Advanced Programming Workshop #3

Sergio Nicolás Mendivelso

V 3.0

I feel good with the backend process, the user stories and the UML diagrams, so, it won't change anymore, in this new version of the programm, I will create an graphic interface for the programm.

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:

Login an register: The user has to register to access the information to keep the security

Validation: The programm has a way to verify the data when somebody is registered or a vehicle is created (The year of creation has to be a number.. Etc).

Vehicles: A designer is the one how can create a vehicle.

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

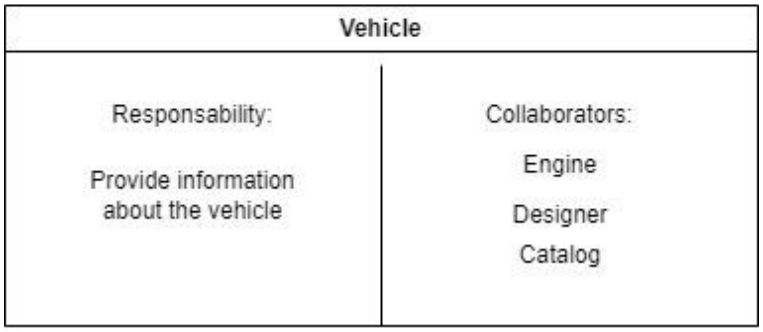
Designer

Vehicles

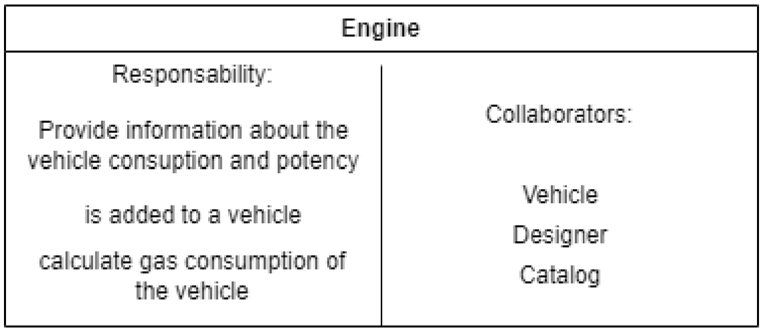
Catalog

Account

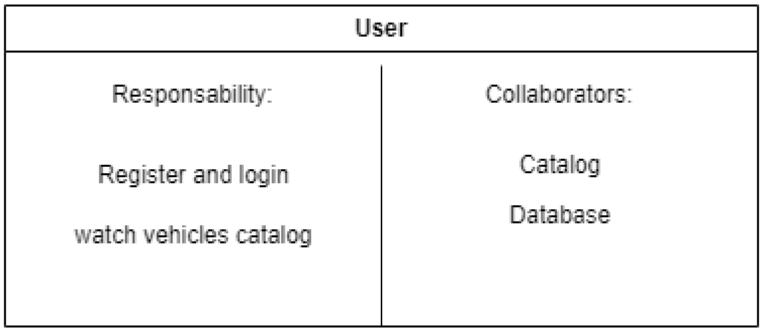
CRC cards:



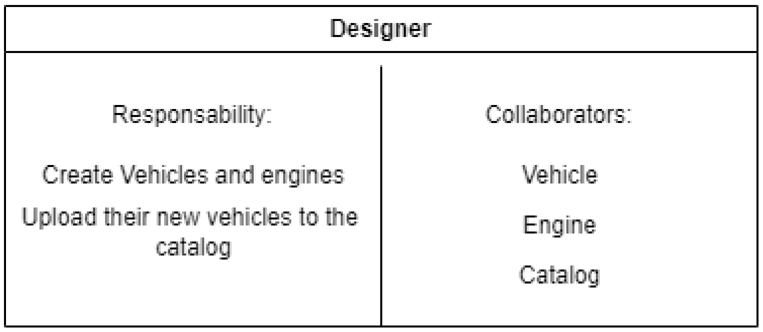
Vehicle	
Responsability: Provide information about the vehicle	Collaborators: Engine Designer Catalog



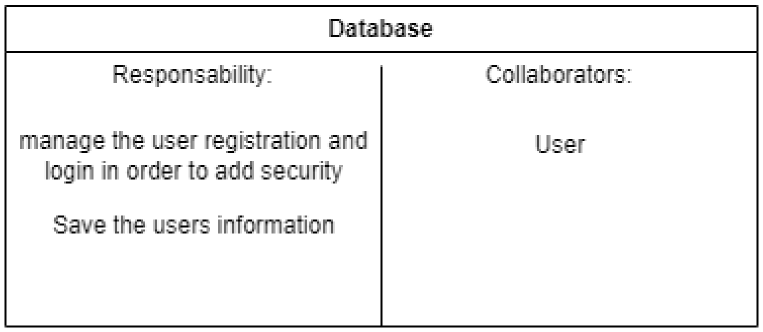
Engine	
Responsability: Provide information about the vehicle consuption and potency is added to a vehicle calculate gas consumption of the vehicle	Collaborators: Vehicle Designer Catalog



