WT LAB Assignment 6: Grocery shop App

Name : Aditya Sakhare

Roll No: 26 TYCS-D Batch 2

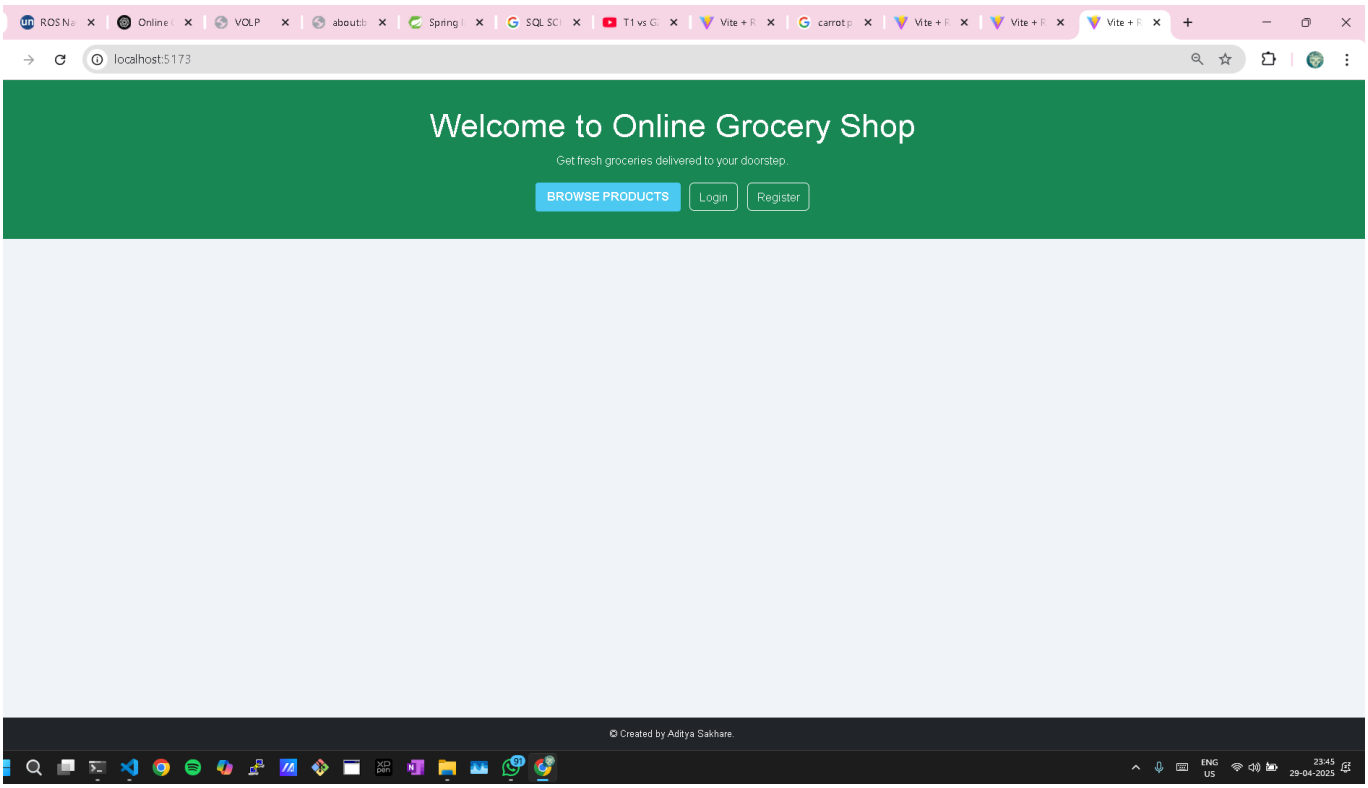
Design:

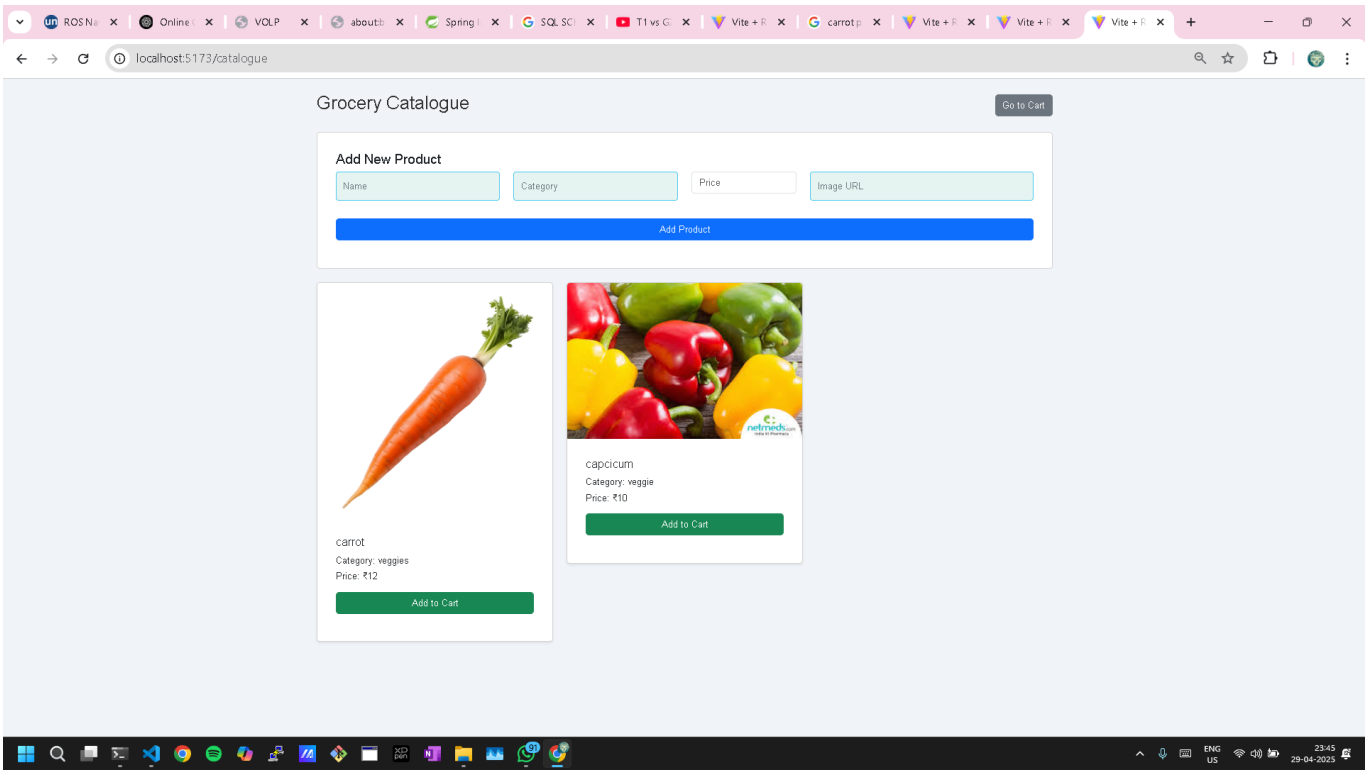
Tables:

