

MINI PROJECT REPORT

ON

CRAFT HIVE

Submitted by

RINTA MARIA RAJU(SJC20CS106)

to

the A P J Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of the degree

of

Bachelor of Technology

in

Computer Science and Engineering



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ST.JOSEPH'S COLLEGE OF ENGINEERING AND TECHNOLOGY PALAI

JULY :: 2023

DECLARATION

I undersigned hereby declare that the mini project report on “CRAFT HIVE Web Application” submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of Prof. Jaya John, Assistant Professor, Dept of CSE. This submission represents my ideas in my own words and where ideas and words of others have been included. I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a course for disciplinary action by the institute and/or the University and can also evoke panel action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Name and Signature of Student

RINTA MARIA RAJU(SJC20CS106)

Place: Choondacherry

Date: 27-06-2023

ST. JOSEPH'S COLLEGE OF ENGINEERING AND TECHNOLOGY, PALAI

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the report entitled **CRAFT HIVE** Web Application submitted by **RINTA MARIA RAJU(SJC20CS106)** to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering is a bonafide record of the mini project work carried out by them under my guidance and supervision.

Mini Project Guide

Prof. Jaya John

Assistant Professor

Dept of CSE

Mini Project Coordinator

Prof. Kishore Sebastian

Assistant Professor

Dept of CSE

Head of the Department

Dr. Joby P.P

Professor & Head

Dept of CSE

Place: Choondacherry

Date: 27-06-2022

ACKNOWLEDGMENT

The success and final outcome of this mini project required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of this project. All that we have done is only due to such supervision and assistance and we would not forget to thank them.

I respect and thank the management of St. Joseph's College of Engineering and Technology for providing as an opportunity and platform for doing the project work.

I am extremely thankful to our beloved Principal, **Dr. V. P Devassia** for providing such nice support and facilities to carry out this project.

I am extremely indebted to **Dr. Joby P.P**, Professor & Head, Department of Computer Science and Engineering for his valuable suggestions and encouragement during the course of this project work.

I would like to express my gratitude to my project coordinator **Prof. Kishore Sebastian**, Asst. Professor, Department of Computer Science and Engineering for his valuable suggestions and guidelines throughout the period of this project work. His contributions and technical support in preparing this report are greatly acknowledged.

I owe my deep gratitude to our project guide **Prof. Jaya John**, Asst. Professor, Department of Computer Science and Engineering who took keen interest in this project and guided as all along, till the completion of this project work by providing all the necessary information for developing a good system.

I am thankful and fortunate enough to get constant encouragement, support and guidance from all the staff members of the department of Computer Science and Engineering, which helped as in successfully completing mini project work.

ABSTRACT

Arts and crafts is one of our oldest past times. People craft things all the time. Sometime it's Silly little things with clay or popsicle sticks. The project is aimed to help small scale Craft makers to gain an income by selling their products. Nowadays most of the craft makers sell their products through social media like Instagram, Facebook etc. The problem is that these small scale sellers doesn't get enough reach on these platforms, though they post it on social media if they aren't among the top their works are not recognized and it is a waste of time and money for them. The application help small scale makers to get enough reach through advertisements. The customers can buy, pay, review the product .This application also gives the buyers an option for requesting customized product. If the request is accepted by the seller ,buyer have to pay an advance amount. If they have any complaints about the product they can place complaints.

Chapter 1 : INTRODUCTION

1.1 Problem Statement	10
1.2 Objectives and Scope	10

Chapter 2 : LITERATURE SURVEY

2.1 ArtFire	12
2.2 A-Crafty-Market	12
2.3 I CraftGifts	13
2.4 Made It Myself	13

Chapter 3 :SOFTWARE REQUIREMENT SPECIFICATION(SRS)

3.1 Functional Requirements	14
3.2 Non Functional Requirements	15
3.3 Module Description	16

Chapter 4: SOFTWARE DESIGN(SDD)

4.1 Activity Diagram	18
4.2 Class Diagram	22
4.3 ER Diagram	23
4.4 Use Case Diagram	25

Chapter 5 :SYSTEM IMPLEMENTATION

5.1 Technology Stack	26
5.1.1 HTML	26
5.1.2 CSS	26
5.1.3 Java Script	27
5.1.4 Node js	27
5.1.5 Mongo db	27

5.2 Frontend	28
5.2.1 Signup and Login	29
5.2.2 Admin	30
5.2.3 Cart	30
5.2.4 Payment	30
5.2.5 Shipment	31
5.3 Backend	32
Chapter 6 : TESTING	
6.1 Various Testing method	33
6.1.1 Unit Testing	33
6.1.2 Integration Testing	33
6.1.3 Functional Testing	33
6.1.4 Load Testing	33
CONCLUSION	38
REFERENCES	39

LIST OF FIGURES

4.1	Activity Diagram	19
4.1	Class Diagram	22
4.3	ER Diagram	24
4.4	Use Case Diagram	25
5.2.1	Home Page	28
5.2.2	Sign up and Login	29
5.2.3	Cart Dashboard	30
5.2.4	Payment Dashboard	31
5.2.5	Shipment Dashboard	31
5.3	Backend	32
6.1.4	Testing	34

LIST OF ABBREVIATIONS

HTML - Hypertext Markup Language

CSS - Cascading Style Sheets

SQL - Structured Query Language

DBMS - Database Management System

SNS - Social Networking Service

CHAPTER 1

INTRODUCTION

Web development refers to the building, creating, and maintaining of websites. It includes aspects such as web design, web publishing, web programming, and database management. It is the creation of an application that works over the internet i.e. websites. Web development can range from developing a simple single static page of plain text to complex web applications, electronic businesses, and social network services. A more comprehensive list of tasks to which Web development commonly refers, may include Web engineering, Web design, Web content development, client liaison, client-side/server-side scripting, Web server and network security configuration, and e-commerce development. Among Web professionals, "Web development" usually refers to the main non-design aspects of building Web sites: writing markup and coding. Web development may use content management systems (CMS) to make content changes easier and available with basic technical skills.

1.1 Problem Statement

People craft things all the time. Sometime it's silly little things with clay or popsicle sticks. The project is aimed to help small scale craft makers to gain an income by selling their products. Nowadays most of the craft makers sell their products through social media like Instagram, Facebook etc. The problem is that these small scale sellers doesn't get enough reach on these platforms, though they post it on social media if they aren't among the top their works are not recognized and it is a waste of time and money for them.

1.2 Objectives and Scope

This web application is a user-friendly platform designed to empower small-scale craft makers by providing them with a convenient marketplace to sell their products. The application offers a range of essential features to facilitate a seamless experience for both sellers and buyers. Sellers can easily add and delete products based on product availability, ensuring accurate and up-to-date listings. Buyers, on the other hand, can share their

valuable feedback and opinions by reviewing the products they have purchased, helping other users make informed decisions. The application also includes a secure payment and refund system, ensuring that buyers can make hassle-free payments and receive refunds if they choose to cancel their orders. For customized products, buyers are able to request personalized items, and a chat feature is provided to enable direct communication with the sellers. This fosters collaboration and allows buyers to discuss their specific requirements, ensuring the creation of a tailored and satisfactory product. Overall, this web application serves as a valuable platform for craft makers and buyers alike, fostering a vibrant and supportive community while offering a seamless and enjoyable user experience.

CHAPTER -2

LITERATURE SURVEY

A craft product selling website is an online platform that provides a dedicated space for independent crafters, artisans, and small businesses to showcase and sell their unique handmade creations. Crafters can create individual profiles, sharing their background and creative process, while product listings feature compelling descriptions, high-quality images, and details about materials used. The website offers a secure shopping cart and checkout system, allowing customers to browse and purchase craft products with ease. Integration with trusted payment gateways ensures secure transactions, and customer support channels enable communication between buyers and sellers. Ratings and reviews contribute to building trust, while social sharing options allow customers to spread the word about their favorite craft products. Ultimately, the craft product selling website serves as a vibrant online marketplace, connecting passionate crafters with customers who appreciate the artistry and personal touch behind each handmade item.

2.1 Art Fire

Art Fire: While Art Fire is a platform for selling handmade and vintage items, one disadvantage is that it can be complex to navigate and set up for some users. The interface and features may not be as user-friendly as other platforms, which could make it more challenging for sellers, especially those who are less tech-savvy.

2.2 A-Crafty-Market

A-Crafty-Market: A-Crafty-Market is a craft marketplace that requires sellers to apply and be accepted. One disadvantage is that it has limited spots available for sellers, making it a more exclusive platform with a smaller number of sellers compared to larger marketplaces. This limited availability may reduce the potential customer base and exposure for craft makers.

2.3 ICraftGifts

ICraftGifts is an online marketplace that charges a monthly subscription fee for sellers to list their products. This payment requirement can be a disadvantage, especially for small-scale craft makers or those starting out, as it adds an additional financial commitment to participate on the platform.

2.4 Made It Myself

Made It Myself: Made It Myself is a craft marketplace where sellers can create their own online store. One potential disadvantage is that the website design and user interface may not be as modern or visually appealing compared to other platforms. This could impact the overall shopping experience for buyers and potentially reduce sales for sellers.

CHAPTER -3

SOFTWARE REQUIREMENTS SPECIFICATION

3.1 FUNCTIONAL REQUIREMENTS

- Hardware Requirements

RAM:4GB and above

Processor: Pentium Processor and above

- Software Requirements

Front end: HTML 5.0,CSS,React.js

Back end : node.js , MongoDB

- User registration and login: The website should allow users to create accounts and login securely.
- Search and filtering: Buyers should be able to search for products using keywords and filter by various criteria such as price range, product category, brand, and other relevant attributes.
- Product catalogue: The platform should have a catalogue of products that sellers can list and buyers can search and browse. The catalogue should have details such as product descriptions, images, price, and stock availability.
- Shopping cart and checkout: Buyers should be able to add products to their shopping cart and proceed to checkout, where they can review their order, select a payment method, and provide shipping details.
- Product Customization: A chatting platform is provided for customized products.
- Payment processing: The platform should facilitate secure payment processing, allowing buyers to pay for their orders using payment gateways.

- Order management: Sellers should be able to manage their orders, updating the status of orders.
- Reviews and ratings: Buyers should be able to leave reviews and ratings for products and sellers, which can help other buyers make informed decisions and also help sellers improve their products and services.
- Account management: The website should allow users to update their account information, manage their addresses, and update their payment methods.

3.2 NON FUNCTIONAL REQUIREMENTS

- Performance Requirements

Maximum possible quick response to the orders is required, also should provide fast updating of records. The changes if any made should be reflected automatically in the next screens. In order to maintain an acceptable speed at maximum number of uploads allowed from a particular customer as any number of users can access to the system at any time. Also the connections to the servers will be based on the attributes of the user like his location and server will be working 24X7 times.

- Safety Requirement

Payment security: Ensure that the payment process is secure and encrypted. Use trusted payment gateways, which have their own security measures in place.

Strong passwords: Use strong passwords and change them regularly to prevent unauthorized access to your account.

Privacy policy: Make sure that the online shop has a clear and transparent privacy policy, which outlines how they collect, store, and use your personal information.

- Software Quality Attributes

Security: The application is password protected and also any updation of new product entries and order processing is done by only privileged users.

Maintainability: The application is to be designed so that it is easily maintained. Also it should allow incorporating new requirements in any module of system.

Reliability: The application will be able to handle two orders. When a user confirms his/her order the database will be updated immediately and the next user will not face problems in ordering.

Portability :The application will be easily portable on any window based system.

3.3 MODULE DESCRIPTION

- **User Module**

Login Page : The login page is the initial access point for users to securely authenticate and gain access to a specific system or platform, typically requiring a username and password. It serves as a gateway for users to enter and utilize the features and functionalities of the application or website.

Browsing Product : A user action that allows them to browse through the available products in the online shop.

- **Online Shop**

Product Catalogue : A product catalog is a comprehensive listing of items providing detailed information such as descriptions, prices, and images. It serves as a reference guide for customers to browse and select products, facilitating their purchasing decisions.

Cart : A cart allows users to collect and store items they intend to purchase while shopping. It provides a convenient way for users to review, modify, and proceed to checkout with their selected products before completing the transaction.

Customized Product Request : Buyer can request a product to be customized for their need.

- **Payment**

Payment Gateway : Razor pay is a popular payment gateway that enables businesses to securely accept online payments from customers. It offers a seamless integration, robust security features, and supports various payment methods, providing a reliable solution for processing transactions and ensuring a smooth checkout experience.

Payment Processing : A service that processes payments on behalf of the online shop.

- Orders

Order Management : Order management involves the end-to-end process of handling customer orders, from initial placement to fulfillment and delivery. It encompasses activities such as order processing, inventory management, tracking, and customer communication, ensuring efficient and timely order fulfillment.

Order Status : Status about where order shipped or not.

- Analytics

Feedback : Feedback or reviews are valuable insights provided by customers that reflect their opinions, experiences, and satisfaction with a product. They serve as a crucial source of information for businesses to evaluate and improve their offerings, build credibility, and make informed decisions based on customer feedback.

Data Management : Management of all the data in the database

CHAPTER 4

SOFTWARE DESIGN

4.1 Activity Diagram

Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that models the flow from one activity to another activity. Activity diagrams can be regarded as a form of a structured flowchart combined with a traditional data flow diagram. Typical flowchart techniques lack constructs for expressing concurrency. However, the join and split symbols in activity diagrams only resolve this for simple cases; the meaning of the model is not clear when they are arbitrarily combined with decisions or loops.

In fig 4.1.1 the admin starts by logging into the admin panel and accessing the admin dashboard. the admin can manage customers by verifying them as buyer and seller. The admin can perform these actions as needed and can log out when finished.

There are two types of users buyer and seller. In case of seller they can start by logging into the seller panel from there they can manage their product by adding ,editing or deleting them they can also view the order listings as shown in fig 4.1.2 and 4.1.3.

Buyer can start by logging in to buyer panel. they can view the products the seller has added and are given a feature to add the product into cart ,from there they can confirm the order and goes to payment dashboard as shown in fig 4.1.2 and 4.1.3.

FOR ADMIN

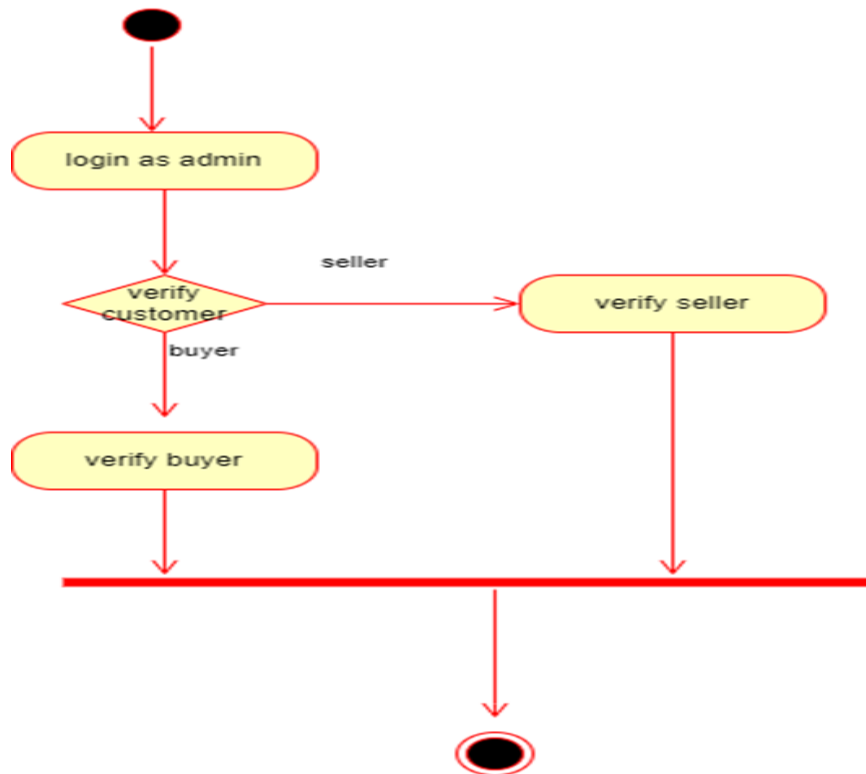


Fig: 4.1.1: Activity Diagram

FOR USER (NOT CUSTOMIZED PRODUCT)

