**CAR ACCESSORIES SHOP MANAGEMENT SYSTEM**

***A project report submitted to***

***Bishop Heber College (Autonomous), Tiruchirappalli***

***affiliated to Bharathidasan University, Tiruchirappalli – 620024***

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

# **Bachelor of Vocation in Information Technology**

*By*

**S. MOHANRAJ**

**(REGISTER NO: 225915135)**

*Under the guidance of*

**Mrs. P. USHA**,MCA .,M.Phil., SET., NET.,



**Department of Information Technology**

**Bishop Heber College(Autonomous)**

*(Nationally Re-accredited with ‘A++’ Grade by NAAC with a CGPA of 3.69 out of 4)*

*(Recognized by UGC as “College of Excellence”)*

*Tiruchirappalli 620 017*

**NOVEMBER 2024**

**Department of Information Technology**



**Bishop Heber College(Autonomous)**

**Tiruchirappalli-620 017,Tamilnadu,India**

**Phone No:0431-277 0136**

## CERTIFICATE

This Viva-Voce examination for the candidate  **S.MOHANRAJ**

### **(Reg No:225915135)** was held on\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Signature of HOD**

**Examiners:**

1.

2.

**Mrs. P.USHA, MCA .,M. Phil.,SET.,NET..**



Assistant Professor,

Department of Information Technology,

Bishop Heber College (Autonomous),

Tiruchirapalli-620 017

**Date:**

## CERTIFICATE

This is to certify that the project work entitled **“CAR ACCESSORIES SHOP MANAGEMENT SYSTEM”** is a bonafide work done under my supervision by **S.MOHANRAJ (Reg No:225915135)** and submitted to Bishop Heber College (Autonomous), Tiruchirappalli – 620 017 in partial fulfillment of the requirements for the award of the degree of Bachelor of Vocation in Information Technology during the odd semester of the academic year (2024-2025).

**Signature of the Guide**

## DECLARATION

I hereby declare that the work presented in this project work report is the original work done by me under the guidance of **Mrs. P.USHA, MCA., M.Phil., SET., NET.,** Assistant Professor of Information Technology, Bishop Heber College (Autonomous),Tiruchirapalli-620 017 and has not been included in any other project work submitted for any other degree.

Name of the Candidate :S.MOHANRAJ

Register Number :225915135

Semester :FIFTH

Academic Year :2024 – 2025

Course Code :U21ITPJ5

**Signature of the Candidate**

**ACKNOWLEDGEMENTS**

First of foremost, I thank **THE ALMIGHTY GOD** for granting abundant grace. Good health and knowledge to do this Project.

I express my sincere gratitude to **Dr.J.PRINCY MERLIN, M.Sc., SET., B.Ed., M.Phil., Ph.D., PGDCI.,** Principal., Bishop Heber College (Autonomous), Tiruchirappalli for providing the opportunity to pursue this programme.

I am highly indebted to thanks **Dr. J. JOHN RAYBIN JOSE, M.sc.,MCA, M.Phill, PGDCA ,Ph.D, SET.**  Associate professor and Head , Department of Information Technology , Bishop Heber College (Autonomous), Trichy for providing all the support and facilities to do this project work.

I wish to place on record my gratitude to **Mrs. P.USHA, MCA., M.Phil., SET., NET.,** Assistant Professor in Department of Information Technology, Bishop Heber College (Autonomous), Tiruchirappalli for grant me permission to pursue carry on with this project and guiding me to complete the project successfully.

I thank all the Staff members of the Department of Information Technology for their contribution to complete this project work successfully.

I record my deep sense of gratitude to my beloved parents and my friends for their encouragement and moral support extended during the period of my project.

**S.MOHANRAJ**

**ABSTRACT**

The **Car Accessories Shop Management System** is a PHP-based web application with SQL as the database, designed to simplify the management of a car accessories store. It allows the admin to manage products, track inventory, and view customer orders, while providing customers with a smooth shopping experience. The admin can add, update, and delete products, monitor stock levels, and generate sales reports for better business insights. Customers can register, log in, browse car accessories, add items to their cart, and place orders. The system maintains a detailed order history, allowing customers to track past and ongoing purchases. The database includes tables for users, products, orders, and order items, ensuring efficient data organization and retrieval. Built with a responsive design using frameworks like Bootstrap, the platform is user-friendly on both desktop and mobile devices. JavaScript and AJAX are used for real-time updates, enhancing the shopping experience. The system is scalable and easy to modify, making it adaptable to the evolving needs of the store.

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S. NO.** | **TITLE** | **PAGE NO.** |
| **1** | **INTRODUCTION** | 1 |
| **2** | **SYSTEM STUDY** | 2 |
| **2.1. Project Description** | 2 |
| 2.1.1. Existing System | 2 |
| 2.1.2. Proposed System | 3 |
|  | 2.1.3 Modules and Description | 4 |
|  | **2.2 Requirement Specification** | 5 |
|  | 2.2.1 Hardware Requirements | 5 |
|  | 2.2.2 Software Requirements | 5 |
| **3.** | **SYSTEM DESIGN** | 8 |
| 3.1. Logical Design | 8 |
| 3.2 Database Design | 9 |
| **4.** | **SYSTEM DEVELOPMENT** | 10 |
| 4.1. Program Code | 10 |
| **5.** | **SYSTEM TESTING** | 26 |
| 5.1. Unit testing | 26 |
| 5.2.Integration testing | 27 |
| 5.3 Validation Testing | 28 |
| **6.** | **SYSTEM IMPLEMENTATION** | 29 |
| **7.** | **CONCLUSION** | 36 |
|  | **BIBLIOGRAPHY** | 37 |

# **1.INTRODUCTION**

The Car Accessories Shop Management System is an application designed for maintaining and managing various aspects of a car accessories store. This project covers the basic functionality of managing inventory, sales, and customer orders within a car accessories shop environment. Car accessory shops play a critical role in the automotive industry by providing a wide range of parts and accessories to enhance the performance, comfort, and aesthetics of vehicles. These shops also offer services such as product installations, maintenance, and customization for vehicles.

The goal of this project is to develop a system that solves the operational needs of a car accessories store, ensuring efficient management of products and customer demands. The system should provide various ways to perform tasks such as inventory tracking, order management, sales processing, and customer service. Additionally, it should enhance the store's workspace with extra functionalities that go beyond the scope of a traditional management system, such as customer feedback tracking and product recommendations.

There are different types of car accessory stores, including those specializing in performance upgrades, aesthetic modifications, and essential maintenance parts. Some stores may focus on performance accessories, offering parts to improve speed, handling, and overall vehicle dynamics. Others may focus on aesthetic enhancements such as custom wheels, lighting, or body kits. There are also general car accessory shops that provide a broad range of products, including basic items like seat covers, air fresheners, and car care products. This system will cater to the specific needs of these different types of car accessory shops, ensuring smooth operations and improved customer satisfaction.

# **2.SYSTEM STUDY**

System analysis is a process of gathering the facts concerning the system, breaking them into elements, and understanding the relationships between these elements. It provides a framework for visualizing the organizational and environmental factors that operate on a system. The quality of work performed by a machine is usually uniform, neat, and more reliable when compared to performing the same operations manually.

## 2.1.Project Description

The scope of the project is to store and access the database consisting of customer details, product inventories, and transaction records. This data can be shared with the concerned departments, such as sales, inventory management, and customer service, to streamline operations. The Car Accessories Shop Management System is an application that manages information on customer orders, available products, and transactions, typically for small to medium-sized car accessory shops.

### **2.1.1.Existing System**

The existing system operates manually. It involves a lot of complexity and requires significant human effort, including paperwork and manual tracking of sales, inventory, and customer data. All data must be recorded on physical ledgers, making it a tedious and risky process. As the number of transactions increases, so does the volume of data, making data management more challenging. To retrieve any information, an extensive manual search through papers is required, making it time-consuming and inefficient.

### **2.1.2.Proposed System.**

The proposed system is designed to provide the best services to both the car accessory shop and its customers by being user-friendly and reducing the time it takes to complete tasks compared to the current manual system. The new system is highly computerized, ensuring that data related to customer orders, product inventories, and sales are securely stored and managed with high accuracy. This reduces errors caused by human mistakes or machine malfunctions. The system also validates data as it is entered, ensuring data integrity. When necessary, appropriate messages are displayed to prevent user confusion. The data entry screen offers features for viewing records and modifying various types of data as needed.

**Advantages:**

1. It satisfies the user’s requirements.
2. The system generates various types of reports and information for different purposes (sales, inventory, etc.).
3. It is easy for both users and operators to understand.
4. It is simple to operate.
5. The system is expandable, allowing for future upgrades or the addition of new features.

**2.1.3 Modules and Description**

Car Accessories Shop Management System is the add record and delete record to all details.

* **Register Account :**

This module is use to create an account with personal information for easier access to the shop’s services.

* **View Transaction Details :**

This module is use to view all transaction details.

* **Manage Inventory :**

Enables the administrator to add, update, or remove products from the inventory, manage stock levels, and set prices.

* **Manage Orders :**

Administrator to view, approve, or cancel customer orders, and track order statuses.

* **View customer Details :**

This module is use to view all customer data.

* **View Sales Report :**

This module is use to displays sales reports, helping administrators analyze sales trends and business performance.

* **Manage Service Bookings :**

This module is use to view, schedule, and manage customer service bookings, including confirmations and cancellations.

* **View Profile :**

This module is use to customers to view and edit their account details, such as name, email, and contact information.

* **Add Product to Cart :**

This module is use to add selected products to their shopping cart for future purchase.

* **Checkout :**

This module is use to process for customers to review their cart, select payment options, and place orders.

* **View Delivery Status :**

customers to track the delivery status of their purchased products, including shipment and estimated delivery time.

* **Book Service :**

This module is use to schedule and book services, such as car maintenance or installation of accessories.

### **2.2. Requirement Specification**

Requirements specification involves frequent communication with system users to determine specific feature expectations, resolve conflicts or ambiguities, and ensure that the system meets the needs of its users. The goal is to ensure the system or product conforms to the client’s needs rather than forcing users to adapt to predefined requirements. Requirements analysis is a team effort that demands a combination of hardware, software, and human factors engineering, along with strong communication and interpersonal skills.

### **2.2.1. Hardware Requirement**

The hardware specification of the computer system required for developing the **Car Accessories Shop Management System** is as follows:

* **Processor**: Intel Core i3
* **Hard Disk**: 500 GB
* **RAM**: 8 GB
* **Keyboard**: Standard Keyboard
* **Mouse**: Standard Mouse Pad

### **2.2.2. Software Requirement**

A **Software Requirement Specification** is a detailed description of the system's behavior. It includes the use cases that describe all user interactions with the system. The software tools used for the **Car Accessories Shop Management System** are as follows:

* **Operating System**: Windows 10
* **Software Applications**: WAMP
* **Front End**: PHP, HTML, CSS, JavaScript
* **Back End**: MySQL Database

#### **Operating System**

The **Operating System** manages the communication between hardware and software, allowing the system to function. This project will use **Windows 10** , which is widely supported and ideal for running the required software stack.

#### **Development Tools**

* **PHP**: PHP (Hypertext Preprocessor) is a widely-used, open-source scripting language that is especially suited for web development and can be embedded in HTML. PHP is server-side and allows the creation of dynamic content and interaction with the database.
* **HTML/CSS**: HTML (HyperText Markup Language) and CSS (Cascading Style Sheets) are used for structuring and designing the web pages of the system. HTML creates the structure, while CSS is responsible for the layout, design, and responsiveness.
* **JavaScript**: JavaScript is a client-side scripting language used to add dynamic elements, interactivity, and enhanced functionality to the web pages. It enables features such as form validation, dynamic content updates, and user interaction with the interface without reloading the page.

#### **Database**:

* **MySQL**: MySQL is an open-source relational database management system that will be used to manage all data related to customers, products, orders, and transactions. MySQL is known for its reliability, ease of use, and support for high transaction loads.

# **3.SYSTEM DESIGN**

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. System design could be seen as the application of system theory to product development.

**3.1. Logical Design**

Logical design is an abstract concept in computer programming by which programmers arrange data in a series of logical relationships known as attributes or entities. An entity refers to a chunk of information, whereas an attribute defines the unique properties of an entity.

* Fig.3.1 Car Accessories Shop Management System

**3.2 Database Design**

Database design is the process of producing a detailed data model of the database. This data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity. A good database design is important in ensuring consistent data, elimination of data redundancy, efficient execution of queries and high-performance application.

**Users Table :**

|  |  |
| --- | --- |
| **Field** | **Data Type** |
| Id | Int |
| Username | Varchar |
| Email | Varchar |
| Password | Varchar |
| Image | Longblob |
| Address | Varchar |
| Phone No | Int |
| Date of Birth | Date |
| Created\_at | Timestamp |

**Products Table :**

|  |  |
| --- | --- |
| **Field** | **Data Type** |
| Id | Int |
| Name | Varchar |
| Category | Varchar |
| Price | Decimal |
| Image | Longblob |
| Original\_Price | Decimal |
| Discount\_Percentage | Int |
| Description | Varchar |
| Sold\_By | Varchar |
| Total\_Products | Int |
| Warranty | Varchar |

**Cart Table :**

|  |  |
| --- | --- |
| **Field** | **Data Type** |
| Id | Int |
| User\_Id | Int |
| Product\_Id | Int |
| Quantity | Int |

**Orders Table :**

|  |  |
| --- | --- |
| **Field** | **Data Type** |
| Order\_Id | Int |
| User\_Id | Int |
| Order\_Date | Datetime |
| Delivery\_Status | Varchar |
| Total\_Amount | Decimal |
| Payment\_Method | Varchar |
| Payment\_Status | Varchar |
| Shipping\_Address | Varchar |
| Billing\_Address | Varchar |
| Created\_At | Timestamp |
| Updated\_At | Timestamp |
| Customer\_Name | Varchar |

**Service\_Booking Table :**

|  |  |
| --- | --- |
| **Field** | **Data Type** |
| Id | Int |
| User\_Id | Int |
| Service | Varchar |
| Booking\_date | Date |

**Payment Table :**

|  |  |
| --- | --- |
| **Field** | **Data Type** |
| Id | Int |
| User\_Id | Int |
| Product\_Id | Int |
| Amount | Decimal |
| Payment\_Status | Varchar |

# **4.SYSTEM DEVELOPMENT**

The Software Development Life Cycle(SDLC), also referred to as the application development life-cycle, is a term used in system engineering, information system and software engineering to describe a process for planning, creating testing and deploying an information system. The system developments life-cycle concept applies to a range of hardware and software configurations, as a system can be composed of hardware only, software only, or a combination of both.

## 4.1 Program Code

Here I displayed the coding of my program:

**db.php:**

<?php

$servername = "localhost";

$username = "root";

$password = "";

$dbname = "car\_accessories";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);}?>

**Login.php:**

<?php

session\_start();

require '../db.php';

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$email = $\_POST['email'];

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

$sql = "SELECT \* FROM users WHERE email = ? LIMIT 1";

$stmt = $conn->prepare($sql);

$stmt->bind\_param('s', $email);

$stmt->execute();

$result = $stmt->get\_result();

if ($result->num\_rows > 0) {

$user = $result->fetch\_assoc();

if (password\_verify($password, $user['password'])) {

$\_SESSION['email'] = $user['email'];

$\_SESSION['user\_id'] = $user['id'];

$\_SESSION['loggedin'] = true;

if ($email === 'admin@gmail.com') {

header("Location: ../Admin/Dashboard.php");

} else {

header("Location: ../index.html"); }exit;

} else {

echo "Invalid password.";}

} else {

echo "No user found with that email address."; }

$stmt->close();}

$conn->close();?>

**Add\_Product.php:**

<?php

require '../db.php';

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

$name = isset($\_POST['name']) ? mysqli\_real\_escape\_string($conn, $\_POST['name']) : '';

$category = isset($\_POST['category']) ? mysqli\_real\_escape\_string($conn, $\_POST['category']) : '';

$price =isset($\_POST['price'])?mysqli\_real\_escape\_string($conn, $\_POST['price']) :'';

$rating=isset($\_POST['rating'])?mysqli\_real\_escape\_string($conn,$\_POST['rating'])'';

$original\_price=isset($\_POST['original\_price'])?mysqli\_real\_escape\_string($conn,$\_POST['original\_price']) : NULL;

$discount\_percentage = isset($\_POST['discount\_percentage']) ? mysqli\_real\_escape\_string($conn, $\_POST['discount\_percentage']) : NULL;

$description = isset($\_POST['description']) ? mysqli\_real\_escape\_string($conn, $\_POST['description']) : NULL;

$material = isset($\_POST['material']) ? mysqli\_real\_escape\_string($conn, $\_POST['material']) : NULL;

$care = isset($\_POST['care']) ? mysqli\_real\_escape\_string($conn, $\_POST['care']) : NULL;

$sold\_by = isset($\_POST['sold\_by']) ? mysqli\_real\_escape\_string($conn, $\_POST['sold\_by']) : NULL;

$positive\_feedback\_percentage = isset($\_POST['positive\_feedback\_percentage']) ? mysqli\_real\_escape\_string($conn, $\_POST['positive\_feedback\_percentage']) : NULL;

$total\_products = isset($\_POST['total\_products']) ? mysqli\_real\_escape\_string($conn, $\_POST['total\_products']) : NULL;

$warranty = isset($\_POST['warranty']) ? mysqli\_real\_escape\_string($conn, $\_POST['warranty']) : NULL;

if (isset($\_FILES['image']['tmp\_name'])) {

$image = $\_FILES['image']['tmp\_name'];

$imageData = addslashes(file\_get\_contents($image));} else {

$imageData = NULL; }

$sql = "INSERT INTO products (name, category, price, rating, original\_price, discount\_percentage, description, material, care, sold\_by, positive\_feedback\_percentage, total\_products, warranty, image)

VALUES ('$name', '$category', '$price', '$rating', '$original\_price', '$discount\_percentage', '$description', '$material', '$care', '$sold\_by', '$positive\_feedback\_percentage', '$total\_products', '$warranty', '$imageData')";

if ($conn->query($sql) === TRUE) {

echo "New product added successfully!";

echo '<br><a href="add\_product.php"><button>Add Another Product</button></a>';

echo '<br><a href="inventory.php"><button>View Inventory</button></a>';

} else {

echo "Error: " . $sql . "<br>" . $conn->error;

echo '<br><a href="inventory.php"><button>Go TO Inventry</button></a>';}}?>

**Edit\_Product.php:**

<?php

require '../db.php';

if (isset($\_GET['id'])) {

$product\_id = $\_GET['id'];

$sql = "SELECT \* FROM products WHERE id = $product\_id";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

