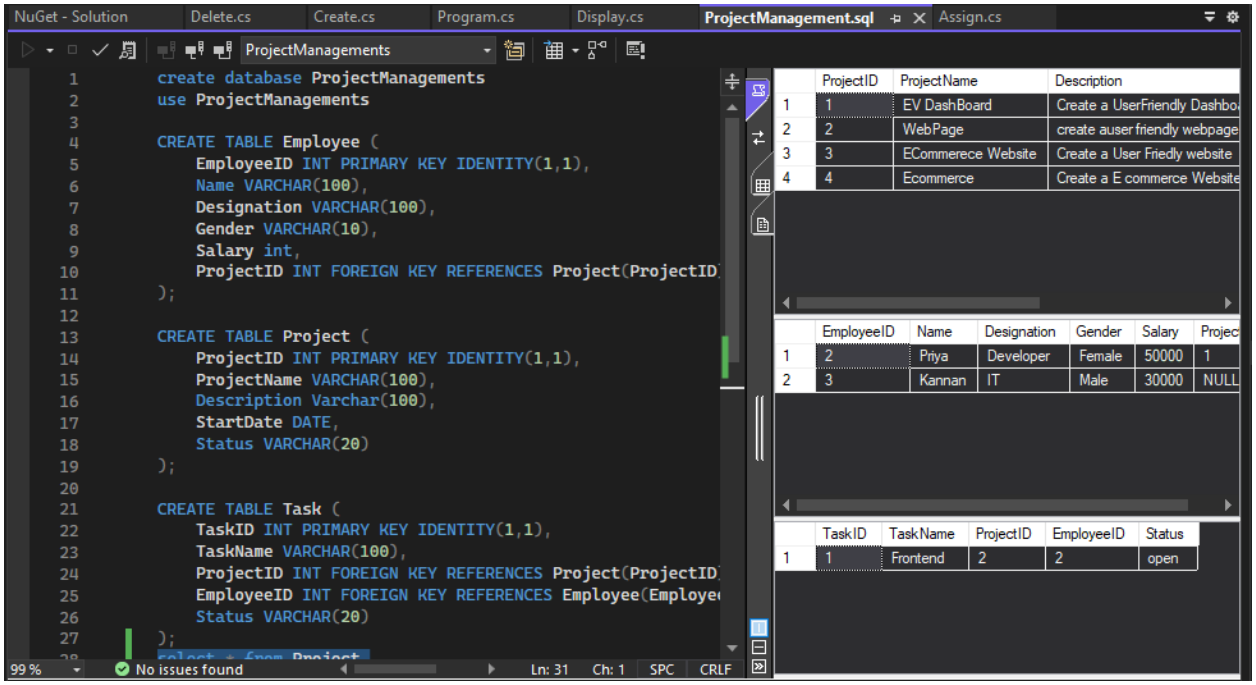
CASE STUDY PROJECT MANAGEMENT

CREATE DATABASE SCHEMA :



```
create database ProjectManagements
use ProjectManagements

CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY IDENTITY(1,1),
    Name VARCHAR(100),
    Designation VARCHAR(100),
    Gender VARCHAR(10),
    Salary int,
    ProjectID INT FOREIGN KEY REFERENCES Project(ProjectID)
);

CREATE TABLE Project (
    ProjectID INT PRIMARY KEY IDENTITY(1,1),
    ProjectName VARCHAR(100),
    Description Varchar(100),
    StartDate DATE,
    Status VARCHAR(20)
);

CREATE TABLE Task (
    TaskID INT PRIMARY KEY IDENTITY(1,1),
    TaskName VARCHAR(100),
    ProjectID INT FOREIGN KEY REFERENCES Project(ProjectID),
    EmployeeID INT FOREIGN KEY REFERENCES Employee(Employee
    Status VARCHAR(20)
);
```

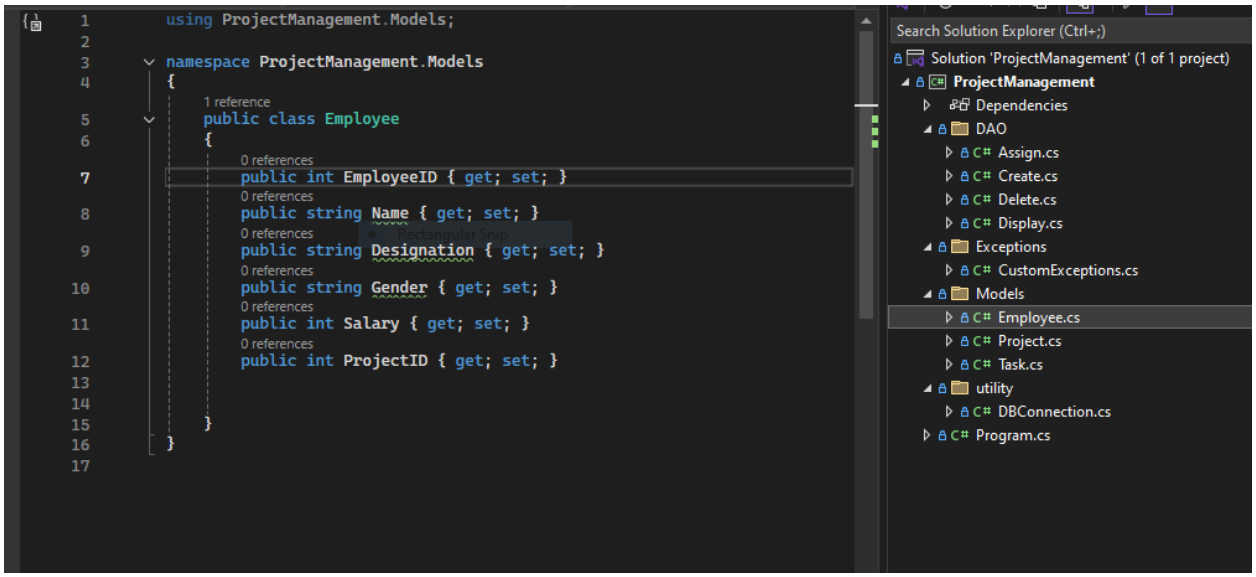
ProjectID	ProjectName	Description
1	EV DashBoard	Create a UserFriendly Dashbo
2	WebPage	create auser friendly webpage
3	ECommerece Website	Create a User Friedly website
4	Ecommerce	Create a E commerce Website

EmployeeID	Name	Designation	Gender	Salary	Projec
2	Priya	Developer	Female	50000	1
3	Kannan	IT	Male	30000	NULL

TaskID	TaskName	ProjectID	EmployeeID	Status
1	Frontend	2	2	open

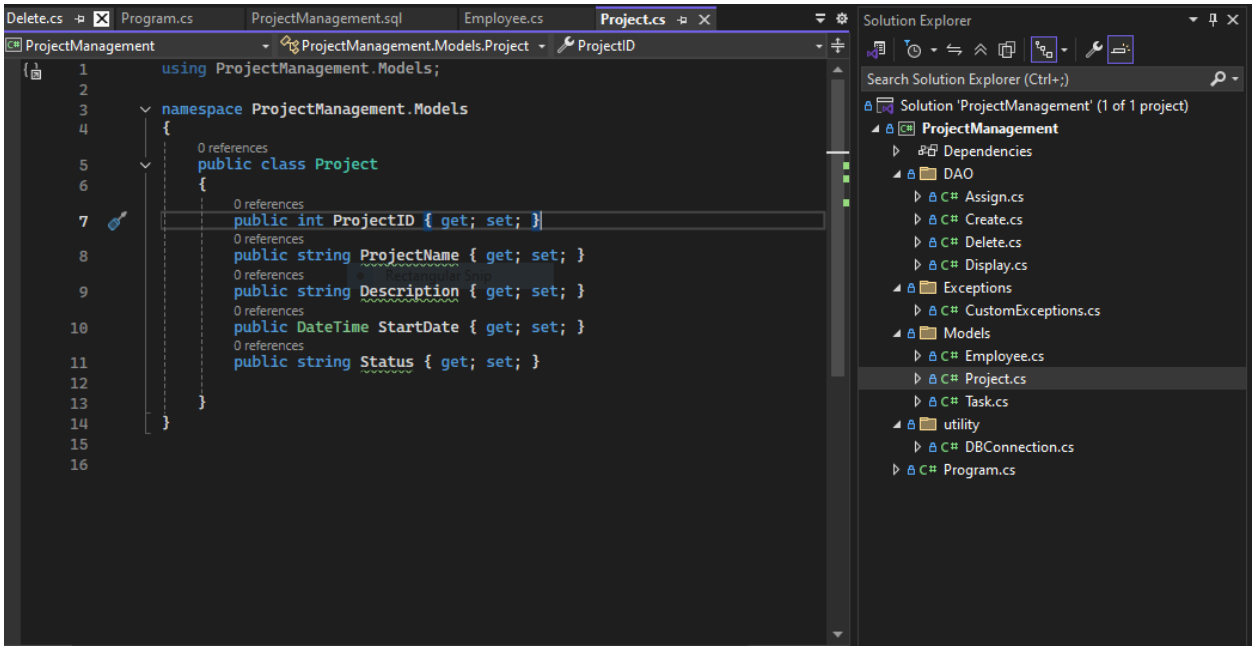
CREATE MODELS:

1. **Employee**

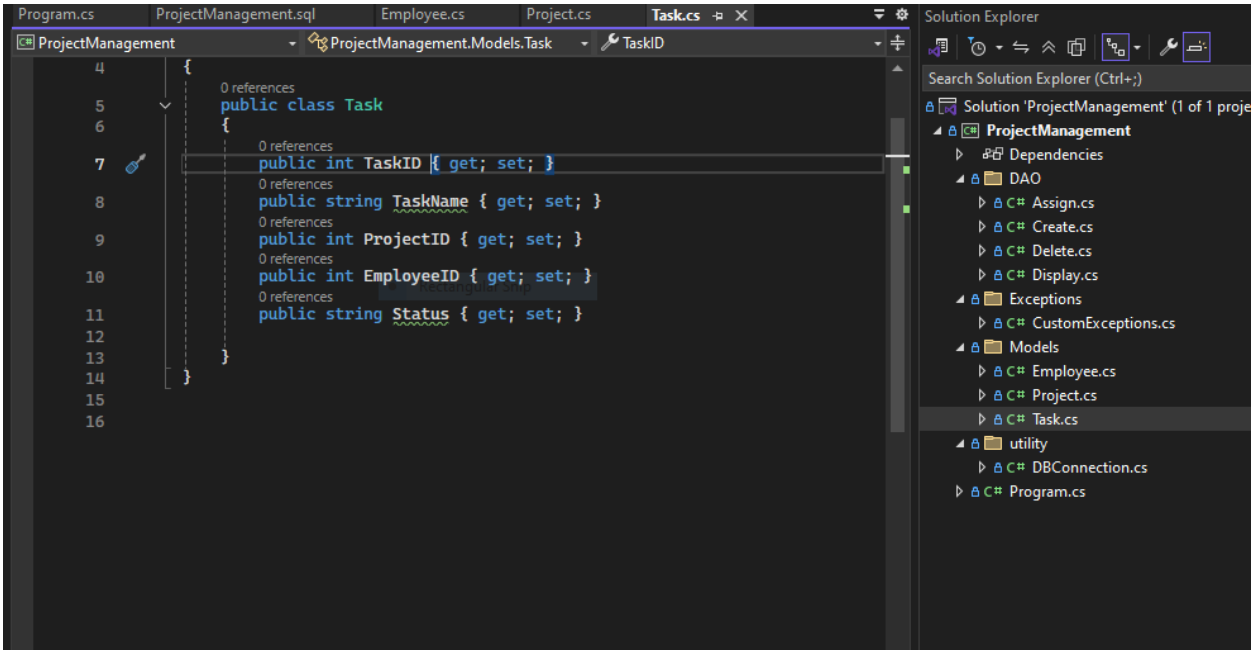


```
using ProjectManagement.Models;

namespace ProjectManagement.Models
{
    public class Employee
    {
        public int EmployeeID { get; set; }
        public string Name { get; set; }
        public string Designation { get; set; }
        public string Gender { get; set; }
        public int Salary { get; set; }
        public int ProjectID { get; set; }


    }
}
```

2. **Project**

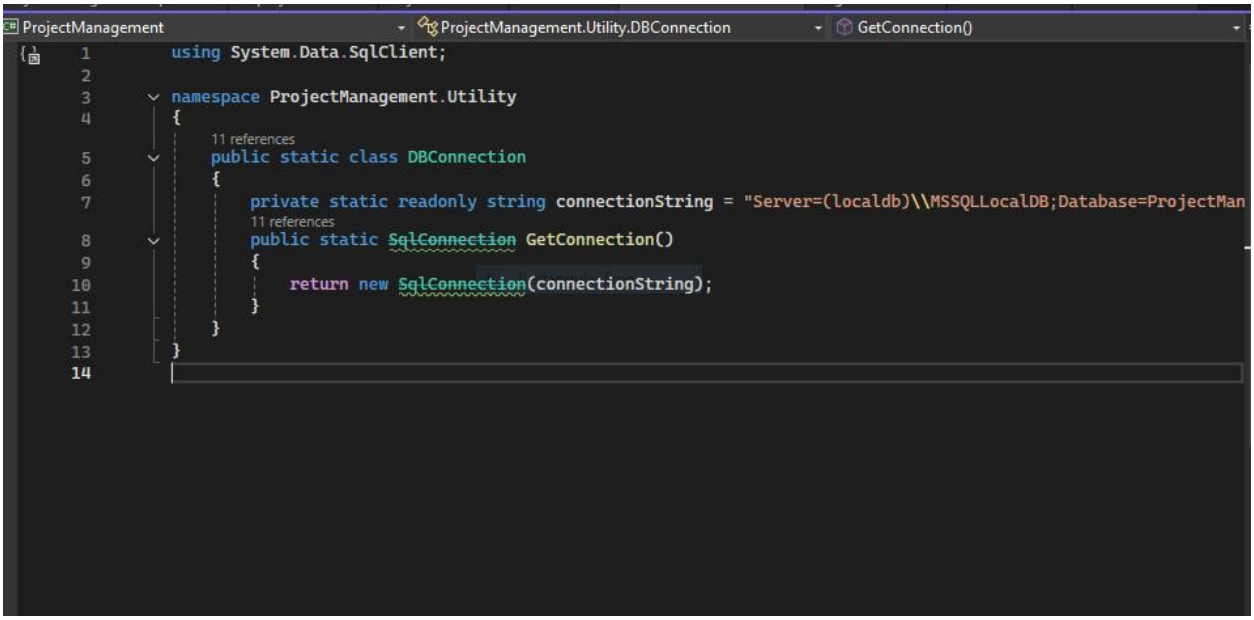


3. Task



CREATE UTILITY

1. DBConnection



CREATE DAO

1. Assign

```
using ProjectManagement.Exceptions;
using ProjectManagement.Models;
using ProjectManagement.Utility;
using System;
using System.Data.SqlClient;
using System.Threading.Tasks;

namespace ProjectManagement.DAO
{
  public class Assign
  {
    public static void AssignProject()
    {

      Console.Write("Enter Employee ID to Assign Project: ");
      int employeeid = int.Parse(Console.ReadLine());

      Console.Write("Enter ProjectID: ");
      int projectid = int.Parse(Console.ReadLine());

      using SqlConnection conn = DBConnection.GetConnection();
      string query = "UPDATE Employee SET ProjectID = @projectid WHERE EmployeeID = @employeeid";

      using SqlCommand cmd = new SqlCommand(query, conn);
      cmd.Parameters.AddWithValue("@employeeid", employeeid);
      cmd.Parameters.AddWithValue("@projectid", projectid);

      conn.Open();

      int result = cmd.ExecuteNonQuery();
      Console.WriteLine($"{result} Project Assigned Sucessfully.");
```

```
        }
        public static void AssignTask()
        {
            Console.Write("Enter Task ID to Assign Project and Employee: ");
            int taskid = int.Parse(Console.ReadLine());

            Console.Write("Enter ProjectID: ");
            int projectid = int.Parse(Console.ReadLine());

            Console.Write("Enter EmployeeID: ");
            int employeeid = int.Parse(Console.ReadLine());

            using SqlConnection conn = DBConnection.GetConnection();
            string query = "UPDATE Task SET ProjectID = @projectid, EmployeeID = @employeeid WHERE TaskID =
@taskid";

            using SqlCommand cmd = new SqlCommand(query, conn);
            cmd.Parameters.AddWithValue("@taskid", taskid);
            cmd.Parameters.AddWithValue("@employeeid", employeeid);
            cmd.Parameters.AddWithValue("@projectid", projectid);

            conn.Open();
            int result = cmd.ExecuteNonQuery();
            Console.WriteLine($"{result} Task Assigned Sucessfully.");

        }
    }

}
```

2. Create

```
using ProjectManagement.Models;
using ProjectManagement.Utility;
using System;
using System.Data.SqlClient;

namespace ProjectManagement.DAO
{
    public class Create
    {
        public static void CreateEmployee()
        {
            Console.Write("Enter Full Name: ");
            string name = Console.ReadLine();

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

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

            Console.Write("Enter Salary: ");
            int salary = int.Parse(Console.ReadLine());

            int EmployeeID = 0;
```

```
        using SqlConnection conn = DBConnection.GetConnection();

        // Insert Employee
        string studentQuery = "INSERT INTO Employee (Name,Designation,Gender,Salary) OUTPUT
INSERTED.EmployeeID VALUES (@name, @designation,@gender,@salary)";
        using SqlCommand cmd = new SqlCommand(studentQuery, conn);
        cmd.Parameters.AddWithValue("@name", name);
        cmd.Parameters.AddWithValue("@designation", designation);
        cmd.Parameters.AddWithValue("@gender", gender);
        cmd.Parameters.AddWithValue("@salary", salary);

        conn.Open();
        EmployeeID = (int)cmd.ExecuteScalar();
        conn.Close();
    }
    public static void CreateProject()
    {
        Console.Write("Enter ProjectName: ");
        string projectname = Console.ReadLine();

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

        Console.Write("Enter Start Date (YYYY-MM-DD): ");
        string startdate = Console.ReadLine();

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

        int ProjectID = 0;

        using SqlConnection conn = DBConnection.GetConnection();

        // Insert Project
        string studentQuery = "INSERT INTO Project ( ProjectName,Description,StartDate,Status) OUTPUT
INSERTED.ProjectID VALUES (@projectname, @description,@startdate,@status)";
        using SqlCommand cmd = new SqlCommand(studentQuery, conn);
        cmd.Parameters.AddWithValue("@projectname", projectname);
        cmd.Parameters.AddWithValue("@description", description);
        cmd.Parameters.AddWithValue("@startdate", startdate);
        cmd.Parameters.AddWithValue("@status", status);

        conn.Open();
        ProjectID = (int)cmd.ExecuteScalar();
        conn.Close();
    }
    public static void CreateTask()
    {
        Console.Write("Enter Task: ");
        string taskname = Console.ReadLine();

        //Console.Write("Enter ProjectID: ");
        //int projectid = int.Parse(Console.ReadLine());

        //Console.Write("Enter EmployeeID: ");
        //int employeeid = int.Parse(Console.ReadLine());

        Console.Write("Enter Status: ");
```

```
            string status = Console.ReadLine();

            int TaskID = 0;

            using SqlConnection conn = DBConnection.GetConnection();

            // Insert Task
            string studentQuery = "INSERT INTO Task ( TaskName,Status) OUTPUT INSERTED.TaskID VALUES
(@taskname, @status)";

            using SqlCommand cmd = new SqlCommand(studentQuery, conn);

            cmd.Parameters.AddWithValue("@taskname", taskname);
            //cmd.Parameters.AddWithValue("@projectid", projectid);
            //cmd.Parameters.AddWithValue("@employeeid", employeeid);
            cmd.Parameters.AddWithValue("@status", status);

            conn.Open();
            TaskID = (int)cmd.ExecuteScalar();
            conn.Close();

        }

        internal bool CreateEmployee(Employee emp)
        {
            throw new NotImplementedException();
        }
    }
}
```

3. Delete

```
using ProjectManagement.Models;
using ProjectManagement.Utility;
using System;
using System.Data.SqlClient;

namespace ProjectManagement.DAO
{
    public class Display
    {
        public static void DisplayEmployee()
        {
            using SqlConnection conn = DBConnection.GetConnection();
            string query = "SELECT * FROM Employee";

            using SqlCommand cmd = new SqlCommand(query, conn);
            conn.Open();
            using SqlDataReader reader = cmd.ExecuteReader();
            Console.WriteLine("\nEmployee Details:");
            while (reader.Read())
            {
                Console.WriteLine($"EmployeeID: {reader["EmployeeID"]}, Name: {reader["Name"]}, Designation:
{reader["Designation"]}, Gender: {reader["Gender"]},Salary: {reader["Salary"]},ProjectID: {reader["ProjectID"]}");
            }
        }
        public static void DisplayProject()
        {
```

```
            using SqlConnection con = DBConnection.GetConnection();
            string query = "SELECT * FROM Project";

            using SqlCommand cmd = new SqlCommand(query, con);
            con.Open();
            using SqlDataReader reader = cmd.ExecuteReader();
            Console.WriteLine("\nProject Details:");
            while (reader.Read())
            {
                Console.WriteLine($"ProjectID: {reader["ProjectID"]}, ProjectName: {reader["ProjectName"]},Description:
{reader["Description"]}, StartDate: {reader["StartDate"]},Status: {reader["Status"]}");
            }
        }
        public static void DisplayTask()
        {
            using SqlConnection con = DBConnection.GetConnection();
            string query = "SELECT * FROM Task";

            using SqlCommand cmd = new SqlCommand(query, con);
            con.Open();
            using SqlDataReader reader = cmd.ExecuteReader();
            Console.WriteLine("\nTask Details:");
            while (reader.Read())
            {
                Console.WriteLine($"TaskID: {reader["TaskID"]}, TaskName: {reader["TaskName"]},ProjectID:
{reader["ProjectID"]}, EmployeeID: {reader["EmployeeID"]},Status: {reader["Status"]}");
            }
        }
        public static void DisplayProjectAssigned()
        {
            string query = @"
                SELECT p.ProjectId, p.ProjectName, e.Name, t.TaskName
                FROM Task t
                JOIN Employee e ON e.EmployeeID = t.EmployeeID
                JOIN Project p ON p.ProjectID = t.ProjectID";

            using SqlConnection conn = DBConnection.GetConnection();
            using SqlCommand cmd = new SqlCommand(query, conn);
            conn.Open();
            SqlDataReader reader = cmd.ExecuteReader();
            Console.WriteLine("\nProjects Assigned");
            while (reader.Read())
            {
                Console.WriteLine($"ProjectID: {reader["ProjectID"]}, ProjectName: {reader["ProjectName"]}, Name:
{reader["Name"]}, TaskName: {reader["TaskName"]}");
            }
        }

    }
}
```

