

ONLINE MARKETPLACE

CS321: Project 1

By

Akshat Kumar

21010113

Under the guidance of

Dr. Rajkumari Bidyalakshmi



Department of Computer Science Engineering,
Indian Institute of Information Technology, Manipur
November, 2023



Department of Computer Science & Engineering
Indian Institute of Information Technology Manipur

Dr. Rajkumari Bidyalakshmi Devi

Email: bidyalakshmi@iiitmanipur.ac.in

Assistant Professor

Certificate

This is to certify that the project report entitled **ONLINE MARKETPLACE** submitted to Department of Computer Science Engineering, Indian Institute of Information Technology Senapati, Manipur in partial fulfillment for the award of the degree of Bachelor of Technology in Computer Science & Engineering is a record of Bonafide work carried out by **Akshat Kumar** bearing Roll Number **21010113**.

Signature of Supervisor

(Dr. Rajkumari Bidyalakshmi Devi)

DECLARATION

The work embodied in the present report entitled “Online Marketplace” has been carried out in the Computer Science department. The work reported herein is original and does not form part of any other report or dissertation on the basis of which a degree or award was conferred on an earlier occasion or to any other student. I understand the Institute’s policy on plagiarism and declare that the report and publications are my own work, except where specifically acknowledged and has not been copied from other sources or been previously submitted for award or assessment.

Date:

(Signature)

Akshat Kumar

21010113

Department of Computer Science & Engineering

IIIT Senapati, Manipur

ABSTRACT

This project is a digital platform that facilitates the buying and selling of goods and services over the internet. It serves as a virtual space where buyers and sellers can buy and sell their products . Online marketplaces can cover a wide range of products and services, including retail goods, handmade crafts, digital products, freelance services, and more. Key features of online marketplaces typically include user profiles for buyers and sellers, product listings with detailed descriptions and images, a secure payment system, and a feedback or rating system to build trust among participants. Online marketplaces have become increasingly popular due to their convenience, accessibility, and the ability to reach a global audience. They provide opportunities for small businesses, entrepreneurs, and individuals to establish an online platform to reach far places. However, they also face challenges such as ensuring the security of transactions, managing product quality, and addressing issues related to trust and customer satisfaction. The success of an online marketplace often depends on factors like user experience, effective marketing, and the ability to foster a trustworthy community of buyers and sellers.

CONTENTS

1	INTRODUCTION	ix
1.1	AIM AND BACKGROUND	ix
1.2	PROBLEM STATEMENT	x
1.3	EXISTING SYSTEMS AND BRIEF MOTIVATION:	xi
1.4	ROAD-MAP:	xi
1.5	TECHNOLOGY USED:	xiii
2	LITERATURE REVIEW	xv
2.1	E-commerce Platform Analysis:	xv
2.2	Backend Development with Guidance:	xvi
2.3	QR Code Generation for Enhanced Security:	xvii
2.4	Data Visualization with Chart.js:	xviii
3	DESIGN AND IMPLEMENTATION	xix
3.1	Part 1: Creation of the Database:	xix
3.2	Part 2:-Interconnection between the Tables:	xxii
3.2.1	Users Table:	xxii
3.2.2	Userroles Table::	xxii
3.2.3	Product Table:	xxiii
3.2.4	Orders Table:	xxiii
3.2.5	Cart Table:	xxiii
3.2.6	Addresses Table:	xxiv
3.2.7	Returns Table:	xxiv
3.3	Part 3:-Functionality OF PHP EACH SESSIONS/PAGE:	xxvi

3.3.1	LOGIN AND REGISTRATION PAGE	xxvi
3.3.2	Userroles page	xxviii
3.3.3	Buyer page	xxxi
3.3.4	Cart-page	xxxii
3.3.5	Address-page	xxxv
3.3.6	Transcation-page	xxxvi
3.3.7	Profile-page-for-buyer	xxxviii
3.3.8	Seller-dashboard	xxxix
4	DIAGRAMS	xliii
4.1	ER DIAGRAM	xliii
4.2	GHANTT CART	xliii
4.3	USE CASE	xliii
4.4	BUYER ACTIVITY	xliii
4.5	SELLER ACTIVITY	xliii
5	RESULTS AND DISCUSSION	xlix
5.1	Homepage	l
5.2	Login	li
5.3	Registration	lii
5.4	User-role-selection	liii
5.5	Shopping-page	liv
5.6	Cart-page	lv
5.7	Select-address	lvi
5.8	Transcation	lvii
5.9	Buyers-profile-page	lviii
5.10	Seller-dashboard/profile-page	lix
5.11	Product-submission-page	lx
5.12	Discussion	lxi

6	CONCLUSION AND FUTURE SCOPE	lxii
6.1	Conclusion	lxii
6.2	Future scope	lxii

LIST OF FIGURES

3.1	ER DIAGRAM	xxi
3.2	login	xxix
3.3	registration	xxix
3.4	userroles	xxx
3.5	buyer page	xxxii
3.6	cart page	xxxiv
3.7	address	xxxvi
4.1	ER DIAGRAM	xliv
4.2	ghantt chart	xlv
4.3	use case	xlvi
4.4	buyer activity	xlvii
4.5	seller activity	xlviii

Chapter 1

INTRODUCTION

1.1 AIM AND BACKGROUND

The online marketplace aims to create a user-friendly and inclusive platform where people from all over the world can easily buy and sell products. The main goal is to make global transactions simple and to give businesses, especially small ones, the chance to reach a wider audience. I want to make it easy for users by providing a website that's easy to navigate and understand. Using the latest technology, we plan to offer features like personalized suggestions and an effective search system. Security is a top priority, so I am making sure to protect users' information and verify the sellers on our platform. I am also focusing on offering a variety of products so that users can find unique and interesting items. The background of the online marketplace comes from noticing changes in how people shop, the growing impact of technology on traditional stores, and the general trend toward digital solutions. This online marketplace is our response to the shift in how people buy things and our effort to create a platform that meets the needs of modern consumers. We aim to make commerce more accessible, especially for smaller businesses, and to adapt to the way people want to shop in today's digital age.

Our commitment to user ease is reflected in the design of our website, ensuring it is not only straightforward to navigate but also user-friendly, catering to individuals with varying levels of digital literacy. Leveraging cutting-edge technology, I am dedicated to introducing features that enhance the user experience, including personalized suggestions based on individual preferences and an effective search system to streamline product discovery.

Ensuring the security of our users is a paramount concern, and we are implementing stringent measures to safeguard personal information while also rigorously verifying sellers on our platform. This commitment to security is fundamental to building a trustworthy and reliable online marketplace environment

1.2 PROBLEM STATEMENT

Currently existing marketplaces are fragmented, inefficient, and opaque. They are often difficult to use for both buyers and sellers, and they can be expensive to participate in. This can lead to a number of problems, including:

- High costs for buyers and sellers: Buyers and sellers often have to pay high fees to participate in marketplaces, and they may also have to pay additional fees for services such as shipping and payment processing.
- Limited selection and competition: Many marketplaces offer a limited selection of products and services, and there is often little competition among sellers. This can lead to higher prices for buyers and lower profits for sellers.
- Lack of trust and transparency: It can be difficult for buyers to trust sellers on marketplaces, and sellers may be reluctant to disclose information about their products and services. This can lead to fraud and other problems.
- Inefficient operations: Many marketplaces are inefficient, with slow order processing and delivery times. This can frustrate buyers and lead to lost sales for sellers.

Solution A new marketplace platform is needed that addresses the problems of existing marketplaces. This platform should be:

- Cost-effective:** Buyers and sellers should be able to participate in the marketplace at a low cost.
- Efficient:** The platform should have efficient order processing and delivery mechanisms.
- Transparent:** The platform should provide buyers and sellers with all the information they need to make informed decisions.
- Trustworthy:** The platform should have mechanisms in place to protect buyers and sellers from fraud.

Benefits A new marketplace platform that addresses the problems of existing marketplaces would have a number of benefits, including:

- Lower costs for buyers and sellers:** Buyers and sellers would save money on fees and other costs.
- More selection and competition:** Buyers would have a wider selection of products and services to choose from, and sellers would face more competition.
- Increased trust and transparency:**

Buyers would be more likely to trust sellers, and sellers would be more likely to disclose information about their products and services. More efficient operations: Buyers would experience faster order processing and delivery times. Overall, a new marketplace platform that addresses the problems of existing marketplaces would make it easier and more efficient for buyers and sellers to connect and transact. This would lead to lower costs, more selection, and a better overall experience for everyone involved.

1.3 EXISTING SYSTEMS AND BRIEF MOTIVATION:

The motivation behind the development of an online marketplace emerges from a critical examination of existing systems and the challenges inherent in traditional commerce models. Conventional retail often confines businesses to local markets, operating within specific hours, limiting accessibility for consumers. The online marketplace seeks to break down these barriers, providing a global platform accessible 24/7, thus allowing businesses to reach a diverse audience beyond geographic constraints. Furthermore, the initiative is fueled by a commitment to empower small and medium enterprises (SMEs), offering them a stage to compete globally and fostering competition against larger counterparts. The focus on enhancing user experience through personalized recommendations, efficient search systems, and a user-friendly interface aims to cater to the evolving preferences of modern consumers. Leveraging the latest technologies, the online marketplace integrates features like secure payment gateways. In essence, the development of the online marketplace is a proactive response to challenges in existing systems, aiming to create a more accessible, inclusive, and technologically advanced commerce environment aligned with the expectations of a contemporary, global consumer base.

1.4 ROAD-MAP:

In dissecting the challenges of traditional commerce, our project uncovers a multifaceted problem landscape that has long hindered the seamless exchange of goods and services. Foremost among these challenges are the geographical restrictions that brick-and-mortar establishments inherently impose. Traditional commerce is confined within specific physi-

cal locations, limiting the market reach of businesses and restricting their ability to engage with a global audience. This geographical constraint not only stifles the growth potential of enterprises but also overlooks the evolving consumer trend of seeking products and services beyond local boundaries.

Operational complexities further compound the challenges faced by traditional commerce models. The intricate logistics involved in managing inventory, coordinating supply chains, and orchestrating transactions within the constraints of a physical storefront contribute to inefficiencies. These complexities not only elevate operational costs but also introduce potential bottlenecks that impede the smooth flow of commerce. This issue is particularly pronounced for small and medium enterprises (SMEs), which may lack the resources to navigate these operational intricacies, further limiting their competitiveness.

Scalability issues pose yet another significant challenge within the traditional commerce paradigm. As businesses aim to expand and diversify their offerings, the conventional retail model struggles to accommodate this growth seamlessly. Expanding product lines, reaching new markets, and accommodating increased transaction volumes become arduous tasks, often requiring substantial investments in infrastructure and resources. This lack of scalability inhibits businesses from fully realizing their market potential, hindering agility and responsiveness to evolving consumer demands.

By articulating these inherent problems within traditional commerce, our project recognizes the pressing need for innovative solutions. The online marketplace i propose is poised to redefine the very nature of commerce by addressing these challenges head-on. It is an ambitious response to geographical limitations, offering a digital platform that knows no bounds. It is a strategic solution to operational complexities, streamlining processes and reducing inefficiencies. It is a scalable and adaptable model that empowers businesses to grow organically, fostering an environment where enterprises of all sizes can not only survive but thrive in the ever-evolving landscape of global commerce. Through this acknowledgment of challenges, our project sets the stage for a transformative approach that paves the way for a more accessible, efficient, and globally connected marketplace.

