TECHNICAL REPORT
Workshop 4

User stories:

- As an **admin**, I want to create a vehicle, so what I can save it in the catalog.
- As an **admin**, I want to update a vehicle, so what the users can see the new information about it.
- As an **admin,** I want to delete a vehicle, so what the users can't see it anymore in the catalog.
- As a **user,** I want to see vehicles, so what I can have information about it.
- As a **user**, I want to search vehicles, so what I can have information of a specific vehicle.
- As a **user,** I want to register on Vehicle Application, so what I can keep me informed about the latest vehicles created.
- As an **admin**, I want to register on Vehicle Application, so what I can make changes or create vehicles.
- As a **user** and an **admin**, I want to log in application, so what I can scroll on it.
- As an admin, I want to send newsletter, so what I can send the last five vehicles created.
- As a user, I want to subscribe to newsletter, so what I can know about the last vehicles created.

Entities:

- Client (user or admin)
- Vehicle
- Engine
- Catalog
- Newsletter

CRC Cards:

User	
• See vehicles. • Search vehicles.	• CatalogFacadeDecorator

Admin	
• Create vehicles. • Update vehicles. • Delete vehicles	• CatalogFacadeDecorator

Client
Authentication

• Users' authentication • Users' Register	• LoginFacade

Catalog	
• Show list of vehicles	• Vehicle • Engines

Vehicles	
• Provide vehicles information. • Create vehicles	• CatalogFacade

Truck	
• provide truck information. • calculate gas consumption	• Vehicle

Observability	
• check the memory consumption. • check time execution of the searches in the application. • save stats in a performance_log file.	• Catalog

Time Performance	
• check time execution of the searches in the application.	• Catalog

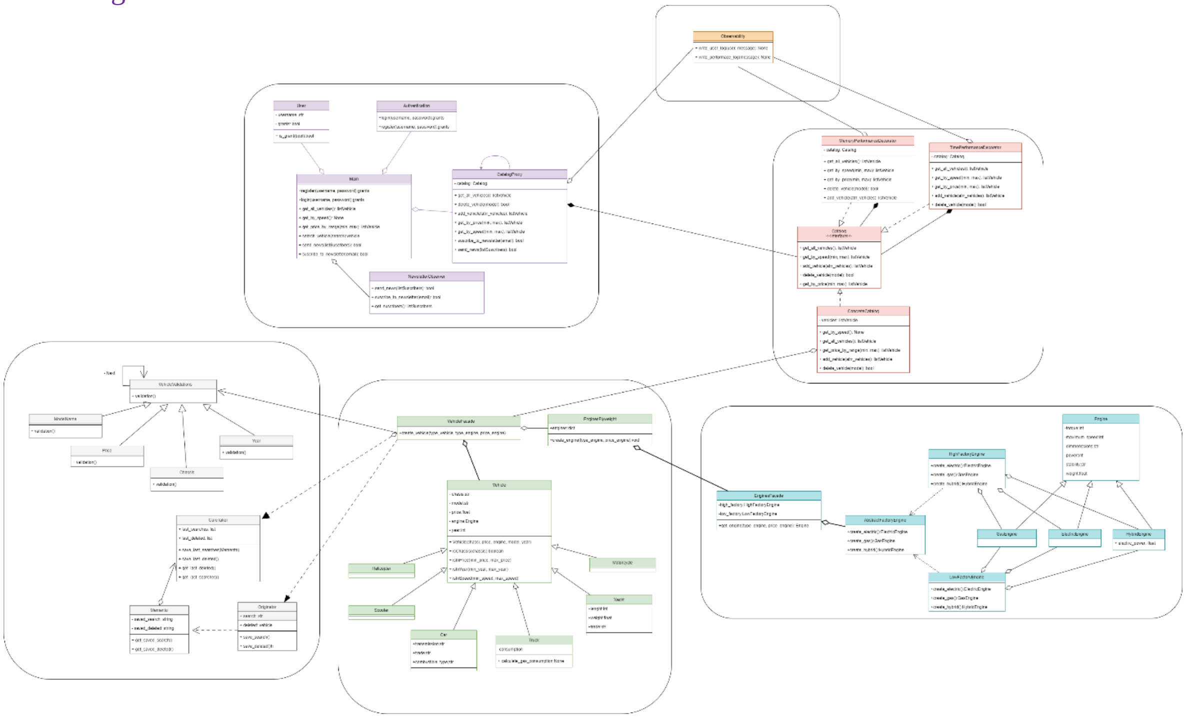
Memory Performance	
• check the memory consumption.	• Catalog

Engines	
- provide engine information	Vehicles

FactoryLowEngines	
- create low price engines.	Vehicles

FactoryHighEngines	
– create high price engines	Vehicles

Class diagram:



For the design of the Project, some additional behavioral design patterns were considered. Such as, Chain of Responsibility, to manage validations when creating vehicles, since it lets you pass requests along a chain of handlers, therefore, the same object created by these validations is passed, allowing not to pass to the next one until it is fulfilled.

Another design is memento, which lets you save and restore the previous state of an object, in this case, it is useful to store the last searched and deleted vehicles in a list, to obtain their information at any time.

The last two mentioned were added to a sub-system, in order not to overload the existing ones. Besides, they perform functions on the same class.
For the newsletter, the implementation of an observer is proposed, since that lets you define a subscription mechanism to notify multiple objects about any events that happen to the object they're observing, in this case the administrator is the one who has the power to send it whenever he wants, to the registered subscribers.

As for the implementation of this proposal, we started by changing the grands with only one value, either true or false, according to this, a menu is established for each type of user, which are two: the admin and the traditional user. The idea was to implement a decorator, however, it was complicated due to the methods already implemented, but anyway they were separated and added other methods, such as register, along with their validations. As for the new hybrid engine, it was added within the already established code.