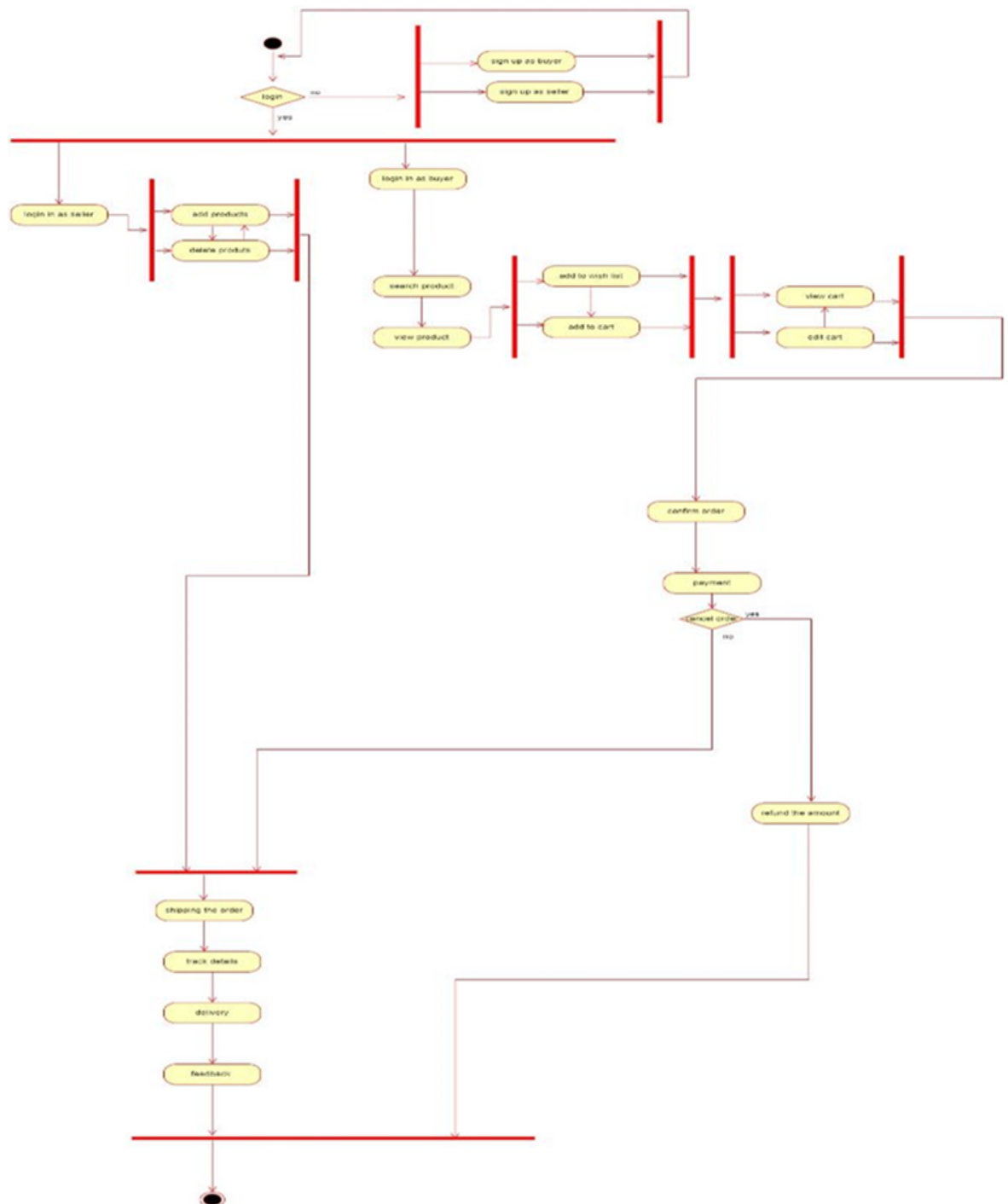


Fig: 4.1.2: Activity Diagram

FOR USER (CUSTOMIZED PRODUCT)

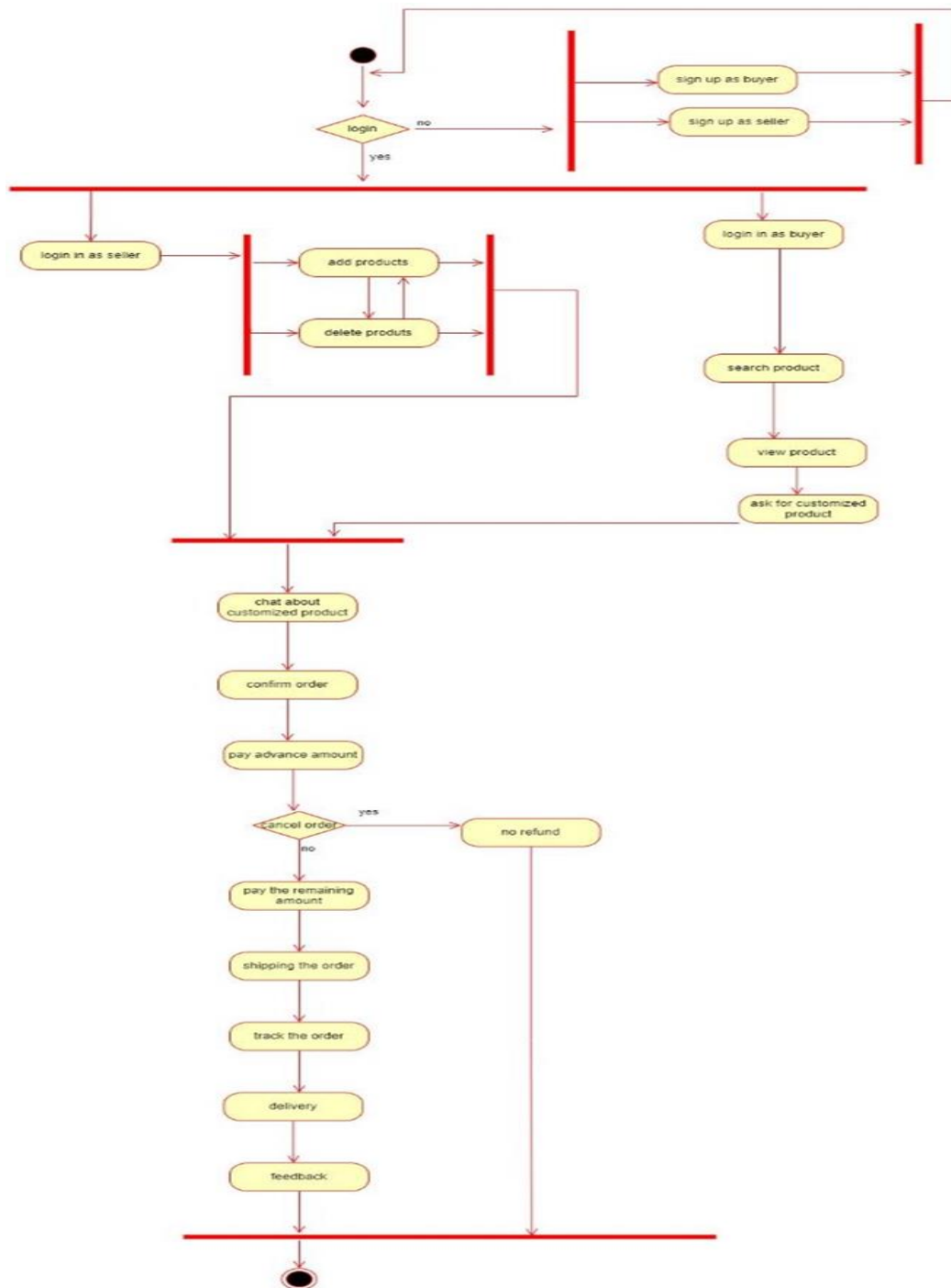


Fig: 4.1.3: Activity Diagram

4.2 Class Diagram

Fig 4.2 shows class diagram. Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

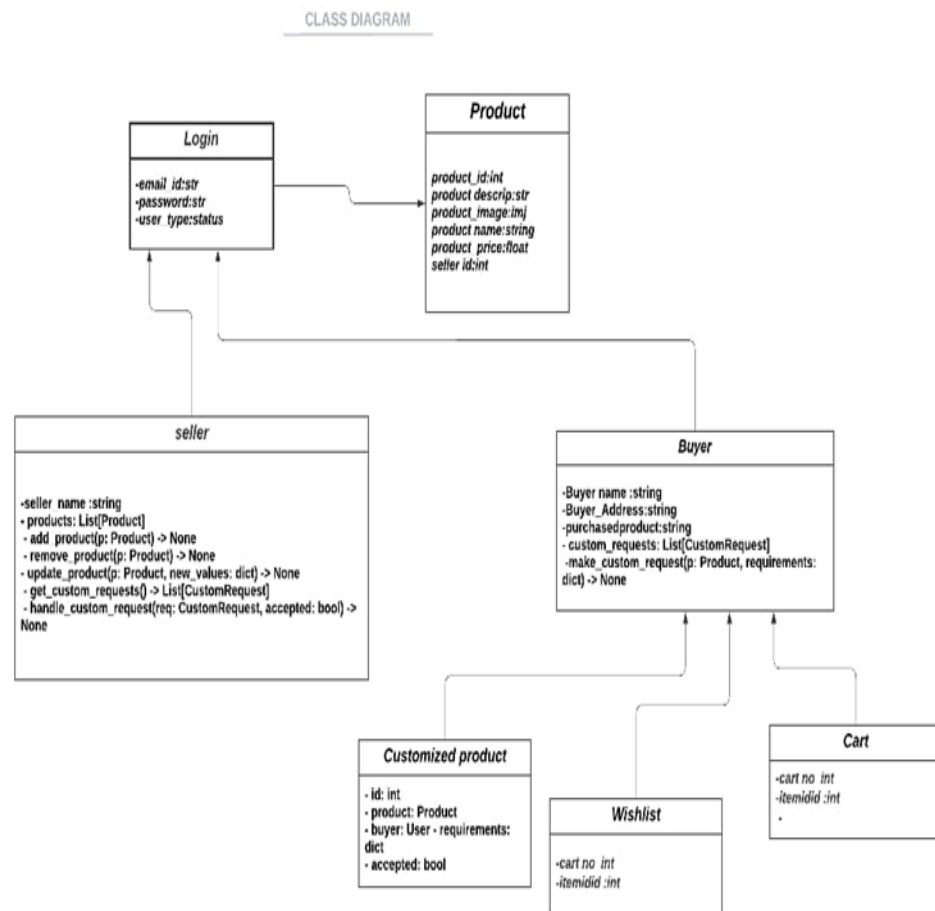


Fig: 4.2: Class Diagram

4.3 ER Diagram

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships. ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.

In fig 4.3, the ER diagram consists of entities Admin, seller, buyer, feedback, wishlist, payment, cart, product, purchase along with their attributes. Admin logs in with specific mail id and password. There are multiple sellers and buyers and they login with their email id and password. The seller sells the products. The buyer can view these products, add them to cart, can make payment and give feedback.

1. User/Customer: Represents the individuals who visit the ecommerce platform to browse and purchase products.
2. Product : Represents the items available for purchase on the ecommerce platform
3. Order: Represents a customer's purchase transaction.
4. Payment: Represents the payment details associated with an order.
5. Cart: Represents the temporary container where users can store products before proceeding to checkout.
6. Feedback: Represents user-generated feedback or ratings for products they have purchased.

ER DIAGRAM

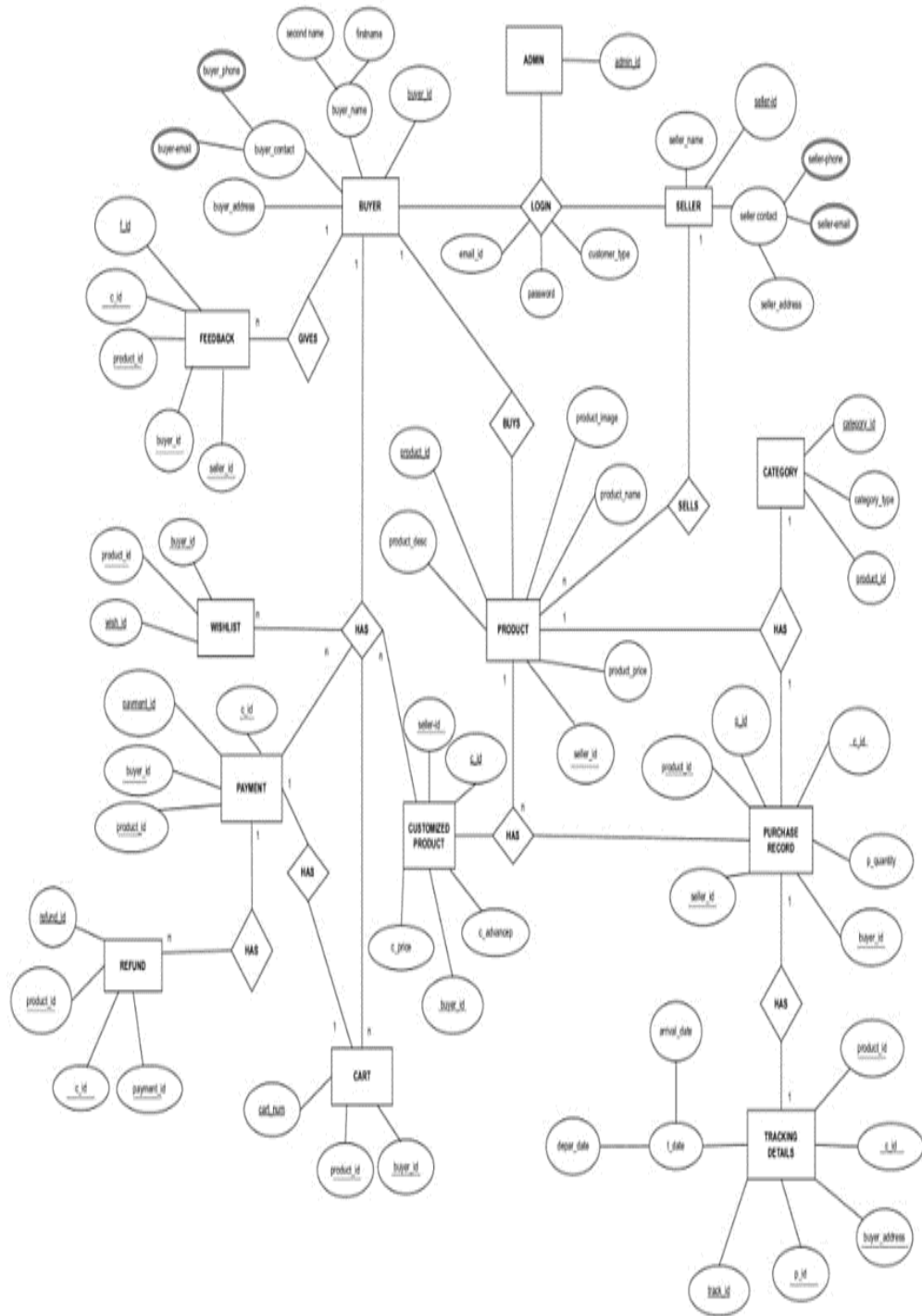


Fig: 4.3: ER Diagram

4.4 Use Case Diagram

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

In fig 4.4 ,the use case diagram consists of three actors Admin, buyer ,seller. Users can register and login to the application.It also shows the different functions provided for each actor as usecases.

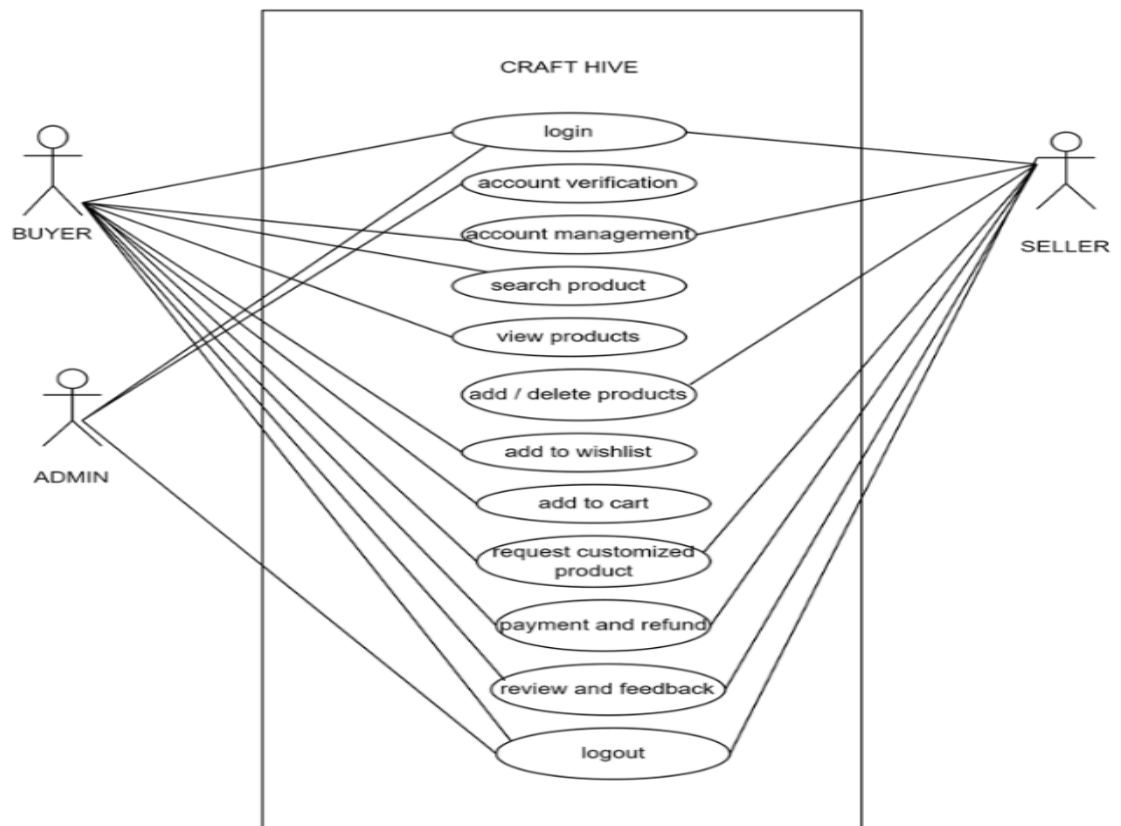


Fig: 4.4: Use Case Diagram

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 Technologies Used

5.1.1 HTML

The Hypertext Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as `` and `<input />` directly introduce content into the page. Other tags such as `<p>` surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

5.1.2 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple webpages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

5.1.3 JavaScript

JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM). The ECMAScript standard does not include any input/output (I/O), such as networking, storage, or graphics facilities. In practice, the web browser or other runtime system provides JavaScript APIs for I/O.

