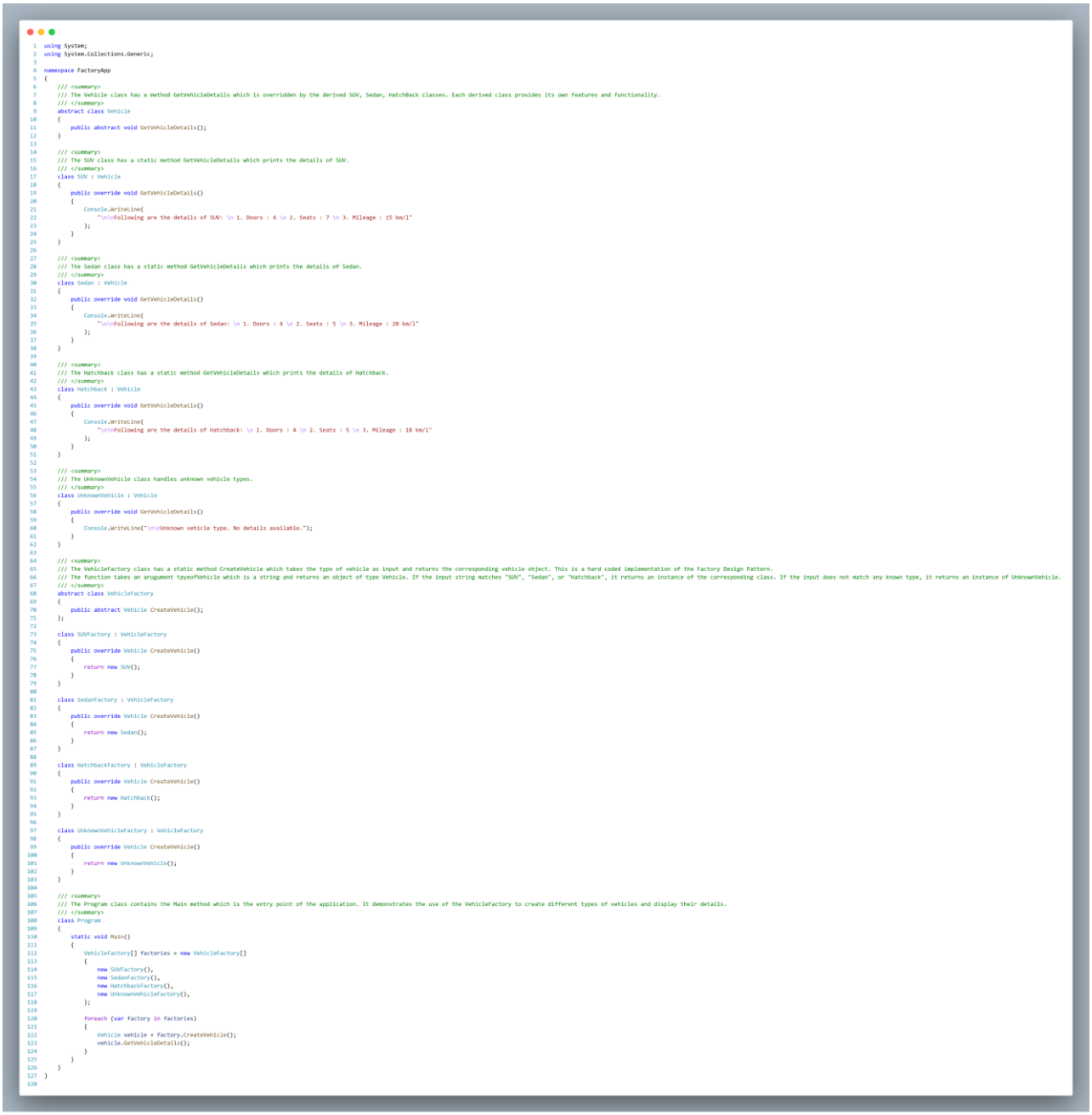
INTERNATIONAL ISLAMIC UNIVERSITY, ISLAMABAD

SOFTWARE ENGINEERING DEPARTMENT

Course: Software Design and Architecture

Lab Task # 01 : Factory Method

Submitted To	:	Mr. Nasir
Submitted By	:	Ahmed Khursheed
Registration No	:	4796-FOC/BSSE/F23



```
using System;
using System.Collections.Generic;

namespace FactoryApp
{
    /// <summary>
    /// The Vehicle class has a method GetVehicleDetails which is overridden by the derived SUV, Sedan, HatchBack classes. Each derived class provides its own features and functionality.
    /// </summary>
    abstract class Vehicle
    {
        public abstract void GetVehicleDetails();
    }

    /// <summary>
    /// The SUV class has a static method GetVehicleDetails which prints the details of SUV.
    /// </summary>
    class SUV : Vehicle
    {
        public override void GetVehicleDetails()
        {
            Console.WriteLine(
                "\n\nFollowing are the details of SUV: \n 1. Doors : 4 \n 2. Seats : 7 \n 3. Mileage : 15 km/l"
            );
        }
    }

    /// <summary>
    /// The Sedan class has a static method GetVehicleDetails which prints the details of Sedan.
    /// </summary>
    class Sedan : Vehicle
    {
        public override void GetVehicleDetails()
        {
            Console.WriteLine(
                "\n\nFollowing are the details of Sedan: \n 1. Doors : 4 \n 2. Seats : 5 \n 3. Mileage : 20 km/l"
            );
        }
    }

    /// <summary>
    /// The Hatchback class has a static method GetVehicleDetails which prints the details of Hatchback.
    /// </summary>
    class Hatchback : Vehicle
    {
        public override void GetVehicleDetails()
        {
            Console.WriteLine(
                "\n\nFollowing are the details of Hatchback: \n 1. Doors : 4 \n 2. Seats : 5 \n 3. Mileage : 18 km/l"
            );
        }
    }

    /// <summary>
    /// The UnknownVehicle class handles unknown vehicle types.
    /// </summary>
    class UnknownVehicle : Vehicle
    {
        public override void GetVehicleDetails()
        {
            Console.WriteLine("\n\nUnknown vehicle type. No details available.");
        }
    }

    /// <summary>
    /// The VehicleFactory class has a static method CreateVehicle which takes the type of vehicle as input and returns the corresponding vehicle object. This is a hard coded implementation of the Factory Design Pattern.
    /// The function takes an argument tpyeofVehicle which is a string and returns an object of type Vehicle. If the input string matches "SUV", "Sedan", or "Hatchback", it returns an instance of the corresponding class. If the input does not match any known type, it returns an instance of UnknownVehicle.
    /// </summary>
    abstract class VehicleFactory
    {
        public abstract Vehicle CreateVehicle();
    };

    class SUVFactory : VehicleFactory
    {
        public override Vehicle CreateVehicle()
        {
            return new SUV();
        }
    }

    class SedanFactory : VehicleFactory
    {
        public override Vehicle CreateVehicle()
        {
            return new Sedan();
        }
    }

    class HatchbackFactory : VehicleFactory
    {
        public override Vehicle CreateVehicle()
        {
            return new Hatchback();
        }
    }

    class UnknownVehicleFactory : VehicleFactory
    {
        public override Vehicle CreateVehicle()
        {
            return new UnknownVehicle();
        }
    }

    /// <summary>
    /// The Program class contains the Main method which is the entry point of the application. It demonstrates the use of the VehicleFactory to create different types of vehicles and display their details.
    /// </summary>
    class Program
    {
        static void Main()
        {
            VehicleFactory[] factories = new VehicleFactory[]
            {
                new SUVFactory(),
                new SedanFactory(),
                new HatchbackFactory(),
                new UnknownVehicleFactory(),
            };

            foreach (var factory in factories)
            {
                Vehicle vehicle = factory.CreateVehicle();
                vehicle.GetVehicleDetails();
            }
        }
    }
}
```

```
Following are the details of Sedan:
 1. Doors : 4
 2. Seats : 5
 3. Mileage : 20 km/l


Following are the details of Hatchback:
 1. Doors : 4
 2. Seats : 5
 3. Mileage : 18 km/l


Unknown vehicle type. No details available.
PS D:\SDA>
```