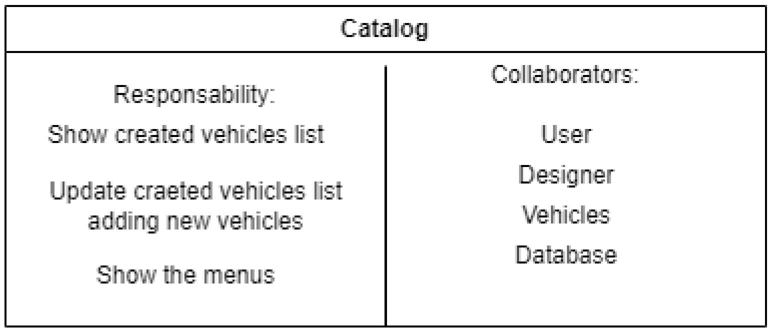
User	
Responsability: Register and login watch vehicles catalog	Collaborators: Catalog Database



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



Database	
Responsability:	Collaborators:
manage the user registration and login in order to add security	User
Save the users information	

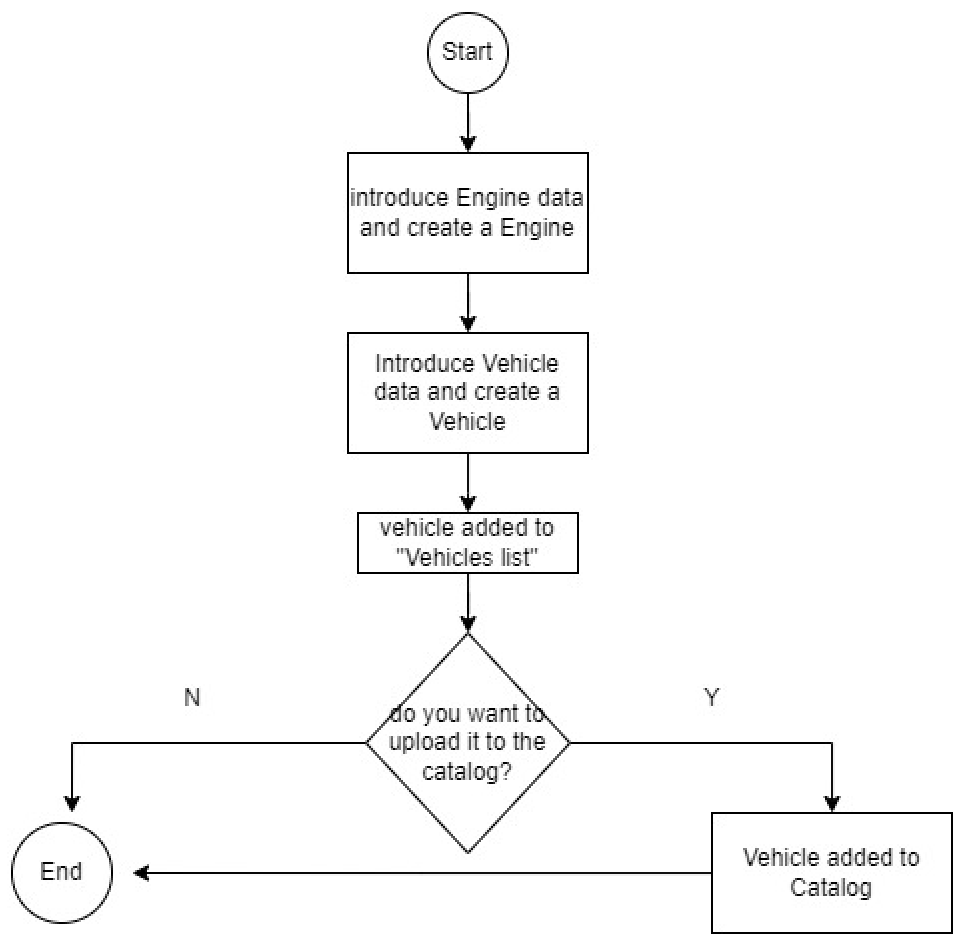


Catalog	
Responsability:	Collaborators:
Show created vehicles list	User
Update craeted vehicles list adding new vehicles	Designer
Show the menus	Vehicles
	Database

