E-COMMERCE WEBSITE

A Project Report for Industrial Training and Internship

submitted by

In the partial fulfillment of the award of the degree of

ABDUL SUHEL
AMRITMOY SAMANTA
RAJASHRI CHANDA
SILPA MONDAL
SONALI DAS

BACHELOR OF COMPUTER APPLICATION

in the

BCA

of

BENGAL SCHOOL OF TECHNOLOGY & MANAGEMENT

Ardent Computech Pvt. Ltd.

CERTIFICATE FROM SUPERVISOR

This is to certify that **ABDUL SUHEL, AMRITMOY SAMANTA , RAJASHRI CHANDA, SILPA MONDAL, SONALI DAS** that **232661010001, 232661010003, 232661010043, 232661010058, 232661010061** have completed the project titled **E-COMMERCE WEBSITE** under my supervision during the period from **15/07/2025** to **06/08/2025** which is in partial fulfillment of requirements for the award of the **BCA** degree of **Bengal School Of Technology & Management.**

Signature of the Supervisor.

Date: 12/08/2025

Name of the project supervisor: **Subhojit Santra**

BONAFIDE CERTIFICATE

Certified that this project work was carried out under my supervision

E- COMMERCE WEBSITE is the bonafide work of

Name of the student: ABDUL SUHEL SIGN: Abdul Suhel

Name of the student: AMRITMOY SAMANTA SIGN: Amritmoy Samanta

Name of the student: RAJASHRI CHANDA SIGN: Rajashri Chanda

Name of the student: SILPA MONDAL SIGN: Silpa Mondal

Name of the student: SONALI DAS SIGN: Sonali Das

SIGNATURE

Name:

PROJECT MENTOR

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.

CONTENT PAGE

1. COMPANY PROFILE

ARDENT (Ardent Computech Pvt. Ltd.), formerly known as Ardent Computech Private Limited, is an ISO 9001:2015 certified Software Development and Training Company based in India. Operating independently since 2003, the organization has recently undergone a strategic merger with ARDENT Technologies, enhancing its global outreach and service offerings.

ARDENT Technologies

ARDENT Technologies delivers high-end IT services across the UK, USA, Canada, and India. Its core competencies lie in the development of customized application software, 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 institution upholds strict quality standards under ISO 9001:2015 certification and is dedicated to bridging the gap between academic knowledge and industry skills through innovative training programs.

2. INTRODUCTION

SMART MART is a modern e-commerce platform developed using the latest web technologies. The frontend is built with React.js for a smooth and dynamic user experience, while the backend runs on Express.js, ensuring fast and scalable server-side operations. We use MongoDB to efficiently manage user data, orders, and product information.

With SMART MART, customers can easily browse a wide range of products, place orders in just a few clicks, and have them delivered right to their doorstep — quickly, reliably, and hassle-free.

2A. OBJECTIVE

The objective of building this e-commerce website using the MERN stack (MongoDB, Express.js, React, Node.js) is to create a dynamic, user-friendly web application that allows users to purchase products. Key goals include :

1. **User authentication —** secure sign up, sign in, and login.
2. **Category & product browsing —** users can choose product categories and view items.
3. **Order system —** users can add products to a cart and place orders.
4. **Payment options —** support Cash on Delivery and online payments**.**
5. **Delivery —** deliver ordered products to the user's location.
6. **Real-time notifications** — show success/failure notifications (e.g., order placed, payment success).

2B. UNIQUE FEATURES

1. Customizable Products – Allow users to personalize products with colors, texts, or features before buying.

2. First Payment - Secure and seamless payment process with multiple options and instant confirmation.

3. Filter - Advanced multi-criteria filtering for quick and precise product search.

4. Wishlist & Price Drop Alerts – Notify users when their saved items go on sale or are back in stock.

3. SYSTEM ANALYSIS

3A.IDENTIFICATION OF NEED :-

System analysis is a crucial phase in the development of our SMART MART e-commerce application. It helps in understanding user expectations, defining system requirements, and ensuring the platform meets both business goals and customer needs. By analyzing the requirements at an early stage, we can avoid costly changes later and ensure the app is user-friendly, scalable, and efficient.

Key Needs Identified :

1. Accessibility to purchasing :

The system should be designed so that all users, regardless of technical knowledge or device limitations, can easily access information, resources, and tutorials related to using SMART MART. This ensures inclusivity and a better onboarding experience.

2. Online shopping platform development :

The platform development process should adopt an iterative approach, continuously improving features based on user feedback. Engaging learning resources, such as interactive guides or tips, help users understand and use SMART MART effectively.

3. Efficient product delivery :

If training materials or seller onboarding courses are provided, they must be delivered quickly, organized logically, and be easy to follow. This ensures both sellers and customers can maximize the benefits of SMART MART with minimal delay.

3B.Feasibility Study : –

The feasibility study of Smart Mart shows that the project is practical and achievable. From a technical point of view, the MERN stack (MongoDB, Express.js, React.js, Node.js) provides a modern, scalable, and secure foundation for development, and the required tools and expertise are already available within the team. Operationally, the app will meet customer and seller needs through a user-friendly interface, secure payment gateway, and efficient product search and filtering, ensuring smooth day-to-day operations. Economically, the estimated development and maintenance costs are within budget limits, and revenue from product sales, advertisements, and premium seller accounts will provide a good return on investment. With strong technical support, clear operational goals, and a sustainable financial plan, SMART MART is considered highly feasible for successful implementation.

3C. Workflow :–

For the development of SMART MART, we are using a hybrid approach that combines the Waterfall Model and Iterative Model. The Waterfall approach provides a clear, sequential process for development, while the Iterative approach allows continuous refinement and improvement of features like product categories (mobiles, laptops, watches, headphones) based on user feedback.

Stages in the Workflow

1. Requirement Gathering and Analysis :-

Identify customer and seller needs, such as easy navigation, secure payments, and category-based filtering.

Research market trends and competitor features.

2. System and software Design :-

Plan database for product listings, orders, and user accounts.

Create UI/UX designs for a smooth shopping experience.

3. Implementation and Integration :-

Develop core features like product display, cart, and checkout. Integrate payment gateways and secure login systems.

4. Testing :-

Test performance, usability, and security of SMART MART.

Fix bugs before deployment.

5. Deployment :-

Launch the platform for public use.

6. Maintenance :-

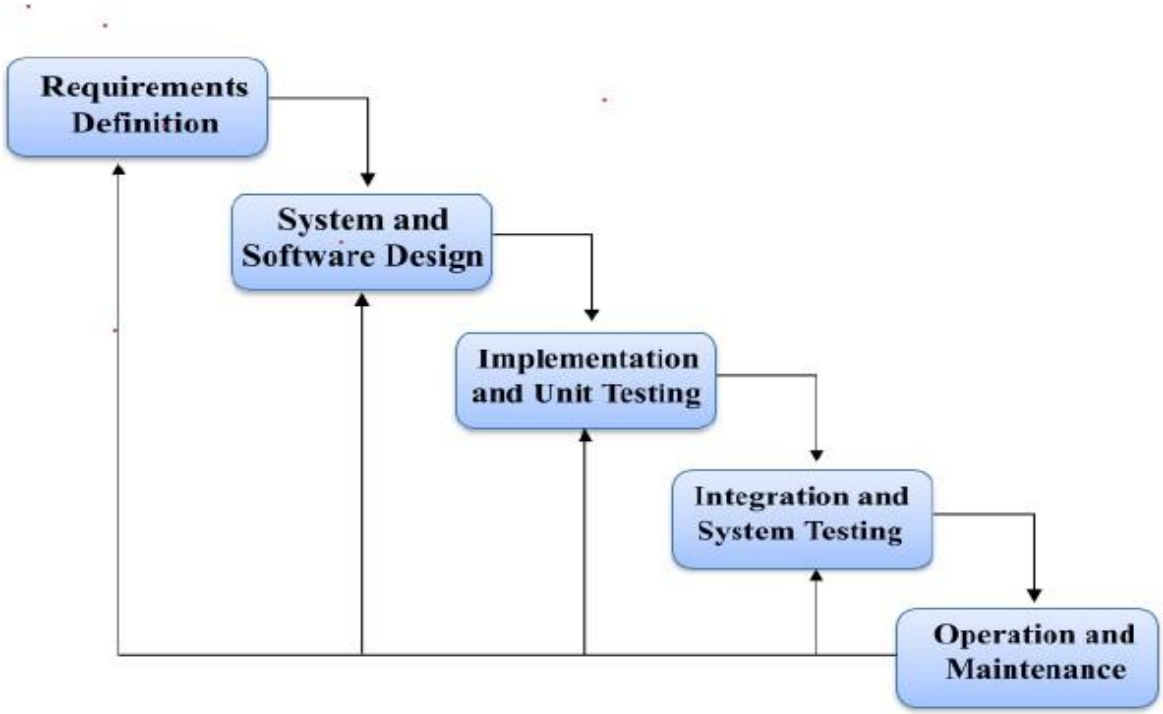
Update categories, add new features, and fix issues regularly.

Advantages :-

- **Flexibility –** Changes can be made based on feedback without starting from scratch.
- **Early Delivery –** Working features can be delivered in stages, allowing faster market entry.
- **Risk Management** – Problems can be detected and resolved earlier in the process.

Disadvantages :-

- **Increased Complexity –** Managing both structured and iterative processes requires strong coordination.
- **Potential for Scope Creep –** Adding features during iteration can delay final delivery.
- **Research Intensive** – Requires constant analysis of user feedback and market trends.



▪ **Applications:**

The Iterative Waterfall Model is suitable for projects with evolving or unclear requirements. It is commonly used in software development projects where regular feedback and refinement are essential.

Additionally, it is applicable in scenarios where partial system delivery is beneficial, allowing stakeholders to assess progress and make adjustments.

3D. Study of the System

1. **Home Page :-**
    a. User View: Displays featured products, promotions, and easy navigation to product categories.
    b. Admin View: Dashboard overview showing recent activities, sales stats, and quick access to management tools.
2. **Shop Page :-**
    a. User Access: Browse and filter products by categories such as mobiles, laptops, watches, and headphones.
    b. Admin Access: Add new products, update existing listings, and organize product categories.
3. **About Us Page :-**
    a. Accessible to all users, this page provides information about SMART MART's mission, vision, and company background.
4. **Contact Page :-**
    a. User Access: Submit inquiries, feedback, or support requests through a contact form.
    b. Admin Access: Receive, review, and respond to user messages and support tickets.
5. **Cart (Shopping Cart) Page :-**
    a. User Access: View selected products, modify quantities, apply discounts, and proceed to checkout.
    b. Admin Access: Monitor current orders, update order statuses, and manage payment processing.
6. **Login / Sign In Page :-**
    a. User Login: Authenticate to access personal accounts, view order history, and manage preferences.
    b. Admin Login: Authenticate to access the admin panel for product, user, and order management.

3E. INPUT AND OUTPUT

Input:

The user accesses the login page and enters their registered email address along with the corresponding password.

Output:

The system authenticates the credentials and allows the user to browse the full range of available products on SMART MART.

3F.SOFTWARE REQUIREMENT SPECIFICATIONS

Software Requirements Specification provides an overview of the entire project. It is a description of a software system to be developed, laying out functional and non-functional requirements. The software requirements specification document enlists enough necessary requirements that are required for the project development. To derive the requirements we need to have a clear and thorough understanding of the project to be developed. This is prepared after detailed communication with the project team and the customer.

The developer is responsible for:

● Developing the system, which meets the SRS and solves all the requirements of the system.

● Demonstrating the system and installing the system at the client's location after acceptance

testing is successful.

● Submitting the required user manual describing the system interfaces to work on it and also the documents of the system.

Functional Requirements:

A. User Registration and Authentication:

1. Users should be able to create accounts securely.
2. The system should authenticate users and manage login sessions.

B. Browse and Search:

1. Users should be able to browse and search for products.

C. Courses Display:

1. Each customer should have detailed and up-to-date items with prices.
2. Users should be able to view products.

Hardware Requirements:

1 . Computer has Intel I3 Processor

2 . 8 GB RAM

3. SSD-ROM Drive

Software Requirements:

1. Windows 11 OS
2. Visual Studio Code
3. Mongo DB Atlas

4.SYSTEM DESIGN

4A. DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated.

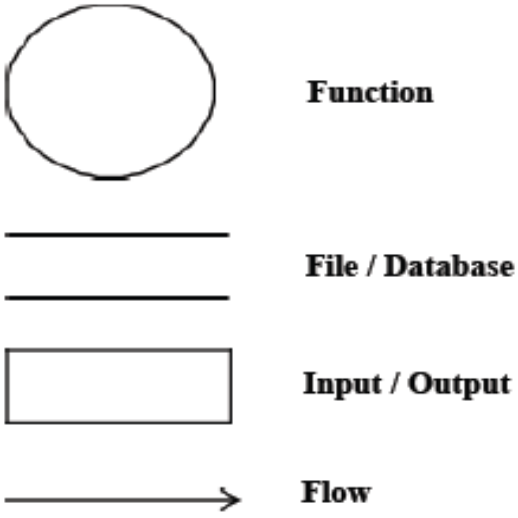
DFD can also be used for the visualization of data processing (structured design). A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of the process or information about whether processes will operate in sequence or in parallel (which is shown on a flowchart).

This context-level DFD is next "exploded", to produce a Level 1 DFD that shows some of the detail of the system being modeled. The Level 1 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present for the system to do its job and shows the flow of data between the various parts of the system.

Data flow diagrams are one of the three essential perspectives of the structured- systems analysis and design method SSADM. The sponsor of a project and the end users will need to be briefed and consulted throughout all stages of a system's evolution. With a data flow diagram, users can visualize how the system will operate, what the system will accomplish, and how the system will be implemented. The old system's data-flow diagrams can be drawn up and compared with.

How any system is developed can be determined through a data flow diagram model. In the course of developing a set of leveled data flow diagrams, the analyst/designer is forced to address how the system may be decomposed into component sub-systems and to identify the transaction data in the data model. Data flow diagrams can be used in both the Analysis and Design phase of the SDLC. There are different notations to draw data flow diagrams. Defining different visual representations for processes, data stores, data flow, and external entities.

DFD Notation:



DFD Example:

Steps to Construct Data Flow Diagram:

Four Steps are generally used to construct a DFD.

□ **Process should be named and referred for easy reference. Each name should be representative of the reference.**

□ **The destination of flow is from top to bottom and from left to right.**

□ **When a process is distributed into lower-level details they are numbered.**

□ **The names of data stores, sources, and destinations are written in capital letters.**

Rules for constructing a Data Flow Diagram:

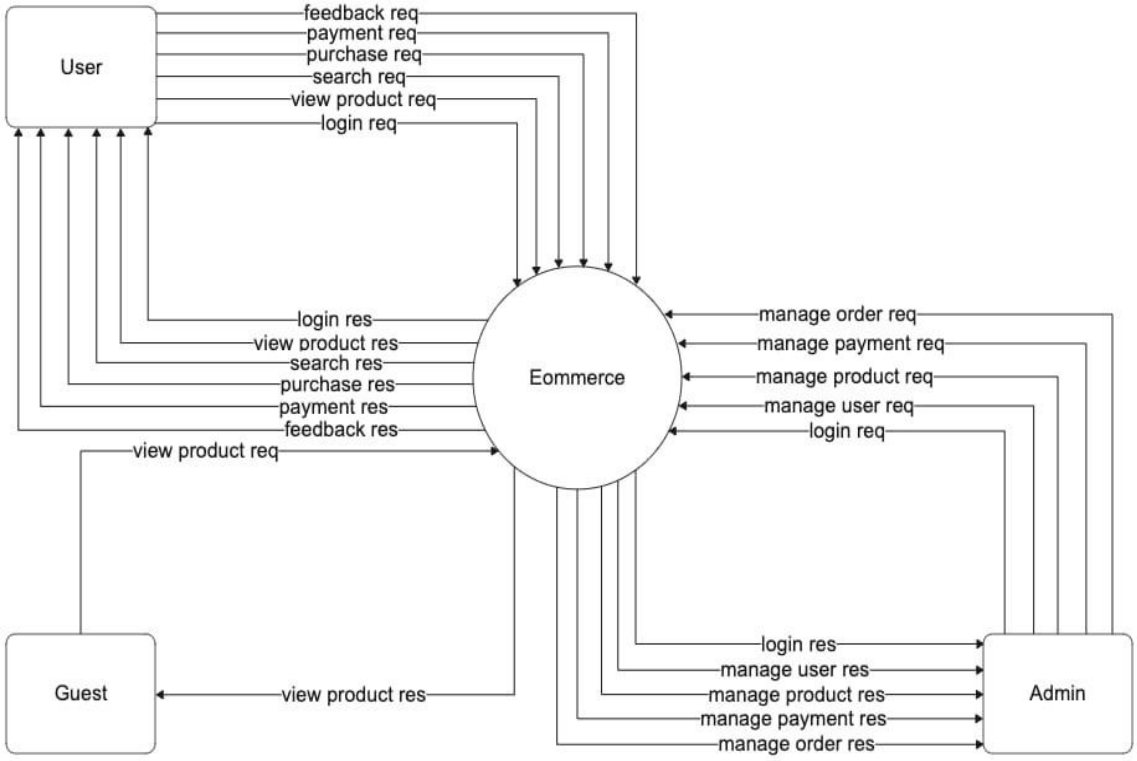
□ **Arrows should not cross each other.**

□ **Squares, Circles, and Files must bear a name.**

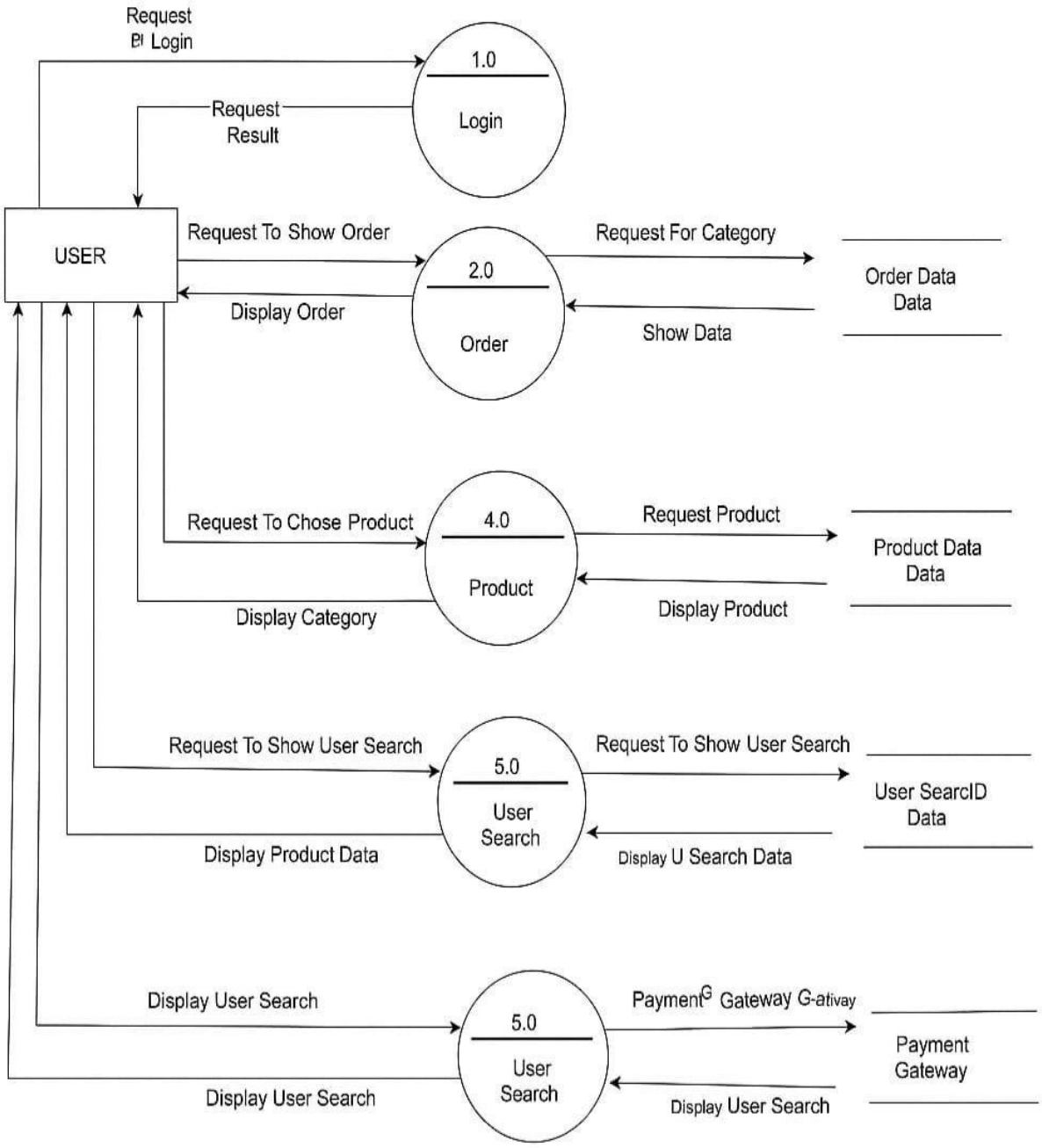
□ **Decomposed data flow squares and circles can have the same names.**

□ **Draw all data flow around the outside of the diagram.**

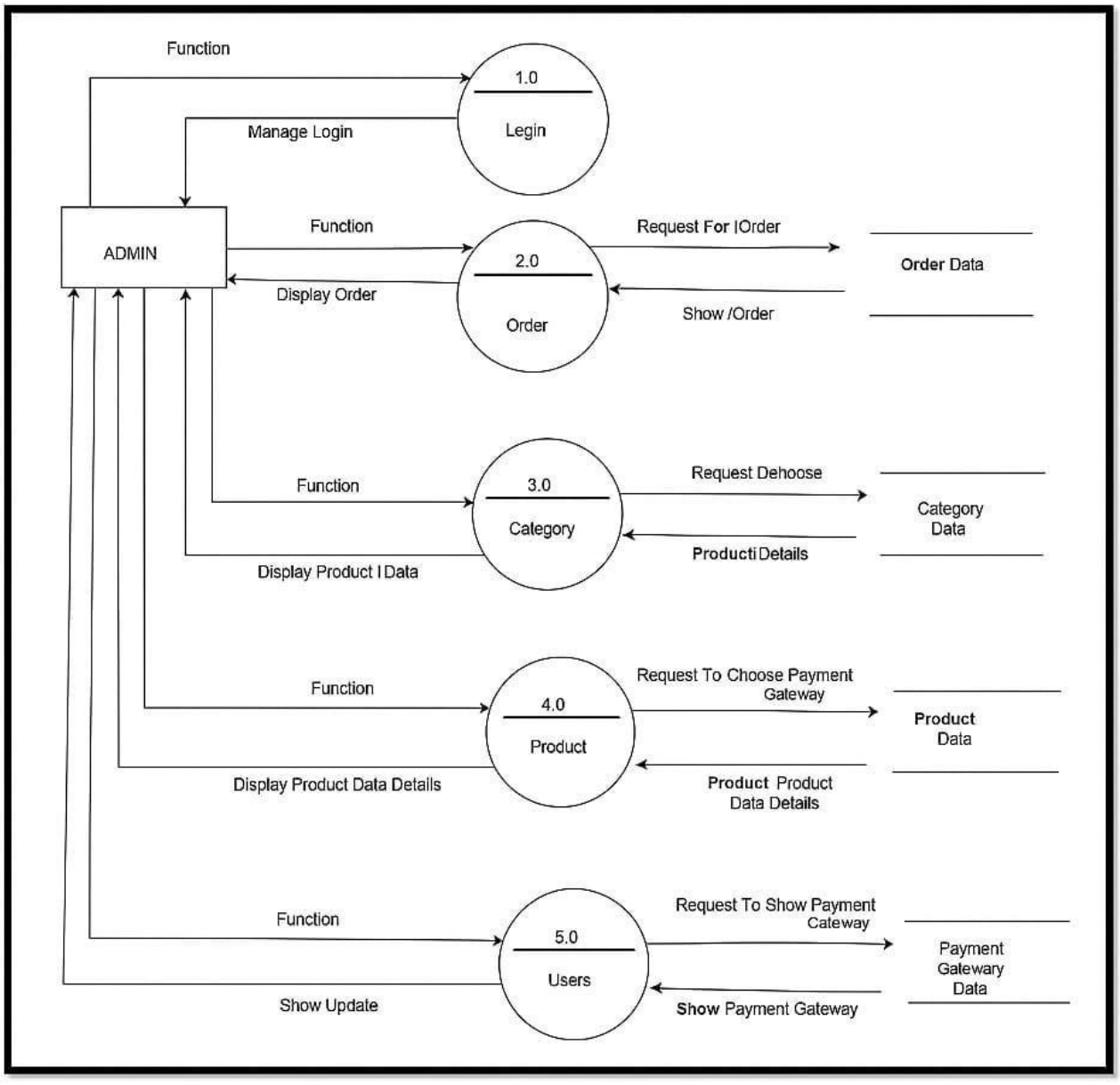
● LEVEL 0 DFD OR CONTEXT DIAGRAM:



- **LEVEL 1 DFD:**


Request
Login
1.0
Login
Request
Result
USER
Request To Show Order
Display Order
2.0
Order
Request For Category
Show Data
Order Data
Data
Request To Chose Product
Display Category
4.0
Product
Request Product
Display Product
Product Data
Data
Request To Show User Search
Display Product Data
5.0
User
Search
Request To Show User Search
Display U Search Data
User SearcID
Data
Display User Search
Display User Search
5.0
User
Search
Payment Gateway
Display User Search
Payment
Gateway

LEVEL 1 DFD: ADMIN



4B.SEQUENCE DIAGRAM

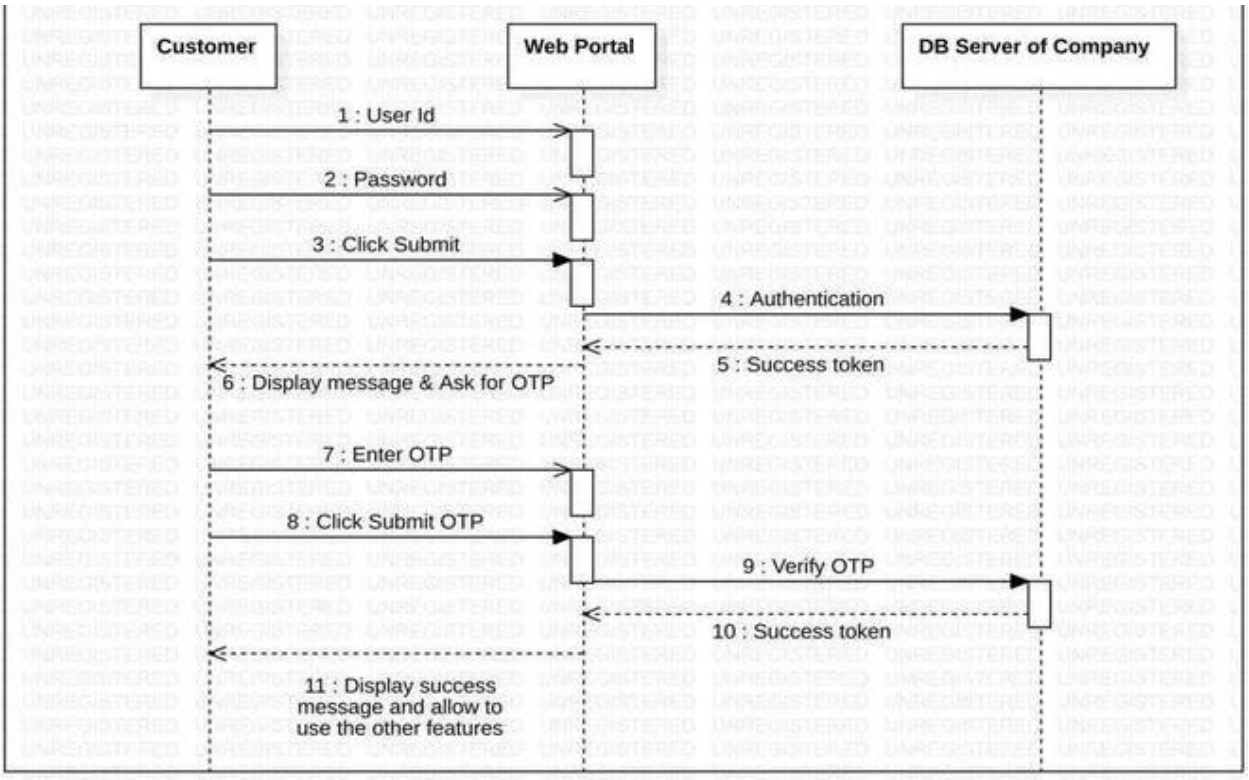
A Sequence diagram is an interaction diagram that shows how processes operate with one another and what is their order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in a time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development.

Sequence diagrams are sometimes called event diagrams or event scenarios.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

A sequence diagram is the most common kind of interaction diagram, which focuses on the message interchange between several life lines .A sequence diagram describes an interaction by focusing on the sequence of messages that are exchanged, along with their corresponding occurrence specifications on the lifelines.

The following nodes and edges are typically drawn in a UML sequence diagram: lifeline, execution specification, message, fragment, interaction, state invariant, continuation, and destruction occurrence.



A Use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the use rand the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

So only static behavior is not sufficient to model a system rather dynamic behavior is more important than static behavior. In UML there are five diagrams available to model dynamic nature and a use case diagram is one of them. Now as we have to discuss that the use case diagram is dynamic in nature there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors. So, use case diagrams consist of actors, use cases, and their relationships. The diagram is used to model the system/subsystem of an application. A single-use case diagram captures a particular functionality of a system.

So, to model the entire system numbers of use case diagrams are used. The purpose of a use case diagram is to capture the dynamic aspect of a system. But this definition is too generic to describe the purpose.

Because the other four diagrams (activity, sequence, collaboration, and State chart) are also having the same purpose. So, we will look into some specific purpose that will distinguish it from the other four diagrams.

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. So, when a system is analyzed to gather its functionalities use cases are prepared and actors are identified.

Now when the initial task is complete use case diagrams are modeled to present the outside view. So, in brief, the purposes of use case diagrams can be as follows:

Used to gather requirements of a system.

Used to get an outside view of a system.

Identify external and internal factors influencing the system.

How to draw Use Case Diagram?

Use case diagrams are considered for high level requirement analysis of a system. So, when the requirements of a system are analyzed, the functionalities are captured in use cases.

So, we can say that uses cases are nothing but the system functionalities written in an organized manner. Now the second things which are relevant to the use cases are the actors. Actors can be defined as something that interacts with the system.

The actors can be human user, some internal applications or may be some external applications. So, in a brief when we are planning to draw use case diagram, we should have the following items identified.

Functionalities to be represented as a use case

Actors

Relationships among the use cases and actors.

Use case diagrams are drawn to capture the functional requirements of a system. So, after identifying the above items we have to follow the following guidelines to draw an efficient use case diagram.

The name of a use case is very important. So, the name should be chosen in such a way so that it can identify the functionalities performed.

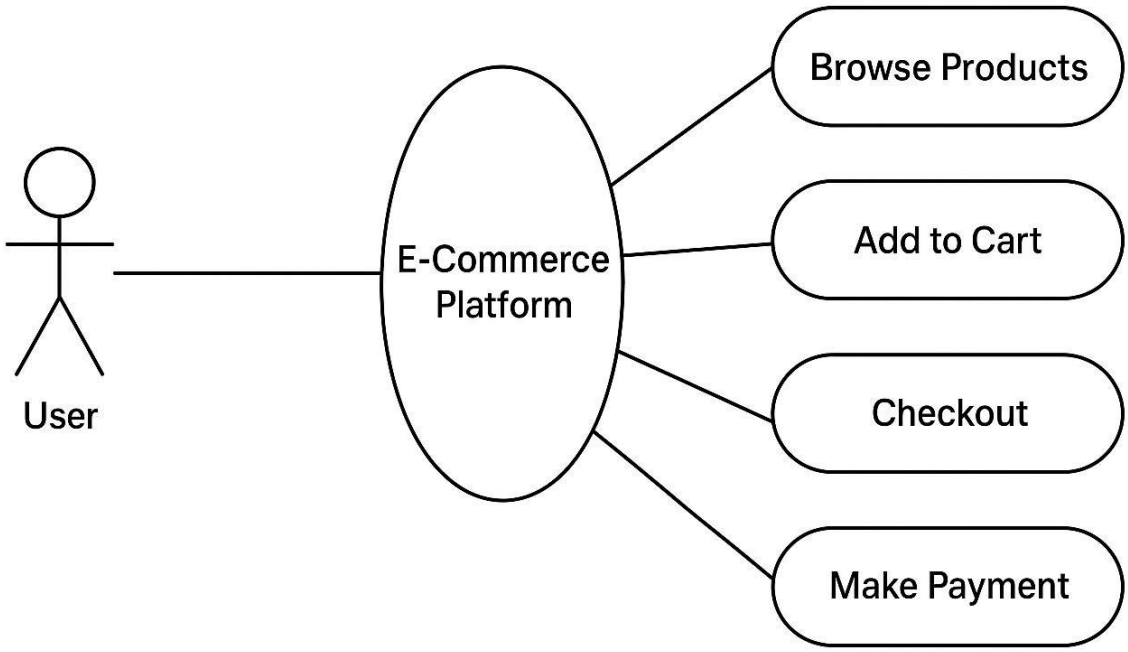
Give a suitable name for actors.

Show relationships and dependencies clearly in the diagram.

Do not try to include all types of relationships. Because the main purpose of the diagram is to identify requirements.

Use note whenever required to clarify some important point

• USE CASE DIAGRAM:

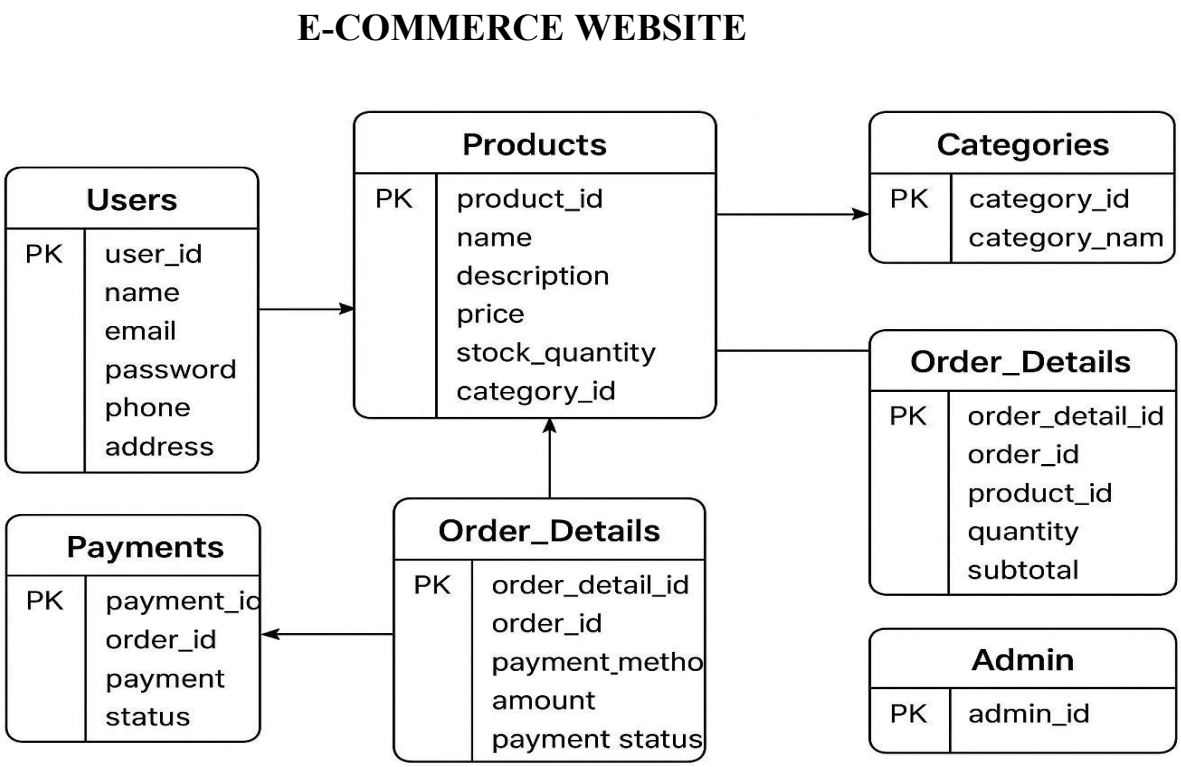


4D.SCHEMA DIAGRAM

The schema is an abstract structure or outline representing the logical view of the database as a whole. Defining categories of data and relationships between those categories, database schema design makes data much easier to retrieve, consume, manipulate, and interpret.

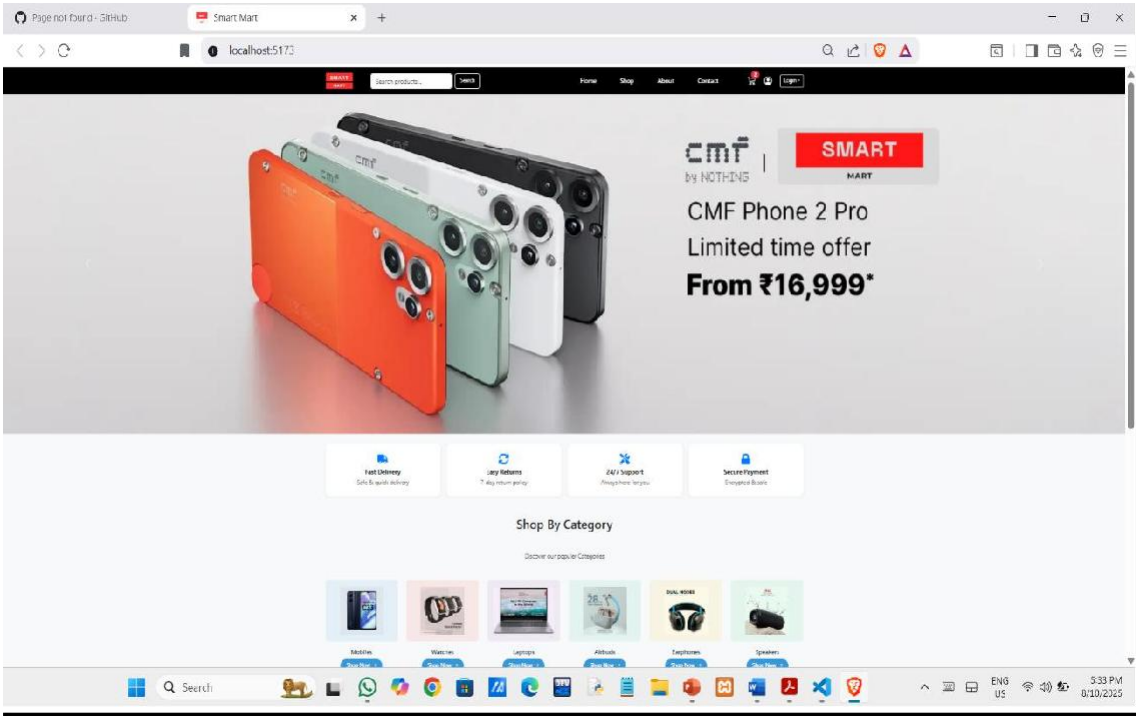
DB schema design organizes data into separate entities, determines how to create relationships between organized entities, and influences the applications of constraints on data. Designers create database schema to give other database users, such as progra mmers and analysts, a logical understanding of data.

- **SCHEMA DESIGN:**



5. UI SNAPSHOT

- **FRONTEND :-**
  - **HOME:-**



CODE:-

```
import React from 'react'
import Header from '../components/Header'
import Carouselslider from '../components/Carouselslider'
import CategoryGrid from '../components/categoryGrid'
import FeaturedProducts from '../components/FeaturedProducts'
import HomeServices from '../components/HomeServices'
const Home = () => {
  return <>
    <Carouselslider/>
    <HomeServices/>
```

```
    <CategoryGrid/>
    <FfeaturedProducts/>
  </>
}

export default Home;
```

- **HEADER:**

CODE –

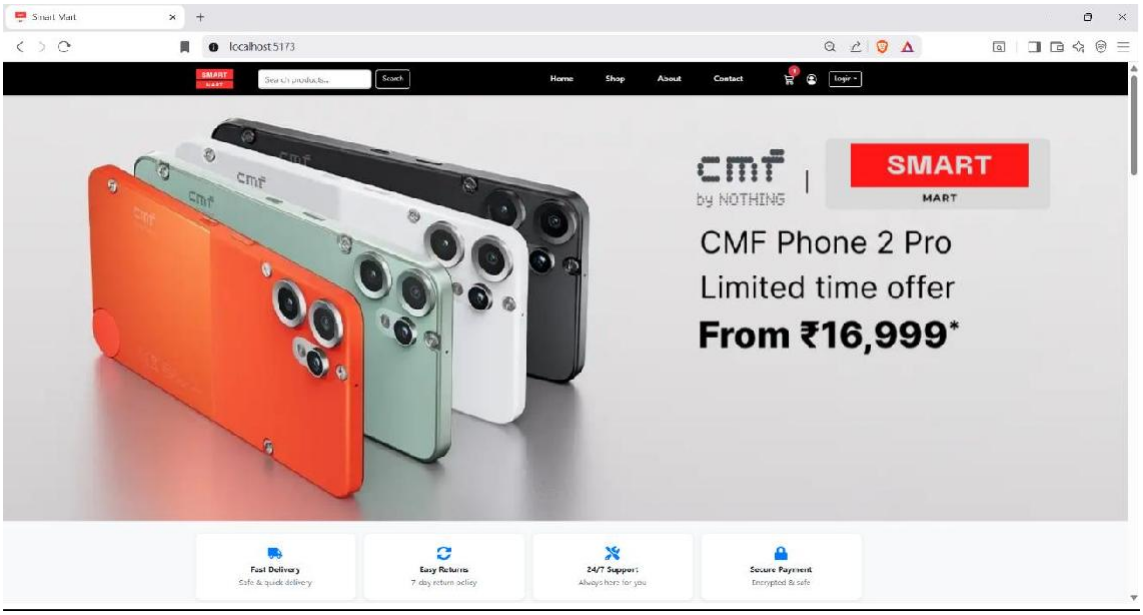
```
import React, { useContext } from 'react';
import { Link, useNavigate } from 'react-router-dom';
import { useAuth } from '../context/AuthContext';
import { useCart } from '../context/CartContext';
import logo from '../assets/logo.png';
import { toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';
import SearchBox from './SearchBox';


const Header = () => {
  const { user, logout } = useAuth();
  const { cartItems } = useCart();
  const navigate = useNavigate();


  const handleLogout = () => {
    logout();
    toast.success("Logged out successfully!");
    navigate('/');
  };


  return (
    <nav className="navbar navbar-expand-lg bg-black shadow-sm py-2 sticky-top">
      <div className="container d-flex align-items-center justify-content-between">
```

```
{/* Logo */}
<Link className="navbar-brand me-5" to="/">
  <img src={logo} alt="Smart Mart" height="40" />
</Link>

{/* Search Box */}
<SearchBox/>

{/* Navbar toggler */}
<button className="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarContent">
  <span className="navbar-toggler-icon"></span>
</button>

<div className="collapse navbar-collapse" id="navbarContent">
  <div className="d-flex flex-wrap align-items-center justify-content-end w-100 gap-4 mt-3 mt-lg-0">
    <ul className="navbar-nav flex-row gap-5 mr-5">
      <li className="nav-item"><Link className="nav-link text-light fw-semibold" to="/">Home</Link></li>
      <li className="nav-item"><Link className="nav-link text-light fw-semibold" to="/shop">Shop</Link></li>
      <li className="nav-item"><Link className="nav-link text-light fw-semibold" to="/about">About</Link></li>
      <li className="nav-item"><Link className="nav-link text-light fw-semibold" to="/contact">Contact</Link></li>
      {user && user.role === 'admin' && (
        <li className="nav-item"><Link className="nav-link text-light fw-semibold" to="/dashboard">Dashboard</Link></li>
      )}
    </ul>
```

```
{/* CART ICON */}
<Link to="/cart" className="text-light position-relative">
 <i className="bi bi-cart3 fs-5"></i>
 {cartItems.length > 0 && (
  <span className="position-absolute top-0 start-100 translate-middle badge rounded-pill bg-danger">
   {cartItems.length}
  </span>
 )}
</Link>

{/* AUTH SECTION */}
{!user ? (
 <>
  <i className="bi bi-person-circle fs-5 text-light" data-bs-toggle="tooltip" title="Account"></i>
  <div className="dropdown">
   <button className="btn btn-outline-light btn-sm dropdown-toggle" type="button" id="loginDropdown" data-bs-toggle="dropdown" aria-expanded="false">Login</button>
   <ul className="dropdown-menu dropdown-menu-end" aria-labelledby="loginDropdown">
    <li className="dropdown-item small d-flex justify-content-between">
     <span>New user?</span>
     <Link to="/signup" className="text-primary ms-2">Sign Up</Link>
    </li>
    <li><hr className="dropdown-divider" /></li>
    <li><Link to="/login" className="dropdown-item">Login</Link></li>
    <li><Link to="/orders" className="dropdown-item">Orders</Link></li>
   </ul>
```

```
                </div>
              </>
            ) : (
              <>
                <Link to={user.role === 'admin' ? '/dashboard' : '/profile'} 
                  className="d-flex align-items-center text-light gap-2">
                  <i className="bi bi-person-circle fs-5" />
                  <span>{user.name?.split(' ')[0]}</span>
                </Link>
                <button onClick={handleLogout} className="btn btn-outline-light 
                  btn-sm" data-bs-toggle="tooltip" title="Logout">
                  <i className="bi bi-box-arrow-right"></i>
                </button>
              </>
            )}

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

export default Header;
```

- **SLIDER:-**

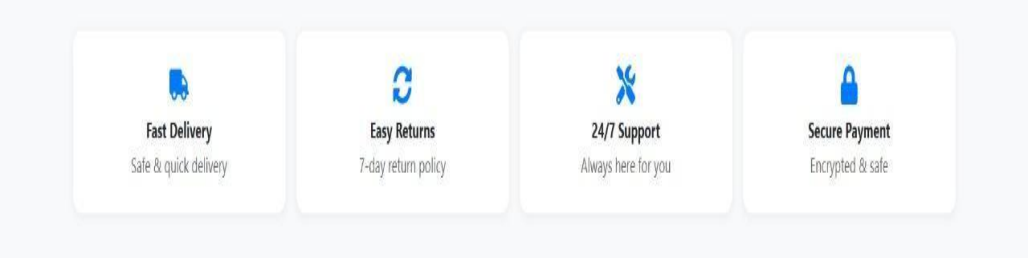


CODE :-

```
import React from 'react';
import banner1 from '../assets/mainbanner.png';
import banner2 from '../assets/laptop.jpg';


const Carouselslider = () => {
 return (
  <div id="carouselExampleControls" className="carousel slide" data-bs-ride="carousel">
    <div className="carousel-inner">
      <div className="carousel-item active">
        <img src={banner1} className="d-block w-100" alt="mainbanner 1" />
```

```
      </div>

      <div className="carousel-item">
        <img src={banner2} className="d-block w-100" alt="laptops" />
      </div>
    </div>

    <button className="carousel-control-prev" type="button" data-bs-target="#carouselExampleControls" data-bs-slide="prev">
      <span className="carousel-control-prev-icon" aria-hidden="true"></span>
      <span className="visually-hidden">Previous</span>
    </button>

    <button className="carousel-control-next" type="button" data-bs-target="#carouselExampleControls" data-bs-slide="next">
      <span className="carousel-control-next-icon" aria-hidden="true"></span>
      <span className="visually-hidden">Next</span>
    </button>
  </div>
 );
};

export default Carouselslider;
```

- **SERVICE :-**

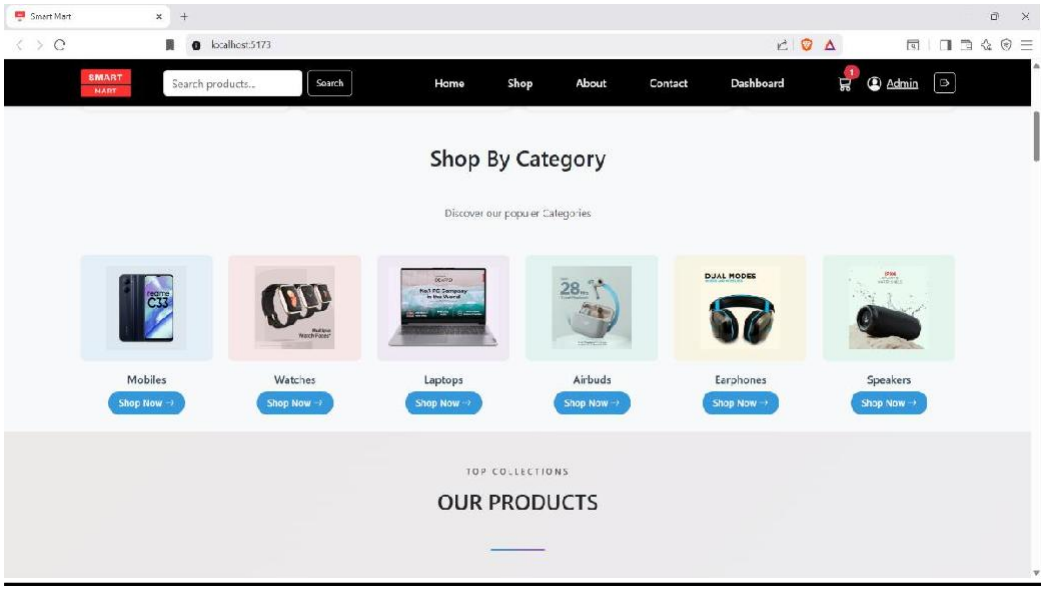


CODE :-

```
import React from 'react';
import { Container, Row, Col, Card } from 'react-bootstrap';
import { FaTools, FaTruck, FaSyncAlt, FaLock } from 'react-icons/fa';
import './HomeServices.css'


const services = [
  {
    icon: <FaTruck size={28} className="service-icon" />,
    title: 'Fast Delivery',
    description: 'Safe & quick delivery',
  },
  {
    icon: <FaSyncAlt size={28} className="service-icon" />,
    title: 'Easy Returns',
    description: '7-day return policy',
  },
  {
    icon: <FaTools size={28} className="service-icon" />,
    title: '24/7 Support',
    description: 'Always here for you',
  },
  {
    icon: <FaLock size={28} className="service-icon" />,
    title: 'Secure Payment',
```

```
    description: 'Encrypted & safe',
  },
];

const HomeServices = () => {
  return (
    <section className="py-4 bg-light">
      <Container fluid="lg">
        <Row className="g-3 justify-content-center">
          {services.map((service, index) => (
            <Col xs={6} md={3} key={index}>
              <Card className="text-center service-card h-100 border-0">
                <Card.Body className="d-flex flex-column align-items-center justify-content-
center py-4">
                  <div className="mb-2">{service.icon}</div>
                  <Card.Title className="fw-bold mb-1">{service.title}</Card.Title>
                  <Card.Text className="text-muted small text-center mb-0">
                    {service.description}
                  </Card.Text>
                </Card.Body>
              </Card>
            </Col>
          ))}
        </Row>
      </Container>
    </section>
  );
};

export default HomeServices;
```