1.5 TECHNOLOGY USED:

Creating a dynamic and feature-rich online marketplace requires a thoughtful selection of technologies to ensure a seamless user experience and robust functionality. The foundation of this project rests on a carefully curated technology stack that harmoniously integrates HTML, CSS, PHP, MySQL, JavaScript, Chart.js, and qrcode.min.js. Each of these elements plays a crucial role in shaping the frontend aesthetics, server-side connectivity, database management, interactivity, and data visualization, as well as ensuring secure and efficient payment transactions.

- **HTML for Frontend Structure:** HTML (HyperText Markup Language) serves as the backbone of the project, providing the necessary structure for the frontend. It defines the layout, content hierarchy, and user interface elements that users interact with. Through the use of HTML, the online marketplace achieves a clear and organized presentation of information, facilitating user navigation and engagement.
- **CSS for Styling and Presentation:** CSS (Cascading Style Sheets) is employed to enhance the visual appeal of the online marketplace. It is responsible for styling HTML elements, ensuring a cohesive and aesthetically pleasing design. By leveraging CSS, the project attains a professional and responsive layout, creating a visually engaging environment that enhances the overall user experience.
- **PHP for Server-Side Connection:** PHP (Hypertext Preprocessor) functions as the server-side scripting language, establishing a connection between the frontend and backend components. It is instrumental in handling dynamic content, user authentication, and communication with the MySQL database. PHP ensures that the online marketplace operates smoothly, securely processing user requests and delivering dynamic content in real-time.
- **MySQL for Database Creation and Management:** MySQL, a powerful relational database management system, is chosen to store and manage the vast array of data associated with the online marketplace. It handles user information, prod-

uct details, transaction records, and more. The structured and efficient nature of MySQL ensures reliable data storage, retrieval, and management, contributing to the seamless functioning of the platform.

- **Chart.js for Data Visualization:** Data visualization is paramount for sellers to monitor their performance effectively. Chart.js is employed to create visually appealing and informative graphs and charts. Sellers can gain insights into sales trends, product popularity, and revenue metrics, enhancing their decision-making process through intuitive data visualization.
- **qrcode.min.js Library for Secure Payment Transactions:** Ensuring secure payment transactions is of utmost importance in an online marketplace. The integration of qrcode.min.js facilitates the creation of secure QR codes, providing a reliable method for users to initiate and confirm transactions. This enhances the overall security of the payment process, contributing to the trustworthiness of the platform.

Chapter 2

LITERATURE REVIEW

2.1 E-COMMERCE PLATFORM ANALYSIS:

- Looked into the user interfaces of Flipkart, Amazon, and Myntra to understand how they optimize the overall shopping experience.
- Examined the navigation flow, product display, and checkout processes to identify user-centric design elements.
- Conducted a feature-by-feature analysis to comprehend the diverse functionalities offered by these platforms
- Analyzed the end-to-end transaction processes, focusing on the efficiency of the checkout process and the security measures implemented during payment.
- Investigated how social elements, such as product sharing, customer reviews, and social media integration, contribute to the overall user experience.

2.2 BACKEND DEVELOPMENT WITH GUIDANCE:

- Incorporated insights and best practices from tutorials by Dani Krossing and Code with Harry, industry-recognized experts in web development.
- Leveraged their guidance to ensure adherence to industry standards, coding best practices, and effective problem-solving approaches
- Ensured that theoretical knowledge was translated into actionable solutions within the context of our project.
- Learned and implemented efficient coding techniques, optimizing PHP code for performance and readability.
- Implemented strategies to ensure the backend infrastructure can scale seamlessly with growing user demands..
- Conducted performance profiling and optimization of PHP code to ensure efficient resource utilization.
- implemented secure coding practices, including input validation, data sanitization.

2.3 QR CODE GENERATION FOR ENHANCED SECURITY:

- Chose qrcode.min.js due to its lightweight nature, enabling quick and efficient QR code generation without imposing unnecessary overhead on the website.
- The minimized version ensures fast loading times and a smooth user experience.
- Integrated qrcode.min.js seamlessly into the frontend of the online marketplace.
- Utilized the simplicity and ease of use provided by the library to generate QR codes dynamically based on transaction-related data.
- Enabled users to engage in secure transactions by swiftly generating QR codes for payment or order verification.
- Streamlined the user journey by providing a visual and intuitive means of processing transactions, reducing friction in the payment process..
- Utilized qrcode.min.js to encode sensitive transaction data securely within the QR code.

2.4 DATA VISUALIZATION WITH CHART.JS:

- Chart.js for its versatility in supporting various chart types, including bar charts, line charts, pie charts, and more.
- Adapted different chart types based on the nature of the data to provide a comprehensive and visually appealing representation.
- Integrated Chart.js with real-time data sources to enable live updates on the seller dashboard.
- Enabled sellers to gain detailed insights by interacting with data points on the charts, fostering a more immersive experience.
- Ensured that charts dynamically reflect changes in data, providing sellers with up-to-the-minute analytics for prompt decision-making..
- Designed the integration to render charts dynamically based on the evolving dataset, avoiding the need for manual chart updates
- Ensured that the charts are not only informative but also aesthetically aligned with the overall visual theme of the dashboard..

Chapter 3

DESIGN AND IMPLEMENTATION

3.1 PART 1: CREATION OF THE DATABASE:

In this important step, I've set up the basic structure for our online marketplace. I created the Marketplace database using MySQL, a powerful system that helps organize and store all the important information for our platform. Think of it like the backbone of our marketplace—it's where we keep track of user profiles, product details, and all the transactions that happen. Using MySQL, which is like a super organized filing system, I carefully planned out how all the pieces of information fit together. This includes things like what details we store about each product, how we keep track of users, and how these different parts are related. This database is more than just storage; it's the foundation for features like making sure users are who they say they are, helping sellers manage their products, and ensuring transactions go smoothly. By making thoughtful choices about how we organize the data, we're setting things up to work well as our marketplace grows. So, creating this Marketplace database with MySQL is a big deal—it's like building a solid base that will support our entire online marketplace and make sure everything runs smoothly for both sellers and buyers.

- users Table : this table contains userid , firstname,lastname ,username ,email,password, PhoneNo, profilepicture.
- userroles: this table contains userroleid ,userd(foreignkey), role .

- product Table: this table contains productid , title ,description ,price , quantity , image , category , userid(foreignkey).
- cart: this table contains cart-id ,user-id(foreign-key),quantity.
- orders Table: this table contains its id ,user, order time, paymentdetails, address-id , products, quantity
- addresses Table: this table contains its id ,user-id ,address line 1 , address line2 , city, state, pincode , country
- paymentdetails Table: this table contains its id ,user-id ,address-id , payment-process-time
- return Table: this table contains its id ,user-id ,address-id , payment-process-time

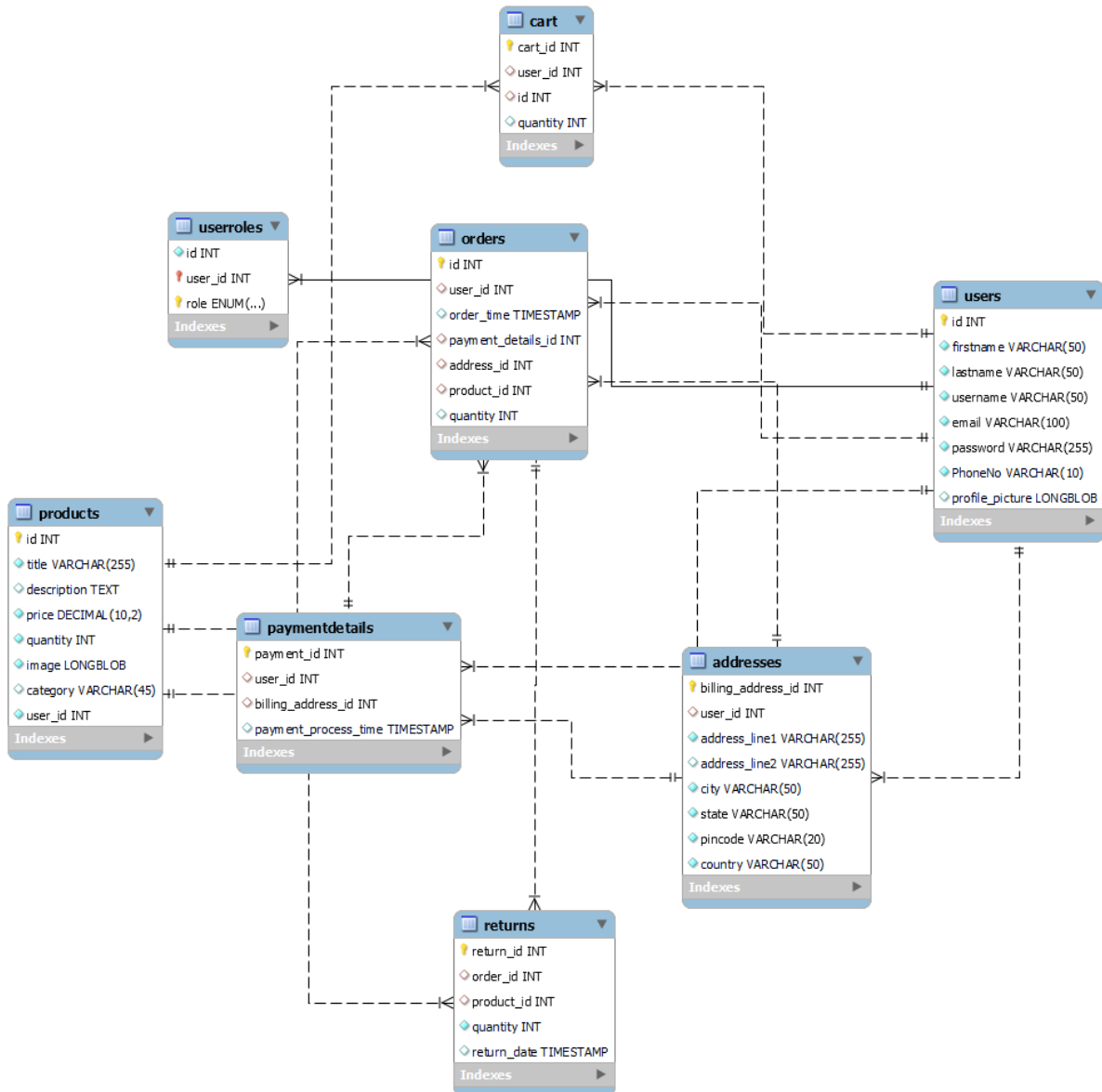


Figure 3.1: ER DIAGRAM

3.2 PART 2:-INTERCONNECTION BETWEEN THE TABLES:

3.2.1 *USERS TABLE:*

- Primary Key: The "id" field serves as the primary key for the users table, ensuring each user has a unique identifier.
- Foreign Key Relationships:
- userroles Table: The "user-id" in the userroles table refers to the "id" in the users table. This establishes a connection between users and their assigned roles, allowing for the identification of whether a user is a buyer or seller.
- Products Table: The "user-id" in the products table is a foreign key pointing to the "id" in the users table. This association links each product to the user who created or manages it.

3.2.2 *USERROLES TABLE::*

- Primary Key: The "id" field serves as the primary key, ensuring each user role entry has a unique identifier.
- Foreign Key Relationships:
- Users Table: The "userid" in the userroles table is a foreign key referencing the "id" in the users table. This establishes a connection between user roles and specific users.

3.2.3 *PRODUCT TABLE:*

- Primary Key: The "id" field is the primary key for the products table, providing a unique identifier for each product.
- Foreign Key Relationship:
- Table: The "userid" in the products table is a foreign key pointing to the "id" in the users table. This relationship indicates the user associated with each product.

3.2.4 *ORDERS TABLE:*

- Primary Key: The "id" field serves as the primary key for the orders table, ensuring each order has a unique identifier.
- Foreign Key Relationships:
- Users Table: The "userid" in the orders table is a foreign key connecting to the "id" in the users table. This associates each order with a specific user.
- Products Table: The "productid" in the orders table is a foreign key pointing to the "id" in the products table, indicating which product is included in the order.
- Addresses Table: The "addressid" in the orders table is a foreign key referencing the "billingaddressid" in the addresses table. This links each order to a billing address.
- Payment Details: The "paymentdetailsid" likely refers to a payment details table. This establishes a relationship with payment information associated with each order.

3.2.5 *CART TABLE:*

- Primary Key: The "cartid" field serves as the primary key for the cart table, ensuring each cart has a unique identifier.
- Foreign Key Relationships:
- Users Table: The "userid" in the cart table is a foreign key pointing to the "id" in the user table, connecting

- Products Table: The "id" in the cart table likely refers to the "id" in the products table, specifying which product is in the cart.
- Addresses Table: The "addressid" in the orders table is a foreign key referencing the "billingaddressid" in the addresses table. This links each order to a billing address.
- Payment Details: The "paymentdetailsid" likely refers to a payment details table. This establishes a relationship with payment information associated with each order.

3.2.6 ADDRESSES TABLE:

- Primary Key: The "billingaddressid" field serves as the primary key for the addresses table, ensuring each address has a unique identifier.
- Foreign Key Relationships: Users Table: The "userid" in the addresses table is a foreign key pointing to the "id" in the users table. This associates each address with a specific user
- Products Table: The "id" in the cart table likely refers to the "id" in the products table, specifying which product is in the cart.
- Addresses Table: The "addressid" in the orders table is a foreign key referencing the "billingaddressid" in the addresses table. This links each order to a billing address.
- Payment Details: The "paymentdetailsid" likely refers to a payment details table. This establishes a relationship with payment information associated with each order.

3.2.7 RETURNS TABLE:

- Primary Key: The "returnid" field serves as the primary key for the returns table, ensuring each return has a unique identifier.
- Foreign Key Relationships:
- Orders Table: The "orderid" in the returns table is a foreign key pointing to the "id" in the orders table, indicating the order associated with each return.

- Products Table: The "productid" in the returns table is a foreign key pointing to the "id" in the products table, specifying which product is being returned.

3.3 PART 3:-FUNCTIONALITY OF PHP EACH SESSIONS/PAGE:

3.3.1 LOGIN AND REGISTRATION PAGE

In the page register.php, is designed to handle user registration for a website. Upon receiving a POST request (indicating form submission), it retrieves user registration data from the form fields, such as first name, last name, username, email, password, and phone number. It then prepares a SQL statement to insert this information into a MySQL database table named "users." The script uses prepared statements to prevent SQL injection vulnerabilities. If the database insertion is successful, it notifies the user through a JavaScript alert and redirects them to a login page. If the insertion fails, likely due to the existence of a user with the same first and last name in the database, it displays an alert indicating that the user already exists. The HTML section of the document contains a registration form with fields for the user's first name, last name, username, phone number, email, and password. The form is styled using custom CSS. Additionally, there's a navigation bar and a footer with social media icons. The website's design suggests a creative or artistic theme, given the brand name "doodle" and the use of artistic fonts. This script is part of a larger web application. It employs PHP for server-side processing, HTML for the user interface, and Bootstrap for styling and layout.

This PHP and HTML page serves as a login mechanism for a web application. The PHP script begins by starting a session and requiring a connection to the database. Upon receiving a POST request (presumably from a login form), it retrieves the entered username and password. It then prepares and executes a SQL SELECT statement to fetch user data (id, username, password) from the 'users' table based on the provided username. If a matching record is found, it compares the entered password with the one stored in the database. If the credentials match, the user is considered authenticated. The script sets a session variable 'userid' to the user's ID and redirects them to the 'userroles.php' page, indicating a successful login. If the credentials don't match, it displays an alert for an "Invalid Password" and redirects to the login page. If no matching username is

found in the 'users' table, it displays an "Invalid Username" alert and directs the user back to the login page. The connection to the database is then closed. The HTML part includes a navigation bar, a login form with fields for username and password, and styling using Bootstrap and custom CSS. The form includes a "Remember me" checkbox and a "Forgot password" link. Additionally, there's a link to redirect users to the registration page if they don't have an account. The page also incorporates social media icons in the footer. Overall, this page provides a functional and visually appealing login interface with error handling and session management.

3.3.2 USERROLES PAGE

This PHP and HTML page manages user role selection for a web application after they've logged in. The PHP script begins by checking if a user is authenticated, as determined by the presence of a 'userid' in the session. If authenticated, it proceeds to handle the form submission when the user selects a role and clicks the "Submit" button. The script retrieves the user ID from the session and the selected role from the form. It then sanitizes these inputs to prevent SQL injection by using the 'realescapestring' method. The script checks if a record with the same user ID and role already exists in the 'userroles' table using a SELECT query. If a match is found, an alert is displayed indicating the existence of the entry, and the user is redirected to the appropriate page based on their role (buyer or seller). If no match is found, a new record is inserted into the 'userroles' table with the user ID and selected role. The script then redirects the user to the corresponding page based on their role. If an error occurs during the SQL operations, an alert is displayed with the specific error message. The HTML section includes a navigation bar, a form allowing users to select their role from a dropdown menu, and a "Submit" button. The page has a visually appealing design using Bootstrap and custom CSS, with a responsive layout. The user is presented with a clean interface to choose their role, facilitating a seamless transition to the buyer or seller dashboard based on their selection.

login.php

```

5  if ($_SERVER["REQUEST_METHOD"] === "POST") {
6      $username = $_POST['username'];
7      $password = $_POST['password'];
8
9      $stmt = $conn->prepare("SELECT id, username, password FROM users WHERE username = ?");
10     $stmt->bind_param("s", $username);
11     $stmt->execute();
12     $stmt->store_result();
13
14     $stmt->bind_result($id, $dbUsername, $dbPassword);
15     $stmt->fetch();
16
17     if ($stmt->num_rows > 0) {
18         if($password != $dbPassword) {
19             echo '<script>alert("Invalid Password")</script>';
20             echo "<script>window.location.href = './login.php'</script>";
21
22             exit;
23         }
24         else
25         {
26             $_SESSION['user_id'] = $id;
27
28             echo '<script>alert("Login successful") </script>';
29             echo "<script>window.location.href = './user_roles.php'</script>";
30
31         }
32     }
33     else {
34         echo '<script>alert("Invalid Username")
35         window.location.assign("login.php")</script>';
36     }
37 }

```

Figure 3.2: login
registration.php

```

require_once('../connection/connection.php');

if ($_SERVER["REQUEST_METHOD"] === "POST") {

    $first_name = $_POST['first_name'];
    $last_name = $_POST['last_name'];
    $username = $_POST['username'];
    $email = $_POST['email'];

    $password = $_POST['password'];
    $phonenumber = $_POST['phonenumber'];

    $stmt = $conn->prepare("INSERT INTO users (firstname, lastname, username, email, password, PhoneNo) VALUES (?, ?, ?, ?, ?, ?)");
    $stmt->bind_param("ssssss", $first_name, $last_name, $username, $email, $password, $phonenumber);

    if ($stmt->execute()) {
        $user_id = $_SESSION['user_id'];
        echo "<script>alert('Registration For $first_name $last_name is Successful ')</script>";
        echo "<script>window.location.href = './login.php'</script>";
    } else {
        echo "<script>alert('Entry For $first_name $last_name is Already Exist In database')</script>";
    }

    $stmt->close();
}
$conn->close();

```

Figure 3.3: registration

```

if(isset($_SESSION['user_id'])){
if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $user_id=$_SESSION['user_id'];
    $role = $_POST['role'];

    $user_id = $conn->real_escape_string($user_id);
    $role = $conn->real_escape_string($role);

    $checkIfExistsSQL = "SELECT id FROM userroles WHERE user_id = '$user_id' AND role = '$role'";
    $result = $conn->query($checkIfExistsSQL);

    if ($result->num_rows > 0) {

        $row = $result->fetch_assoc();
        $id = $row["id"];
        echo "<script>alert('Entry already exists with id: $id');</script>";

        if ($role == 'buyer') {
            header("Location: ../buyer/buyer.php?UserID=" . $user_id . "&RoleID=" . $role);
            exit();
        } elseif ($role == 'seller') {
            header("Location: ../seller/dashboard.php?UserID=" . $user_id . "&RoleID=" . $role);
            exit();
        }
    } else {

        $insertSQL = "INSERT INTO userroles (user_id, role) VALUES ('$user_id', '$role')";
        if ($conn->query($insertSQL) === TRUE) {
            echo "<script>alert('New record inserted successfully');</script>";

            if ($role == 'buyer') {
                header("Location: ../buyer/buyer.php?UserID=" . $user_id . "&RoleID=" . $role);
                exit();
            } elseif ($role == 'seller') {
                header("Location: ../seller/dashboard.php?UserID=" . $user_id . "&RoleID=" . $role);
                exit();
            }
        } else {
            echo "<script>alert('Error: " . $conn->error . "');</script>";
        }
    }
}
}

```

Figure 3.4: userroles

3.3.3 BUYER PAGE

This PHP and HTML page represents a buyer dashboard in a web application. The PHP section begins by starting a session and retrieving the user ID from the session variable. It then queries the 'userroles' table to fetch the role associated with the user ID. Following this, a SELECT query is used to retrieve all products from the 'products' table where the user ID is not equal to the current user's ID. The retrieved products are stored in the all-product variable. Moving on to the HTML section, the page includes a navigation bar with links to home, shop, cart, and orders. The main content area features a product container where products from the database are dynamically displayed. For each product, a product card is generated, showing the product image, title, price, and interactive buttons for changing the quantity and adding the product to the cart. The script utilizes JavaScript to handle quantity changes and adding products to the cart. It maintains a quantities object to store the quantity for each product. The changeQuantity function allows users to increment or decrement the quantity of a specific product, and the addToCart function sends an AJAX request to the 'addtocart.php' script, passing the product ID, quantity, user ID, and role. The response from the server is then displayed in an alert, informing the user whether the product was successfully added to the cart. The page is designed with Bootstrap and custom CSS, featuring a responsive layout and a visually appealing image carousel. It also includes a footer with social media icons. Overall, the page provides a user-friendly interface for buyers to browse products and add them to their carts.

```

ver.php
<div class="product-container">
  <?php
  if ($all_product->num_rows > 0) {
    while ($row = $all_product->fetch_assoc()) {
      echo '<div class="product-card">
        <div class="image-card">';
      if ($row['image'] != NULL){
        echo '';
      } else {
        echo '';
      }
      echo '</div>
        <div class="captions">
          <h2 class="Product Name">' . $row['title'] . '</h2>
          <p class="price">' . $row['price'] . '</p>
        </div>
        <div class="cart-actions">
          <button class="quantity-button" id="decrease-' . $row['id'] . '" onclick="changeQuantity(' . $row['id'] . ', -1)"></button>
          <span id="quantity-' . $row['id'] . '">0</span>
          <button class="quantity-button" id="increase-' . $row['id'] . '" onclick="changeQuantity(' . $row['id'] . ', 1)"></button>
          <button class="add-to-cart-button" onclick="addToCart(' . $row['id'] . ')">Add to Cart</button>
        </div>
      </div>';
    }
  } else {
    echo "No products found in the database.";
  }
  ?>
  <script>
    var quantities = {};

    function changeQuantity(productId, change) {
      if (!quantities[productId]) {
        quantities[productId] = 0;
      }
      quantities[productId] += change;
      if (quantities[productId] < 0) {
        quantities[productId] = 0;
      }
      document.getElementById('quantity-' + productId).innerText = quantities[productId];
    }

    function addToCart(productId) {
      var quantity = quantities[productId] || 0;
      var userId = <?php echo $user_id; >;
    }
  </script>

```

Figure 3.5: buyer page

3.3.4 CART-PAGE

This PHP and HTML page represents a shopping cart functionality within a web application. The PHP section starts by establishing a connection to the database and retrieving the user's role and profile information based on their session. It also includes a function, `removecartitem()`, which handles the removal of items from the cart upon form submission. The HTML section comprises a navigation bar, user profile information, and a main content area displaying the shopping cart. The cart includes a table listing the selected products, their quantities, prices, and options to remove them. The total price, including tax and shipping, is calculated dynamically. Users can select items to remove and proceed to either purchase more items or go to the payment process by clicking corresponding buttons. The page is designed using css for a clean and responsive layout. It

features dynamic content, such as the user's profile picture and username, retrieved from the database. The shopping cart table is populated with product information fetched from the database, providing a comprehensive view of the user's selected items. Additionally, the page includes buttons for users to proceed to add an address, facilitating a seamless transition to the next step in the purchase process. The PHP script handles the logic for removing items from the cart and redirects users based on their actions. Overall, this page serves as a user-friendly interface for managing and reviewing items in the shopping cart, incorporating dynamic data from the database and allowing users to seamlessly proceed with their shopping experience.

```

session_start();
$hostname = "localhost";
$username = "root";
$password = "akshat12345";
$dbname = "project";

$conn = new mysqli($hostname, $username, $password, $dbname);
global $conn;
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$user_id = $_SESSION['user_id'];
$select_query = "select role from userroles where user_id =$user_id";
$result=mysqli_query($conn,$select_query);
$fetching = mysqli_fetch_assoc($result);
$role=$fetching['role'];

$user_query = "SELECT username, profile_picture FROM users WHERE id = $user_id";
$user_result = mysqli_query($conn, $user_query);
$user_data = mysqli_fetch_assoc($user_result);

$username = $user_data['username'];
$profile_picture = $user_data['profile_picture'];

function remove_cart_item() {
    global $conn;
    if (isset($_POST['remove_cart'])) {
        if (isset($_POST['removeitem']) && is_array($_POST['removeitem'])) {
            foreach ($_POST['removeitem'] as $remove_id) {
                $remove_id = (int) $remove_id;
                $deletequery = "DELETE FROM cart WHERE id = $remove_id";
                $run_delete = mysqli_query($conn, $deletequery);
                if ($run_delete) {
                    echo "<script>window.open('marora.php', '_self')</script>";
                }
            }
        }
    }
}

$remove_item = remove_cart_item();

```

Figure 3.6: cart page

3.3.5 ADDRESS-PAGE

This PHP and HTML code is designed for an e-commerce platform, specifically a page that handles the selection of billing addresses for transactions. The page begins by checking if a user is logged in through session management. If a user is logged in, their user ID is retrieved from the session, and essential user information, such as the username and profile picture, is fetched from the database. The user is presented with a form containing a list of saved addresses associated with their account. These addresses are retrieved from the "addresses" table in the database based on the user's ID. Each address is displayed with its details, including address lines, city, state, and pin code. Users can select one or more addresses by checking the corresponding checkboxes. Upon form submission, the PHP code processes the selected billing addresses. It checks if the billing address IDs are set and not empty. If valid addresses are selected, it iterates through each selected address, checking if the address is already associated with the user in the "paymentdetails" table. If not, the selected billing address is inserted into the "paymentdetails" table, indicating a successful payment process. Error handling is also implemented. If a selected address is already associated with the user, an alert is shown, preventing duplicate entries. Additionally, if no addresses are selected, another alert prompts the user to choose at least one address. The HTML section of the code structures the page layout. It includes css and Boxicons for styling and responsiveness. The user's profile picture and username are displayed in the header. The main content includes a form with a list of addresses and a submit button. The page is designed with a clean and user-friendly interface, making it easy for users to interact with and select their billing addresses.

```

$main_query = "SELECT * FROM addresses WHERE user_id = $user_id";
$resultant_query = mysqli_query($conn, $main_query);

if (isset($_POST['submit'])) {

    if (isset($_POST['billing_address_id']) && !empty($_POST['billing_address_id'])) {

        $billing_address_ids = $_POST['billing_address_id'];
        $billing_address_ids = array_map(function ($id) use ($conn) {
            return mysqli_real_escape_string($conn, $id);
        }, $billing_address_ids);

        foreach ($billing_address_ids as $billing_address_id) {
            $select_query_check = "SELECT * FROM `paymentdetails` WHERE user_id = $user_id AND billing_address_id = '$billing_address_id'";
            $check = mysqli_query($conn, $select_query_check);
            $result_mains = mysqli_fetch_assoc($check);

            if ($result_mains) {
                echo "<script>alert('For $user_id, address is already existed with id $billing_address_id')</script>";
                echo "<script>window.open('trans.php', '_self')</script>";
            } else {
                $insert_query = "INSERT INTO `paymentdetails`(`user_id`, `billing_address_id`, `payment_process_time`)
                    VALUES ('$user_id', '$billing_address_id', NOW())";
                $result_main = mysqli_query($conn, $insert_query);
                echo "<script>window.open('trans.php', '_self')</script>";

                if ($result_main) {
                    echo "<script>alert('For $user_id, address is successfully inserted with id $billing_address_id')</script>";
                    echo "<script>window.open('trans.php', '_self')</script>";
                } else {
                    echo "Error: " . mysqli_error($conn);
                }
            }
        }
    }
} else {

```

Figure 3.7: address

3.3.6 TRANSACTION-PAGE

This PHP and HTML code is designed for a payment completion page in an e-commerce application. It assumes that the user has completed their shopping, and their cart information is stored in the "cart" table. The page begins by checking if the user is logged in using session management. If the user is logged in and the form is submitted via POST request, the code retrieves relevant information from the "cart" and "paymentdetails" tables in the database. For each product in the user's cart, the code calculates the total amount considering the quantity and individual product prices. Additional charges, such as shipping fees, are added to the total amount. The seller's UPI ID, phone number, and the calculated total amount are displayed on the page. The code also generates a QR code for the payment details, assuming a UPI-based payment method. The user is presented with their payment information, including the UPI ID, phone number, and total amount. A QR code is dynamically generated using JavaScript, making it easy for users

```

session_start();
$hostname = "localhost";
$username = "root";
$password = "akshat12345";
$dbname = "project";

$conn = new mysqli($hostname, $username, $password, $dbname);
global $conn;
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

if (isset($_SESSION['user_id'])) {
    $user_id = $_SESSION['user_id'];
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $cartQuery = "SELECT * FROM cart WHERE user_id = $user_id";
        $cartResult = mysqli_query($conn, $cartQuery);

        $paymentQuery = "SELECT * FROM paymentdetails WHERE user_id = $user_id";
        $paymentResult = mysqli_query($conn, $paymentQuery);

        $rowPayment = mysqli_fetch_assoc($paymentResult);
        $billing_address_id = $rowPayment['billing_address_id'];
        $payment_details_id = $rowPayment['payment_id'];

        while ($rowCart = mysqli_fetch_assoc($cartResult)) {
            $product_id = $rowCart['id'];
            $quantity = $rowCart['quantity'];
            $timestamp = date("Y-m-d H:i:s");
            insertIntoOrders($user_id, $product_id, $quantity, $payment_details_id, $billing_address_id, $timestamp);

            $deleteCartQuery = "DELETE FROM cart WHERE user_id = $user_id ";
            mysqli_query($conn, $deleteCartQuery);
        }

        echo "<script>window.open('completion.php', '_self')</script>";
    }
}

function insertIntoOrders($user_id, $product_id, $quantity, $payment_details_id, $billing_address_id, $timestamp) {
    global $conn;
    $insertQuery = "INSERT INTO orders (user_id, product_id, quantity, payment_details_id, address_id, order_time)
        VALUES ('$user_id', '$product_id', '$quantity', '$payment_details_id', '$billing_address_id', '$timestamp')";

    mysqli_query($conn, $insertQuery);

    echo "<script>window.open('completion.php', '_self')</script>";
}

```

to scan and complete the payment. A form with a submit button is included on the page, suggesting that the user is expected to confirm the payment. Upon submission, the PHP code inserts the order details into the "orders" table, effectively completing the purchase. The user is then redirected to a completion page ("completion.php"). The page is styled using Bootstrap for a clean and responsive layout, providing a seamless user experience throughout the payment process.

3.3.7 PROFILE-PAGE-FOR-BUYER

The PHP section at the beginning includes the necessary database connection code and starts a session. It checks if a user is logged in by verifying the presence of the 'userid' in the session. If a user is logged in, it retrieves user information such as first name, last name, email, username, and profile picture from the 'users' table in the database using SQL queries. The retrieved user data is then utilized to dynamically generate the content of the HTML page. The page includes a navigation bar that features the website logo and navigation links. The user's account information, such as their username, is displayed in the upper part of the page. A profile picture, along with the user's email and full name, is presented in a structured layout. The main content of the page consists of various sections, each represented by HTML elements. There's a section for general user details like email and name. Additionally, there are sections for different aspects of the user's account, such as orders, returns, and the shopping cart. Placeholder sections like location, favorites, offers, coupons, shopping bags, and customer support are also included. The code focuses on creating a clean and straightforward user interface . It aims to provide a personalized and interactive experience for users managing their accounts on an e-commerce platform. The footer contains social media icons and a copyright notice, contributing to the overall structure of the page. This code serves as a foundation for a simple and customizable account dashboard in an e-commerce application.

3.3.8 SELLER-DASHBOARD

The PHP script, `getproductdata.php`, serves as a backend endpoint responsible for retrieving specific product data associated with the user currently logged in. The script begins by initiating a session, a crucial step for maintaining user-specific data. Following this, the necessary database connection is established through the inclusion of an external connection file. The core functionality involves constructing an SQL query that selects product titles and quantities from the `'products'` table based on the user's session ID. The script then executes this query, processes the obtained results into a PHP array named `'data'`, and subsequently encodes this array into JSON format using `'json-encode'`. In the event of a successful database interaction, the encoded product data is echoed back to the client. However, if an error occurs during the execution of the SQL query, a generic error message is echoed instead. Finally, to ensure proper resource management, the script closes the database connection. On the client side, the JavaScript file, `'productchart.js'`, plays a pivotal role in dynamically rendering a bar chart using the Chart.js library. First, an array of predefined colors is established to enhance the visual representation of the chart. The script then utilizes the Fetch API to asynchronously retrieve product data from the server by making a request to the aforementioned PHP endpoint. Upon successful data retrieval, the script processes the JSON-formatted data, extracting both product titles and quantities. Subsequently, a function named `'createChart'` is invoked, passing the extracted data and predefined colors as parameters. Within `'createChart'`, the Chart.js library is employed to generate a bar chart on an HTML canvas element with the ID `'myChart'`. This chart is configured using the extracted data, specified colors, and additional options, such as ensuring the y-axis begins at zero. The resulting chart provides an interactive and dynamic visualization of the user's product quantities. Importantly, the JavaScript script includes error handling, which logs any potential issues that may arise during the data-fetching process, contributing to improved robustness in handling unexpected situations.

7.php

```
session_start();
require_once('../connection/connection.php');
$user_id = $_SESSION['user_id'];

$sql = "SELECT * FROM project.users WHERE id = $user_id";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    $row = $result->fetch_assoc();
    $username = $row["username"];
    $profile_picture = $row["profile_picture"];
    $firstname = $row["firstname"];
    $lastname = $row["lastname"];
    $email = $row["email"];

} else {
    echo "No user found with the given user_id";
}
if (isset($_POST['submit'])) {
    echo "<script>window.open('formm.php', '_self')</script>";
}
|>
```



```

get_product_data.php
<?php
session_start();
require_once('../connection/connection.php');

$user_id = $_SESSION['user_id'];

$sql = "SELECT title, quantity FROM project.products WHERE user_id = $user_id";
$result = $conn->query($sql);

if ($result) {
    $data = array();
    while ($row = $result->fetch_assoc()) {
        $data[] = array(
            'title' => $row['title'],
            'quantity' => $row['quantity']
        );
    }

    echo json_encode($data);
} else {
    echo "Error: " . $conn->error;
}

$conn->close();
?>

```

```

const colors = [
  'rgba(255, 99, 132, 0.2)',
  'rgba(255, 159, 64, 0.2)',
  'rgba(255, 205, 86, 0.2)',
  'rgba(75, 192, 192, 0.2)',
  'rgba(54, 162, 235, 0.2)',
  'rgba(153, 102, 255, 0.2)',
  'rgba(201, 203, 207, 0.2)'
];

fetch('get_product_data.php')
  .then(response => response.json())
  .then(data => {

    const titles = data.map(item => item.title);
    const quantities = data.map(item => item.quantity);

    // Create the chart using the extracted data and predefined colors
    createChart(titles, quantities, colors);
  })
  .catch(error => console.error('Error fetching product data:', error));

// Function to create the Chart.js chart
function createChart(titles, quantities, colors) {
  const options = {
    scales: {
      y: {
        beginAtZero: true
      }
    }
  };

  const ctx = document.getElementById('myChart').getContext('2d');
  const myChart = new Chart(ctx, {
    type: 'bar',
    data: {
      labels: titles,
      datasets: [{
        label: 'bar',
        data: quantities,
        backgroundColor: colors,
        borderColor: 'rgba(75, 192, 192, 1)',
        borderWidth: 1,
      }]
    },
    options: options
  });
}

```

Chapter 4

DIAGRAMS

4.1 ER DIAGRAM

4.2 GHANTT CART

4.3 USE CASE

4.4 BUYER ACTIVITY

4.5 SELLER ACTIVITY

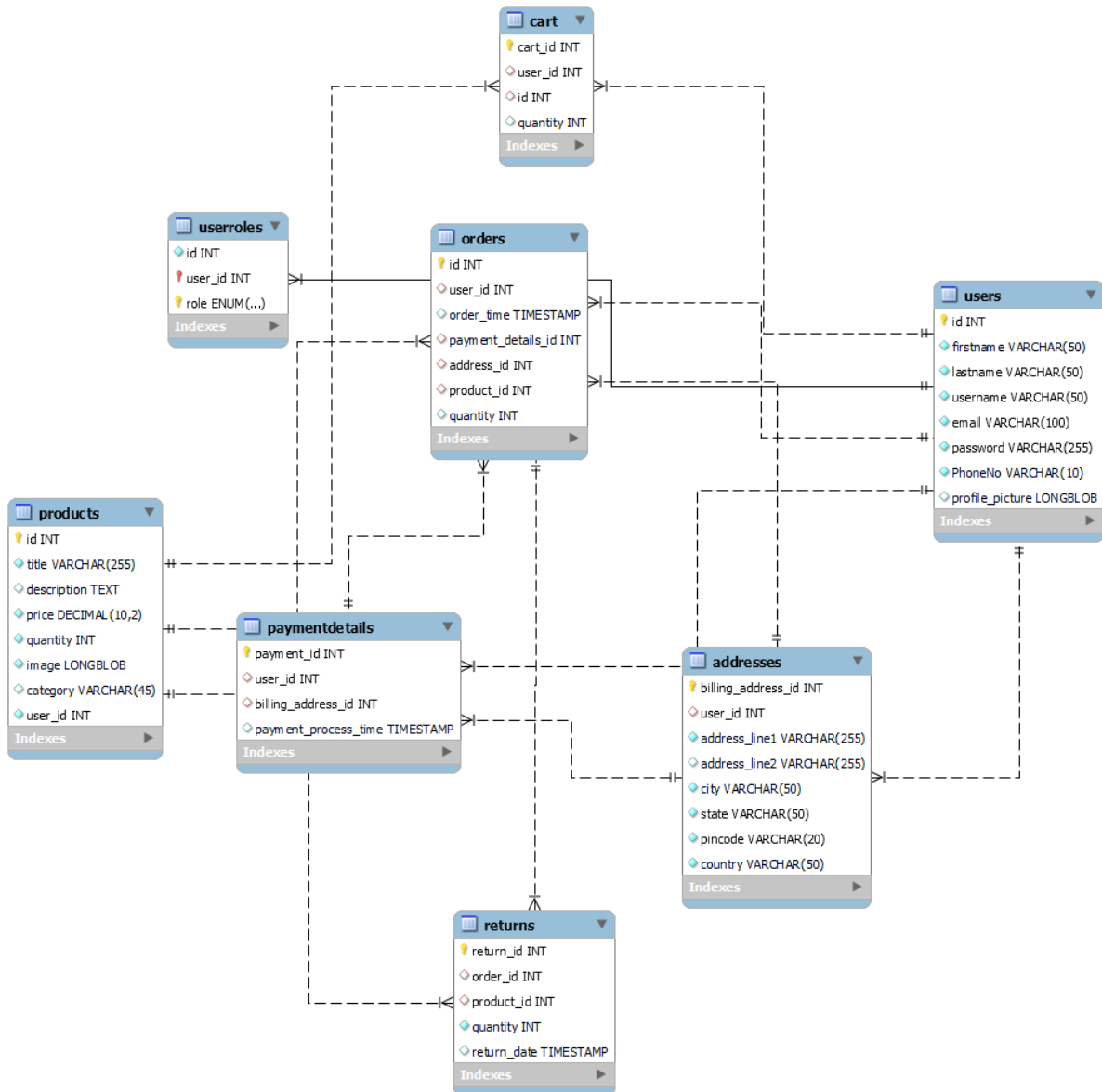


Figure 4.1: ER DIAGRAM

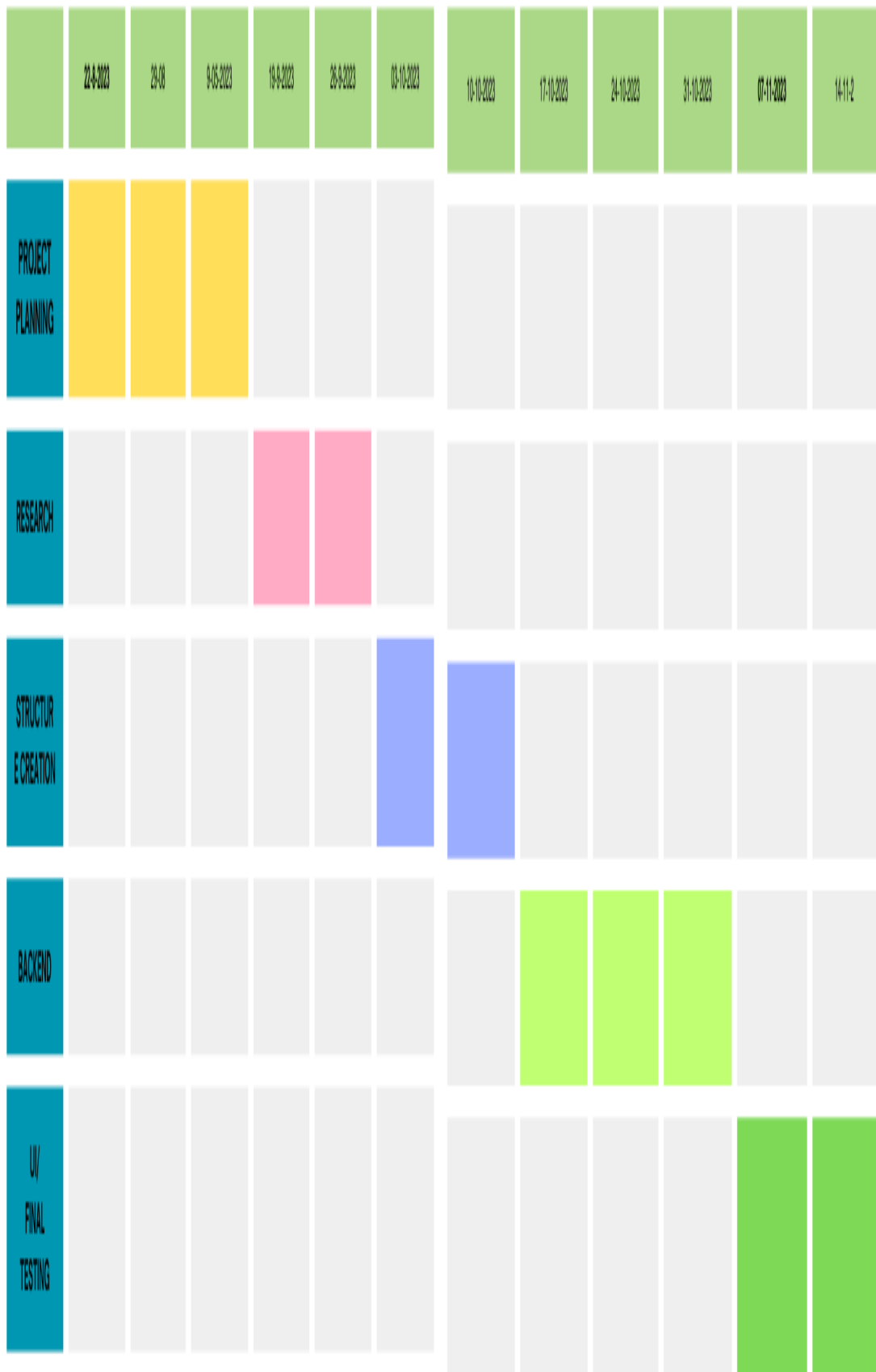


Figure 4.2: gantt chart

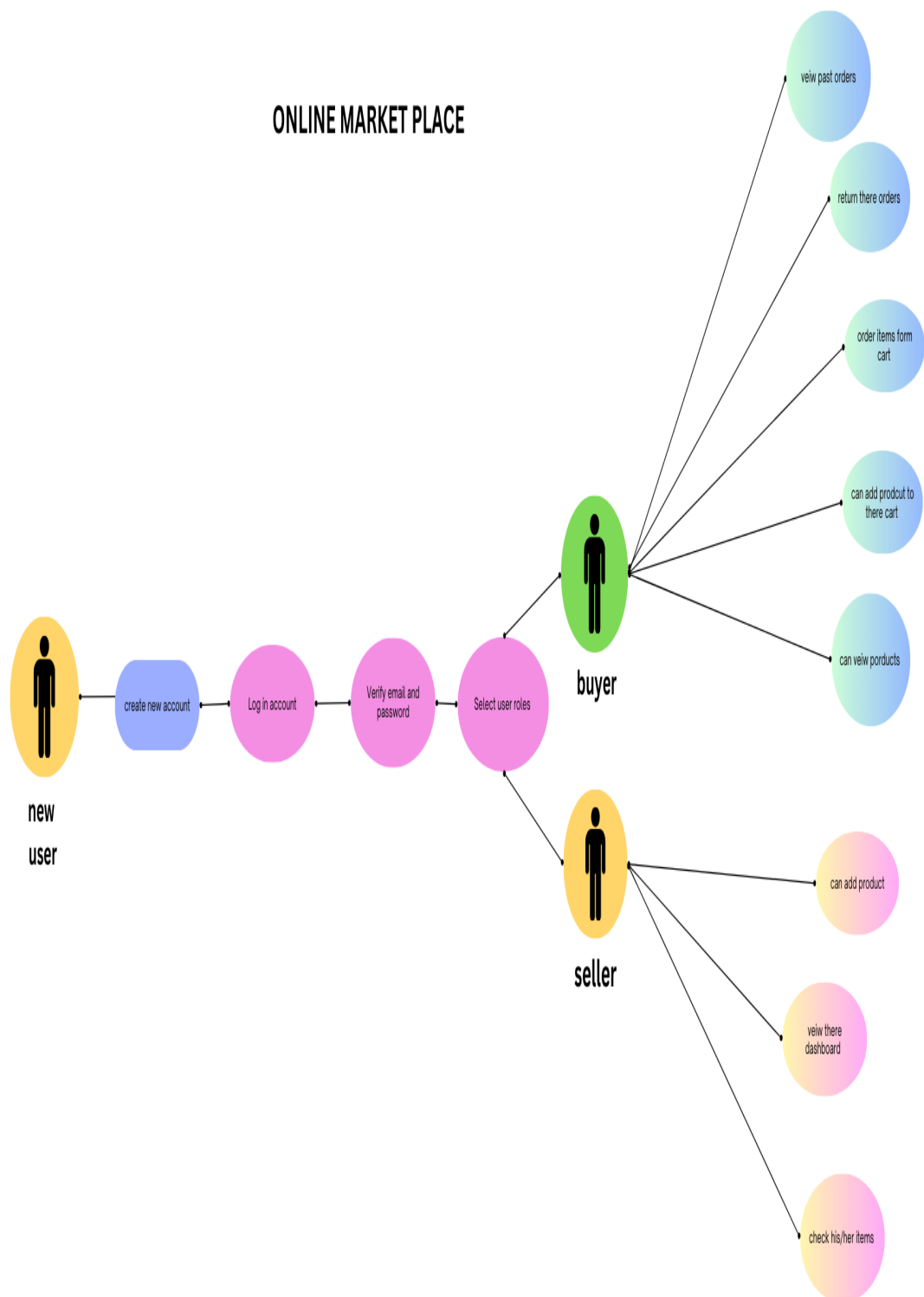


Figure 4.3: use case

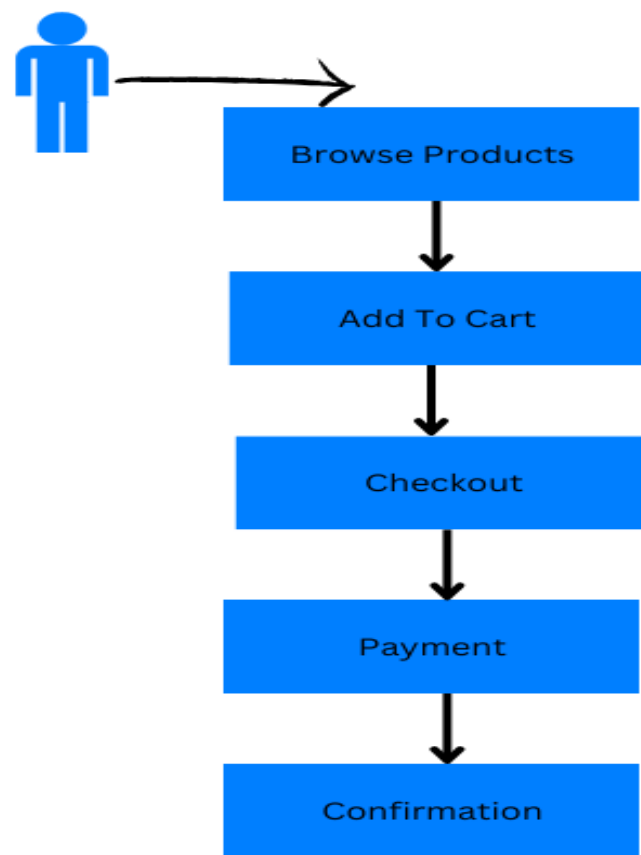


Figure 4.4: buyer activity

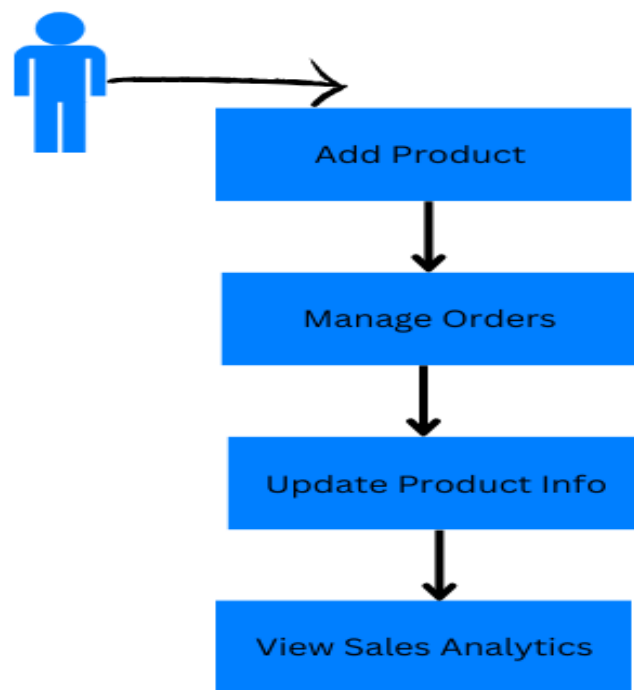
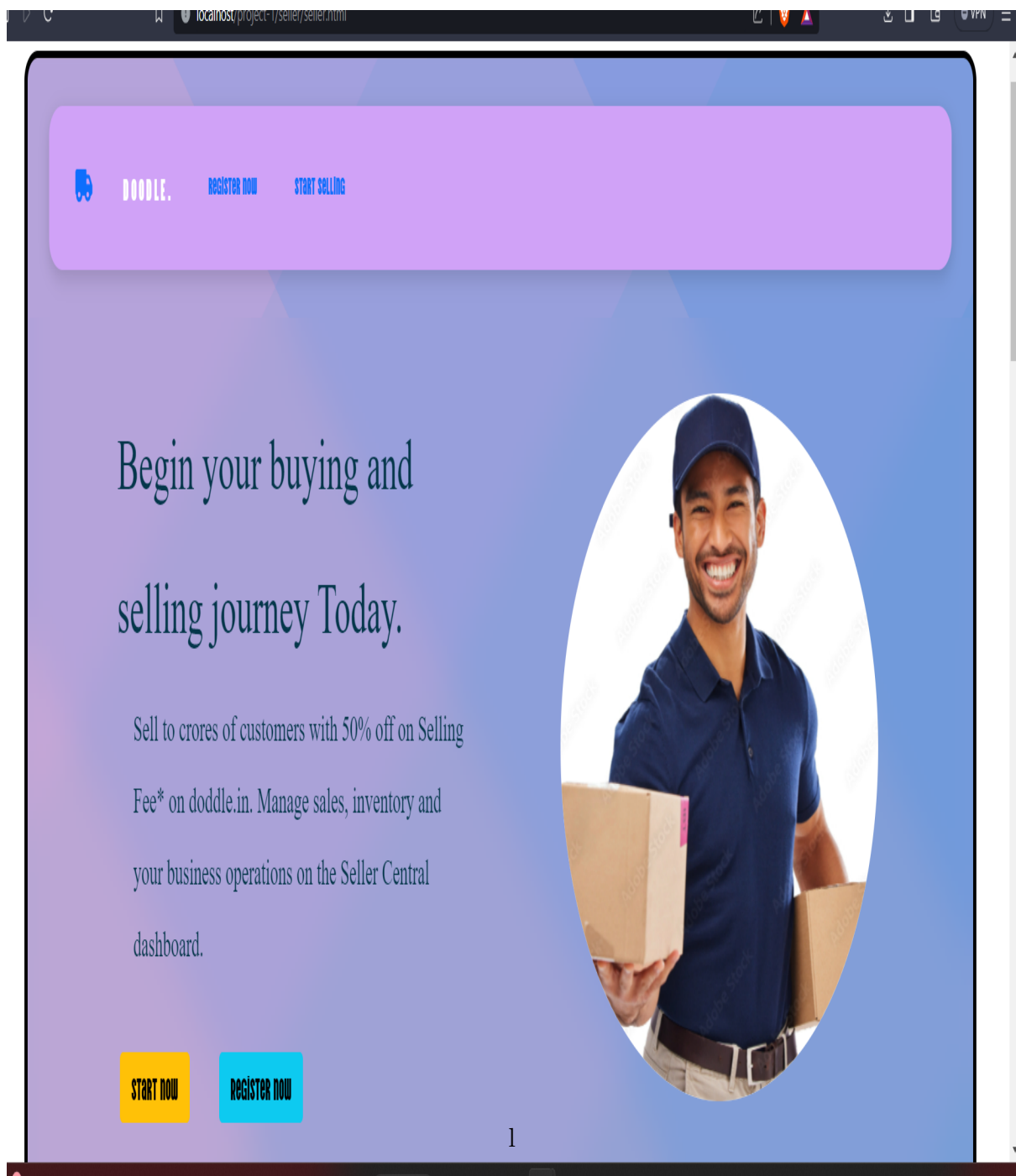


Figure 4.5: seller activity

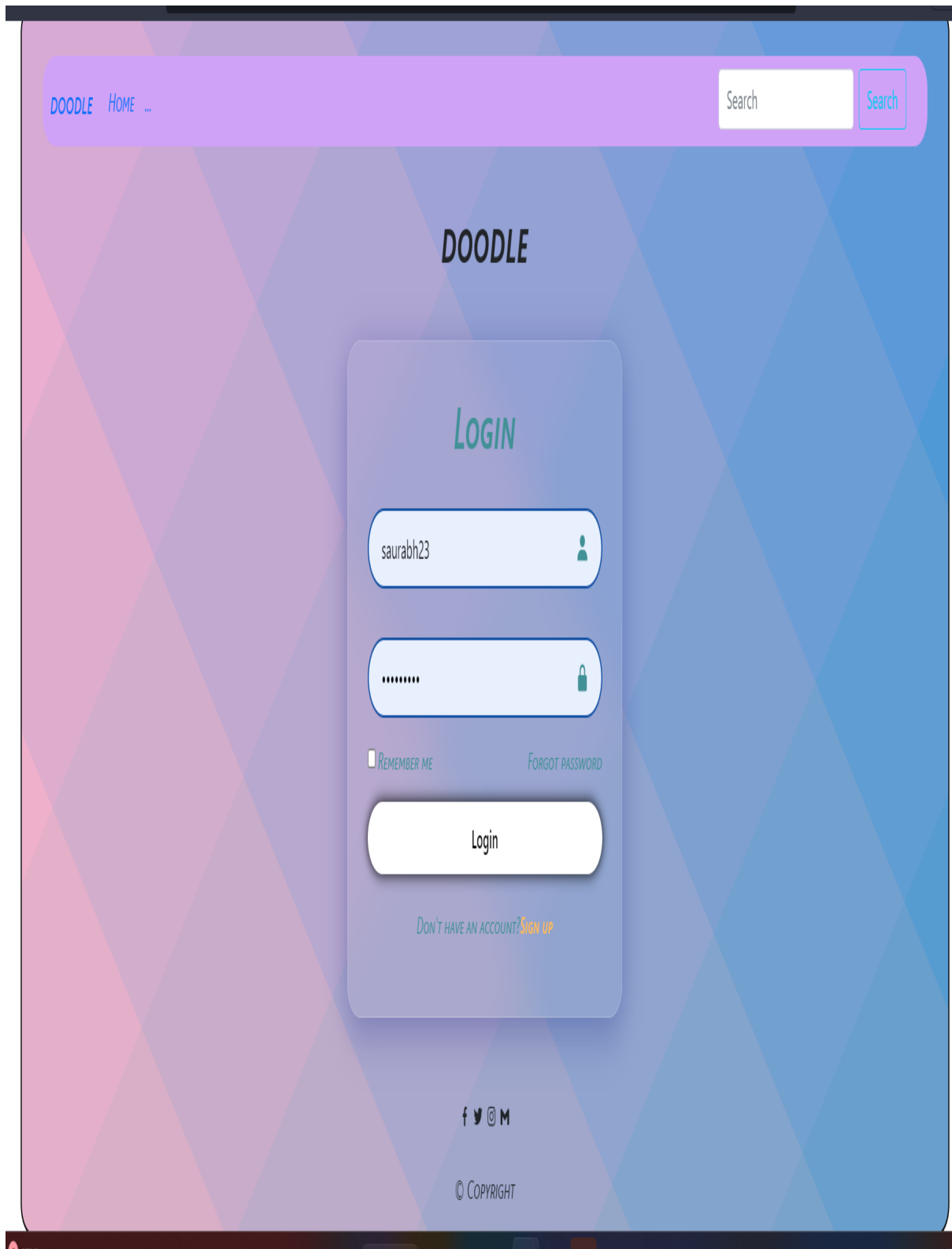
Chapter 5

RESULTS AND DISCUSSION

5.1 HOMEPAGE



5.2 LOGIN



5.3 REGISTRATION

SIGN UP

First Name

Last Name

Select Username

phonenumber

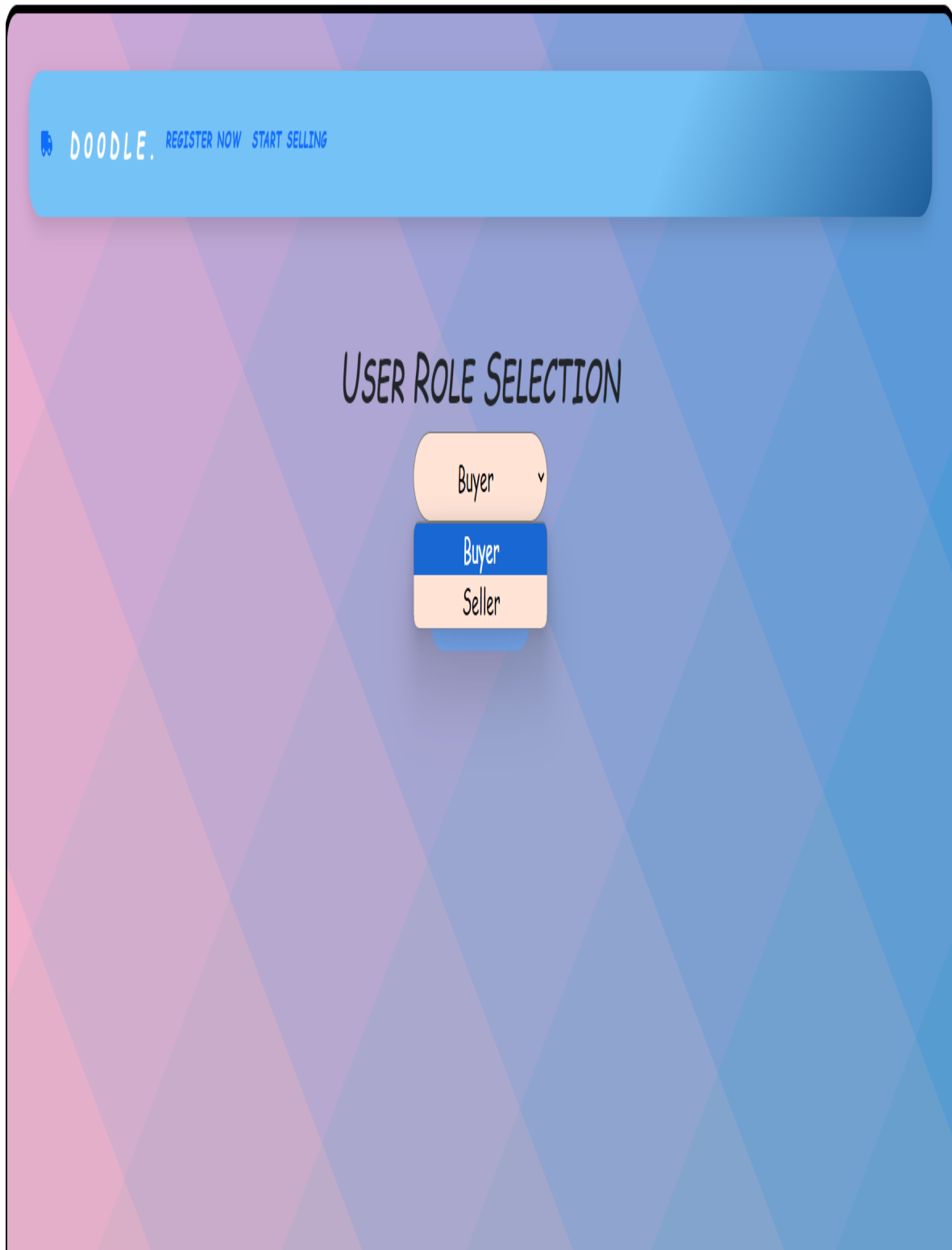
Email

Password

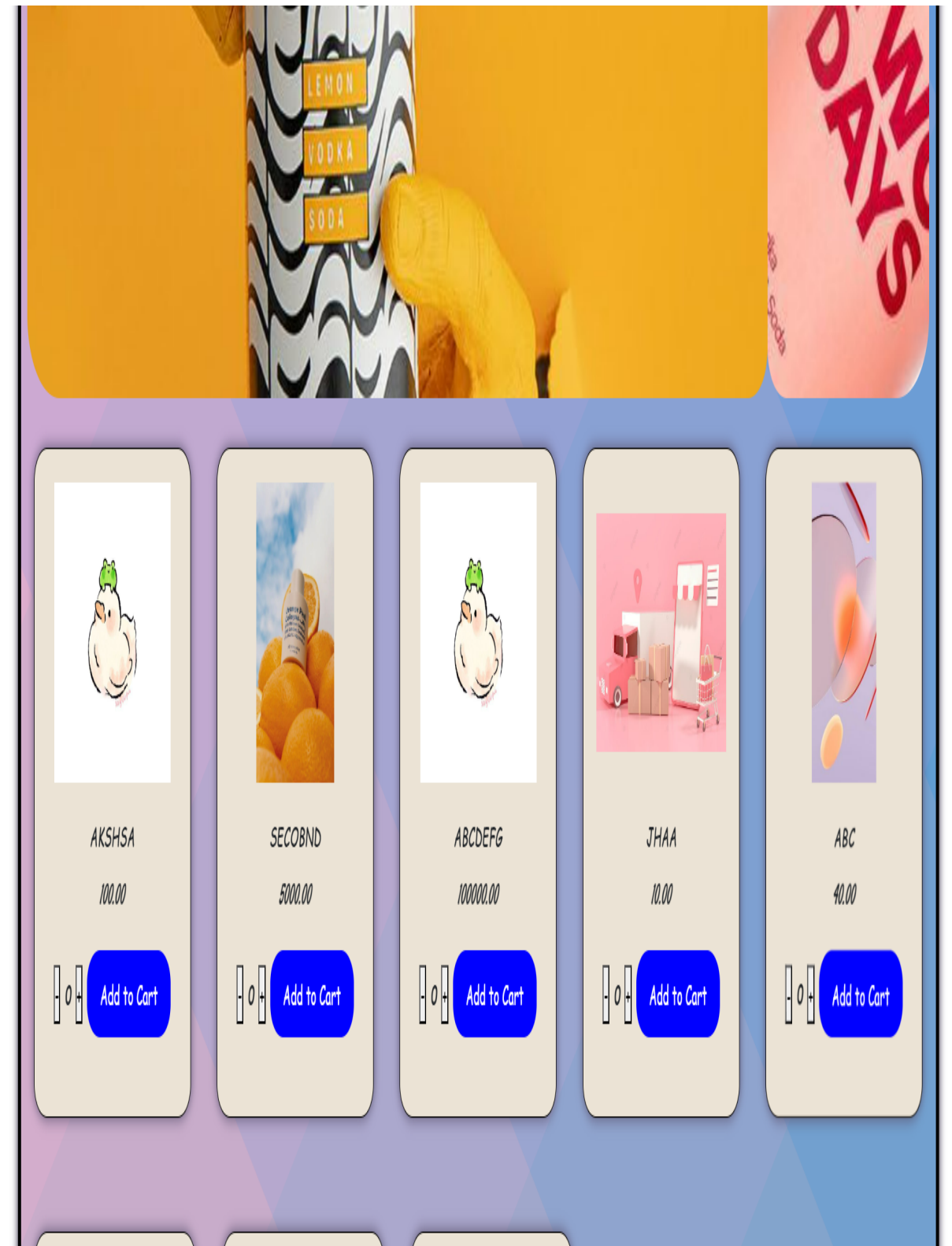
Sign up

ALREADY HAVE AN ACCOUNT? [LOGIN](#)

