# FINAL PROJECT

# (BY G V S S SRI LASYA)

|  |
| --- |
| FINAL PROJECT |
| CODE FOR CLIENT APP |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using BusinessLogicLibrary;  namespace ClientApp  {  internal class Program  {  static void Main(string[] args)  {  int option;  string choice;  do  {  Console.Write("\n1.Add Employee : ");  Console.Write("\n2.Search Employee by Id: ");  Console.Write("\n3.Search Employee byname: ");  Console.Write("\n4.Display all Employees : ");  Console.Write("\nEnter your choice : ");  option = Convert.ToInt32(Console.ReadLine());  switch (option)  {  case 1:  AddEmployee();  break;  case 2:  GetEmployeeById();  break;  case 3:  GetEmployeeByName();  break;  case 4:  GetAllEmployees();  break;  default:  Console.Write("Invalid option");  break;  }  Console.Write("\nDo you want to choose more options(y/n)? :");  choice = Console.ReadLine();  } while (choice.Equals("y"));  }  public static void AddEmployee()  {  int id, salary, age;  string name;    Console.Write("\n Enter id : ");  id = Convert.ToInt32(Console.ReadLine());  Console.Write("\n Enter name : ");  name = Console.ReadLine();  Console.Write("\n Enter salary : ");  salary = Convert.ToInt32(Console.ReadLine());  Console.Write("\n Enter age : ");  age = Convert.ToInt32(Console.ReadLine());  var result = EmployeeBLL.AddEmployee(id, name, salary, age);  if (result)  Console.Write("\nEmployee details are succesfuly saved");  else  Console.Write("\nSome error occured,check input");  }  public static void GetEmployeeById()  {  int id;  Console.Write("Enter id : ");  id =Convert.ToInt32(Console.ReadLine());  var result = EmployeeBLL.GetEmployeeById(id);  if (result.Count == 0)  Console.Write("\n\nNo records exist witg this id");  else  result.ForEach(p => Console.Write("\n" + p));  }  public static void GetEmployeeByName()  {  string name;  Console.Write("Enter name : ");  name = Console.ReadLine();  var result = EmployeeBLL.GetEmployeeByName(name);  if (result.Count == 0)  Console.Write("\n\nNo records exist with this name");  else  result.ForEach(p => Console.Write("\n" + p));  }  public static void GetAllEmployees()  {  Console.Write("\nComplete details of all the empoyees : ");  var result = EmployeeBLL.GetAllEmployees();  result.ToList().ForEach(p => Console.Write("\n" + p));  }  }  } |
| CODE FOR BLL |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Text.RegularExpressions;  using System.Threading.Tasks;  using DataAccessLibrary;  using System.IO;  namespace BusinessLogicLibrary  {  public class EmployeeBLL  {  public static string filePath = "C:\\Users\\gottu\\Desktop\\csc images\\Employees.txt";  public static bool AddEmployee(int empId, string empName, int empSalary, int empAge)  {  //todo add validations  //if all validations are succesful then call dal  var allEmployees = File.ReadAllLines(filePath);  bool isFound = false;    foreach (string employee in allEmployees)  {  var employeeDetails = employee.Split(',');  if (Convert.ToInt32(employeeDetails[0]) == empId)  {  isFound = true;  break;  }  }      if (Regex.Match(empName, "^[a-zA-Z.]{3,50}$").Success && Regex.Match(empId.ToString(), @"^[1-9]\d\*$").Success && isFound == false)    {  var result = EmployeeDAL.AddEmployee(empId, empName, empSalary, empAge);  return result;  }  else  {  return false;    }  }  public static List<string> GetEmployeeById(int id)  {  var result = EmployeeDAL.GetEmployeeById(id);  return result;  }  public static List<string> GetEmployeeByName(string name)  {  var result = EmployeeDAL.GetEmployeeByName(name);  return result;  }  public static string[] GetAllEmployees()  {  var result = EmployeeDAL.GetAllEmployees();  return result;  }  }  } |
| CODE FOR DAL |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.IO;  namespace DataAccessLibrary  {  public static class EmployeeDAL  {  public static string filePath = "C:\\Users\\gottu\\Desktop\\csc images\\Employees.txt";  public static bool AddEmployee(int empId, string empName, int empSalary, int empAge)  {  try  {  string textContent = string.Concat(empId, ",",empName, ",",empSalary, ",", empAge);  File.AppendAllText(filePath, textContent + Environment.NewLine);  return true;  }  catch (Exception ex)  {  return false;  }  }  public static List<string> GetEmployeeById(int id)  {  var allEmployees = File.ReadAllLines(filePath);  bool isFound = false;  List<string> employeeFound = new List<string>();  foreach(string employee in allEmployees)  {  var employeeDetails = employee.Split(',');  if(Convert.ToInt32(employeeDetails[0]) == id)  {  isFound = true;  employeeFound.Add(employee);  break;  }  }  return employeeFound;  }  public static List<string> GetEmployeeByName(string name)  {  var allEmployees = File.ReadAllLines(filePath);  bool isFound = false;  List<string> employeeFound = new List<string>();  foreach (string employee in allEmployees)  {  var employeeDetails = employee.Split(',');  if (employeeDetails[1].Contains(name))  {  isFound = true;  employeeFound.Add(employee);  }  }  return employeeFound;  }  public static string[] GetAllEmployees()  {  var allEmployees = File.ReadAllLines(filePath);  return allEmployees;  }  }  } |
| OUTPUTS |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |