Self Driving RC Car

Case Study

USN : 1MS17IS094

Name : Rohit P N

Course Code : ISEC2

Course Name : Internet of Things

Department of ISE, Ramaiah Institute of Technology

Step 1: Purpose and Requirements

Purpose: A self driving RC car that autonomously drives on a track, while sending real-time analytics to a web dashboard.

Behavior: The car should have 2 controllable parameters, a binary throttle and a steering control. These will allow the car to go forward, stop and take a left or a right turn autonomously by looking at the path ahead.

System Management Requirement: The system should provide remote monitoring.

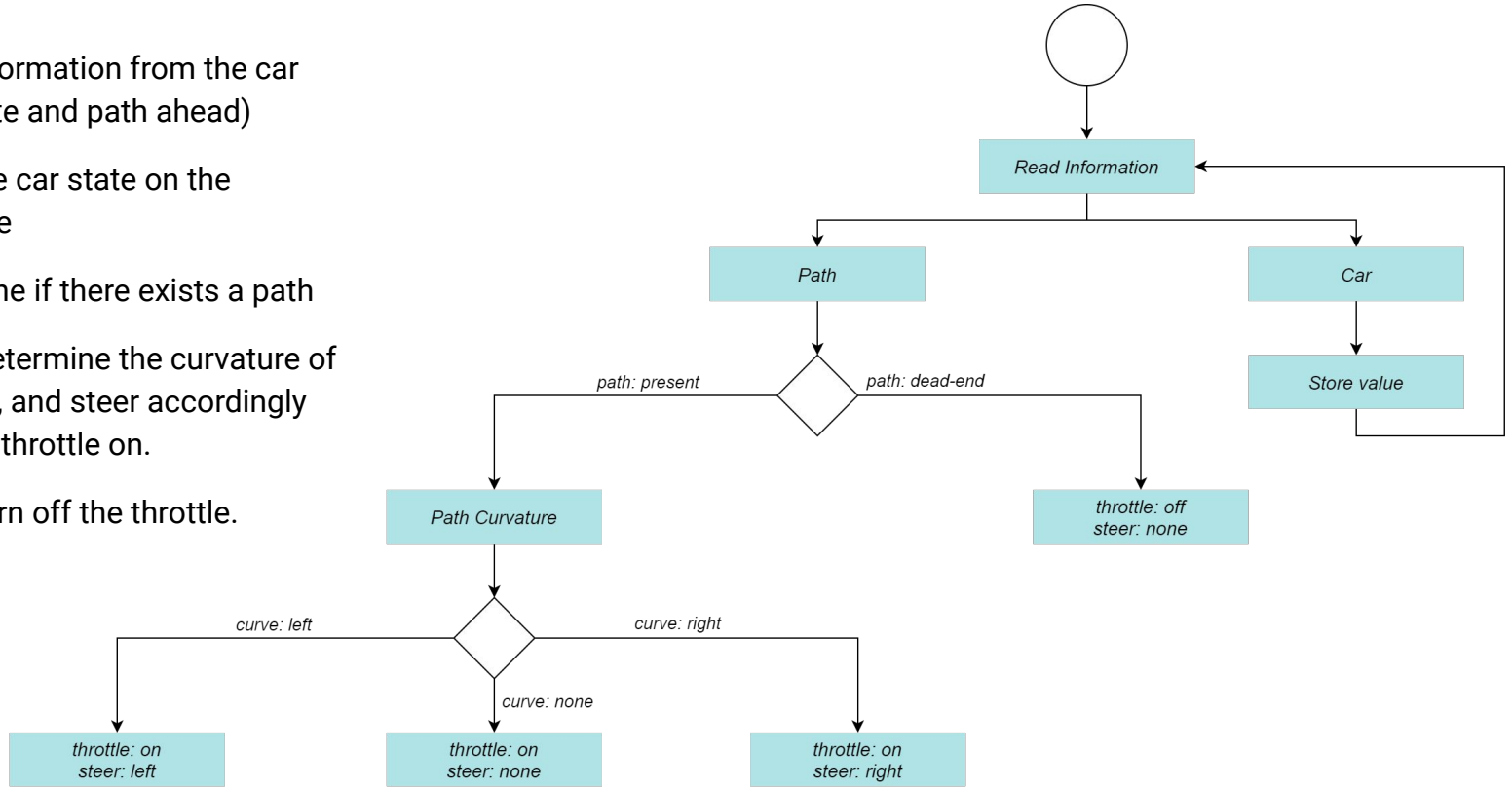
Data Analysis Requirement: The system should push real-time data to the cloud.

Application Deployment Requirement: The application should be deployed locally with servers for processing on the cloud.

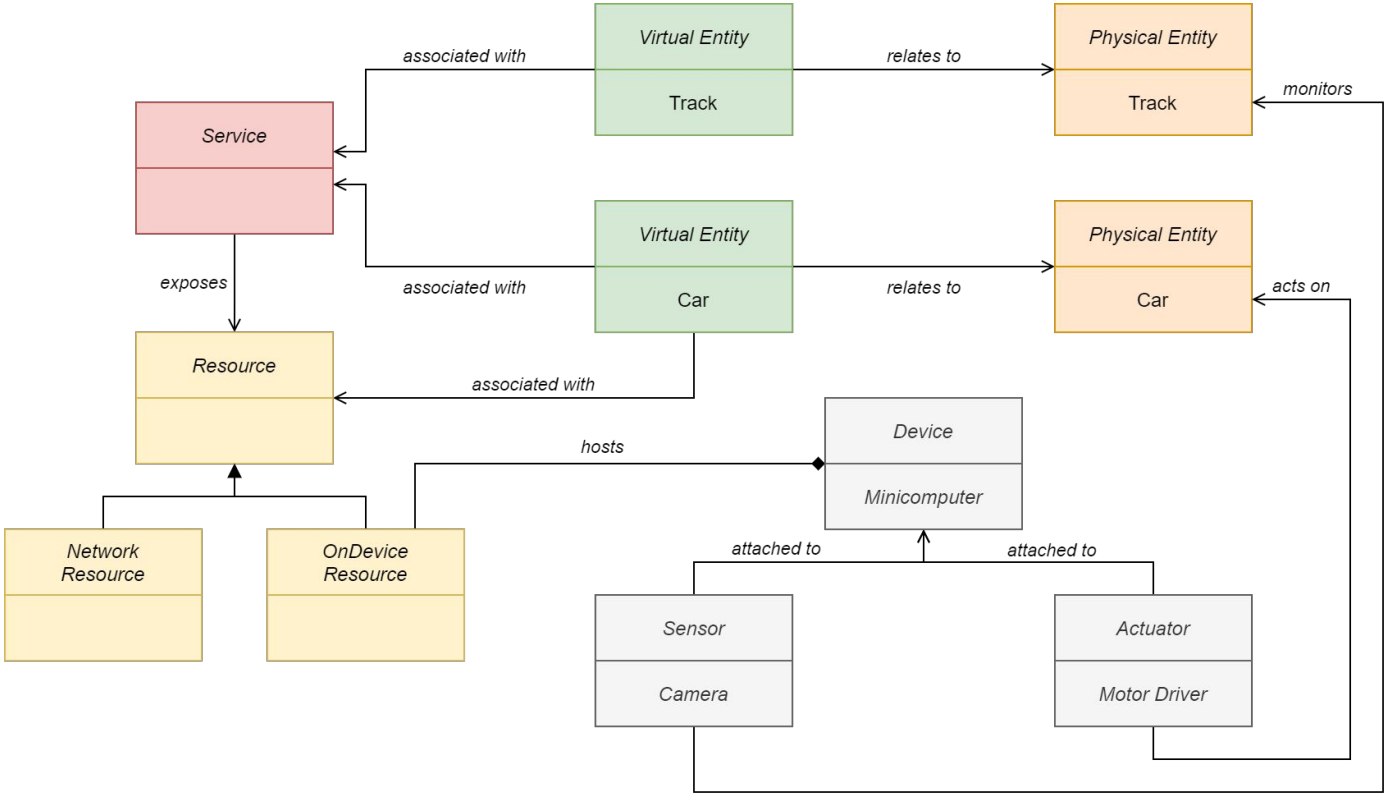
Security Requirement: The system should have basic user authentication capability.

Step 2: Process Specification

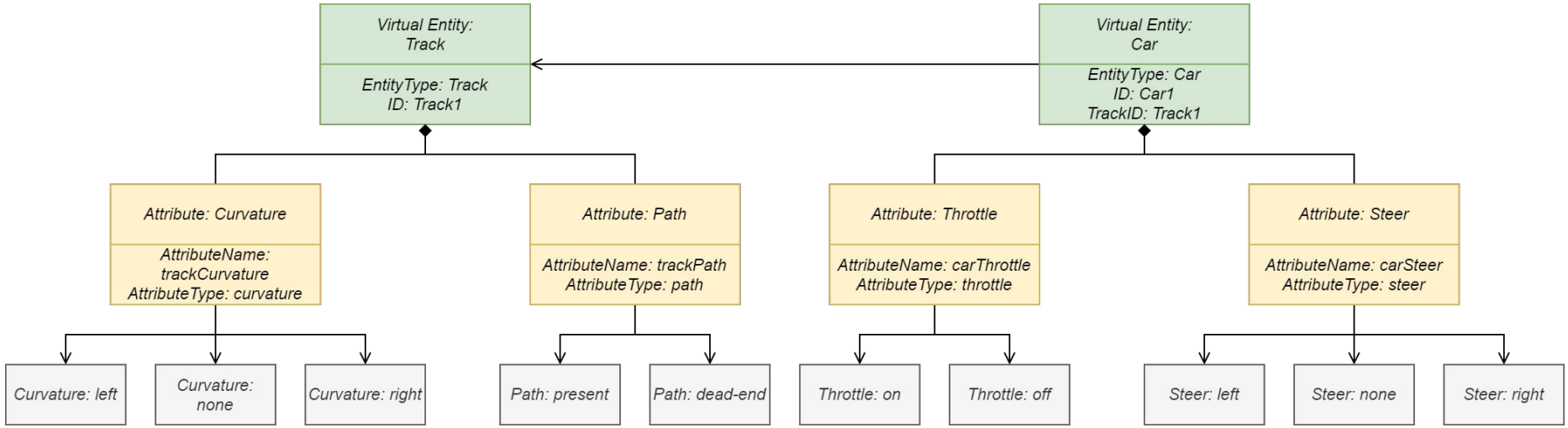
- Read information from the car (Car state and path ahead)
- Store the car state on the database
- Determine if there exists a path
- If yes, determine the curvature of the path, and steer accordingly with the throttle on.
- If not, turn off the throttle.



Step 3: Domain Model Specification



Step 4: Information Model Specification



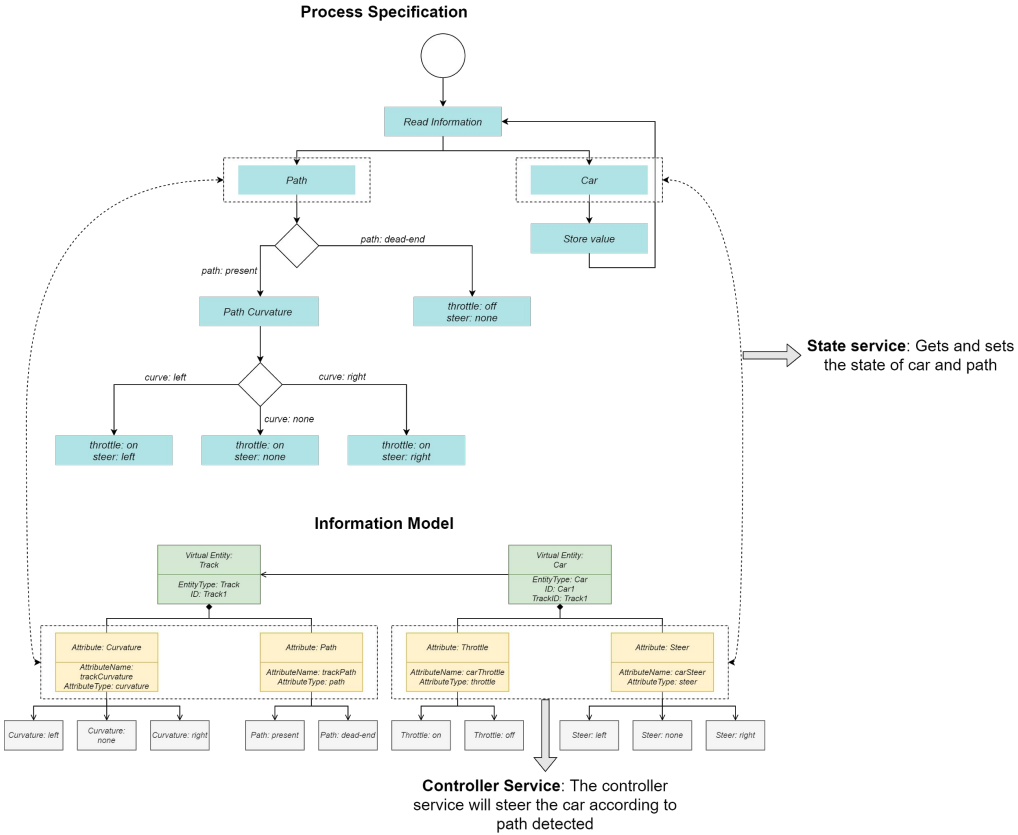
Track's Attributes:

- **Curvature**: The angle of curvature of the path in front of the car
- **Path**: If there is a path in front of the car or not

Car's Attributes:

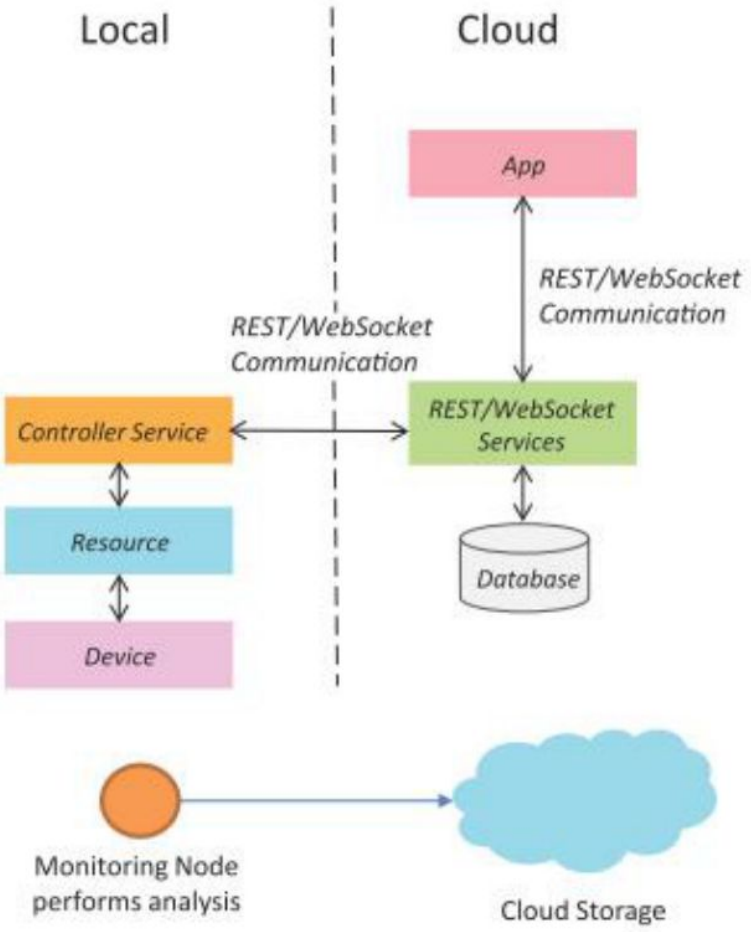
- **Throttle**: Whether the throttle is on or off
- **Steer**: The steering direction of the car

Step 5: Service Specifications



1. **State Service:** Gets and sets the state of the car and gets the state of the path