- **CATEGORY :-**



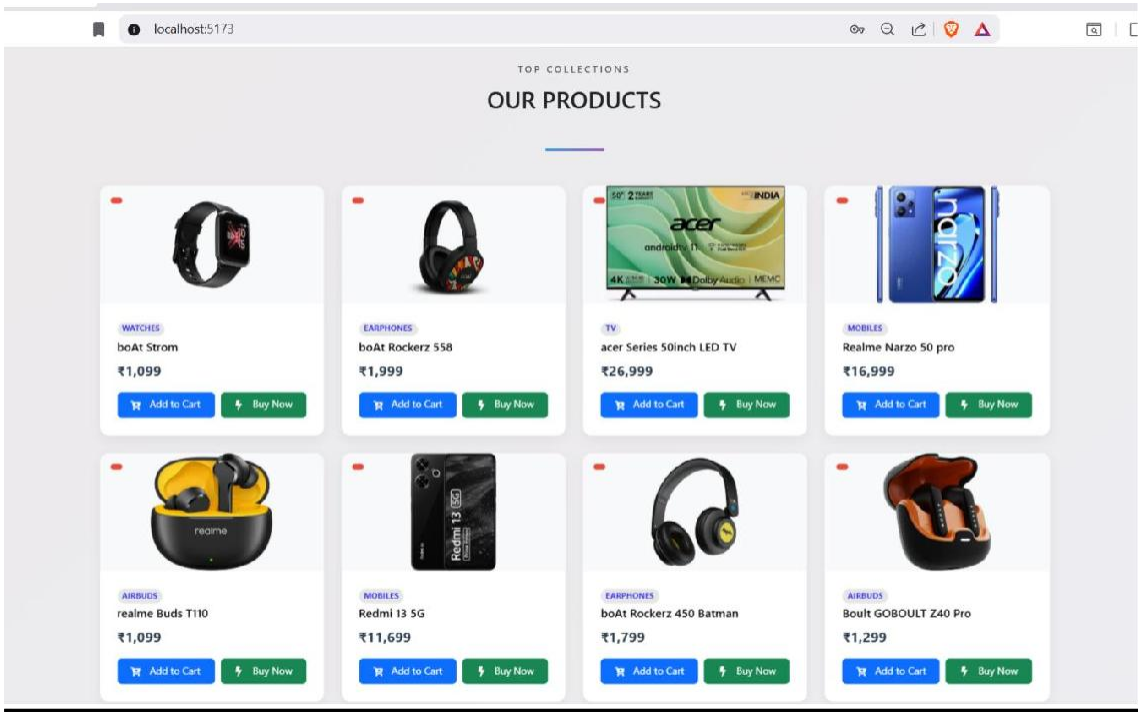
CODE:-

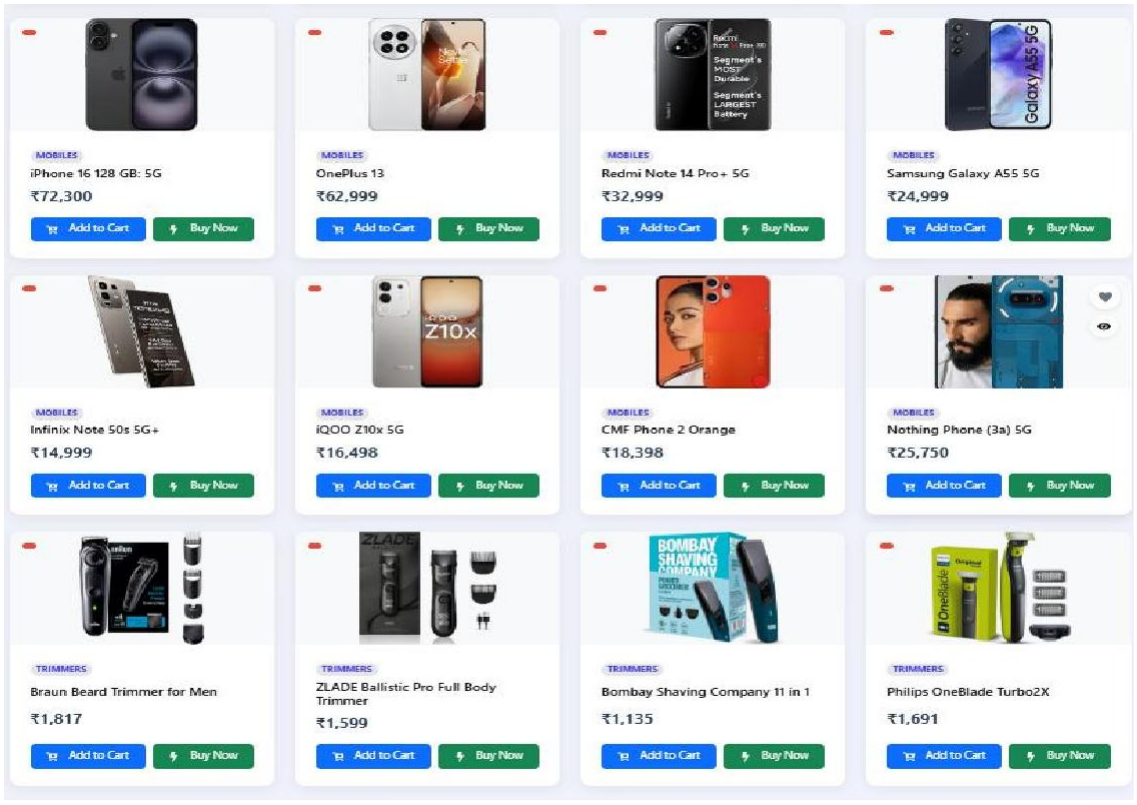
```
import React from 'react';
import { Link } from 'react-router-dom';

import mobileImg from '../assets/categories/mobile.jpg';
import watchImg from '../assets/categories/watches.jpg';
import laptopImg from '../assets/categories/laptop.jpg';
import airbudsImg from '../assets/categories/airbuds.jpg';
import headsetImg from '../assets/categories/headset.jpg';
import speakerImg from '../assets/categories/speakers.jpg';

import './CategoryGrid.css';

const categories = [

  { name: 'Mobiles', image: mobileImg, link: '/category/mobiles', bgColor: 'rgba(52, 152, 219, 0.1)' },

  { name: 'Watches', image: watchImg, link: '/category/watches', bgColor: 'rgba(231, 76, 60, 0.1)' },
```

```
  { name: 'Laptops', image: laptopImg, link: '/category/laptops', bgColor: 'rgba(155, 89, 182, 0.1)' },
  { name: 'Airbuds', image: airbudsImg, link: '/category/airbuds', bgColor: 'rgba(26, 188, 156, 0.1)' },
  { name: 'Earphones', image: headsetImg, link: '/category/headsets', bgColor: 'rgba(241, 196, 15, 0.1)' },
  { name: 'Speakers', image: speakerImg, link: '/category/speakers', bgColor: 'rgba(46, 204, 113, 0.1)' },
];

const CategoryGrid = () => {
  return (
    <section className="category-section py-4">
      <div className="container">
        <div className="section-header text-center mb-5">
          <h2 className="text-center fw-semibold mb-5 text-dark">Shop By Category</h2>
          <p className="section-subtitle text-muted">Discover our populer Categories</p>
        </div>

        <div className="row g-4">
          {categories.map((category, index) => (
            <div className="col-6 col-md-4 col-lg-2" key={index}>
              <Link to={category.link} className="text-decoration-none">
                <div className="category-card h-100 transition-all">
                  <div
                    className="category-img-container position-relative overflow-hidden rounded-3"
                    style={{ backgroundColor: category.bgColor }}
                  >
                    <img
                      src={category.image}
                      alt={category.name}
                      className="category-img img-fluid"
```

```
                />
                <div className="category-overlay"></div>
              </div>
              <div className="category-content text-center mt-3">
                <h5 className="category-name mb-2">{category.name}</h5>
                <button className="btn btn-shop-now">
                  Shop Now <i className="bi bi-arrow-right ms-1"></i>
                </button>
              </div>
            </div>
          </Link>
        </div>
      ))}
    </div>
  </div>
 </section>
 );
};

export default CategoryGrid;
```

- **FEATURES PRODUCT:-**





CODE :-

```
import React, { useEffect, useState } from 'react';
import axios from 'axios';
import {
  Spinner,
  Button,
  Badge,
  OverlayTrigger,
  Tooltip,
  Modal,
  Toast,
  ToastContainer
} from 'react-bootstrap';
import { FaCartPlus, FaHeart, FaEye, FaStar, FaBolt } from 'react-icons/fa';
import './FeaturedProducts.css';
import { useCart } from '../context/CartContext';
import { useNavigate } from 'react-router-dom';

const FeaturedProducts = () => {
  const [products, setProducts] = useState([]);
  const [loading, setLoading] = useState(true);
  const [wishlist, setWishlist] = useState([]);
  const [showToast, setShowToast] = useState(false);
  const [quickViewProduct, setQuickViewProduct] = useState(null);
  const navigate = useNavigate();
  const { addToCart } = useCart();

  useEffect(() => {
    const fetchFeaturedProducts = async () => {
      try {
```

```
      const response = await axios.get('http://localhost:5000/api/products/featured');
      setProducts(response.data);
    } catch (error) {
      console.error('Failed to fetch featured products:', error);
    } finally {
      setLoading(false);
    }
  };

  fetchFeaturedProducts();
}, []);

const toggleWishlist = (productId) => {
  setWishlist(prev =>
  prev.includes(productId)
    ? prev.filter(id => id !== productId)
    : [...prev, productId]
  );
};

const handleBuyNow = (id) => {
  setShowToast(true);
  navigate(/product/${id});
};

return (
  <section className="featured-products-section">
    <div className="container py-5">
      <div className="section-intro text-center mb-5">
        <h6 className="section-pretitle text-uppercase text-muted mb-3">Top
Collections</h6>
```

```
      <h2 className="text-center fw-semibold mb-5 text-dark">OUR PRODUCTS</h2>
      <div className="divider mx-auto"></div>
     </div>

     {loading ? (
      <div className="text-center py-5">
       <Spinner animation="border" variant="primary" className="spinner-grow-lg" />
       <p className="mt-3 text-muted">Curating our finest selections...</p>
      </div>
     ) : (
      <div className="row g-4">
       {products.length > 0 ? (
        products.map(product => (
         <div className="col-12 col-sm-6 col-md-4 col-lg-3" key={product._id}>
          <div className="product-card-wrapper">
           <div className="product-card card border-0 h-100">
            <div className="product-img-container position-relative overflow-hidden">
             <div className="product-badge position-absolute">
              {product.isNew && (
               <Badge pill bg="danger" className="new-badge">New</Badge>
              )}
              {product.discount > 0 && (
               <Badge pill bg="success" className="discount-badge">
                -{product.discount}%
               </Badge>
              )}
             </div>

             <div className="product-actions position-absolute">
```

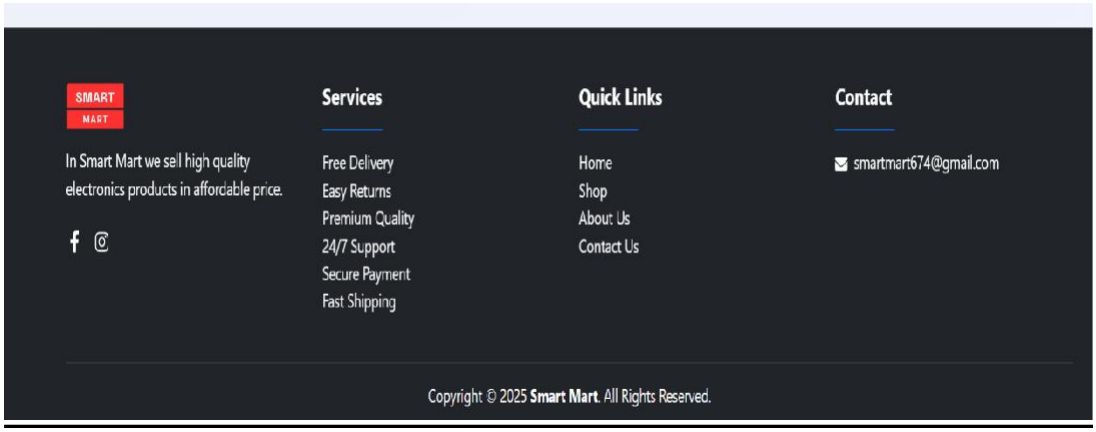
```
                    <button
                      className={wishlist-btn ${wishlist.includes(product._id) ? 'active' : ''}}
                      onClick={() => toggleWishlist(product._id)}
                    >
                      <FaHeart />
                    </button>
                    <OverlayTrigger placement="top" overlay={<Tooltip>Quick View</Tooltip>}>
                      <button className="quickview-btn" onClick={() => setQuickViewProduct(product)}>
                        <FaEye />
                      </button>
                    </OverlayTrigger>
                  </div>
                  <img
                    src={http://localhost:5000${product.image}}
                    alt={product.name}
                    className="img-fluid product-img"
                    onError={(e) => {
                      e.target.src = 'https://via.placeholder.com/300x300?text=Product+Image';
                    }}
                  />
                </div>

                <div className="card-body position-relative d-flex flex-column">
                  <div className="product-category mb-1">
                    <Badge pill bg="light" text="dark" className="text-uppercase">
                      {product.category?.name || 'General'}
                    </Badge>
                  </div>
```

```
                <h5 className="product-title mb-2">
                  <a href={/product/${product._id}}>{product.name}</a>
                </h5>

                <div className="product-price-wrapper mb-3">
                  {product.discount > 0 ? (
                    <>
                      <span className="current-price">₹{(product.price * (1 - product.discount / 100)).toFixed(2)}</span>
                      <span className="original-price">₹{product.price.toLocaleString()}</span>
                    </>
                  ) : (
                    <span className="current-price">₹{product.price.toLocaleString()}</span>
                  )}
                </div>

                    <div className="d-flex justify-content-between gap-2">
  <Button
    variant="primary"
    className="flex-fill"
    onClick={() => addToCart(product)}
  >
    <FaCartPlus className="me-2" /> Add to Cart
  </Button>
  <Button
    variant="success"
    className="flex-fill"
    onClick={() => handleBuyNow(product._id)}
  >
    <FaBolt className="me-2" /> Buy Now
```

```
  </Button>
</div>


                </div>
              </div>
            </div>
          </div>
        ))
      ) : (
        <div className="col-12 text-center py-5">
          <div className="alert alert-light border">
            <i className="bi bi-exclamation-circle display-4 text-muted mb-3"></i>
            <h4 className="mb-3">No Featured Products Available</h4>
            <p className="text-muted">Check back later for our special selections</p>
          </div>
        </div>
      )}
    </div>
  )}
</div>


{/* Quick View Modal */}
<Modal show={!!quickViewProduct} onHide={() => setQuickViewProduct(null)}
centered>
  <Modal.Header closeButton>
    <Modal.Title>{quickViewProduct?.name}</Modal.Title>
  </Modal.Header>
  <Modal.Body>
    <img
      src={http://localhost:5000${quickViewProduct?.image}}
      alt={quickViewProduct?.name}
      className="img-fluid mb-3"
```

```
        />
        <p><strong>Price:</strong> ₹{quickViewProduct?.price}</p>
        <p><strong>Description:</strong> {quickViewProduct?.description || 'No description available.'}</p>
      </Modal.Body>
      <Modal.Footer>
        <Button variant="secondary" onClick={() => setQuickViewProduct(null)}>Close</Button>
        <Button variant="primary" onClick={() => {
          addToCart(quickViewProduct);
          setShowToast(true);
        }}>
          <FaCartPlus className="me-2" /> Add to Cart
        </Button>
      </Modal.Footer>
    </Modal>

    {/* Toast Notification */}
    <ToastContainer position="bottom-end" className="p-3">
      <Toast onClose={() => setShowToast(false)} show={showToast} delay={2000} autohide bg="info">
        <Toast.Header closeButton={false}>
          <strong className="me-auto">SMART MART</strong>
        </Toast.Header>
        <Toast.Body>Redirecting to product page...</Toast.Body>
      </Toast>
    </ToastContainer>
  </section>
 );
};

export default FeaturedProducts;
```

- **FOOTER:-**



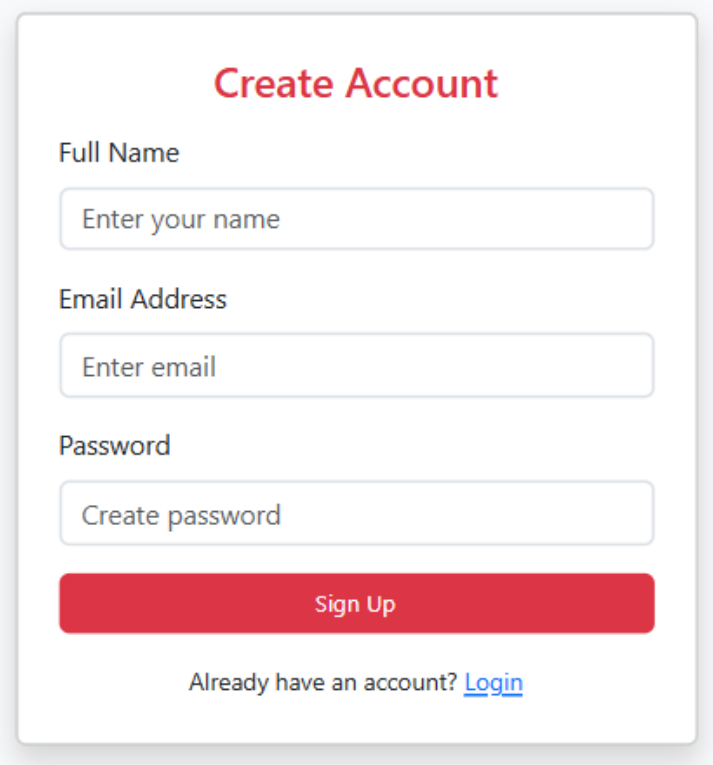
CODE:-

```
import React from 'react';
import { FaFacebookF, FaInstagram, FaEnvelope, } from 'react-icons/fa';
import logo from '../assets/logo.png';

const Footer = () => {
  return (
    <footer className="bg-dark text-white pt-5 pb-3">
      <div className="container">
        <div className="row gy-4">
          {/* Logo & About */}
          <div className="col-md-3">
            <img src={logo} alt="Smart Mart" height="40" />
            <p className="mt-3">
              In Smart Mart we sell high quality electronics products in affordable price.
            </p>
            <div className="d-flex gap-3 mt-2">
              <a href="#" className="text-white fs-5"><FaFacebookF /></a>
              <a href="#" className="text-white fs-5"><FaInstagram /></a>
```

```
  </div>
</div>

{/* Services */}
<div className="col-md-3">
  <h5 className="fw-bold">Services</h5>
  <hr className="border-primary border-2 opacity-100 w-25" />
  <ul className="list-unstyled">
    <li>Free Delivery</li>
    <li>Easy Returns</li>
    <li>Premium Quality</li>
    <li>24/7 Support</li>
    <li>Secure Payment</li>
     <li>Fast Shipping</li>
  </ul>
</div>

{/* Quick Links */}
<div className="col-md-3">
  <h5 className="fw-bold">Quick Links</h5>
  <hr className="border-primary border-2 opacity-100 w-25" />
  <ul className="list-unstyled">
    <li><a href="/" className="text-white text-decoration-none">Home</a></li>
    <li><a href="/shop" className="text-white text-decoration-none">Shop</a></li>
    <li><a href="/about" className="text-white text-decoration-none">About Us</a></li>
    <li><a href="/contact" className="text-white text-decoration-none">Contact Us</a></li>
  </ul>
</div>

{/* Contact */}
<div className="col-md-3">
```

```
          <h5 className="fw-bold">Contact</h5>
          <hr className="border-primary border-2 opacity-100 w-25" />
          <p className="mb-2"><FaEnvelope className="me-2" />smartmart674@gmail.com</p>
        </div>
      </div>
      <hr className="text-secondary mt-4" />
      <p className="text-center mb-0">Copyright © 2025 <strong>Smart Mart</strong>. All Rights Reserved.</p>
    </div>
  </footer>
 );
};

export default Footer;
```

- **SIGN UP:-**



CODE:-

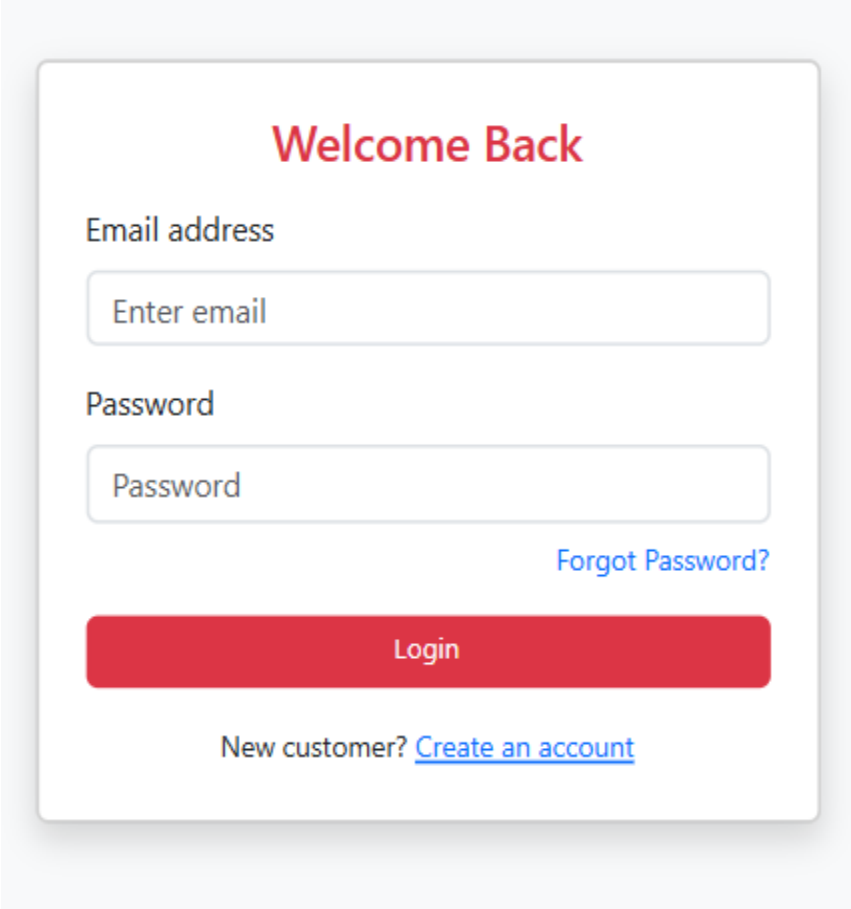
```
import React, { useState } from 'react';
import { Link, useNavigate } from 'react-router-dom';
import axios from 'axios';
import { useAuth } from '../context/AuthContext'; // Optional if you use global auth context

function Signup() {
  const [name, setName] = useState('');
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const navigate = useNavigate();
  const { login } = useAuth(); // Optional: for auto-login after signup

  const handleSignup = async (e) => {
```

```
    e.preventDefault();

    try {
      const { data } = await axios.post("http://localhost:5000/api/signup", {
        name,
        email,
        password,
      });

      if (data.success) {
        alert("Signup successful! Please login.");
        // Optional: auto-login
        // login({ ...data.data.user, token: data.data.token });
        navigate("/login");
      } else {
        alert(data.message || "Signup failed");
      }
    } catch (err) {
      console.error(err);
      alert("Signup failed. Please try again.");
    }
  };

  return (
    <div  classNam="d-flex  justify-content-center  align-items-center  vh-100  bg-
light">
      <div classNam="card shadow p-4" style={{ width: '100%', maxWidth: '400px'
}}>
        <h4 classNam="text-center text-danger mb-3">Create Account</h4>

        <form onSubmit={handleSignup}>
          <div classNam="mb-3">
```

```
 <label htmlFor="name" className="form-label">Full Name</label>
 <input
  type="text"
  className="form-control"
  id="name"
  placeholder="Enter your name"
  value={name}
  onChange={(e) => setName(e.target.value)}
  required
 />
</div>

<div className="mb-3">
 <label htmlFor="email" className="form-label">Email Address</label>
 <input
  type="email"
  className="form-control"
  id="email"
  placeholder="Enter email"
  value={email}
  onChange={(e) => setEmail(e.target.value)}
  required
 />
</div>

<div className="mb-3">
 <label htmlFor="password" className="form-label">Password</label>
 <input
  type="password"
  className="form-control"
  id="password"
```

```
            placeholder="Create password"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
            required
            minLength={6}
          />
        </div>

        <div className="d-grid mb-3">
          <button type="submit" className="btn btn-danger">Sign Up</button>
        </div>

        <div className="text-center">
          <small>
            Already have an account?{" "}
            <Link to="/login" className="text-primary">Login</Link>
          </small>
        </div>
      </form>
    </div>
  </div>
 );
}

export default Signup;
```

- **LOG-IN :-**



CODE:-

```
import axios from 'axios';
import React, { useState } from 'react';
import { Link, useNavigate } from 'react-router-dom';
import { useAuth } from '../context/AuthContext';
import { toast } from 'react-toastify';

const Login = () => {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [loading, setLoading] = useState(false);

  const { login } = useAuth();
```

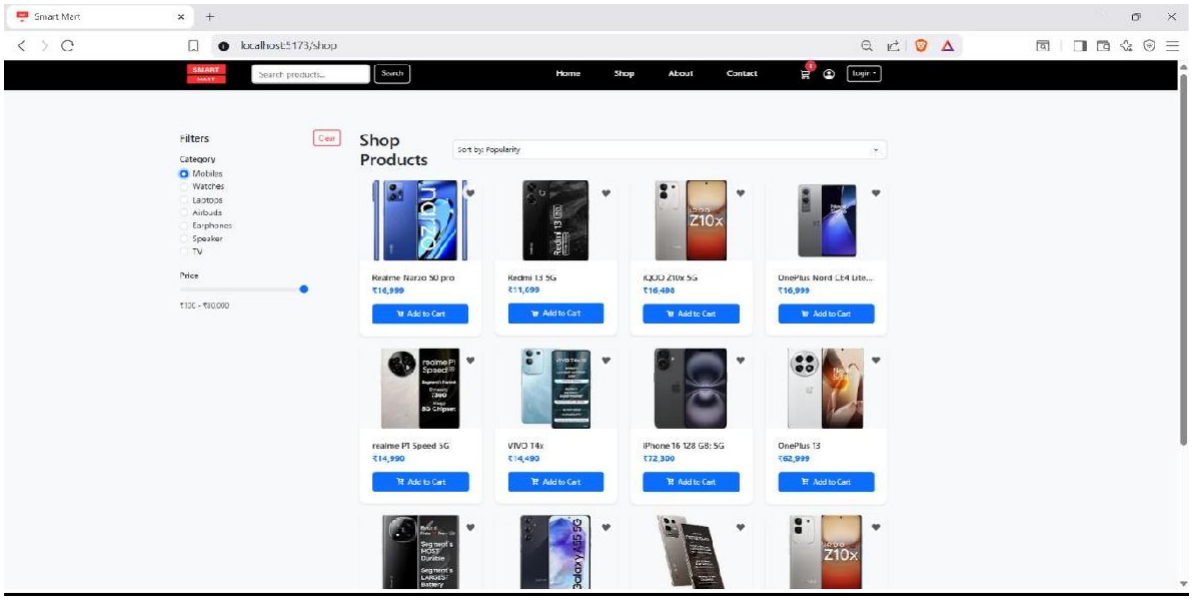
```
const navigate = useNavigate();

const handleLogin = async (e) => {
e.preventDefault();
setLoading(true);


try {
  const { data } = await axios.post(
    "http://localhost:5000/api/login",
    { email, password },
    { withCredentials: true }
  );

  if (data.success) {
    const user = data.data.user;
    const token = data.data.token;

    // ⬛ Fix: Save token to
    localStorage
    localStorage.setItem('token', token);

    // Save to context (optional)
    login({ ...user, token });

    toast.success('Login successful');

    // Redirect based on role
    if (user && user.role === 'admin') {
      navigate('/dashboard', { replace: true });
    } else {
      navigate('/profile', { replace: true });
    }
```

```
    } else {
      toast.error(data.message || 'Login failed');
    }
  } catch (err) {
    console.error(err);
    toast.error("Login failed. Check your credentials.");
  } finally {
    setLoading(false);
  }
};



  return (
    <div className="d-flex justify-content-center align-items-center vh-100 bg-light">
      <div className="card shadow p-4" style={{ width: '100%', maxWidth: '400px' }}>
        <h4 className="text-center text-danger mb-3">Welcome Back</h4>

        <form onSubmit={handleLogin}>
          <div className="mb-3">
            <label htmlFor="email" className="form-label">Email address</label>
            <input
              type="email"
              className="form-control"
              id="email"
              placeholder="Enter email"
              value={email}
              onChange={(e) => setEmail(e.target.value)}
              required
            />
```

```
        </div>


        <div className="mb-1">
          <label htmlFor="password" className="form-label">Password</label>
          <input
            type="password"
            className="form-control"
            id="password"
            placeholder="Password"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
            required
          />
        </div>


        <div className="mb-3 text-end">
          <Link to="/forgot-password" className="text-decoration-none small text-primary">
            Forgot Password?
          </Link>
        </div>


        <div className="d-grid mb-3">
          <button type="submit" className="btn btn-danger" disabled={loading}>
            {loading ? (
              <>
                <span className="spinner-border spinner-border-sm me-2" role="status" />
                Logging in...
              </>
            ) : (
```

```
            'Login'
          )}
        </button>
      </div>

      <div className="text-center">
        <small>
          New customer? <Link to="/signup" className="text-primary">Create an account</Link>
        </small>
      </div>
    </form>
  </div>
 </div>
 );
};

export default Login;
```

- **SHOP PAGE:-**



CODE:-

```
import React, { useState, useEffect } from 'react';
import { Container, Row, Col, Spinner, Form } from 'react-bootstrap';
import ShopFilters from './ShopFilters';
import ProductCard from './Productcard';

import './Shop.css';

const Shop = () => {
  const [products, setProducts] = useState([]);
  const [filteredProducts, setFilteredProducts] = useState([]);
  const [loading, setLoading] = useState(true);
  const [sortBy, setSortBy] = useState('popularity');
  const [wishlist, setWishlist] = useState([]);

  const [filters, setFilters] = useState({
    category: '',
    priceRange: [100, 80000],
    brand: ''
  });
```

```
// Fetch products with error handling
useEffect(() => {
  const fetchProducts = async () => {
    try {
      setLoading(true);
      const response = await fetch('http://localhost:5000/api/products');

      if (!response.ok) {
        throw new Error(`HTTP error! status: ${response.status}`);
      }

      const data = await response.json();

      // Validate and format products
      const validatedProducts = Array.isArray(data) ? data.map(p => ({
        _id: p._id?.toString() || '',
        name: p.name?.toString() || 'Unnamed Product',
        description: p.description?.toString() || 'No description available',
        price: Number(p.price) || 0,
        image: p.image?.toString() || '',
        category: p.category?.name?.toString() || 'Uncategorized',
        brand: p.brand?.toString() || 'Unknown Brand',
        discount: Number(p.discount) || 0,
        isNew: Boolean(p.isNew),
        popularity: Number(p.popularity) || 0,
        createdAt: p.createdAt ? new Date(p.createdAt) : new Date()
      })) : [];

      setProducts(validatedProducts);
      setFilteredProducts(validatedProducts);
    } catch (error) {
      console.error('Failed to load products:', error);
    } finally {
      setLoading(false);
```

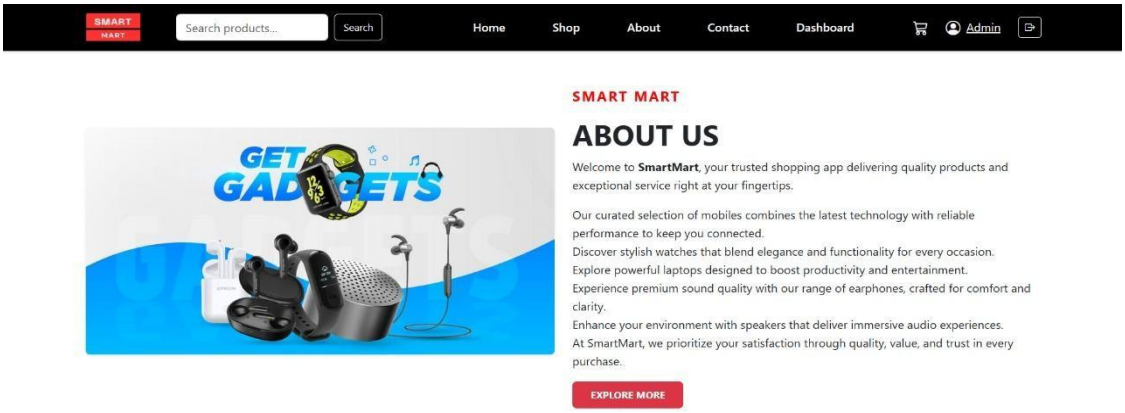
```
    }
  };

  fetchProducts();
}, []);

// Apply filters and sorting
useEffect(() => {
  try {
    let results = [...products];

    // Apply filters
    if (filters.category) {
      results = results.filter(p => p.category === filters.category);
    }
    if (filters.brand) {
      results = results.filter(p => p.brand === filters.brand);
    }
    results = results.filter(p =>
      p.price >= filters.priceRange[0] &&
      p.price <= filters.priceRange[1]
    );

    // Apply sorting
    switch(sortBy) {
      case 'price-low-high':
        results.sort((a, b) => a.price - b.price);
        break;
      case 'price-high-low':
        results.sort((a, b) => b.price - a.price);
        break;
      case 'newest':
        results.sort((a, b) => b.createdAt - a.createdAt);
        break;
      default: // popularity
```

```
      results.sort((a, b) => b.popularity - a.popularity);
    }

    setFilteredProducts(results);
  } catch (error) {
    console.error('Error filtering products:', error);
    setFilteredProducts([]);
  }
}, [filters, products, sortBy]);

// Extract unique categories and brands
const categories = [...new Set(products.map(p => p.category).filter(Boolean))];
const brands = [...new Set(products.map(p => p.brand).filter(Boolean))];

const toggleWishlist = (productId) => {
  setWishlist(prev =>
  prev.includes(productId)
    ? prev.filter(id => id !== productId)
    : [...prev, productId]
  );
};

if (loading) {
  return (
    <div className="text-center py-5">
      <Spinner animation="border" variant="primary" />
      <p className="mt-3">Loading products...</p>
    </div>
  );
}

return (
  <Container className="shop-page py-5">
    <Row>
      {/* Filters Sidebar */}
```

```
<Col lg={3} className="pe-lg-4 mb-4 mb-lg-0">
  <ShopFilters
    filters={filters}
    setFilters={setFilters}
    categories={categories}
    brands={brands}
  />
</Col>

{/* Products Grid */}
<Col lg={9}>
  <div className="d-flex justify-content-between align-items-center mb-4">
    <h2 className="mb-0">Shop Products</h2>
    <Form.Select
      className="sort-select"
      value={sortBy}
      onChange={(e) => setSortBy(e.target.value)}
    >
      <option value="popularity">Sort by: Popularity</option>
      <option value="price-low-high">Price: Low to High</option>
      <option value="price-high-low">Price: High to Low</option>
      <option value="newest">Newest Arrivals</option>
    </Form.Select>
  </div>

  {filteredProducts.length === 0 ? (
    <div className="text-center py-5 bg-white rounded-3 shadow-sm">
      <h5>No products match your filters</h5>
      <button
        className="btn btn-outline-primary mt-3"
        onClick={() => setFilters({
          category: '',
          priceRange: [100, 80000],
          brand: ''
        })}
```

```
            >
              Reset Filters
            </button>
          </div>
        ) : (
          <Row className="g-4">
            {filteredProducts.map(product => (
              <Col key={product._id} xs={6} md={4} lg={4} xl={3}>
                <ProductCard
                  product={product}
                  isInWishlist={wishlist.includes(product._id)}
                  onWishlistToggle={toggleWishlist}
                />
              </Col>
            ))}
          </Row>
        )}
      </Col>
    </Row>
  </Container>
 );
};

export default Shop;
```

- **ABOUT PAGE:-**



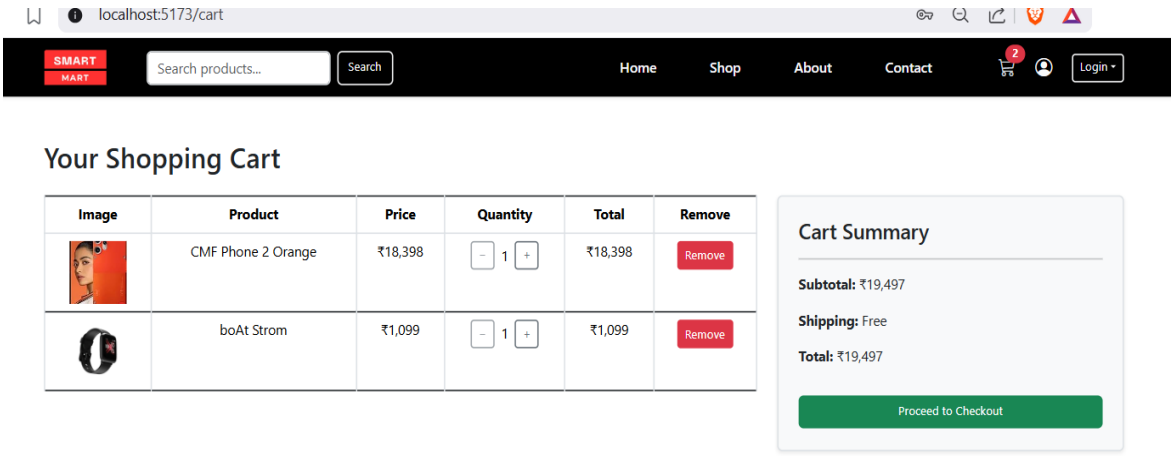
CODE:-

```
import React from "react";
import { Container, Row, Col, Button } from "react-bootstrap";
import banner from '../assets/gadgets.jpg'

const AboutUs = () => {
  return (
    <section className="py-5">
      <Container>
        <Row className="align-items-center">
          {/* Left Side - Single Image */}
          <Col md={6}>
            <div className="mb-3">
              <img
  src={banner} // ⬛ No
  quotes alt="gadgets"
  className="img-fluid rounded shadow-sm"
  style={{ maxWidth: "100%", height: "auto" }}
/>

            </div>
```

```
</Col>

{/* Right Side - Text */}
<Col md={6}>
 {/* Smaller SMART MART */}
 <div
  style={{
   color: "red",
   fontWeight: "bold",
   fontSize: "1.2rem",
   letterSpacing: "2px",
  }}
 >
  SMART MART
 </div>

 <h1 className="fw-bold mt-3">ABOUT US</h1>

 {/* Simple intro */}
 <p style={{ lineHeight: "1.6" }}>
  Welcome to <strong>SmartMart</strong>, your trusted shopping app delivering quality products and exceptional service right at your fingertips.
 </p>

 {/* Professional, trustworthy product lines */}
 <p style={{ lineHeight: "1.6" }}>
  Our curated selection of mobiles combines the latest technology with reliable performance to keep you connected. <br />
  Discover stylish watches that blend elegance and functionality for every occasion. <br />
  Explore powerful laptops designed to boost productivity and entertainment. <br />
  Experience premium sound quality with our range of earphones, crafted for comfort and clarity. <br />
```

```
            Enhance your environment with speakers that deliver immersive audio experiences. <br />
            At SmartMart, we prioritize your satisfaction through quality, value, and trust in every
purchase.
          </p>

          <Button
            href="https://your-smart-smart-website-link.com" // Replace with your actual link
            variant="danger" // red background
            className="fw-bold text-white px-4 py-2"
          >
            EXPLORE MORE
          </Button>
        </Col>
      </Row>
    </Container>
  </section>
 );
};

export default AboutUs;
```

- **CART PAGE:-**



CODE:-

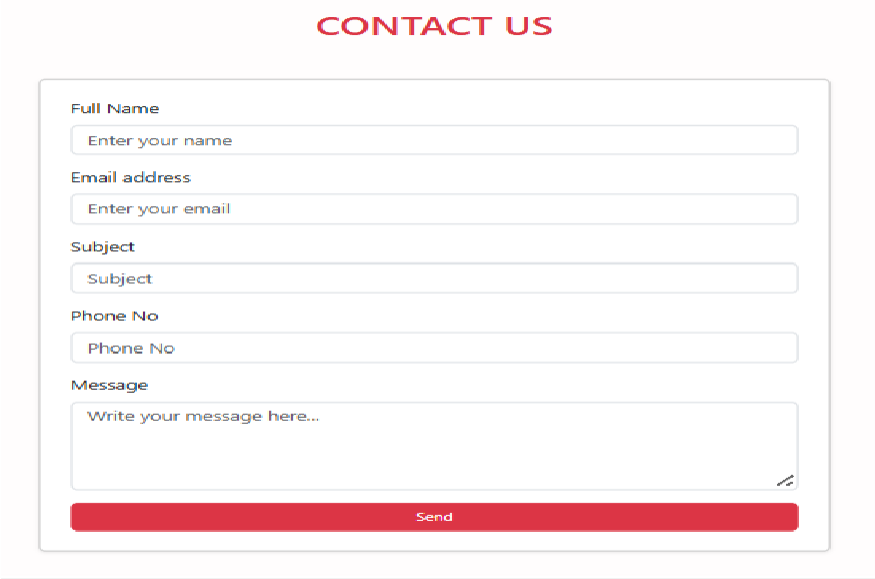
```
// src/pages/CartPage.jsx
import React from 'react';
import { useCart } from '../context/CartContext';
import { Button, Table } from 'react-bootstrap';
import { Link } from 'react-router-dom';

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

  // Calculate totals
  const subtotal = cartItems.reduce((acc, item) => acc + item.price * item.quantity, 0);

  return (
    <div className="container my-5">
      <h2 className="mb-4">Your Shopping Cart</h2>

      {cartItems.length === 0 ? (
        <div className="text-center">
          <h4>Your cart is empty </h4>
          <Link to="/shop" className="btn btn-primary mt-3">Continue Shopping</Link>
```

```
  </div>
 ) : (
  <div className="row">
   {/* Cart Items Table */}
   <div className="col-lg-8 mb-4">
    <Table responsive bordered hover className="text-center align-middle">
     <thead>
      <tr className="bg-light">
       <th>Image</th>
       <th>Product</th>
       <th>Price</th>
       <th>Quantity</th>
       <th>Total</th>
       <th>Remove</th>
      </tr>
     </thead>
     <tbody>
      {cartItems.map(item => (
       <tr key={item._id}>
        <td>
         <img
          src={http://localhost:5000${item.image}}
          alt={item.name}
          style={{ width: '70px', height: '70px', objectFit: 'cover' }}
         />
        </td>
        <td>{item.name}</td>
        <td>₹{item.price.toLocaleString()}</td>
        <td>
         <div className="d-flex justify-content-center align-items-center gap-2">

          <Button size="sm" variant="outline-secondary" onClick={() => updateQuantity(item._id, item.quantity - 1)} disabled={item.quantity <= 1}>−</Button>
```

```
                <span>{item.quantity}</span>
                <Button size="sm" variant="outline-secondary" onClick={() => updateQuantity(item._id, item.quantity + 1)}>+</Button>
              </div>
            </td>
            <td>₹{(item.price * item.quantity).toLocaleString()}</td>
            <td>
              <Button variant="danger" size="sm" onClick={() => removeFromCart(item._id)}>
                Remove
              </Button>
            </td>
          </tr>
        ))}
      </tbody>
    </Table>
  </div>

  {/* Summary Section */}
  <div className="col-lg-4">
    <div className="border rounded p-4 bg-light shadow-sm">
      <h4>Cart Summary</h4>
      <hr />
      <p><strong>Subtotal:</strong> ₹{subtotal.toLocaleString()}</p>
      <p><strong>Shipping:</strong> Free</p>
      <p><strong>Total:</strong> ₹{subtotal.toLocaleString()}</p>

      <Link to="/checkout" className="btn btn-success w-100 mt-3">
        Proceed to Checkout
      </Link>

    </div>
  </div>
```

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

export default CartPage;
```

- **CONTACT PAGE:-**

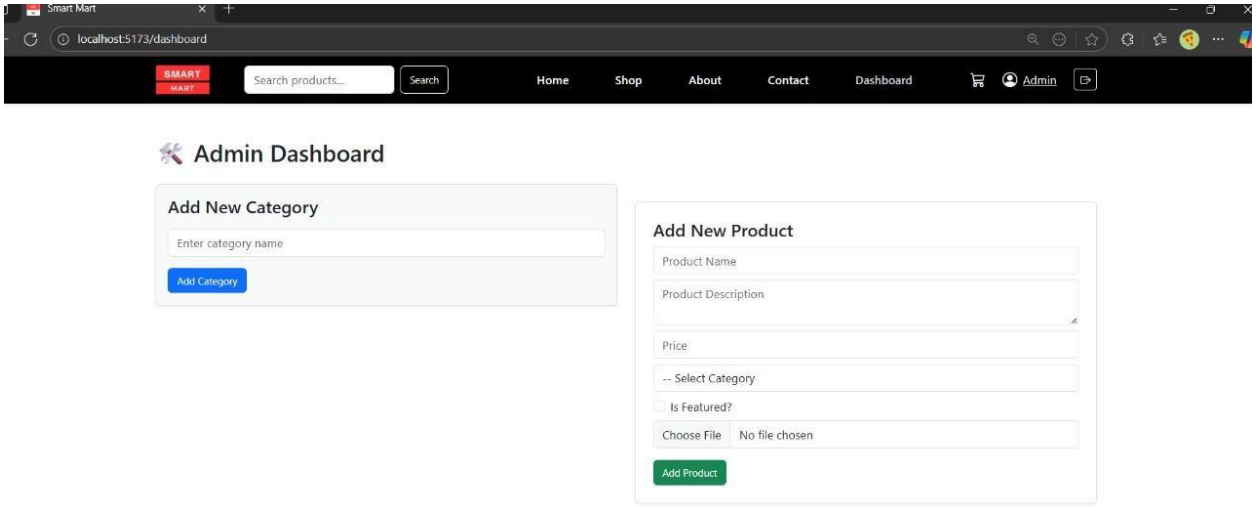


CODE:-

```
import React from 'react';

const Contact = () => {
 return (
  <div style={{ backgroundColor: '#fffdfdff', minHeight: '100vh' }}>
   <div className="container py-5">
    <h2 className="text-center text-danger mb-5">CONTACT US</h2>

    <div className="row justify-content-center">
     <div className="col-md-6">
      <div className="card p-4 shadow-sm">
       <form>
        <div className="mb-3">
         <label htmlFor="name" className="form-label">Full Name</label>
         <input type="text" className="form-control" id="name" placeholder="Enter your name" required />
        </div>

        <div className="mb-3">
         <label htmlFor="email" className="form-label">Email address</label>
         <input type="email" className="form-control" id="email" placeholder="Enter your email" required />
```

```
            </div>

            <div className="mb-3">
              <label htmlFor="subject" className="form-label">Subject</label>
              <input type="text" className="form-control" id="subject" placeholder="Subject" />
            </div>

            <div className="mb-3">
              <label htmlFor="phone" className="form-label">Phone No</label>
              <input type="text" className="form-control" id="phone" placeholder="Phone No" />
            </div>

            <div className="mb-3">
              <label htmlFor="message" className="form-label">Message</label>
              <textarea className="form-control" id="message" rows="4" placeholder="Write your message here..." required></textarea>
            </div>

            <button type="submit" className="btn btn-danger w-100">Send</button>
          </form>
        </div>
      </div>
    </div>
  </div>
 );
};

export default Contact;
```

- **DASHBOARD PAGE:-**

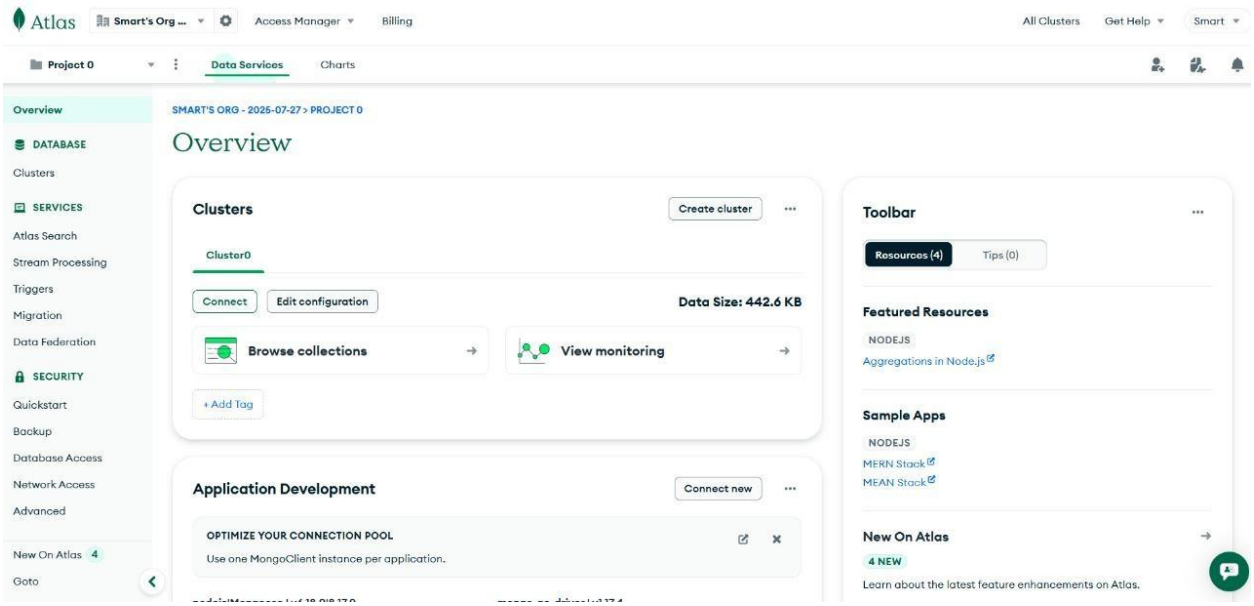


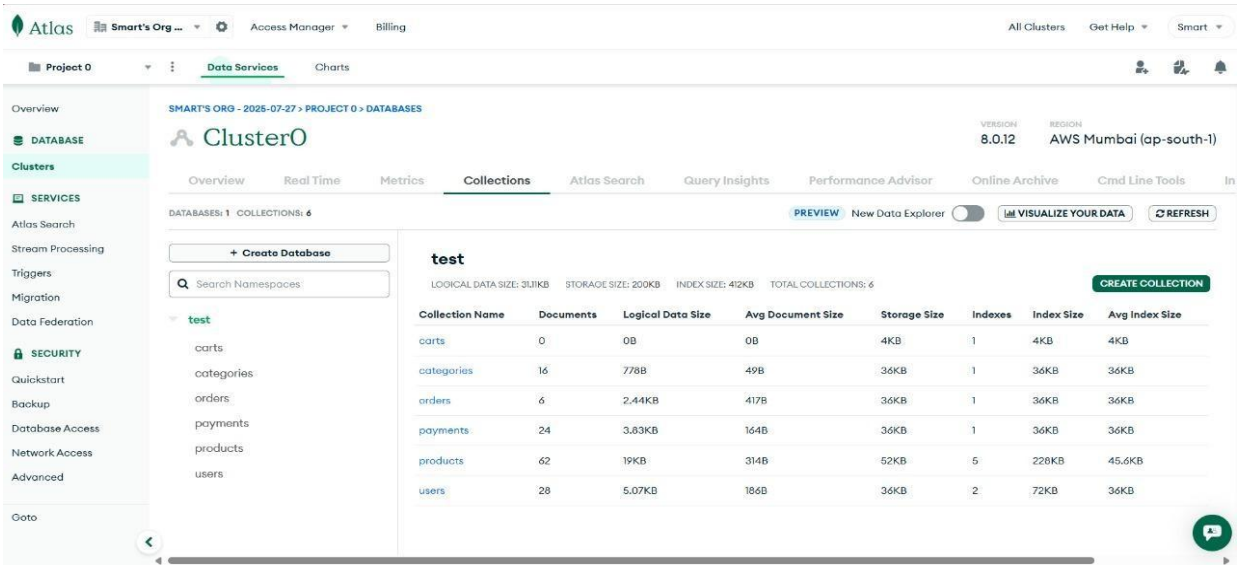
CODE:-

```
import React from 'react';
import AddCategoryForm from '../components/AddCategoryForm';
import AddProductForm from '../components/AddProductForm';

const Dashboard = () => {
  return (
    <div className="container my-5">
      <h2 className="mb-4">🛠 Admin Dashboard</h2>
      <div className="row">
        <div className="col-md-6">
          <AddCategoryForm />
        </div>
        <div className="col-md-6">
          <AddProductForm />
        </div>
      </div>
```

```
    </div>
  );
};

export default Dashboard;
```

6. DATABASE:-


Atlas
Smart's Org ...
Access Manager
Billing
All Clusters
Get Help
Smart
Project 0
Data Services
Charts
Overview
DATABASE
Clusters
SERVICES
Atlas Search
Stream Processing
Triggers
Migration
Data Federation
SECURITY
Quickstart
Backup
Database Access
Network Access
Advanced
New On Atlas 4
Goto
SMART'S ORG - 2025-07-27 > PROJECT 0
Overview
Clusters
Create cluster
Cluster0
Connect
Edit configuration
Data Size: 442.6 KB
Browse collections
View monitoring
+ Add Tag
Application Development
Connect new
OPTIMIZE YOUR CONNECTION POOL
Use one MongoClient instance per application.
Toolbar
Resources (4)
Tips (0)
Featured Resources
NODEJS
Aggregations in Node.js
Sample Apps
NODEJS
MERN Stack
MEAN Stack
New On Atlas
4 NEW
Learn about the latest feature enhancements on Atlas.


Atlas
Smart's Org ...
Access Manager
Billing
All Clusters
Get Help
Smart
Project 0
Data Services
Charts
Overview
DATABASE
Clusters
SERVICES
Atlas Search
Stream Processing
Triggers
Migration
Data Federation
SECURITY
Quickstart
Backup
Database Access
Network Access
Advanced
Goto
SMART'S ORG - 2025-07-27 > PROJECT 0 > DATABASES
Cluster0
VERSION 8.0.12
REGION AWS Mumbai (ap-south-1)
Overview
Real Time
Metrics
Collections
Atlas Search
Query Insights
Performance Advisor
Online Archive
Cmd Line Tools
DATABASES: 1 COLLECTIONS: 6
PREVIEW
New Data Explorer
VISUALIZE YOUR DATA
REFRESH
+ Create Database
Search Namespaces
test
carts
categories
orders
payments
products
users
test
LOGICAL DATA SIZE: 31.1KB
STORAGE SIZE: 200KB
INDEX SIZE: 412KB
TOTAL COLLECTIONS: 6
CREATE COLLECTION
Collection Name | Documents | Logical Data Size | Avg Document Size | Storage Size | Indexes | Index Size | Avg Index Size
carts | 0 | 0B | 0B | 4KB | 1 | 4KB | 4KB
categories | 16 | 778B | 49B | 36KB | 1 | 36KB | 36KB
orders | 6 | 2.44KB | 417B | 36KB | 1 | 36KB | 36KB
payments | 24 | 3.83KB | 164B | 36KB | 1 | 36KB | 36KB
products | 62 | 19KB | 314B | 52KB | 5 | 228KB | 45.6KB
users | 28 | 5.07KB | 186B | 36KB | 2 | 72KB | 36KB

CONCLUSION

In conclusion, the development of an E-Commerce website using the MERN stack demonstrates the effectiveness of modern web technologies in building scalable, responsive, and us er-friendly applications. By leveraging MongoDB for efficient data storage, Express.js and Node.js for robust backend services, and React.js for a dynamic and interactive frontend, this project successfully showcases a full-stack solution capable of handling real-time user interactions, order management, and secure user authentication. This project not only fulfills the basic requirements of an E-Commerce system—such as browsing products, viewing details, placing orders, and tracking deliveries—but also provides a foundation for future enhancements like online payment integration, real-time chat support, and delivery partner tracking. The modular structure and API-driven approach ensure maintainability and scalability, making it well-suited for real-world deployment. Ultimately, this project highlights the potential of MERN STACK applications in solving everyday problems and creating a seamless shopping experience for users.

8. Future Scope & Further Enhancements

- **Future Scope:**

  The future of SMART MART is filled with opportunities to grow, innovate, and deliver an even better shopping experience. Possible areas of development include:

- **AI-Powered Product Recommendations:**

  Implement artificial intelligence to provide personalized shopping suggestions, offers, and deals based on customer preferences, purchase history, and browsing patterns.

- **Gamification in Shopping:**

  Introduce reward points, badges, and interactive challenges to make shopping more engaging and fun for customers.

- **AR/VR Shopping Experience:**

  Integrate augmented and virtual reality to allow customers to virtually try on products such as clothing, accessories, or furniture before purchasing.

- **Offline Shopping Mode:**

  Enable browsing of selected product catalogs without an active internet connection, making the platform more accessible in low-connectivity areas.

- **Multilingual Support:**

  Add support for multiple languages to cater to customers from different regions and countries.

- **Collaborative Shopping Tools:**

  Allow customers to share wishtlist or carts with friends and family, making group shopping easier and more interactive.

- **Mail sending :-**

  The system can be enhanced to automatically send notifications and updates via email to improve communication with users.

9. BIBLIOGRAPHY

1) www.w3schools.com

2) www.youtube.com

3) www.pexels.com

4) www.codepen.ic

5) www.google.com

6) www.googlefont.com

7) www.react.our