

A REPORT OF 4-Weeks Industrial Training

at

S.T.E.P - Science & Technology Entrepreneurs' Park (STEP-GNDEC)

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:

Abhishek Singh

URN - 2435222

DEPARTMENT OF Computer Science & Engineering

GURU NANAK DEV ENGINEERING COLLEGE LUDHIANA

(An Autonomous College under UGC ACT)

Training Certificate

Declaration

I Abhishek Singh hereby declare that I have undertaken 4 weeks training **S.T.E.P - Science & Technology Entrepreneurs' Park (STEP-GNDEC)** during a period from 24th June, 2025 to 18th July, 2025 in partial fulfillment of requirements for the award of degree of B.Tech (Computer Science & Engineering) at GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA. The work which is being presented in the training report submitted to Department of Computer Science & Engineering at GURU NANAK DEV ENGINEERING COLLEGE, LUDHIANA is an authentic record of training work.

Signature of the Student

The 4 weeks industrial training Viva–Voce Examination of _____ has been held on _____ and accepted.

Signature of Internal Examiner

Signature of External Examiner

Abstract

This report presents a comprehensive overview of the four-week industrial training program on "Web Development Using MERN", conducted at S.T.E.P - Science & Technology Entrepreneurs' Park (STEP-GNDEC). The program aimed to provide hands-on experience with the MERN stack- MongoDB, Express.js, React.js, and Node.js and equip students with practical skills to build modern full-stack web applications.

During the training, I developed a B2B e-commerce platform designed for interaction between Retailers, Sellers, and Admins. Key features of the project include role-based authentication, product upload and review mechanisms, secure login and registration with document upload, and dynamic product categorization managed via MongoDB.

This training significantly strengthened my understanding of frontend and backend integration, database design, RESTful API development, and deployment practices. The practical exposure gained through this project has enhanced my ability to build scalable, responsive, and user-friendly web applications using the MERN stack.

Acknowledgement

I would like to extend my sincere gratitude to **Prof. Dr. Satinderpal Singh**, Trainer at the GNDEC MERN Stack Cell, for his unwavering support, expert guidance, and insightful feedback throughout the duration of the training.

I am also thankful to the **Department of Computer Science and Engineering, GNDEC Ludhiana**, for offering me the opportunity to participate in this industrial training and for providing the necessary resources and environment to learn and grow.

My heartfelt appreciation goes to my peers and teammates for their cooperation, encouragement, and teamwork during the training journey.

About the Institute

S.T.E.P – Science & Technology Entrepreneurs’ Park, GNDEC Ludhiana

Science & Technology Entrepreneurs’ Park (STEP-GNDEC) is a premier incubation and innovation hub established at **Guru Nanak Dev Engineering College (GNDEC), Ludhiana**. It is jointly promoted by the **Department of Science and Technology (DST), Government of India, Punjab State Council for Science & Technology (PSCST), and National Science & Technology Entrepreneurship Development Board (NSTEDB)**. The institute plays a pivotal role in nurturing entrepreneurship and technological innovation among students and young professionals.

Situated within the **autonomous GNDEC campus**, STEP-GNDEC offers a dynamic ecosystem that fosters intellectual curiosity, skill development, and business incubation. The center provides cutting-edge facilities such as co-working spaces, startup incubation cells, innovation labs, and a range of short-term training and diploma programs.

The **primary objectives** of STEP-GNDEC include:

- Promoting entrepreneurship among science and technology students.
- Providing infrastructure and mentorship for startups and innovative ideas.
- Offering training, internships, and diploma programs in emerging technologies.
- Facilitating idea-to-market transitions through funding and administrative support.

With over **38 years of experience** and more than **25,000 students** trained, STEP-GNDEC is one of the most established and impactful entrepreneurship cells in North India. The institute also organizes regular bootcamps, tech festivals (such as STEP-TECH), Web3 promotion events, and innovation challenges that empower students to develop real-world solutions.

Under the leadership of **Dr. Arvind Dhingra (Executive Director, STEP-GNDEC)**, the institute continues to advance its mission of producing skilled professionals and innovative entrepreneurs equipped to meet the demands of a rapidly evolving tech-driven economy.

Index

S. No.	Contents	Page No
1	Certificate	<i>i</i>
2	Declaration	<i>ii</i>
3	Abstract	<i>iii</i>
4	Acknowledgement	<i>iv</i>
5	About Institute	<i>v</i>
6	Index	<i>vi</i>
7	List of Figures	<i>vii</i>
8	List of Tables	<i>vii</i>
9	Definitions, Acronyms and Abbreviations	<i>viii</i>
11	Chapter 1 – Introduction	<i>12-29</i>
12	Chapter 2 - Training Work Undertaken	<i>30-</i>
13	Chapter 3 - Results and Discussion	
14	Chapter 4 – Conclusion and Future Scope	
	References	

List of Figures

Figure No.	Figure Name	Page No
1	Logo of STEP GNDEC	
2	MERN Stack Architecture	
3	CRUD Operation Overview	22

List of Tables

Table No.	Table Name	Page No
1	Week-wise Training Schedule Summary	
2	Key Features of Each MERN Stack Technology	

Definitions, Acronyms and Abbreviations

Term/Acronym	Definition
MERN	A technology stack consisting of MongoDB, Express.js, React.js, and Node.js
MongoDB	A NoSQL database that stores data in flexible, JSON-like documents
Express.js	A web application framework for Node.js used to build APIs and server-side applications
React.js	A front-end JavaScript library for building user interfaces, maintained by Meta (Facebook)
Node.js	A JavaScript runtime built on Chrome's V8 engine, used for server-side development
API	Application Programming Interface – a set of functions that allow applications to communicate
CRUD	Create, Read, Update, Delete – basic operations of persistent storage
UI	User Interface – the layout and visual elements through which users interact with an application
UX	User Experience – the overall experience and satisfaction of a user using the application
HTML	HyperText Markup Language – standard language for creating web pages
CSS	Cascading Style Sheets – used for describing the presentation of web pages
JS	JavaScript – a scripting language used to create and control dynamic website content
eB2B	Ecommerce Business-to-Business – a type of commerce between businesses rather than between a business and individual consumers
STEP- GNDEC	Science & Technology Entrepreneurs' Park – Guru Nanak Dev Engineering College, Ludhiana.

Chapter 1: Introduction

1.1 Overview of Industrial Training

Industrial training plays a crucial role in bridging the gap between theoretical knowledge and practical implementation in professional education. It helps students gain hands-on experience, develop technical skills, and understand the actual working environment of the industry. Such training empowers students to become industry-ready and increases their chances of employment in today's competitive job market.

The 4-week industrial training was conducted at **S.T.E.P – Science and Technology Entrepreneurs' Park**, Guru Nanak Dev Engineering College (GNDEC), Ludhiana. The training was focused on **MERN Stack Development**, a powerful and modern technology stack used for full-stack web development. This training provided a comprehensive learning environment for understanding both frontend and backend web development using tools and technologies relevant to today's IT industry.

Objectives of the 4-week training:

- To understand the fundamentals of full-stack web development.
- To gain practical experience in building dynamic web applications using the MERN Stack (MongoDB, Express.js, React.js, and Node.js).
- To bridge the gap between academic concepts and real-world development practices.
- To complete a full-fledged project based on B2B E-commerce using MERN Stack.
- To improve problem-solving and debugging skills.
- To collaborate and communicate effectively in a development environment.

1.1.1 Table 1: Week-wise Training Schedule Summary

Week	Topics Covered
Week 1	Introduction to MERN stack, React setup, basic HTML/CSS revision
Week 2	Web Development using HTML,Github responsive UI using Bootstrap.
Week 3	React components, props, state, routing, responsive UI using Tailwind
Week 4	Working on actual project building (B2B E-commerce)

1.1.1 Relevance of MERN Stack in Modern Web Development

The MERN Stack is widely adopted in the industry due to its use of JavaScript across the entire development process — from client-side to server-side. React offers a component-based architecture that simplifies frontend development, while Node.js and Express provide a scalable backend environment. MongoDB, being a NoSQL database, ensures flexibility in data management. The demand for MERN Stack developers is increasing rapidly, making it a valuable skill set for aspiring web developers.



Figure 1: STEP logo (Source: <https://stepgndec.com/>)

1.2 About the Training Program

1.2.1 Web Development using MERN

Web development refers to the process of building and maintaining websites and web applications that run online on a browser. It involves a combination of frontend (client-side) and

backend (server-side) technologies. The frontend focuses on what users see and interact with, while the backend handles data storage, server logic, and communication with databases.

Modern web development has evolved beyond static pages to include interactive, data-driven applications. Full-stack development combines both frontend and backend skills, enabling developers to manage all aspects of web applications.

1.2.2 Why MERN Stack?

The MERN stack is a powerful combination of technologies that enables full-stack development using a single programming language — **JavaScript** — for both client-side and server-side code. It is a popular choice among developers and companies due to its speed, flexibility, and large community support.

1.2.3 M: MongoDB

MongoDB is a NoSQL, document-oriented database that stores data in flexible, JSON-like format called BSON. It allows for easy scalability and is well-suited for handling large amounts of unstructured data.

- **Advantages:** Schema-less structure, horizontal scaling, fast read/write operations.
- **Role in MERN:** Stores data for the backend (e.g., user accounts, product details).

1.2.4 E: Express.js

Express.js is a lightweight and flexible Node.js web application framework that simplifies building backend APIs and handling HTTP requests.

- **Advantages:** Minimalist framework, middleware support, fast routing.
- **Role in MERN:** Acts as the backend server framework to handle business logic.

1.2.5 R: React.js

React is a powerful JavaScript library developed by Facebook for building interactive and dynamic user interfaces.

- **Advantages:** Component-based architecture, virtual DOM, fast rendering.
- **Role in MERN:** Powers the frontend of the application for an interactive UI/UX.

1.2.6 N: Node.js

Node.js is a runtime environment that allows JavaScript to be executed on the server-side.

- **Advantages:** Non-blocking I/O, event-driven architecture, fast performance.
- **Role in MERN:** Runs the backend code, connects to the database, and serves the frontend.

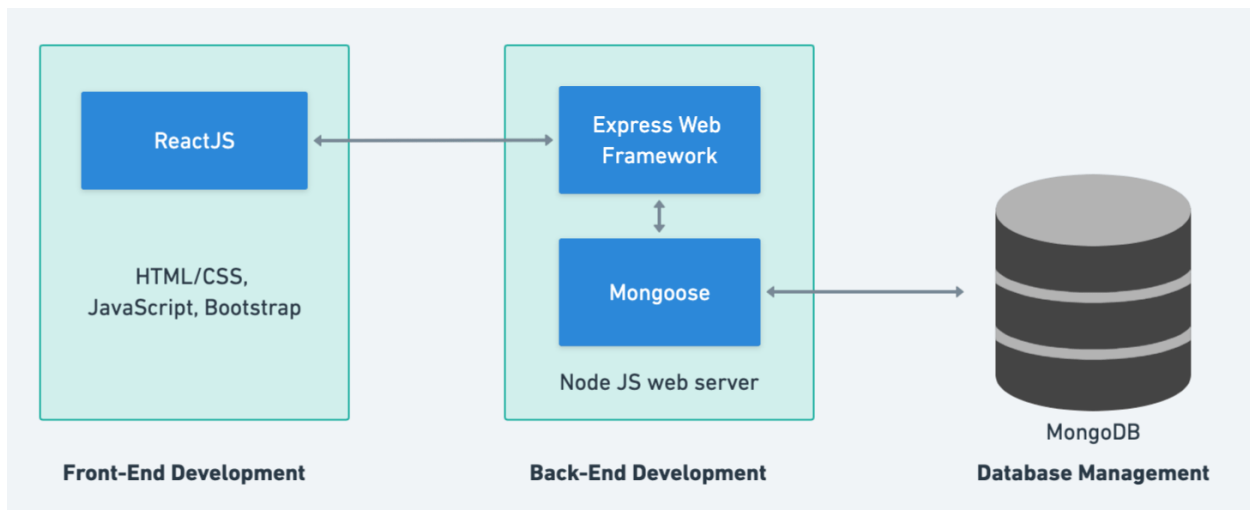


Figure 2: MERN Stack Architecture (Source: <https://ellow.io/>)

1.2.7 Goals of the Training Program

The core goals of the training program were:

- To provide hands-on experience in full-stack web development.
- To introduce students to industry-relevant tools and workflows.
- To build a complete web application using the MERN Stack.

- To strengthen coding skills, debugging abilities, and problem-solving techniques.
- To prepare students for real-world development environments and internships/jobs.

1.2.8 Key Areas Covered During Training

The training program was structured to move from basics to advanced topics in a progressive manner. Major areas covered include:

- HTML, CSS, and Bootstrap, Tailwind for UI Design.
- JavaScript fundamentals and DOM manipulation.
- React.js setup, component creation, state and props handling.
- Express.js for API development and server routing.
- MongoDB for database integration and CRUD operations.
- Node.js for backend logic and application execution.
- Full-stack integration of frontend, backend, and database.
- Deployment of the final project to hosting platforms.

1.2.9 Table 1: Key Features of Each MERN Stack Technology

Technology	Type	Language Used	Key Features	Role in Stack
MongoDB	NoSQL Database	JSON/BSON	Flexible schema, scalable, fast data access	Stores application data
Express.js	Backend Framework	JavaScript	Lightweight, supports middleware, RESTful APIs	Server logic & routing
React.js	Frontend Library	JavaScript	Component-based, virtual DOM, reusable UI parts	User interface builder
Node.js	Runtime Environment	JavaScript	Event-driven, non-blocking I/O, scalable	Executes backend code

1.3 Theoretical Concepts Learned

During the MERN Stack industrial training, we explored a wide range of theoretical and practical concepts crucial for full-stack web development. The training covered frontend and backend technologies, database operations, and authentication systems — equipping us to build modern, scalable web applications.

1.3.1 Frontend Development

1.3.1.1 HTML, CSS, JavaScript, Bootstrap

- **HTML (HyperText Markup Language):** The base structure of any website. It organizes web content using tags like headings, paragraphs, forms, images, etc.
- **CSS (Cascading Style Sheets):** Used to style HTML pages. It controls the layout, fonts, colors, and responsiveness of the site.
- **JavaScript:** A scripting language that adds interactivity to websites. It handles events, validates forms, manipulates DOM elements, and enables dynamic behavior.
- **Bootstrap:** A CSS framework that offers prebuilt components like buttons, cards, and grid systems. It helps in building responsive and mobile-friendly web designs quickly.
- **Tailwind CSS**

Tailwind CSS is a utility-first CSS framework that allows developers to build custom designs directly in HTML markup using utility classes.

Key Features:

- Highly customizable and fast.
- Encourages consistency using design tokens.
- Removes the need for writing custom CSS in many cases.
- Works well with modern frameworks like React.

1.3.1.2 React.js

React is a JavaScript library for building user interfaces. It allows developers to create reusable UI components and efficiently update the DOM.

Key Concepts:

- **Components:** Fundamental units of a React app. Can be reused and nested.
- **Props:** Short for "properties", used to pass data to components.
- **State:** Local data storage within components.
- **Hooks:** Introduced in React 16.8. Common hooks include:
 - `useState()` for state management
 - `useEffect()` for side-effects and lifecycle handling
- **Routing:** `react-router-dom` is used to navigate between pages in a single-page application.

React Components using Tailwind

```
• import { Link } from "react-router-dom";  
• import SectionHeadStyle from "../SectionHeadStyle";  
• import ProductCard from "../ProductCard";  
• const ProductSection = ({ heading, img, products, linkTo }) => {  
•   return (  
•     <div className="w-full h-fit mt-10">  
•       <SectionHeadStyle img={img} heading={heading} />  
•       <div className="grid gap-6 grid-cols-2 sm:[grid-template-columns:repeat(auto-fit,_minmax(280px,_1fr))]">  
•         {products.map((product, index) => (  

```

- `<ProductCard key={index} product={product} />`
- `)))`
- `{/* See More */}`
- `<div className="min-h-[340px] flex items-center justify-center text-themeColor italic font-bold text-sm border-4 border-dashed rounded-xl p-6">`
- `<Link to={linkTo}>`
- `<button className="border-2 cursor-pointer p-5 border-themeColor">`
- `See More Products`
- `</button>`
- `</Link>`
- `</div>`
- `</div>`
- `</div>`
- `);`
- `};`
- `export default ProductSection;`

1.4.2 Backend Development

Node.js and Express.js

- **Node.js:** A JavaScript runtime that allows backend development using JavaScript. It uses an event-driven, non-blocking I/O model.

- **Express.js:** A minimal Node.js framework for handling backend logic and routing.

Key Concepts:

- **Server Setup:** Creating a server using `express()` and listening on a port.
- **Routing:** Handling endpoints using `app.get()`, `app.post()`, etc.
- **Middleware:** Functions that execute between request and response (e.g., logging, authentication).

REST API Development

RESTful APIs allow the frontend and backend to communicate via HTTP. During training, REST APIs were developed to manage users, products, and authentication features.