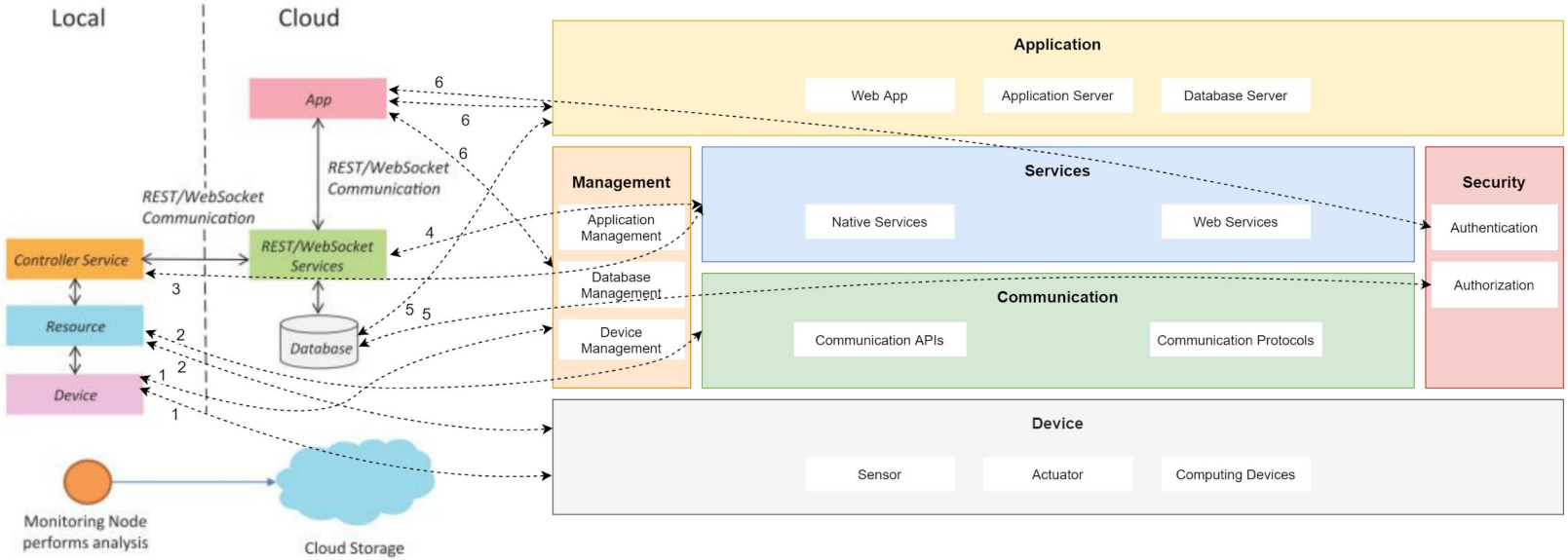
2. **Controller Service:** The controller service will steer the car and control its movements according to the path detected



Step 6: IoT Level Specification

- This system is an **IoT Level 2 system**
- A level-2 IoT system has a single node that performs sensing and/or actuation and local analysis
- Data is stored in the cloud and application is usually cloud-based
- Level-2 IoT systems are suitable for solutions where the data involved is big, however, the primary analysis requirement is not computationally intensive and can be done locally itself

Step 7: Functional View Specification



1. IoT device maps to the Device FG (sensors, actuators, devices, computing devices) and the Management FG (device management)

2. Resources map to the device FG (on-device resource) and Communication FG (communication APIs and protocols)

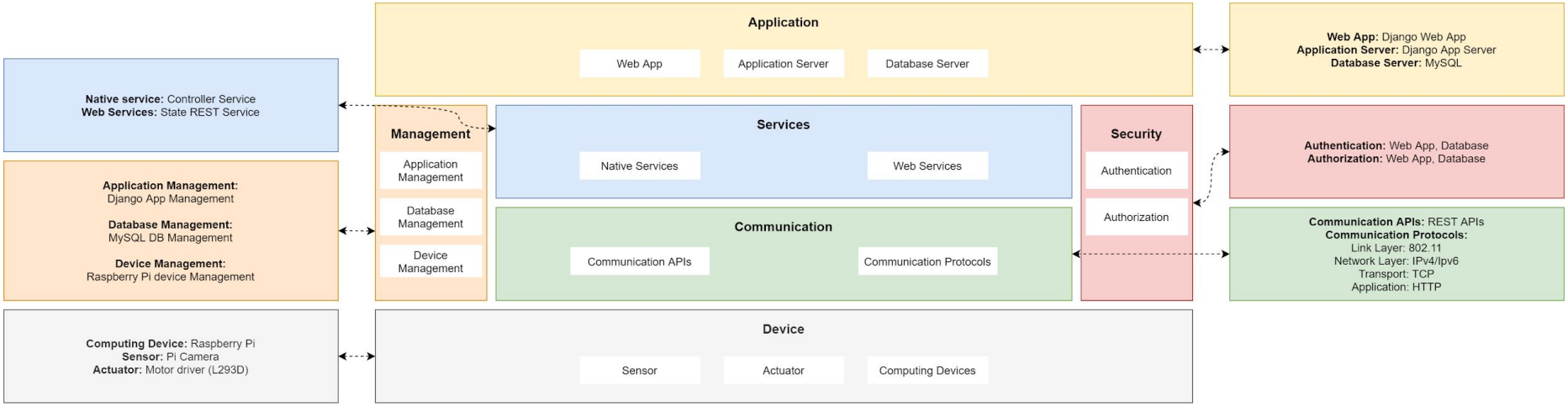
3. Controller service maps to the services FG (native service). Web services map to Services FG (web services)

