**מטלה 3**

**מגישים:**

**אמיר דניאלי – 207016692**

**עמרי שקד - 207404484**

**Console UI**

Program – Contains the main and runs the program.

User Interface – Is in charge of all the user interface and wraps the logic of the program.

**GarageLogic**

ValueOutOfRangeException (Inherits from exception) – Exceptions of values that are not in the desired range (for example when a user exceeds amount of fuel available to fill)

WrongEngineTypeException (Inherits from exception) –Exceptions where user tries to fill fuel in electric vehicle or electricity in fuel vehicle.

Garage – A garage in which all new vehicles are stored, and from which the needed actions are executed.

GarageVehicle – Holds object of type Vehicle, and fields with the owner details and car status. Each vehicle in the Garage is represented by a GarageVehicle that holds all its needed details.

eVehicleStatus – Enum that holds status of car, repair, done or paid.

Vehicle – One of the fields of GarageVehicle. This is an object that represents a vehicle, the father class of all vehicle types. It contains fields that are relevant for any vehicle (License plate, model, num of tires, Engine, Tires type...) and a dictionary with questions to ask and their corresponding methods (questions that we ask the user when inserting a new car – for example what type of vehicle it is. The corresponding methods initiate the relevant setter). There are also methods such as set tires (for a given vehicle type) and adding new KVP to the dictionary explained.

Engine – One of the fields of vehicle. It serves as a father class of fuel and electric engine types. Contains fields that are relevant to all engine types (Current energy levels, energy remaining %, max energy capacity). This class is abstract and contains an abstract method – FillEnergy.

ElectricEngine (Inherits from Engine) – Engine that is powered by electricity. Relevant for electrical cars or motorcycles. This class contains the relevant fields for the engine (hours remaining, maximum hours on a charge), and functionality such as – Charge battery and FillEnergy (overrides).

FuelEngine (inherits from Engine) – Engine that is powered by fuel. Relevant for fuel powered cars, motorcycles, and lorry. This class contains the relevant fields for the engine (Fuel tank size, current fuel level, fuel type), and functionality such as – FillFuel and FillEnergy (overrides).

eFuelType – enum, one of the fields in FuelEngine. Contains all relevant fuel types (Octan95, Octan96, Octan98 and Soler).

Tire – Another one of Vehicles fields. It contains fields that are relevant for each tire, such as the manufacturer name, tire pressure and max pressure possible. It also contains functionalities such as – Inflate tire.

Lorry (Inherits from vehicle) – This class represents a lorry. It has fields that are relevant for lorries, such as volume of cargo and a bool field for cold transportation. It also contains fields for the number of tires in a lorry, the maximum pressure for each tire and the capacity of its fuel tank.

Car – (Inherits from vehicle) – This is a father class for car types (electric/fuel). It contains fields that are relevant for any car type, such as its number of tires, max pressure in them, colour (enum) and number of doors (enum).

eCarColour – Enum that represents the colour of the car (Black, White, Red, Blue).

eNumOfDoors -Enum that represents the number of doors the car has (ranging from 2-5).

ElectricCar (Inherits from car) – Represents an electric car with the relevant fields (Max hours of charge on battery).

FuelCar (Inherits from car) – Represents a fuel powered car with the relevant fields (Maximum fuel capacity).

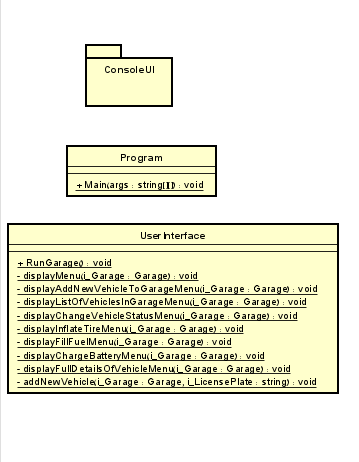
Motorcycle (Inherits from vehicle) – This is the father class for motorcycle types (electric/fuel). It contains fields that are relevant for any motorcycle type, such as the number of tires, max pressure in them and the engine volume.

eMotorcycleLicenseType – Enum that represents the license type of the motorcycle (A, A1, A2, AB).

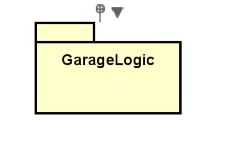
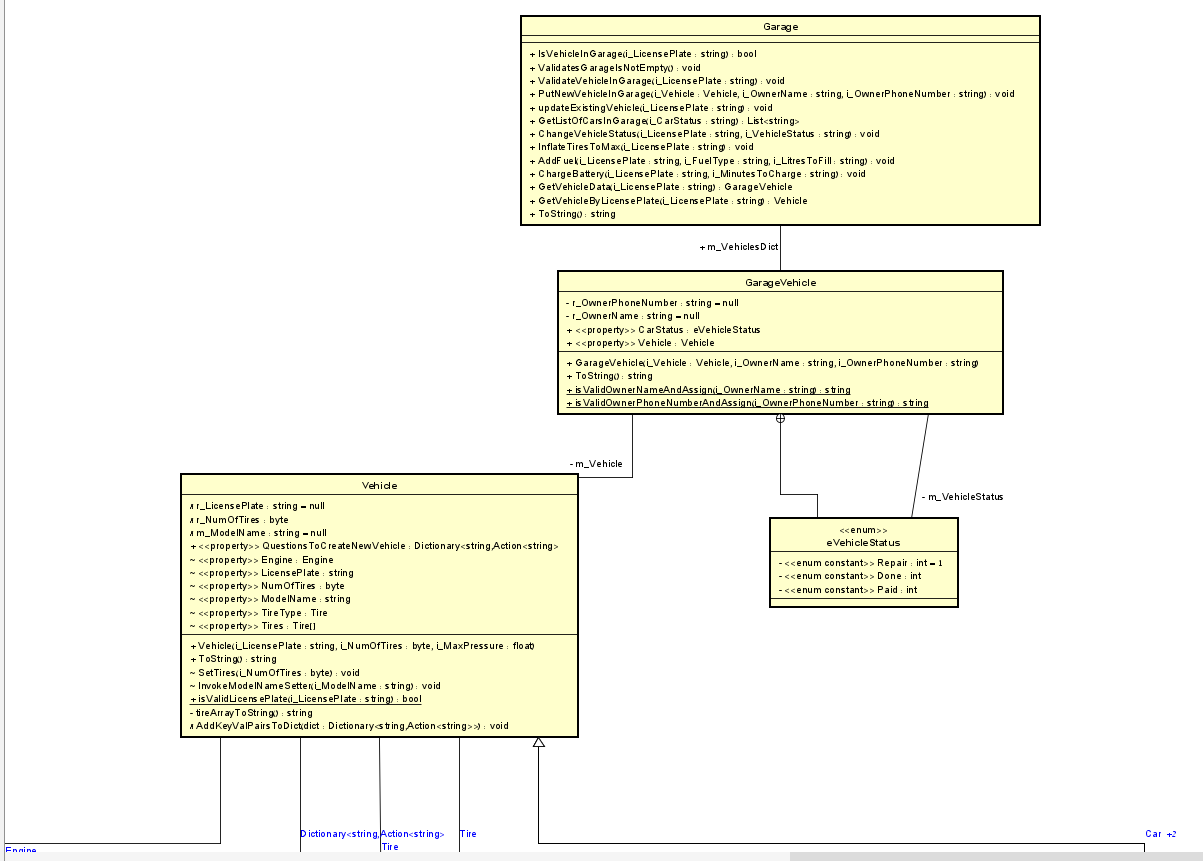
FuelMotorcycle (Inherits from motorcycle) – Represents a fuel powered motorcycle with the relevant fields (Max fuel capacity).

ElectricMotorcycle (Inherits from motorcycle) – Represents an electrical motorcycle with the relevant fields (Max hours of charge on battery).

Dictionary<string, Action<string>> - As explained, the r\_QuestionsToCreateNewVehicle dictionary consists of relevant questions and the corresponding methods. Each one of Engine and Tire consist of a similar dictionary with the relevant questions, and a union operation is conducted on the dictionaries to help create a new vehicle object.   
Additionally, each one of Car, FuelEngine, Lorry, Motorcycle adds its own relevant questions to the dictionary (for example how many doors does the car have? – For a car type) and the relevant method).

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated