Source Code for Phase 1 Project:

```
using System;
using System.Collections.Generic;
using System.Text;
using System.IO;


namespace Phase1_Project
{
    class project
    {
        class Teacher

        {
            public int id { get; set; }                 //class elements: Teacher's Id
            public string name { get; set; }                            // Teacher's Name
            public int std { get; set; }                                // Teacher's
Standard
            public char sec { get; set; }                               // Teacher's
Section

        }
        class TeacherData

        {
            // Method for Adding Teachers details
            static public void GetData(Teacher t, List<Teacher> teacher_list)
            {
                Console.WriteLine("\nEnter number of Teacher's Data to be added\n");
                int n = int.Parse(Console.ReadLine());

                for (int i = 0; i < n; i++)
                {
                    Console.WriteLine("\nEnter Teacher's Id :\n");
                    int ID = int.Parse(Console.ReadLine());

                    Console.WriteLine("\nEnter Teacher's Name :\n");
                    string NAME = Console.ReadLine();

                    Console.WriteLine("\nEnter Standard :\n");
                    int STD = int.Parse(Console.ReadLine());

                    Console.WriteLine("\nEnter Section :\n");
                    char SEC = char.Parse(Console.ReadLine());
                    // Object (t) for class (Teacher) and list of object (teacher_list)
to store teachers details
                    t = new Teacher();
                    t.id = ID;
                    t.name = NAME;
                    t.std = STD;
                    t.sec = SEC;
                    teacher_list.Add(t);
                }
            }
```

```
            // Method for Updating the existing teacher's details
            static public void UpdateData(List<Teacher> teacher_list)
            {
                Console.WriteLine("\nEnter Teacher's Id for updation :\n");
                int u = int.Parse(Console.ReadLine());
                // To FIND Id
                int i = teacher_list.FindIndex(x => x.id == u);
                // If Id NOT FOUND!!
                if (i<0)
                {
                    Console.WriteLine("\nInvalid Id!!\n");
                }
                // If Id FOUND!!
                else
                {
                    Console.WriteLine("\nEnter Name:\n");
                    string a = Console.ReadLine();
                    teacher_list[i].name = a;
                    Console.WriteLine("\nEnter Standard:\n");
                    int b = int.Parse(Console.ReadLine());
                    teacher_list[i].std = b;
                    Console.WriteLine("\nEnter Section:\n");
                    char c = Console.ReadLine()[0];
                    teacher_list[i].sec = c;
                }
            }
            // Method to Delete Teacher's Details
            static public void RemoveData(List<Teacher> teacher_list)
            {
                Console.WriteLine("\nEnter Id of Teacher whose Data to be deleted\n");
                int d = int.Parse(Console.ReadLine());
                // Search of Id
                int i = teacher_list.FindIndex(x => x.id == d);
                // If Id NOT FOUND!!
                if (i < 0)
                {
                    Console.WriteLine("\nInvalid Id!!\n");
                }
                // If Id FOUND!!
                else
                {
                    Console.WriteLine("\nTeacher's Name :" + teacher_list[i].name +
"\n\nStandard :" + teacher_list[i].std + "\n\nSection :" + teacher_list[i].sec + "\n");
                    Console.WriteLine("\nDATA DELETED!!");
                    teacher_list.RemoveAt(i);
                }
            }
            // Method to Display Teacher's Details
            static public void DisplayData(Teacher t, List<Teacher> teacher_list)
            {
                for (int i = 0; i < teacher_list.Count; i++)
                {
                    Console.WriteLine("\nTeacher's ID = " + teacher_list[i].id +
"\n\nTeacher's Name = " + teacher_list[i].name + "\n\nStandard = " + teacher_list[i].std
+ "\n\nSection = " + teacher_list[i].sec + "\n");
                }
                // If the list is EMPTY!!
                if (teacher_list.Count == 0)
```

```
                Console.WriteLine("\nEmpty List\n");
            }
            // Method to Store Teacher's Details in Text File
            static public void StoreData(List<Teacher> teacher_list)
            {
                string p = @"E:\VS_C#_Programs\Teacher_Data.txt";
                string q = "";
                // To Store list into Text File
                for (int i = 0; i < teacher_list.Count; i++)
                {
                    q+= teacher_list[i].id + " " + teacher_list[i].name + " " +
teacher_list[i].std + " " + teacher_list[i].sec + "\n";
                }

                File.WriteAllText(p, q);
                Console.WriteLine("\nTeachers Details STORED in Text File!\n");
            }
            static void Main(string[] args)
            {
                // Creating list (teacher_list) for class Teacher
                List<Teacher> teacher_list = new List<Teacher>();
                Teacher t = null;
                int e = 1;
                do
                {
                    Console.WriteLine("\n====================Teacher's
Data====================\n");
                    Console.WriteLine("\nEnter the Operation to be performed:-\n\n1. ADD
Teacher's Data\n\n2. UPDATE Teacher's Data\n\n3. DELETE Teacher's Data\n\n4. DISPLAY
Teacher's Data\n\n5. STORE Teacher's Data\n\n6. EXIT\n");

                    int x = int.Parse(Console.ReadLine());

                    switch (x)
                    {
                        case 1: // To ADD teachers Details
                            GetData(t, teacher_list);

Console.WriteLine("\n====================**********================\n");
                            break;

                        case 2: // To UPDATE teachers Details
                            UpdateData(teacher_list);

Console.WriteLine("\n====================**********================\n");
                            break;

                        case 3: // To DELETE teacher's Details
                            RemoveData(teacher_list);

Console.WriteLine("\n====================**********================\n");
                            break;

                        case 4: // To DISPLAY teacher's Details
                            DisplayData(t, teacher_list);

Console.WriteLine("\n====================**********================\n");
                            break;
```

```
                        case 5: // To STORE teacher's Details in Text File
                            StoreData(teacher_list);

Console.WriteLine("\n====================**********================\n");
                            break;

                        case 6: // To EXIT from loop
                             e = 0;
                            break;

                        default: // For INVALID choice
                            Console.WriteLine("\nInvalid Operation...\n");
                            break;
                    }
                } while (e != 0);
            }
        }
    }
}
```