NEW SUMMIT COLLEGE

(Affiliated to Tribhuvan University)

Lab Report of

NET Centric Computing

CSC 367

Bachelors of Science in Computer Science and Information Technology

Institute of Science and Technology

Submitted by:

Name: Lokesh Datta Ojha

Roll No: 20

Semester: VI

Program: BSc CSIT

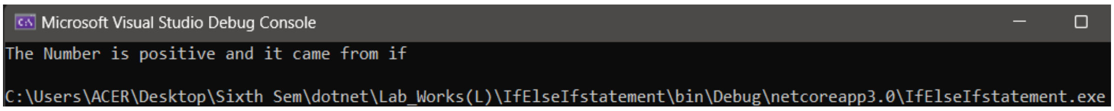
Submitted to:

Name: Bishow Shrestha

Chapter1:

If, else if statement

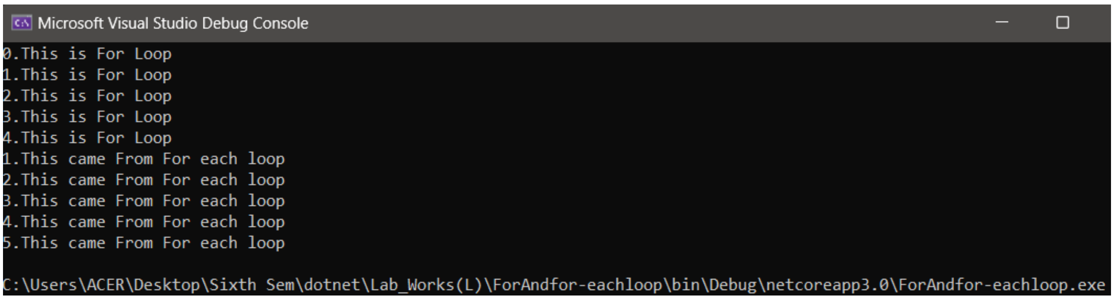
```
using System;
namespace IfElseIfstatement
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int x = 5;
            if (x > 0)
            {
                Console.WriteLine("The Number is positive and it came from if ");
            }
            else if (x < 0)
            {
                Console.WriteLine("The Number is negative and it came from else if");
            }
            else
            {
                Console.WriteLine("The Number is zero and it came from else");
            }
            Console.ReadKey();
        }
    }
}
```



```
The Number is positive and it came from if

C:\Users\ACER\Desktop\Sixth Sem\dotnet\Lab_Works(L)\IfElseIfstatement\bin\Debug\netcoreapp3.0\IfElseIfstatement.exe
```

For and for-each loop

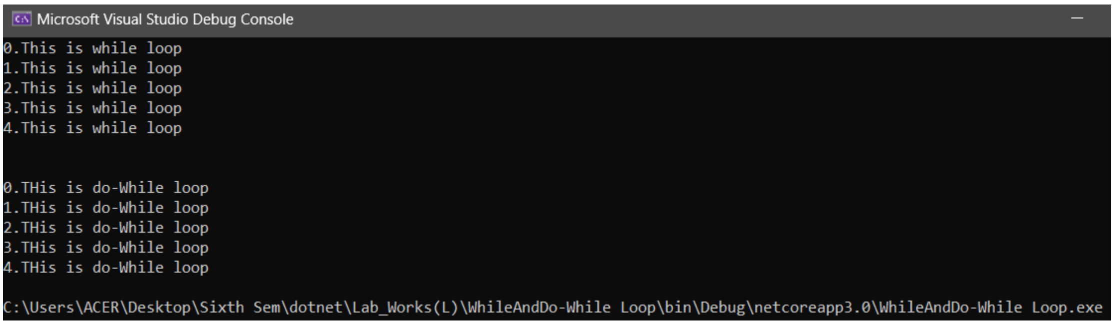
```
using System;

namespace ForAndforEachLoop
{
    internal class Program
    {
        static void Main(string[] args)
        {
            for (int i = 0; i < 5; i++)
            {
                Console.WriteLine(i+".This is For Loop");
            }

            int[] numbers = { 1, 2, 3, 4, 5 };
            foreach (int num in numbers)
            {
                Console.WriteLine(num+ ".This came From For each loop");
            }
        }
    }
}
```



```
0.This is For Loop
1.This is For Loop
2.This is For Loop
3.This is For Loop
4.This is For Loop
1.This came From For each loop
2.This came From For each loop
3.This came From For each loop
4.This came From For each loop
5.This came From For each loop

C:\Users\ACER\Desktop\Sixth Sem\dotnet\Lab_Works(L)\ForAndfor-eachloop\bin\Debug\netcoreapp3.0\ForAndfor-eachloop.exe
```

while and do while loop

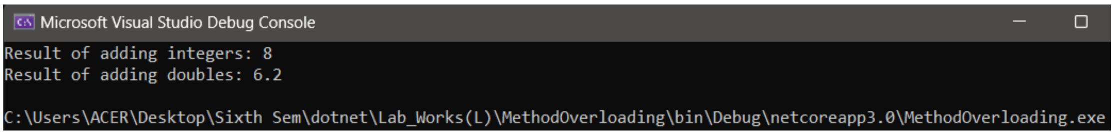
```
using System;

namespace WhileAndDo_While_Loop
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int i = 0;
            while (i < 5)
            {
                Console.WriteLine(i+".This is while loop");
                i++;
            }
            Console.WriteLine("\n");
            int j = 0;
            do
            {
                Console.WriteLine(j+".THis is do-While loop");
                j++;
            } while (j < 5);
        }
    }
}
```



```
0.This is while loop
1.This is while loop
2.This is while loop
3.This is while loop
4.This is while loop


0.THis is do-While loop
1.THis is do-While loop
2.THis is do-While loop
3.THis is do-While loop
4.THis is do-While loop

C:\Users\ACER\Desktop\Sixth Sem\dotnet\Lab_Works(L)\WhileAndDo-While Loop\bin\Debug\netcoreapp3.0\WhileAndDo-While Loop.exe
```

Method Overloading

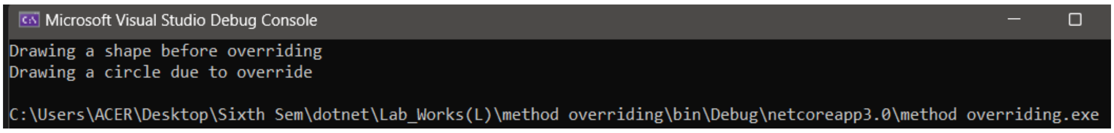
```
using System;

namespace MethodOverloading
{
    internal class Program
    {
        static void Main(string[] args)
        {
            MathOp math = new MathOp();

            int result1 = math.Addition(5, 3);
            Console.WriteLine("Result of adding integers: " + result1);

            double result2 = math.Addition(2.5, 3.7);
            Console.WriteLine("Result of adding doubles: " + result2);
        }

        class MathOp
        {
            public int Addition(int a, int b)
            {
                return a + b;
            }

            public double Addition(double a, double b)
            {
                return a + b;
            }
        }
    }
}
```



```
Result of adding integers: 8
Result of adding doubles: 6.2

C:\Users\ACER\Desktop\Sixth Sem\dotnet\Lab_Works(L)\MethodOverloading\bin\Debug\netcoreapp3.0\MethodOverloading.exe
```

Method Overriding

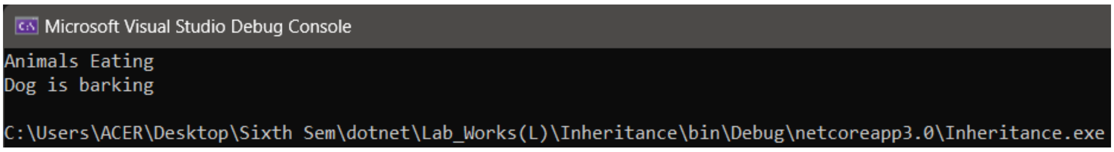
```
using System;

namespace method_overriding
{
   internal class Program
   {
      static void Main(string[] args)
      {
         Shape shape = new Shape();
         shape.Draw();

         Circle circle = new Circle();
         circle.Draw();
      }

      class Shape
      {
         public virtual void Draw()
         {
            Console.WriteLine("Drawing a shape before overriding");
         }
      }

      class Circle : Shape
      {
         public override void Draw()
         {
            Console.WriteLine("Drawing a circle due to override");
         }
      }
   }
}
```



```
Drawing a shape before overriding
Drawing a circle due to override

C:\Users\ACER\Desktop\Sixth Sem\dotnet\Lab_Works(L)\method overriding\bin\Debug\netcoreapp3.0\method overriding.exe
```

Inheritance example

```
using System;

namespace Inheritance
{
   internal class Program
   {
      static void Main(string[] args)
      {
         Dog dog = new Dog();
         dog.Eat();
         dog.Bark();
      }
   }
   class Animal
   {
      public void Eat()
      {
         Console.WriteLine("Animals Eating");
      }
   }

   class Dog : Animal
   {
      public void Bark()
      {
         Console.WriteLine("Dog is barking");
      }
   }
}
```



```
Animals Eating
Dog is barking

C:\Users\ACER\Desktop\Sixth Sem\dotnet\Lab_Works(L)\Inheritance\bin\Debug\netcoreapp3.0\Inheritance.exe
```

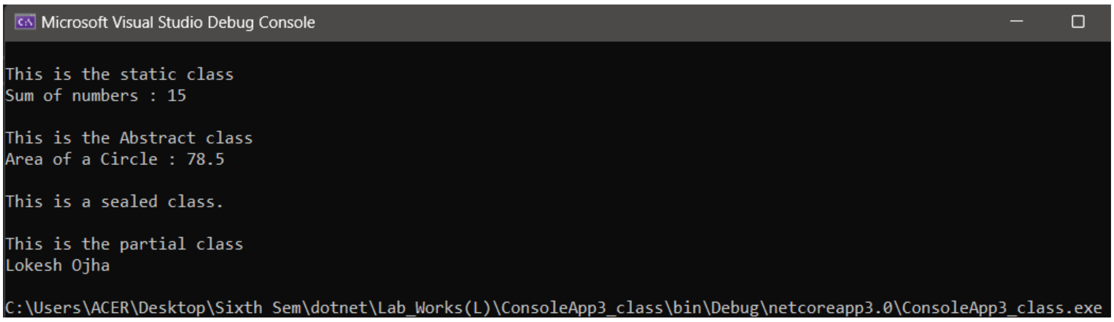
Chapter 2:
Class and it type with examples

In Main.cs

```
using System;
using System.Collections.Generic;
using System.Text;
using static ConsoleApp3_class.Program;
namespace ConsoleApp3_class
{
  class Program
  {
    static void Main()
    {
      int sum = Mathsum.Add(10,5);
      Console.WriteLine("\nThis is the static class ");
      Console.WriteLine($"Sum of numbers : {sum}");
      Circle circle = new Circle { Radius = 5 };
      double area = circle.CalculateArea();
      Console.WriteLine("\nThis is the Abstract class ");
      Console.WriteLine($"Area of a Circle : {area}");
      FinalClass finalObj = new FinalClass();
      Console.WriteLine();
      finalObj.DisplayMessage();
      Student student = new Student { FirstName = "Lokesh", LastName = "Ojha"};
      Console.WriteLine("\nThis is the partial class ");
      student.DisplayFullName();
    }
  }
}
```

In Program.cs

```
using System;

namespace ConsoleApp3_class
{
    public static class Mathsum
    {
      public static int Add(int a, int b)
      {
        return a + b;
      }
    }
    public abstract class Area
    {
        public abstract double CalculateArea();
    }

    public class Circle : Area
    {
        public double Radius { get; set; }
        public override double CalculateArea()
        {
            double Pi = 3.14;
            return Pi* Radius * Radius;
        }
    }
    public sealed class FinalClass
    {
        public void DisplayMessage()
        {
            Console.WriteLine("This is a sealed class.");
        }
    }
    public partial class Student
    {
        public string FirstName { get; set; }
        public string LastName { get; set; }
    }
    public partial class Student
    {
        public void DisplayFullName()
        {
            Console.WriteLine($"{FirstName} {LastName}");
        }
    }
}
```

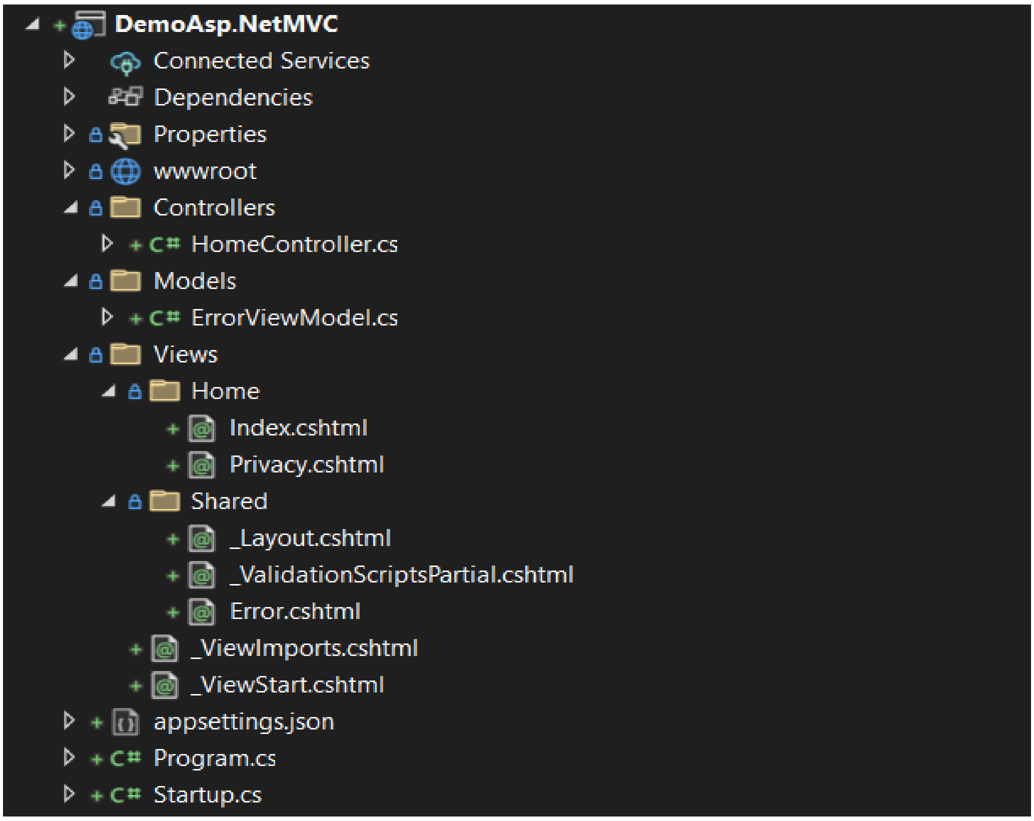


```
This is the static class
Sum of numbers : 15

This is the Abstract class
Area of a Circle : 78.5

This is a sealed class.

This is the partial class
Lokesh Ojha

C:\Users\ACER\Desktop\Sixth Sem\dotnet\Lab_Works(L)\ConsoleApp3_class\bin\Debug\netcoreapp3.0\ConsoleApp3_class.exe
```

Chapter 3:

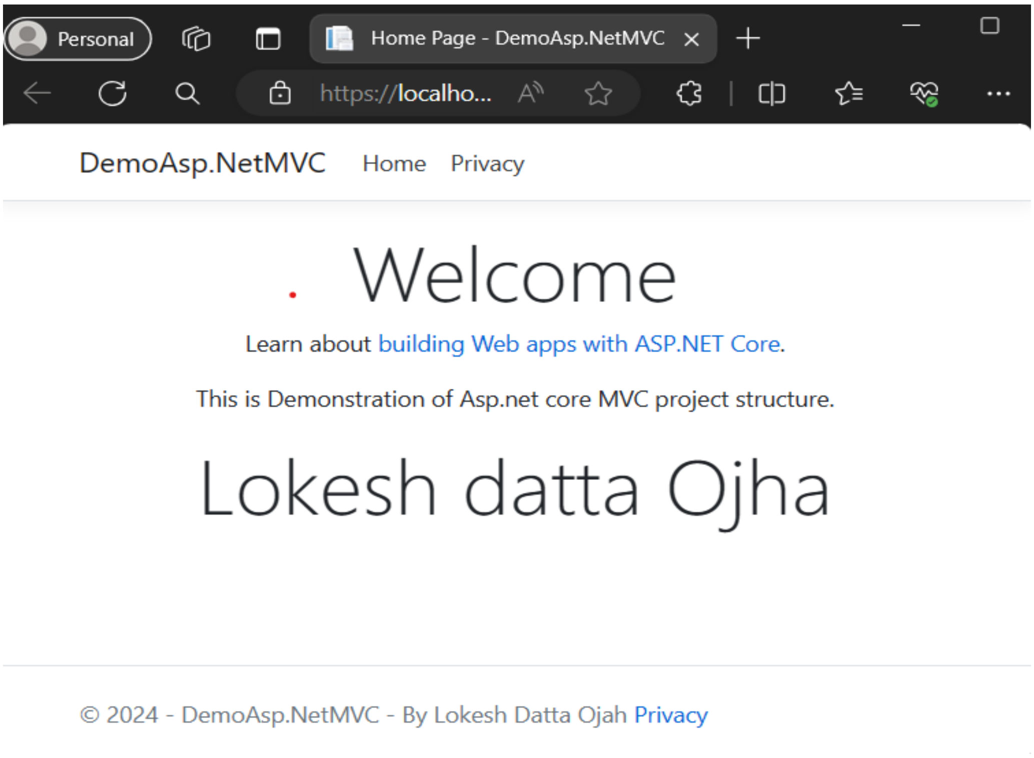
Demonstration of Asp.net core MVC project structure.

Project Folder Structure



```
DemoAsp.NetMVC
    Connected Services
    Dependencies
    Properties
    wwwroot
    Controllers
        HomeController.cs
    Models
        ErrorViewModel.cs
    Views
        Home
            Index.cshtml
            Privacy.cshtml
        Shared
            _Layout.cshtml
            _ValidationScriptsPartial.cshtml
            Error.cshtml
        _ViewImports.cshtml
        _ViewStart.cshtml
    appsettings.json
    Program.cs
    Startup.cs
```

Default Page



DemoAsp.NetMVC Home Privacy

Welcome

Learn about building Web apps with ASP.NET Core.

This is Demonstration of Asp.net core MVC project structure.

