A REPORT OF ONE MONTH TRAINING

at

SENSATION SOFTWARE SOLUTIONS PVT. LTD.

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

(Computer Science & Engineering)

JUNE-JULY ,2025

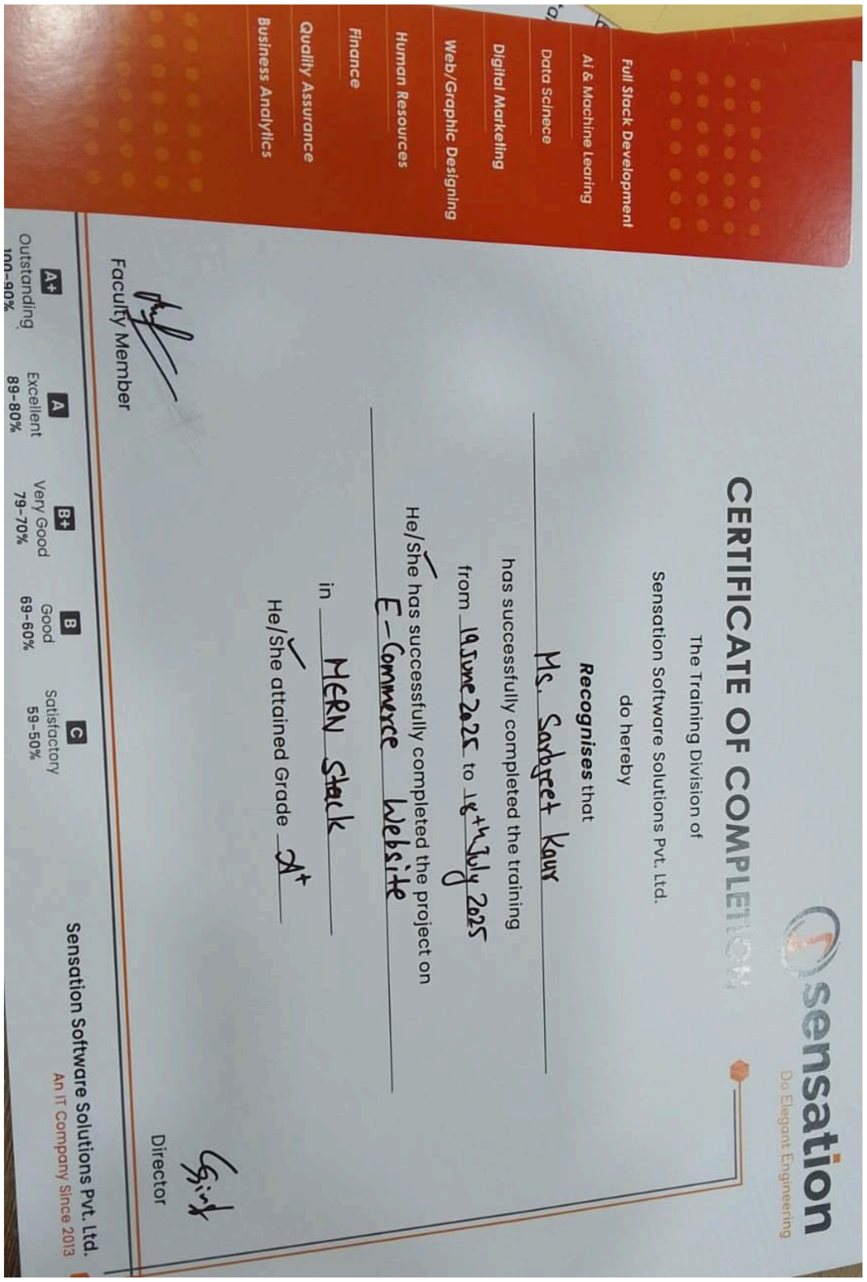
SUBMITTED BY:
NAME :SARBJEET KAUR
UNIVERSITY ROLL NO. :2302665

DEPARTMENT OF COMPUTER & SCIENCE ENGINEERING

GURU NANAK DEV ENGINEERING COLLEGE LUDHIANA

(An Autonomous College Under UGC ACT)

Certificate by SENSATION SOFTWARE SOLUTIONS PVT. LTD.



sensation

Do Elegant Engineering

Full Stack Development

AI & Machine Learing

Data Scinece

Digital Marketing

Web/Graphic Designing

Human Resources

Finance

Quality Assurance

Business Analytics

CERTIFICATE OF COMPLETION

The Training Division of

Sensation Software Solutions Pvt. Ltd.

do hereby

Recognises that

Ms. Sarbjeet Kaur

has successfully completed the training

from 19June2025 to 18th July 2025

He/She has successfully completed the project on

E-Commerce Website

in MERN Stack

He/She attained Grade A+

Faculty Member

Director

Sensation Software Solutions Pvt. Ltd.

An IT Company Since 2013

A+	A	B+	B	C
Outstanding	Excellent	Very Good	Good	Satisfactory
100-90%	89-80%	79-70%	69-60%	59-50%

CANDIDATE'S DECLARATION

I **SARBJEET KAUR** hereby declare that I have undertaken one month training at **“Sensation Software Solution Pvt. Ltd.”** during a period from **June2025** to **July2025** in partial fulfillment of requirements for the award of degree of B.Tech (Computer Science and Engineering) at GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA. The work which is being presented in the training report submitted to Department of Computer Science and Engineering at GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA is an authentic record of training work.

Signature of the Student

The one month industrial training Viva–Voce Examination of______________has been held on ______________and accepted.

Signature of Internal Examiner   Signature of External Examiner

ABSTRACT

This project presents a full-stack e-commerce web application developed using the MERN stack (MongoDB, Express.js, React.js, Node.js). The application provides users with a seamless shopping experience, including product browsing, detailed product views, secure user authentication, and shopping cart management. Users can search and filter products by category and price, while the system maintains real-time synchronization between the frontend interface and the backend database. An admin interface allows authorized users to efficiently manage products, update inventory, and control user roles.

The backend implements RESTful APIs with JWT-based authentication, bcrypt-encrypted passwords, and middleware for request validation and error handling, ensuring security and reliability. The frontend leverages React.js for dynamic, responsive, and component-based UI development, integrating API calls to provide real-time data updates. Comprehensive testing using Postman and MongoDB Compass verified functionality, data integrity, and performance.

The application was successfully deployed on cloud platforms—backend on Render and frontend on Vercel—allowing public access and demonstrating practical skills in modern web development, cloud deployment, and full-stack integration. This project highlights best practices in software architecture, modular design, and secure web application development, offering a scalable and user-friendly solution for online shopping.

ACKNOWLEDGEMENT

I extend my heartfelt gratitude to **Dr. Sehajpal Singh**, Principal of Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing me the invaluable opportunity to undertake this four-week training program. His encouragement and support motivated me to make the most of this learning experience.

I am sincerely thankful to **Dr. Kiran Jyoti**, Head of the Department at GNDEC, for her guidance and inspiration throughout the training period.

I express my deep appreciation to the mentors and trainers at **Sensations Software Solutions Pvt. Ltd.**, whose expertise and valuable insights helped me understand practical applications of web development using the **MERN stack (MongoDB, Express.js, React.js, Node.js)**. Their mentorship enabled me to work on an **e-commerce project**, which significantly enriched my technical skills.

My thanks also go to the faculty of the **Computer Science & Engineering Department, GNDEC**, for their continuous support and encouragement.

Lastly, I acknowledge the effort and dedication I invested in completing this project independently. Although challenging at times, the process was highly rewarding, allowing me to strengthen my **full-stack development skills**, deepen my technical understanding, and gain practical experience in building dynamic web applications.

ABOUT THE SENSATION SOFTWARE SOLUTION

SOLUTIONS PVT. LTD.

Sensations Software Solutions Pvt. Ltd. is a leading IT services and software development company headquartered in Mohali, Punjab. Established in 2013, it has grown into a trusted technology partner for clients across India and abroad. The company specializes in **web development, mobile applications, e-commerce platforms, UI/UX design, and digital marketing**.

Following agile methodologies, the company emphasizes quality, transparency, and client satisfaction. It has delivered over 200 projects for more than 150 clients worldwide, covering industries like healthcare, logistics, real estate, and education.

The organization encourages continuous learning and skill development, providing exposure to modern technologies, frameworks, and cloud computing.

Founder: Mr. Parveen Singh

Headquarters: Mohali, Punjab, India

Mission: To empower businesses and individuals with efficient, high-quality digital solutions that bridge the gap between technology and business success.

List of Figures

Figure No. **Page No.**

Figure 1.1 Mern stack 2

Figure 2.1 Application Workflow 15

Figure 3.1 MERN Stack Project Architecture or Workflow Diagram 18

Figure 3.2 Home Page showing product listings 19

Figure 3.3 Product details page interface 20

Figure 3.4 Shopping cart interface with update and remove options 20

Figure 3.5 User login and registration forms 21

Figure 3.6 Admin Dashboard 22

Figure 3.7: Postman showing API endpoints for user 23

Figure 3.8: JWT authentication response after successful login 24

Figure 3.9 Collections in MongoDB Compass 26

Figure 3.10: Filtered product results in MongoDB Compass 27

Figure 3.11: API testing 29

Figure 3.12 Responsive design on mobile 30

Figure 3.13 live backend API response from render 31

Figure 3.14 Live frontend interface hosted on Vercel 32

CONTENTS

Topic **Page No.**

Certificates by SENSATION SOFTWARE SOLUTIONS PVT. LTD. ***i***

CANDIDATE'S DECLARATION ***ii***

ABSTRACT ***iii***

ACKNOWLEDGEMENT ***vi***

ABOUT THE SENSATION SOFTWARE SOLUTIONS PVT. LTD ***v***

List of Figures ***vi***

CHAPTER 1 INTRODUCTION **1-7**

1.1 What is the MERN Stack? 1

1.2 Background 2

1.3 Purpose and Objective of the Training 3

1.4 Theoretical Concepts 4

1.5 E-Commerce Concepts 5

1.6 Tools and Technologies Learned 6-7

CHAPTER 2 TRAINING WORK UNDERTAKEN **8-16**

2.1 Overview of the Training Program 8

2.2 Curriculum and Course Content 9-11

2.3 Course Duration 11-12

2.4 Training Methodology and Approach 12-14

2.5 Project Undertaken 14-16

CHAPTER 3 RESULTS AND DISCUSSION **17-34**

3.1 Overview of the Project Results 17-18

3.2 Frontend Results 18-22

3.3 Backend Results 22-25

3.4 Database Results 25-27

3.5 Testing and Deployment Results 27-32

3.6 Overall Discussion 34-36

CHAPTER 3 RESULTS AND DISCUSSION **35-37**

4.1 Conclusion 35

4.2 Future Scope 35-37

REFERENCES **38-39**

APPENDIX **40-44**

CHAPTER 1 INTRODUCTION

1.1 What is the MERN Stack?

The MERN Stack is a popular and powerful web development technology stack that combines four key technologies—MongoDB, Express.js, React.js, and Node.js—to enable full-stack development using a single programming language, JavaScript. It allows developers to build dynamic, efficient, and scalable web applications by seamlessly connecting the frontend, backend, and database layers.

- **MongoDB** is a NoSQL database that stores information in flexible, JSON-like documents. It provides scalability and high performance, making it ideal for applications that handle large amounts of data such as e-commerce websites.
- **Express.js** is a lightweight web application framework for Node.js that simplifies backend development. It provides built-in tools for routing, handling requests, and managing middleware, allowing developers to build APIs quickly and efficiently.
- **React.js** is a JavaScript library used to build responsive and interactive user interfaces. It uses a component-based structure, enabling developers to create reusable UI components and manage application state effectively.
- **Node.js** is a runtime environment that allows JavaScript to run on the server side. It is designed for building scalable and high-performance network applications capable of handling multiple client requests simultaneously.

By integrating these four technologies, the MERN stack provides an end-to-end development environment where all layers of the application communicate seamlessly. This eliminates the need for multiple programming languages across different parts of the project, ensuring faster development, easier maintenance, and better performance.

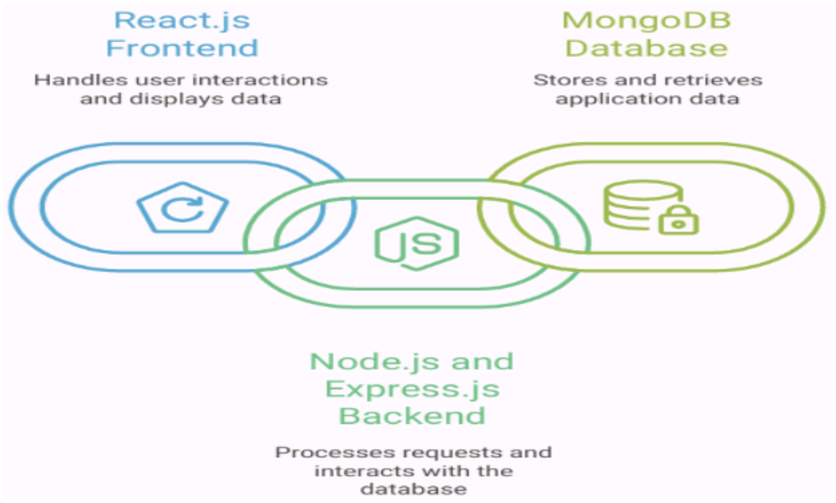

React.js
Frontend
Handles user interactions and displays data
MongoDB
Database
Stores and retrieves application data
Node.js and
Express.js
Backend
Processes requests and interacts with the database

Figure 1.1 Mern stack

1.2 Background

The rapid evolution of web technologies has significantly transformed the way businesses interact with customers. In today's digital era, online retail has become an essential channel for commerce, making e-commerce platforms a vital component for business growth. Developing a full-stack web application requires both frontend and backend knowledge, as well as the ability to connect them seamlessly.

The MERN stack—comprising MongoDB, Express.js, React.js, and Node.js—has emerged as one of the most popular full-stack development frameworks. MongoDB provides a flexible NoSQL database for storing structured and unstructured data, Express.js simplifies backend routing and API creation, React.js enables the development of dynamic and responsive user interfaces, and Node.js allows server-side programming with JavaScript.

During my one-month industrial training at Sensation Solutions, I focused on gaining hands-on experience with these technologies by developing a functional e-commerce web application. This training provided exposure to designing database schemas, creating RESTful APIs, building interactive user interfaces, and deploying a live application. It also strengthened my understanding of how modern web applications are structured, maintained, and scaled in real-world environments.

1.3 Purpose and Objective of the Training

1.3.1 Purpose:

The primary purpose of the MERN stack training at Sensation Solutions was to provide practical, hands-on experience in full-stack web development. The training aimed to equip participants with the skills necessary to design, develop, and deploy a complete web application using modern web technologies. It focused on bridging theoretical knowledge with real-world application, preparing participants for professional web development roles.

1.3.2 Objectives:

- Develop proficiency in frontend development using React.js, creating responsive and interactive user interfaces.
- Build backend APIs using Node.js and Express.js for server-side processing.
- Design and manage databases using MongoDB, including CRUD operations and schema management.
- Implement secure user authentication and session management for web applications.
- Integrate frontend, backend, and database to create a full-stack e-commerce application.

- Utilize version control (Git/GitHub) and deploy applications using cloud platforms such as Render and Vercel.
- Gain experience in testing, debugging, and deploying a functional web application in a real-world scenario.

1.4 Theoretical Concepts

Each component of the MERN stack contributes a specific function that enables the development of efficient, scalable, and interactive web applications.

- **MongoDB**: A NoSQL database that stores data in flexible, JSON-like documents, allowing developers to create complex data structures without rigid schemas. MongoDB supports efficient querying, indexing, and aggregation, making it ideal for applications like e-commerce platforms where product catalogs and user data must be dynamically managed.
- **Express.js**: A lightweight and flexible web application framework for Node.js that simplifies server-side development. Express.js provides features such as routing, middleware support, and HTTP utilities, which help in creating robust backend APIs for handling requests and responses efficiently.
- **React.js:** A JavaScript library for building dynamic user interfaces. React uses a component-based architecture, virtual DOM, and state management to render views efficiently. It allows developers to create responsive and interactive web pages, which is essential for e-commerce applications with features like product listings, shopping carts, and real-time updates.
- **Node.js:** A server-side JavaScript runtime environment that enables asynchronous programming and high-performance networking. Node.js allows developers to handle

multiple requests concurrently, making it suitable for applications that require fast response times and scalability, such as e-commerce platforms.

- By combining these technologies, the MERN stack enables developers to build full-stack applications where the frontend, backend, and database communicate seamlessly. This approach reduces the complexity of using multiple programming languages and tools, providing a unified development environment. During my training, I learned how each component interacts with the others to support a complete web application workflow—from data storage and server-side processing to rendering dynamic frontend interfaces.

1.5 E-Commerce Concepts

E-commerce, or electronic commerce, refers to the buying and selling of goods and services over the internet. Modern e-commerce platforms provide users with a seamless shopping experience, allowing them to browse products, place orders, make payments, and track deliveries—all from a web browser or mobile device.

In my MERN stack project, the following key features were implemented:

- **Product Catalog:** A dynamic display of available products, including names, images, descriptions, categories, and prices. Users can filter and search products to quickly find what they need.
- **Shopping Cart:** Functionality that allows users to add, remove, and modify product quantities before proceeding to checkout. The cart maintains session data so users can continue shopping without losing selected items.

- **User Authentication:** Secure login and registration system using technologies like JWT (JSON Web Tokens) and bcrypt for password hashing. Authentication ensures that sensitive user data is protected and only authorized users can access certain features.
- **Responsive Design**: The frontend interface is built using React.js to ensure that the platform is accessible and visually consistent across different devices and screen sizes.
- **Admin Dashboard:** A secure admin interface that allows administrators to manage products, view users, and monitor site activity. This feature enables efficient content and inventory management directly from the web application.

Implementing these features in a MERN stack project allowed me to understand how frontend components, backend APIs, and database operations interact to deliver a smooth and functional e-commerce application. The practical exposure to these concepts strengthened my understanding of both theoretical and applied aspects of full-stack web development.

1.6 Tools and Technologies Learned

During the one-month industrial training, I gained hands-on experience with various tools and technologies essential for developing a full-stack e-commerce application using the MERN stack. These tools helped me understand real-world development practices and improved my proficiency in both frontend and backend development.

- **Visual Studio Code (VS Code):** A versatile integrated development environment (IDE) used for writing, debugging, and managing code efficiently. It supports extensions that simplify working with JavaScript, React.js, Node.js, and MongoDB.

- **MongoDB Compass:** A graphical user interface for managing MongoDB databases. It allowed me to create collections, insert and query documents, and visually explore the database structure.
- **Postman:** A tool for testing RESTful APIs. I used it to verify backend endpoints, test HTTP requests (GET, POST, PUT, DELETE), and ensure proper communication between the frontend and backend.
- **Git and GitHub:** Version control tools used for managing code changes, collaborating with others, and maintaining a project repository. GitHub also provided a platform to store the project securely and track development progress.
- **Render / Vercel:** Cloud platforms used to deploy the backend and frontend of the full-stack application. Render was used for deploying the Node.js backend, while Vercel was used to host the React.js frontend. These platforms allowed me to make the project live, accessible through web URLs, and demonstrate deployment processes in a professional environment.

CHAPTER 2 TRAINING WORK UNDERTAKEN

2.1 Overview of the Training Program

The MERN stack training program at Sensation Solutions was designed to provide participants with a comprehensive understanding of full-stack web development using modern web technologies. The primary objective was to equip participants with both theoretical knowledge and practical experience in building dynamic, responsive, and scalable web applications.

The program emphasized hands-on learning through the development of a complete e-commerce application. Participants gained exposure to the entire web development workflow, including frontend design, backend API development, database management, version control, and cloud deployment.

The training was structured to gradually build participants' skills, starting from fundamental concepts in HTML, CSS, and JavaScript, and progressing to advanced topics such as React.js component architecture, Node.js server-side programming, Express.js API routing, and MongoDB database operations. This step-by-step approach ensured that learners understood individual technologies before integrating them into a full-stack project.

By the end of the training, participants were able to design, implement, and deploy a functional web application, gaining insights into real-world development practices, problem-solving strategies, and professional workflows used in IT organizations.

2.2 Curriculum and Course Content

The MERN stack training program was designed to provide a balanced mix of theoretical understanding and practical application. The curriculum was structured in progressive modules, enabling participants to gradually acquire skills required for full-stack development and apply them to real-world projects.

Module 1: Introduction to Web Development and JavaScript

Objective: Build foundational knowledge of web technologies and core JavaScript concepts.

Topics Covered:

- Overview of web development: Frontend, backend, and full-stack concepts
- Basics of HTML, CSS, and JavaScript
- JavaScript concepts: Variables, loops, functions, events, and DOM manipulation

  Hands-on Exercises: Creation of static web pages and simple interactive scripts

Module 2: React.js Frontend Development

Objective: Learn to build dynamic, responsive, and interactive user interfaces.

Topics Covered:

- Component-based architecture and reusable components
- Props, state management, and hooks
- React Router for single-page application navigation
- Event handling and form management

  Hands-on Exercises: Build product catalog, shopping cart, and user interface components

Module 3: Node.js and Express.js Backend Development

Objective: Develop server-side applications and APIs.

Topics Covered:

- Node.js fundamentals: Asynchronous programming and event-driven architecture
- Express.js: Routing, middleware, and RESTful API development
- Handling HTTP requests and responses
- Error handling and authentication mechanisms

  Hands-on Exercises: Create backend APIs for product management, user authentication, and order processing

Module 4: MongoDB Database Management

Objective: Learn to store, retrieve, and manage data efficiently.

Topics Covered:

- NoSQL database concepts
- Creating collections and performing CRUD operations
- Schema design and data validation
- Indexing and querying techniques

  Hands-on Exercises: Implement product catalogs, user profiles, and order management systems

Module 5: Integration and Full-Stack Development

Objective: Combine frontend, backend, and database to build a complete application.

Topics Covered:

- Connecting React frontend to Node/Express backend
- API integration and data fetching
- Managing state and session data across the application

  Hands-on Exercises: Implement end-to-end functionalities such as user authentication, shopping cart, and order processing

Module 6: Deployment and Version Control

Objective: Understand professional workflows and deploy applications.

Topics Covered:

- Version control using Git and GitHub
- Deployment of backend on Render and frontend on Vercel
- Continuous testing and debugging

  Hands-on Exercises: Deploy the e-commerce application to live servers and ensure proper functionality

This structured curriculum ensured that participants gained practical experience at every stage of development, from foundational concepts to building and deploying a full-stack web application.

2.3 Course Duration

The MERN stack training program was conducted over a four-week period, designed to gradually build knowledge and hands-on skills in full-stack web development. The schedule combined lectures, practical exercises, and project work to ensure comprehensive learning:

- Week 1: Introduction to web development, HTML, CSS, and JavaScript fundamentals. Hands-on exercises included creating static web pages and understanding basic DOM manipulation.
- Week 2: Frontend development using React.js. Participants learned about components, state management, hooks, and navigation using React Router. Hands-on tasks included developing the product catalog and shopping cart interfaces.
- Week 3: Backend development with Node.js and Express.js, and database management using MongoDB. Participants implemented RESTful APIs, designed schemas, and performed CRUD operations on the database.
- Week 4: Full-stack integration and deployment. Participants connected the frontend with the backend, tested APIs, and deployed the live application using Render (backend) and Vercel (frontend).

Each week included lectures, coding exercises, mini-projects, and instructor-led guidance, ensuring that participants could apply theoretical concepts in practical scenarios and develop a fully functional e-commerce web application by the end of the program.

2.4 Training Methodology and Approach

The MERN stack training adopted a structured and practical methodology, blending theoretical knowledge with hands-on exercises to ensure participants gained both understanding and real-world skills. The approach focused on active learning, problem-solving, and project-based application.

1. **Blended Learning**

   a. Interactive Sessions: Core concepts of frontend, backend, and database development were delivered through lectures and demonstrations.
   b. Hands-On Exercises: Participants implemented code examples, practiced building components, and developed APIs to reinforce learning.

2. **Project-Based Learning**
   a. Mini-Projects: Small assignments such as building a product catalog or a shopping cart helped participants understand the workflow of a full-stack application.
   b. Main Project: The e-commerce web application served as the capstone project, integrating all learned skills, from frontend design to backend APIs and database management.

3. **Experiential Learning**
   a. Participants learned by doing, creating real features such as user authentication, product listing, and order management.
   b. This approach encouraged problem-solving and understanding the interaction between frontend, backend, and database systems.

4. **Version Control and Collaboration**
   a. Git and GitHub were used to maintain code, track changes, and simulate a professional development environment.
   b. Collaborative exercises emphasized teamwork and proper code management.

5. **Testing and Deployment**
   a. Postman was used to test API endpoints and ensure correct communication between frontend and backend.

b. Render (backend) and Vercel (frontend) were used for deployment, giving participants experience in hosting a live application.

This methodology ensured that participants gained a holistic understanding of full-stack development, improved their coding skills, and were prepared to apply MERN stack technologies in professional projects.

2.5 Project Undertaken

The capstone project for the training program involved the development of a full-stack e-commerce web application using the MERN stack. This project integrated all the skills and concepts acquired during the training, providing comprehensive hands-on experience in designing, implementing, and deploying a functional web application.

2.5.1 Project Overview:

The application allows users to browse products, manage a shopping cart, register and log in securely, and interact with the product catalog. The backend handles data management and API development, while the frontend delivers a responsive and interactive user interface.

2.5.2 Key Features Implemented:

1. **Dynamic Product Catalog:** Displays products with images, descriptions, categories, and pricing. Users can filter and search for products seamlessly.
2. **Shopping Cart Functionality:** Users can add, remove, or update items in the cart. The cart maintains session data to enhance user convenience.
3. **Secure User Authentication**: Implemented using JWT (JSON Web Tokens) and bcrypt for password encryption, ensuring data security and controlled access.

4. **Responsive Frontend Interface:** Developed using React.js to ensure compatibility across devices and screen sizes.
5. **Database Integration:** MongoDB is used for efficient storage and retrieval of product and user information.
6. **API Testing:** Postman was employed to test RESTful API endpoints, ensuring smooth communication between frontend and backend.
7. **Deployment:** The backend was deployed on Render, and the frontend on Vercel, making the application live and accessible online.
8. **Admin Dashboard:** A separate dashboard for administrators to manage product listings, user accounts, and application data. It includes functionalities such as adding or removing products and viewing

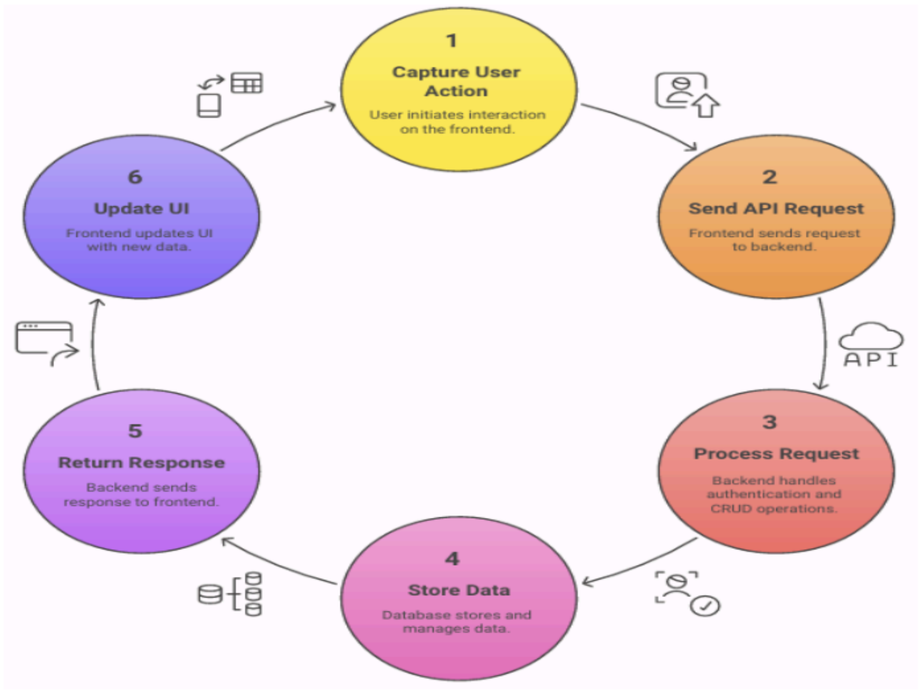

1
Capture User Action
User initiates interaction on the frontend.
2
Send API Request
Frontend sends request to backend.
API
3
Process Request
Backend handles authentication and CRUD operations.
4
Store Data
Database stores and manages data.
5
Return Response
Backend sends response to frontend.
6
Update UI
Frontend updates UI with new data.

Figure 2.1 Application Workflow

2.5.3 Learning Outcomes:

- Gained practical experience in end-to-end full-stack development, from database design to frontend implementation.
- Applied theoretical knowledge of MongoDB, Express.js, React.js, and Node.js in a real-world project setting.
- Developed skills in API testing, debugging, and deployment using cloud platforms.
- Strengthened problem-solving, logical thinking, and project management capabilities.

This project reinforced technical skills learned during training and provided insight into professional development workflows, preparing me for real-world full-stack development challenges.

CHAPTER 3 RESULTS AND DISCUSSION

3.1 Overview of the Project Results

The capstone project undertaken during the training was a full-stack e-commerce web application developed using the MERN stack — comprising MongoDB, Express.js, React.js, and Node.js. The project successfully demonstrated the complete development workflow, from frontend design and backend logic to database integration and deployment.

The primary outcome of the project was a fully functional and responsive e-commerce platform that allowed users to browse products, register or log in securely, manage shopping carts, and interact with the backend through RESTful APIs. The system effectively integrated all four MERN technologies, showcasing seamless communication between the client-side interface, server-side logic, and database.

3.1.1 Key Achievements:

- Developed a responsive user interface using React.js with smooth navigation and dynamic rendering of products.
- Implemented secure user authentication using JWT (JSON Web Tokens) and bcrypt for password encryption.
- Created RESTful APIs in Node.js and Express.js for handling product, user, and cart operations.
- Designed and managed a MongoDB database to store product, user, and order information efficiently.
- Deployed the backend on Render and the frontend on Vercel, ensuring live accessibility and testing of the project.

The project served as a comprehensive demonstration of full-stack development, integrating all learned concepts from the training into a real-world application that was both scalable and interactive.

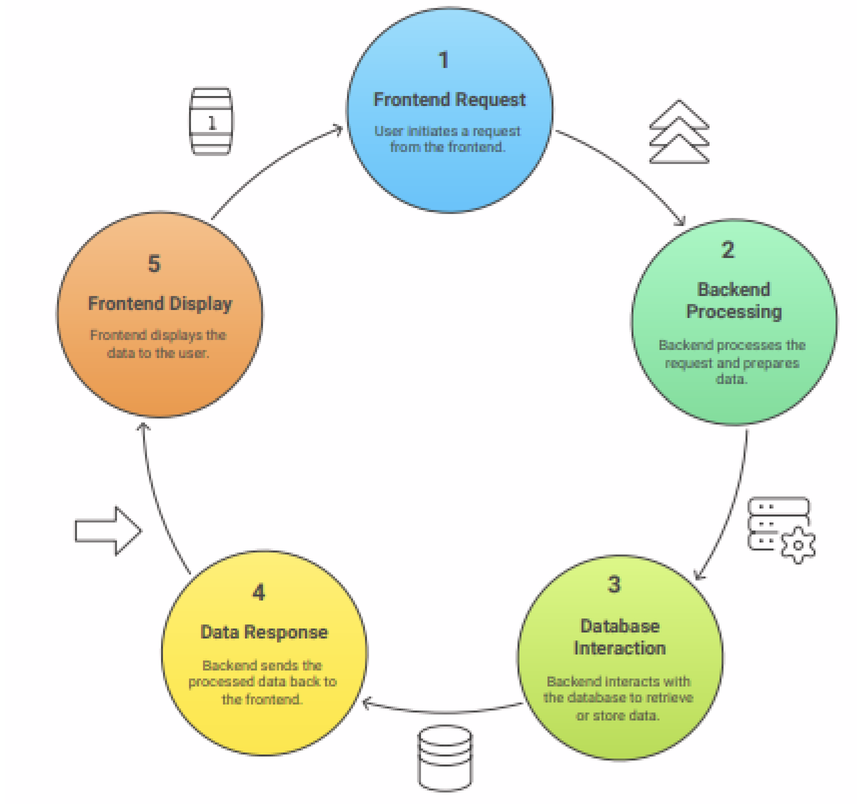

1
Frontend Request
User initiates a request from the frontend.
2
Backend Processing
Backend processes the request and prepares data.
3
Database Interaction
Backend interacts with the database to retrieve or store data.
4
Data Response
Backend sends the processed data back to the frontend.
5
Frontend Display
Frontend displays the data to the user.

Figure 3.1 MERN Stack Project Architecture or Workflow Diagram

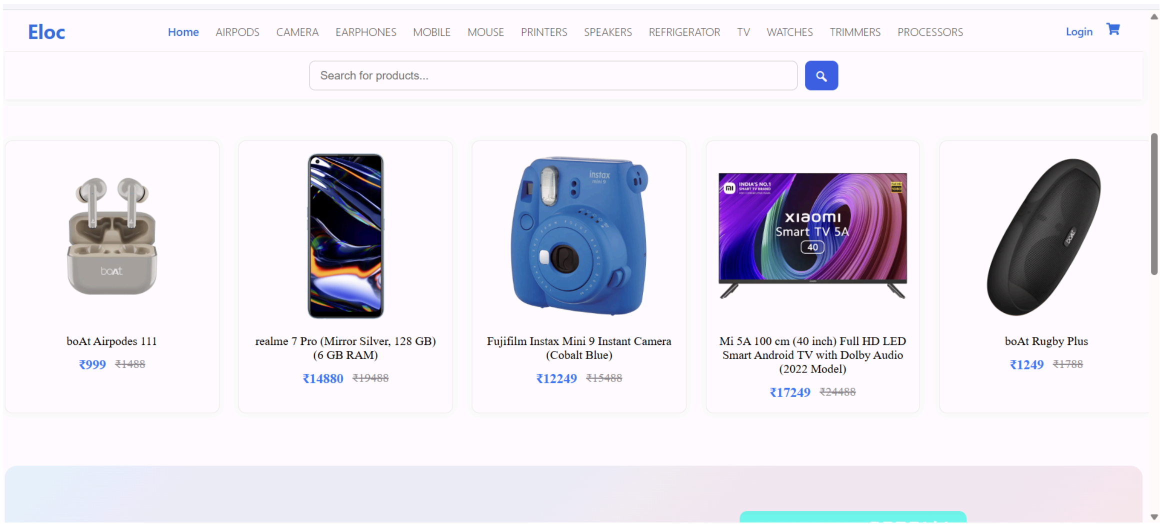
3.2 Frontend Results

The frontend of the e-commerce application was developed using React.js, focusing on creating an intuitive, responsive, and visually appealing user interface. The design emphasized simplicity, usability, and smooth navigation to ensure a positive user experience across different devices and screen sizes.

3.2.1 Key Features and Results:

- **Home Page and Product Listings:**

  Displayed products dynamically with names, images, categories, and prices. The interface allowed users to scroll, view multiple items, and navigate easily between sections.



boAt Airpodes 111
₹999 ~~₹1488~~

realme 7 Pro (Mirror Silver, 128 GB) (6 GB RAM)
₹14880 ~~₹19488~~

Fujifilm Instax Mini 9 Instant Camera (Cobalt Blue)
₹12249 ~~₹15488~~

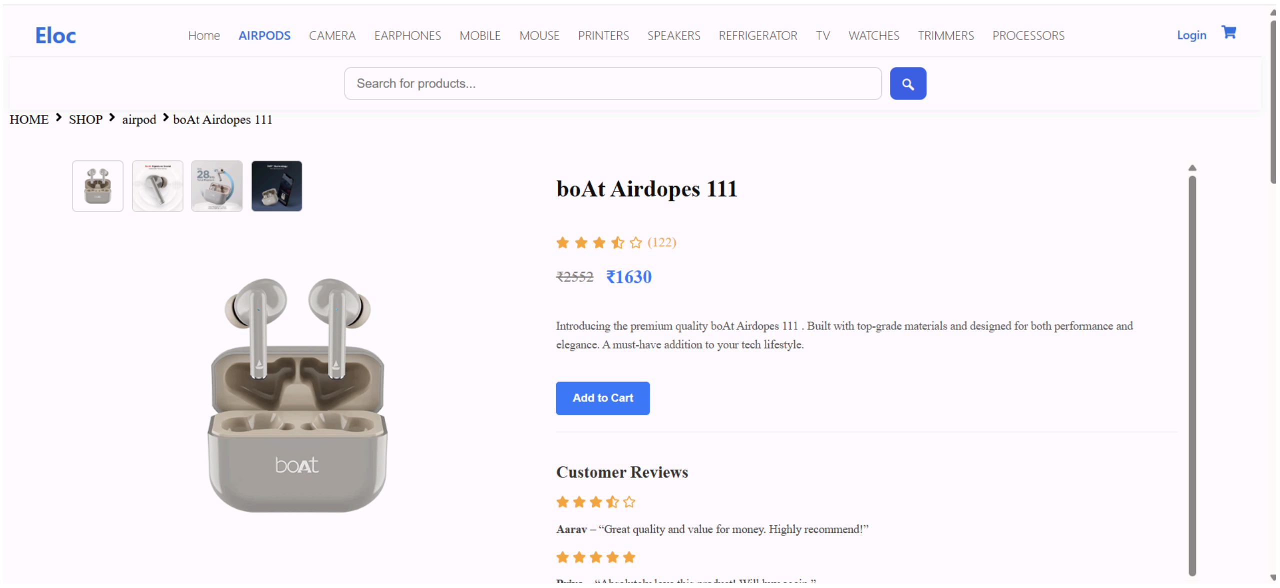
Mi 5A 100 cm (40 inch) Full HD LED Smart Android TV with Dolby Audio (2022 Model)
₹17249 ~~₹24488~~

boAt Rugby Plus
₹1249 ~~₹1788~~

Figure 3.2 Home Page showing product listings

- **Product Details Page:**

  Provided detailed product information including description, price, and an option to add items to the shopping cart.


Eloc

Home AIRPODS CAMERA EARPHONES MOBILE MOUSE PRINTERS SPEAKERS REFRIGERATOR TV WATCHES TRIMMERS PROCESSORS Login

Search for products...

HOME › SHOP › airpod › boAt Airdopes 111

boAt Airdopes 111

★★★★☆ (122)

~~₹2552~~ ₹1630

Introducing the premium quality boAt Airdopes 111 . Built with top-grade materials and designed for both performance and elegance. A must-have addition to your tech lifestyle.

Add to Cart

Customer Reviews

★★★★☆

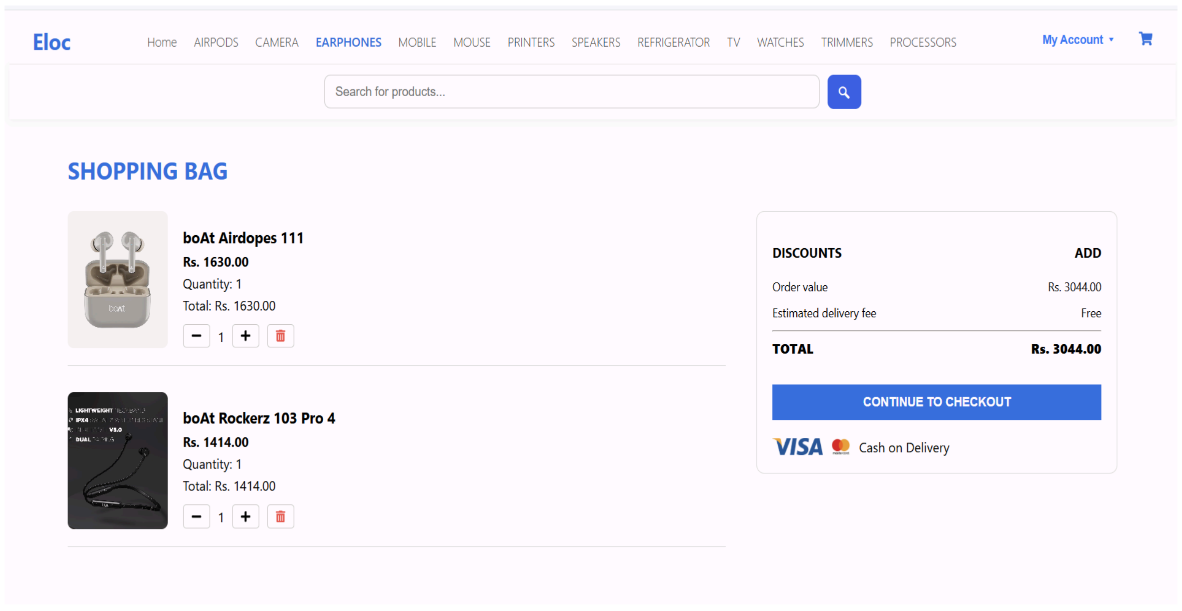
Aarav – "Great quality and value for money. Highly recommend!"

★★★★★

Figure 3.3 Product details page interface

- **Shopping Cart:**

Implemented functionality for adding, updating, and removing items. The cart maintained state across navigation, ensuring a smooth shopping experience


Eloc

Home AIRPODS CAMERA EARPHONES MOBILE MOUSE PRINTERS SPEAKERS REFRIGERATOR TV WATCHES TRIMMERS PROCESSORS My Account ▾

Search for products...

SHOPPING BAG

boAt Airdopes 111
Rs. 1630.00
Quantity: 1
Total: Rs. 1630.00
− 1 +

boAt Rockerz 103 Pro 4
Rs. 1414.00
Quantity: 1
Total: Rs. 1414.00
− 1 +

DISCOUNTS	**ADD**
Order value	Rs. 3044.00
Estimated delivery fee	Free
TOTAL	**Rs. 3044.00**

CONTINUE TO CHECKOUT

VISA Cash on Delivery

Figure 3.4 Shopping cart interface with update and remove options

- **User Authentication Interface:**

  Designed login and registration forms with real-time validation to enable secure access to user accounts.

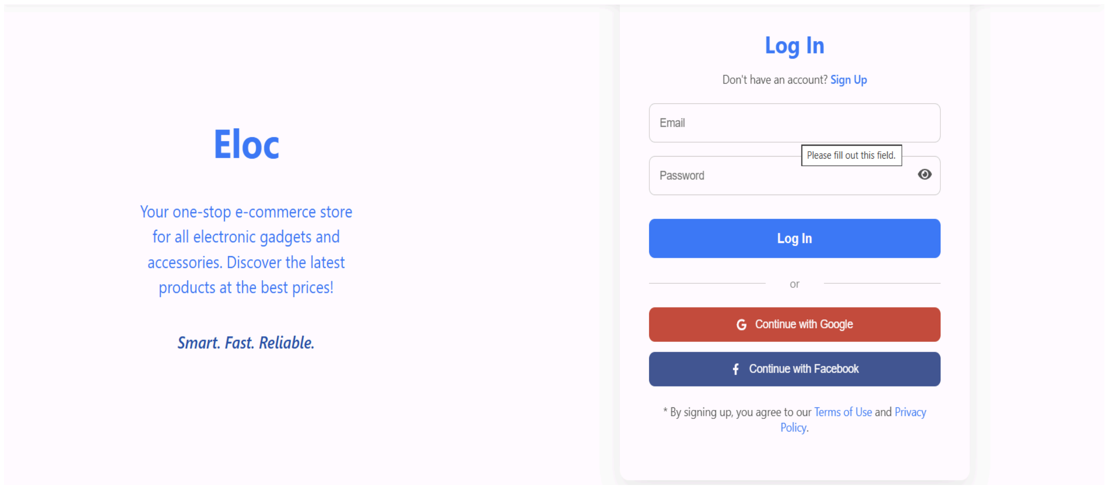

Eloc
Your one-stop e-commerce store for all electronic gadgets and accessories. Discover the latest products at the best prices!
Smart. Fast. Reliable.
Log In
Don't have an account? Sign Up
Email
Please fill out this field.
Password
Log In
or
Continue with Google
Continue with Facebook
* By signing up, you agree to our Terms of Use and Privacy Policy.

Figure 3.5 User login and registration forms

- **Responsive Layout:**

  Ensured full responsiveness using CSS Flexbox and media queries, adapting seamlessly to desktop, tablet, and mobile devices.

- **Admin Interface:**

  Provided administrators with a secure dashboard to manage products and users. Features included adding new products with images, updating or removing existing products, and promoting users to admin roles. The interface ensured smooth interaction with backend APIs and maintained role-based access control.

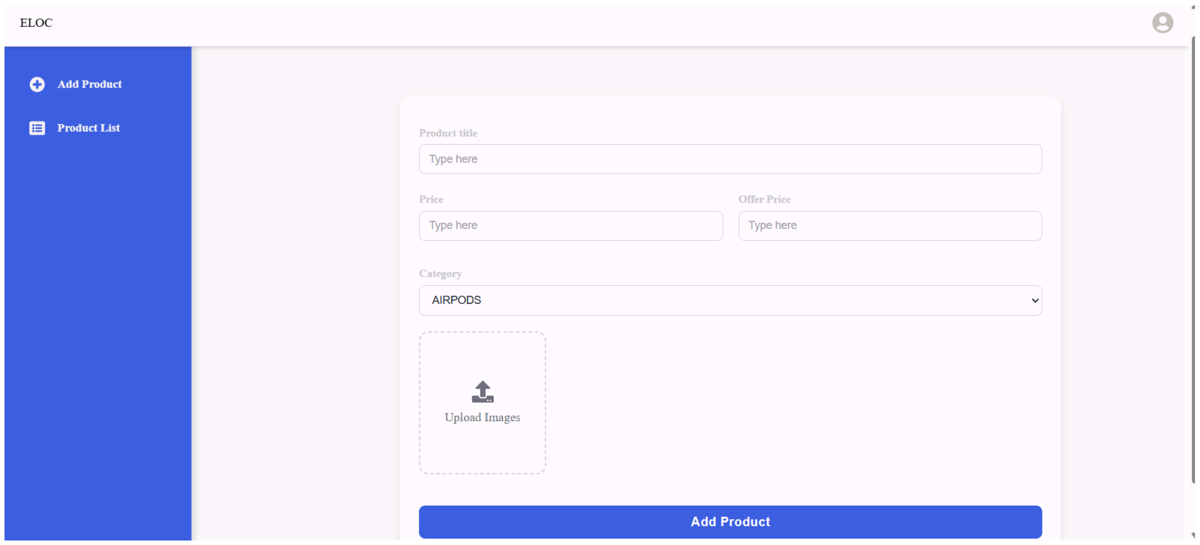


Figure 3.6 Admin Dashboard

- **Navigation and Routing:**

  Integrated React Router to handle navigation between pages without full page reloads, resulting in a faster and smoother user experience.

3.2.2 Discussion:

The frontend architecture leveraged React's component-based design, enabling reusable UI elements and easier code maintenance. React hooks such as useState and useEffect were utilized for managing component states and lifecycle events efficiently. API data was fetched using Fetching , allowing real-time synchronization between the user interface and backend data.

This approach ensured modularity, scalability, and responsiveness throughout the application. The frontend successfully provided a polished interface that interacted seamlessly with backend services.

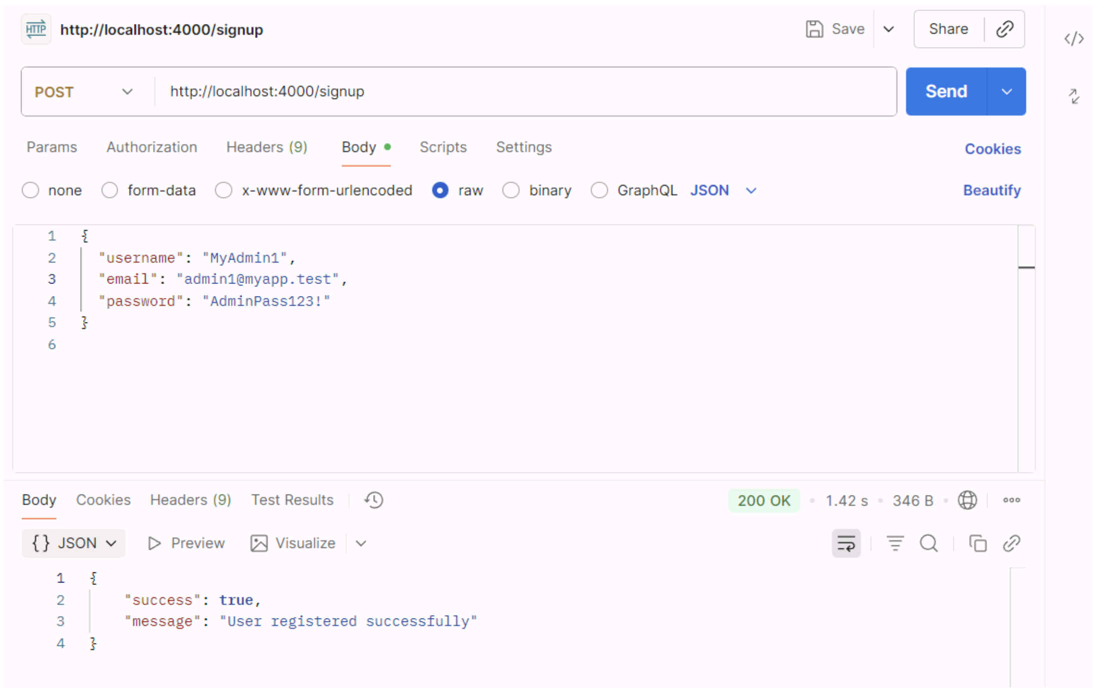
3.3 Backend Results

The backend of the e-commerce web application was developed using Node.js and Express.js, forming the core of the application's server-side logic. It was responsible for handling API requests, processing business logic, managing authentication, and facilitating data exchange between the frontend and database.

3.3.1 Key Features and Results:

- **RESTful API Development:**

  Implemented multiple API endpoints to manage users, products, and shopping cart operations. Each route was structured to follow RESTful conventions, ensuring clarity and scalability.



```
POST http://localhost:4000/signup
```

```
{
  "username": "MyAdmin1",
  "email": "admin1@myapp.test",
  "password": "AdminPass123!"
}
```

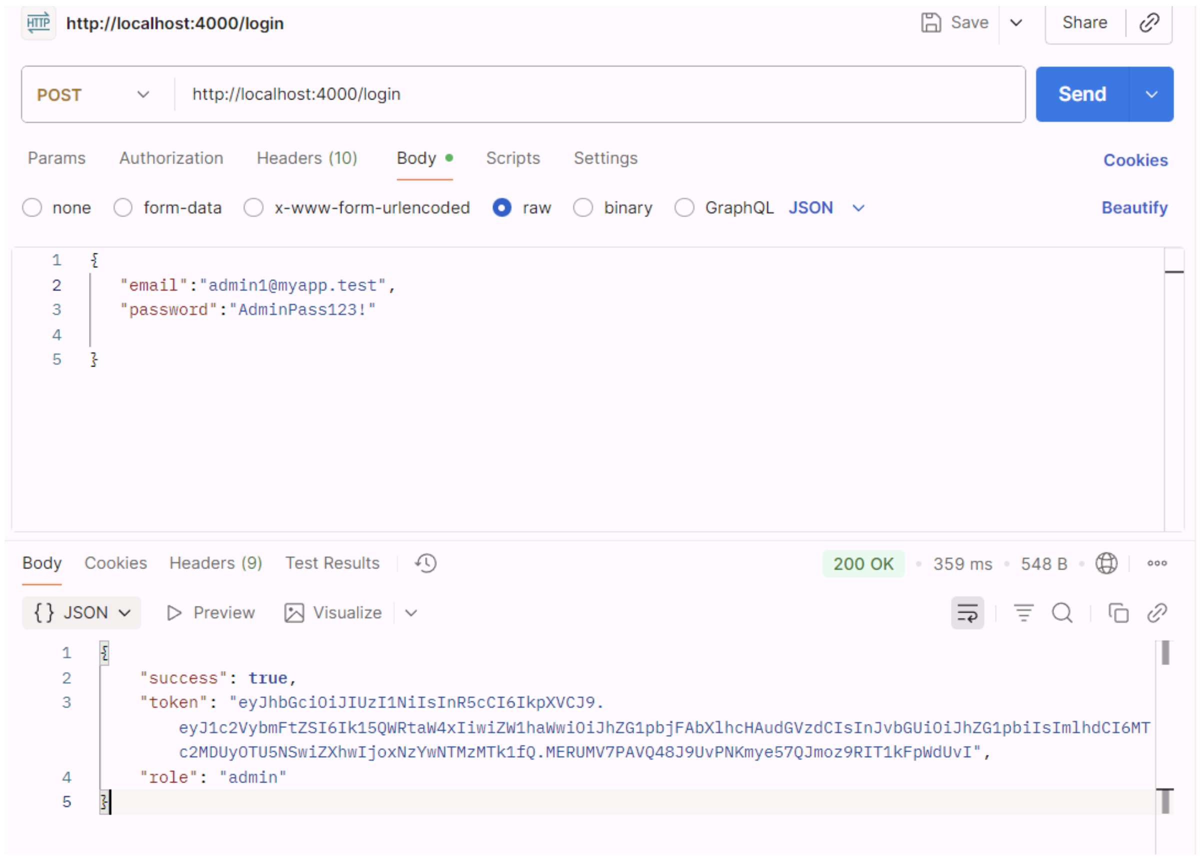
```
{
    "success": true,
    "message": "User registered successfully"
}
```

Figure 3.7: Postman showing API endpoints for user

- **User Authentication:**

  Integrated JWT (JSON Web Tokens) for secure user authentication and session handling.

  Used bcrypt for password encryption to protect user credentials in the database.



POST http://localhost:4000/login

```
{
   "email":"admin1@myapp.test",
   "password":"AdminPass123!"

}
```

```
{
    "success": true,
    "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJ1c2VybmFtZSI6Ik15QWRtaW4xIiwiZW1haWwiOiJhZG1pbjFAbXlhcHAudGVzdCIsInJvbGUiOiJhZG1pbiIsImlhdCI6MT
        c2MDUyOTU5NSwiZXhwIjoxNzYwNTMzMTk1fQ.MERUMV7PAVQ48J9UvPNKmye57QJmoz9RIT1kFpWdUvI",
    "role": "admin"
}
```

Figure 3.8: JWT authentication response after successful login

- **Product and Cart Management:**

  Developed routes for CRUD (Create, Read, Update, Delete) operations on products and cart items.

  Enabled seamless interaction between the user interface and database for real-time data updates.

- **Middleware Integration:**

  Configured custom middleware for request validation, authentication verification, and error handling to enhance security and reliability.

- **Server Configuration:**

  Established a Node.js server using Express.js to manage HTTP requests efficiently and ensure smooth API communication with the frontend.

3.3.2 Discussion:

The backend was designed following a modular architecture, separating route handlers, controllers, and database models for better code organization.
Express.js simplified API routing and middleware management, while Node.js provided a non-blocking event-driven environment, allowing the application to handle multiple requests simultaneously.

This structure resulted in a fast, secure, and reliable backend, capable of supporting a scalable full-stack application.

3.4 Database Results

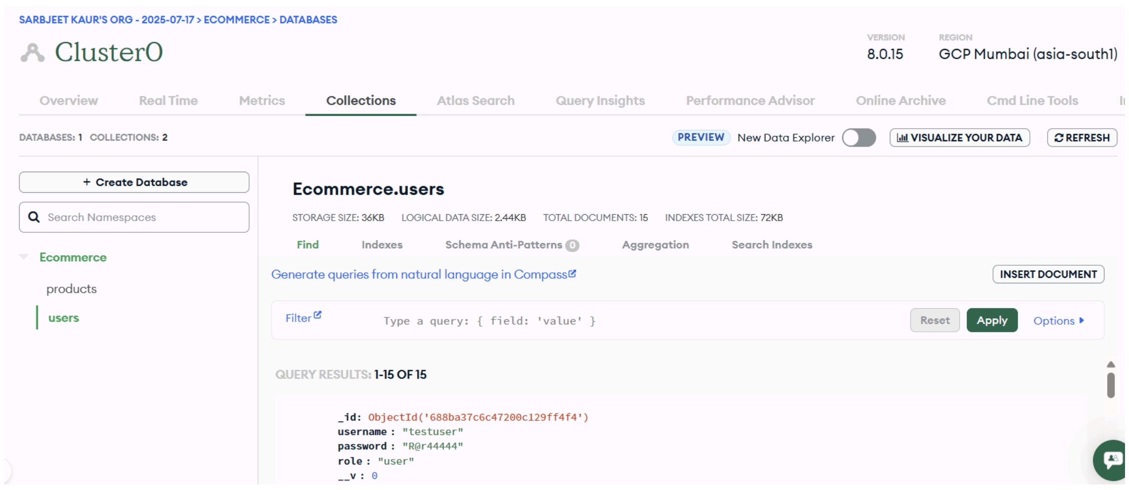
The database component of the e-commerce web application was developed using MongoDB, a NoSQL database system that stores data in a flexible, document-oriented format. It was used to manage all persistent data such as users, products, and shopping cart details.

3.4.1 Key Features and Results:

- **Database Design:**

  Created multiple collections — including Users, Products, and Carts — to handle different data entities within the application.

Each collection was structured using appropriate schemas defined in Mongoose, ensuring data consistency and validation.



```
_id: ObjectId('688ba37c6c47200c129ff4f4')
username : "testuser"
password : "R@r44444"
role : "user"
__v : 0
```

Figure 3.9 Collections in MongoDB Compass

- **CRUD Operations:**

  Implemented Create, Read, Update, and Delete operations via Express.js APIs for efficient database manipulation.

  This allowed real-time updating of product details, user information, and cart status.

- **Data Relationships:**

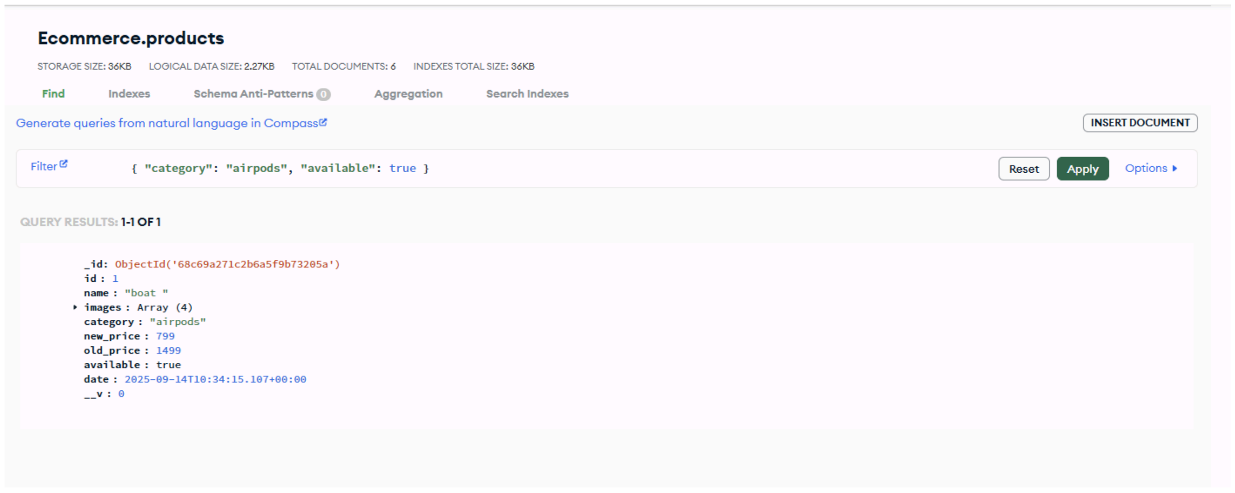
  Established logical relationships between collections — for example, linking user IDs with their respective cart and order data.

  Used ObjectIDs in Mongoose to reference related documents and maintain database integrity.

- **Efficient Querying and Filtering:**

  Developed optimized queries to filter products by category, price, and other attributes, enhancing user experience on the frontend.

Aggregation pipelines were used where necessary to generate dynamic search results.


Ecommerce.products
STORAGE SIZE: 36KB LOGICAL DATA SIZE: 2.27KB TOTAL DOCUMENTS: 6 INDEXES TOTAL SIZE: 36KB
Find
Indexes
Schema Anti-Patterns 0
Aggregation
Search Indexes
Generate queries from natural language in Compass
INSERT DOCUMENT
Filter
Reset
Apply
Options

```
{ "category": "airpods", "available": true }
```

QUERY RESULTS: 1-1 OF 1

```
  _id: ObjectId('68c69a271c2b6a5f9b73205a')
  id : 1
  name : "boat "
▸ images : Array (4)
  category : "airpods"
  new_price : 799
  old_price : 1499
  available : true
  date : 2025-09-14T10:34:15.107+00:00
  __v : 0
```

Figure 3.10: Filtered product results in MongoDB Compass

- **Performance and Security:**

  Indexes were created on frequently accessed fields to improve query performance.

