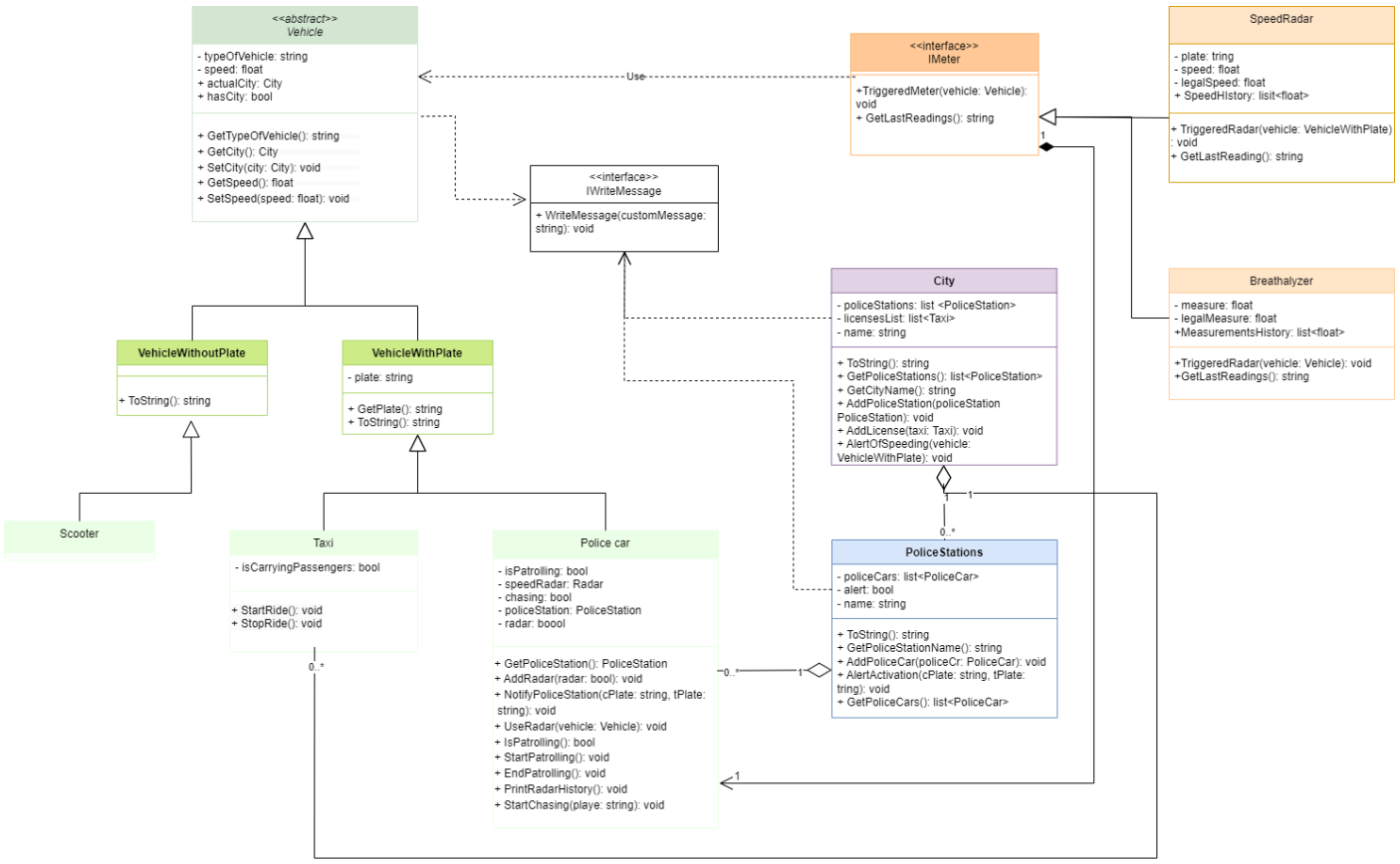
REPORT PRACTICE 2

Victoria Guillén de la Torre

3ºB iMat

1.UML Class diagram



2. SOLID Principles

- **SRP - Single Responsibility Principle:** Each of the classes have their own responsibility.

- **OCP - Open/Closed Principle**: As demonstrated in the code, the project has carried out, without the need to significantly modify any of the classes., the implementation of the class 'Scooter' which inherits from 'Vehicle'. Many of the methods are applicable to any type of Vehicle without the need to specify exactly the type. "We can conclude that the code adheres to this principle, making it modular for the implementation of additional vehicle classes.

- **LSP - Liskov Substitution Principle**: the adherence to this principle can be clearly seen in the 'VehicleWithPlate' and 'VehicleWithoutPlate' classes, where vehicles that may or may not have license plates are abstracted at an intermediate layer.

- **ISP - Interface Segregation Principle:** the implemented interface IMessageWritter focuses solely on the message writing functionality. This approach ensures that classes like Vehicle, PoliceCar or PoliceStation only include methods relevant to their specific context without being forced to implement unnecessary methods. In this case the IMessage interface only provides a writing method used by all the classes that inherit from it.

- **DIP - Dependency Inversion Principle**: depending on abstractions rather than concrete implementations makes the program adhere to this last principle.

3. Question 7

The SOLID principle we are violating is the Single Responsibility Principle which says that each class can only have one single responsibility. In this case, the vehicle class would not only be responsible for handling the logic of various vehicles but also of the different measuring devices, thereby assuming multiple responsibilities.

One possible solution can be the implementation of the 'Bridge pattern' which will be able to separate class inheritance from object composition. The vehicle class contains a reference to the radar implementation and will delegate its functionality to it. Adding new radars to the system won´t depend on creating new subclasses of vehicles.

4. GitHub link

For the moment the project is being built in the 'develop' branch.
https://github.com/victoriaguillen/Software_Architecture_Basics/tree/develop