Lokesh datta Ojha

© 2024 - DemoAsp.NetMVC - By Lokesh Datta Ojah Privacy

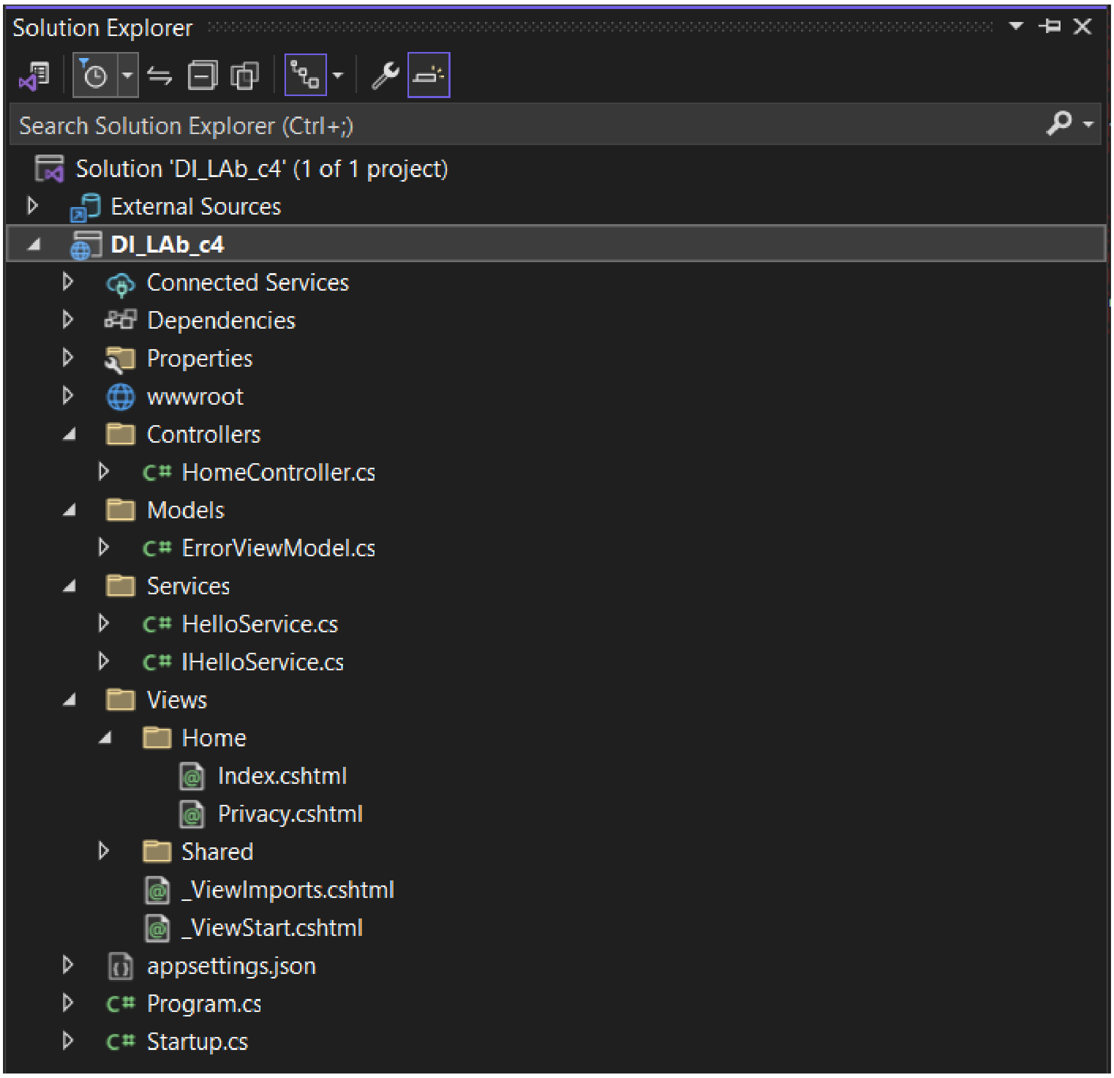
Chapter 4

WAP to demonstrate dependency injection and register them in .net core.

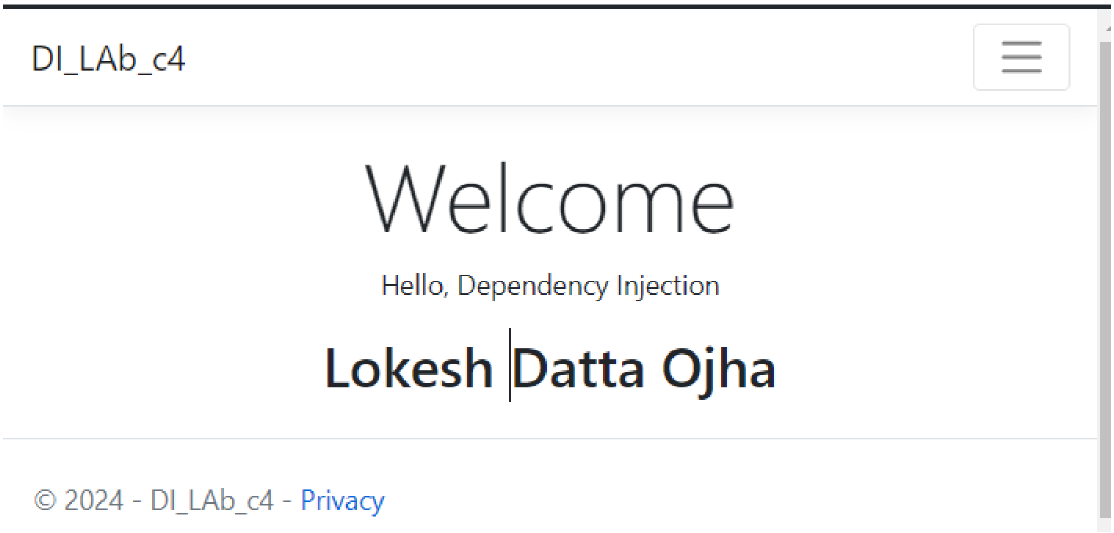
HomeController.cs

```
using DI_LAb_c4.Models;
using DI_LAb_c4.Services;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Extensions.Logging;
using System.Diagnostics;

namespace DI_LAb_c4.Controllers
{
   public class HomeController : Controller
   {
      private readonly ILogger<HomeController> _logger;
      private readonly IHelloService _helloService;

      public HomeController(ILogger<HomeController> logger, IHelloService
helloService)
      {
         _logger = logger;
         _helloService = helloService;
      }

      public IActionResult Index()
      {
         ViewBag.Message = _helloService.SayHello();
         return View();
      }

      public IActionResult Privacy()
      {
         return View();
      }
      [ResponseCache(Duration = 0, Location = ResponseCacheLocation.None,
NoStore = true)]
      public IActionResult Error()
      {
         return View(new ErrorViewModel { RequestId = Activity.Current?.Id ??
HttpContext.TraceIdentifier });
      }
   }
}
```

Project Folder Structure



```
Solution 'DI_LAb_c4' (1 of 1 project)
  External Sources
  DI_LAb_c4
    Connected Services
    Dependencies
    Properties
    wwwroot
    Controllers
      HomeController.cs
    Models
      ErrorViewModel.cs
    Services
      HelloService.cs
      IHelloService.cs
    Views
      Home
        Index.cshtml
        Privacy.cshtml
      Shared
      _ViewImports.cshtml
      _ViewStart.cshtml
    appsettings.json
    Program.cs
    Startup.cs
```



DI_LAb_c4

Welcome

Hello, Dependency Injection

Lokesh Datta Ojha

© 2024 - DI_LAb_c4 - Privacy

WAP to demonstrate different types of return types of a controller action method.

DemoController.cs

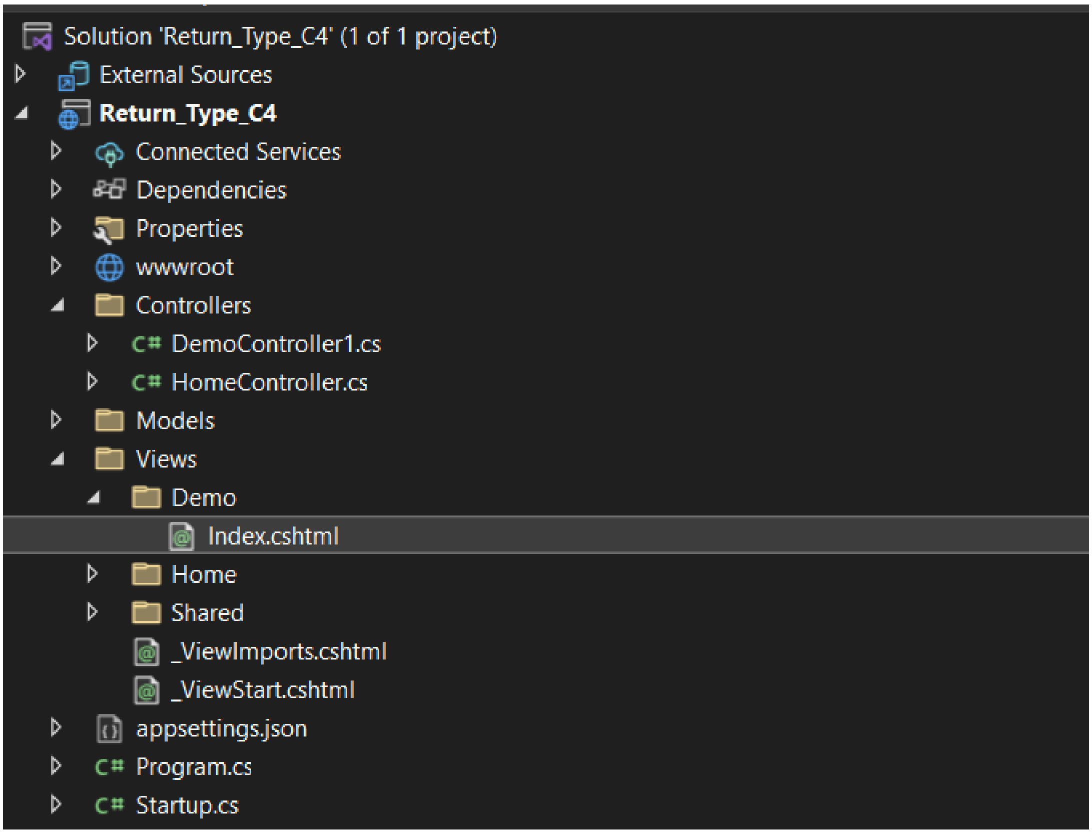
```
using Microsoft.AspNetCore.Mvc;

namespace Return_Type_C4.Controllers
{
  public class DemoController : Controller
  {
    // Return a view
    public IActionResult Index()
    {
      return View();
    }

    // Return JSON data
    public IActionResult JsonData()
    {
      var data = new { Name = "Lokesh Datta Ojha", Age = 30 };
      return Json(data);
    }

    // Return plain text content
    public IActionResult PlainText()
    {
      return Content("This is plain text content.");
    }

    // Return a file (assuming you have a file named sample.txt in wwwroot/files)
    public IActionResult FileDownload()
    {
      var filePath = "wwwroot/files/sample.txt";
      var contentType = "text/plain";
      var fileName = "sample.txt";
      return PhysicalFile(filePath, contentType, fileName);
    }

    // Return a redirect to another action
    public IActionResult RedirectToAction()
    {
      return RedirectToAction("Index", "Home");
    }

    // Return a redirect to an external URL
```

```
        public IActionResult RedirectToUrl()
        {
            return Redirect("https://www.Google.com");
        }

        // Return a status code
        public IActionResult NotFoundStatus()
        {
            return NotFound();
        }

        // Return no content
        public IActionResult NoContentResult()
        {
            return NoContent();
        }
    }
}
```