$product = $result->fetch\_assoc();} else {

echo "No product found!";exit;}

} else {

echo "Invalid product ID!";exit;}

if ($\_SERVER['REQUEST\_METHOD'] == 'POST') {

$name = $\_POST['name'];

$category = $\_POST['category'];

$price = $\_POST['price'];

$rating = $\_POST['rating'];

$original\_price = $\_POST['original\_price'];

$discount\_percentage = $\_POST['discount\_percentage'];

$description = $\_POST['description'];

$material = $\_POST['material'];

$care = $\_POST['care'];

$sold\_by = $\_POST['sold\_by'];

$positive\_feedback\_percentage = $\_POST['positive\_feedback\_percentage'];

$total\_products = $\_POST['total\_products'];

$warranty = $\_POST['warranty'];

if (!empty($\_FILES['image']['name'])) {

$image = addslashes(file\_get\_contents($\_FILES['image']['tmp\_name']));

$image\_sql = ", image='$image'";

} else {

$image\_sql = '';}

$update\_sql = "UPDATE products SET

name='$name',

category='$category',

price='$price',

rating='$rating',

original\_price='$original\_price',

discount\_percentage='$discount\_percentage',

description='$description',

material='$material',

care='$care',

sold\_by='$sold\_by',

positive\_feedback\_percentage='$positive\_feedback\_percentage',

total\_products='$total\_products',

warranty='$warranty'

$image\_sql

WHERE id=$product\_id";

if ($conn->query($update\_sql) === TRUE) {

echo "Product updated successfully!";

header('Location: inventory.php');

exit;

} else {

echo "Error updating product: " . $conn->error;}}?>

**Add\_to\_Cart.php:**

<?php

session\_start();

require '../db.php';

if (!isset($\_SESSION['user\_id'])) {

echo 'Please log in to add items to the cart.';

exit;

}

$user\_id = $\_SESSION['user\_id'];

$product\_id = $\_POST['product\_id'];

$quantity = $\_POST['quantity'];

$query = "SELECT \* FROM cart\_items WHERE user\_id = ? AND product\_id = ?";

$stmt = $conn->prepare($query);

$stmt->bind\_param('ii', $user\_id, $product\_id);

$stmt->execute();

$result = $stmt->get\_result();

if ($result->num\_rows > 0) {

$update\_query = "UPDATE cart\_items SET quantity = quantity + ? WHERE user\_id = ? AND product\_id = ?";

$stmt\_update = $conn->prepare($update\_query);

$stmt\_update->bind\_param('iii', $quantity, $user\_id, $product\_id);

$stmt\_update->execute();

} else {

$insert\_query = "INSERT INTO cart\_items (user\_id, product\_id, quantity) VALUES (?, ?, ?)";

$stmt\_insert = $conn->prepare($insert\_query);

$stmt\_insert->bind\_param('iii', $user\_id, $product\_id, $quantity);

$stmt\_insert->execute();

}

header("Location: cart.php");

exit();?>

**Checkout.php:**

<?php

session\_start();

require '../db.php';

require\_once 'stripe-php-master/init.php';

\Stripe\Stripe::setApiKey('sk\_test\_51Q2Zi0Lzf1Nkvl1Jkp5MvnrDv7FyVnDJSMmPrs0GAPxYwGaLcbJRsSSzO3ZtZXJUldwAmzOUso5QaGWfhusOeaKt00gSVTacvk');

$user\_id = $\_SESSION['user\_id'];

$name = isset($\_GET['name']) ? htmlspecialchars($\_GET['name']) : 'N/A';

$address = isset($\_GET['address']) ? htmlspecialchars($\_GET['address']) : 'N/A';

$phone = isset($\_GET['phone']) ? htmlspecialchars($\_GET['phone']) : 'N/A';

$payment\_method = isset($\_GET['payment\_method']) ? htmlspecialchars($\_GET['payment\_method']) : 'N/A';

$sql = "SELECT products.id, products.name, products.price, cart\_items.quantity

FROM cart\_items

JOIN products ON cart\_items.product\_id = products.id

WHERE cart\_items.user\_id = ?";

$stmt = $conn->prepare($sql);

$stmt->bind\_param("i", $user\_id);

$stmt->execute();

$result = $stmt->get\_result();

$total\_price = 0;

$line\_items = [];

while ($row = $result->fetch\_assoc()) {

$total\_price += $row['price'] \* $row['quantity'];

$line\_items[] = [

'price\_data' => [

'currency' => 'inr',

'product\_data' => [

'name' => $row['name'],

],

'unit\_amount' => $row['price'] \* 100,

],

'quantity' => $row['quantity'],];

}if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

// Get shipping details from the session or POST request

$name = $\_SESSION['shipping\_details']['name'];

$address = $\_SESSION['shipping\_details']['address'];

$phone = $\_SESSION['shipping\_details']['phone'];

$payment\_method = $\_SESSION['shipping\_details']['payment\_method'];

$session = \Stripe\Checkout\Session::create([

'payment\_method\_types' => ['card'],

'line\_items' => $line\_items,

'mode' => 'payment',

'success\_url' => 'http://localhost/Car-Accessories/User/checkout\_success.php?session\_id={CHECKOUT\_SESSION\_ID}' .

'&name=' . urlencode($name) .

'&address=' . urlencode($address) .

'&phone=' . urlencode($phone) .

'&payment\_method=' . urlencode($payment\_method), // Include the shipping details in the success URL

'cancel\_url' => 'http://localhost/Car-Accessories/User/checkout\_cancel.php', // Change to your cancel URL]);

header("Location: " . $session->url);

exit();}?>

**Checkout\_Success.php:**

<?php

session\_start();

require '../db.php';

require\_once 'stripe-php-master/init.php';

\Stripe\Stripe::setApiKey('sk\_test\_51Q2Zi0Lzf1Nkvl1Jkp5MvnrDv7FyVnDJSMmPrs0GAPxYwGaLcbJRsSSzO3ZtZXJUldwAmzOUso5QaGWfhusOeaKt00gSVTacvk');

if (!isset($\_GET['session\_id'])) {

header("Location: checkout.php"); // Redirect if no session\_id

exit();}

$session\_id = $\_GET['session\_id'];

$session = \Stripe\Checkout\Session::retrieve($session\_id);

$user\_id = $\_SESSION['user\_id'];

$name = isset($\_GET['name']) ? htmlspecialchars($\_GET['name']) : 'N/A';

$address = isset($\_GET['address']) ? htmlspecialchars($\_GET['address']) : 'N/A';

$phone = isset($\_GET['phone']) ? htmlspecialchars($\_GET['phone']) : 'N/A';

$payment\_method = isset($\_GET['payment\_method']) ? htmlspecialchars($\_GET['payment\_method']) : 'N/A';

$billing\_address = $address;

$order\_date = new DateTime();

$order\_date\_formatted = $order\_date->format('Y-m-d H:i:s');

$delivery\_status = "Processing";

$total\_amount = $session->amount\_total / 100;

$currency\_code = $session->currency;

$txn\_id = $session->id;

$payment\_status = $session->payment\_status;

$payment\_response = json\_encode($session);

$stmt = $conn->prepare("INSERT INTO tbl\_payment (`user\_id`, `amount`, `currency\_code`, `txn\_id`, `payment\_status`, `payment\_response`) VALUES (?, ?, ?, ?, ?, ?)");

$stmt->bind\_param("isssss", $user\_id, $total\_amount, $currency\_code, $txn\_id, $payment\_status, $payment\_response);

if (!$stmt->execute()) {

echo "Error inserting payment details: " . $stmt->error;}

$stmt1 = $conn->prepare("INSERT INTO orders(`user\_id`, `customer\_name`, `order\_date`, `delivery\_status`, `total\_amount`, `payment\_method`, `payment\_status`, `shipping\_address`, `billing\_address`) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)");

$stmt1->bind\_param("issssssss", $user\_id, $name, $order\_date\_formatted, $delivery\_status, $total\_amount, $payment\_method, $payment\_status, $shipping\_address, $billing\_address);

if ($stmt1->execute()) {

$order\_id = $stmt1->insert\_id;

$cart\_query = $conn->prepare("SELECT cart\_items.\*, products.price AS product\_price FROM cart\_items

JOIN products ON cart\_items.product\_id = products.id

WHERE cart\_items.user\_id = ?");

$cart\_query->bind\_param("i", $user\_id);

$cart\_query->execute();

$cart\_result = $cart\_query->get\_result();

$insert\_item\_stmt = $conn->prepare("INSERT INTO order\_items (`order\_id`, `product\_id`, `quantity`, `price`) VALUES (?, ?, ?, ?)");

while ($row = $cart\_result->fetch\_assoc()) {

$product\_id = $row['product\_id'];

$quantity = $row['quantity'];

$price = isset($row['price']) ? $row['price'] : $row['product\_price'];

$insert\_item\_stmt->bind\_param("iiid", $order\_id, $product\_id, $quantity, $price);

if (!$insert\_item\_stmt->execute()) {

echo "Error inserting order item: " . $insert\_item\_stmt->error;

}}

$insert\_item\_stmt->close();

$delete\_stmt = $conn->prepare("DELETE FROM cart\_items WHERE user\_id = ?");

$delete\_stmt->bind\_param("i", $user\_id);

if (!$delete\_stmt->execute()) {

echo "Error removing items from cart: " . $delete\_stmt->error; }

$\_SESSION['success\_message'] = "Thank you for your purchase!";

} else {

echo "Error inserting order details: " . $stmt1->error;}?>

**Remove\_From\_Cart.php:**

<?php

session\_start();

require'../db.php';

if (!isset($\_SESSION['user\_id'])) {

$\_SESSION['error\_message'] = "You must be logged in to remove items from the cart.";

header("Location: cart.php");

exit;}

$user\_id = $\_SESSION['user\_id'];

$product\_id = isset($\_POST['product\_id']) ? intval($\_POST['product\_id']) : 0;

$sql = "DELETE FROM cart\_items WHERE user\_id = ? AND product\_id = ?";

$stmt = $conn->prepare($sql);

$stmt->bind\_param("ii", $user\_id, $product\_id);

if ($stmt->execute()) {

$\_SESSION['success\_message'] = "Item removed from cart successfully.";

} else {

$\_SESSION['error\_message'] = "Error removing item from cart. Please try again.";

}

$stmt->close();

$conn->close();

header("Location: cart.php");

exit;

?>

# **5.SYSTEM TESTING**

System testing is the process of evaluation and software item to detect differences between given input and expected output.Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words, software testing is a verification and validation process.

### **5.1. Unit Testing:**

Testing of individual software components or modules. Typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. may require developing test driver modules or test harnesses.

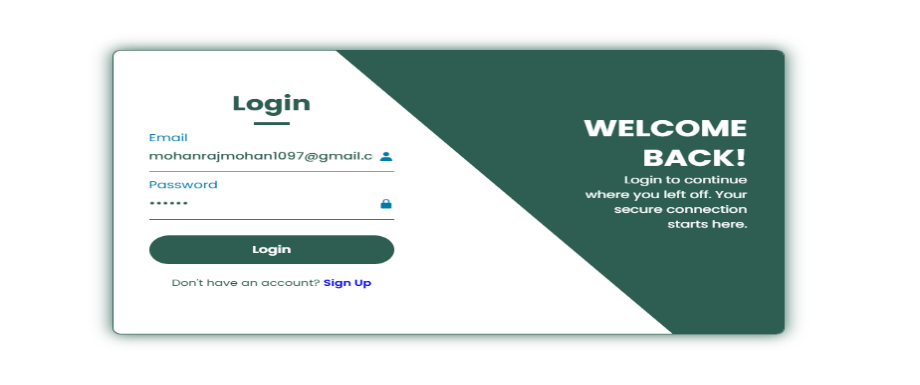


Fig:5.1 Login Page

### **5.2. Integration Testing:**

Testing of integrated modules to verify combined functionality after integration. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

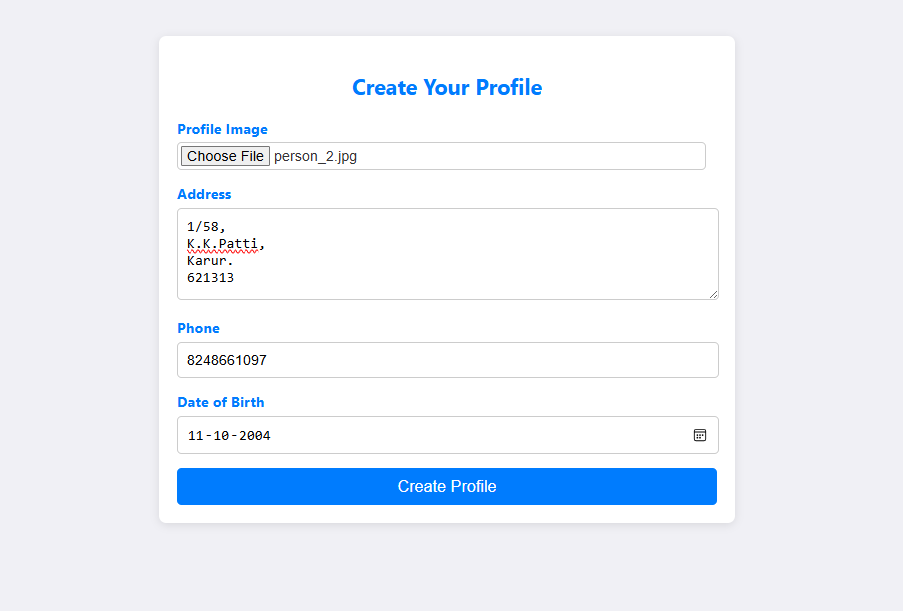


Fig:5.2 Register Account Page

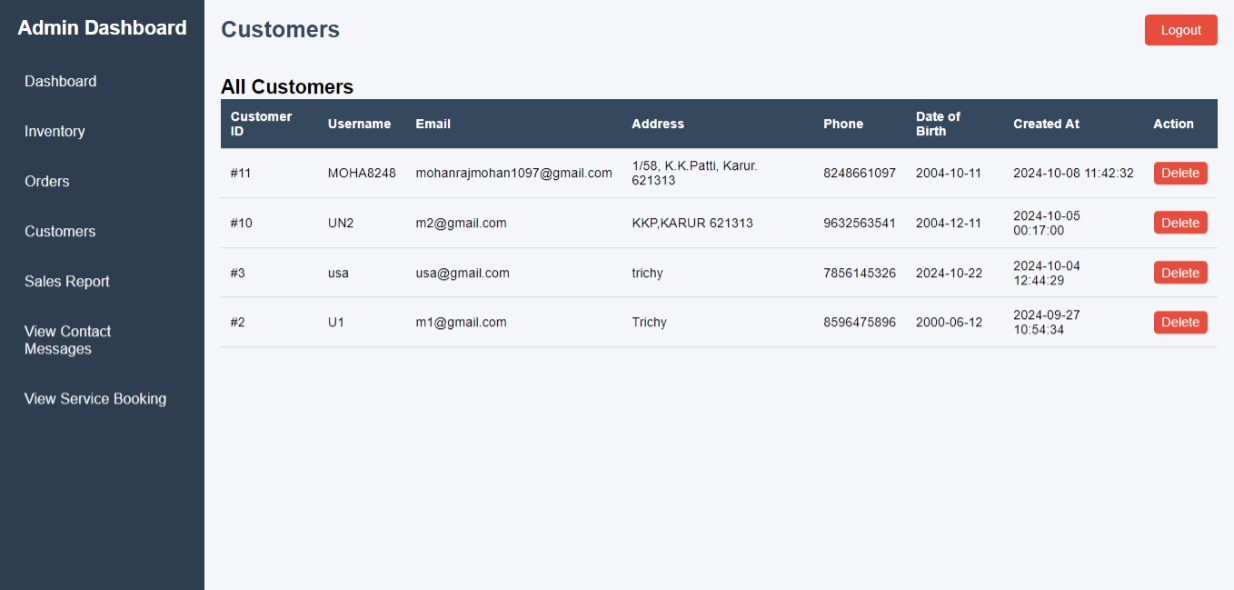
****

Fig:5.3 View Customer Details Page

### **5.3. Validation Testing:**

The process of validating a software product, or determining if it meets high level criteria, involves determining whether it is up to part. It is the procedure used to verify that the product we are producing is the proper one. The actual and anticipated product are being validated.

Testing in motion is validation.

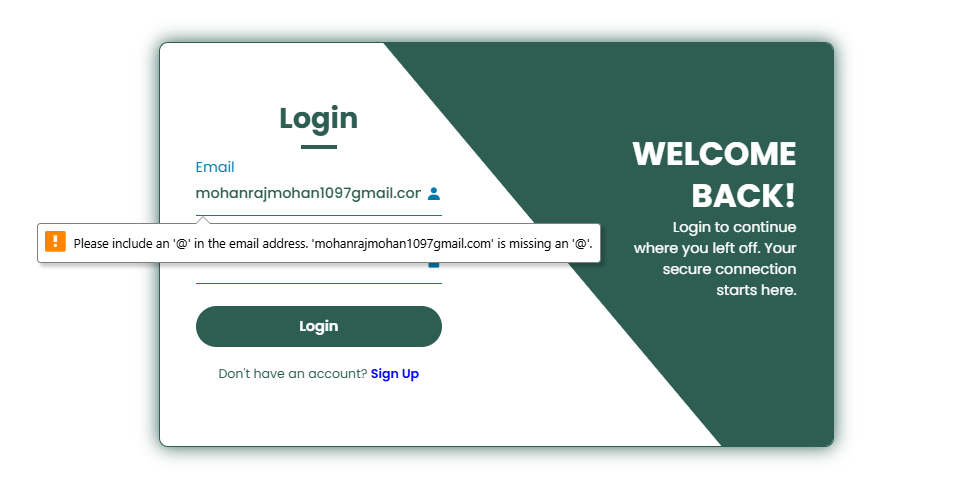


Fig:5.4 Validation Testing on Login

**6.SYSTEM IMPLEMENTATION**

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus, it can be considering the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

**6.1 SOFTWARE DEMONSTRATION**

**6.1.1 Home Page**

The home page of the Car Accessories Shop Management System offers a user-friendly interface for browsing products, accessing services, and discovering the latest deals.

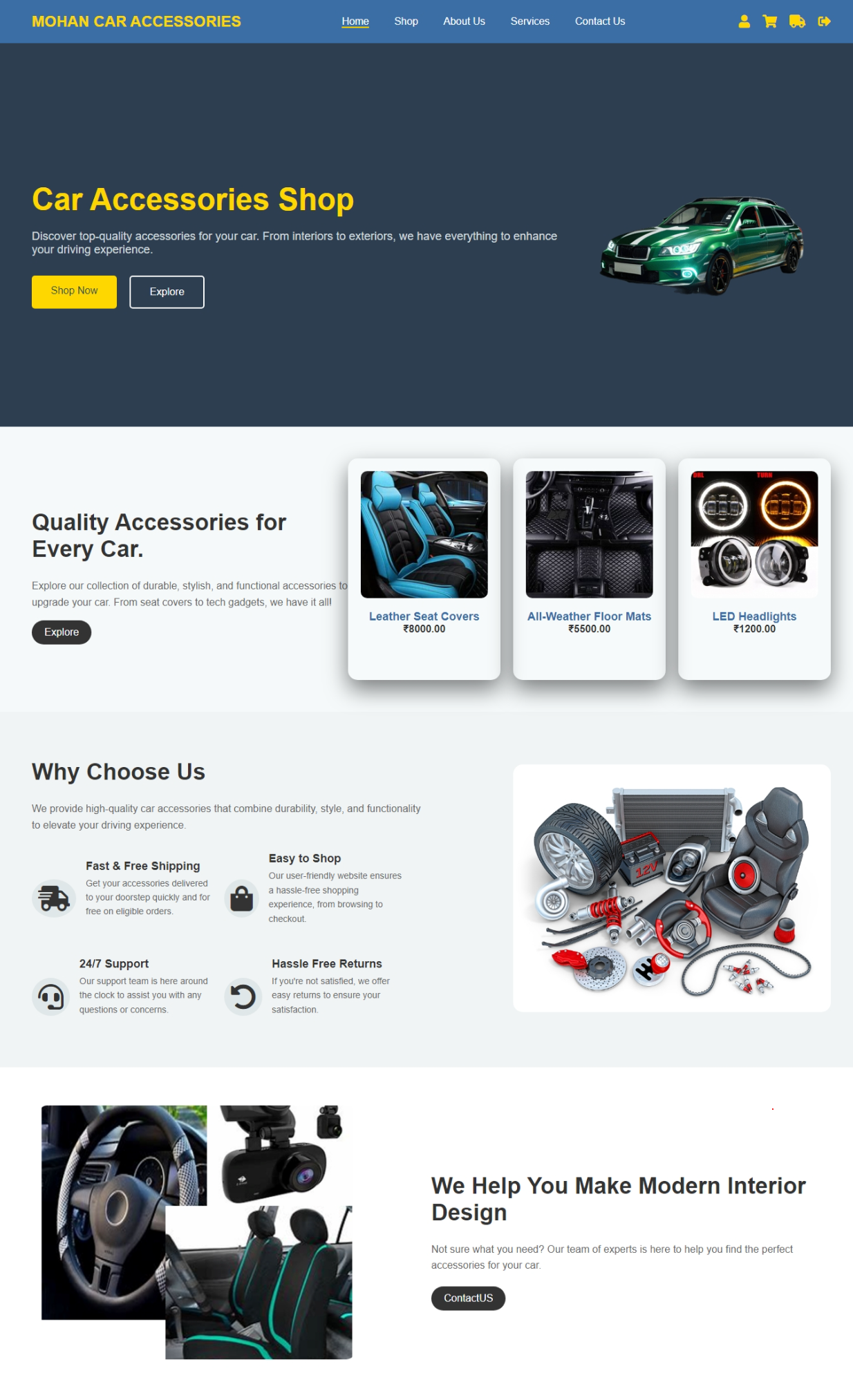


Fig:6.1 Home Page

**6.1.2 Login Page**

A user authentication interface allowing access to an website by entering credentials such as email and password.

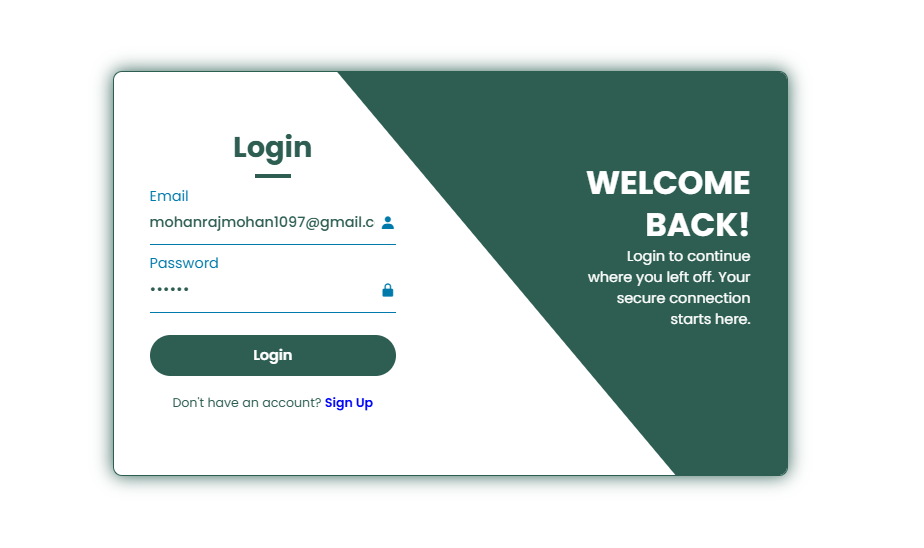


Fig:6.2 Login Page

**6.1.2 Signup Page**

An interface within the website enabling users to input email and password to signup.

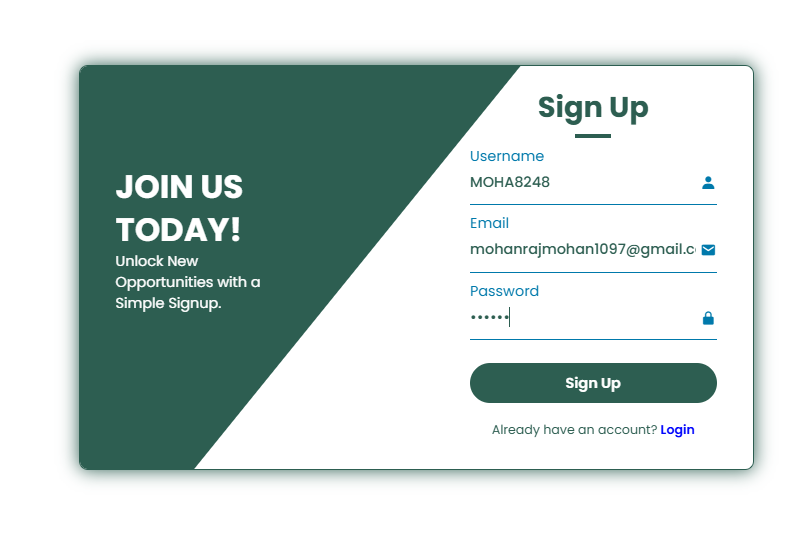


Fig:6.3 SignUp Page

**6.1.3 Dashboard**

The main screen of the Android app, serving as the entry point for users.It typically showcases essential features, content, or navigation options for easy access.

