

Advanced Programming Workshop #2

Sergio Nicolás Mendivelso

V 2.0

This program create various types of vehicles and add them to a list. It have classes for Engine, Vehicle ,Car, Truck, Yacht, and Motorcycle. The program will not manage MVC model anymore, it will be like a monolith system.

Bussines Model:

In this type of platforms, the users can see a catalog of the vehicles. So, exists the designers who creates the vehicles an upload it to the platform. This can be usefull to sell their products.

Business rules:

User Stories:

As User (Designer)I want to create an account to log in and see the created vehicles.

As Designer I want to create a various types of vehicles to have a list of them and upload it

As Designer I want to watch the information of my created vehicles list

As User I want to watch another created vehicles in a catalog

Entities:

User

Designer

Vehicles

Catalog

Account

CRC cards:

Vehicle	
Responsability: Show it owns attributes calculate gas consumption	Collaborators: Engine Designer

Engine	
Responsability: Show it owns attributes is added to a vehicle	Collaborators: Vehicle Designer

User	
Responsability: Register and login watch vehicles catalog	Collaborators: Catalog DataBase

Designer	
Responsability: Create Vehicles and engines Upload their new vehicles	Collaborators: Vehicle Engine Catalog

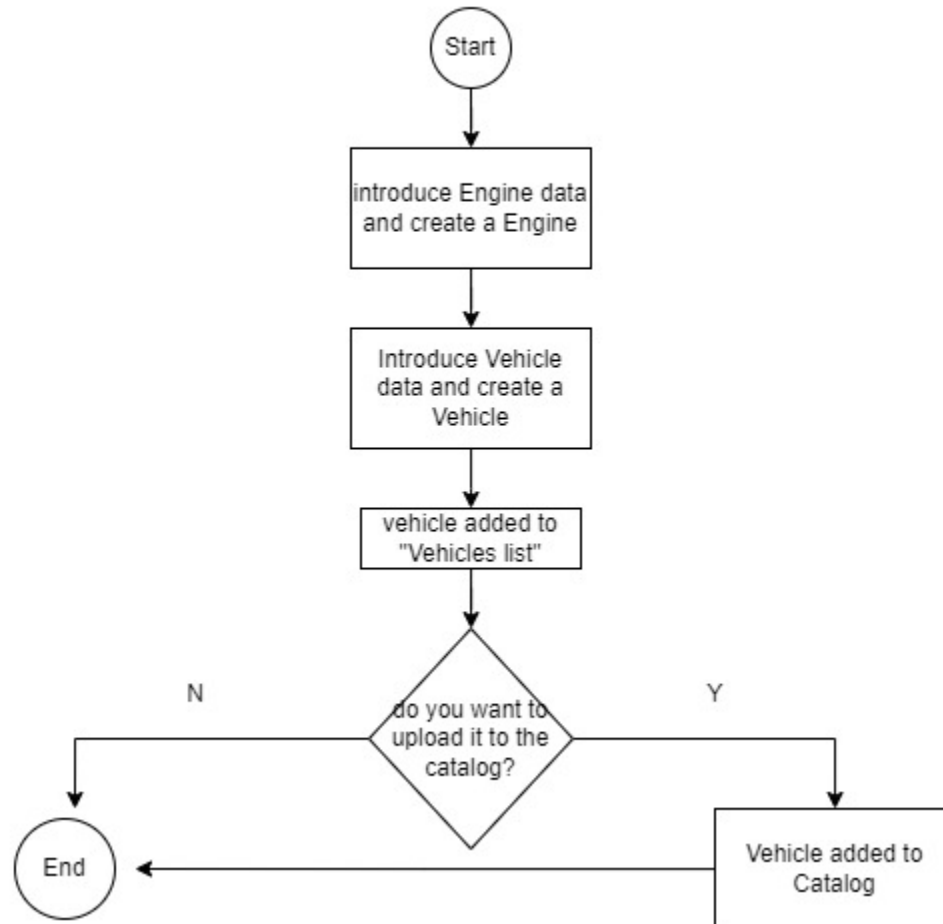
Data_base	
Responsability: Save users information return access verification	Collaborators: User

Catalog	
Responsability: Show created vehicles list Update created vehicles list adding new vehicles	Collaborators: User Designer

Note: These are the enough CRC Cards, because the yacht, car, motorcycle and truck classes are just concrete versions of vehicle.

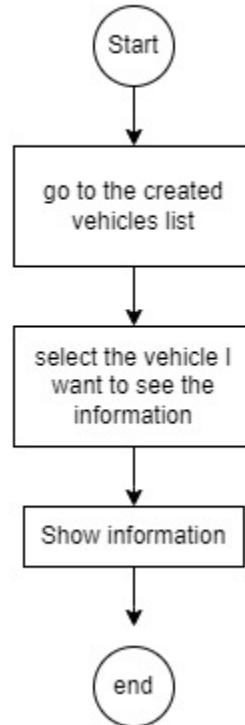
Activity Diagrams:

As Designer I want to create a various types of vehicles to have a list of them and upload it



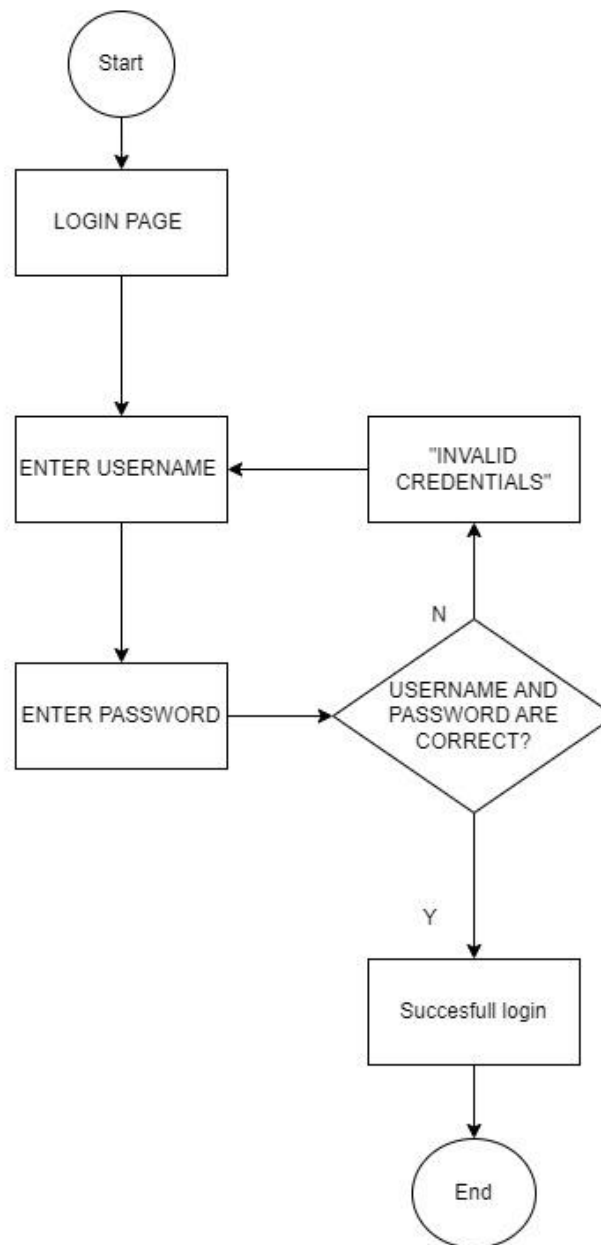
This diagram shows how to create a vehicle being a Designer, and how it can be upload to the catalog.

As Designer I want to watch the information of my created vehicles list



This diagram shows how to watch the created vehicles list for the Designers.

As User I want to create an account to log in and see the created vehicles.

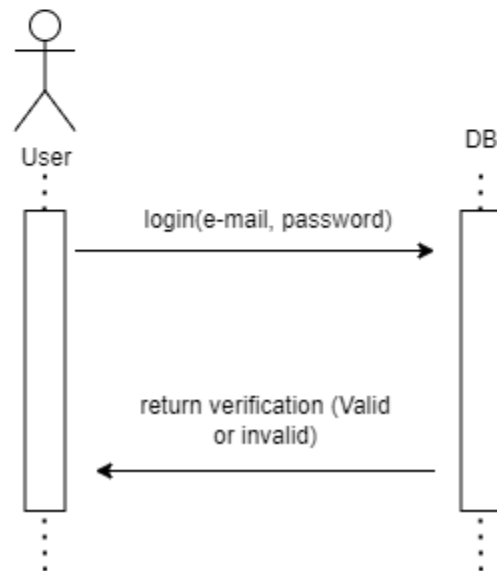
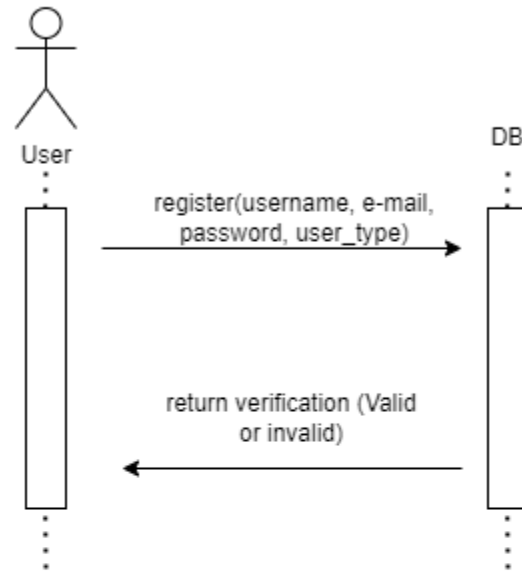


This diagram shows how log int to the system.

Note: The user Story “As User I want to watch another created vehicles in a catalog” Has not an activity diagramm, because it is a obvious and short process.

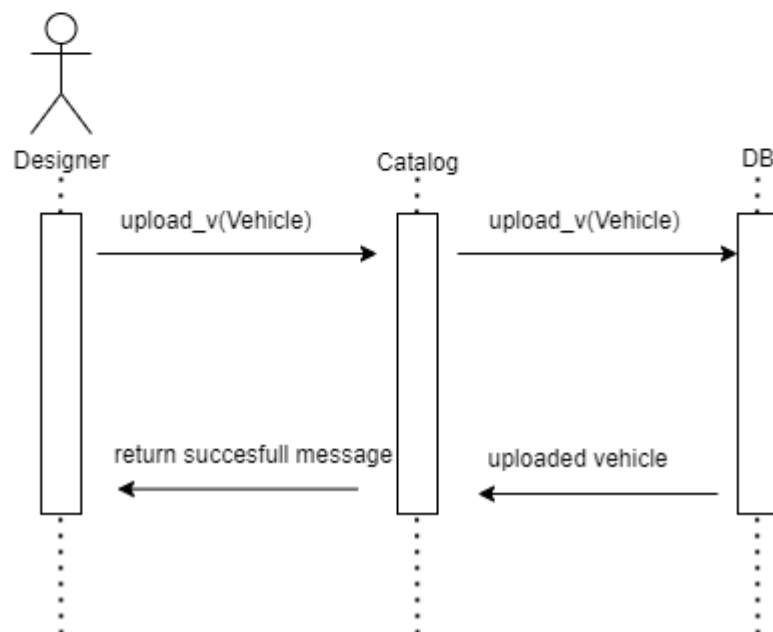
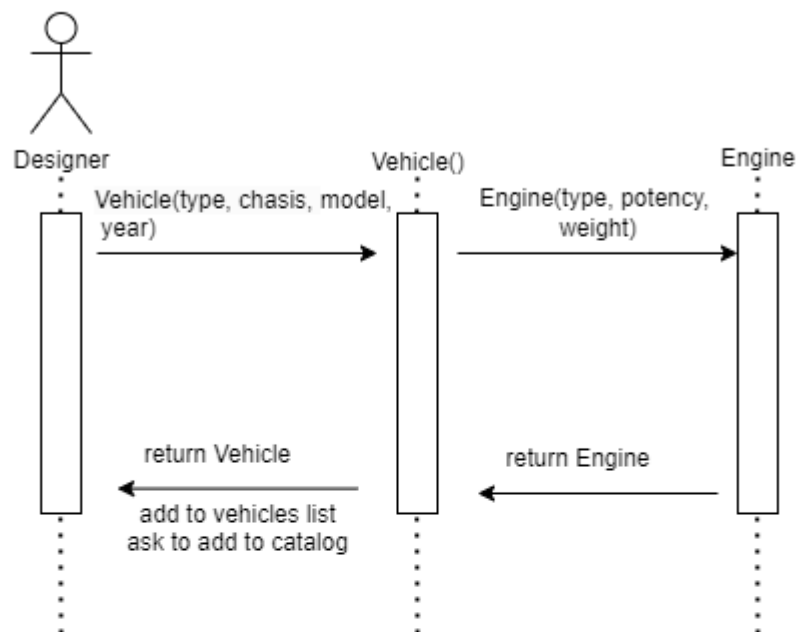
Sequence Diagrams:

As User I want to create an account to log in and see the created vehicles.



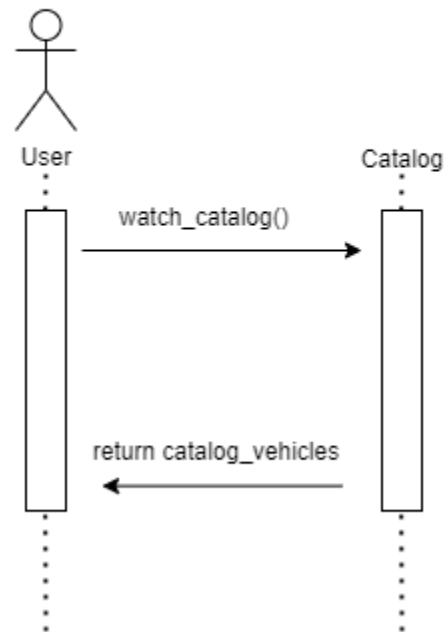
These diagrams shows how an user object and a DataBase object interact to obtain the verification of the register or the login.

As Designer I want to create a various types of vehicles to have a list of them and upload it



These diagrams shows how a designer object create a vehicle, so it has to create an engine too, also shows how it can upload a vehicle to the catalog and to the DataBase object.

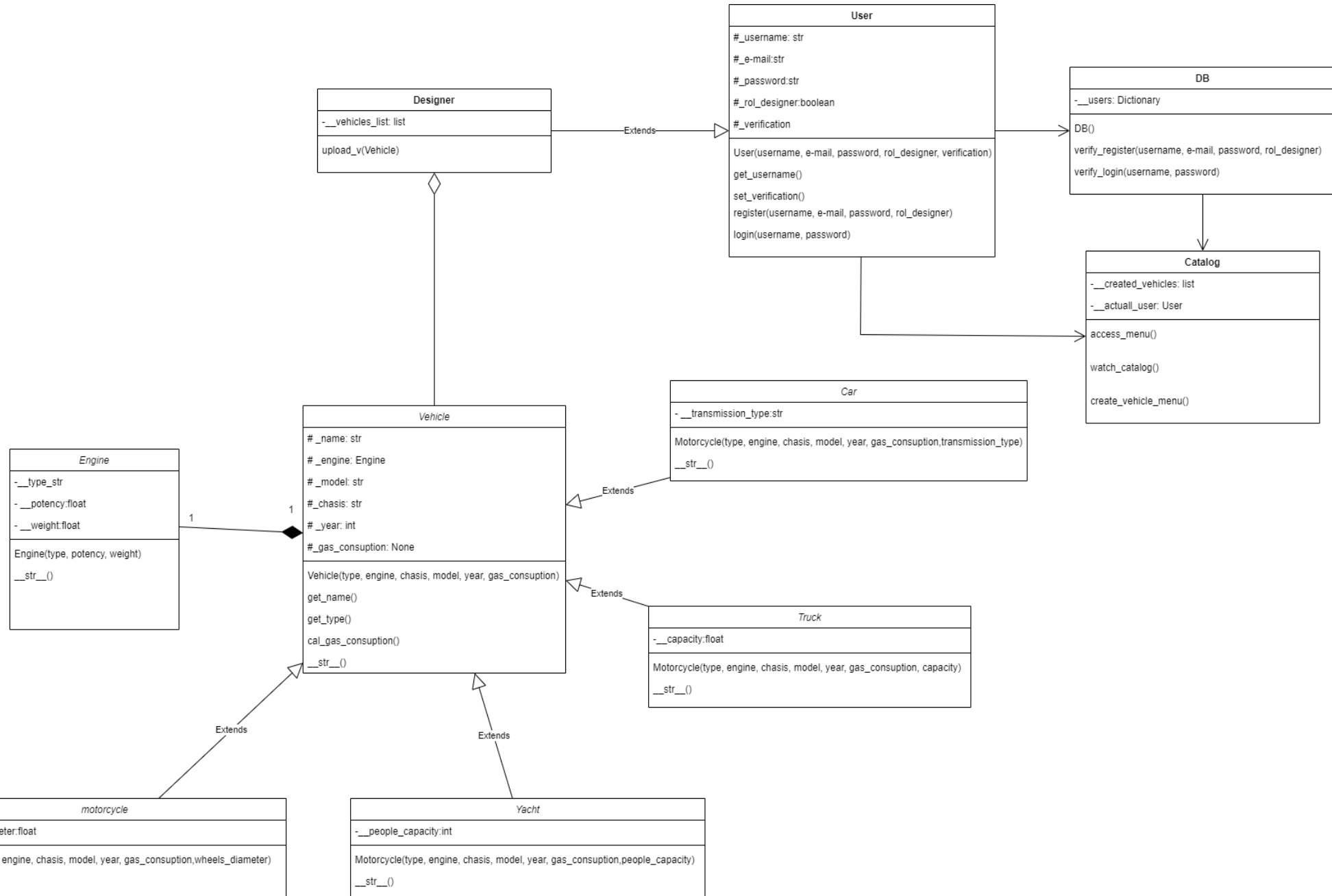
As User I want to watch another created vehicles in a catalog



This diagram shows how a User can watch the catalog and its information.

Note: The User story “As Designer I want to watch the information of my created vehicles list” has not an object interaction. So it has not a sequence diagram.

Class Diagram New Version:



New changes:

1. No more MVC model. Eliminated classes: Main_controller, View, Launcher
2. New Classes: User, Designer, DB and Catalog
3. New module separation: vehicles and users.
4. New Relations, aggregation, association, composition.
5. Now vehicle has a name attribute
6. No more get-Setter methods, just one that method that return all the attributes of the vehicles, and the engine (`__str__()` method). The method to get the vehicle type keeps.
7. `cal_gas_consumption()` method will pass from Main class to Engine class

General Description:

- The first change I did, were the vehicles classes. I create an `__str__()` method in all the vehicle classes and also add the name attribute in Vehicle class
- I use a composition relation in the Engine class to vehicle class.
- I change the `cal_gas_consumption()` method from the controller to the Engine class, I think to add it into the Vehicle class, but the method has more engine attributes than vehicle attributes, so the method is shorter and readable in the Engine class.
- After that, I create the Users module.
- I created the "users" file composed by User and Designer class. I created all its methods and attributes in both classes
- Login have the mail and password parameters.