

C# ASSIGNMENT COURIER MANAGEMENT SYSTEM

NAME: GOKUL T M

```
using System.Collections;
using System.Globalization;

namespace CourierManagementTask2_4
{
    internal class Program
    {

static void Main(string[] args)
{
    Console.WriteLine("Courier Management System");
    Console.WriteLine("Enter Question Number to Run (1 to 15): ");
    int choice = int.Parse(Console.ReadLine());

    switch (choice)
    {
        case 1:
            Question1();
            break;
        case 2:
            Question2();
            break;
        case 3:
            Question3();
            break;
        case 4:
            Question4();
            break;
        case 5:
            Question5();
            break;
        case 6:
            Question6();
            break;
        case 7:
            Question7();
            break;
        case 8:
            Question8();
            break;
        case 9:
            Question9();
            break;
        case 10:
            Question10();
            break;
        case 11:
            Question11();
            break;
        case 12:
            Question12();
            break;
        case 13:
            Question13();
            break;
        case 14:
            Question14();
            break;
    }
}
```

```

        case 15:
            Question15();
            break;
        default:
            Console.WriteLine("Invalid Choice");
            break;
    }
}

```

Coding Task 1: Control Flow Statements

1. Write a program that checks whether a given order is delivered or not based on its status (e.g., "Processing," "Delivered," "Cancelled"). Use if-else statements for this.

```

private static void Question1()
{
    // Write a program that checks whether a given order is delivered or not
    // based on its status (e.g.,
    // "Processing," "Delivered," "Cancelled"). Use if-else statements for
    // this.

    Console.WriteLine("Enter the order status (Processing, Delivered,
Cancelled) : ");
    string? status = Console.ReadLine();

    if (status == null)
    {
        Console.WriteLine("Please Enter a valid status in
(Processing,Delivered or Cancelled)");
    }
    else if (status.Equals("Processing", StringComparison.OrdinalIgnoreCase))
    {
        Console.WriteLine("The order is Processing and is not delivered");
    }
    else if (status.Equals("Delivered", StringComparison.OrdinalIgnoreCase))
    {
        Console.WriteLine("The order is delivered");
    }
    else if (status.Equals("Cancelled", StringComparison.OrdinalIgnoreCase))
    {
        Console.WriteLine("The order is Cancelled");
    }
    else
    {
        Console.WriteLine("Please Enter a Valid Status!");
    }
}

```

```

Courier Management System
Enter Question Number to Run (1 to 15):
1
Enter the order status (Processing, Delivered, Cancelled) :
Delivered
The order is delivered

```

2. Implement a switch-case statement to categorize parcels based on their weight into "Light," "Medium," or "Heavy."

```

private static void Question2()
{
    // 2. Implement a switch-case statement to categorize parcels based on
    // their weight into "Light,"
    // "Medium," or "Heavy."

    Console.WriteLine("Enter the Courier Weight: ");
    double CourierWeight = Convert.ToDouble(Console.ReadLine());

    switch (CourierWeight)
    {
        case <= 0:
            Console.WriteLine("Weight Cannot be less than Zero");
            break;
        case <= 1.5:
            Console.WriteLine("Light Weight");
            break;
        case <= 2.5:
            Console.WriteLine("Medium Weight");
            break;
        case >= 2.5:
            Console.WriteLine("Heavy Weight");
            break;
        default:
            Console.WriteLine("Please Enter Correct Value");
            break;
    }
}

```

```

Courier Management System
Enter Question Number to Run (1 to 15):
2
Enter the Courier Weight:
2.0
Medium Weight

C:\Users\gokul\OneDrive\Desktop\C#\workspace\CourierManagementSystemC#\CourierManagementTask2-4\bin\Debug\net8.0\Courier
ManagementTask2-4.exe (process 4076) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .

```

3. Implement User Authentication 1. Create a login system for employees and customers using Java control flow statements.

```
// 3. Implement User Authentication 1. Create a login system for employees
and customers using Java
// control flow statements.

Dictionary<string, string> customers = new Dictionary<string, string>();
customers.Add("gokul@gmail.com", "pass#12");

Dictionary<string, string> employees = new Dictionary<string, string>();
employees.Add("vignesh@gmail.com", "vicky007");

Console.WriteLine("Choose 1 or 2: ");
Console.WriteLine("1.User/customer");
Console.WriteLine("2.Employee");
string choice = Console.ReadLine();

Console.WriteLine("Please Enter your Email ID: ");
string emailId = Console.ReadLine();
Console.WriteLine("Please Enter your Password ");
string password = Console.ReadLine();

switch (choice)
{
    case "1":
        if (customers.ContainsKey(emailId) && customers[emailId] ==
password)
        {
            Console.WriteLine("Customer login successfull");
        }
        else
        {
            Console.WriteLine("invalid userId or Password");
        }
        break;

    case "2":
        if (employees.ContainsKey(emailId) && employees[emailId] ==
password)
        {
            Console.WriteLine("Employee login successfull");
        }
        else
        {
            Console.WriteLine("invalid userId or Password");
        }
        break;

    default:
        Console.WriteLine("Please Enter a valid Choice in(1 or 2)");
        break;
}
}
```

```
Microsoft Visual Studio Debu X + v
Courier Management System
Enter Question Number to Run (1 to 15):
3
Choose 1 or 2:
1.User/customer
2.Employee
1
Please Enter your Email ID:
gokul@gmail.com
Please Enter your Password
pass#12
Customer login successfull

C:\Users\gokul\OneDrive\Desktop\C#_workspace\CourierManagementSystemC#\CourierManagementTask2-4\bin\Debug\net8.0\Courier
ManagementTask2-4.exe (process 14088) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

4. Implement Courier Assignment Logic 1. Develop a mechanism to assign couriers to shipments based on predefined criteria (e.g., proximity, load capacity) using loops.

```
private static void Question4()
{
    // 4. Implement Courier Assignment Logic 1. Develop a mechanism to assign
    couriers to shipments based
    // on predefined criteria(e.g., proximity, load capacity) using loops.

    string[] couriers = { "karthick", "Sam", "Ravi" };
    string[] areas = { "Chennai", "Mumbai", "Chennai" };
    int[] capacities = { 100, 80, 60 };
    int[] loads = { 30, 20, 50 };

    string shipmentArea = "Chennai";
    int shipmentWeight = 20;

    bool assigned = false;

    for (int i = 0; i < couriers.Length; i++)
    {
        if (areas[i] == shipmentArea && (loads[i] + shipmentWeight <=
        capacities[i]))
        {
            Console.WriteLine($"Shipment assigned to {couriers[i]}");
            loads[i] += shipmentWeight;
            assigned = true;
            break;
        }
    }

    if (!assigned)
        Console.WriteLine("No courier available for this shipment.");
}
```

```
Microsoft Visual Studio Debu... X
Courier Management System
Enter Question Number to Run (1 to 15):
4
Shipment assigned to karthick

C:\Users\goku1\OneDrive\Desktop\C#_workspace\CourierManagementSystemC#\
ManagementTask2-4.exe (process 12552) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->
le when debugging stops.
Press any key to close this window . . .
```

Task 2: Loops and Iteration

5. Write a Java program that uses a for loop to display all the orders for a specific customer

```
private static void Question5()
{
    Console.WriteLine("Enter the name of Customer:");
    string name = Console.ReadLine();

    Console.WriteLine("Enter the number of orders");
    int ordercount = int.Parse(Console.ReadLine());

    string[] orders = new string[ordercount];

    for (int i = 0; i < orders.Length; i++)
    {
        Console.Write($"Order no{i + 1}:");
        orders[i] = Console.ReadLine();
    }
    Console.WriteLine($"
Orders for customer {name}");
    for (int i = 0; i < orders.Length; i++)
    {
        Console.WriteLine($"orders {i + 1}:{orders[i]}");
    }
}
```

```
Microsoft Visual Studio Debug Console
Courier Management System
Enter Question Number to Run (1 to 15):
5
Enter the name of Customer:
Gokul
Enter the number of orders
3
Order no1:shoes
Order no2:watch
Order no3:bag

Orders for customer Gokul
orders 1:shoes
orders 2:watch
orders 3:bag

C:\Users\gokul\OneDrive\Desktop\C#\workspace\CourierManagementSystem
ManagementTask2-4.exe (process 21728) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tool
le when debugging stops.
Press any key to close this window . . .
```

6. Implement a while loop to track the real-time location of a courier until it reaches its destination.

```
private static void Question6()
{
    int start = 1;
    int destination = 10;

    while (start <= destination)
    {
        Console.WriteLine($"The Courier has reached destination number :
{start}");
        start++;
    }
}
```

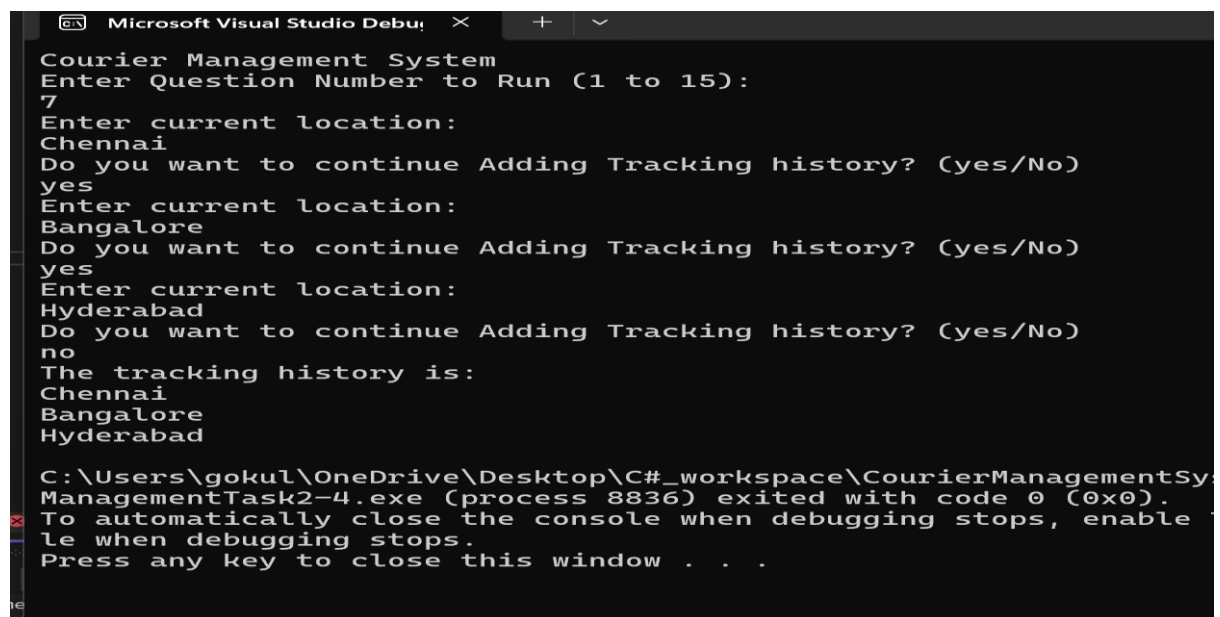
```
Microsoft Visual Studio Debug Console
Courier Management System
Enter Question Number to Run (1 to 15):
6
The Courier has reached destination number : 1
The Courier has reached destination number : 2
The Courier has reached destination number : 3
The Courier has reached destination number : 4
The Courier has reached destination number : 5
The Courier has reached destination number : 6
The Courier has reached destination number : 7
The Courier has reached destination number : 8
The Courier has reached destination number : 9
The Courier has reached destination number : 10

C:\Users\gokul\OneDrive\Desktop\C#\workspace\CourierManagementSystem
ManagementTask2-4.exe (process 31776) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tool
le when debugging stops.
Press any key to close this window . . .
```

. Task 3: Arrays and Data Structures

7. Create an array to store the tracking history of a parcel, where each entry represents a location update.

```
}  
  
private static void Question7()  
{  
    List<string> location = new List<string>();  
    string choice;  
    string loc;  
  
    do  
    {  
        Console.WriteLine("Enter current location: ");  
        loc = Console.ReadLine();  
        location.Add(loc);  
  
        Console.WriteLine("Do u want to continue Adding Tracking history?  
(yes/No)");  
        choice = Console.ReadLine();  
  
    } while (choice == "y");  
  
    Console.WriteLine($"The tracking history is:");  
  
    foreach (string item in location)  
    {  
        Console.WriteLine(item);  
    }  
}
```



```
Microsoft Visual Studio Debug Console  
Courier Management System  
Enter Question Number to Run (1 to 15):  
7  
Enter current location:  
Chennai  
Do you want to continue Adding Tracking history? (yes/No)  
yes  
Enter current location:  
Bangalore  
Do you want to continue Adding Tracking history? (yes/No)  
yes  
Enter current location:  
Hyderabad  
Do you want to continue Adding Tracking history? (yes/No)  
no  
The tracking history is:  
Chennai  
Bangalore  
Hyderabad  
C:\Users\gokul\OneDrive\Desktop\C#_workspace\CourierManagementSystem\ManagementTask2-4.exe (process 8836) exited with code 0 (0x0).  
To automatically close the console when debugging stops, enable  
the option in the Debug menu.  
Press any key to close this window . . .
```


8. Implement a method to find the nearest available courier for a new order using an array of couriers.

```
private static void Question8()
{
    int[] CourierLocation = { 10, 5, 8 };

    Console.WriteLine("Enter the location of new order to find the nearest courier location: ");
    int orderLocation = Convert.ToInt32(Console.ReadLine());

    int minIndex = -1;
    int min = int.MaxValue;

    for (int i = 0; i < CourierLocation.Length; i++)
    {
        if (Math.Abs(CourierLocation[i] - orderLocation) < min)
        {
            min = Math.Abs(CourierLocation[i] - orderLocation);
            minIndex = i;
        }
    }

    Console.WriteLine($"Index of nearest available Courier is: {minIndex + 1} with a distance of {min}");
}
```

```
Microsoft Visual Studio Debug Console
Courier Management System
Enter Question Number to Run (1 to 15):
8
Enter the location of new order to find the nearest courier location:
7
Index of nearest available Courier is: 3 with a distance of 1

C:\Users\gokul\OneDrive\Desktop\C#\workspace\CourierManagementSystemC#\CourierManagementTask2-4.exe (process 15884) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debug Console->Close when debugging stops.
Press any key to close this window . . .
```

Task 4: Strings, 2d Arrays, user defined functions, Hashmap

9. Parcel Tracking: Create a program that allows users to input a parcel tracking number. Store the tracking number and Status in 2d String Array. Initialize the array with values. Then, simulate the tracking process by displaying messages like "Parcel in transit," "Parcel out for delivery," or "Parcel delivered" based on the tracking number's status.

```

private static void Question9()
{
    string[,] parcels = {
        { "TRK123", "In Transit" },
        { "TRK456", "Out for Delivery" },
        { "TRK789", "Delivered" }
    };

    Console.WriteLine("Enter tracking number: ");
    string input = Console.ReadLine();

    bool found = false;

    for (int i = 0; i < parcels.GetLength(0); i++)
    {
        if (parcels[i, 0].Equals(input, StringComparison.OrdinalIgnoreCase))
        {
            Console.WriteLine($"Status: Parcel {parcels[i, 1]}");
            found = true;
            break;
        }
    }

    if (!found)
    {
        Console.WriteLine("Tracking number not found.");
    }
}

```

```

Courier Management System
Enter Question Number to Run (1 to 15):
9
Enter tracking number: TRK456
Status: Parcel Out for Delivery

C:\Users\gokul\OneDrive\Desktop\C#_workspace\CourierManagementSystem\
ManagementTask2-4.exe (process 16604) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable To
le when debugging stops.
Press any key to close this window . . .

```

10. Customer Data Validation: Write a function which takes 2 parameters, data-denotes the data and detail-denotes if it is name address or phone number. Validate customer information based on following criteria. Ensure that names contain only letters and are properly capitalized, addresses do not contain special characters, and phone numbers follow a specific format (e.g., ###-###-####).

```

private static void Question10()
{
    Console.WriteLine("Enter data to validate: ");
    string data = Console.ReadLine();
}

```

```

Console.WriteLine("Enter type (name/address/phone): ");
string detail = Console.ReadLine();

if (detail == "name")
{
    bool valid = true;

    if (!char.IsUpper(data[0]))
        valid = false;

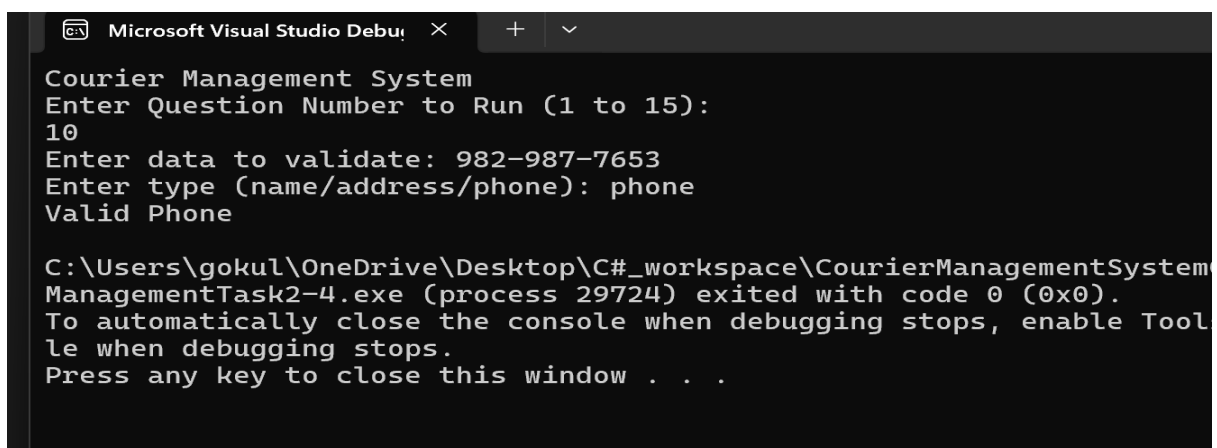
    foreach (char c in data)
    {
        if (!char.IsLetter(c))
            valid = false;
    }

    Console.WriteLine(valid ? "Valid Name" : "Invalid Name");
}
else if (detail == "address")
{
    bool valid = true;

    foreach (char c in data)
    {
        if (!(char.IsLetterOrDigit(c) || c == '-' || c == ','))
            valid = false;
    }

    Console.WriteLine(valid ? "Valid Address" : "Invalid Address");
}
else if (detail == "phone")
{
    bool valid = data.Length == 12 && data[3] == '-' && data[7] == '-';
    Console.WriteLine(valid ? "Valid Phone" : "Invalid Phone");
}
else
{
    Console.WriteLine("Invalid type entered.");
}
}

```



The screenshot shows the Microsoft Visual Studio Debug Console window. The title bar indicates it is a debug console window. The output text is as follows:

```

Courier Management System
Enter Question Number to Run (1 to 15):
10
Enter data to validate: 982-987-7653
Enter type (name/address/phone): phone
Valid Phone

C:\Users\goku\OneDrive\Desktop\C#\workspace\CourierManagementSystem
ManagementTask2-4.exe (process 29724) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tool
le when debugging stops.
Press any key to close this window . . .

```

11. Address Formatting: Develop a function that takes an address as input (street, city, state, zip code) and formats it correctly, including capitalizing the first letter of each word and properly formatting the zip code.

```
private static void Question11()
{
    Console.Write("Enter street: ");
    string street = Console.ReadLine();

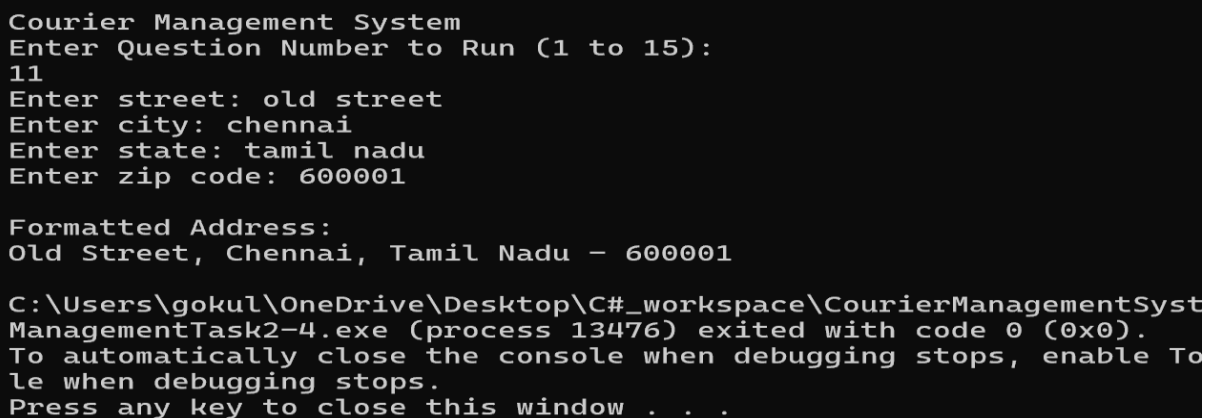
    Console.Write("Enter city: ");
    string city = Console.ReadLine();

    Console.Write("Enter state: ");
    string state = Console.ReadLine();

    Console.Write("Enter zip code: ");
    string zip = Console.ReadLine();

    TextInfo ti = CultureInfo.CurrentCulture.TextInfo;
    street = ti.ToTitleCase(street.ToLower());
    city = ti.ToTitleCase(city.ToLower());
    state = ti.ToTitleCase(state.ToLower());

    if (zip.Length == 6 && int.TryParse(zip, out _))
    {
        Console.WriteLine("\nFormatted Address:");
        Console.WriteLine($"{street}, {city}, {state} - {zip}");
    }
    else
    {
        Console.WriteLine("Invalid zip code. It should be 6 digits.");
    }
}
```



The screenshot shows a console window titled "Courier Management System". It prompts the user to "Enter Question Number to Run (1 to 15):" and the user enters "11". The program then prompts for address details: "Enter street: old street", "Enter city: chennai", "Enter state: tamil nadu", and "Enter zip code: 600001". It then displays the "Formatted Address: Old Street, Chennai, Tamil Nadu - 600001". At the bottom, a Windows error message is visible: "C:\Users\gokul\OneDrive\Desktop\C#_workspace\CourierManagementSystem\ManagementTask2-4.exe (process 13476) exited with code 0 (0x0). To automatically close the console when debugging stops, enable Tools menu | Enable automatic console closure. Press any key to close this window . . .".

12. Order Confirmation Email: Create a program that generates an order confirmation email. The email should include details such as the customer's name, order number, delivery address, and expected delivery date.

```
private static void Question12()
```

```

{
    Console.WriteLine("Enter Customer Name: ");
    string name = Console.ReadLine();

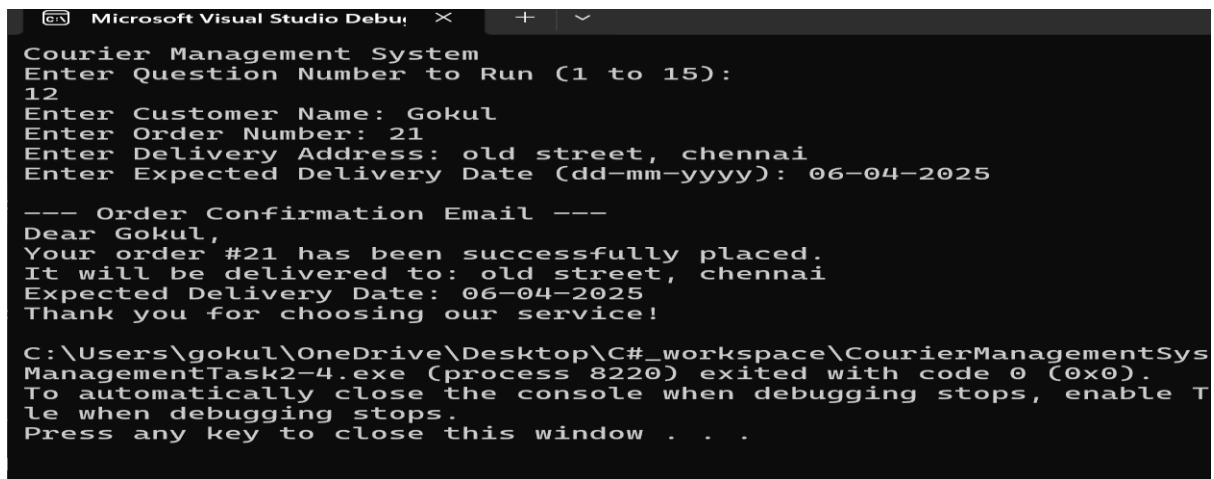
    Console.WriteLine("Enter Order Number: ");
    string orderNo = Console.ReadLine();

    Console.WriteLine("Enter Delivery Address: ");
    string address = Console.ReadLine();

    Console.WriteLine("Enter Expected Delivery Date (dd-mm-yyyy): ");
    string deliveryDate = Console.ReadLine();

    Console.WriteLine("\n--- Order Confirmation Email ---");
    Console.WriteLine($"Dear {name},");
    Console.WriteLine($"Your order #{orderNo} has been successfully placed.");
    Console.WriteLine($"It will be delivered to: {address}");
    Console.WriteLine($"Expected Delivery Date: {deliveryDate}");
    Console.WriteLine("Thank you for choosing our service!");
}

```



```

Microsoft Visual Studio Debug Console
Courier Management System
Enter Question Number to Run (1 to 15):
12
Enter Customer Name: Gokul
Enter Order Number: 21
Enter Delivery Address: old street, chennai
Enter Expected Delivery Date (dd-mm-yyyy): 06-04-2025

--- Order Confirmation Email ---
Dear Gokul,
Your order #21 has been successfully placed.
It will be delivered to: old street, chennai
Expected Delivery Date: 06-04-2025
Thank you for choosing our service!

C:\Users\gokul\OneDrive\Desktop\C#\workspace\CourierManagementSys
ManagementTask2-4.exe (process 8220) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable T
le when debugging stops.
Press any key to close this window . . .

```

13. Calculate Shipping Costs: Develop a function that calculates the shipping cost based on the distance between two locations and the weight of the parcel. You can use string inputs for the source and destination addresses.

```

private static void calculateShippingCost(string source, string destination,
int distance, double weight)
{
    double cost = (distance * 5) + (weight * 10);

    Console.WriteLine($"Shipping from {source} to {destination}");
    Console.WriteLine($"Total shipping cost: {cost}");
}

private static void Question13()
{
    Console.WriteLine("Enter source location: ");
    string source = Console.ReadLine();

```

```

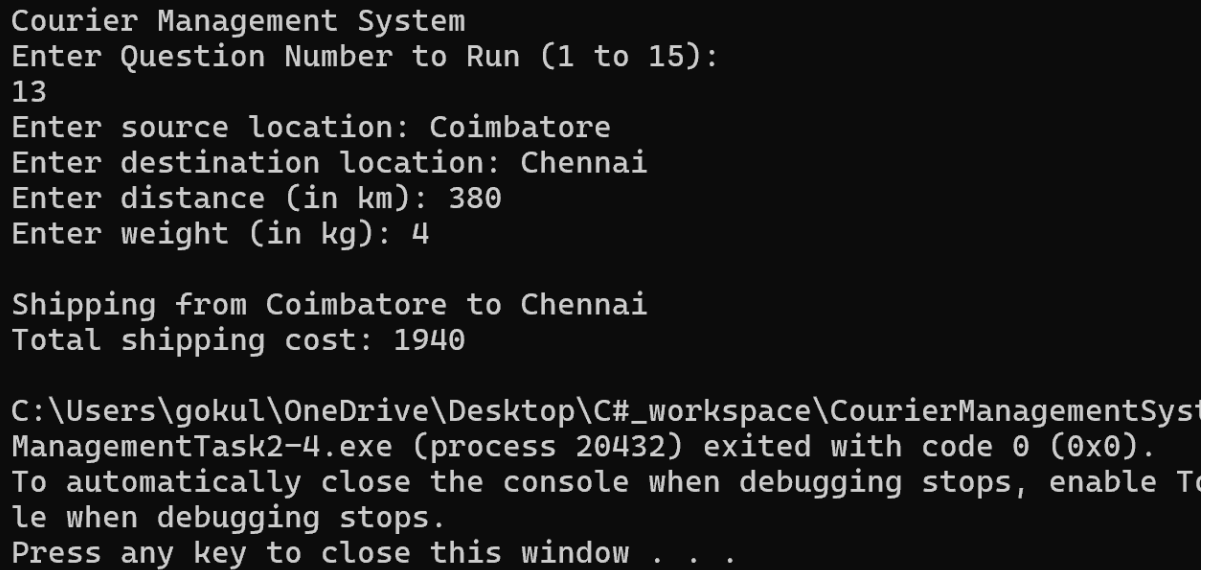
    Console.WriteLine("Enter destination location: ");
    string destination = Console.ReadLine();

    Console.WriteLine("Enter distance (in km): ");
    int distance = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine("Enter weight (in kg): ");
    double weight = Convert.ToDouble(Console.ReadLine());

    calculateShippingCost(source, destination, distance, weight);
}

```



```

Courier Management System
Enter Question Number to Run (1 to 15):
13
Enter source location: Coimbatore
Enter destination location: Chennai
Enter distance (in km): 380
Enter weight (in kg): 4

Shipping from Coimbatore to Chennai
Total shipping cost: 1940

C:\Users\gokul\OneDrive\Desktop\C#_workspace\CourierManagementSystem\
ManagementTask2-4.exe (process 20432) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Te
le when debugging stops.
Press any key to close this window . . .

```

14. Password Generator: Create a function that generates secure passwords for courier system accounts. Ensure the passwords contain a mix of uppercase letters, lowercase letters, numbers, and special characters.

