Technical report workshop 3

This report is meant to show the technical desitions and strategies used during workshop 3

Decision making

* Since it is meant to use less memory it will be tried to implement flyweight pattern
* Facade pattern is seen to be the easiest way to approach the subsystem interaction
* Since user is meant to be monitored, and also the performance of the application it is gonna be implemented a proxy pattern
* Starting the encapsulated method it is et to private some of the methods for the vehicle class
* Added attribute combustible type attribute for easier search
* Decided to create a base user class to later use decorators on it
* For speed purposes abstract factory is eliminated to completely rely on the engine factor class that uses flyweight
* Engine class still uses completely its own attributes that inheritance into gas and electric engines
* High end and low end classes doesn’t exist anymore and the way its represented in the vehicles in the program is the predefined assets of that kind of engines for both electric and gas
* Electric and gas engines still a class since they have to exist to represent the combustion type of the engine
* Further encapsulation meant to be done with the proxy made with the catalog
* For no more time consuming coding facade is still missing when it was meant to be the compilation of the subsystem that were born from the catalog options
* I know that Engine could be an abstract class but for lack of experience couldn’t implement in the given time

Class diagram:

