Task 6: Service Provider Interface /Abstract class

Create Interface /Abstract class ICourierUserService and ICourierAdminService interface

ICourierUserService {

placeOrder()

Place a new courier order.

@param courierObj Courier object created using values entered by users

@return The unique tracking number for the courier order .

Use a static variable to generate unique tracking number. Initialize the static variable in Courier class with some random value. Increment the static variable each time in the constructor to generate next values.

getOrderStatus();

Get the status of a courier order.

@param trackingNumber The tracking number of the courier order.

return The status of the courier order (e.g., yetToTransit, In Transit, Delivered).

cancelOrder()

Cancel a courier order.

@param trackingNumber The tracking number of the courier order to be canceled.

@return True if the order was successfully canceled, false otherwise.

getAssignedOrder();

Get a list of orders assigned to a specific courier staff member

@param courierStaffId The ID of the courier staff member.

@return A list of courier orders assigned to the staff member.

ICourierAdminService

int addCourierStaff(Employee obj);

Add a new courier staff member to the system.

@param name The name of the courier staff member.

@param contactNumber The contact number of the courier staff member.

@return The ID of the newly added courier staff member.

Code:

ICourierUserService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CMS
{
    public interface ICourierUserService
    {

        string PlaceOrder(Courier courierObj);
        string GetOrderStatus(string trackingNumber);
        bool CancelOrder(string trackingNumber);
        List<Courier> GetAssignedOrders(int courierStaffId);
    }
}
```

CourierUserService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CMS.Services
{
    public class CourierUserService : ICourierUserService
    {
        private List<Courier> courierOrders;
```

```
    public CourierUserService()
    {
        courierOrders = new List<Courier>();
    }
 public string PlaceOrder(Courier courierObj)
    {
        courierOrders.Add(courierObj);
        return courierObj.TrackingNumber;
    }

    public string GetOrderStatus(string trackingNumber)
    {
        var order = courierOrders.FirstOrDefault(c => c.TrackingNumber == trackingNumber);
        return order != null ? order.Status : "Order not found!";
    }

    public bool CancelOrder(string trackingNumber)
    {
        var order = courierOrders.FirstOrDefault(c => c.TrackingNumber == trackingNumber);
        if (order != null && order.Status != "Delivered")
        {
            order.Status = "Cancelled";
            return true;
        }
        return false;
    }
public List<Courier> GetAssignedOrders(int courierStaffId)
    {
        return courierOrders.Where(c => c.UserId == courierStaffId).ToList();
    }
  }}
```

ICourierAdminService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CMS
{
   public interface ICourierAdminService
   {
      int AddCourierStaff(Employee obj);
   }
}
```

CourierAdminService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace CMS.Services
{
   public class CourierService : ICourierAdminService
   {
      private List<Employee> employees = new List<Employee>();
      private static int employeeIdSeed = 1;
      public int AddCourierStaff(Employee obj)
      {
         obj.EmployeeID = employeeIdSeed++;
          employees.Add(obj);
          return obj.EmployeeID;  }}}
```