VIETNAM NATIONAL UNIVERSITY HCMC

INTERNATIONAL UNIVERSITY

_____

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Academic Year: Second semester 2024 – 2025

WEB APPLICATION DEVELOPMENT
E-commerce website for automotive parts

Member of group:

No.	Full name	Student ID	Email
1	Nguyễn Vũ Minh Thư	ITITWE21112	ITITWE21112@student.hcmiu.edu.vn
2	Nguyễn Trung Tín	ITITDK21061	ITITDK21061@student.hcmiu.edu.vn
3	Nguyễn Hải Quân	ITITWE21104	ITITWE21104@student.hcmiu.edu.vn

Instructor: Assoc. Prof. Nguyen Van Sinh

OUTLINES

I.   INTRODUCTION

1.   About Us

In this section, we will introduce the nature of our project software development team, the "automotive parts selling project". We delve into the foundational aspects of our endeavor – an online platform for selling automotive parts – and highlight the challenges encountered throughout the project's development journey. We joined hands to realize a platform dedicated to trading automotive parts. Our collaboration harnesses our shared skills and vision, creating a user-centric digital space for buying and selling automotive parts.

2.   The Product's Information

In today's increasingly modern era, vehicles are everywhere, but the wide range of sources for automotive parts often leaves users overwhelmed. Consumers must carefully evaluate price and supplier reputation to avoid low-quality products or scams. The challenge lies in striking a balance between affordability and user-friendly features that appeal to a diverse global audience. The solution we propose is an automotive parts exchange website called "Auto parts Hub", designed specifically to tackle these issues. By offering a streamlined and intuitive user experience combined with powerful functionality, Auto parts Hub aims to create a trustworthy marketplace that connects sellers and buyers efficiently. This initiative is fueled by our commitment to simplifying the process of finding the right part for

every vehicle's need and fostering a global community where users can quickly find exactly what they're looking for—reliably and with confidence.

3.   Development Environment
- **Front-end**: Using NodeJS: Embedded JavaScript
- **Back-end**: Implementing NodeJS to handle server-side logic
- **Database**: Employing MySQL for data storage and management
- **Architecture**: Following the Model-View-Controller (MVC) for structuring code.
- **Version Control**: Managing code with GitHub to track changes and collaborate. Link GitHub: E-commerce website for automotive parts
- **Communication**: Using Discord for team meetings and discussions.
- **Task management**: Keeping track of work with Google sheet
- **Documentation**: Using JSON to store raw data

II.   REQUIREMENT ANALYSIS AND DESIGN

This section presents a customized requirements analysis and design framework for the development of the trading platform. It acts as a blueprint for the project implementation, specifying each feature along with the defined conditions, as well as the functional and non-functional requirements outlined by the client. During the development process, we will continuously update and improve future iterations, ensuring an accurate and comprehensive record of the project's progress.

1. REQUIREMENT ANALYSIS

Funtional Requirements

Use case 1: Log in

+ Product scope: Allows users to log into the system to perform their functions.

+ Main action:

Actor: User	System
1. Select the login function	1.1 Display login screen
2. Enter username and password	2.1 Fail -> move to sub-action 2.2 Successful -> display screen
3. End of use case	

+ Sub action

Actor: User	System
1. Password checking	1.1 Noticed that there is wrong username or password 1.2 Pre-display login screen 1.3 Return to step 2 in main action
2. Change password 2.1 Click "change password" in login screen 2.2 Enter old password and enter new password, replay new password 2.3 Enter new password	2.1 Display screen to change password 2.2.1 Check old password 2.2.2 Check if the new password and re-entered new password are the same 2.3 Display successful notification screen

+ Precondition: User must have an account on the system

+ Post-condition: User successfully logged in

Use case 2: Manage accounts

+ Product scope: Allows admin to manage customer accounts

+ Main action:

User: Admin	System
1. Click "manage account" to request	1.1 Show form "manage account"
2. Action (more details, lock, open account)	2.1 Return the corresponding operation result

+ Precondition: Already have an admin account

+ Post-condition: Manage all accounts

Use case 3: Manage automotive parts

+ Product scope: Allows admin to manage automotive parts in website

+ Main action:

User: Admin	System
1.1 Send a part management request 1.2 Select "Add Product" 1.3 Enter part details (e.g., name, category, compatibility, price, description) 1.4 Save the new part entry	1.1 Show manage form 1.2 Show add product form 1.3 Send information of part and check information 1.4 Send notice back
2.1 Send a part management request 2.2 Select "Update Product" 2.3 Modify the part details as needed 2.4 Save and update the product information	2.1 Show manage form 2.2 Show update product form 2.3 Send information of part and check information 2.4 Send notice back
3.1 Send a part management request 3.2 Select "Delete Product" 3.3 Confirm deletion and refresh the Manage Parts section	3.1 Show manage form 3.2 Confirm delete and check validity 3.3 Send notice and result back

+ Precondition: Already have an admin account

+ Post-condition: Manage all automotive parts

Use case 4: Manage type of parts

+ Product scope: Allows admin to manage types of all automotive parts

+ Main action:

User: Admin	System
1.1 Send a part type management request 1.2 Choose "Add Type of Product" 1.3 Enter new part type information 1.4 Save the new part type	1.1 Display manage form 1.2 Show add type of product form 1.3 Submit part type information and validate the data 1.4 Send a success or error notice back to the user
2.1 Send a part type management request 2.2 Choose "Update Type of Product" 2.3 Edit the existing part type information 2.4 Save and update the part type	2.1 Display manage form 2.2 Show update type of product form 2.3 Submit updated part type information and validate 2.4 Send a success or error notice back to the user
3.1 Send a part type management request 3.2 Choose "Delete Type of Product" 3.3 Update the Manage Part Types list	3.1 Show manage form 3.2 Confirm delete and check validity 3.3 Send notice and result back

+ Precondition: Already have an admin account

+ Post-condition: Manage all types of automotive parts

Use case 5: Manage origin of products

+ Product scope: Allows admin to manage origin of automotive parts.

+ Main action:

User: Admin	System
1.1 Send an origin management request 1.2 Choose "add origin" 1.3 Add information of origin	1.1 Show origin form 1.2 Show add origin form

1.4 Save new type	1.3 Send information of origin and check validity 1.4 Send notice back
2.1 Send an origin management request 2.2 Choose "Update origin" 2.3 Change information of origin 2.4 Update information of origin	2.1 Show manage form 2.2 Show update origin form 2.3 Send information of origin and check information 2.4 Send notice back
3.1 Send an origin management request 3.2 Choose "delete origin" 3.3 Update "manage origin"	3.1 Show manage form 3.2 Confirm delete and check validity 3.3 Send notice and result back

+ Precondition: Already have an admin account

+ Post-condition: Manage origin of automotive parts

Use case 6: Manage bills

+ Product scope: Allows admin to manage bills

+ Main action:

User: Admin	System
1.1 Send manage bill request 1.2 Choose to see more details 1.3 Choose back	1.1 Show bill form 1.2.1 Check id of bill 1.2.2 Send back result 1.2.3 Send back notice 1.3 Return to home page
2.1 Send manage bill request 2.2 Click received	2.1 Show bill form 2.2 Update the bill
3.1 Send manages bill request 3.2 Click cancelled	3.1 Show bill form 3.2 Update the bill

+ Precondition: Already have an admin account

+ Post-condition: Manage all bills

Use case 7: Revenue statistics

+ Product scope: Allows admin to compile revenue statistics

+ Main action:

User: Admin	System
1.1. Send a request for statistical management	1.1 display statistics form
1.2 Statistics of orders received	1.2 request time information
1.3 Choose time	1.3.1 Check information
	1.3.2 Send back result
2.1. Send a request for statistical management	2.1 display statistics form
2.2 Statistics of cancelled orders	2.2 request time information
2.3 Choose time	2.3.1 Check information
	2.3.2 Send back result
3.1. Send a request for statistical management	3.1 display statistics form
3.2 Statistics of best-selling orders	3.2 request time information
3.3 Choose time	3.3.1 Check information
	3.3.2 Send back result

+ Precondition: Already have an admin account

+ Post-condition: Revenue Analysis

Non-Functional Requirements

1. Usability requirements:

- It is crucial to guarantee that all software components operate accurately and efficiently.
- The hosting environment must hold an active license to manage incoming requests effectively.
- A designated administrator should oversee regular software updates and actively monitor system errors.

2. Usability Requirements:

- Structured and straightforward to understand.
- Clear and informative.
- User-friendly, enabling seamless learning and navigation.
- Allows users to accomplish tasks effortlessly and error-free.

3. Humanity Requirements:

+ Detailed documentation is provided to ensure users can fully comprehend the system through thorough reading.

+ The graphical user interface (GUI) is designed intuitively to facilitate easy learning and quick user adaptation.

+ Responsive interfaces are a critical aspect, enhancing the overall user experience.

4. Maintainability Requirements:

+ The system is built to support regular updates and modifications. Since users may have limited understanding of its internal processes, the development team must ensure the platform is user-friendly and straightforward for both users and maintenance staff.

+ The platform should offer intuitive features enabling users to easily update and manage information related to videos, channels, and viewer interactions.

+ The source code should be cleanly organized and thoroughly documented to improve readability and simplify future updates or debugging.

+ A well-defined and stable database architecture is essential to ensure a reliable foundation for maintenance and scalability.

