ASSIGNMENT-1

WRITEUP

M SHREE VAMSHIKA

Software engineer

Dover India private limited.

ABSTRACT

This project is a console-based application which can be used to manage teacher's data in school/college. The user can use command line interface to interact with the application's backend that is text-file is used to store, update and retrieve teacher's data. C# programming language is used to write this program.

SOFTWARE REQUIREMENTS

- Visual studio
- .Net Framework
- Windows OS

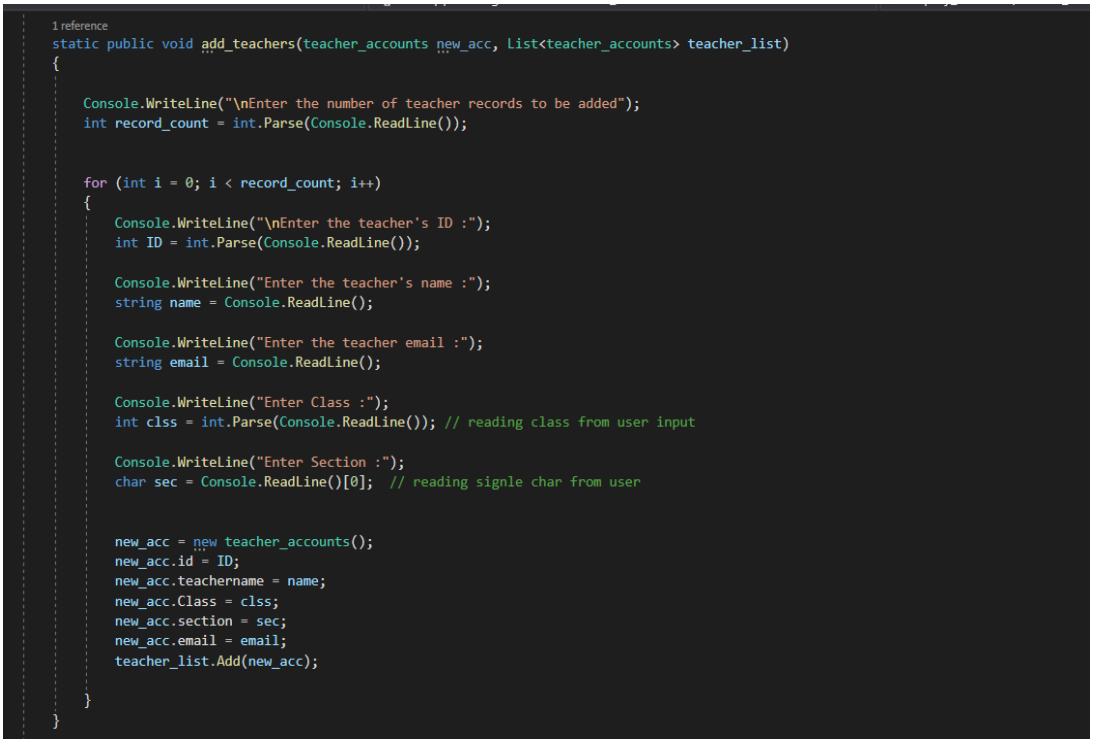
FEATURES PROVIDED

1. Adding new teacher
2. Displaying all teacher's
3. Updating teacher's data
4. Delete a teacher record
5. Saving the record to text file

1. <u>Adding teacher data</u>:

   When the user chooses 1st option this means he wants to add teacher data.

   User will have to enter number of records to be added along with ID, name, email, class, section of the teacher.



```
1 reference
static public void add_teachers(teacher_accounts new_acc, List<teacher_accounts> teacher_list)
{

    Console.WriteLine("\nEnter the number of teacher records to be added");
    int record_count = int.Parse(Console.ReadLine());


    for (int i = 0; i < record_count; i++)
    {
        Console.WriteLine("\nEnter the teacher's ID :");
        int ID = int.Parse(Console.ReadLine());

        Console.WriteLine("Enter the teacher's name :");
        string name = Console.ReadLine();

        Console.WriteLine("Enter the teacher email :");
        string email = Console.ReadLine();

        Console.WriteLine("Enter Class :");
        int clss = int.Parse(Console.ReadLine()); // reading class from user input

        Console.WriteLine("Enter Section :");
        char sec = Console.ReadLine()[0];  // reading signle char from user


        new_acc = new teacher_accounts();
        new_acc.id = ID;
        new_acc.teachername = name;
        new_acc.Class = clss;
        new_acc.section = sec;
        new_acc.email = email;
        teacher_list.Add(new_acc);

    }
}
```

2. Displaying the teacher record:

   When the users choose 2nd option all the teacher details is displayed.

```
1 reference
static public void display_teachers(teacher_accounts new_acc, List<teacher_accounts> teacher_list)
{


    for (int i = 0; i < teacher_list.Count; i++)

    {
        Console.WriteLine("\n\nID = " + teacher_list[i].id);
        Console.WriteLine("Teacher Name = " + teacher_list[i].teachername);
        Console.WriteLine("Class = " + teacher_list[i].Class);
        Console.WriteLine("Section = " + teacher_list[i].section);
        Console.WriteLine("Teacher Email = " + teacher_list[i].email + "\n\n");
    }

    if (teacher_list.Count < 1)
        Console.WriteLine("\n  No teacher's data found \n");

}
1 reference
```

3. Update teacher's data:
   Here the user has to provide the ID of the teacher to update and select the field which he wants to update.

```
1 reference
static public void update_teachers_data(List<teacher_accounts> teacher_list)
{
    Console.WriteLine("Enter the teacher's id to update account ..");
    int update_id = int.Parse(Console.ReadLine());

    int index = teacher_list.FindIndex(x => x.id == update_id);
    if (index > -1)
    {
        Console.WriteLine("Enter option \n1. Name \n2. Class\n3. Section\n4. email");
        int option = int.Parse(Console.ReadLine());

        if (option == 1)
        {
            Console.WriteLine("Enter new name to be updated ..");
            string nn = Console.ReadLine();
            teacher_list[index].teachername = nn;
        }

        if (option == 2)
        {
            Console.WriteLine("Enter new class to be updated ..");
            int nc = int.Parse(Console.ReadLine());
            teacher_list[index].Class = nc;
        }

        if (option == 3)
        {
            Console.WriteLine("Enter new section to be updated ..");
```

4. Deleting teacher record:
   Here the user has to mention the teacher ID to delete the record of that particular teacher.

```
1 reference
static public void delete_records(List<teacher_accounts> teacher_list)
{
    Console.WriteLine("\nEnter the teacher's id to delete record ..");
    int delete_id = int.Parse(Console.ReadLine());
    int index = teacher_list.FindIndex(x => x.id == delete_id);

    if (index > -1)
    {
        Console.WriteLine("\nTeacher's Name :" + teacher_list[index].teachername + "\nAllocated Class :" + teacher_list[index].Cla
        Console.WriteLine("Will be deleted from records ..\n");
        teacher_list.RemoveAt(index);
    }

    else
        Console.WriteLine("ID not found in records \n");
```

5. Saving the teacher details to file:
   After all the necessary operations are done the user can finally save all the records to a text-file.

```
}
1 reference
static public void save_to_text_file(List<teacher_accounts> teacher_list)
{
    string path_to_file = @"C:\Users\11035922\Desktop\teachers record.txt";
    string rec = "";


    for (int i = 0; i < teacher_list.Count; i++)
    {
        rec += teacher_list[i].id + " " + teacher_list[i].teachername + " " + teacher_list[i].Class +
            " " + teacher_list[i].section + " "+teacher_list[i].email+"\n";
    }

    File.WriteAllText(path_to_file, rec);

}
```