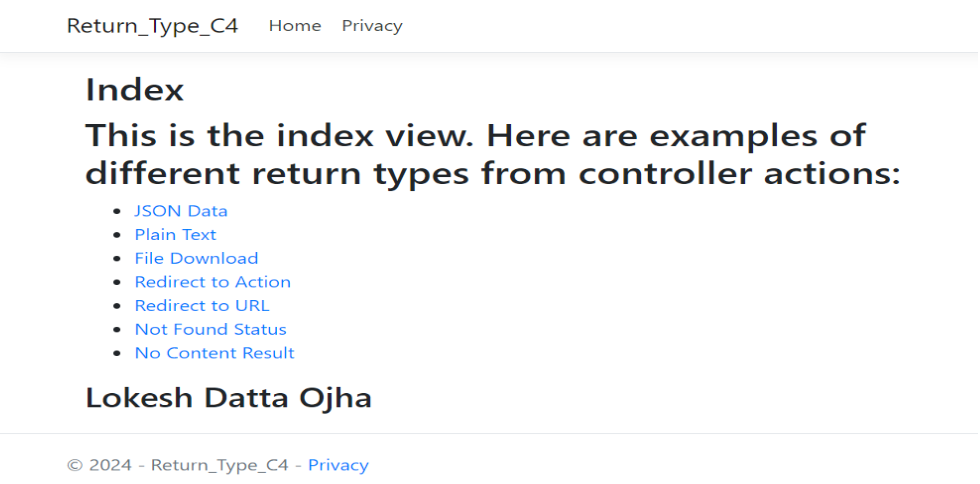
Index.cshtml

```
    @{
    ViewData["Title"] = "Index";
}

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewData["Title"]</title>
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
</head>
<body>
    <div class="container">
        <h2>Index</h2>
        <h2>This is the index view. Here are examples of different return types from
controller actions:</h2>

        <ul>
            <li><a href="/Demo/JsonData">JSON Data</a></li>
            <li><a href="/Demo/PlainText">Plain Text</a></li>
            <li><a href="/Demo/FileDownload">File Download</a></li>
            <li><a href="/Demo/RedirectToActionDemo">Redirect to Action</a></li>
            <li><a href="/Demo/RedirectToUrl">Redirect to URL</a></li>
            <li><a href="/Demo/NotFoundStatus">Not Found Status</a></li>
            <li><a href="/Demo/NoContentResult">No Content Result</a></li>
        </ul>
        <h3>Lokesh Datta Ojha</h3>
    </div>
</body>
</html>
```

Project Folder Structure



```
Solution 'Return_Type_C4' (1 of 1 project)
▷ External Sources
◢ Return_Type_C4
    ▷ Connected Services
    ▷ Dependencies
    ▷ Properties
    ▷ wwwroot
    ◢ Controllers
        ▷ C# DemoController1.cs
        ▷ C# HomeController.cs
    ▷ Models
    ◢ Views
        ◢ Demo
            Index.cshtml
        ▷ Home
        ▷ Shared
          _ViewImports.cshtml
          _ViewStart.cshtml
    ▷ appsettings.json
    ▷ C# Program.cs
    ▷ C# Startup.cs
```



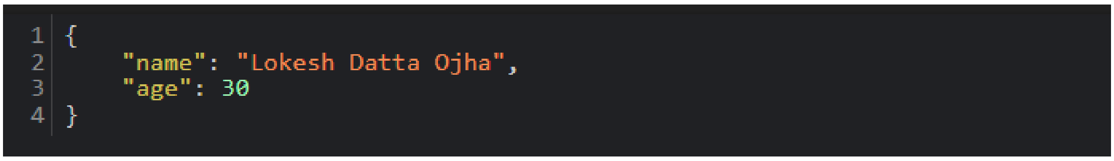
Index

This is the index view. Here are examples of different return types from controller actions:

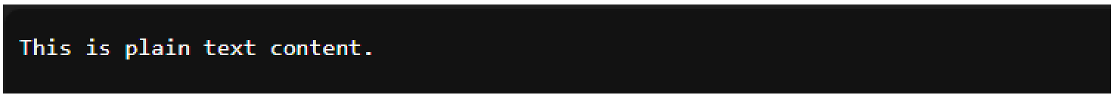
- JSON Data
- Plain Text
- File Download
- Redirect to Action
- Redirect to URL
- Not Found Status
- No Content Result

Lokesh Datta Ojha

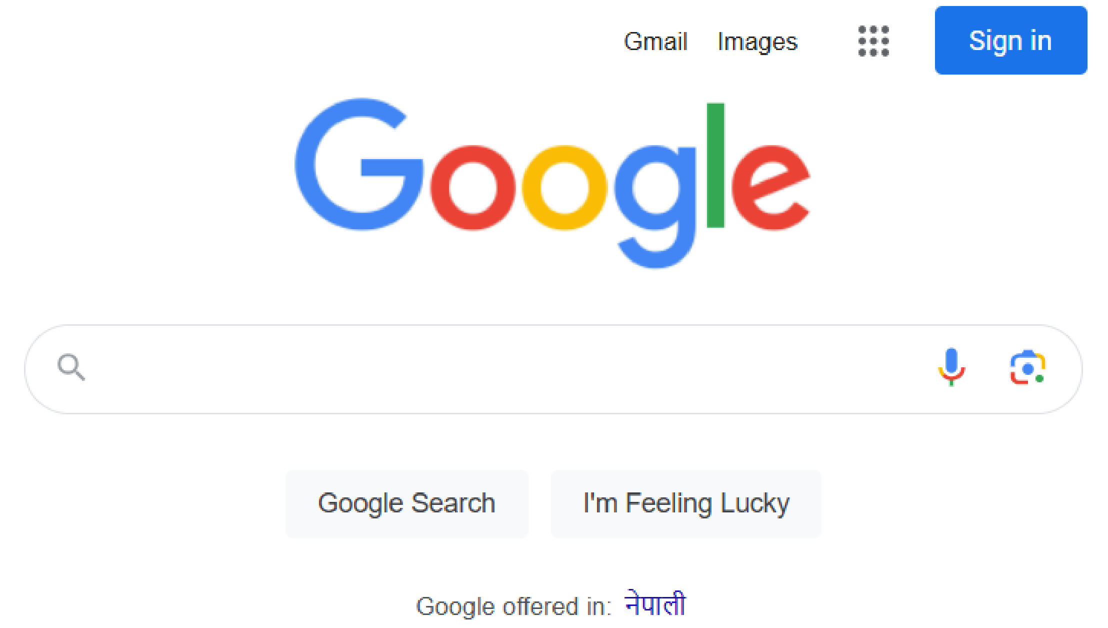
© 2024 - Return_Type_C4 - Privacy

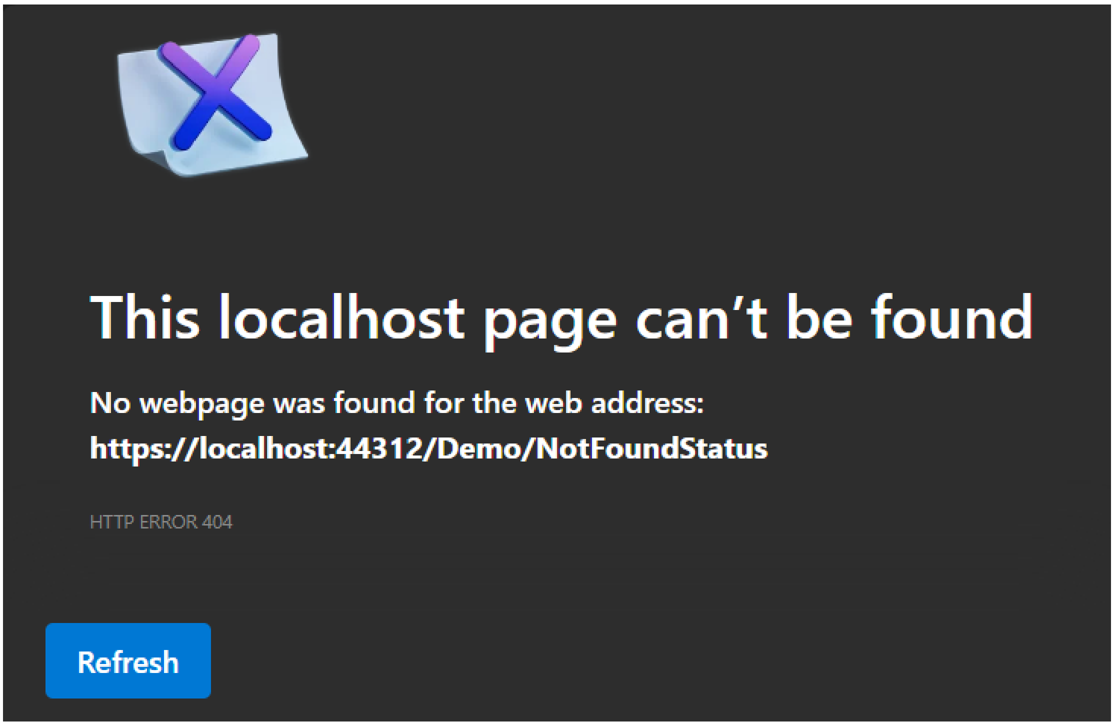


```
{
    "name": "Lokesh Datta Ojha",
    "age": 30
}
```



```
This is plain text content.
```





This localhost page can’t be found

No webpage was found for the web address: **https://localhost:44312/Demo/NotFoundStatus**

HTTP ERROR 404

Use Viewbag, viewdata, and temp data to pass data from controller to views.

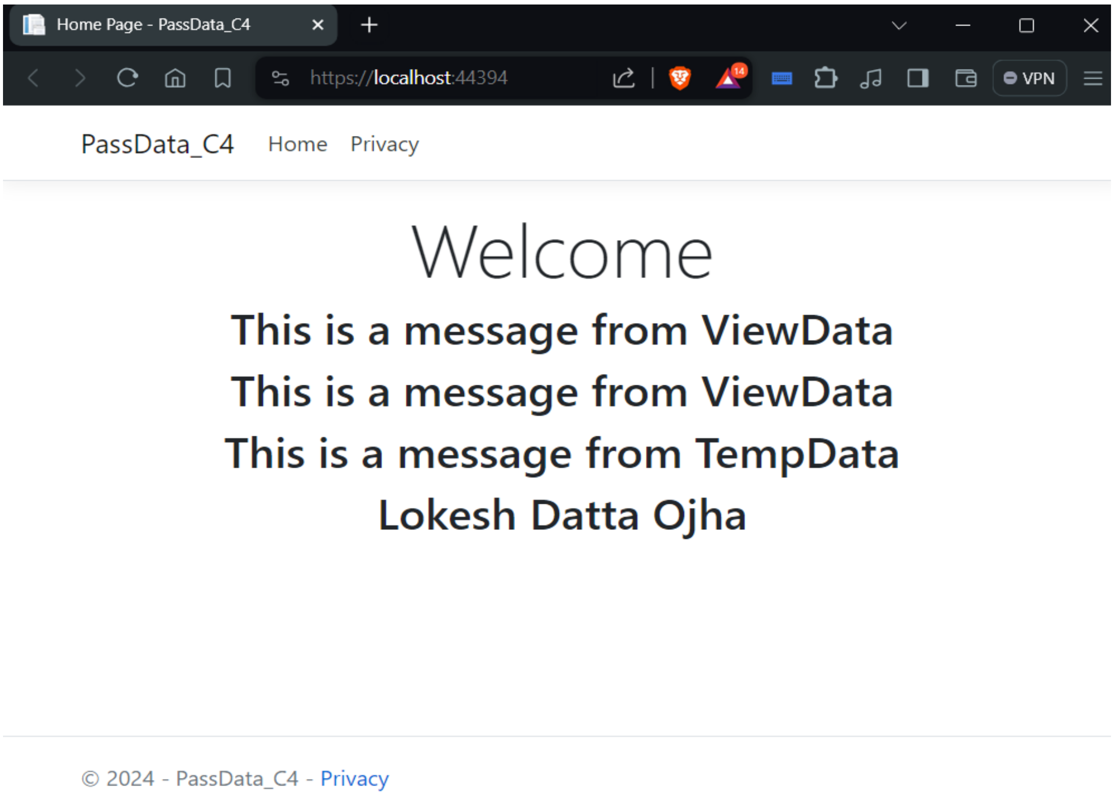
HomeControllwe.cs

```
public IActionResult Index()
        {
            ViewBag.Message = "This is a message from ViewBag";
            ViewData["Message"] = "This is a message from ViewData";
            TempData["Message"] = "This is a message from TempData";
            return View();
        }
```

Index.cshtml

```
@{
    ViewData["Title"] = "Home Page";
    var message = TempData["Message"] as string;
}
<div class="text-center">
    <h1 class="display-4">Welcome</h1>
    <h2>@ViewBag.Message</h2>
    <h2>@ViewData["Message"]</h2>
    <h2>@message</h2>
    <h2>Lokesh Datta Ojha<h2>
</div>
```

Output



Welcome

This is a message from ViewData

This is a message from ViewData

This is a message from TempData

Lokesh Datta Ojha

© 2024 - PassData_C4 - Privacy

Use a Model with appropriate data annotation and display a validation error msg on the page if the input is invalid.

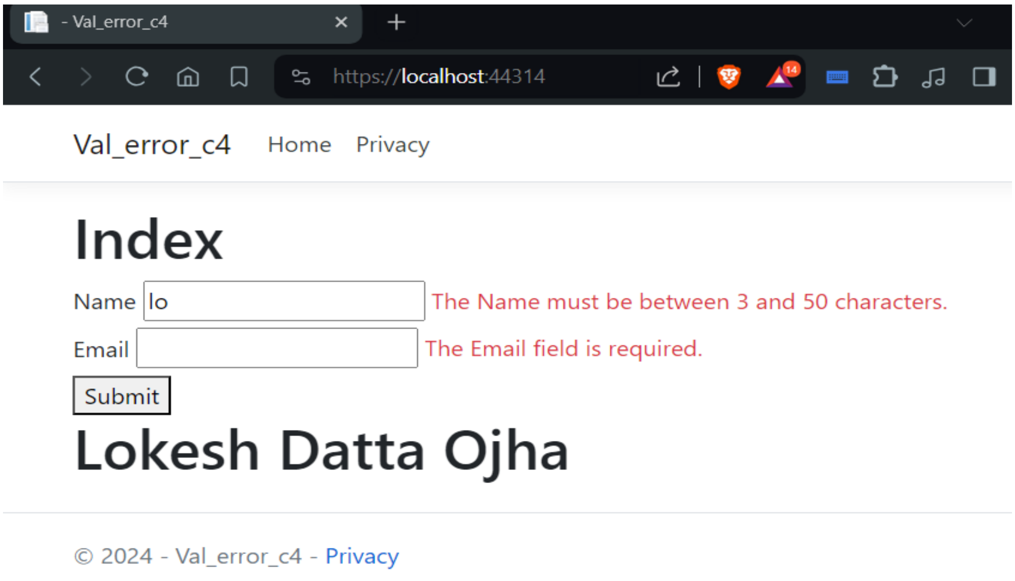
ErrorViewModel.cs

```
  public class User
  {
     [Required(ErrorMessage = "The Name field is required.")]
     [StringLength(50, MinimumLength = 5, ErrorMessage = "The Name must be
between 3 and 50 characters.")]
     public string Name { get; set; }

     [Required(ErrorMessage = "The Email field is required.")]
     [EmailAddress(ErrorMessage = "Invalid email format.")]
     [StringLength(50, ErrorMessage = "The Email must not exceed 100 characters.")]
     public string Email { get; set; }
  }
```

Index.cshtml

```
@model Val_error_c4.Models.User

<!DOCTYPE html>
<html>
<head>
  <title>Index</title>
</head>
<body>
  <h1>Index</h1>
  <form method="post">
     <div>
        <label asp-for="Name"></label>
        <input asp-for="Name" />
        <span asp-validation-for="Name " class="text-danger"></span>
     </div>
     <div>
        <label asp-for="Email"></label>
        <input asp-for="Email" />
        <span asp-validation-for="Email" class="text-danger"></span>
     </div>
     <button type="submit">Submit</button>
  </form>
  <h1>Lokesh Datta Ojha</h1>
</body>
</html>
```

Output



Val_error_c4 Home Privacy

Index

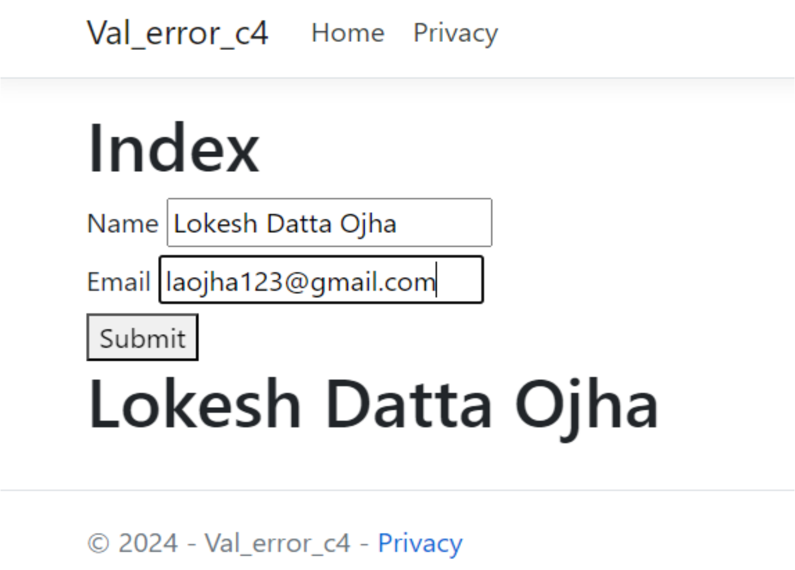
Name lo The Name must be between 3 and 50 characters.

Email The Email field is required.

Submit

Lokesh Datta Ojha

© 2024 - Val_error_c4 - Privacy



Val_error_c4 Home Privacy

Index

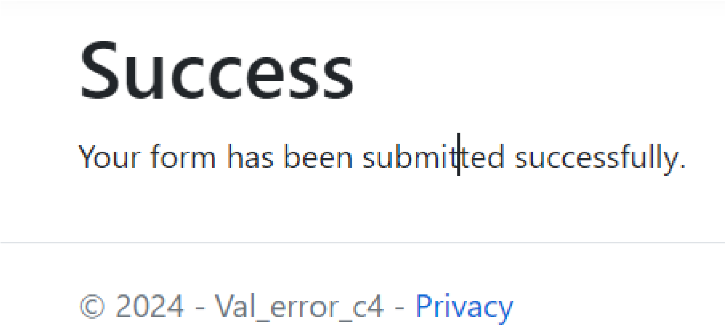
Name Lokesh Datta Ojha

Email laojha123@gmail.com

Submit

Lokesh Datta Ojha

© 2024 - Val_error_c4 - Privacy



Success

Your form has been submitted successfully.

© 2024 - Val_error_c4 - Privacy

Chapter 5:
Create a table named BAG(ID, Brand, Price, Date). Perform the following operations using Entity Framework Core.Update the Price of those Books to 2000 which are published on 2022.Delete the Books which are published in 1981Retrieve the Brand of the Book having a price greater than 500.
Bags.cs

```
using System;

namespace BooksManag_c4.Models
{
   public class Bags
   {

         public int ID { get; set; }
         public string Brand { get; set; }
         public decimal Price { get; set; }
         public DateTime Date { get; set; }
   }

}
```

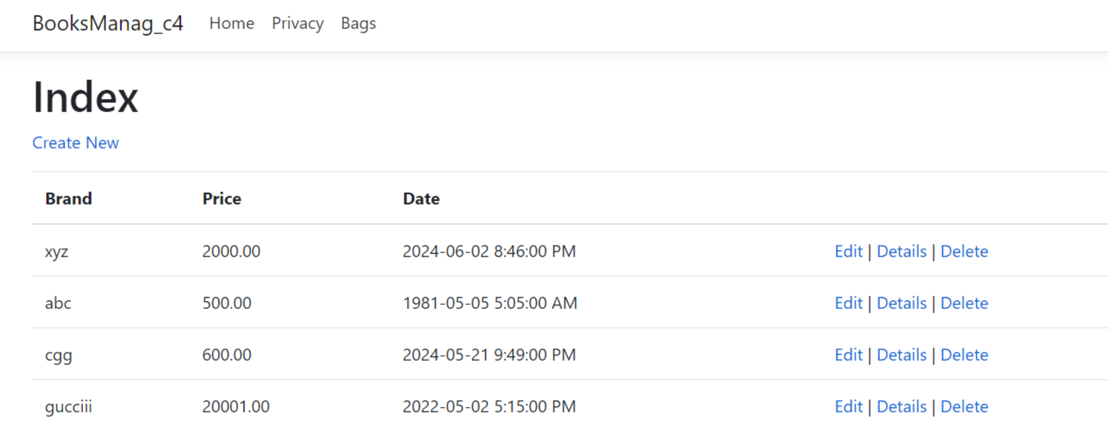
BagController.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.Rendering;
using Microsoft.EntityFrameworkCore;
using BooksManag_c4.Data;
using BooksManag_c4.Models;

namespace BooksManag_c4.Controllers
{
   public class BagsController : Controller
   {
      private readonly BagsContext _context;

      public BagsController(BagsContext context)
      {
         _context = context;
      }
```

```
        // GET: Bags
        public async Task<IActionResult> Index()
        {
            return View(await _context.Bags.ToListAsync());
        }