Examples:

- `POST /api/register`
- `GET /api/products`
- `PUT /api/product/:id`
- `DELETE /api/product/:id`

```
• const express = require("express");
• const cors = require("cors");
• const mongoose = require("mongoose");
• const dotenv = require("dotenv");
• const cookieParser = require("cookie-parser");
• const path = require("path");
• dotenv.config();
• const app = express();
• // CORS: allow frontend origin
```

- `app.use(cors({ origin: "http://localhost:5173", credentials: true }));`
- `app.use(express.json());`
- `app.use(cookieParser());`
- `app.use("/uploads", express.static(path.join(__dirname, "uploads"))); // serve static files`
- *// MongoDB connection*
- `mongoose`
- `.connect(process.env.MONGO_URI)`
- `.then(() => console.log("MongoDB connected"))`
- `.catch((err) => console.error(err));`
- *// Routes*
- `app.use("/api/auth", require("./routes/authRoutes"));`
- *// Start server*
- `const PORT = process.env.PORT || 5000;`
- `app.listen(PORT, () => console.log(`Server running on port ${PORT}`));`
-

1.4.3 Database Handling

MongoDB and Mongoose

- **MongoDB:** A document-based NoSQL database used to store application data in a flexible format.
- **Mongoose:** A Node.js library that provides schema-based solutions to model MongoDB data.

CRUD Operations:

- **Create:** `Model.create()`

- **Read:** Model.find() or Model.findById()
- **Update:** Model.findByIdAndUpdate()
- **Delete:** Model.findByIdAndDelete()

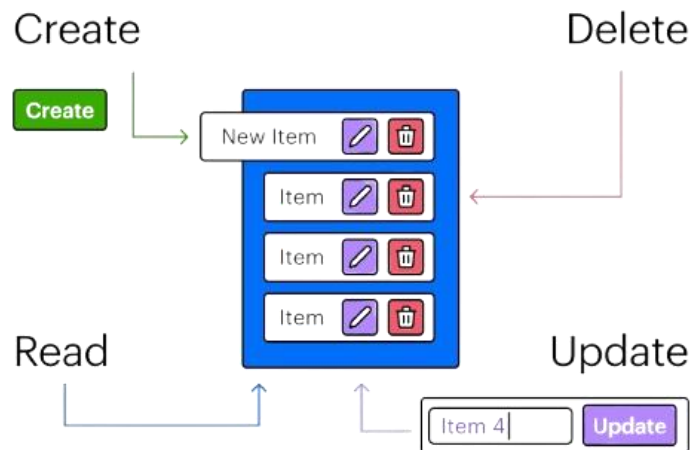


Figure 2: CRUD Operation Overview (Source: [https:// ramotion.com/](https://ramotion.com/))

Mongoose schema definition

- ```

• const mongoose = require("mongoose");

• const productSchema = new mongoose.Schema({

• name: String,

• category: String,

• price: Number,

• images: [String],

• });

• module.exports = mongoose.model("Product", productSchema);

```

#### 1.4.4 Authentication

Authentication ensures that users can securely log in and access resources.

## Login/Register System

- Users can register with email/mobile and password.
- Passwords are hashed using **bcrypt** before being stored in MongoDB.

## JWT (JSON Web Token)

- After login, a token is generated using **jsonwebtoken**.
- This token is sent in headers (Authorization) for secure access to protected routes.

### Steps:

1. User submits login credentials.
2. Server validates and returns a token.
3. Token is stored on the client (usually in localStorage).
4. Client sends token with requests to access protected data.

## JWT-based Login Route Setup (Express + JWT)

```
• // Required Modules
• const express = require('express');
• const jwt = require('jsonwebtoken');
• const bcrypt = require('bcryptjs');
• const User = require('./models/User'); // Mongoose user model
•
• const router = express.Router();
•
• // Secret key (in practice, store this in environment variables)
• const JWT_SECRET = 'your_jwt_secret_key';
•
```

- *// LOGIN ROUTE*
- `router.post('/login', async (req, res) => {`
- `const { email, password } = req.body;`
- `try {`
- *// Find user by email*
- `const user = await User.findOne({ email });`
- 
- `if (!user) return res.status(404).json({ message: 'User not found' });`
- *// Compare entered password with hashed password*
- `const isMatch = await bcrypt.compare(password, user.password);`
- `if (!isMatch) return res.status(400).json({ message: 'Invalid credentials' });`
- 
- *// Generate JWT token*
- `const token = jwt.sign({ id: user._id }, JWT_SECRET, { expiresIn: '1h' });`
- 
- `res.json({ token, user: { id: user._id, email: user.email } });`
- `} catch (error) {`
- `res.status(500).json({ message: 'Server error' });`
- `}`
- `});`
- 
- `module.exports = router;`



## 1.5 Tools and Technologies Used

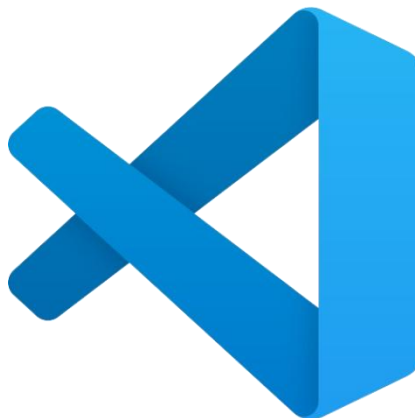
The MERN Stack industrial training involved the use of various tools and technologies that helped streamline the development process. These tools were used for coding, backend development, database management, API testing, UI design, and version control. Familiarity with such tools is essential for any modern web developer.

### 1. Visual Studio Code

**Visual Studio Code (VS Code)** is a powerful and widely-used code editor developed by Microsoft. It supports a wide range of programming languages and has a rich extension marketplace.

#### Key Features:

- Syntax highlighting, code completion, and IntelliSense
- Integrated terminal and Git support
- Extensions for ESLint, Prettier, Tailwind CSS, etc.



**Figure 3:** VS code editor logo (Source: <https://vscode.com/>)

### 2. Node.js

**Node.js** is a JavaScript runtime that allows execution of JavaScript code outside the browser. It is commonly used for developing backend services and APIs.

### Why Used in Training:

- Running Express server
- Installing dependencies using npm
- Building REST APIs



**Figure 4:** Node js logo (Source: <https://nodejs.com/>)

### 3. MongoDB

**MongoDB** is a NoSQL database that stores data in JSON-like format. It allows developers to handle flexible and scalable data models.

#### Tools Used:

- **MongoDB Compass:** A GUI to explore and manage the database.
- **Mongoose:** An ODM used to define schemas and perform CRUD operations in Node.js.



**Figure 5:** MongoDB logo (Source: <https://mongodb.com/>)

#### 4. Git & GitHub

**Git** is a distributed version control system, and **GitHub** is a hosting platform for Git repositories.

##### **How Used in Training:**

- Track project changes and revisions
- Push code to GitHub repositories
- Collaborate using branches and pull requests



**Figure 6:** GitHub logo (Source: <https://github.com/>)

#### 5. Postman

**Postman** is a tool used to test and debug RESTful APIs.

##### **Use Cases:**

- Testing login and register routes
- Sending custom headers and payloads
- Checking response time, status, and data format

#### 6. Tailwind CSS

**Tailwind CSS** is a utility-first CSS framework that enables rapid UI development directly in HTML/JSX markup.

**Why Used:**

- Fast, responsive designs using utility classes
- No need for custom CSS files
- Works seamlessly with React components



**Figure 6:** Tailwind logo (Source: <https://tailwind.com/>)

## **7. Operating System**

Most of the development was carried out on **Windows 10/11 OS**, which provided a compatible environment for installing tools like Node.js, MongoDB, VS Code, and Git. The OS also supported Chrome Developer Tools for frontend debugging.



**Figure 6:** Tailwind logo (Source: <https://mswindows.com/>)

## 8. Hardware Configuration Used

- Processor (CPU) Intel Core i5 (4 cores or more)
- RAM 8 GB
- Storage 256 GB SSD and 1000 Hard Disk
- Display Full HD (1920x1080)
- Operating System Windows 11
- Internet Stable Wi-Fi connection (5 Mbps+)

## Chapter 2: TRAINING WORK UNDERTAKEN

This chapter provides a detailed account of the step-by-step learning process, the tools and technologies practiced, the development methodology followed, and the final project undertaken — a B2B E-Commerce Website using the MERN stack.

### 2.1 Introduction to MERN Stack

#### 2.1.1 Overview of MERN Stack

The **MERN stack** is a popular set of JavaScript-based technologies used to develop modern web applications. The term **MERN** is an acronym for the four key technologies used:

##### Technology Role

**MongoDB** NoSQL database for storing application data

**Express.js** Backend framework running on Node.js

**React.js** Frontend library for building user interfaces

**Node.js** JavaScript runtime for building server-side apps

Each component in the MERN stack is open-source and supports end-to-end development using JavaScript. This allows developers to build the frontend, backend, and database interactions using a single programming language — JavaScript.

#### 2.1.2 Why MERN for Modern Web Development?

MERN has become the go-to stack for developers due to the following reasons:

- **Full JavaScript Stack:** All four technologies use JavaScript, enabling seamless communication between client and server.
- **Component-Based Frontend (React):** React allows building dynamic and responsive UIs using reusable components.

