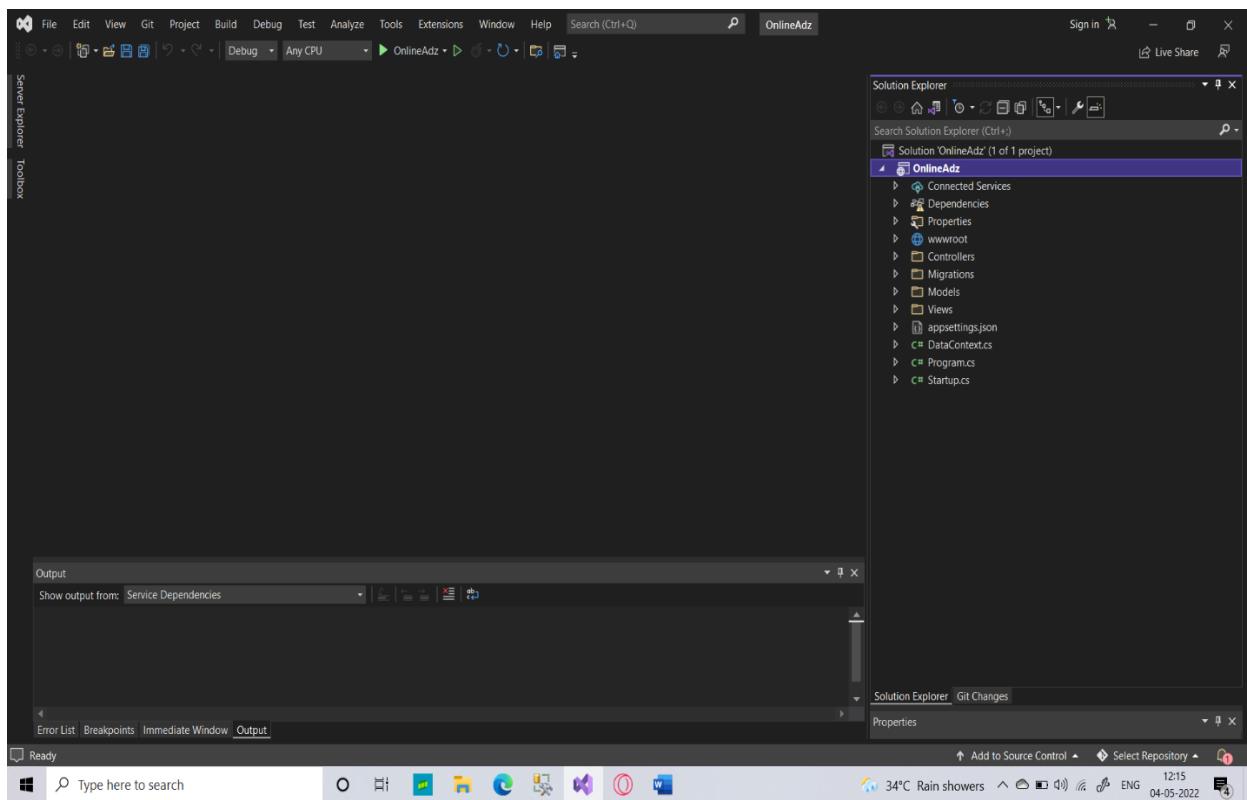


Online Ads

Team-24

Basic structure of the Project:



Controllers: AccountController

The screenshot shows the Visual Studio IDE interface with the following details:

- Title Bar:** OnlineAdz
- Toolbars:** Standard, Debug, Windows, Help.
- Code Editor:** AccountController.cs (part of the OnlineAdz project). The code implements an AccountController with Register and Login actions using ASP.NET Core Identity.
- Solution Explorer:** Shows the project structure with files like AccountController.cs, CategoryAdController.cs, HomeController.cs, PlatformController.cs, UserAdController.cs, UserInterestController.cs, Migrations, Models, Views, appsettings.json, ApplicationDbContext.cs, Program.cs, and Startup.cs.
- Status Bar:** Shows the date (04-05-2022), time (12:17), and weather (34°C Rain showers).

```
using OnlineAdz.Models;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;
using System.Linq;
using System.Threading.Tasks;

namespace OnlineAdz.Controllers
{
    [Authorize]
    public class AccountController : Controller
    {
        private readonly UserManager<IdentityUser> _userManager;
        private readonly SignInManager<IdentityUser> _signInManager;
        public AccountController(UserManager<IdentityUser> userManager, SignInManager<IdentityUser> signInManager)
        {
            _userManager = userManager;
            _signInManager = signInManager;
        }

        [HttpGet]
        public IActionResult Register()
        {
            return View();
        }

        [HttpPost]
        public async Task<IActionResult> Register(RegisterViewModel model)
        {
            if (ModelState.IsValid)
            {
                var user = new IdentityUser
                {
                    UserName = model.Name,
                    Email = model.Email
                };
                var result = await _userManager.CreateAsync(user, model.Password);
                if (result.Succeeded)
                {
                    //add role here
                    //await _userManager.AddToRoleAsync(user, "Admin");
                    return RedirectToAction("Login", "Account");
                }
            }
            ModelState.AddModelError("", "Invalid Register.");
            return View(model);
        }

        [HttpGet]
        public IActionResult Login()
        {
            return View();
        }
    }
}
```

The screenshot shows the Visual Studio IDE interface with the following details:

- Title Bar:** OnlineAdz
- Toolbars:** Standard, Debug, Windows, Help.
- Code Editor:** AccountController.cs (part of the OnlineAdz project). The code has been modified to include a redirect to the Admin role after successful registration.
- Solution Explorer:** Shows the project structure with files like AccountController.cs, CategoryAdController.cs, HomeController.cs, PlatformController.cs, UserAdController.cs, UserInterestController.cs, Migrations, Models, Views, appsettings.json, ApplicationDbContext.cs, Program.cs, and Startup.cs.
- Status Bar:** Shows the date (04-05-2022), time (12:18), and weather (34°C Rain showers).

```
using OnlineAdz.Models;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Mvc;
using System.Linq;
using System.Threading.Tasks;

namespace OnlineAdz.Controllers
{
    [Authorize]
    public class AccountController : Controller
    {
        private readonly UserManager<IdentityUser> _userManager;
        private readonly SignInManager<IdentityUser> _signInManager;
        public AccountController(UserManager<IdentityUser> userManager, SignInManager<IdentityUser> signInManager)
        {
            _userManager = userManager;
            _signInManager = signInManager;
        }

        [HttpGet]
        public IActionResult Register(RegisterViewModel model)
        {
            if (ModelState.IsValid)
            {
                var user = new IdentityUser
                {
                    UserName = model.Name,
                    Email = model.Email
                };
                var result = await _userManager.CreateAsync(user, model.Password);
                if (result.Succeeded)
                {
                    //add role here
                    //await _userManager.AddToRoleAsync(user, "Admin");
                    return RedirectToAction("Login", "Account");
                }
            }
            ModelState.AddModelError("", "Invalid Register.");
            return View(model);
        }

        [HttpGet]
        public IActionResult Login()
        {
            return View();
        }
    }
}
```

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the `AccountController.cs` file from the `OnlineAdz` project. The code implements a `Logout` action method that returns a redirect to the `Login` action of the `Account` controller. The `ModelState` is checked for errors related to email and password. The Solution Explorer on the right shows the project structure with files like `Startup.cs`, `Program.cs`, and various controller classes.

```
        public IActionResult Logout()
        {
            return RedirectToAction("Login", "Account");
        }
    }

    0 references
    public IActionResult Logout()
    {
        return RedirectToAction("Login", "Account");
    }
}
```

Controllers:CategoryAdController:

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the `CategoryAdController.cs` file from the `OnlineAdz` project. The code defines a `CategoryAdController` class that inherits from `Controller`. It contains an `Index` action method that returns a view of all `CategoryAds` from the database. It also contains a `Create` action method with `[HttpPost]` and `[ValidateAntiForgeryToken]` attributes, which adds a new `CategoryAd` to the database and redirects back to the index. The Solution Explorer on the right shows the project structure with files like `Startup.cs`, `Program.cs`, and various controller classes.

```
        public IActionResult Index()
        {
            return View(context.CategoryAds.ToList());
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public IActionResult Create(CategoryAd cad)
        {
            if (cad != null)
            {
                context.CategoryAds.Add(cad);
                context.SaveChanges();
                return RedirectToAction("Index");
            }
            return View();
        }
    }

    0 references
    public IActionResult Index()
    {
        return View(context.CategoryAds.ToList());
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    public IActionResult Create(CategoryAd cad)
    {
        if (cad != null)
        {
            context.CategoryAds.Add(cad);
            context.SaveChanges();
            return RedirectToAction("Index");
        }
        return View();
    }
}
```

