**COMP4442 Service and Cloud Computing**

Group Project

|  |  |  |
| --- | --- | --- |
| **Name** | **Student ID** | **Email** |
| Cheung Sui Wing | 21027547D | 21027547D@connect.polyu.hk |
| Lau Man Chun | 21027257D | 21027257D@connect.polyu.hk |
| Kwong Chun Him | 21028468D | 21028468D@connect.polyu.hk |
| Cheng Chi Kit | 21028079D | 21028079D@connect.polyu.hk |

Table of Contents

[Group tasks performed by each member 1](#_Toc101709782)

[Cheung Sui Wing 1](#_Toc101709783)

[Lau Man Chun 1](#_Toc101709784)

[Kwong Chun Him 1](#_Toc101709785)

[Cheng Chi Kit 1](#_Toc101709786)

[AWS App link: 1](#_Toc101709787)

[Folder structure: 1](#_Toc101709788)

[System Specification 2](#_Toc101709789)

[Operation system 2](#_Toc101709790)

[Programming language with version 2](#_Toc101709791)

[System software 2](#_Toc101709792)

[Required packages 2](#_Toc101709793)

[Functional modules 3](#_Toc101709794)

[Key functions 3](#_Toc101709795)

[Summary Web Page 3](#_Toc101709796)

[Monitor Web Page 5](#_Toc101709797)

[Monitor Record Page 5](#_Toc101709798)

[Spark 5](#_Toc101709799)

[Upload data to database 6](#_Toc101709800)

[Generate data for real time monitor 6](#_Toc101709801)

[Relationship among them 6](#_Toc101709802)

[System architecture 7](#_Toc101709803)

[Deployment procedures 7](#_Toc101709804)

[Operations in AWS 7](#_Toc101709805)

[Scripts to be executed locally 10](#_Toc101709806)

[Testing reports for different functions 11](#_Toc101709807)

[Use cases 11](#_Toc101709808)

[Screenshots of testing results 12](#_Toc101709809)

[References: 20](#_Toc101709810)

# Group tasks performed by each member

## Cheung Sui Wing

* Spark script
* Database SQL, upload data script, initSQL
* Flask App: summary bar chart and pie chart
* Setup/deploy the app in AWS S3, EC2, RDS, EMR
* Report Document

## Lau Man Chun

* Summary table/ fetch SQL/ filter etc.
* Monitor record page, bug fix
* Doc

## Kwong Chun Him

* Monitor function and monitor page
* Monitor record page, record database/SQL
* Report Document

## Cheng Chi Kit

* Testing
* Deploy the app in AWS
* Report Document

# AWS App link:

<http://web-env.eba-qttvhmqs.us-east-1.elasticbeanstalk.com/>

# Folder structure:

Graphical user interface, text

Description automatically generated

# System Specification

## **Operation system**

AWS EC2:

* Python 3.8 running on 64bit Amazon Linux 2/3.3.12

In Local:

* Windows 10 / Google Colab

## **Programming language with version**

Python 3.8

MySQL 8.0.28

## **System software**

In AWS:

* AWS Spark,
* AWS EMR,
* AWS S3,
* AWSElastic BeanStalk,
* AWSElastic BeanStalk Loadbalancer

In Local:

* Visual Studio code
* Google colab
* MySQL workbench 8.0

## **Required packages**

certifi==2020.6.20

click==8.1.2

colorama==0.4.4

Flask==2.1.1

itsdangerous==2.1.2

Jinja2==3.1.1

MarkupSafe==2.1.1

mysql-connector==2.2.9

numpy==1.22.3

pandas==1.4.2

py4j==0.10.9.3

PyMySQL==1.0.2

pyspark==3.2.1

python-dateutil==2.8.2

pytz==2022.1

six==1.16.0

Werkzeug==2.1.1

wincertstore==0.2

# Functional modules

## **Key functions**

### **Summary Web Page**

The Summary page includes all the functions below, it is for viewing the driving behavior of the people in a selected amount of date and time or viewing a specific person’s driving behavior.

* **Summary data table**

Show the summary data according to the the selecting day in table form

Table

Description automatically generated

* **Date and time Selection**

Select data and time for data to be shown and for generating bar chart and pie chart.

Calendar

Description automatically generated

* **Filter**

Find driving behavior according to DriverID or Car Plate Number, etc. (only filter the data existing in the table)

Graphical user interface, text, application

Description automatically generated

* **Generating Bar Chart**

Generate a Bar Chart according to the data of each or every driver, which include abrupt acceleration times, abrupt brake times, overspeed times, etc.

Chart, bar chart

Description automatically generated

* **Generating Pie Chart**

Generate a Pie Chart according to the data of each or every driver, which include abrupt acceleration times, abrupt brake times, overspeed times, etc.

Chart, pie chart

Description automatically generated

### **Monitor Web Page**

* Can search the driver ID that you want to monitor.

Graphical user interface

Description automatically generated with low confidence

* Show the real time driving speed of this driver (update the data every 30 seconds), when the speed of driver high than 80 will alert “speed too fast” message.

Graphical user interface, application

Description automatically generated

### **Monitor Record Page**

Show the Monitor Record according to the selected driver and select time period (show the speed data from the dataset provided)

Graphical user interface, application

Description automatically generated

### **Spark**

* File for using in EMR: DB/Spark\_aws\_EMR.py
* File for using in development (Google Colab) : DB/ spark\_colab.ipynb
* We hard code the S3 input output path, no need to use sys.args.

Basic logic

* First, read all 10 text data
* Then, Filter out all the record that do no have any special behaviour

e.g.:

A screenshot of a computer

Description automatically generated with medium confidence

In our case, only count the line that have more than 8 records



* Then use driver, carPlateNumber, Time as a key to sum up all other behaviour primary in that day.
* Final output as a single csv file

*Reference: Huawai Driving Behavior Data Analysis [1]*

### **Upload data to database**

* Path: DB/upload\_data\_to\_DB.py
* Upload csv data generate by the spark function

### **Generate data for real time monitor**

* Path: DB/ gen\_data\_for\_monitor.py
* Generate random data for real time monior function

## **Relationship among them**

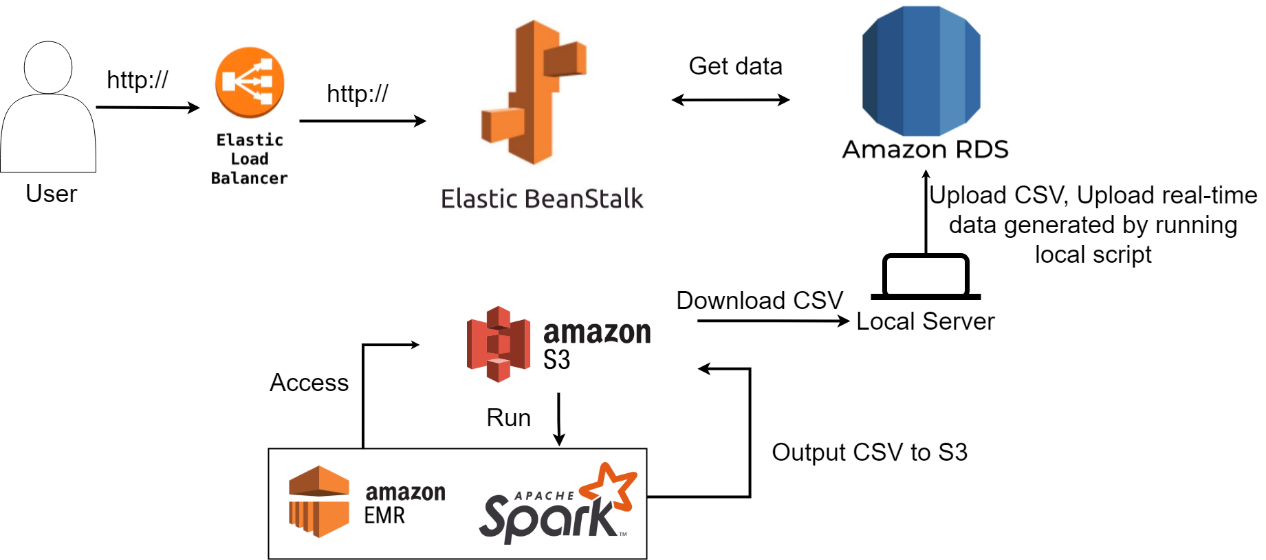
Summary and Moniotr are the fontend function

