# INDEX

### Introduction (1 – 4)

* 1. Introduction of the System
     1. Project Title
     2. Category
     3. Overview
  2. Objectives of the System
  3. Scope of the System
  4. Structure of the System
  5. System Architecture
  6. End Users
  7. Software/Hardware Used for Development
  8. Software/Hardware Required for Implementation

### Software Requirements Specification (SRS) (5 – 8)

* 1. Introduction
  2. Overall Description
     1. Product Perspective
     2. Product Functions
     3. User Characteristics
     4. General Constraints
  3. Special Requirements
  4. Functional Requirements
     1. Module 1: Authentication
     2. Module 2: Music Management
     3. Module 3: Design Constraints
     4. Module 4: System Attributes
  5. Design Constraints
  6. System Attributes
  7. Other Requirements

### System Design (9 – 13)

* 1. Introduction
  2. Assumptions and Constraints
  3. Functional Decomposition
  4. Description of Programs
     1. Context Flow Diagram
     2. Data Flow Diagrams (Level 0, 1, 2)
  5. Description of Components

### Database Design (14 – 17)

* 1. Introduction
  2. Purpose and Scope
  3. Table Definition
  4. ER Diagram

### Detailed Design ..(18 – 21)

* 1. Introduction
  2. Structure of the Software Package
  3. Modular Decomposition
     1. Module 1: Authentication
     2. Module 2: Audi Streaming & Playback
     3. Module 3: Administrative Module

### User Interface (26 – 32)

* 1. Login
  2. Main Screen / Home Page
  3. On-Screen Reports
  4. Alerts
  5. Error Messages

### Testing……………………………..……………………….. (25 – 27)

* 1. Introduction
     1. Unit Testing
     2. Integration Testing
     3. System Testing
  2. Test Reports

1. **Conclusion………………………………………….……(29 - 30)**
2. **Limitations………………………………………....……(31 – 32)**

**10. Scope for Enhancement………….…………………….(33 – 34)**

**11. Abbreviations and Acronyms…………………………(35 – 36)**

**12. Bibliography / References…………………………….(37 – 38)**

**Chapter -1**

**Introduction**

## Introduction

### Introduction of the System

* + 1. **Project Title**

LapHub – Premium E-Commerce Platform for Laptop

### Category

Web Application Development, E-Commerce, Cloud Technology, Payment Processing, and Inventory Management.

### Overview

The demand for laptops has significantly increased due to the growing need for remote work, online education, gaming, and personal use. LapHub is a comprehensive web-based e-commerce platform designed to provide a seamless shopping experience for purchasing laptops online, providing customers with a wide range of options, competitive pricing, and exceptional service. The platform caters to different user segments such as students, professionals, gamers, and businesses, ensuring accessibility and convenience.

It features a responsive interface for both desktop and mobile browsers, ensuring an immersive user experience with advanced filtering, comparison tools, and secure checkout.

### Objectives of the System

* Provide a responsive and user-friendly e-commerce shopping experience for laptop buyers
* Enable customers to search, filter, and compare laptops from various brands and categories
* Support secure payment processing through multiple payment gateways
* Implement an effective order management system with real-time tracking
* Store product information and user data securely in the cloud
* Provide comprehensive product specifications, reviews, and ratings

### Scope of the System

* Targeted for students, professionals, gamers, and business buyers looking for laptops
* Designed for modern web browsers with responsive support across all devices
* Supports Customer, Admin, and System Manager user roles
* Scalable architecture suitable for high-traffic e-commerce scenarios
* Can be extended to mobile platforms (Android and iOS)
* Integration with popular payment providers (Razorpay)

### Structure of the System

1. Authentication Module – Secure login for users with role-based access.
2. Product Management Module – Admin can add, update, or delete songs, albums, and playlists.
3. Shopping Cart Module – Real-time synchronized playback with queue management.
4. Order Management Module – Cloud-based media storage for fast and reliable streaming
5. Payment Processing– Responsive web design for all devices.
6. Admin Dashboard – Manage products, categories, inventory, and orders

### System Architecture

The LapHub architecture follows a modern full-stack web model with clear separation between frontend, backend.

### Main Layers:

##### Client Layer (Frontend)

* + Built using React and Vite for dynamic, responsive interfaces
  + Tailwind CSS for modern, responsive design
  + Redux for state management

##### Server Layer (Backend)

* + Node.js with Express manages API requests, authentication, and payment processing
  + RESTful API architecture for scalability

##### Dashboard UI

* + When a user visits the dashboard, they will experience a professional and visually appealing interface.

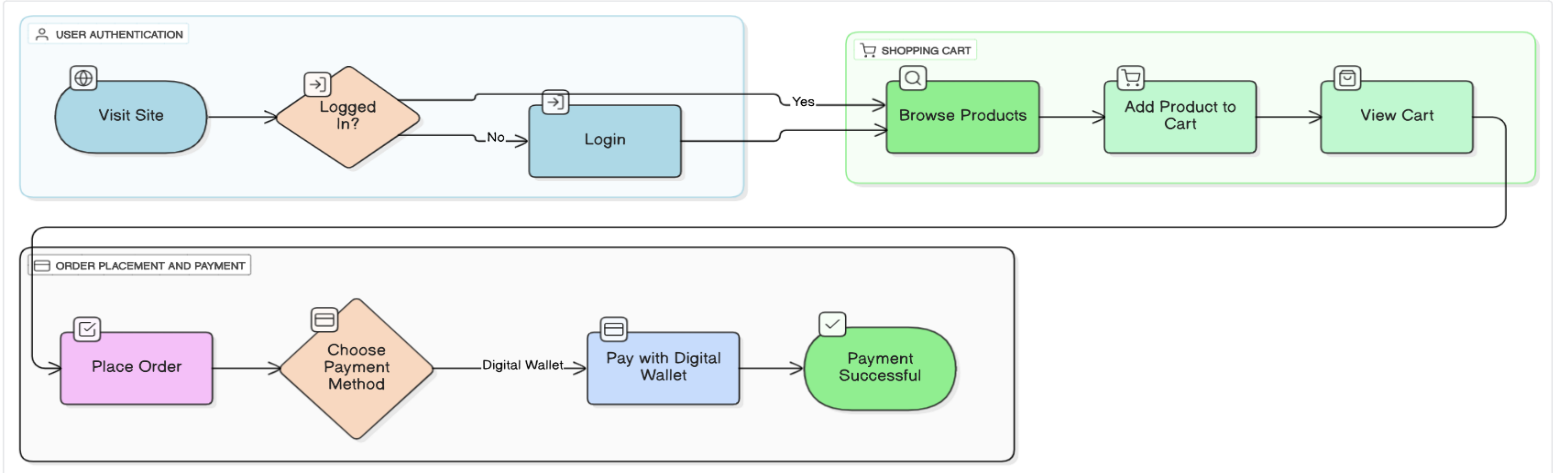
##### Database Layer

* + MongoDB handles all data storage for users, products, orders, categories, and payments

##### Payment Layer

* + Razorpay integration.
  + Secure payment processing and webhook handling

##### Authentication Layer

* + JWT-based authentication for secure user sessions
  + bcryptjs for password hashing

### End Users

* Customers – Browse, purchase, and track orders in real time
* Administrators– Manage products, categories, inventory, and orders
* System Managers – Monitor server, database performance, and system health
* Guest Users – Browse products without account creation.

### Software/Hardware Used for Development

##### Software:

##### React.js with Vite for fast builds

##### Tailwind CSS for responsive design

##### Node.js with Express.js backend

##### MongoDB 6.x or MongoDB Atlas

##### Razorpay and Stripe API integration

##### JWT authentication

##### WebSocket support (for future real-time features)

##### Hardware:

* Minimum Intel Core i5 (8th Gen or higher) / AMD Ryzen 5 equivalent
* Windows/Mac/Linux Laptop/Desktop
* 8 GB RAM minimum for development
* Stable high-speed internet connection (10 Mbps minimum)
* SSD storage (256 GB minimum)

### Software/Hardware Required for Implementation

##### Software:

* Node.js runtime environment
* MongoDB Atlas database
* JWT for Secure Authentication
* Web browsers (Chrome, Firefox, Edge)

##### Hardware:

* Server Requirements: 2 – 4 CPU cores, 4–8 GB RAM, SSD storage
* Client Requirements: Any device with 2 GB RAM, audio output,
* Stable internet connection.

**Chapter -2**

**SRS - Software Requirements Specifications**

## Software Requirement Specification (SRS)

### Introduction

This Software Requirement Specification (SRS) defines the functional and non-functional requirements of the LapHub e-commerce system. It explains how the system operates under various conditions, including user interactions, software dependencies, and system constraints. The SRS serves as a guide for design, development, and testing phases. It ensures a common understanding among developers, testers, and stakeholders regarding system objectives, performance expectations, and scalability.

### Overall Description

* + 1. **Product Perspective**

LapHub is a full-stack web application that provides a complete e-commerce platform for buying and selling laptops and computer hardware. It integrates React (frontend), Node.js/Express (backend), MongoDB (database), Razorpay/Stripe (payment processing), and Cloudinary (media storage) to deliver a seamless shopping experience with secure transactions, inventory management, and admin control.

### Product Functions

* User Authentication & Authorization – Secure signup, login, and role-based access control
* Product Catalog – Browse products with detailed specifications, images, and pricing
  + Search & Filtering – Advanced search and filtering by category, price, specifications, and ratings
* Shopping Cart – Add, remove, update quantities, and manage items before checkout
* Order Management – Place orders, track status, and view order history
  + Payment Processing – Secure payment through Razorpay and Stripe with webhook integration
* User Profile Management – Manage personal information, addresses, and preferences
* Admin Dashboard – Full control over products, categories, inventory, orders, and users
* Product Reviews & Ratings – Users can leave reviews and ratings for purchased products
* Responsive Design – Optimized for mobile, tablet, and desktop browsers

### User Characteristics

* Customers – Tech-savvy and casual users who browse and purchase products online
* Repeat Customers – Users with saved addresses, payment methods, and order history
* Admin Users – Product managers who add, edit, and manage product catalogs
* Administrators – Full system access for managing users, orders, and system settings
  + System Admins – Handle server deployment, database management, and performance monitoring

### General Constraints

* Requires stable internet connection for browsing and checkout
* Depends on third-party services (Razorpay, Stripe, Cloudinary, MongoDB Atlas)
* Payment processing requires PCI DSS compliance
* Image uploads limited by Cloudinary plan
* Performance may vary with user device and network speed
* Inventory updates must be in real-time to prevent overselling

### Special Requirements (Software/Hardware – if any) Software:

* React.js with Vite for fast builds
* Tailwind CSS for responsive design
* Node.js with Express.js backend
* MongoDB 6.x or MongoDB Atlas
* Razorpay API integration
* JWT authentication

### Hardware:

* Minimum Intel Core i5 (8th Gen or higher) / AMD Ryzen 5 equivalent
* Windows Laptop/Desktop/Mobile
* Stable internet connection

### Functional Requirements

* + 1. **Module 1: Authentication & User Management**
* User registration with email verification
* Secure login with JWT token-based sessions
* Role-based access control (Customer, Admin)
* JWT-based session management and validation
* Profile edits, name and image
* Logout and token refresh system

### 2.4.1 Module 2: Product Management

* Admin can create, read, update, and delete products (laptops)
* Product categorization with hierarchical categories (By Usage, By Price, By Brand)
* Product specifications (Processor, RAM, Storage, Graphics, Display, Battery, etc.)
* Image upload with Cloudinary integration (thumbnail, primary, gallery images)
* Advanced filtering by specifications and category

### Module 3: Shopping Cart & Checkout

* Add/remove products from cart
* Update product quantities
* Calculate subtotal, taxes, and shipping
* Real-time cart updates

### Module 4: Payment Processing

* **Integrate Razorpay for payments (UPI, Net Banking, Card, Wallet)**
* **Payment status tracking and confirmation**
* **Multiple payment method support for flexibility**
* **Payment failure handling and retry mechanism**

**Chapter-3**

**System Design (Functional Design)**

## System Design

### Introduction

System design for LapHub defines how the e-commerce platform operates, integrates, and delivers a seamless shopping experience. The architecture ensures secure, scalable, and responsive interaction through modular components that handle user authentication, product management, order processing, and payment handling. The system focuses on performance optimization and user experience—managing product catalogs, delivering secure payments, and enabling efficient admin control. It uses MongoDB for data storage, Razorpay/Stripe for payments, and Cloudinary for media delivery. The design supports smooth transaction processing while maintaining optimal performance and data security.

### Assumptions and Constraints Assumptions:

* Users have stable internet connection for browsing and checkout
* Modern browsers with JavaScript enabled and cookie support
* Users are familiar with basic e-commerce interfaces
* Payment gateways are always available and responsive
* Cloudinary service is available for image storage and delivery
* MongoDB Atlas is accessible and maintains uptime

### Constraints:

* Dependent on external services (Razorpay, Stripe, Cloudinary, MongoDB Atlas)
* Payment processing subject to PCI DSS compliance
* File upload size restrictions based on Cloudinary plan
* API rate limiting from payment providers
* Concurrent user load dependent on server capacity
* Image quality limited by bandwidth and device capabilities
* International payments subject to regional regulations

### Functional Decomposition

The system is divided into the following major modules:

##### Authentication & User Management

* + Secure login/signup with JWT tokens
  + User profile management and role-based access control
  + Session management with token refresh

1. **Product Catalog & Discovery**
   * Browse products with specifications and reviews
   * Product recommendations and featured items
2. **Shopping Cart & Checkout**
   * Cart management with quantity updates
   * Shipping and tax calculation

##### Payment Processing

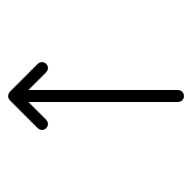
* + Razorpay integration for Indian payments
  + Secure payment gateway communication

### Description of Program:

* + 1. **Context Flow Diagram (CFD) Entities:**
* Customers (Primary Users)
* Administrators (System Managers)
* Payment Gateways (Razorpay)
* Cloudinary (Image Storage & Delivery)
* MongoDB Database (Data Storage)

### System Block:

* User Authentication & Authorization Management
* Product Catalog & Search System
* Shopping Cart & Checkout Management
* Payment Processing & Security
* Order Management System
* Administrative Content Management Portal



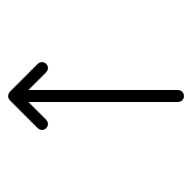
User

Authentication &

User Management

LapHub System

Admin



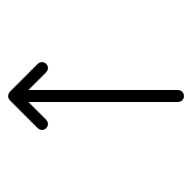
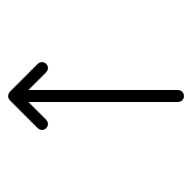
Cart/Order

Manage Products

Manage Orders

### Data Flow Diagrams (DFDs)

* **Level 0**



LAPHUB

User

Database

### Level 1

Authentication

User

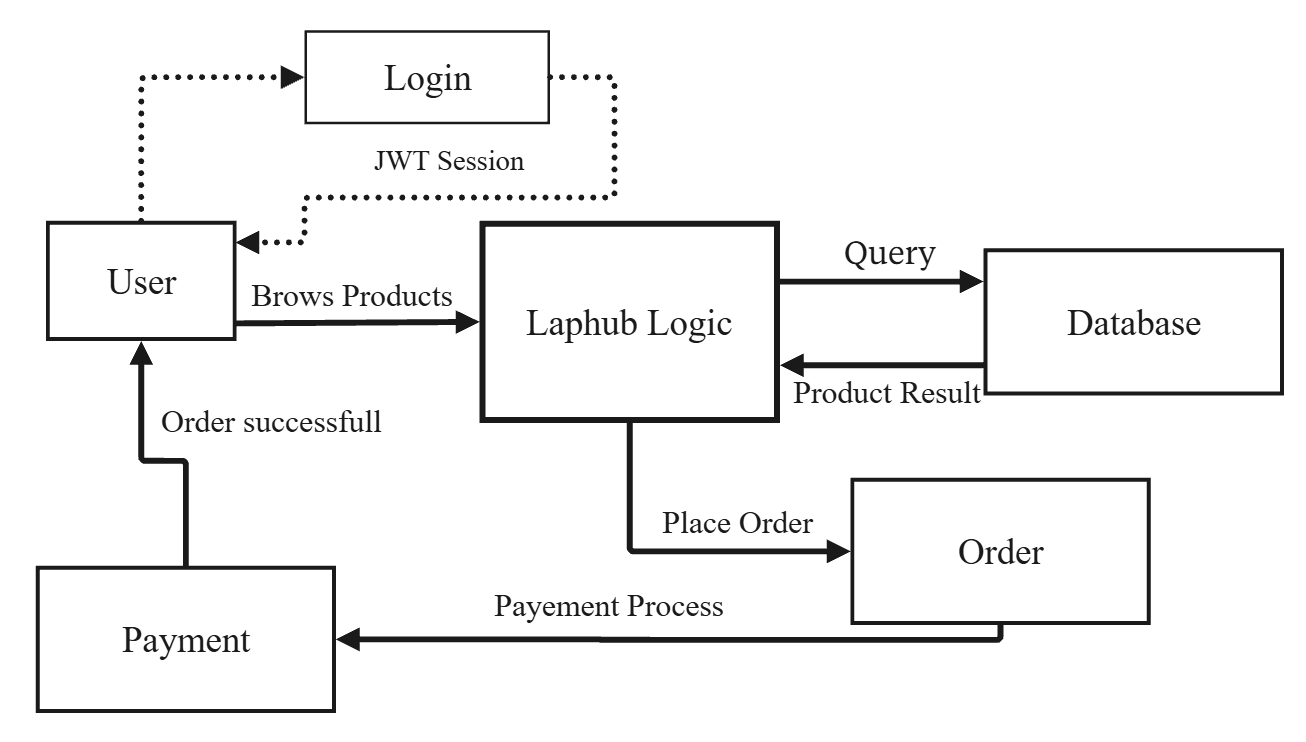
Product Browsing

Order confirmation

Add to cart /Checkout

### Level 2

### 



### Description of Components

* + 1. **Functional Component 1: Authentication**
* Purpose: Secure user identity verification and session management
* Technology: JWT tokens, bcryptjs, Express middleware
* Input: User credentials (email/password), authentication requests
* Output: Authenticated sessions, user profiles, access tokens
* Implementation: HTTPS communication, secure token storage, session validation

### Functional Component 2: Product Catalog Management

* Purpose: Organize and deliver product information with search capabilities
* Technology: MongoDB with Mongoose ODM, Cloudinary for images
* Input: Search queries, filter parameters, product requests
* Output: Product lists, product details, filtered results, recommendations
* Implementation: Database indexing, caching, image optimization

### Functional Component 3: Shopping Cart & Checkout

* Purpose: Manage user shopping sessions and checkout process
* Technology: MongoDB, Express.js
* Input: Product additions/removals, quantity updates, checkout requests
* Output: Cart totals, order confirmation, invoice generation
* Implementation: Real-time updates, persistent storage, validations

### Functional Component 3: Payment Processing

* Purpose: Secure handling of payments and transaction management
* Technology: Razorpay API, Stripe API, Webhooks, JWT
* Input: Payment information, order details, customer data
* Output: Payment confirmation, transaction records, receipts

### Order Management

* Purpose: Provide admin control over products, orders, and users
* Technology: Express.js API, MongoDB, Cloudinary, Node.js
* Input: Media files, product metadata, administrative commands
* Output: Updated catalog, user management reports, analytics
* Implementation: Role-based access control, audit logs, secure operations

**Chapter-4**

**Database Design**

### Database Design

* 1. **Introduction**

The database design of LapHub securely manages user credentials, product catalogs, orders, payments, and inventory using MongoDB. It supports role-based access control for customers and administrators, enables efficient product discovery and order tracking, and maintains data integrity while supporting real-time updates. The design integrates seamlessly with Cloudinary for media storage, payment gateways for secure transactions, and provides optimized query performance for fast user experience.

### Purpose and Scope Purpose:

* Securely store and manage user profiles, products, categories, orders, and payments
* Provide real-time access for browsing, shopping, and order tracking
* Enable efficient product discovery with advanced searching and filtering
* Maintain system performance with scalable indexing and caching
* Support admin operations for inventory and order management

### Scope:

* Database used by the LapHub web application

##### Stores:

* User profiles and authentication metadata
* Products, categories, and specifications
* Shopping carts and order details
* Payment information and transaction logs
* Order history and invoices
* Product reviews and ratings
* Administrative records and activity logs

The system can later be extended for **AI-driven recommendations**, **advanced analytics**, and **cross-platform data synchronization**.

* 1. **Table Definition**
     1. **Users Model**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id (PK) | INTEGER | Unique identifier for doctor |
| name | TEXT | User’s full display name |
| role | TEXT | Interviewer Or Candidate |
| email | TEXT | User’s email address |
| Password | TEXT | Hashed Password |
| Phone | TEXT | Phone Number |

* + 1. **Categories Model**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| \_id | ObjectId (PK) | Unique identifier for category |
| name | TEXT | Category name |
| artist | TEXT | Album artsit |

* + 1. **Products Model**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id | ObjectId (PK) | Unique Id for Product |
| name | TEXT | Product name |
| Description | TEXT | Product description Name |
| imageUrl | TEXT | Product image url |
| Review | TEXT | Review of the product |
| CateogoryID | ObjectId | Category id |
| |  | | --- | | Ratings |  |  | | --- | |  | | NUMBER | |  | | --- | | Rating of the Product |  |  | | --- | |  | |

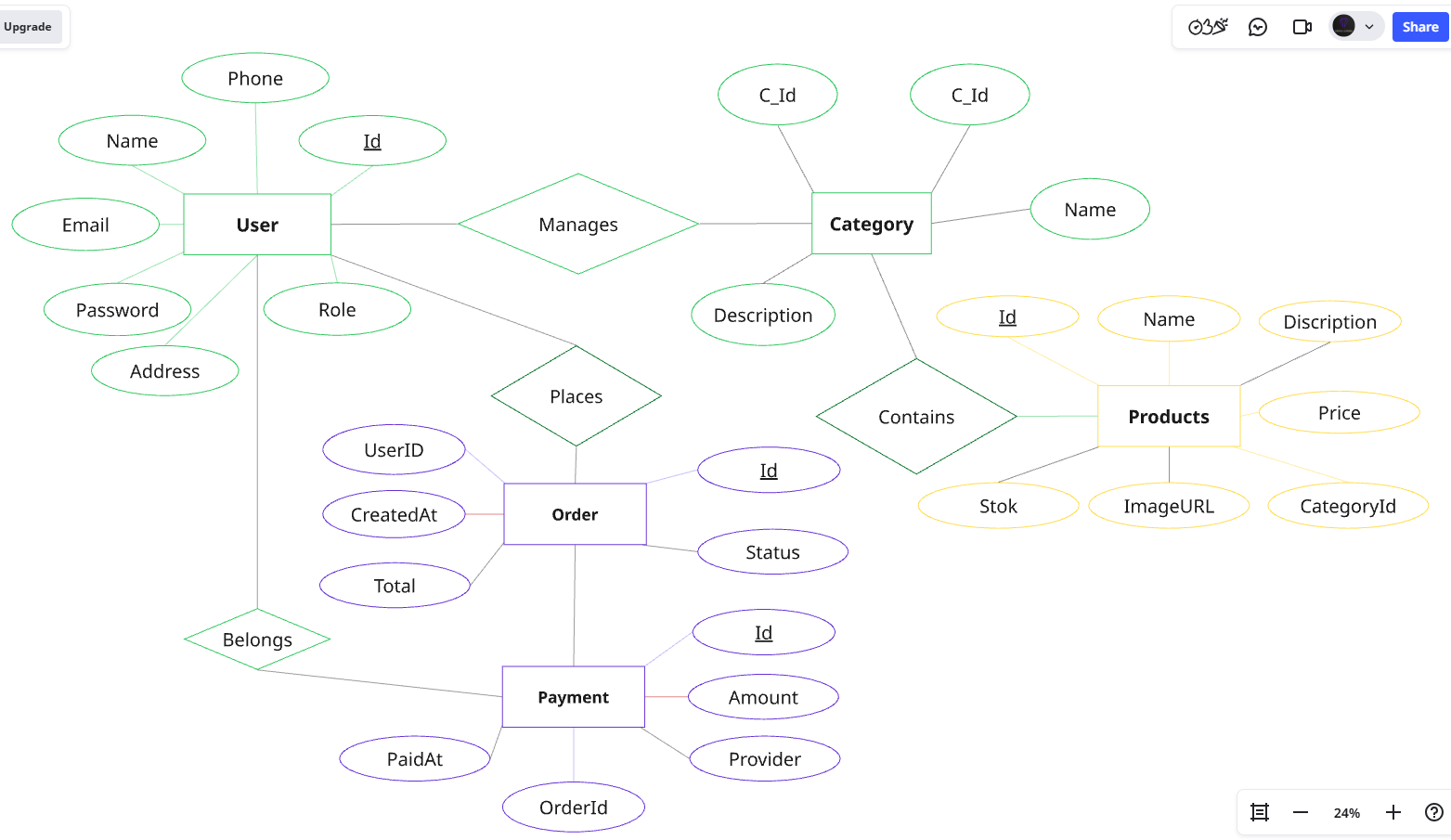
**4.3.4 Orders Model**

|  |  |  |
| --- | --- | --- |
| **Field name** | **Data Type** | **Description** |
| Id | ObjectId | Unique id |
| UserId | ObjectId | User id |
| Items | ARRAY | Item list |
| Total | NUMBER | Total of Price |
| Status | STRING | Current status of the product |

**4.3.5 Payment Model**

|  |  |  |
| --- | --- | --- |
| **Field name** | **Data Type** | **Description** |
| Id | ObjectId | Unique id |
| Amount | ObjectId | User id |
| Provider | STRING | Payment Provider |
| OrderId | NUMBER | Order Id for the payment |
| PaidAt | STRING | Date and time of the payment |

* 1. **ER Diagram**

****

**Chapter-5**

**Detailed Design**

### Detailed Design

* 1. **Introduction**

The detailed design of LapHub focuses on the internal logic of each functional module. This includes input handling, API interaction, payment processing, and output generation. Each module performs a specific function such as authentication, catalog management, shopping, payment processing, or content administration. The modular design ensures maintainability, scalability, and flexibility for future feature extensions.

### Structure of the Software Package (Structure Chart) Main Application

**Main Application (Interview Management System)**

LapHub E-Commerce Application

│

├── Authentication & Authorization Module (Module 1)

│ ├── User Registration & Login

│ ├── JWT Token Management

│ ├── Role-Based Access Control (RBAC)

│ └── Password Reset & Recovery

│

├── Product Management Module (Module 2)

│ ├── Product CRUD Operations

│ ├── Category Management

│ ├── Inventory Tracking

│ ├── Image Upload (Cloudinary)

│ └── Product Specifications

│

├── Product Discovery Module (Module 3)

│ ├── Search Functionality

│ ├── Advanced Filtering

│ └── Rating & Reviews

│

├── Shopping Cart Module (Module 4)

│ ├── Add/Remove Items

│ ├── Quantity Management

│ ├── Cart Calculations

│

├── Checkout & Order Module (Module 5)

│ ├── Multi-step Checkout

│ ├── Shipping Address Management

│ ├── Order Confirmation

│ └── Order History Tracking

│

├── Payment Processing Module (Module 6)

│ ├── Razorpay Integration

│ ├── Payment Validation

│

├── Admin Dashboard Module (Module 7)

│ ├── Product Management UI

│ ├── Order Management

│ ├── User Management

* 1. **Modular Decomposition of the System**
     1. **Module 1: Authentication Management**
        1. **Inputs**
           + User credentials, registration data, OAuth tokens

### Procedural Details

* + - * + Hash passwords, generate JWT tokens, validate credentials.

### File I/O Interfaces

* + - * + MongoDB Users collection, JWT libraries

### Outputs

* + - * + Authentication tokens, user sessions, access confirmation

### Implementation Aspects

* + - * + bcryptjs for hashing, JWT for tokens, HTTPS for transmission

### Module 2: Product Management

* + - 1. **Inputs**
         * Product data, images, specifications, category info

### Procedural Details

* + - * + Validate data, upload images, store product details.

### File I/O Interfaces

* + - * + MongoDB Products collection, Cloudinary API.

### Outputs

* + - * + Product ID, confirmation response, image URLs

### Implementation Aspects

* + - * + Data validation, image optimization, database indexing

### Module 3: Payment Processing

### Inputs

* + - * + Order data, payment credentials, amount

### Procedural Details

* + - * + Validate payment, communicate with gateway, log transaction.

### File I/O Interfaces

* + - * + Razorpay API, MongoDB Payments.

### Outputs

* + - * + Payment confirmation, receipt, transaction record.

### Implementation Aspects

* + - * + SSL encryption, webhook handling.

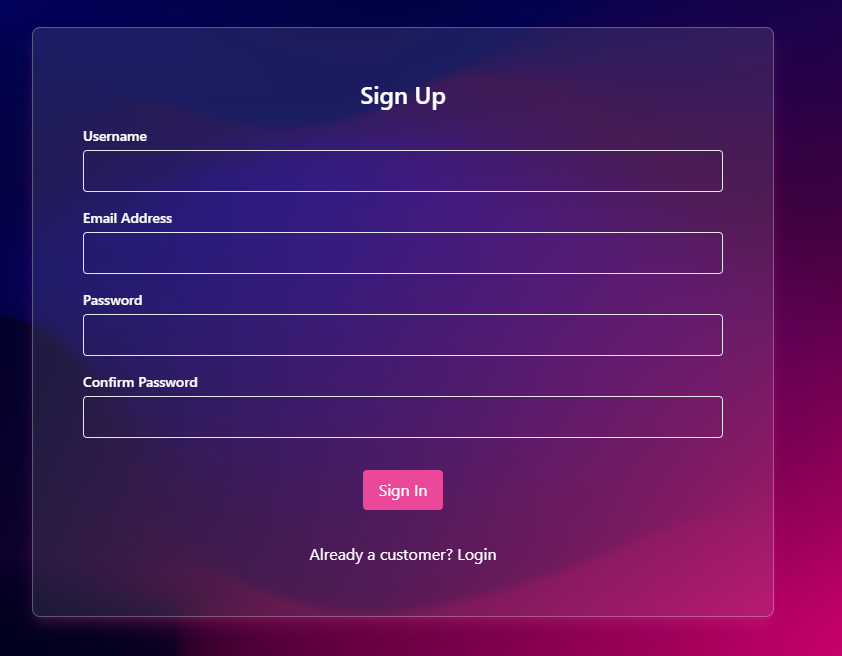
**Chapter-6**

**User Interface**

# User Interface

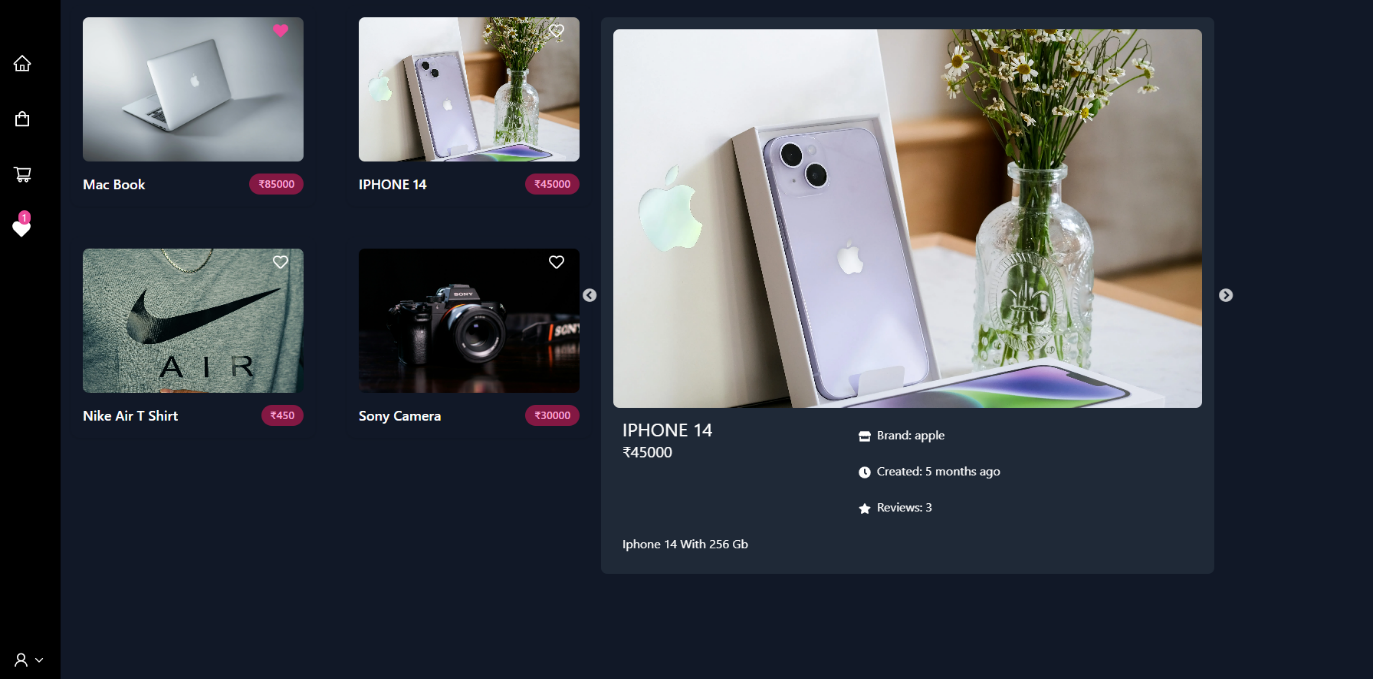
#### Login & Registration Page

* + Clean, modern login form with email and password fields
  + New user registration link with sign-up form
  + Option for new users to register
  + Secure authentication using encryption (HTTPS, password hashing)



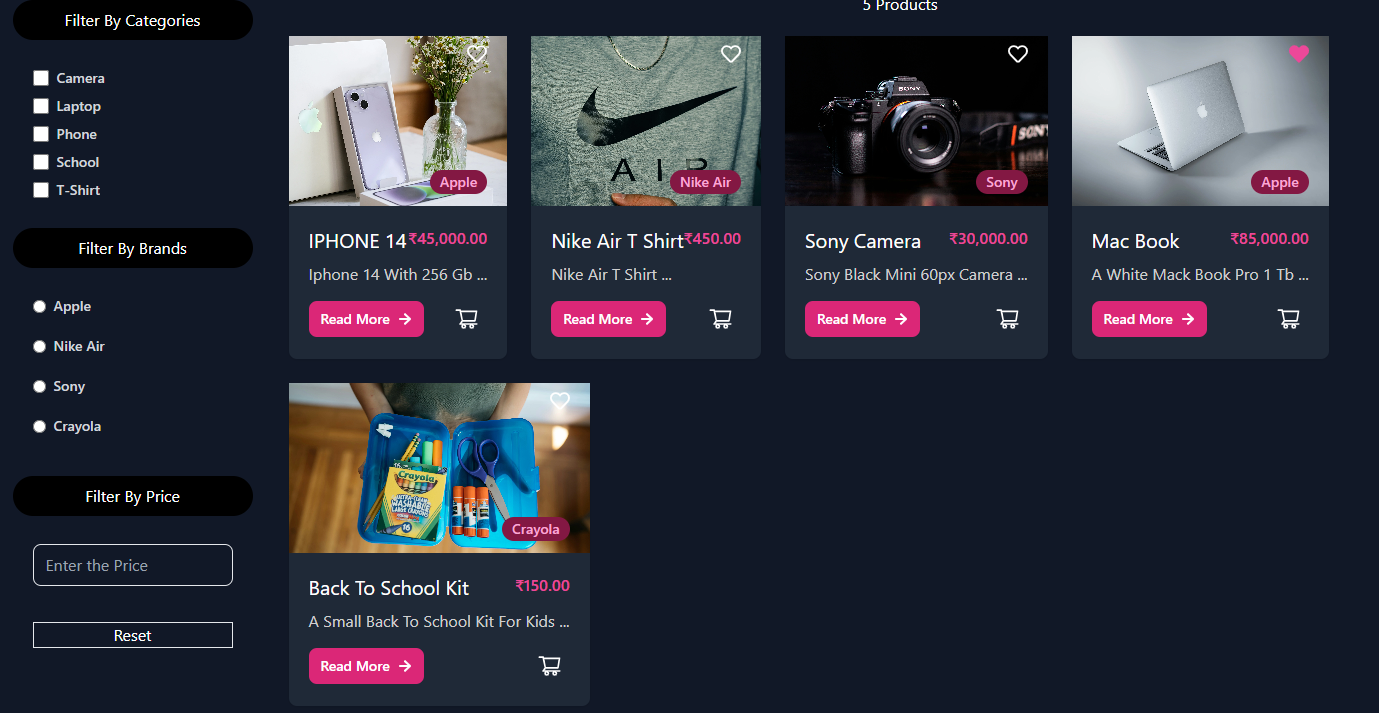
### Main Screen / Home Page

The homepage provides a personalized greeting for logged-in users, creating a welcoming and engaging experience. It features a carousel of top or featured products, allowing users to easily browse through the latest or most popular items. A trending products section highlights what’s currently in demand, while a category navigation menu helps users quickly explore different product types. The search bar with autocomplete ensures fast and accurate searches, enhancing usability. Each item is displayed in visually appealing product cards that include images, prices, and ratings for easy comparison..



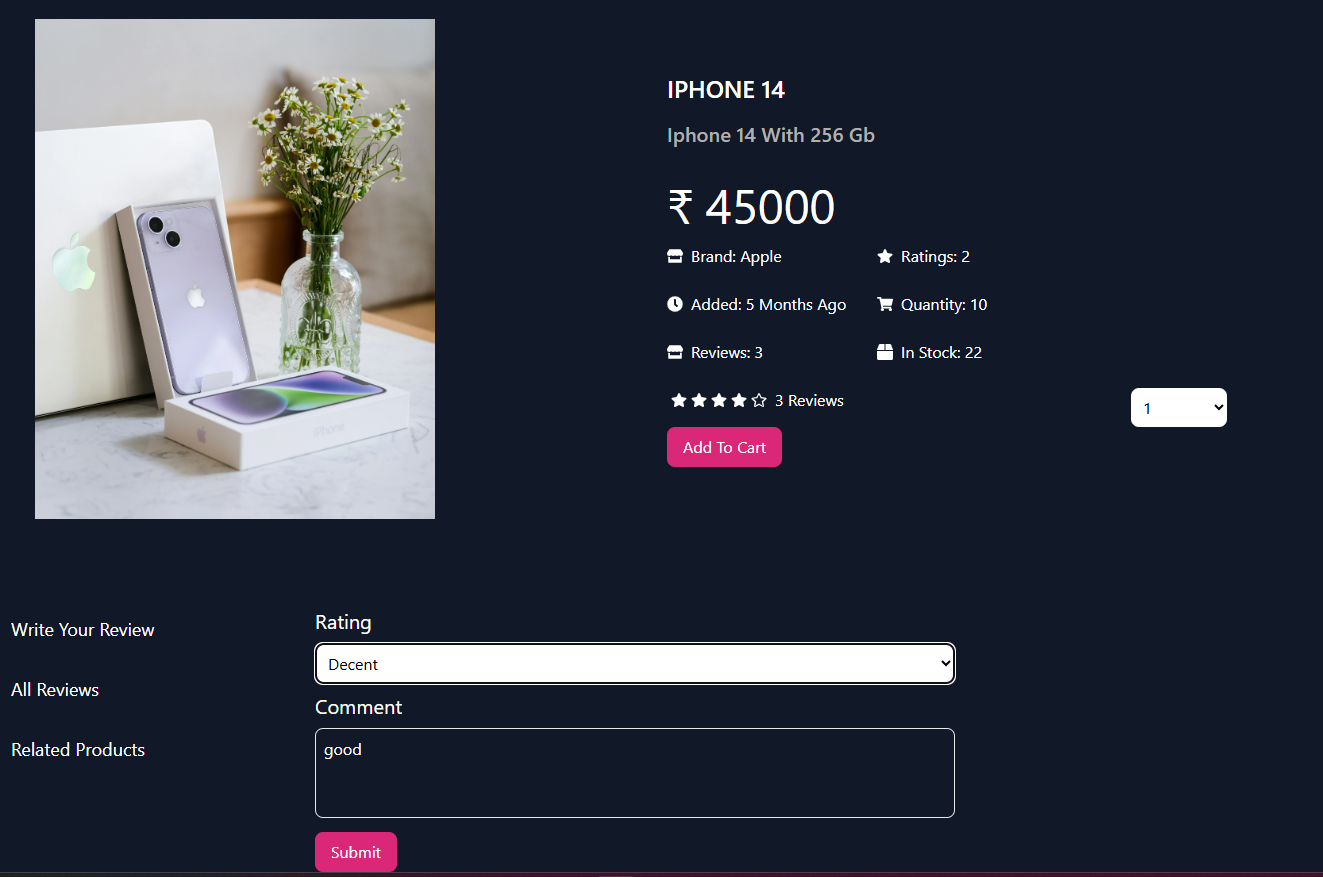
### Shop Management

The product page includes a grid/list view toggle, advanced filters by usage, price, brand, processor, RAM, storage, and graphics. Users can sort by price, rating, or popularity and navigate using pagination or infinite scroll. Each product card displays the image, name, price, and key specs, with quick view and Add to Cart options. Features like search term highlighting, filter count display, and result count enhance the browsing experience.



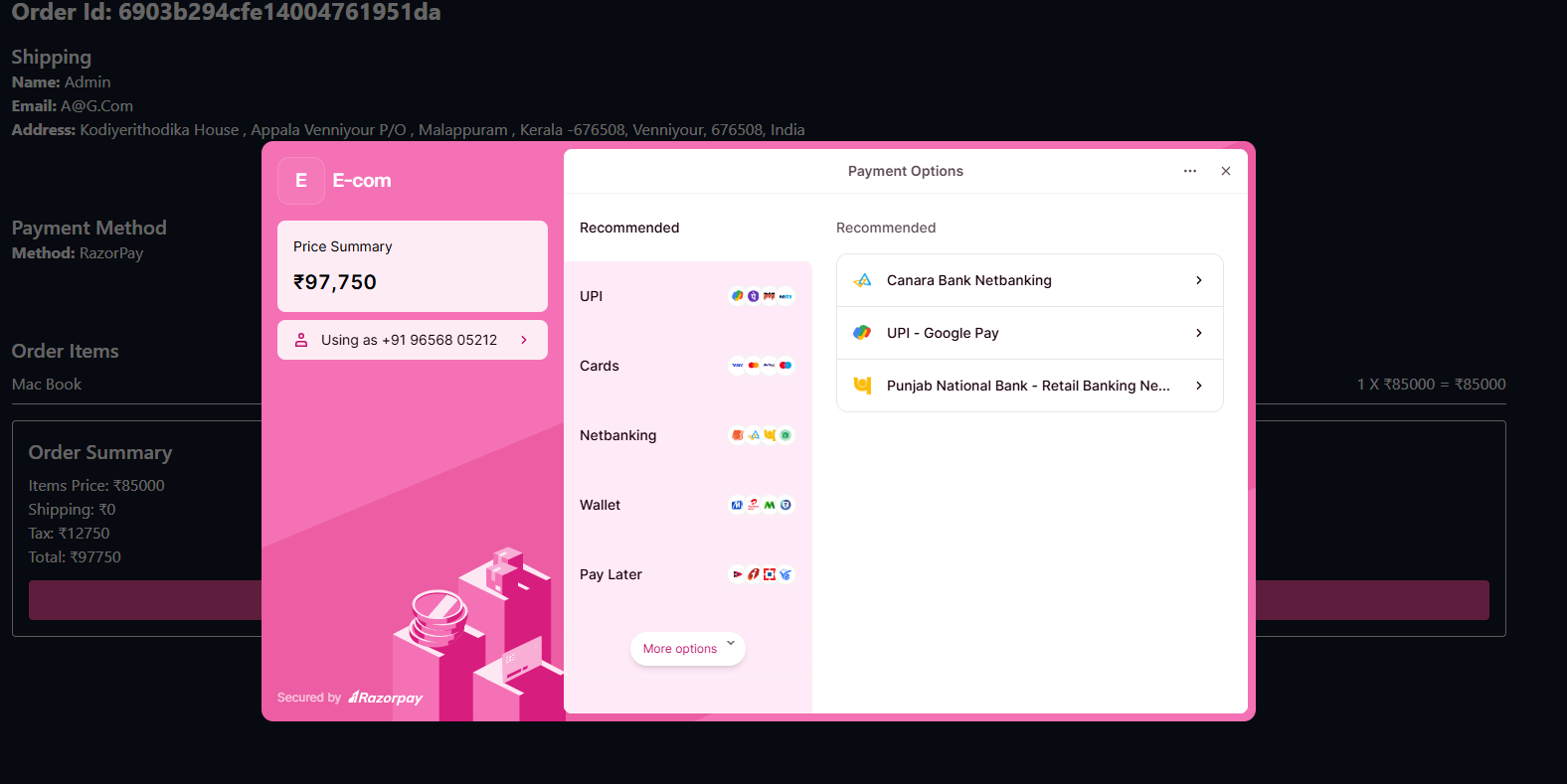
### Product Detail Page

The product detail page displays complete **laptop information**, including specifications, pricing, and availability. Users can easily **add the product to their cart** for purchase. A dedicated **reviews section** shows customer feedback and ratings. Additionally, users have the option to **add their own reviews**, making the experience more interactive and information.



### 6.5 Payment

In the payment step of the checkout process, users can securely complete their purchase through Razorpay integration. The interface allows customers to select Razorpay as their preferred payment method and proceed with multiple options, including credit/debit cards, UPI, net banking, and wallets. The system ensures encrypted and seamless transactions, providing instant confirmation upon successful payment. Once processed, users are automatically directed to the order review and confirmation page, ensuring a smooth and reliable checkout experience.



**Chapter-7 Testing**

### Testing

* 1. **Introduction**

Testing ensures the LapHub e-commerce system works correctly, securely, and efficiently. We test each module individually (unit testing), check interactions between modules (integration testing), and evaluate the full system in real-world scenarios (system testing). Edge cases like invalid logins, payment failures, out-of-stock items, and concurrent orders are also tested.

### Unit Testing

Each module of the LapHub system is individually tested:

##### Authentication: Verify login, registration, password reset, JWT generation.

##### Product Management: Test CRUD operations, image uploads, specification storage.

##### Cart Operations: Add/remove items, quantity updates, calculations.

##### Payment Processing:  Payment validation, transaction logging, error handling.

##### Admin Operations:  Product management, user management, order fulfillment.

##### User Profile: Profile updates, address management, preference settings.

### Integration Testing

Modules were tested together to ensure data flowed correctly:

* Authentication → Product Browsing → Cart → Checkout → Payment → Order Confirmation
* Admin Login → Product Upload → Product Display → Inventory Update
* User Registration → Profile Creation → Address Management → Order Placement
* Payment Processing → Order Update → Notification → Invoice Generation
* Verified modules work together and data flows correctly across system

### System Testing

Full system evaluation in realistic e-commerce scenarios:

* Complete user journey from registration to order delivery
* Admin workflows for product management and order fulfillment
* Payment gateway integration with real transactions
* Inventory management across concurrent orders
* Real-time stock updates and availability
* Database integrity and consistency
* Performance under load testing (concurrent users)

### Test Reports

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Result** | **Status** |
| **TC\_001** | **User registration** | **Account created successfully** | **Pass** |
| **TC\_002** | **User login** | **User authenticated with valid JWT** | **Pass** |
| **TC\_003** | **Invalid login** | **Error message displayed** | **Pass** |
| **TC\_004** | **Browse products** | **Products loaded from database** | **Pass** |
| **TC\_005** | **Search products** | **Matching products displayed** | **Pass** |
| **TC\_006** | **Filter products** | **Filtered results displayed correctly** | **Pass** |
| **TC\_007** | **Add to cart** | **Item added to cart** | **Pass** |
| **TC\_008** | **Remove from cart** | **Item removed from cart** | **Pass** |
| **TC\_009** | **Update cart quantity** | **Quantity updated correctly** | **Pass** |
| **TC\_010** | **Apply discount** | **Discount calculated correctly** | **Pass** |
| **TC\_011** | **Proceed to checkout** | **Checkout form displayed** | **Pass** |
| **TC\_012** | **Payment processing (Razorpay)** | **Payment successful** | **Pass** |
| **TC\_013** | **Payment processing (Stripe)** | **Payment successful** | **Pass** |
| **TC\_014** | **Order creation** | **Order stored in database** | **Pass** |
| **TC\_015** | **Order confirmation email** | **Email sent to user** | **Pass** |
| **TC\_016** | **View order history** | **User’s orders displayed** | **Pass** |
| **TC\_017** | **Admin product upload** | **Product added to catalog** | **Pass** |
| **TC\_018** | **Admin order management** | **Order status updated** | **Pass** |
| **TC\_019** | **Admin user management** | **Users listed and managed** | **Pass** |
| **TC\_020** | **Out of stock handling** | **Product unavailable for purchase** | **Pass** |
| **TC\_021** | **Responsive design (Mobile)** | **Layout adjusted for mobile** | **Pass** |
| **TC\_022** | **Responsive design (Tablet)** | **Layout adjusted for tablet** | **Pass** |
| **TC\_023** | **Cross-browser (Chrome)** | **Works correctly** | **Pass** |
| **TC\_024** | **Cross-browser (Firefox)** | **Works correctly** | **Pass** |
| **TC\_025** | **Session timeout** | **User logged out after inactivity** | **Pass** |
| **TC\_018** | **Admin order management** | **Order status updated** | **Pass** |
| **TC\_019** | **Admin user management** | **Users listed and managed** | **Pass** |
| **TC\_020** | **Out of stock handling** | **Product unavailable for purchase** | **Pass** |
| **TC\_021** | **Responsive design (Mobile)** | **Layout adjusted for mobile** | **Pass** |
| **TC\_022** | **Responsive design (Tablet)** | **Layout adjusted for tablet** | **Pass** |
| **TC\_023** | **Cross-browser (Chrome)** | **Works correctly** | **Pass** |
| **TC\_024** | **Cross-browser (Firefox)** | **Works correctly** | **Pass** |
| **TC\_025** | **Session timeout** | **User logged out after inactivity** | **Pass** |

**Chapter - 8**

**Conclusion**

1. **Conclusion**

The LapHub E-Commerce Platform efficiently integrates authentication, product management, shopping, payment processing, and order management into a comprehensive web-based platform. It streamlines the shopping experience for customers while providing administrators with powerful tools to manage inventory and orders.

With secure JWT authentication and role-based access, customers can browse and purchase with confidence while administrators maintain full control over the platform. Responsive design ensures seamless experience on all devices, while Razorpay and Stripe integration enables secure, flexible payment options.

The modular architecture of LapHub makes it scalable and adaptable for future integrations with mobile apps, advanced recommendation systems, loyalty programs, or marketplace features. The platform is built to handle high-traffic scenarios while maintaining performance and security standards.

Key Achievements:

* Secure, scalable e-commerce platform
* Seamless user experience across all devices
* Flexible payment processing with multiple gateways
* Efficient inventory and order management
* Extensible architecture for future enhancements

**Chapter-9**

**Limitation**

### Limitations

* + **Product Scope: Initial version limited to laptops only; accessories and peripherals planned for future phases**
  + **Internet Dependency: Real-time features require stable internet connection for smooth operation**
  + **Payment Gateway Dependency: Platform relies on Razorpay (India) and Stripe (International) availability**
  + **Cloudinary Dependency: Image storage and delivery dependent on Cloudinary service**
  + **Database Dependency: MongoDB Atlas uptime affects system availability**
  + **Concurrent User Load: Performance depends on server capacity and scaling infrastructure**
  + **Geographic Restrictions: International payments subject to regional regulations and gateway limitations**
  + **Browser Compatibility: Older browsers may not support all features**
  + **Mobile App: No dedicated mobile application in first release (website only)**
  + **Offline Functionality: Limited offline capabilities; PWA support planned for future**
  + **Payment Processing: PCI compliance requirements limit certain customizations**

**Chapter-10**

**Scope for Enhancement**

### Scope for Enhancement

* Laptop Accessories: Expansion to sell laptop accessories such as chargers, bags, and external storage devices
* AI-Based Recommendations: Implement machine learning for personalized laptop suggestions based on user preferences and browsing history
* Mobile Applications: Develop native iOS and Android applications for better mobile experience
* Multi-Language Support: Support for multiple languages to reach international customers
* Multi-Currency Support: Support for multiple currencies for global transactions
* Product Comparison: Enhanced product comparison tool with specification highlighting and side-by-side view
* Wishlist & Price Tracking: Advanced wishlist with price drop notifications
* Extended Hardware Categories: Addition of computer hardware, peripherals, and accessories
* Advanced Inventory: Warehouse management and multi-location inventory tracking
* Shipping Integration: Direct integration with shipping providers (FedEx, DHL, Local Couriers)
* Return Management: Automated return and refund processing with tracking
* Live Chat Support: Real-time customer support functionality
* Affiliate Program: Commission-based affiliate marketing program for sellers
* Refurbished Laptops: Support for selling refurbished and used laptops with certification

**Chapter-11**

**Abbreviations and Acronyms**

1. **Abbreviations and Acronyms**

|  |  |
| --- | --- |
| Acronym | Full Form |
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| RBAC | Role-Based Access Control |
| JWT | JSON Web Token |
| HTTPS | HyperText Transfer Protocol Secure |
| SSL | Secure Sockets Layer |
| PCI DSS | Payment Card Industry Data Security Standard |
| CRUD | Create, Read, Update, Delete |
| UI/UX | User Interface / User Experience |
| SEO | Search Engine Optimization |
| CDN | Content Delivery Network |
| HTML | HyperText Markup Language |
| CSS | Cascading Style Sheets |
| REST | Representational State Transfer |
| OAuth | Open Authorization |
| SQL | Structured Query Language |
| DB | Database |
| DBMS | Database Management System |
| URL | Uniform Resource Locator |
| CMS | Content Management System |
| OTP | One-Time Password |
| 2FA | Two-Factor Authentication |
| CSRF | Cross-Site Request Forgery |
| XSS | Cross-Site Scripting |
| RAM | Random Access Memory |
| CPU | Central Processing Unit |
| SSD | Solid State Drive |
| Mbps | Megabits per second |
| GB | Gigabytes |

**Chapter-12**

**Bibliography / References**

1. **Bibliography / References**
   1. **React.js Documentation** [- https://react.dev](-%20https:/react.dev)
   2. **Node.js & Express.js - https://nodejs.org / https://expressjs.com**
   3. **MongoDB Documentation -** <https://www.mongodb.com/docs>
   4. **Razorpay API Documentation - https://razorpay.com/docs/api**
   5. **Stripe API Documentation - https://stripe.com/docs/api**
   6. **Cloudinary API - https://cloudinary.com/documentation**
   7. **Tailwind CSS - https://tailwindcss.com**
   8. **Vite Documentation - https://vitejs.dev**
   9. **JWT.io - https://jwt.io**
   10. **bcryptjs Documentation - https://www.npmjs.com/package/bcryptjs**
   11. **GitHub - https://github.com**
   12. **ChatGPT (OpenAI) - https://openai.com/chatgpt**
   13. **GitHub Copilot - https://github.com/features/copilot**
   14. **MDN Web Docs - https://developer.mozilla.org**
   15. **Postman API Documentation - https://www.postman.com**