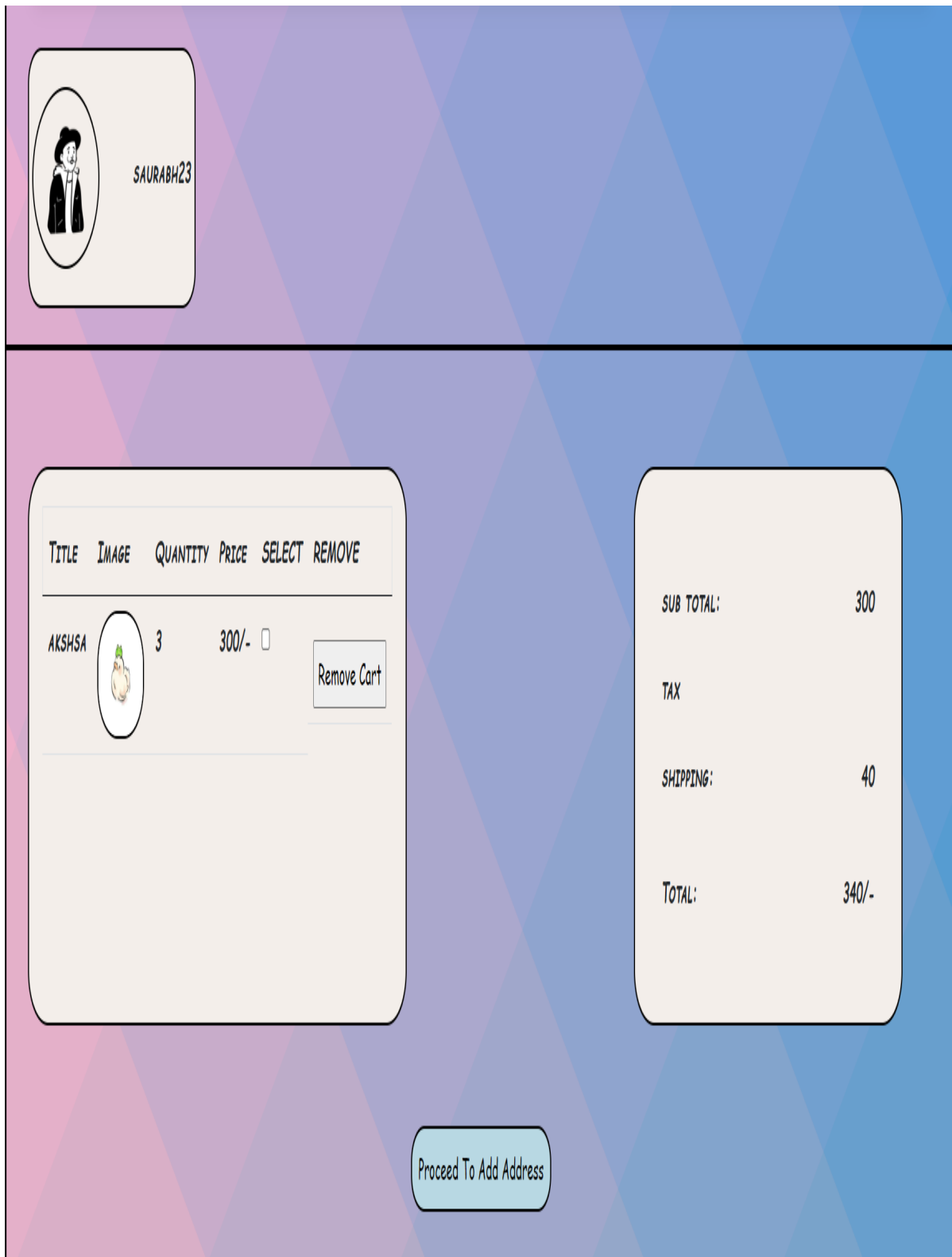
5.4 USER-ROLE-SELECTION



5.5 SHOPPING-PAGE




5.6 CART-PAGE



5.7 SELECT-ADDRESS



 SAURABH23

SELECT ADDRESS GIVEN BELOW

GORAKHPUR

CITY

UP

463163

☒

AGRA

CITY


UP

463163

☐

Submit

5.8 TRANSCATION




PAYMENT DETAILS:

UPI ID: KAKSHAT43@OKSBI

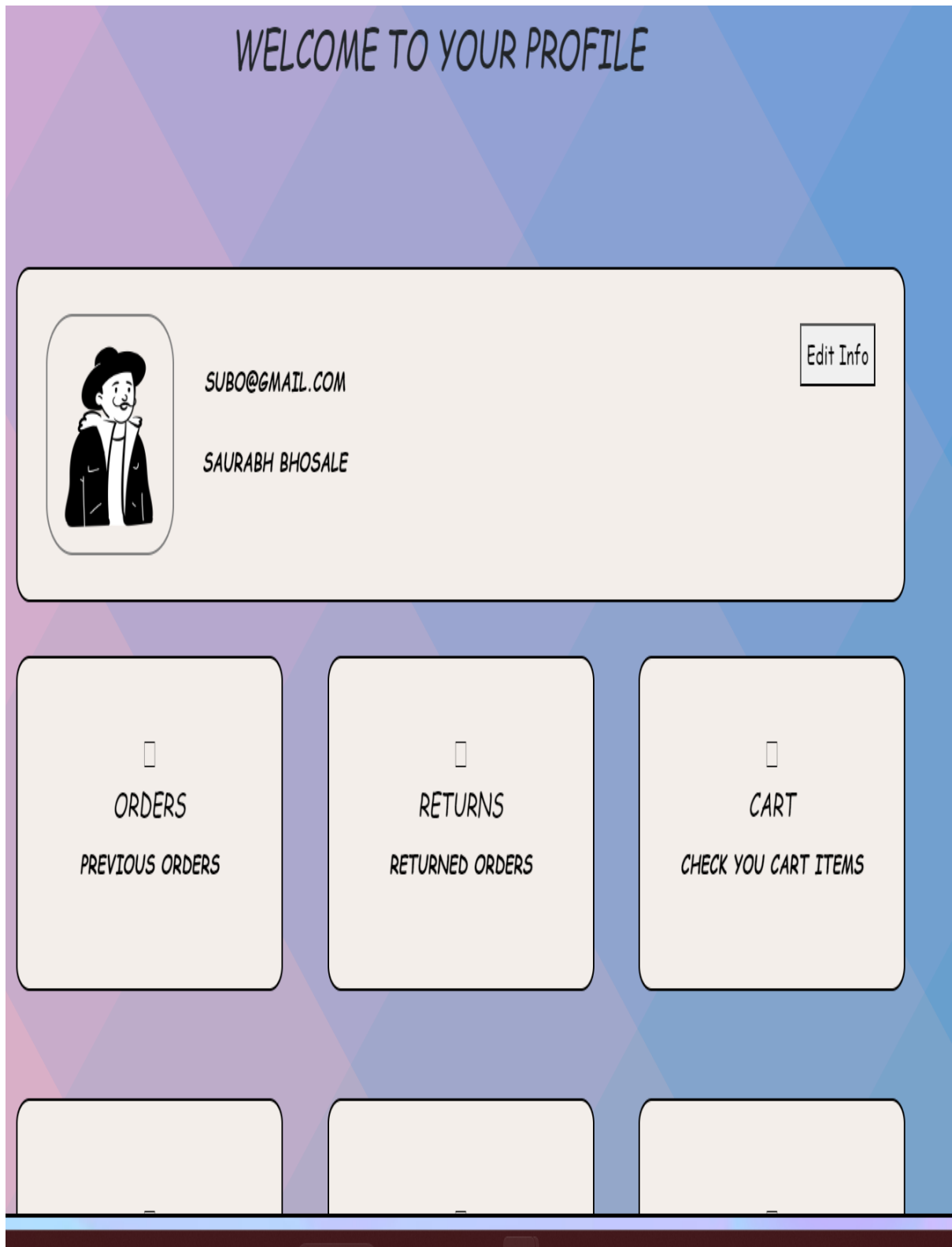
PHONE NUMBER: 6307257216

TOTAL AMOUNT: 340



Submit

5.9 BUYERS-PROFILE-PAGE



5.10 SELLER-DASHBOARD/PROFILE-PAGE



5.11 PRODUCT-SUBMISSION-PAGE

PRODUCT SUBMISSION FORM

PRODUCT TITLE:

abc

PRODUCT DESCRIPTION:

...

PRICE: 500

QUANTITY: 540

CATEGORY: Clothing ▾

PRODUCT IMAGE:

Choose File Screenshot 2...-15 094958.png

Submit

5.12 DISCUSSION

Within the realm of e-commerce, my implemented online marketplace stands as an innovative hub, revolutionizing the dynamics for both sellers and buyers. Sellers experience a seamless onboarding process, allowing them to effortlessly showcase their products through an intuitive interface. This includes detailed product descriptions, high-quality images, and flexible pricing options. The real game-changer lies in the personalized dashboard, offering sellers a comprehensive view of their inventory's performance. Dynamic bar graphs visually represent product quantities and sales trends, enabling sellers to make informed decisions about their offerings. Buyers, on the other hand, are greeted with a user-friendly interface designed for easy navigation and exploration. Advanced search functionalities and personalized recommendations enhance the shopping experience, providing buyers with a diverse and curated selection of products spanning various categories. Beyond mere transactions, the marketplace nurtures a sense of community. Robust review and rating systems empower users to make informed decisions, while direct messaging facilitates communication between buyers and sellers. This engagement fosters trust and collaboration within the digital ecosystem. Security is paramount, with the implementation of secure payment gateways, and verification processes. This ensures a safe and reliable environment for all participants. In essence, this online marketplace transcends conventional boundaries, transforming into a dynamic digital space where commerce meets community. It caters to the evolving needs of both sellers and buyers, offering not just products but an immersive and secure online shopping experience.

Chapter 6

CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

In the realm of e-commerce, the concept of an online marketplace where users seamlessly transition between being sellers and buyers opens up new dimensions of convenience and flexibility. This project has successfully implemented a dynamic platform that empowers individuals to engage in both selling and purchasing activities within a single environment. The dual-role functionality not only streamlines the user experience but also fosters a sense of community and interconnectedness. The project's user-friendly interface, secure transactions, and robust features contribute to a positive online trading environment. This multifaceted approach to online commerce provides users with the autonomy to manage their transactions efficiently and fosters a self-sustaining ecosystem. Furthermore, the project introduces innovative features such as [mention any unique features or functionalities], setting it apart in the competitive landscape of online marketplaces. The successful integration of these elements showcases the adaptability of the platform and its potential to revolutionize the way individuals approach online buying and selling.

6.2 FUTURE SCOPE

The scope of this project extends beyond its current state, offering ample opportunities for future enhancements and expansions. Some potential areas of growth include:

Enhanced User Features: integrate advanced user profiling, recommendation systems, and personalized dashboards to provide a tailored experience for each user.

Blockchain Integration: Explore the incorporation of blockchain technology to enhance security, transparency, and traceability of transactions, thereby building trust among users.

Collaborations: Establish collaborations with payment gateways, logistics companies, and other relevant partners to improve the overall efficiency and reliability of the platform.

ML Algorithms for Personalized Product Recommendations: ML algorithms analyze user behavior, including past purchases, viewed products, and search history. This data is then used to understand user preferences and patterns.

Natural Language Processing (NLP): AI-driven chatbots use NLP to understand and respond to customer queries in a human-like manner. This enables them to provide instant and accurate responses to a wide range of customer inquiries.