CIS 525 Fall 2023 Term Project
Final Report

Dr. Jinhua Guo

I. Project Title and Team members

Vybhavi Badugu (02821334)
Sai Venkata Tejaswini Betina (57969830)
Pooja Alumalla (93297444)

II. A web link to our project

http://vehicle-app-bucket.s3-website.us-east-2.amazonaws.com/

III. Proposal sections update

1. **Introduction:**

Background:
The challenge that we have considered is to make vehicle purchasing hassle-free as the buyer/user goes through a lot of paperwork and signatures etc. Thus, we have come up with a solution to this project aims to revolutionize the vehicle purchasing experience by creating a user-friendly online platform. Users can seamlessly search and buy vehicles from various dealerships. The system aggregates vehicle listings, providing a comprehensive selection in one place. Streamlining the traditional vehicle buying process, it provides immense value to consumers seeking a convenient online purchasing experience.

Scope:
This project encompasses the essential stages of the vehicle purchasing journey, incorporating features such as browsing listings, adding to the cart, finalizing purchase orders, and overseeing transaction history. It seamlessly integrates website hosting, user authentication, and purchase notifications for a comprehensive user experience. It's important to note that aspects such as physical vehicle handling, financing, loans, and interactions with external payment processors are intentionally excluded from the project's scope. The focus remains on streamlining the online vehicle purchasing process while keeping the system agile and specialized.

Objectives:

- Digital Transformation: Develop an online Vehicle Management System to transition from manual to digital processes.
- Security: Implement robust user authentication and authorization mechanisms.
- Scalability: Utilize AWS services for scalable and reliable infrastructure.

User Authentication and Authorization:

- Implement secure user authentication and role-based authorization to ensure data privacy and system integrity.

Responsive User Interface:

- Develop an intuitive and responsive user interface using Angular, ensuring a seamless experience across various devices and screen sizes.

Backend Services:

- Create a robust backend using Angular, providing restful APIs for efficient data retrieval, manipulation, and communication with the frontend.

Database Management:

- Utilize AWS Dynamo DB as the database management system to store and manage application data efficiently and securely.

Data Integrity:

- Implement comprehensive data validation and integrity checks to ensure consistency and reliability of the stored data.

User Interaction:

- Enable users to perform essential tasks such as CRUD operations, searching, and filtering data.

2. **<u>Modules:</u>**

User Management:
Efficiently handles user interactions by overseeing account registration, login processes, and profile management. This module serves as a secure repository for storing customer data.

Vehicle Catalogue:
Empowers users with a visually rich display of the vehicle inventory, offering comprehensive details such as make, model, specifications, and pricing. Users can

seamlessly browse, explore, and add vehicles to their cart for a user-friendly experience.

Shopping Cart:

Provides users with a convenient platform to curate their desired vehicle selections. Users can effortlessly add vehicles to their cart and manage selected items, ensuring a smooth and personalized shopping journey.

Checkout/Order Processing:

Streamlines the final stages of the purchase journey by concluding purchase orders, processing transactions, and generating detailed order invoices. This module ensures a seamless and secure transition from selection to purchase.

Transaction History:

Acts as a comprehensive archive, storing detailed records of purchase transactions. Users can access vital information, including vehicle details and transaction statuses, enabling them to effectively track and manage their purchase history.

Inventory Database:

Serves as the backbone of the system, functioning as the dynamic inventory system. Admins and dealers can efficiently update the database, ensuring that the site reflects the most current and accurate information about available vehicle listings.

Website Hosting and Security:

Ensures the platform's vitality by providing robust website hosting capabilities. Additionally, this module takes charge of access controls, safeguarding the site's integrity and security for both users and administrators.

Administration Management:

Efficiently handles user interactions by helping in user account management and profile management where the admin plays a seller role in the web application and will be able to add and modify the list of cars. This module serves as a secure repository for modifying data.

3. From Modules to Operations (CRUD Operations)

User Management:

- **Create:** Facilitates user sign-up and registration.
- **Read:** Allows users to view their profile details.
- **Update:** Empowers users to edit their account settings.
- **Delete:** Enables the deactivation of a user account.

Vehicle Catalogue/Administration Managment:

- **Create:** Permits admins to add new vehicle listings.
- **Read:** Provides users with the ability to browse and view vehicle listings.
- **Update:** Allows for the editing and updating of existing vehicle listings.
- **Delete:** Facilitates the removal of vehicle listings that are sold.

Shopping Cart:

- **Create:** Enables users to add vehicle listings to their cart.
- **Read:** Provides a view of items currently added to the cart.
- **Update:** Allows users to change quantities of items in the cart.
- **Delete:** Facilitates the removal of items from the shopping cart.

Checkout/Order Processing:

- **Create:** Generates new purchase orders and transactions.
- **Read:** Allows users to review order invoices and purchase details.
- **Update:** Permits the editing of order details before finalizing.
- **Delete:** Facilitates the cancellation or removal of orders.

Transaction History:

- **Create:** Records purchase transactions completed at checkout.
- **Read:** Enables customers to review their past transactions.
- **Update:** Allows for the changing of transaction record statuses after completion.
- **Delete:** While not typical, historical records persist.

4. User Roles and Access Control

Standard User:

- **Create Account:** By entering information like name and email.
- **Login/Logout:** Access limited to their own account.
- **View Vehicle Catalog Listings:** Access details like pricing and specifications.
- **Search and Filter:** Allows searching and filtering of vehicle listings based on attributes.
- **View Full Details:** Provides access to complete vehicle details.
- **Shopping Cart Operations:** Adds, edits, and removes items from the personal shopping cart.
- **Checkout Process:** Completes the purchase order flow to buy vehicles in the cart.
- **Access Transaction History:** Views past purchase orders and transaction history for their account.
- **Edit Profile:** Modifies profile information like email, phone, and password.
- **No Access:** No create/modify access to the vehicle inventory catalog and no access to other user profiles or transaction data.

Admin User/Dealer:

- **Highest Privilege:** Admin account with the highest level of privilege.
- **Vehicle Inventory Management:** Creates and modifies all vehicle inventory catalog listings.
- **Dealership Signups:** Approves/Rejects new dealership signups.
- **Create/Edit Vehicles:** Creates and edits vehicle listings for any dealerships.
- **Access User Data:** Views user profiles and transaction data for all standard buyer accounts.
- **Purchase Reports:** Views purchase reports across all dealership inventories.

- **Generate Sales Reports:** Generates overall sales reports for the platform.
- **Access Controls:** Adds and modifies all access controls and user roles.

IV. From Operations to Pages:

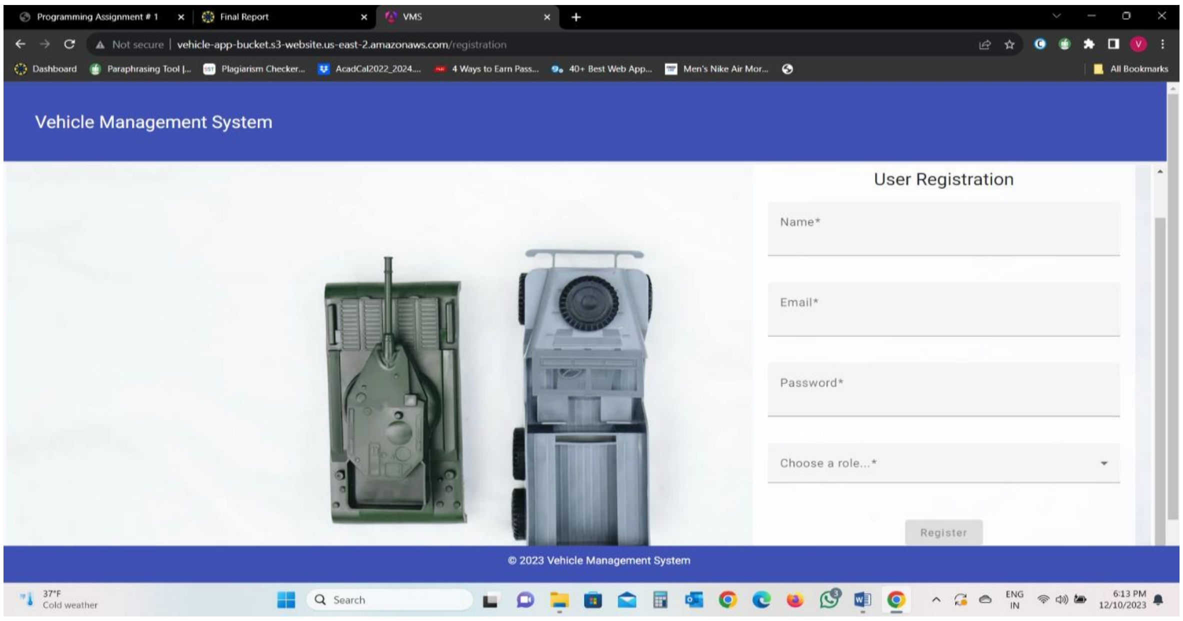
<u>Frontend:</u>

AngularJS serves as the JavaScript framework for UI components and logic, while HTML and CSS are employed for structuring and styling the user interface.

Key Frontend Pages/Features:

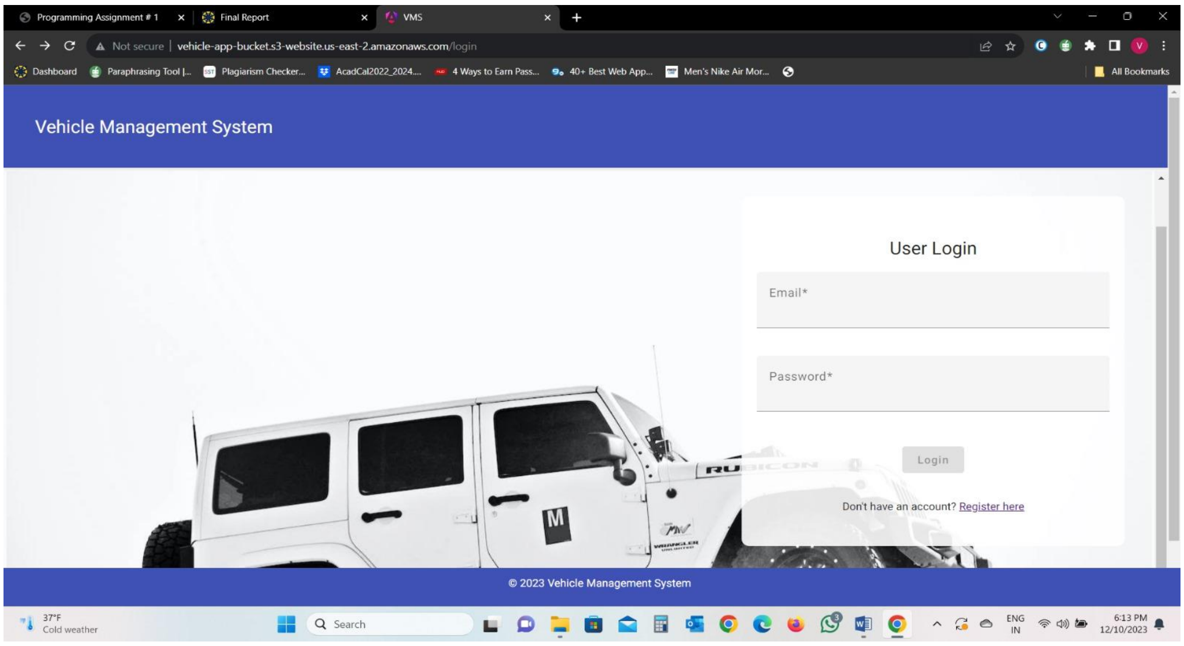
→ **Registration/user/admin**

The user will have to register themselves to login to the web application to purchase vehicles online.


vehicle-app-bucket.s3-website.us-east-2.amazonaws.com/registration
Vehicle Management System
User Registration
Name*
Email*
Password*
Choose a role...*
Register
© 2023 Vehicle Management System

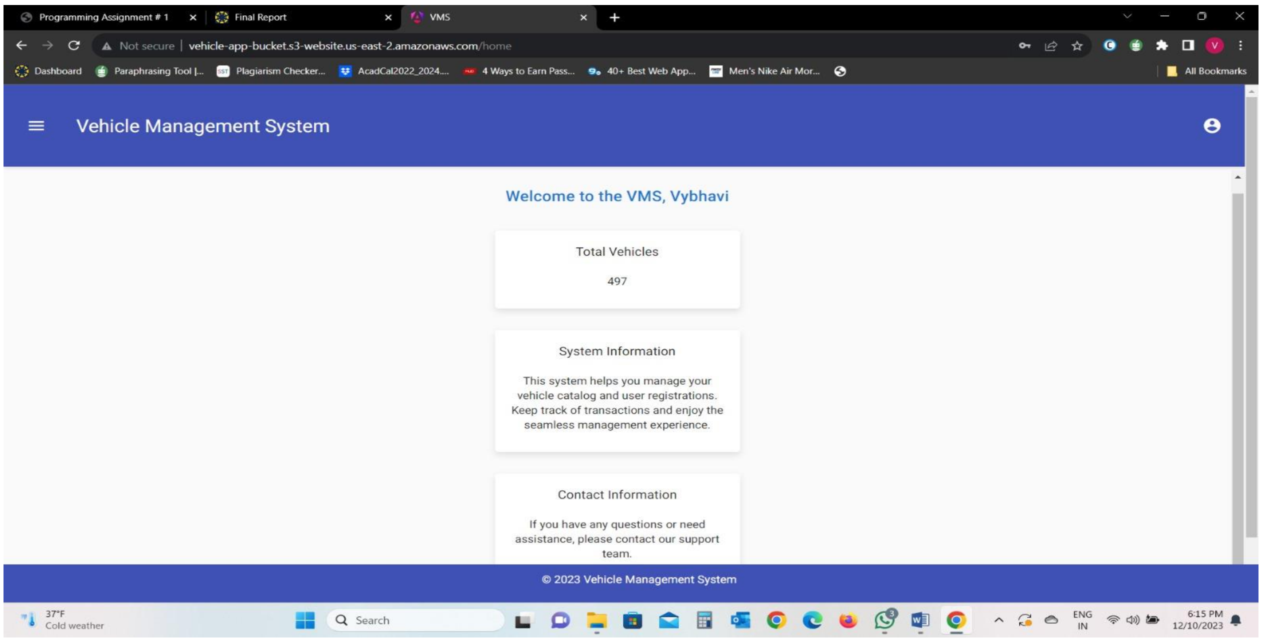
→ **Login/Signup:** Components manage authentication processes.

The user will have to enter their email and password to login to the web application for purchasing vehicle online.



Vehicle Management System
User Login
Email*
Password*
Login
Don't have an account? Register here
© 2023 Vehicle Management System

→ **Home Page:** Angular controller and view display the total count of vehicles. Here in the home page, we can see the total number of cars available for the sale.