        // GET: Bags/Details/5
        public async Task<IActionResult> Details(int? id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var bags = await _context.Bags
                .FirstOrDefaultAsync(m => m.ID == id);
            if (bags == null)
            {
                return NotFound();
            }

            return View(bags);
        }

        // GET: Bags/Create
        public IActionResult Create()
        {
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Create([Bind("ID,Brand,Price,Date")] Bags
bags)
        {
            if (ModelState.IsValid)
            {
                _context.Add(bags);
                await _context.SaveChangesAsync();
                return RedirectToAction(nameof(Index));
            }
            return View(bags);
        }
        // GET: Bags/Edit/5
        public async Task<IActionResult> Edit(int? id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var bags = await _context.Bags.FindAsync(id);
            if (bags == null)
            {
                return NotFound();
            }
            return View(bags);
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Edit(int id, [Bind("ID,Brand,Price,Date")]
Bags bags)
        {
            if (id != bags.ID)
            {
                return NotFound();
            }

            if (ModelState.IsValid)
            {
                try
                {
                    _context.Update(bags);
                    await _context.SaveChangesAsync();
                }
                catch (DbUpdateConcurrencyException)
                {
                    if (!BagsExists(bags.ID))
                    {
                        return NotFound();
                    }
                    else
                    {
                        throw;
                    }
                }
        return RedirectToAction(nameof(Index));
    }
    return View(bags);
}

// GET: Bags/Delete/5
public async Task<IActionResult> Delete(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var bags = await _context.Bags
        .FirstOrDefaultAsync(m => m.ID == id);
    if (bags == null)
    {
        return NotFound();
    }

    return View(bags);
}

// POST: Bags/Delete/5
[HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]
public async Task<IActionResult> DeleteConfirmed(int id)
{
    var bags = await _context.Bags.FindAsync(id);
    _context.Bags.Remove(bags);
    await _context.SaveChangesAsync();
    return RedirectToAction(nameof(Index));
}

private bool BagsExists(int id)
{
    return _context.Bags.Any(e => e.ID == id);
}
[HttpPost]
public IActionResult UpdatePrice()
{
    var bagsToUpdate = _context.Bags.Where(b => b.Date.Year == 2022);
    foreach (var bag in bagsToUpdate)
    {
```

```
                bag.Price = 2000;
            }
            _context.SaveChanges();

            return Ok("Prices updated.");
        }

        // Delete the Bags which are published in 1981
        [HttpPost]
        public IActionResult DeleteBags()
        {
            var bagsToDelete = _context.Bags.Where(b => b.Date.Year == 1981);
            _context.Bags.RemoveRange(bagsToDelete);
            _context.SaveChanges();

            return Ok("Bags deleted.");
        }

        // Retrieve the Brand of the Bags having a price greater than 500
        [HttpGet]
        public IActionResult GetBrands()
        {
            var brands = _context.Bags
                         .Where(b => b.Price > 500)
                         .Select(b => b.Brand)
                         .ToList();

            return Ok(brands);
        }
    }
}
```

Output:

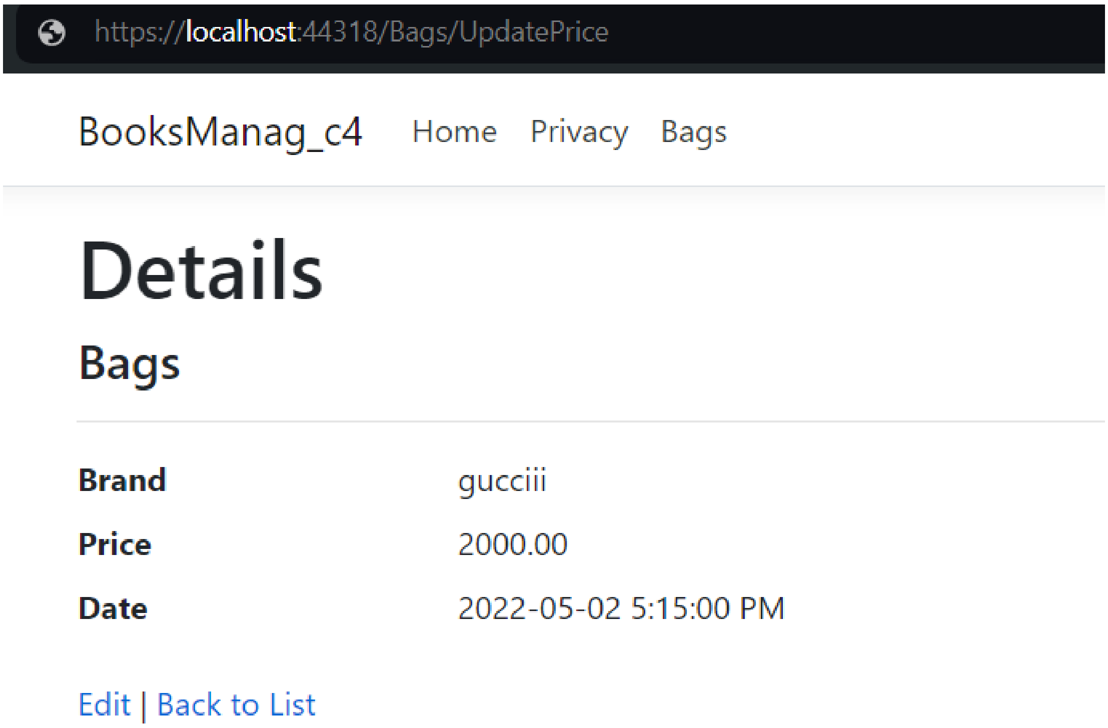


Index

Create New

Brand	Price	Date	
xyz	2000.00	2024-06-02 8:46:00 PM	Edit | Details | Delete
abc	500.00	1981-05-05 5:05:00 AM	Edit | Details | Delete
cgg	600.00	2024-05-21 9:49:00 PM	Edit | Details | Delete
gucciii	20001.00	2022-05-02 5:15:00 PM	Edit | Details | Delete

Update



https://localhost:44318/Bags/UpdatePrice

BooksManag_c4 Home Privacy Bags

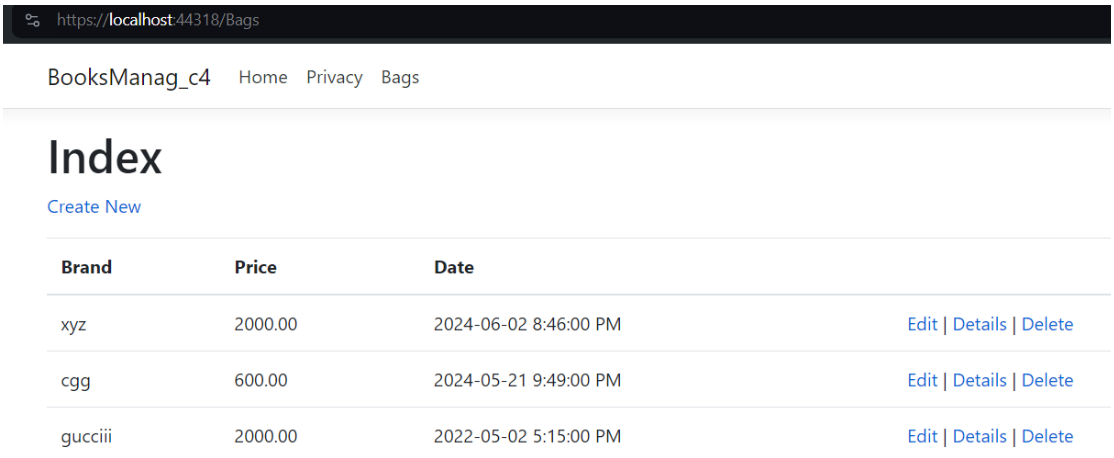
Details

Bags

Brand	gucciii
Price	2000.00
Date	2022-05-02 5:15:00 PM

Edit | Back to List

Delete



https://localhost:44318/Bags

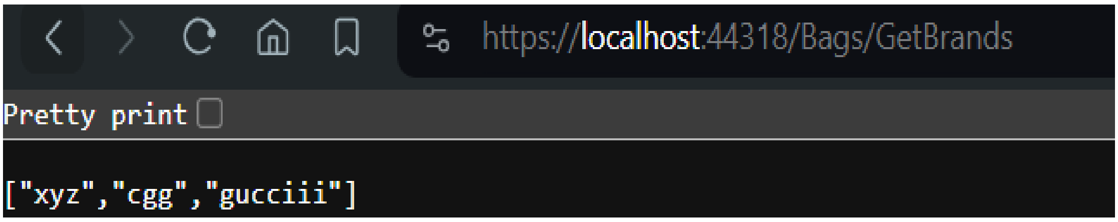
BooksManag_c4 Home Privacy Bags

Index

Create New

Brand	Price	Date	
xyz	2000.00	2024-06-02 8:46:00 PM	Edit | Details | Delete
cgg	600.00	2024-05-21 9:49:00 PM	Edit | Details | Delete
gucciii	2000.00	2022-05-02 5:15:00 PM	Edit | Details | Delete

Retrive



https://localhost:44318/Bags/GetBrands

Pretty print □

```
["xyz","cgg","gucciii"]
```

Chapter 6

Give an example to manage the session state.

HomeController.cs

```
public IActionResult Index()
        {
            // Set session values
            HttpContext.Session.SetString("UserName", "Lokesh Datta Ojha");
            HttpContext.Session.SetInt32("UserAge", 22);

            return View();
        }
```

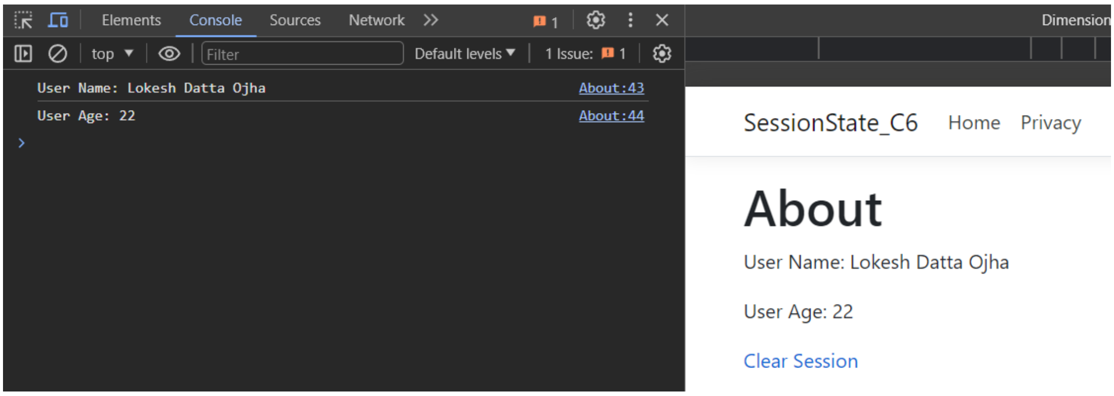
Index.cshtml

```
@{
    ViewData["Title"] = "Home Page";
}

<div>
    <h1>Welcome to the Home Page</h1>
    <p>Session values have been set.</p>
    <a href="/Home/About">Go to About page</a>
    <h1>Lokesh Datta Ojha
    </h1>
</div>
```

About.cshtml

```
@{
    ViewData["Title"] = "About Page";
}

<div>
    <h1>About</h1>
    <p>User Name: @ViewBag.UserName</p>
    <p>User Age: @ViewBag.UserAge</p>
    <a href="/Home/ClearSession">Clear Session</a>
</div>
<script>
    console.log("User Name: @ViewBag.UserName");
    console.log("User Age: @ViewBag.UserAge");
</script>
```

Output:



```
User Name: Lokesh Datta Ojha                                    About:43
User Age: 22                                                    About:44
```

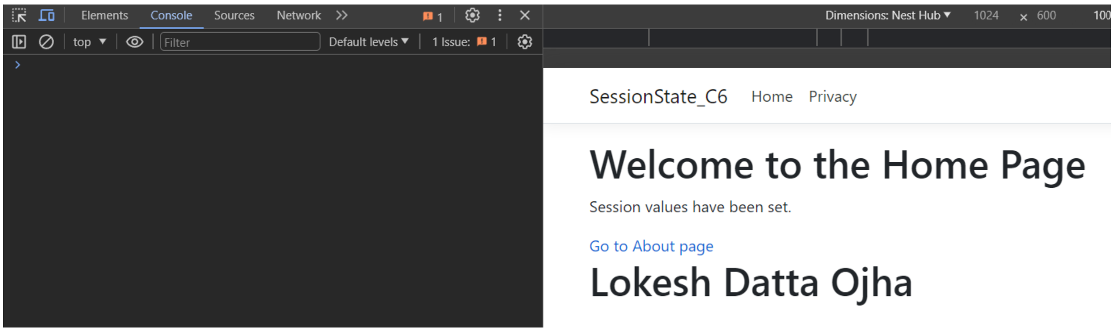
SessionState_C6 Home Privacy

About

User Name: Lokesh Datta Ojha

User Age: 22

Clear Session



SessionState_C6 Home Privacy

Welcome to the Home Page

Session values have been set.

Go to About page

Lokesh Datta Ojha

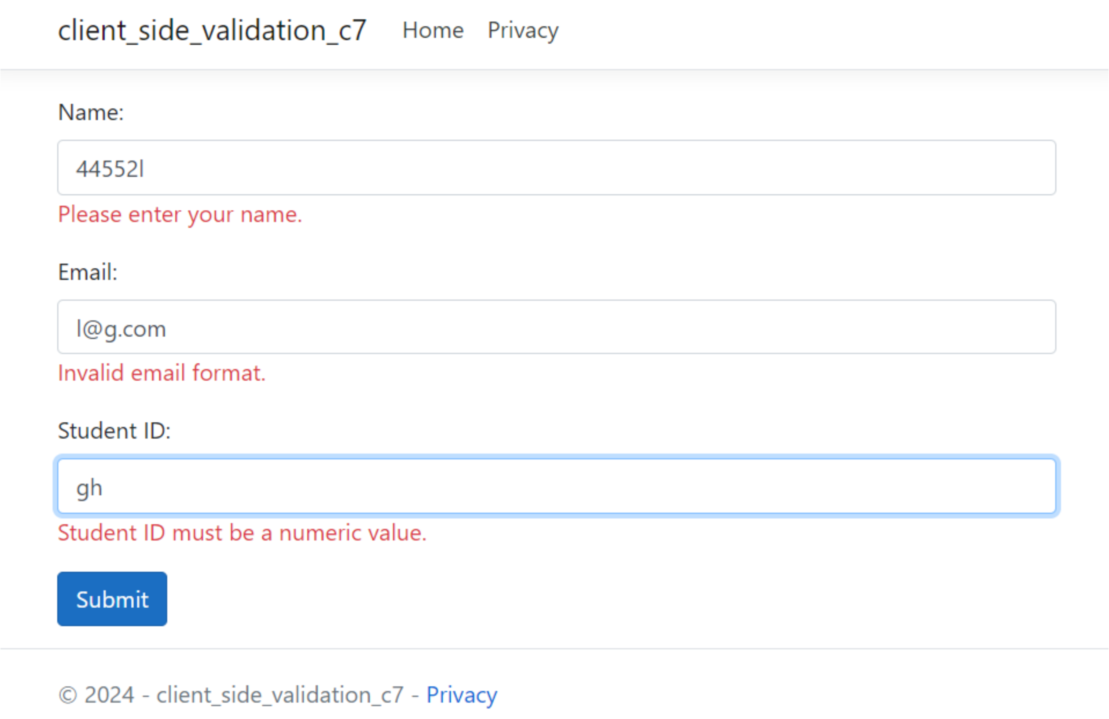
Chapter :7

Design a client-side validation application using jQuery for taking input data from students in the Library Management System, assume your own assumptions.

AddStudent.cshtml

```
@{
    ViewData["Title"] = "Add Student";
}

<form id="studentForm">
    <div class="form-group">
        <label for="name">Name:</label>
        <input type="text" class="form-control" id="name" name="name" required>
        <span class="text-danger" id="nameError"></span>
    </div>

    <div class="form-group">
        <label for="email">Email:</label>
        <input type="email" class="form-control" id="email" name="email" required>
        <span class="text-danger" id="emailError"></span>
    </div>

    <div class="form-group">
        <label for="studentId">Student ID:</label>
        <input type="text" class="form-control" id="studentId" name="studentId" required>
        <span class="text-danger" id="studentIdError"></span>
    </div>

    <button type="submit" class="btn btn-primary">Submit</button>
</form>

@section Scripts {
    <script>
        $(document).ready(function () {
            $('#studentForm').submit(function (event) {
                event.preventDefault(); // Prevent form submission

                var name = $('#name').val();
                var email = $('#email').val();
                var studentId = $('#studentId').val();
        // Validate Name
        if (name.trim() === '' || name.length < 5) {
          $('#nameError').text('Please enter your name.');
          return false; // Prevent form submission
        }

        // Validate Email
        if (email.trim() === '') {
          $('#emailError').text('Please enter your email.');
          return false; // Prevent form submission
        }
        // Validate Email Format
        if (email.length < 10) {
            $('#emailError').text('Invalid email format.');
            return false; // Prevent form submission
        }


        // Validate Student ID
        if (studentId.trim() === '') {
          $('#studentIdError').text('Please enter your student ID.');
          return false; // Prevent form submission
        }
        if (isNaN(studentId)) {
          $('#studentIdError').text('Student ID must be a numeric value.');
          return false; // Prevent form submission
        }

        // If all validations pass, submit the form
        return true;
      });
    });
  </script>
}
```

Output



Name:

44552l

Please enter your name.

Email:

l@g.com

Invalid email format.

Student ID:

gh

Student ID must be a numeric value.

© 2024 - client_side_validation_c7 - Privacy

Chapter 8:

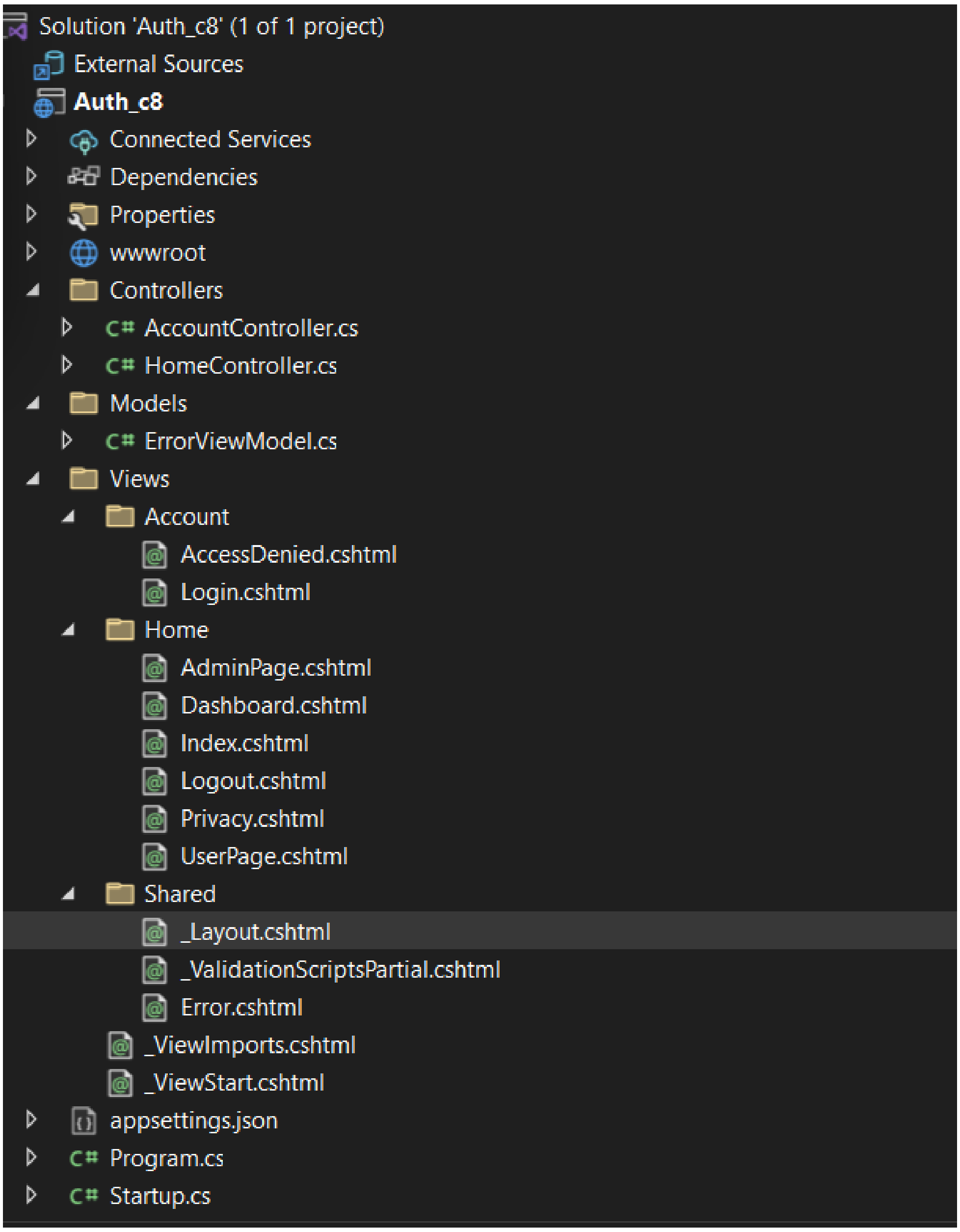
Design a web application and secure it using authentication, and authorization.

AccountController.cs

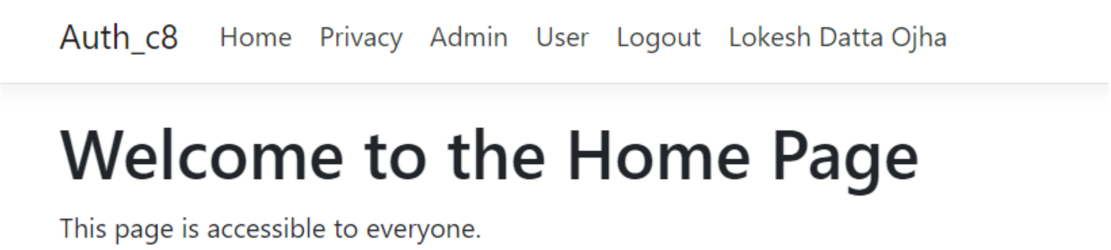
```
using Microsoft.AspNetCore.Authentication.Cookies;
using Microsoft.AspNetCore.Authentication;
using Microsoft.AspNetCore.Mvc;
using System.Collections.Generic;
using System.Security.Claims;
using System.Threading.Tasks;

namespace Auth_c8.Controllers
{
    public class AccountController : Controller
    {
        [HttpGet]
        public IActionResult Login(string returnUrl)
        {
            ViewBag.ReturnUrl = returnUrl;
            return View();
        }

        [HttpPost]
        public async Task<IActionResult> Login(string username, string password, string
returnUrl)
        {
            if (username == "admin" && password == "admin")
            {
                var claims = new List<Claim>
                {
                    new Claim(ClaimTypes.Name, username),
                    new Claim(ClaimTypes.Role, "Admin")
                };

                var claimsIdentity = new ClaimsIdentity(claims,
CookieAuthenticationDefaults.AuthenticationScheme);
                var claimsPrincipal = new ClaimsPrincipal(claimsIdentity);

                await
HttpContext.SignInAsync(CookieAuthenticationDefaults.AuthenticationScheme,
claimsPrincipal);

                return Redirect(returnUrl ?? "/");
            }
            if (username == "user" && password == "user")
            {
                var claims = new List<Claim>
                {
                    new Claim(ClaimTypes.Name, username),
                    new Claim(ClaimTypes.Role, "User")
                };

                var claimsIdentity = new ClaimsIdentity(claims,
CookieAuthenticationDefaults.AuthenticationScheme);
                var claimsPrincipal = new ClaimsPrincipal(claimsIdentity);

                await
HttpContext.SignInAsync(CookieAuthenticationDefaults.AuthenticationScheme,
claimsPrincipal);

                return Redirect(returnUrl ?? "/");
            }

            ViewBag.Error = "Invalid username or password";
            return View();
        }

        public async Task<IActionResult> Logout()
        {
            await
HttpContext.SignOutAsync(CookieAuthenticationDefaults.AuthenticationScheme);
            return RedirectToAction("Index", "Home");
        }

        public IActionResult AccessDenied()
        {
            return View();
        }
    }
    }
```

Startup.cs

```
using Microsoft.AspNetCore.Authentication.Cookies;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;

public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        services.AddControllersWithViews();


services.AddAuthentication(CookieAuthenticationDefaults.AuthenticationScheme)
            .AddCookie(options =>
            {
                options.LoginPath = "/Account/Login";
                options.AccessDeniedPath = "/Account/AccessDenied";
            });

        services.AddAuthorization(options =>
        {
            options.AddPolicy("Admin", policy => policy.RequireRole("Admin"));
            options.AddPolicy("User", policy => policy.RequireRole("User"));
            options.AddPolicy("AdminOrUser", policy => policy.RequireRole("Admin", "User"));
        });
    }

    public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
    {
        if (env.IsDevelopment())
        {
            app.UseDeveloperExceptionPage();
        }
        else
        {
            app.UseExceptionHandler("/Home/Error");
            app.UseHsts();
        }

        app.UseHttpsRedirection();
        app.UseStaticFiles();

        app.UseRouting();

        app.UseAuthentication();
        app.UseAuthorization();
        app.UseEndpoints(endpoints =>
        {
            endpoints.MapControllerRoute(
                name: "default",
                pattern: "{controller=Home}/{action=Index}/{id?}");
        });
    }
}
```

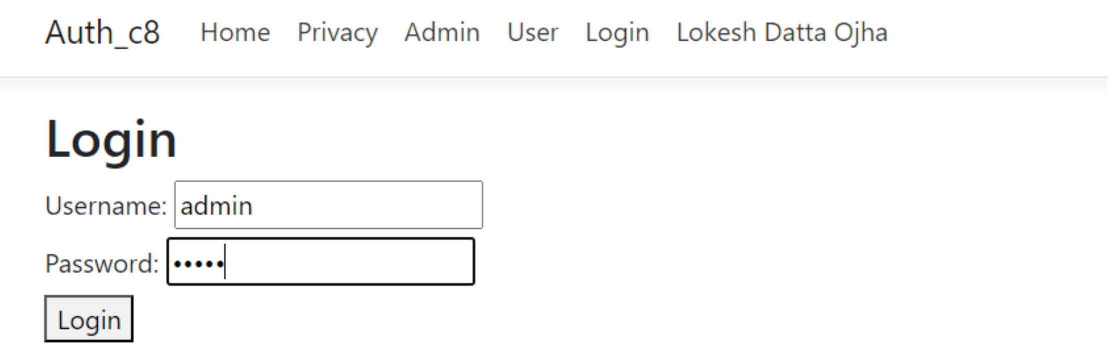


```
Solution 'Auth_c8' (1 of 1 project)
  External Sources
  Auth_c8
    Connected Services
    Dependencies
    Properties
    wwwroot
    Controllers
      AccountController.cs
      HomeController.cs
    Models
      ErrorViewModel.cs
    Views
      Account
        AccessDenied.cshtml
        Login.cshtml
      Home
        AdminPage.cshtml
        Dashboard.cshtml
        Index.cshtml
        Logout.cshtml
        Privacy.cshtml
        UserPage.cshtml
      Shared
        _Layout.cshtml
        _ValidationScriptsPartial.cshtml
        Error.cshtml
      _ViewImports.cshtml
      _ViewStart.cshtml
    appsettings.json
    Program.cs
    Startup.cs
```



Welcome to the Home Page

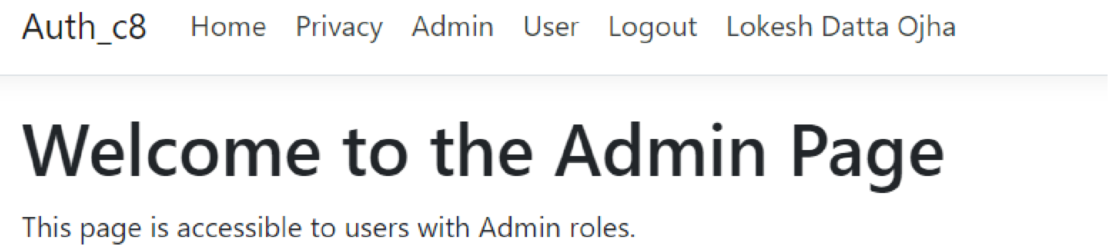
This page is accessible to everyone.



Login

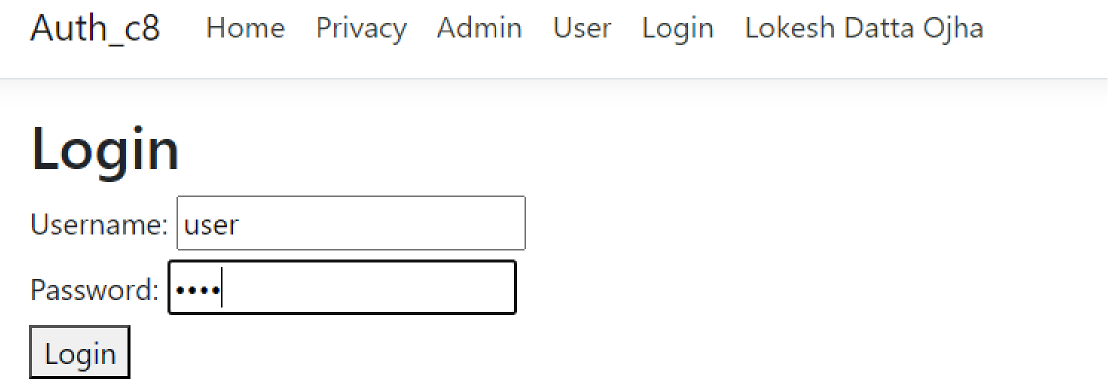
Username: admin

Password: •••••



Welcome to the Admin Page

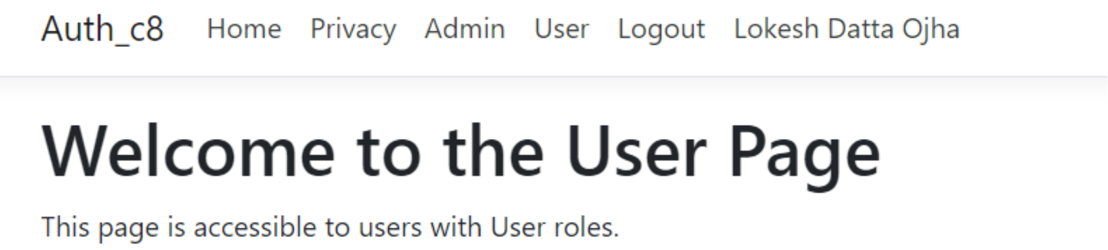
This page is accessible to users with Admin roles.



Login

Username: user

Password: ••••



Welcome to the User Page

This page is accessible to users with User roles.