Summary includes functions that are Data and time selection, Filter, Generating Bar Chart, and Generating Pie Chart, they all are configed into a single page for easier browsing. The data used is come from spark processing.

The Monitor Web Page shows the real-time driving data generated by using the local script.

# System architecture

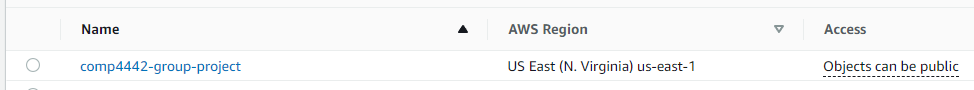
(System architecture to show how you orchestrate the AWS services.)



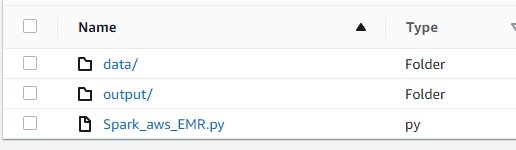
# Deployment procedures

## Operations in AWS

1. Create storage Buckets



1. Upload the python script and data to the bucket



**data/** storing the 10 orginal data

**output/** use to store the csv output after running the python script

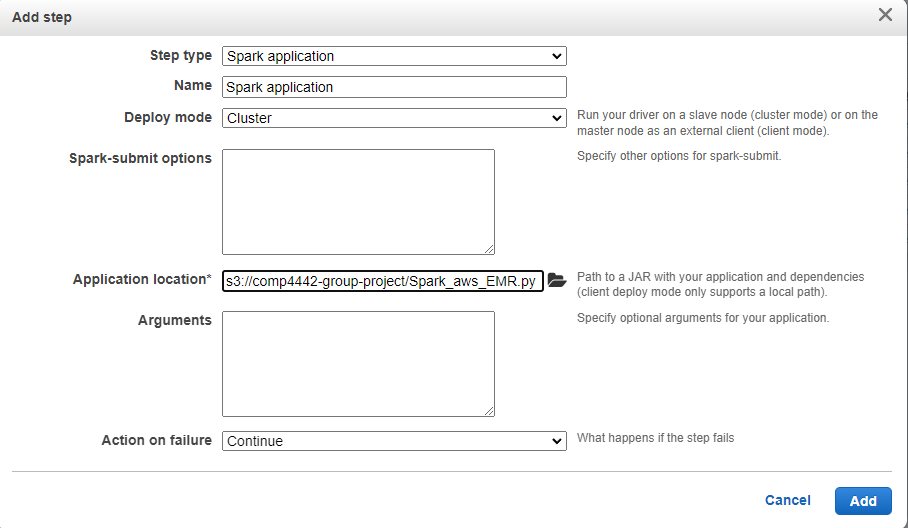
and the **python script**

1. Setup EMR

Graphical user interface, text, application

Description automatically generated

1. Add Step

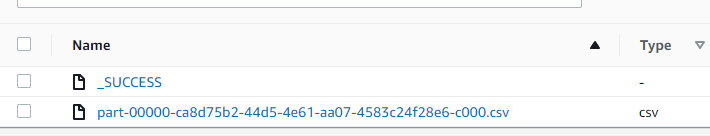


(hard code path, no args are needed)

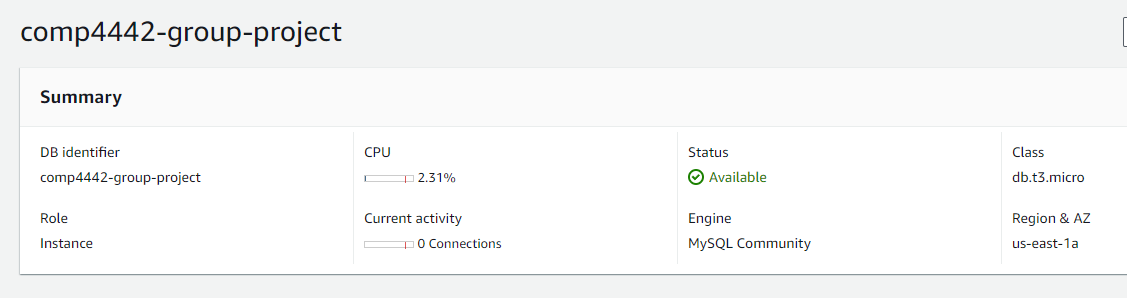
1. Wait for complete



1. Find the csv in S3://comp4442-group-project/output/



1. Download the csv and store in local
2. Set up AWS RDS database



1. Add groupmate’s IP and 0.0.0.0/0 to security Inbound rule

Graphical user interface, application

Description automatically generated

1. Setup MySQL workbench

Graphical user interface, text, application, chat or text message

Description automatically generated

1. Copy the create table query in DB/initDB.sql and execute it to create DrivingRecords and Monitor tables.

Graphical user interface

Description automatically generated

1. Run upload\_data\_to\_DB.py to upload the csv data to DB

python upload\_data\_to\_DB.py

1. Develop the Flask application locally

python application.py

1. Load balancer

Graphical user interface, application

Description automatically generated

1. Deploy the Flask app to AWS Elastic Beanstalk

Graphical user interface, text, application

Description automatically generated

url: <http://web-env.eba-qttvhmqs.us-east-1.elasticbeanstalk.com/>

Graphical user interface, application

Description automatically generated

## Scripts to be executed locally

* DB/ upload\_data\_to\_DB.py

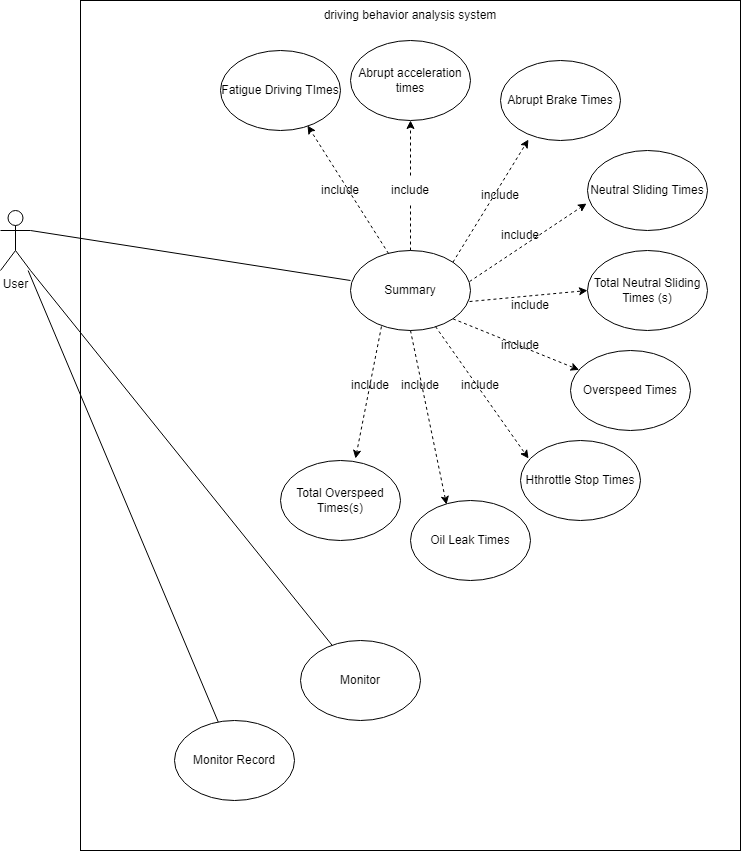
Use once to upload the csv data to database

* DB/ gen\_data\_for\_monitor.py

Run this script to generate fake data for real time monitor function

# Testing reports for different functions

## Use cases



## Screenshots of testing results

* Home Page

Graphical user interface, application

Description automatically generated

* Summary button

Graphical user interface, text, application, email

Description automatically generated

* Monitor button

Graphical user interface, application

Description automatically generated with medium confidence

* Summary-Date and time button

Calendar

Description automatically generated

* Summary-Choosing date and time



* Summary-Submit button

Table

Description automatically generated

* Summary-Filter

Table

Description automatically generated with medium confidence

* Summary-Abrupt acceleration times

Chart, bar chart

Description automatically generated

* Summary-Abrupt Brake Times

Chart, bar chart

Description automatically generated

* Summary-Neutral Sliding Times

Chart, bar chart

Description automatically generated

* Summary-Total Neutral Sliding Times (s)

Chart, bar chart

Description automatically generated

* Summary-Overspeed Times

Chart, bar chart

Description automatically generated

* Summary-Total Overspeed Times(s)

Chart, bar chart

Description automatically generated

* Summary-Fatigue Driving Times

Chart, bar chart

Description automatically generated

* Summary-Hthrottle Stop Times

Chart

Description automatically generated

* Summary-Oil Leak Times

Chart, bar chart

Description automatically generated

* Monitor-Submit

Graphical user interface, application

Description automatically generated

* Monitor-Over Speed

Graphical user interface, chart, application

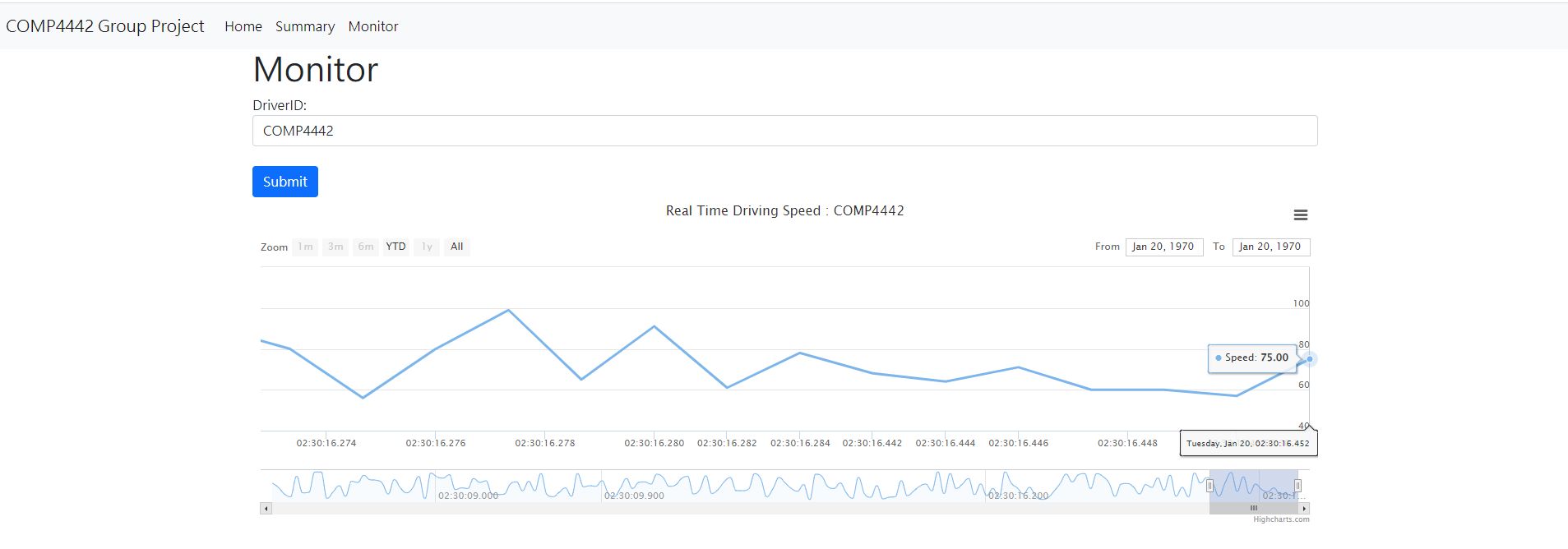
Description automatically generated

* The data of the chat should same with the database
* Data of database

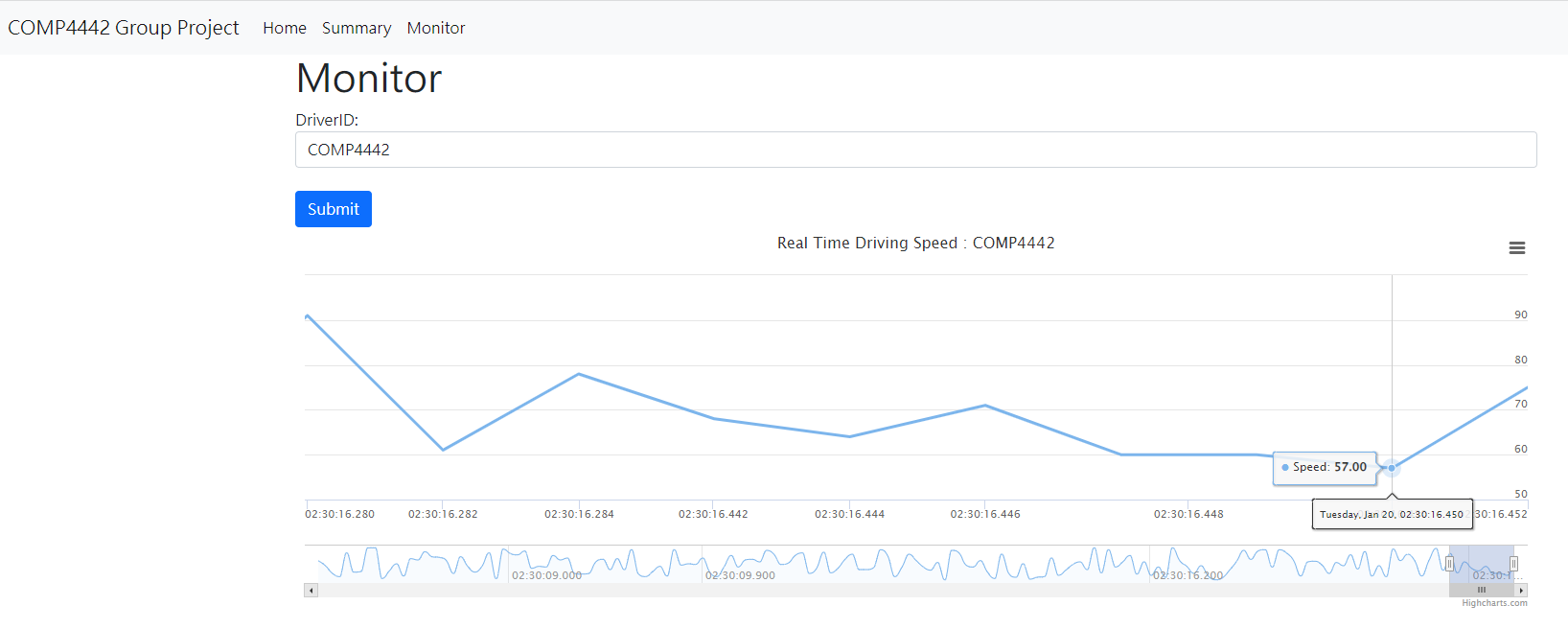
一張含有 文字 的圖片

自動產生的描述

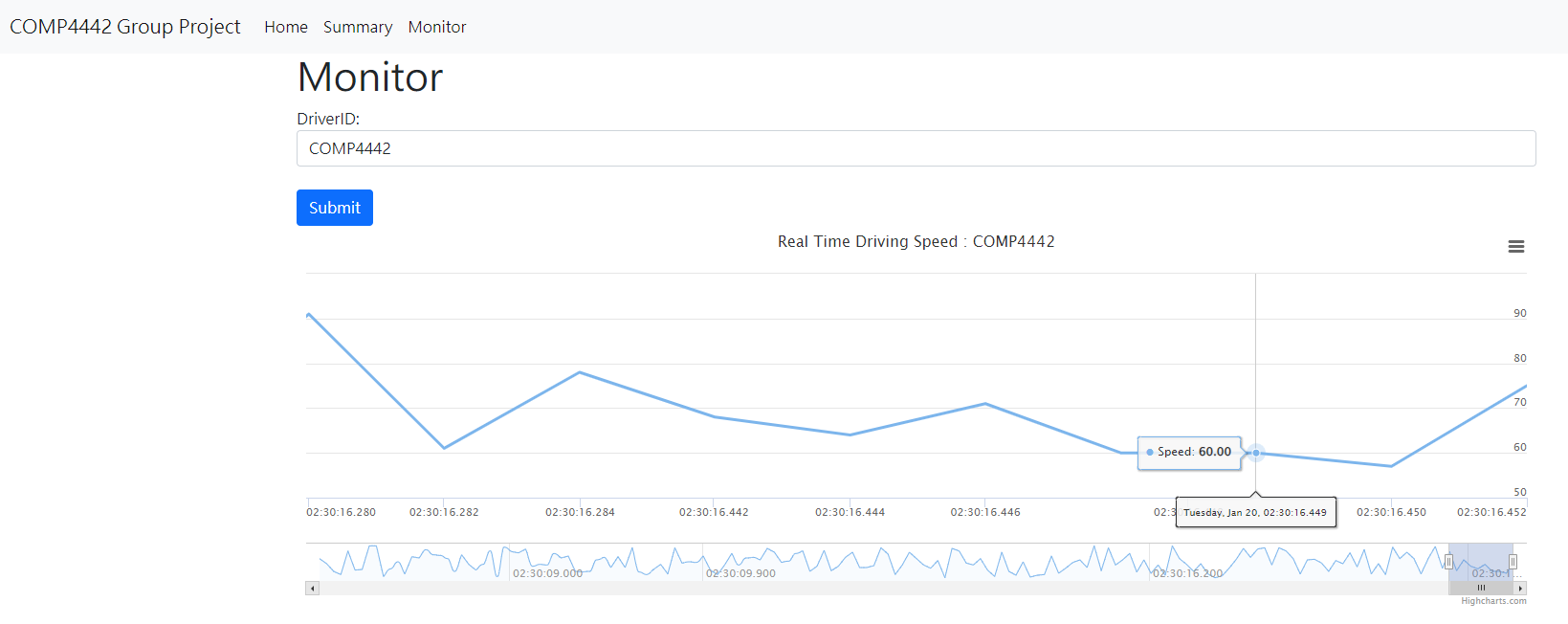
* Simple 1 : The last data of chart



* Simple 2 : The second to last data of chart



* Simple 3 : The third to last data of chart



* Result: all the data of the chat is same with the data base
* Monitor Record

Graphical user interface, application

Description automatically generated

* 5M

Graphical user interface, chart, application

Description automatically generated

* 1M

Graphical user interface

Description automatically generated

# References:

[1] Haiwai, Driving Behavior Data Analysis, Apr 01, 2022, <https://support.huaweicloud.com/intl/en-us/bestpractice-dli/dli_05_0001.html>