5.1.4 Node js

Node.js is an open-source, server-side JavaScript runtime environment that allows you to run JavaScript code on the server. It uses the V8 JavaScript engine, which is the same engine that powers the Google Chrome web browser. Node.js provides an event-driven, non-blocking I/O model, making it lightweight and efficient for building scalable network applications. Node.js allows developers to use JavaScript on both the client-side and server-side, providing a unified programming language for full-stack web development. This enables code reuse and simplifies the transition between client-side and server-side programming.

5.1.5 Mongo DB

MongoDB is a popular, open-source, NoSQL (non-relational) database management system. It is designed to store and manage large amounts of data across distributed systems, providing scalability, flexibility, and high performance. MongoDB stores data in a flexible, JSON-like format called BSON (Binary JSON), which allows for easy manipulation and retrieval of data.

5.2 RESULTS

The system architecture consists of a web application which is the front end that is developed in HTML, CSS, JAVASCRIPT. The Backend of this application is built using Mongo db. and Node js. The live server is deployed using the XAMPP.

5.2.1 Frontend

A person can authenticate themselves, and after doing so, they are taken to a dashboard where they can buy or sell craft products. The administrator can keep track of money transactions, list out products, and confirm roles. Person registered as sellers could add new items and sell at various prices, while others can buy new products or buy newly customized products. Fig 5.2.1 A and fig 5.2.1 B shows the home page.

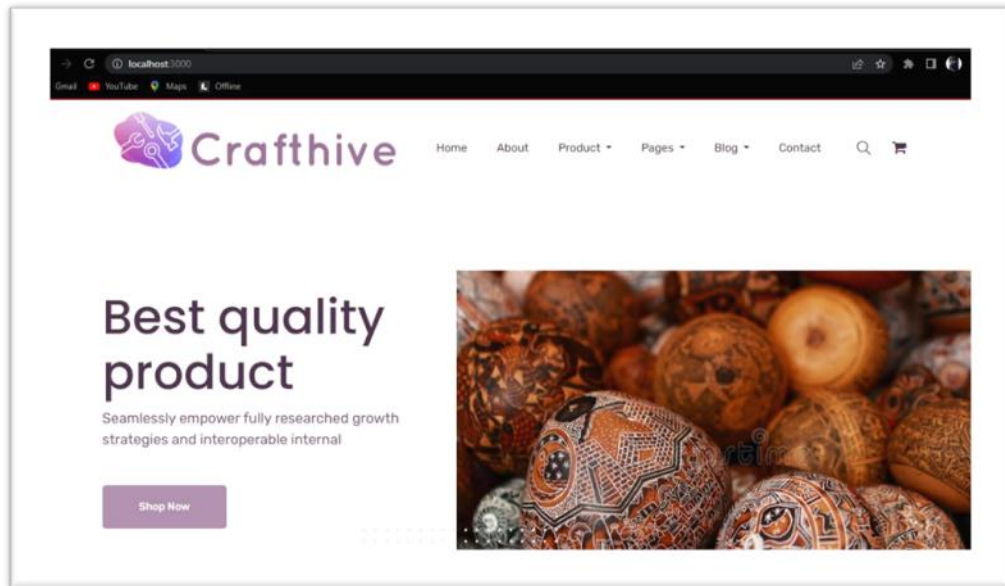


Fig 5.2.1 A Home Page

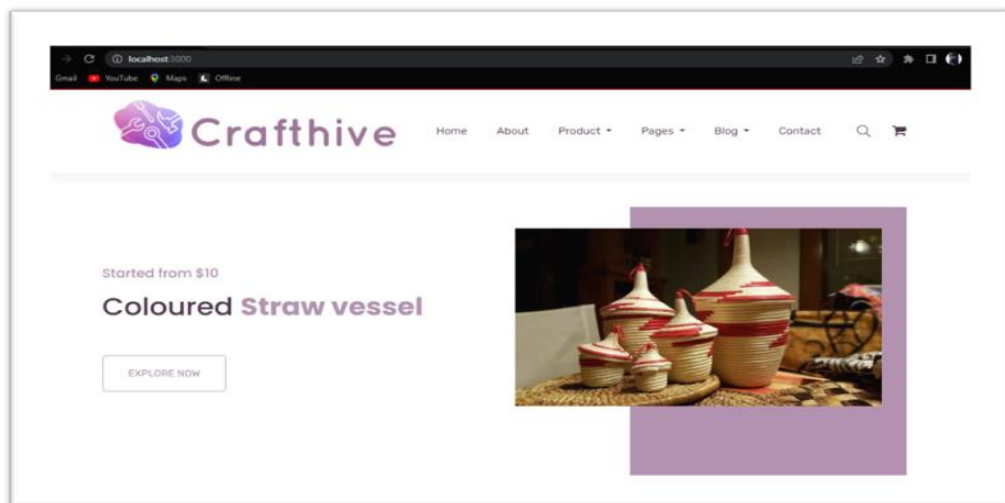


Fig 5.2.1 B Home Page

5.2.2 Signup and Login

In fig 5.2.2 ,when the user enters the application, he will have to authenticate himself. A new user or unregistered user entering will have to get registered using name, email id and password and continue to the login page. New user can be buyer or seller, where sellers need to verify themselves to nullify any types of frauds.

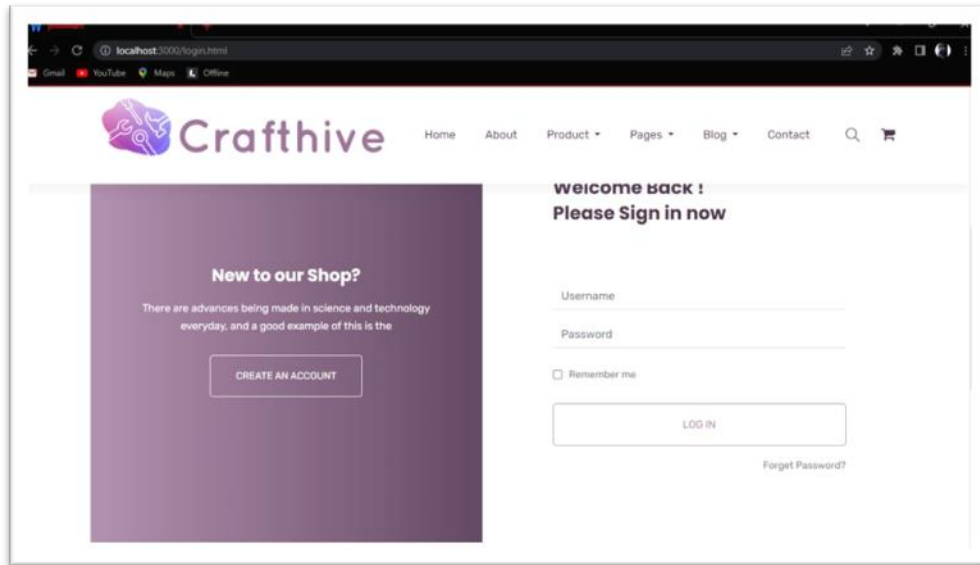


Fig: 5.2.2: Sign Up and Login

5.2.3 Admin

When the admin enters the application, he can view the list of persons registered as sellers and buyers and can authenticate purchases and view processes.

5.2.3 Cart

In fig 5.2.3, the items that needed to be purchased are added to the cart , buyer can proceed for payment or delete items from the cart .

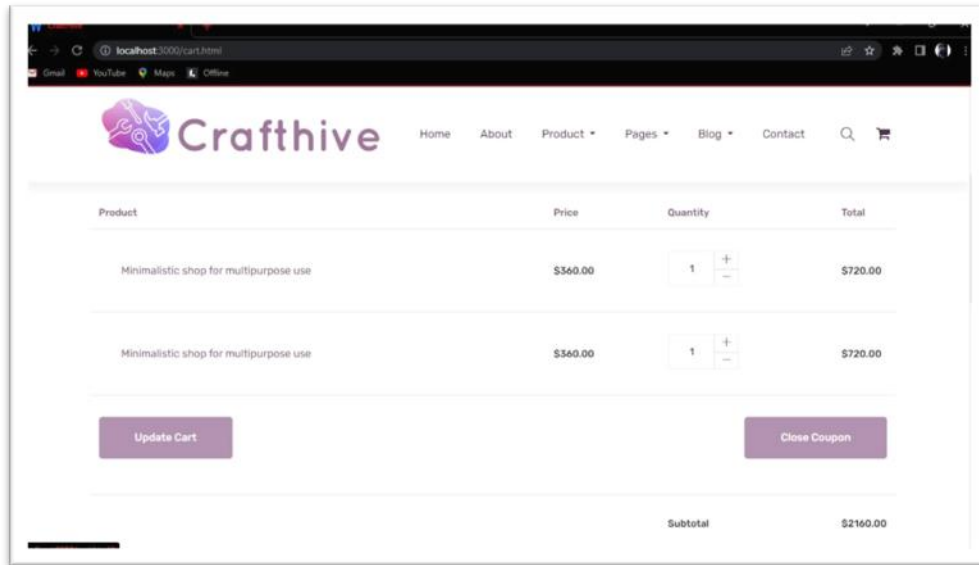


Fig 5.2.3 cart

5.2.4: Payment Dashboard

In fig 5.2.4, payment details for the purchase is being processed. In online payment users can pay the fee through the razor pay gateway which includes multiple options like UPI, credit or debit card and even net banking. This functionality is done using API calls.

Billing Details

First name * Last name *

Company name

Phone number * Email Address *

Country
Country
Country
Country

Address line 01 *

Address line 02 *

Your Order

Product		Total
Fresh Blackberry	x 02	\$720.00
Fresh Tomatoes	x 02	\$720.00
Fresh Broccoli	x 02	\$720.00
SUBTOTAL		\$2160.00
SHIPPING	Flat Rate: \$50.00	
TOTAL		\$2210.00

☐ CHECK PAYMENTS

Please send a check to Store Name,

Fig 5.2.4 Payment Dashboard

5.2.5: Shipment Dashboard

In fig 5.2.5,a shipment dashboard provides a comprehensive overview of the shipping process, allowing businesses to track and manage their shipments in real-time. It offers insights into shipping statuses, carrier information, delivery timelines, and enables businesses to streamline logistics operations for efficient order fulfillment.

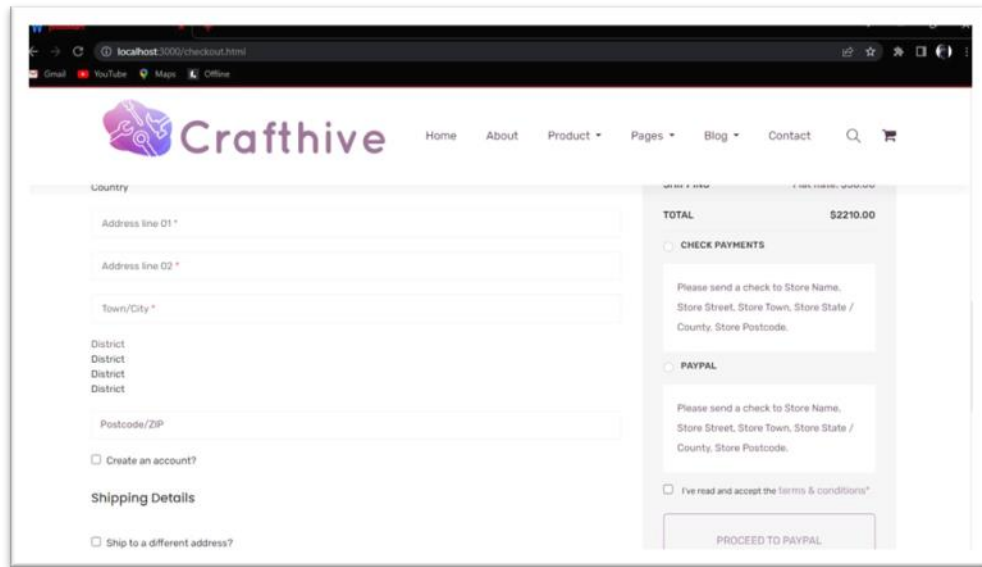
The image is a screenshot of a web browser displaying the 'Craffthive' checkout page. The browser's address bar shows 'localhost:3000/checkout.html'. The page features a navigation bar with links for Home, About, Product, Pages, Blog, and Contact, along with a search icon and a shopping cart icon. The main content area is divided into two columns. The left column contains a form for shipping details, including fields for Country, Address line 01, Address line 02, Town/City, District (with three dropdown menus), Postcode/ZIP, and checkboxes for 'Create an account?' and 'Ship to a different address?'. The right column displays the 'TOTAL' amount as '\$2210.00' and offers two payment options: 'CHECK PAYMENTS' and 'PAYPAL'. Both options include instructions to send a check to the store name, street, town, state, and county/postcode. A checkbox for 'I've read and accept the terms & conditions*' is located below the payment options. At the bottom of the right column is a 'PROCEED TO PAYPAL' button.

Fig 5.2.5 Shipment Dashboard

5.3 Backend

The backend can be used by the admin. Registration details including name and email and data regarding donations are stored in a database. Verification for successful registration is provided in the code built on Nodejs. The database used for this purpose is Mongo dB. It can store very large numbers of records efficiently (they take up little space). It is very quick and easy to find information. It is easy to add new data and edit or delete old data. This is deployed using the XAMPP platform. Any different developer or tester can easily test without losing time to set up and configure it. Fig 5.3.1 and fig 5.3.2 shows the backend.

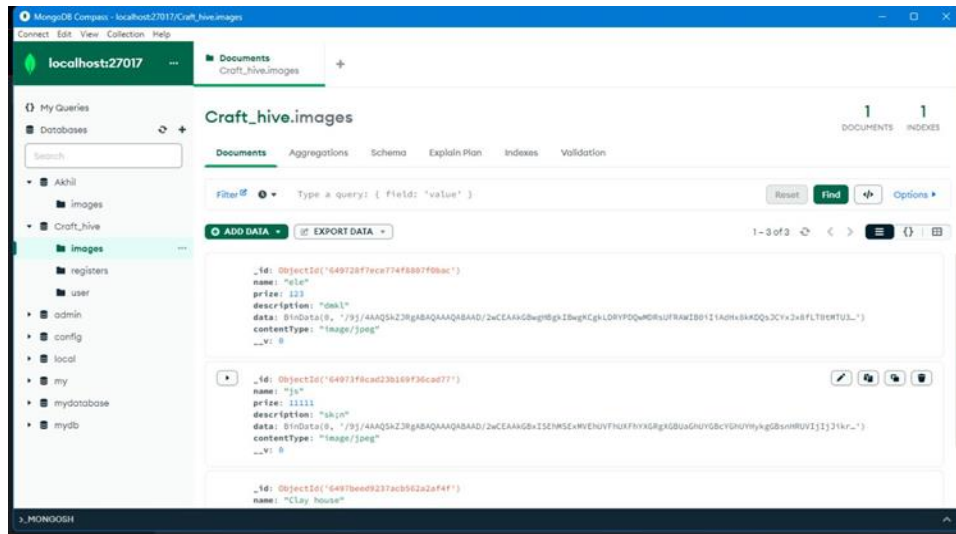


Fig 5.3.1 Back End

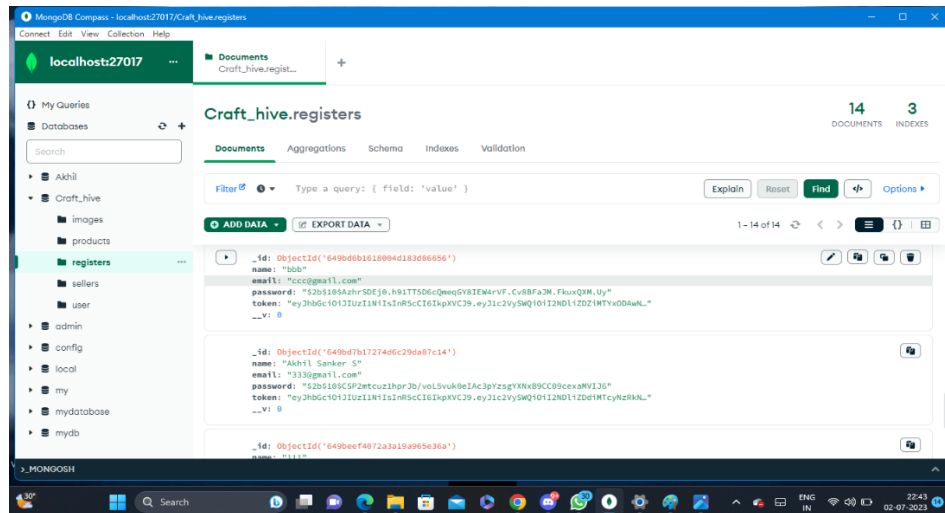


Fig 5.3.2 Back End

CHAPTER 6

TESTING

6.1 Various Testing Methods

6.1.1 Unit Testing

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. This testing methodology is done during the development process by the software developers and sometimes QA staff. The main objective of unit testing is to isolate written code to test and determine if it works as intended.