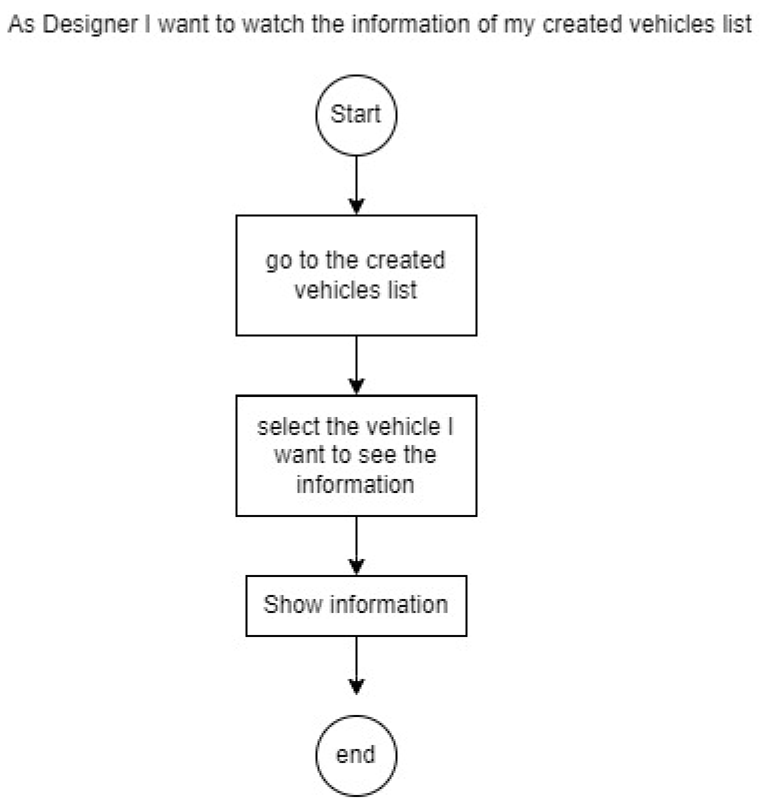
Note: These are the enough CRC Cards, because the yacht, car, motorcycle and truck classes are just concrete versions of vehicle.

Activity Diagrams:

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

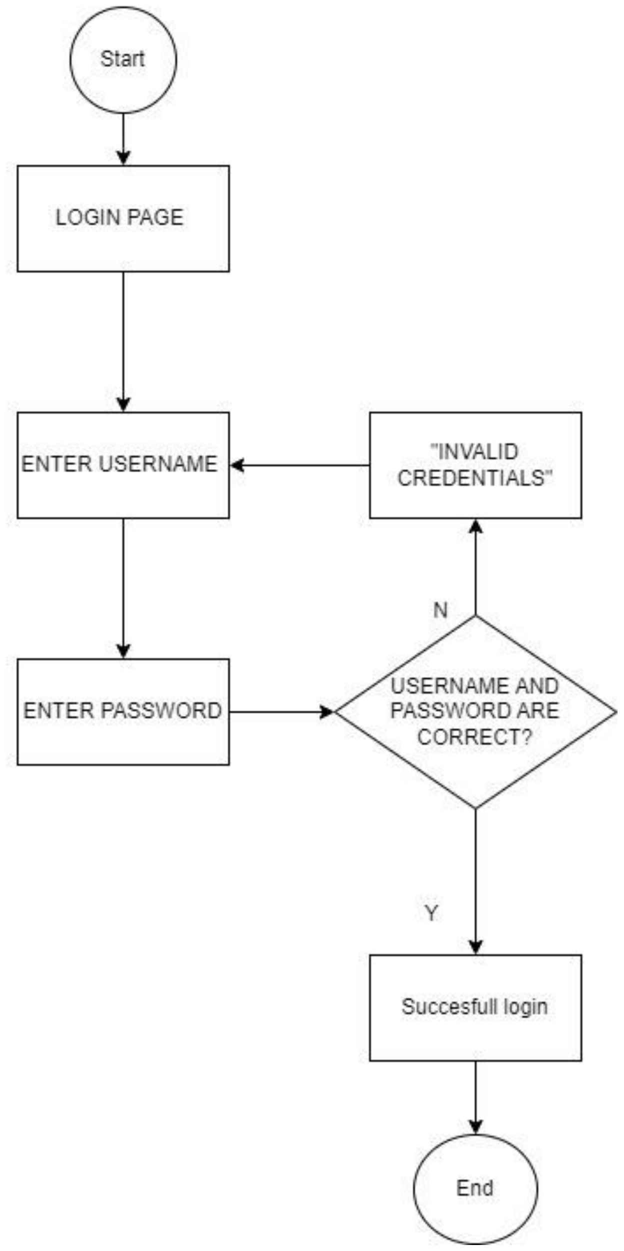

Start
introduce Engine data and create a Engine
Introduce Vehicle data and create a Vehicle
vehicle added to "Vehicles list"
do you want to upload it to the catalog?
N
Y
End
Vehicle added to Catalog

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
Start
go to the created vehicles list
select the vehicle I want to see the information
Show information
end

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.

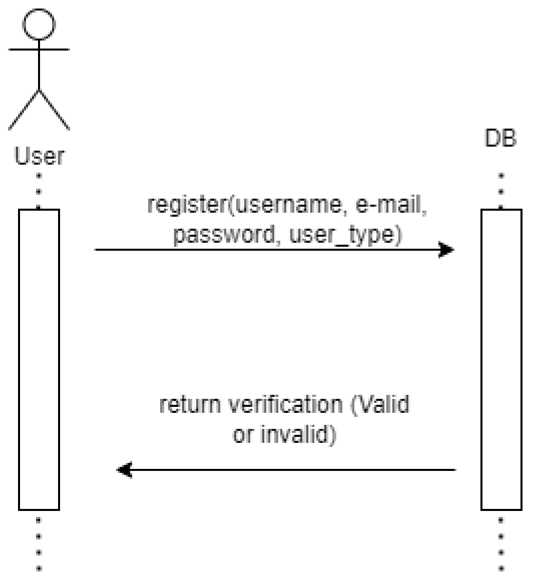


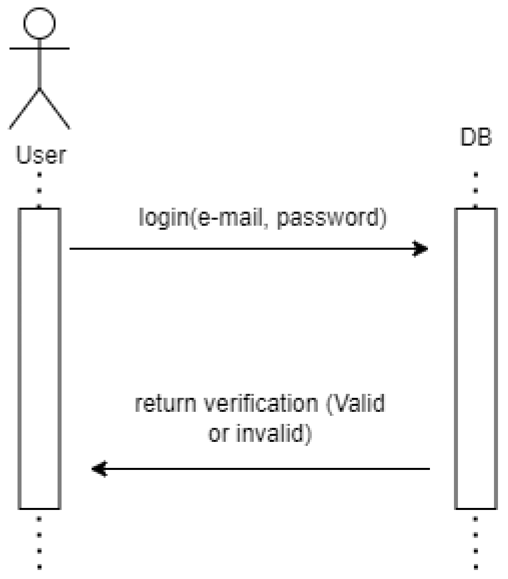
This diagram shows how log int to the system.

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:

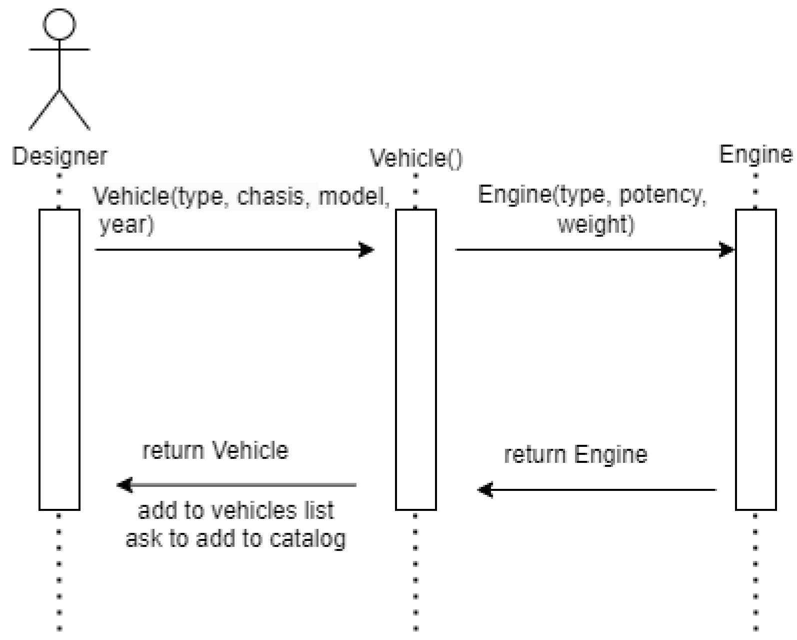
As User I want to create an account to log in and see the created vehicles.

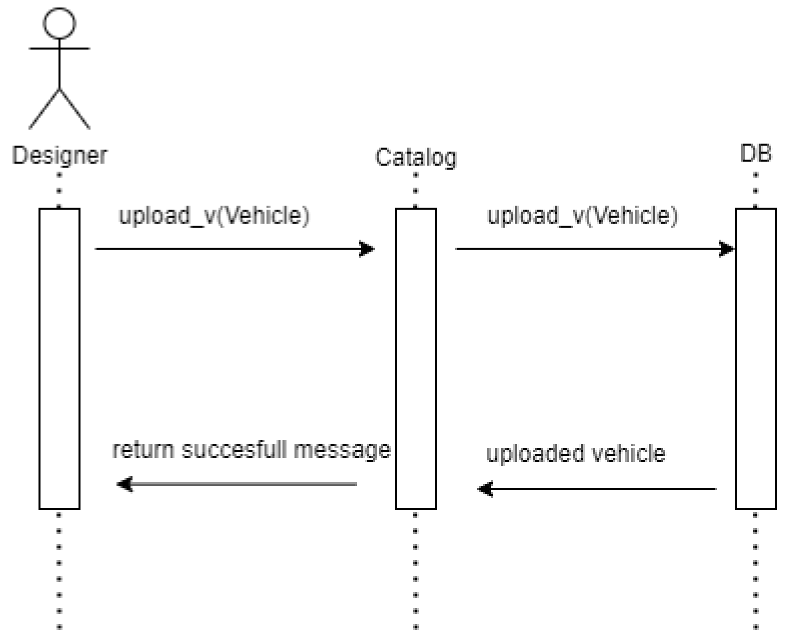

User
DB
register(username, e-mail, password, user_type)
return verification (Valid or invalid)



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

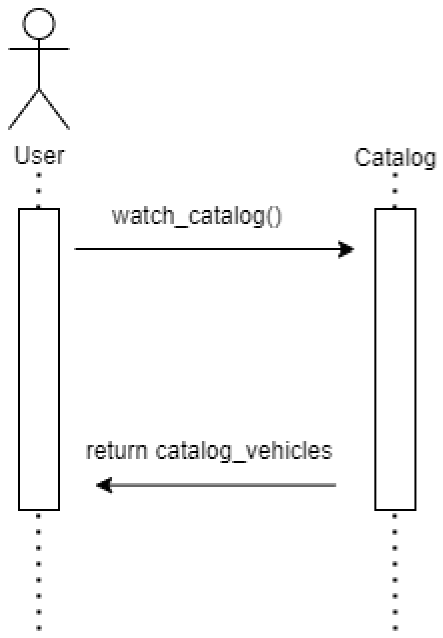
As Designer I want to create a various types of vehicles to have a list of them and upload it





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.

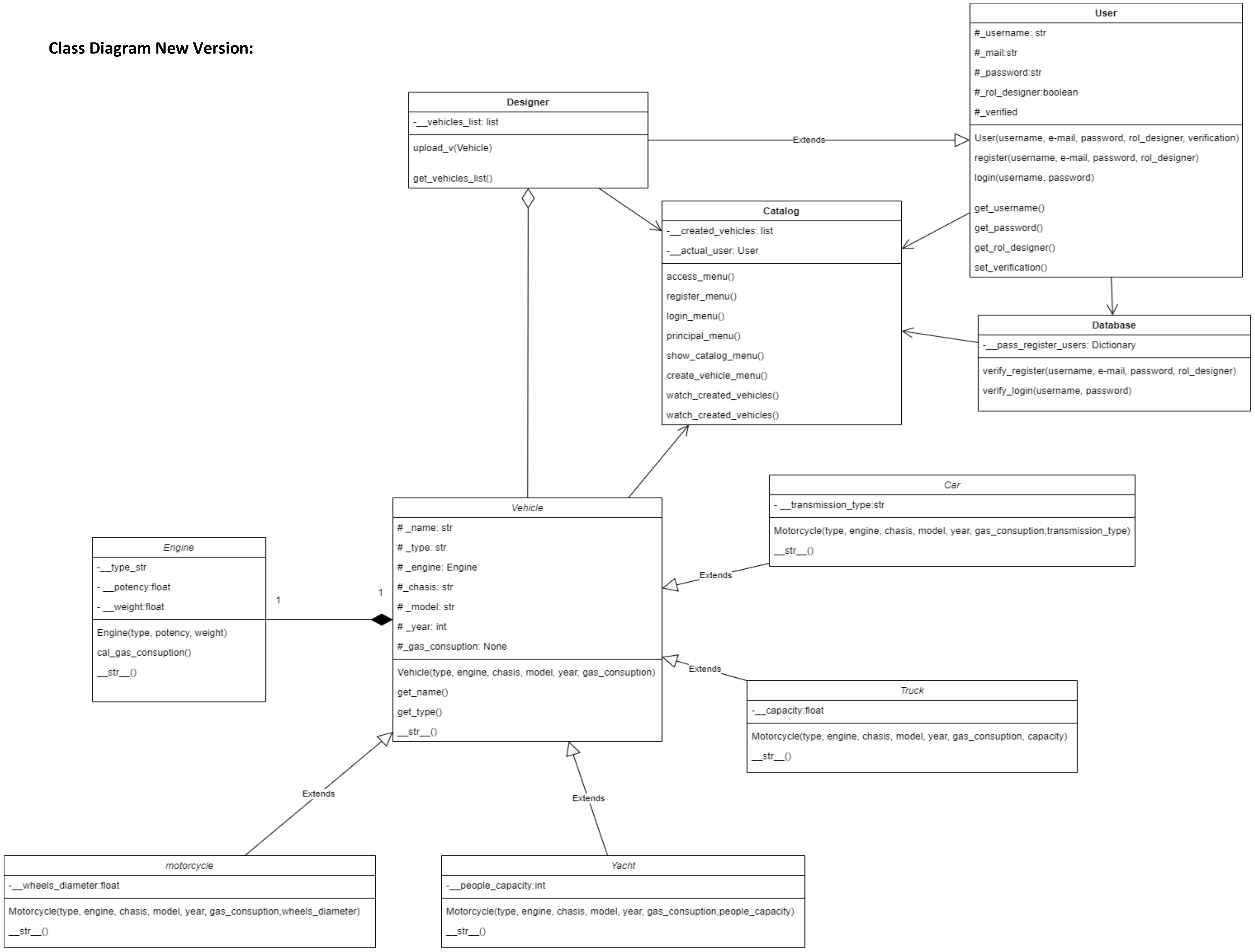
As User I want to watch another created vehicles in a catalog



This diagram shows how a User can watch the catalog and it 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 clasess: Main_controller, View, Launcher
2. New Classes: User, Designer, DB and Catalog
3. New module separation: vehicles and users.
4. New Relations, agreggation, association, composition.
5. Now vehicle has a name attribute
6. No more get-Setter methods, just one that method that return all the atrributes of the vehicles, and the engine (__str__() method). The method to get the vehicle type keeps.
7. cal_gas_consuption() method will pass from Main class to Engine class