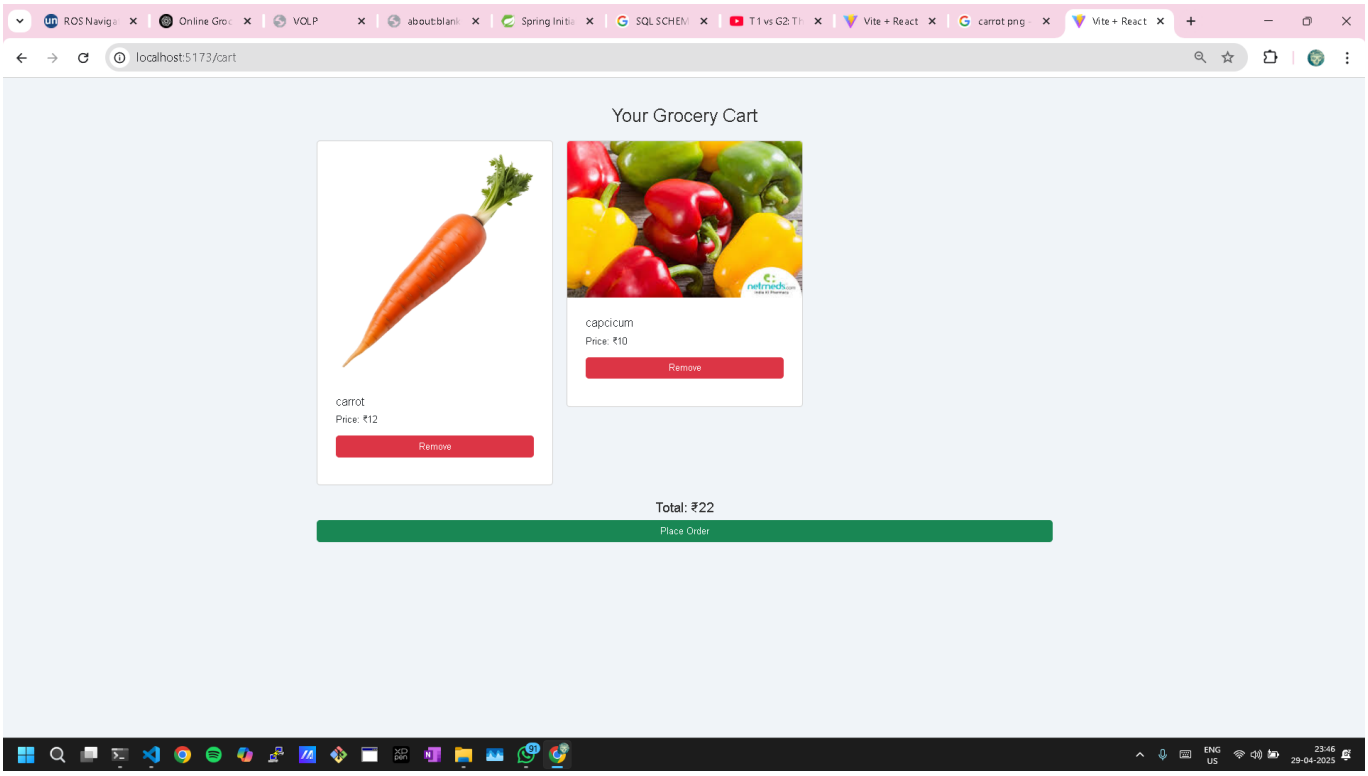
```
mysql> use ass6;
Database changed
mysql>
mysql>
mysql> show tables;
+----------------+
| Tables_in_ass6 |
+----------------+
| order_items    |
| orders         |
| orders_seq     |
| products       |
| user           |
| user_seq       |
+----------------+
6 rows in set (0.00 sec)

mysql> select * from user;
+----+-------+--------+----------+
| id | email | name   | password |
+----+-------+--------+----------+
|  1 | a@a   | aditya | a        |
+----+-------+--------+----------+
1 row in set (0.01 sec)

mysql> select * from products;
+----+----------+-------------------------------------------------------------------
-------------------------------------------------------------+----------+-------+
| id | category | image
                                                | name     | price |
+----+----------+-------------------------------------------------------------------
-------------------------------------------------------------+----------+-------+
|  1 | veggies  |
https://static.vecteezy.com/system/resources/previews/044/764/062/non_2x/fresh-
```

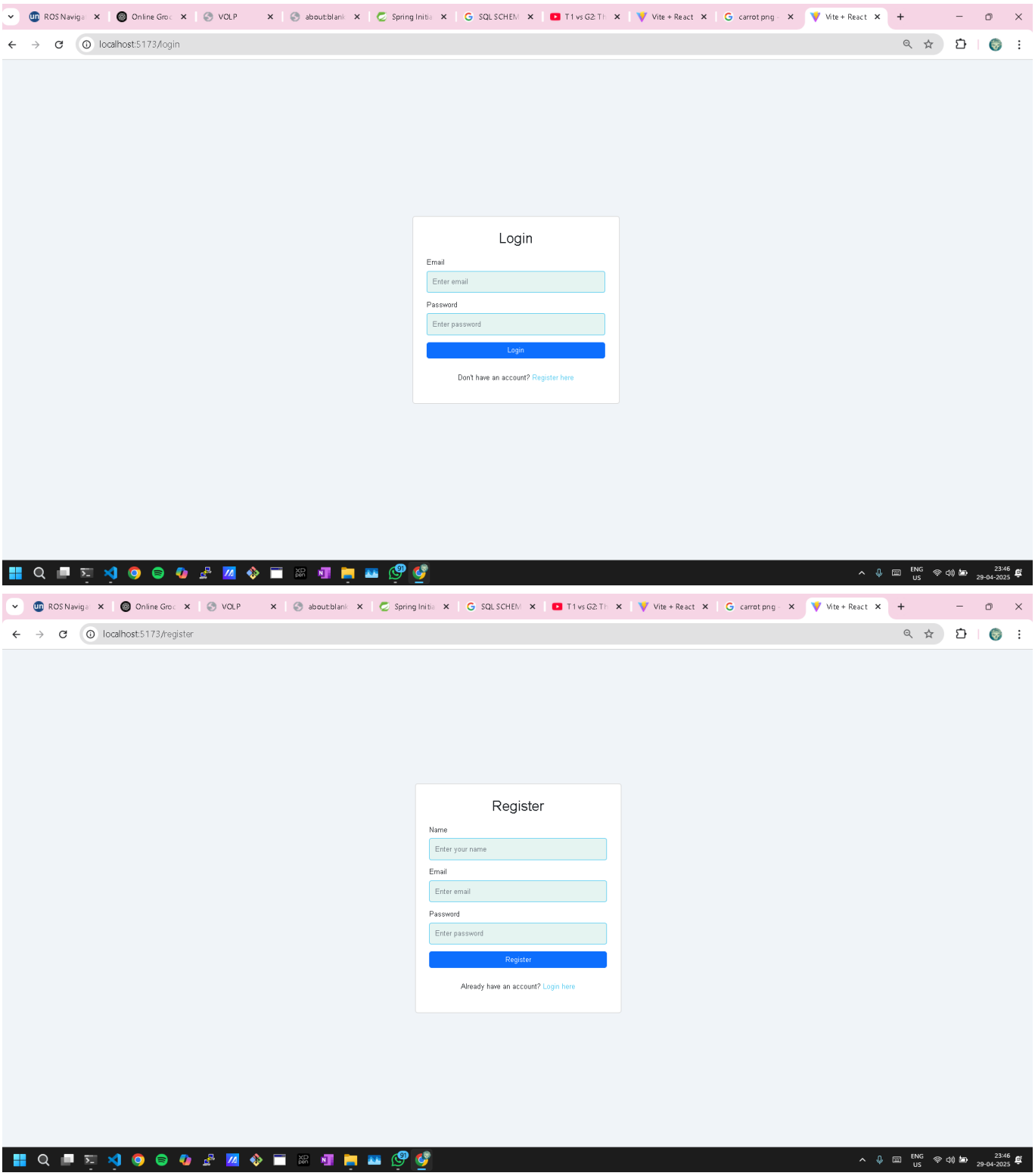
```
carrots-bundle-with-transparency-free-png.png | carrot   |    12 |
|  2 | veggie   | https://encrypted-tbn0.gstatic.com/images?
q=tbn:ANd9GcQZoSst2ipInsKN43I_uVJ3xdR1gC-vz7WOnw&s                       |
capcicum |    10 |
+----+----------+-------------------------------------------------------------------------
-----------------------------------------------------------+----------+-------+
2 rows in set (0.00 sec)

mysql> select * from orders;
+----+----------------------------+---------+
| id | order_date                 | user_id |
+----+----------------------------+---------+
|  1 | 2025-04-29 16:09:51.521556 |       1 |
|  2 | 2025-04-29 17:04:33.161147 |       1 |
+----+----------------------------+---------+
2 rows in set (0.01 sec)

mysql> select * from order_items;
+----+----------+----------+------------+
| id | quantity | order_id | product_id |
+----+----------+----------+------------+
|  1 |        1 |        1 |          1 |
|  2 |        1 |        2 |          2 |
|  3 |        1 |        2 |          1 |
+----+----------+----------+------------+
3 rows in set (0.00 sec)

mysql>
```

Screenshots:


localhost:5173
Welcome to Online Grocery Shop
Get fresh groceries delivered to your doorstep.
BROWSE PRODUCTS
Login
Register
© Created by Aditya Sakhare.


localhost:5173/catalogue
Grocery Catalogue
Go to Cart
Add New Product
Name
Category
Price
Image URL
Add Product
carrot
Category: veggies
Price: ₹12
Add to Cart
capcicum
Category: veggie
Price: ₹10
Add to Cart


localhost:5173/cart
Your Grocery Cart
carrot
Price: ₹12
Remove
capcicum
Price: ₹10
Remove
Total: ₹22
Place Order



Codes:

Frontend (React + Vite):

src/App.jsx

```
import React from 'react';
import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';
import HomePage from './components/HomePage';
import LoginPage from './components/LoginPage';
```

```
import CataloguePage from './components/CataloguePage';
import RegistrationPage from './components/RegistrationPage';
import 'bootstrap/dist/css/bootstrap.min.css';
import './style.css'
import CartPage from './components/CartPage';
function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" exact element={ <HomePage/> } />
        <Route path="/login" element={<LoginPage/>} />
        <Route path="/catalogue" element={<CataloguePage/>} />
        <Route path="/register" element={<RegistrationPage/>} />
        <Route path='/cart' element={<CartPage />} />
      </Routes>
    </Router>
  );
}

export default App;
```

src/componenets/

```
import React, { useState, useEffect } from 'react';
import { useNavigate } from 'react-router-dom';
import axios from 'axios';

function CartPage() {
  const [cart, setCart] = useState([]);
  const [total, setTotal] = useState(0);
  const navigate = useNavigate();

  useEffect(() => {
    const storedCart = JSON.parse(localStorage.getItem('cart')) || [];
    setCart(storedCart);
    calculateTotal(storedCart);
  }, []);

  const calculateTotal = (items) => {
    const sum = items.reduce((acc, item) => acc + item.price, 0);
    setTotal(sum);
  };

  const removeItem = (id) => {
    const updated = cart.filter(item => item.id !== id);
    setCart(updated);
    localStorage.setItem('cart', JSON.stringify(updated));
    calculateTotal(updated);
  };

  const placeOrder = () => {
```

```
    const payload = {
      userId: 1, // Replace with actual user ID
      items: cart.map(p => ({ productId: p.id, quantity: 1 }))
    };
    axios.post('http://localhost:8080/api/orders', payload)
      .then(() => {
        alert('Order placed!');
        setCart([]);
        localStorage.removeItem('cart');
        navigate('/catalogue');
      })
      .catch(err => {
        console.error('Order error:', err);
        alert('Failed to place order.');
      });
  };

  return (
    <div className="container py-5">
      <h2 className="text-center mb-4">Your Grocery Cart</h2>
      <div className="row">
        {cart.length === 0 ? (
          <p>Your cart is empty.</p>
        ) : (
          cart.map(item => (
            <div className="col-md-4" key={item.id}>
              <div className="card mb-4">
                <img src={item.image} alt={item.name} className="card-img-top" />
                <div className="card-body">
                  <h5 className="card-title">{item.name}</h5>
                  <p>Price: ₹{item.price}</p>
                  <button className="btn btn-danger" onClick={() =>
removeItem(item.id)}>Remove</button>
                </div>
              </div>
            </div>
          ))
        )}
      </div>
      <div className="text-center">
        <h4>Total: ₹{total}</h4>
        <button className="btn btn-success" onClick={placeOrder}>Place
Order</button>
      </div>
    </div>
  );
}

export default CartPage;
```

```
import React, { useState, useEffect } from 'react';
import { Link } from 'react-router-dom';
import axios from 'axios';

function CataloguePage() {
  const [name, setName] = useState('');
  const [category, setCategory] = useState('');
  const [price, setPrice] = useState('');
  const [image, setImage] = useState('');
  const [products, setProducts] = useState([]);

  useEffect(() => {
    axios.get('http://localhost:8080/api/products')
      .then(response => setProducts(response.data))
      .catch(error => console.error('Error fetching products', error));
  }, []);

  const handleAddProduct = (e) => {
    e.preventDefault();
    const newProduct = {
      name,
      category,
      price: parseFloat(price),
      image: image || "https://via.placeholder.com/150"
    };

    axios.post('http://localhost:8080/api/products', newProduct)
      .then(response => {
        setProducts([...products, response.data]);
        setName('');
        setCategory('');
        setPrice('');
        setImage('');
        alert('Product added!');
      })
      .catch(error => {
        console.error('Error adding product', error);
        alert('Failed to add product');
      });
  };

  const addToCart = (product) => {
    const cart = JSON.parse(localStorage.getItem('cart')) || [];
    cart.push(product);
    localStorage.setItem('cart', JSON.stringify(cart));
    alert('Product added to cart!');
  };

  return (
    <div className="container py-4">
      <div className="d-flex justify-content-between align-items-center mb-4">
        <h2>Grocery Catalogue</h2>
        <Link to="/cart" className="btn btn-secondary">Go to Cart</Link>
```

```
      </div>

      <div className="card mb-4">
        <div className="card-body">
          <h4>Add New Product</h4>
          <form onSubmit={handleAddProduct}>
            <div className="row">
              <div className="col-md-3 mb-3">
                <input
                  type="text"
                  className="form-control"
                  placeholder="Name"
                  value={name}
                  onChange={e => setName(e.target.value)}
                  required
                />
              </div>
              <div className="col-md-3 mb-3">
                <input
                  type="text"
                  className="form-control"
                  placeholder="Category"
                  value={category}
                  onChange={e => setCategory(e.target.value)}
                  required
                />
              </div>
              <div className="col-md-2 mb-3">
                <input
                  type="number"
                  className="form-control"
                  placeholder="Price"
                  value={price}
                  onChange={e => setPrice(e.target.value)}
                  required
                />
              </div>
              <div className="col-md-4 mb-3">
                <input
                  type="text"
                  className="form-control"
                  placeholder="Image URL"
                  value={image}
                  onChange={e => setImage(e.target.value)}
                />
              </div>
            </div>
            <button type="submit" className="btn btn-primary">Add Product</button>
          </form>
        </div>
      </div>

      <div className="row">
        {products.map(product => (
```

```
          <div className="col-md-4" key={product.id}>
            <div className="card mb-4 shadow-sm">
              <img src={product.image} alt={product.name} className="card-img-top"
/>
              <div className="card-body">
                <h5>{product.name}</h5>
                <p className="mb-1">Category: {product.category}</p>
                <p>Price: ₹{product.price}</p>
                <button className="btn btn-success w-100" onClick={() =>
addToCart(product)}>
                  Add to Cart
                </button>
              </div>
            </div>
          </div>
        ))}
      </div>
    </div>
  );
}

export default CataloguePage;
```

```
import React from 'react';
import { Link } from 'react-router-dom';
import 'bootstrap/dist/css/bootstrap.min.css';

function HomePage() {
  return (
    <div className="main-container">
      {/* Hero Section */}
      <section className="bg-success text-white text-center py-5">
        <div className="container">
          <h1 className="display-4">Welcome to Online Grocery Shop</h1>
          <p className="lead mb-4">Get fresh groceries delivered to your doorstep.
</p>
          <div>
            <Link to="/catalogue" className="btn btn-light btn-lg mx-2">Browse
Products</Link>
            <Link to="/login" className="btn btn-outline-light btn-lg mx-
2">Login</Link>
            <Link to="/register" className="btn btn-outline-light btn-lg mx-
2">Register</Link>
          </div>
        </div>
      </section>

      {/* Footer */}
```

```
      <footer className="bg-dark text-white text-center py-3">
        <div className="container">
          <p className="mb-0">&copy; Created by Aditya Sakhare.</p>
        </div>
      </footer>
    </div>
  );
}

export default HomePage;
```

```
import React, { useState } from 'react';
import API from '../api/api';
import { useNavigate } from 'react-router-dom';
import 'bootstrap/dist/css/bootstrap.min.css';

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

  const handleSubmit = (e) => {
    e.preventDefault();
    API.post('auth/login', { email, password })
      .then(res => {
        alert('Login successful!');
        navigate('/catalogue');
      })
      .catch(err => {
        console.error('Login error:', err.response?.data || err.message);
        alert('Invalid credentials.');
      });
  };

  return (
    <div className="container d-flex justify-content-center align-items-center"
style={{ height: '100vh' }}>
      <div className="card" style={{ width: '30rem' }}>
        <div className="card-body">
          <h2 className="card-title text-center mb-4">Login</h2>
          <form onSubmit={handleSubmit}>
            <div className="mb-3">
              <label className="form-label">Email</label>
              <input
                type="email"
                className="form-control"
                placeholder="Enter email"
                value={email}
                onChange={e => setEmail(e.target.value)}
                required
```

```
              />
            </div>
            <div className="mb-3">
              <label className="form-label">Password</label>
              <input
                type="password"
                className="form-control"
                placeholder="Enter password"
                value={password}
                onChange={e => setPassword(e.target.value)}
                required
              />
            </div>
            <button className="btn btn-primary w-100" type="submit">Login</button>
          </form>
          <div className="text-center mt-3">
            <p>Don't have an account? <a href="/register">Register here</a></p>
          </div>
        </div>
      </div>
    </div>
  );
}

export default LoginPage;
```

```
import React, { useState } from 'react';
import API from '../api/api';
import { useNavigate } from 'react-router-dom';
import 'bootstrap/dist/css/bootstrap.min.css';

function RegistrationPage() {
  const [name, setName] = useState('');
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const navigate = useNavigate();

  const handleSubmit = (e) => {
    e.preventDefault();
    API.post('auth/register', { name, email, password })
      .then(() => {
        alert('Registration successful!');
        navigate('/login');
      })
      .catch(error => {
        console.error('Registration error:', error);
        alert('Registration failed.');
      });
  };
```

```
  return (
    <div className="container d-flex justify-content-center align-items-center"
style={{ height: '100vh' }}>
      <div className="card" style={{ width: '30rem' }}>
        <div className="card-body">
          <h2 className="card-title text-center mb-4">Register</h2>
          <form onSubmit={handleSubmit}>
            <div className="mb-3">
              <label className="form-label">Name</label>
              <input
                type="text"
                className="form-control"
                placeholder="Enter your name"
                value={name}
                onChange={e => setName(e.target.value)}
                required
              />
            </div>
            <div className="mb-3">
              <label className="form-label">Email</label>
              <input
                type="email"
                className="form-control"
                placeholder="Enter email"
                value={email}
                onChange={e => setEmail(e.target.value)}
                required
              />
            </div>
            <div className="mb-3">
              <label className="form-label">Password</label>
              <input
                type="password"
                className="form-control"
                placeholder="Enter password"
                value={password}
                onChange={e => setPassword(e.target.value)}
                required
              />
            </div>
            <button className="btn btn-primary w-100"
type="submit">Register</button>
          </form>
          <div className="text-center mt-3">
            <p>Already have an account? <a href="/login">Login here</a></p>
          </div>
        </div>
      </div>
    </div>
  );
}

export default RegistrationPage;
```

src/api/

```
import axios from 'axios';

const API_URL = axios.create({
    baseURL: 'http://localhost:8080/api'
});


export default API_URL;
```

Backend:

backend\backend\src\main\java\com\result\backend\config

```
package com.result.backend.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.CorsRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
public class WebConfig implements WebMvcConfigurer {
    @Override
    public void addCorsMappings(CorsRegistry registry) {
        // Allow all origins, methods, and headers
        registry.addMapping("/**")
                .allowedOrigins("*")  // Allow all origins
                .allowedMethods("*")  // Allow all HTTP methods
                .allowedHeaders("*");  // Allow all headers
    }
}
```

backend\backend\src\main\java\com\result\backend\controller

```
package com.grocery.backend.controller;
```

```
import com.grocery.backend.model.User;
import com.grocery.backend.repository.UserRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

@RestController
@RequestMapping("/api/auth")
@CrossOrigin(origins = "*")
public class AuthController {

    @Autowired
    private UserRepository userRepository;

    @PostMapping("/register")
    public ResponseEntity<?> registerUser(@RequestBody User user) {
        if (userRepository.findByEmail(user.getEmail()) != null) {
            return ResponseEntity.badRequest().body("Email already exists");
        }
        userRepository.save(user);
        return ResponseEntity.ok("User registered successfully");
    }

    @PostMapping("/login")
    public ResponseEntity<?> loginUser(@RequestBody User loginData) {
        User user = userRepository.findByEmail(loginData.getEmail());
        if (user != null && user.getPassword().equals(loginData.getPassword())) {
            return ResponseEntity.ok(user);
        }
        return ResponseEntity.status(401).body("Invalid credentials");
    }
}
```

```
package com.grocery.backend.controller;

import com.grocery.backend.dto.OrderDTO;
import com.grocery.backend.dto.OrderItemDTO;
import com.grocery.backend.model.*;
import com.grocery.backend.repository.*;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

@RestController
@RequestMapping("/api/orders")
@CrossOrigin(origins = "*")
public class OrderController {
```

```
    @Autowired private UserRepository userRepository;
    @Autowired private ProductRepository productRepository;
    @Autowired private OrderRepository orderRepository;

    @PostMapping
    public String placeOrder(@RequestBody OrderDTO orderDTO) {
        User user = userRepository.findById(orderDTO.getUserId()).orElse(null);
        if (user == null) return "User not found";

        Order order = new Order();
        order.setUser(user);
        order.setOrderDate(LocalDateTime.now());

        List<OrderItem> orderItems = new ArrayList<>();
        for (OrderItemDTO itemDTO : orderDTO.getItems()) {
            Product product =
productRepository.findById(itemDTO.getProductId()).orElse(null);
            if (product == null) continue;

            OrderItem orderItem = new OrderItem();
            orderItem.setProduct(product);
            orderItem.setQuantity(itemDTO.getQuantity());
            orderItem.setOrder(order);
            orderItems.add(orderItem);
        }

        order.setItems(orderItems);
        orderRepository.save(order);
        return "Order placed successfully!";
    }
}
```

```
package com.grocery.backend.controller;

import com.grocery.backend.model.Product;
import com.grocery.backend.repository.ProductRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/api/products")
@CrossOrigin(origins = "*")
public class ProductController {

    @Autowired
    private ProductRepository productRepository;
```

```
    @GetMapping
    public List<Product> getAllProducts() {
        return productRepository.findAll();
    }

    @PostMapping
    public Product addProduct(@RequestBody Product product) {
        return productRepository.save(product);
    }
}
```

backend\backend\src\main\java\com\result\backend\model

```
package com.grocery.backend.model;
import jakarta.persistence.*;
import lombok.*;
import java.time.LocalDateTime;
import java.util.List;

@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
@Table(name = "orders")
public class Order {
    @Id @GeneratedValue private Long id;

    @ManyToOne private User user;
    private LocalDateTime orderDate;

    @OneToMany(mappedBy = "order", cascade = CascadeType.ALL)
    private List<OrderItem> items;
}
```

```
package com.grocery.backend.model;

import jakarta.persistence.*;
import lombok.*;

@Entity
@Table(name = "order_items")
@Data
@NoArgsConstructor
@AllArgsConstructor
```

```
public class OrderItem {

    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private int quantity;

    @ManyToOne
    @JoinColumn(name = "product_id")
    private Product product;

    @ManyToOne
    @JoinColumn(name = "order_id")
    private Order order;
}
```

```
package com.grocery.backend.model;

import jakarta.persistence.*;
import lombok.*;

@Entity
@Table(name = "products")
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Product {

    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String name;
    private String category;
    private Double price;
    @Column(length = 1000)  // Increase length from default 255
    private String image;
}
```

```
package com.grocery.backend.model;


import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
```

```
import jakarta.persistence.Id;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;




@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class User {
    @Id @GeneratedValue private Long id;
    private String name;
    private String email;
    private String password;
}
```

backend\backend\src\main\java\com\result\backend\repository

```
package com.grocery.backend.repository;

import com.grocery.backend.model.Order;
import org.springframework.data.jpa.repository.JpaRepository;
import java.util.List;



public interface OrderRepository extends JpaRepository<Order, Long> {
    // Custom query methods can be defined here if needed
    List<Order> findByUserId(Long userId);

}
```

```
package com.grocery.backend.repository;

import com.grocery.backend.model.Product;
import org.springframework.data.jpa.repository.JpaRepository;

public interface ProductRepository extends JpaRepository<Product, Long> {}
```

```
package com.grocery.backend.repository;

import com.grocery.backend.model.User;
import org.springframework.data.jpa.repository.JpaRepository;


public interface UserRepository extends JpaRepository<User, Long> {
    User findByEmail(String email);
}
```

backend\backend\src\main\java\com\result\backend\service

```
package com.grocery.backend.service;

import com.grocery.backend.dto.OrderDTO;
import com.grocery.backend.dto.OrderItemDTO;
import com.grocery.backend.model.Order;
import com.grocery.backend.model.OrderItem;
import com.grocery.backend.model.Product;
import com.grocery.backend.model.User;
import com.grocery.backend.repository.OrderRepository;
import com.grocery.backend.repository.ProductRepository;
import com.grocery.backend.repository.UserRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;


@Service
public class OrderService {
    @Autowired private UserRepository userRepo;
    @Autowired private ProductRepository productRepo;
    @Autowired private OrderRepository orderRepo;

    public Order placeOrder(OrderDTO dto) {
        User user = userRepo.findById(dto.getUserId()).orElseThrow();
        Order order = new Order(null, user, LocalDateTime.now(), new ArrayList<>
());
        for (OrderItemDTO item : dto.getItems()) {
            Product p = productRepo.findById(item.getProductId()).orElseThrow();
            order.getItems().add(new OrderItem(null, item.getQuantity(), p,
order));
        }
        return orderRepo.save(order);
    }
```

```
    public List<Order> getOrdersByUser(Long userId) {
        return orderRepo.findByUserId(userId);
    }
}
```