2. DESIGN

ERD Diagram
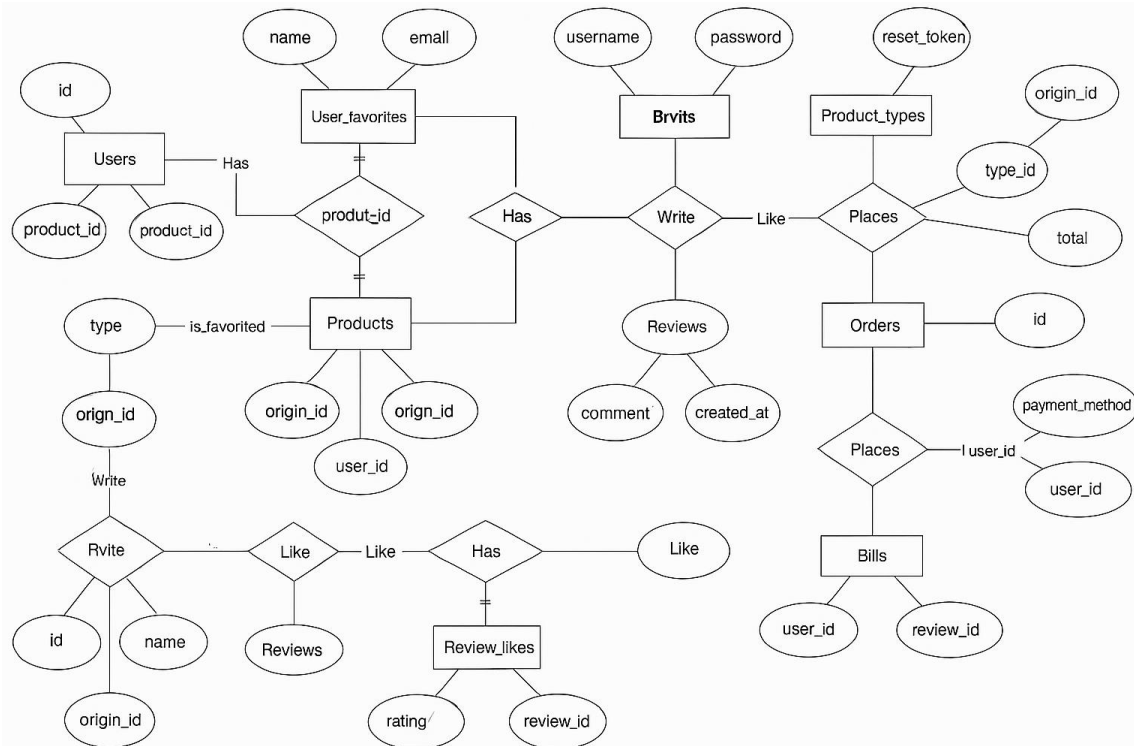
This Entity-Relationship Diagram (ERD) illustrates the structure of the e-commerce platform's database. It includes all relevant entities such as Users, Admins, Products, Product Types, Origins, Orders, and Bills. Each entity is associated with its attributes and relationships, providing a comprehensive overview of how data is connected across the system.

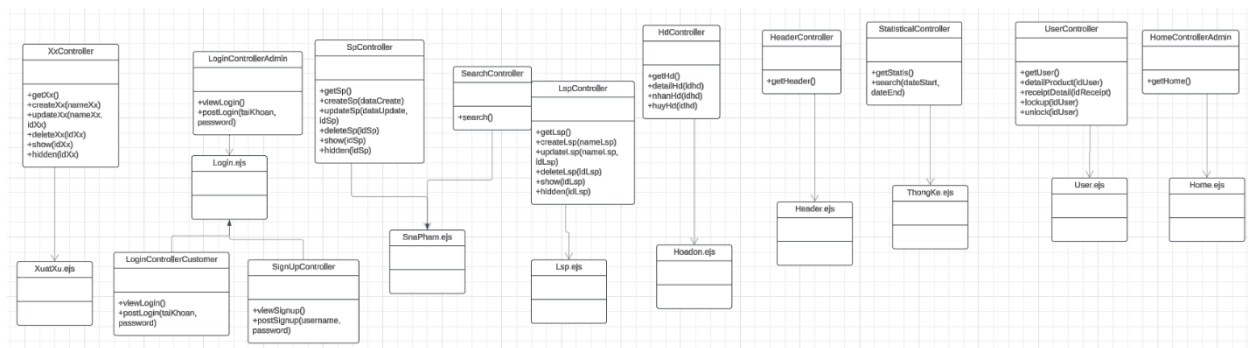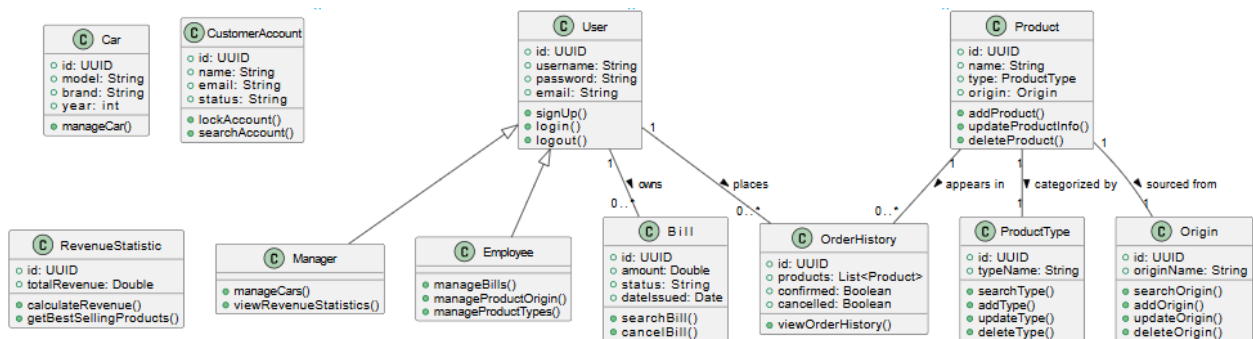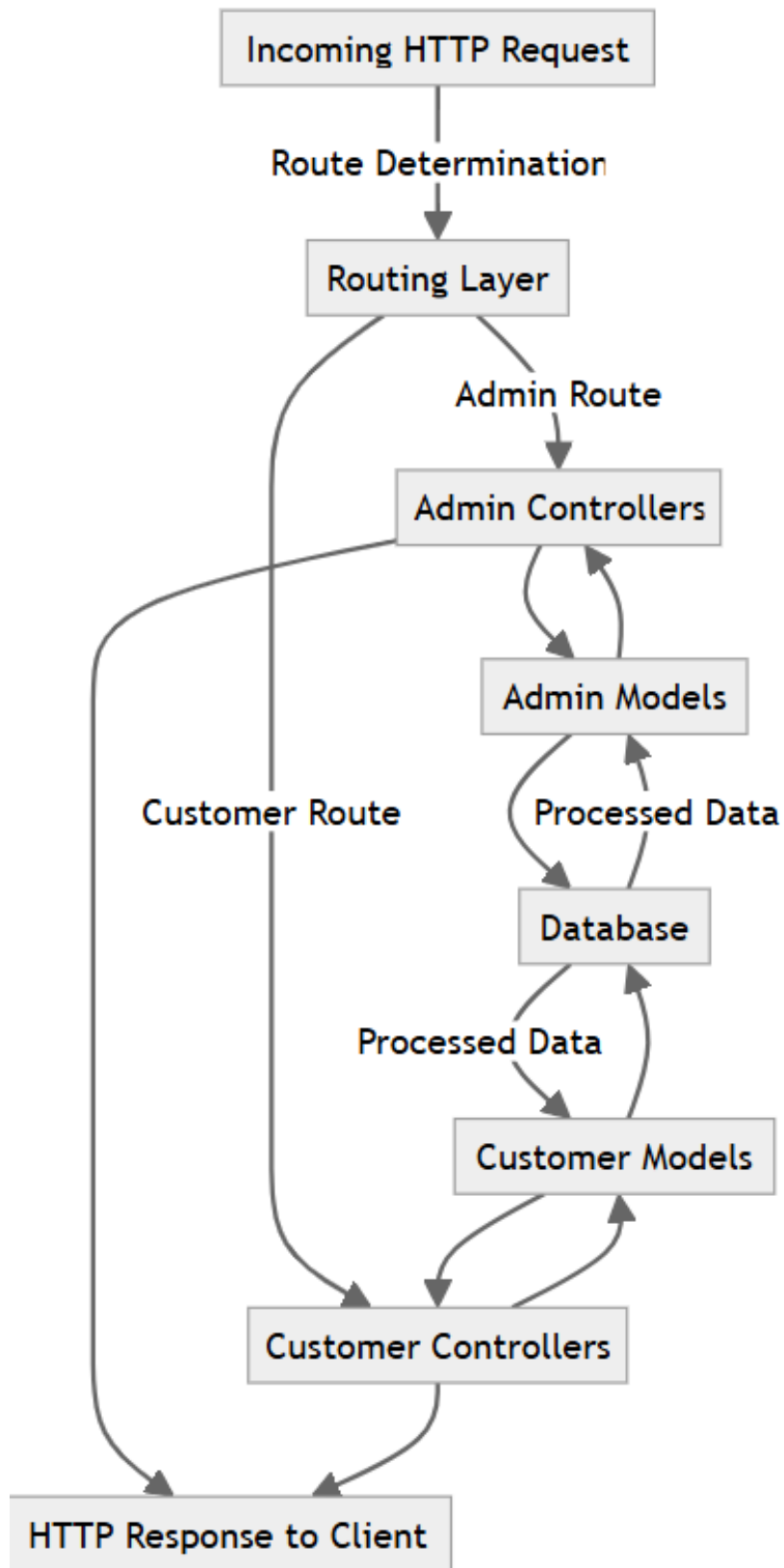# Relational Schema Diagram

Customer Back-end:

Admin Back-end:

13
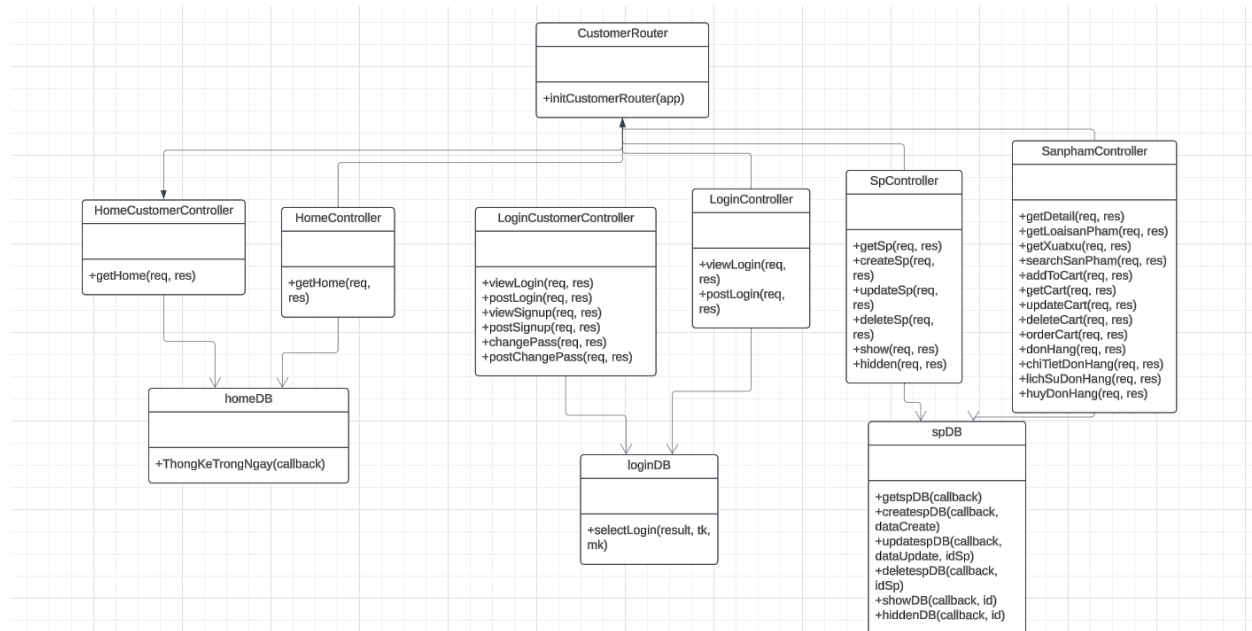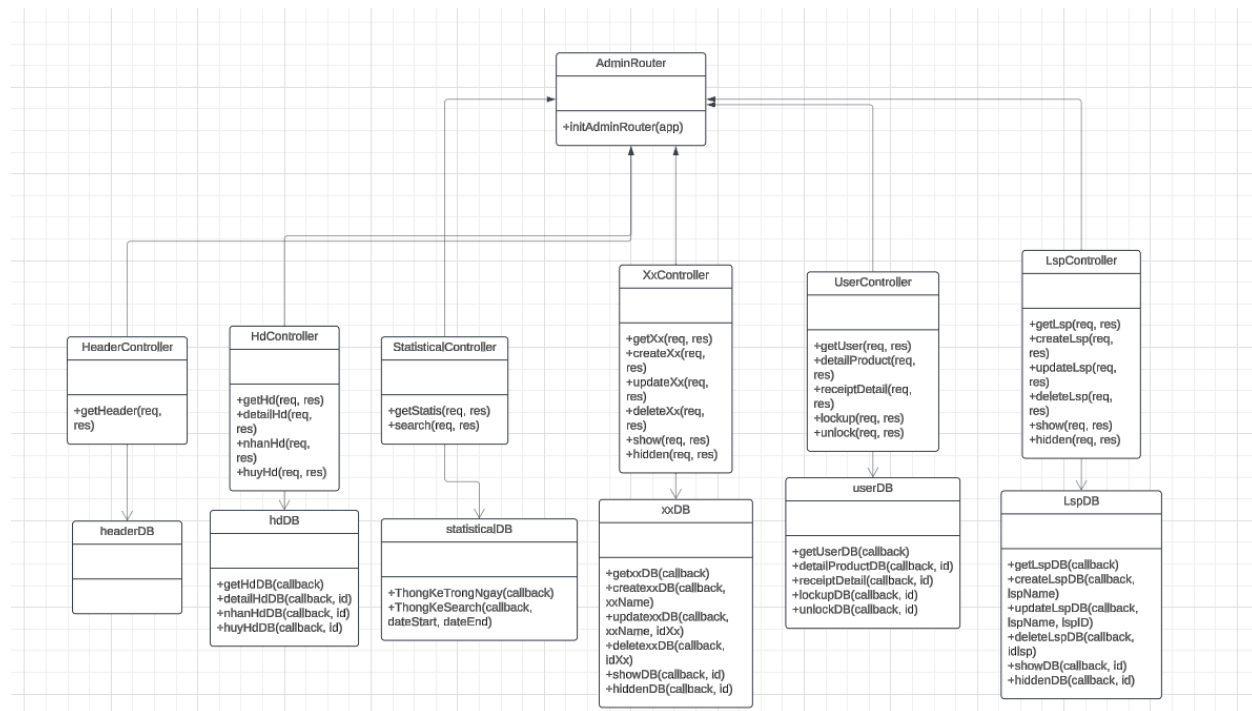
Class-Diagram

Front-end

Back-end

LOGIC FLOW OF BACKEND API
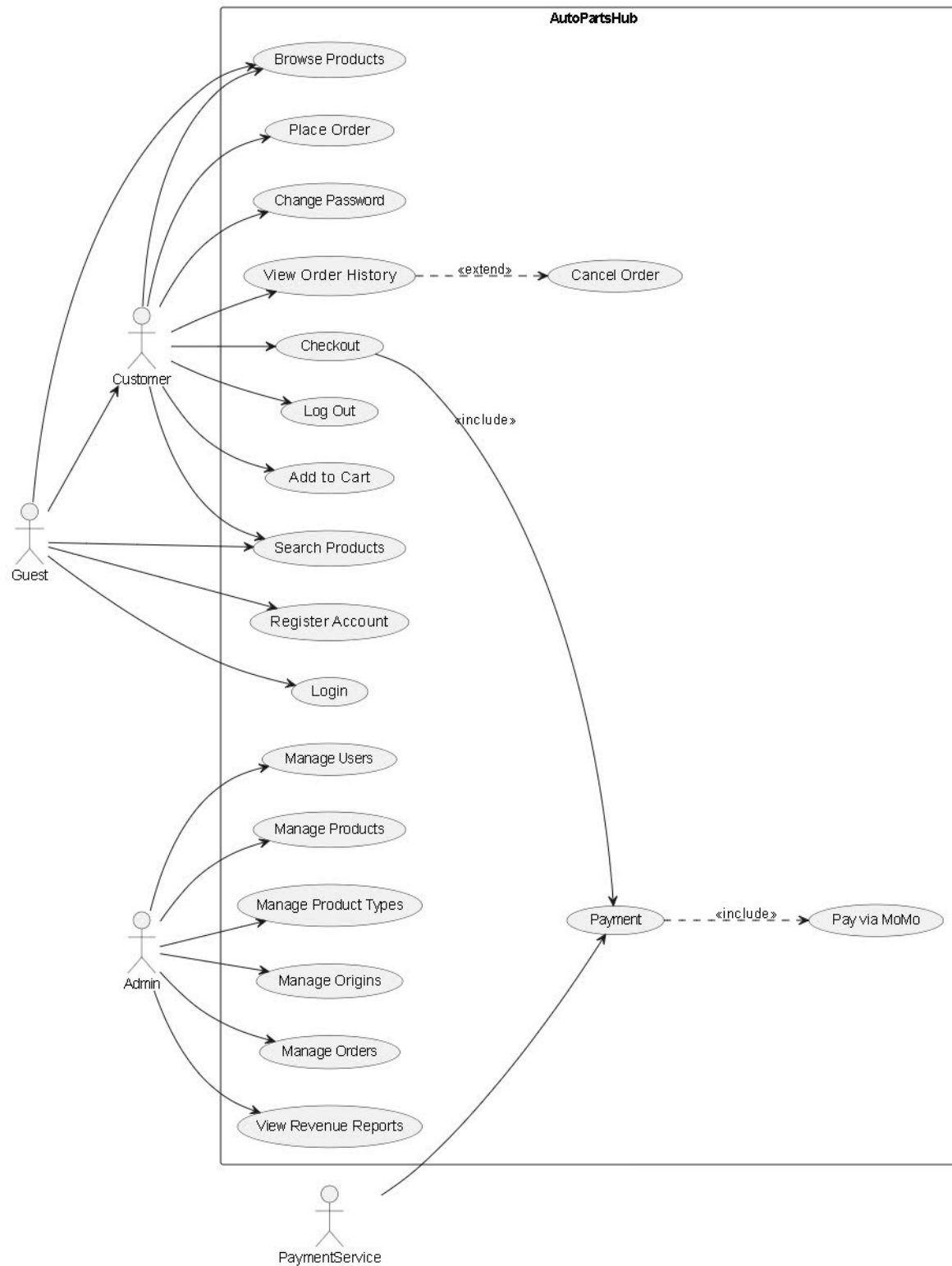
User Case Diagram

API

FRONT-END:

- /login: Page for users to log in.
- /home: Homepage displaying main content for users.
- /header: Header section displayed on customer pages.
- /sanpham: Page displaying products for customers.
- /admin: Admin login page for administrative access.

BACK-END:

- /api/admin/hd: Endpoint to manage admin header content.
- /api/admin/home: Endpoint to manage admin home data.
- /api/admin/login: Endpoint for admin login authentication.
- /api/admin/statistical: Endpoint for retrieving statistical data.
- /api/admin/user: Endpoint to manage user accounts.
- /api/admin/sp: Endpoint for managing products.
- /api/customer/header: Endpoint for retrieving customer page headers.
- /api/customer/home: Endpoint for fetching customer home data.
- /api/customer/login: Endpoint for customer login authentication.
- /api/customer/sanpham: Endpoint for managing customer products.

III.   Frontend

1. Frontend Structure

The frontend of the project is responsible for rendering the user interface and handling user interactions. It ensures a responsive and intuitive experience, built using web technologies such as HTML, CSS, JavaScript, and a UI library is Ant Design.

- Static Files: Located in the public/ folder (e.g., CSS stylesheets, JavaScript files, images).

- Dynamic Templates: Found in the views/ folder, rendered by Express using res.render(). These are usually EJS (.ejs) or HTML files.

2. Functionality

- **Layout Rendering**: Pages such as home, login, vehicle listings, and product details are dynamically rendered based on user actions.

- **Styling**: Handled by custom CSS and/or external libraries to ensure consistent design and responsiveness across devices.

- **User Interaction**:

  o Forms for login/registration

  o Buttons and filters for vehicle/product searches

  o Dynamic elements that update without reloading the page

- **AJAX / Fetch API**: Frontend uses JavaScript (via fetch() or similar) to communicate with the backend in real-time. For example:

  o Fetching car data to populate listings

  o Submitting forms without page reload

  o Comparing multiple vehicles

3. Frontend-Backend Integration

- Backend APIs (possibly written in Java or Node.js) are called from the frontend to fetch or send data.

- JSON is commonly used for data exchange.

- Example flow: User selects a car → JavaScript sends request → Server responds with car info → Page updates dynamically.

4. Frontend Implementation with Code Examples

4.1. Google Chart for admin

- Loads the Google Charts library, enabling the visualization of data.

- Loads the corechart package and sets drawChart as the callback function once the chart library is ready.

```html
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
<!-- <script type="text/javascript" src="/admin/js/home.js"></script> -->
<script type="text/javascript">
  google.charts.load('current', { 'packages': ['corechart'] });
  google.charts.setOnLoadCallback(drawChart);
```

4.2 Google Column Chart in Statistical for admin

- ColumnChart is used for clear comparison of product quantities.

- Displays the selected date range dynamically using <%= dateStart %> and <%= dateEnd %>.

```
var options = { 'title': 'Statistics from: <%= dateStart %> to: <%= dateEnd %>', 'width': 950, 'height': 550 ,

var chart = new google.visualization.ColumnChart(document.getElementById('piechart'));
chart.draw(data, options);
```

4.3 Customer Cart Page

- Solve and update data when customers have change

```
citis.onchange = function() {
    district.length = 1;
    ward.length = 1;
    if (this.value != "") {
        const result = data.filter(n => n.Name === this.value);

        for (const k of result[0].Districts) {
            district.options[district.options.length] = new Option(k.Name, k.Name);
        }
    }
};
```

IV. Backend

1. Backend Structure

The backend is organized under the 'nodejs' directory. It includes the following main components:
- connectDB.js: Establishes database connection
- index.js: Main entry point of the server
- controller/: Contains business logic split by user roles
  - admin/: Admin-specific operations (user management, statistics, login, etc.)
  - customer/: Customer-specific operations
- package.json: Project dependencies

2. Database Design

The database schema is defined in the file 'data_create.sql'. It contains tables for managing users, products, orders, and related business entities. The design follows normalization principles for relational databases.

3. System Architecture Overview

The system architecture consists of three layers:
1. Controller Layer: Handles HTTP requests and routes them to appropriate services.

2. Service Layer (if applied): Encapsulates business logic (partially integrated into controllers).
3. Data Access Layer: Interacts with MySQL database via SQL queries.

Example flow:
Client Request → Express Router → Controller → SQL Query → Database → Response

4. Backend Implementation with Code Examples

This section highlights key parts of the backend implementation with annotated code examples. Each snippet is essential to the operation and organization of the Node.js backend.

4.1. Database Connection Setup

File: nodejs/connectDB.js
This file establishes a connection to the MySQL database using the 'mysql2' library.

```
1   const mysql = require('mysql2');
2
3   const connection = mysql.createConnection({
4     host: 'localhost',
5     user: 'root',
6     password: 'root',
7     database: 'car_fragments',
8     multipleStatements: true
9   });
10
11  connection.connect(function(err) {
12    if (err) {
13      console.error('Connect Error:', err);
14      throw err;
15    }
16    console.log('Connect Success!');
17  });
18
19  module.exports = connection;
20
21
```

4.2. Express Server Initialization

File: nodejs/index.js

The entry point of the backend application, initializing Express and configuring middleware.

```
1   const express = require("express");
2   const expressLayouts = require('express-ejs-layouts');
3   const initAdminRouter = require('./route/admin/initAdminRouter');
4   const initCustomerRouter = require('./route/customer/initCustomerRouter');
5   const fileUpload = require('express-fileupload');
6   var cookieParser = require('cookie-parser');
7
8   let session = require('express-session');
9
10
11  const app = express();
12  const port = 3000;
```

```
app.listen(port, () => {
    console.log(`Run http://localhost:${port}/customer/home`)
})
```

4.3. Admin Login Controller

File: nodejs/controller/admin/loginController.js

Handles the logic for admin login authentication.

```
1   const loginDB = require('../../model/admin/loginDB');
2
3   let viewLogin = (req, res) => {
4       req.session.isLoginAdmin = false;
5       res.render('admin/loginAdmin.ejs', {layout: 'admin/loginAdmin.ejs', dataLogin: ''});
6   }
7
8   let postLogin = (req, res) => {
9
10      loginDB.selectLogin(function(data){
11          if(!data[0]){
12              res.render('admin/loginAdmin.ejs', {dataLogin: 'Wrong username or password'});
13              res.end();
14          }
15          else{
16              req.session.isLoginAdmin = true;
17              res.redirect('/admin');
18              res.end();
19          }
20      }, req.body.taiKhoan, req.body.password);
21  }
22
23  module.exports = {viewLogin, postLogin};
```

4.4. User Management Controller

File: nodejs/controller/admin/userController.js

Manages CRUD operations related to users.

```
1    const userDB = require('../../model/admin/userDB');
2
3    const layout = './admin/layoutAdmin';
4    let getUser = (req, res) => {
5        userDB.getUserDB(function(data) {
6            res.render('admin/user.ejs', {layout: layout, dataUser: data});
7            res.end();
8        })
9    }
10
11   let detailProduct = (req, res) => {
12       userDB.detailProductDB(function(data){
13           res.render('admin/userDetailProduct.ejs', {layout: layout, dataDetail: data})
14           res.end();
15       }, req.params.idUser)
16   }
17
18   let receiptDetail = (req, res) => {
19       userDB.receiptDetail(function(data){
20           res.render('admin/receiptDetail.ejs', {layout: layout, dataDetail: data, STT: 0})
21           res.end();
22       }, req.params.idReceipt)
23   }
24
25   let lockup = (req, res) => {
26       userDB.lockupDB(function(data){
27           res.redirect('/admin/user');
28           res.end();
29       }, req.params.idUser)
30   }
31
32   let unlock = (req, res) => {
33       userDB.unlockDB(function(data){
34           res.redirect('/admin/user');
35           res.end();
36       }, req.params.idUser)
37   }
```

4.5. Statistical Report Controller

File: nodejs/controller/admin/statisticalController.js

Retrieves and responds with summarized statistics from the database.

```
nodejs > controller > admin > JS statisticalController.js > ...
1    const statisticalDB = require('../../model/admin/statisticalDB');
2
3    const layout = './admin/layoutAdmin';
4    let getStatis = (req, res) => {
5        statisticalDB.ThongKeTrongNgay(function(data) {
6            res.render('admin/statistical.ejs', {layout: layout, dataStatis: data, Stt: 0});
7            res.end();
8        })
9    }
10
11   let search = (req, res) => {
12       statisticalDB.ThongKeSearch(function(data) {
13           res.render('admin/statisticalSearch.ejs', {layout: layout, dataStatis: data, dateStart: req.query.dateStart, dateEnd: req.q
14           res.end();
15       }, req.query.dateStart, req.query.dateEnd)
16   }
17
18   module.exports = {
19       getStatis, search
20   }
```
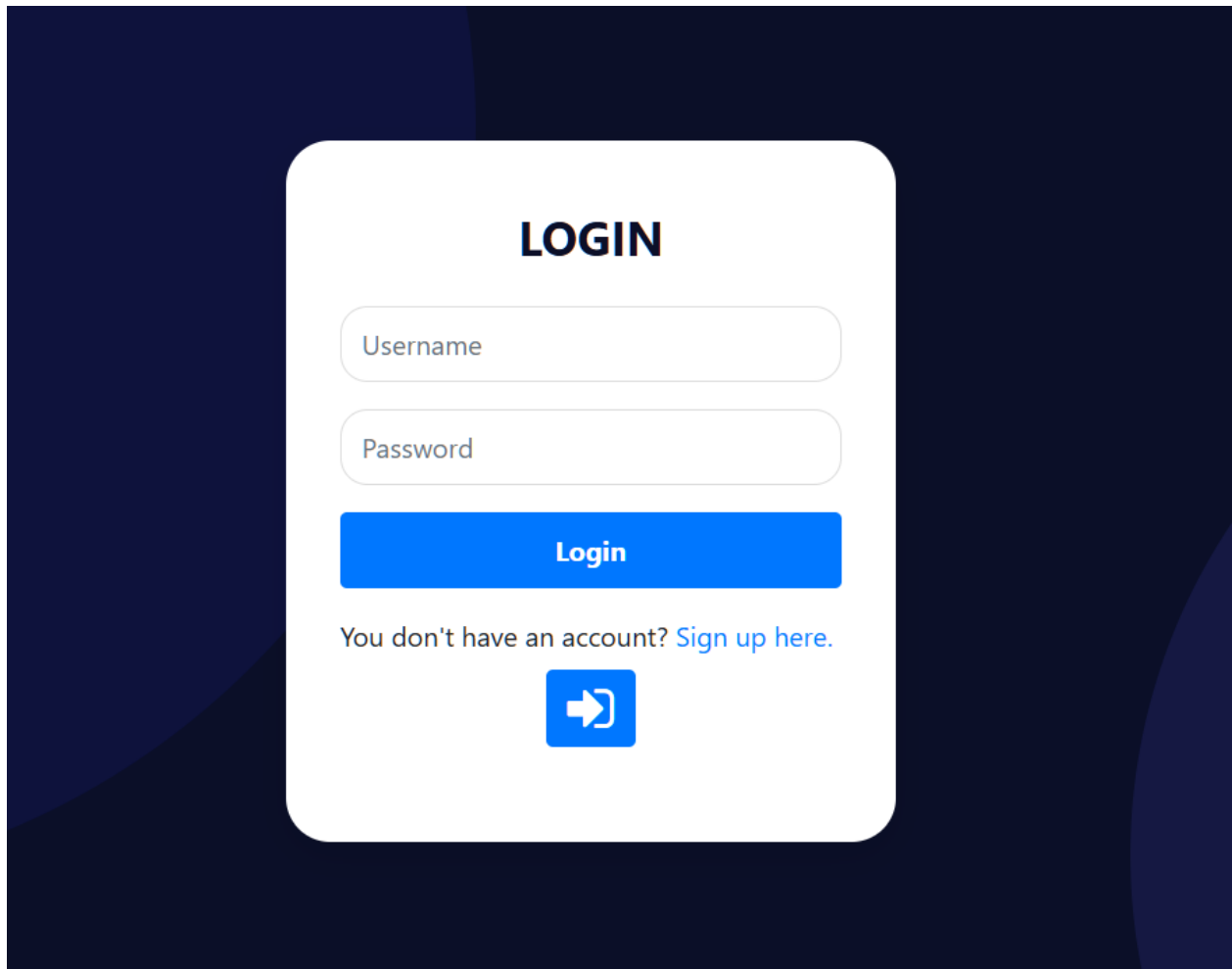
V. IMPLEMENTATION

o Users' account

1.1 Account Management

1.1.1 Login to application

Step 1: Click Account icon

Step 2: Enter the username and password of your account. Then click "**Login**", the system will check if username and password is true it redirects to Home page. Otherwise, it will return the error message.
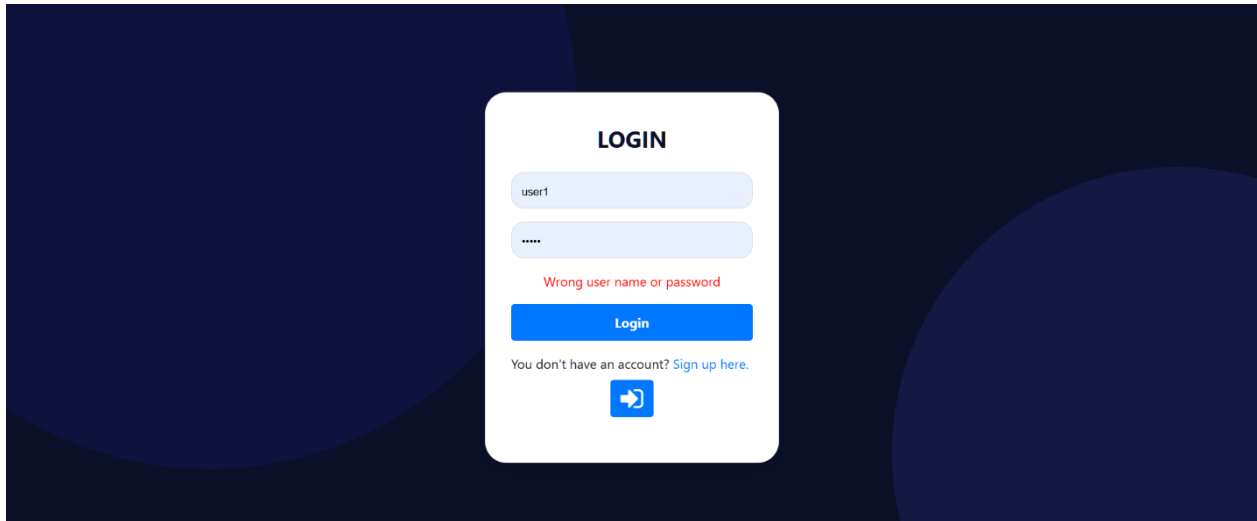
Error message will be appeared If the system recognizes that your account is incorrect or does not exist.
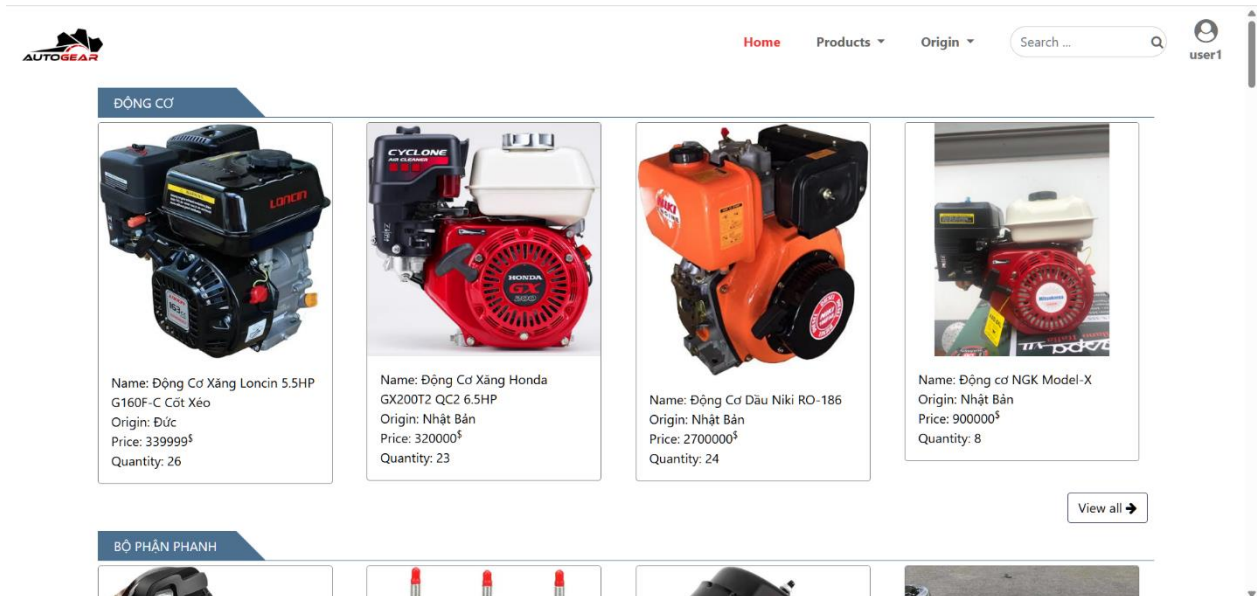
You will move to the homepage when login successful.

1.1.2 Register

Step 1: If you don't have any account, click "Sign up here".

Step 2: Input username, password and re-password

Sign up page

1.2. Home
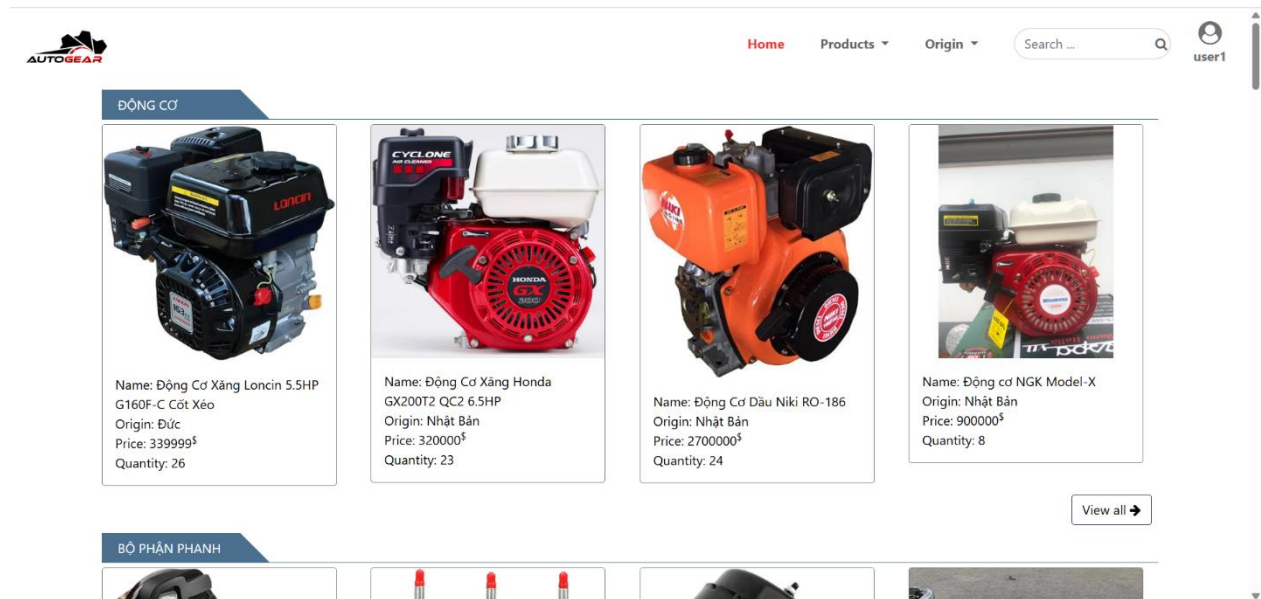
Homepage

1.2.1 Products

When you click "Products", the web will show all types of products, you click on the type you want, and the web will show the one you choose.

1.2.2 Origin

When you click "Origin", the web will show all origins of automotive parts, you click on the type you want, and the web will show the one you choose.
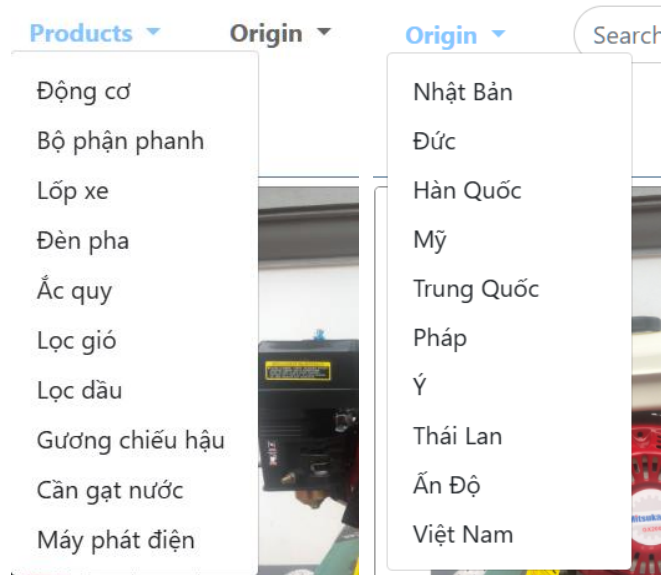
List of Products/Origin

1.2.3 "Search" engine

You can search the car you want by type in "search" box
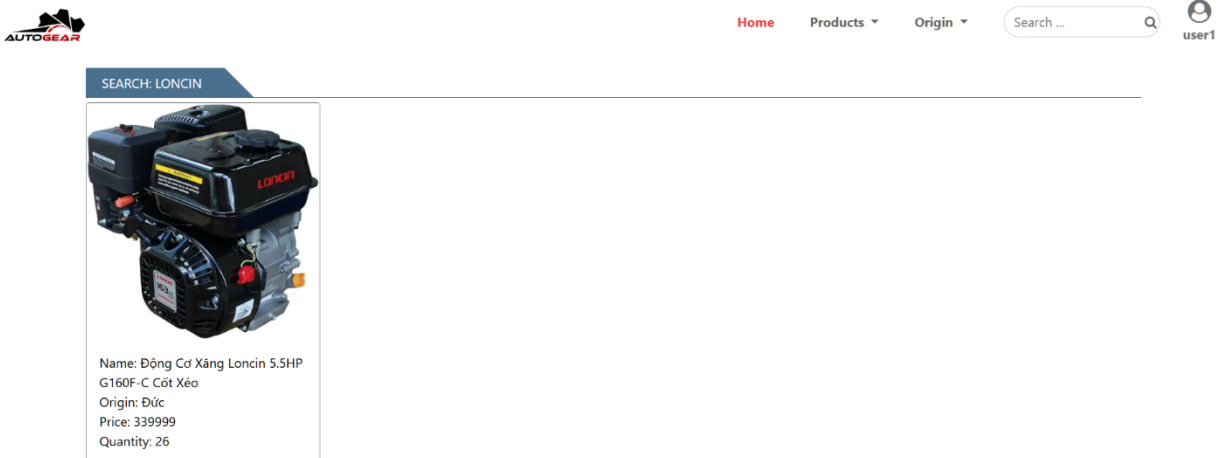
Searching Loncin

1.2.4 Account
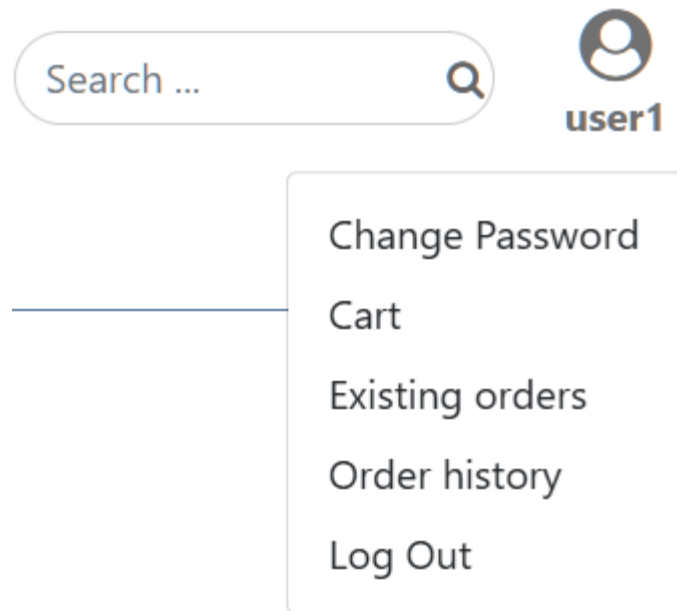
Account drops down

1.2.4.a Change password
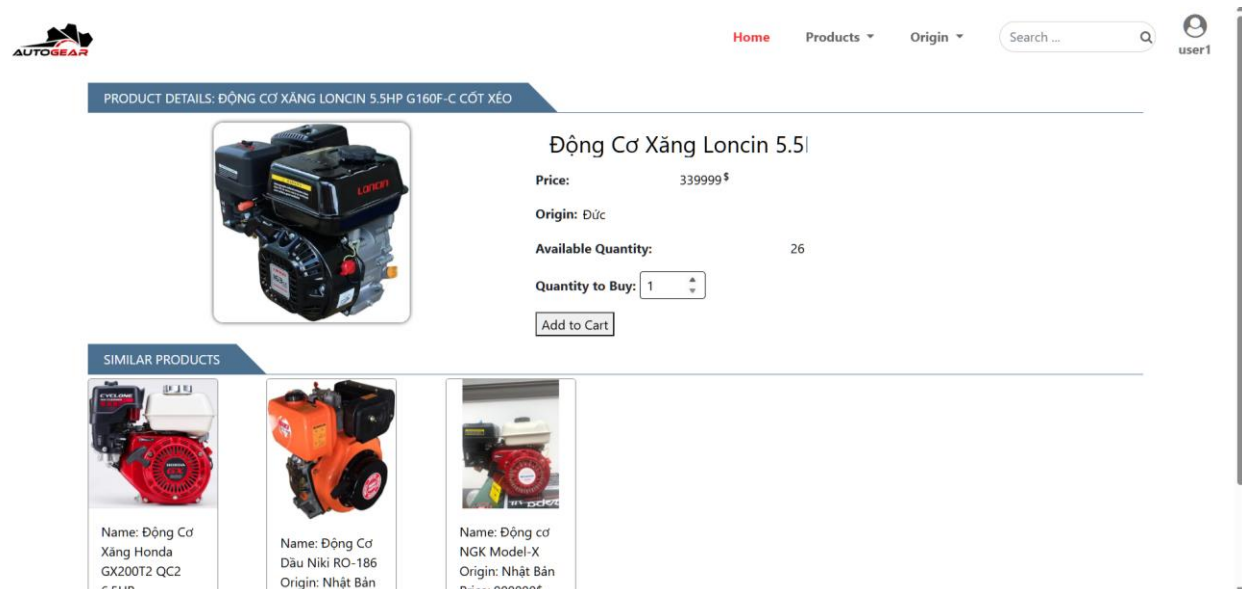
Step 1: Click on "Change password" in account dropdown

Step 2: Input old password, new password and re-new password. After that, click "Change password". If you don't want to change your password, click "Back"