- **RESTful API Development (Express + Node):** Express provides a lightweight way to build APIs while Node.js handles concurrent requests efficiently.
- **Scalable and Flexible Database (MongoDB):** MongoDB, being a NoSQL database, offers high scalability and schema-less structure suitable for real-time and growing applications.
- **Large Community Support:** Each tool in the stack is widely adopted and supported by active developer communities.
- **Rapid Development:** The unified language and ecosystem reduce context switching and speed up the development cycle.

### 2.1.3 MERN Stack Example: Simple Product API and Display

#### 1. MongoDB Schema (using Mongoose)

```

• // models/Product.js

• const mongoose = require('mongoose');

• const ProductSchema = new mongoose.Schema({
• name: String,
• price: Number,
• category: String
• });
• module.exports = mongoose.model('Product', ProductSchema);

```

#### 2. Express.js Backend API

```

• // server.js

• const express = require('express');
• const mongoose = require('mongoose');

```

- `const Product = require('./models/Product');`
- `const cors = require('cors');`
- 
- `const app = express();`
- `app.use(express.json());`
- `app.use(cors());`
- 
- `mongoose.connect('mongodb://localhost:27017/mern_demo');`
- 
- `app.get('/products', async (req, res) => {`
- `const products = await Product.find();`
- `res.json(products);`
- `});`
- 
- `app.listen(5000, () => console.log('Server running on port 5000'));`

### 3. React.js Frontend to Fetch Products

- `// App.jsx`
- `import { useEffect, useState } from 'react';`
- `function App() {`
- `const [products, setProducts] = useState([]);`
- `useEffect(() => {`
- `fetch('http://localhost:5000/products')`
- `.then(res => res.json())`



```

• .then(data => setProducts(data));
• }, []);
•
• return (
• <div className="p-5">
• <h1 className="text-2xl font-bold mb-4">Products</h1>
•
• {products.map(p => (
• <li key={p._id}>{p.name} - ₹{p.price}
•))}
•
• </div>
•);
• }
•
• export default App;

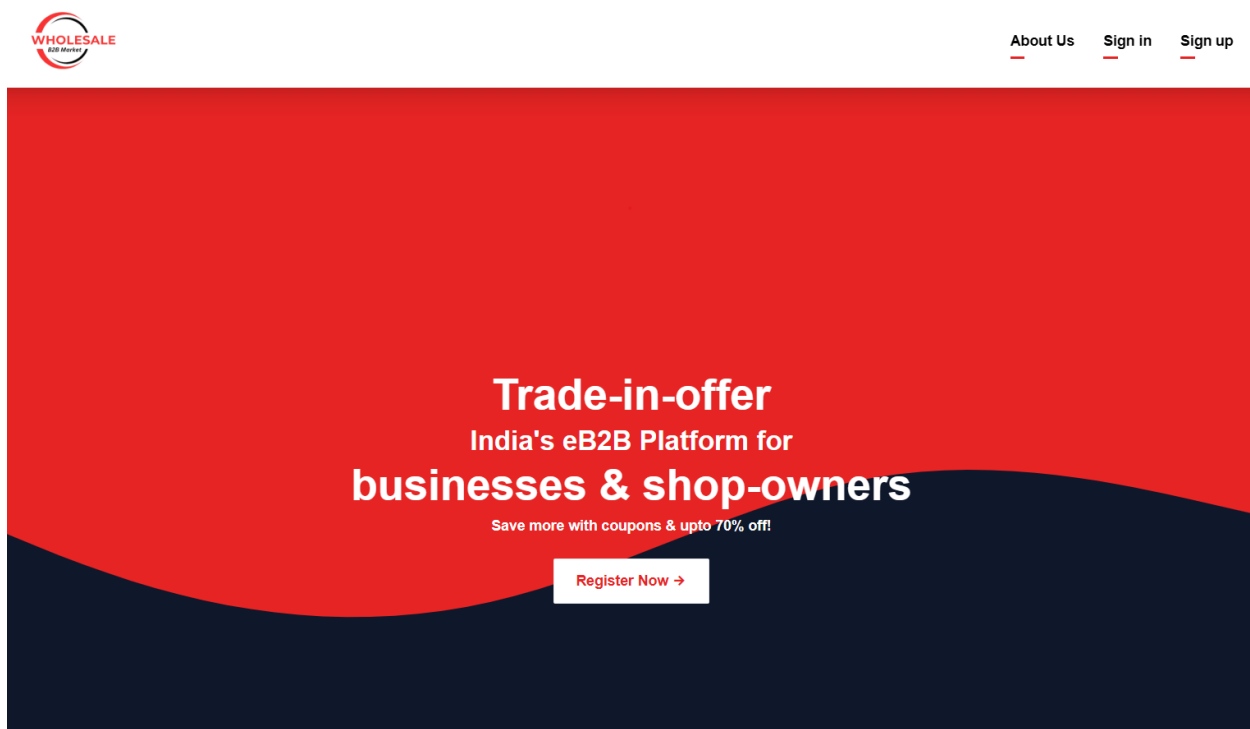
```

#### 2.1.4 Relevance to B2B Platforms

For B2B (Business-to-Business) platforms, performance, scalability, and responsiveness are crucial. MERN perfectly fits this requirement:

- **Real-Time Data Handling:** MongoDB and Node.js support real-time data operations, important for inventory, orders, and customer management in B2B systems.

- **Modular Architecture:** Using React and Express allows separating concerns and building scalable features like product management, admin panels, and role-based dashboards.
- **Custom APIs and Integrations:** MERN allows secure RESTful APIs which are essential for payment gateways, vendor integration, analytics, and user management in B2B solutions.
- **Cost-Efficient Development:** Being open-source and JavaScript-based, MERN reduces infrastructure and development costs, making it ideal for startups and growing businesses.



**Figure 6: Wholesale eB2B Market Dashboard**

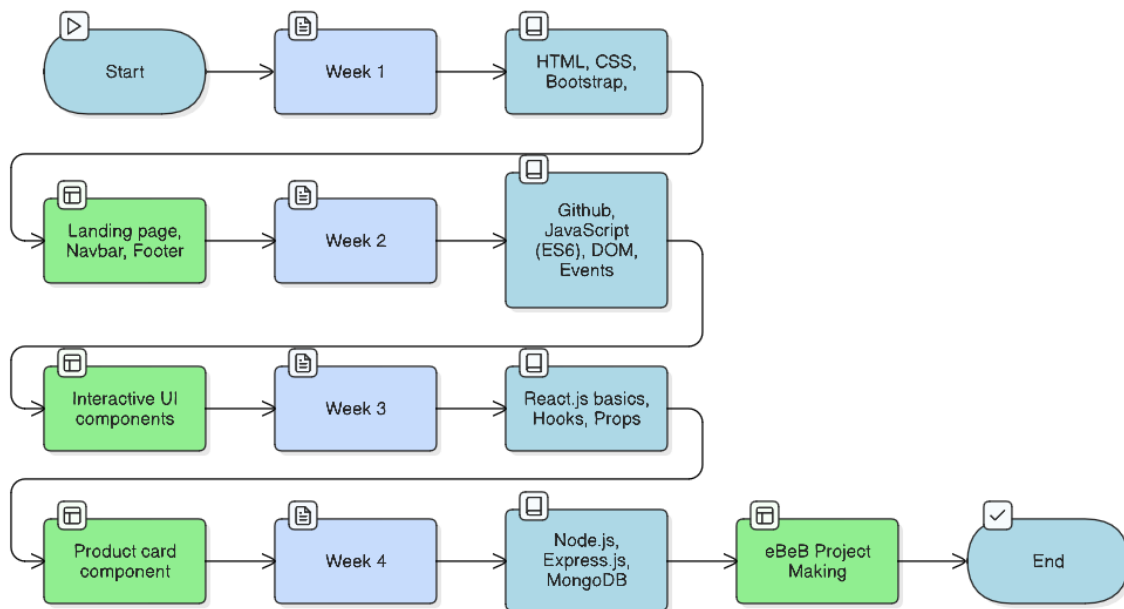
## **2.2 Sequential Learning Steps Followed**

### **2.2.1 Week-wise Training Structure**

Our 4-week industrial training followed a structured learning path, focusing on frontend to backend development in a progressive manner. Each week introduced new technologies and practical tasks aligned with real-world B2B e-commerce website development.

**Table 3: Weekly Breakdown of MERN Stack Training**

| Week   | Technologies/Topic Covers            | Practical Implementation               |
|--------|--------------------------------------|----------------------------------------|
| Week 1 | HTML, CSS, Bootstrap, Tailwind       | Practice, Landing Page, Navbar, footer |
| Week 2 | Github, JavaScript(ES6), DOM, Events | Practice, Intractive UI components     |
| Week 3 | React.js basics, Hooks, Props        | Practice                               |
| Week 4 | Working on Project (eB2B)            | MERN                                   |



**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)

## 2.3 Technologies Covered in Sequence

### 1. HTML, CSS, Bootstrap, Tailwind

- Built static layouts using HTML.
- Styled elements with CSS and Bootstrap utilities.

- Applied Tailwind for utility-first responsive design.
- **Activity:** Created a responsive login/register UI with Tailwind.

## 2. JavaScript (DOM, Events, ES6)

- Used modern JavaScript features (let, const, arrow functions).
- Manipulated DOM and handled form validations.
- Implemented event listeners for buttons, modals, etc.
- **Activity:** Created an image carousel using vanilla JS.

## 3. React.js Basics

- Understood JSX, props, and component-based architecture.
- Used useState and useEffect for state and lifecycle.
- Created dynamic product cards with props.
- **Activity:** Developed a functional ProductCard component.

### React Component Code

```
// App.jsx

import React from "react";

import { BrowserRouter as Router, Routes, Route } from "react-router-dom";

// Homepage components

import Navbar from "../components/Navbar";

import HeroSection from "../components/HeroSection";

import Quick from "../components/Quick";

import Category from "../components/Category";

import StartTrading from "../components/StartTrading";

import WelcomeSection from "../components/WelcomeSection";
```

```

// Pages

import Auth from "./pages/Auth";

import Products from "./pages/Products";

import CommomFooter from "./components/CommomFooter";

import About from "./pages/About";

import BrandLogoSlideBar from "./components/BrandLogoSlideBar";

import RetailerDashboard from "./pages/RetailerDashboard";

import SellerDashboard from "./pages/SellerDashboard";

import AdminDashboard from "./pages/AdminDashboard";

// Home page layout

const Home = () => {

 return (

 <>

 <Navbar />

 <HeroSection />

 <Quick />

 <Category />

 <StartTrading />

 <WelcomeSection />

 <BrandLogoSlideBar />

 <CommomFooter />

 </>
)
}

```

```

);

};

const App = () => {

return (

<Router>

<Routes>

 {/* Home page */}

 <Route path="/" element={<Home />} />

 {/* Login/Register combined page */}

 <Route path="/auth" element={<Auth />} />

 <Route path="/products" element={<Products />} />

 <Route path="/retailer-dashboard" element={<RetailerDashboard />} />

 <Route path="/seller-dashboard" element={<SellerDashboard />} />

 <Route path="/admin-dashboard" element={<AdminDashboard />} />

 {/* Product page (accessible after login) */}

 <Route path="/products" element={<Products />} />

 {/* About Us page */}

 <Route path="/about" element={<About />} />

</Routes>

</Router>

);

};

export default App;

```

#### 4. Node.js and Express.js

- Built server-side RESTful APIs with Express.
- Used routes, middleware, and error handling.
- Connected to MongoDB using Mongoose.
- **Activity:** Created /products GET API for frontend fetch.

Sample Express Route for fetching product data.

```
• const mongoose = require("mongoose");

• const productSchema = new mongoose.Schema({
• name: String,
• category: String,
• price: Number,
• images: [String],
• });

• module.exports = mongoose.model("Product", productSchema);
```

#### 5. MongoDB (CRUD Operations)

- Designed schema with Mongoose.
- Implemented CRUD operations for product data.
- Tested APIs using Postman and connected them to React.
- **Activity:** Full-stack integration to fetch and display MongoDB data in React.

##### Mongodb Connection Code

```
• const mongoose = require("mongoose");

• const connectDB = async () => {
• try {
• await mongoose.connect(process.env.MONGO_URI, {
```

- useNewUrlParser: true,
- useUnifiedTopology: true,
- });
- console.log("□ MongoDB connected");
- } **catch** (error) {
- console.error("□ MongoDB connection failed:", error.*message*);
- process.exit(1);
- }
- };
- module.exports = connectDB;

## 2.3 Methodology Followed

During the industrial training on web development using the MERN stack, a structured and agile-based methodology was followed to ensure efficient learning and timely project completion. The training was focused not only on learning concepts but also on applying them practically through real-world project development techniques.

### 1. Project Development Lifecycle

The development lifecycle followed in this training mimicked professional software development practices. It included the following phases:

#### 1. Requirement Gathering

Understanding project requirements, features, user roles (Retailer, Seller, Admin), and functional expectations.



## 2. Planning

Defining project roadmap, deciding milestones (weekly goals), and allocating tasks (frontend/backend).

## 3. Design

Wireframing UI components and page layouts. Using Tailwind CSS for faster, utility-first design.

## 4. Development

Implementing the project in phases: frontend using React.js, backend using Node.js and Express.js, and database operations with MongoDB.

## 5. Testing

Manual testing of frontend functionality, API responses using Postman, and data operations with MongoDB Compass.

## 6. Deployment

Final deployment using GitHub and platforms like Netlify for frontend and Render/Glitch/Vercel for backend.

## 2. Agile Development Model with Weekly Milestones

An Agile methodology was adopted to divide the 4-week training into manageable sprints. Each week had specific deliverables:

**Table 5: Weekly Goal/Milestone**

| Week   | Goal/Milestone                                           |
|--------|----------------------------------------------------------|
| Week 1 | Learn HTML, CSS, Bootstrap/Tailwind, create basic UI     |
| Week 2 | Learn JavaScript, React.js basics, and dynamic routing   |
| Week 3 | Learn Node.js, Express.js, connect to MongoDB            |
| Week 4 | Complete full-stack integration and project finalization |

## Chapter 3: RESULTS AND DISCUSSION

### 3.1 Overview of Project Outcomes

During the 4-week industrial training on **MERN Stack Development** at **S.T.E.P GNDEC**, the primary goal was to build a full-stack web application titled “**B2B E-Commerce Website**”. The project was successfully developed using the MERN stack (MongoDB, Express.js, React.js, and Node.js) and aimed to facilitate an online platform where **Retailers**, **Sellers**, and **Admins** could interact securely and efficiently. This section summarizes the overall outcomes and performance of the project during the training.

#### 3.1.1 Project Summary:

The B2B website was designed to provide a scalable, responsive, and role-based user interface where:

- **Retailers** can browse approved products.
- **Sellers** can register and submit their products.
- **Admins** have the authority to approve sellers and products, manage users, and organize products into relevant categories like FMCG, Electronics, etc.

The platform simulates a real-world business-to-business model and focuses on key aspects like secure authentication, image/file uploads, product management, and admin control.

#### 3.1.2 Key Modules/Features Developed:

- **Authentication System** (Login/Register with role distinction: Retailer, Seller, Admin)
- **File Upload System** for documents (e.g., shop proof, identity) using Multer
- **Product Submission Form** with multiple image support (for sellers)
- **Admin Dashboard** to approve sellers/products and categorize items
- **Role-based Access Control** using protected routes in React

- **Frontend Interface** built with **React + Tailwind CSS + Bootstrap**
- **Backend APIs** developed using **Node.js and Express.js**
- **MongoDB Database Design** for storing user, product, and admin data
- **Responsive UI** compatible with both desktop and mobile views

### 3.1.3 Tech Stack Performance:

- The **React frontend** performed efficiently in terms of component reusability and state handling using `useState` and `useEffect`.
- **Tailwind CSS and Bootstrap** enabled fast styling and responsive layout design.
- **Node.js and Express** offered a smooth backend experience with organized REST API routes and middleware.
- **MongoDB** handled data operations effectively with flexible schema design for users and product storage.
- The integration of **Multer** allowed seamless image and file uploads to the backend.
- Deployment and testing were initially done locally and could later be shifted to cloud platforms like **Render, Vercel, or Firebase** (for frontend).

## 3.2 Frontend Results and UI Snapshots

During the industrial training, the frontend of the B2B E-commerce website was built using **Tailwind CSS**. This approach allowed for a modern, responsive, and utility-first design, providing a seamless user experience across different devices.

### Tailwind CSS UI Results

The UI components were designed using a hybrid approach — Tailwind CSS for flexibility and responsiveness, and Bootstrap for components like modals, buttons, and grid layouts. This combination enhanced development speed while maintaining visual consistency.

### 3.2.1 Key Features and Layouts Implemented

- **Navigation Bar**

A responsive navbar was created using Bootstrap and styled with Tailwind classes to support dropdown menus and authentication-based UI rendering.

- **Home Page Design**

The homepage showcases product categories, featured banners, and call-to-action buttons using Tailwind's utility classes for spacing, colors, and typography.

- **Product Cards**

Each product card displays an image, title, price, and quantity-based discount breakdown. Multiple product images can be scrolled within each card using Bootstrap carousels styled with Tailwind.

