# HopTraf: Driving Behavior Analysis Component of Intelligent Transportation System.

Author: CHEN Yi Pu, HAN Jiaming, LIU Qianqi,
SHU Kunxin, YANG Rongtao

HKPU 24 Spring, Service and Cloud Computing

# Checklist

Driving Behavior Analysis Component of Intelligent Transportation System.

Build on Centos Linux 7.9 Pass | Maven Build Pass | Coverage 93% | Release Version V1.0

- Project Name: Hoptraf
- Group 12 in COMP4442

The team has completed all of Project's basic and additional requirements.

- Online Demo: https://aws.hoptraf.hirsun.tech
- Native Link: https://main.d3j623nxuvghcy.amplifyapp.com/
- Github Repository: https://github.com/guomaimang/Hoptraf/
- Testing Report: https://test-report.hoptraf.hirsun.tech/

## Requirements

- [x] A Website for Driving Behavior Analysis
- [x] Real Time
- [x] Generate a summary to show the driving behavior of each driver.
- [x] Monitor the driving speed of each driver in real time.
- [x] Use Amazon Web Services (AWS) to develop the website.
- [x] Analyze the driving behavior with **Spark** (Spark-SQL)
- [x] The cumulative number of times for each driver
- [x] Use a diagram to visualize the driving speed
- [x] When the driver is speeding, a warning will be issued
- [x] Automatically Updated every 30 seconds
- [x] Running of the website: no run-time error
- [x] Effective and user-friendly user interface

## Submission

- [x] Source code in a folder
- [x] Report

## Environment

- Pruduction Env - Operation system: Centos 7.9 (Linux)
- Development Env - Operation system: Macos 14.4.1 (23E224)
- Programming language: Java 17.0.10
- Application Framework: StringBoot 3
- Dependencies/Required packages in Java:

```
lombok
spring-boot-configuration-processor
jedis
spark-core_2.13
spark-sql_2.13
janino
fastjson2
mysql-connector-j
mybatis-spring-boot-starter
druid-spring-boot-starter
```

- Software for Building: Maven
- Other Components in used

```
Mysql
Redis
Nginx
```

## Division of Work

| NAME | STUDENT ID | TASKS | CONTRIBUTION |
|---|---|---|---|
| HAN Jiaming | 20075519d | Program Structure Works | 20% |
| YANG Rongtao | 20075563d | Service Design | 20% |
| SHU Kunxin | 20081657d | Testing | 20% |
| CHEN Yi pu | 19084858d | Workflow Works | 20% |
| LIU Qianqi | 20080796d | AWS Works | 20% |

# Table of Contents

# 1 Introduction

This report provides an in-depth overview of a web service developed to analyze driving behavior data.

The service offers various features, including real-time data analysis and visualization, driver statistics calculation, and data query. It operates on data provided in CSV files, which includes driving behavior records from 10 drivers over 10 consecutive days. The web service was developed using Java with the SpringBoot framework and integrates with Apache Spark for data operations.

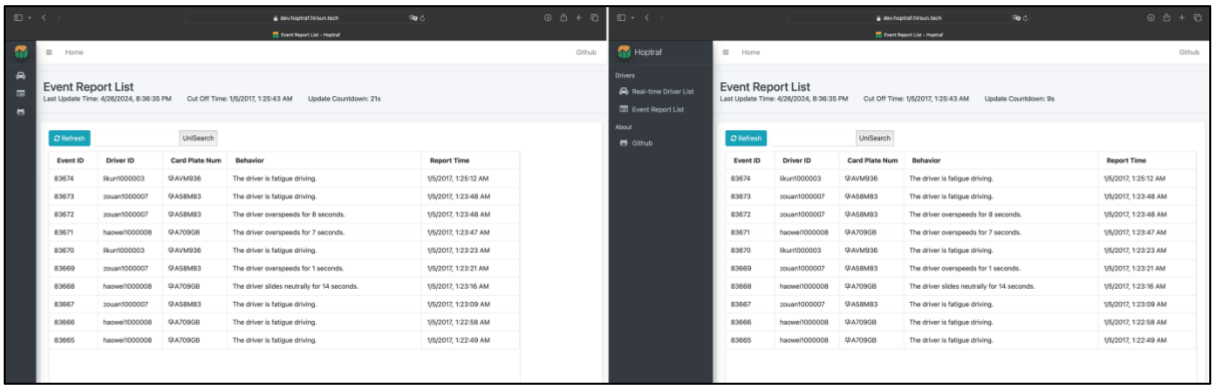
The system design incorporates multi-layer architecture and asynchronous design for enhanced performance. The project delves into implementation details, cloud computing with AWS, evaluation of the service, and testing procedures.

# 2 Features and Functionalities

In order to analyze the driving behavior data, a web service is developed with a series of functionalities including data time simulation, real-time data analysis, real-time driving behavior comprehension, driver statistics calculation, data query, and data visualization.

These functionalities not only provide interactions with the web service to search according to different criteria but also satisfy the two main requirements specified in the documentation: to display summary information during the given time and monitor driving speed in real-time with diagrams and notifications.

The web service can be accessed through here, while users should see the following web page indicating the availability of the service.



Event Report List
Last Update Time: 4/26/2024, 8:36:35 PM  Cut Off Time: 1/5/2017, 1:25:43 AM  Update Countdown: 21s

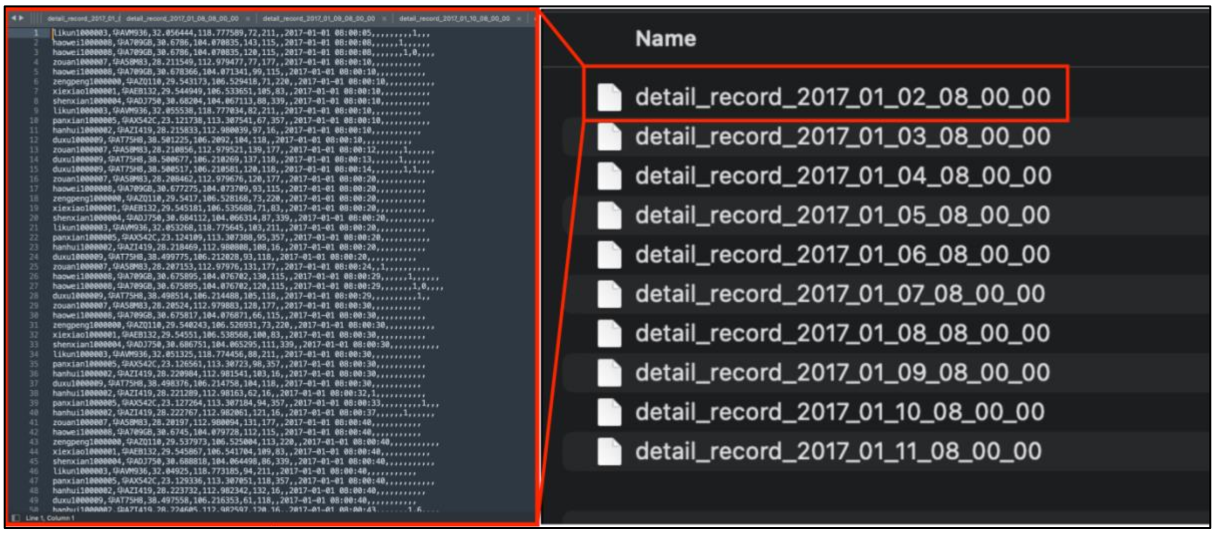
| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 83674 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 1:25:12 AM |
| 83673 | zouan1000007 | 华A5BM83 | The driver is fatigue driving. | 1/5/2017, 1:23:48 AM |
| 83672 | zouan1000007 | 华A5BM83 | The driver overspeeds for 8 seconds. | 1/5/2017, 1:23:48 AM |
| 83671 | haowei1000008 | 华A709GB | The driver overspeeds for 7 seconds. | 1/5/2017, 1:23:47 AM |
| 83670 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 1:23:23 AM |
| 83669 | zouan1000007 | 华A5BM83 | The driver overspeeds for 1 seconds. | 1/5/2017, 1:23:21 AM |
| 83668 | haowei1000008 | 华A709GB | The driver slides neutrally for 14 seconds. | 1/5/2017, 1:23:16 AM |
| 83667 | zouan1000007 | 华A5BM83 | The driver is fatigue driving. | 1/5/2017, 1:23:09 AM |
| 83666 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 1:22:58 AM |
| 83665 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 1:22:49 AM |

Event Report List
Last Update Time: 4/26/2024, 8:36:35 PM  Cut Off Time: 1/5/2017, 1:25:43 AM  Update Countdown: 9s

| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 83674 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 1:25:12 AM |
| 83673 | zouan1000007 | 华A5BM83 | The driver is fatigue driving. | 1/5/2017, 1:23:48 AM |
| 83672 | zouan1000007 | 华A5BM83 | The driver overspeeds for 8 seconds. | 1/5/2017, 1:23:48 AM |
| 83671 | haowei1000008 | 华A709GB | The driver overspeeds for 7 seconds. | 1/5/2017, 1:23:47 AM |
| 83670 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 1:23:23 AM |
| 83669 | zouan1000007 | 华A5BM83 | The driver overspeeds for 1 seconds. | 1/5/2017, 1:23:21 AM |
| 83668 | haowei1000008 | 华A709GB | The driver slides neutrally for 14 seconds. | 1/5/2017, 1:23:16 AM |
| 83667 | zouan1000007 | 华A5BM83 | The driver is fatigue driving. | 1/5/2017, 1:23:09 AM |
| 83666 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 1:22:58 AM |
| 83665 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 1:22:49 AM |

## 2.1 Data

The data of this project was provided in advance containing 10 files of 10 drivers' driving behavior over 10 consecutive days. These data are stored in CSV files without headers with a total of 413,450 records. Each record consists of 19 features (columns) vary from driver information to specific driving behavior indications.



detail_record_2017_01_(
detail_record_2017_01_08_08_00_00
detail_record_2017_01_09_08_00_00
detail_record_2017_01_10_08_00_00
Line 1, Column 1
Name

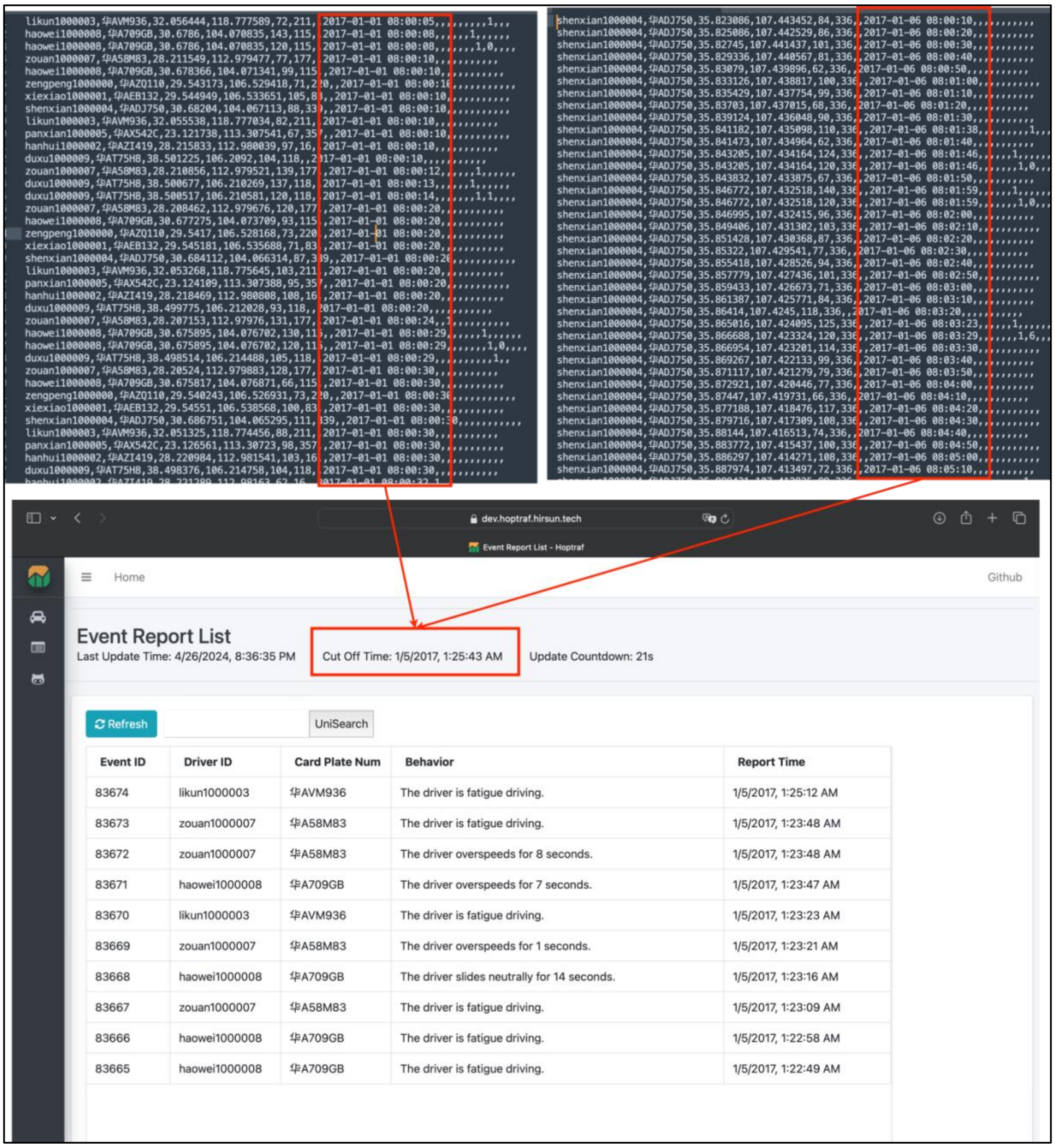
```
likun1000003,华AVM936,32.056444,118.777589,72,211,,2017-01-01 08:00:05,,,,,,,,,1,,,
haowei1000008,华A709GB,30.6786,104.070835,143,115,,2017-01-01 08:00:08,,,,,,1,,,,,,
haowei1000008,华A709GB,30.6786,104.070835,120,115,,2017-01-01 08:00:08,,,,,,,,1,0,,,,
zouan1000007,华A58M83,28.211549,112.979477,77,177,,2017-01-01 08:00:10,,,,,,,,,,,
haowei1000008,华A709GB,30.678366,104.071341,99,115,,2017-01-01 08:00:10,,,,,,,,,,,
zengpeng1000000,华AZQ110,29.543173,106.529418,71,220,,2017-01-01 08:00:10,,,,,,,,,,,
xiexiao1000001,华AEB132,29.544949,106.533651,105,83,,2017-01-01 08:00:10,,,,,,,,,,,
shenxian1000004,华ADJ750,30.68204,104.067113,88,339,,2017-01-01 08:00:10,,,,,,,,,,,
likun1000003,华AVM936,32.055538,118.777034,82,211,,2017-01-01 08:00:10,,,,,,,,,,,
panxian1000005,华AX542C,23.121738,113.307541,67,357,,2017-01-01 08:00:10,,,,,,,,,,,
hanhui1000002,华AZI419,28.215833,112.980039,97,16,,2017-01-01 08:00:10,,,,,,,,,,,
duxu1000009,华AT75H8,38.501225,106.2092,104,118,,2017-01-01 08:00:10,,,,,,,,,,,
zouan1000007,华A58M83,28.210856,112.979521,139,177,,2017-01-01 08:00:12,,,,,,1,,,,,,
duxu1000009,华AT75H8,38.500677,106.210269,137,118,,2017-01-01 08:00:13,,,,,,1,,,,,,
duxu1000009,华AT75H8,38.500517,106.210581,120,118,,2017-01-01 08:00:14,,,,,,,,1,1,,,,
zouan1000007,华A58M83,28.208462,112.979676,120,177,,2017-01-01 08:00:20,,,,,,,,,,,
haowei1000008,华A709GB,30.677275,104.073709,93,115,,2017-01-01 08:00:20,,,,,,,,,,,
zengpeng1000000,华AZQ110,29.5417,106.528168,73,220,,2017-01-01 08:00:20,,,,,,,,,,,
xiexiao1000001,华AEB132,29.545181,106.535688,71,83,,2017-01-01 08:00:20,,,,,,,,,,,
shenxian1000004,华ADJ750,30.684112,104.066314,87,339,,2017-01-01 08:00:20,,,,,,,,,,,
likun1000003,华AVM936,32.053268,118.775645,103,211,,2017-01-01 08:00:20,,,,,,,,,,,
panxian1000005,华AX542C,23.124109,113.307388,95,357,,2017-01-01 08:00:20,,,,,,,,,,,
hanhui1000002,华AZI419,28.218469,112.980808,108,16,,2017-01-01 08:00:20,,,,,,,,,,,
duxu1000009,华AT75H8,38.499775,106.212028,93,118,,2017-01-01 08:00:20,,,,,,,,,,,
zouan1000007,华A58M83,28.207153,112.97976,131,177,,2017-01-01 08:00:24,,1,,,,,,,,,,
haowei1000008,华A709GB,30.675895,104.076702,130,115,,2017-01-01 08:00:29,,,,,,1,,,,,,
haowei1000008,华A709GB,30.675895,104.076702,120,115,,2017-01-01 08:00:29,,,,,,,,1,0,,,,
duxu1000009,华AT75H8,38.498514,106.214488,105,118,,2017-01-01 08:00:29,,,,,,,,,,1,,
zouan1000007,华A58M83,28.20524,112.979883,128,177,,2017-01-01 08:00:30,,,,,,,,,,,
haowei1000008,华A709GB,30.675817,104.076871,66,115,,2017-01-01 08:00:30,,,,,,,,,,,
zengpeng1000000,华AZQ110,29.540243,106.526931,73,220,,2017-01-01 08:00:30,,,,,,,,,,,
xiexiao1000001,华AEB132,29.54551,106.538568,100,83,,2017-01-01 08:00:30,,,,,,,,,,,
shenxian1000004,华ADJ750,30.686751,104.065295,111,339,,2017-01-01 08:00:30,,,,,,,,,,,
likun1000003,华AVM936,32.051325,118.774456,88,211,,2017-01-01 08:00:30,,,,,,,,,,,
panxian1000005,华AX542C,23.126561,113.30723,98,357,,2017-01-01 08:00:30,,,,,,,,,,,
hanhui1000002,华AZI419,28.220984,112.981541,103,16,,2017-01-01 08:00:30,,,,,,,,,,,
duxu1000009,华AT75H8,38.498376,106.214758,104,118,,2017-01-01 08:00:30,,,,,,,,,,,
hanhui1000002,华AZI419,28.221289,112.98163,62,16,,2017-01-01 08:00:32,1,,,,,,,,,,,
panxian1000005,华AX542C,23.127264,113.307184,94,357,,2017-01-01 08:00:33,,,,,,,,,,1,,,
hanhui1000002,华AZI419,28.222767,112.982061,121,16,,2017-01-01 08:00:37,,,,,,1,,,,,,
zouan1000007,华A58M83,28.20197,112.980094,131,177,,2017-01-01 08:00:40,,,,,,,,,,,
haowei1000008,华A709GB,30.6745,104.079728,112,115,,2017-01-01 08:00:40,,,,,,,,,,,
zengpeng1000000,华AZQ110,29.537973,106.525004,113,220,,2017-01-01 08:00:40,,,,,,,,,,,
xiexiao1000001,华AEB132,29.545867,106.541704,109,83,,2017-01-01 08:00:40,,,,,,,,,,,
shenxian1000004,华ADJ750,30.688818,104.064498,86,339,,2017-01-01 08:00:40,,,,,,,,,,,
likun1000003,华AVM936,32.04925,118.773185,94,211,,2017-01-01 08:00:40,,,,,,,,,,,
panxian1000005,华AX542C,23.129336,113.307051,118,357,,2017-01-01 08:00:40,,,,,,,,,,,
hanhui1000002,华AZI419,28.223732,112.982342,132,16,,2017-01-01 08:00:40,,,,,,,,,,,
duxu1000009,华AT75H8,38.497558,106.216353,61,118,,2017-01-01 08:00:40,,,,,,,,,,,
hanhui1000002,华AZI419,28.224605,112.982597,120,16,,2017-01-01 08:00:43,,,,,,,1,6,,,,
```

detail_record_2017_01_02_08_00_00
detail_record_2017_01_03_08_00_00
detail_record_2017_01_04_08_00_00
detail_record_2017_01_05_08_00_00
detail_record_2017_01_06_08_00_00
detail_record_2017_01_07_08_00_00
detail_record_2017_01_08_08_00_00
detail_record_2017_01_09_08_00_00
detail_record_2017_01_10_08_00_00
detail_record_2017_01_11_08_00_00

## 2.2 Features

### 2.2.1 Data Time Simulation

All the data are recorded with timestamps lying between 2027-01-01 to 2017-01-10. In order to perform real-time operations, a specific design is implemented that the web service simulates the actual time when the data is recorded. As shown in the figure provided below, whenever the service is initialized, the internal time will be set to 2017-01-01 and continues to flow normally. Such design allows the service to load data according to the timestamps and perform truly real-time analysis and statistical calculations.



```
likun1000003,华AVM936,32.056444,118.777589,72,211,,2017-01-01 08:00:05,,,,,,,,,1,,,
haowei1000008,华A709GB,30.6786,104.070835,143,115,,2017-01-01 08:00:08,,,,,,1,,,,,,
haowei1000008,华A709GB,30.6786,104.070835,120,115,,2017-01-01 08:00:08,,,,,,,1,0,,,,
zouan1000007,华A58M83,28.211549,112.979477,77,177,,2017-01-01 08:00:10,,,,,,,,,,
haowei1000008,华A709GB,30.678366,104.071341,99,115,,2017-01-01 08:00:10,,,,,,,,,,
zengpeng1000000,华AZQ110,29.543173,106.529418,71,220,,2017-01-01 08:00:10,,,,,,,,,,,
xiexiao1000001,华AEB132,29.544949,106.533651,105,83,,2017-01-01 08:00:10,,,,,,,,,,
shenxian1000004,华ADJ750,30.68204,104.067113,88,339,,2017-01-01 08:00:10,,,,,,,,,,
likun1000003,华AVM936,32.055538,118.777034,82,211,,2017-01-01 08:00:10,,,,,,,,,,
panxian1000005,华AX542C,23.121738,113.307541,67,357,,2017-01-01 08:00:10,,,,,,,,,,
hanhui1000002,华AZI419,28.215833,112.980039,97,16,,2017-01-01 08:00:10,,,,,,,,,,
duxu1000009,华AT75H8,38.501225,106.2092,104,118,,2017-01-01 08:00:10,,,,,,,,,,
zouan1000007,华A58M83,28.210856,112.979521,139,177,,2017-01-01 08:00:12,,,,,,1,,,,,,
duxu1000009,华AT75H8,38.500677,106.210269,137,118,,2017-01-01 08:00:13,,,,,,1,,,,,,
duxu1000009,华AT75H8,38.500517,106.210581,120,118,,2017-01-01 08:00:14,,,,,,,1,1,,,,
zouan1000007,华A58M83,28.208462,112.979676,120,177,,2017-01-01 08:00:20,,,,,,,,,,
haowei1000008,华A709GB,30.677275,104.073709,93,115,,2017-01-01 08:00:20,,,,,,,,,,
zengpeng1000000,华AZQ110,29.5417,106.528168,73,220,,2017-01-01 08:00:20,,,,,,,,,,
xiexiao1000001,华AEB132,29.545181,106.535688,71,83,,2017-01-01 08:00:20,,,,,,,,,,
shenxian1000004,华ADJ750,30.684112,104.066314,87,339,,2017-01-01 08:00:20,,,,,,,,,,,
likun1000003,华AVM936,32.053268,118.775645,103,211,,2017-01-01 08:00:20,,,,,,,,,,
panxian1000005,华AX542C,23.124109,113.307388,95,357,,2017-01-01 08:00:20,,,,,,,,,,
hanhui1000002,华AZI419,28.218469,112.980808,108,16,,2017-01-01 08:00:20,,,,,,,,,,
duxu1000009,华AT75H8,38.499775,106.212028,93,118,,2017-01-01 08:00:20,,,,,,,,,,
zouan1000007,华A58M83,28.207153,112.97976,131,177,,2017-01-01 08:00:24,,1,,,,,,,,,
haowei1000008,华A709GB,30.675895,104.076702,130,115,,2017-01-01 08:00:29,,,,,,1,,,,,,
haowei1000008,华A709GB,30.675895,104.076702,120,115,,2017-01-01 08:00:29,,,,,,,1,0,,,,
duxu1000009,华AT75H8,38.498514,106.214488,105,118,,2017-01-01 08:00:29,,,,,,,,,1,,
zouan1000007,华A58M83,28.20524,112.979883,128,177,,2017-01-01 08:00:30,,,,,,,,,,
haowei1000008,华A709GB,30.675817,104.076871,66,115,,2017-01-01 08:00:30,,,,,,,,,,
zengpeng1000000,华AZQ110,29.540243,106.526931,73,220,,2017-01-01 08:00:30,,,,,,,,,,,
xiexiao1000001,华AEB132,29.54551,106.538568,100,83,,2017-01-01 08:00:30,,,,,,,,,,
shenxian1000004,华ADJ750,30.686751,104.065295,111,339,,2017-01-01 08:00:30,,,,,,,,,,,
likun1000003,华AVM936,32.051325,118.774456,88,211,,2017-01-01 08:00:30,,,,,,,,,,
panxian1000005,华AX542C,23.126561,113.30723,98,357,,2017-01-01 08:00:30,,,,,,,,,,
hanhui1000002,华AZI419,28.220984,112.981541,103,16,,2017-01-01 08:00:30,,,,,,,,,,
duxu1000009,华AT75H8,38.498376,106.214758,104,118,,2017-01-01 08:00:30,,,,,,,,,,
```

```
shenxian1000004,华ADJ750,35.823086,107.443452,84,336,,2017-01-06 08:00:10,,,,,,,,,,
shenxian1000004,华ADJ750,35.825086,107.442529,86,336,,2017-01-06 08:00:20,,,,,,,,,,
shenxian1000004,华ADJ750,35.82745,107.441437,101,336,,2017-01-06 08:00:30,,,,,,,,,,
shenxian1000004,华ADJ750,35.829336,107.440567,81,336,,2017-01-06 08:00:40,,,,,,,,,,
shenxian1000004,华ADJ750,35.83079,107.439896,62,336,,2017-01-06 08:00:50,,,,,,,,,,
shenxian1000004,华ADJ750,35.833126,107.438817,100,336,,2017-01-06 08:01:00,,,,,,,,,,,
shenxian1000004,华ADJ750,35.835429,107.437754,99,336,,2017-01-06 08:01:10,,,,,,,,,,
shenxian1000004,华ADJ750,35.83703,107.437015,68,336,,2017-01-06 08:01:20,,,,,,,,,,
shenxian1000004,华ADJ750,35.839124,107.436048,90,336,,2017-01-06 08:01:30,,,,,,,,,,
shenxian1000004,华ADJ750,35.841182,107.435098,110,336,,2017-01-06 08:01:38,,,,,,,,,1,,,
shenxian1000004,华ADJ750,35.841473,107.434964,62,336,,2017-01-06 08:01:40,,,,,,,,,,
shenxian1000004,华ADJ750,35.843205,107.434164,124,336,,2017-01-06 08:01:46,,,,,,1,,,,,,
shenxian1000004,华ADJ750,35.843205,107.434164,120,336,,2017-01-06 08:01:46,,,,,,,1,0,,,
shenxian1000004,华ADJ750,35.843832,107.433875,67,336,,2017-01-06 08:01:50,,,,,,,,,,
shenxian1000004,华ADJ750,35.846772,107.432518,140,336,,2017-01-06 08:01:59,,,,,,1,,,,,,
shenxian1000004,华ADJ750,35.846772,107.432518,120,336,,2017-01-06 08:01:59,,,,,,,1,0,,,
shenxian1000004,华ADJ750,35.846995,107.432415,96,336,,2017-01-06 08:02:00,,,,,,,,,,
shenxian1000004,华ADJ750,35.849406,107.431302,103,336,,2017-01-06 08:02:10,,,,,,,,,,,
shenxian1000004,华ADJ750,35.851428,107.430368,87,336,,2017-01-06 08:02:20,,,,,,,,,,
shenxian1000004,华ADJ750,35.85322,107.429541,77,336,,2017-01-06 08:02:30,,,,,,,,,,
shenxian1000004,华ADJ750,35.855418,107.428526,94,336,,2017-01-06 08:02:40,,,,,,,,,,
shenxian1000004,华ADJ750,35.857779,107.427436,101,336,,2017-01-06 08:02:50,,,,,,,,,,,
shenxian1000004,华ADJ750,35.859433,107.426673,71,336,,2017-01-06 08:03:00,,,,,,,,,,
shenxian1000004,华ADJ750,35.861387,107.425771,84,336,,2017-01-06 08:03:10,,,,,,,,,,
shenxian1000004,华ADJ750,35.86414,107.4245,118,336,,2017-01-06 08:03:20,,,,,,,,,,
shenxian1000004,华ADJ750,35.865016,107.424095,125,336,,2017-01-06 08:03:23,,,,,,1,,,,,,
shenxian1000004,华ADJ750,35.866688,107.423324,120,336,,2017-01-06 08:03:29,,,,,,,1,6,,,
shenxian1000004,华ADJ750,35.866954,107.423201,114,336,,2017-01-06 08:03:30,,,,,,,,,,,
shenxian1000004,华ADJ750,35.869267,107.422133,99,336,,2017-01-06 08:03:40,,,,,,,,,,
shenxian1000004,华ADJ750,35.871117,107.421279,79,336,,2017-01-06 08:03:50,,,,,,,,,,
shenxian1000004,华ADJ750,35.872921,107.420446,77,336,,2017-01-06 08:04:00,,,,,,,,,,
shenxian1000004,华ADJ750,35.87447,107.419731,66,336,,2017-01-06 08:04:10,,,,,,,,,,
shenxian1000004,华ADJ750,35.877188,107.418476,117,336,,2017-01-06 08:04:20,,,,,,,,,,,
shenxian1000004,华ADJ750,35.879716,107.417309,108,336,,2017-01-06 08:04:30,,,,,,,,,,,
shenxian1000004,华ADJ750,35.88144,107.416513,74,336,,2017-01-06 08:04:40,,,,,,,,,,
shenxian1000004,华ADJ750,35.883772,107.415437,100,336,,2017-01-06 08:04:50,,,,,,,,,,,
shenxian1000004,华ADJ750,35.886297,107.414271,108,336,,2017-01-06 08:05:00,,,,,,,,,,,
shenxian1000004,华ADJ750,35.887974,107.413497,72,336,,2017-01-06 08:05:10,,,,,,,,,,
```

Event Report List

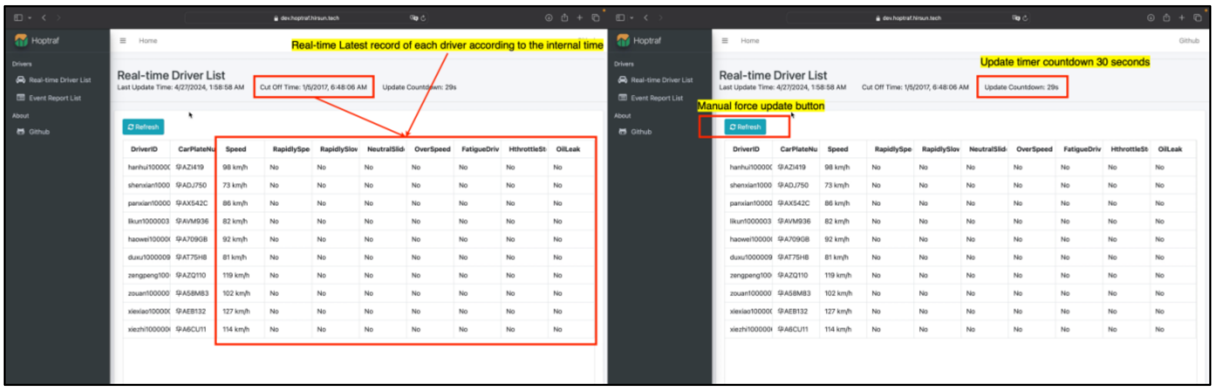
Last Update Time: 4/26/2024, 8:36:35 PM  Cut Off Time: 1/5/2017, 1:25:43 AM  Update Countdown: 21s

| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 83674 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 1:25:12 AM |
| 83673 | zouan1000007 | 华A58M83 | The driver is fatigue driving. | 1/5/2017, 1:23:48 AM |
| 83672 | zouan1000007 | 华A58M83 | The driver overspeeds for 8 seconds. | 1/5/2017, 1:23:48 AM |
| 83671 | haowei1000008 | 华A709GB | The driver overspeeds for 7 seconds. | 1/5/2017, 1:23:47 AM |
| 83670 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 1:23:23 AM |
| 83669 | zouan1000007 | 华A58M83 | The driver overspeeds for 1 seconds. | 1/5/2017, 1:23:21 AM |
| 83668 | haowei1000008 | 华A709GB | The driver slides neutrally for 14 seconds. | 1/5/2017, 1:23:16 AM |
| 83667 | zouan1000007 | 华A58M83 | The driver is fatigue driving. | 1/5/2017, 1:23:09 AM |
| 83666 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 1:22:58 AM |
| 83665 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 1:22:49 AM |

### 2.2.2 Real-time Monitoring

The data monitoring feature performs real-time driving behavior comprehension and driver statistics calculation, and can be categorized into three different aspects, drivers-based analysis, time-based analysis, and driver information analysis.

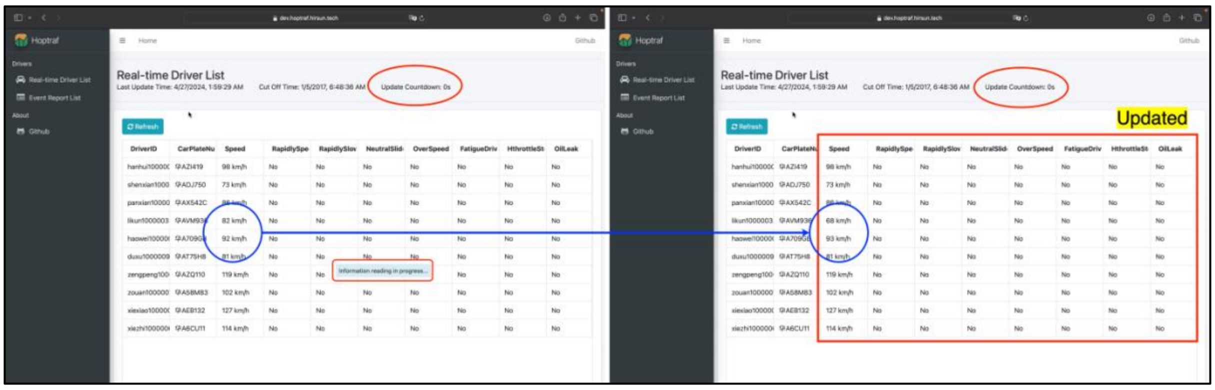
By navigating to the Real-time Driver List page, a driving behavior summary of all the 10 drivers will be displayed in a table. This drivers-based analysis satisfies the first requirement of this project. The table contains not only driver ID and car plates but also latest driving speed, and whether that particular driver is undergoing rapidly speeding up, rapidly slowing down, neutral sliding, over speeding, fatigue driving, throttle stopping, or oil leaking. These data can be automatically updated every 30 seconds, as indicated explicitly by a countdown timer, or manually updated by toggling the Refresh button that prompts and force the web service to read in the next data.


dev.hoptraf.hirsun.tech
Hoptraf
Drivers
Real-time Driver List
Event Report List
About
Github
Home
Refresh
Real-time Latest record of each driver according to the internal time
Update timer countdown 30 seconds
Manual force update button

Real-time Driver List
Last Update Time: 4/27/2024, 1:58:58 AM Cut Off Time: 1/5/2017, 6:48:06 AM Update Countdown: 29s

| DriverID | CarPlateNu | Speed | RapidlySpe | RapidlySlov | NeutralSlid | OverSpeed | FatigueDriv | HthrottleSt | OilLeak |
|---|---|---|---|---|---|---|---|---|---|
| hanhui10000C | 苏AZI419 | 98 km/h | No | No | No | No | No | No | No |
| shenxian1000 | 苏ADJ750 | 73 km/h | No | No | No | No | No | No | No |
| panxian10000 | 苏AX542C | 86 km/h | No | No | No | No | No | No | No |
| likun1000003 | 苏AVM936 | 82 km/h | No | No | No | No | No | No | No |
| haowei10000( | 苏A709GB | 92 km/h | No | No | No | No | No | No | No |
| duxu1000009 | 苏AT75H8 | 81 km/h | No | No | No | No | No | No | No |
| zengpeng100 | 苏AZQ110 | 119 km/h | No | No | No | No | No | No | No |
| zouan100000 | 苏A58M83 | 102 km/h | No | No | No | No | No | No | No |
| xiexiao10000C | 苏AEB132 | 127 km/h | No | No | No | No | No | No | No |
| xiezhi100000 | 苏A6CU11 | 114 km/h | No | No | No | No | No | No | No |

Apart from drivers-based analysis, more sophisticated services are provided for detailed understandings of the data and finer interactions with the service from the user perspective. The Event Report List page performs time-based analysis. In other words, instead of analyzing the latest driving behavior of all 10 drivers at the same time, the web service reads in the latest data according to the timestamp and comprehends the corresponding record.



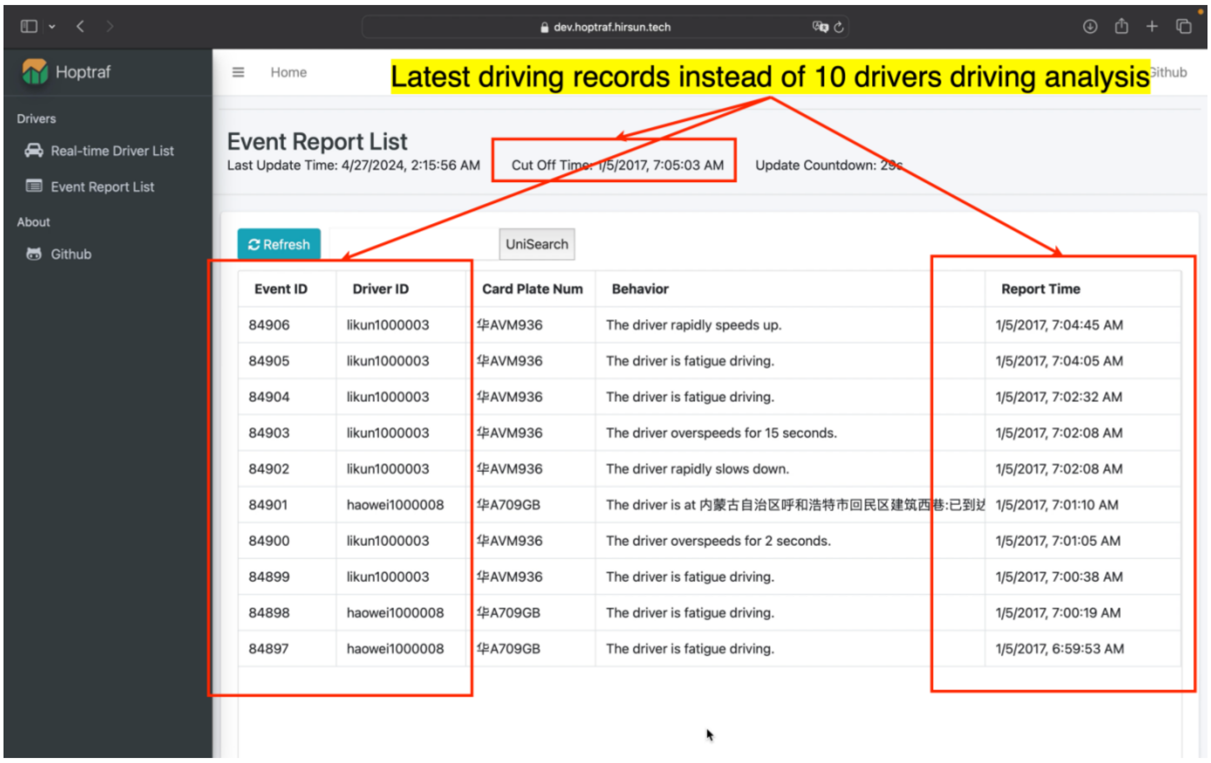
Real-time Driver List
Last Update Time: 4/27/2024, 1:59:29 AM Cut Off Time: 1/5/2017, 6:48:36 AM Update Countdown: 0s

| DriverID | CarPlateNu | Speed | RapidlySpe | RapidlySlov | NeutralSlid | OverSpeed | FatigueDriv | HthrottleSt | OilLeak |
|---|---|---|---|---|---|---|---|---|---|
| hanhui10000C | 苏AZI419 | 98 km/h | No | No | No | No | No | No | No |
| shenxian1000 | 苏ADJ750 | 73 km/h | No | No | No | No | No | No | No |
| panxian10000 | 苏AX542C | 86 km/h | No | No | No | No | No | No | No |
| likun1000003 | 苏AVM936 | 82 km/h | No | No | No | No | No | No | No |
| haowei10000( | 苏A709GB | 92 km/h | No | No | No | No | No | No | No |
| duxu1000009 | 苏AT75H8 | 81 km/h | No | No | No | No | No | No | No |
| zengpeng100 | 苏AZQ110 | 119 km/h | No | No | No | No | No | No | No |
| zouan100000 | 苏A58M83 | 102 km/h | No | No | No | No | No | No | No |
| xiexiao10000C | 苏AEB132 | 127 km/h | No | No | No | No | No | No | No |
| xiezhi100000 | 苏A6CU11 | 114 km/h | No | No | No | No | No | No | No |

Real-time Driver List
Last Update Time: 4/27/2024, 1:59:29 AM Cut Off Time: 1/5/2017, 6:48:36 AM Update Countdown: 0s

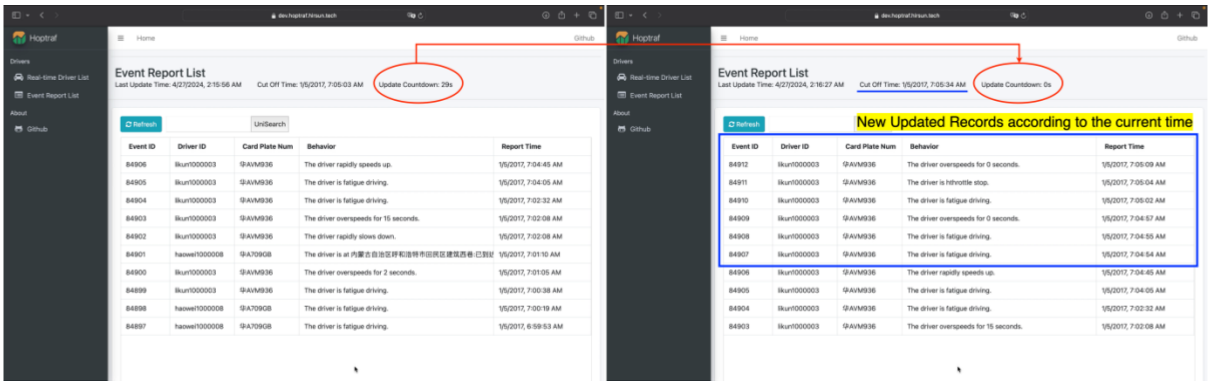
| DriverID | CarPlateNu | Speed | RapidlySpe | RapidlySlov | NeutralSlid | OverSpeed | FatigueDriv | HthrottleSt | OilLeak |
|---|---|---|---|---|---|---|---|---|---|
| hanhui10000C | 苏AZI419 | 98 km/h | No | No | No | No | No | No | No |
| shenxian1000 | 苏ADJ750 | 73 km/h | No | No | No | No | No | No | No |
| panxian10000 | 苏AX542C | 86 km/h | No | No | No | No | No | No | No |
| likun1000003 | 苏AVM936 | 68 km/h | No | No | No | No | No | No | No |
| haowei10000( | 苏A709GB | 93 km/h | No | No | No | No | No | No | No |
| duxu1000009 | 苏AT75H8 | 81 km/h | No | No | No | No | No | No | No |
| zengpeng100 | 苏AZQ110 | 119 km/h | No | No | No | No | No | No | No |
| zouan100000 | 苏A58M83 | 102 km/h | No | No | No | No | No | No | No |
| xiexiao10000C | 苏AEB132 | 127 km/h | No | No | No | No | No | No | No |
| xiezhi100000 | 苏A6CU11 | 114 km/h | No | No | No | No | No | No | No |

As illustrated in the figure below, the service displays the information of the driver and provides a short but detailed description of the latest behavior even with time duration. This service also allows automatic update every 30 seconds or manual update through pressing the button.



Event Report List
Last Update Time: 4/27/2024, 2:15:56 AM Cut Off Time: 1/5/2017, 7:05:03 AM Update Countdown: 29s

| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 84906 | likun1000003 | 华AVM936 | The driver rapidly speeds up. | 1/5/2017, 7:04:45 AM |
| 84905 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:04:05 AM |
| 84904 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:02:32 AM |
| 84903 | likun1000003 | 华AVM936 | The driver overspeeds for 15 seconds. | 1/5/2017, 7:02:08 AM |
| 84902 | likun1000003 | 华AVM936 | The driver rapidly slows down. | 1/5/2017, 7:02:08 AM |
| 84901 | haowei1000008 | 华A709GB | The driver is at 内蒙古自治区呼和浩特市回民区建筑西巷:已到达 | 1/5/2017, 7:01:10 AM |
| 84900 | likun1000003 | 华AVM936 | The driver overspeeds for 2 seconds. | 1/5/2017, 7:01:05 AM |
| 84899 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:00:38 AM |
| 84898 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 7:00:19 AM |
| 84897 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 6:59:53 AM |



Event Report List
Last Update Time: 4/27/2024, 2:15:56 AM Cut Off Time: 1/5/2017, 7:05:03 AM Update Countdown: 29s

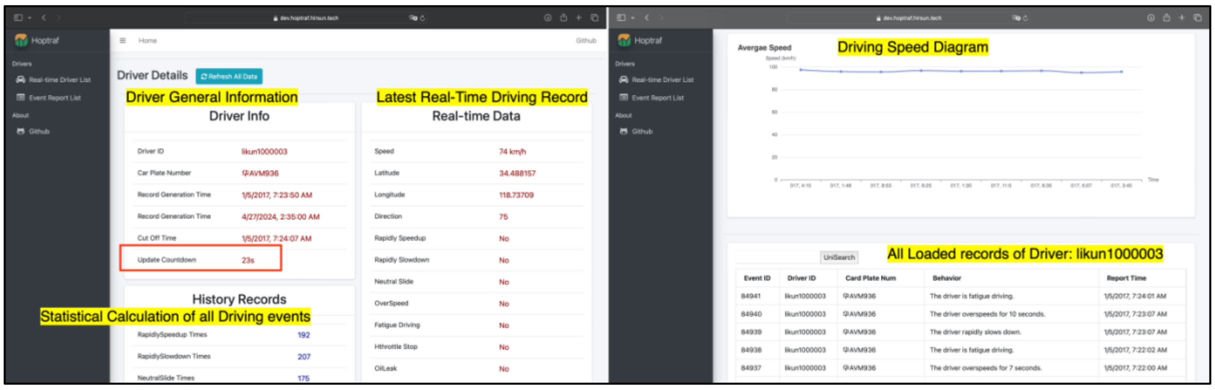
| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 84906 | likun1000003 | 华AVM936 | The driver rapidly speeds up. | 1/5/2017, 7:04:45 AM |
| 84905 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:04:05 AM |
| 84904 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:02:32 AM |
| 84903 | likun1000003 | 华AVM936 | The driver overspeeds for 15 seconds. | 1/5/2017, 7:02:08 AM |
| 84902 | likun1000003 | 华AVM936 | The driver rapidly slows down. | 1/5/2017, 7:02:08 AM |
| 84901 | haowei1000008 | 华A709GB | The driver is at 内蒙古自治区呼和浩特市回民区建筑西巷:已到达 | 1/5/2017, 7:01:10 AM |
| 84900 | likun1000003 | 华AVM936 | The driver overspeeds for 2 seconds. | 1/5/2017, 7:01:05 AM |
| 84899 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:00:38 AM |
| 84898 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 7:00:19 AM |
| 84897 | haowei1000008 | 华A709GB | The driver is fatigue driving. | 1/5/2017, 6:59:53 AM |

Event Report List
Last Update Time: 4/27/2024, 2:16:27 AM Cut Off Time: 1/5/2017, 7:05:34 AM Update Countdown: 0s

| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 84912 | likun1000003 | 华AVM936 | The driver overspeeds for 0 seconds. | 1/5/2017, 7:05:09 AM |
| 84911 | likun1000003 | 华AVM936 | The driver is hthrottle stop. | 1/5/2017, 7:05:04 AM |
| 84910 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:05:02 AM |
| 84909 | likun1000003 | 华AVM936 | The driver overspeeds for 0 seconds. | 1/5/2017, 7:04:57 AM |
| 84908 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:04:55 AM |
| 84907 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:04:54 AM |
| 84906 | likun1000003 | 华AVM936 | The driver rapidly speeds up. | 1/5/2017, 7:04:45 AM |
| 84905 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:04:05 AM |
| 84904 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/5/2017, 7:02:32 AM |
| 84903 | likun1000003 | 华AVM936 | The driver overspeeds for 15 seconds. | 1/5/2017, 7:02:08 AM |

By clicking on any driving records, regardless in the Real-time Driver List page or the Event Report List page, users will navigate to the driver information analysis service. This service can be seen as an integration of the drivers-based analysis and time-based analysis but focuses on one particular driver only. To be more specific, it contains all forms of analysis results in the previous two services on one driver.

Different from the previous analysis, this service contains additional details such as the statistic calculations of all driving events, diagram of the driving speed, and a notification box that pops up when the driver is speeding in real-time. The notification box serves as a real-time monitoring warning message if the latest data read in indicates the driver is over speeding. Along with the speed diagram, the two functionalities satisfy the second requirements of this project. Similarly, this service also updates every 30 seconds with a button that can force update the application.



Driver Info

| | |
|---|---|
| Driver ID | likun1000003 |
| Car Plate Number | 苏AVM936 |
| Record Generation Time | 1/5/2017, 7:23:50 AM |
| Record Generation Time | 4/27/2024, 2:35:00 AM |
| Cut Off Time | 1/5/2017, 7:24:07 AM |
| Update Countdown | 23s |

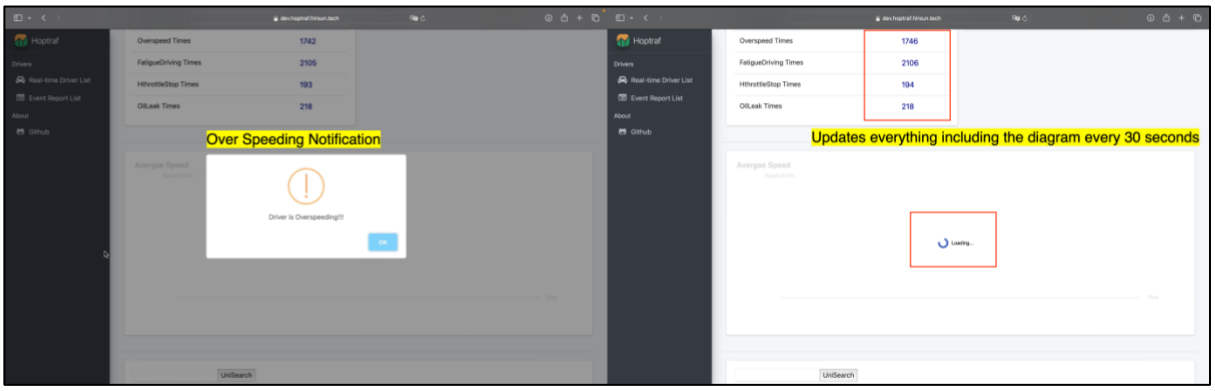
History Records

| | |
|---|---|
| RapidlySpeedup Times | 192 |
| RapidlySlowdown Times | 207 |
| NeutralSlide Times | 175 |

Real-time Data

| | |
|---|---|
| Speed | 74 km/h |
| Latitude | 34.488157 |
| Longitude | 118.73709 |
| Direction | 75 |
| Rapidly Speedup | No |
| Rapidly Slowdown | No |
| Neutral Slide | No |
| OverSpeed | No |
| Fatigue Driving | No |
| Hthrottle Stop | No |
| OilLeak | No |

| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 84941 | likun1000003 | 苏AVM936 | The driver is fatigue driving. | 1/5/2017, 7:24:01 AM |
| 84940 | likun1000003 | 苏AVM936 | The driver overspeeds for 10 seconds. | 1/5/2017, 7:23:07 AM |
| 84939 | likun1000003 | 苏AVM936 | The driver rapidly slows down. | 1/5/2017, 7:23:07 AM |
| 84938 | likun1000003 | 苏AVM936 | The driver is fatigue driving. | 1/5/2017, 7:22:02 AM |
| 84937 | likun1000003 | 苏AVM936 | The driver overspeeds for 7 seconds. | 1/5/2017, 7:22:00 AM |

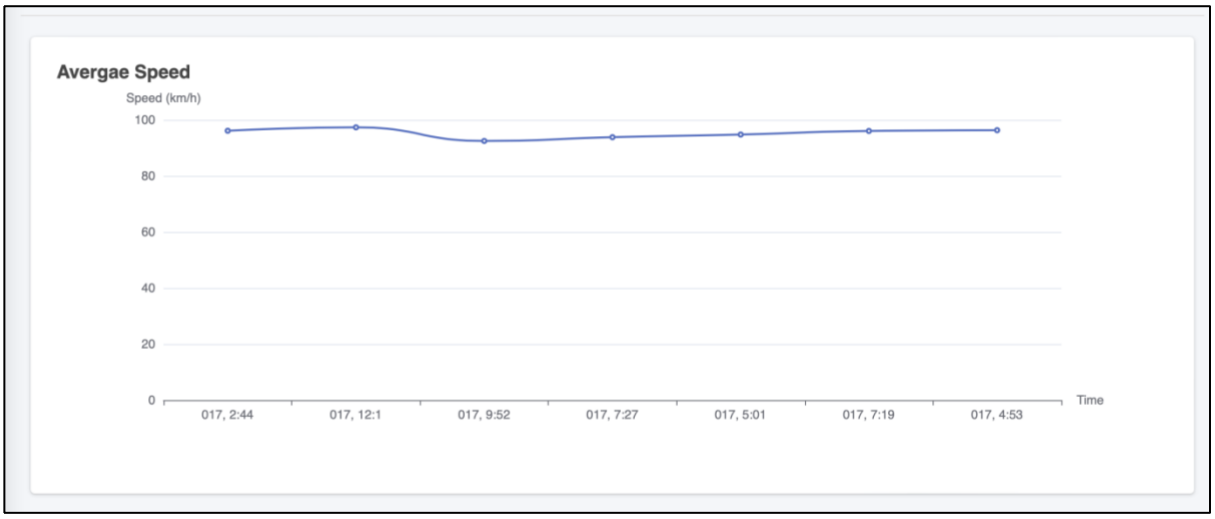


| | |
|---|---|
| Overspeed Times | 1742 |
| FatigueDriving Times | 2105 |
| HthrottleStop Times | 193 |
| OilLeak Times | 218 |

| | |
|---|---|
| Overspeed Times | 1746 |
| FatigueDriving Times | 2106 |
| HthrottleStop Times | 194 |
| OilLeak Times | 218 |

### 2.2.3 Data Visualization

Data visualization is a feature that compensate to the previous real-time monitoring feature, particularly, the driver information analysis service. Each driver has their own diagram illustrating the past ten (maximum) driving speed in a line chart. While all the other records, data, and information are updated, the line chart also reloads and display the latest chart in real-time manner less than five seconds. Details can be seen in the figures provided below.

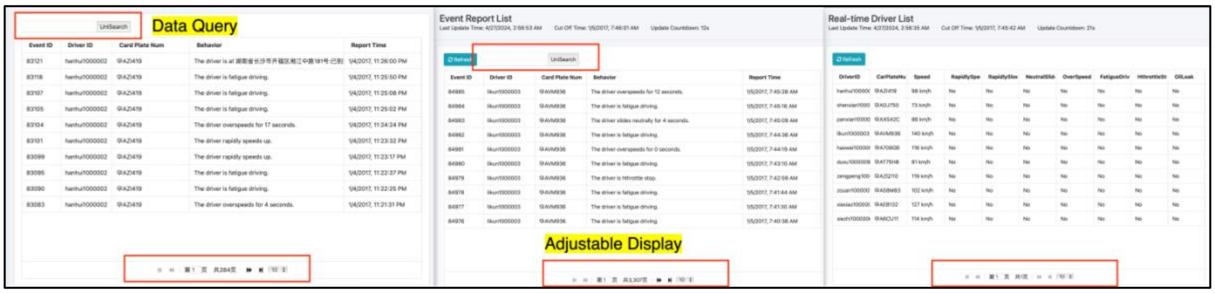


### 2.2.4 Data Query

Besides thorough analysis from different aspects, some features are added to make the web service interactive, extending the nature of the application from static contents with dynamic interactions. Accordingly, as well as navigation through pages, a querying feature allows users to search for particular record or records of particular driver simply by inputting driver ID and event ID (where event ID represents the read in record ID in chronological order). This feature can be accessed mainly through two pages, where two text boxes are shown in the Event Report List page and the driver information page.

### 2.2.5 Adjustable Display

To improve and optimize the user experience when accessing the web service, a minor feature targeting display optimization is implemented. The application allows user to adjust the number of rows of data and records to be displayed in the same page. The default number of rows is set to 10 while two other options are available, 20 and 50. It is assured that changing to different options does not affect the performance of the application and remain real-time consistency. Such feature is available in all pages where data are displayed in table or row format.



| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 83121 | hanhui1000002 | 湘AZ1419 | The driver is at 湖南省长沙市开福区湘江中路181号-已到 | 1/4/2017, 11:26:00 PM |
| 83118 | hanhui1000002 | 湘AZ1419 | The driver is fatigue driving. | 1/4/2017, 11:25:50 PM |
| 83107 | hanhui1000002 | 湘AZ1419 | The driver is fatigue driving. | 1/4/2017, 11:25:08 PM |
| 83105 | hanhui1000002 | 湘AZ1419 | The driver is fatigue driving. | 1/4/2017, 11:25:02 PM |
| 83104 | hanhui1000002 | 湘AZ1419 | The driver overspeeds for 17 seconds. | 1/4/2017, 11:24:24 PM |
| 83101 | hanhui1000002 | 湘AZ1419 | The driver rapidly speeds up. | 1/4/2017, 11:23:32 PM |
| 83099 | hanhui1000002 | 湘AZ1419 | The driver rapidly speeds up. | 1/4/2017, 11:23:17 PM |
| 83095 | hanhui1000002 | 湘AZ1419 | The driver is fatigue driving. | 1/4/2017, 11:22:37 PM |
| 83090 | hanhui1000002 | 湘AZ1419 | The driver is fatigue driving. | 1/4/2017, 11:22:25 PM |
| 83083 | hanhui1000002 | 湘AZ1419 | The driver overspeeds for 4 seconds. | 1/4/2017, 11:21:31 PM |

第1 页 共284页 10

**Event Report List**

Last Update Time: 4/27/2024, 2:58:53 AM    Cut Off Time: 1/5/2017, 7:46:01 AM    Update Countdown: 12s

| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 84985 | likun1000003 | 湘AVM936 | The driver overspeeds for 12 seconds. | 1/5/2017, 7:45:28 AM |
| 84984 | likun1000003 | 湘AVM936 | The driver is fatigue driving. | 1/5/2017, 7:45:16 AM |
| 84983 | likun1000003 | 湘AVM936 | The driver slides neutrally for 4 seconds. | 1/5/2017, 7:45:09 AM |
| 84982 | likun1000003 | 湘AVM936 | The driver is fatigue driving. | 1/5/2017, 7:44:36 AM |
| 84981 | likun1000003 | 湘AVM936 | The driver overspeeds for 0 seconds. | 1/5/2017, 7:44:19 AM |
| 84980 | likun1000003 | 湘AVM936 | The driver is fatigue driving. | 1/5/2017, 7:43:10 AM |
| 84979 | likun1000003 | 湘AVM936 | The driver is hthrottle stop. | 1/5/2017, 7:42:59 AM |
| 84978 | likun1000003 | 湘AVM936 | The driver is fatigue driving. | 1/5/2017, 7:41:44 AM |
| 84977 | likun1000003 | 湘AVM936 | The driver is fatigue driving. | 1/5/2017, 7:41:30 AM |
| 84976 | likun1000003 | 湘AVM936 | The driver is fatigue driving. | 1/5/2017, 7:40:36 AM |

第1 页 共3,307页 10

**Real-time Driver List**

Last Update Time: 4/27/2024, 2:58:35 AM    Cut Off Time: 1/5/2017, 7:45:42 AM    Update Countdown: 21s

| DriverID | CarPlateNu | Speed | RapidlySpe | RapidlySlov | NeutralSlid | OverSpeed | FatigueDriv | HthrottleSt | OilLeak |
|---|---|---|---|---|---|---|---|---|---|
| hanhui100000 | 湘AZ1419 | 98 km/h | No | No | No | No | No | No | No |
| shenxian1000 | 湘ADJ755 | 73 km/h | No | No | No | No | No | No | No |
| panxian10000 | 湘AXS42C | 88 km/h | No | No | No | No | No | No | No |
| likun1000003 | 湘AVM936 | 140 km/h | No | No | No | No | No | No | No |
| haowei10000 | 湘A709G8 | 116 km/h | No | No | No | No | No | No | No |
| duxu1000008 | 湘AT75H8 | 81 km/h | No | No | No | No | No | No | No |
| zengpeng100 | 湘AZQ110 | 119 km/h | No | No | No | No | No | No | No |
| zouan100000 | 湘AS8MB3 | 102 km/h | No | No | No | No | No | No | No |
| xiexiao10000 | 湘AEB132 | 127 km/h | No | No | No | No | No | No | No |
| xiechi100000 | 湘A6CU11 | 114 km/h | No | No | No | No | No | No | No |

第1 页 共1页 10

# 3 System Design

As required, all data should be analyzed with the Spark framework. On the other hand, the motivation is to provide a public web service that can be accessed like normal web applications. Thus, the design of the system should be targeting web service as a core basis while being compatible and integrating with Spark. As a result, the program is written in Java with the SpringBoot framework for building RESTful APIs and using Java Spark for data related operations.

## 3.1 SpringBoot

SpringBoot is an opensource Java-based framework ideal for creating micro services, web services, RESTful APIs, and standalone applications, just to name a few. Owing to its robustness, easy development and maintenance with dependency injection, minimum configuration, and scalability, the SpringBoot framework has been adopted in various enterprise level solutions in the past decades.

Being a Java framework also indicates its cross-platform availability. Beans and dependency injection (DI) are the two core basis of all Spring or SpringBoot applications. Managing object dependencies could be critical and troublesome in large scaled applications or could possibly affect the whole system with slight modifications. SpringBoot address the issue by introducing Beans.

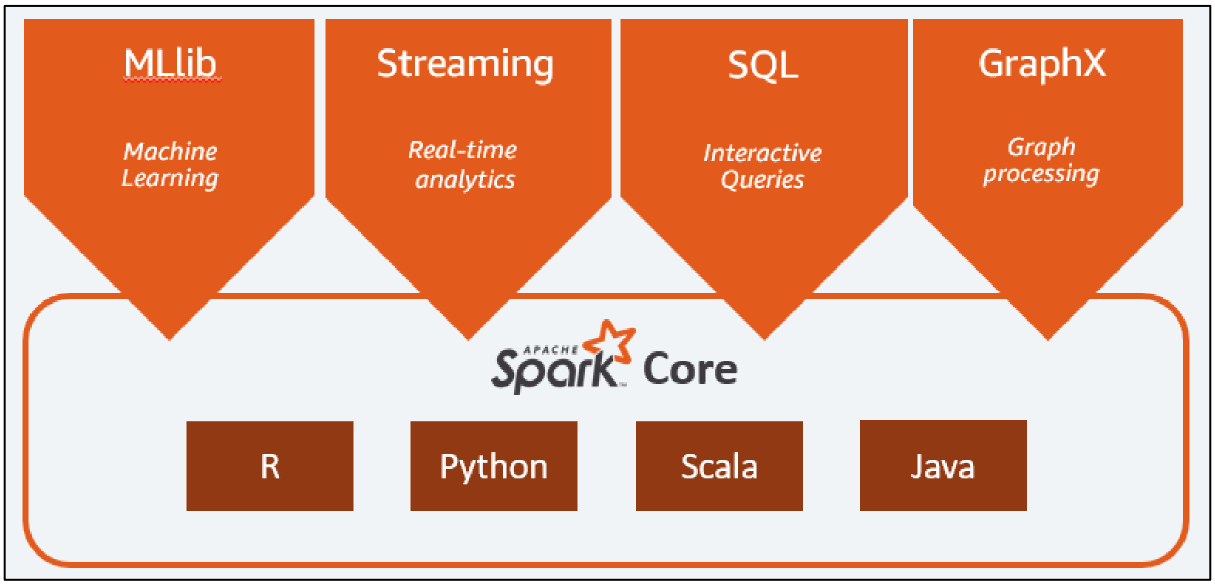
Beans function like normal Java objects but managed by the IoC container, considered the SpringBoot framework itself, with DI feature. DI is an implementation of Inversion of Control, to put it simply, instead of initializing objects in different classes during compiling, one would only need to specify the dependencies between objects. The IoC container will inject these Beans according to the specification as objects during runtime. Such design decouples the architecture, modulars the program, and lowers the difficulties switching implementations.

On the other hand, building RESTful web service with SpringBoot requires minimum configurations. SpringBoot bootstraps all necessary tools and packages to launch a web service automatically during runtime and provides simple annotations for development to ease the implementation process, including the servlet, the web configurations, the application container, and the Beans container. SpringBoot scans the whole project during runtime and configures the application according to the annotations specified in the code. By configuring the POMs, a set of convenient dependency (library) descriptors, developers can focus solely on the application logic instead of handling all the version controls and dependencies.

## 3.2 Spark

The Apache Spark framework is a multi-language engine for big data related operations. The distributed processing system of Spark addresses the limitation of the Hadoop programming model and allows in-memory caching and real-time analytic queries against data of any size. By creating DataFrames or Resilient Distributed Datasets (RDDs), the Spark framework is not only fast but can be executed on different nodes and different workloads with fault tolerance feature.

Although Spark is mainly written in Scala, it supports different languages for development, including Python, Scala, Java, and R. Spark can be used in different cases, according to the official documentation, Spark has mainly four streams, MLlib, Streaming, SQL, and GraphX, all built upon Spark Core. Spark Core is the foundation of the framework providing memory management, fault recovery, scheduling, and tasks management. As the goal of this project is to analyze and provide insights with the given driving data, only the Streaming and SQL tools are utilized.

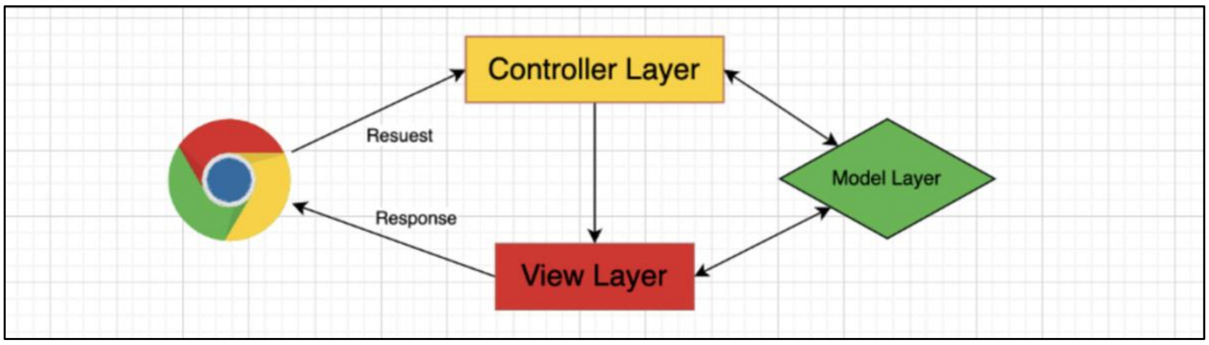


## 3.3 Program Architecture

Traditionally, web services are implemented and designed using the Model-View-Controller (MVC) architecture. The MVC model is made up of three components, the model, view, and controller layer.

- The view layer renders views to the client end users for either listening to request or returning the execution results.
- The controller layer serves as a manager similar to master node that manages both the web configuration and the whole application.
- The model layer is responsible for holding the application logic, data, and service to performed according to the incoming requests.

Even though the MVC model seems to be an optimal solution, various drawbacks have been introduced recently. A major issue is the inconvenient of CI/CD control that hinders the easiness of maintaining the web service in the long run. Hence, this project inherits the nature of SpringBoot and implements the application with a multi-layer architecture.



### 3.3.1 Multi-layer Architecture

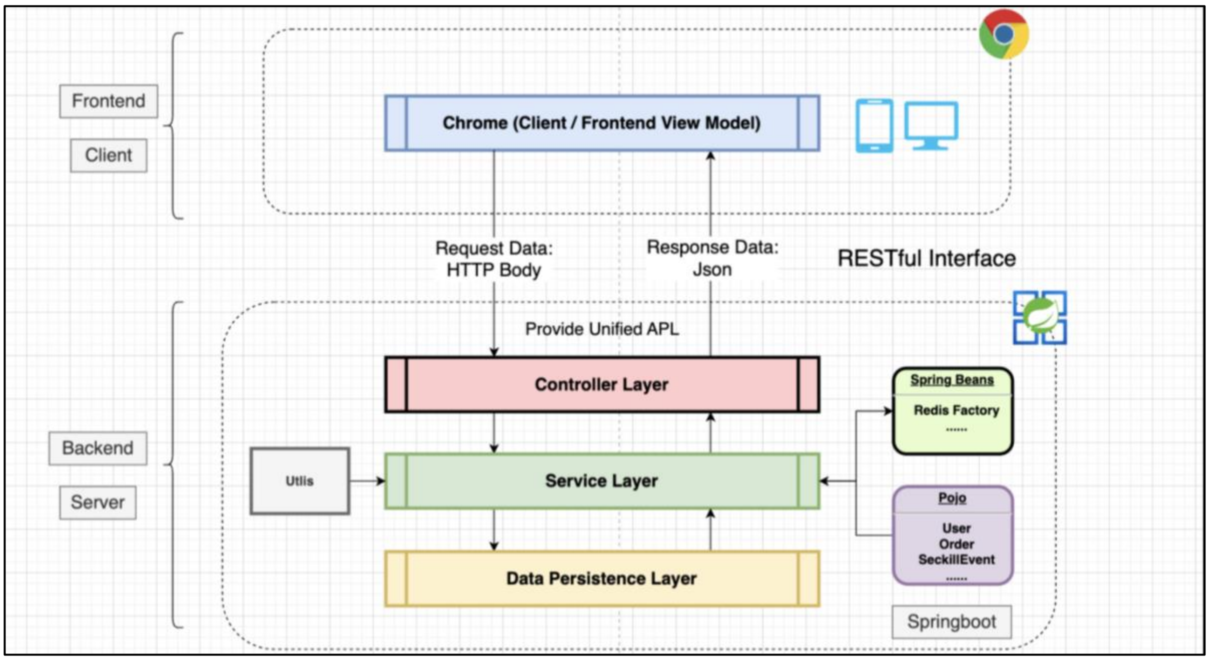
The multi-layer architecture completely separates the front end and backend, while the backend can further be divided into three different layers. As illustrated in the figure below, the front end model, also known as the view model, is responsible for listening to request data and rendering response data. The model accepts input request then pass it to the backend embedded in a HTTP body and return response data in JSON format in a web page.

Since SpringBoot web services are RESTful, the response data are automatically configured into JSON format. As for the backend, it consists of three layers, the Controller layer, Service layer, and Data persistence layer.

The Controller layer manages the input and output data but only interacts with the service layer. According to different input paths, Controller pass the data to different service interfaces in the service layer.

The service layer holds the interface and implements the core application logic and functionalities. This layer interacts with the Controller, the data, Spring Beans, and different entities (Plain-Old-Java-Objects, POJOs).

Lastly, the Data Persistence Layer fetches data via SparkSQL and data persistence via MySQL. With the multi-layer design, this project is able to achieve parallel development, improved maintainability, and seamless upgrade in CI/CD.

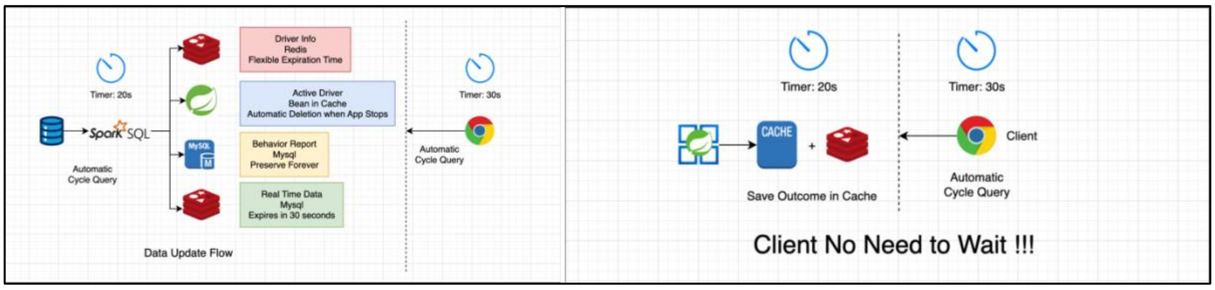


### 3.3.2 Asynchronous Design

HopTraf's client queries and server data updates are asynchronous. This asynchronous design was adopted with the aim of achieving faster response times and reducing the pressure of service downtime.

Clients periodically request data from the server. Instead of waiting for a whole process to complete before returning the data, the server responds immediately with cached data from memory. This approach allows the clients to receive the necessary information without delay, enhancing the overall user experience and efficiency of the system.

On the server side, data is periodically queried via SpsqrSQL and the analyzed results are stored in Redis until the next cycle. During this cycle, when a client request is received, the server directly returns the data from the cache. This methodology not only speeds up the data retrieval process but also reduces the load on the server, as it doesn't need to execute a full process for every client request. Therefore, the server can handle more client requests simultaneously, improving the overall performance of the system.

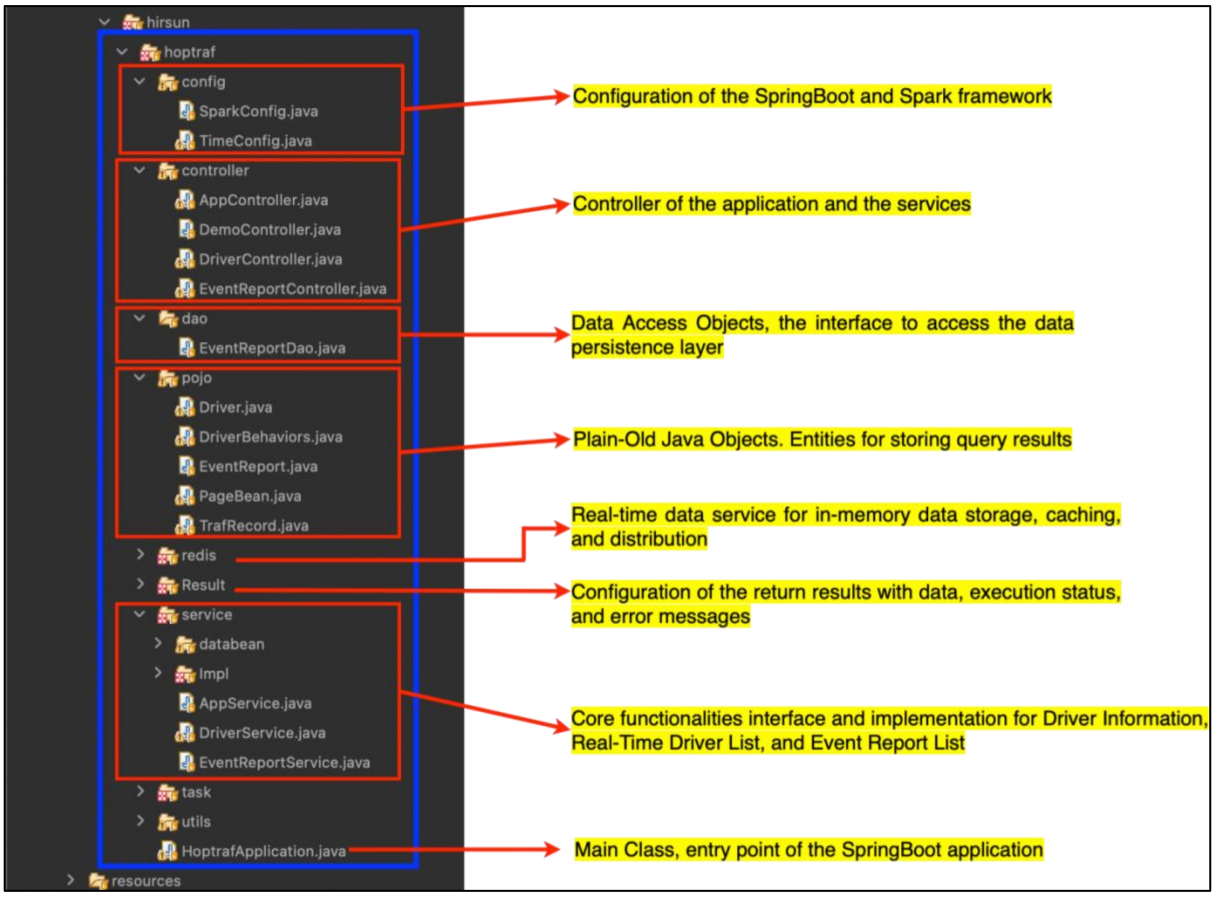


# 4 Implementation

## 4.1 Packages

The implementation codes are grouped into different packages to form the multi-layer architecture. Brief descriptions of each package can be seen in the following figure.

- The HoptrafApplication class is the entry point of the application, where the SpringBoot starts the whole web service. Apart from the configurations and controllers for different services, the main and core components that made up the services lies under the dao, pojo, redis, and service packages.
- The pojo package contains the Driver, DriverBehaviors, and EventReport classes. These classes serve as entities to be configured and return as part of the response JSON data.
- The dao package is an implementation of the data access object that separates the business logic with the persistence layer and provide access to the data storage using MySQL.
- Together with the redis package, frequently used data are stored and cached in-memory for real-time operations.
- In terms of the core functionalities, the AppService class manages the redis service, event report service, and the data time simulation feature; the DriverService is responsible for querying data for driver information and comprehends the driving behavior while the EventReportService configures a page Bean.



Configuration of the SpringBoot and Spark framework

Controller of the application and the services

Data Access Objects, the interface to access the data persistence layer

Plain-Old Java Objects. Entities for storing query results

Real-time data service for in-memory data storage, caching, and distribution

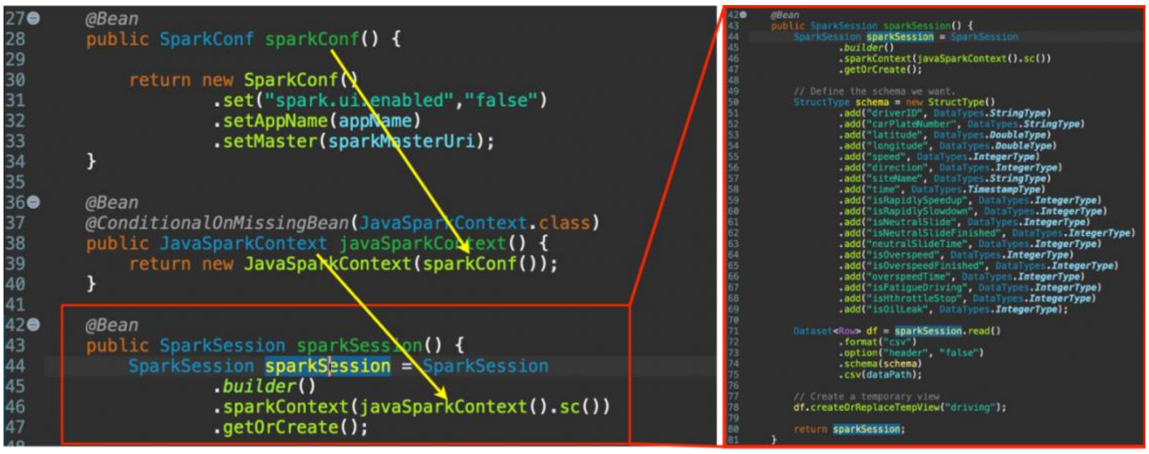
Configuration of the return results with data, execution status, and error messages

Core functionalities interface and implementation for Driver Information, Real-Time Driver List, and Event Report List

Main Class, entry point of the SpringBoot application

## 4.2 Spark Configuration

To utilize Spark for data analysis and operations, configuration of Spark is needed. The SparkConfig class is a configuration file containing Beans that will create Spark Session during runtime before the web service starts. The Spark Session initializes a Spark Context which sets the application name and master node URI. Furthermore, the Session reads in the 10 data source CSV files into a Dataset object for creating a temporal homepage view when accessing the web service.

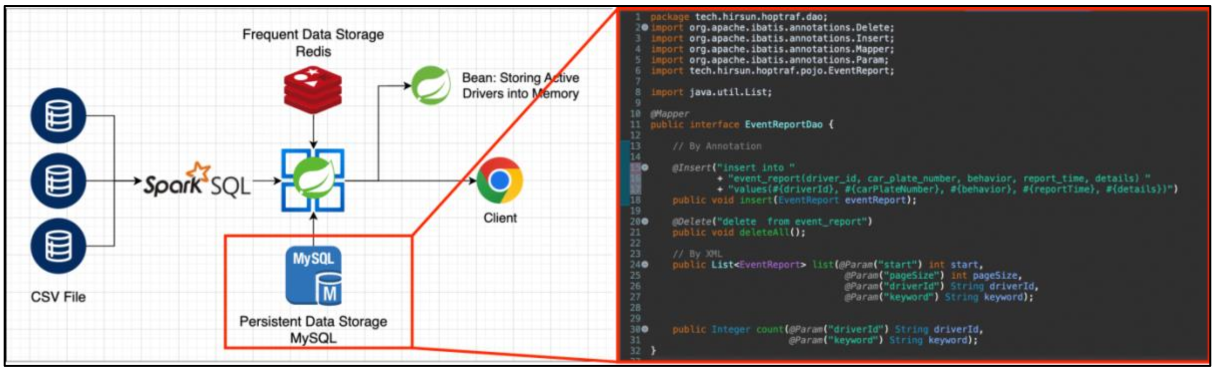


```java
    @Bean
    public SparkConf sparkConf() {

        return new SparkConf()
                .set("spark.ui.enabled","false")
                .setAppName(appName)
                .setMaster(sparkMasterUri);
    }

    @Bean
    @ConditionalOnMissingBean(JavaSparkContext.class)
    public JavaSparkContext javaSparkContext() {
        return new JavaSparkContext(sparkConf());
    }

    @Bean
    public SparkSession sparkSession() {
        SparkSession sparkSession = SparkSession
                .builder()
                .sparkContext(javaSparkContext().sc())
                .getOrCreate();
```

```java
    @Bean
    public SparkSession sparkSession() {
        SparkSession sparkSession = SparkSession
                .builder()
                .sparkContext(javaSparkContext().sc())
                .getOrCreate();

        // Define the schema we want.
        StructType schema = new StructType()
                .add("driverID", DataTypes.StringType)
                .add("carPlateNumber", DataTypes.StringType)
                .add("latitude", DataTypes.DoubleType)
                .add("longitude", DataTypes.DoubleType)
                .add("speed", DataTypes.IntegerType)
                .add("direction", DataTypes.IntegerType)
                .add("siteName", DataTypes.StringType)
                .add("time", DataTypes.TimestampType)
                .add("isRapidlySpeedup", DataTypes.IntegerType)
                .add("isRapidlySlowdown", DataTypes.IntegerType)
                .add("isNeutralSlide", DataTypes.IntegerType)
                .add("isNeutralSlideFinished", DataTypes.IntegerType)
                .add("neutralSlideTime", DataTypes.IntegerType)
                .add("isOverspeed", DataTypes.IntegerType)
                .add("isOverspeedFinished", DataTypes.IntegerType)
                .add("overspeedTime", DataTypes.IntegerType)
                .add("isFatigueDriving", DataTypes.IntegerType)
                .add("isHthrottleStop", DataTypes.IntegerType)
                .add("isOilLeak", DataTypes.IntegerType);

        Dataset<Row> df = sparkSession.read()
                .format("csv")
                .option("header", "false")
                .schema(schema)
                .csv(dataPath);

        // Create a temporary view
        df.createOrReplaceTempView("driving");

        return sparkSession;
    }
```

## 4.3 Data Access Object

The data access object interface uses MySQL to create the data persistence layer. Due to the fact that different data are accessed with different frequencies, frequently access and fetched data are further cached in-memory with redis. The SparkSQL is used only when fetching real-time data after 30 seconds countdown or manually refresh to update the view layer. In other words, the data access object and redis can be viewed as two persistent source of data for the application to fetch from, as shown in the left of the figure below.

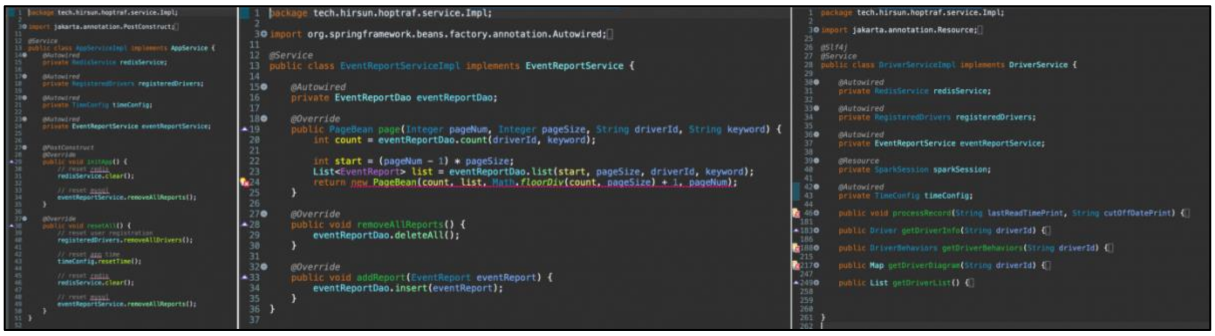


```java
package tech.hirsun.hoptraf.dao;
import org.apache.ibatis.annotations.Delete;
import org.apache.ibatis.annotations.Insert;
import org.apache.ibatis.annotations.Mapper;
import org.apache.ibatis.annotations.Param;
import tech.hirsun.hoptraf.pojo.EventReport;

import java.util.List;

@Mapper
public interface EventReportDao {

    // By Annotation

    @Insert("insert into "
            + "event_report(driver_id, car_plate_number, behavior, report_time, details) "
            + "values(#{driverId}, #{carPlateNumber}, #{behavior}, #{reportTime}, #{details})")
    public void insert(EventReport eventReport);

    @Delete("delete  from event_report")
    public void deleteAll();

    // By XML
    public List<EventReport> list(@Param("start") int start,
                                  @Param("pageSize") int pageSize,
                                  @Param("driverId") String driverId,
                                  @Param("keyword") String keyword);


    public Integer count(@Param("driverId") String driverId,
                         @Param("keyword") String keyword);
}
```

## 4.4 Services

The three service classes build up the core logics behind the three pages of the web service. As mentioned, the AppService initializes and resets the status of other services. The DriverService and EventReportService class query the required data using SparkSQL, jointly process the incoming request and update the Real-time Driver List, Event Report List, and Driver Information page.

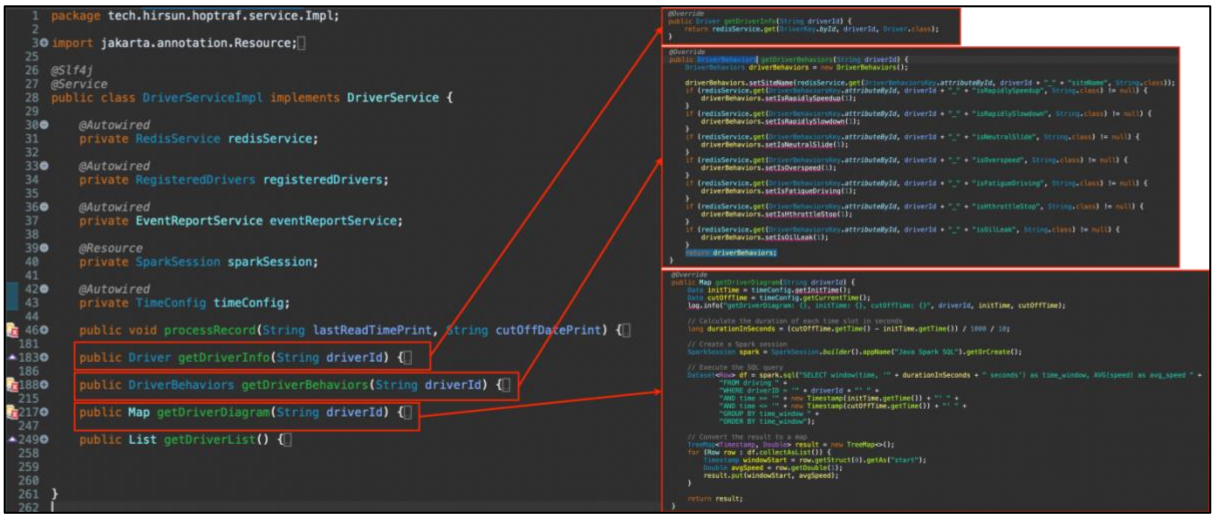
Majority of the functionalities can be seen implemented in the DriverService class, including the getting driver information function, getting driver driving behavior function, and even updating the real-time driving speed diagram function. These functions receive the query results and configure the Driver and DriverBehaviors entities into the Result object to form the response data.



```java
package tech.hirsun.hoptraf.service.Impl;

import jakarta.annotation.PostConstruct;

@Service
public class AppServiceImpl implements AppService {
    @Autowired
    private RedisService redisService;

    @Autowired
    private RegisteredDrivers registeredDrivers;

    @Autowired
    private TimeConfig timeConfig;

    @Autowired
    private EventReportService eventReportService;


    @PostConstruct
    @Override
    public void initApp() {
        // reset redis
        redisService.clear();

        // reset mysql
        eventReportService.removeAllReports();
    }

    @Override
    public void resetAll() {
        // reset user registration
        registeredDrivers.removeAllDrivers();

        // reset app time
        timeConfig.resetTime();

        // reset redis
        redisService.clear();

        // reset mysql
        eventReportService.removeAllReports();
    }
}
```

```java
package tech.hirsun.hoptraf.service.Impl;

import org.springframework.beans.factory.annotation.Autowired;

@Service
public class EventReportServiceImpl implements EventReportService {

    @Autowired
    private EventReportDao eventReportDao;

    @Override
    public PageBean page(Integer pageNum, Integer pageSize, String driverId, String keyword) {
        int count = eventReportDao.count(driverId, keyword);

        int start = (pageNum - 1) * pageSize;
        List<EventReport> list = eventReportDao.list(start, pageSize, driverId, keyword);
        return new PageBean(count, list, Math.floorDiv(count, pageSize) + 1, pageNum);
    }

    @Override
    public void removeAllReports() {
        eventReportDao.deleteAll();
    }

    @Override
    public void addReport(EventReport eventReport) {
        eventReportDao.insert(eventReport);
    }
}
```

```java
package tech.hirsun.hoptraf.service.Impl;

import jakarta.annotation.Resource;

@Slf4j
@Service
public class DriverServiceImpl implements DriverService {

    @Autowired
    private RedisService redisService;

    @Autowired
    private RegisteredDrivers registeredDrivers;

    @Autowired
    private EventReportService eventReportService;

    @Resource
    private SparkSession sparkSession;

    @Autowired
    private TimeConfig timeConfig;

    public void processRecord(String lastReadTimePrint, String cutOffDatePrint) {

    public Driver getDriverInfo(String driverId) {

    public DriverBehaviors getDriverBehaviors(String driverId) {

    public Map getDriverDiagram(String driverId) {

    public List getDriverList() {



}
```



```java
package tech.hirsun.hoptraf.service.Impl;

import jakarta.annotation.Resource;

@Slf4j
@Service
public class DriverServiceImpl implements DriverService {

    @Autowired
    private RedisService redisService;

    @Autowired
    private RegisteredDrivers registeredDrivers;

    @Autowired
    private EventReportService eventReportService;

    @Resource
    private SparkSession sparkSession;

    @Autowired
    private TimeConfig timeConfig;

    public void processRecord(String lastReadTimePrint, String cutOffDatePrint) {

    public Driver getDriverInfo(String driverId) {

    public DriverBehaviors getDriverBehaviors(String driverId) {

    public Map getDriverDiagram(String driverId) {

    public List getDriverList() {



}
```

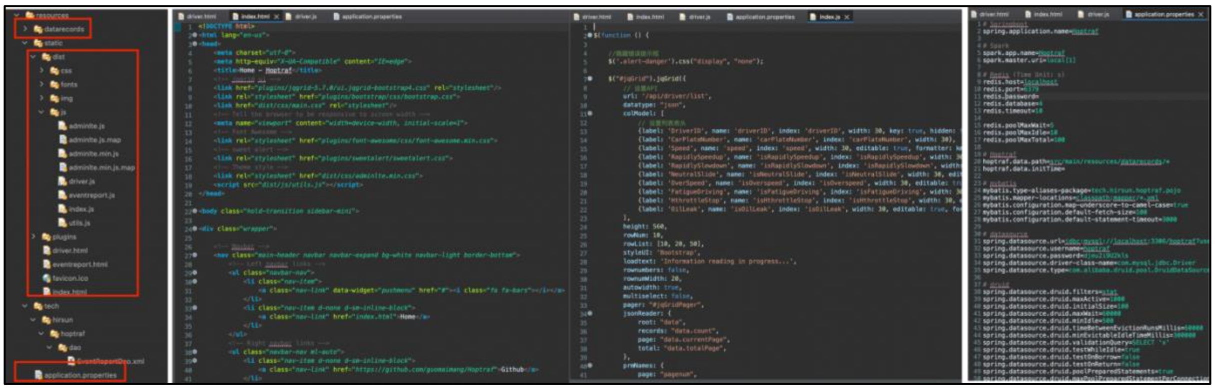
```java
@Override
public Driver getDriverInfo(String driverId) {
    return redisService.get(DriverKey.byId, driverId, Driver.class);
}
```

```java
@Override
public DriverBehaviors getDriverBehaviors(String driverId) {
    DriverBehaviors driverBehaviors = new DriverBehaviors();

    driverBehaviors.setSiteName(redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "siteName", String.class));
    if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isRapidlySpeedup", String.class) != null) {
        driverBehaviors.setIsRapidlySpeedup(1);
    }
    if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isRapidlySlowdown", String.class) != null) {
        driverBehaviors.setIsRapidlySlowdown(1);
    }
    if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isNeutralSlide", String.class) != null) {
        driverBehaviors.setIsNeutralSlide(1);
    }
    if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isOverspeed", String.class) != null) {
        driverBehaviors.setIsOverspeed(1);
    }
    if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isFatigueDriving", String.class) != null) {
        driverBehaviors.setIsFatigueDriving(1);
    }
    if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isHthrottleStop", String.class) != null) {
        driverBehaviors.setIsHthrottleStop(1);
    }
    if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isOilLeak", String.class) != null) {
        driverBehaviors.setIsOilLeak(1);
    }
    return driverBehaviors;
}
```

```java
@Override
public Map getDriverDiagram(String driverId) {
    Date initTime = timeConfig.getInitTime();
    Date cutOffTime = timeConfig.getCurrentTime();
    log.info("getDriverDiagram: {}, initTime: {}, cutOffTime: {}", driverId, initTime, cutOffTime);

    // Calculate the duration of each time slot in seconds
    long durationInSeconds = (cutOffTime.getTime() - initTime.getTime()) / 1000 / 10;

    // Create a Spark session
    SparkSession spark = SparkSession.builder().appName("Java Spark SQL").getOrCreate();

    // Execute the SQL query
    Dataset<Row> df = spark.sql("SELECT window(time, '" + durationInSeconds + " seconds') as time_window, AVG(speed) as avg_speed " +
            "FROM driving " +
            "WHERE driverID = '" + driverId + "' " +
            "AND time >= '" + new Timestamp(initTime.getTime()) + "' " +
            "AND time <= '" + new Timestamp(cutOffTime.getTime()) + "' " +
            "GROUP BY time_window " +
            "ORDER BY time_window");

    // Convert the result to a map
    TreeMap<Timestamp, Double> result = new TreeMap<>();
    for (Row row : df.collectAsList()) {
        Timestamp windowStart = row.getStruct(0).getAs("start");
        Double avgSpeed = row.getDouble(1);
        result.put(windowStart, avgSpeed);
    }

    return result;
}
```

## 4.5 Static Resources

All the aforementioned implementations are focusing on the backend logic. The front end web pages, properties files, as well as the given 10 CSV files are stored as static resources outside the packages under the resource folder. The front end view layer is implemented in HTML with JavaScript and jQuery for dynamic content update and display. The application properties file, on the other hand, stores the static values of certain properties and references to the underlying classes that consume them. For instance, the SpringBoot application name, Spark configuration values, redis host, port, database connection settings and credentials, just to name a few.



```
<!DOCTYPE html>
<html lang="en-us">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Home - Hoptraf</title>
    <!-- jqGrid ui -->
    <link href="plugins/jqgrid-5.7.0/ui.jqgrid-bootstrap4.css" rel="stylesheet"/>
    <link rel="stylesheet" href="plugins/bootstrap/css/bootstrap.css">
    <link href="dist/css/main.css" rel="stylesheet"/>
    <!-- Tell the browser to be responsive to screen width -->
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Font Awesome -->
    <link rel="stylesheet" href="plugins/font-awesome/css/font-awesome.min.css">
    <!-- Sweet Alert -->
    <link rel="stylesheet" href="plugins/sweetalert/sweetalert.css">
    <!-- Theme style -->
    <link rel="stylesheet" href="dist/css/adminlte.min.css">
    <script src="dist/js/utils.js"></script>
</head>

<body class="hold-transition sidebar-mini">

<div class="wrapper">

    <!-- Navbar -->
    <nav class="main-header navbar navbar-expand bg-white navbar-light border-bottom">
        <!-- Left navbar links -->
        <ul class="navbar-nav">
            <li class="nav-item">
                <a class="nav-link" data-widget="pushmenu" href="#"><i class="fa fa-bars"></i></a>
            </li>
            <li class="nav-item d-none d-sm-inline-block">
                <a class="nav-link" href="index.html">Home</a>
            </li>
        </ul>
        <!-- Right navbar links -->
        <ul class="navbar-nav ml-auto">
            <li class="nav-item d-none d-sm-inline-block">
                <a class="nav-link" href="https://github.com/guomaimang/Hoptraf">Github</a>
            </li>
```

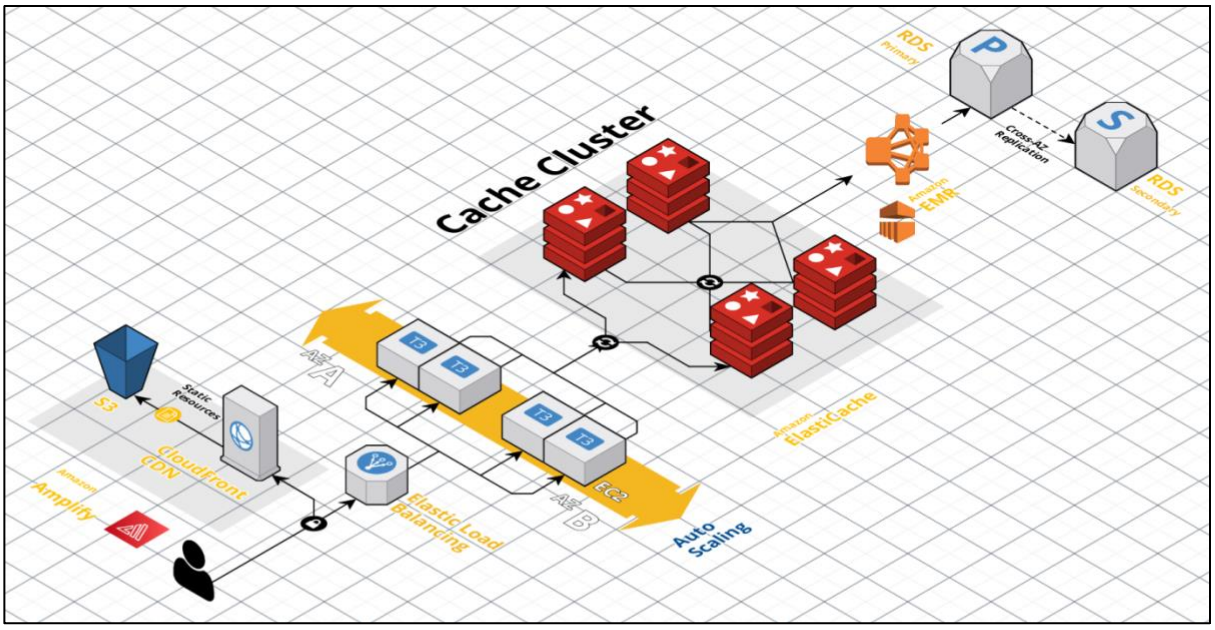
```

$(function () {

    //[illegible]
    $('.alert-danger').css("display", "none");

    $("#jqGrid").jqGrid({
        // [illegible]API
        url: '/api/driver/list',
        datatype: "json",
        colModel: [
            // [illegible]
            {label: 'DriverID', name: 'driverID', index: 'driverID', width: 30, key: true, hidden:
            {label: 'CarPlateNumber', name: 'carPlateNumber', index: 'carPlateNumber', width: 30},
            {label: 'Speed', name: 'speed', index: 'speed', width: 30, editable: true, formatter: k
            {label: 'RapidlySpeedup', name: 'isRapidlySpeedup', index: 'isRapidlySpeedup', width: 3
            {label: 'RapidlySlowdown', name: 'isRapidlySlowdown', index: 'isRapidlySlowdown', width
            {label: 'NeutralSlide', name: 'isNeutralSlide', index: 'isNeutralSlide', width: 30, edi
            {label: 'OverSpeed', name: 'isOverspeed', index: 'isOverspeed', width: 30, editable: tr
            {label: 'FatigueDriving', name: 'isFatigueDriving', index: 'isFatigueDriving', width: 3
            {label: 'HthrottleStop', name: 'isHthrottleStop', index: 'isHthrottleStop', width: 30,
            {label: 'OilLeak', name: 'isOilLeak', index: 'isOilLeak', width: 30, editable: true, fo
        ],
        height: 560,
        rowNum: 10,
        rowList: [10, 20, 50],
        styleUI: 'Bootstrap',
        loadtext: 'Information reading in progress...',
        rownumbers: false,
        rownumWidth: 20,
        autowidth: true,
        multiselect: false,
        pager: "#jqGridPager",
        jsonReader: {
            root: "data",
            records: "data.count",
            page: "data.currentPage",
            total: "data.totalPage",
        },
        prmNames: {
            page: "pagenum",
```

```
# Springboot
spring.application.name=hoptraf

# Spark
spark.app.name=hoptraf
spark.master.uri=local[1]

# Redis (Time Unit: s)
redis.host=localhost
redis.port=6379
redis.password=
redis.database=6
redis.timeout=10

redis.poolMaxWait=5
redis.poolMaxIdle=10
redis.poolMaxTotal=100

# hoptraf
hoptraf.data.path=src/main/resources/datarecords/*
hoptraf.data.initTime=

# mybatis
mybatis.type-aliases-package=tech.hirsun.hoptraf.pojo
mybatis.mapper-locations=classpath:mapper/*.xml
mybatis.configuration.map-underscore-to-camel-case=true
mybatis.configuration.default-fetch-size=100
mybatis.configuration.default-statement-timeout=3000

# datasource
spring.datasource.url=jdbc:mysql://localhost:3306/hoptraf?use
spring.datasource.username=hoptraf
spring.datasource.password=[illegible]
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.datasource.type=com.alibaba.druid.pool.DruidDataSource

# druid
spring.datasource.druid.filters=stat
spring.datasource.druid.maxActive=1000
spring.datasource.druid.initialSize=100
spring.datasource.druid.maxWait=60000
spring.datasource.druid.minIdle=500
spring.datasource.druid.timeBetweenEvictionRunsMillis=60000
spring.datasource.druid.minEvictableIdleTimeMillis=300000
spring.datasource.druid.validationQuery=SELECT 'x'
spring.datasource.druid.testWhileIdle=true
spring.datasource.druid.testOnBorrow=false
spring.datasource.druid.testOnReturn=false
spring.datasource.druid.poolPreparedStatements=true
spring.datasource.druid.maxPoolPreparedStatementPerConnection
```

# 5 Cloud Computing with AWS

## 5.1 AWS Deployment Architecture

Amazon Web Services (AWS) offers flexible, scalable services that are instrumental for processing the requirements of the HopTraf application, from hosting a simple website to processing large-scale data analysis. The deployment is designed in a way to handle static and dynamic content, process large amount of data effectively, and maintain strongly scalable database system. Several AWS services are string-ed together in the following way:



### 5.1.1 AWS Amplify: Hosting Static Websites

The AWS Amplify is utilized for hosting static components of our application. It automatically deploys the front end contents from the code repository into a global Content Delivery Network (CDN). This will save on deployment processes and optimize content delivery. Users in different geographical locations experience little latency and hence faster loading time for the page hosting HTML, CSS, and JavaScript files in S3 buckets and serving it from a CDN. Furthermore, AWS Amplify enables easy scalability during unexpected spikes in traffic, ensuring that applications are still responsive and available without any manual interference.

### 5.1.2 Apache Spark on Amazon EMR

For processing the large datasets involving driving behaviors, the Apache Spark framework is hosted on Amazon EMR. EMR grants the capability to work on a managed Hadoop framework, making it convenient to carry out activities in big data processing. This environment provides the ability to use Apache Spark in effecting very complex data transformations and analyses at speed. Such design is considered important because the scaling of Amazon EMR makes it possible for resources to be resized dynamically in the face of the processing requirements, thus improving cost optimizations, and increasing processing speed. Running EMR also reduced the operational overhead to maintain a big data ecosystem.

### 5.1.3 Amazon ElastiCache for Redis

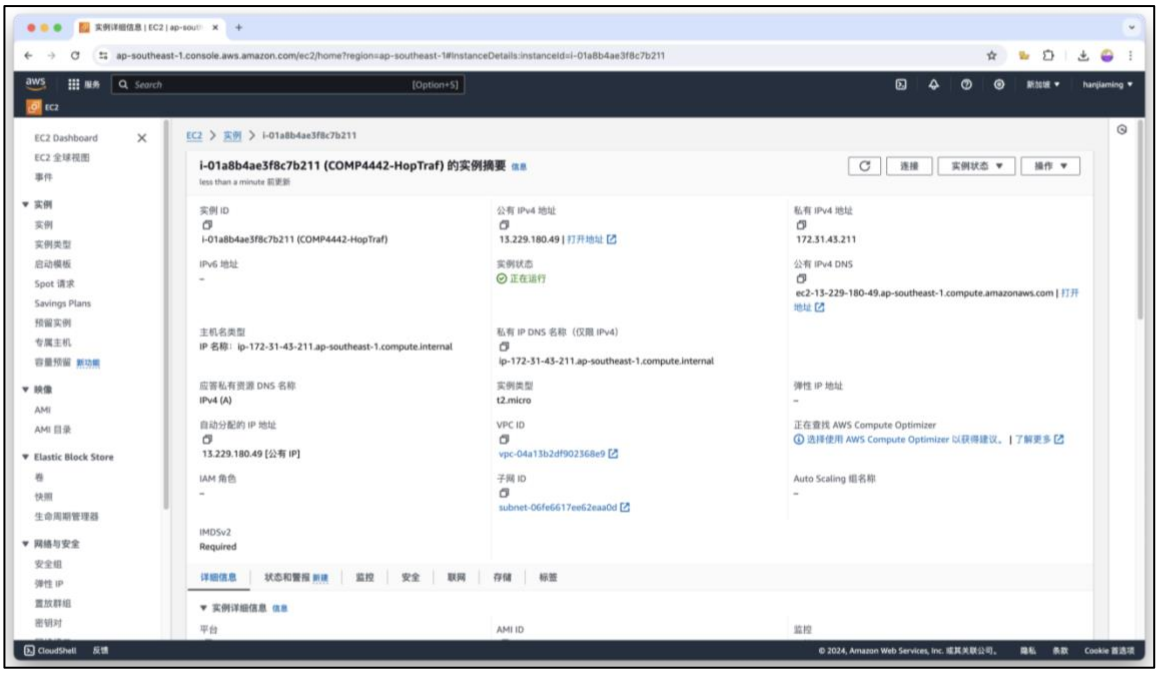
Deploying Redis service with Amazon ElastiCache helps increase application performance by caching application-level frequent queries on data. This lightens the database load and shortens response times for user requests, especially those that include dynamic content dependent on speedy retrieval of data. ElastiCache acts as a middle layer by caching data from the backend, meaning that user experience does not have to bear the brunt of latencies that would otherwise occur from direct database querying. This setup is ideal for real-time functionalities that need on-the-go access to data, such as real-time analytics and live data visualization.

### 5.1.4 Amazon RDS for MySQL

In terms of the Data Persistence layer, the Amazon RDS is chosen to satisfy the requirements of MySQL. Amazon RDS makes it easy to set up, operate, and scale relational schemas in the cloud. The service provides a resizable and cost-efficient capacity, which in turn, helps to offload administration tasks such as hardware provisioning, database setup, patching, and backups that are time-consuming by automation. This is significant for the application since it guarantees data integrity and robustness. Being able to scale the resources elastically with minimal downtime is of absolute importance, not to mentioned that the read in data continues to grow in time and the demands from the application keep on evolving.

## 5.2 Deployment on AWS

### 5.2.1 Backend (EC2)



i-01a8b4ae3f8c7b211 (COMP4442-HopTraf) 的实例摘要 信息
less than a minute 前更新

实例 ID
i-01a8b4ae3f8c7b211 (COMP4442-HopTraf)

公有 IPv4 地址
13.229.180.49 | 打开地址

私有 IPv4 地址
172.31.43.211

IPv6 地址
–

实例状态
正在运行

公有 IPv4 DNS
ec2-13-229-180-49.ap-southeast-1.compute.amazonaws.com | 打开地址

主机名类型
IP 名称：ip-172-31-43-211.ap-southeast-1.compute.internal

私有 IP DNS 名称（仅限 IPv4）
ip-172-31-43-211.ap-southeast-1.compute.internal

应答私有资源 DNS 名称
IPv4 (A)

实例类型
t2.micro

弹性 IP 地址
–

自动分配的 IP 地址
13.229.180.49 [公有 IP]

VPC ID
vpc-04a13b2df902368e9

正在查找 AWS Compute Optimizer
选择使用 AWS Compute Optimizer 以获得建议。 | 了解更多

IAM 角色
–

子网 ID
subnet-06fe6617ee62eaa0d

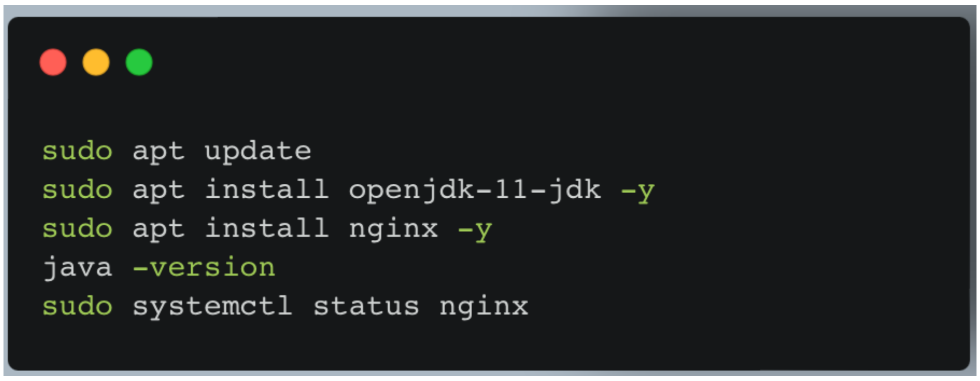
Auto Scaling 组名称
–

IMDSv2
Required

实例详细信息 信息
平台
AMI ID
监控

The deployment of the backend starts with the initialization of an EC2 (Elastic Compute Cloud) instance. EC2 provides scalable computing capacity in the AWS cloud which makes it an ideal candidate for hosting our backend operations, including the SpringBoot application.

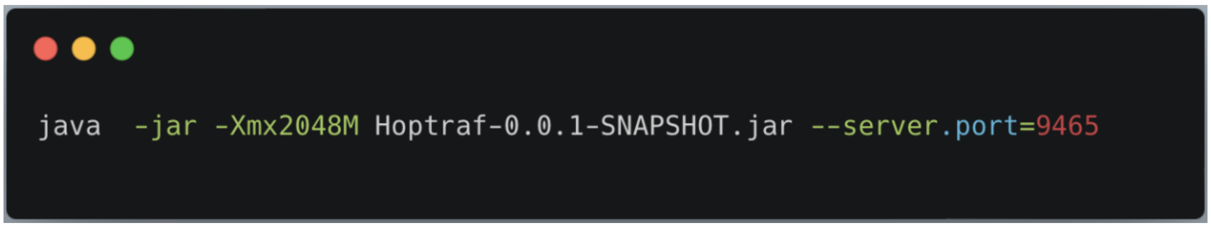
**Environment Setup.** The first step in setting up the backend on an EC2 instance is to configure the required environments. This includes setting up the Java environment necessary for running our SpringBoot application and configuring the Nginx server, which acts as a reverse proxy to enhance security and load balancing. Detailed instructions for configuring the Nginx environment can be found in the appendix.



```
sudo apt update
sudo apt install openjdk-11-jdk -y
sudo apt install nginx -y
java -version
sudo systemctl status nginx
```

**Starting the Backend Framework.** Once the environments are configured, the backend framework is started by executing specific commands which launch the SpringBoot application. This process involves:

- Running a series of shell commands to initiate the Java application. These commands typically involve navigating to the directory containing the executable jar file and using a Java command to run the application, such as:



```
java  -jar -Xmx2048M Hoptraf-0.0.1-SNAPSHOT.jar --server.port=9465
```

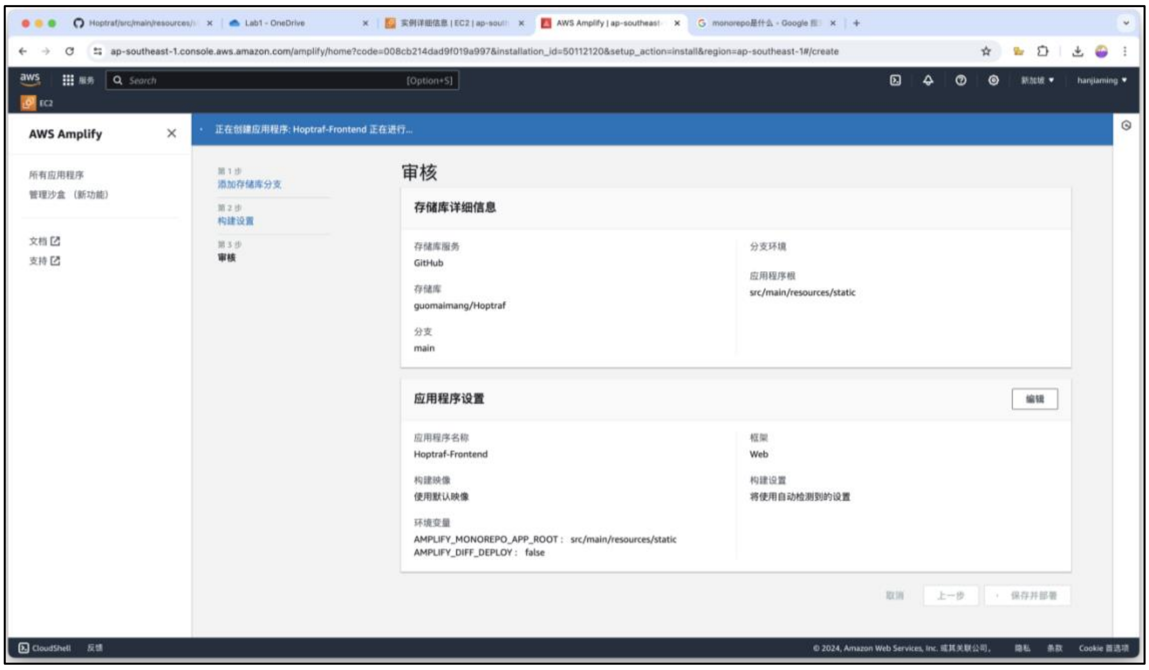
- Ensuring that all environment variables and necessary configurations are correctly set up to link the application with other AWS services and external dependencies it might have.

**Verification.** After the application starts, it's crucial to verify that it is running correctly and accessible. This might involve checking log files for any immediate errors thrown during startup and confirming that the EC2 instance is accepting connections on the appropriate ports.

### 5.2.2 Deploying the Front-end on AWS Amplify

For the front-end, AWS Amplify simplifies the deployment process significantly. Amplify integrates seamlessly with continuous integration and deployment pipelines and supports direct linking to code repositories for automated builds and deployments.

**Creating an Amplify Application.** To deploy the front-end, first, create a new application in AWS Amplify and link it to the GitHub repository (e.g., guomaimang/Hoptraf). This linkage allows Amplify to automatically detect changes to the repository, build the updated version, and deploy it without manual intervention.



ap-southeast-1.console.aws.amazon.com/amplify/home?code=008cb214dad9f019a997&installation_id=50112120&setup_action=install&region=ap-southeast-1#/create

正在创建应用程序: Hoptraf-Frontend 正在进行...

第 1 步 添加存储库分支
第 2 步 构建设置
第 3 步 审核

审核

**存储库详细信息**

| | |
|---|---|
| 存储库服务 | GitHub |
| 存储库 | guomaimang/Hoptraf |
| 分支 | main |
| 分支环境 | |
| 应用程序根 | src/main/resources/static |

**应用程序设置**

| | |
|---|---|
| 应用程序名称 | Hoptraf-Frontend |
| 构建映像 | 使用默认映像 |
| 框架 | Web |
| 构建设置 | 将使用自动检测到的设置 |

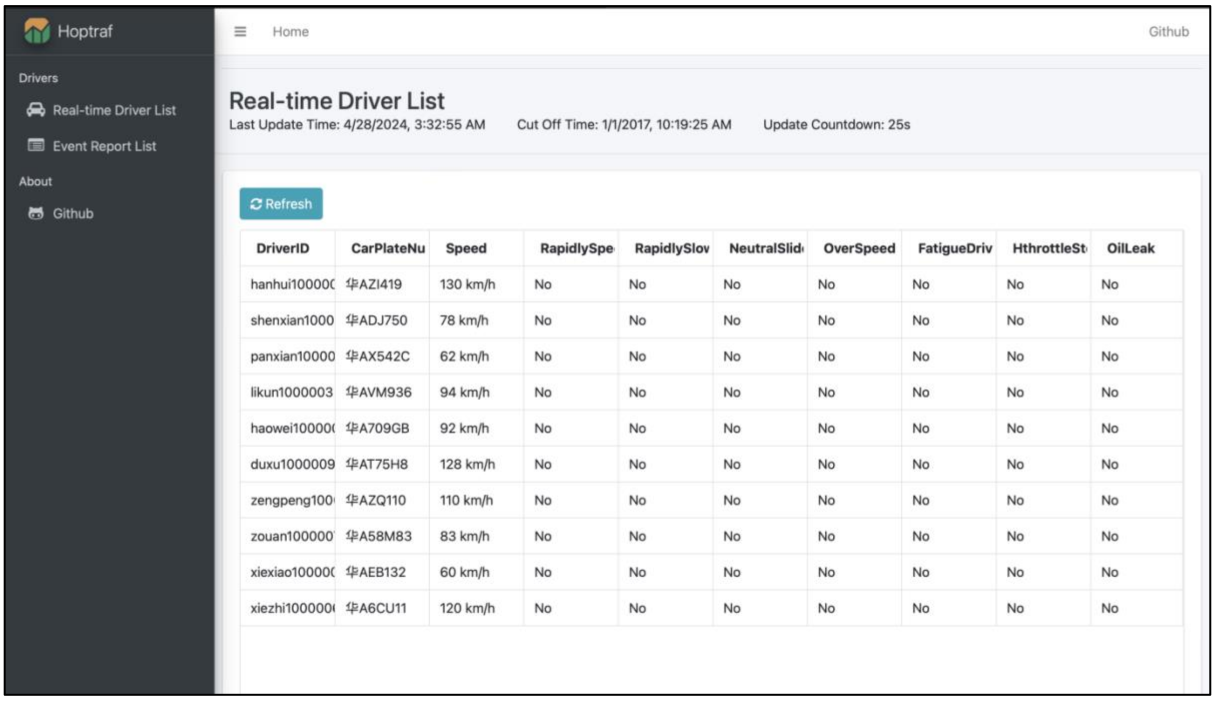
环境变量
```
AMPLIFY_MONOREPO_APP_ROOT :  src/main/resources/static
AMPLIFY_DIFF_DEPLOY :  false
```

**Automatic Deployment.** Upon creation, AWS Amplify automatically sets up the environment and starts the deployment process. This includes installing dependencies, building the project according to the specifications provided in the amplify.yml file, and deploying the static files to a globally distributed CDN.

**Verification.** After deployment, this project utilizes a dial-up testing tool provided by Alibaba to verify that the site is accessible worldwide. This tool simulates requests from various global locations to ensure that the website is reachable and performs as expected across different regions. The verification results can be seen in the following link here.

## 6 Evaluation

The application is evaluated in different use cases with provided screenshots in this section. Real-Time driving behavior of 10 drivers that updates every 30 seconds:

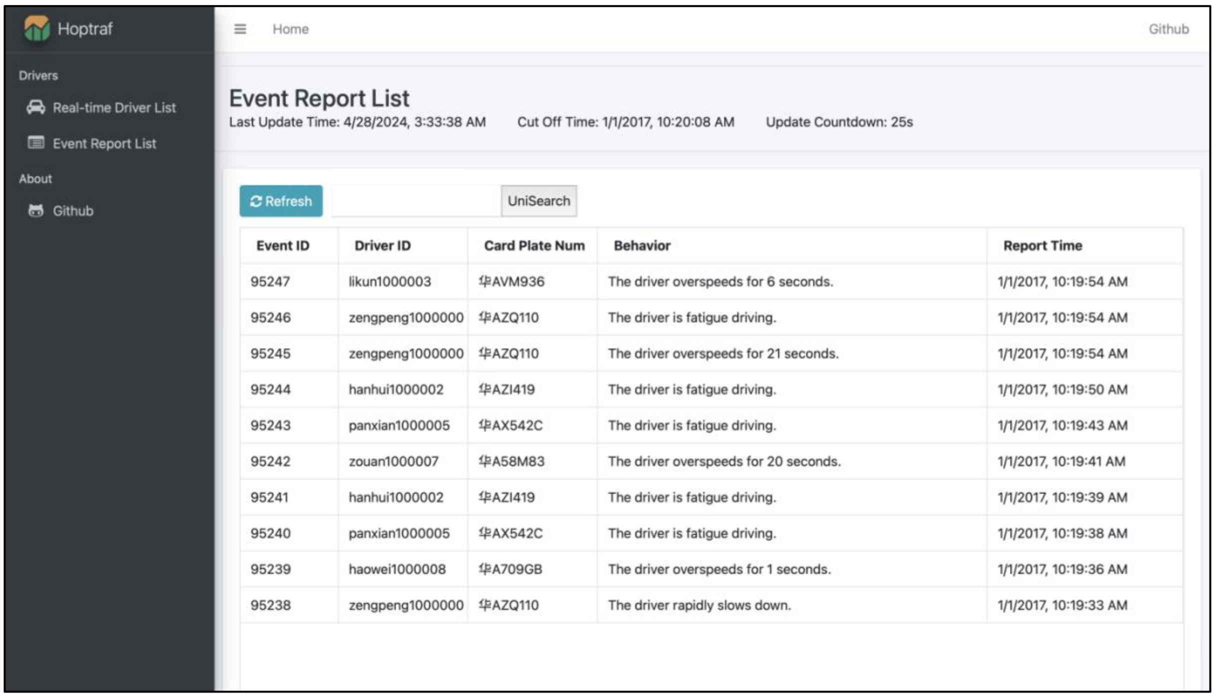


Real-time Driver List

Last Update Time: 4/28/2024, 3:32:55 AM  Cut Off Time: 1/1/2017, 10:19:25 AM  Update Countdown: 25s

| DriverID | CarPlateNu | Speed | RapidlySpe | RapidlySlov | NeutralSlid | OverSpeed | FatigueDriv | HthrottleSt | OilLeak |
|---|---|---|---|---|---|---|---|---|---|
| hanhui10000 | 华AZI419 | 130 km/h | No | No | No | No | No | No | No |
| shenxian1000 | 华ADJ750 | 78 km/h | No | No | No | No | No | No | No |
| panxian10000 | 华AX542C | 62 km/h | No | No | No | No | No | No | No |
| likun1000003 | 华AVM936 | 94 km/h | No | No | No | No | No | No | No |
| haowei10000 | 华A709GB | 92 km/h | No | No | No | No | No | No | No |
| duxu1000009 | 华AT75H8 | 128 km/h | No | No | No | No | No | No | No |
| zengpeng100 | 华AZQ110 | 110 km/h | No | No | No | No | No | No | No |
| zouan100000 | 华A58M83 | 83 km/h | No | No | No | No | No | No | No |
| xiexiao10000 | 华AEB132 | 60 km/h | No | No | No | No | No | No | No |
| xiezhi100000 | 华A6CU11 | 120 km/h | No | No | No | No | No | No | No |

Latest driving records in time descending order that updates every 30 seconds:

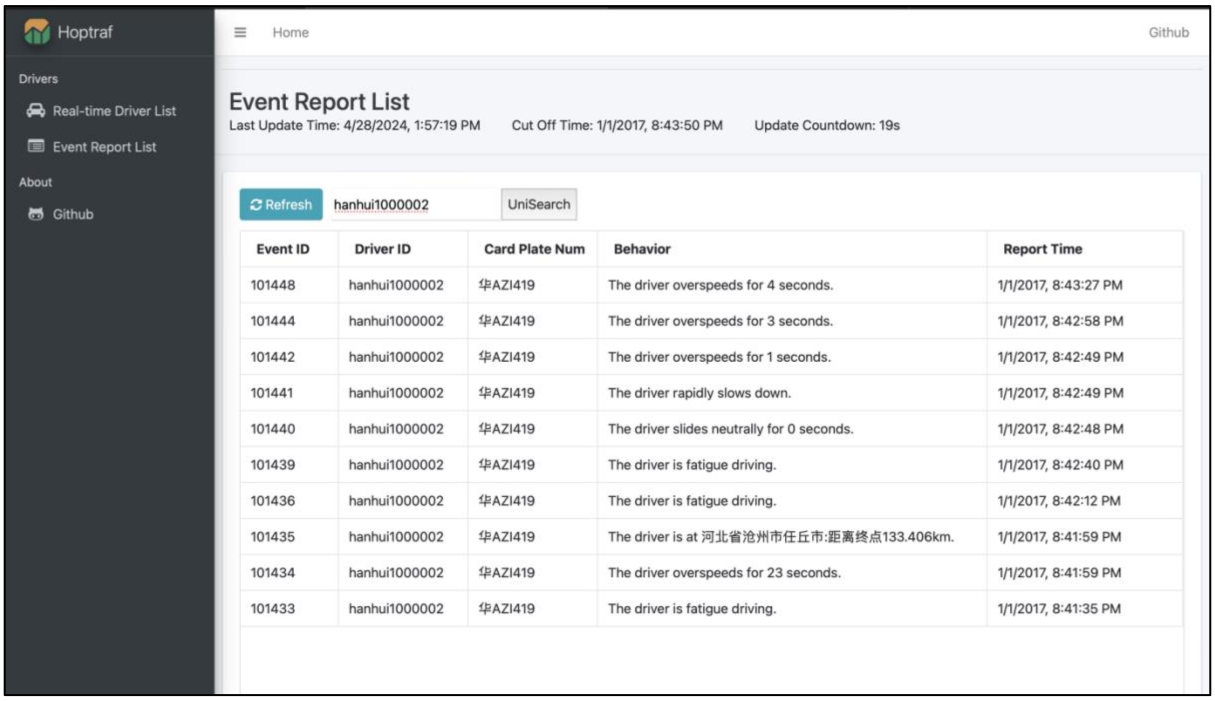


Event Report List

Last Update Time: 4/28/2024, 3:33:38 AM  Cut Off Time: 1/1/2017, 10:20:08 AM  Update Countdown: 25s

| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 95247 | likun1000003 | 华AVM936 | The driver overspeeds for 6 seconds. | 1/1/2017, 10:19:54 AM |
| 95246 | zengpeng1000000 | 华AZQ110 | The driver is fatigue driving. | 1/1/2017, 10:19:54 AM |
| 95245 | zengpeng1000000 | 华AZQ110 | The driver overspeeds for 21 seconds. | 1/1/2017, 10:19:54 AM |
| 95244 | hanhui1000002 | 华AZI419 | The driver is fatigue driving. | 1/1/2017, 10:19:50 AM |
| 95243 | panxian1000005 | 华AX542C | The driver is fatigue driving. | 1/1/2017, 10:19:43 AM |
| 95242 | zouan1000007 | 华A58M83 | The driver overspeeds for 20 seconds. | 1/1/2017, 10:19:41 AM |
| 95241 | hanhui1000002 | 华AZI419 | The driver is fatigue driving. | 1/1/2017, 10:19:39 AM |
| 95240 | panxian1000005 | 华AX542C | The driver is fatigue driving. | 1/1/2017, 10:19:38 AM |
| 95239 | haowei1000008 | 华A709GB | The driver overspeeds for 1 seconds. | 1/1/2017, 10:19:36 AM |
| 95238 | zengpeng1000000 | 华AZQ110 | The driver rapidly slows down. | 1/1/2017, 10:19:33 AM |

Querying particular driver records with driver ID:



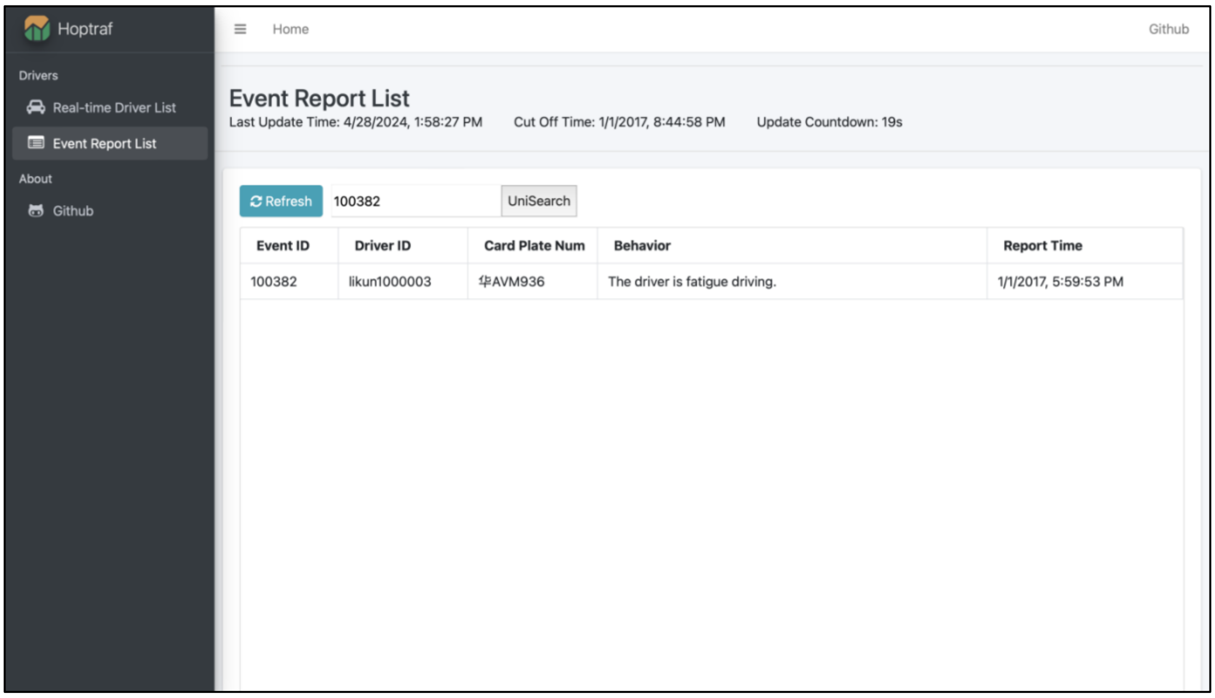
Event Report List

Last Update Time: 4/28/2024, 1:57:19 PM Cut Off Time: 1/1/2017, 8:43:50 PM Update Countdown: 19s

hanhui1000002

| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 101448 | hanhui1000002 | 华AZI419 | The driver overspeeds for 4 seconds. | 1/1/2017, 8:43:27 PM |
| 101444 | hanhui1000002 | 华AZI419 | The driver overspeeds for 3 seconds. | 1/1/2017, 8:42:58 PM |
| 101442 | hanhui1000002 | 华AZI419 | The driver overspeeds for 1 seconds. | 1/1/2017, 8:42:49 PM |
| 101441 | hanhui1000002 | 华AZI419 | The driver rapidly slows down. | 1/1/2017, 8:42:49 PM |
| 101440 | hanhui1000002 | 华AZI419 | The driver slides neutrally for 0 seconds. | 1/1/2017, 8:42:48 PM |
| 101439 | hanhui1000002 | 华AZI419 | The driver is fatigue driving. | 1/1/2017, 8:42:40 PM |
| 101436 | hanhui1000002 | 华AZI419 | The driver is fatigue driving. | 1/1/2017, 8:42:12 PM |
| 101435 | hanhui1000002 | 华AZI419 | The driver is at 河北省沧州市任丘市:距离终点133.406km. | 1/1/2017, 8:41:59 PM |
| 101434 | hanhui1000002 | 华AZI419 | The driver overspeeds for 23 seconds. | 1/1/2017, 8:41:59 PM |
| 101433 | hanhui1000002 | 华AZI419 | The driver is fatigue driving. | 1/1/2017, 8:41:35 PM |

Querying specific record with event ID:



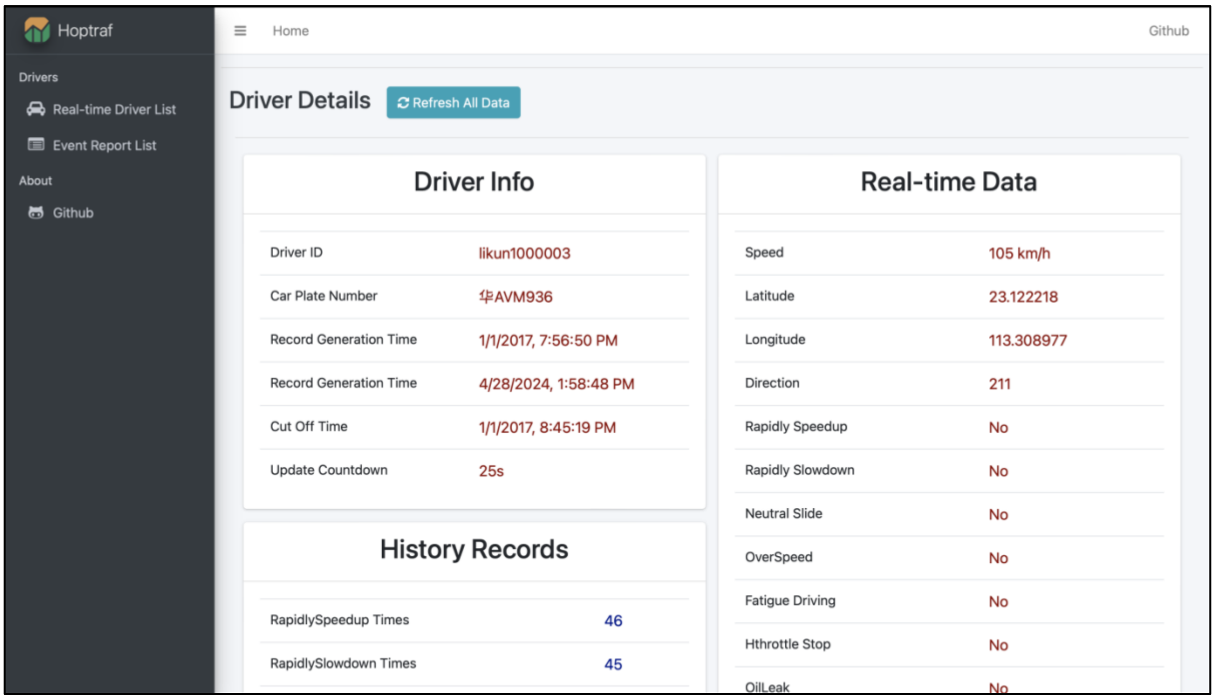
Event Report List

Last Update Time: 4/28/2024, 1:58:27 PM Cut Off Time: 1/1/2017, 8:44:58 PM Update Countdown: 19s

100382

| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 100382 | likun1000003 | 华AVM936 | The driver is fatigue driving. | 1/1/2017, 5:59:53 PM |

Real-Time detailed driver information:



Driver Details

Driver Info

| | |
|---|---|
| Driver ID | likun1000003 |
| Car Plate Number | 华AVM936 |
| Record Generation Time | 1/1/2017, 7:56:50 PM |
| Record Generation Time | 4/28/2024, 1:58:48 PM |
| Cut Off Time | 1/1/2017, 8:45:19 PM |
| Update Countdown | 25s |

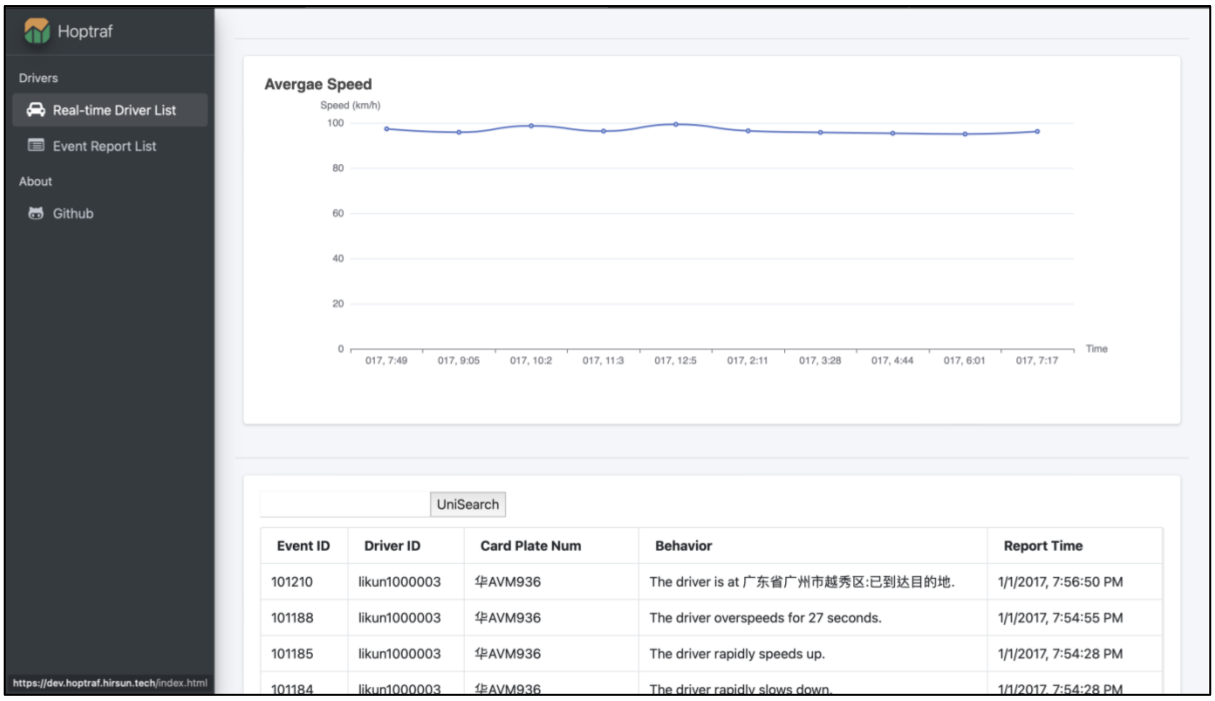
History Records

| | |
|---|---|
| RapidlySpeedup Times | 46 |
| RapidlySlowdown Times | 45 |

Real-time Data

| | |
|---|---|
| Speed | 105 km/h |
| Latitude | 23.122218 |
| Longitude | 113.308977 |
| Direction | 211 |
| Rapidly Speedup | No |
| Rapidly Slowdown | No |
| Neutral Slide | No |
| OverSpeed | No |
| Fatigue Driving | No |
| Hthrottle Stop | No |
| OilLeak | No |

Real-Time driving speed visualization diagram:

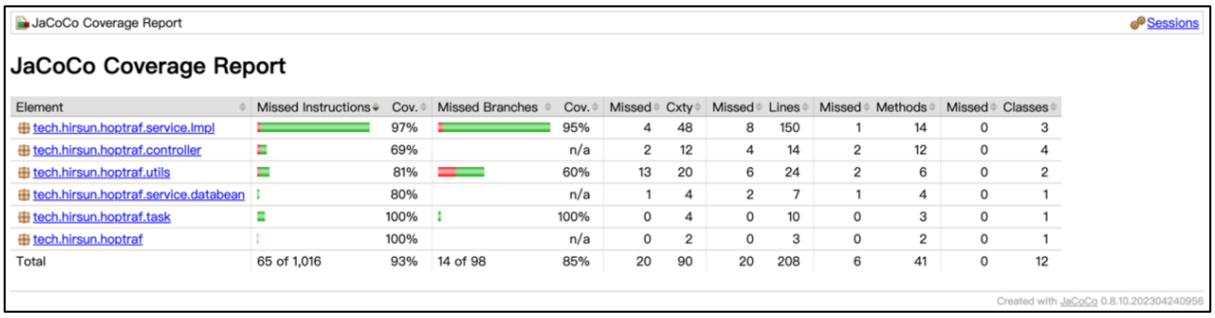


| Event ID | Driver ID | Card Plate Num | Behavior | Report Time |
|---|---|---|---|---|
| 101210 | likun1000003 | 华AVM936 | The driver is at 广东省广州市越秀区:已到达目的地. | 1/1/2017, 7:56:50 PM |
| 101188 | likun1000003 | 华AVM936 | The driver overspeeds for 27 seconds. | 1/1/2017, 7:54:55 PM |
| 101185 | likun1000003 | 华AVM936 | The driver rapidly speeds up. | 1/1/2017, 7:54:28 PM |
| 101184 | likun1000003 | 华AVM936 | The driver rapidly slows down. | 1/1/2017, 7:54:28 PM |

# 7 Testing

HopTraf takes the whole project forward with a software engineering process. The team members pay special attention to the high quality and availability of the software, which cannot be separated from software testing. As a result, the tool *JaCoCO* is leveraged to test the quality of the code to ensure that all code is covered and tested at runtime. Unlike unit testing, *JaCoCO* supports not only code unit testing, but also calculating global test code coverage as a proxy while the program is running, which helps to analyze the work of the whole project instead of one small unit.

As shown in the figure below, the project achieved 93% code coverage for the logical implementation, with 97% test coverage for the implementation classes. Developers can even clearly analyze which part of the code is covered by tests and which is not.

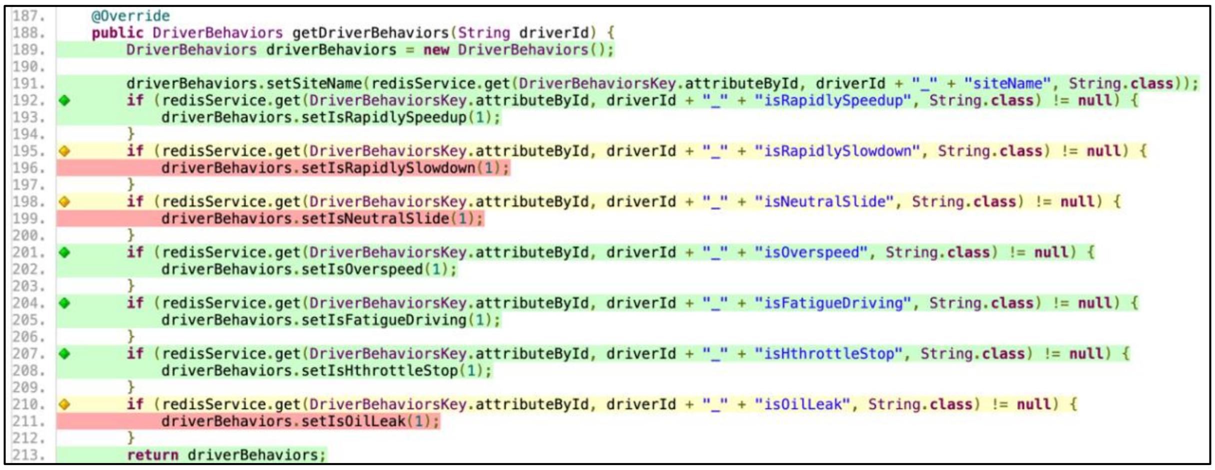


JaCoCo Coverage Report    Sessions

JaCoCo Coverage Report

| Element | Missed Instructions | Cov. | Missed Branches | Cov. | Missed | Cxty | Missed | Lines | Missed | Methods | Missed | Classes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| tech.hirsun.hoptraf.service.Impl | | 97% | | 95% | 4 | 48 | 8 | 150 | 1 | 14 | 0 | 3 |
| tech.hirsun.hoptraf.controller | | 69% | | n/a | 2 | 12 | 4 | 14 | 2 | 12 | 0 | 4 |
| tech.hirsun.hoptraf.utils | | 81% | | 60% | 13 | 20 | 6 | 24 | 2 | 6 | 0 | 2 |
| tech.hirsun.hoptraf.service.databean | | 80% | | n/a | 1 | 4 | 2 | 7 | 1 | 4 | 0 | 1 |
| tech.hirsun.hoptraf.task | | 100% | | 100% | 0 | 4 | 0 | 10 | 0 | 3 | 0 | 1 |
| tech.hirsun.hoptraf | | 100% | | n/a | 0 | 2 | 0 | 3 | 0 | 2 | 0 | 1 |
| Total | 65 of 1,016 | 93% | 14 of 98 | 85% | 20 | 90 | 20 | 208 | 6 | 41 | 0 | 12 |

Created with JaCoCo 0.8.10.202304240956



```java
    @Override
    public DriverBehaviors getDriverBehaviors(String driverId) {
        DriverBehaviors driverBehaviors = new DriverBehaviors();

        driverBehaviors.setSiteName(redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "siteName", String.class));
        if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isRapidlySpeedup", String.class) != null) {
            driverBehaviors.setIsRapidlySpeedup(1);
        }
        if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isRapidlySlowdown", String.class) != null) {
            driverBehaviors.setIsRapidlySlowdown(1);
        }
        if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isNeutralSlide", String.class) != null) {
            driverBehaviors.setIsNeutralSlide(1);
        }
        if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isOverspeed", String.class) != null) {
            driverBehaviors.setIsOverspeed(1);
        }
        if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isFatigueDriving", String.class) != null) {
            driverBehaviors.setIsFatigueDriving(1);
        }
        if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isHthrottleStop", String.class) != null) {
            driverBehaviors.setIsHthrottleStop(1);
        }
        if (redisService.get(DriverBehaviorsKey.attributeById, driverId + "_" + "isOilLeak", String.class) != null) {
            driverBehaviors.setIsOilLeak(1);
        }
        return driverBehaviors;
```

The test results have been made available for global access for public, at here.

# 8 Conclusion

The HopTraf web service provides a robust platform for real-time analysis of driving behavior data. It leverages various features such as data time simulation, real-time data analysis, driver statistics calculation, data query, and data visualization to enable insightful interactions and meet specified requirements.

The project utilizes a multi-layer architecture and asynchronous design to ensure efficient performance and scalability. It is built with the SpringBoot framework and Apache Spark, and is deployed on Amazon Web Services, taking advantage of services like AWS Amplify, Amazon EMR, ElastiCache for Redis, and RDS for MySQL.

The system has demonstrated high code coverage and successful operation in various use cases, making it a reliable tool for driving behavior analysis.

# Reference

1. SpringBoot: https://www.tutorialspoint.com/spring_boot/spring_boot_introduction.htm
2. Beans and DI: https://www.tutorialspoint.com/spring_boot/spring_boot_beans_and_dependency_injection.htm
3. IoC and DI: https://www.baeldung.com/inversion-control-and-dependency-injection-in-spring
4. Spark: https://spark.apache.org/
5. Spark: https://aws.amazon.com/what-is/apache-spark/#:~:text=Apache%20Spark%20is%20an%20open,against%20data%20of%20any%20size.
6. DAO: https://www.baeldung.com/java-dao-pattern
7. DAO: https://stackoverflow.com/questions/19154202/what-is-data-access-object-dao-in-java
8. Redis: https://www.baeldung.com/jedis-java-redis-client-library