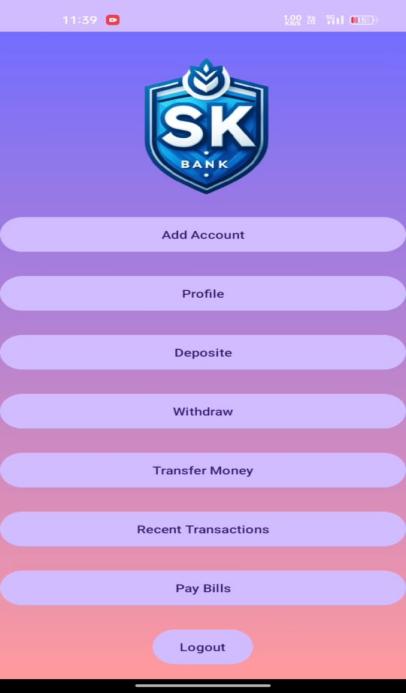


Fig:6.3 Dashboard

**6.1.4 RegisterAccount**

A page within the Android app where users can input information to register and create a new account.It typically collects personal details and credentials necessary for account creation..

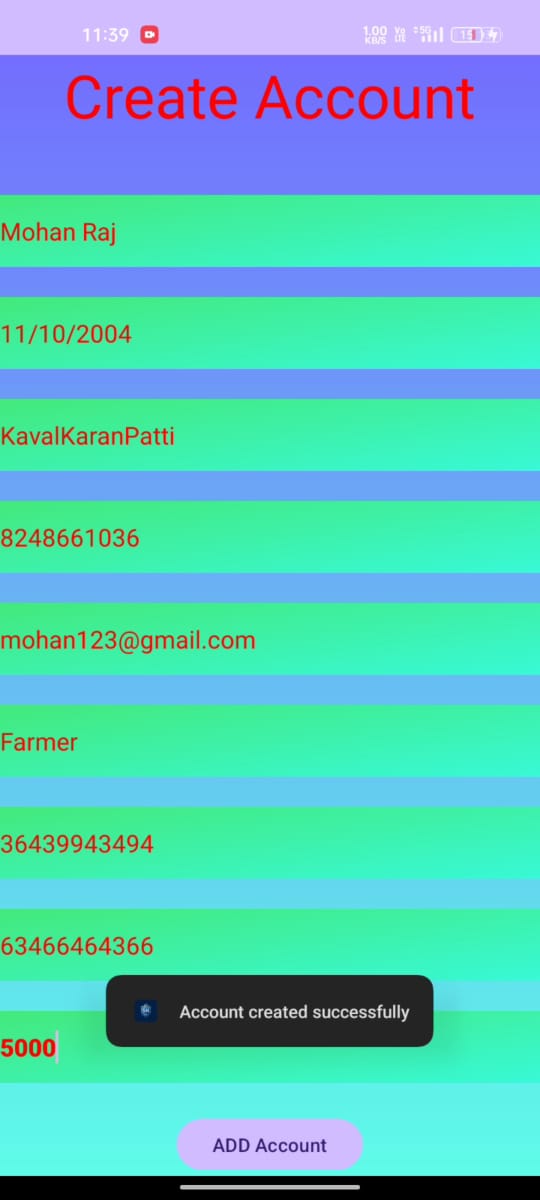


Fig:6.4 Register Account

**6.1.5 Profile**

The page displays a user's personal information, such as their name, accountno., bio, and contact details. It serves as a hub for users to manage their settings, preferences, and interactions within an application.



Fig:6.5 Profile

**6.1.6 DepositAmount**

The deposit amount page allows users to input and confirm the amount of money they want to deposit into their account.



Fig:6.6 Deposite

**6.1.7 WithdrawAmount**

The withdraw amount page allows users to input and confirm the amount of money they want to withdraw into their account.



Fig:6.7 Withdraw

**6.1.8 TransferMoney**

The transfer money using account number page facilitates users in sending funds to another account by inputting the recipient's account number or phonr number and specifying the transfer amount.



Fig:6.8 Transfer Money

**6.1.9 RecentTransactions**

The recent transaction page displays your latest banking activities, providing a clear overview of your financial transactions. It allows you to track your spending, monitor deposits, and stay informed about your account's activity.



Fig:6.9 Recent Transactions

**6.1.10 PayBills**

The pay bills page streamlines bill payment, enabling quick and efficient settlement of obligations. It offers a convenient platform to schedule payments, manage recurring expenses, and maintain financial organization.



Fig:6.10 Pay Bills

**6.1.11 ViewCustomerDetails**

The view customer details page offers comprehensive insights into client information, facilitating personalized service." "It provides a consolidated view of account status, transaction history, and contact details for efficient customer management.



Fig:6.11 View Customer Details

**6.1.12 View TransactionDetails**

The view transaction details page offers a detailed breakdown of individual transactions, including , amount, and sender-id. It provides a comprehensive overview of account activity, empowering admin to track expenses and reconcile their finances.



# **CONCLUSION**

In the **Car Accessories Shop Management System**, the platform provides a user-friendly interface that ensures efficient management of shop services, offering convenience for both customers and administrators. The admin can oversee product inventory, manage customer orders, and track sales effectively. Customers can easily register, view product listings, add items to their cart, place orders, and view their purchase history. The platform also ensures secure and seamless transactions. Future improvements will focus on expanding functionality and further enhancing the overall user experience.

**BIBLIOGRAPHY**

**BOOK REFERENCES:**

1. Robin Nixon, “Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5”, 2021.
2. Luke Welling, Laura Thomson, “PHP and MySQL Web Development”, 2016.

**WEBSITE REFERENCES:**

1. <https://www.php.net/manual/en/index.php>
2. https://www.w3schools.com/php/
3. <https://www.mysql.com/>