```

private static string generatePassword(int length)
{
    string digits = "0123456789";
    string special = "!@#$%^&*";
    string upper = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    string lower = "abcdefghijklmnopqrstuvwxyz";

    string allCharacters = upper + lower + digits + special;
    Random rand = new Random();
    string password = "";

    for (int i = 0; i < length; i++)
    {
        int index = rand.Next(allCharacters.Length);
        password += allCharacters[index];
    }
}

```

```

        return password;
    }
    private static void Question14()
    {
        Console.WriteLine("Enter desired password length: ");
        int length = Convert.ToInt32(Console.ReadLine());

        string password = generatePassword(length);
        Console.WriteLine("Generated Password: " + password);
    }
}

```

```

Microsoft Visual Studio Debug Console
Courier Management System
Enter Question Number to Run (1 to 15):
14
Enter desired password length: 8
Generated Password: ir@$UkWf

C:\Users\gokul\OneDrive\Desktop\C#_workspace\CourierManagementSystem\
ManagementTask2-4.exe (process 26368) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable To
le when debugging stops.
Press any key to close this window . . .

```

15. Find Similar Addresses: Implement a function that finds similar addresses in the system. This can be useful for identifying duplicate customer entries or optimizing delivery routes. Use string functions to implement this.

```

private static void Question15()
{
    List<string> addresses = new List<string>
    {
        "10 Gandhi Road",
        "22 Nehru Street",
        "22 Nehru St",
        "45 Patel Nagar",
        "78 Anna Salai",
        "91 MG Avenue",
        "91 mg ave"
    };

    for (int i = 0; i < addresses.Count; i++)
    {
        for (int j = i + 1; j < addresses.Count; j++)
        {
            string addr1 = addresses[i].ToLower();
            string addr2 = addresses[j].ToLower();

            if (addr1.Contains(addr2) || addr2.Contains(addr1))
            {

```

```

        Console.WriteLine($"Similar addresses are :\n\n
        \"{addresses[i]}\" \nand\n \"{addresses[j]}\" \n\n");
    }
}
}
}
}

```

```

Courier Management System
Enter Question Number to Run (1 to 15):
15
Similar addresses are :

    "22 Nehru Street"
and
    "22 Nehru St"

Similar addresses are :

    "91 MG Avenue"
and
    "91 mg ave"

C:\Users\gokul\OneDrive\Desktop\C#\workspace\CourierManagementSystem\
ManagementTask2-4.exe (process 10912) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable
the option 'Automatically close console when debugging stops'.
Press any key to close this window . . .

```

Following tasks are incremental stages to build an application and should be done in a single project Task 5: Object Oriented Programming Scope : Entity classes/Models/POJO, Abstraction/Encapsulation Create the following model/entity classes within package entities with variables declared private, constructors(default and parametrized, getters, setters and toString())

1. User Class: Variables: userID , userName , email , password , contactNumber , address
2. Courier Class Variables: courierID , senderName , senderAddress , receiverName , receiverAddress , weight , status, trackingNumber , deliveryDate ,userId
3. Employee Class: Variables employeeID , employeeName , email , contactNumber , role String, salary
4. Location Class Variables LocationID , LocationName , Address
5. CourierCompany Class Variables companyName , courierDetails -collection of Courier Objects, employeeDetails- collection of Employee Objects, locationDetails - collection of Location Objects.

6. Payment Class: Variables PaymentID long, CourierID long, Amount double, PaymentDate Date

Courier class:

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Microsoft.Identity.Client;

namespace CourierManagementSystem.model
{
    public class Courier
    {
        private static int _nextCourierId = 112;
        public int CourierID { get; set; }
        public string? SenderName { get; set; }
        public string? SenderAddress { get; set; }
        public string? ReceiverName { get; set; }
        public string? ReceiverAddress { get; set; }
        public decimal? CourierWeight { get; set; }
        public string? CourierStatus { get; set; }
        public string? TrackingNumber { get; set; }
        public DateTime? DeliveryDate { get; set; }

        //Foreign keys
        public int? UserID { get; set; }
        public int? EmployeeID { get; set; }
        public int? ServiceID { get; set; }
        public int? LocationID { get; set; }

        public Courier() { }
        public Courier(int courierId, string? senderName, string? senderAddress,
            string? receiverName, string? receiverAddress, decimal courierWeight, string?
            courierStatus, string? trackingNumber, DateTime deliveryDate, int userId, int
            employeeId, int serviceId, int locationId)
        {
            CourierID = courierId;
            SenderName = senderName;
            SenderAddress = senderAddress;
            ReceiverName = receiverName;
            ReceiverAddress = receiverAddress;
            CourierWeight = courierWeight;
            CourierStatus = courierStatus;
            TrackingNumber = trackingNumber;
            DeliveryDate = deliveryDate;

            UserID = userId;
            EmployeeID = employeeId;
            ServiceID = serviceId;
            LocationID = locationId;
        }

        public override string ToString()
        {
            return $"CourierID: {CourierID}\n" +
                $"Sender Name: {SenderName}\n" +
                $"Sender Address: {SenderAddress}\n" +
```

```

        $"Receiver Name: {ReceiverName}\n" +
        $"Receiver Address: {ReceiverAddress}\n" +
        $"Courier Weight: {CourierWeight} kg\n" +
        $"Courier Status: {CourierStatus}\n" +
        $"Tracking Number: {TrackingNumber}\n" +
        $"Delivery Date: {DeliveryDate?.ToString("dd-MM-yyyy")}\n" +
        $"UserID: {UserID}\n" +
        $"EmployeeID: {EmployeeID}\n" +
        $"ServiceID: {ServiceID}\n" +
        $"LocationID: {LocationID}";
    }

}
}

```

CourierCompany class:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using CourierManagementSystem.model;

namespace CourierManagementSystem.entity
{
    public class CourierCompany
    {
        public string CompanyName { get; set; }
        public List<Courier> CourierDetails { get; set; }
        public List<Employee> EmployeeDetails { get; set; }
        public List<Location> LocationDetails { get; set; }

        public CourierCompany(string companyName)
        {
            CompanyName = companyName;
            CourierDetails = new List<Courier>();
            EmployeeDetails = new List<Employee>();
            LocationDetails = new List<Location>();
        }
    }
}

```

CourierServices Class:

```

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

namespace CourierManagementSystem.model
{
    public class CourierServices

```

```

    {
        public int ServiceID { get; set; }
        public string? ServiceName { get; set; }
        public decimal? ServiceCost { get; set; }

        public CourierServices(int serviceID, string? serviceName, decimal?
serviceCost)
        {
            ServiceID = serviceID;
            ServiceName = serviceName;
            ServiceCost = serviceCost;
        }
    }
}

```

Employee class:

```

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

namespace CourierManagementSystem.model
{
    public class Employee
    {
        public int EmployeeID { get; set; }
        public string? EmployeeName { get; set; }
        public string? EmployeeEmail { get; set; }
        public string? EmployeeContact { get; set; }
        public string? EmployeeRole { get; set; }
        public decimal EmployeeSalary { get; set; }

        public Employee(int employeeId, string? employeeName, string?
employeeEmail, string? employeeContact, string? employeeRole, decimal
employeeSalary)
        {
            EmployeeID = employeeId;
            EmployeeName = employeeName;
            EmployeeEmail = employeeEmail;
            EmployeeContact = employeeContact;
            EmployeeRole = employeeRole;
            EmployeeSalary = employeeSalary;
        }

        public override string ToString()
        {
            return $"Employee ID: {EmployeeID}, " +
                $"Name: {EmployeeName}, " +
                $"Email: {EmployeeEmail}, " +
                $"Contact: {EmployeeContact}, " +
                $"Role: {EmployeeRole}, " +
                $"Salary: {EmployeeSalary}";
        }
    }
}

```

Location Class:

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

namespace CourierManagementSystem.model
{
    public class Location
    {
        public int LocationId { get; set; }
        public string? LocationName { get; set; }
        public string? LocationAddress { get; set; }

        public Location(int locationId, string? locationName, string?
locationAddress)
        {
            LocationId = locationId;
            LocationName = locationName;
            LocationAddress = locationAddress;
        }

        public override string ToString()
        {
            return $"Location ID: {LocationId}, " +
                $"Name: {LocationName}, " +
                $"Address: {LocationAddress}";
        }
    }
}
```

Payment Class:

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

namespace CourierManagementSystem.model
{
    class Payment
    {
        public int PaymentID { get; set; }
        public int? CourierID { get; set; }
        public int? LocationID { get; set; }
        public int? EmployeeID { get; set; }
        public decimal? Amount { get; set; }
        public DateTime? PaymentDate { get; set; }

        public Payment(int paymentID, int? courierID, int? locationID, int?
employeeID, decimal? amount, DateTime? paymentDate)
        {
            PaymentID = paymentID;
            CourierID = courierID;
        }
    }
}
```

```

        LocationID = locationID;
        EmployeeID = employeeID;
        Amount = amount;
        PaymentDate = paymentDate;
    }

    public override string ToString()
    {
        return $"Payment ID: {PaymentID}, " +
            $"Courier ID: {CourierID}, " +
            $"Location ID: {LocationID}, " +
            $"Employee ID: {EmployeeID}, " +
            $"Amount: {Amount}, " +
            $"Payment Date: {PaymentDate}";
    }
}

```

User Class:

```

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

namespace CourierManagementSystem.model
{
    public class User
    {
        public int UserID { get; set; }
        public string? UserName { get; set; }
        public string? UserEmail { get; set; }
        public string? UserPassword { get; set; }
        public string? UserContact { get; set; }
        public string? UserAddress { get; set; }

        public User(int userID, string? userName, string? userEmail, string?
userPassword, string? userContact, string? userAddress)
        {
            UserID = userID;
            UserName = userName;
            UserEmail = userEmail;
            UserPassword = userPassword;
            UserContact = userContact;
            UserAddress = userAddress;
        }

        public override string ToString()
        {
            return $"User ID: {UserID}, " +
                $"Name: {UserName}, " +
                $"Email: {UserEmail}, " +
                $"Password: {UserPassword}, " +
                $"Contact: {UserContact}, " +
                $"Address: {UserAddress}";
        }
    }
}

```

```
}
```

Task 6: Service Provider Interface /Abstract class

Create 2 Interface /Abstract class ICourierUserService and ICourierAdminService
interface ICourierUserService { // Customer-related functions

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.*/

```
namespace CourierManagementSystem.dao
{
    interface ICourierUserService
    {
        public string PlaceOrder(Courier courier);
        public string GetOrderStatus(string trackingNumber);
        public bool CancelOrder(string trackingNumber);

        public List<Courier> GetAssignedOrder(int employeeId);

        public List<Courier> RetrieveDeliveryHistory(string trackingNumber);

        public Dictionary<int, decimal> GenerateRevenueReportByLocationId();
    }
}
```

```
namespace CourierManagementSystem.dao
{
    interface ICourierAdminService
    {
        int AddCourierStaff(Employee obj);
    }
}
```

Task 7: Exception Handling (Scope: User Defined Exception/Checked /Unchecked Exception/Exception handling using try..catch ,finally,throw & throws keyword usage)
Define the following custom exceptions and throw them in methods whenever needed .
Handle all the exceptions in main method,

1. TrackingNumberNotFoundException :throw this exception when user try to withdraw amount or transfer amount to another acco

2. InvalidEmployeeIdException throw this exception when id entered for the employee not existing in the system

```

}

namespace CourierManagementSystem.exceptions
{
    class InvalidEmployeeIDException:Exception
    {
        public InvalidEmployeeIDException(string msg) : base(msg) { }
    }
}

namespace CourierManagementSystem.exceptions
{
    class TrackingNumberNotFoundException : Exception
    {
        public TrackingNumberNotFoundException(string msg) : base(msg) { }
    }
}

```

Task 8: Service implementation

1.Create CourierUserServiceImpl class which implements ICourierUserService interface which holds a variable named companyObj of type CourierCompany. This variable can be used to access the Object Arrays to access data relevant in method implementations.

2. Create CourierAdminService Impl class which inherits from CourierUserServiceImpl and implements ICourierAdminService interface.

3. Create CourierAdminServiceCollectionImpl class which inherits from CourierUserServiceCollectionImpl and implements ICourierAdminService interface.

```

namespace CourierManagementSystem.dao
{
    interface ICourierUserService
    {
        public string PlaceOrder(Courier courier);
        public string GetOrderStatus(string trackingNumber);
        public bool CancelOrder(string trackingNumber);

        public List<Courier> GetAssignedOrder(int employeeId);
    }
}

```

```

        public List<Courier> RetrieveDeliveryHistory(string trackingNumber);

        public Dictionary<int, decimal> GenerateRevenueReportByLocationId();

    }

}

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using CourierManagementSystem.exceptions;
using CourierManagementSystem.model;
using CourierManagementSystem.util;
using Microsoft.Data.SqlClient;
using Microsoft.Identity.Client;

namespace CourierManagementSystem.dao
{
    class CourierUserServiceImpl : ICourierUserService
    {

        private SqlConnection con; //=
        DBConnUtil.GetConnection("appsettings.json");

        public string PlaceOrder(Courier courier)
        {

            con = DBConnUtil.GetConnection("appsettings.json");
            SqlCommand cmd = con.CreateCommand();

            cmd.CommandText = "insert into
tblCourier(CourierID,SenderName,SenderAddress,ReceiverName," +

"ReceiverAddress,CourierWeight,CourierStatus,TrackingNumber,DeliveryDate,UserID
,EmployeeID,ServiceID,LocationID) " +
            "values
(@CourierID,@SenderName,@SenderAddress,@ReceiverName,@ReceiverAddress,@CourierW
eight,@CourierStatus,@TrackingNumber," +
            "@DeliveryDate,@UserID,@EmployeeID,@ServiceID,@LocationID) ";
            cmd.Connection = con;

            cmd.Parameters.AddWithValue("@CourierID", courier.CourierID);
            cmd.Parameters.AddWithValue("@SenderName", courier.SenderName);
            cmd.Parameters.AddWithValue("@SenderAddress",
courier.SenderAddress);
            cmd.Parameters.AddWithValue("@ReceiverName", courier.ReceiverName);
            cmd.Parameters.AddWithValue("@ReceiverAddress",
courier.ReceiverAddress);
            cmd.Parameters.AddWithValue("@CourierWeight",
courier.CourierWeight);
            cmd.Parameters.AddWithValue("@CourierStatus",
courier.CourierStatus);
            cmd.Parameters.AddWithValue("@TrackingNumber",
courier.TrackingNumber);
            cmd.Parameters.AddWithValue("@DeliveryDate", courier.DeliveryDate);

            cmd.Parameters.AddWithValue("@UserID", courier.UserID);
            cmd.Parameters.AddWithValue("@EmployeeID", courier.EmployeeID);

```



```

cmd.Parameters.AddWithValue("@ServiceID", courier.ServiceID);
cmd.Parameters.AddWithValue("@LocationID", courier.LocationID);

con.Open();
int rows = cmd.ExecuteNonQuery();
con.Close();

if (rows > 0)
{
    return "Value added into Couriers Successfully!";
}
else
{
    return "Problem adding values to Couriers";
}

}
public string GetOrderStatus(string trackingNumber)
{
    con = DBConnUtil.GetConnection("appsettings.json");
    SqlCommand cmd = new SqlCommand("select CourierStatus from
tblCourier where TrackingNumber = @TrackingNumber", con);

    cmd.Parameters.AddWithValue("@TrackingNumber", trackingNumber);

    con.Open();
    string orderStatus = (string)cmd.ExecuteScalar();
    con.Close();
    if (orderStatus == null)
    {
        throw (new TrackingNumberNotFoundException("Tracking Number Not
Found ! "));
    }

    return orderStatus;
}
public bool CancelOrder(string trackingNumber)
{
    con = DBConnUtil.GetConnection("appsettings.json");
    SqlCommand cmd = new SqlCommand("update tblCourier set
CourierStatus = 'Order Cancelled' where TrackingNumber = @trackingNumber",
con);

    cmd.Parameters.AddWithValue("@trackingNumber", trackingNumber);

    con.Open();
    int rows = cmd.ExecuteNonQuery();
    con.Close();

    if(rows == 0)
    {
        throw new TrackingNumberNotFoundException("Tracking Number not
found for Cancelling the Courier Order");
    }
    return true;
}

public List<Courier> GetAssignedOrder(int employeeId)
{
    List<Courier> assignedCouriers = new List<Courier>();

```

```

        con = DBConnUtil.GetConnection("appsettings.json");

        SqlCommand cmd = new SqlCommand("Select * from tblCourier where
EmployeeID = @employeeId", con);
        cmd.Parameters.AddWithValue("@employeeId", employeeId);

        con.Open();
        SqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Courier courier = new Courier(

                Convert.ToInt32(dr["CourierID"]),
                Convert.ToString(dr["SenderName"]),
                Convert.ToString(dr["SenderAddress"]),
                Convert.ToString(dr["ReceiverName"]),
                Convert.ToString(dr["ReceiverAddress"]),
                Convert.ToDecimal(dr["CourierWeight"]),
                Convert.ToString(dr["CourierStatus"]),
                Convert.ToString(dr["TrackingNumber"]),
                Convert.ToDateTime(dr["DeliveryDate"]),
                Convert.ToInt32(dr["UserID"]),
                Convert.ToInt32(dr["EmployeeID"]),
                Convert.ToInt32(dr["ServiceID"]),
                Convert.ToInt32(dr["LocationID"])
            );

            assignedCouriers.Add(courier);
        }

        con.Close();

        if (assignedCouriers.Count == 0) throw new
InvalidEmployeeIDException("Entered EmployeeID is Invalid!");

        return assignedCouriers;
    }

    public List<Courier> RetrieveDeliveryHistory(string trackingNumber)
    {
        List<Courier> history = new List<Courier>();

        con = DBConnUtil.GetConnection("appsettings.json");
        SqlCommand cmd = new SqlCommand("SELECT * FROM tblCourier WHERE
TrackingNumber = @TrackingNumber", con);
        cmd.Parameters.AddWithValue("@TrackingNumber", trackingNumber);

        con.Open();
        SqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Courier courier = new Courier(
                Convert.ToInt32(dr["CourierID"]),
                Convert.ToString(dr["SenderName"]),
                Convert.ToString(dr["SenderAddress"]),
                Convert.ToString(dr["ReceiverName"]),
                Convert.ToString(dr["ReceiverAddress"]),
                Convert.ToDecimal(dr["CourierWeight"]),

```

```

        Convert.ToString(dr["CourierStatus"]),
        Convert.ToString(dr["TrackingNumber"]),
        Convert.ToDateTime(dr["DeliveryDate"]),
        Convert.ToInt32(dr["UserID"]),
        Convert.ToInt32(dr["EmployeeID"]),
        Convert.ToInt32(dr["ServiceID"]),
        Convert.ToInt32(dr["LocationID"])
    );

    history.Add(courier);
}

dr.Close();
con.Close();

if (history.Count == 0)
{
    throw new TrackingNumberNotFoundException("Tracking Number Not
Found!");
}

return history;
}

public Dictionary<int, decimal> GenerateRevenueReportByLocationId()
{
    Dictionary<int, decimal> revenueByLocation = new Dictionary<int,
decimal>();

    con = DBConnUtil.GetConnection("appsettings.json");
    string query = @"SELECT LocationID, SUM(Amount) as TotalRevenue
FROM tblPayment
GROUP BY LocationID";

    SqlCommand cmd = new SqlCommand(query, con);

    try
    {
        con.Open();
        SqlDataReader reader = cmd.ExecuteReader();

        while (reader.Read())
        {
            int locationId = Convert.ToInt32(reader["LocationID"]);
            decimal revenue =
Convert.ToDecimal(reader["TotalRevenue"]);

            revenueByLocation.Add(locationId, revenue);
        }
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error fetching revenue by location: " +
ex.Message);
    }
    finally
    {
        con.Close();
    }

    return revenueByLocation;
}

```

```

    }
}

```

```

namespace CourierManagementSystem.dao
{
    interface ICourierAdminService
    {
        int AddCourierStaff(Employee obj);
    }
}

```

```

namespace CourierManagementSystem.dao
{
    class CourierAdminServiceImpl:CourierUserServiceImpl,ICourierAdminService
    {
        private SqlConnection con;
        public int AddCourierStaff(Employee obj)
        {
            con = DBConnUtil.GetConnection("appsettings.json");

            SqlCommand cmd = new SqlCommand("insert into tblEmployee
values(@EmployeeID,@EmployeeName,@EmployeeEmail," +
            "@EmployeeContact,@EmployeeRole,@EmployeeSalary)", con);

            cmd.Parameters.AddWithValue("@EmployeeID", obj.EmployeeID);
            cmd.Parameters.AddWithValue("@EmployeeName", obj.EmployeeName);
            cmd.Parameters.AddWithValue("@EmployeeEmail", obj.EmployeeEmail);
            cmd.Parameters.AddWithValue("@EmployeeContact",
obj.EmployeeContact);
            cmd.Parameters.AddWithValue("@EmployeeRole", obj.EmployeeRole);
            cmd.Parameters.AddWithValue("@EmployeeSalary", obj.EmployeeSalary);

            con.Open();
            int rows = cmd.ExecuteNonQuery();
            con.Close();

            if(rows==0)
            {
                Console.WriteLine("Cannot Add Employee");
                return -1;
            }

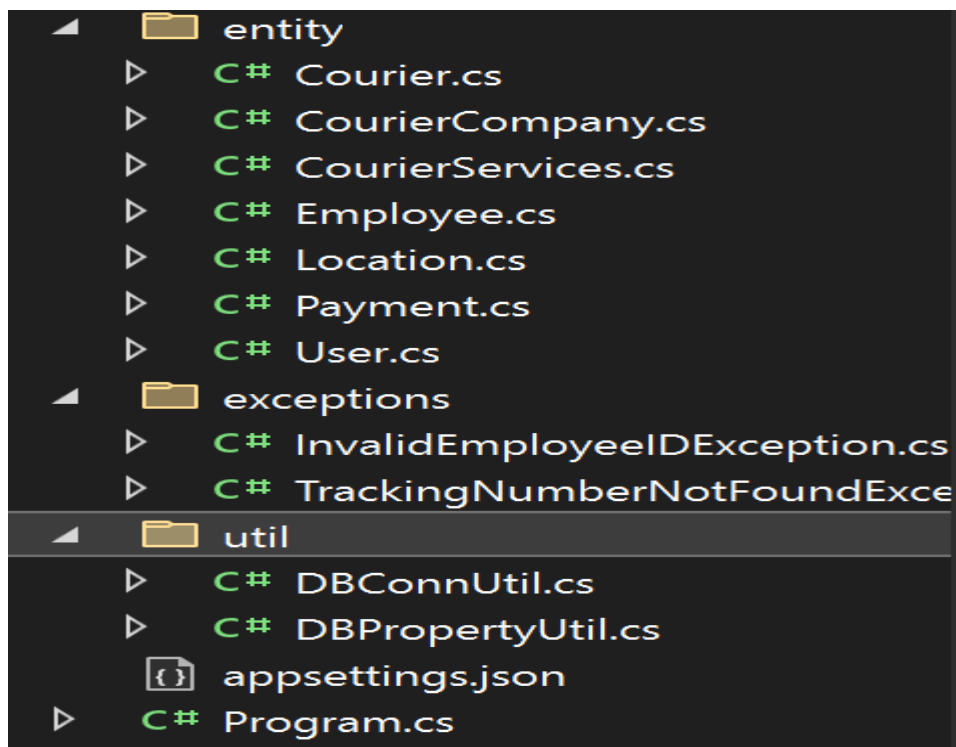
            return obj.EmployeeID;
        }
    }
}

```

Task 9: Database Interaction Connect your application to the SQL database for the Courier Management System

1. Write code to establish a connection to your SQL database. Create a class **DBConnection** in a package **connectionutil** with a static variable **connection** of Type **Connection** and a static method **getConnection()** which returns **connection**. **Connection** properties supplied in the connection string should be read from a property file.
2. Create a Service class **CourierServiceDb** in **dao** with a static variable named **connection** of type **Connection** which can be assigned in the constructor by invoking the method in **DBConnection** Class.
3. Include methods to insert, update, and retrieve data from the database (e.g., inserting a new order, updating courier status).
4. Implement a feature to retrieve and display the delivery history of a specific parcel by querying the database. 1. Generate and display reports using data retrieved from the database (e.g., shipment status report, revenue report).

Folder structure:



Appsettings.json:

```

{
  "ConnectionStrings": {
    "DefaultConnection": "data source = GVIVOBOKPRO14; initial catalog = dbCourierManagement; integrated security = true;Encrypt = false"
  }
}
  
```

```

namespace CourierManagementSystem.util
{
    static class DBPropertyUtil
  
```

```

    {
        public static string GetConnectionString(string fileName)
        {
            var builder = new
ConfigurationBuilder().SetBasePath(Directory.GetCurrentDirectory()).AddJsonFile
(fileName);
            IConfiguration config = builder.Build();

            string conStr = config.GetConnectionString("DefaultConnection");

            return conStr;
        }
    }
}

```

```

namespace CourierManagementSystem.util
{
    static class DBConnUtil
    {
        public static SqlConnection? GetConnection(string fileName)
        {
            try
            {
                string? conString =
DBPropertyUtil.GetConnectionString(fileName);
                return new SqlConnection(conString);
            }
            catch (InvalidOperationException e)
            {
                Console.WriteLine("Cannot create connection object " +
e.Message);
                return null;
            }
        }
    }
}

```

Program.cs:

```

using CourierManagementSystem.dao;
using CourierManagementSystem.exceptions;
using CourierManagementSystem.model;

namespace CourierManagementSystem
{
    internal class Program
    {
        private ICourierUserService _userService = new
CourierUserServiceImpl();
        private ICourierAdminService _adminService = new
CourierAdminServiceImpl();

        static void Main(string[] args)
        {

```

```

Program program = new Program();
bool exit = false;

while (!exit)
{
    Console.WriteLine("\n==== Courier Management System ====");
    Console.WriteLine("1. Place a Courier Order");
    Console.WriteLine("2. Track Courier Status");
    Console.WriteLine("3. Cancel a Courier Order");
    Console.WriteLine("4. Add Courier Staff");
    Console.WriteLine("5. View Assigned Orders for Employee");
    Console.WriteLine("6. Retrieve Delivery History by Tracking
Number");

    Console.WriteLine("7. Generate Revenue Report by Location ID");
    Console.WriteLine("8. Exit");
    Console.Write("Enter your choice: ");

    string choice = Console.ReadLine();

    switch (choice)
    {
        case "1":
            program.PlaceOrder();
            break;
        case "2":
            program.GetOrderStatus();
            break;
        case "3":
            program.CancelOrder();
            break;
        case "4":
            program.AddCourierStaff();
            break;
        case "5":
            program.GetAssignedOrders();
            break;
        case "6":
            program.RetrieveDeliveryHistory();
            break;
        case "7":
            program.GenerateRevenueReportByLocationId();
            break;
        case "8":
            exit = true;
            Console.WriteLine("Exiting the system. Thank you!");
            break;
        default:
            Console.WriteLine("Invalid choice. Please try again.");
            break;
    }
}

```

```

private void PlaceOrder()
{
    Console.WriteLine("Enter CourierId: ");
    int courierId = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine("Enter sender name:");
    string? senderName = Console.ReadLine();

    Console.WriteLine("Enter sender address:");
}

```

```

        string? senderAddress = Console.ReadLine();

        Console.WriteLine("Enter receiver name:");
        string? receiverName = Console.ReadLine();

        Console.WriteLine("Enter receiver address:");
        string? receiverAddress = Console.ReadLine();

        Console.WriteLine("Enter weight:");
        decimal courierWeight = Convert.ToDecimal(Console.ReadLine());

        Console.WriteLine("Enter status:");
        string? courierStatus = Console.ReadLine();

        Console.WriteLine("Enter Tracking Number:");
        string? trackingNumber = Console.ReadLine();

        Console.WriteLine("Enter delivery date (yyyy-mm-dd):");
        DateTime deliveryDate = Convert.ToDateTime(Console.ReadLine());

        Console.WriteLine("Enter user ID:");
        int userId = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine("Enter employee Id");
        int employeeId = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine("Enter service Id");
        int serviceId = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine("Enter location ID:");
        int locationId = Convert.ToInt32(Console.ReadLine());

        Courier courier = new Courier(courierId, senderName, senderAddress,
receiverName, receiverAddress, courierWeight, courierStatus, trackingNumber,
deliveryDate, userId, employeeId, serviceId, locationId);

        Console.WriteLine(_userService.PlaceOrder(courier));
    }

    private void GetOrderStatus()
    {
        Console.WriteLine("Enter the Tracking number to retrieve Courier
Status: ");
        String trackingNumber = Console.ReadLine();

        try
        {
            string orderStatus =
_userService.GetOrderStatus(trackingNumber);
            Console.WriteLine($"The Order status for Tracking number:
{trackingNumber} is: {orderStatus}");
        }
        catch (TrackingNumberNotFoundException te)
        {
            Console.WriteLine(te.Message);
        }
    }

    private void CancelOrder()
    {

```



```

        Console.WriteLine("Enter the tracking number for cancelling the
order: ");
        string trackingNumber = Console.ReadLine();
        try
        {
            if(_userService.CancelOrder(trackingNumber)) Console.WriteLine("The Order is
Cancelled! ");
        }
        catch(TrackingNumberNotFoundException te)
        {
            Console.WriteLine(te.Message);
        }
    }

    private void AddCourierStaff()
    {
        Console.WriteLine("Enter the EmployeeId: ");
        int employeeId = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine("Enter the Employee Name: ");
        string? employeeName = Console.ReadLine();

        Console.WriteLine("Enter the Employee email: ");
        string? employeeEmail = Console.ReadLine();

        Console.WriteLine("Enter the Employee Contact: ");
        string? employeeContact = Console.ReadLine();

        Console.WriteLine("Enter the Employee Role: ");
        string? employeeRole = Console.ReadLine();

        Console.WriteLine("Enter the Employee Salary: ");
        decimal employeeSalary = Convert.ToDecimal(Console.ReadLine());

        Employee e = new Employee(employeeId, employeeName, employeeEmail,
employeeContact, employeeRole, employeeSalary);

        int EmployeeId = _adminService.AddCourierStaff(e);

        if(EmployeeId!=-1) Console.WriteLine($"Courier Staff of EmployeeID
{EmployeeId} added successfully");
    }

    private void GetAssignedOrders()
    {
        Console.WriteLine("Enter the Employee Id to List orders assigned:
");
        int employeeId = Convert.ToInt32(Console.ReadLine());

        try
        {
            List<Courier> assignedOrders =
_userService.GetAssignedOrder(employeeId);

            foreach (var assignedOrder in assignedOrders)
            {
                Console.WriteLine("CourierID : " + assignedOrder.CourierID
+ " SenderName : " + assignedOrder.SenderName + " ReceiverName : " +
assignedOrder.ReceiverName + " EmployeeID : " + assignedOrder.EmployeeID);
            }
        }
    }

```

```

        catch(InvalidEmployeeIDException ieid)
        {
            Console.WriteLine(ieid.Message);
        }

    }

    private void RetrieveDeliveryHistory()
    {
        try
        {
            Console.WriteLine("Enter the Tracking Number to view delivery
history: ");
            string trackingNumber = Console.ReadLine();

            List<Courier> historyList =
            _userService.RetrieveDeliveryHistory(trackingNumber);

            foreach (var courier in historyList)
            {
                Console.WriteLine("CourierID: " + courier.CourierID +
                                " SenderName: " + courier.SenderName +
                                " ReceiverName: " + courier.ReceiverName
+
                                " Status: " + courier.CourierStatus +
                                " DeliveryDate: " +
courier.DeliveryDate);
            }
        }
        catch (TrackingNumberNotFoundException ex)
        {
            Console.WriteLine(ex.Message);
        }
    }

    private void GenerateRevenueReportByLocationId()
    {
        Dictionary<int, decimal> revenueReport =
        _userService.GenerateRevenueReportByLocationId();

        Console.WriteLine("\n--- Revenue by Location ID ---");
        foreach (var entry in revenueReport)
        {
            Console.WriteLine($"Location ID: {entry.Key}, Revenue:
{entry.Value}");
        }
    }

}

```

```

Enter receiver address:
Coimbatore, Tamil Nadu
Enter weight:
3.7
Enter status:
Processing
Enter Tracking Number:
TN34598
Enter delivery date (yyyy-mm-dd):
2025-04-06
Enter user ID:
3
Enter employee Id
3
Enter service Id
3
Enter location ID:
3
Value added into Couriers Successfully!

```

	CourierID	SenderName	SenderAddress	ReceiverName	ReceiverAddress	CourierWeight	CourierStatus	TrackingNumber	DeliveryDate	UserID	EmployeeID	ServiceID	LocationID
1	101	Arun Kumar	Chennai, Tamil Nadu	Deepak R	Madurai, Tamil Nadu	2.50	In Transit	TN12345	2025-03-25	1	1	2	4
2	102	Priya Sharma	Coimbatore, Tamil Nadu	Anjali K	Chennai, Tamil Nadu	1.20	Delivered	TN67890	2025-03-18	2	2	3	1
3	103	Dinesh Kumar	Madurai, Tamil Nadu	Ravi T	Chennai, Tamil Nadu	3.10	Pending	TN11223	2025-03-27	3	3	1	3
4	104	Sowmya R	Trichy, Tamil Nadu	Sneha P	Coimbatore, Tamil Nadu	2.00	In Transit	TN33445	2025-03-22	4	4	4	4
5	105	Vignesh B	Salem, Tamil Nadu	Meera S	Madurai, Tamil Nadu	1.50	Processing	TN55667	2025-03-21	5	5	5	5
6	106	Karthikeyan	Chennai, Tamil Nadu	Deepak R	Madurai, Tamil Nadu	3.70	In Transit	TN12445	2025-03-25	1	1	2	4
7	107	Newton	Trichy, Tamil Nadu	Einstein	Coimbatore, Tamil Nadu	0.90	In Transit	TN33445	2025-03-22	4	4	4	2
8	108	Vikash	Madurai, Tamil Nadu	Ragul	Chennai, Tamil Nadu	3.10	delivered	TN63824	2025-03-22	3	3	1	3
9	109	Prashanth	Trichy, Tamil Nadu	Sneha P	Coimbatore, Tamil Nadu	2.00	Order Cancelled	TN65345	2025-03-20	4	4	4	3
10	110	Priya	Salem, Tamil Nadu	Shankar	Madurai, Tamil Nadu	1.50	delivered	TN90809	2025-03-23	5	5	NULL	5
11	111	Prashanth	Trichy, Tamil Nadu	Prem	Coimbatore, Tamil Nadu	2.00	Order Cancelled	TN65345	2025-03-20	4	4	4	3
12	112	Gokul	Coimbatore, Tamil Nadu	Ravi	Salem, Tamil Nadu	5.00	In Transit	TN27374	2025-04-04	2	2	2	2
13	113	raja	Trichy, Tamil Nadu	Harish	Coimbatore, Tamil Nadu	3.70	Processing	TN34598	2025-04-06	3	3	3	3

```

==== Courier Management System ====
1. Place a Courier Order
2. Track Courier Status
3. Cancel a Courier Order
4. Add Courier Staff
5. View Assigned Orders for Employee
6. Retrieve Delivery History by Tracking Number
7. Generate Revenue Report by Location ID
8. Exit
Enter your choice: 2
Enter the Tracking number to retrieve Courier Status:
TN34598
The Order status for Tracking number: TN34598 is: Processing

```

12	112	Gokul	Coimbatore, Tamil Nadu	Ravi	Salem, Tamil Nadu	5.00	In Transit	TN27374	2025-04-04	2	2	2	2
13	113	raja	Trichy, Tamil Nadu	Harish	Coimbatore, Tamil Nadu	3.70	Processing	TN34598	2025-04-06	3	3	3	3

==== Courier Management System ====

1. Place a Courier Order
2. Track Courier Status
3. Cancel a Courier Order
4. Add Courier Staff
5. View Assigned Orders for Employee
6. Retrieve Delivery History by Tracking Number
7. Generate Revenue Report by Location ID
8. Exit

Enter your choice: 3

Enter the tracking number for cancelling the order:

TN34598

The Order is Cancelled!

11	111	Prashanth	Trichy, Tamil Nadu	Prem	Coimbatore, Tamil Nadu	2.00	Order Cancelled	TN65345	2025-03-20	4	4	4	3
12	112	Gokul	Coimbatore, Tamil Nadu	Ravi	Salem, Tamil Nadu	5.00	In Transit	TN27374	2025-04-04	2	2	2	2
13	113	raja	Trichy, Tamil Nadu	Harish	Coimbatore, Tamil Nadu	3.70	Order Cancelled	TN34598	2025-04-06	3	3	3	3