1.2.4.b Cart

Step 1: Click to a product you want to buy, choose "Quantity to Buy" and click "Add to Cart"
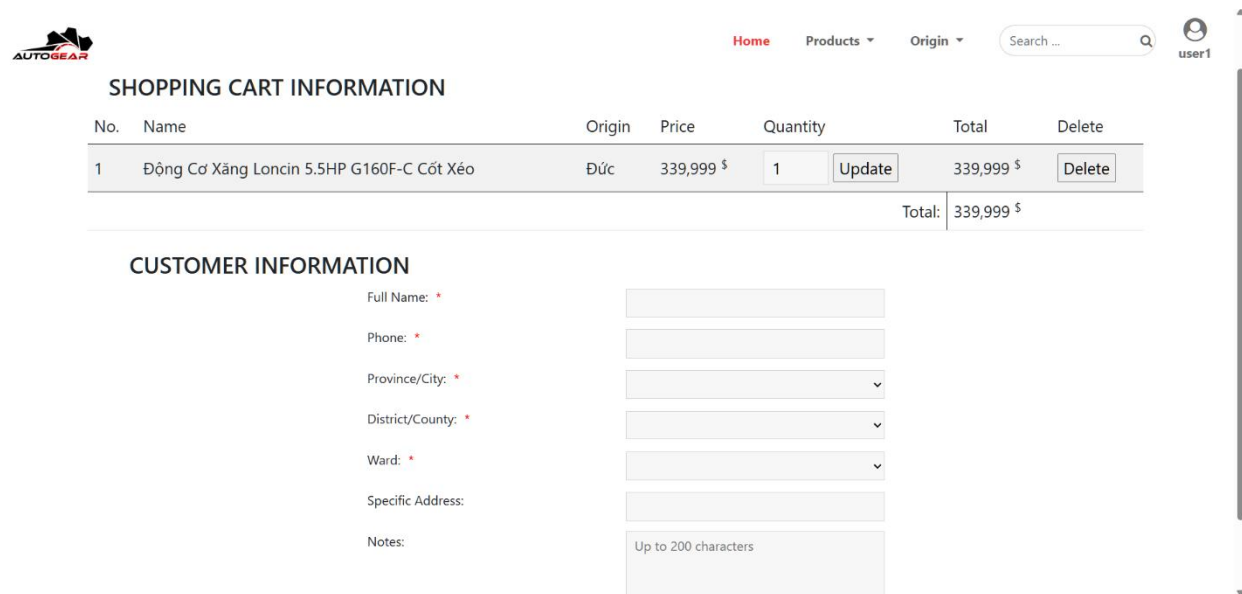
Step 2: Fill in your personal information and click on "Continue shopping" if you want to add products or "Order" if you want to place an order

Cart Form

You can update the quantity of automotive parts or add automotive parts or delete automotive parts in cart.

32

Step 3: When done, the QR code will appear shortly and will be ready to receive your payment.

NGUYEN HAI QUAN
31410009194022
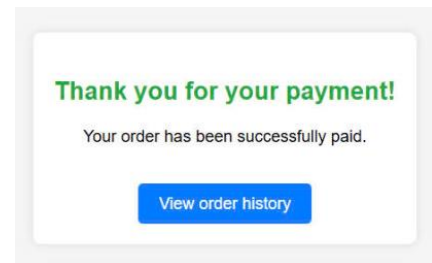BIDV – CN Đông Sài Gòn

Or we can insert the Transaction code

After that, you click "I have completed the payment on the App" and there will be a confirmation message:

No.	Recipient Name	Address	Phone	Total Amount	Purchase Date	Notes	Status	Actions	
1	Nguyễn Hải Quân	Thành phố Hải Phòng, Huyện Cát Hải, Xã Xuân Đám, ffs	0123456789	217030 $	2025/06/01 15:18:23	gg	Pending	Details	Cancel Order

You can recognize the information by clicking the "order information"

When you click "Cancel Order", the Order will be cancelled.

ORDER

No.	Recipient Name	Address	Phone Number	Total Amount	Purchase Date	Notes	Status
1	Henry	Thành phố Hà Nội, Quận Đống Đa, Phường Cát Linh	0908336061	136,240 $	2025/05/02 09:33:45	tung tung sahur	Completed

DETAILS

No.	Product Name	Origin	Price	Quantity	Total
1	Bầu trợ lực phanh Ô tô chính hãng	Trung Quốc	120 $	2	240 $
2	Lốp ô tô Kenda 23550R18 97 H KR52	Nhật Bản	17,000 $	8	136,000 $

DETAILS

No.	Product Name	Origin	Price	Quantity	Total
1	Bầu trợ lực phanh Ô tô chính hãng	Trung Quốc	120 $	2	240 $
2	Lốp ô tô Kenda 23550R18 97 H KR52	Nhật Bản	17,000 $	8	136,000 $

1.2.4.c Existing orders

When you click "existing orders", the web will show your all waiting order

ORDER INFORMATION

No.	Recipient Name	Address	Phone	Total Amount	Purchase Date	Notes	Status	Actions	
1	Nguyễn Hải Quân	Thành phố Hải Phòng, Huyện Cát Hải, Xã Xuân Đám, ffs	0123456789	217030 $	2025/06/01 15:18:23	gg	Pending	Details	Cancel Order

1.2.4.d Order history

When you click "Order history", the web will show all your cancelled/completed order

ORDER HISTORY

| #	Recipient's Name	Address	Phone Number	Total Amount	Purchase Date	Notes	Status	
1	Nguyễn Hải Quân	Thành phố Hải Phòng, Huyện Cát Hải, Xã Việt Hải, ffs	0123456789	360 $	2025/06/01 11:58:34	good	Completed	Details
2	Nguyễn Hải Quân	Thành phố Hải Phòng, Huyện Cát Hải, Xã Xuân Đảm, ffs	0123456789	240 $	2025/06/01 11:40:36	gg	Completed	Details

o Admins' Account

2.1 Login

Step 1: Admin will have a private path to access, so customers will not be able to enter without knowing it.

Step 2: When you fill up the admin account, the web will show the manager screen
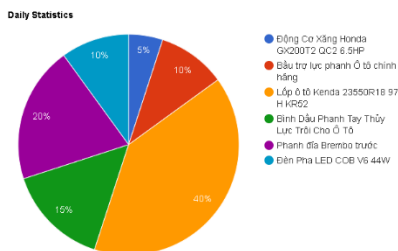
Main

The main screen will show daily statistics

2.2 Admins manager

Admin manages dropdown will be show when hover or click on it.

Manage ▾

Account

Type Of Product

Origin

Product

Bill

Statistic

2.2.1 Account

When you click "Account", the web show lists of accounts.

Main	Manage ▾	Website				▲▾

Customer Accounts

🔍 Search username ...

ID	Account Name	Status	Sign up Date	Bills	Actions
1	user1	1	2024/12/11 14:50:51	View	Lock Account
2	user2	1	2024/12/11 14:50:51	View	Lock Account
3	user3	1	2024/12/11 14:50:51	View	Lock Account
8	123	1	2025/05/02 16:28:30	View	Lock Account

List of users

You can find Account by search.

Customer Accounts

🔍 user2

ID	Account Name	Status	Sign up Date	Bills	Actions
2	user2	1	2024/12/11 14:50:51	View	Lock Account

Searching "user2"

When you click "Views", it will show all details as pending/ cancelled/ completed order.

Main	Manage ▾	Website		▲▾

ID	Account	Status	Sign up Date
8	123	1	2025/05/02 16:28:30

🔍 Finding...

ID	Name	Address	Phone Number	Total	Purchased Date	Note	Status	More details
52	David	Thành phố Hải Phòng, Quận Lê Chân, Phường Hồ Nam	0833341233	804290	2025/05/02 09:38:15		Completed	View
51	Henry	Thành phố Hà Nội, Quận Đống Đa, Phường Cát Linh	0908336061	136240	2025/05/02 09:33:45	tung tung sahur	Completed	View
50	Henry Truong	Tỉnh Vĩnh Phúc, Huyện Lập Thạch, Xã Tử Du, 345 Park Avenue	0908336061	320000	2025/05/02 09:28:53		Pending	View

Back

Click "view" to View more details in that bill.

Main	Manage ▼	Website							👤▼

ID	Account	Password	Status		Sign Up Date			
8	123	123	1		2025/05/02 16:28:30			

Bill's ID	Customer's Name	Address		Phone Number	Total	Purchased Date	Note	Status
51	Henry	Thành phố Hà Nội, Quận Đống Đa, Phường Cát Linh		0908336061	136240$	2025/05/02 09:33:45	tung tung sahur	Completed

Q Search for names..

No.	Product's ID	Product		Origin	Cost	Quantity	Total
1	47	Bầu trợ lực phanh Ô tô chính hãng		Trung Quốc	120$	2	240$
2	15	Lốp ô tô Kenda 23550R18 97 H KR52		Nhật Bản	17000$	8	136000$

Back

You can unlock/lock account of customer.

Main	Manage ▼	Website				👤▼

Customer Accounts

Q Search username ...

ID	Account Name	Status	Sign up Date	Bills	Actions
1	user1	1	2024/12/11 14:50:51	View	Lock Account
2	user2	1	2024/12/11 14:50:51	View	Lock Account
3	user3	1	2024/12/11 14:50:51	View	Lock Account
8	123	1	2025/05/02 16:28:30	View	Lock Account

2.2.2 Manage type of products

When you click "Type of product", it will show all types you have.

Manage type of product

Q Search for names.. | Add Product

No.	ID	Type of Product	Added Date	Updated Date	Available	⚙	
1	1	Động cơ	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
2	2	Bộ phận phanh	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
3	3	Lốp xe	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
4	4	Đèn pha	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
5	5	Ắc quy	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
6	6	Lọc gió	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
7	7	Lọc dầu	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
8	8	Gương chiếu hậu	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
9	9	Cần gạt nước	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
10	10	Máy phát điện	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide

Manage type of product

No.	ID	Type of Product	Added Date	Updated Date	Available	⚙	
1	1	Động cơ	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
2	2	Bộ phận phanh	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
3	3	Lốp xe	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
4	4	Đèn pha	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
5	5	Ắc quy	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
6	6	Lọc gió	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
7	7	Lọc dầu	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
8	8	Gương chiếu hậu	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
9	9	Cần gạt nước	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
10	10	Máy phát điện	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide

You can change the name of type of products when click "Change"
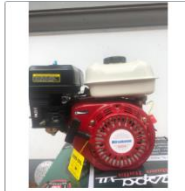
Then the name of products will be changed in the customer's web

Manage type of product

No.	ID	Type of Product	Added Date	Updated Date	Available	⚙	
1	1	Động cơ	2025/04/24 14:36:05	2025/04/24 14:36:05	NO	Change	Show
2	2	Bộ phận phanh	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
3	3	Lốp xe	2025/04/24 14:36:05	2025/04/24 14:36:05	NO	Change	Show
4	4	Đèn pha	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
5	5	Ắc quy	2025/04/24 14:36:05	2025/04/24 14:36:05	NO	Change	Show
6	6	Lọc gió	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
7	7	Lọc dầu	2025/04/24 14:36:05	2025/04/24 14:36:05	NO	Change	Show
8	8	Gương chiếu hậu	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide
9	9	Cần gạt nước	2025/04/24 14:36:05	2025/04/24 14:36:05	NO	Change	Show
10	10	Máy phát điện	2025/04/24 14:36:05	2025/04/24 14:36:05	YES	Change	Hide

Hide/Show the type of products.

Then this type of product will be unavailable
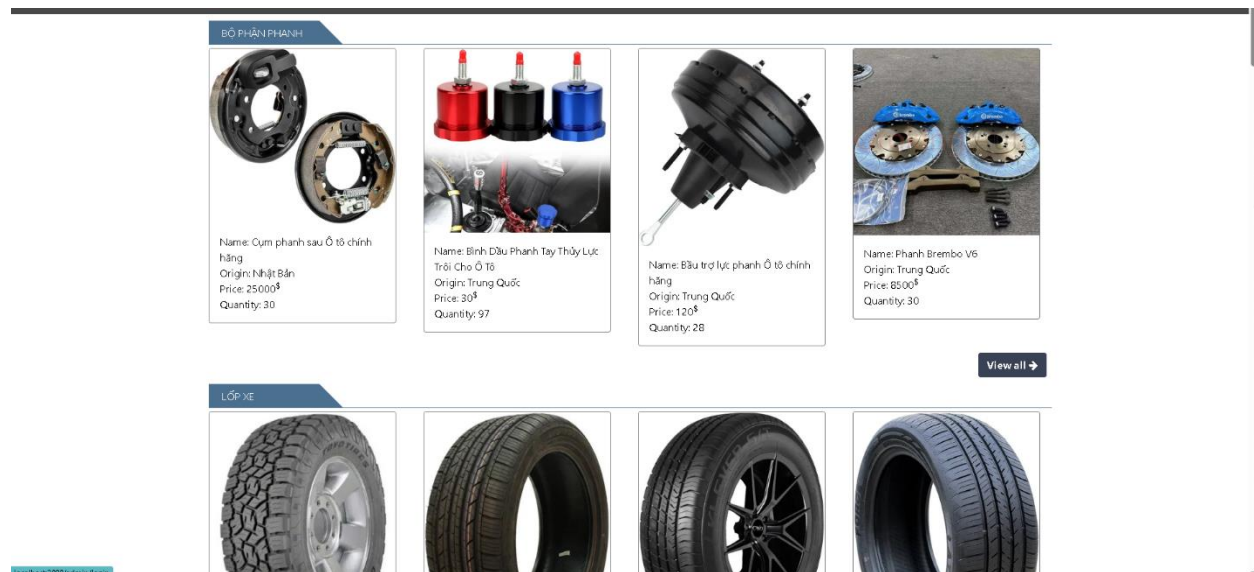
2.2.3 Manage origin of products

You can check the origin of car by click "Origin", the action will the same as " Type of Product".

2.2.4 Manage products

Add products

2.2.5 Bill management

Bill Management

No.	Account ID	Name	Address	Phone Number	Total	Purchased Date	Note	Status	Done		
1	8	Henry Truong	Tỉnh Vĩnh Phúc, Huyện Lập Thạch, Xã Tử Du, 345 Park Avenue	0908336061	320000$	2025/05/02 09:28:53		Pending	More Information	Received	Cancelled
2	8	Henry	Thành phố Hà Nội, Quận Đống Đa, Phường Cát Linh	0908336061	136240$	2025/05/02 09:33:45	tung tung sahur	Completed	More Information		
3	8	David	Thành phố Hải Phòng, Quận Lê Chân, Phường Hồ Nam	0833341233	804290$	2025/05/02 09:38:15		Completed	More Information		

41

Bill

No.	Account ID	Name	Address	Phone Number	Total	Purchased Date	Note	Status
3	8	David	Thành phố Hải Phòng, Quận Lê Chân, Phường Hồ Nam	0833341233	804290$	2025/05/02 09:38:15		Completed

More Information

No.	Product ID	Product	Image	Origin	Cost	Quantity	Total
1	48	Bình Dầu Phanh Tay Thủy Lực Trôi Cho Ô Tô		Trung Quốc	30	3	90$
2	2	Phanh đĩa Brembo trước		Đức	200000	4	800000$
3	19	Đèn Pha LED COB V6 44W		Hàn Quốc	2100	2	4200$

Close

More Information in Bill

2.2.6 Statistical

Statistical

From: 05/02/2025 To: 05/02/2025   statistics

Today statistics
- Động Cơ Xăng Honda GX200T2 QC2 6.5HP
- Bầu trợ lực phanh Ô tô chính hãng
- Lốp ô tô Kenda 23550R18 97 H KR52
- Bình Dầu Phanh Tay Thủy Lực Trôi Cho Ô Tô
- Phanh đĩa Brembo trước
- Đèn Pha LED COB V6 44W

No.	Product	Price	Quantity	Total
1	Động Cơ Xăng Honda GX200T2 QC2 6.5HP	320000	1	320000
2	Bầu trợ lực phanh Ô tô chính hãng	120	2	240
3	Lốp ô tô Kenda 23550R18 97 H KR52	17000	8	136000
4	Bình Dầu Phanh Tay Thủy Lực Trôi Cho Ô Tô	30	3	90
5	Phanh đĩa Brembo trước	200000	4	800000
6	Đèn Pha LED COB V6 44W	2100	2	4200

Statistical

No.	Product	Price	Quantity	Total
1	Động Cơ Xăng Honda GX200T2 QC2 6.5HP	320000	1	320000
2	Bầu trợ lực phanh Ô tô chính hãng	120	2	240
3	Lốp ô tô Kenda 23550R18 97 H KR52	17000	8	136000
4	Bình Dầu Phanh Tay Thủy Lực Trởi Cho Ô Tô	30	3	90
5	Phanh đĩa Brembo trước	200000	4	800000
6	Đèn Pha LED COB V6 44W	2100	2	4200

Main   Manage ▼   Website

Statistical

Statistics from: 2025-05-02 to: 2025-05-02

No.	Product	Price	Quantity	Total
1	Động Cơ Xăng Honda GX200T2 QC2 6.5HP	320000$	1	320000$
2	Bầu trợ lực phanh Ô tô chính hãng	120$	2	240$
3	Lốp ô tô Kenda 23550R18 97 H KR52	17000$	8	136000$
4	Bình Dầu Phanh Tay Thủy Lực Trởi Cho Ô Tô	30$	3	90$

VI. DISCUSSION AND CONCLUSION

Developing an e-commerce platform tailored for automotive parts posed significant challenges, especially in managing an extensive inventory system and implementing a seamless parts comparison feature. One of the key hurdles involved efficiently storing and retrieving detailed vehicle data, encompassing images, specifications, and pricing information.

For the front-end, Node.js and the Ant Design (antd) library served as robust foundations, ensuring a modern, responsive, and intuitive interface. On the backend, built with Servlets, meticulous optimization was crucial to handle complex queries for filtering and sorting vehicle data. Docker containers were configured to enhance scalability, complemented by AWS services for secure image storage and rapid retrieval.

Implementing the vehicle comparison feature introduced another layer of complexity, necessitating precise synchronization between front-end and backend to ensure real-time updates to comparison lists, thereby enriching user experience.

This project provided an invaluable learning experience, offering insights into comprehensive web development and contemporary deployment techniques. Skills acquired included managing diverse data types, optimizing server performance, and enhancing user interaction. Utilizing these tools not only enhanced deployment proficiency but also underscored the importance of leveraging cloud services for scalable applications. Ultimately, this project fortified our technical expertise and prepared us for future advancements in web development and cloud computing.

VII. REFFERENCES

[1] Oracle, "Java EE 7 Tutorial: Servlets," Oracle Documentation. [Online].

[2] Vite. (n.d.). *Vite guide*.

[3] MySQL Tutorial. (n.d.). *MySQL Tutorial*.

[4] Java point. (n.d.). *Example to connect to the MySQL database*.

[5] MySQL Documentation. (n.d.). *MySQL Documentation*.

[6] W3Schools. (n.d.). *W3Schools Online Web Tutorials*

[7] Baeldung. (2024, May 11). *Spring Boot Tutorial – Bootstrap a Simple Application*. Retrieved from Baeldung.

[8] Rascia, T. (2017, December 07). *How to Connect to an API with JavaScript*. Retrieved from Tania Rascia Blog