- **Login and Registration Forms**

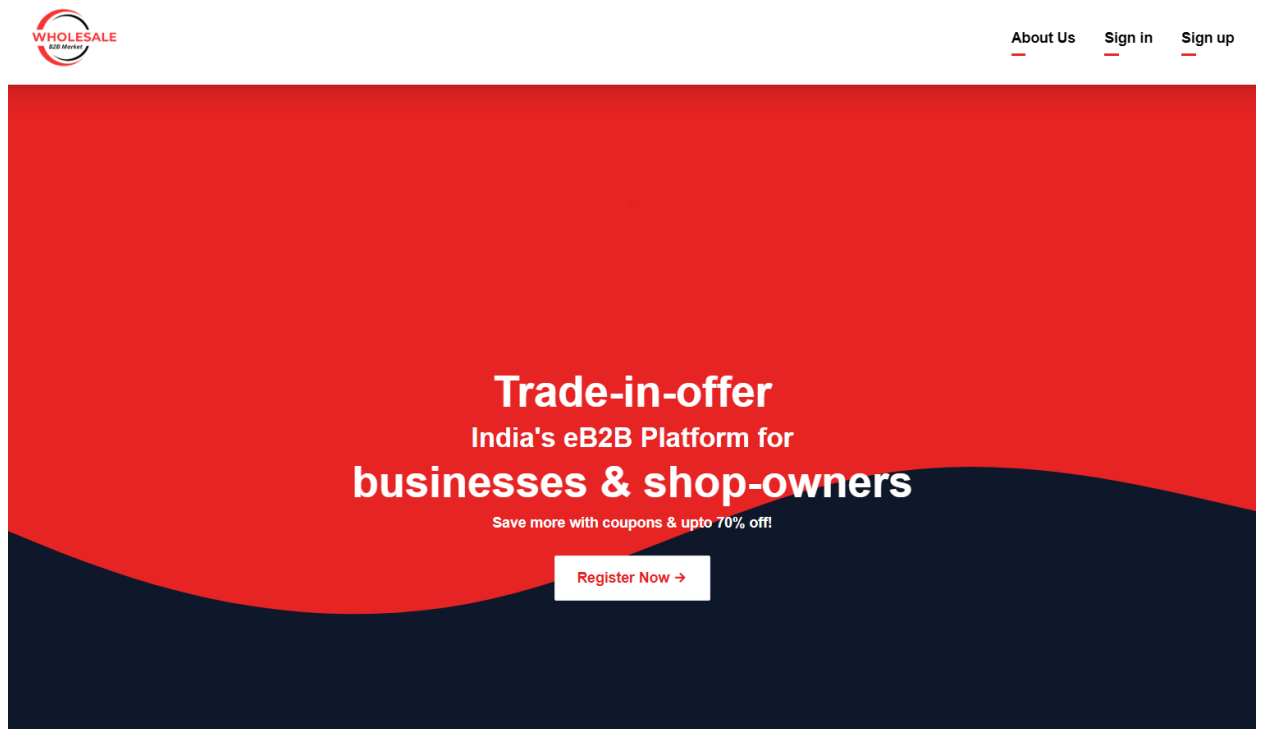
Forms support validation, mobile number-based login, document upload (PDF/images), and user role (Retailer/Seller/Admin) selection.

### 3.2.2 Mobile Responsiveness Results

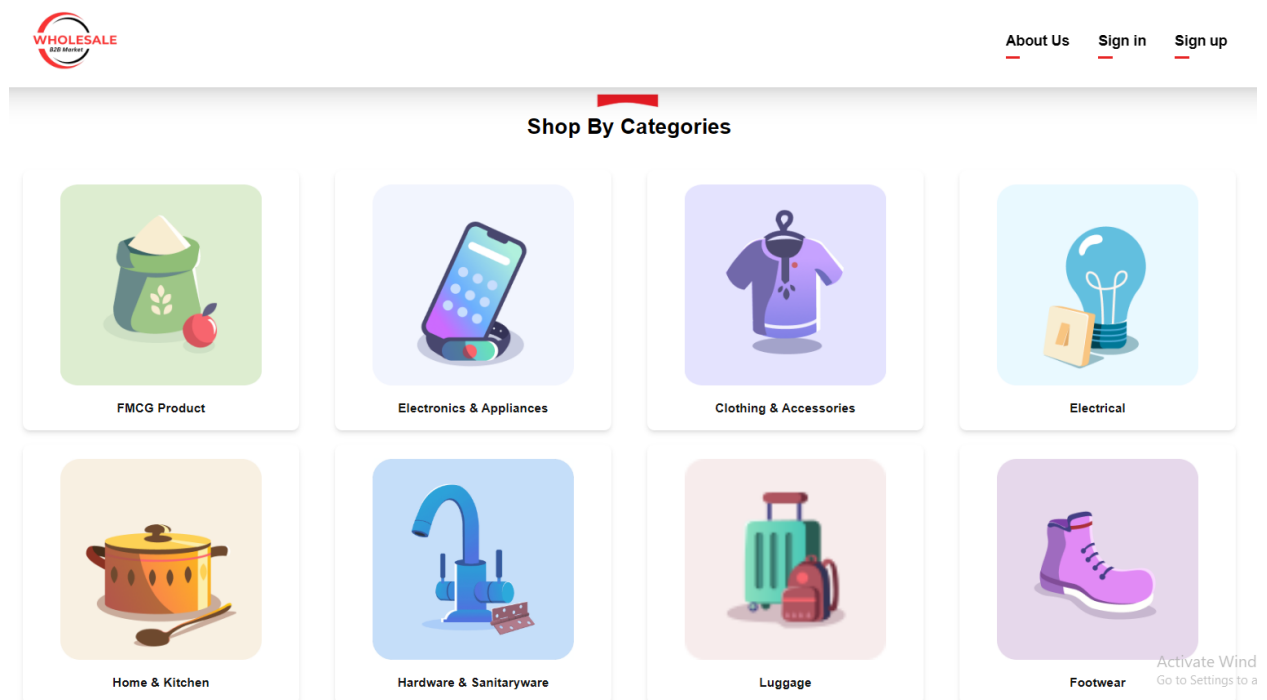
Using Tailwind's mobile-first breakpoints (**sm:**, **md:**, **lg:**), every page is optimized for small screens. Product cards stack vertically, the navbar collapses into a hamburger menu, and font sizes are scaled appropriately.

### 3.2.3 UI screenshots here with labels below each image:

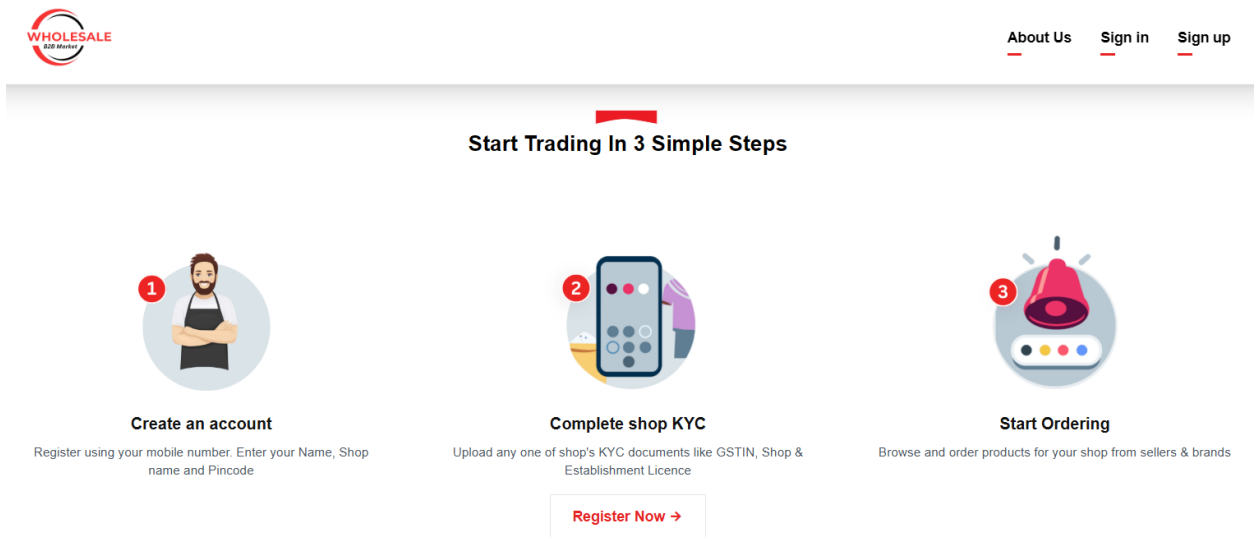
## 1. Figure 3.1: Home Page UI Using Tailwind CSS



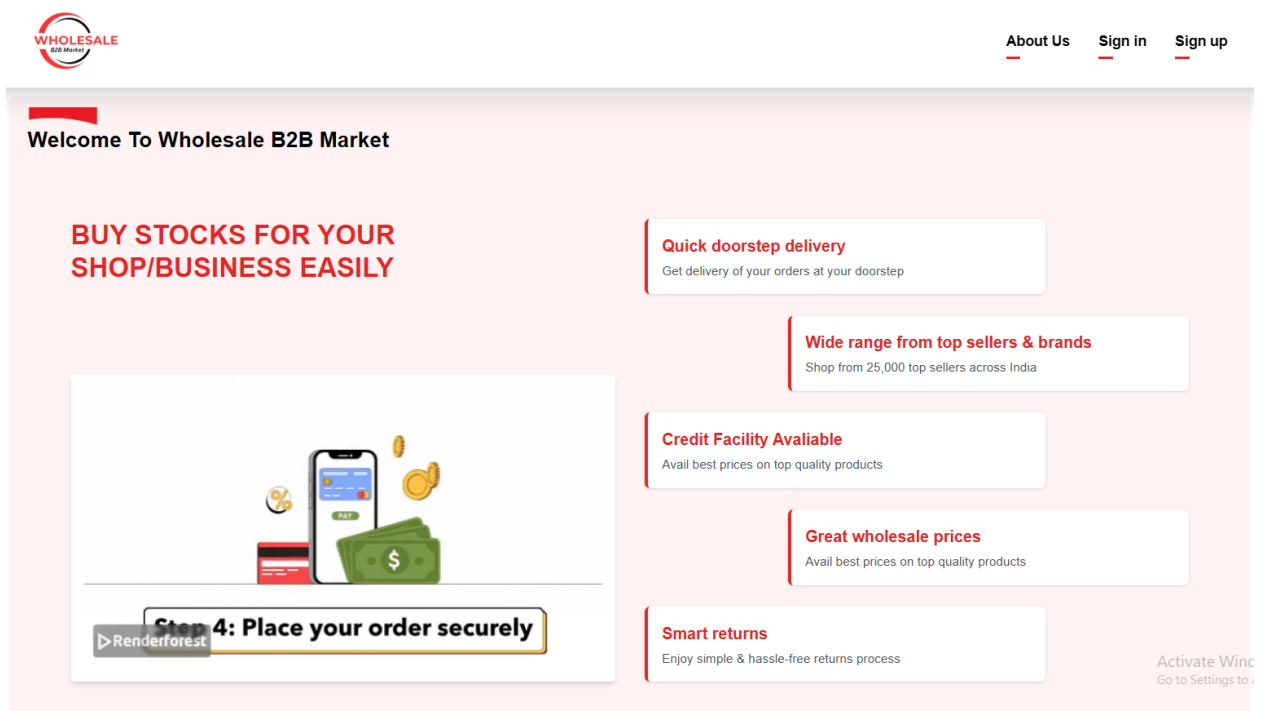
**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)



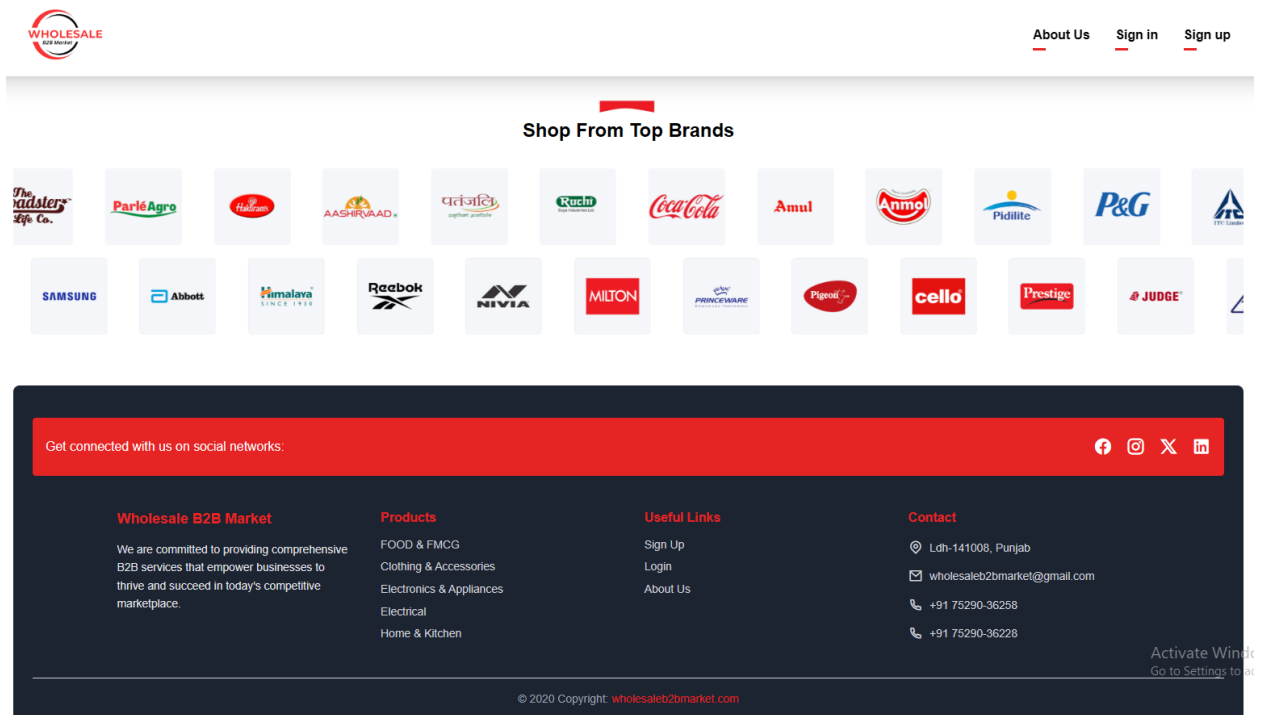
**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)



**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)



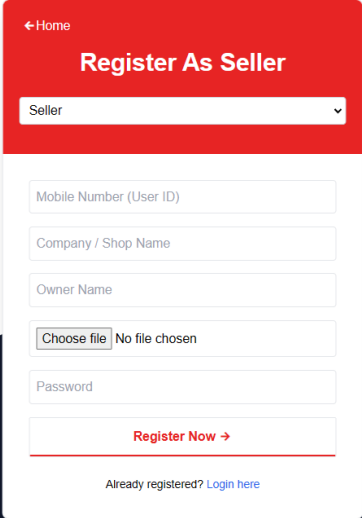
**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)



**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)

## 2. Figure 3.2: Login/Register Forms with File Upload

**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)



← Home

### Register As Seller

Seller ▾

Mobile Number (User ID)

Company / Shop Name

Owner Name

No file chosen

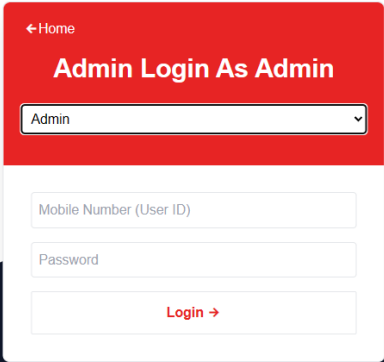
Password

[Register Now →](#)

Already registered? [Login here](#)

Activate Wind

**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)



← Home

### Admin Login As Admin

Admin ▾

Mobile Number (User ID)

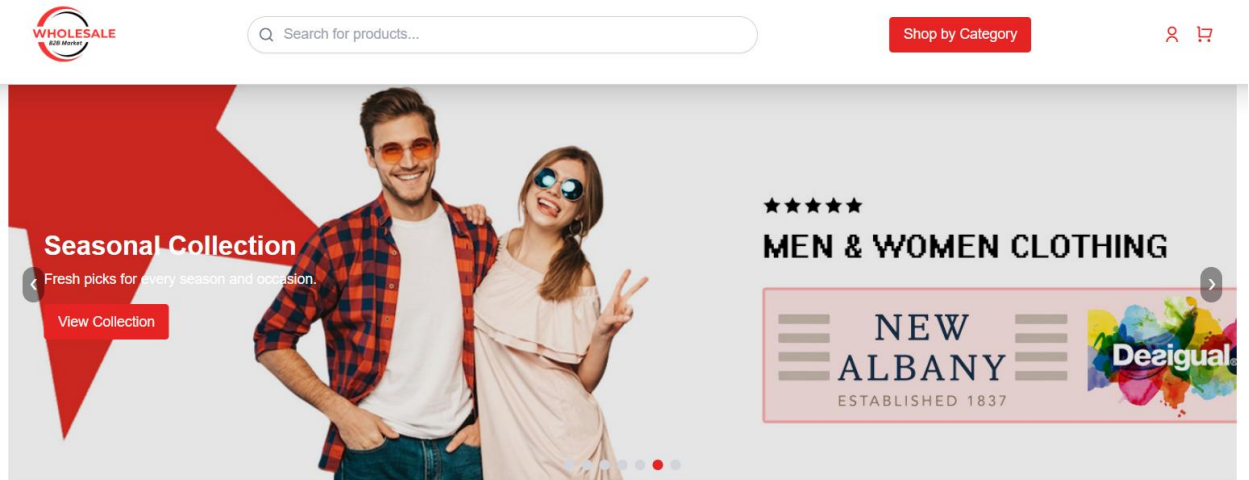
Password

[Login →](#)