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the `CategoryAdController.cs` file under the `OnlineAdz` project. The code implements the `IActionResult` interface with methods for `Edit`, `Delete`, and `Details`. It uses Entity Framework to interact with the `CategoryAds` database context. The Solution Explorer on the right shows the project structure with files like `AccountController.cs`, `CategoryAdController.cs`, `HomeController.cs`, `PlatformController.cs`, `UserAdController.cs`, and `UserInterestController.cs`. The status bar at the bottom indicates the user is AWL, has 5.00% completion, and the date is 04-05-2022.

```
    public IActionResult Edit(string id)
    {
        return View(context.CategoryAds.Find(id));
    }

    [HttpPost]
    public IActionResult Edit(string id, CategoryAd cad)
    {
        if (cad != null)
        {
            CategoryAd c = context.CategoryAds.Find(id);
            c.CategoryName = cad.CategoryName;
            context.SaveChanges();
            return RedirectToAction("Index");
        }
        return View();
    }

    public IActionResult Delete(string id)
    {
        return View(context.CategoryAds.Find(id));
    }

    [HttpPost]
    public IActionResult Delete(string id, CategoryAd cad)
    {
        if (cad != null)
        {
            CategoryAd c = context.CategoryAds.Find(id);
            context.CategoryAds.Remove(c);
            context.SaveChanges();
            return RedirectToAction("Index");
        }
        return View();
    }

    public IActionResult Details(string id)
    {
    }
```

This screenshot is identical to the one above, showing the `CategoryAdController.cs` file in the Microsoft Visual Studio IDE. The code implements the `IActionResult` interface with methods for `Edit`, `Delete`, and `Details`. It uses Entity Framework to interact with the `CategoryAds` database context. The Solution Explorer on the right shows the project structure with files like `AccountController.cs`, `CategoryAdController.cs`, `HomeController.cs`, `PlatformController.cs`, `UserAdController.cs`, and `UserInterestController.cs`. The status bar at the bottom indicates the user is AWL, has 5.00% completion, and the date is 04-05-2022.

```
    public IActionResult Edit(string id)
    {
        if (cad != null)
        {
            CategoryAd c = context.CategoryAds.Find(id);
            c.CategoryName = cad.CategoryName;
            context.SaveChanges();
            return RedirectToAction("Index");
        }
        return View();
    }

    public IActionResult Delete(string id)
    {
        return View(context.CategoryAds.Find(id));
    }

    [HttpPost]
    public IActionResult Delete(string id, CategoryAd cad)
    {
        if (cad != null)
        {
            CategoryAd c = context.CategoryAds.Find(id);
            context.CategoryAds.Remove(c);
            context.SaveChanges();
            return RedirectToAction("Index");
        }
        return View();
    }

    public IActionResult Details(string id)
    {
    }
```

The screenshot shows the Visual Studio IDE interface with the following details:

- Title Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Search (Ctrl+Q), OnlineAdz.
- Solution Explorer:** Shows the project structure for "OnlineAdz" with files like AccountController.cs, CategoryAdController.cs, HomeController.cs, PlatformController.cs, UserAdController.cs, UserInterestController.cs, Migrations, Models, Views, appsettings.json, ApplicationDbContext.cs, Program.cs, and Startup.cs.
- Code Editor:** Displays the "CategoryAdController.cs" file with C# code for managing category ads. It includes methods for Delete (HTTP GET and POST), Details, and Index.
- Status Bar:** Shows the status bar with "AWL -5.00%" and the date "04-05-2022".

Controllers:HomeController:

The screenshot shows the Visual Studio IDE interface with the following details:

- Title Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Search (Ctrl+Q), OnlineAdz.
- Solution Explorer:** Shows the project structure for "OnlineAdz" with files like AccountController.cs, CategoryAdController.cs, HomeController.cs, PlatformController.cs, UserAdController.cs, UserInterestController.cs, Migrations, Models, Views, appsettings.json, ApplicationDbContext.cs, Program.cs, and Startup.cs.
- Code Editor:** Displays the "HomeController.cs" file with C# code for the Home controller. It includes actions for Index, Privacy, and AboutUs.
- Status Bar:** Shows the status bar with "28°C Light rain" and the date "04-05-2022".

The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q).
- Solution Explorer:** Shows the project structure for "OnlineAdz".
- Code Editor:** Displays the `HomeController.cs` file with the following code:

```
using Microsoft.AspNetCore.Mvc;
using OnlineAdz.Models;

namespace OnlineAdz.Controllers
{
    public class HomeController : Controller
    {
        public IActionResult Privacy()
        {
            return View();
        }

        public IActionResult AboutUs()
        {
            return View();
        }

        public IActionResult ContactUs()
        {
            return View();
        }

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

The code editor includes a status bar at the bottom with "90 %", "No issues found", "Ln: 1 Ch: 1 SPC CRLF", and a search bar with "Type here to search".

Controllers:PlatformController:

The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q).
- Solution Explorer:** Shows the project structure for "OnlineAdz".
- Code Editor:** Displays the `PlatformController.cs` file with the following code:

```
using Microsoft.AspNetCore.Mvc;
using OnlineAdz.Models;

namespace OnlineAdz.Controllers
{
    public class PlatformController : Controller
    {
        private popularityContext context = new popularityContext();

        public IActionResult Index()
        {
            return View(context.Platforms.ToList());
        }
    }
}
```

The code editor includes a status bar at the bottom with "90 %", "No issues found", "Ln: 1 Ch: 1 SPC CRLF", and a search bar with "Type here to search".

Controllers:UserAdController:

The screenshot shows the Visual Studio IDE interface with the UserAdController.cs file open in the main editor. The code implements a controller for managing user ads. It includes methods for indexing ads by email or ad name, creating new ads, and listing ads. The Solution Explorer on the right shows the project structure with various controllers like AccountController, CategoryAdController, HomeController, PlatformController, and UserAdController.

```
using Microsoft.AspNetCore.Mvc;
using OnlineAdz.Models;

namespace OnlineAdz.Controllers
{
    public class UserAdController : Controller
    {
        private readonly AdApplicationContext _context = new AdApplicationContext();

        public ActionResult Index(string option, string search)
        {
            if (option == "Email")
            {
                return View(_context.UserAds.Where(x => x.Email == search).ToList());
            }
            else
            {
                return View(_context.UserAds.Where(x => x.AdName == search).ToList());
            }
        }

        [HttpPost]
        public IActionResult Create()
        {
            return View();
        }
    }
}
```

This screenshot shows the same Visual Studio interface with the UserAdController.cs file open. It adds two more methods: Edit and Edit. The Edit method takes an integer ID and returns a view of the user ad with that ID. The Edit method takes an integer ID and a UserAd object, updates the user ad in the database, and then returns a redirect to the index action. The Solution Explorer remains the same, showing the project structure.

```
        return View();
    }

    [HttpPost]
    public IActionResult Create(UserAd ad)
    {
        if (ad != null)
        {
            _context.UserAds.Add(ad);
            _context.SaveChanges();
            return RedirectToAction("Index");
        }
        return View();
    }

    public IActionResult Edit(int id)
    {
        var ad = _context.UserAds.Find(id);
        return View(ad);
    }

    [HttpPost]
    public IActionResult Edit(int id, UserAd ad)
    {
        if (ad != null)
        {
            UserAd a = _context.UserAds.Find(id);
            a.CategoryId = ad.CategoryId;
        }
    }
}
```

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the `UserAdController.cs` file under the `OnlineAdz` project. The code implements two methods: `Index` and `Delete`. The `Index` method retrieves a user ad by ID and returns a view. The `Delete` method removes a user ad from the database and redirects to the index action. The Solution Explorer on the right lists all files in the `OnlineAdz` project, including controllers like `AccountController.cs`, `CategoryAdController.cs`, `HomeController.cs`, `PlatformController.cs`, `UserAdController.cs`, and `UserInterestController.cs`.

```
if (ad != null)
{
    UserAd a = context.UserAds.Find(id);
    a.CategoryId = ad.CategoryId;
    a.Email = ad.Email;
    a.AdTitle = ad.AdTitle;
    a.AdStatus = ad.AdStatus;
    a.UserInterest = ad.UserInterest;
    context.SaveChanges();
    return RedirectToAction("Index");
}

return View();
}

[HttpPost]
public IActionResult Delete(int id)
{
    return View(context.UserAds.Find(id));
}
[HttpPost]
public IActionResult Delete(int id, UserAd ad)
{
    if (ad != null)
    {
        UserAd a = context.UserAds.Find(id);
        context.UserAds.Remove(a);
        context.SaveChanges();
        return RedirectToAction("Index");
    }
}
```

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the `UserAdController.cs` file under the `OnlineAdz` project. The code implements methods for listing, creating, updating, and deleting user ads. The `Index` method lists ads by option and search term. The `Create` and `Edit` methods handle form submissions. The `Delete` method removes an ad from the database. The `Details` method retrieves a specific ad for display. The `OnActionExecuting` filter ensures ads are only listed if they belong to the current user.

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

namespace OnlineAdz.Controllers
{
    public class UserAdController : Controller
    {
        private readonly ApplicationDbContext _context;

        public UserAdController(ApplicationDbContext context)
        {
            _context = context;
        }

        // GET: UserAdz/UserAd
        public async Task Index(string option, string search)
        {
            var ads = _context.UserAds
                .Include(u => u.User)
                .Where(u => u.User.Id == User.Id);

            if (!string.IsNullOrEmpty(option))
            {
                ads = ads.Where(u => u.Option == option);
            }

            if (!string.IsNullOrEmpty(search))
            {
                ads = ads.Where(u => u.Search == search);
            }

            return View(await ads.ToListAsync());
        }

        // GET: UserAdz/UserAd/5
        [HttpGet]
        public async Task Details(int? id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var userAd = await _context.UserAds
                .Include(u => u.User)
                .FirstOrDefaultAsync(u => u.Id == id);

            if (userAd == null)
            {
                return NotFound();
            }

            return View(userAd);
        }

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

        // POST: UserAdz/UserAd/Create
        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task Create([Bind("Id,User,Option,Search,Image,IsApproved")] UserAd userAd)
        {
            if (ModelState.IsValid)
            {
                _context.Add(userAd);
                await _context.SaveChangesAsync();
                return RedirectToAction(nameof(Index));
            }
            return View(userAd);
        }

        // GET: UserAdz/UserAd/Edit/5
        public async Task Edit(int? id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var userAd = await _context.UserAds.FindAsync(id);

            if (userAd == null)
            {
                return NotFound();
            }

            return View(userAd);
        }

        // POST: UserAdz/UserAd/Edit/5
        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task Edit([Bind("Id,User,Option,Search,Image,IsApproved")] UserAd userAd)
        {
            if (ModelState.IsValid)
            {
                _context.Update(userAd);
                await _context.SaveChangesAsync();
                return RedirectToAction(nameof(Index));
            }
            return View(userAd);
        }

        // GET: UserAdz/UserAd/Delete/5
        public async Task Delete(int id)
        {
            var userAd = await _context.UserAds.FindAsync(id);

            if (userAd == null)
            {
                return NotFound();
            }

            _context.UserAds.Remove(userAd);
            await _context.SaveChangesAsync();

            return RedirectToAction(nameof(Index));
        }

        // This filter ensures ads are only listed if they belong to the current user
        [OnActionExecuting]
        public void OnActionExecuting(ActionExecutingContext context)
        {
            var user = _context.Users
                .Include(u => u.UserAds)
                .FirstOrDefault(u => u.Id == User.Id);

            if (user != null)
            {
                context.Result = new JsonResult(new { user });
            }
        }
    }
}
```

The Solution Explorer on the right shows the project structure for `OnlineAdz`, which includes controllers like `AccountController.cs`, `CategoryAdController.cs`, `HomeController.cs`, `PlatformController.cs`, `UserAdController.cs`, and `UserInterestController.cs`.

Controllers:UserInterestController:

The screenshot shows the Visual Studio IDE interface with the UserInterestController.cs file open in the main editor. The code defines a controller for managing user interests. It includes imports for Microsoft.AspNetCore.Mvc and OnlineAdz.Models, and a namespace declaration. The controller has two actions: Index and Index. The Index action takes a UserInterest object (ui) and returns a view, setting the email, category ID, ad name, and ad description from the ui object. The Solution Explorer on the right shows the project structure with files like AccountController.cs, CategoryAdController.cs, HomeController.cs, PlatformController.cs, UserAdController.cs, and UserInterestController.cs.

```
OnlineAdz
using Microsoft.AspNetCore.Mvc;
using OnlineAdz.Models;

namespace OnlineAdz.Controllers
{
    public class UserInterestController : Controller
    {
        public IActionResult Index()
        {
            return View();
        }

        [HttpPost]
        public IActionResult Index(UserInterest ui)
        {
            string email = ui.Email;
            string categoryid = ui.CategoryId;
            string adname = ui.AdName;
            string addesc = ui.AdDesc;
            return View();
        }
    }
}
```

Models: AdApplicationContext:

The screenshot shows the Visual Studio IDE interface with the AdApplicationContext.cs file open in the main editor. This is a partial class that inherits from DbContext. It has a constructor that takes an DbContextOptions<AdApplicationContext> parameter. It also has properties for CategoryAds and UserAds, both of type DbSet<CategoryAd> and DbSet<UserAd> respectively. The OnConfiguring method overrides the base implementation to set the connection string to "AdApplication" (which is commented out). The Solution Explorer on the right shows the project structure with files like AdApplication.cs, CategoryAd.cs, ErrorViewModel.cs, LoginViewModel.cs, Platform.cs, popularityContext.cs, RegisterViewModel.cs, UserAd.cs, UserInterest.cs, appsettings.json, DataContext.cs, Program.cs, and Startup.cs.

```
OnlineAdz
using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Metadata;

namespace OnlineAdz.Models
{
    public partial class AdApplicationContext : DbContext
    {
        public AdApplicationContext(DbContextOptions<AdApplicationContext> options)
        {
        }

        public virtual DbSet<CategoryAd> CategoryAds { get; set; } = null!;
        public virtual DbSet<UserAd> UserAds { get; set; } = null!;

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            if (!optionsBuilder.IsConfigured)
            {
                #warning To protect potentially sensitive information in your connection string, you should move it out of source code. You can avoid scaffolding the connection string by configuring it outside your application.
                optionsBuilder.UseSqlServer("Server=.;Initial Catalog=AdApplication;User Id=sa;Password=12345");
            }
        }
    }
}
```

```
0 references
protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
{
    if (!optionsBuilder.IsConfigured)
    {
        #warning To protect potentially sensitive information in your connection string, you should move it out of source code. You can avoid scaffolding the connection string by using the Name= setting in App.config or Appsettings.json file
        optionsBuilder.UseSqlServer("Server=.;Initial Catalog=AdApplication;User Id=sa;Password=12345");
    }
}

0 references
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<CategoryAd>(entity =>
    {
        entity.HasKey(e => e.CategoryId)
            .HasName("PK__Category__19893A8BF1942AB5");

        entity.Property(e => e.CategoryId)
            .HasMaxLength(50)
            .IsUnicode(false);

        entity.Property(e => e.CategoryName)
            .HasMaxLength(50)
            .IsUnicode(false);
    });

    modelBuilder.Entity<UserAd>(entity =>
    {
        entity.HasKey(e => e.AdId)
            .HasName("PK__UserAds__713B05AE87D9A2E0");
    });
}
```

```
modelBuilder.Entity<UserAd>(entity =>
{
    entity.HasKey(e => e.AdId)
        .HasName("PK__UserAds__713B05AE87D9A2E0");

    entity.Property(e => e.AdName)
        .HasMaxLength(50)
        .IsUnicode(false);

    entity.Property(e => e.AdStatus)
        .HasMaxLength(50)
        .IsUnicode(false);

    entity.Property(e => e.CategoryId)
        .HasMaxLength(50)
        .IsUnicode(false);

    entity.Property(e => e.Email).HasMaxLength(256);

    entity.Property(e => e.UserInterest)
        .HasMaxLength(50)
        .IsUnicode(false);

    //entity.HasOne(d => d.Category)
    //    .WithMany(p => p.UserAds)
    //    .HasForeignKey(d => d.CategoryId)
    //    .OnDelete(DeleteBehavior.ClientSetNull)
    //    .HasConstraintName("FK__UserAds__Category__1273C1CD");
});

OnModelCreatingPartial(modelBuilder);
}
```

Models:CategoryAd:

The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q) - OnlineAdz
- Solution Explorer:** Shows the project structure for "OnlineAdz" with files like AdApplicationContext.cs, CategoryAd.cs, ErrorViewModel.cs, LoginViewModel.cs, Platform.cs, popularityContext.cs, RegisterViewModel.cs, UserAd.cs, UserInterest.cs, Views, appsettings.json, ApplicationDbContext.cs, Program.cs, and Startup.cs.
- Code Editor:** Displays the content of CategoryAd.cs, which defines a partial class CategoryAd with properties CategoryId, UserAds, and CategoryName.
- Status Bar:** Shows "Ready", "Type here to search", and system icons.

Models:ErrorViewModel:

The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q) - OnlineAdz
- Solution Explorer:** Shows the project structure for "OnlineAdz" with files like AdApplicationContext.cs, CategoryAd.cs, ErrorViewModel.cs, LoginViewModel.cs, Platform.cs, popularityContext.cs, RegisterViewModel.cs, UserAd.cs, UserInterest.cs, Views, appsettings.json, ApplicationDbContext.cs, Program.cs, and Startup.cs.
- Code Editor:** Displays the content of ErrorViewModel.cs, which defines a public class ErrorViewModel with a RequestId property and a validation rule for ShowRequestId.
- Status Bar:** Shows "Ready", "Type here to search", and system icons.

Models:LoginViewModel:

```
using System.ComponentModel.DataAnnotations;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations.Schema;

namespace OnlineAdz.Models
{
    public class LoginViewModel
    {
        [Required]
        [EmailAddress]
        [DataType(DataType.Password)]
        public string Email { get; set; }

        [Required]
        [DataType(DataType.Password)]
        public string Password { get; set; }

        public string Role { get; set; }
    }
}
```

Models:Platform:

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace OnlineAdz.Models
{
    public partial class Platform
    {
        [Required]
        public string CategoryId { get; set; } = null!;

        [Required]
        public string CategoryName { get; set; } = null!;

        [Required]
        public string AdDescription { get; set; } = null!;

        [Required]
        public string MainPlatform { get; set; } = null!;
    }
}
```

Models:popularityContext:

```
using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore.Metadata;

namespace OnlineAdz.Models
{
    public partial class popularityContext : DbContext
    {
        public popularityContext()
        {
        }

        public popularityContext(DbContextOptions<popularityContext> options)
        {
        }

        public virtual DbSet<Platform> Platforms { get; set; } = null!;

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            if (!optionsBuilder.IsConfigured)
            {
                #warning To protect potentially sensitive information in your connection string, you should move it out of source code. You can avoid scaffolding the connection string by using the Name= setting in App.config or Web.config, as described in https://go.microsoft.com/fwlink/?linkid=852773.
                optionsBuilder.UseSqlServer("Server=.\\Initial Catalog=popularity;User Id=sa;Password=12345");
            }
        }
    }
}
```

```
#warning To protect potentially sensitive information in your connection string, you should move it out of source code. You can avoid scaffolding the connection string by using the Name= setting in App.config or Web.config, as described in https://go.microsoft.com/fwlink/?linkid=852773.

protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<Platform>(entity =>
    {
        entity.HasKey();

        entity.Property(e => e.AdDescription)
            .HasMaxLength(90)
            .IsUnicode(false);

        entity.Property(e => e.CategoryId)
            .HasMaxLength(50)
            .IsUnicode(false);

        entity.Property(e => e.CategoryName)
            .HasMaxLength(50)
            .IsUnicode(false);

        entity.Property(e => e.MainPlatform)
            .HasMaxLength(50)
            .IsUnicode(false);
    });
}

OnModelCreatingPartial(modelBuilder);
}
```

Models: RegisterViewModel:

The screenshot shows the Visual Studio IDE interface with the RegisterViewModel.cs file open in the main editor. The code defines a class RegisterViewModel with properties Name, Email, Password, Role, State, and City, each annotated with validation attributes like Required and EmailAddress. The Solution Explorer on the right shows the project structure, including files like AdApplicationContext.cs, CategoryAd.cs, ErrorViewModel.cs, LoginViewModel.cs, Platform.cs, popularityContext.cs, RegisterViewModel.cs, UserAd.cs, UserInterest.cs, and Startup.cs.

```
using System.ComponentModel.DataAnnotations;
```

```
namespace OnlineAdz.Models
```

```
{
```

```
    public class RegisterViewModel
```

```
    {
```

```
        [Required(ErrorMessage = "Name Required")]
        public string Name { get; set; }
```

```

        [Required(ErrorMessage = "Email is required")]
        [EmailAddress]
        public string Email { get; set; }
```

```

        [Required(ErrorMessage = "Password is required")]
        [DataType(DataType.Password)]
        public string Password { get; set; }
```

```

        [Required(ErrorMessage = "Please Specify your Role : Admin or User")]
        public string Role { get; set; }
```

```

        public string State { get; set; }
```

```

        public string City { get; set; }
```

```
    }
```

Models:UserAd:

The screenshot shows the Visual Studio IDE interface with the UserAd.cs file open in the main editor. The code defines a partial class UserAd with properties AdId, CategoryId, Email, AdName, AdStatus, and UserInterest, each annotated with validation attributes like Required and EmailAddress. The Solution Explorer on the right shows the project structure, including files like AdApplicationContext.cs, CategoryAd.cs, ErrorViewModel.cs, LoginViewModel.cs, Platform.cs, popularityContext.cs, RegisterViewModel.cs, UserAd.cs, UserInterest.cs, and Startup.cs.

```
using System;
using System.Collections.Generic;
```

```
namespace OnlineAdz.Models
```

```
{
```

```
    public partial class UserAd
```

```
    {
```

```
        [Required]
        public int AdId { get; set; }
```

```

        public string CategoryId { get; set; } = null;
```

```

        [Required]
        public string? Email { get; set; }
```

```

        public string AdName { get; set; } = null;
```

```

        public string AdStatus { get; set; } = null;
```

```

        public string UserInterest { get; set; } = null;
```

```

        //public virtual CategoryId Category { get; set; } = null;
```

```
    }
```

Models:UserInterest:

```

namespace OnlineAdz.Models
{
    public class UserInterest
    {
        2 references
        public string Email { get; set; }
        2 references
        public string CategoryId { get; set; }
        2 references
        public string Adname { get; set; }
        2 references
        public string AdDesc { get; set; }
    }
}

```

Views: Account:

Login:

```

<body style="background-image: url('../Images/fiverrsize.jpg'); ">
<model OnlineAdz.Models.LoginViewModel>
<h1>Login

```

Register:

```
<body style="background-image: url('~/css/Images/Fiverrsize.jpg');">
@model OnlineAdz.Models.RegisterViewModel
<h1>Register</h1>
<hr/>


