**E-COMMERCE WEBSITE**

**(MERN Stack)**

**A PROJECT REPORT**

**for**

**Mini Project (KCA353)**

**Session (2023-24)**

**Submitted by**

**Vikul Dhiman**

**( 2200290140178 )**

**Vipin Sharma**

**( 2200290140179 )**

**Submitted in partial fulfillment of the**

**Requirements for the Degree of**

**MASTER OF COMPUTER APPLICATION**

**Under the Supervision of**

**Dr. Sangeeta Arora**

**(Associate Professor)**



**Submitted to**

**Department Of Computer Applications**

**KIET Group of Institutions, Ghaziabad**

**Uttar Pradesh-201206**

**( MARCH 2024 )**

**CERTIFICATE**

Certified that **Vikul Dhiman 2200290140178, Vipin Sharma 2200290140179** has/ have carried out the project work having “E-COMMERCE WEBSITE” (**Mini Project-KCA353**) for **Master of Computer Applications** from Dr. A.P.J. Abdul Kalam Technical University (AKTU**)** (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

**Date:**

**Vikul Dhiman ( 2200290140178 )**

**Vipin Sharma ( 2200290140179 )**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

**Dr. Sangeeta Arora Dr. Arun Kumar Tripathi**

**Associate Professor Head**

**Department of Computer Applications Department of Computer Applications**

**KIET Group of Institutions, Ghaziabad KIET Group of Institutions, Ghaziabad**

**Vikul Dhiman**

**Vipin Sharma**

**MarketWiZ**

**ABSTRACT**

MarketWiZ is a web-based application intended for online buyers and sellers. The main objective of this application is to make it interactive and its ease of use. It would make viewing, selection etc functions of a products easier. It contains a sophisticated search engine for user’s to search for products specific to their needs. This project make easier to add products which includes updating their name, description, category etc. provides an easy and convenient way to search for products where a user can select the category of products which user want and then the website will show only that category products.

This project report provides an overview of the development and implementation of an E-commerce website aimed at optimizing the online shopping experience. The report covers the key aspects of the project, including the design, functionality, and features incorporated to create a user-friendly and efficient platform.

**ACKNOWLEDGEMENTS**

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Dr. Sangeeta Arora** for her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Arun Kumar Tripathi Professor and Head, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

**Vikul Dhiman**

**Vipin Sharma**

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Certificate | | ii |
|  | Abstract | | iii |
|  | Acknowledgements | | iv |
|  | Table of Contents | | v |
|  | List of Tables | | vi |
|  | List of Figures | | vii |
| 1 | Introduction | | 1-3 |
|  | 1.1 | Overview | 1 |
|  | 1.2 | Description | 1 |
|  | 1.3 | Key features | 2 |
|  | 1.4 | Objectives | 3 |
| 2 | Feasibility Study | | 4-5 |
|  | 2.1 | Technical feasibility | 4 |
|  | 2.2 | Operational feasibility | 4 |
|  | 2.3 | Economic feasibility | 5 |
| 3 | Architecture Design | | 6-9 |
|  | 3.1 DFD | | 6-7 |
|  | 3.2 Use case diagram | | 7-8 |
|  | 3.3 ER diagram | | 9 |
| 4 | Project Process | | 10-17 |
|  | 4.1 Homecard.Js | | 10 |
|  | 4.2 Login.Js & SignUp.Js | | 11-13 |
|  | 4.3 Adding new products | | 13-14 |
|  | 4.4 Card.Js | | 14-15 |
|  | 4.5 Payment.Js | | 16 |
|  | 4.6 Backend | | 17 |
| 5 | Testing | | 18-20 |
|  | 5.1 Unit testing | | 18-20 |
|  | 5.1.1 Backend(Express.Js) | | 18 |
|  | 5.1.2 Adding new products | | 19 |
|  | 5.1.3 Frontend(React.Js) | | 19 |
|  | 5.2 Database testing | | 20 |
| 6 | Conclusion | | 21 |
|  | Bibliography | | 22 |
|  | References | | 23 |
|  |  |  |  |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table No.** | **Name of Table** | **Page** |
| 5.1 | Unit Testing table | 18 |

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fig.ure No.** | **Name of Fig.ure** | **Page No.** | |
| 3.1 | Level 0 DFD | | 6 | |
| 3.2 | Level 1 DFD | | 6 | |
| 3.3 | Level 2 DFD | | 7 | |
| 3.4 | User Use case diagram | | 7 | |
| 3.5 | Admin Use case diagram | | 8 | |
| 3.6 | ER diagram | | 9 | |
| 4.1 | Homecard | | 10 | |
| 4.2 | Homecard.Js | | 11 | |
| 4.3 | Login | | 11 | |
| 4.4 | SignUp | | 12 | |
| 4.5 | Login.Js | | 12 | |
| 4.6 | SignUp.Js | | 12 | |
| 4.7 | Add new product | | 13 | |
| 4.8 | Newproduct.Js | | 14 | |
| 4.10 | Adding products to cart | | 14 | |
| 4.11 | Cart.Js | | 15 | |
| 4.12 | Payment.Js | | 16 | |
| 4.16 | Express.Js | | 17 | |
| 5.1 | Testing of backend | | 18 | |
| 5.2 | Testing of new products add | | 19 | |
| 5.3 | Testing of frontend | | 20 | |
| 5.4 | Testing of database(MongoDB) | | 20 | |

**CHAPTER 1**

**INTRODUCTION**

The MarketWiZ website is a comprehensive and user friendly e-commerce website made with techstack MERN designed to simplify the process of buying and selling the products. The website caters to individuals who want to buy the products at reasonable price and who want to sell the products at best rate for making some profit. The website is equipped with essential features, login to website, signup to the website, Adding products to cart, Checking description, searching of project with categories, making payment etc.

**1.1 Overview**

The MarketWiZ E-Commerce website is a robust and user-friendly platform developed using the MERN (MongoDB, Express.js, React, Node.js) tech stack. This comprehensive online marketplace is meticulously designed to streamline the buying and selling process, catering to individuals who seek both reasonable prices for purchasing products and the opportunity to sell items at the best possible rates to make a profit.

**1.2 Description**

MarketWiZ is a state-of-the-art online shopping platform built with the advanced MERN technology. It's a one-stop solution for people who want to buy and sell products effortlessly and securely. Easy Sign-In, Log in or sign up. We prioritize the security of your personal information. Product Management: Sellers can easily add, update, and manage their products. Buyers get a wide range of choices. User-Friendly Design The website is designed to be simple and attractive, making it easy for everyone to use. Convenient Shopping Cart Add items to your cart with a click, making the shopping process smooth and enjoyable. Detailed Product Info: Know what you're buying with detailed product descriptions, helping you make informed decisions. Efficient Search and Categories: Find what you need quickly with powerful search features and well-organized categories. Secure Payments: Make transactions confidently with our integrated secure Payment gateway, keeping your information safe simplified.

Transactions We aim to make buying and selling as easy as possible, creating a hassle-free experience for users. Engaging Experience Enjoy exploring the platform with a user-friendly design that encourages effortless transactions. Profitable Selling Sellers can maximize profits by offering their products at competitive rates, benefiting both buyers and sellers. Scalable and Reliable: We've built a platform that can grow with demand, ensuring a smooth experience even during busy times.

**1.3 Key Features**

**User-Friendly Interface**: The website boasts an intuitive and user-friendly interface, ensuring a seamless and enjoyable experience for both buyers and sellers.

**Authentication and Authorization:** Secure login and signup functionalities ensure a personalized experience for users, offering a secure environment for their transactions.

**Product Management**: Sellers can effortlessly add and manage their products, providing a diverse range of offerings for potential buyers.

**Shopping Cart Functionality**: Users can easily add products to their cart, creating a convenient and efficient shopping experience.

**Product Descriptions**: Detailed product descriptions empower buyers with essential information, aiding them in making informed purchase decisions.

**Search and Categorization**: The website facilitates a powerful search functionality, allowing users to find products swiftly through categories, enhancing the overall browsing experience.

**Secure Payment Gateway**: The incorporation of a secure payment gateway ensures safe and reliable transactions, instilling confidence in users while making payments.

**1.4 Objectives**

**Simplify Buying and Selling:** The primary goal is to simplify the process of buying and selling, providing a convenient platform for users to engage in seamless transactions.

**User Engagement:** Foster user engagement through an appealing and intuitive design, encouraging users to explore the website and make transactions effortlessly.

**Profitable Selling Opportunities:** Create a marketplace that not only benefits buyers but also offers sellers the opportunity to maximize profits by providing a platform for selling products at competitive rates.

**Scalability and Performance:** Develop a scalable architecture that can accommodate growth while ensuring optimal performance, even during peak usage periods**.** online at any time. Unlike traditional stores, this site doesn't have set times for when it opens and when it closes.

**Personalizing Shopping Experiences:** Consumers could have a customized shopping experience. Consumers can browse for a large range of products based on their preferences and needs without restriction. On e-commerce websites, customers are presented products depending on their interests and geography.

**Elevating Buying Process:** E-commerce has accelerated the entire purchasing procedure for consumers. They can purchase things from the comfort of their own homes, without the need to visit actual stores. It saves enormous amounts of time and expedites transactions.

**Retargeting Customers:** Online purchasing has simplified the process of retargeting customers for businesses. While clients are shopping online, the ecommerce enterprise collects a large amount of information about them. Periodically, customers can be contacted by sending them personalized emails, messages, promotions, and discounts

**Access to Customer Data:** E-commerce platforms gather valuable data about customer behavior, preferences, and demographics. This data can be analyzed to make informed business decisions, refine marketing strategies, and enhance the overall customer experience..

**Flexibility and Scalability:** Online businesses can easily adapt to changing market conditions and scale their operations to accommodate growth. This flexibility is crucial for businesses experiencing fluctuations in demand or those looking to expand.

**CHAPTER 2**

**FEASIBILITY STUDY**

The MERN E-Commerce project aims to assess the viability and practicality of developing an online marketplace using MongoDB, Express.js, React, and Node.js. This feasibility study evaluates technical, operational, economic, and aspects to determine the project's viability.

**2.1 Technical Feasibility**

Technology Stack: MERN stack is widely used and has proven its effectiveness in building scalable and responsive web applications. Availability of skilled developers and a large community for support. Scalability: MongoDB's scalability allows the system to handle increased data and user loads. Node.js supports asynchronous, non-blocking I/O, making it suitable for handling a large number of concurrent connections. Third-Party Integrations: Availability of APIs and SDKs for common e-commerce functionalities such as payment gateways, shipping providers, and inventory management. Development Tools: A rich set of development tools and libraries for React.js, Node.js, and MongoDB simplifies the development process. Security Measures: Implementing secure coding practices, encryption, and authentication mechanisms to ensure the security of user data and transactions.

**2.2 Operational Feasibility**

User Experience: Ensuring the website is user-friendly and provides a positive shopping experience. Regularly updating and optimizing the user interface based on user feedback. Scalability and Performance: Ensuring the system can handle increased traffic during peak times. Monitoring and optimizing performance to provide a responsive experience. Training and Support: Providing training for administrators and support staff to manage and maintain the e-commerce platform. Ensuring there is a support system in place for users encountering issues. Compliance and Regulations: Adhering to legal and regulatory requirements related to e-commerce, data protection, and online transactions.

Sustainability: Evaluating the long-term viability of the technology stack and ensuring it can adapt to future changes in the e-commerce landscape.

**2.3 Economic Feasibility**

Development Costs: Initial development costs may include hiring experienced MERN stack developers. Licensing costs for any third-party tools or services used in the development process. Infrastructure Costs: Cloud hosting services may incur ongoing operational costs based on usage. Costs for securing the infrastructure and implementing backup and recovery mechanisms. Maintenance Costs: Regular maintenance and updates to ensure compatibility with evolving technologies. Costs associated with bug fixes, security patches, and feature enhancements. Return on Investment: Estimation of the revenue generation potential and how it compares to the overall investment in development and maintenance..

**CHAPTER 3**

**ARCHITECTURE DESIGN**

**3.1 Data Flow Diagram**

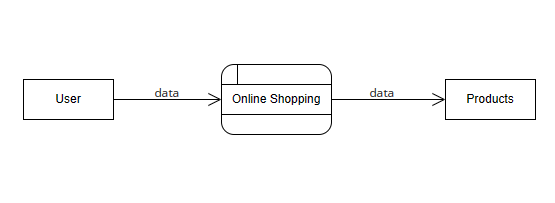
****

Fig. 3.1 Level 0 DFD

User initiates the online shopping process by providing input and receiving outputs. Online shopping module will manages the overall online shopping functionality. Products represents the range of items available for purchase in the online store.

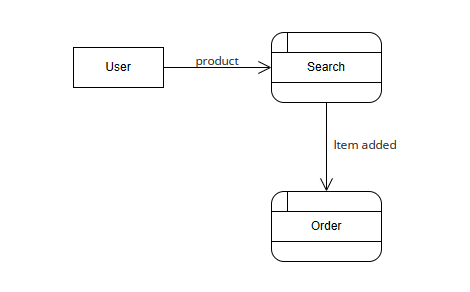
****

Fig. 3.2 Level 1 DFD

Search Receives user search queries from the User. Utilizes the search algorithm to find matching products. Sends the search results back to the User. Order module will place order.

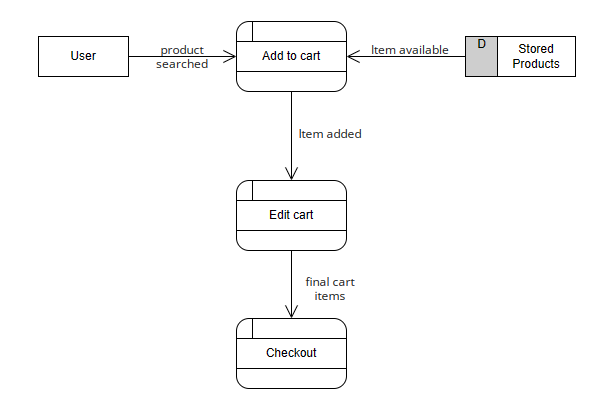
****

Fig. 3.3 Level 2 DFD

Add to cart Receives selected products from the User. Updates the user's shopping cart with the selected items. Sends an acknowledgment to the User. Edit cart will allows the user to modify the contents of their shopping cart. Checkout process manages the completion of the user's order, including payment and shipping details.

**3.2 Use Case diagram**

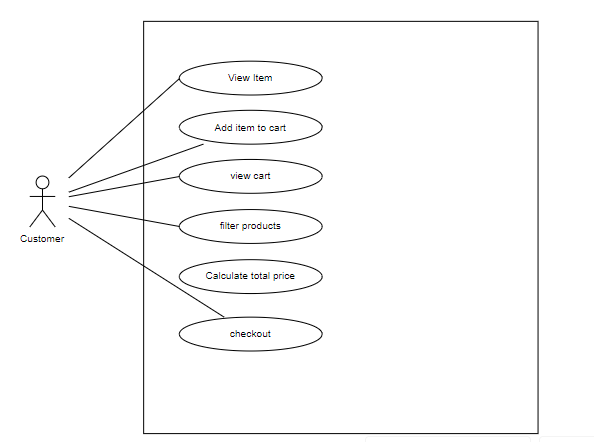
****

Fig. 3.4 User Use case diagram

The "View Items" use case represents the user's ability to browse and explore the available products in the system. The "Add Item to Cart" use case reflects the action of selecting a product and placing it into the virtual shopping cart. This process enables users to accumulate desired items for future consideration.

The "View Cart" use case allows users to review the contents of their shopping cart, providing an overview of selected items and quantities. This feature enhances the user's shopping experience by facilitating easy management of their selected products. The "Filter Products" use case permits users to refine their product search based on specific criteria such as category, price range, or other relevant filters. The "Calculate Total Prices" use case involves the system determining the cumulative cost of items in the shopping cart. This step is crucial for users to be aware of the total amount they will be charged before proceeding to checkout. Finally, the "Checkout" use case represents the user's intent to complete the purchase. This involves providing necessary details for shipping and payment, ultimately concluding the transaction.

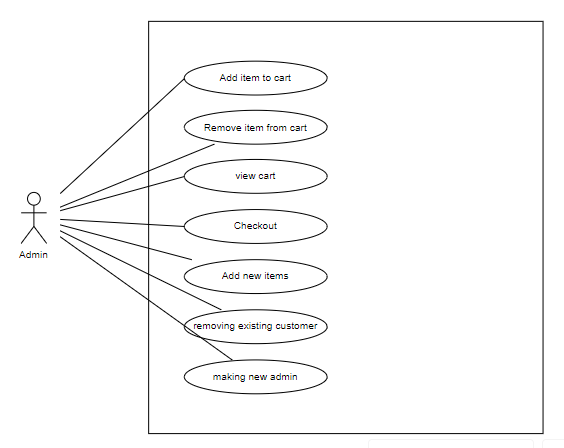


Fig. 3.5 Admin Use case diagram

The admin use case diagram has the "Remove Existing Customers" use case represents the admin's ability to manage customer accounts by removing or deactivating existing customer profiles. The "Make New Customers" use case reflects the admin's authority to create new customer accounts within the system. The "Add Items" use case signifies the admin's role in managing the product catalog.

**3.3 ER Diagram**

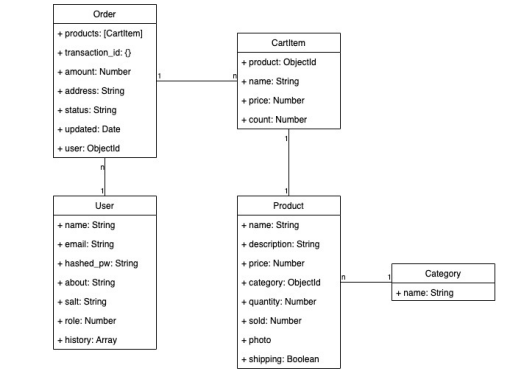
****

Fig. 3.6 ER diagram

**CHAPTER 4**

**PROJECT PROCESS**

ShopVerseisanE-commerceWebApplicationusingtheMERNstackthat

canhelpcompaniesbringtheirproductstothecustomers.Mainfunction:

•Signupandlogin:RequiresUserstoregisterusingtheirphonenumber

oremail

•Shoppingcart:thisfeaturehelpsusersbuyandcheckgoodsdirectlyon

theapplication

•Search:Userscansearchdirectlybytypinginthesearchboxforthe

producttheywanttosee.

•Buyandpay:Customerswhobuythroughtheappcanpaythroughmany

differentpaymentgateways.

ShopVerseisanE-commerceWebApplicationusingtheMERNstackthat

canhelpcompaniesbringtheirproductstothecustomers.Mainfunction:

•Signupandlogin:RequiresUserstoregisterusingtheirphonenumber

oremail

•Shoppingcart:thisfeaturehelpsusersbuyandcheckgoodsdirectlyon

theapplication

•Search:Userscansearchdirectlybytypinginthesearchboxfor99the

producttheywanttosee.

•Buyandpay:Customerswhobuythroughtheappcanpaythroughmany

differentpaymentgateways.

**4.1 HomeCard.Js**

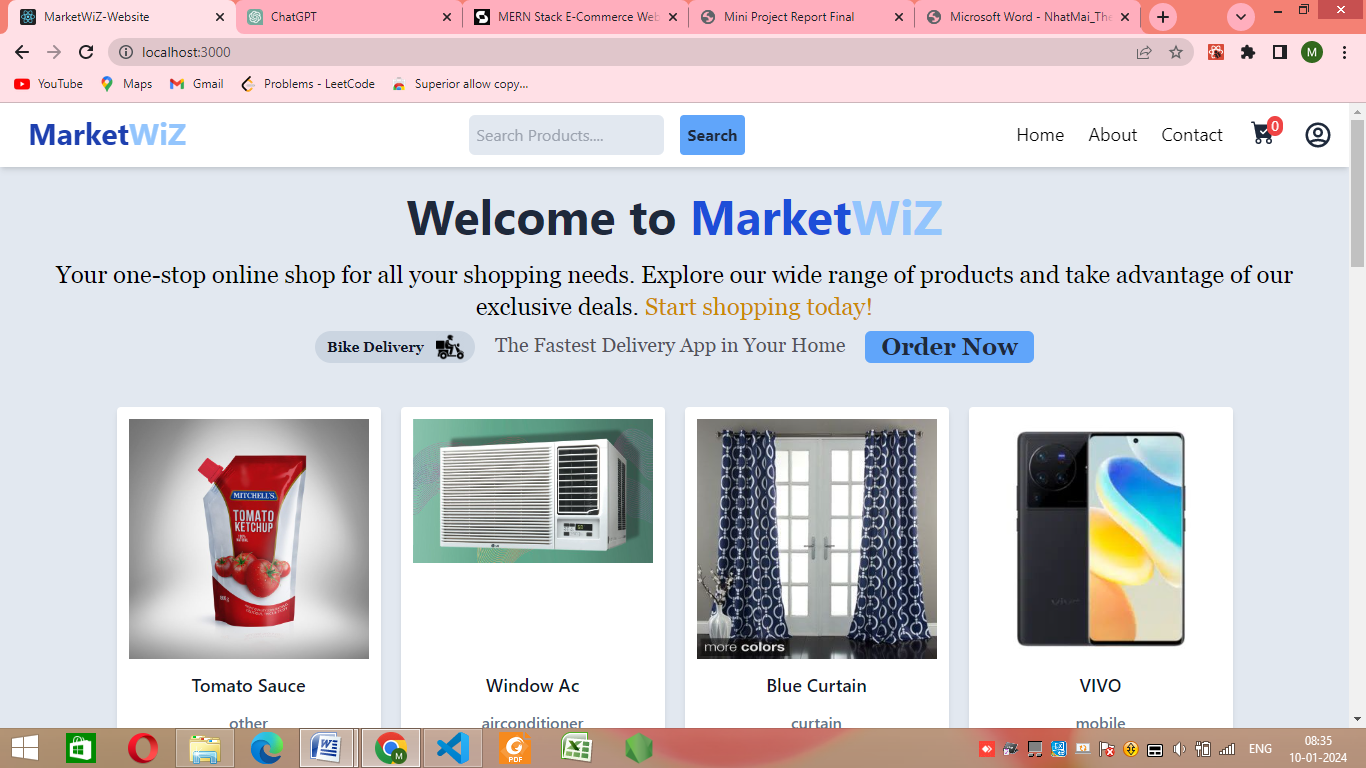
****

Fig. 4.1 Homecard

On the above Figure we can see the home page of the app. It includes an online store like the function bar, the search bar and a list of products. Customers who visit the store can search for the products they want to buy and they can also chose to select from

Categories to buy the best products.

