# Instruction

# **EV Charger Dashboard Project**

This project consists of three parts:

- 1. evcharger-dashboard-backend: This contains the backend components.
- 2. evcharger-dashboard-frontend: This contains the frontend components.
- 3. evcharger-availability-prediction: This contains the machine learning components.

## Setup Instructions

#### Step 1: Import the Database

1. Create the database "db\_evcharger\_0731" with command

```
Plain Text

1 CREATE DATABASE db_evcharger_0731;
```

- 2. Locate the file export\_total.sql in the submitted project folder.
- 3. Use the console to run the following command to import the database into your local machine:

```
Plain Text

1 mysql -u your_username -p db_evcharger_0731 < export_total.sql
```

Replace your\_username with your MySQL username.

#### Step 2: Update Machine Learning Script Database Configuration

- 1. Navigate to the evcharger-availability-prediction directory.
- 2. Open the file data\_cleaning\_and\_model\_training.py .
- 3. Update the database configuration information at the beginning of the script to match your local database settings.

```
from sqlalchemy import create_engine
                        import pandas as pd
                              from datetime import datetime
                               import urllib.parse
                               # Database connection information
    7
                            db config = {
                                                'host': 'localhost'
   8
                                                'user': 'root',
   9
                                                'password': '123456',
10
                                                'database': 'db_evcharger_0731'
11
12
13
                               # Create the database URL (URL-encode the password to handle special characters)
                              \label{locality} $$ db_url = f'''mysql+pymysql:/{db_config['user']}: \{urllib.parse.quote_plus(db_config['password'])\} @\{db_config['hoselet all black of the password']\} and the password of 
15
16
                              # Create the SQLAlchemy engine
17
18
                              engine = create engine(db_url)
19
20
                               # Updated SQL query, including weather information
21 V query = """
                                                                                                                                                                                                                                                                                                                                                                                                         🗲 中 🕠 🎍 🗏 😭 🕿 🖁
```

### Step 3: Install Dependencies for Machine Learning Module

- 1. In the evcharger-availability-prediction directory, locate the requirements.tx t file.
- 2. Install the required dependencies using the following command:

```
Plain Text

pip install -r requirements.txt
```

## Step 4: Run the Machine Learning Script

1. In the evcharger-availability-prediction directory, run the machine learning script:

```
python data_cleaning_and_model_training.py
```

Depending on your Python version, you may need to use <a href="python">python</a> instead of <a href="python">python</a>.

### **Step 5: Update Backend Configuration**

1. Navigate to the backend directory evcharger-dashboard-backend\src\main\resource s.

- 2. Open the file application properties.
- 3. Update the database configuration information to match your local database settings.

```
spring.application.name=EV Charger Utilisation Dashboard
spring.datasource.type=com.alibaba.druid.pool.DruidDataSource
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/db_evcharger_0722?useSSL=false
spring.datasource.username=root
spring.datasource.password=123456
mybatis-plus.configuration.log-impl=org.apache.ibatis.logging.stdout.StdOutImpl
mybatis-plus.configuration.use-deprecated-executor=false
mybatis-plus.mapper-locations=classpath:/mapper/*.xml
server.port=8088
```

### Step 6: Update Model Path in Backend

- 1. In the same directory, open prediction.py.
- 2. Modify the base\_directory in the model loading path to reflect the absolute path of
  the evcharger-availability-prediction directory on your local machine./

```
For Windows:
base_directory = r"<your_path>\evcharger-availability-prediction"

For Linux:
base_directory = "<your_path>/evcharger-availability-prediction"

For MacOS:
base_directory = "<your_path>/evcharger-availability-prediction"
```

```
| Project | Cardinoprocessive | P man | StationController Feature | P man | StationController Feature | P man | StationController Feature | P man | P
```

## Step 7: Adjust Python Command in Backend Service

- Navigate to evcharger-dashboard-backend\src\main\java\com\evcharger\dashboard-backend\src\main\java\langle\lan
- 2. Open PredictionServiceImpl.java.
- 3. Update the Python command (python or python3) based on your local Python version.

```
    ■ E evcharger-dashboard-backend ∨ ♀ main ∨

                                                                                                                                                                         a x Q Ø
                                                                                                                           StationControllerTest > D & :
                                                            lication.properties
                         ① ConnectorMapper
                                                                                         JsonNode lastDayWeather = list.get(list.size() - 1);
Ġ
                        ③ SiteMapper
                                                                                         weather = lastDayWeather.path( s: "weather").get(0).path( s: "main").asText();
                        ① StationMapper
ĮΊ

∨ 
iii service

                                                                                } catch (Exception e) {
                      Igmi 📵 🗸
86
                           @ RouteServiceImpl
                           © SiteServiceImpl
                           C StationServiceImpl

    AvailabilityService

                                                                                                  "--coordinatesX", String.valueOf(connector.getCoordinatesX()),
"--coordinatesY", String.valueOf(connector.getCoordinatesY()),
                        ① RouteService
                         SiteService

    StationService

                      @ EvChargerUtilisationDashboardApplica
                                                                                                   '--connectorTypeRapid", String.vglueOf(connector.getConnectorTypeRapid()),
                                                                                                     -connectorTypeUltraRapid", String.valueOf(connector.getConnectorTypeUltraRapid()),
Ð

    com.evcharger.dashboard

                       AvailabilityControllerTest
                                                                                                   --stationConnectorCount", String.valueOf(connector.getStationConnectorCount()),
```

#### Step 8: Start the Backend Server

1. In the evcharger-dashboard-backend directory, start the backend server using the following command:

```
plain Text

mvn clean install
mvn spring-boot:run
```

### Step 9: Start the Frontend Server

- 1. Navigate to the evcharger-dashboard-frontend directory.
- 2. Install the necessary dependencies and start the frontend server using the following commands:

```
Plain Text

1 npm install
2 npm run serve
```

# Step 10: Access the System

Open your browser and go to <a href="http://localhost:8081">http://localhost:8081</a> to access the EV Charger Dashboard system.

# Step 11: Detailed Result

Detailed Results of prediction and analysis is in the evcharger-availability-prediction folder. They are shown in evcharger-prediction.pdf and analysis.pdf