  Sensitive information such as passwords was stored securely after encryption using bcrypt on the backend.

3.4.2 Discussion:

MongoDB's flexible schema design made it well-suited for handling dynamic product and user data. Integration with Mongoose simplified data modeling and validation processes.

The NoSQL nature of MongoDB provided scalability, allowing the application to handle large datasets efficiently.

Additionally, MongoDB Compass facilitated the visualization and verification of stored data, making debugging and data management more convenient during development.

3.5 Testing and Deployment Results

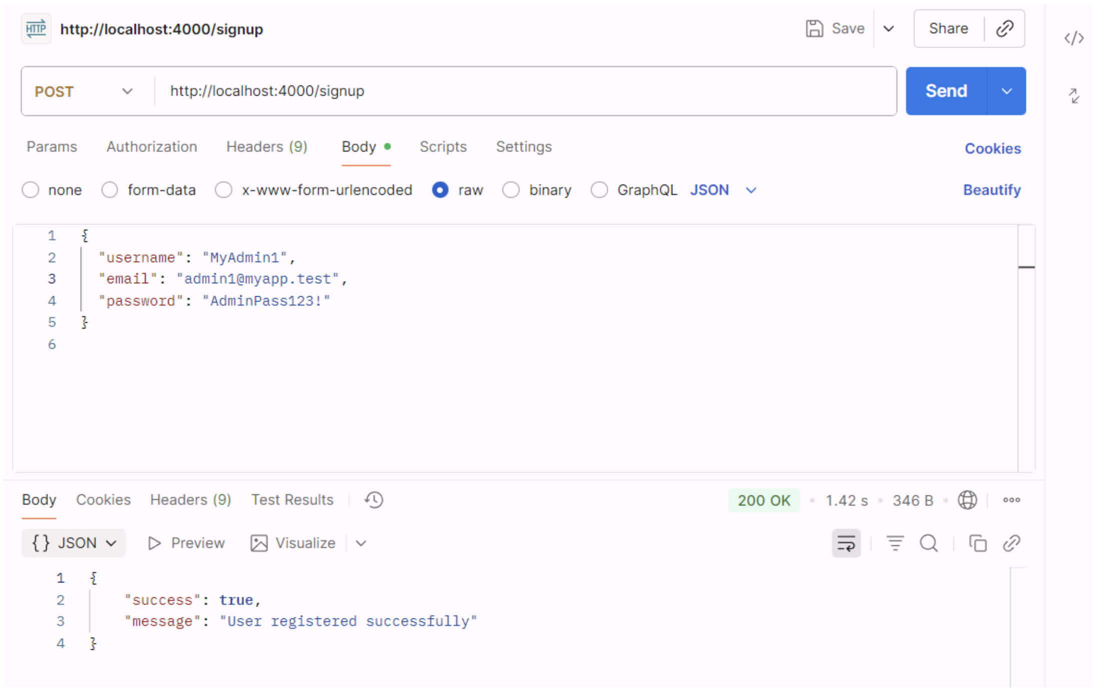
The testing and deployment phase was a crucial part of the MERN stack e-commerce application development. It ensured that all components — frontend, backend, and database — worked seamlessly together and that the application could be accessed publicly after successful deployment.

3.5.1 Testing Phase

Comprehensive testing was conducted throughout the project to ensure functionality, reliability, and performance of the web application.

1. API Testing using Postman:

- All backend APIs were tested using Postman to verify request and response accuracy.
- HTTP methods such as GET, POST, PUT, and DELETE were tested for product, user, and cart routes.
- Response codes, authentication headers (JWT), and database updates were verified for consistency and correctness.
- Proper error handling was confirmed for invalid or unauthorized requests.



```
{
  "username": "MyAdmin1",
  "email": "admin1@myapp.test",
  "password": "AdminPass123!"
}
```

```
{
    "success": true,
    "message": "User registered successfully"
}
```

Figure 3.11: API testing

2. Frontend Testing:

- The React.js frontend was tested for proper component rendering, form validation, and data fetching from APIs.
- Navigation and routing were checked using React Router to ensure smooth transitions between pages.
- The shopping cart and authentication modules were validated for correct functionality and data synchronization with the backend.
- Cross-device and responsive design testing were performed to confirm consistent UI across desktop and mobile views.

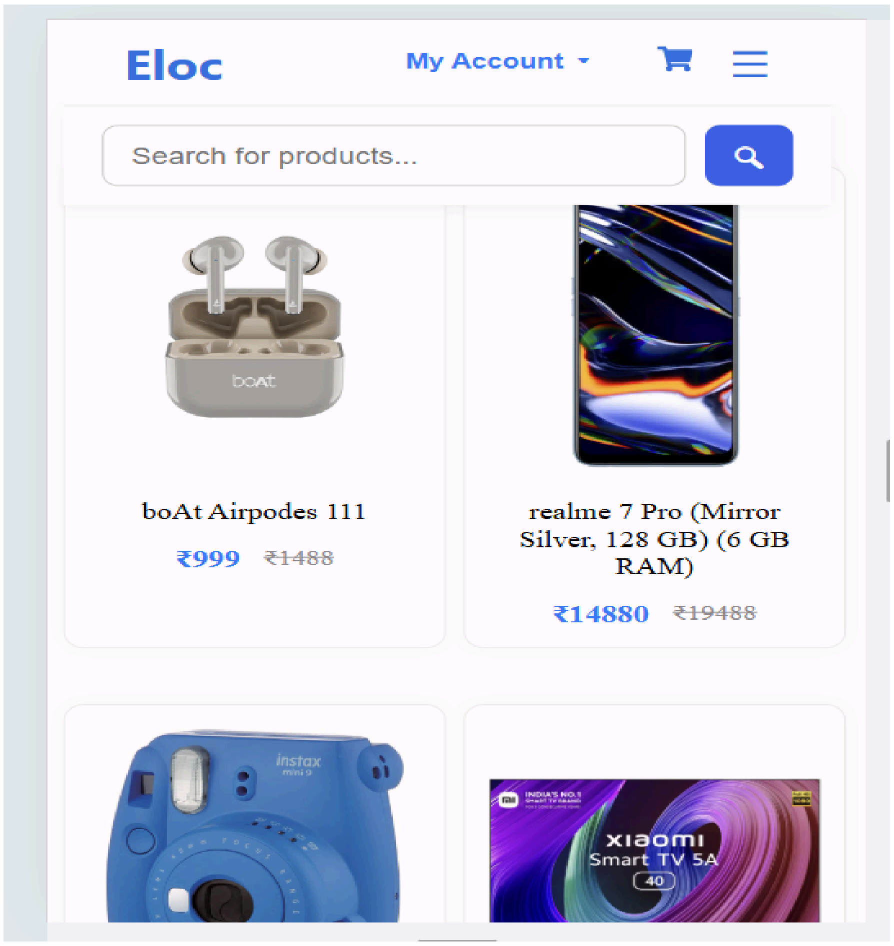
- 

Figure 3.12 Responsive design on mobile

3. Database Testing:

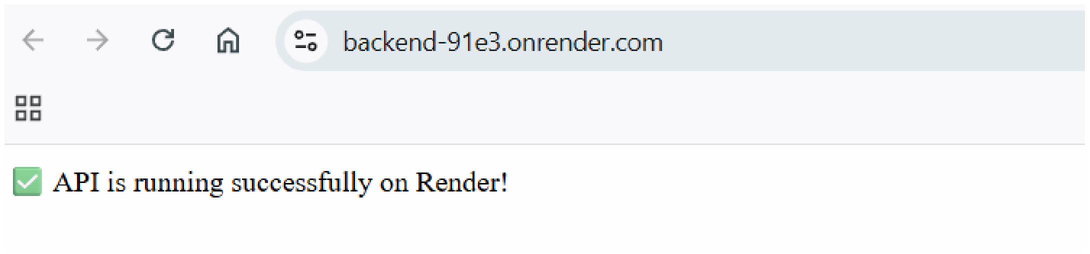
- Verified that CRUD operations in MongoDB reflected accurately in the collections through MongoDB Compass.
- Ensured that data integrity was maintained when performing updates or deletions.
- Checked user authentication flow with encrypted password storage and token generation.

3.5.2 Deployment Phase

After successful testing, the application was deployed on cloud platforms for public access and demonstration.

1. Backend Deployment (Render):

- The Node.js and Express.js server was deployed on Render, which automatically built and hosted the backend service.
- Environment variables such as MongoDB connection URI and JWT secret keys were securely configured.
- The deployed API endpoints were tested again post-deployment to confirm proper connectivity and functionality.

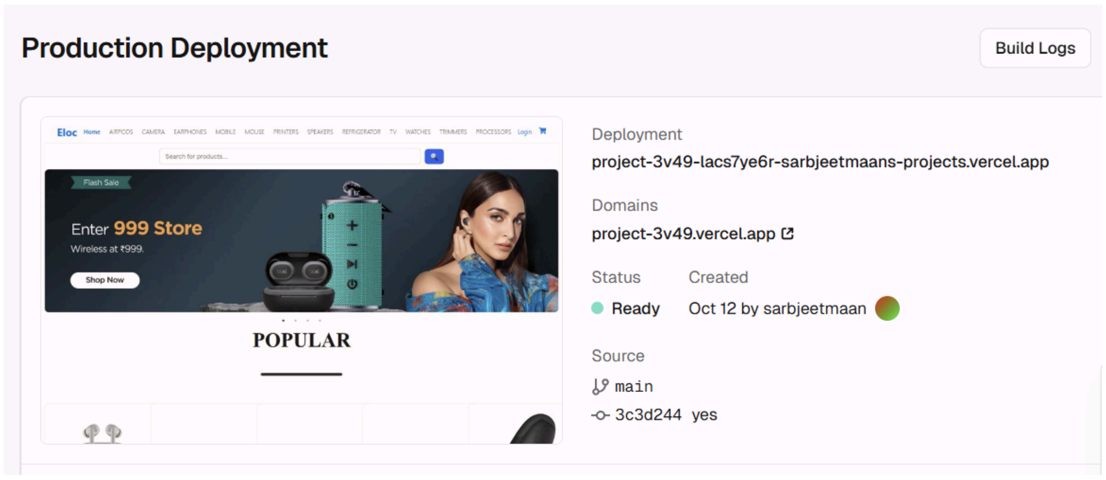


✅ API is running successfully on Render!

Figure 3.13 live backend API response from render .

2. Frontend Deployment (Vercel):

- The React.js frontend was deployed on Vercel, providing a live and accessible web interface for users.
- Environment variables were configured to connect the frontend with the deployed backend API.
- The live version of the website was tested to ensure smooth operation and proper communication between all components.


Production Deployment

Build Logs

Deployment
project-3v49-lacs7ye6r-sarbjeetmaans-projects.vercel.app

Domains
project-3v49.vercel.app

Status Ready

Created Oct 12 by sarbjeetmaan

Source
main
3c3d244 yes

Figure 3.14 Live frontend interface hosted on Vercel

3.5.3 Discussion:

Testing and deployment were essential to ensure that all parts of the MERN stack application functioned cohesively in real-world conditions. Postman validated the API reliability, while manual and responsive testing confirmed user interface stability.

Deploying the backend on Render and the frontend on Vercel demonstrated practical knowledge of cloud-based web hosting and continuous deployment workflows, enabling public accessibility and professional-grade presentation of the project.

3.6 Overall Discussion

The completion of the MERN stack e-commerce web application marked the successful culmination of the industrial training program. The project provided comprehensive exposure to full-stack web development, from designing the frontend interface to implementing backend APIs and managing a cloud-hosted database.

The integration of MongoDB, Express.js, React.js, and Node.js enabled the development of a modern, scalable, and efficient application that reflects the workflow of real-world software projects. Each phase of the project—frontend development, backend implementation, database management, testing, and deployment—contributed to strengthening both technical and analytical skills.

3.6.1 Key Observations and Achievements

- Full-Stack Integration:

  The seamless communication between frontend, backend, and database layers demonstrated the power of the MERN stack in creating complete web applications using a single programming language—JavaScript.

- Efficient Frontend Design:

  The React.js framework enabled the creation of a dynamic and user-friendly interface, ensuring responsiveness and interactivity across devices.

- Robust Backend and Database Management:

  The use of Node.js and Express.js facilitated efficient request handling and API management, while MongoDB provided flexibility in handling user and product data.

- Security and Authentication:

  Implementing JWT-based authentication enhanced data security and ensured that only authorized users could access protected routes.

- Practical Exposure to Deployment:

  Deploying the project on Render (backend) and Vercel (frontend) provided valuable experience in modern cloud deployment practices, preparing for real-world software delivery environments.

- API and Functional Testing:

  Through Postman and manual verification, all modules were tested to ensure data consistency, error handling, and smooth interaction between client and server components.

3.6.2 Learning Outcomes

The project served as a practical demonstration of full-stack development concepts and provided real-world learning in:

- Developing complete web applications using the MERN stack.
- Applying problem-solving and debugging techniques in code implementation.
- Managing version control using Git and GitHub for professional collaboration.
- Understanding project structuring, API lifecycle, and deployment workflows.
- Strengthening logical thinking, coding efficiency, and adaptability to new technologies.

3.6.3 Conclusion

The e-commerce web application developed during the training not only met its functional objectives but also provided an in-depth understanding of full-stack development.
It successfully bridged the gap between theoretical learning and industrial application, enhancing both technical proficiency and practical experience.

This training experience has significantly contributed to professional growth and has laid a solid foundation for future endeavors in web development, particularly in building scalable, interactive, and user-centric applications using the MERN stack.

CHAPTER 4: CONCLUSION AND FUTURE SCOPE

4.1 Conclusion

The one-month industrial training on MERN Stack Development at Sensation Solutions provided a comprehensive understanding of full-stack web application development. Through the development of a fully functional e-commerce web application, I gained practical experience in integrating frontend, backend, and database technologies within a single, unified environment.

This training enhanced my technical proficiency in React.js, Node.js, Express.js, and MongoDB, as well as my understanding of RESTful API development, authentication, and deployment processes. I also became proficient in using development tools such as Visual Studio Code, Postman, MongoDB Compass, GitHub, and cloud platforms like Render and Vercel.

The hands-on project enabled me to apply theoretical concepts in a practical setting, improving my skills in coding, debugging, and problem-solving. The addition of an Admin Dashboard further strengthened my understanding of role-based authentication and management features in modern web applications.

Overall, this training served as an excellent bridge between academic learning and real-world application development. It not only enhanced my technical skills but also provided valuable exposure to professional workflows, project management, and software deployment practices—preparing me for a successful career in full-stack web development.

4.2 Future Scope

The e-commerce web application developed during this training demonstrates the core functionality of a modern online shopping platform, including product management, shopping cart operations, user authentication, and an admin dashboard for administrative control. However, there are several opportunities for further enhancement and scalability.

Future enhancements may include:

- Payment Gateway Integration:

  Incorporating secure online payment systems such as Razorpay, Stripe, or PayPal to enable real-time transactions and improve user convenience.
- Order Tracking and Delivery Management:

  Adding features for users to track order status, expected delivery dates, and shipment updates to enhance post-purchase experience.
- Advanced Admin Dashboard:

  Expanding the admin panel to include sales analytics, graphical reports, user management, and product performance tracking for better business insights.
- Product Reviews and Ratings:

  Allowing users to review and rate products, improving engagement and providing feedback for future enhancements.
- Inventory and Stock Management:

  Automating inventory updates and notifications for low-stock products to streamline store operations.
- AI-Based Recommendation System:

  Implementing machine learning algorithms to suggest products based on user preferences, browsing history, and purchase patterns.

- Progressive Web Application (PWA):

  Converting the application into a PWA to allow offline usage and installation on mobile devices, improving accessibility.
- Enhanced Security and Performance:

  Applying advanced security measures such as SSL certificates, data encryption, and optimization for faster load times.

Summary

The project lays a solid foundation for a scalable e-commerce solution built using the MERN stack. With future enhancements such as payment integration, intelligent recommendations, and expanded admin analytics, the application can evolve into a comprehensive and feature-rich commercial platform. The training not only strengthened technical skills but also fostered a mindset of continuous learning and innovation in full-stack web development.

REFERENCES

[1] MongoDB Inc., *MongoDB: The Developer Data Platform*, [Online]. Available: https://www.mongodb.com/docs/.

[2] Express.js Foundation, *Express.js – Fast, Unopinionated, Minimalist Web Framework for Node.js*, [Online]. Available: https://expressjs.com/.

[3] Meta Platforms Inc., *React.js – A JavaScript Library for Building User Interfaces*, [Online]. Available: https://react.dev/.

[4] Node.js Foundation, *Node.js Documentation – JavaScript Runtime Built on Chrome's V8 Engine*, [Online]. Available: https://nodejs.org/en/docs/. Accessed: Oct. 2025.

[5] Render Inc., *Render: Cloud Deployment Platform for Developers*, [Online]. Available: https://render.com/.

[6] Vercel Inc., *Vercel: Frontend Deployment and Hosting Platform*, [Online]. Available: https://vercel.com/docs.

[7] Postman Inc., *Postman API Platform – Design, Test, and Automate APIs*, [Online]. Available: https://www.postman.com/.

[8] GitHub Inc., *GitHub Documentation – Collaboration and Version Control using Git*, [Online]. Available: https://docs.github.com/.

[9] Mozilla Foundation, *MDN Web Docs – Web Development Resources for JavaScript, HTML, and CSS*, [Online]. Available: https://developer.mozilla.org/.

[10] W3Schools, *W3Schools Online Web Development Tutorials*, [Online]. Available: https://www.w3schools.com/.

[11] Brad Traversy and Florin Pop, *MERN Stack Front To Back: Full Stack React, Redux & Node.js*, Udemy Online Course, 2023.

[12] A. Banks and E. Porcello, *Learning React: Modern Patterns for Developing React Apps*, 3rd ed. Sebastopol, USA: O'Reilly Media, 2023.

[13] A. Beaulieu, *Learning MongoDB*, 2nd ed. Sebastopol, USA: O'Reilly Media, 2021.

[14] A. Williams, *Node.js 18 Cookbook: Over 130 Recipes to Build Scalable and Efficient Node.js Applications*, Birmingham, UK: Packt Publishing, 2022.

APPENDIX

A.1 Code Snippets

Backend Route Example – Add Product (/addproduct):

```
app.post('/addproduct', async (req, res) => {
  try {
    const products = await Product.find({});
    const id = products.length > 0 ? products[products.length - 1].id + 1 : 1;
    const product = new Product({
      id,
      name: req.body.name,
      images: req.body.images,
      category: req.body.category,
      new_price: req.body.new_price,
      old_price: req.body.old_price,
    });
    await product.save();
    res.json({ success: true, product });
  } catch (err) {
    console.error(err);
    res.status(500).json({ success: false, message: "Server Error" });
  }
});
```

Frontend Component Example – Product Card (ProductCard.jsx):

```
const ProductCard = ({ product }) => {
  return (
    <div className="product-card">
      <img src={product.images[0]} alt={product.name} />
      <h3>{product.name}</h3>
      <p>Category: {product.category}</p>
      <p>Price: ${product.new_price}</p>
    </div>
  );
};
```

A.2 Database Schemas

User Collection: Stores user information with fields for username, email, password (hashed), and role (user/admin).

Product Collection: Contains product details including id, name, images, category, prices, availability, and date added.

Cart Collection: Links products to users with a userId and an array of items in the cart.

A.3 Postman Collection

- All backend APIs (user, product, cart) were tested using Postman.
- Collection includes: signup, login, addproduct, allproducts, removeproduct, makeadmin.

- Exported collection file: MERN_Ecommerce_API.postman_collection.json

A.4 Additional Screenshots

1. Responsive Views: Mobile and tablet versions of the homepage.
2. Admin Workflows: Screenshots of product addition and user management in admin dashboard.
3. Database Queries: Sample filtered query results in MongoDB Compass.
4. Error Handling: Example of invalid login or unauthorized request response.

A.5 Environment Setup

Example .env file structure:

```
PORT=4000

MONGO_URL=mongodb+srv://<username>:<password>@cluster0.mongodb.net/myDatabase

JWT_SECRET=your_jwt_secret_key

BASE_URL=http://localhost:4000

NODE_ENV=development
```

Steps to run the project locally:

1. Clone the repository.
2. Run npm install in both backend and frontend directories.
3. Set environment variables as shown above.
4. Start backend: npm start
5. Start frontend: npm start

A.6 Testing Logs and Results

- API request/response examples from Postman.
- MongoDB Compass views showing CRUD operations and data integrity.
- Backend console logs demonstrating middleware execution, authentication checks, and server requests