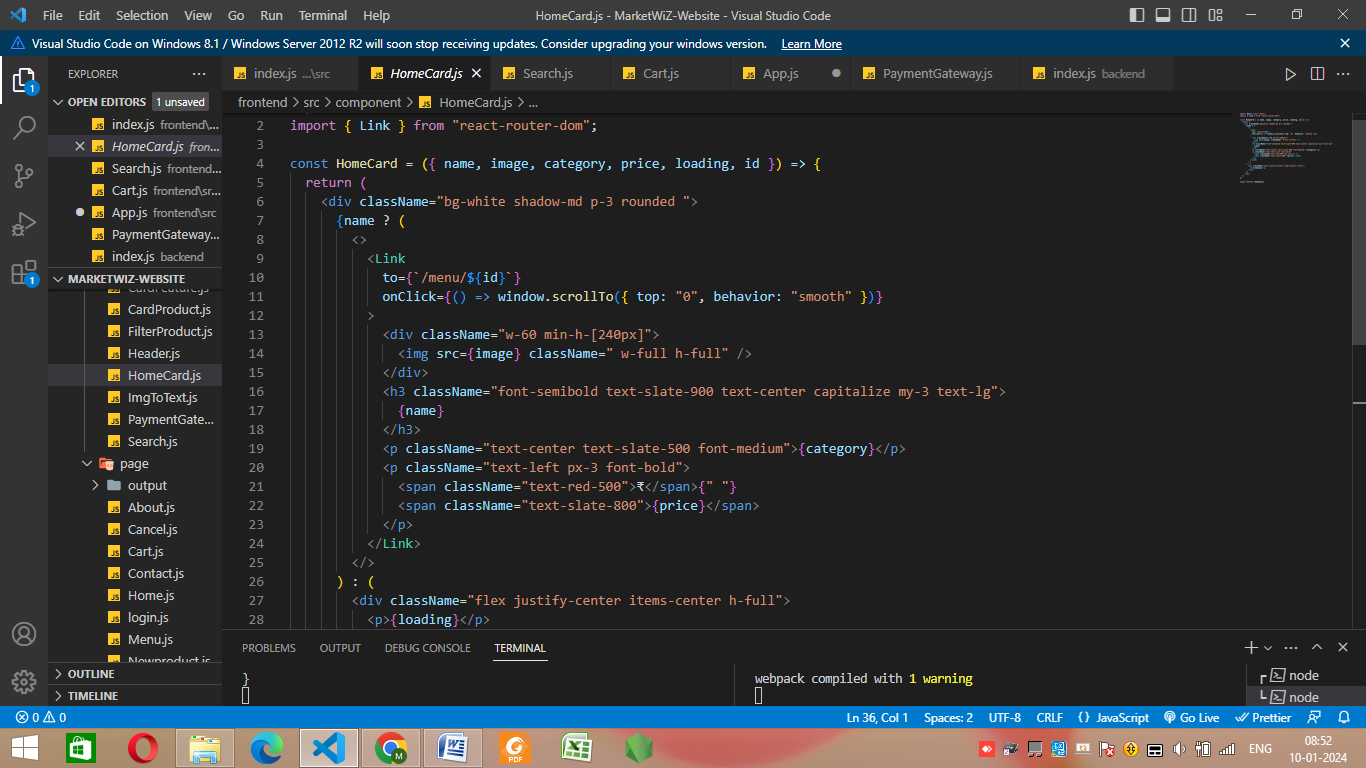


Fig. 4.2 Homecard.Js

**4.2 Login.Js & SignUp.Js System**

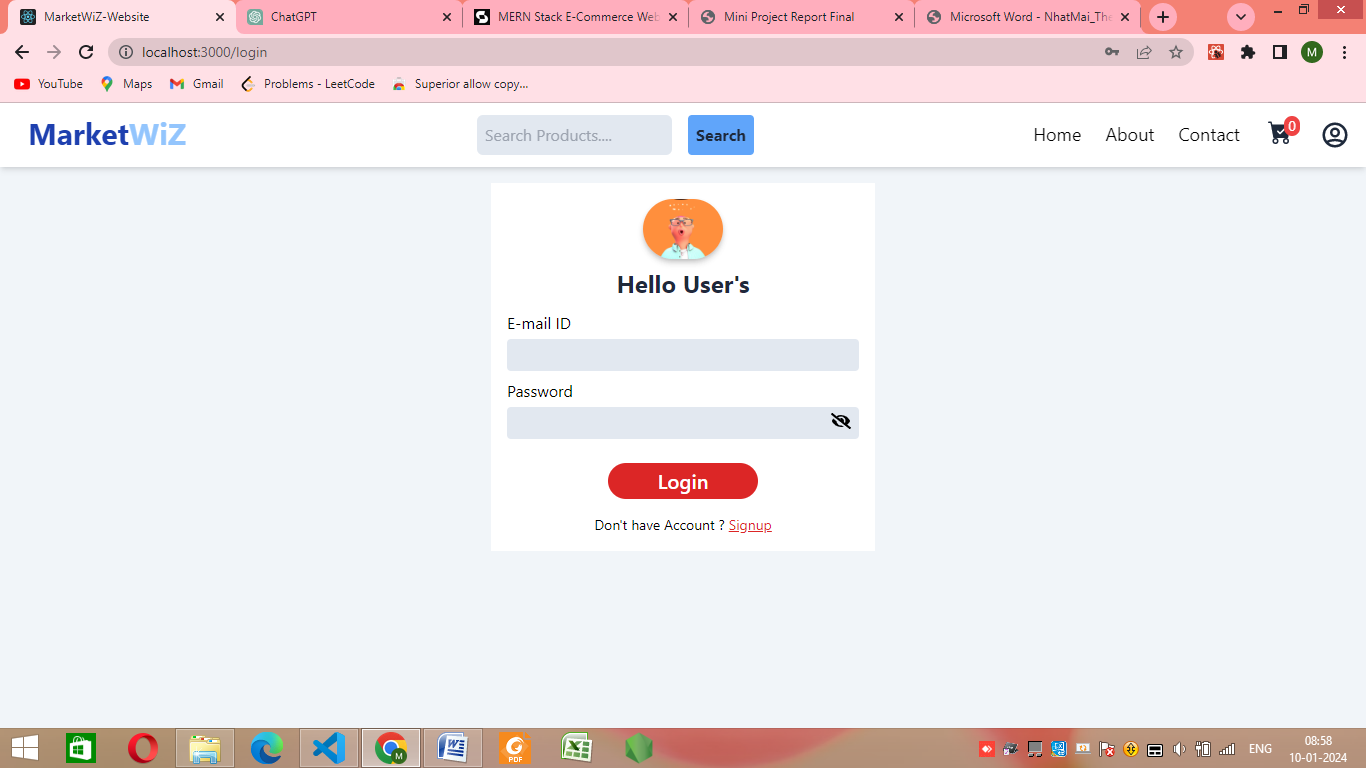
****

Fig. 4.3 Login

Above Figure is showing login page of the MarketWiZ website, the user can provide the email and password. If the email and password were found correct then user

will get logged in otherwise message “incorrect credentials”.

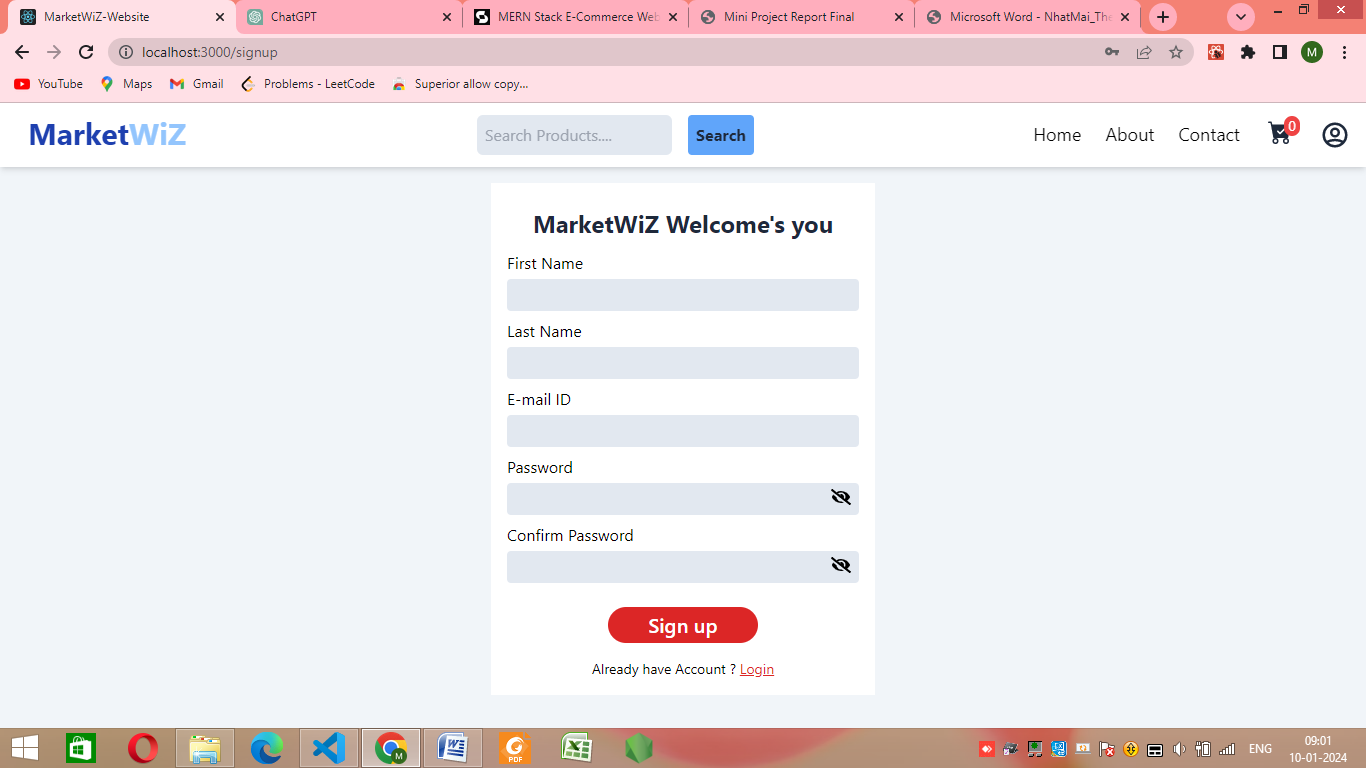
****

Fig. 4.4 SignUp

Above Figure is showing Sign Up page of the MarketWiZ website, the user has to provide the first name, last name, email and password. The record will be first put on the database after that the user can login using that credentials.

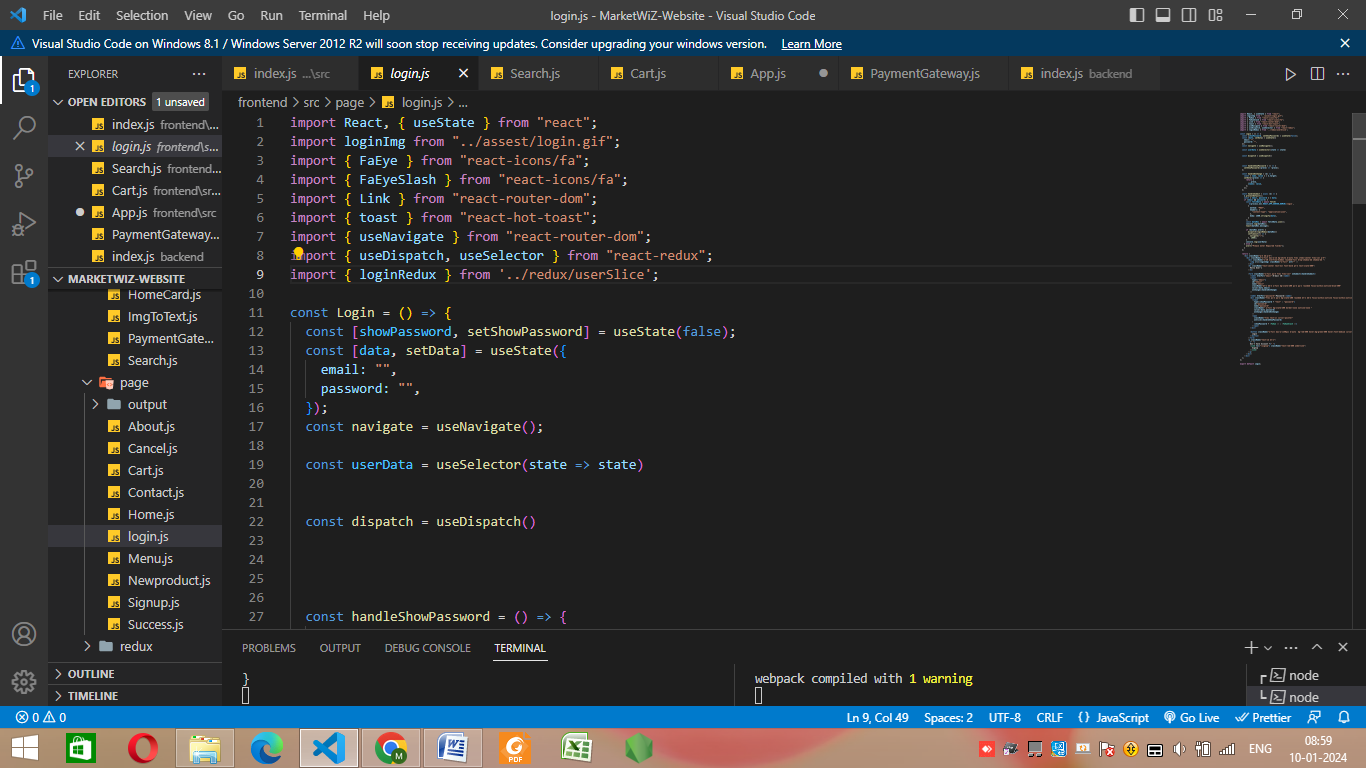
****

Fig. 4.5 Login.Js

logInImg imported from “../asset/login.gif” and useNavigate() is imported from “react-router-dom”. Login() function is created in which useNavigate(), useSelector() and dispatch() methods are used.

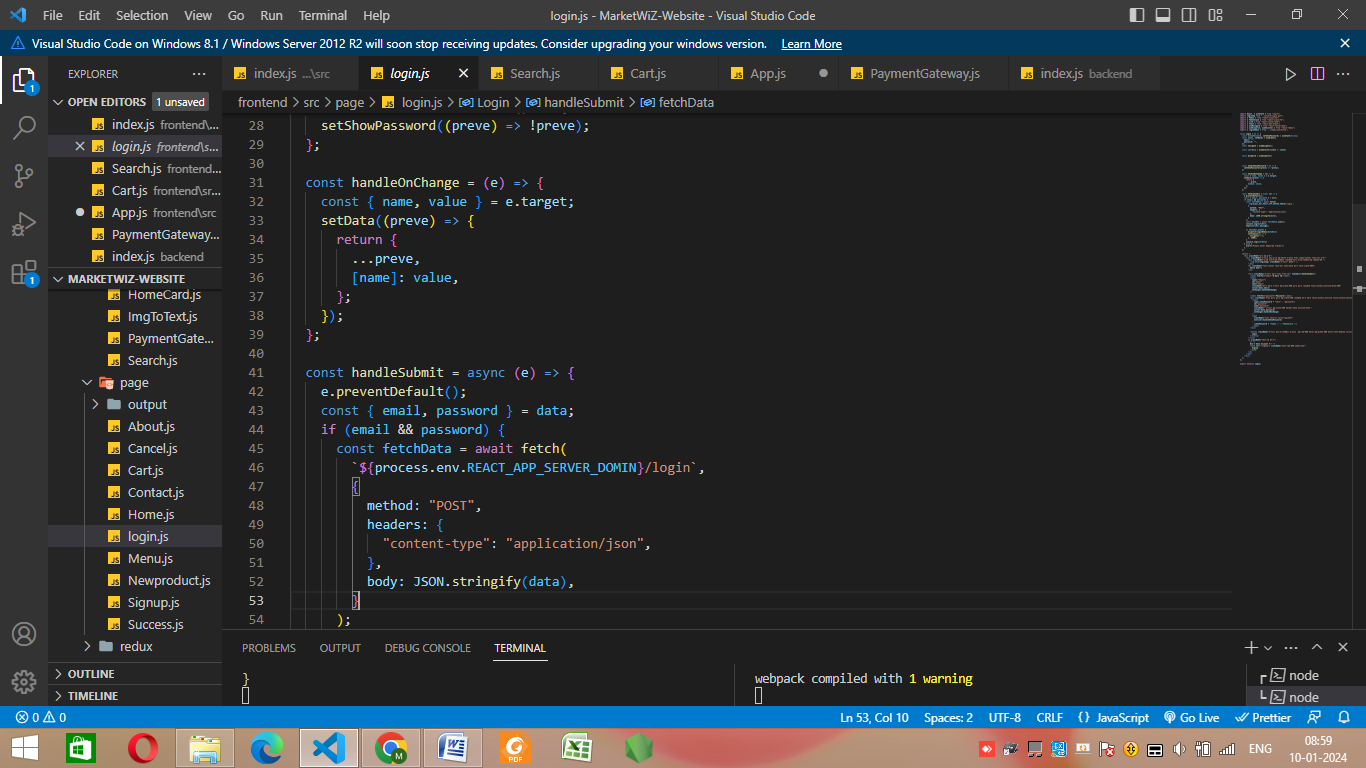
****

Fig. 4.6 Login.Js

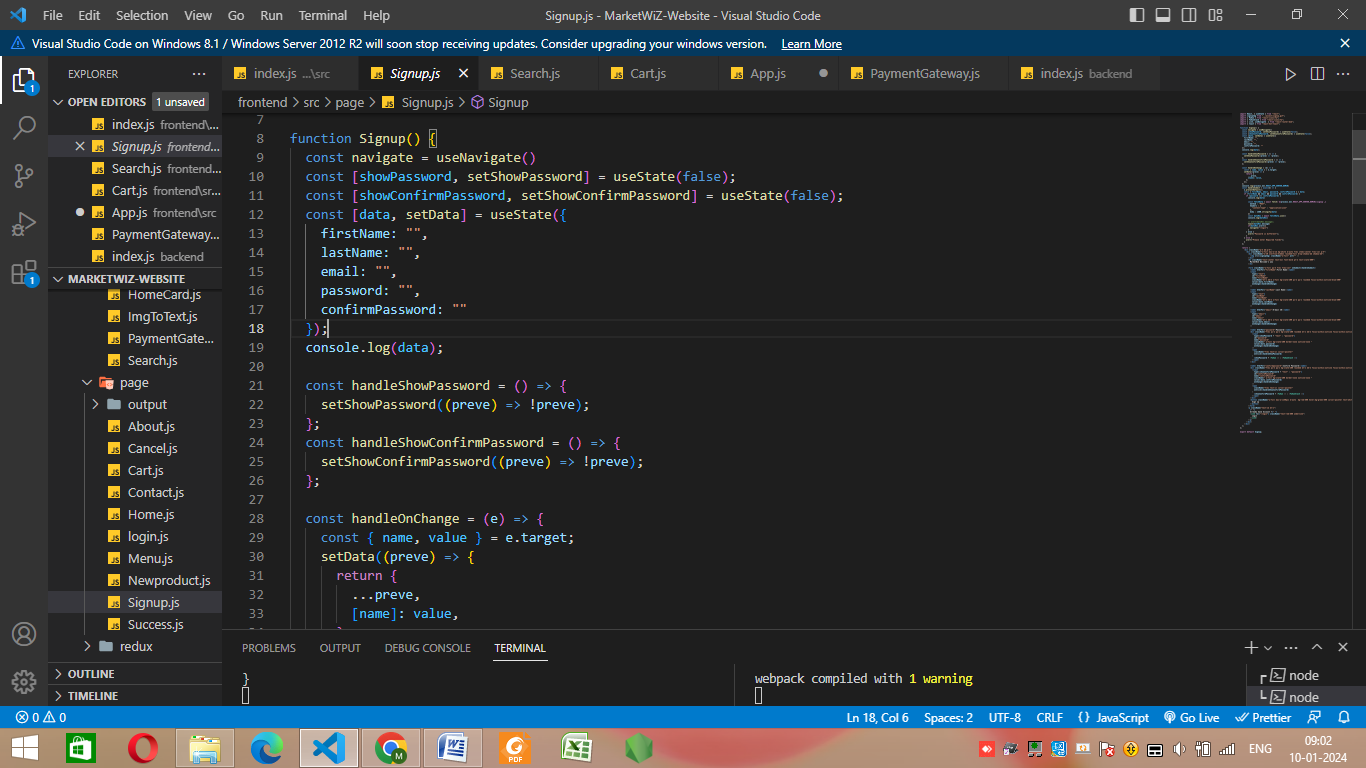
****

Fig. 4.6 SignUp.Js

In Signup() function handleShowPassword() and handleShowConfirmPassword() is used. In handleShowPassword() setShowPassword is used. handleOnChange() method is implemented using setData and preve. In handleShowConfirmPassword() setShowConfirmPassword() is used.

**4.3 Adding New Product**

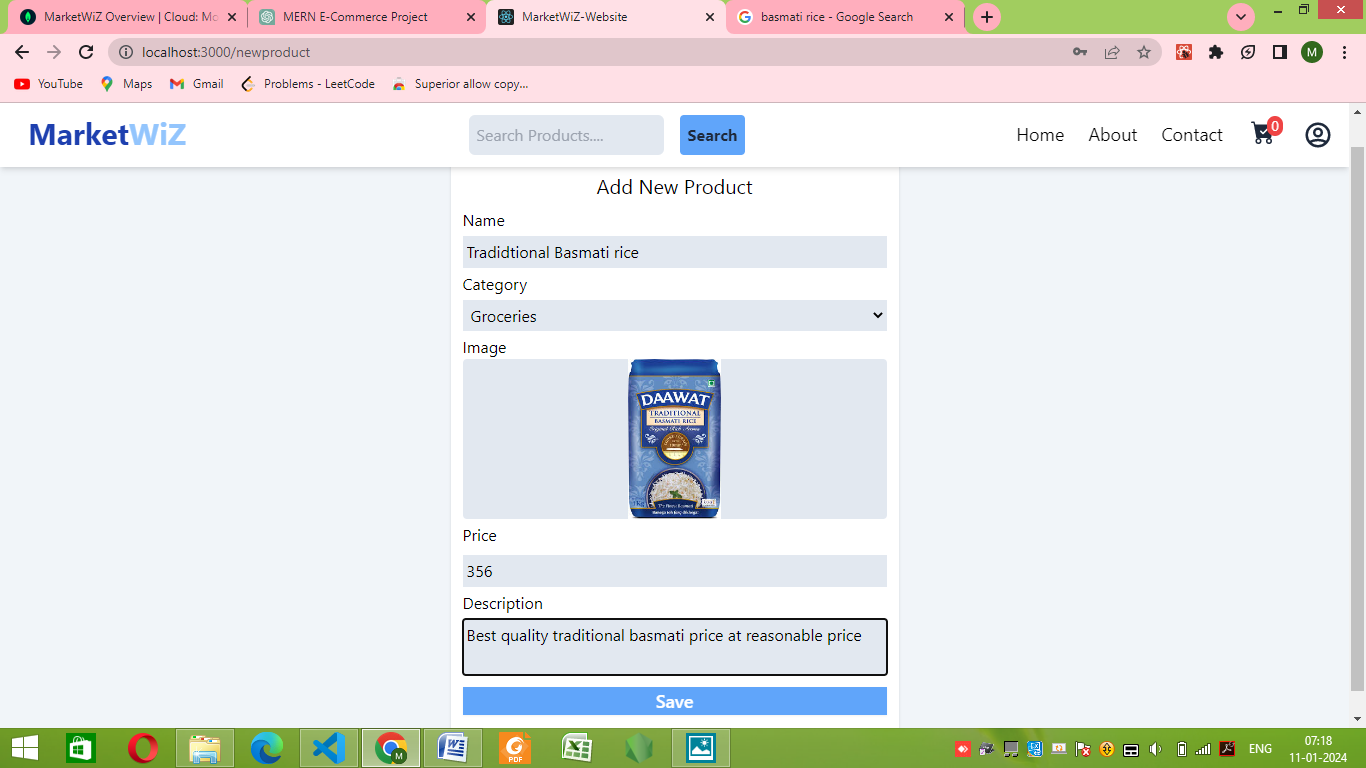
****

Fig. 4.7 Add new product

Above Figure is showing how to add new products in the website this part will be done by the admin as it has the authority to do it, the name of the product, categories, price and description need to be given. The record will be first put on the database after that the website will have this product and can be seen in categories section too.

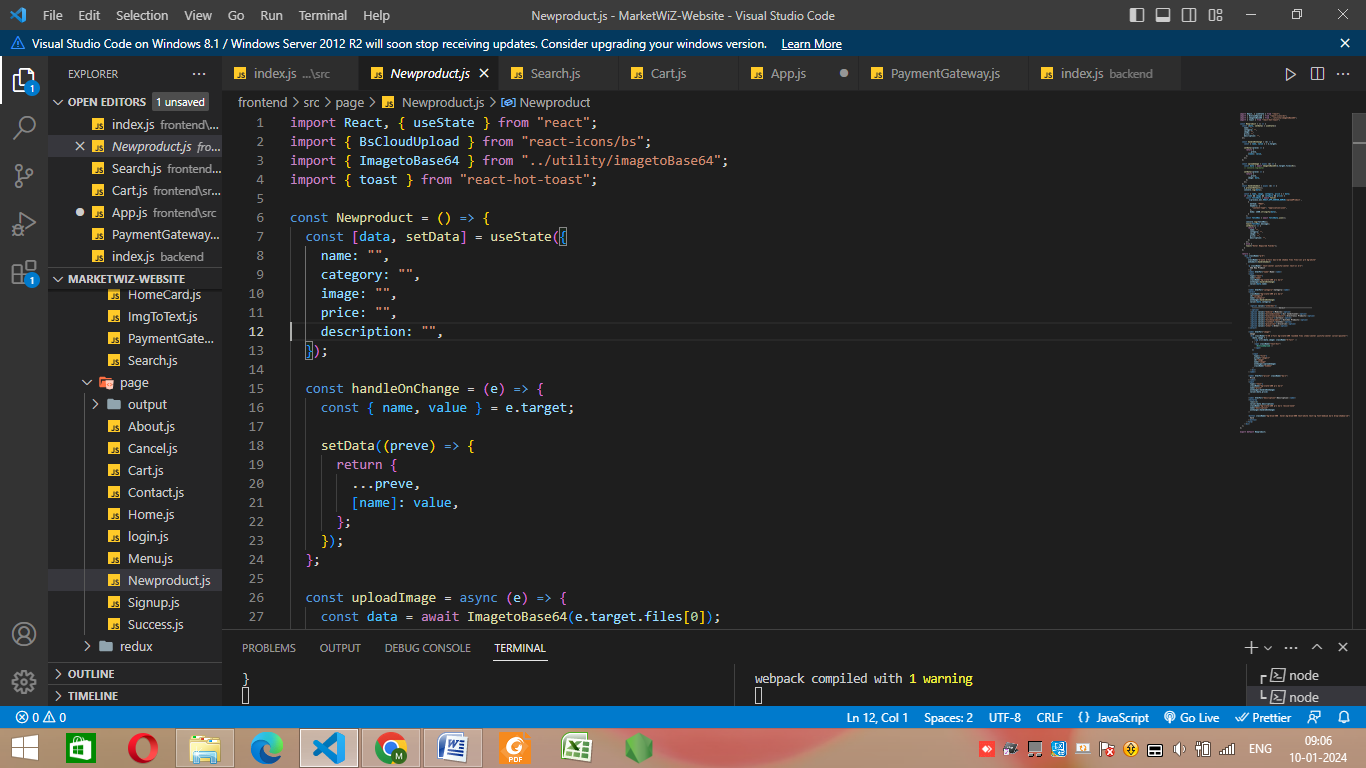
****

Fig. 4.8 Newproduct.Js

Importing BsCloudUpload from “react-icons/bs” for the products image and ImageToBase64 from “../utility/imagetoBase64”. Newproduct() function is used for handing the “adding new product” in the website. In this function useState() is used.

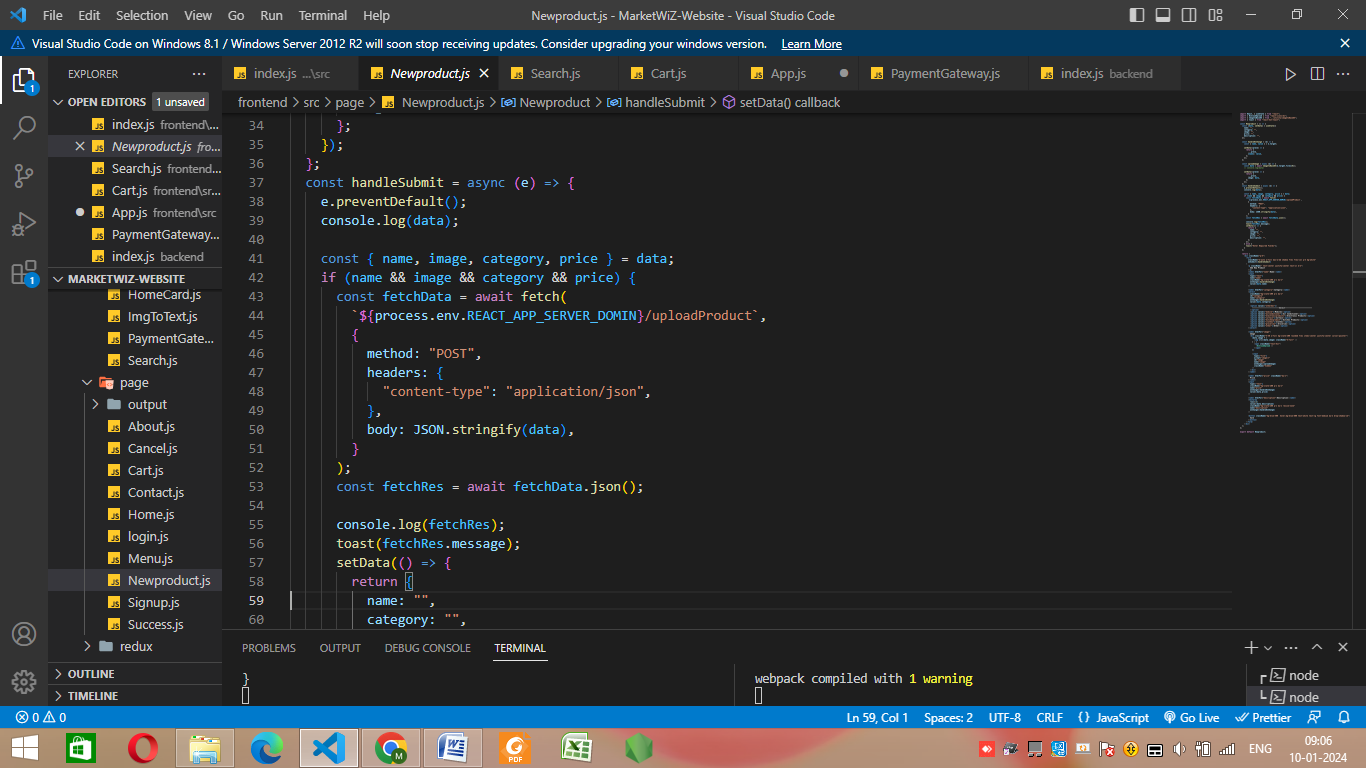
****

Fig. 4.9 Newproduct.Js

Above Figures are showing the coding part of New product add page and how they are implemented.

**4.4 Cart.Js**

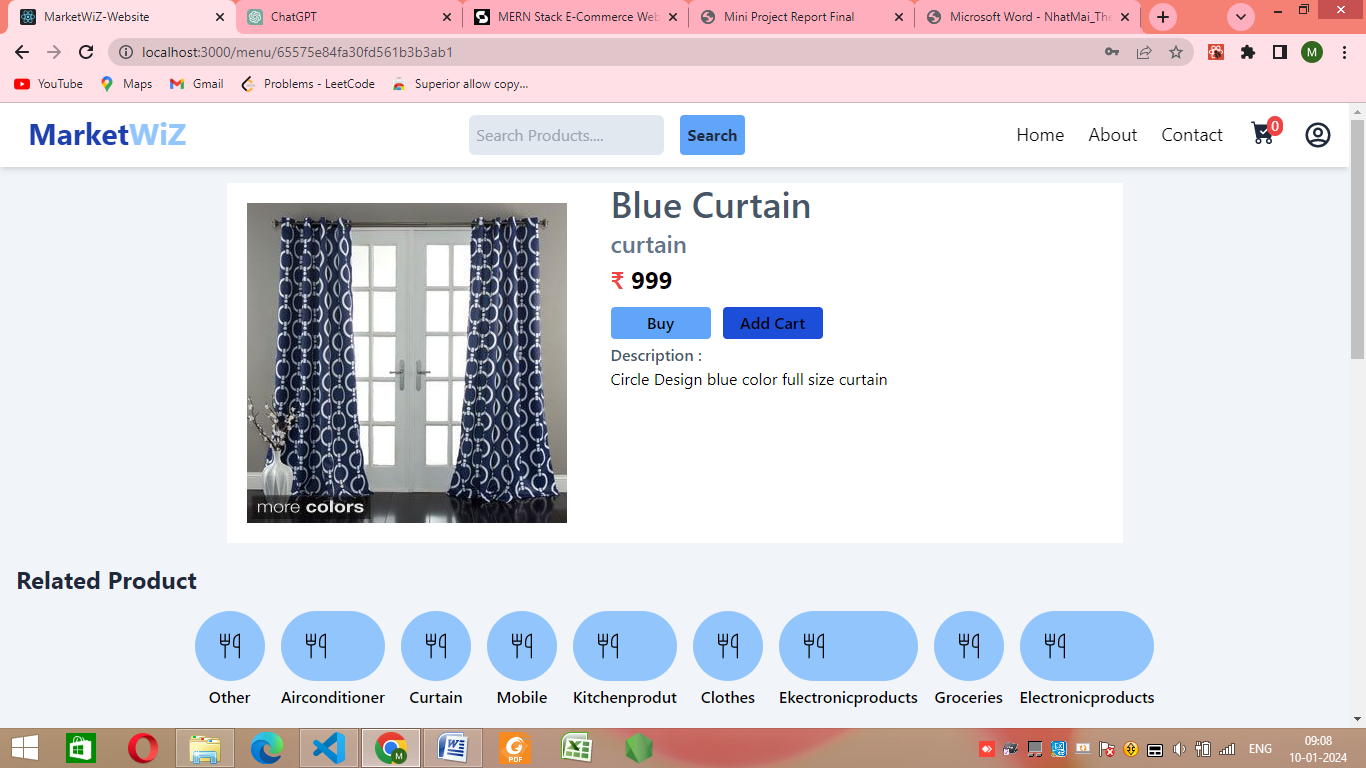
****

Fig. 4.10 Adding products to cart

Products can be added by just clicking on the add to cart button or buy now button. User can check the price and description the product.

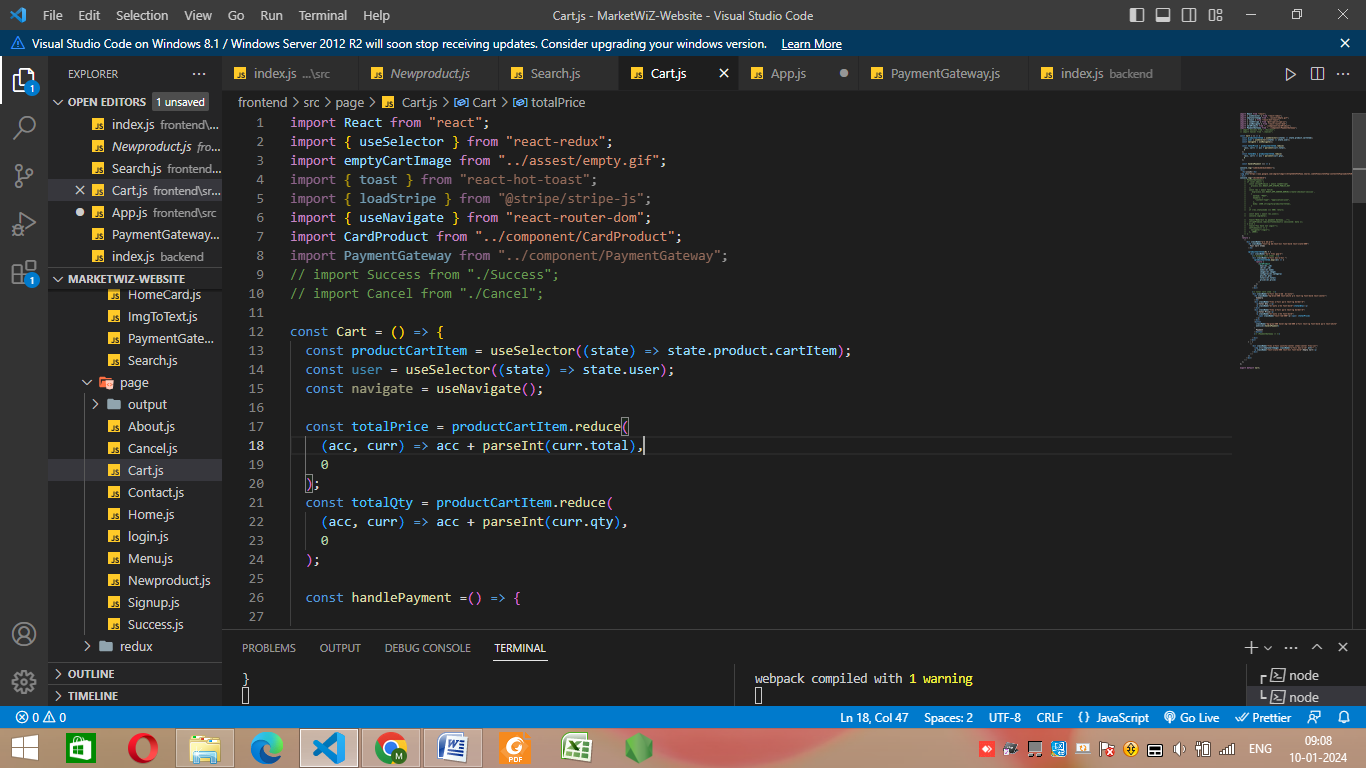
****

Fig. 4.11 Cart.Js

productCartItem() method is implemented using useSelector(). In this method state.product.catItem is returned. In Cart() method totalPrice() and totalQuantity() methods are implemented using productCartItem.reduce() method. productCartItem returns acc+parseInt(curr.total) for price calculation.

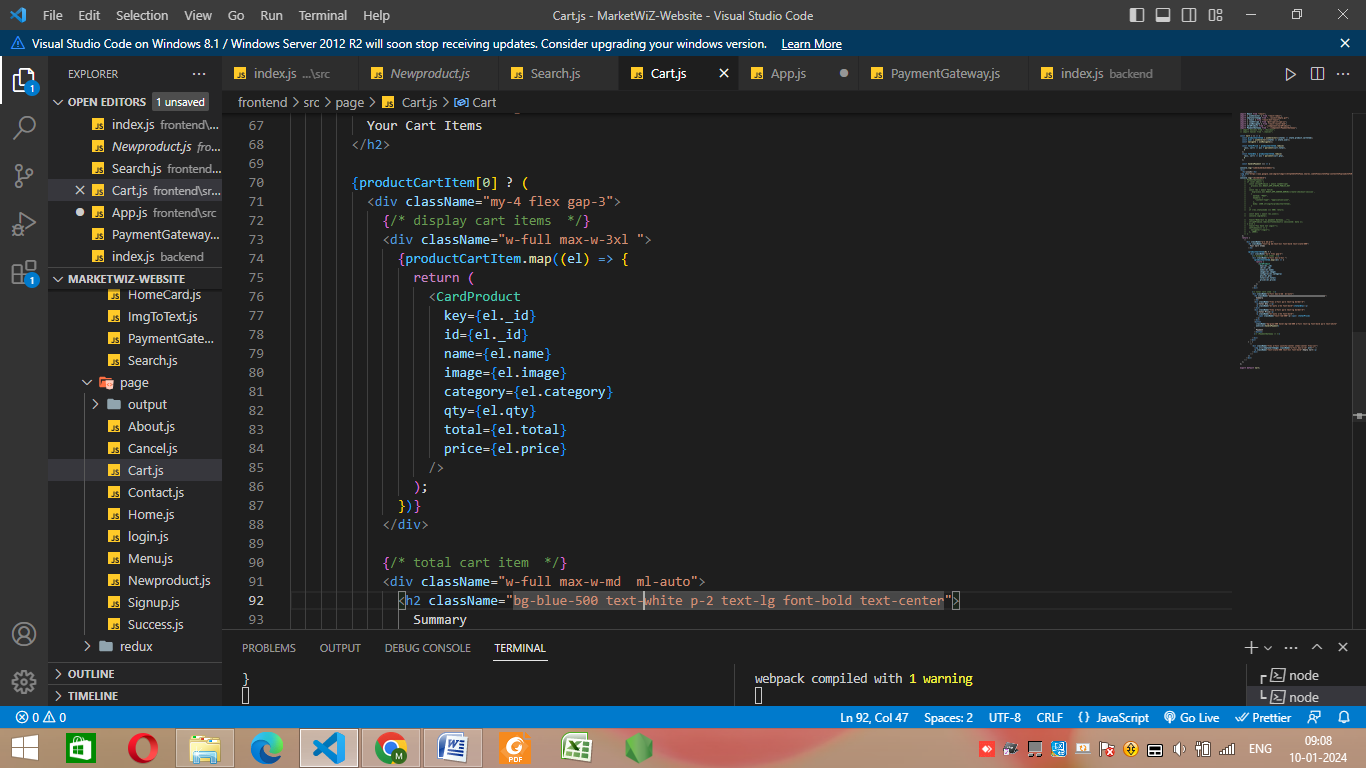
****

Fig. 4.12 Cart.Js

productCartItem[0] if it is true then it will return <div className=”my-4 flex gap-3”></div> . In this div <cartProduct key id name image qty total price> is returned.

****

Fig. 4.13 Cart.Js

**4.5 Payment.Js**

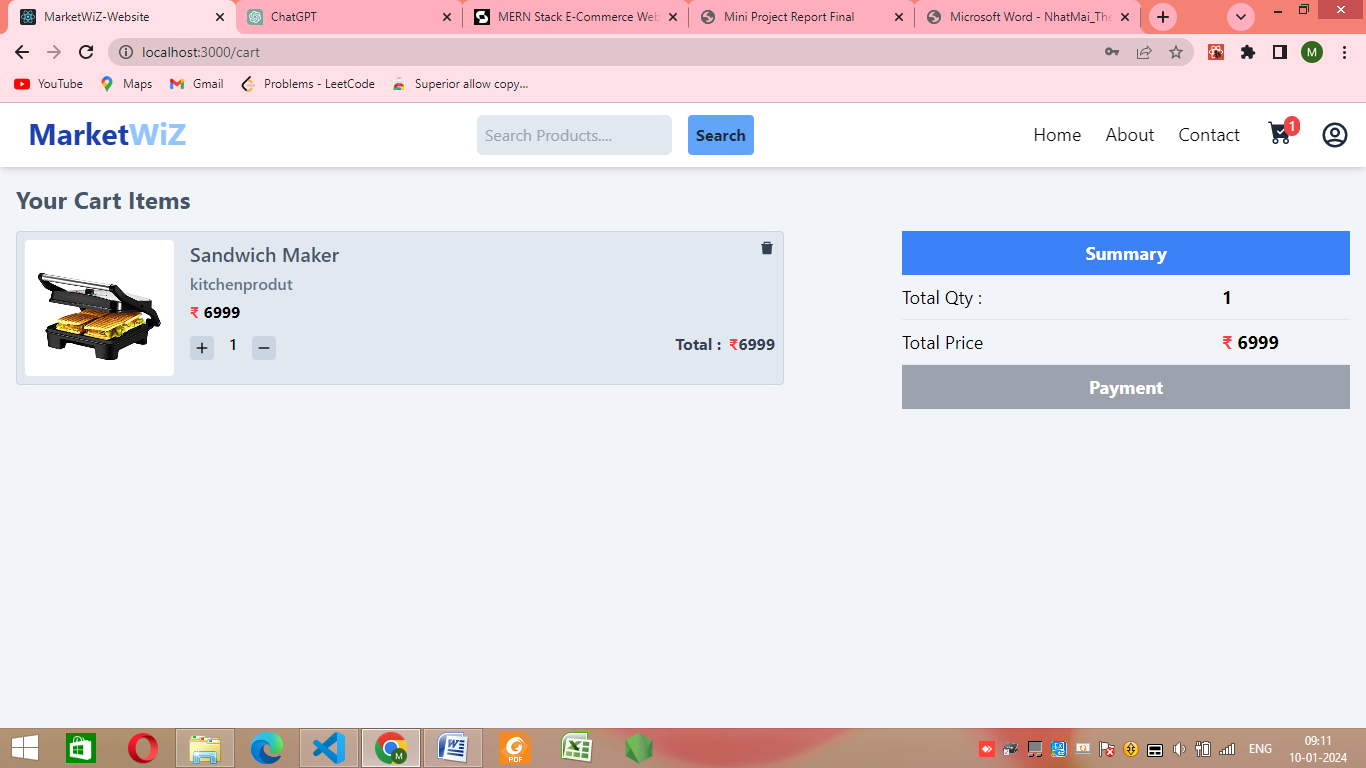
****

Fig. 4.14 Making payment

This Fig.ure is showing how to increment or decrement or delete the product from cart and making payment. The price is shown and user can make payment by clicking on payment.

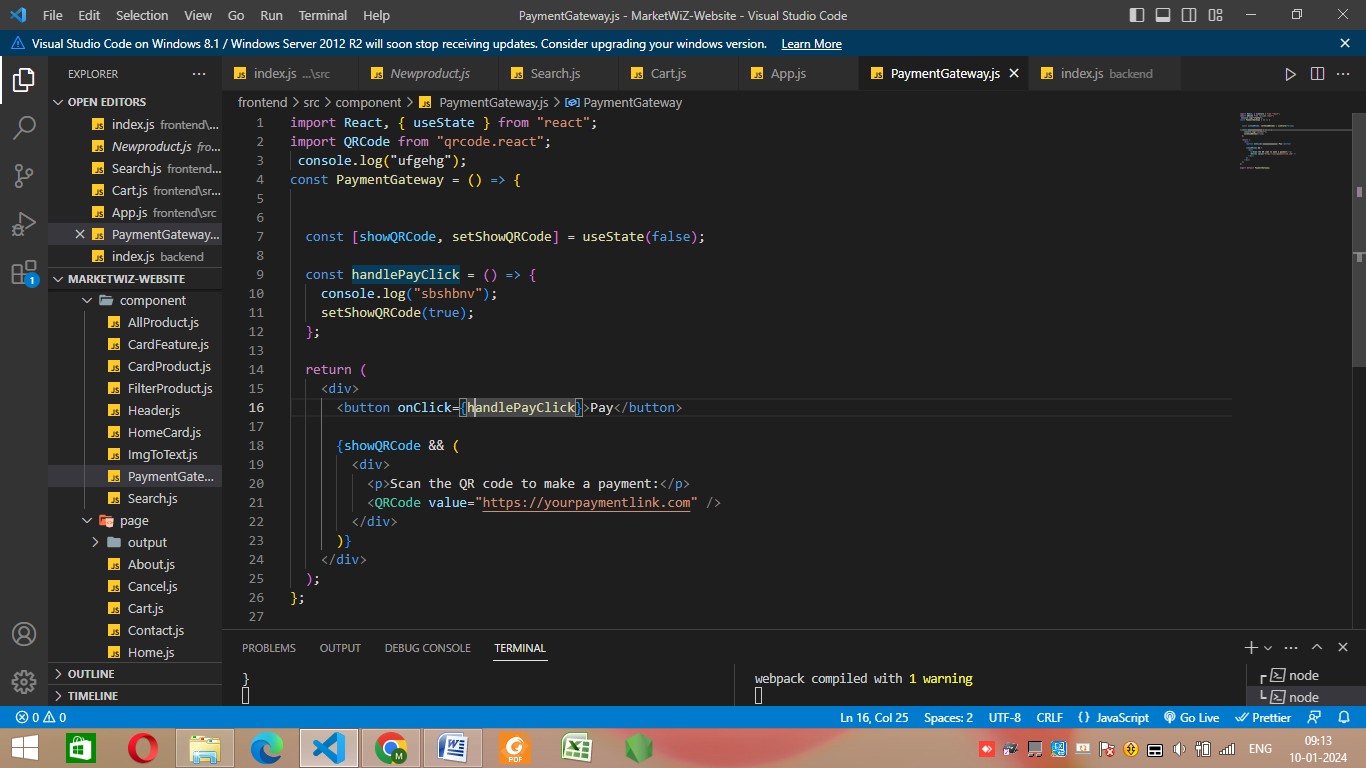
****

Fig. 4.15 Payment.Js

Above Figures are showing the coding part Payment page and how they are implemented.

**4.6 Backend**

Backend is made from Express.Js. Firstly moongose.model() is used for setting the mongoose and app.get() and app.post() for api for serach pages and for login pages.

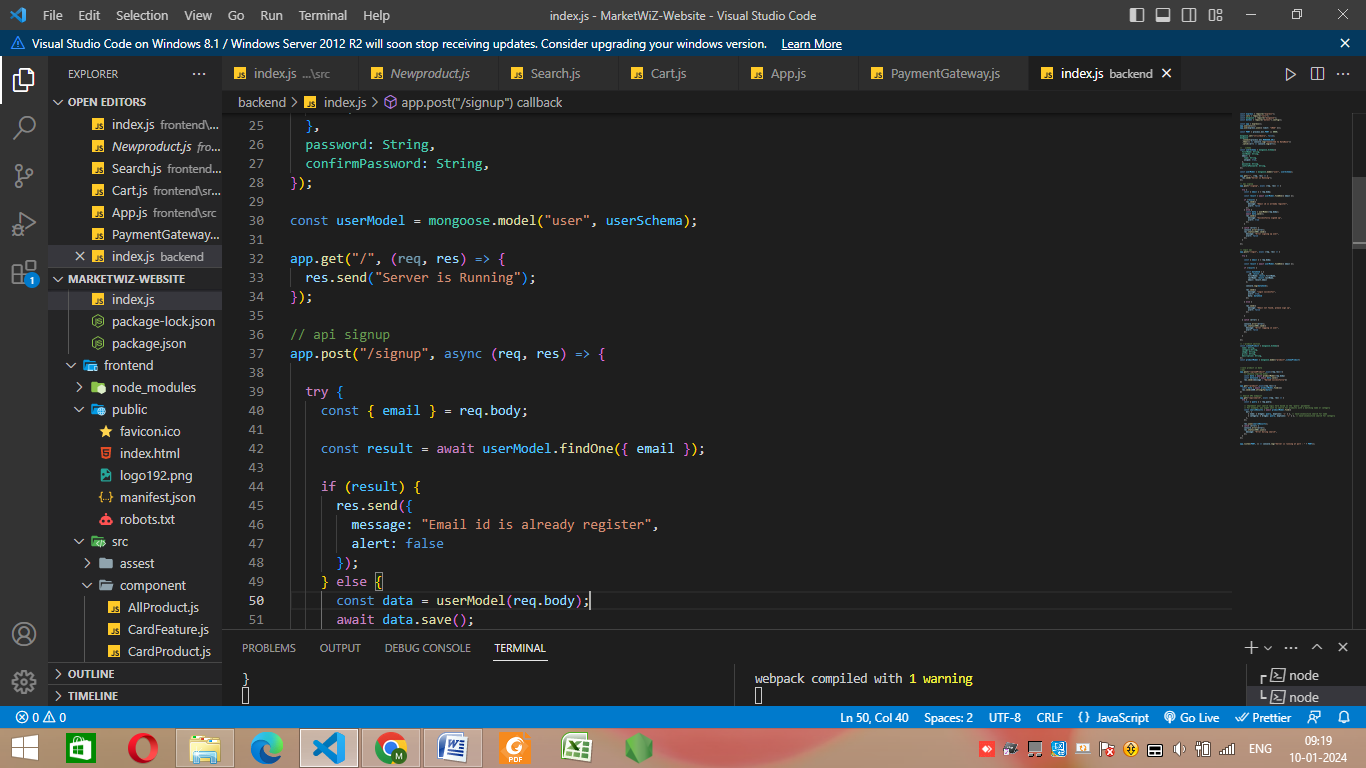
****

Fig. 4.16. Express.Js

api.get(“/product”,async(req,res)) method is implemented using productModel.find() res.json(searchResults) is used for sending response, res.send(JSON.stringify(data)) is used for response send. Similarly app.get(“/api/search”) is implemented using req.query and searchResults is implemented using productModel.find() in this res.json() is used for response send.

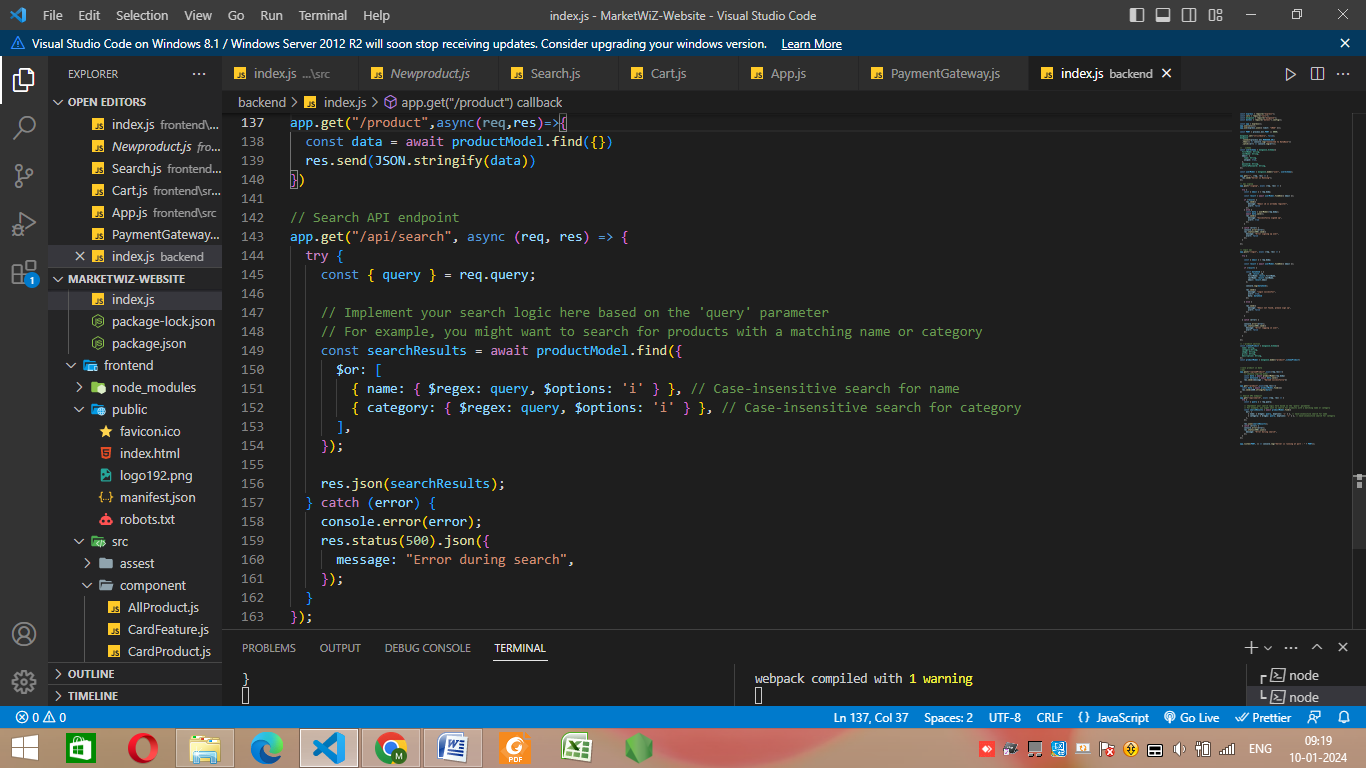
****

Fig. 4.17 Express.Js

**CHAPTER 5**

**TESTING**

Testing is a critical phase in the development of a MERN (MongoDB, Express.js, React, Node.js) E-Commerce website. It ensures that the application functions as expected, performs well, and meets the requirements of users. Here's an overview of the key testing areas for a MERN E-Commerce project.

**5.1 Unit Testing**

This testing is used for testing individual units or modules in the project.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test**  **Case ID** | **Description** | **Input** | **Website**  **Output** | **Actual**  **Output** |
| TC001 | Validate User  Login credentials | ID: [groupno42@gmail.com](mailto:groupno42@gmail.com)  Password: 12345 | Login  Successful | Login Successful |
| TC002 | Validate User  Login credentials | ID: [groupno42@gmail.com](mailto:groupno42@gmail.com)  Password: 12345678 | Invalid  credentials | Invalid  credentials |
| TC003 | Add valid Product name and category | Name: Basmati Rice  Category: groceries | Product  Added  successfully | Product  Added  successfully |
| TC004 | Add invalid Product name and category | Name: abcxyz123  Category: abc123 | No abc123 category  exist | No abc123  category  exist |
| TC005 | Remove items  From cart | Product: Basmati Rice  Quantity: 15 | Successfully removed  Basmati rice | Successfully removed  Basmati rice |
| TC006 | Calculate total  Order amount | Cart items: Basmati rice  Quantity:12  Price:100 | Order amount  =1200 rs | Order amount  =1200 rs |
| TC007 | Making Payment | Cart items: Basmati rice  Quantity:12 | Order placed  successfully | Order placed  successfully |

Table 5.1 Unit Testing Table

**5.1.1 Backend(Express.js)**

Input: Valid username and password

Expected Output: Successful authentication

****

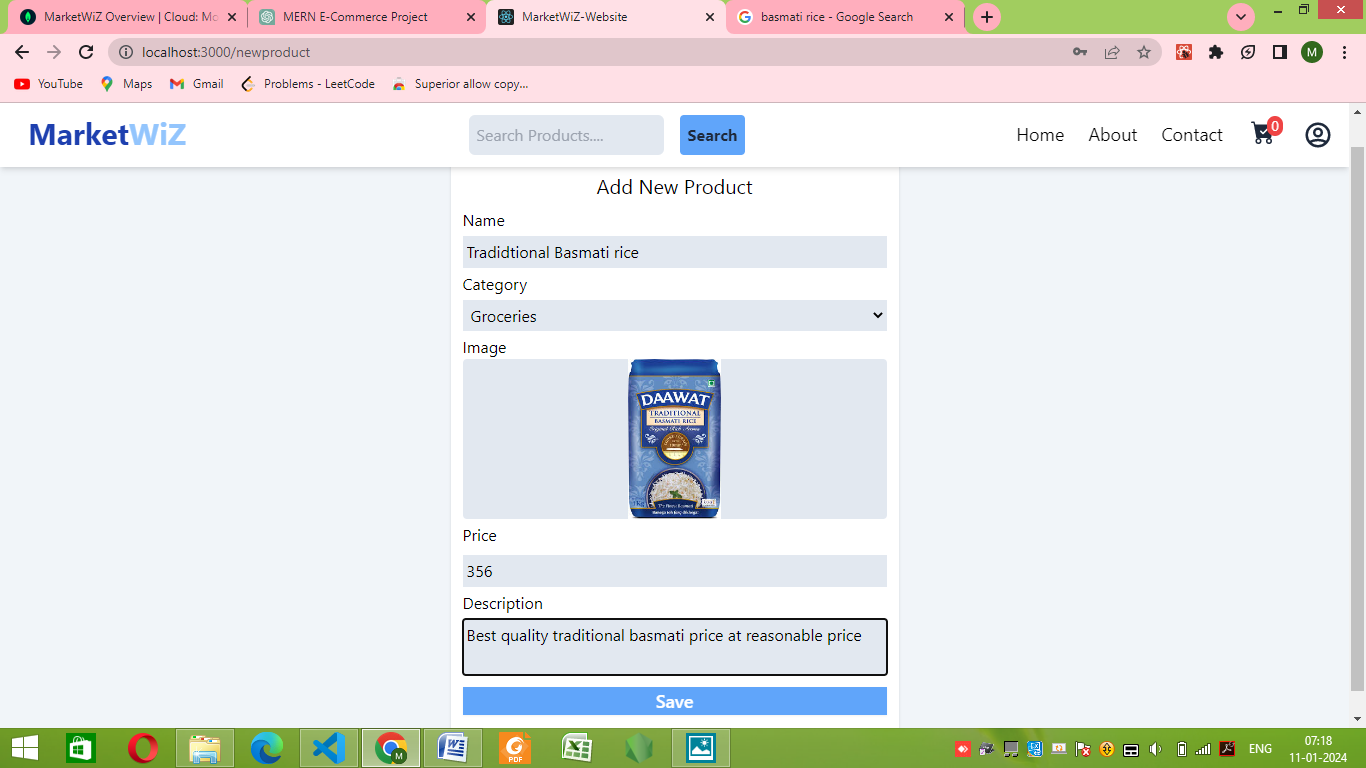
Fig. 5.1 Testing of backend

In this figure firstName: “group” lastName: “no” email: [groupno42@gmail.com](mailto:groupno42@gmail.com) is entered and server returned the object having id as 655afcc7f8f09937c2c1f70f which is unique in nature and can be used to identify object in database uniquely. nodemon index.js is used for starting the server.

**5.1.2 Adding new products**

Input: adding new product with picture and description

Output: product successfully added

****  
 Fig. 5.2 Testing of new products add

**5.1.3 Frontend(React.Js)**

Input: Adding/removing items from the shopping cart Expected

Output: Shopping cart reflects the correct number of items and total cost

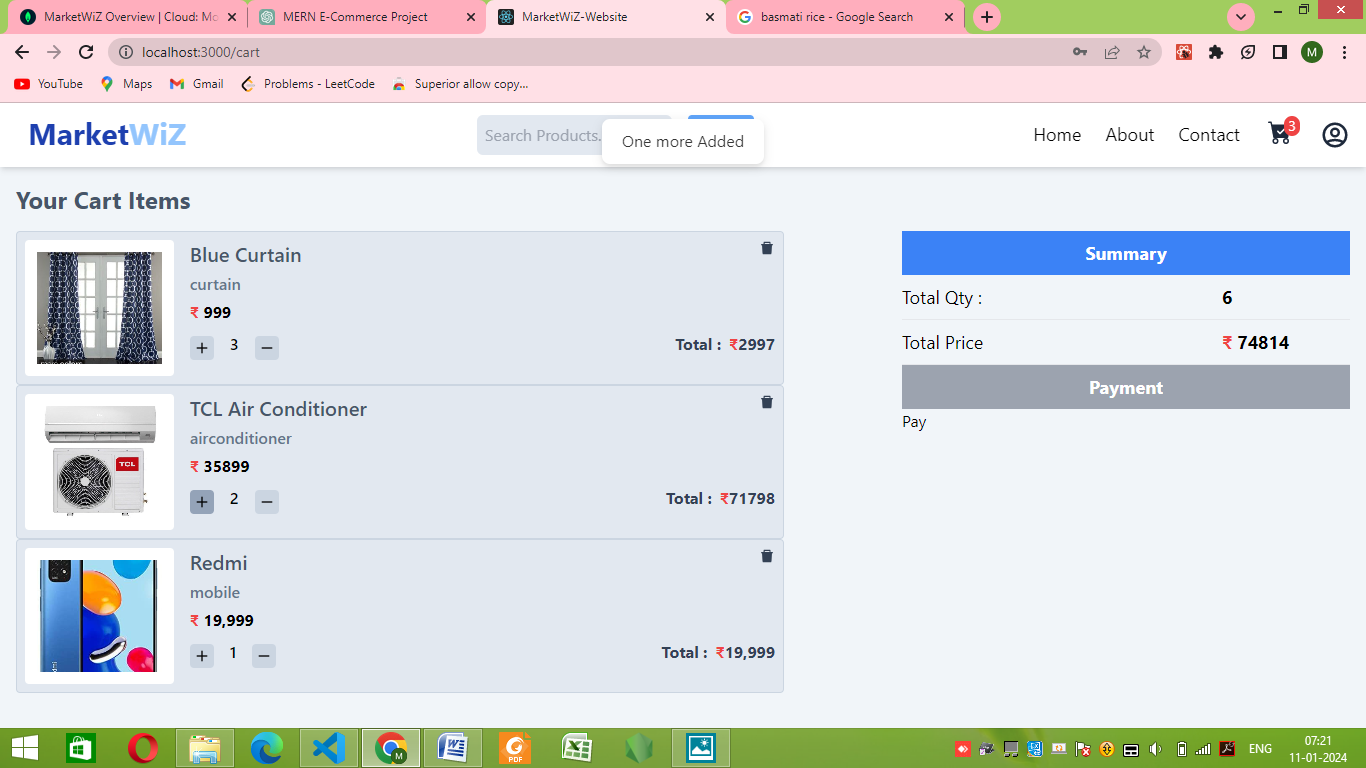
****

Fig. 5.3 Testing of frontend

**5.2 Database Testing**

Checking if database is working fine in mongoDB atlas interface.

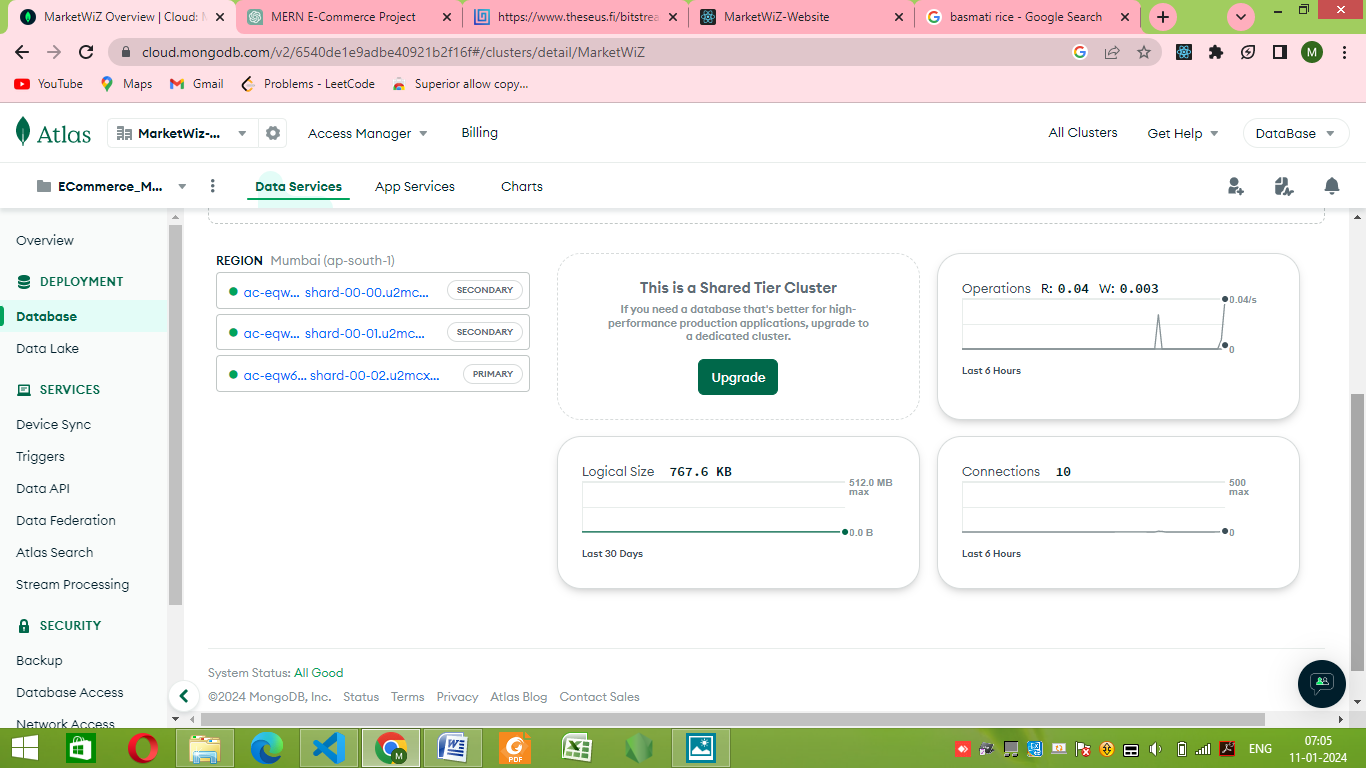
****

Fig. 5.4 Testing of database(mongoDB)

In this figure health of the database is shown in terms of operations, Logical size and connections. Operations indicate how many times database is accessed in the last 6 hours, Logical Size indicating the size of the database and connections indicating the number of users accessed the database etc.

**CHAPTER 6**

**CONCLUSION**

The achievement of the thesis is researching the basic components of MERN stack technology: MongoDB, ExpressJS framework, ReactJS library and NodeJS platform. Using MERN stack technology in conjunction with Braintree to build an e-commerce web application.

The advantages are performing the basic functions of a product search website for customers, making it easy for customers to find categories that have the products they are looking for. Gives small stores a platform to store information and promote their products. Password data of accounts when logging in to the system is stored in a secure database. The management interface, statistics of the user and admin are easy to use for everyone.

The disadvantages are online chat functions between shop owners and customers are not yet supported as well as between shop owners and administrators. The current product search algorithm locates by the user location that is not really optimal, needs to be improved to speed up the search even more.

Since the purpose of the thesis is the e-commerce application, the understanding about MERN technologies and applying it to this app is the most important.

Overcome current shortcomings, listen to customers’ comments and making improvements, helping users have a great experience in the future.

**BIBLIOGRAPHY**

**Books:**

Flanagan, D. (2017). "JavaScript: The Definitive Guide."

O'Reilly Media. Robson, S. (2014). "Node.js in Action.

"Manning Publications. Holmes, K. (2019).

"Learning React: Functional Web Development with React and Redux." O'Reilly Media.

**Articles and Blog Posts:**

"Building a Full-Stack MERN Application." FreeCodeCamp. (<https://www.freecodecamp.org/news/how-to-build-a-full-stack-mern-application/>)

"Creating a RESTful API with Node.js and Express." Scotch.io. (<https://scotch.io/tutorials/build-a-restful-api-using-node-and-express-4>)

"Introduction to React Router." reactrouter.com. (<https://reactrouter.com/web/guides/quick-start>)

**Forums and Community:**

Stack Overflow.https://stackoverflow.com/ Reddit - r/MERN. (Year). https://www.reddit.com/r/MERN/ MongoDB Community Forums. (Year). https://community.mongodb.com/

**REFERENCES**

**Online Documentation:**

MongoDB Documentation. (<https://docs.mongodb.com/>)

Express.js Documentation. (<https://expressjs.com/>)

React Documentation. (<https://reactjs.org/docs/getting-started.html>)

Node.js Documentation. (https://nodejs.org/en/docs/)

**Frameworks and Libraries:**

Bootstrap. (<https://getbootstrap.com/>)

Redux. (<https://redux.js.org/>)

Axios. (<https://axios-http.com/>)

Mongoose. (https://mongoosejs.com/)