***Project: Part 01***

# Case Study

# Implementing a 3-Tiers Application Vehicles’ Factory

A factory of vehicles manufactures different types of vehicles.

A vehicle could run on gasoline or electric batteries. Therefore, we have two different types of vehicles.

Each type would have its own ***fuel-efficiency measure*.**

For gasoline vehicles, it is the ***gasoline consumed per kilometer***, and for electric vehicles it is the ***kilowatts of power consumed per kilometer.***

Because, this is the common functionality and must be implemented by every type of car, including those that will come on the market in the future such as hybrid vehicles, you have to create a standard interface that every car will implement. You will call this interface *IMileageEfficiency. Any vehicle that implements this interface will get a standard set of methods for obtaining the vehicle’s efficiency.*

*You will define this interface with a method, as follows:   
 float getMilesPerGallon ( )*

*Therefore, the class Vehicle will implement this interface, and the Gasoline Vehicle and Electric Vehicle classes will provide an appropriate implementation for its sole method <<getMilesPerGallon>>*

Applying the polymorphism:  
The two extended classes has 2 common behaviors:  
1- make trip ( where the trip counter has to start and the fuel will be consumed) is following:   
 public void makeTrip ( )  
2- get miles per gallon has following implementation:  
 The Gasoline Vehicle class named **GasVehicle** must implements the 2 methods:  
 - *makeTrip’s method implementation* of the class **GasVehicle** must initialize the 2 attributes to their   
 initial values: : (for example: *trip counter* to 100 and *fuel consumed* to 7.77f)  
 - *getMilesPerGallon’s method implementation* of the class GasVehicle must return:  
 tripCounter / fuelConsumed

The Electric Vehicle class named **ElectricVehicle** must implement the 2 methods:   
 - *makeTrip’s method implementation* of the class ElectricVehicle must initialize the 2 attributs to their   
 initial values: (for example: trip counter to 100 and KW power consumed to 4.44f).  
 - getMilesPerGallon’s method implementation of the class ElectricVehicle must return:   
 tripCounter / KwPowerConsumed

A ***vehicle*** is identified by a serial number, made, model and trip counter, where trip counter is a protected attribute (it means: trip counter will be accessible by inherited classes)

A ***gasoline vehicle*** is identified by a *fuel consumed***,** while an ***electric vehicle*** is identified by a *kilowatts power consumed*

.

**I-** You have to create the ***business layer*:** **(package bus)**

1.1 Design the class diagram and add the relationship between classes.

1.2 Create the necessary enumerations, validation class and, all classes governing the system that contains:

* relevant attributs,
* get / set accessors,
* overloaded constructors,
* overridden method that prints *the* state *of the object*
* other necessary services

1.3 Create a validation class named *Validator* ensuring the validity of data.   
  
1.4 Create ***the interface*** between the **client side** and the **business side.**   
This interface named *IMileageEfficiency* must be implemented by the Vehicle base class.   
The gasoline vehicle and the electric vehicle classes must add the implementation to the *getMilePerGallon* service.

II- You have to create the ***presentation layer*:** (package client)  
  
To effectively manage the stock of vehicles, you must implement the following tasks:

1. Adding a new vehicle into a list of vehicles.
2. Searching for a vehicle by serial number.
3. Removing a specific vehicle from the list of vehicles.
4. Listing vehicles, listing gasoline vehicles, listing electric vehicles

Requirements to meet:

1. Data abstraction
2. Encapsulation
3. Inheritance
4. Polymorphism