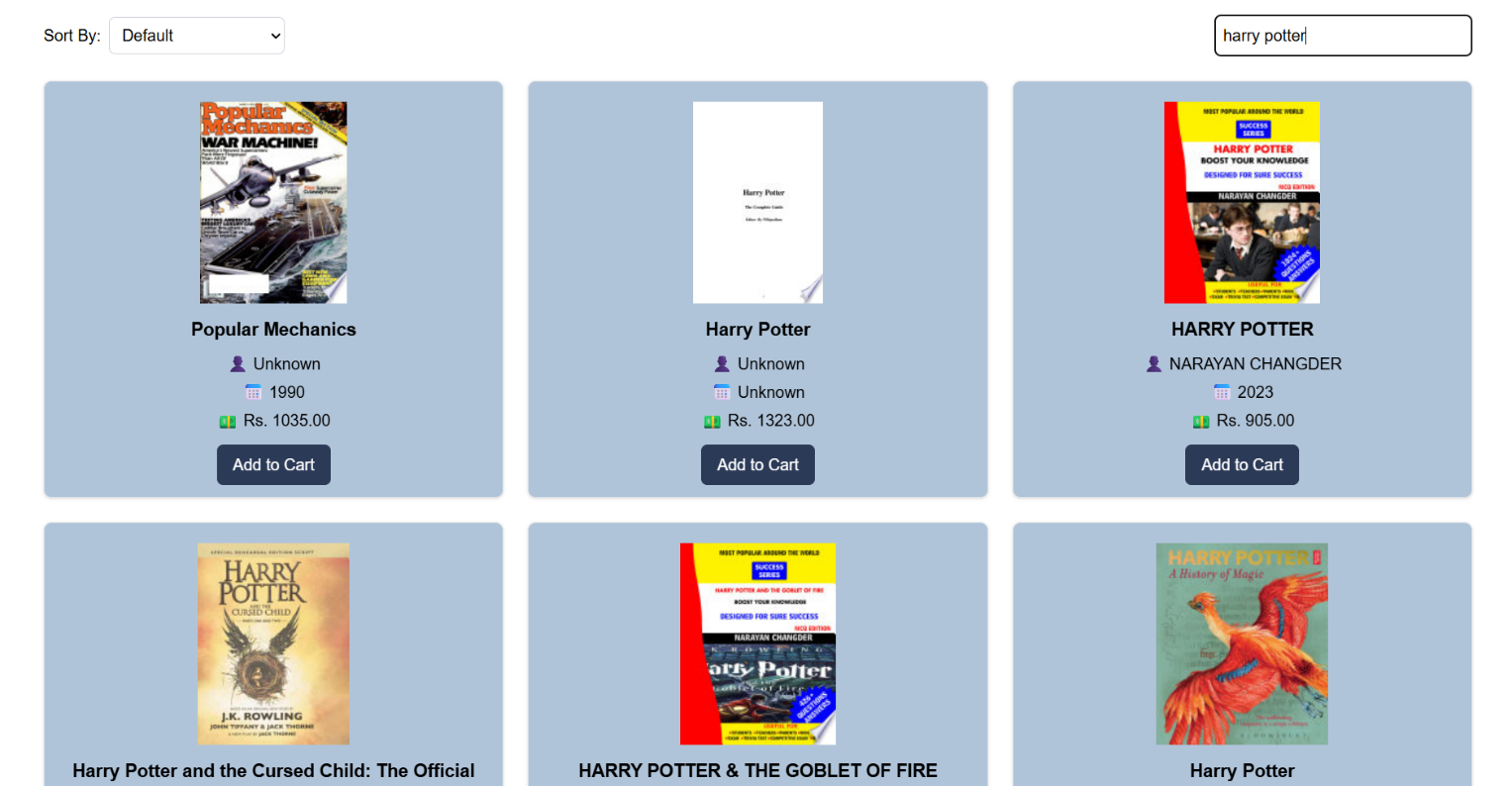
    </div>

  );

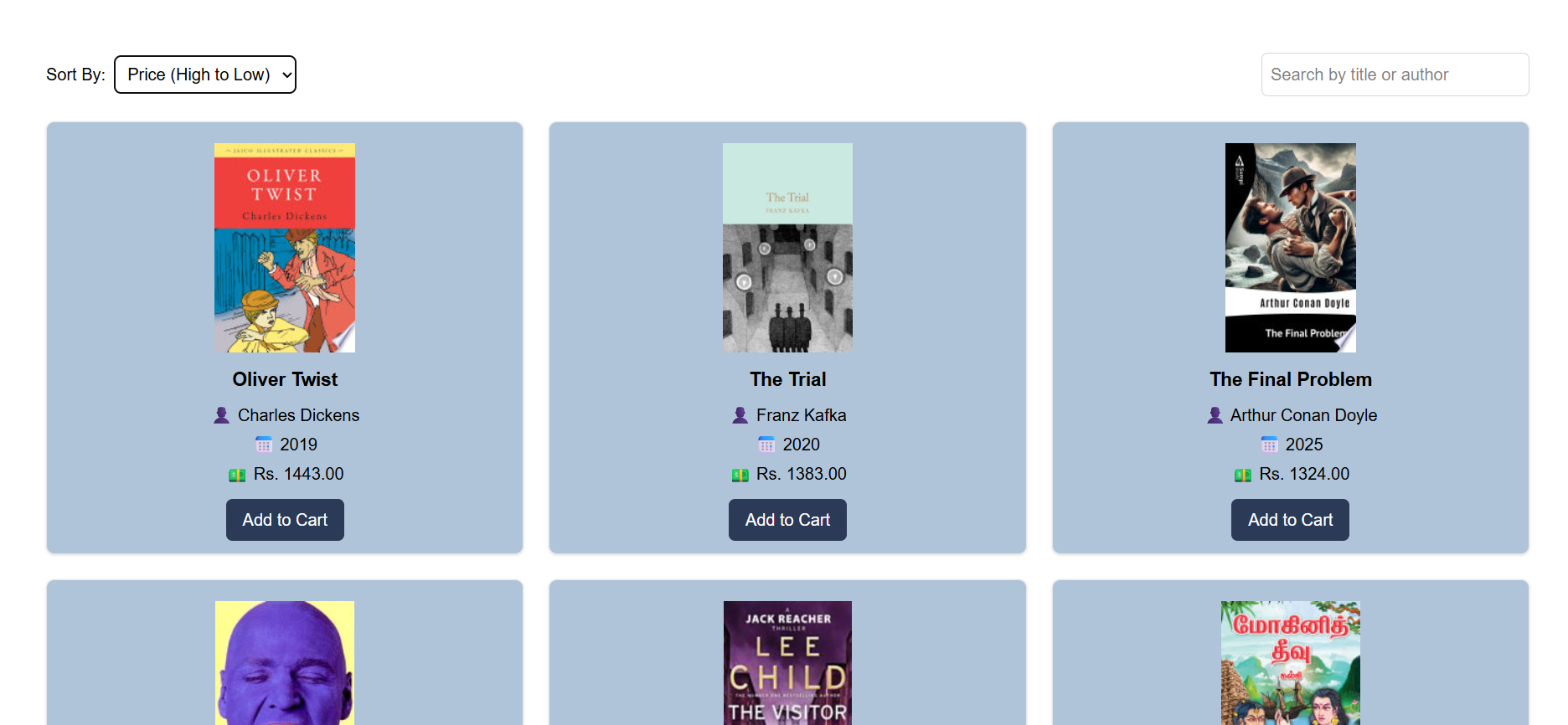
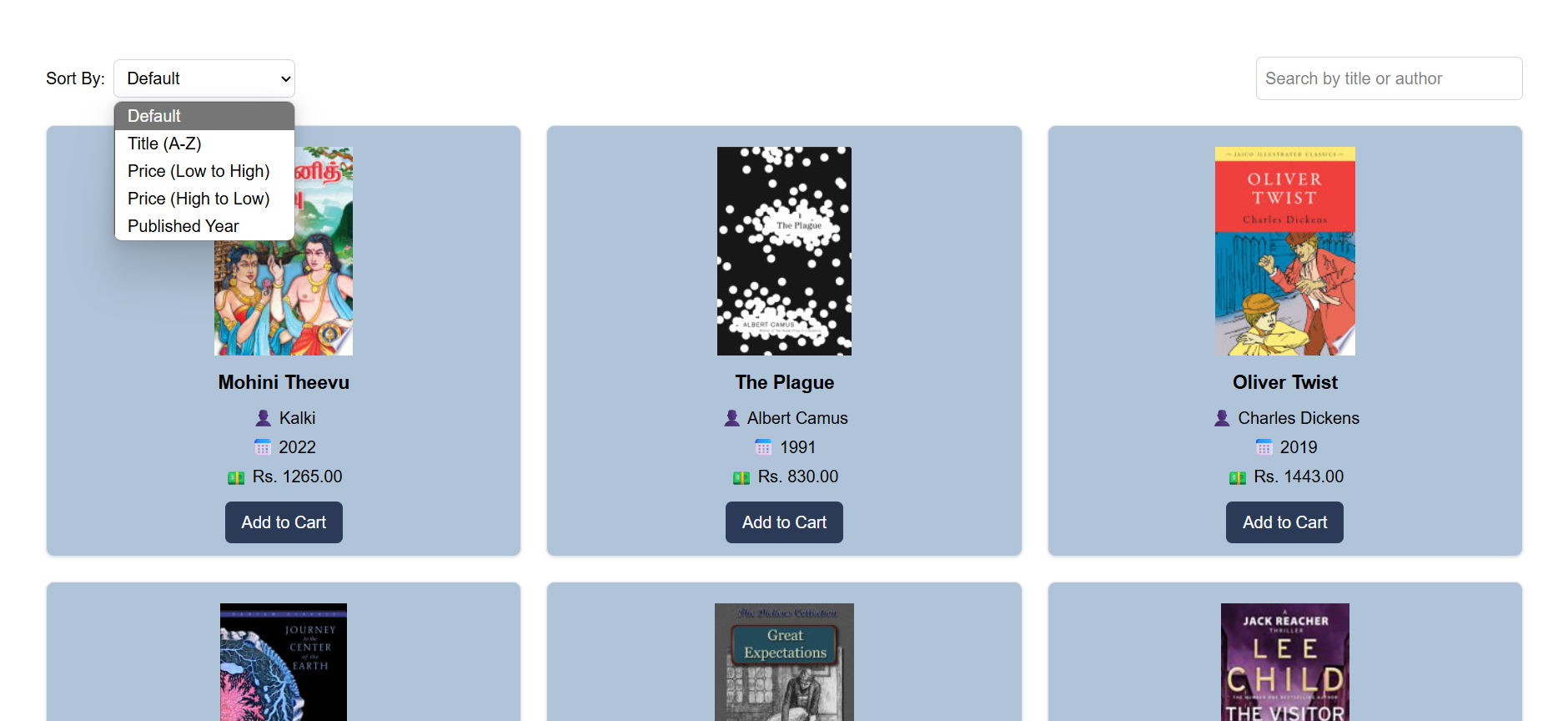
};

export default BookCard;

**Search Functionality**

* **Screenshot:**

**Filter/Sort Options**

* **Screenshot:**
* **Functionality:**

**Book Details View**

* **Screenshot:**
* A screenshot of a computer

  AI-generated content may be incorrect.This book is available in store.
* This book is not available in store.
* **Code:**

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

import { useParams, useNavigate, useLocation } from "react-router-dom";

import { useCart } from "../context/CartContext";

const BookDetailsPage = () => {

  const { id } = useParams();

  const navigate = useNavigate();

  const location = useLocation();

  const { addToCart } = useCart();

  const [book, setBook] = useState(null);

  useEffect(() => {

    fetch(`https://www.googleapis.com/books/v1/volumes/${id}`)

      .then((res) => res.json())

      .then((data) => {

        setBook({

          title: data.volumeInfo.title,

          authors: data.volumeInfo.authors || ["Unknown"],

          year: data.volumeInfo.publishedDate?.slice(0, 4) || "Unknown",

          image:

            data.volumeInfo.imageLinks?.thumbnail ||

            "https://via.placeholder.com/128x193?text=No+Cover",

          price: Math.floor(Math.random() \* 1000) + 500,

          description:

            data.volumeInfo.description || "No description available.",

          stock: Math.floor(Math.random() \* 4),

        });

      });

  }, [id]);

  if (!book) return <p>Loading...</p>;

  return (

    <div className="flex flex-wrap items-start gap-[70px] p-10">

      <img

        src={book.image}

        alt={book.title}

        className="w-[300px] h-auto object-cover"

      />

      <div className="flex-1 max-w-[1000px]">

        <h2 className="text-2xl font-semibold text-[#2B3A57]">

          {book.title}

        </h2>

        <p className="mt-2 text-base">

          <strong>Author:</strong> {book.authors.join(", ")}

        </p>

        <p className="mt-2 text-base">

          <strong>Year:</strong> {book.year}

        </p>

        <p className="mt-2 text-base">

          <strong>Price:</strong> Rs. {book.price}.00

        </p>

        <p className="mt-2 text-base">

          <strong>Status:</strong>{" "}

          {book.stock === 0

            ? "Out of stock"

            : book.stock === 1

            ? "1 book available"

            : `${book.stock} books available`}

        </p>

        <p className="mt-2 text-base">

          <strong>Description:</strong>

        </p>

        <div

          className="text-black leading-relaxed text-justify mt-2"

          dangerouslySetInnerHTML={{ \_\_html: book.description }}

        ></div>

        <button

          disabled={book.stock === 0}

          className="mt-5 bg-[#2B3A57] text-white px-4 py-2 rounded-lg transition-colors duration-300 disabled:opacity-80 disabled:cursor-default disabled:text-gray-400 hover:bg-[#3A4C73]"

          onClick={() => {

            addToCart({ ...book, id }, location.pathname);

            navigate("/cart");

          }}

        >

          Add to Cart

        </button>

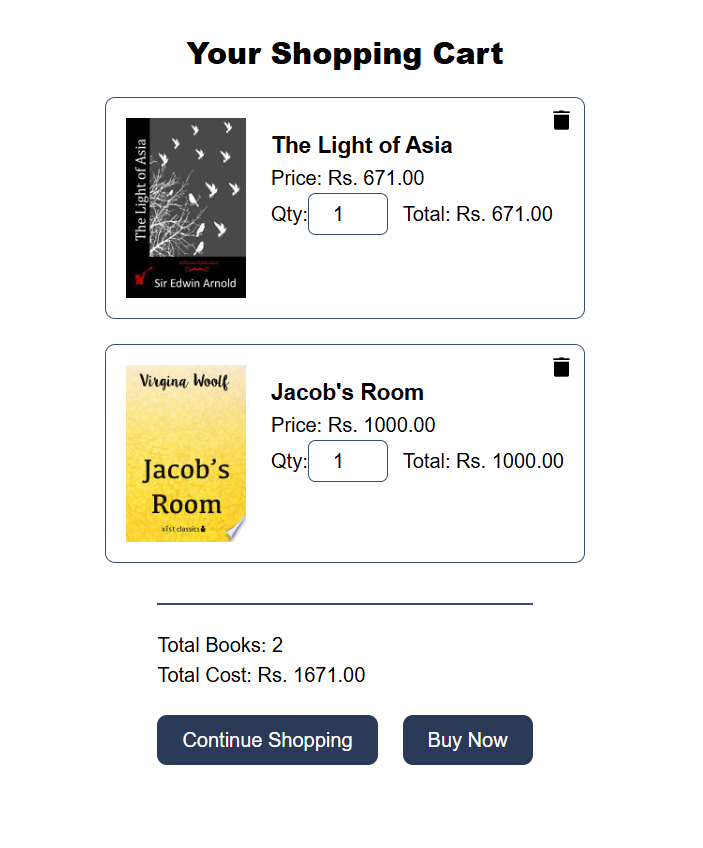
      </div>

    </div>

  );

};

export default BookDetailsPage;

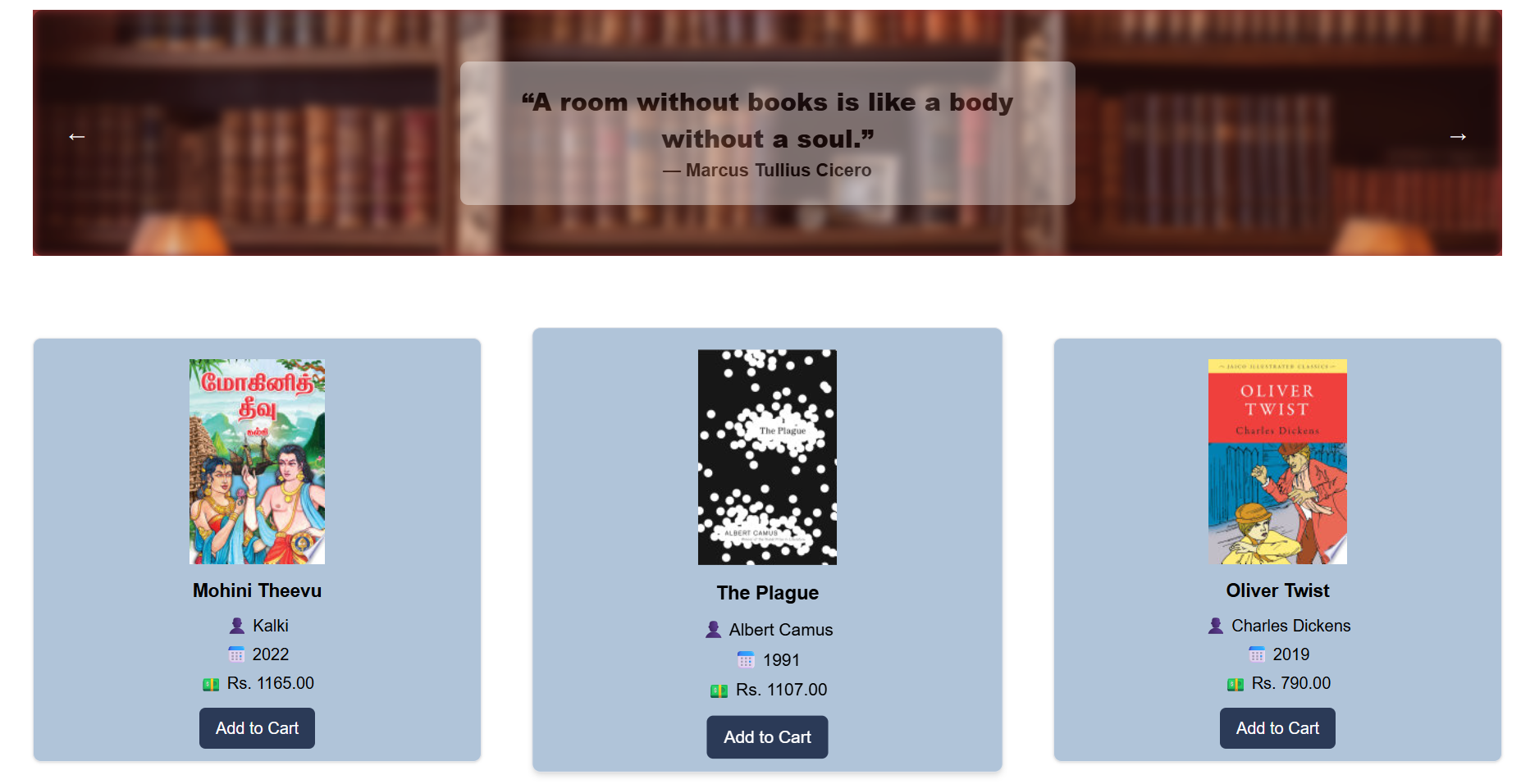
**1.3 Shopping Cart**

**Cart View**

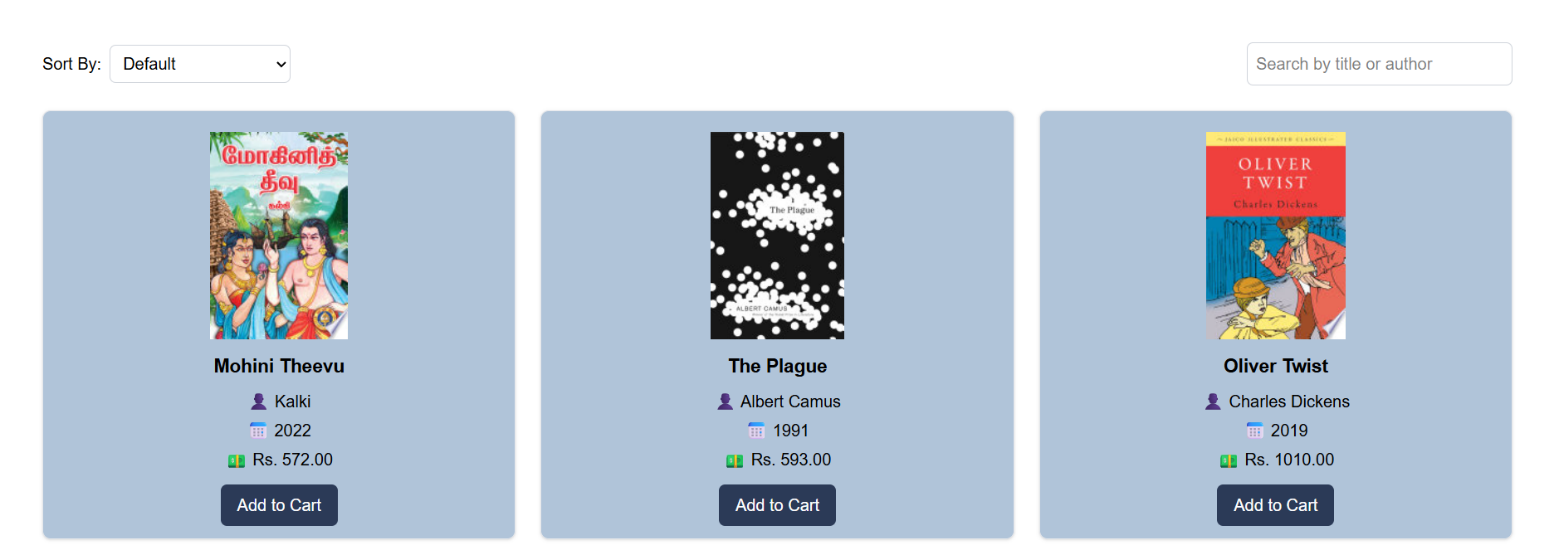
* **Screenshot:**

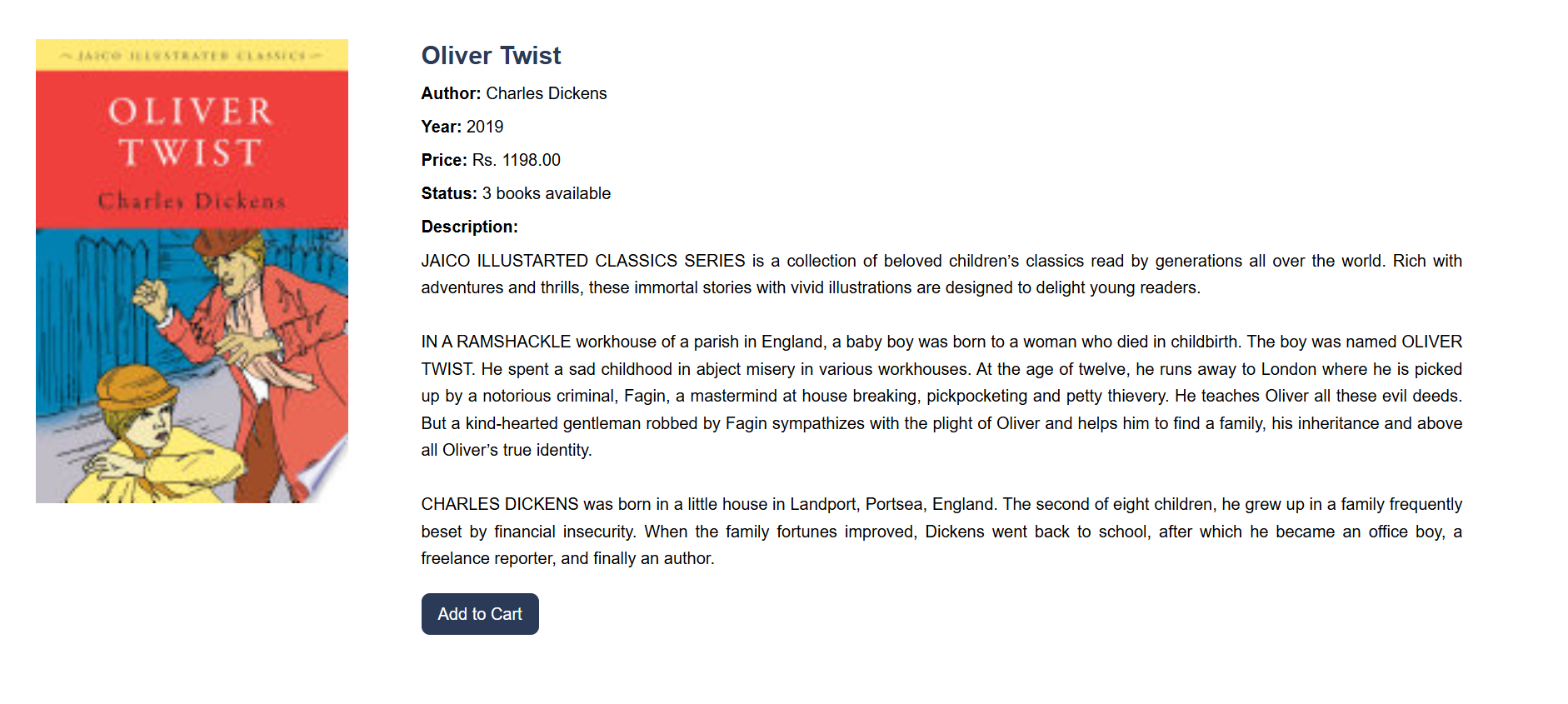
**Add to Cart Functionality**

* **Screenshot:**
* From About page



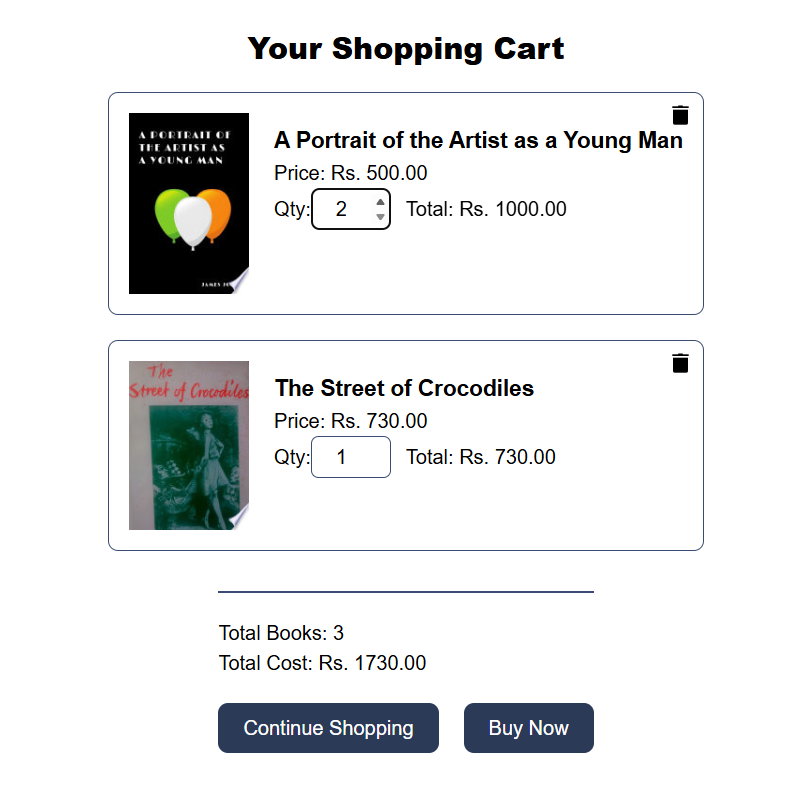
* From Book page



* From Book details page

**Update Quantity Interface, Remove Items Interface**, **Cart Total Calculation**

* **Screenshot:**



* **Code:**

import React from 'react';

import { useCart } from '../context/CartContext';

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

const CartPage = () => {

  const { cartItems, updateQuantity, removeFromCart, lastPage } = useCart();

  const navigate = useNavigate();

  const totalBooks = cartItems.reduce((sum, item) => sum + item.quantity, 0);

  const totalCost = cartItems.reduce(

    (sum, item) => sum + item.quantity \* item.price,

    0

  );

  const handleContinueShopping = () => {

    if (lastPage === '/about') {

      navigate('/about');

    } else {

      navigate('/books');

    }

  };

  const handleBuyNow = async () => {

    try {

      const res = await fetch('http://localhost:4000/api/user/data', {

        credentials: 'include'

      });

      const data = await res.json();

      if (data.success) {

        navigate('/shipping');

      } else {

        navigate('/login', { state: { redirectTo: '/shipping' } });

      }

    } catch (err) {

      navigate('/login', { state: { redirectTo: '/shipping' } });

    }

  };

  return (

    <div className="cart-wrapper px-8 py-8 flex items-center justify-center flex-col">

      <h2 className="text-2xl font-extrabold text-black mb-5 text-center">

        Your Shopping Cart

      </h2>

      {cartItems.length === 0 ? (

        <p>Your cart is empty.</p>

      ) : (

        <>

          <div className="cart-list flex flex-col gap-5">

            {cartItems.map((item) => (

              <div

                className="cart-item relative bg-white p-4 border border-[#3A4C73] rounded-lg flex gap-5"

                key={item.id}

              >

                <button

                  className="delete-btn absolute top-2 right-2 bg-none border-none cursor-pointer"

                  onClick={() => removeFromCart(item.id)}

                >

                  <img

                    src="https://img.icons8.com/ios-glyphs/30/000000/delete.png"

                    alt="Delete"

                    className="w-5 h-5 filter-brightness-0 transition-transform duration-200 hover:scale-110 hover:filter-brightness-40"

                  />

                </button>

                <img src={item.image} alt={item.title} className="w-24 h-auto" />

                <div className="info flex flex-col">

                  <h4 className="text-lg font-bold mt-2">{item.title}</h4>

                  <p>Price: Rs. {item.price}.00</p>

                  <div className="flex items-center">

                    Qty:{' '}

                    <input

                      type="number"

                      min="1"

                      value={item.quantity}

                      onChange={(e) =>

                        updateQuantity(item.id, e.target.value)

                      }

                      className="w-16 py-1 text-base border border-[#3A4C73] rounded-md text-center appearance-none"

                    />

                    <span className="ml-3">

                      Total: Rs. {item.quantity \* item.price}.00

                    </span>

                  </div>

                </div>

              </div>

            ))}

          </div>

          <div className="cart-summary mt-8 pt-5 border-t-2 border-[#3A4C73]">

            <h4>Total Books: {totalBooks}</h4>

            <h4>Total Cost: Rs. {totalCost}.00</h4>

            <div className="cart-buttons mt-5 flex gap-5">

              <button

                onClick={handleContinueShopping}

                className="px-5 py-2 bg-[#2B3A57] text-white rounded-lg hover:bg-[#3A4C73]"

              >

                Continue Shopping

              </button>

              <button

                onClick={handleBuyNow}

                className="px-5 py-2 bg-[#2B3A57] text-white rounded-lg hover:bg-[#3A4C73]"

              >

                Buy Now

              </button>

            </div>

          </div>

        </>

      )}

    </div>

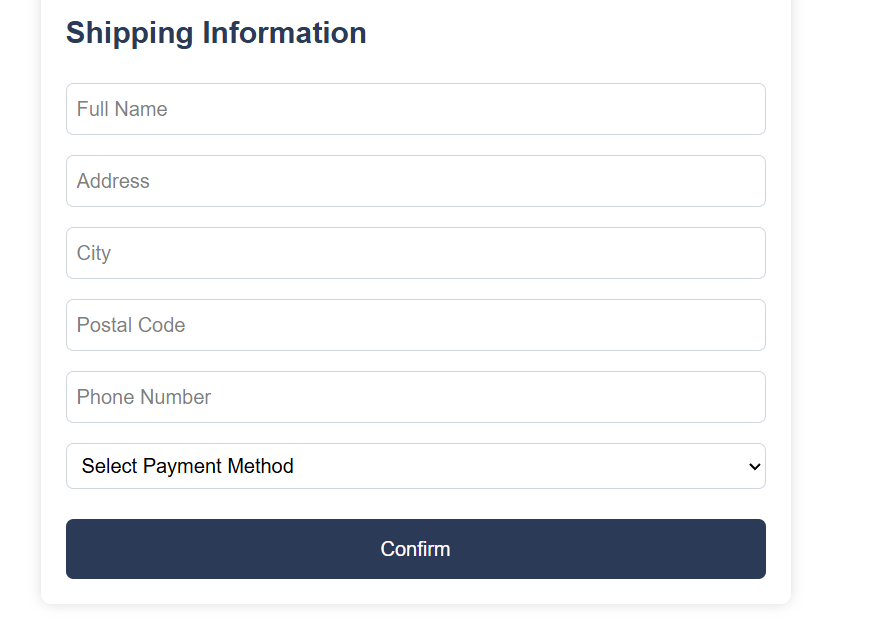
  );

};

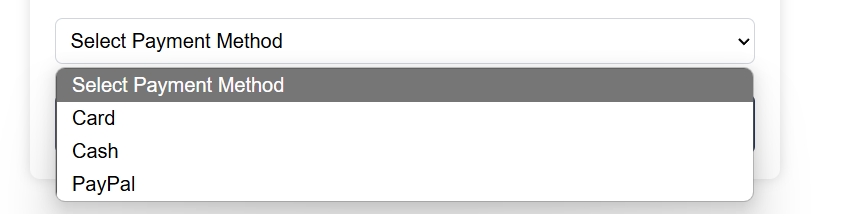
export default CartPage;

**1.4 Checkout Process**

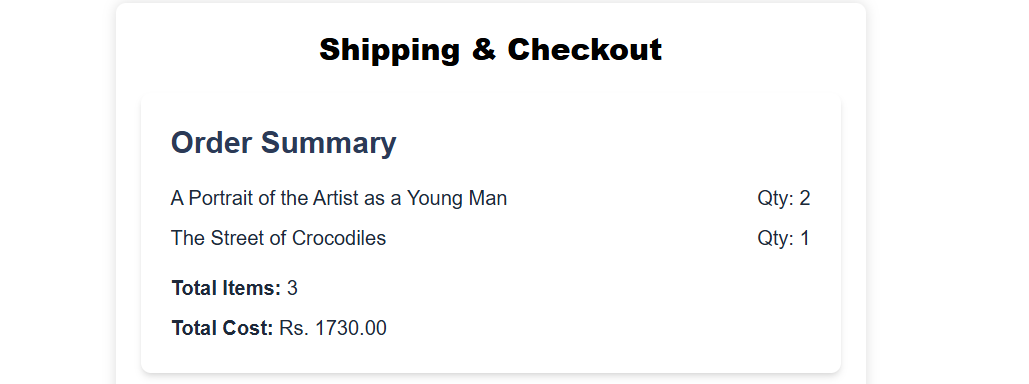
**Checkout Form (Shipping Information)**

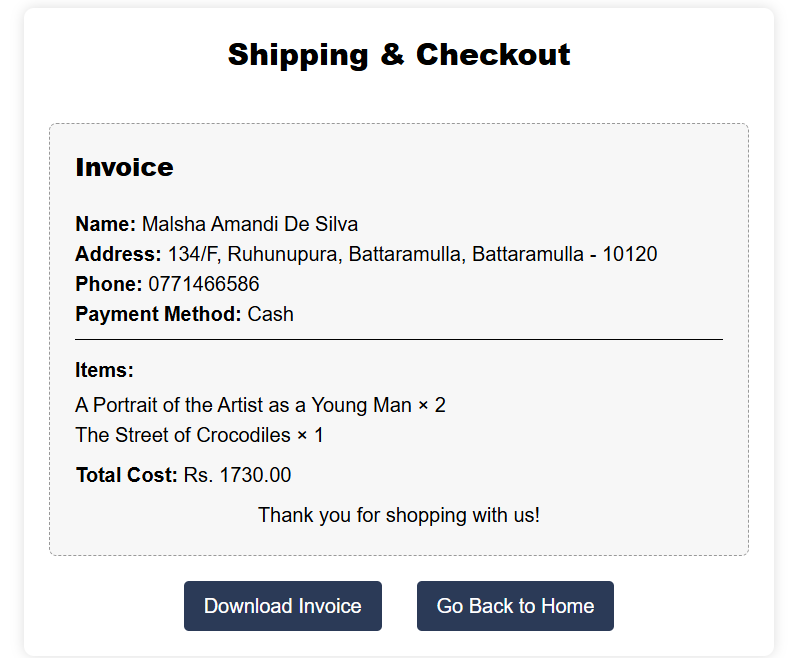
* **Screenshot:**

**Payment Interface**

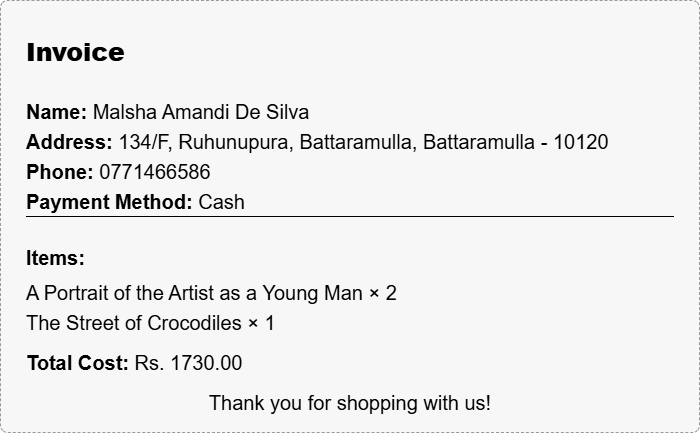
* **Screenshot:**

**Order Confirmation Page**

* **Screenshot:**



* **Invoice:**
* Download invoice as PNG



* **Code**

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

import { useCart } from '../context/CartContext';

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

import html2canvas from 'html2canvas';

const ShippingPage = () => {

  const { cartItems } = useCart();

  const navigate = useNavigate();

  const invoiceRef = useRef();

  const [showInvoice, setShowInvoice] = useState(false);

  const [error, setError] = useState('');

  const [shipping, setShipping] = useState({ fullName:'', address:'', city:'', postalCode:'', phone:'' });

  const [paymentMethod, setPaymentMethod] = useState('');

  const totalItems = cartItems.reduce((sum, i) => sum + i.quantity, 0);

  const totalCost = cartItems.reduce((sum, i) => sum + i.quantity \* i.price, 0);

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

  const handleConfirm = async () => {

    const { fullName, address, city, phone } = shipping;

    if (!fullName || !address || !city || !phone || !paymentMethod) {

      setError('Please fill in all required fields'); return;

    }

    try {

      const res = await fetch('http://localhost:4000/api/orders/createorder', {

        method:'POST', credentials:'include', headers:{'Content-Type':'application/json'},

        body:JSON.stringify({ items: cartItems, shipping, paymentMethod, totalItems, totalCost })

      });

      const data = await res.json();

      if (data.success) {

        setShowInvoice(true);

      } else setError(data.message);

    } catch(err) {

      setError('Error placing order');

    }

  };

  const handleDownload = async () => {

    const canvas = await html2canvas(invoiceRef.current);

    const link = document.createElement('a'); link.download='invoice.png';

    link.href=canvas.toDataURL(); link.click();

  };

  return (

    <div className="max-w-[600px] mx-auto my-10 p-5 bg-white rounded-lg shadow-[0\_0\_10px\_#ddd]">

      <h2 className="text-2xl font-extrabold text-black mb-5 text-center">Shipping & Checkout</h2>

      {!showInvoice ? (

        <>

          <div className="bg-white p-6 rounded-lg shadow-md">

  <h3 className="text-2xl font-bold text-[#2B3A57] mb-4">

    Order Summary

  </h3>

  {cartItems.map((item) => (

    <div

      key={item.id}

      className="flex justify-between mb-2"

    >

      <span className="text-base text-gray-800">

        {item.title}

      </span>

      <span className="text-base text-gray-800">

        Qty: {item.quantity}

      </span>

    </div>

  ))}

  <p className="mt-4 text-base text-gray-800">

    <strong>Total Items:</strong> {totalItems}

  </p>

  <p className="mt-2 text-base text-gray-800">

    <strong>Total Cost:</strong> Rs. {totalCost}.00

  </p>

</div>

<div className="shipping-form">

  <h3 className="text-2xl font-bold text-[#2B3A57] mb-4 mt-6">Shipping Information</h3>

  {error && (

  <p className="text-red-600 font-medium text-sm mb-3">{error}</p>

)}

  <input

   required

    type="text"

    name="fullName"

    placeholder="Full Name"

    onChange={handleChange}

    className="w-full my-2 p-2 border border-gray-300 rounded-md"

  />

  <input

  required

    type="text"

    name="address"

    placeholder="Address"

    onChange={handleChange}

    className="w-full my-2 p-2 border border-gray-300 rounded-md"

  />

  <input

  required

    type="text"

    name="city"

    placeholder="City"

    onChange={handleChange}

    className="w-full my-2 p-2 border border-gray-300 rounded-md"

  />

  <input

    type="text"

    name="postalCode"

    placeholder="Postal Code"

    onChange={handleChange}

    className="w-full my-2 p-2 border border-gray-300 rounded-md"

  />

  <input

  required

    type="text"

    name="phone"

    placeholder="Phone Number"

    onChange={handleChange}

    className="w-full my-2 p-2 border border-gray-300 rounded-md"

  />

  <select

  name="paymentMethod"

  value={paymentMethod}

  onChange={(e) => setPaymentMethod(e.target.value)}

  className="w-full my-2 p-2 border border-gray-300 rounded-md"

>

  <option value="">Select Payment Method</option>

  <option value="Card">Card</option>

  <option value="Cash">Cash</option>

  <option value="PayPal">PayPal</option>

</select>

            <button

              className="confirm-btn mt-4 p-3 bg-[#2B3A57] text-white rounded-md w-full cursor-pointer hover:bg-[#3A4C73]"

              onClick={handleConfirm}

            >

              Confirm

            </button>

          </div>

        </>

      ) : (

        <div className="invoice-section">

          <div className="bg-[#f7f7f7] p-5 mt-10 border border-dashed border-[#999] rounded-md" ref={invoiceRef}>

            <h2 className="text-xl font-extrabold text-black mb-5 text-left">Invoice</h2>

            <p><strong>Name:</strong> {shipping.fullName}</p>

            <p><strong>Address:</strong> {shipping.address}, {shipping.city} - {shipping.postalCode}</p>

            <p><strong>Phone:</strong> {shipping.phone}</p>

            <p className='mb-2'><strong>Payment Method:</strong> {paymentMethod}</p>

            <hr />

            <h3 className='mt-3 mb-1'><strong>Items:</strong></h3>

            <ul>

              {cartItems.map((item) => (

                <li key={item.id}>{item.title} × {item.quantity}</li>

              ))}

            </ul>

            <p className='mt-2 mb-2'><strong>Total Cost:</strong> Rs. {totalCost}.00</p>

            <p className='text-center'>Thank you for shopping with us!</p>

          </div>

          <div className="mt-5 flex justify-center gap-5">

            <button

              onClick={handleDownload}

              className="bg-[#2B3A57] text-white py-2 px-4 rounded hover:bg-[#3A4C73]"

            >

              Download Invoice

            </button>

            <button

              onClick={() => navigate("/about")}

              className="bg-[#2B3A57] text-white py-2 px-4 rounded hover:bg-[#3A4C73] ml-2"

            >

              Go Back to Home

            </button>

          </div>

        </div>

      )}

    </div>

  );

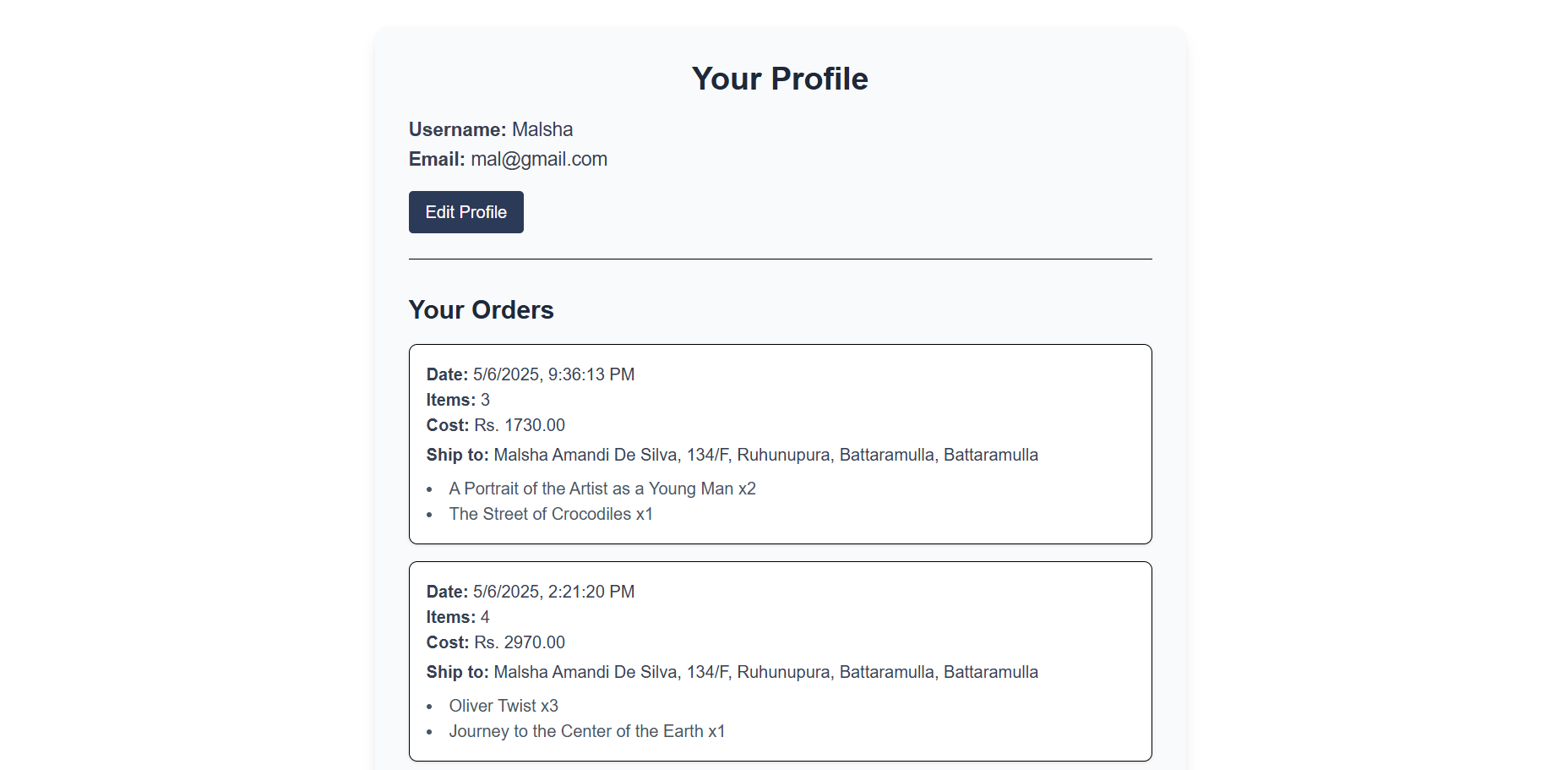
};

export default ShippingPage;

**1.5 User Profile**

**Profile View, Order History View**

* **Screenshot:**



* **Code:**

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

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

const ProfilePage = () => {

  const [user, setUser] = useState(null);

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

  const [loading, setLoading] = useState(true);

  const navigate = useNavigate();

  useEffect(() => {

    const fetchData = async () => {

      try {

        const userRes = await fetch('http://localhost:4000/api/user/data', { credentials: 'include' });

        if (userRes.status === 401) throw new Error('Unauthorized');

        const userData = await userRes.json();

        if (userData.success) {

          setUser(userData.user);

        } else {

          navigate('/login', { state: { redirectTo: '/profile' } });

        }

        const orderRes = await fetch('http://localhost:4000/api/orders/getorders', { credentials: 'include' });

        const orderData = await orderRes.json();

        if (orderData.success) {

          setOrders(orderData.orders);

        }

      } catch (err) {

        navigate('/login', { state: { redirectTo: '/profile' } });

      } finally {

        setLoading(false);

      }

    };

    fetchData();

  }, [navigate]);

  const handleLogout = () => {

    fetch('http://localhost:4000/api/auth/logout', { method: 'POST', credentials: 'include' })

      .then(() => navigate('/login'));

  };

  if (loading) return <p className="text-center mt-20">Loading...</p>;

  return (

    <div className="max-w-3xl mx-auto p-8 bg-gray-50 rounded-xl shadow-lg m-10">

      {user && (

        <div className="mb-8 border-b pb-6">

          <h2 className="text-3xl font-bold text-gray-800 mb-4 text-center">Your Profile</h2>

          <p className="text-lg text-gray-700"><strong>Username:</strong> {user.username}</p>

          <p className="text-lg text-gray-700"><strong>Email:</strong> {user.email}</p>

          <button

            onClick={() => navigate('/profile/edit')}

            className="mt-4 px-4 py-2 bg-[#2B3A57] text-white rounded hover:bg-[#3A4C73]"

          >

            Edit Profile

          </button>

        </div>

      )}

      <h2 className="text-2xl font-semibold text-gray-800 mb-4">Your Orders</h2>

      {orders.length === 0 ? (

        <p className="text-gray-600">No orders yet.</p>

      ) : (

        orders.map(o => (

          <div key={o.\_id} className="border p-4 mb-4 rounded-lg bg-white shadow-sm">

            <p className="text-gray-700"><strong>Date:</strong> {new Date(o.createdAt).toLocaleString()}</p>

            <p className=" text-gray-700 mb-1">

          <span className="font-semibold">Items:</span> {o.totalItems}

          <br />

          <span className=" font-semibold">Cost:</span> Rs. {o.totalCost}.00

        </p>

            <p className="text-gray-700">

              <strong>Ship to:</strong> {o.shipping.fullName}, {o.shipping.address}, {o.shipping.city}

            </p>

            <ul className="mt-2 list-disc list-inside text-gray-600">

              {o.items.map(i => (

                <li key={i.bookId}>{i.title} x{i.quantity}</li>

              ))}

            </ul>

          </div>

        ))

      )}

      <button

            onClick={handleLogout}

            className="mt-6 w-full p-3 bg-[#2B3A57] text-white rounded-md hover:bg-[#3A4C73]"

          >

            Logout

          </button>

    </div>

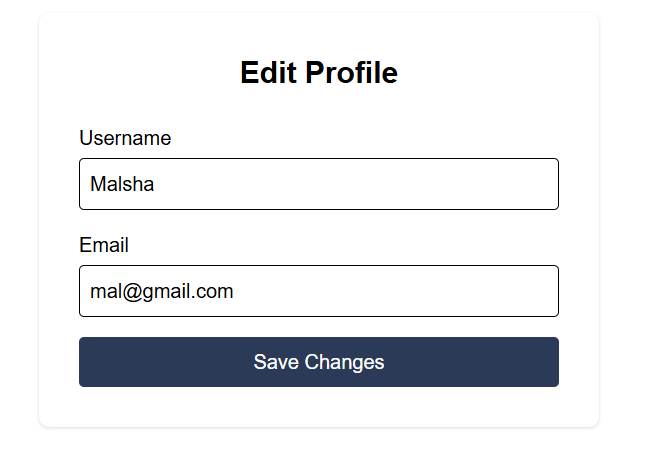
  );

};

export default ProfilePage;

**Edit Profile Interface**

* **Screenshot:**



A screenshot of a login form

AI-generated content may be incorrect.

* **Code:**

// src/pages/ProfileEdit.jsx

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

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

const ProfileEdit = () => {

  const [form, setForm] = useState({

    username: '',

    email: '',

    profileImage: null,

  });

  const [preview, setPreview] = useState(null);

  const [error, setError] = useState('');

  const [success, setSuccess] = useState('');

  const [loading, setLoading] = useState(true);

  const navigate = useNavigate();

  useEffect(() => {

    fetch('http://localhost:4000/api/user/data', { credentials: 'include' })

      .then(res => {

        if (res.status === 401) throw new Error();

        return res.json();

      })

      .then(data => {

        if (data.success) {

          setForm(f => ({ ...f, username: data.user.username, email: data.user.email }));

          setPreview(data.user.profileImage || null);

        } else {

          navigate('/login', { state: { redirectTo: '/profile/edit' } });

        }

      })

      .catch(() => navigate('/login', { state: { redirectTo: '/profile/edit' } }))

      .finally(() => setLoading(false));

  }, [navigate]);

  const handleChange = e => {

    const { name, value, files } = e.target;

    if (name === 'profileImage') {

      const file = files[0];

      setForm(f => ({ ...f, profileImage: file }));

      setPreview(URL.createObjectURL(file));

    } else {

      setForm(f => ({ ...f, [name]: value }));

    }

  };

  const handleSubmit = async e => {

    e.preventDefault();

    setError('');

    setSuccess('');

    const { username, email, profileImage } = form;

    if (!username || !email) {

      setError('Username and email are required');

      return;

    }

    const data = new FormData();

    data.append('username', username);

    data.append('email', email);

    if (profileImage) data.append('avatar', profileImage);

    try {

      const res = await fetch('http://localhost:4000/api/user/update', {

        method: 'PUT',

        credentials: 'include',

        body: data,

      });

      const result = await res.json();

      if (result.success) {

        setSuccess('Profile updated successfully!');

        setTimeout(() => navigate('/profile'), 2000);

      } else {

        setError(result.message || 'Update failed');

      }

    } catch {

      setError('Server error, please try again');

    }

  };

  if (loading) return <p className="text-center mt-20">Loading...</p>;

  return (

    <div className="max-w-md mx-auto p-8 bg-white rounded-lg shadow m-10">

      <h2 className="text-2xl font-bold mb-6 text-center">Edit Profile</h2>

      {error && <p className="text-red-600 mb-4">{error}</p>}

      {success && <p className="text-green-600 mb-4">{success}</p>}

      <form onSubmit={handleSubmit} className="space-y-4">

        <div>

          <label className="block mb-1 font-medium">Username</label>

          <input

            type="text"

            name="username"

            value={form.username}

            onChange={handleChange}

            className="w-full p-2 border rounded"

          />

        </div>

        <div>

          <label className="block mb-1 font-medium">Email</label>

          <input

            type="email"

            name="email"

            value={form.email}

            onChange={handleChange}

            className="w-full p-2 border rounded"

          />

        </div>

        <button

          type="submit"

          className="w-full py-2 bg-[#2B3A57] text-white rounded hover:bg-[#3A4C73]"

        >

          Save Changes

        </button>

      </form>

    </div>

  );

};

export default ProfileEdit;