```

==== Courier Management System ====
1. Place a Courier Order
2. Track Courier Status
3. Cancel a Courier Order
4. Add Courier Staff
5. View Assigned Orders for Employee
6. Retrieve Delivery History by Tracking Number
7. Generate Revenue Report by Location ID
8. Exit
Enter your choice: 4
Enter the EmployeeId:
9
Enter the Employee Name:
Vikash
Enter the Employee email:
vikashram@gmail.com
Enter the Employee Contact:
7878945678
Enter the Employee Role:
Manager
Enter the Employee Salary:
60000
Courier Staff of EmployeeID 9 added successfully

```

Results Messages

	EmployeeID	EmployeeName	EmployeeEmail	EmployeeContact	EmployeeRole	EmployeeSalary
1	1	Vikram R	vikram.r@example.com	9876541230	Delivery Agent	25000.00
2	2	Meena L	meena.l@example.com	9845123456	Manager	50000.00
3	3	Karthik S	karthik.s@example.com	9785612345	Delivery Agent	27000.00
4	4	Lavanya M	lavanya.m@example.com	9823456789	Supervisor	60000.00
5	5	Senthil K	senthil.k@example.com	9798765432	Delivery Agent	26000.00
6	6	John	JohnCena@example.com	9798765431	Supervisor	55000.00
7	7	Johnson	Johnson@example.com	9798765430	Manager	52000.00
8	8	ram	ram@gmail.com	6565587877	Delivery Agent	23000.00
9	9	Vikash	vikashram@gmail.com	7878945678	Manager	60000.00

```

==== Courier Management System ====
1. Place a Courier Order
2. Track Courier Status
3. Cancel a Courier Order
4. Add Courier Staff
5. View Assigned Orders for Employee
6. Retrieve Delivery History by Tracking Number
7. Generate Revenue Report by Location ID
8. Exit
Enter your choice: 5
Enter the Employee Id to List orders assigned:
3
CourierID : 103 SenderName : Dinesh Kumar ReceiverName : Ravi T EmployeeID : 3
CourierID : 108 SenderName : Vikash ReceiverName : Ragul EmployeeID : 3
CourierID : 113 SenderName : raja ReceiverName : Harish EmployeeID : 3

```

	CourierID	SenderName	SenderAddress	ReceiverName	ReceiverAddress	CourierWeight	CourierStatus	TrackingNumber	DeliveryDate	UserID	EmployeeID	ServiceID	LocationID
1	101	Arun Kumar	Chennai, Tamil Nadu	Deepak R	Madurai, Tamil Nadu	2.50	In Transit	TN12345	2025-03-25	1	1	2	4
2	102	Priya Sharma	Coimbatore, Tamil Nadu	Anjali K	Chennai, Tamil Nadu	1.20	Delivered	TN67890	2025-03-18	2	2	3	1
3	103	Dinesh Kumar	Madurai, Tamil Nadu	Ravi T	Chennai, Tamil Nadu	3.10	Pending	TN11223	2025-03-27	3	3	1	3
4	104	Sowmya R	Trichy, Tamil Nadu	Sneha P	Coimbatore, Tamil Nadu	2.00	In Transit	TN33445	2025-03-22	4	4	4	4
5	105	Vignesh B	Salem, Tamil Nadu	Meera S	Madurai, Tamil Nadu	1.50	Processing	TN55667	2025-03-21	5	5	5	5
6	106	Karthikeyan	Chennai, Tamil Nadu	Deepak R	Madurai, Tamil Nadu	3.70	In Transit	TN12445	2025-03-25	1	1	2	4
7	107	Newton	Trichy, Tamil Nadu	Einstein	Coimbatore, Tamil Nadu	0.90	In Transit	TN33445	2025-03-22	4	4	4	2
8	108	Vikash	Madurai, Tamil Nadu	Ragul	Chennai, Tamil Nadu	3.10	delivered	TN63824	2025-03-22	3	3	1	3
9	109	Prashanth	Trichy, Tamil Nadu	Sneha P	Coimbatore, Tamil Nadu	2.00	Order Cancelled	TN65345	2025-03-20	4	4	4	3
10	110	Priya	Salem, Tamil Nadu	Shankar	Madurai, Tamil Nadu	1.50	delivered	TN90809	2025-03-23	5	5	NULL	5
11	111	Prashanth	Trichy, Tamil Nadu	Prem	Coimbatore, Tamil Nadu	2.00	Order Cancelled	TN65345	2025-03-20	4	4	4	3
12	112	Gokul	Coimbatore, Tamil Nadu	Ravi	Salem, Tamil Nadu	5.00	In Transit	TN27374	2025-04-04	2	2	2	2
13	113	raja	Trichy, Tamil Nadu	Harish	Coimbatore, Tamil Nadu	3.70	Order Cancelled	TN34598	2025-04-06	3	3	3	3

```

==== Courier Management System ====
1. Place a Courier Order
2. Track Courier Status
3. Cancel a Courier Order
4. Add Courier Staff
5. View Assigned Orders for Employee
6. Retrieve Delivery History by Tracking Number
7. Generate Revenue Report by Location ID
8. Exit
Enter your choice: 6
Enter the Tracking Number to view delivery history:
TN63824
CourierID: 108 SenderName: Vikash ReceiverName: Ragul Status: delivered DeliveryDate: 22-03-2025 00:00:00

```

```
select * from tblCourier
select * from tblLocation
```

100 %

Results Messages

	CourierID	SenderName	SenderAddress	ReceiverName	ReceiverAddress	CourierWeight	CourierStatus	TrackingNumber	DeliveryDate	UserID	EmployeeID	ServiceID	LocationID
1	101	Arun Kumar	Chennai, Tamil Nadu	Deepak R	Madurai, Tamil Nadu	2.50	In Transit	TN12345	2025-03-25	1	1	2	4
2	102	Priya Sharma	Coimbatore, Tamil Nadu	Anjali K	Chennai, Tamil Nadu	1.20	Delivered	TN67890	2025-03-18	2	2	3	1
3	103	Dinesh Kumar	Madurai, Tamil Nadu	Ravi T	Chennai, Tamil Nadu	3.10	Pending	TN11223	2025-03-27	3	3	1	3
4	104	Sowmya R	Trichy, Tamil Nadu	Sneha P	Coimbatore, Tamil Nadu	2.00	In Transit	TN33445	2025-03-22	4	4	4	4
5	105	Vignesh B	Salem, Tamil Nadu	Meera S	Madurai, Tamil Nadu	1.50	Processing	TN55667	2025-03-21	5	5	5	5
6	106	Karthikeyan	Chennai, Tamil Nadu	Deepak R	Madurai, Tamil Nadu	3.70	In Transit	TN12445	2025-03-25	1	1	2	4
7	107	Newton	Trichy, Tamil Nadu	Einstein	Coimbatore, Tamil Nadu	0.90	In Transit	TN33445	2025-03-22	4	4	4	2
8	108	Vikash	Madurai, Tamil Nadu	Ragul	Chennai, Tamil Nadu	3.10	delivered	TN63824	2025-03-22	3	3	1	3
9	109	Prashanth	Trichy, Tamil Nadu	Sneha P	Coimbatore, Tamil Nadu	2.00	Order Cancelled	TN65345	2025-03-20	4	4	4	3
10	110	Priya	Salem, Tamil Nadu	Shankar	Madurai, Tamil Nadu	1.50	delivered	TN90809	2025-03-23	5	5	NULL	5
11	111	Prashanth	Trichy, Tamil Nadu	Prem	Coimbatore, Tamil Nadu	2.00	Order Cancelled	TN65345	2025-03-20	4	4	4	3
12	112	Gokul	Coimbatore, Tamil Nadu	Ravi	Salem, Tamil Nadu	5.00	In Transit	TN27374	2025-04-04	2	2	2	2
13	113	raja	Trichy, Tamil Nadu	Harish	Coimbatore, Tamil Nadu	3.70	Order Cancelled	TN34598	2025-04-06	3	3	3	3

```
==== Courier Management System ====
1. Place a Courier Order
2. Track Courier Status
3. Cancel a Courier Order
4. Add Courier Staff
5. View Assigned Orders for Employee
6. Retrieve Delivery History by Tracking Number
7. Generate Revenue Report by Location ID
8. Exit
Enter your choice: 7

--- Revenue by Location ID ---
Location ID: 1, Revenue: 5300.00
Location ID: 2, Revenue: 6000.00
Location ID: 3, Revenue: 100.00
Location ID: 4, Revenue: 1500.00
Location ID: 5, Revenue: 150.00
```

Results Messages

	PaymentID	CourierID	locationID	Amount	PaymentDate	EmployeeID
1	1	101	1	3000.00	2025-03-24	5
2	2	102	2	6000.00	2025-03-18	2
3	3	103	3	100.00	2025-03-26	6
4	4	104	4	500.00	2025-03-22	4
5	5	105	5	75.00	2025-03-01	1
6	6	106	1	1100.00	2025-03-20	3
7	7	107	4	500.00	2025-03-21	1
8	8	108	5	75.00	2025-03-15	3
9	9	109	1	1200.00	2025-03-20	2
10	10	110	4	500.00	2025-03-20	2