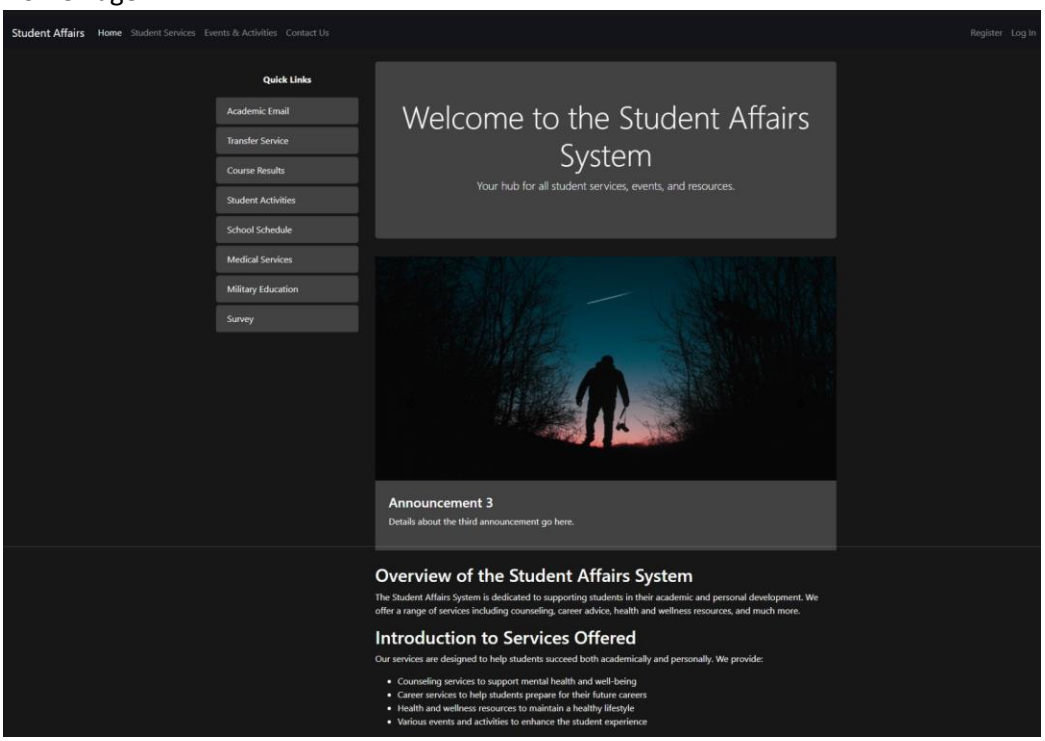
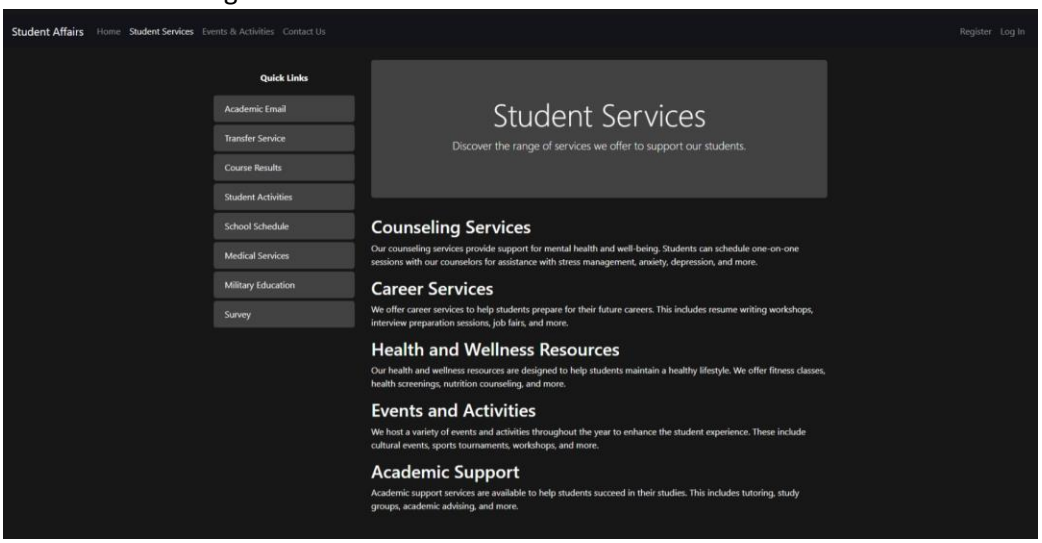
Session 2 → Task 1:

Source Code: https://github.com/Y-Baker/.NET_InnoTech/Session#2

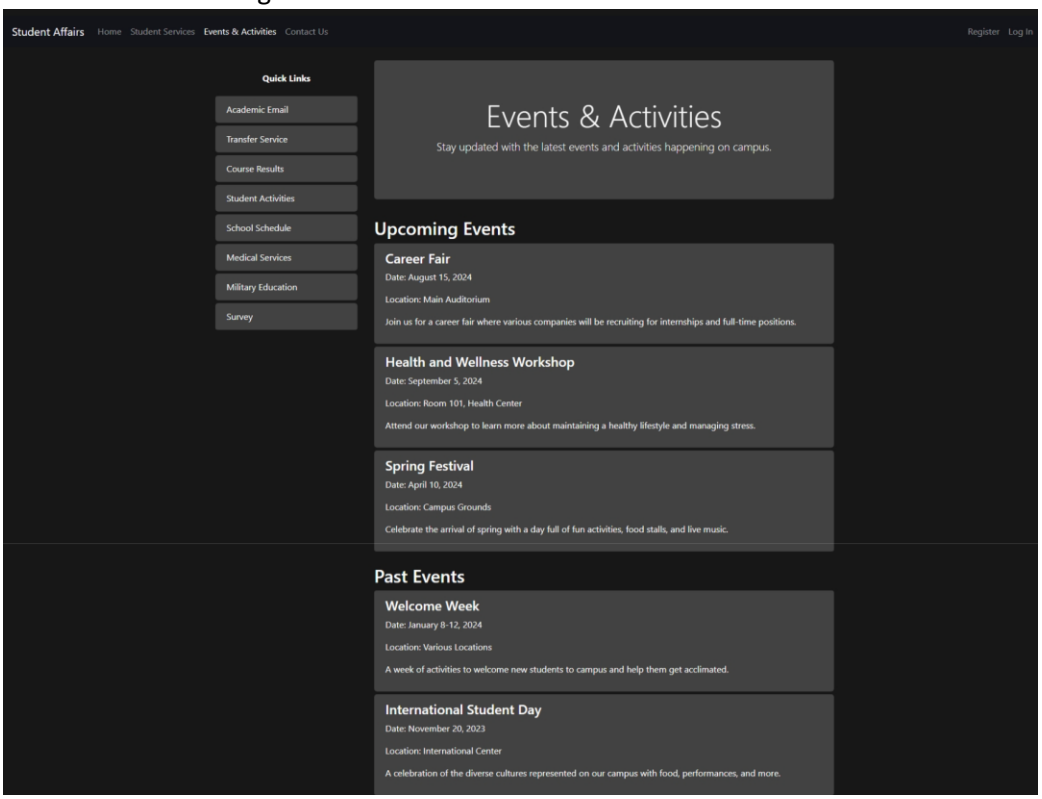
1- Home Page:



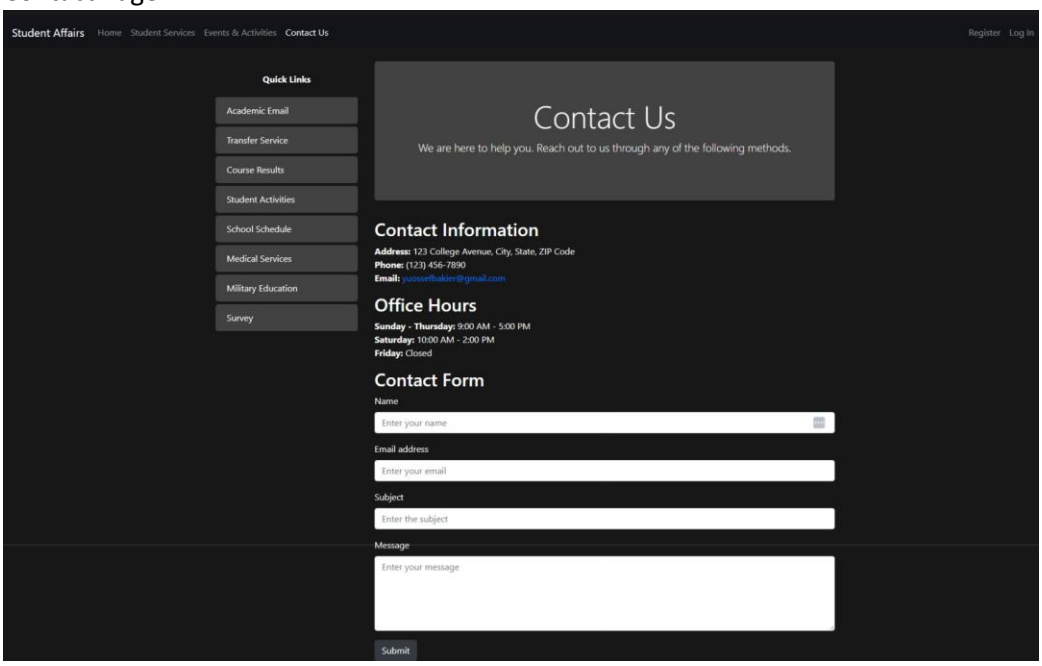
2- Student Service Page



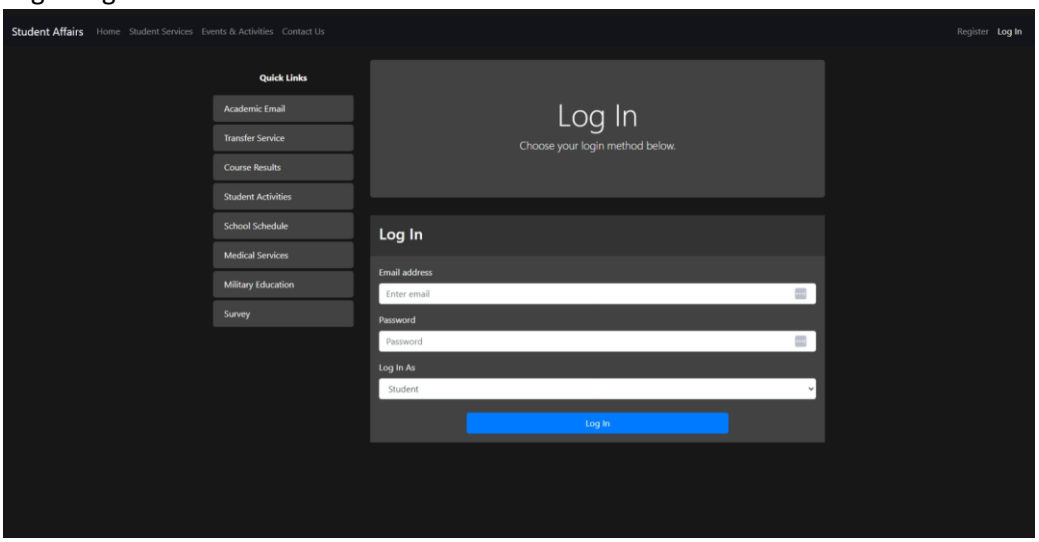
3- Events & Activities Page



4- Contact Page



5- Login Page


Student Affairs
Home
Student Services
Events & Activities
Contact Us
Register
Log In
Quick Links
Academic Email
Transfer Service
Course Results
Student Activities
School Schedule
Medical Services
Military Education
Survey
Log In
Choose your login method below.
Log In
Email address
Enter email
Password
Password
Log In As
Student
Log In

Session 2 → Task 2:

1- CPU usage if the case of for loop will increase with increasing the input. On the other hand, CPU usage will be constant whatever the input is.

2- Validation

```
Console.WriteLine("Hello, Please enter the maximum number you need to add");

string? maximumNumberFromUser;

maximumNumberFromUser = Console.ReadLine();
//TODO:: add validation for the maximumNumber
bool isNumber = int.TryParse(maximumNumberFromUser, out int maximumNumber);
if (!isNumber)
{
    Console.WriteLine("Please enter a valid number");
    return;
}

Console.WriteLine($"Are you sure you need to get the addition up to {maximumNumberFromUser}?");

int sum = 0;
for (int i = 1; i <= maximumNumber; i++)
{
    sum = sum + i;
}

int seriesSum = maximumNumber * (maximumNumber + 1) / 2;

Console.WriteLine($"The sum of numbers from 1 to {maximumNumber} = {sum}");
Console.WriteLine($"The sum of numbers from 1 to {maximumNumber} = {seriesSum} using series form");
```

Session 2 → Task 3:

1- Remove the last comma

```
for (int i = 1; i <= maximumNumber; i++)
{
    // Remove the last comma
    if (i == maximumNumber)
        numbersNewApproach.Append(i);
    else
        numbersNewApproach.Append($"{i},");
}
```

2- Complexity of string (array) method:
As strings are immutable so at each iterate we need to create a new string which take time and process. In the first iteration we no more time as empty string O(1), in the second iteration we need one to copy the string and one to append O(2) and so on

The Complexity will be O(n * (n+1) / 2) which can simplified to O(n^2)

3- How to know about how string builder work under the hood:
- Take a look at Microsoft documentation.
- Search about it you can find your answer on stack overflow or any blogs like this https://www.stevejgordon.co.uk/how-does-the-stringbuilder-work-in-dotnet-part-2-understanding-the-overhead.