Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE CODE: DJ19ITC802 **DATE:** 25-02-2023

COURSE NAME: Design Patterns Laboratory **CLASS:** BE - IT

EXPERIMENT NO. 2

CO/LO: Identify and apply the most suitable design pattern to address a given application design problem.

AIM: Design a UML class diagram for Factory Pattern and implement the same for any real life scenario.

DESCRIPTION:

The factory method is a creational design pattern, i.e., related to object creation. In the Factory pattern, we create objects without exposing the creation logic to the client and the client uses the same common interface to create a new type of object. The idea is to use a static member-function (static factory method) that creates & returns instances, hiding the details of class modules from the user. A factory pattern is one of the core design principles to create an object, allowing clients to create objects of a library in a way such that it doesn't have a tight coupling with the class hierarchy of the library.

SOURCE CODE:

```
from abc import ABC, abstractmethod

# Product Interface
class Vehicle(ABC):
    @abstractmethod
    def drive(self):
        pass

# Concrete Products
class Car(Vehicle):
    def drive(self):
        return "Driving a car"

class Bike(Vehicle):
```

```
    def drive(self):
        return "Riding a bike"

class Truck(Vehicle):
    def drive(self):
        return "Driving a truck"

# Factory Class
class VehicleFactory:
    def get_vehicle(self, vehicle_type: str) -> Vehicle:
        if vehicle_type.lower() == "car":
            return Car()
        elif vehicle_type.lower() == "bike":
            return Bike()
        elif vehicle_type.lower() == "truck":
            return Truck()
        else:
            raise ValueError("Unknown vehicle type")

# Client Code Example
if __name__ == "__main__":
    factory = VehicleFactory()

    vehicle1 = factory.get_vehicle("car")
    print(vehicle1.drive())  # Output: Driving a car

    vehicle2 = factory.get_vehicle("bike")
    print(vehicle2.drive())  # Output: Riding a bike

    vehicle3 = factory.get_vehicle("truck")
    print(vehicle3.drive())  # Output: Driving a truck
```

CONCLUSION:

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

We learned to design a UML class diagram for Factory Pattern and implement the same for any real life scenario.

REFERENCES: