## 

Project Assignment

Ajay Agarwal

Principal Software Engineer

# 

# 

# Overview

Develop a software system which will enable users to create/view/update/delete data for an Electric Vehicle. Customer. Users should have the ability to update Base MSRP based on make and model.

# Functional Requirements

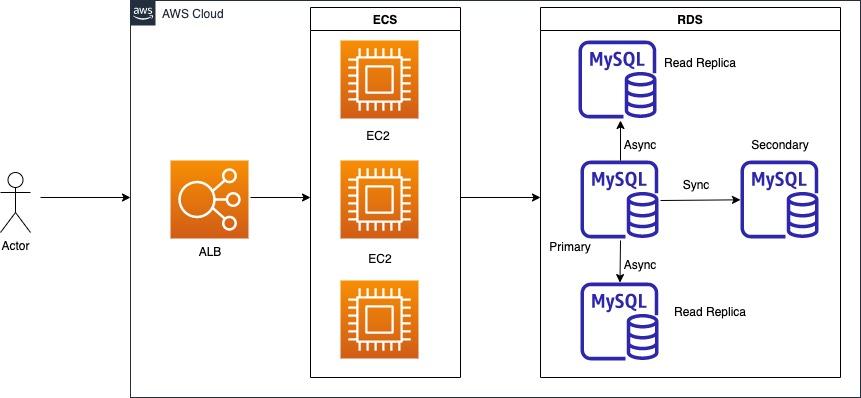
* As a user I should be able to create, update, view and delete electric vehicle data
* As a user I should be able to update Base MSRP based on make and model.

# Non Functional Requirements

* Operations should have sufficient logging, but consider privacy indication with logging
* Operations should leverage telemetry for observability
* Provide Helm chart for project deployment. Utilize a local container registry ([example](https://www.paulsblog.dev/how-to-install-a-private-docker-container-registry-in-kubernetes/)) for publishing and retrieving image.

# 

# Physical Diagram



# Assumptions

* The application is deployed in a single region
* If higher availability is required we can do a multi region deployment.
* The UI is out of scope for this project
* City and County are unique across states
* I am not sure of the Electric Utility company names. Even though they are separated by "|" and "||", not 100% certain what they represent. Hence treating it like a single value.
* VIN is unique. In the dataset since only the first 10 chars are provided, I have added a running number to make the data unique.

# Technology Stack

* Java, Spring Boot for Rest API
* MySql RDBMS for Data Store. Did not use PostGreSQL

# 

# Data Model

## Reference Data

### Name: mststate

This is a reference table which will store the various states. This table might be loaded into the application server memory at startup

| **Column Name** | **Type** | **Size** | **Constraint** | **Comments** |
| --- | --- | --- | --- | --- |
| id | INT |  | Primary Key  Not Null |  |
| state | varchar | 100 | Not Null  Unique |  |

### Name: mstcounty

This is a reference table which will store the various counties. This table might be loaded into the application server memory at startup

| **Column Name** | **Type** | **Size** | **Constraint** | **Comments** |
| --- | --- | --- | --- | --- |
| id | INT |  | Primary Key  Not Null |  |
| county | varchar | 100 | Not Null  Unique |  |

### 

### Name: city

This is a reference table which will store the cities. This table might be loaded into the application server memory at startup

| **Column Name** | **Type** | **Size** | **Constraint** | **Comments** |
| --- | --- | --- | --- | --- |
| id | INT |  | Primary Key  Not Null |  |
| city | varchar | 100 | Not Null  Unique |  |

## Other Reference Data

## Entities

### evdata

This is the master table which will store Electric Vehicle Data

| **Column Name** | **Type** | **Size** | **Constraint** | **Comments** |
| --- | --- | --- | --- | --- |
| VIN | varchar | 100 | Primary Key  Not Null |  |
|  |  |  |  |  |

# Rest Api Design

## Entity: evdata

/api/v1/evdata

| Method | URL | Path Variables | Return Codes |
| --- | --- | --- | --- |
| GET | /{id} | id | 200 OK  404 Not Found |
| POST |  |  | 201 Created  400 Bad Request |
| PUT | /{id} | id | 200 OK  400 Bad Request |
| DELETE | /{id} | id | 200 OK |

### Notes

### Request Format

### Response Format

## Entity: evdata/actions

/api/v1/actions/

| Method | URL | Path Variables | Return Codes |
| --- | --- | --- | --- |
| POST |  |  | 201 Created  400 Bad Request |

### Notes

### Request Format

### Response Format

### 

# Technical Roadmap (Pending Items)

* Unit Testing (JUnit, Mockito)
* API Documentation (Swagger)
* Logging (Log4j)
* Validations (Hibernate Validator)
* DB Migration Framework
* Authentication and Authorization
* Reporting
* Archiving
* Continuous Integration (CI) Pipeline
  + Maven
  + Jenkins
  + Sonarqube
  + Java Code coverage (Jacoco)
  + Artifact Repository (Nexus/Artifactory)
  + Docker
* Continuous Delivery (CD) Pipeline
  + Regression Testing framework
  + Performance/Load Testing framework
  + Mock Server (wiremock)
  + Infrastructure as Code (Terraform)
  + Blue/Green Deployment
* Infrastructure (AWS)
  + Application Load Balancer
  + Elastic Container Service (ECS)
  + EC2
  + Route53
  + RDS (MySQL)
  + AutoScaling
* Logging and Monitoring
  + Splunk
  + Cloudwatch
  + New Relic
  + Prometheus
  + Graffana
* DB Backup
  + Automated if we are using RDS
* Alerting
  + PagerDuty