4. Web services map to services FG (web services)

5. Database maps to the Management FG (database management) and security FG (database security)

6. Application maps to the Application FG (web application, application and database servers), management FG (app management) and security FG (app security)

Step 8: Operational View Specification

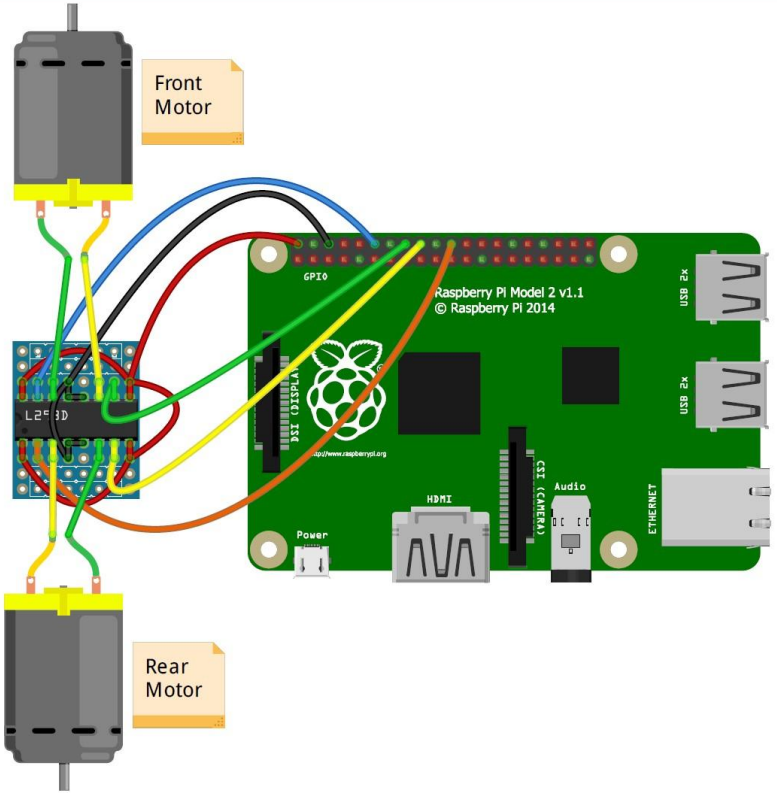


The above diagram shows the intricate details of the workings of the application. We use the Django framework for this project along with a MySQL database which is hosted on the cloud with which our IoT device (RC Car) interacts through HTTP requests and responses over WiFi

Step 9: Device & Component Integration

The components required are as follow:

- Raspberry Pi 3 or higher
- Two 9V motors
- Motor driver (L293D)
- 9V batteries
- RC Car chassis
- Pi Camera



Step 10: Application Development

Car

- On the Car, we need a program that captures the video feed from the Pi Camera, sends every nth frame to an API endpoint hosted on the cloud over HTTP, whose response would be the next state of the car ie, whether to turn left, right, stop or go straight.
- This program would also send the car's steer and throttle state every n seconds to a different API endpoint to enable real-time monitoring.
- This program would run infinitely on the Raspberry Pi until a keyboard exception or other exit statements.
- A path made on the floor that is clearly identifiable by the camera.

Server

- A Django application with a MySQL database, hosted on a cloud service exposing 2 endpoints for the purposes described above.
- The first endpoint would take an image as an input, get a prediction from a trained Convolutional Neural Network, and return the same as the HTTP response.
- The second endpoint would receive the state of the car, which is stored in the MySQL database, which can be used for a real-time dashboard.