Top 3 design patterns that I have extensively used are

**Abstract Factory pattern-**Encapsulates a group of individual factories with common functiaonality.

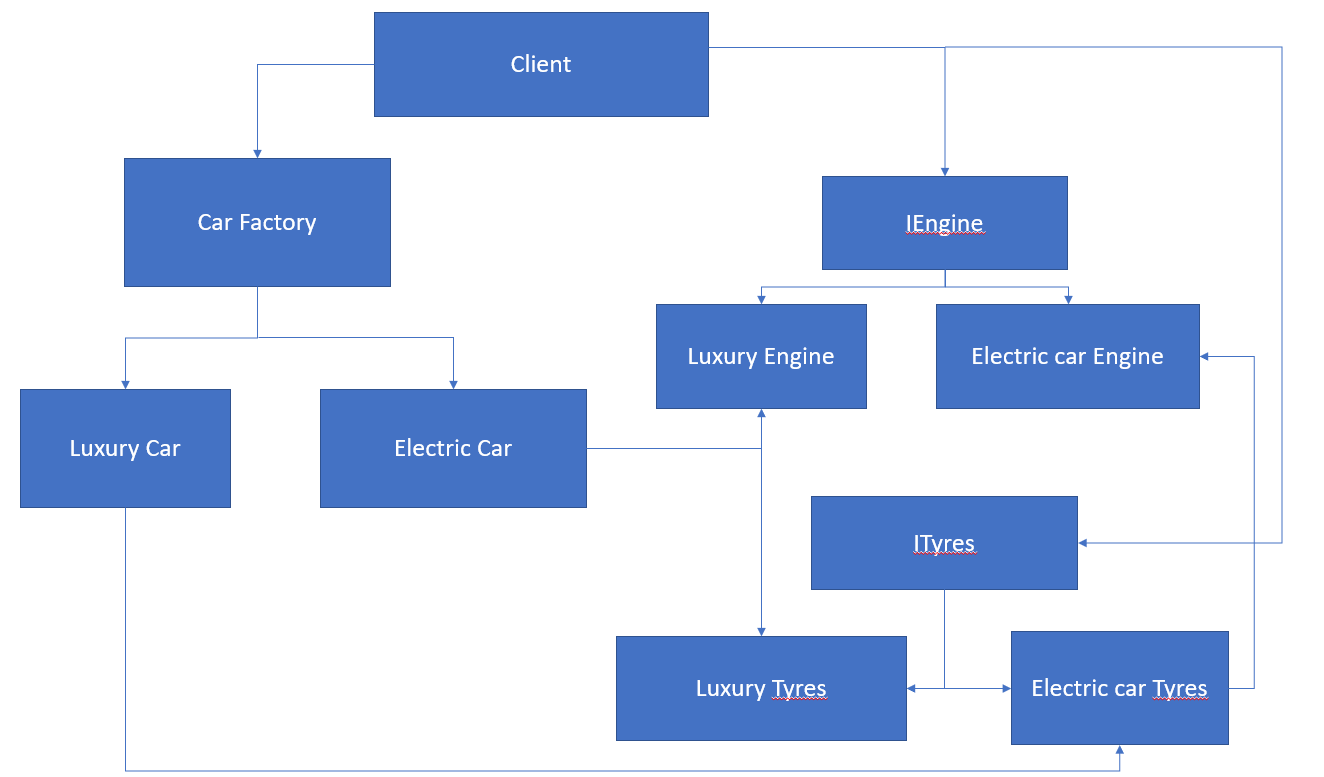
**Mediator pattern-** Defines simplified communication between classes.

**Singleton-**It ensures only one object of a class is ever created.

**Abstract Factory Pattern**

Abstract factory pattern provide an interface for creating related objects without specifying their concrete class. It delegates the responsibility of object instantiation to another object by composition.

It is useful in many complex projects to attain decoupling. A system should be independent of how its products are created. It helps us to create a collection of related products that are designed to be used together by enforcing desired family of products to be at the same time.



In the block diagram given above, Client orders a car, Car factory defines a generic car.This class declares an interface that can be used to create an instance of luxury or electric car.

Each car requires its own specific parts for example Luxury car needs different tyre from electric car, Luxury car requires luxury tyre and Engine.

Each of the components are also provided with abstract class.Their concrete classes are then used to define the implementation details . Car factory declares the methods that can be used to retrieve instances of the objects for each of the abstract component class.Car factory also enforces dependencies that exist between all the component concrete classes.