CAR RENTAL MARKETPLACE

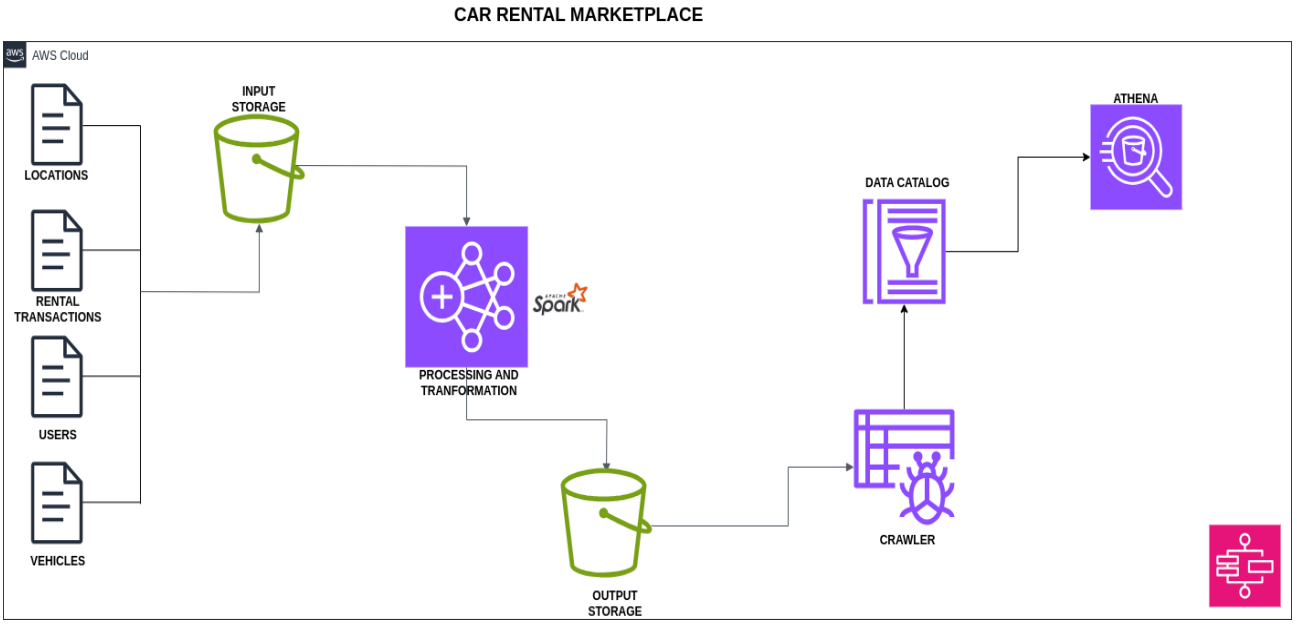
The project is based on a dataset from a car rental marketplace, a digital platform that connects vehicle owners with individuals seeking short-term rental options. This marketplace operates at scale, handling high volumes of user interactions, vehicle listings, rental transactions, and real-time location data. The company relies heavily on data to optimize pricing, enhance user experience, and ensure vehicle availability and operational efficiency. As such, robust data engineering practices and tools are essential to support data-driven decision-making.

The pipeline aims to ingest the data from the Amazon S3 bucket as the source in raw format, Process the data using EMR Cluster. The processing comes in two phases(**Spark Jobs**) written in parquet format;

1. **The Vehicle and Location Performance Metrics.**
    - Revenue per Location
    - Total transactions per location
    - Average, max, and min transaction amounts
    - Unique vehicles used at each location
    - Rental duration and revenue by vehicle type

2. **User and Transaction Analysis**
    - Total transactions per day.
    - Revenue per day
    - User-specific spending and rental duration metrics
    - Maximum and minimum transaction amounts.

FLOW DIAGRAM.



SCHEMA

TABLES

RENTAL TRANSACTION: This table consists of information on rental transactions.

Column Name	Column Type	Column Description
Rental id	String	Unique identifier for each rental transaction.
User id	String	ID of the user who made the rental.
Vehicle id	String	ID of the vehicle rented.
Rental start time	timestamp	Timestamp indicating when the rental began.
Rental end time	timestamp	Timestamp indicating when the rental ended
Pickup location	Integer	Location ID where the vehicle was picked up (FK to location table).
Dropoff location	Integer	Location ID where the vehicle was dropped off (FK to location table).
Total amount	Double	Total cost charged for the rental.

Location: This table contains information on the location of the rental.

Column Name	Column Type	Column Comment
Location id	Integer	Unique identifier for the location.
Location name	String	Descriptive name of the location (e.g., "Downtown Branch").
Address	String	Street address of the location.
City	String	City where the location is situated.
State	String	State where the location is situated.
Zip code	Integer	ZIP/postal code for the location.
Latitude	Double	Latitude coordinate for mapping purposes.
Longitude	Double	Longitude coordinate for mapping purposes.

USERS: This table consists of information on the users of the rental vehicles.

Column Name	Column Type	Column Comment
User id	String	Id referencing the user
First name	String	First name of the user
Last name	String	Last name of the user
Email	String	Users email

Phone number	String	User phone number
Driver license number	String	User driver's license number
Driver license expiry	Date	License number expiry date
Creation date	Date	Date the user was created
Is Active	integer	Status indicator (1 = active, 0 = inactive).

VEHICLES: This table consists of information on the vehicles/cars to be rented out

Column Name	**Column Type**	**Column Comment**
Active	Integer	Status flag for the vehicle (1 = available/active, 0 = inactive).
Vehicle license number	String	Official license plate number of the vehicle.
Registration name	String	Name under which the vehicle is registered.
License type	String	Type of vehicle license (e.g., personal, commercial).
Expiration date	String	Expiry date of the vehicle's registration/license.
Permit license number	String	The permit/license number was issued by the relevant authority.
Certification date	Date	Date the vehicle was certified for use.
Vehicle year	Integer	Year the vehicle was manufactured or registered.
Base telephone number	String	Phone number associated with the vehicle’s operating base.
Base address	String	Physical address of the operating base.
Vehicle id	String	Unique ID for the vehicle (FK to rental table).
Last update timestamp	String	Last recorded update for this vehicle's info.
Brand	String	Manufacturer brand (e.g.,

		Toyota, Ford).
Vehicle type	String	Type or category of the vehicle.

ORCHESTRATION

The pipeline orchestrates a data processing workflow that analyzes car rental data using various AWS services, including EMR, Glue, and Athena.

Architecture

The pipeline follows this sequence: Create EMR Cluster → Run Spark Jobs in Parallel → Trigger Glue Crawler → Query Athena (Multiple Queries) → Terminate EMR Cluster

Pipeline States and Components Explained

1. Create EMR Cluster
    - This state provisions an Amazon EMR cluster to execute Spark jobs later in the workflow.

Key Components

- EMR version 6.12.0
- Installs Spark on the cluster
- 1 master node and 2 core nodes of m5.xlarge type
- IAM role for EC2 instances in the cluster (EMR_EC2_DefaultRole)
- IAM role for EMR service (EMR_DefaultRole)
- S3 location for EMR logs
- Stores cluster information in the execution state JSON

2. Run Spark Jobs in Parallel
    - This state runs two Spark jobs concurrently using the previously created EMR cluster.

Key Components:

- Two parallel branches:

1. Location-Vehicle analysis: Processes location and vehicle performance data
2. User-Transaction analysis: Processes user behavior and transaction data

- Both jobs use spark-submit to execute Python scripts stored in S3

- References the cluster ID from the previous state
- Both jobs allow the workflow to continue even if a job fails

3. Trigger Glue Crawler

- After both Spark jobs are complete, this state triggers an AWS Glue crawler to catalog the processed data.

Key Components

- Initiates the Glue crawler named "car_rental"
- The crawler discovers and catalogs the schema of processed data

4. Wait for Crawler

- This state introduces a 60-second delay to allow the Glue crawler to complete its operation.

5-7. Athena Queries

- These states execute three sequential Athena queries to analyze the processed data.

Key Components

- Each query extracts specific business insights:
  1. Location with highest revenue
  2. Most popular vehicle type
  3. Top 5 users by spending
- All queries store results in the same S3 path
- Queries are executed sequentially

8. Terminate EMR Cluster

- This final state shuts down the EMR cluster to avoid unnecessary costs.

Key Components

- References the cluster ID from the initial state
- Ensures the workflow waits for confirmation that the cluster is terminated

Monitoring and Troubleshooting

1. Execution Monitoring:

- - Step Functions provide visual execution tracking
- - Check the status of each state and transition

2. Logs and Debugging
   - Review EMR logs in S3
   - Check Athena query results
   - Review CloudWatch logs for the state machine execution

3. Common Issues:
   - Permission errors: Verify IAM roles have all required policies
   - Resource limits: Check AWS service quotas if cluster creation fails
   - S3 access: Confirm all S3 paths exist and are accessible
   - Glue crawler: Ensure it's properly configured for the data format

Security Considerations

1. IAM Roles: Follow the least privilege principle by limiting permissions
2. S3 Buckets: Implement appropriate bucket policies
3. EMR Security: Consider using security configurations for encryption
4. Networking: Use VPC and security groups to control access

Cost Optimization

1. EMR Cluster:
   - The pipeline terminates clusters after use
   - Consider using Spot instances for core nodes
   - Evaluate instance types based on workload