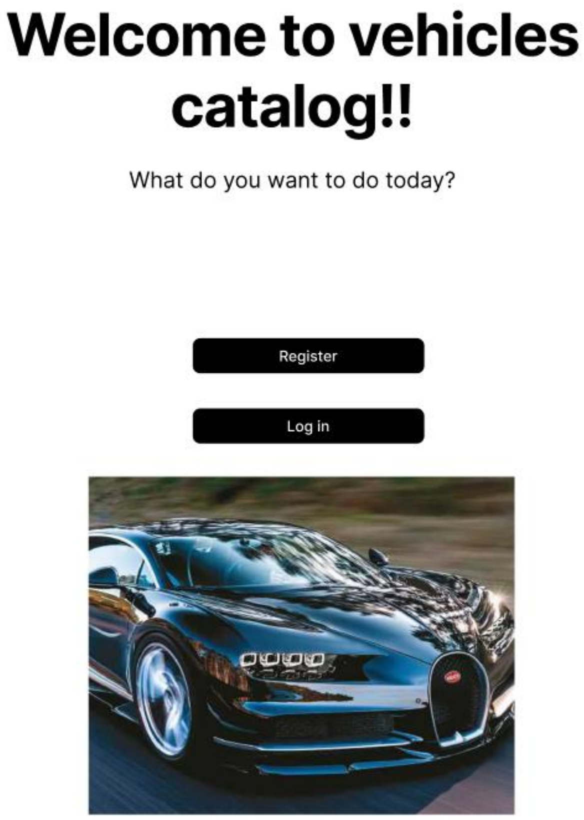
General Description:

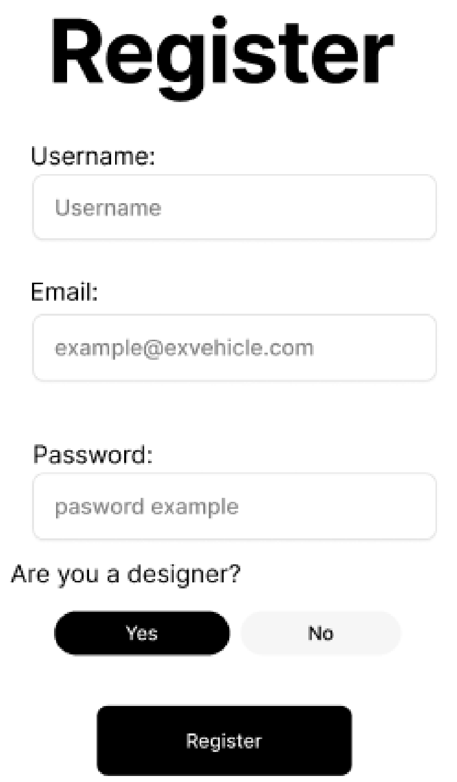
- The first change I did, were the vehicles classes. I create an __str__() method in all the vehicle classes and also add the name attribute in Vehicle class
- I use a composition relation in the Engine class to vehicle class.
- I change the cal_gas_consuption() method from the controller to the Engine 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 an 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.

UI prototipes:

The next protoypes where created with the Figma Tool:

1. The user need a form to register and login





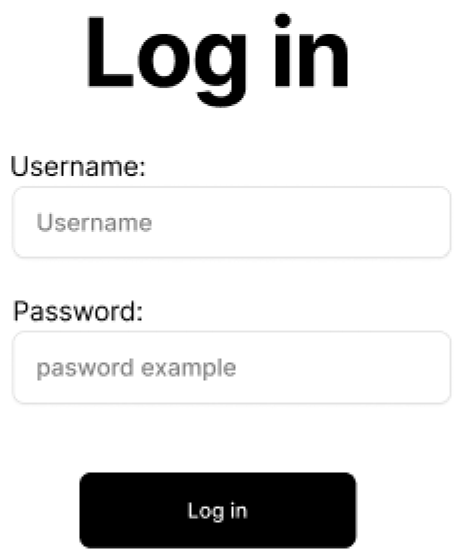
Register

Username:
Username

Email:
example@exvehicle.com

Password:
pasword example

Are you a designer?
[✓ Yes] No



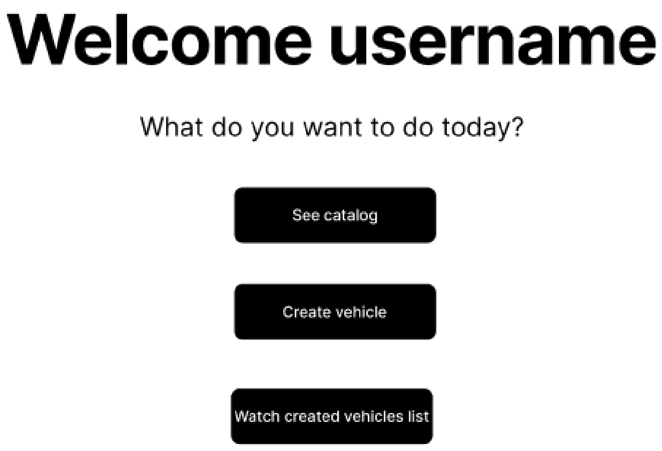
Log in

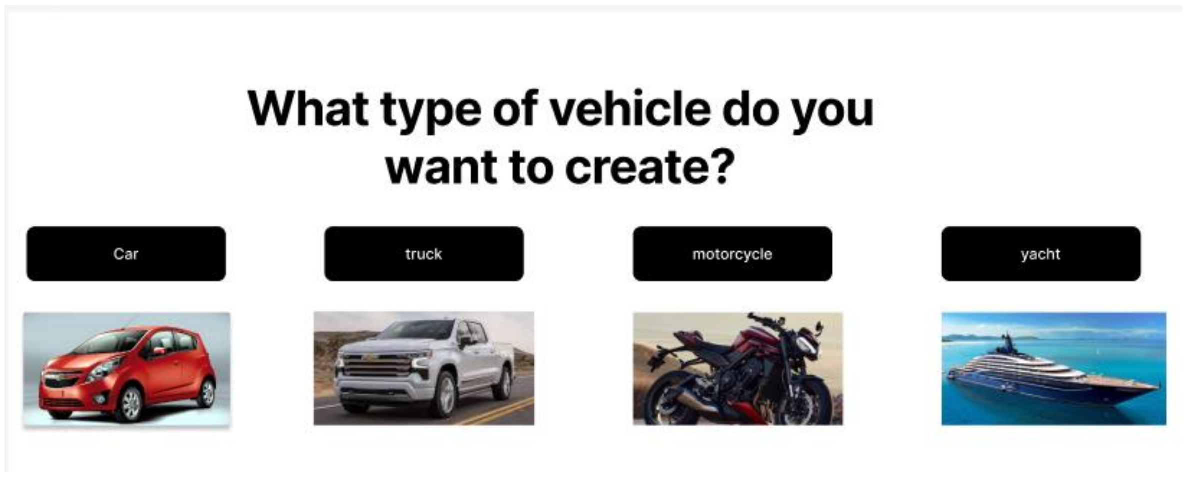
Username:
Username

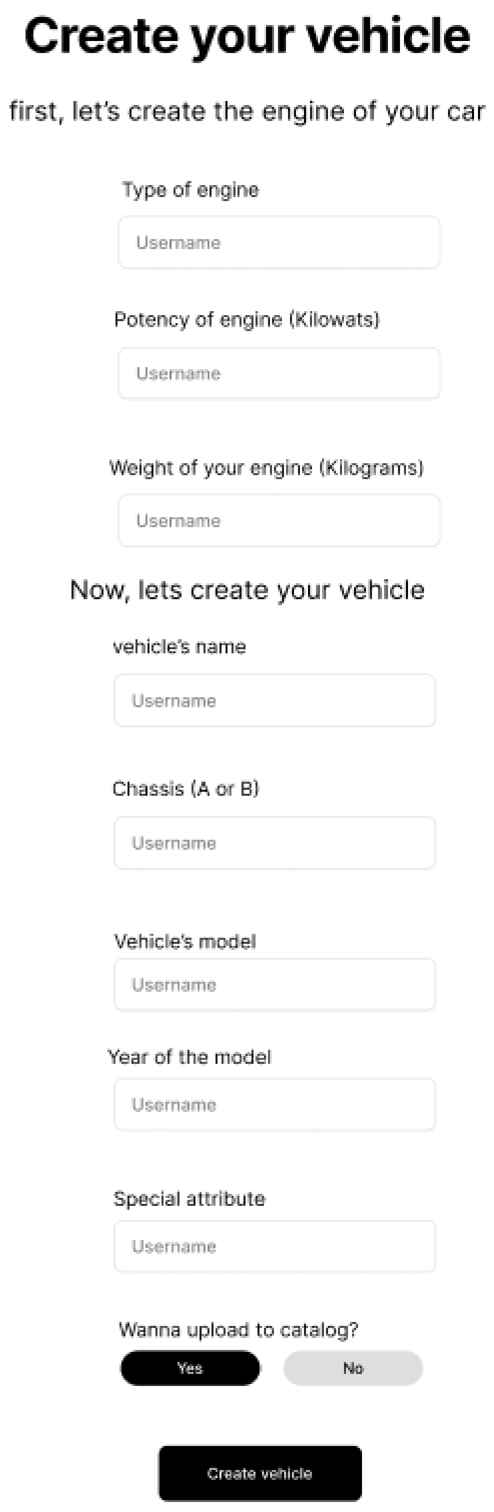
Password:
pasword example

2. The user needs a form to add a new engine to the application.

3. The user needs a form to add a new vehicle to the application.







Create your vehicle

first, let's create the engine of your car

Type of engine

Username

Potency of engine (Kilowats)

Username

Weight of your engine (Kilograms)

Username

Now, lets create your vehicle

vehicle's name

Username

Chassis (A or B)

Username

Vehicle's model

Username

Year of the model

Username

Special attribute

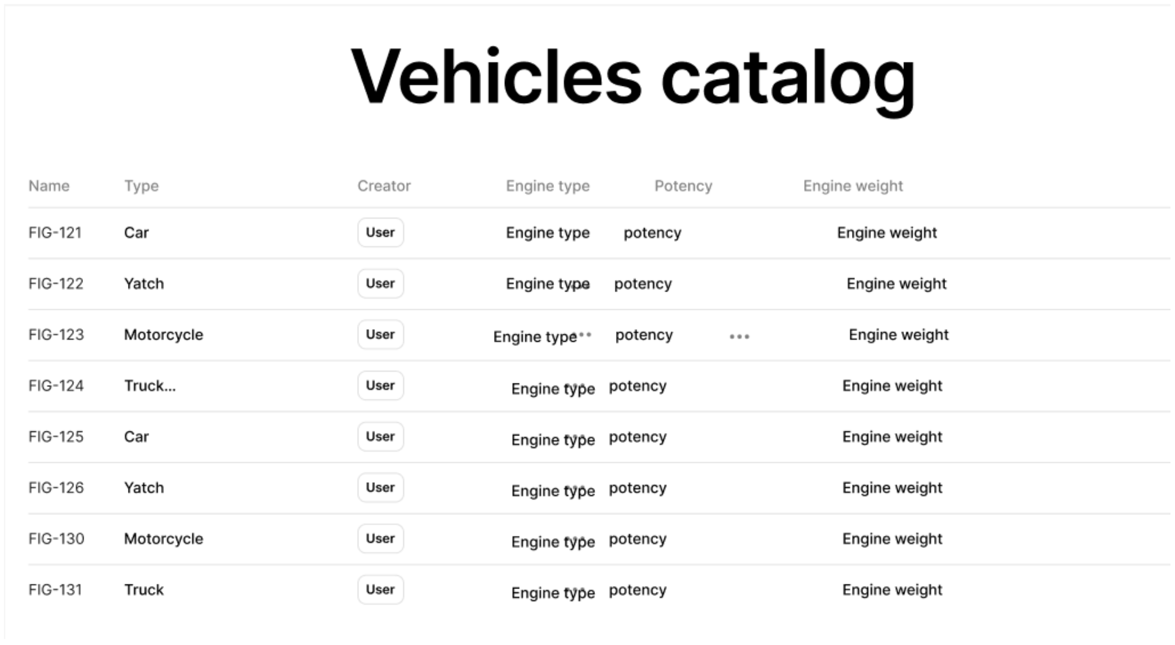
Username

Wanna upload to catalog?

Yes No

Create vehicle

4. The user needs a page to see all engines in the application.
5. The user needs a page to see all vehicles in the application.



Vehicles catalog

Name	Type	Creator	Engine type	Potency	Engine weight
FIG-121	Car	User	Engine type	potency	Engine weight
FIG-122	Yatch	User	Engine type	potency	Engine weight
FIG-123	Motorcycle	User	Engine type	potency	Engine weight
FIG-124	Truck...	User	Engine type	potency	Engine weight
FIG-125	Car	User	Engine type	potency	Engine weight
FIG-126	Yatch	User	Engine type	potency	Engine weight
FIG-130	Motorcycle	User	Engine type	potency	Engine weight
FIG-131	Truck	User	Engine type	potency	Engine weight

Data structures JSON:

1. The first Entity is the user with username, mail, password, rol_designer and a verified
2. The second Entity is the vehicle with name, type, engine, chasis, model, year, gas_consuption.
3. The third Entity is the Engine with type, potency, weight

Transferences Frontend to backend:

1. Register and Login: An user do an register request and these data is sent to the backend:
2. Data base: The created vehicles and users are saved

Web services:

1. **User Register:**
   a. **HTTP Method:** POST
   b. **URL**: /api/register
   c. **Request:** username, email, password, role
   d. **Response:** verification, message

2. **User Log In:**
    a. **HTTP Method:** POST
    b. **URL**: /api/login
    c. **Request:** username, password
    d. **Response:** verification, message
3. **Create vehicle:**
    a. **HTTP Method:** POST
    b. **URL**: /api/create_vehicles
    c. **Request:** name, type, engine, chasis, model, year, gas_consuption
    d. **Response:** vehicle created, succesful message
4. **Create Engine:**
    a. **HTTP Method:** POST
    b. **URL**: /api/create_engine
    c. **Request:** type, potency, weight
    d. **Response:** engine created, succesful message
5. **Get vehicles list**
    a. **HTTP Method:** Get
    b. **URL**: /api/vehicles_list
    c. **Request:** any
    d. **Response:** vehicles list
6. **Get vehicles catalog**
    a. **HTTP Method:** Get
    b. **URL**: /api/catalog
    c. **Request:** any
    d. **Response:** vehicles catalog

NOTE: Some of this services are created in Codes/backend/Services.py