# Student Management System (Mini Project in C# .NET)

## Project Description

A simple Student Management System where you can:  
✔ Add Students  
✔ View Student List  
✔ Search Students  
✔ Delete Students  
✔ Save & Load Student Data

## Topics Covered

OOP Principles, Collections, LINQ, Exception Handling, File Handling, Async Programming

## Step 1: Define the Student Class

The following code defines a Student class with properties and methods.

using System;  
  
public class Student  
{  
 public int Id { get; set; }  
 public string Name { get; set; }  
 public int Age { get; set; }  
 public string Course { get; set; }  
  
 public Student(int id, string name, int age, string course)  
 {  
 Id = id;  
 Name = name;  
 Age = age;  
 Course = course;  
 }  
  
 public override string ToString()  
 {  
 return $"ID: {Id}, Name: {Name}, Age: {Age}, Course: {Course}";  
 }  
}

## Step 2: Implement StudentManager Class

The StudentManager class manages student records including adding, searching, deleting, saving, and loading data.

using System;  
using System.Collections.Generic;  
using System.IO;  
using System.Linq;  
using System.Text.Json;  
using System.Threading.Tasks;  
  
public class StudentManager  
{  
 private List<Student> students = new List<Student>();  
 private string filePath = "students.json";  
  
 public void AddStudent(Student student)  
 {  
 students.Add(student);  
 Console.WriteLine("Student added successfully!");  
 }  
  
 public void DisplayStudents()  
 {  
 if (students.Count == 0)  
 {  
 Console.WriteLine("No students found!");  
 return;  
 }  
  
 foreach (var student in students)  
 {  
 Console.WriteLine(student);  
 }  
 }  
  
 public void SearchStudent(string name)  
 {  
 var foundStudents = students.Where(s => s.Name.Contains(name, StringComparison.OrdinalIgnoreCase)).ToList();  
 if (foundStudents.Count == 0)  
 {  
 Console.WriteLine("No matching students found.");  
 }  
 else  
 {  
 foundStudents.ForEach(s => Console.WriteLine(s));  
 }  
 }  
  
 public void DeleteStudent(int id)  
 {  
 var student = students.FirstOrDefault(s => s.Id == id);  
 if (student != null)  
 {  
 students.Remove(student);  
 Console.WriteLine("Student removed successfully.");  
 }  
 else  
 {  
 Console.WriteLine("Student not found!");  
 }  
 }  
  
 public async Task SaveToFileAsync()  
 {  
 string json = JsonSerializer.Serialize(students);  
 await File.WriteAllTextAsync(filePath, json);  
 Console.WriteLine("Data saved successfully.");  
 }  
  
 public async Task LoadFromFileAsync()  
 {  
 if (File.Exists(filePath))  
 {  
 string json = await File.ReadAllTextAsync(filePath);  
 students = JsonSerializer.Deserialize<List<Student>>(json) ?? new List<Student>();  
 Console.WriteLine("Data loaded successfully.");  
 }  
 else  
 {  
 Console.WriteLine("No saved data found.");  
 }  
 }  
}

## Step 3: Main Program Execution

This is the entry point of the program where users interact with the system.

using System;  
using System.Threading.Tasks;  
  
class Program  
{  
 static async Task Main(string[] args)  
 {  
 StudentManager manager = new StudentManager();  
 await manager.LoadFromFileAsync();  
  
 while (true)  
 {  
 Console.WriteLine("\nStudent Management System");  
 Console.WriteLine("1. Add Student");  
 Console.WriteLine("2. View Students");  
 Console.WriteLine("3. Search Student");  
 Console.WriteLine("4. Delete Student");  
 Console.WriteLine("5. Save & Exit");  
 Console.Write("Enter your choice: ");  
  
 string choice = Console.ReadLine();  
  
 switch (choice)  
 {  
 case "1":  
 Console.Write("Enter ID: ");  
 int id = int.Parse(Console.ReadLine());  
 Console.Write("Enter Name: ");  
 string name = Console.ReadLine();  
 Console.Write("Enter Age: ");  
 int age = int.Parse(Console.ReadLine());  
 Console.Write("Enter Course: ");  
 string course = Console.ReadLine();  
 manager.AddStudent(new Student(id, name, age, course));  
 break;  
  
 case "2":  
 manager.DisplayStudents();  
 break;  
  
 case "3":  
 Console.Write("Enter Student Name to Search: ");  
 string searchName = Console.ReadLine();  
 manager.SearchStudent(searchName);  
 break;  
  
 case "4":  
 Console.Write("Enter Student ID to Delete: ");  
 int deleteId = int.Parse(Console.ReadLine());  
 manager.DeleteStudent(deleteId);  
 break;  
  
 case "5":  
 await manager.SaveToFileAsync();  
 return;  
  
 default:  
 Console.WriteLine("Invalid choice, please try again.");  
 break;  
 }  
 }  
 }  
}

## Project Features Covered

✅ OOP Principles  
✅ Collections  
✅ LINQ Queries  
✅ Exception Handling  
✅ File Handling (JSON)  
✅ Async Programming

## Future Enhancements

📌 Use a Database (SQL Server or SQLite) instead of JSON  
📌 Implement GUI using Windows Forms or WPF  
📌 Add more search & filter options  
📌 Implement a REST API using ASP.NET Core