**Questions Answers:**

1-Yes, our class diagram follows the SOLID principles as it was used as follows:

* **Single Responsibility**:

is used to make sure that classes have one reason to change as the control class was divided into different classes with different responsibilities and one reason to change also entity classes have Single responsibility as it stores the data about the entity and returns it’s data.

* **Interface Segregation Principle:**

no interface in the program have methods that is passed to a class and never used.

* **Open Close Principle:**

classes are open to be extended by inheritance to add new features or attributes , but don't accept being modified.

2-Yes, our class diagram contains some Design patterns:

**1-Singleton pattern:**

we have garage class that uses the singleton pattern as there can only be one instance of this class at any time.

**2-Strategy pattern:**

we created an interface called “Parkin” to be used to identify different park-in methods and to add a new parking method without changing anything in other classes specially the context class which is GarageParkingControl.