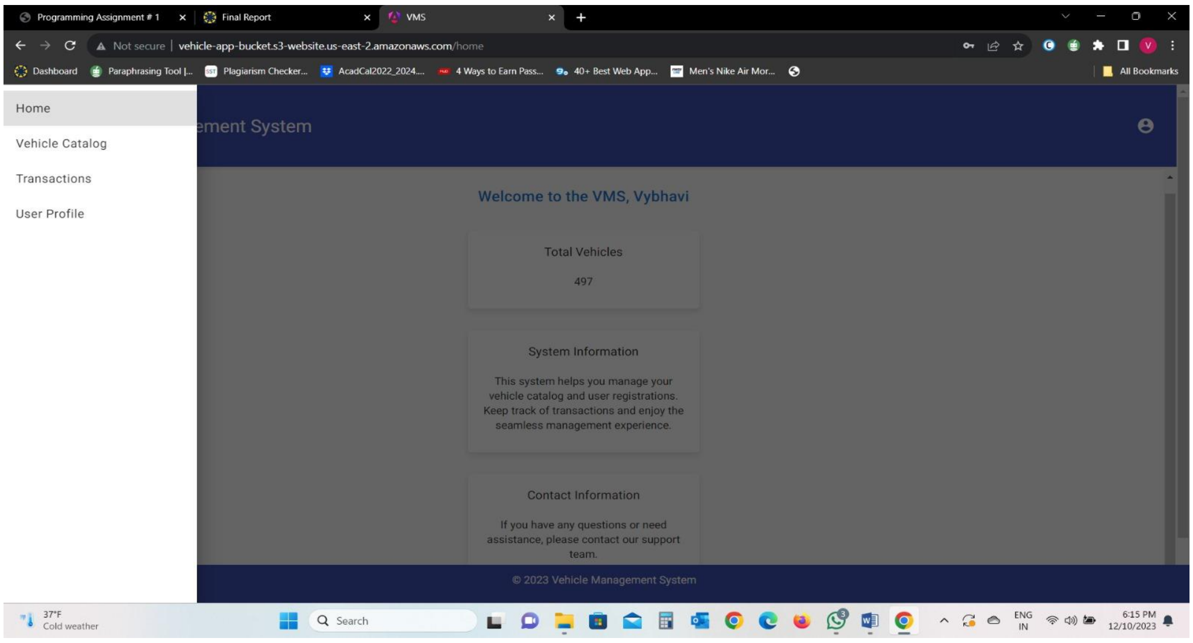


Vehicle Management System
Welcome to the VMS, Vybhavi
Total Vehicles
497
System Information
Contact Information
© 2023 Vehicle Management System

This system helps you manage your vehicle catalog and user registrations. Keep track of transactions and enjoy the seamless management experience.

If you have any questions or need assistance, please contact our support team.

→ **Menu navigation**
There are horizontal lines to the top left of the web application, when clicked it displays the menu card for the application.


Programming Assignment # 1
Final Report
VMS
Not secure | vehicle-app-bucket.s3-website.us-east-2.amazonaws.com/home
Home
Vehicle Catalog
Transactions
User Profile
ement System

Welcome to the VMS, Vybhavi

Total Vehicles

497

System Information

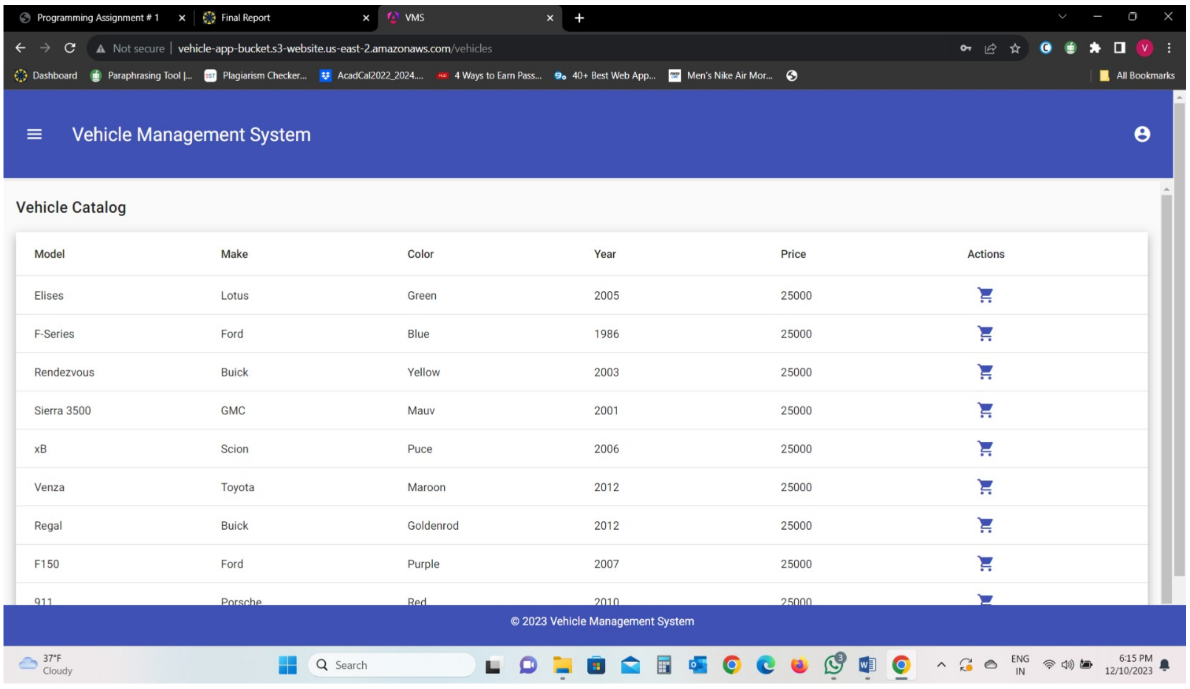
This system helps you manage your vehicle catalog and user registrations. Keep track of transactions and enjoy the seamless management experience.

Contact Information

If you have any questions or need assistance, please contact our support team.

© 2023 Vehicle Management System

→ **Vehicle Catalog Page:** Utilizes the ngRepeat directive to populate vehicle listings. From the listings in the menu page there is a vehicle catalog where u can see all the listed vehicles along with their specifications and prices to purchase .

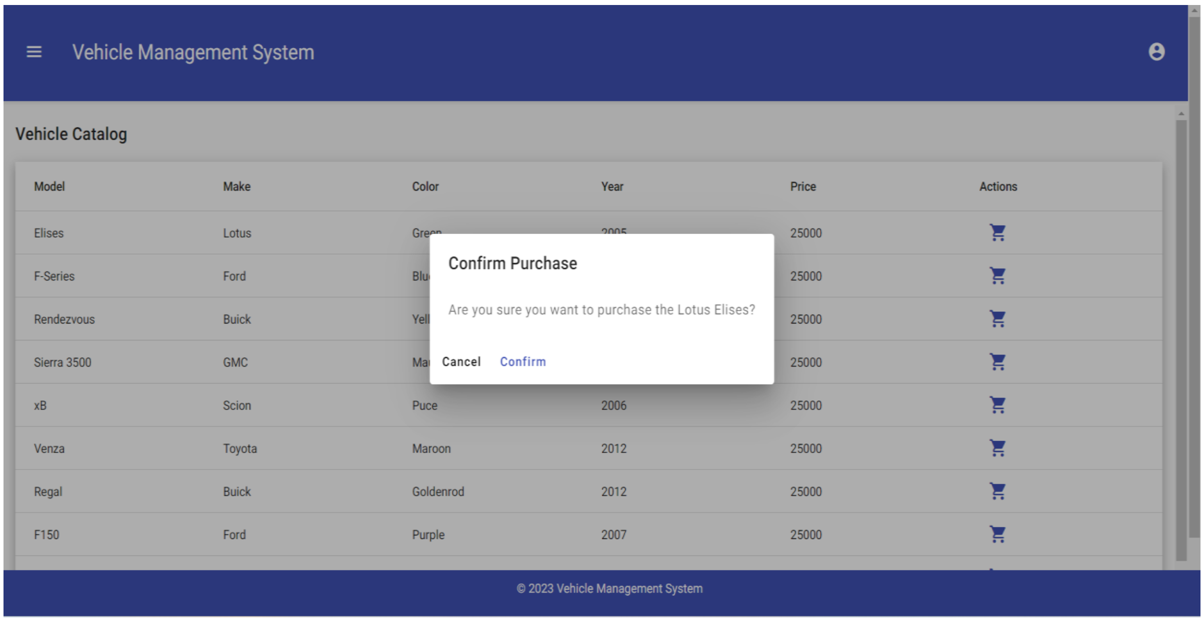

Programming Assignment # 1
Final Report
VMS
Not secure | vehicle-app-bucket.s3-website.us-east-2.amazonaws.com/vehicles
Vehicle Management System

Vehicle Catalog

Model	Make	Color	Year	Price	Actions
Elises	Lotus	Green	2005	25000	🛒
F-Series	Ford	Blue	1986	25000	🛒
Rendezvous	Buick	Yellow	2003	25000	🛒
Sierra 3500	GMC	Mauv	2001	25000	🛒
xB	Scion	Puce	2006	25000	🛒
Venza	Toyota	Maroon	2012	25000	🛒
Regal	Buick	Goldenrod	2012	25000	🛒
F150	Ford	Purple	2007	25000	🛒
911	Porsche	Red	2010	25000	🛒

© 2023 Vehicle Management System

→ **Add-to-Cart:** Controller captures vehicle IDs added for checkout.
In order to purchase the vehicle, the user needs to click on the add to cart icon which is located to the last column of the listing. And the user will be asked to confirm the purchase in order to make a successful purchase.


Vehicle Management System

Vehicle Catalog

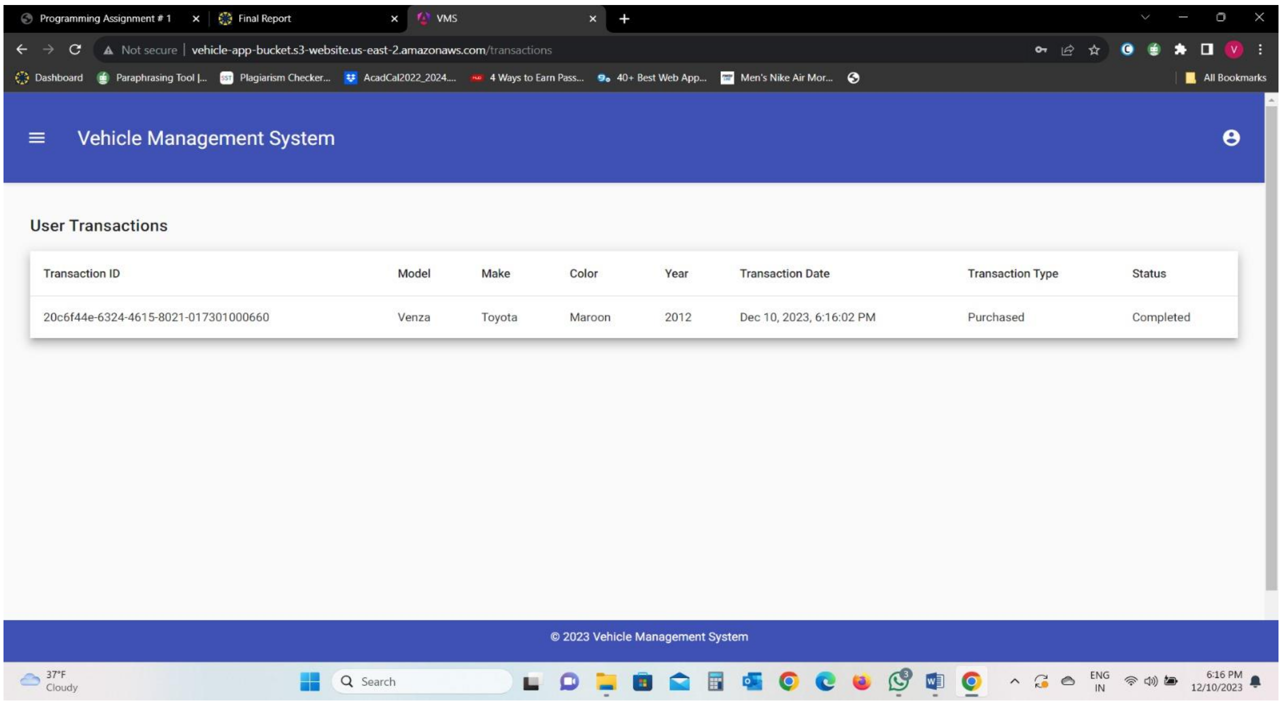
Model	Make	Color	Year	Price	Actions
Elises	Lotus	Green	2005	25000	
F-Series	Ford	Blu		25000	
Rendezvous	Buick	Yell		25000	
Sierra 3500	GMC	Ma		25000	
xB	Scion	Puce	2006	25000	
Venza	Toyota	Maroon	2012	25000	
Regal	Buick	Goldenrod	2012	25000	
F150	Ford	Purple	2007	25000	

Confirm Purchase

Are you sure you want to purchase the Lotus Elises?

© 2023 Vehicle Management System

→ **Transaction History:** Implements filters and sorting for purchase records. All the successful purchases can be seen in the user transactions history with date and time included for the purchase happened.



Vehicle Management System

User Transactions

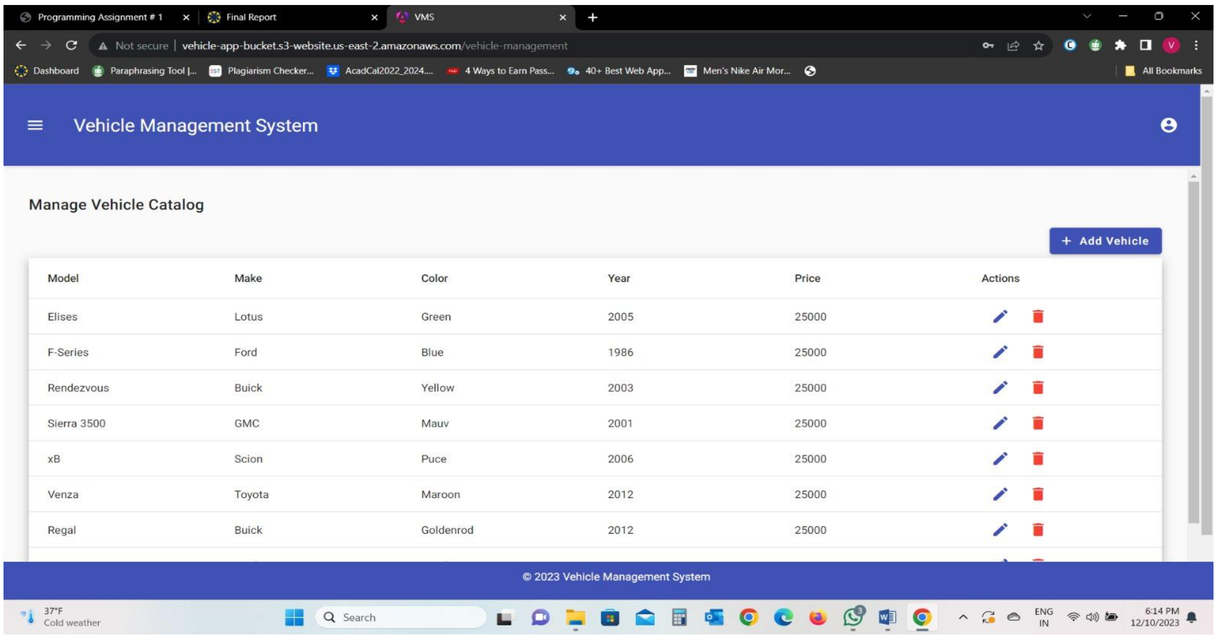
Transaction ID	Model	Make	Color	Year	Transaction Date	Transaction Type	Status
20c6f44e-6324-4615-8021-017301000660	Venza	Toyota	Maroon	2012	Dec 10, 2023, 6:16:02 PM	Purchased	Completed

© 2023 Vehicle Management System

Admin panel implentation

→ Managing Vehical catlog

The admin/seller will be able to see the vehicle management page where they can find the option to add a vehicle, modify the details of the vehicle or delete the vehicles listings.


Vehicle Management System

Manage Vehicle Catalog

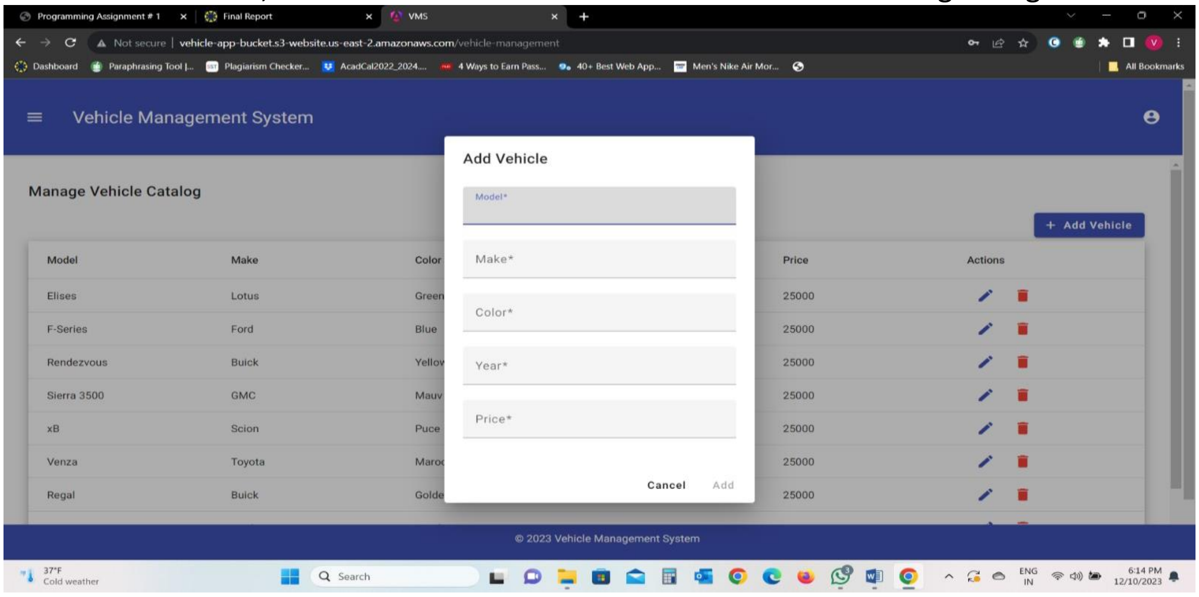
+ Add Vehicle

Model	Make	Color	Year	Price	Actions
Elises	Lotus	Green	2005	25000	
F-Series	Ford	Blue	1986	25000	
Rendezvous	Buick	Yellow	2003	25000	
Sierra 3500	GMC	Mauv	2001	25000	
xB	Scion	Puce	2006	25000	
Venza	Toyota	Maroon	2012	25000	
Regal	Buick	Goldenrod	2012	25000	

© 2023 Vehicle Management System

→ **Adding the vehicle Page**

The admin/seller will be able to enter the details of the vehicle that needs to be sold, when he clicks Add, the new vehicle can be seen in the vehicle catalog listing.


Add Vehicle

Model*

Make*

Color*

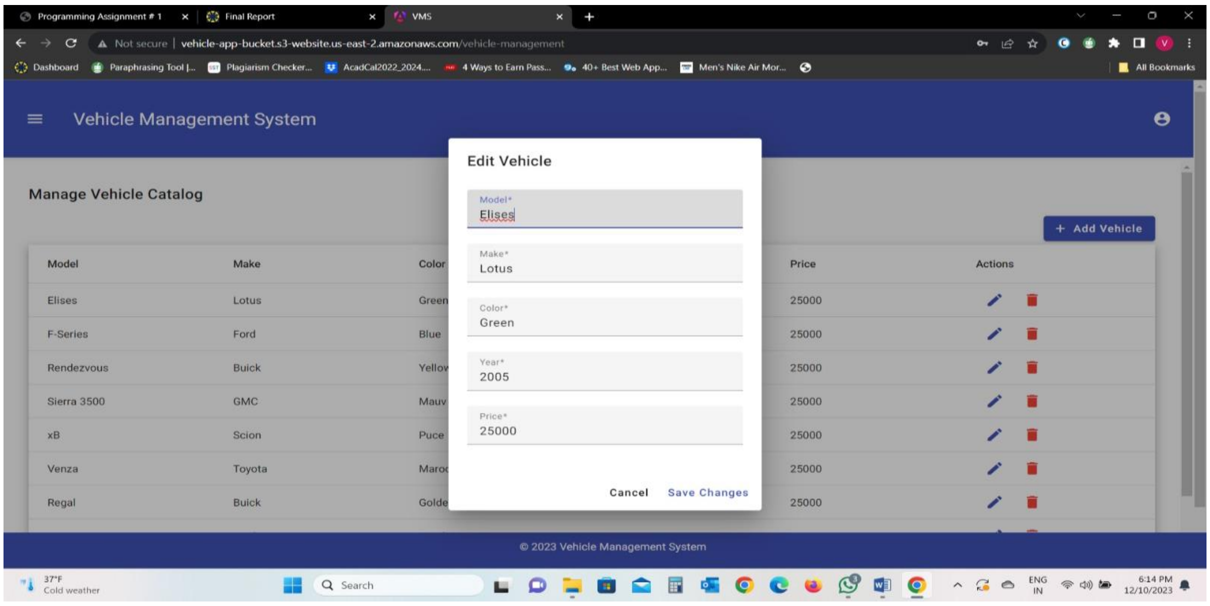
Year*

Price*

Cancel Add

→ **Modification page**

The admin/seller will be able to modify/edit the details of the vehicle that needs to be sold, when he clicks save changes, the new vehicle can be seen in the vehicle catalog listing with modified changes.



Not secure | vehicle-app-bucket.s3-website.us-east-2.amazonaws.com/vehicle-management

Vehicle Management System

Manage Vehicle Catalog

+ Add Vehicle

Model	Make	Color	Price	Actions
Elises	Lotus	Green	25000	
F-Series	Ford	Blue	25000	
Rendezvous	Buick	Yellov	25000	
Sierra 3500	GMC	Mauv	25000	
xB	Scion	Puce	25000	
Venza	Toyota	Maroc	25000	
Regal	Buick	Golde	25000	

Edit Vehicle

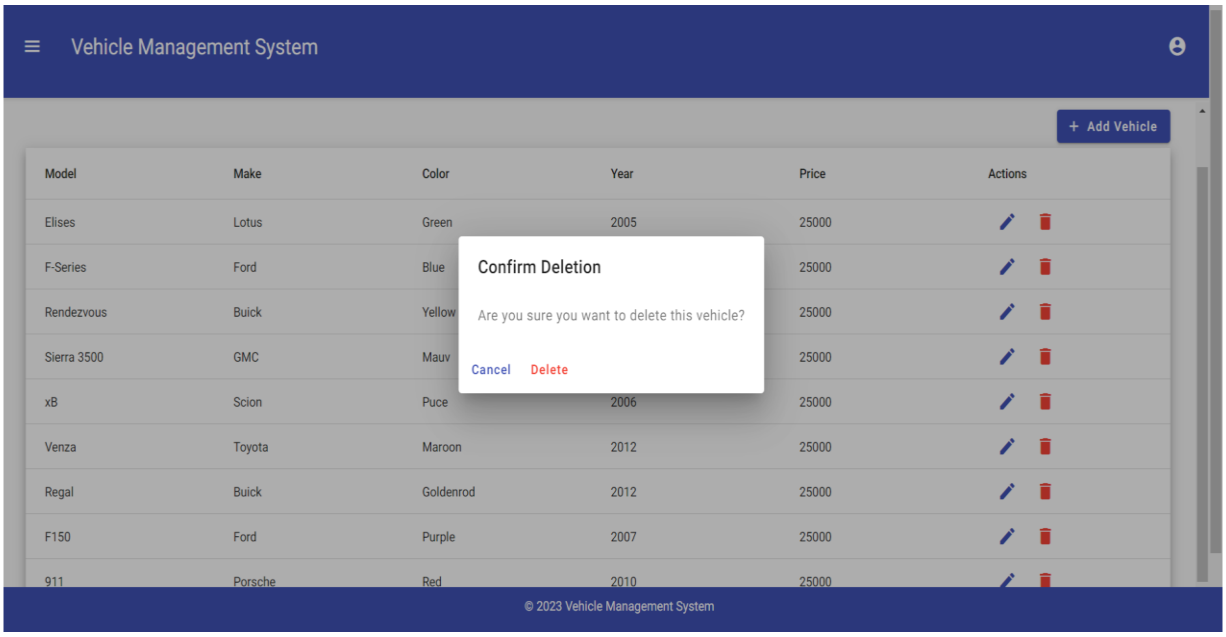
Model*: Elises
Make*: Lotus
Color*: Green
Year*: 2005
Price*: 25000

Cancel Save Changes

© 2023 Vehicle Management System

→ **Deleting the vehicle page**

The Admin/seller will be able to delete the vehicle from the vehicle catalog listing.



Vehicle Management System

+ Add Vehicle

Model	Make	Color	Year	Price	Actions
Elises	Lotus	Green	2005	25000	
F-Series	Ford	Blue		25000	
Rendezvous	Buick	Yellow		25000	
Sierra 3500	GMC	Mauv		25000	
xB	Scion	Puce	2006	25000	
Venza	Toyota	Maroon	2012	25000	
Regal	Buick	Goldenrod	2012	25000	
F150	Ford	Purple	2007	25000	
911	Porsche	Red	2010	25000	

Confirm Deletion

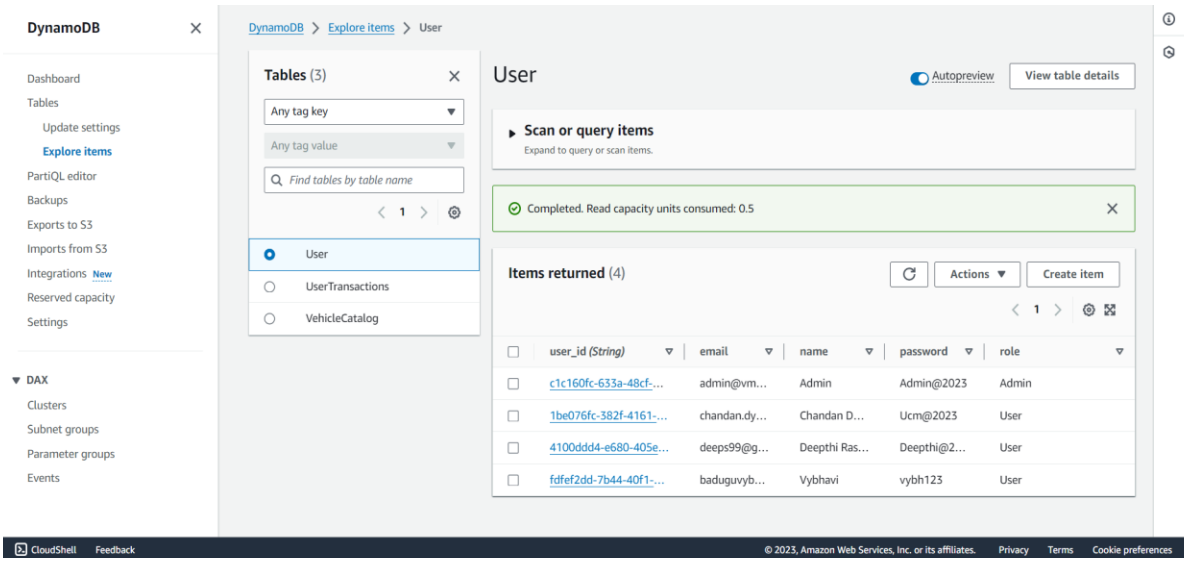
Are you sure you want to delete this vehicle?

Cancel Delete

© 2023 Vehicle Management System

V. Database design schema

User Table:



User

Completed. Read capacity units consumed: 0.5

Items returned (4)

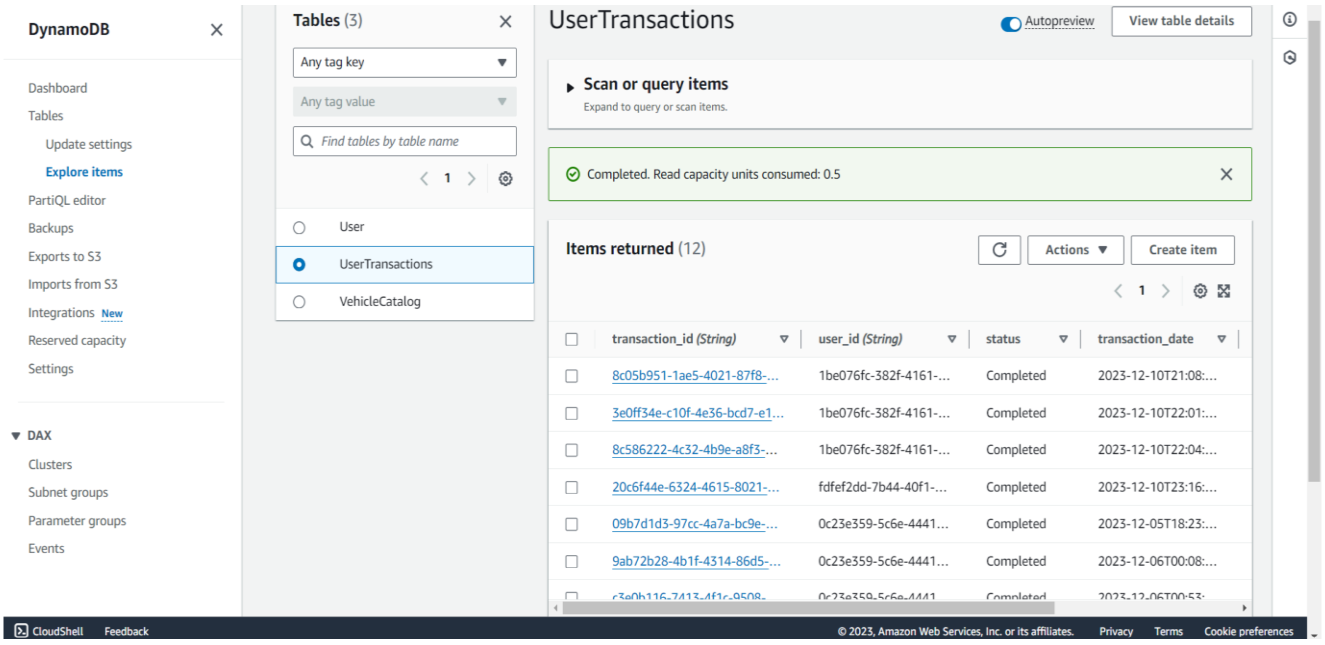
user_id (String)	email	name	password	role
c1c160fc-633a-48cf-…	admin@vm…	Admin	Admin@2023	Admin
1be076fc-382f-4161-…	chandan.dy…	Chandan D…	Ucm@2023	User
4100ddd4-e680-405e…	deeps99@g…	Deepthi Ras…	Deepthi@2…	User
fdfef2dd-7b44-40f1-…	baduguvyb…	Vybhavi	vybh123	User

© 2023, Amazon Web Services, Inc. or its affiliates.

Partition Key: user_id (String)
Sort Key: (None)

```
{
 "user_id": "unique_user_id",
 "name": "user_name",
 "email": "user_email",
 "registration_date": "registration_date",
 "status": "user_status"
```

<u>UserTransaction Table</u>:



UserTransactions

Completed. Read capacity units consumed: 0.5

Items returned (12)

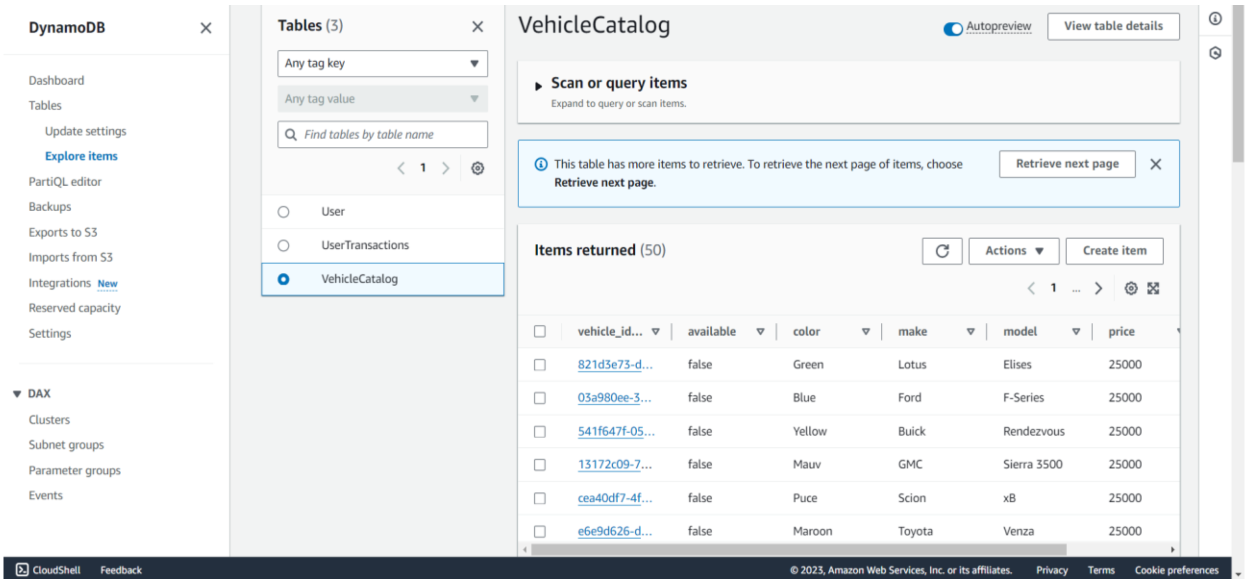
transaction_id (String)	user_id (String)	status	transaction_date
8c05b951-1ae5-4021-87f8-…	1be076fc-382f-4161-…	Completed	2023-12-10T21:08:…
3e0ff34e-c10f-4e36-bcd7-e1…	1be076fc-382f-4161-…	Completed	2023-12-10T22:01:…
8c586222-4c32-4b9e-a8f3-…	1be076fc-382f-4161-…	Completed	2023-12-10T22:04:…
20c6f44e-6324-4615-8021-…	fdfef2dd-7b44-40f1-…	Completed	2023-12-10T23:16:…
09b7d1d3-97cc-4a7a-bc9e-…	0c23e359-5c6e-4441…	Completed	2023-12-05T18:23:…
9ab72b28-4b1f-4314-86d5-…	0c23e359-5c6e-4441…	Completed	2023-12-06T00:08:…
c3e0b116-7413-4f1c-9508-	0c23e359-5c6e-4441	Completed	2023-12-06T00:53:

© 2023, Amazon Web Services, Inc. or its affiliates.

Partition Key: transaction_id (String)
Sort Key: user_id (String)

```
{
  "transaction_id": "unique_transaction_id",
  "user_id": "user_id",
  "vehicle_id": "vehicle_id",
  "transaction_date": "transaction_date",
  "transaction_type": "transaction_type",
  "status": "transaction_status"
}
```

<u>VehicleCatalog Table:</u>



VehicleCatalog

This table has more items to retrieve. To retrieve the next page of items, choose **Retrieve next page**.

Items returned (50)

vehicle_id...	available	color	make	model	price
821d3e73-d...	false	Green	Lotus	Elises	25000
03a980ee-3...	false	Blue	Ford	F-Series	25000
541f647f-05...	false	Yellow	Buick	Rendezvous	25000
13172c09-7...	false	Mauv	GMC	Sierra 3500	25000
cea40df7-4f...	false	Puce	Scion	xB	25000
e6e9d626-d...	false	Maroon	Toyota	Venza	25000

Partition Key: vehicle_id (String)
Sort Key: (None)

```
{
  "vehicle_id": "unique_vehicle_id",
  "make": "vehicle_make",
  "model": "vehicle_model",
  "color": "vehicle_color",
  "year": "vehicle_year",
  "price": "vehicle_price",
  "available": true, // Boolean indicating availability
}
```

VI. Backend Implementation

<u>Main Modules:</u>

- **User Management:** The system allows users to create accounts, login/logout, and manage their profiles securely.
- **Vehicle Catalog:** Admins and dealers can add, update, and delete vehicle listings, while users can browse and add vehicles to their shopping cart.
- **Shopping Cart:** Users can add, view, update, and remove vehicles from their shopping cart.
- **Checkout/Order Processing:** The system supports the creation, viewing, updating, and cancellation of purchase orders.
- **Transaction History:** Users can review their past transactions, and administrators have access to overall sales reports.

AWS services used for the Database layer and API layer for Backend development

Database Layer:
AWS DynamoDB, a NoSQL database, is chosen for its flexibility. The Python boto3 library seamlessly integrates the database with Lambda.

Primary Tables in DynamoDB:

- **VehicleCatalog:** Contains all vehicle listing documents.
- **UserAccounts:** Stores customer signup data.
- **PurchaseOrders:** Records transaction details for completed purchases.

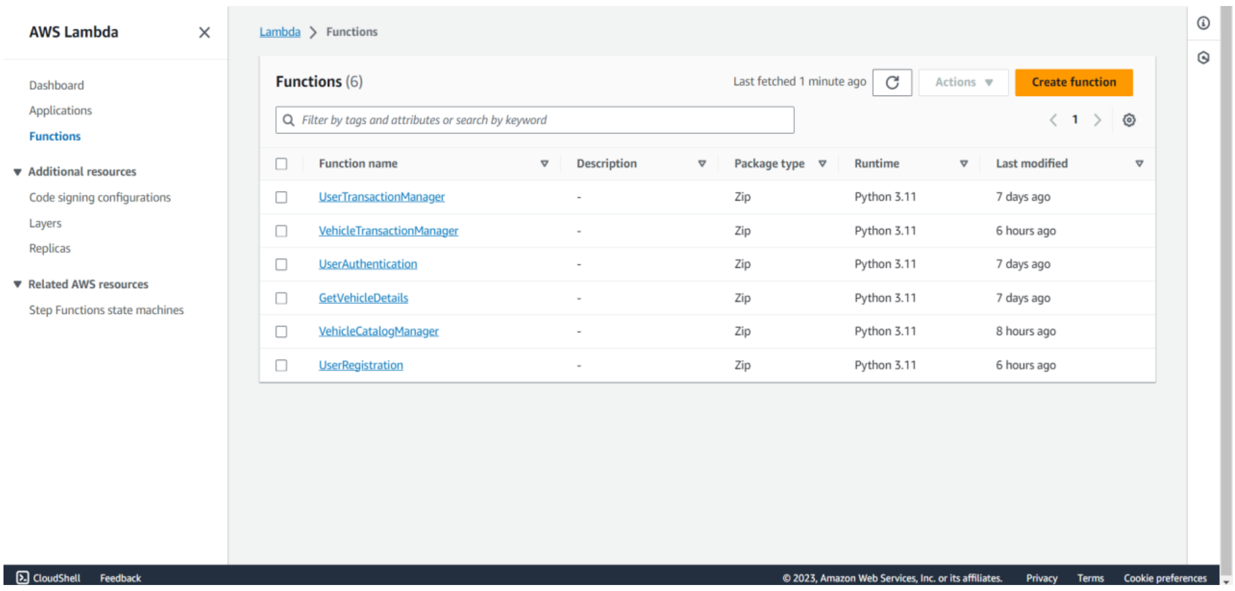
AWS Lambda Functions:

- **GetVehicles API:** Queries DynamoDB for vehicle listings.
- **CreateUser API:** Inserts signup data into the users table.
- **CompleteOrder API:** Orchestrates checkout purchase transactions.