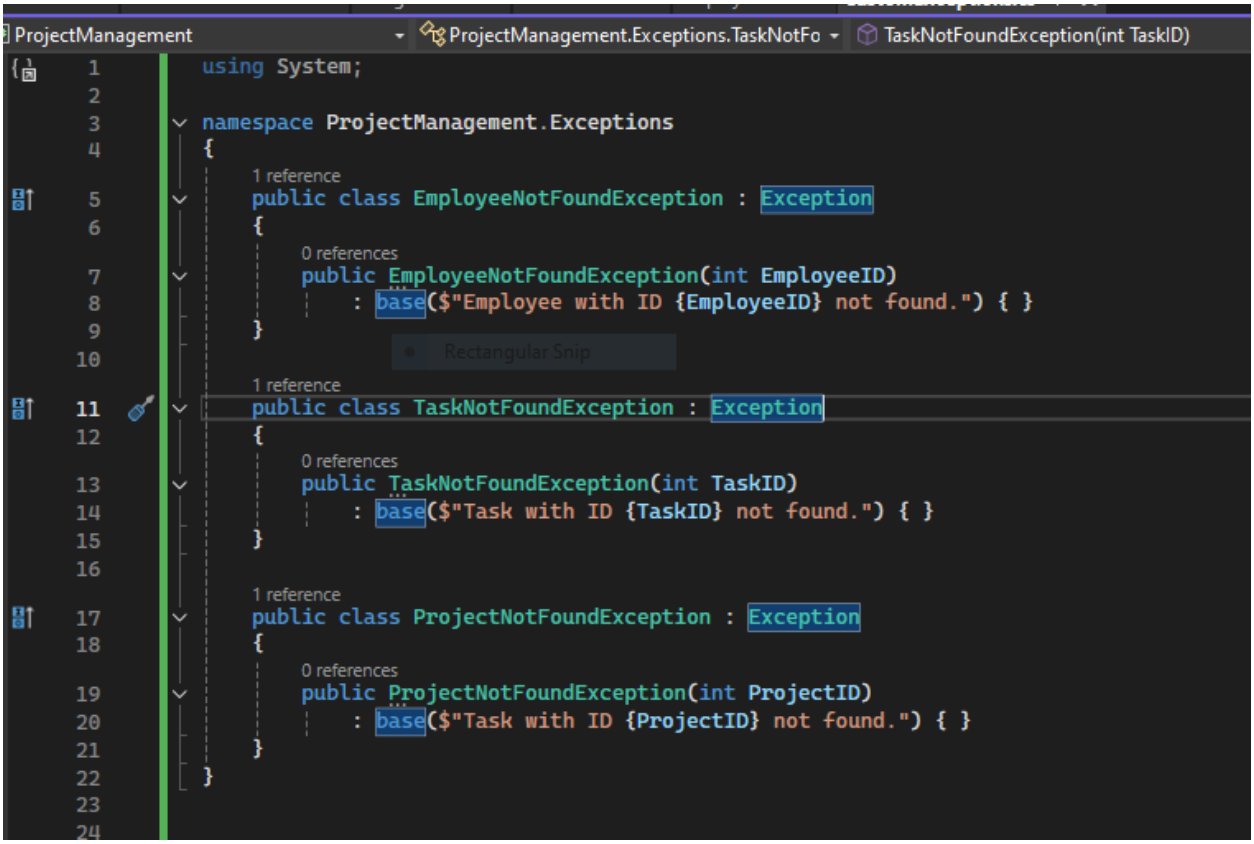
4. Display

```
using ProjectManagement.Models;
using ProjectManagement.Utility;
using System;
using System.Data.SqlClient;
```

```
namespace ProjectManagement.DAO
{
   public class Display
   {
      public static void DisplayEmployee()
      {
         using SqlConnection conn = DBConnection.GetConnection();
         string query = "SELECT * FROM Employee";

         using SqlCommand cmd = new SqlCommand(query, conn);
         conn.Open();
         using SqlDataReader reader = cmd.ExecuteReader();
         Console.WriteLine("\nEmployee Details:");
         while (reader.Read())
         {
            Console.WriteLine($"EmployeeID: {reader["EmployeeID"]}, Name: {reader["Name"]}, Designation:
{reader["Designation"]}, Gender: {reader["Gender"]},Salary: {reader["Salary"]},ProjectID: {reader["ProjectID"]}");
         }
      }
      public static void DisplayProject()
      {
         using SqlConnection con = DBConnection.GetConnection();
         string query = "SELECT * FROM Project";

         using SqlCommand cmd = new SqlCommand(query, con);
         con.Open();
         using SqlDataReader reader = cmd.ExecuteReader();
         Console.WriteLine("\nProject Details:");
         while (reader.Read())
         {
            Console.WriteLine($"ProjectID: {reader["ProjectID"]}, ProjectName: {reader["ProjectName"]},Description:
{reader["Description"]}, StartDate: {reader["StartDate"]},Status: {reader["Status"]}");
         }
      }
      public static void DisplayTask()
      {
         using SqlConnection con = DBConnection.GetConnection();
         string query = "SELECT * FROM Task";

         using SqlCommand cmd = new SqlCommand(query, con);
         con.Open();
         using SqlDataReader reader = cmd.ExecuteReader();
         Console.WriteLine("\nTask Details:");
         while (reader.Read())
         {
            Console.WriteLine($"TaskID: {reader["TaskID"]}, TaskName: {reader["TaskName"]},ProjectID:
{reader["ProjectID"]}, EmployeeID: {reader["EmployeeID"]},Status: {reader["Status"]}");
         }
      }
      public static void DisplayProjectAssigned()
      {
         string query = @"
            SELECT p.ProjectId, p.ProjectName, e.Name, t.TaskName
            FROM Task t
            JOIN Employee e ON e.EmployeeID = t.EmployeeID
            JOIN Project p ON p.ProjectID = t.ProjectID";

         using SqlConnection conn = DBConnection.GetConnection();
         using SqlCommand cmd = new SqlCommand(query, conn);
```

```
            conn.Open();
            SqlDataReader reader = cmd.ExecuteReader();
            Console.WriteLine("\nProjects Assigned");
            while (reader.Read())
            {
                Console.WriteLine($"ProjectID: {reader["ProjectID"]}, ProjectName: {reader["ProjectName"]}, Name:
{reader["Name"]}, TaskName: {reader["TaskName"]}");
            }
        }

    }
}
```

CREATE EXCEPTIONS



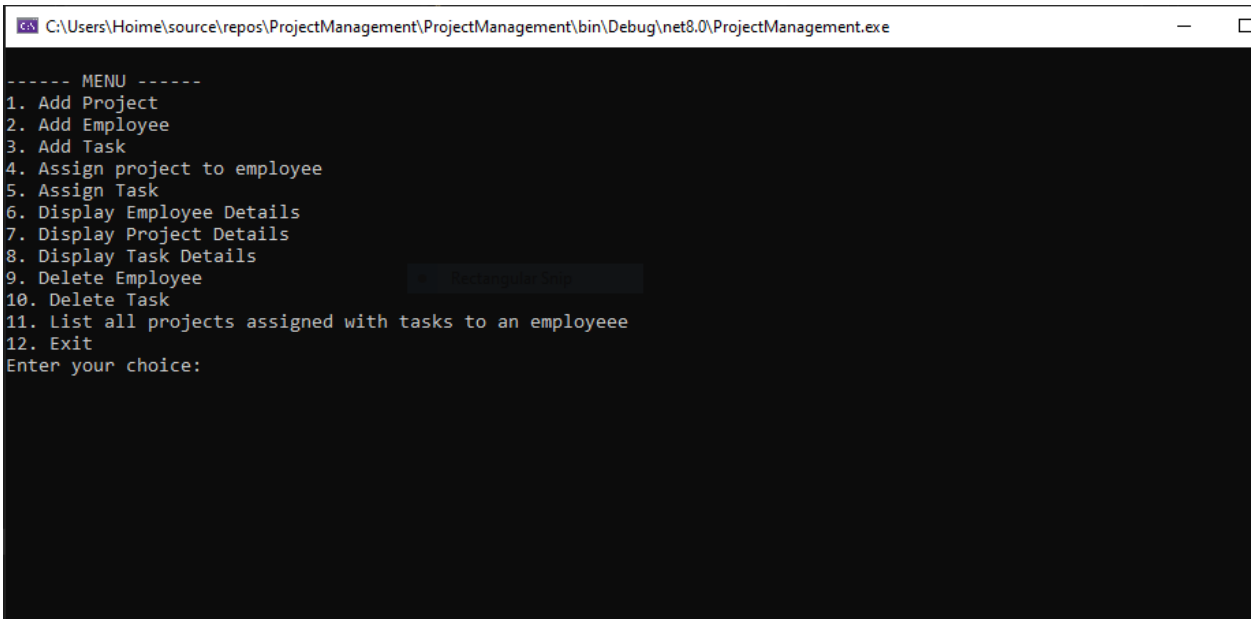
MAIN PROGRAM

```
using ProjectManagement.DAO;

namespace ProjectManagement
{
    public class MainProgram
    {
        static void Main()
        {
            bool exit = false;

            while (!exit)
```

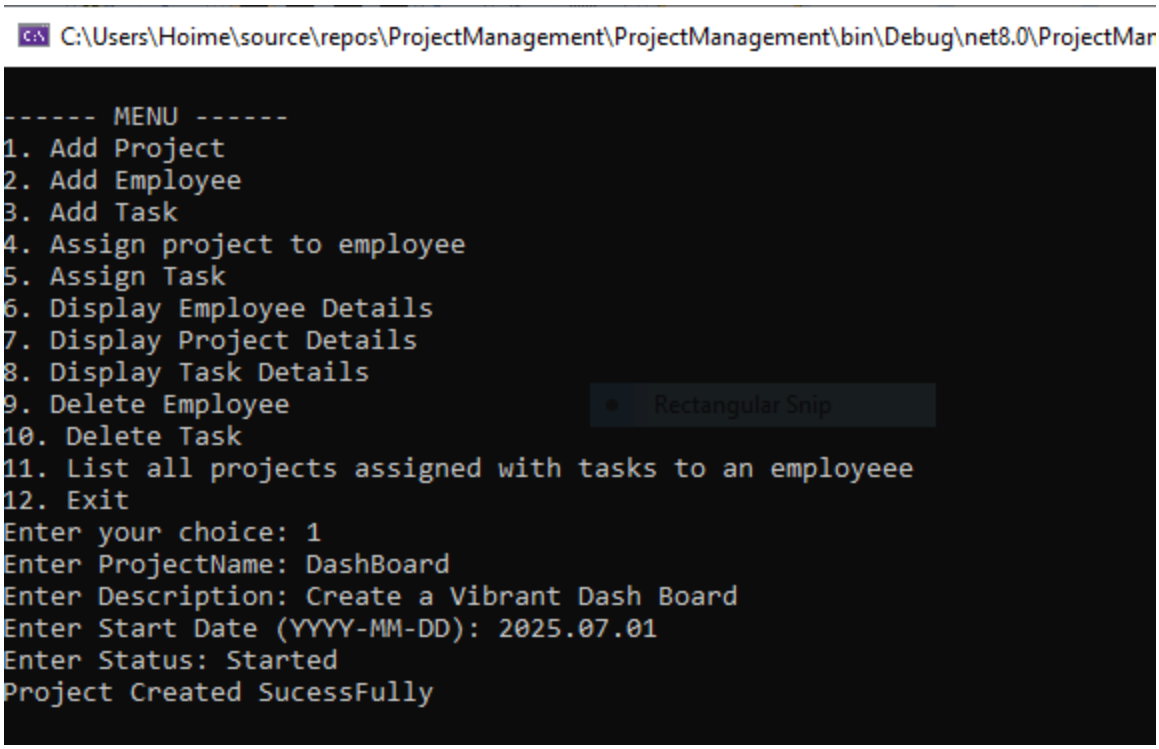
```
{
    Console.WriteLine("\n------ MENU ------");
    Console.WriteLine("1. Add Project");
    Console.WriteLine("2. Add Employee");
    Console.WriteLine("3. Add Task");
    Console.WriteLine("4. Assign project to employee ");
    Console.WriteLine("5. Assign Task");
    Console.WriteLine("6. Display Employee Details");
    Console.WriteLine("7. Display Project Details");
    Console.WriteLine("8. Display Task Details");
    Console.WriteLine("9. Delete Employee");
    Console.WriteLine("10. Delete Task");
    Console.WriteLine("11. List all projects assigned with tasks to an employeee ");


    Console.WriteLine("12. Exit");

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

    switch (choice)
    {
        case "1":
            Create.CreateProject();
            Console.WriteLine("Project Created SucessFully");
            break;
        case "2":
            Create.CreateEmployee();
            Console.WriteLine("Employee Data Created SucessFully");
            break;
        case "3":
            Create.CreateTask();
            Console.WriteLine("Task Created Sucessfully");
            break ;
        case "4":
            Assign.AssignProject();
            break;
        case "5":
            Assign.AssignTask();
            break;
        case "6":
            Display.DisplayEmployee();
            break;
        case "7":
            Display.DisplayProject();
            break;
        case "8":
            Display.DisplayTask();
            break;
        case "9":
            Delete.DeleteEmployee();
            break;
        case "10":
            Delete.DeleteTask();
            break;
        case "11":
            Display.DisplayProjectAssigned();
            break;
        case "12":
```

```
                exit = true;
                break;

            default:
                Console.WriteLine("Invalid choice. Try again.");
                break;
        }

    }

  }
 }
}
```

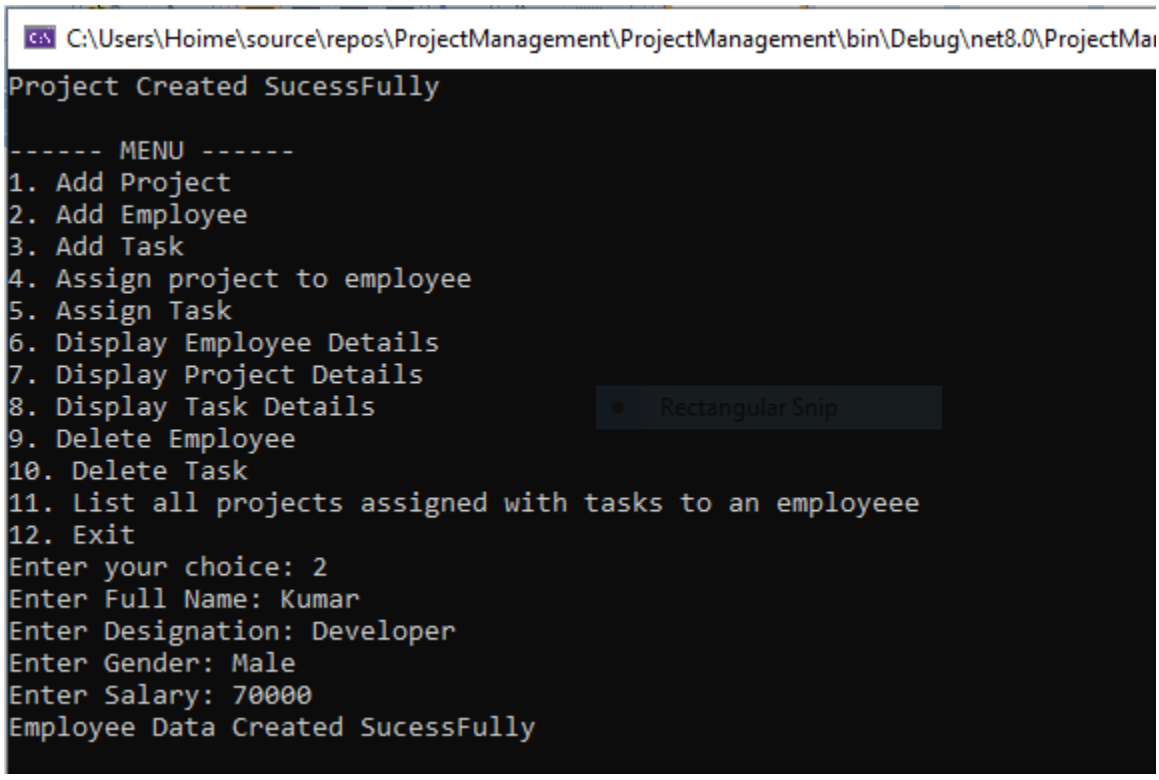
OUTPUT



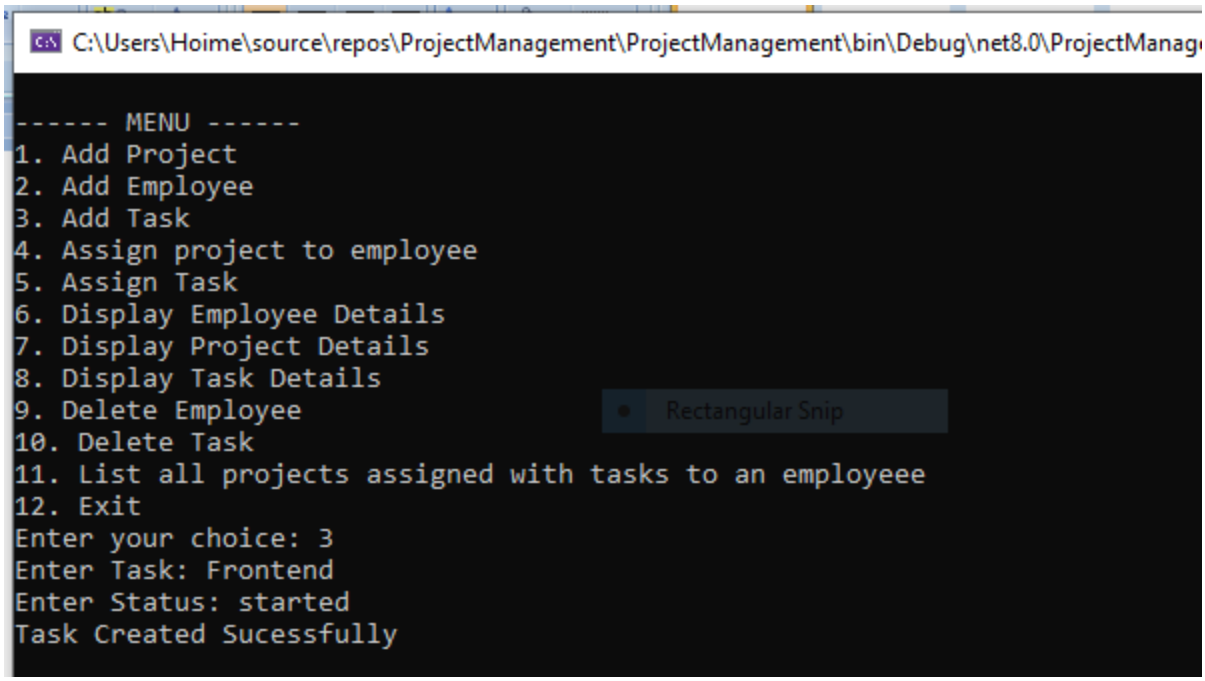
1. Add Project



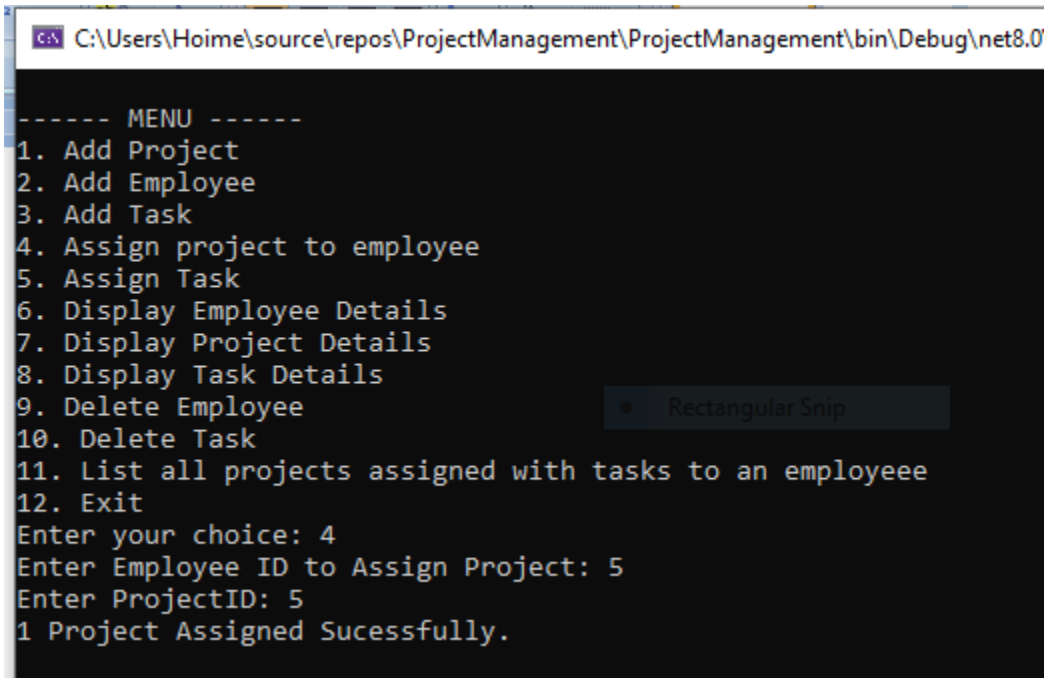
2. Add Employee



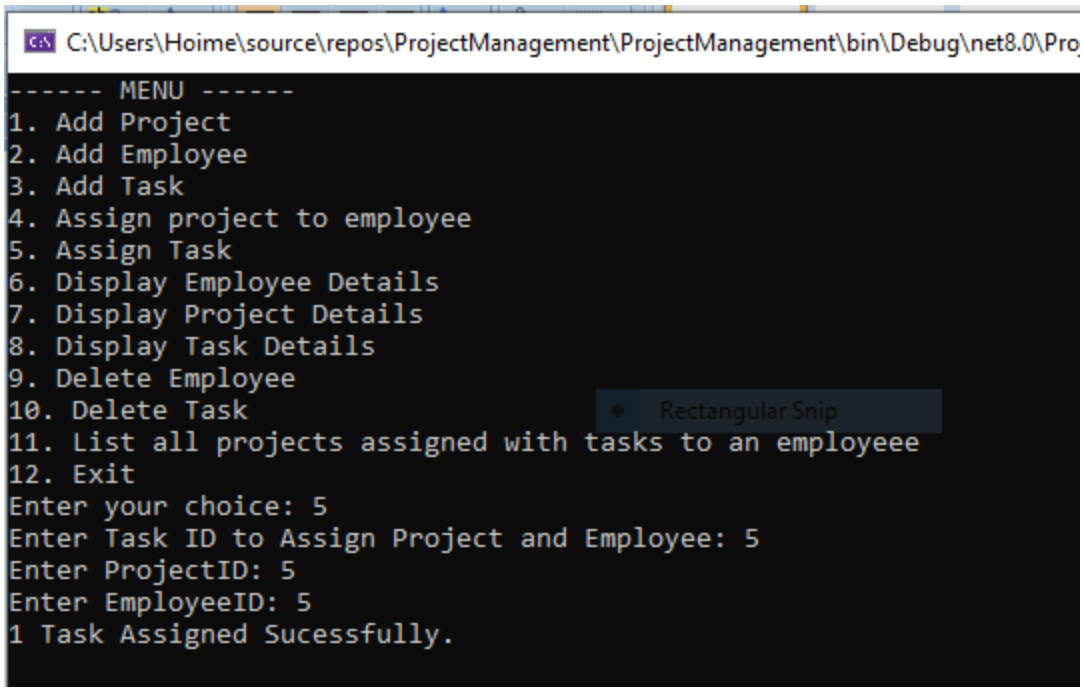
3. Add Task



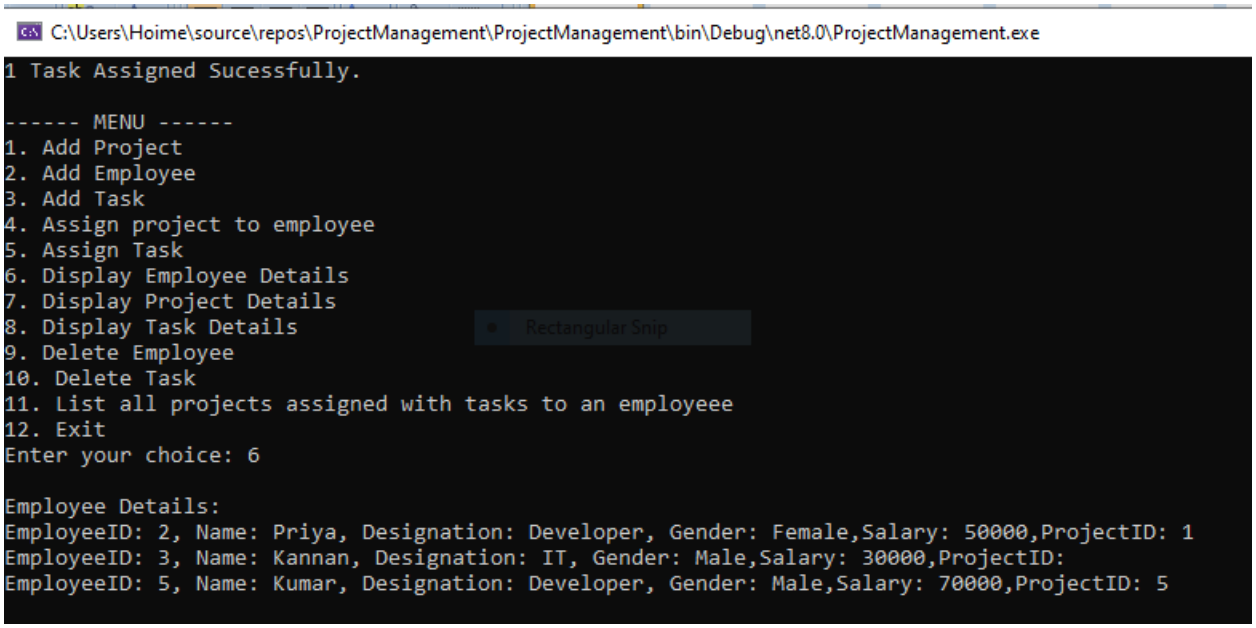
4. Assign Project to Employee



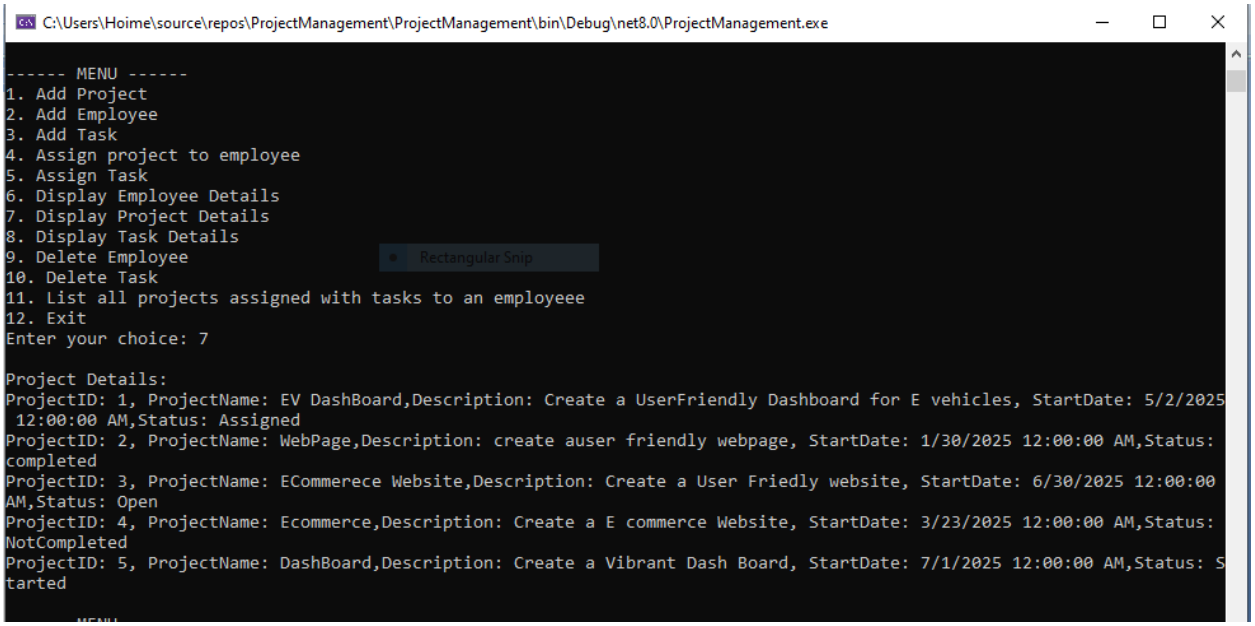
5. Assign Task



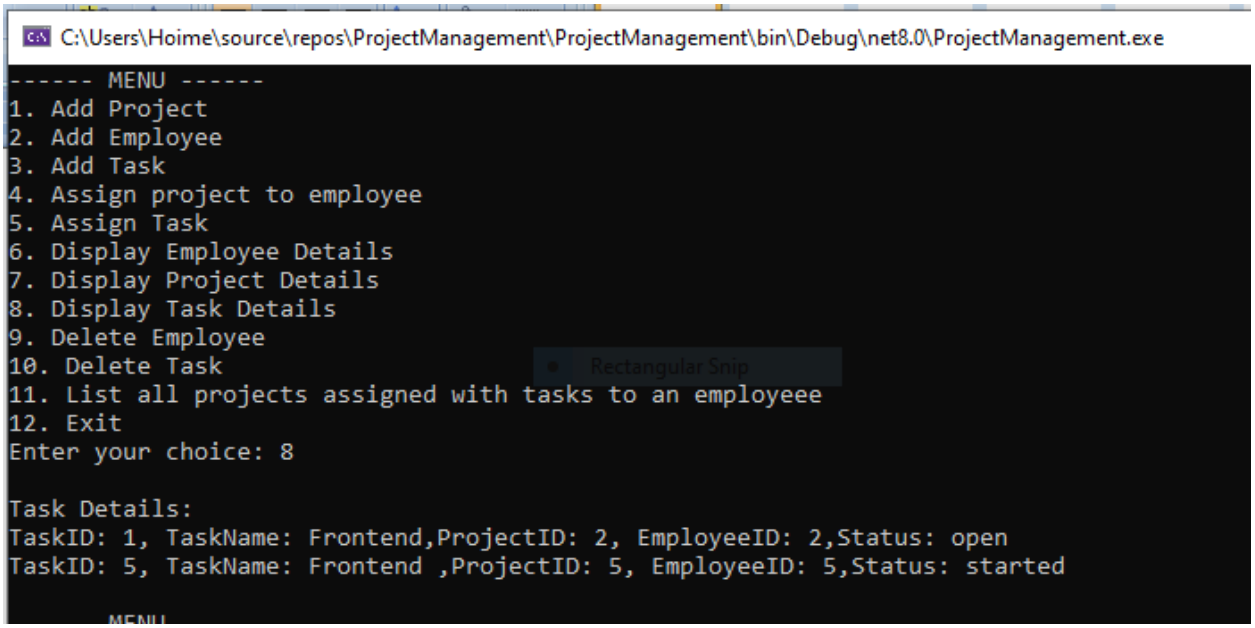
6. Display Employee Details



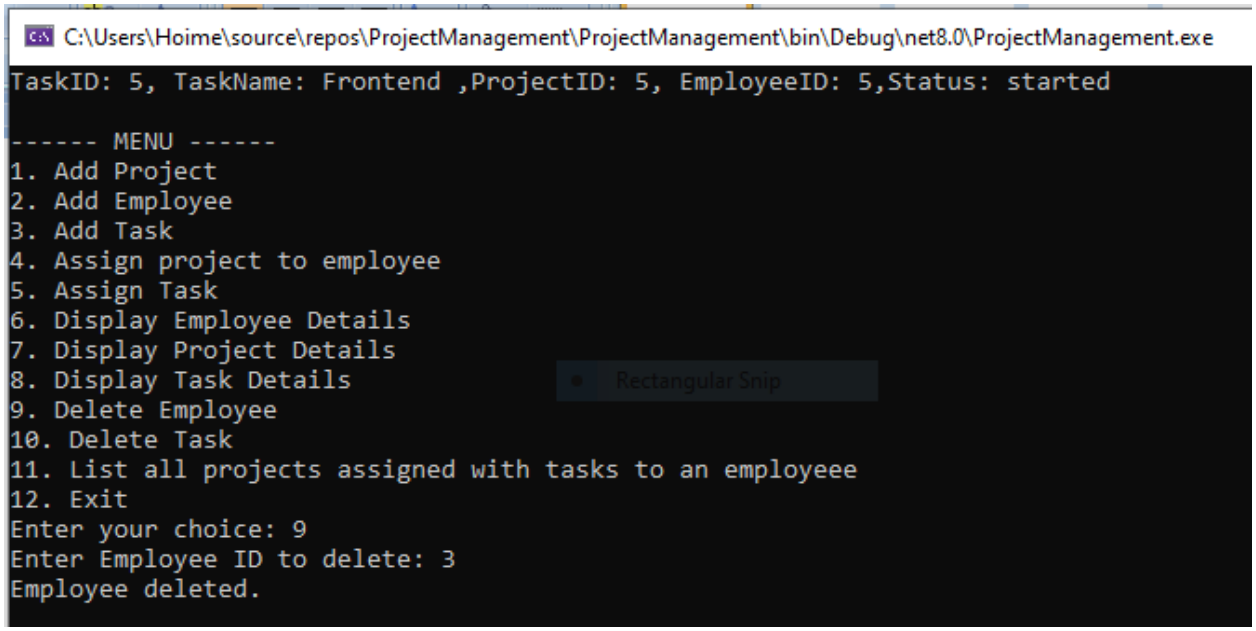
7. Display Project Details



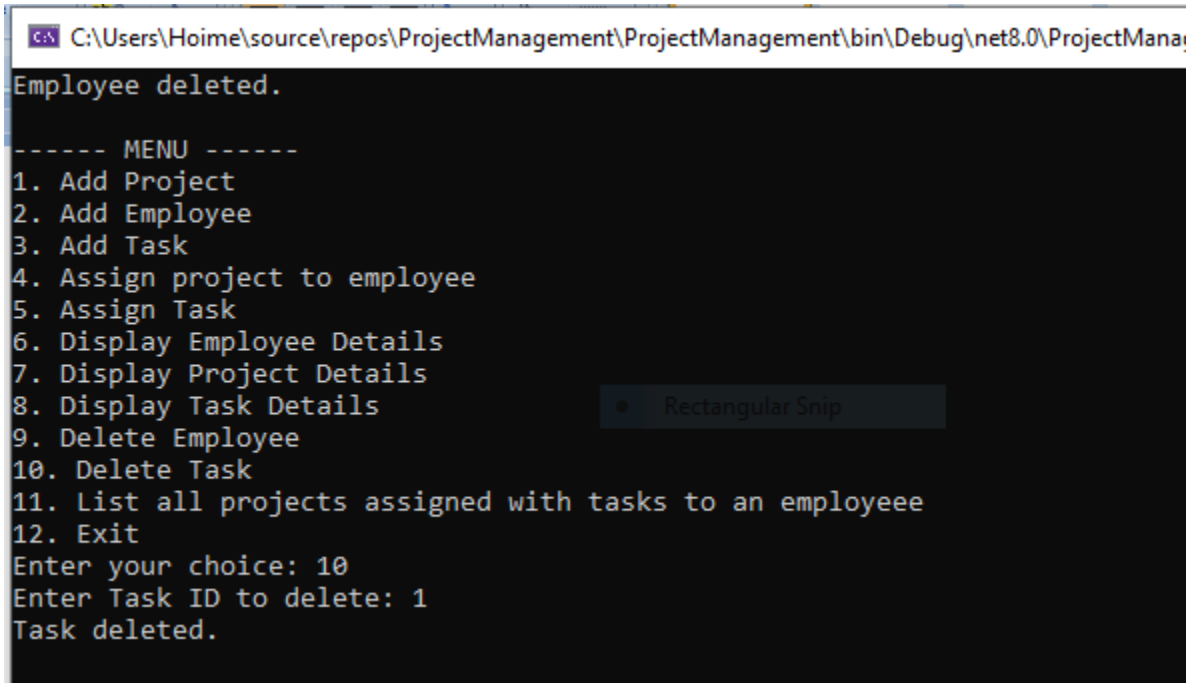
8. Display Task Details



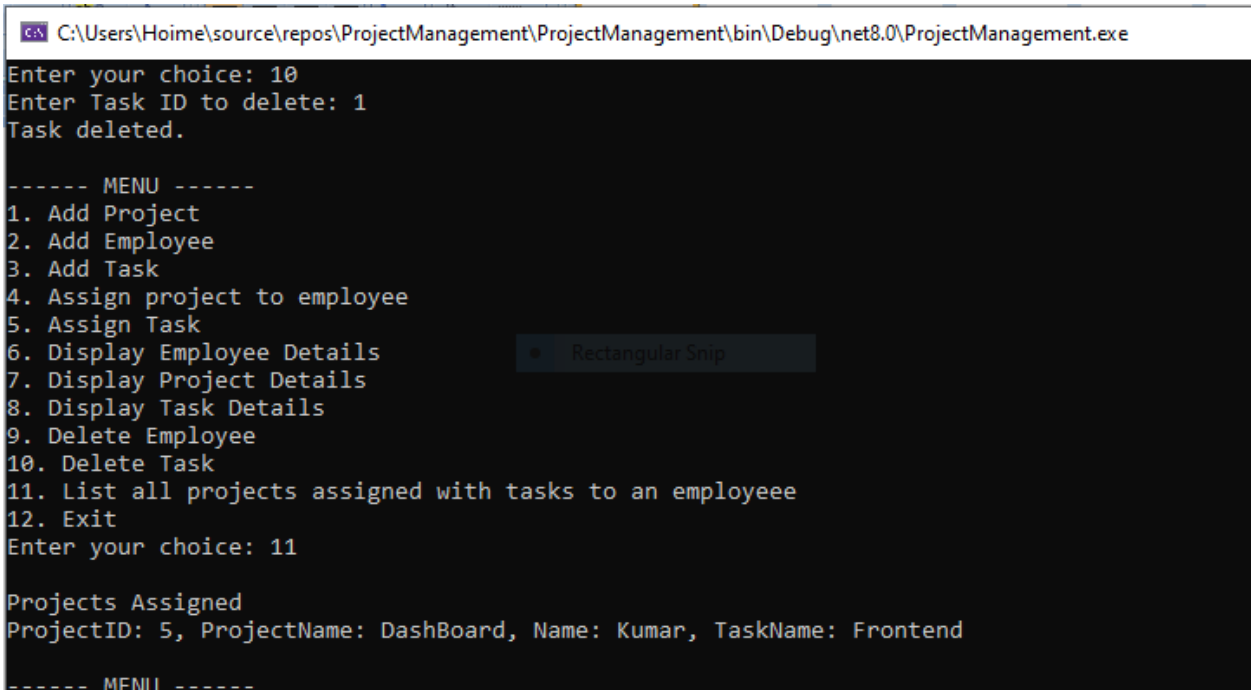
9. Delete Employee



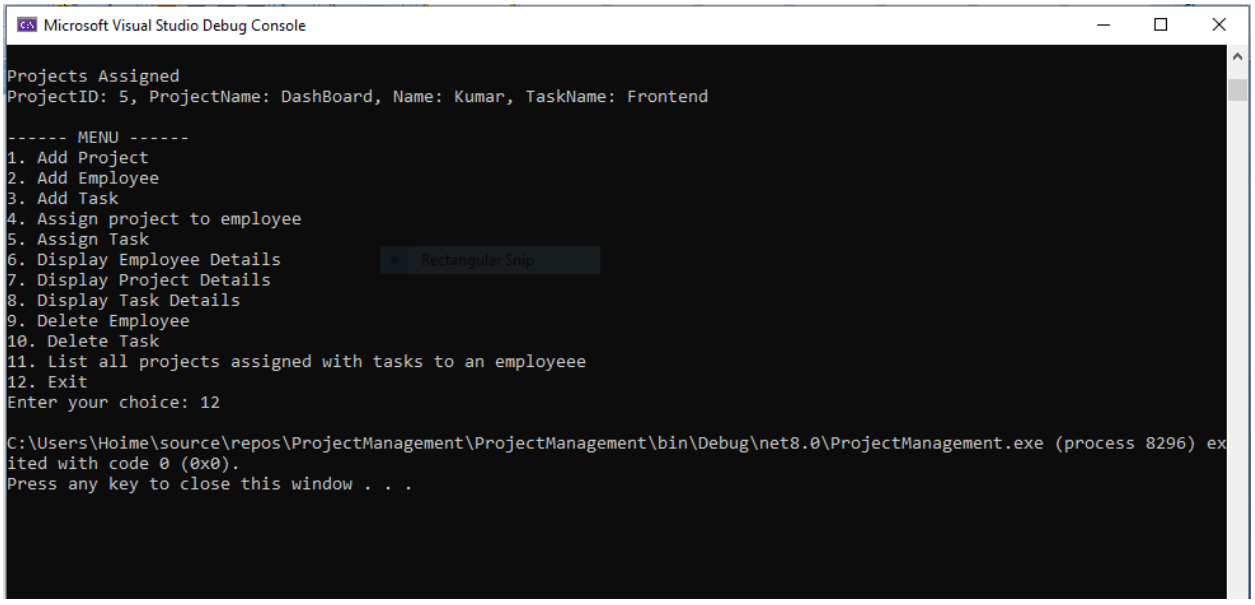
10. Delete Task



11. List all projects assigned with tasks to an employee


C:\Users\Hoime\source\repos\ProjectManagement\ProjectManagement\bin\Debug\net8.0\ProjectManagement.exe
Enter your choice: 10
Enter Task ID to delete: 1
Task deleted.
------ MENU ------
1. Add Project
2. Add Employee
3. Add Task
4. Assign project to employee
5. Assign Task
6. Display Employee Details
7. Display Project Details
8. Display Task Details
9. Delete Employee
10. Delete Task
11. List all projects assigned with tasks to an employeee
12. Exit
Enter your choice: 11
Projects Assigned
ProjectID: 5, ProjectName: DashBoard, Name: Kumar, TaskName: Frontend

12. Exit


Microsoft Visual Studio Debug Console
Projects Assigned
ProjectID: 5, ProjectName: DashBoard, Name: Kumar, TaskName: Frontend
------ MENU ------
1. Add Project
2. Add Employee
3. Add Task
4. Assign project to employee
5. Assign Task
6. Display Employee Details
7. Display Project Details
8. Display Task Details
9. Delete Employee
10. Delete Task
11. List all projects assigned with tasks to an employeee
12. Exit
Enter your choice: 12
C:\Users\Hoime\source\repos\ProjectManagement\ProjectManagement\bin\Debug\net8.0\ProjectManagement.exe (process 8296) exited with code 0 (0x0).
Press any key to close this window . . .