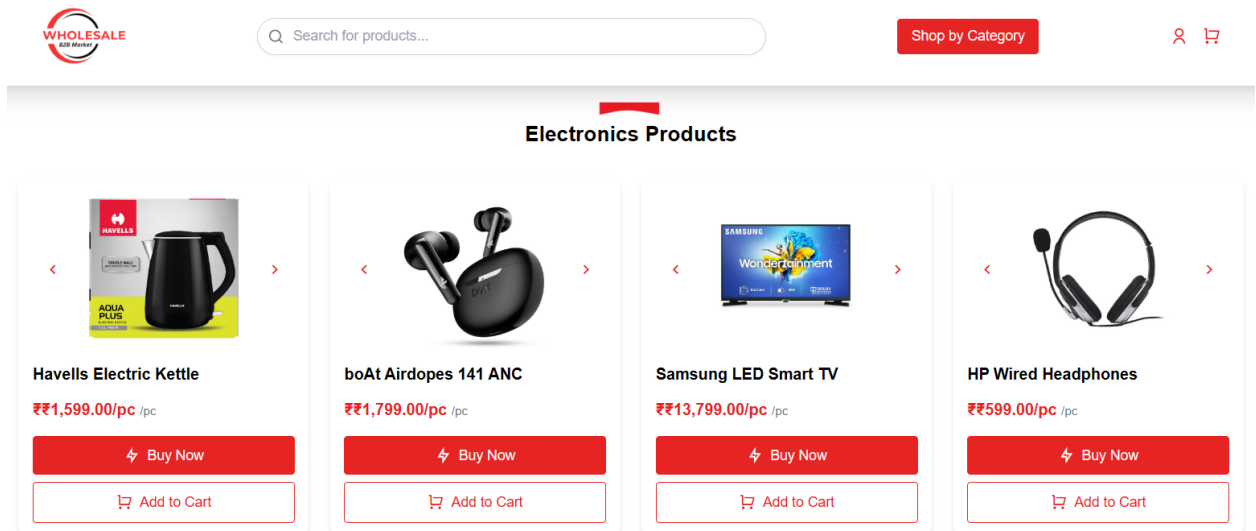
**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)

### 3. Figure 3.3: Product Card with Carousel Images





**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)

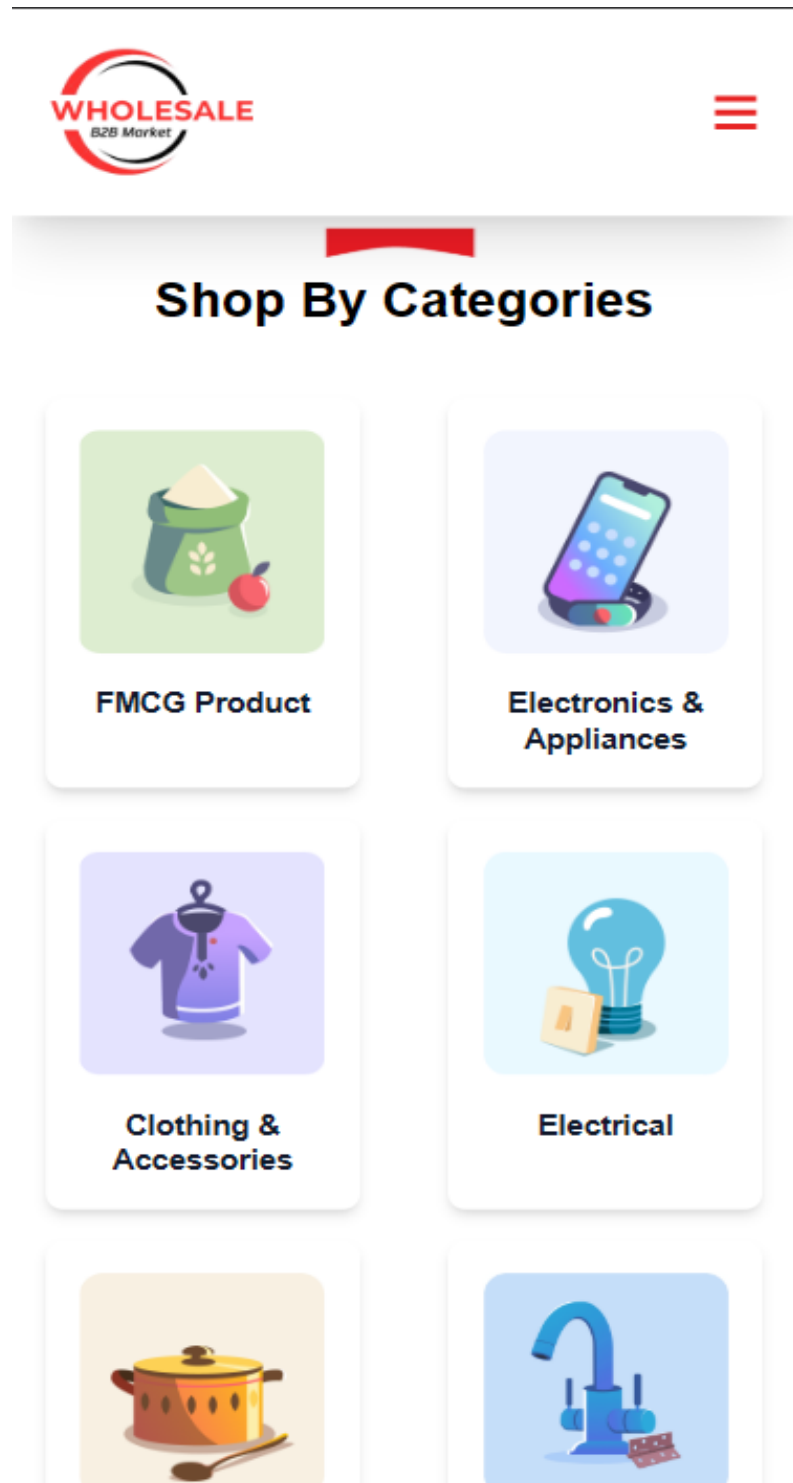


**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)

#### 4. Figure 3.4: Mobile View of Homepage



**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)

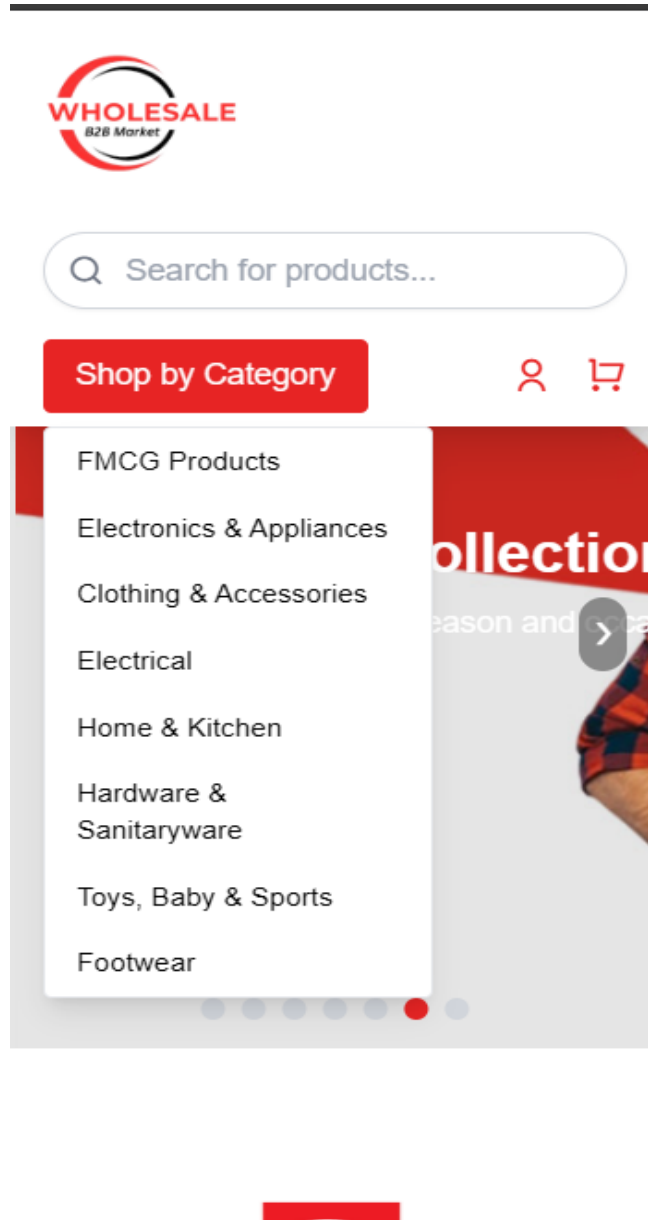


**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)

## 5. Figure 3.5: Navigation Bar on Desktop and Mobile



**Figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)



**figure 6:** Weekly Breakdown of MERN Stack Training (Source: <https://app.eraser.io/>)