Assignment 2 – Simulation of Electric cars and Charging points.

UC(AR)Y is a pioneer automotive company headquartered in Nicosia, Cyprus. UC(AR)Y started from the laboratories of the University of Cyprus and has gradually developed commercially available electronic cars. The brand is now considered one of the biggest competitors of Tesla!

To make the next step the company is recruiting you to design and implement an information system that will meet the requirements explained below.

Problem

This information system in general, needs to contain the information regarding the company's developed *e-cars* and the *e-charger points* that the company will install. Also, they want the system to provide the ability to find a charger point. The system should also include a database where all the data will be stored and retrieved when needed. The data that will be imported in the system are available at blackboard.

Part 1 – Create a database management class

Create a new class **Database** that implements the **DatabaseInterface**. In this class you need to implement the following:

1. Implement the method **connectToDatabase** which creates a connection between the program and the database by using the **jdbc library.** Create an account to **https://www.freesqldatabase.com/** and when you successfully login you will be able to create a database. The name of the database and all the other credentials you will need will be

send to your email. These credentials will be used for creating the connection with the jdbc library but also for logging into the phpMyAdmin page that is a database administration page.

2. Create programmatically the following tables:
   Charging_Point: point_id(Primary key), number_of_outlets, available_outlets, city, address.
   E_Car: model_id(Primary key), model, charging_speed, releate_date,
   Customer: customer_id(Primary key), full_name, car_registration_number, model_id(Foreign key), car_color, car_battery_level, car_last_service, total_millimetres .
   Charging_Process: process_id(Partial Key), customer_id (Foreign key), point_id(Foreign key), starting_timestamp, fully_charge_timestamp.
   In this table the Primary key should consist from : process_id, customer_id and point_id.
3. Use the files that were given to you to add the data to your tables. The files are e-cars.csv, chargePoints.csv, customer.csv, chargingProcess.csv.

**We assume that:*

1. *Each customer can have **only one car.***
2. *Each customer that has an account owns a car.*

Part 2 – Implement the Java Classes

For the purposes of the system, you will need to:

1. Create a **Customer** class with fields based on the database attributes. This class should be able to receive the user's details and the car's details and save them in the database. Keep the car's details in an ElectricCar object *(see below)* as well.
2. Create a class **Vehicle** that will implement the interface **<<*VehicleInterface*>>** that was given in blackboard. Also, this class should include a constructor that initializes each global variable.
3. Create a class **Car** that inherits the class **Vehicle**. This class should be implement the following methods:
   a. A method that calculates the total millimetres that a car has travelled.

   b. A method that updates a global variable that contains the date of the last service of the car.

4. Create a class **ElectricCar** which it inherits the class, Car. This class should also include the following methods:
   a. A method that returns the current battery balance of the car.
5. Create a class **EChargingPoint**. This class should implement the following methods:
   a. **availableOutlets()** that returns the available outlets on this point.
   b. **startCharging(Customer customer, int pointID)**
      i. Prints the beginning of charging
      ii. Updates the attribute AvailableOutlets in the database for the specific charging point.
      iii. Inserts a new record into the table **ChargingProcess.** The field fully_charge_timestamp should remain null (it will be updated after the charging process is finished).
   c. **finishCharging(Customer customer)**
      i. Checks that the customer has indeed a charging car at the moment and prints the right message.
      ii. Finds the record from the table **ChargingProcess** that has null value in the fully_charge_timestamp field and set the value to the current datetime.
      iii. Updates the available outlets for this charging point in the database.
6. Create a **ECarsCompany class** that should contain:
   a. A method that prints all the car models that the company has developed with all the details. The cars must be in alphabetic order based on the model's name.
   b. A method that prints the number of charging points in each city.
   c. The method **availableEChargerPoints(String city, boolean availableOutlets)** that finds the available charging points. The method should take as an input a city a flag named availableOutlets. If the flag is false, then print the available charging points in that city (including charging points with no available outlets at the moment), else print **only** the points that have at least one free outlet.
   d. In this class include the main method that take cares of all the user interaction functionality.

Part 4 – User Interaction

1. Print a welcoming message to the console that prints the name of the company, the number of the different models that it owns and the number of charging stations in each city.
2. Then show a prompt with the main menu:
   1. Show more info about the company.
   2. Find a charging point
   3. Login
   4. Register
   5. Exit

   When the user makes the selection, you need to do the following:

   1. Show more info about the company.:
   a. Show the full catalogue of the company's cars with the details of each car in alphabetic order based on the car's model.
   b. Show the number of charger stations in each city.
   c. Show the main menu again.

   2. Find a charging point
   a. You need to prompt to the user to write the city of his/her interest. Then you need to print the charging points in this city and the available outlets of each point. If the given city doesn't exist in the system show an error message and ask for the city again.
   b. The user should also be able to select the option "Show only the charger points with available outlets".
   c. After showing the results show the main menu

   3. Login
   a. Ask the user to enter his/her ID.
   b. If the user does not exist, you need to show an error message and the main menu.
   c. Otherwise show the **logged in submenu**:
      a. Show car status
      b. Start charging
      c. Complete charging process
      d. Logout

   Based on the user's selection you need to do the following:

Show car status

a. Shows the user's car's battery level, total millimeters, and the last service date.
b. Then show again the **logged in submenu.**

Start charging

a. The user needs to enter the charging's point id.
b. Then you need to check there is not on-going charging process for this user.
c. Check there is indeed empty outlet in the station and then start the charging by calling the proper method from the specific EChargerStation object.

Complete charging process

a. The user receives an informative message about the charging process.
b. Then the **login submenu** is shown again.

Logout

Navigate back to the **main** menu.

4. Register

a. Ask the user to give his/her ID number
b. Check that the user doesn't already exist. In case he/she does exist, show a message saying that the user exists and show the main menu as well.
c. Otherwise ask for the user's full name and the car's model. You need to check that the model provided exists. If the car doesn't exist show an error message and ask the user to re-enter the model.
d. Otherwise ask the rest details about the car.
e. Save the user & the car's details in the database and show the main menu.

5. Exit

a. Terminate the program

Part 4 – Test

Run your program with the following scenarios and make sure that every user can accomplish his/her goal by showing the correct error messages! There are cases that the users insert wrong data, it is up to the program to handle these cases.
The initial data of your database should be the one given in the files. Create a pdf with the screenshots of the scenarios.

Scenario 1:
A user with an id 885566 is trying to login into the system. After successfully logging in he selects "Show car status" to see his car details and then he starts charging his car at the point with id *524*. Finally, he logs out of the system.

Scenario 2:
Helen Smith selects "*Find a charging point* "in the main menu. She wants to see only the points with available outlets in Papho, but when she is asked to write the city, she writes "papho". When she manages to see the points in Papho, she proceeds to register with an id 222222 and a car model "34 MPG". After registration she logs in and selects "Start Charging" in the point with id *525* and then logs out.

Scenario 3
Mike selects "Find charging point" from the main menu and insert the city Papho as well, but he wants to see only the points with available outlets. Then logs out.

Scenario 4
Helen Smith with an id 222222 logs in again this time to Complete the charging process and logout.

Scenario 5
Mike wants to see if the charging point near to him with an id *525* has now available outlets.

Part 5 – Entity Relationship Diagram

Draw the ER diagram for the specified database.

Submission

Compress your project folder into a zip file and upload it into Assignment 2 on Blackboard. Create a PDF file that will include the screenshots from the scenarios and the ER diagram.

Marking

Database implementation	**3**
Java Classes	**3**
User Interaction	**2**
Test	**1.5**
ERD	**0.5**