**Vehicle Telematics and Analytics Platform**

**1. Project Overview**

**1.1 Introduction**

This Vehicle Telematics and Analytics Platform is designed to monitor, analyze, and process data from vehicle fleets in real time. The system is divided into several microservices, each with a specific responsibility, ensuring scalability, modularity, and ease of maintenance.

**1.2 Key Features**

* Real-time vehicle data ingestion and processing.
* Data caching for low-latency access.
* Notification system for alerts based on analytics.
* Scalable microservices architecture.
* OAuth2-based authentication and authorization.
* Efficient message communication using Kafka.

**1.3 Architecture**

This platform consists of several interconnected microservices:

* **User Management Microservice (UMS)**: Handles user registration, authentication, and authorization.
* **Vehicle Data Microservice (VDMS)**: Ingests real-time vehicle data.
* **Analytics Microservice (AMS)**: Consumes data from Kafka, performs analytics, and provides insights.
* **Notification Microservice (NMS)**: Sends notifications based on analytics results.
* **Redis Cache**: Caches frequently accessed data.
* **Kafka**: Handles asynchronous data communication between services.

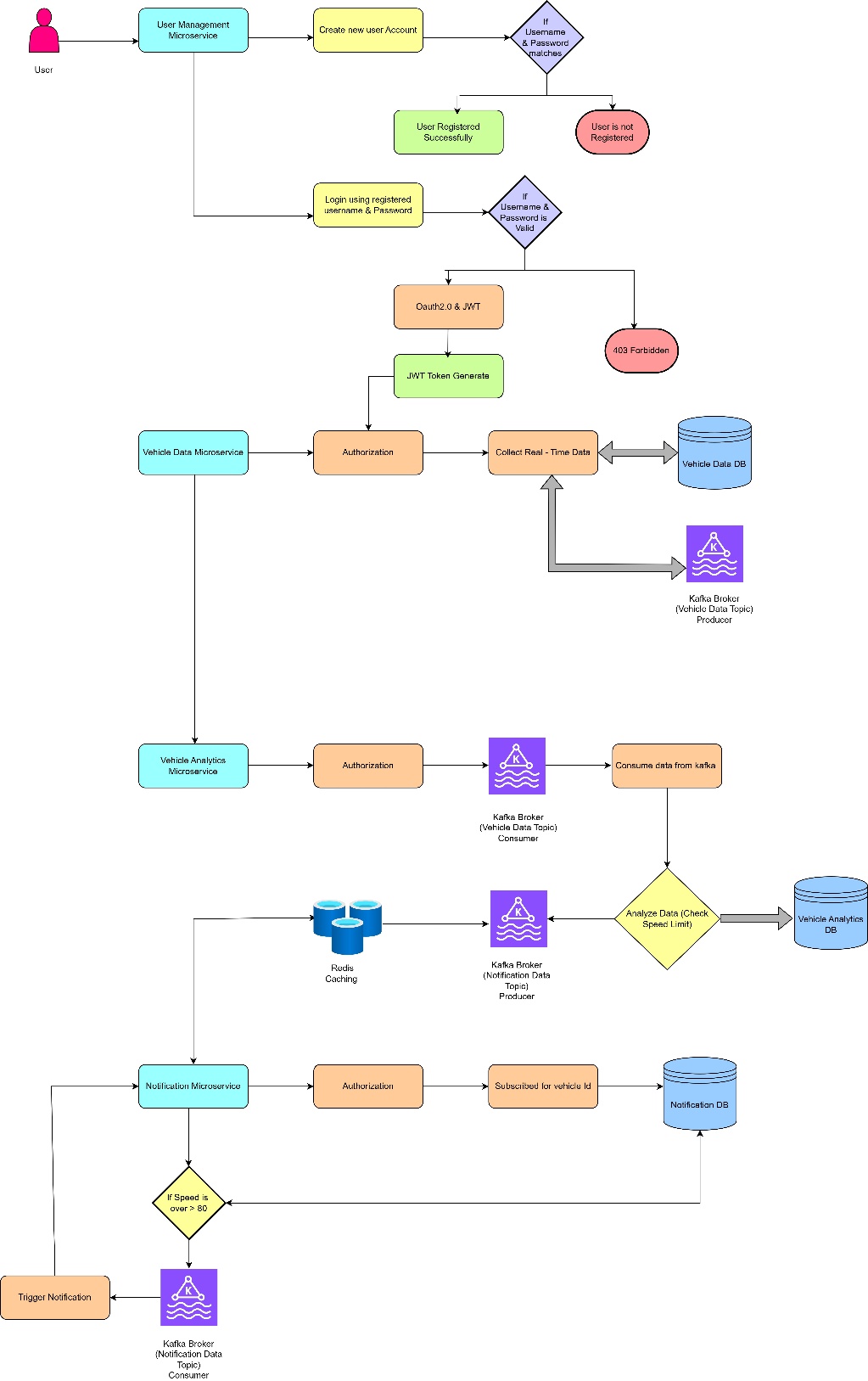
**2. Technology Stack**

|  |  |
| --- | --- |
| **Component** | **Technology** |
| Backend | Java, Spring Boot |
| Database | MySQL |
| Caching | Redis |
| Messaging | Kafka |
| Authentication | OAuth 2.0, JWT |

## ****3. System Architecture****

### ****3.1 High-Level Architecture****

The system architecture is structured in a microservices pattern, each service performing specific functions and communicating through Kafka or HTTP REST APIs. Redis is used for caching, while MySQL stores persistent data. The following diagram shows the data flow across services:



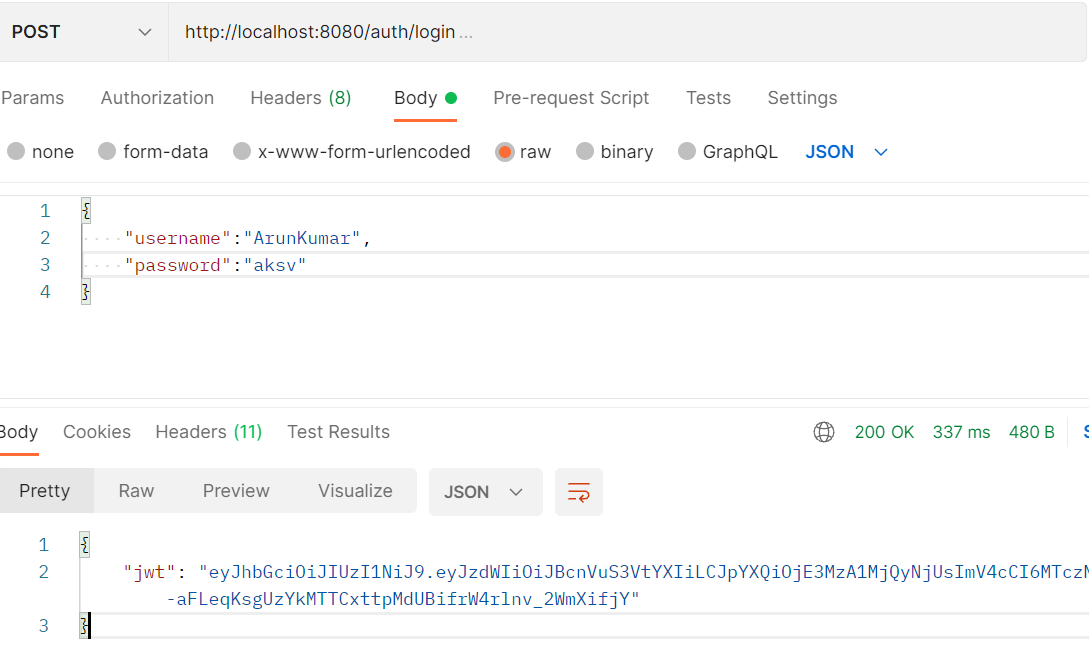
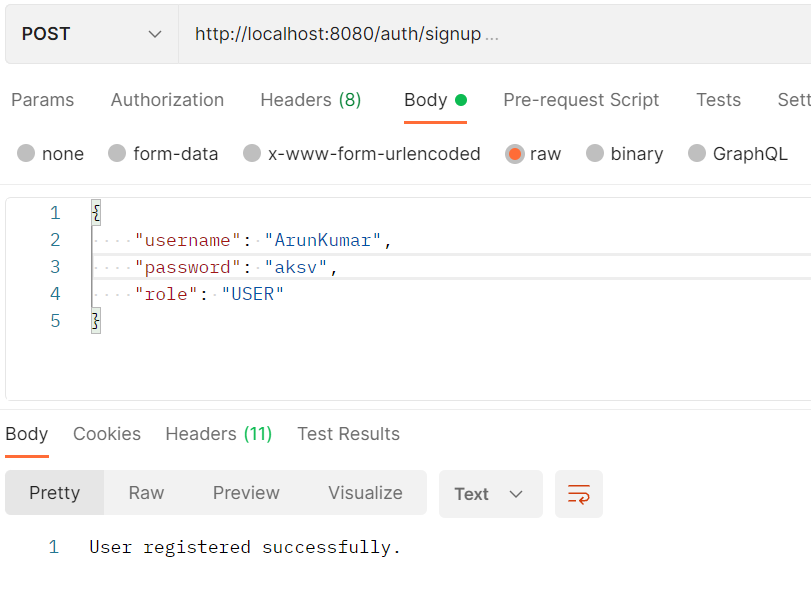
### ****3.2 Flow Explanation:****

1. **User Authentication**:  
   Users authenticate via the User Management Microservice (OAuth2), which provides an access token to interact with APIs.
2. **Vehicle Data Microservice**:  
   The Vehicle Data Microservice gathers real-time data (e.g., speed, location) from vehicles. It publishes this data to a Kafka topic.
3. **Kafka Broker**:  
   Kafka broadcasts the vehicle data to subscribers (Analytics and Notification Microservices). Kafka ensures real-time, asynchronous communication.
4. **Analytics Microservice**:  
   The Analytics Microservice consumes the vehicle data from Kafka, processes it (e.g., calculating the average speed), and stores the analytics in MySQL.
5. **Notification Microservice**:  
   The Notification Microservice monitors the Kafka topic for significant events (e.g., over-speeding) and sends alerts or notifications accordingly.
6. **Redis**:  
   Frequently accessed data, such as vehicle locations, is cached in Redis to speed up subsequent requests.
7. **MySQL**:  
   The MySQL database stores structured information like vehicle details, analytics, and user data.

**4. API**

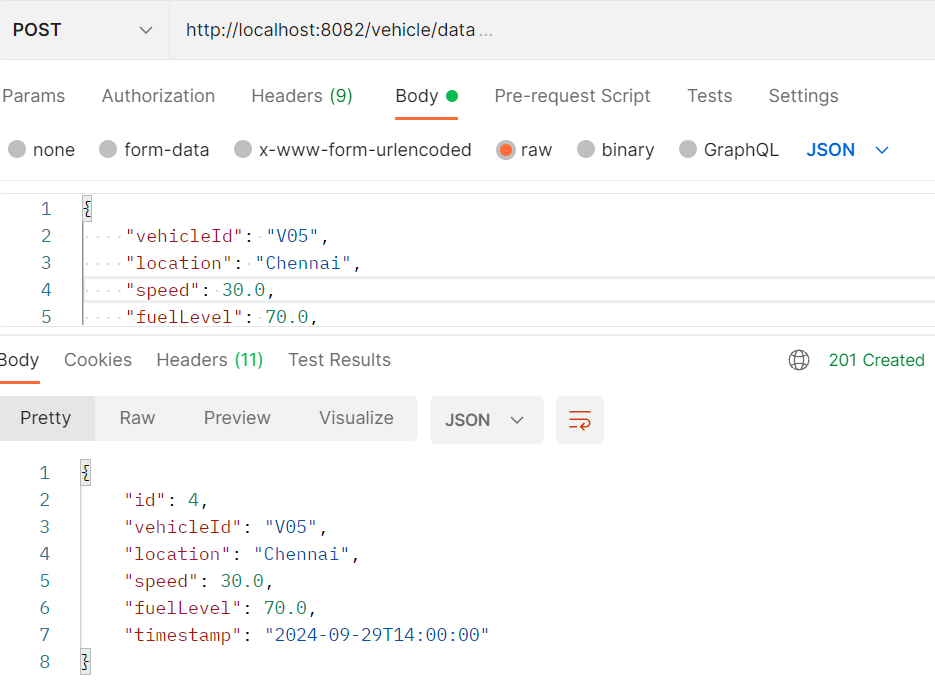
### ****4.1 User Management Microservice (UMS)****

* **Endpoint**: /api/auth/register
  + **Method**: POST
  + **Description**: Registers a new user.
  + **Parameters**: username, password, email
* **Endpoint**: /api/auth/login
  + **Method**: POST
  + **Description**: Authenticates a user and provides a token.
  + **Parameters**: username, password



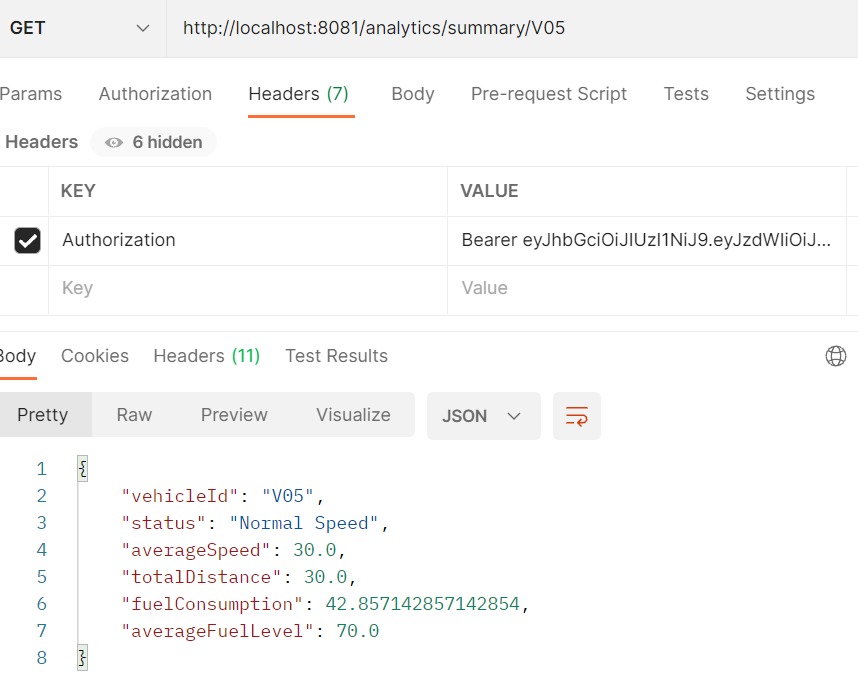
### 4.2 ****Vehicle Data Microservice (VDMS)****

* **Endpoint**: /api/vehicle-data/publish
  + **Method**: POST
  + **Description**: Publishes vehicle data to Kafka.
  + **Parameters**: vehicleId, timestamp, locationData



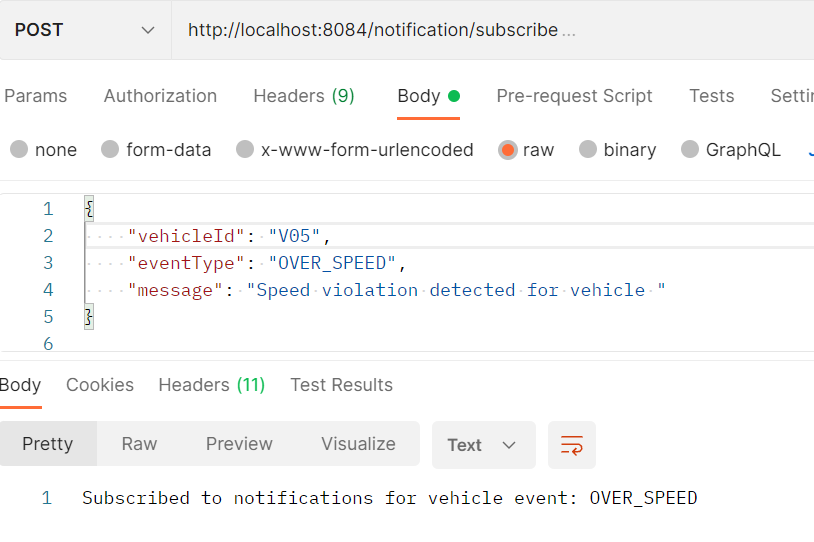
### ****4.3 Analytics Microservice (AMS)****

* **Endpoint**: /api/analytics/report
  + **Method**: GET
  + **Description**: Provides an analytics report.
  + **Parameters**: vehicleId



### ****4.4 Notification Microservice (NMS)****

* **Endpoint**: /api/notifications/send
  + **Method**: POST
  + **Description**: Sends notifications based on vehicle analytics.



## ****6. Redis Caching****

* **Purpose**: To store frequently accessed analytics data and reduce latency.
* **Caching Strategy**: Least Recently Used (LRU)
* **Usage**: The Analytics Microservice caches results of frequent analytics queries to optimize performance.

## ****7. Kafka Configuration****

* **Broker Address**: Defined in the .env file.
* **Topics**:
  + vehicle\_data\_topic: Used by VDMS to publish vehicle data.
  + analytics\_topic: Used by AMS to receive and process data.
* **Configuration**: Each service is configured with proper ProducerConfig and ConsumerConfig in application.yml