6.1.2 Integration Testing

Integration Testing is defined as a type of testing where software modules are integrated logically and tested as a group. A typical software project consists of multiple software modules, coded by different programmers. The purpose of this level of testing is to expose defects in the interaction between these software modules when they are integrated

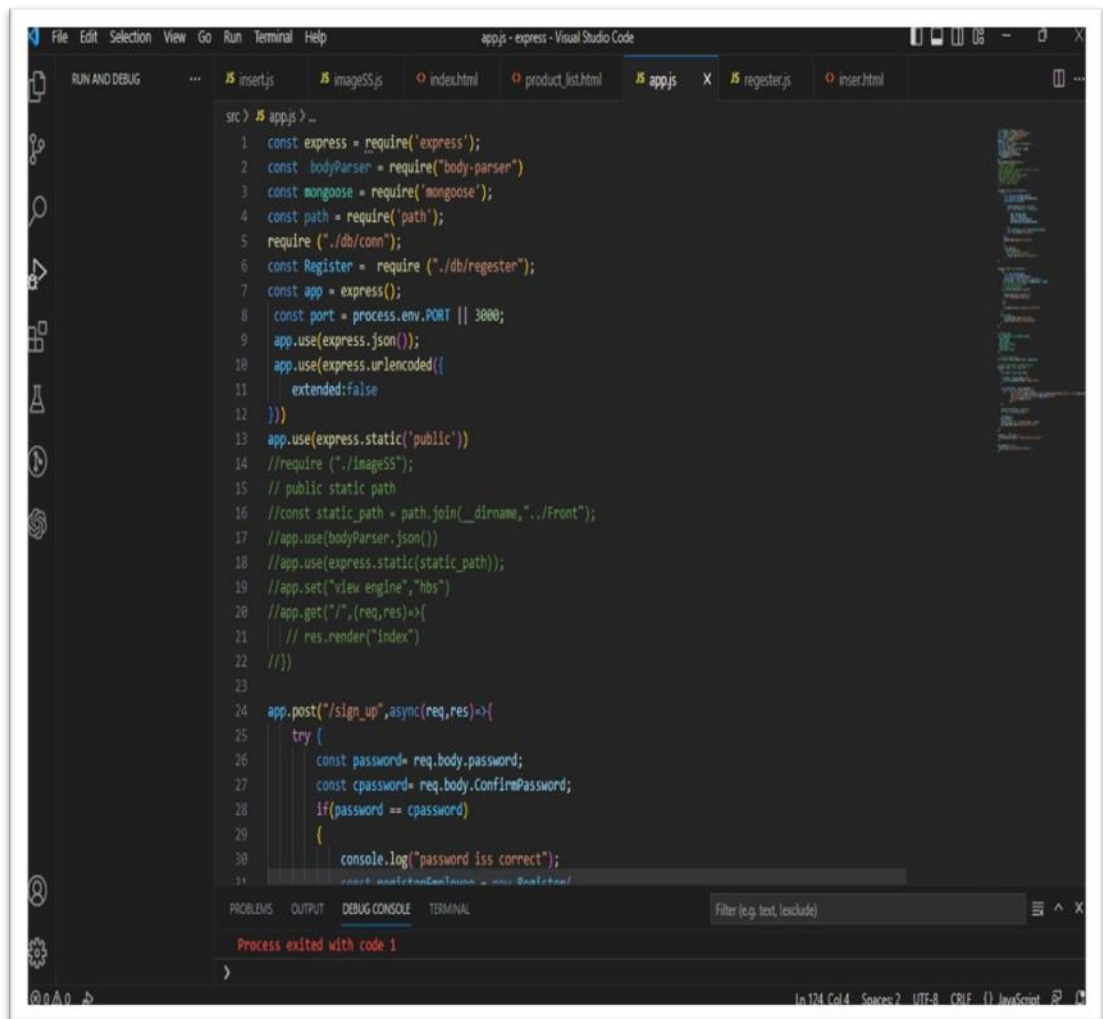
6.1.3 Functional Testing

Functional testing is a type of testing that seeks to establish whether each application feature works as per the software requirements. Each function is compared to the corresponding requirement to ascertain whether its output is consistent with the end user's expectations. The testing is done by providing sample inputs, capturing resulting outputs, and verifying that actual outputs are the same as expected outputs.

6.1.4 Load Testing

Load testing is a type of performance testing that simulates a real-world load on any software, application, or website. Without it, your application could fail miserably in real-world conditions. That's why we build tools like Retrace to help you monitor application performance and fix bugs before your code ever gets to production. Load testing examines how the system behaves during normal and high loads and determines if a system, piece of

software, or computing device can handle high loads given a high demand of end-users. This tool is typically applied when a software development project nears completion.



The image shows a screenshot of the Visual Studio Code editor with a project named 'appjs - express'. The 'appjs' file is open, displaying the following JavaScript code:

```
src> JS appjs > ...
1  const express = require('express');
2  const bodyParser = require('body-parser');
3  const mongoose = require('mongoose');
4  const path = require('path');
5  require('./db/conn');
6  const Register = require('./db/register');
7  const app = express();
8  const port = process.env.PORT || 3000;
9  app.use(express.json());
10 app.use(express.urlencoded({
11   extended: false
12 }));
13 app.use(express.static('public'))
14 //require('./imageSS.js');
15 // public static path
16 //const static_path = path.join(__dirname, '../Front');
17 //app.use(bodyParser.json());
18 //app.use(express.static(static_path));
19 //app.set('view engine', 'hbs')
20 //app.get('/', (req, res) => {
21   // res.render('index')
22 //})
23
24 app.post('/sign_up', async (req, res) => {
25   try {
26     const password = req.body.password;
27     const cpassword = req.body.confirmPassword;
28     if (password === cpassword) {
29       console.log('password is correct');
30     }
31   } catch (error) {
32     console.log(error);
33   }
34 }
```

The bottom of the editor shows the 'DEBUG CONSOLE' tab with the message 'Process exited with code 1'. The status bar at the bottom indicates 'Ln 174, Col 4, Source 2, UTF-8, CRLF, (1) JavaScript'.

Fig 6.1 A

```
1 <!doctype html>
2 <html lang="xxx">
3
4 <head>
5   <!-- Required meta tags -->
6   <meta charset="utf-8">
7   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
8   <title>CraftHive</title>
9   <link rel="icon" href="img/favicon.png">
10  <!-- Bootstrap CSS -->
11  <link rel="stylesheet" href="css/bootstrap.min.css">
12  <!-- animate CSS -->
13  <link rel="stylesheet" href="css/animate.css">
14  <!-- owl carousel CSS -->
15  <link rel="stylesheet" href="css/owl.carousel.min.css">
16  <!-- font awesome CSS -->
17  <link rel="stylesheet" href="css/all.css">
18  <!-- flatIcon CSS -->
19  <link rel="stylesheet" href="css/flatIcon.css">
20  <link rel="stylesheet" href="css/themify-icons.css">
21  <!-- font awesome CSS -->
22  <link rel="stylesheet" href="css/magnific-popup.css">
23  <!-- swiper CSS -->
24  <link rel="stylesheet" href="css/slick.css">
25  <!-- style CSS -->
26  <link rel="stylesheet" href="css/style.css">
27 </head>
28
29 <body>
30   <!--::header part start::-->
31   <div class="main menu item">
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Filter (e.g. text, include)

Process exited with code 1

Ln 191, Col 34 - Spaces: 4 - UTF-8 - LF - HTML

Fig 6.1 B

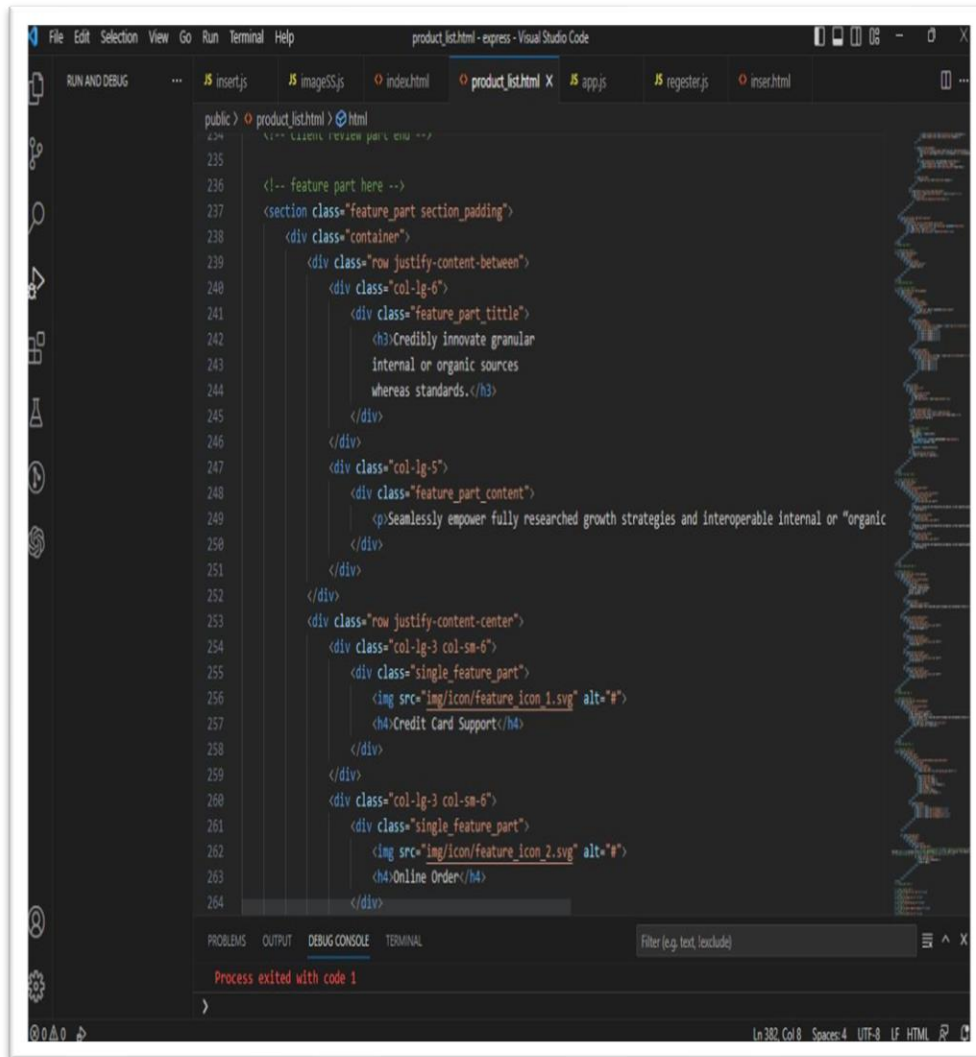
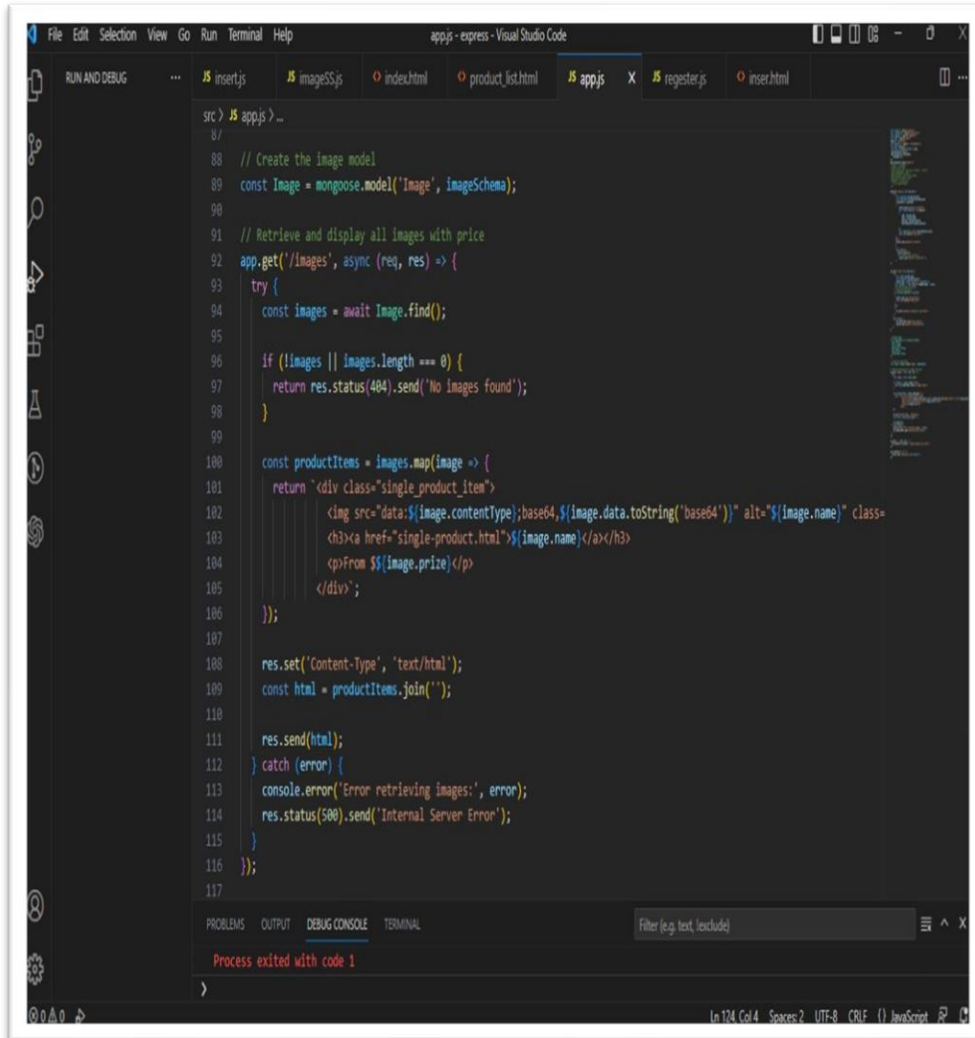


Fig 6.1 C



```
src > JS app.js > ...
87
88 // Create the image model
89 const Image = mongoose.model('Image', imageSchema);
90
91 // Retrieve and display all images with price
92 app.get('/images', async (req, res) => {
93   try {
94     const images = await Image.find();
95
96     if (!images || images.length === 0) {
97       return res.status(404).send('No images found');
98     }
99
100     const productItems = images.map(image => {
101       return `<div class="single_product_item">
102         <a href="single-product.html">${image.name}</a></h3>
104         <p>from $${image.price}</p>
105       </div>`;
106     });
107
108     res.set('Content-Type', 'text/html');
109     const html = productItems.join('');
110
111     res.send(html);
112   } catch (error) {
113     console.error('Error retrieving images:', error);
114     res.status(500).send('Internal Server Error');
115   }
116 });
117
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Filter (e.g. text, include)

Process exited with code 1

Ln 124 Col 4 Spaces 2 UTF-8 CR LF JavaScript

Fig 6.1 D

CONCLUSION

In conclusion, the development and implementation of a multivendor craft product selling ecommerce website have proven to be a transformative project. The website has successfully connected talented crafters with a wider customer base, offering a centralized platform for them to showcase and sell their unique creations. By expanding market reach, diversifying product offerings, empowering small-scale producers, and improving the customer experience, the website has created a vibrant and supportive marketplace. It promotes local and sustainable practices, fosters community building, and presents opportunities for collaboration and growth. While challenges and opportunities exist, the multivendor craft product selling ecommerce website has demonstrated its value in facilitating creative expression, economic growth, and the promotion of handmade crafts.

REFERENCES

- [1] Razorpay: [online],Available:“<https://razorpay.com/>”. [Accessed Jun 8, 2023].
- [2] About Shopify Website: [online],Available: “<https://www.shopify.com/>”.
[Accessed Mar 31, 2023].
- [3] MongoDB: [online],Available: “<https://www.mongodb.com/>”.
[Accessed Apr 28, 2023].