API Gateway Endpoints Integrated with Lambda:

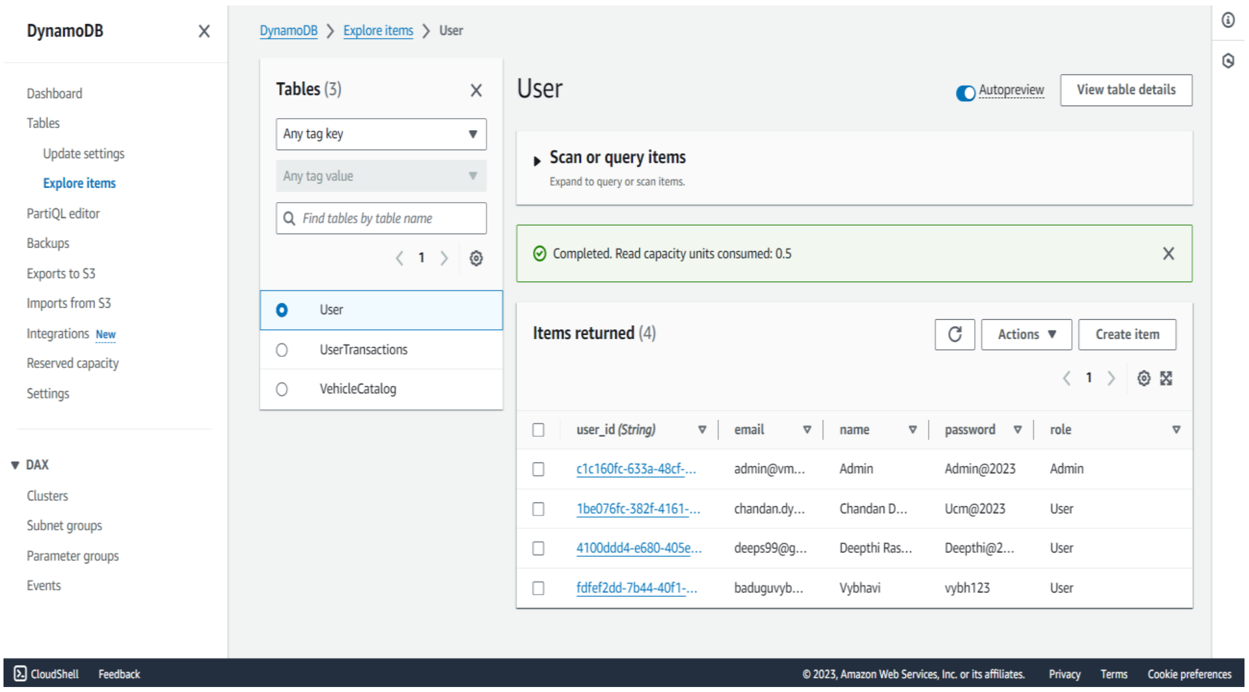
- **/vehicles GET:** Returns paginated vehicle listing JSON.
- **/accounts POST:** Validates and inserts new user accounts.
- **/orders POST:** Accepts order payload and finalizes in the database.

Screenshots of the AWS implementation:-


AWS Lambda
Dashboard
Applications
Functions
Additional resources
Code signing configurations
Layers
Replicas
Related AWS resources
Step Functions state machines
Lambda > Functions
Last fetched 1 minute ago
Actions
Create function
Filter by tags and attributes or search by keyword
CloudShell
Feedback
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Functions (6)

Function name	Description	Package type	Runtime	Last modified
UserTransactionManager	-	Zip	Python 3.11	7 days ago
VehicleTransactionManager	-	Zip	Python 3.11	6 hours ago
UserAuthentication	-	Zip	Python 3.11	7 days ago
GetVehicleDetails	-	Zip	Python 3.11	7 days ago
VehicleCatalogManager	-	Zip	Python 3.11	8 hours ago
UserRegistration	-	Zip	Python 3.11	6 hours ago



User

Completed. Read capacity units consumed: 0.5

Items returned (4)

user_id (String)	email	name	password	role
c1c160fc-633a-48cf-...	admin@vm...	Admin	Admin@2023	Admin
1be076fc-382f-4161-...	chandan.dy...	Chandan D...	Ucm@2023	User
4100ddd4-e680-405e...	deeps99@g...	Deepthi Ras...	Deepthi@2...	User
fdfef2dd-7b44-40f1-...	baduguvyb...	Vybhavi	vybh123	User

Unit Tests:

Unit tests are conducted to validate input/outputs for each API method, ensuring robust functionality.

Admin Panel:

- The Administrators have the highest privileges that help in enabling them to create and modify all vehicle inventory in the vehicle catalogue listings, manage dealership signups, view user profiles, and transactions made for the vehicles, and to generate comprehensive sales reports.

Optimization Techniques:

Some of the optimization techniques used are:-

- **Indexing:** Here additional secondary indexing is done, so that the search for the vehicles can be done in small batches and the car which is required by the user/ Buyer is selected.
- **Caching:** The cache is stored for each page in order to access it afterwards if needed.
- **Unit Testing:** Rigorous unit tests validate the inputs and outputs of each API method.
- **AWS Lambda Functions:** Critical operations, such as transactions and user authentication, are handled efficiently using AWS Lambda functions written in Python.
- **Data Storage:** AWS DynamoDB, a NoSQL database, is utilized for flexibility and scalability.

VII. Testing

Comprehensive testing has been conducted across various aspects of the Vehicle Purchasing System to ensure its reliability and functionality:

- **Functionality Testing:** All aspects of the application, including forms, databases, and cookies, have been thoroughly tested.
- **Usability Testing:** The user-friendliness of the application, including navigation and content, has been evaluated.
- **Interface Testing:** Interactions with APIs, server interactions, and error handling have been rigorously tested.
- **Compatibility Testing:** The application's compatibility with different browsers, devices, and operating systems has been verified.
- **Performance Testing:** Load and stress handling, as well as response time, have been tested to ensure optimal performance.
- **Security Testing:** The application's security features, including data encryption, user authentication, and access controls, have been extensively tested.

Debugging and Logging:

AWS CloudWatch is employed for effective debugging and monitoring, providing insights into API activities and Lambda function executions.

VIII. Individual workload division:

The workload for the Vehicle Purchasing System project was distributed among the team members as follows:

- **Vybhavi Badugu**: Frontend development, User Experience, UX testing, debugging and user authentication and authorization, AWS deployment, and documenting the following for the report.
- **Sai Venkata Tejaswini Betina**: Backend development, AWS Deployment, Database design, and documenting the following for the report.
- **Pooja Alumalla:** Web Application Testing, quality assurance, user interface (UI) design, optimization techniques implementation, and documentation of the report.