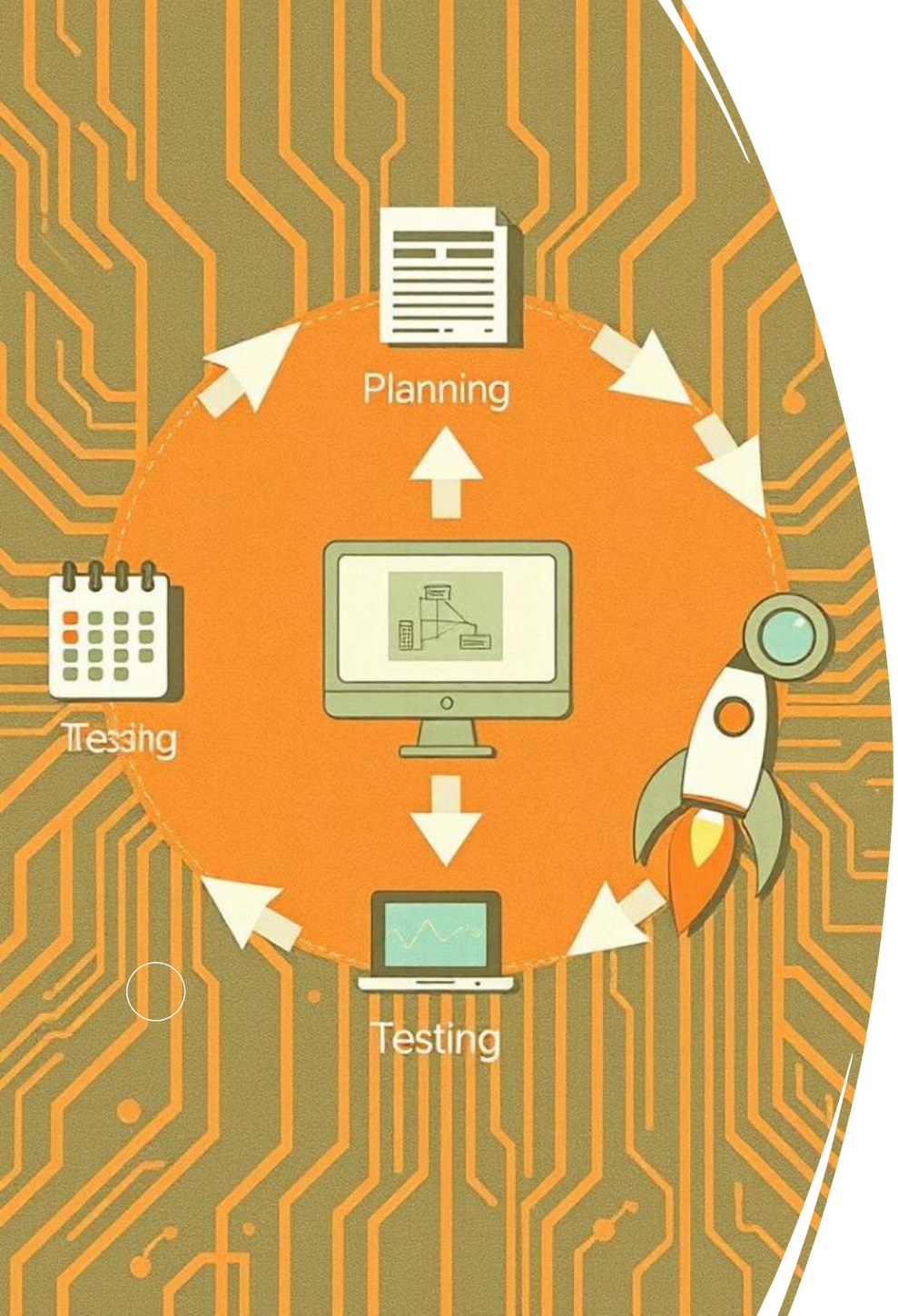


Auto Parts E-Commerce Project: Software Development Lifecycle

Welcome to the Auto Parts E-Commerce project! This presentation outlines our strategic approach to developing a cutting-edge platform for buying and selling automotive parts. We emphasize a structured and collaborative methodology to ensure the system is user-friendly, secure, and efficient. Our goal is to provide a seamless experience for both administrators and customers, meeting the dynamic needs of the automotive industry.

GOAL OF THE PROJECT

- Develop an e-commerce website specializing in providing auto parts, with the aim to:
  - Help users easily search, compare, and purchase auto parts.
  - Expand the business's accessibility to customers nationwide or internationally.
  - Enhance the efficiency of order management, inventory control, and customer support services.

Identifying Stakeholders

Customers

Browse products, place orders, manage accounts

Administrators

Manage products, user accounts, and orders

Developers

Build and maintain the website, ensuring functionality and security

FUNCTION REQUIREMENT

- Both buyers (users) and admins can register and log in to the system.
- Users can browse products and add or remove items from their shopping cart.
- Users can proceed to the checkout process to place orders after finalizing their cart.
- Users can view a history of their past orders.
- Admins can add, edit, and delete products in the shop.
- Admins can view, process, and track orders made by users.
- Admins can access and analyze historical data for performance evaluation.
- Admins can create, update, or remove user accounts.

NON-FUNCTION REQUIREMENT

- The system must respond within 3 seconds for each user action.
- The interface should be easy to use, user-friendly, and support desktop/mobile.
- The system must operate 24/7 with 99.9% uptime.
- All user information must be securely stored, and payment data should be encrypted.
- The system must handle 10,000+ simultaneous users.
- Ensure data consistency across all transactions, especially for inventory and order processes.
- Provide clear error messages and ensure recovery from failures without data loss.
- Track system performance to ensure optimal load management and uptime compliance.

TECHNOLOGY USED

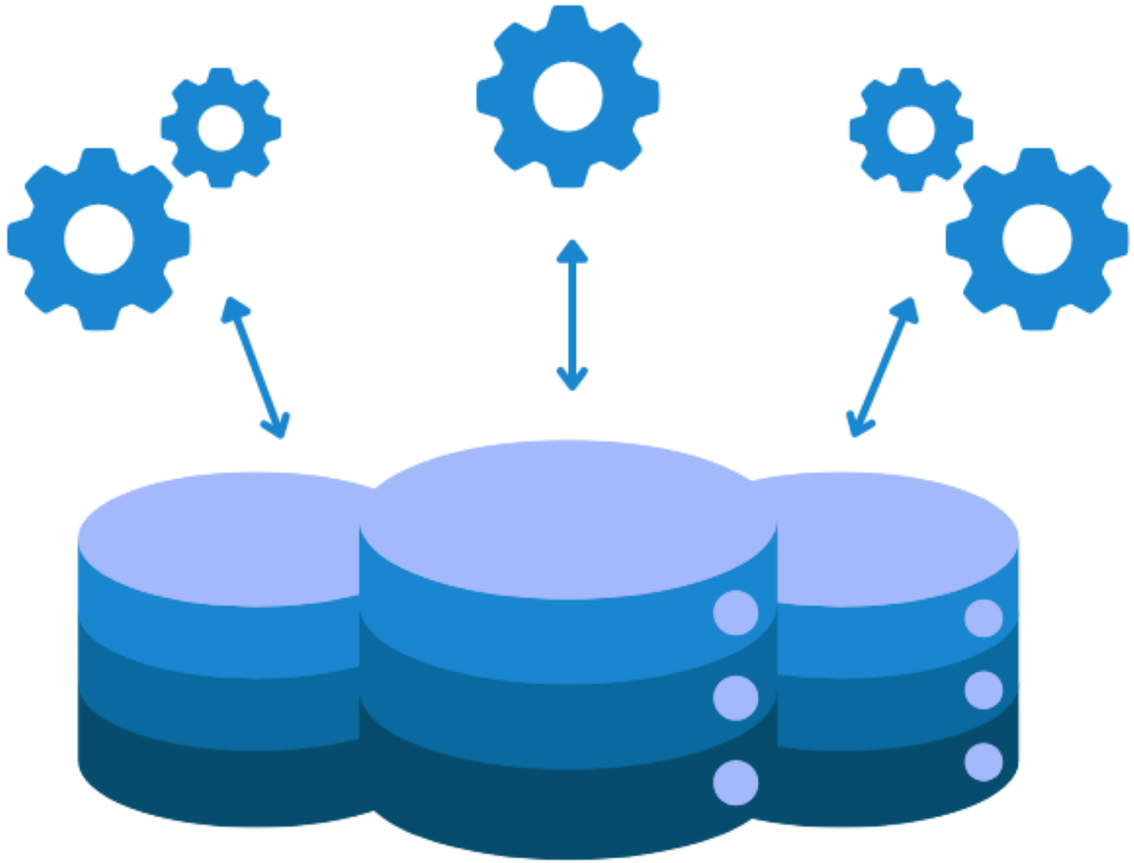
1) Backend: PHP

2) Frontend: HTML, CSS, JavaScript

3) Database: MySQL.

4) Server: Apache on Laragon.

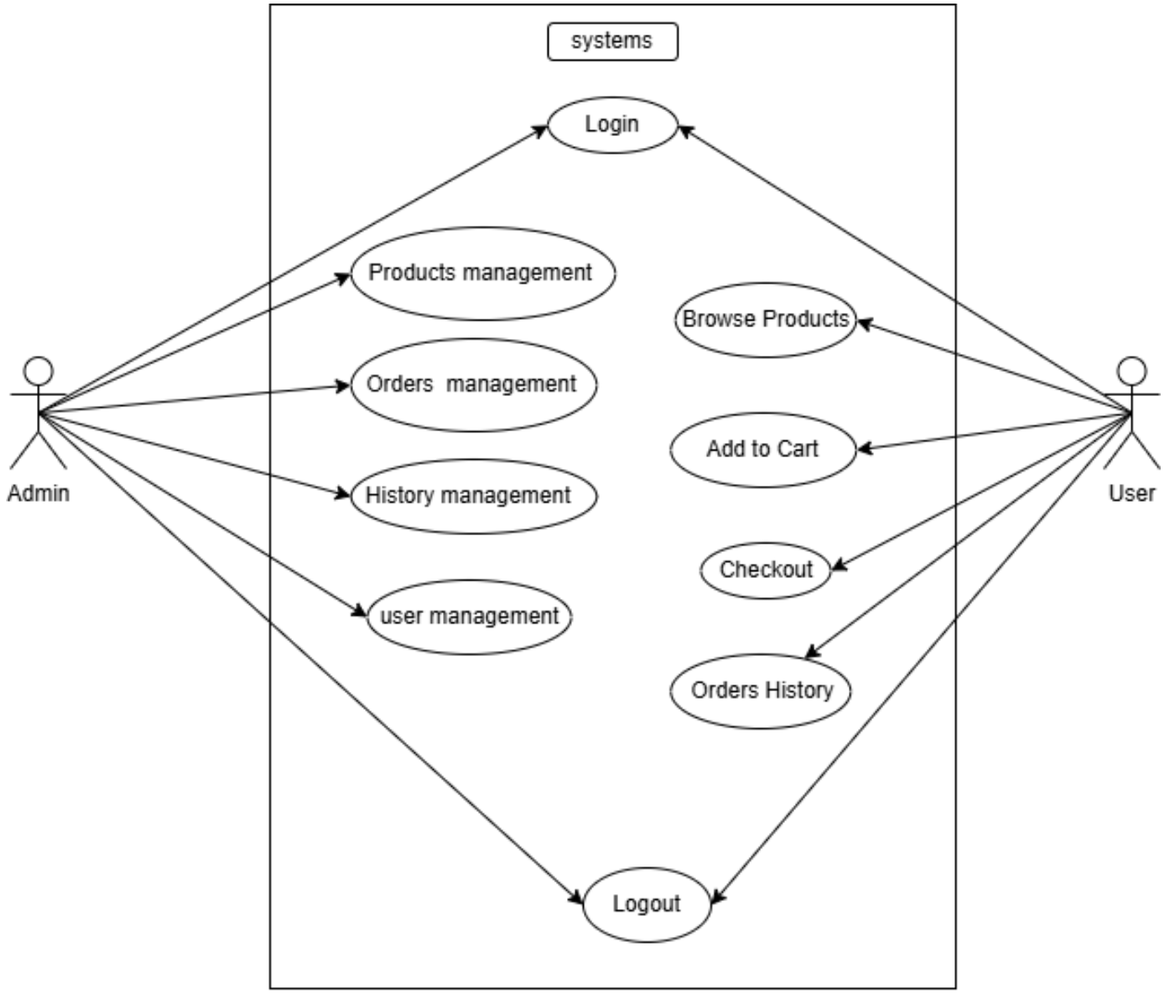
SYSTEM DESIGN



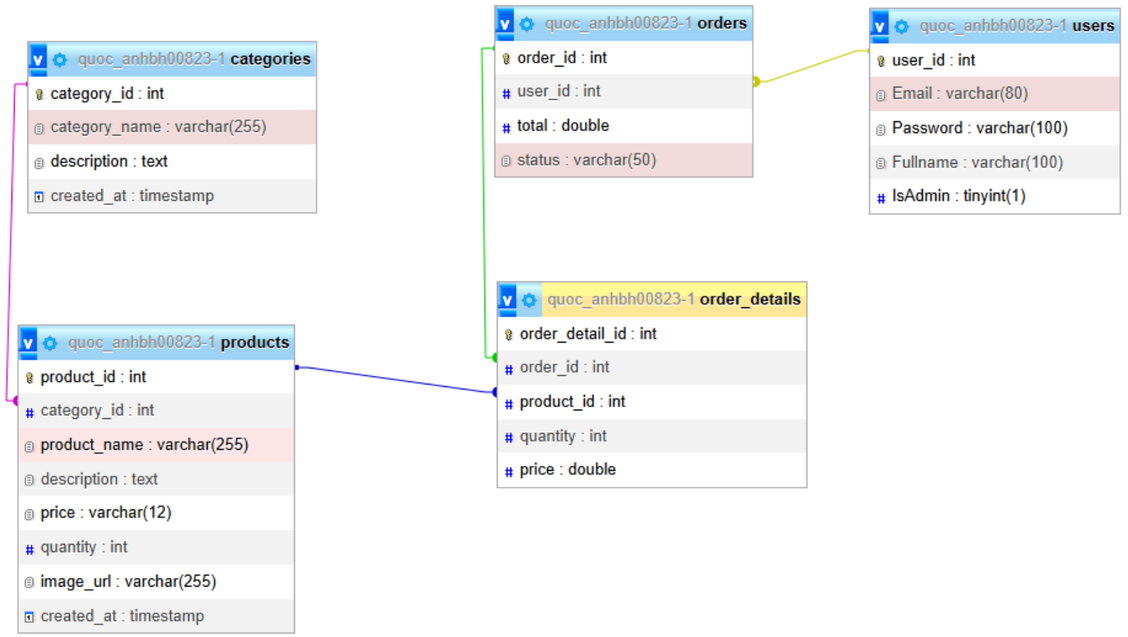
Use Case Diagram

- **Actor Description:**

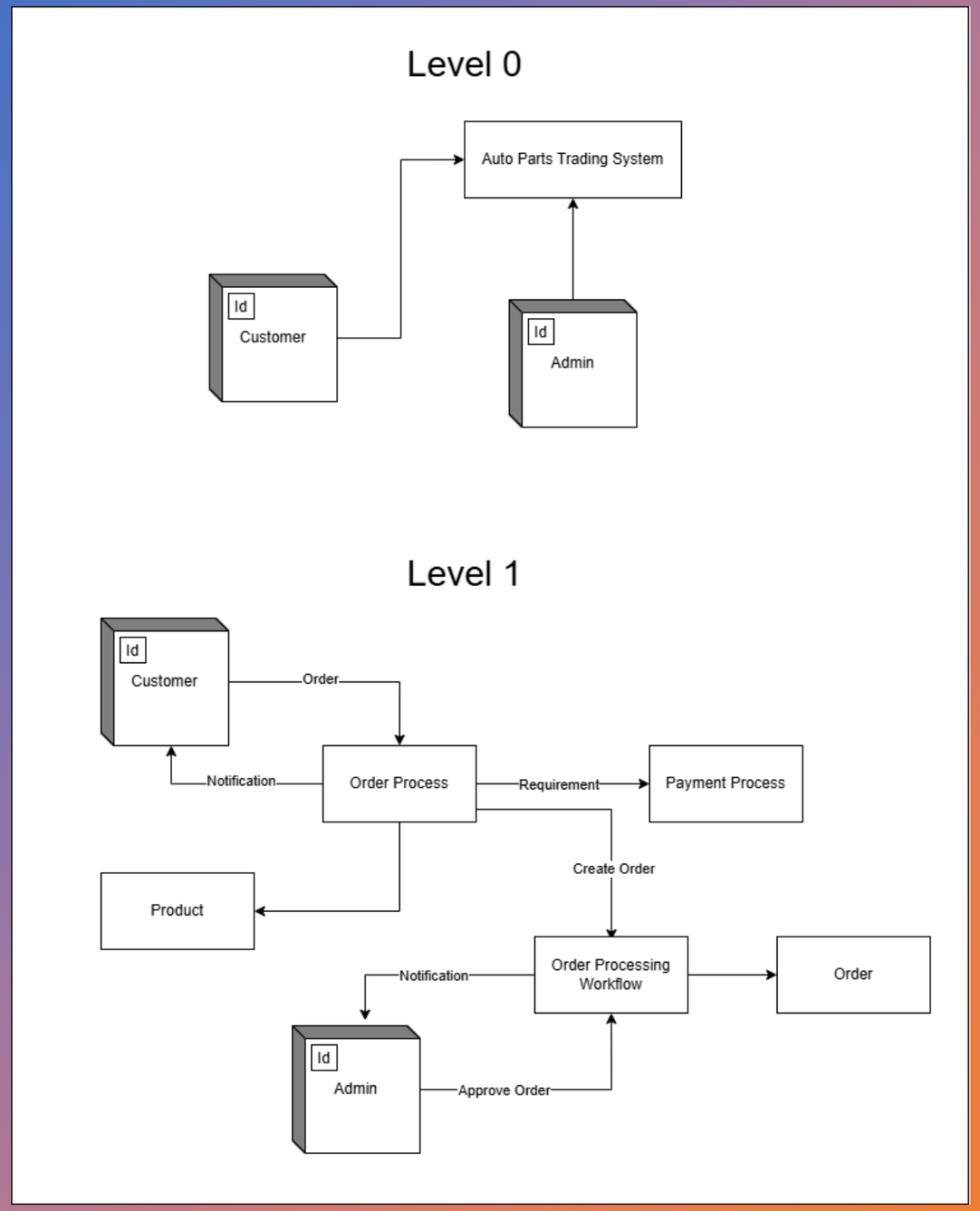
- **Admin**: Manages the system, has full control over products and orders.
- **User**: Regular customer, can browse products, add them to the cart, and proceed to checkout.



Entity-Relationship Diagram (ERD)



- This ERD demonstrates a basic yet complete data structure for an e-commerce system:
  - **Category management**: Organize products into categories for easier searching and classification.
  - **User management**: Differentiate between regular users and admins.
  - **Order management**: Store detailed information about each order, including the products, quantities, and prices.



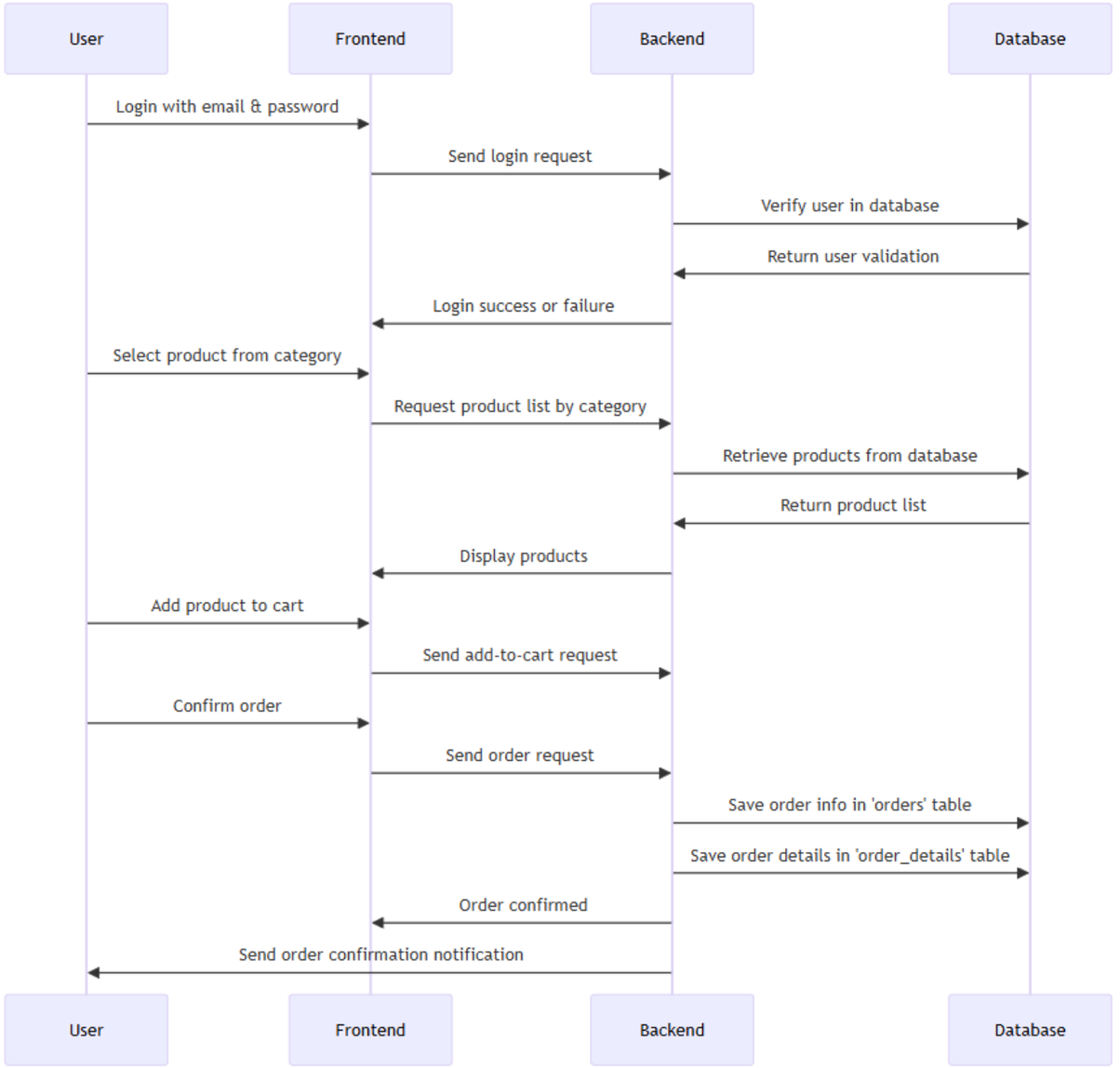
Data Flow Diagram (DFD)

- **Observations:**
  - **Level 0** provides a high-level view of the main entities and their simple relationships.
  - **Level 1** goes into more detail, showing processes like order creation, requirement handling, payment, and order approval.
  - The **relationships between entities** such as **Customer**, **Admin**, **Order Process**, and **Payment Process** are clear, but more clarification is needed about how notifications and data handling occur between steps.

web project for buying and selling auto parts (Sequence Diagram):

Purpose of the Diagram:

The sequence diagram outlines the flow of interactions between the **User**, **Frontend**, **Backend**, and **Database** for a car spare parts e-commerce system. It focuses on the user's journey from login to placing an order, showcasing the system's behavior and interactions.

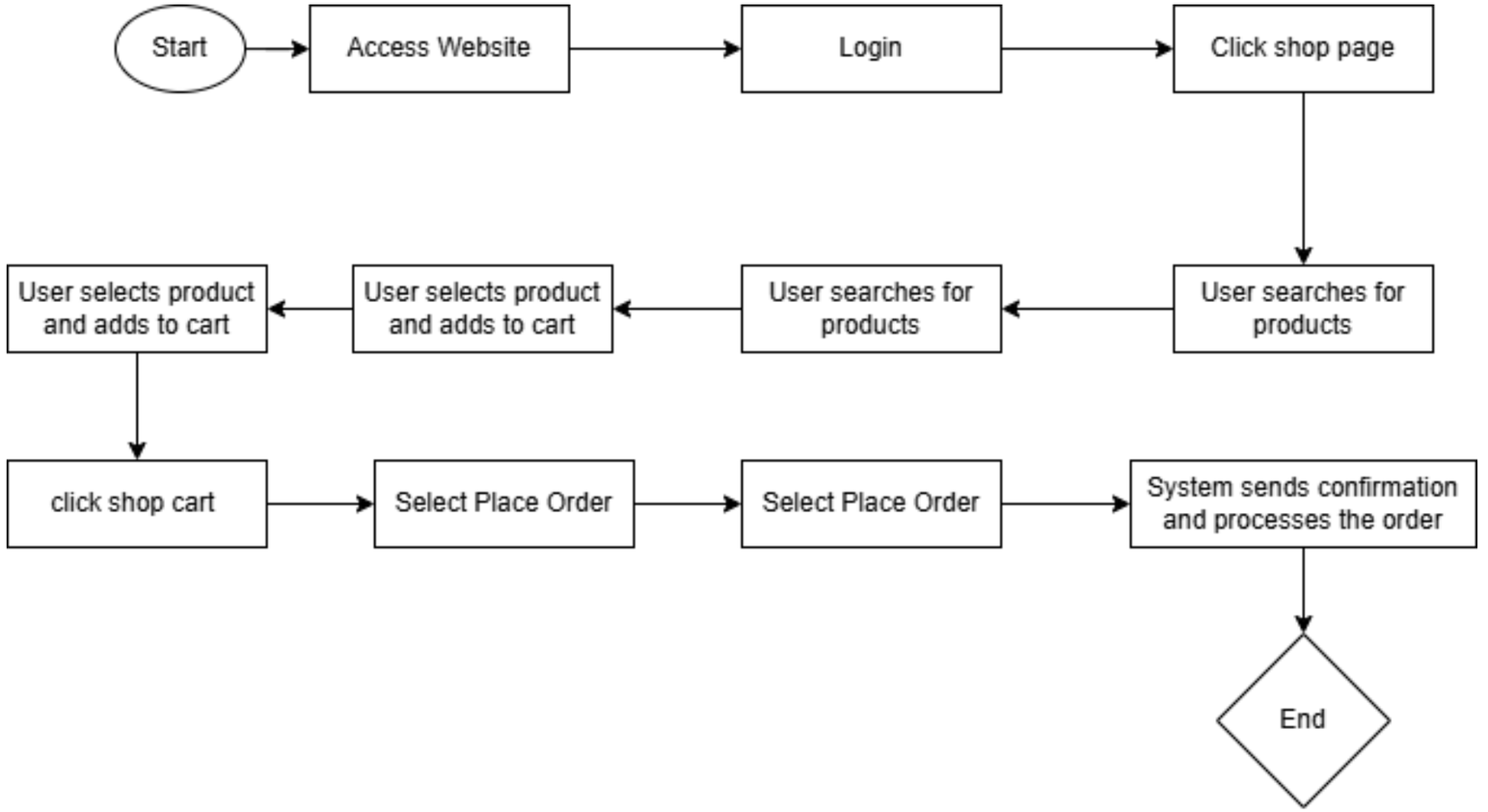


web project for buying and selling auto parts (Sequence Diagram):

- **Analysis of the Sequence Diagram:**
  - **Main Purpose:**
    - Represents the process flow when a **user** logs in, searches for products, adds items to the cart, and places an order.
  - **Key Components:**
    - **User:** Performs actions like logging in, selecting products, and confirming an order.
    - **Frontend:** Sends requests and displays results to the user.
    - **Backend:** Handles business logic and serves as the connection between the frontend and database.
    - **Database:** Stores information about users, products, orders, and order details.

- **3. Process Flow:**
  - **Process Flow:Login:** Validates login credentials via the database.
  - **Product Search:** Retrieves the product list from the database.
  - **Add to Cart:** Backend processes the request to add products to the cart.
  - **Order Confirmation:** Saves order details in the database and sends a confirmation notification.

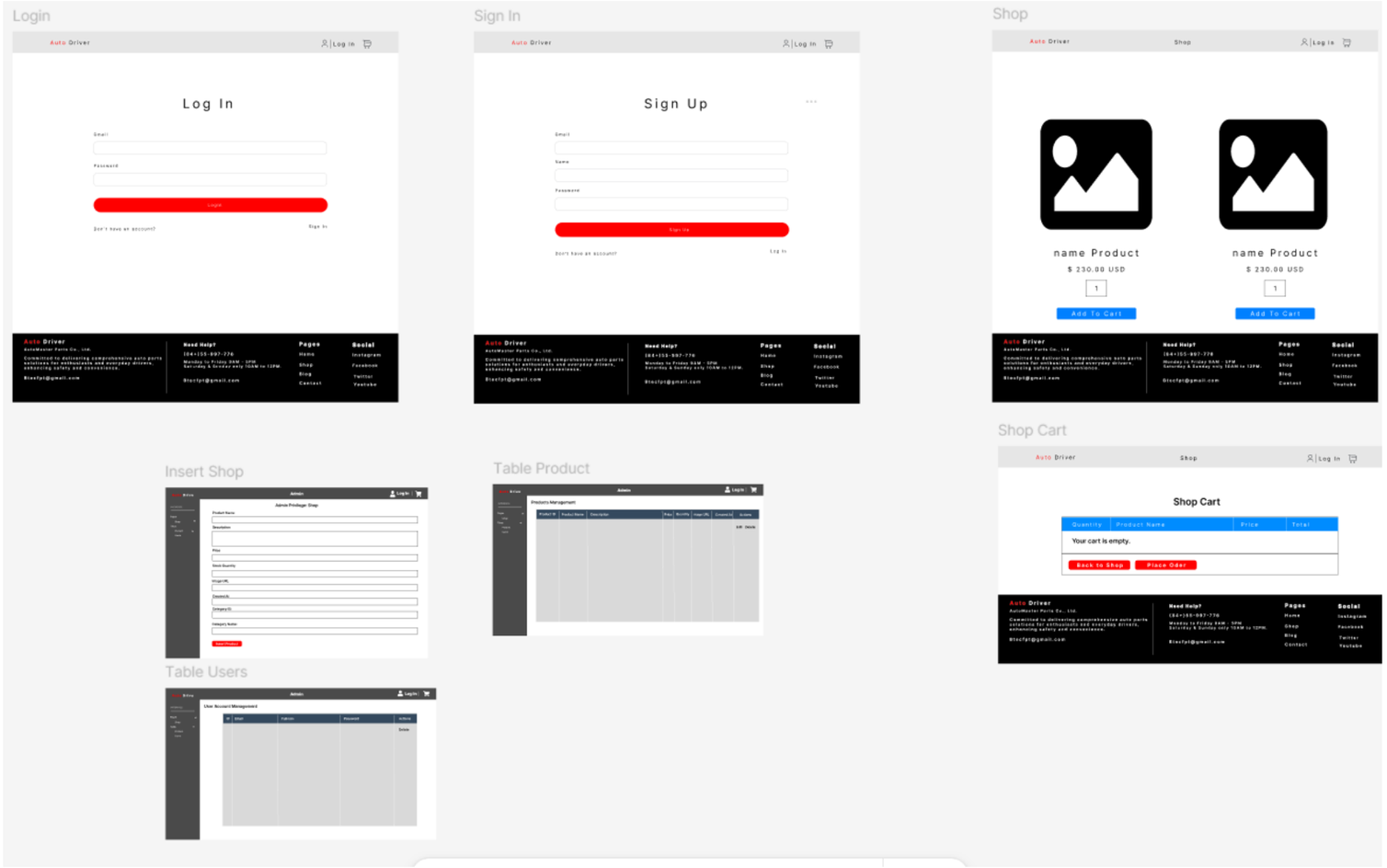
FLOW CHART


Start
Access Website
Login
Click shop page
User searches for products
User searches for products
User selects product and adds to cart
User selects product and adds to cart
click shop cart
Select Place Order
Select Place Order
System sends confirmation and processes the order
End

FLOW CHART

- **Main Flow**:
- Starting from accessing the website, the user searches for products, adds them to the cart, and finally places the order.
- The system responds by confirming the order and completing the transaction.
- **Possible Improvements**:
- The flowchart does not detail the payment process or shipping method selection. These could be added for greater completeness.
- Handling cases where users are not logged in (e.g., creating an account or guest checkout) should be considered.
- **Automation Opportunities**:
- The system could suggest products automatically based on search history.
- Real-time stock availability information could be provided to enhance user experience.

Wireframe Design



Wireframe Design

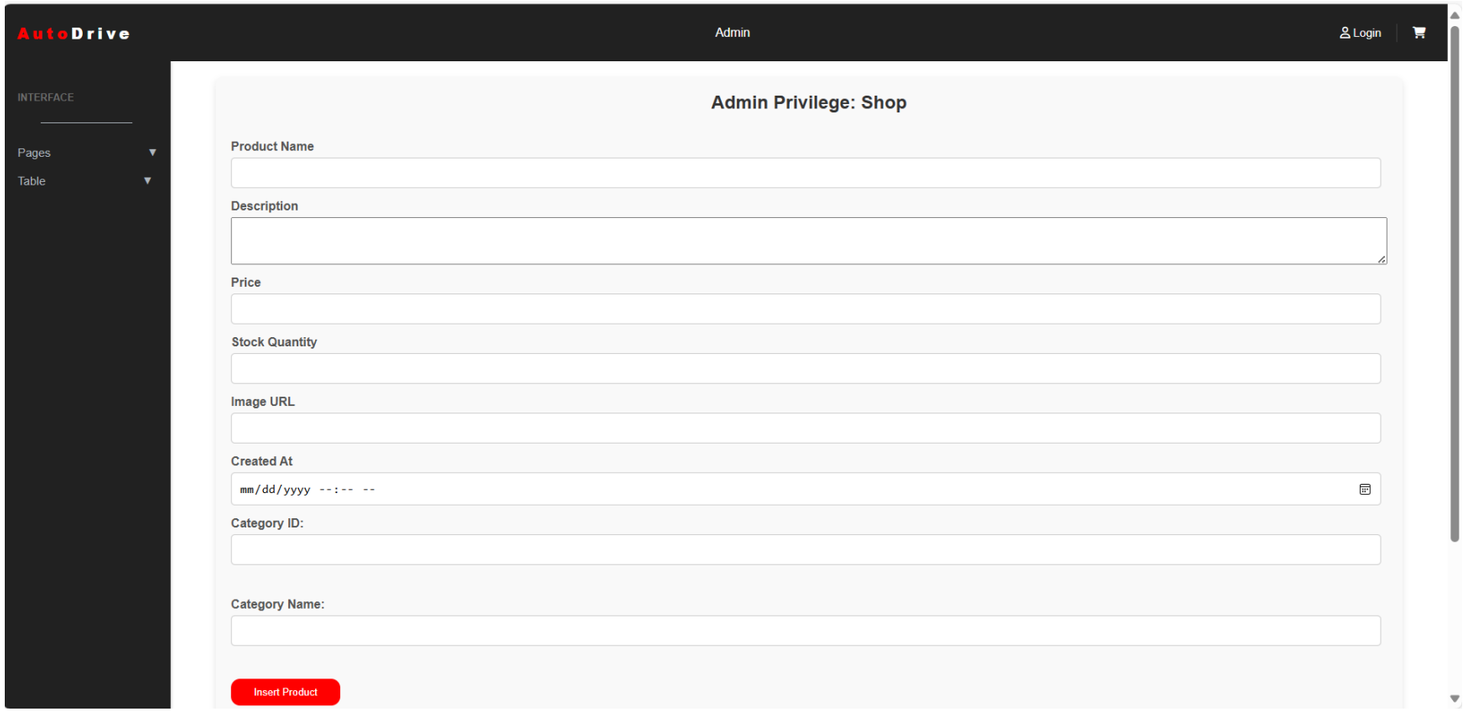
1. Overview of the Interface

The prototype consists of the following key components:

- **Login:** A page where users enter their email and password.
- **Sign Up:** A page where users can create a new account.
- **Shop:** Displays a list of products.
- **Shop Cart:** A shopping cart page that allows users to manage their added products.
- **Insert Shop, Table Product, and Table Users:** Management sections for administrators or data entry personnel.

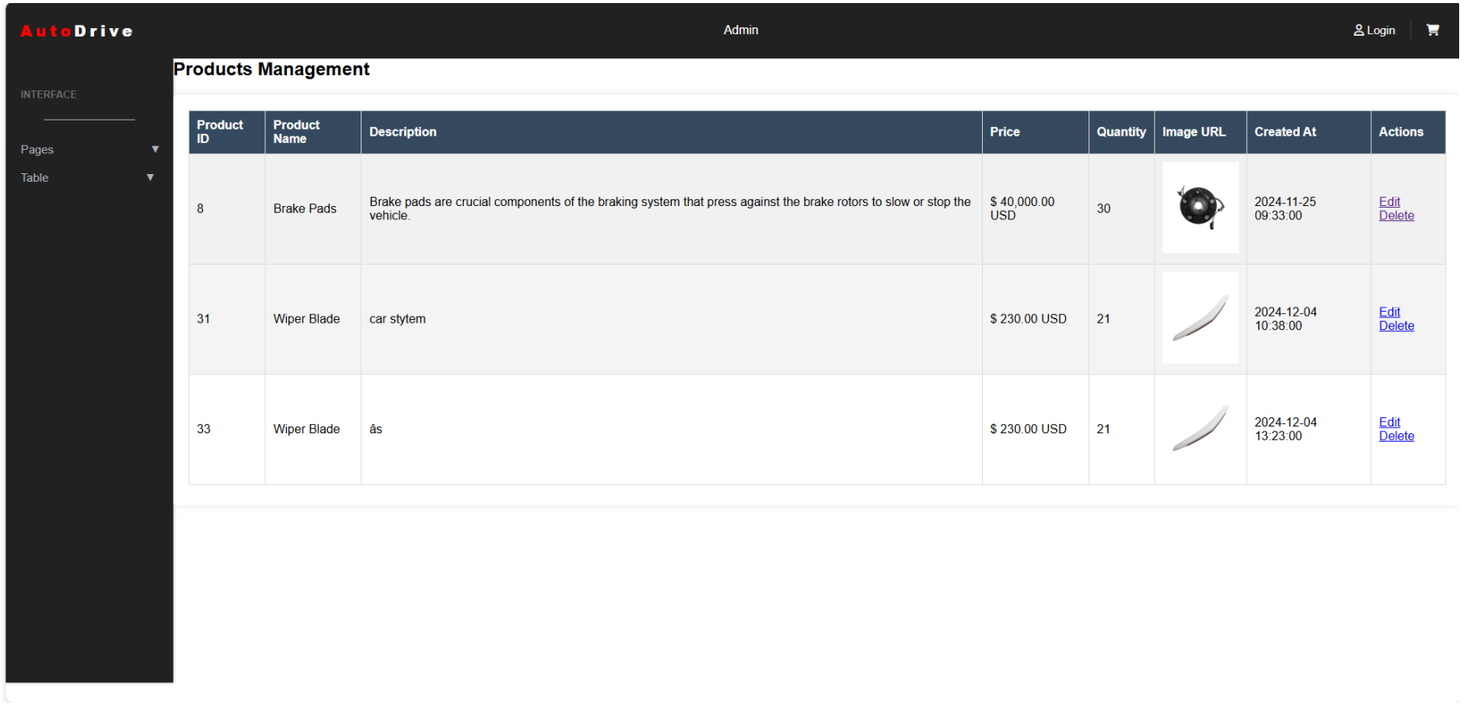
software development

software development(Insert Products)



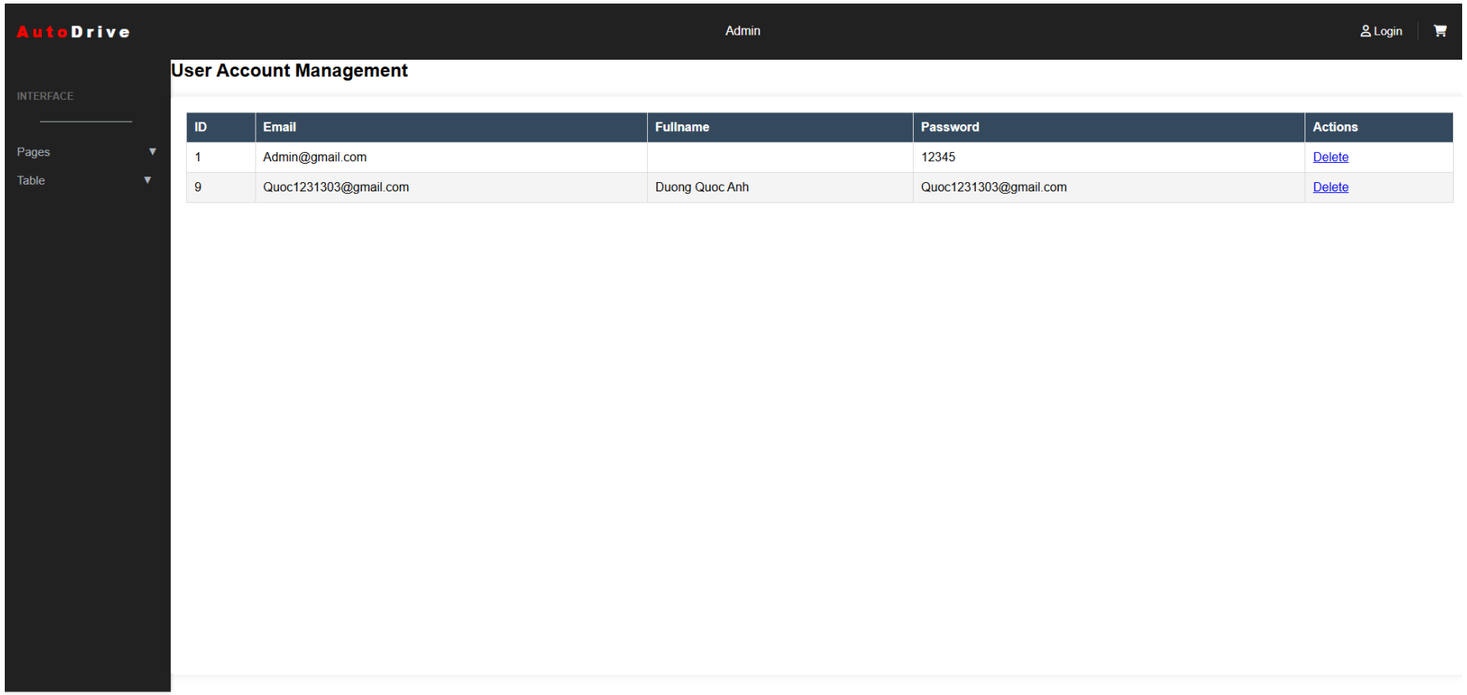
- **Main Function:**
  - A page for admins to add new products with fields: product name, description, price, stock quantity, image URL, creation date, category ID, and category name.

software development (Products Manager)



- **Analysis of "Products Management" Interface:**
  - **Main Function:**
    - A product management page displaying a table of products with columns: **Product ID**, **Product Name**, **Description**, **Price**, **Quantity**, **Image URL**, **Created At**, and **Actions**.
    - Two main actions: **Edit** and **Delete**.

software development (user manager)

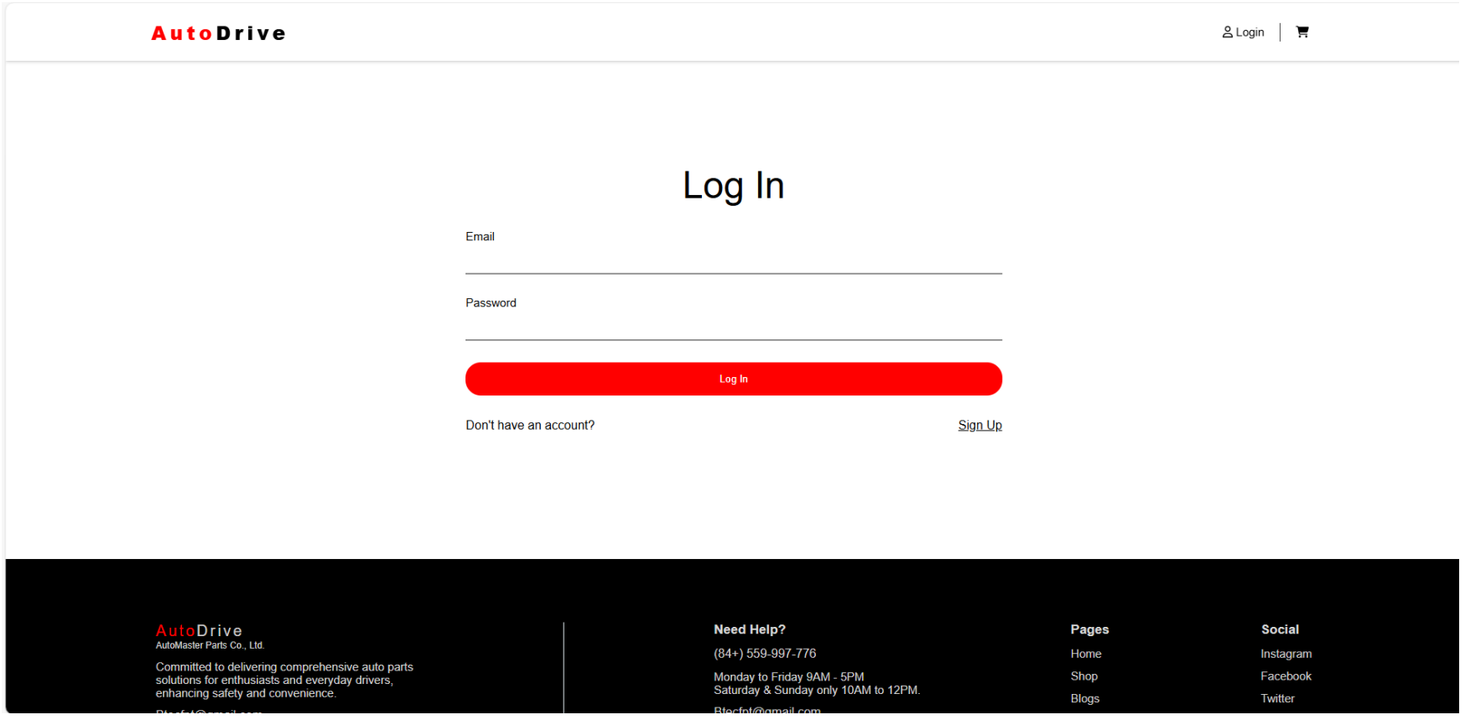


- Title: "User Account Management" represents the functionality of managing user accounts.

Data columns:

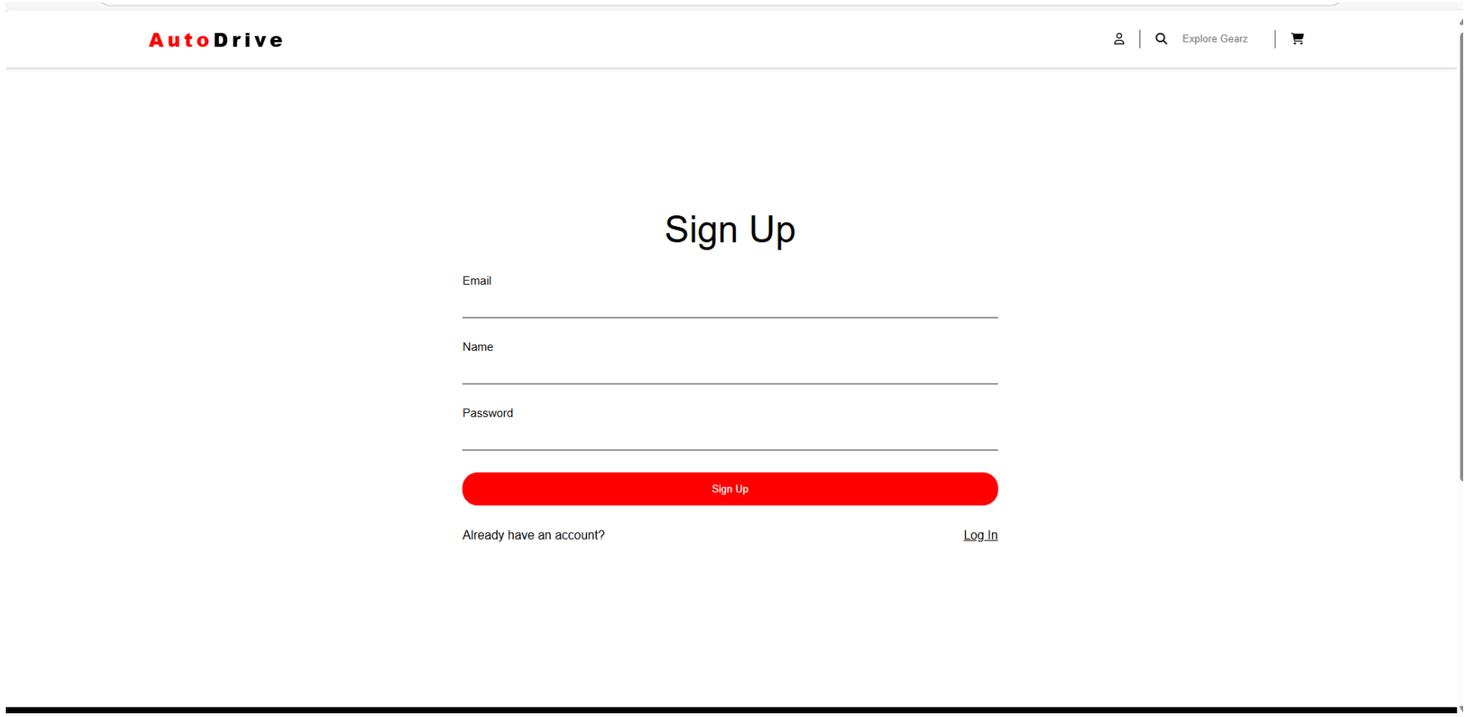
- **ID:** The unique identifier for each user.
- **Email:** The email address associated with the account.
- **Fullname:** The full name of the user.
- **Password:** The account password (displayed in plain text, which is insecure).
- **Actions:** Operations such as deleting an account (Delete).

software development (login pages)



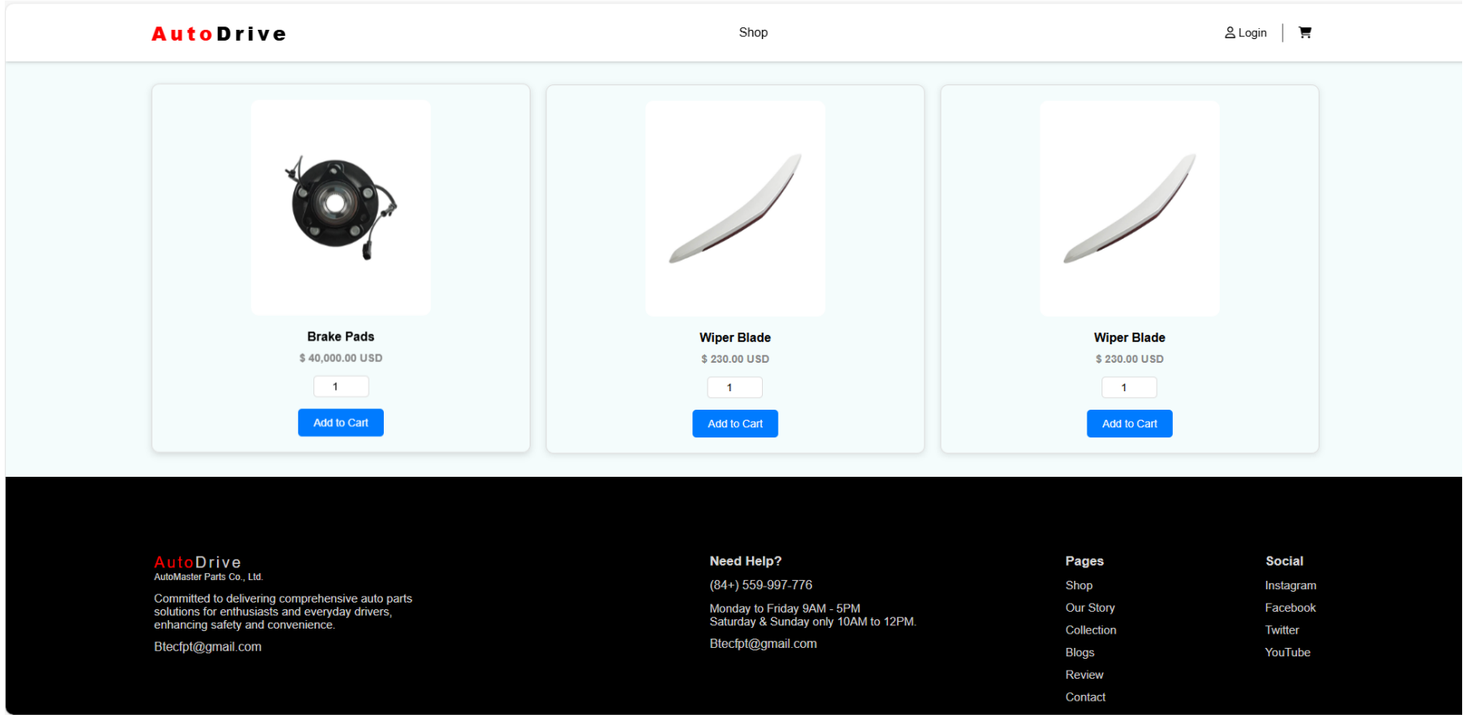
- **Analysis of "Log In" Interface:**
- This is the login page where users can log in to the website. Additionally, the page includes a role-based access feature for Admins. Admins can log in to the admin page to manage products and users.

software development (Sign pages)



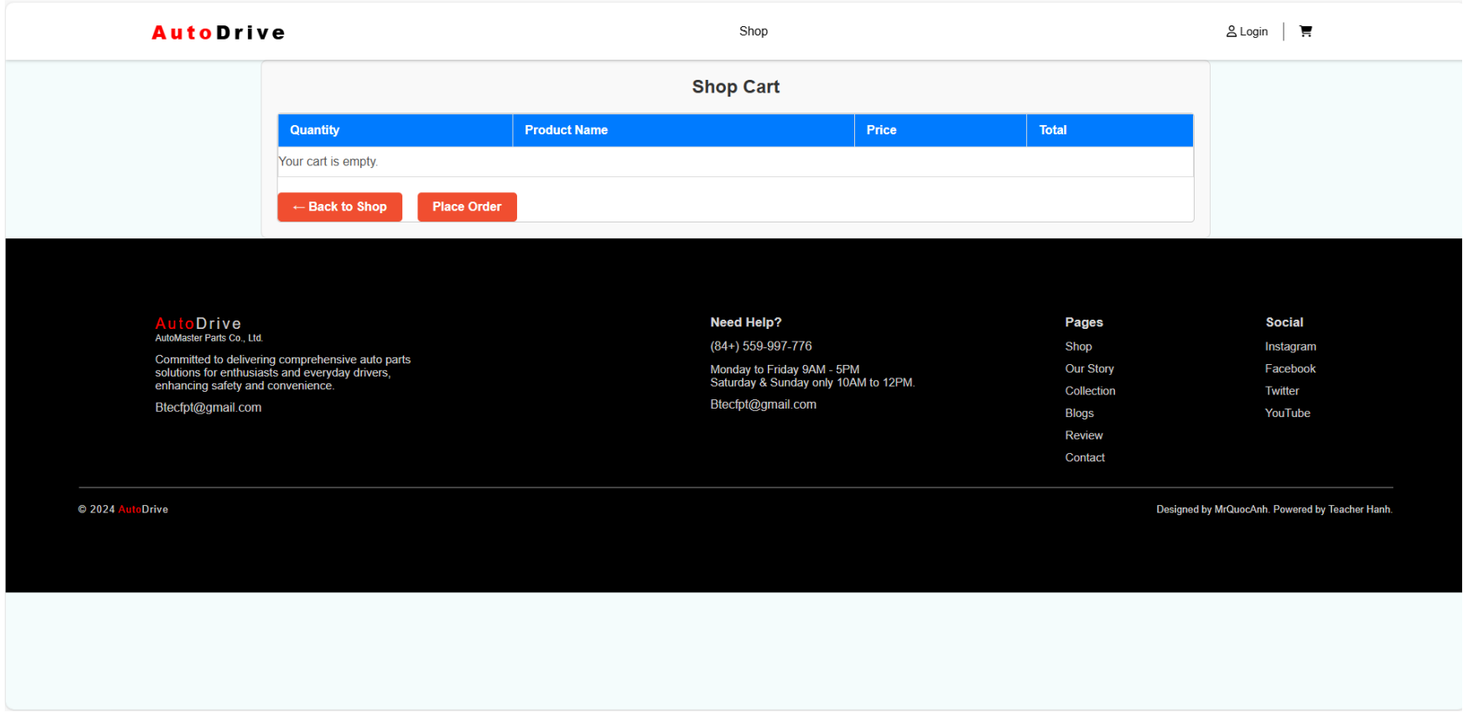
- **Analysis of "Log In" Interface:**
- This is a registration page where users can create accounts, and the accounts will be stored in the database.

software development (Shop pages)



- **Analysis of "Shop pages" Interface:**
- This is the shop page where users can browse, search, and purchase products.

software development (Cart pages)



- **Analysis of "Shop Cart" Interface:**
- After users click 'Add to Cart,' the item will be added to their shopping cart. Users can view their cart and click 'Place Order' to make a purchase.

Auto Parts E-Commerce Project Software Development: Quality Assurance & Testing

Software Testing: A Cornerstone of Quality

Functional Testing

This involves examining fundamental system features like part search, search, payment processing, login, and shopping carts, ensuring ensuring accurate operation.

User Interface Testing

Ensures proper functionality and user-friendliness of the website across website across various browsers, providing a seamless experience for all experience for all users.

Testing for Performance and Security

Performance Testing

Evaluates system responsiveness under heavy user traffic, particularly particularly during payment and part search processes, ensuring smooth ensuring smooth operation.

Security Testing

Identifies and mitigates potential security vulnerabilities like SQL SQL Injection and XSS, safeguarding user data and system integrity. integrity.

Benefits of Software Testing

Early Error Detection

Testing facilitates early detection and correction of errors during development, minimizing costly post-deployment fixes.

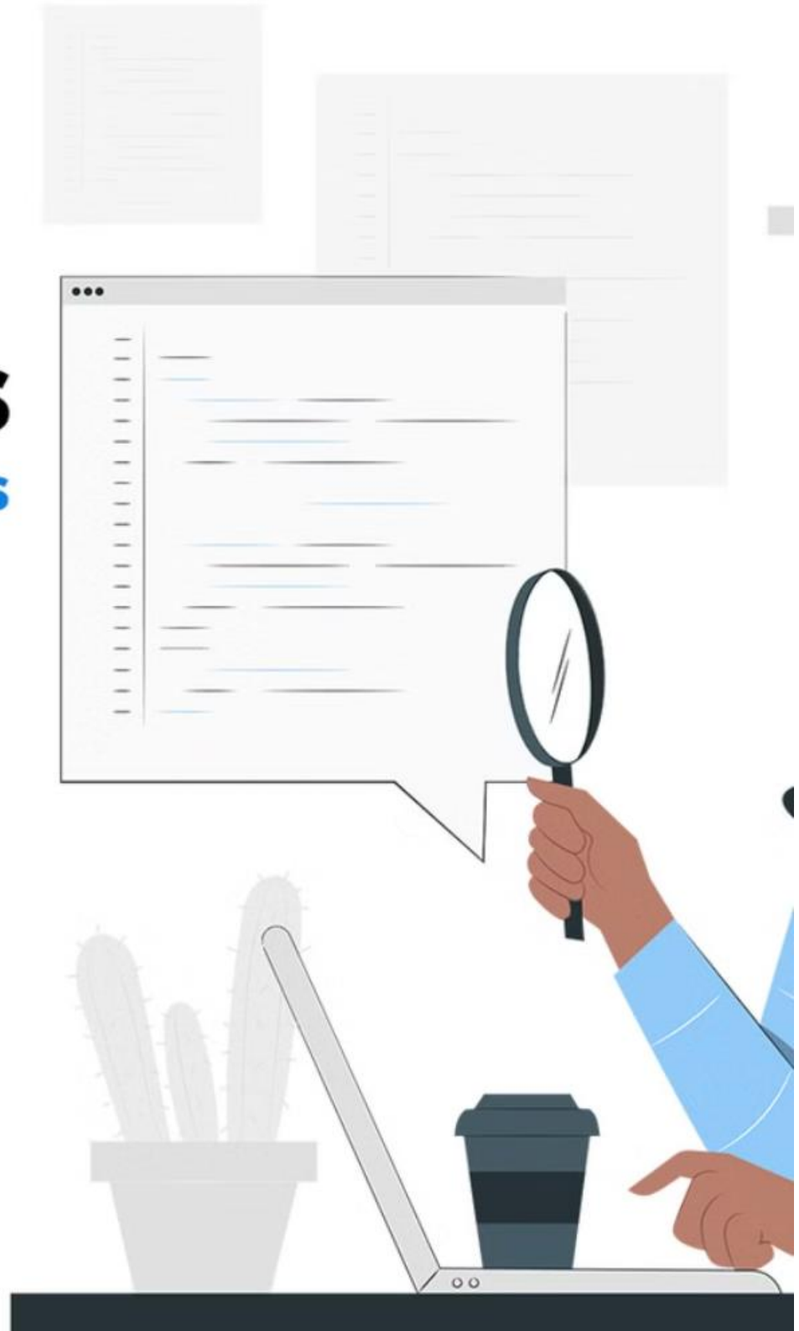
Enhanced Quality

Guarantees software functionality functionality adheres to requirements and quality standards, standards, delivering a robust and and reliable product.

Increased System Reliability

Thorough testing instills user confidence in vital services like payment processing processing and product search, boosting overall system reliability.

ons
views
y
ev



Approach 1: Code Reviews

Detects Errors Early

Peer reviews catch issues before they reach production, minimizing minimizing the risk of bugs affecting affecting users. This saves time and and resources during later stages. stages.

Improves Code Readability Readability and Maintainability

Code reviews ensure clean and well-well-documented code, making it it easier to understand and maintain. maintain. This contributes to long-long-term project success.

Promotes Knowledge Sharing

Peer reviews facilitate knowledge exchange within the team. Developers learn Developers learn from each other, enhancing overall skill levels and project project efficiency.

Benefits of Code Review: Improved Quality and Knowledge Knowledge Sharing

Early Error Detection

Code review helps identify errors overlooked overlooked during development, resulting in resulting in a higher quality product.

Enhanced Code Quality

Leads to optimized and maintainable code, code, reducing future errors as new features features are added.

Team Knowledge Sharing

Provides an opportunity for programmers to programmers to learn from each other, share share expertise, and improve their programming skills.

Low-code & Scripting mode	Both	Scripting Only	Scripting Only	Both	Scripting Only
Supported language(s)	Java & Groovy	Java, C#, Python, JavaScript, Ruby, PHP, Perl	Java, C#, Python, JavaScript, Ruby, PHP, Perl	JavaScript, Python, VBScript, JScript, Delphi, C++, C#	JavaScript

Approach 2: Automated Testing

Selenium

We use Selenium for automated UI testing, ensuring that the platform's platform's user interface functions correctly and meets user expectations.

Postman

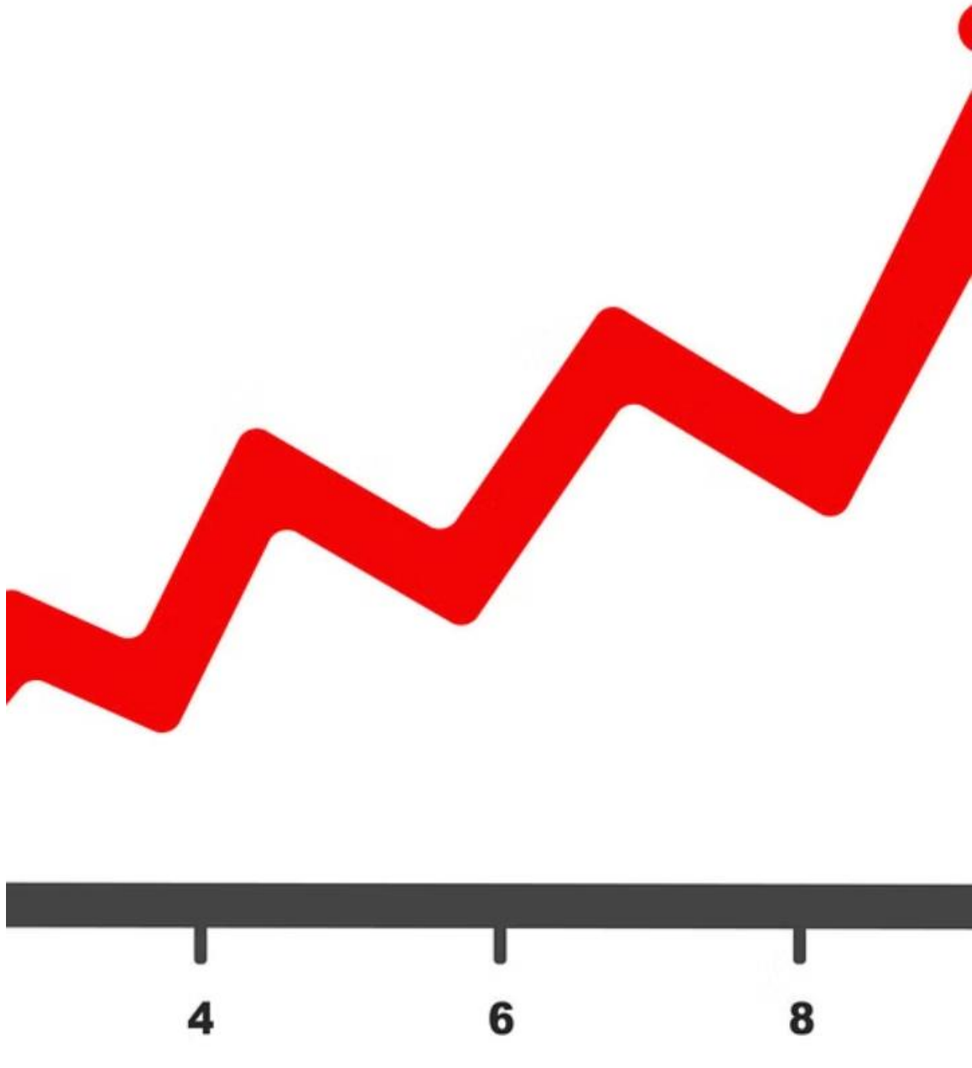
Postman is employed for API testing, verifying the communication and communication and data exchange between different components of our components of our platform.

Results of Quality Approaches Approaches

30%

Reduced Bug Rate

We've observed a 30% reduction in bugs detected during testing and production, production, demonstrating the effectiveness of our quality assurance strategy. strategy.



Conclusion and Next Steps

Code reviews and automated testing are crucial for delivering high-quality software. These practices ensure a reliable platform that meets user expectations. I am committed to continuously improving my quality assurance processes. I plan to expand my test cases, implement performance testing, and leverage monitoring tools to maintain a smooth and efficient platform.

CONCLUSION

- The automotive parts sales system operates efficiently, fully meeting the requirements set. The project successfully integrates a user-friendly interface with key functions such as shopping cart management, online payment, and user management. The system allows users to easily search for and purchase the necessary automotive parts while optimizing the payment and transaction process. Through this project, participants gain a better understanding of the synergy between frontend and backend development in building an e-commerce application, while also improving their skills in database management and system security.

THANKS FOR WATCHING