<div class="col-md-4">
<form asp-action="Register" asp-controller="Account">
    <div asp-validation-summary="All" class="text-danger"></div>
    <div class="form-group">
        <label asp-for="Name" class="control-label"></label>
        <input asp-for="Name" class="form-control" placeholder="Name"/>
        <span asp-validation-for="Name" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Email" class="control-label"></label>
        <input asp-for="Email" class="form-control" placeholder="Email"/>
        <span asp-validation-for="Email" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Password" class="control-label"></label>
        <input asp-for="Password" class="form-control" placeholder="Password" />
        <span asp-validation-for="Password" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Role" class="control-label"></label>
        <input asp-for="Role" class="form-control" placeholder="Role"/>
        <span asp-validation-for="Role" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="State" class="control-label"></label>
        <input asp-for="State" class="form-control" placeholder="State"/>
        <span asp-validation-for="State" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="City" class="control-label"></label>
        <input asp-for="City" class="form-control" placeholder="City"/>
        <span asp-validation-for="City" class="text-danger"></span>
    </div>
    <br />
    <div class="form-group">
        <input type="submit" value="Register" class="btn btn-primary" />
    </div>
</form>
</div>


```

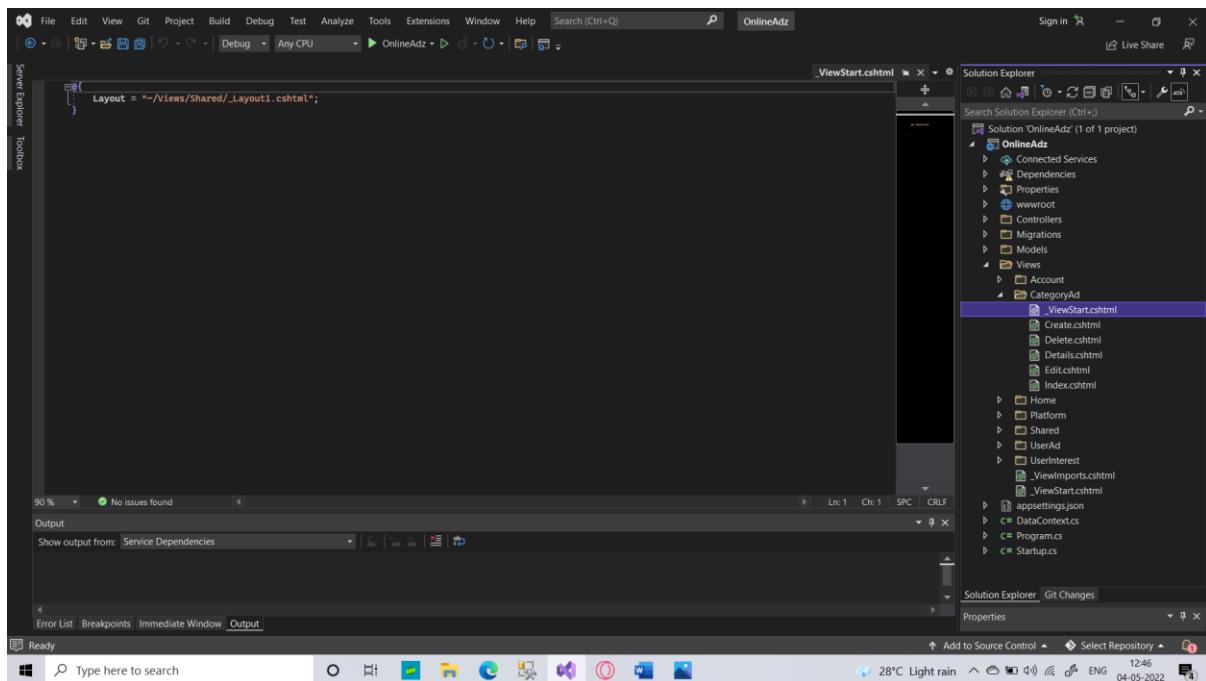
```
<body style="background-image: url('~/css/Images/Fiverrsize.jpg');">
@model OnlineAdz.Models.RegisterViewModel
<h1>Register</h1>
<hr/>


<div class="col-md-4">
<form asp-action="Register" asp-controller="Account">
    <div asp-validation-summary="All" class="text-danger"></div>
    <div class="form-group">
        <label asp-for="Name" class="control-label"></label>
        <input asp-for="Name" class="form-control" placeholder="Name"/>
        <span asp-validation-for="Name" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Email" class="control-label"></label>
        <input asp-for="Email" class="form-control" placeholder="Email"/>
        <span asp-validation-for="Email" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Password" class="control-label"></label>
        <input asp-for="Password" class="form-control" placeholder="Password" />
        <span asp-validation-for="Password" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Role" class="control-label"></label>
        <input asp-for="Role" class="form-control" placeholder="Role"/>
        <span asp-validation-for="Role" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="State" class="control-label"></label>
        <input asp-for="State" class="form-control" placeholder="State"/>
        <span asp-validation-for="State" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="City" class="control-label"></label>
        <input asp-for="City" class="form-control" placeholder="City"/>
        <span asp-validation-for="City" class="text-danger"></span>
    </div>
    <br />
    <div class="form-group">
        <input type="submit" value="Register" class="btn btn-primary" />
    </div>
</form>
</div>

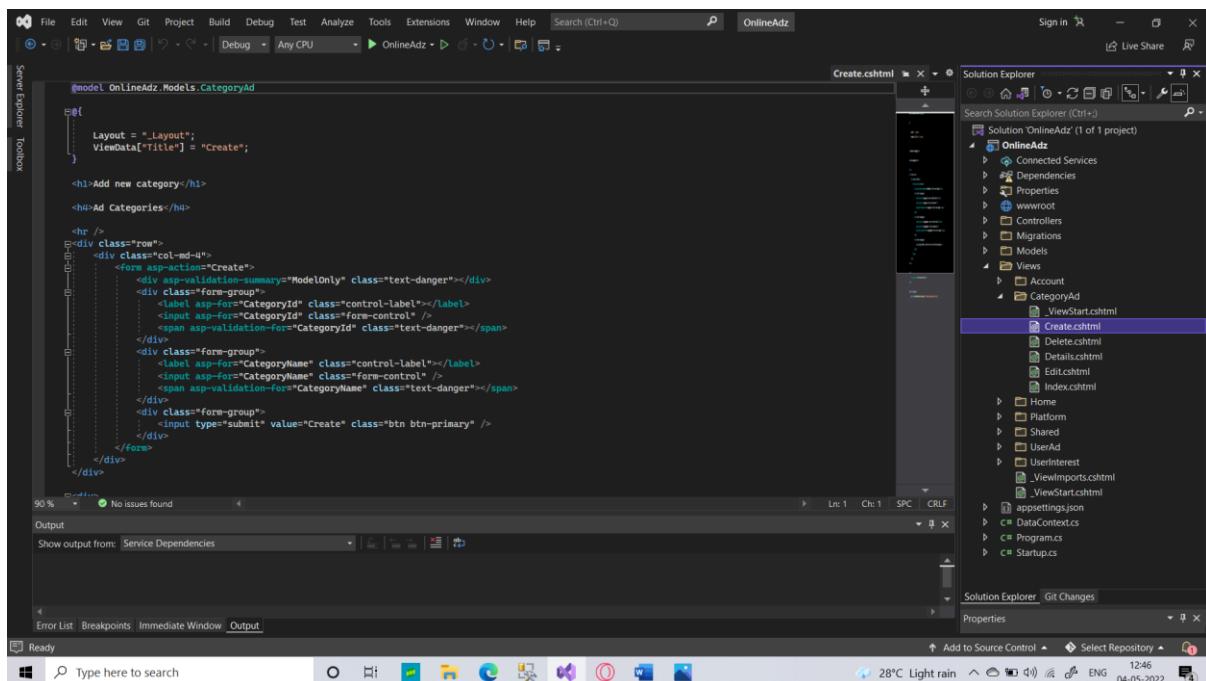

```

Views: CategoryAd:

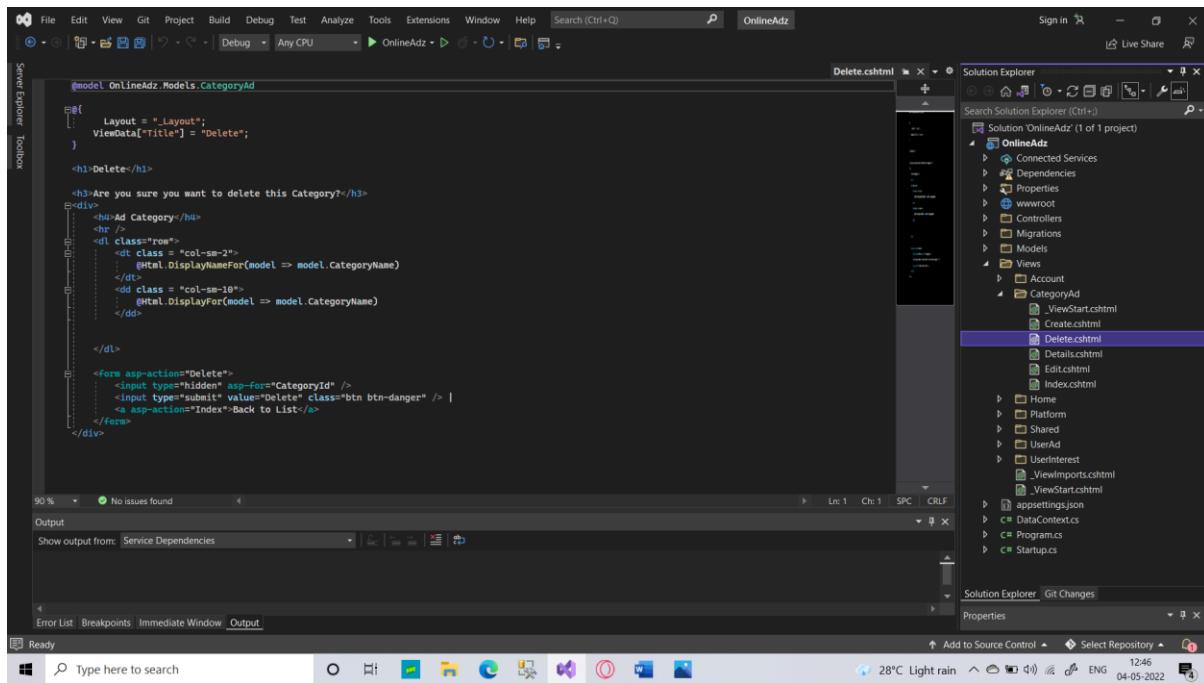
_ViewStart:



Create:



Delete:



The screenshot shows the Visual Studio IDE interface with the 'Delete.cshtml' file open in the main editor. The code is a standard ASP.NET MVC delete view for a 'CategoryAd' model. It includes an H1 header, a confirmation message, a table for displaying category details, and a form with a hidden field for the category ID, a submit button labeled 'Delete', and a link to return to the list.

```
@model OnlineAdz.Models.CategoryAd
@{
    Layout = "_Layout";
    ViewData["Title"] = "Delete";
}



# Delete



Are you sure you want to delete this Category?



#### Category



---



@Html.DisplayNameFor(model => model.CategoryName)


@Html.DisplayFor(model => model.CategoryName)

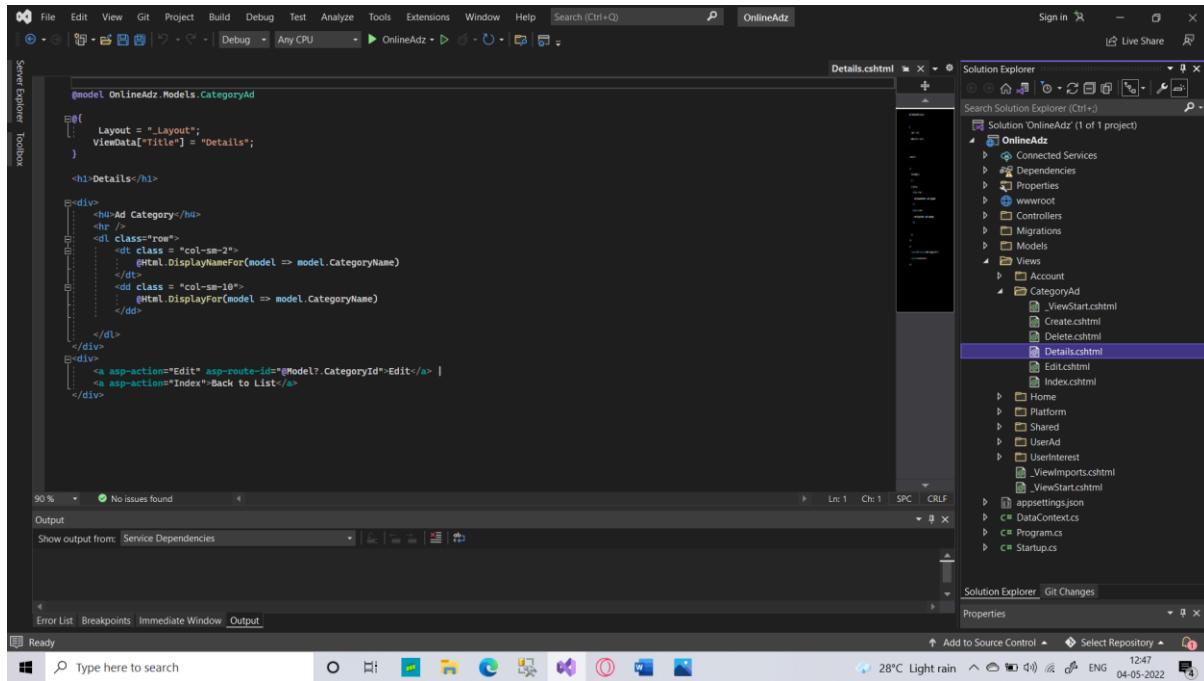


---


<form asp-action="Delete">
    <input type="hidden" asp-for="CategoryId" />
    <input type="submit" value="Delete" class="btn btn-danger" />
    <a asp-action="Index" href="#">Back to List


```

Details:



The screenshot shows the Visual Studio IDE interface with the 'Details.cshtml' file open in the main editor. The code is a standard ASP.NET MVC details view for a 'CategoryAd' model. It includes an H1 header, a confirmation message, a table for displaying category details, and a form with links to edit or back to the list.

```
@model OnlineAdz.Models.CategoryAd
@{
    Layout = "_Layout";
    ViewData["Title"] = "Details";
}



# Details



Category



---



@Html.DisplayNameFor(model => model.CategoryName)


@Html.DisplayFor(model => model.CategoryName)

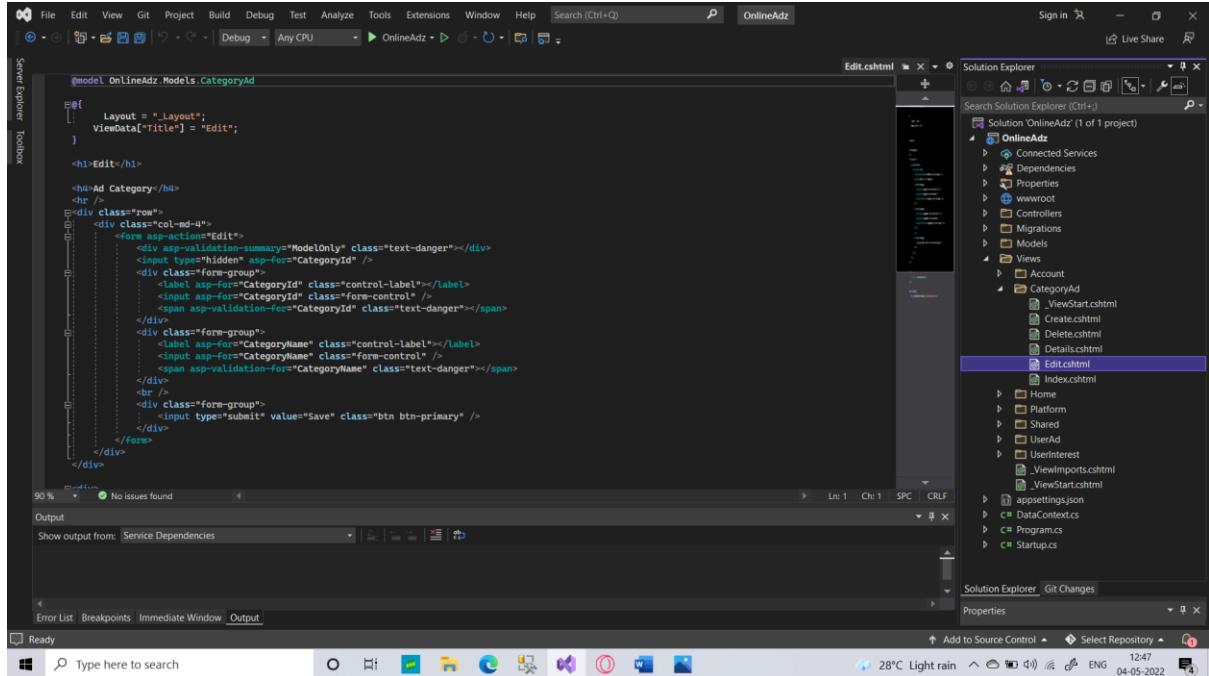


---


<div>
    <a asp-action="Edit" asp-route-id="@Model?.CategoryId" href="#">Edit
    <a asp-action="Index" href="#">Back to List
</div>

```

Edit:



The screenshot shows the Visual Studio IDE interface with the 'Edit.cshtml' file open in the main editor window. The code is an ASP.NET MVC view for editing a category. It includes a form with various input fields like 'CategoryId' and 'CategoryName', and a submit button. The Solution Explorer on the right shows the project structure, including the 'CategoryAd' folder and its files.

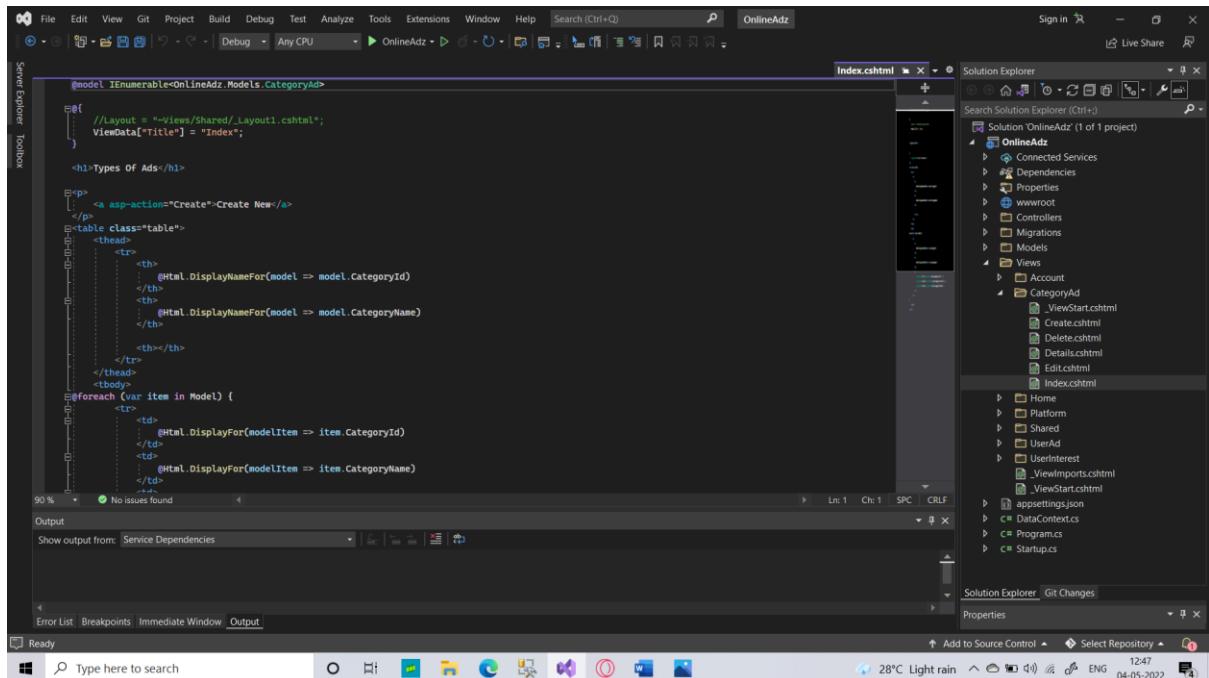
```
@@ -1 @model OnlineAdz.Models.CategoryAd
{
    Layout = "_Layout";
    ViewData["Title"] = "Edit";
}

<h1>Edit</h1>
<h2>Add Category</h2>
<hr />


<div class="col-md-4">
        <form asp-action="Edit">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <input type="hidden" asp-for="CategoryId" />
            <div class="form-group">
                <label asp-for="CategoryId" class="control-label"></label>
                <input asp-for="CategoryId" class="form-control" />
                <span asp-validation-for="CategoryId" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="CategoryName" class="control-label"></label>
                <input asp-for="CategoryName" class="form-control" />
                <span asp-validation-for="CategoryName" class="text-danger"></span>
            </div>
            <br />
            <div class="form-group">
                <input type="submit" value="Save" class="btn btn-primary" />
            </div>
        </form>
    </div>


```

Index:



The screenshot shows the Visual Studio IDE interface with the 'Index.cshtml' file open in the main editor window. The code is an ASP.NET MVC view for displaying a list of categories. It features a table with columns for Category ID and Category Name, and a link to create a new category. The Solution Explorer on the right shows the project structure, including the 'CategoryAd' folder and its files.

```
@@ -1 @model IEnumerable<OnlineAdz.Models.CategoryAd>
{
    //Layout = "_Views/Shared/_Layout.cshtml";
    ViewData["Title"] = "Index";
}

<h1>Types Of Ads</h1>
<p><a href="#" asp-action="Create">Create New</a></p>
<table class="table">
    <thead>
        <tr>
            <th>@Html.DisplayNameFor(model => model.CategoryId)</th>
            <th>@Html.DisplayNameFor(model => model.CategoryName)</th>
            <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model) {
            <tr>
                <td>@Html.DisplayFor(modelItem => item.CategoryId)</td>
                <td>@Html.DisplayFor(modelItem => item.CategoryName)</td>
                <td></td>
            </tr>
        }
    </tbody>
</table>
```

The screenshot shows the Visual Studio IDE interface with the following details:

- Title Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Search (Ctrl+Q), OnlineAdz.
- Solution Explorer:** Shows the project structure for "OnlineAdz" with 1 item: "CategoryAd".
- Code Editor:** Displays the "Index.cshtml" view code for the "CategoryAd" controller. The code includes a table with columns for Category ID and Name, and links for Edit, Details, and Delete actions.
- Status Bar:** Shows "Ready", "Type here to search", and system status like "28°C Light rain" and "12:47 04-05-2022".

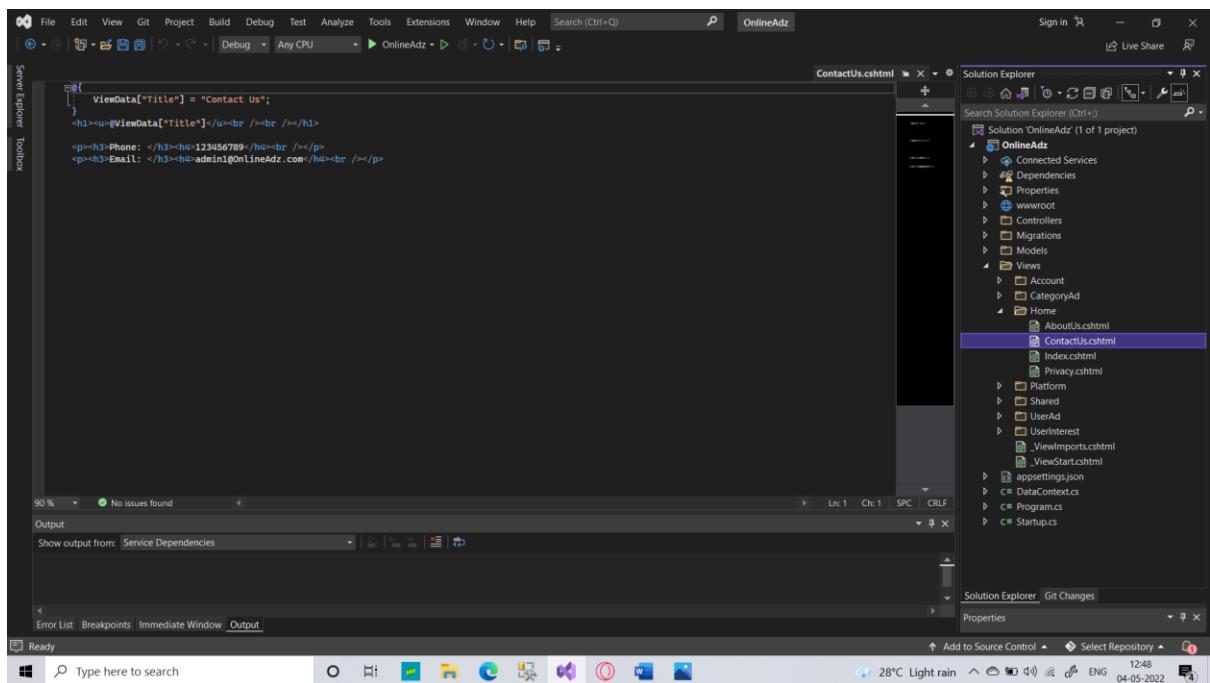
Views: Home:

AboutUs:

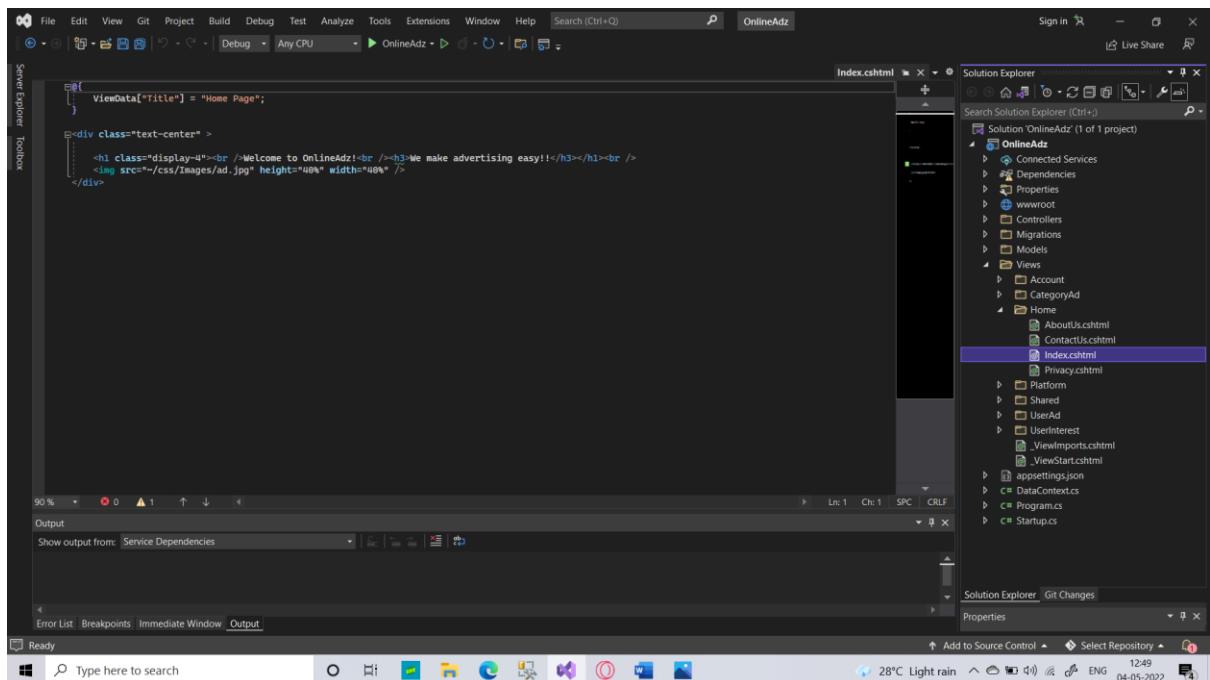
The screenshot shows the Visual Studio IDE interface with the following details:

- Title Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Search (Ctrl+Q), OnlineAdz.
- Solution Explorer:** Shows the project structure for "OnlineAdz" with 1 item: "Home".
- Code Editor:** Displays the "AboutUs.cshtml" view code for the "Home" controller. It sets the ViewData["Title"] to "About Us" and contains a paragraph about the portal.
- Status Bar:** Shows "Ready", "Type here to search", and system status like "28°C Light rain" and "12:48 04-05-2022".

ContactUs:



Index:



Privacy:

The screenshot shows the Visual Studio IDE interface. The main window displays the `Privacy.cshtml` file content:

```
[@{ ViewData["Title"] = "Privacy Policy"; }<h1>@ViewData["Title"]</h1><br /><p>We are committed to protecting and preserving the privacy of our visitors when visiting our site or communicating electronically with us.</p>
```

The Solution Explorer on the right shows the project structure for `OnlineAdz`, including files like `Index.cshtml` and `Privacy.cshtml` under the `Views\Home` folder.

Views: Platform:

Index:

The screenshot shows the Visual Studio IDE interface. The main window displays the `Index.cshtml` file content:

```
@model IEnumerable<OnlineAdz.Models.Platform>
{@
    //Layout = "~/Views/Shared/_Layout.cshtml";
    ViewData["Title"] = "Index";
}

<h1><u>Know About Ads</u></h1><br /><br />

<table class="table">
    <thead>
        <tr>
            <th>@Html.DisplayNameFor(model => model.CategoryId)</th>
            <th>@Html.DisplayNameFor(model => model.CategoryName)</th>
            <th>@Html.DisplayNameFor(model => model.AdDescription)</th>
            <th>@Html.DisplayNameFor(model => model.MainPlatform)</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model) {
            <tr>
                <td>@Html.DisplayFor(modelItem => item.CategoryId)</td>
                <td>@Html.DisplayFor(modelItem => item.CategoryName)</td>
                <td>@Html.DisplayFor(modelItem => item.AdDescription)</td>
                <td>@Html.DisplayFor(modelItem => item.MainPlatform)</td>
            </tr>
        }
    </tbody>
</table>
```

The Solution Explorer on the right shows the project structure for `OnlineAdz`, including files like `Index.cshtml` and `_ViewStart.cshtml` under the `Views\Platform` folder.

The screenshot shows the Visual Studio IDE interface. On the left, the Server Explorer and Toolbox are visible. The main area displays the `Index.cshtml` file content:

```

<table>
    <thead>
        <tr>
            <th></th>
            <th>@Html.DisplayNameFor(model => model.MainPlatform)</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model) {
            <tr>
                <td>@Html.DisplayFor(modelItem => item.CategoryId)</td>
                <td>@Html.DisplayFor(modelItem => item.CategoryName)</td>
                <td>@Html.DisplayFor(modelItem => item.AdDescription)</td>
                <td>@Html.DisplayFor(modelItem => item.MainPlatform)</td>
            </tr>
        }
    </tbody>
</table>

```

The Solution Explorer on the right shows the project structure for "OnlineAdz":

- Connected Services
- Dependencies
- Properties
- wwwroot
- Controllers
- Migrations
- Models
- Views
 - Account
 - CategoryAd
 - Home
 - Platform
 - _ViewStart.cshtml
 - Index.cshtml
 - Shared
 - UserAd
 - UserInterest
 - _ViewImports.cshtml
 - _ViewStart.cshtml
- SharedSettings.json
- DataContext.cs
- Program.cs
- Startup.cs

Views: Shared:

Layout:

The screenshot shows the Visual Studio IDE interface. On the left, the Server Explorer and Toolbox are visible. The main area displays the `_Layout.cshtml` file content:

```

<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <title>ViewData["Title"] - OnlineAdz</title>
        <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
        <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
        <link rel="stylesheet" href="~/OnlineAdz.styles.css" asp-append-version="true" />
    </head>
    <body>
        <header>
            <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-bottom box-shadow mb-3">
                <div class="container-fluid">
                    <a class="navbar-brand" asp-area="" asp-controller="Home" asp-action="Index">OnlineAdz</a>
                    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target=".navbar-collapse" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
                        <span class="navbar-toggler-icon"></span>
                    </button>
                    <div class="navbar-collapse collapse d-sm-inline-flex justify-content-between">
                        <ul class="navbar-nav flex-grow-1">
                            <li class="nav-item">
                                <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Home</a>
                            </li>
                            <li class="nav-item">
                                <a class="nav-link text-dark" asp-area="" asp-controller="Account" asp-action="Register">Register</a>
                            </li>
                            <li class="nav-item">
                                <a class="nav-link text-dark" asp-area="" asp-controller="Account" asp-action="Login">Login</a>
                            </li>
                        </ul>
                    </div>
                </div>
            </nav>
        </header>
        <div class="container">
            <partial name="/_ValidationScriptsPartial" />
            <main>
                <partial name="/_Layout" />
            </main>
            <partial name="/_ValidationScriptsPartial" />
        </div>
    </body>
</html>

```

The Solution Explorer on the right shows the project structure for "OnlineAdz":

- Connected Services
- Dependencies
- Properties
- wwwroot
- Controllers
- Migrations
- Models
- Views
 - Account
 - CategoryAd
 - Home
 - Platform
 - _ViewStart.cshtml
 - Index.cshtml
 - Layout.cshtml
 - Layout2.cshtml
 - _ValidationScriptsPartial.cshtml
 - Error.cshtml
 - Shared
 - _Layout.cshtml
 - Layout.cshtml
 - Layout2.cshtml
 - _ValidationScriptsPartial.cshtml
 - Error.cshtml
 - UserAd
 - UserInterest
 - _ViewImports.cshtml
 - _ViewStart.cshtml
- SharedSettings.json
- DataContext.cs
- Program.cs
- Startup.cs

_Layout1:

The screenshot shows the Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search (Ctrl+Q). The title bar says "OnlineAdz". The main area displays the code for `_Layout1.cshtml`. The code defines a `<head>` section with meta tags for charset, viewport, and title. It includes links to Bootstrap CSS and a custom stylesheet. The `<body>` section has a background image and a navigation bar with items for Home, Category, and User. The Solution Explorer on the right shows the project structure for "OnlineAdz" with files like `Startup.cs`, `Program.cs`, and various controller and model files.

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <title>ViewData["Title"] - OnlineAdz</title>
        <link rel="stylesheet" href="/lib/bootstrap/dist/css/bootstrap.min.css" />
        <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
        <link rel="stylesheet" href="~/OnlineAdz.styles.css" asp-append-version="true" />
    </head>
    <body style="background-image:url('~/css/Images/sixteen.jpg');">
        <header>
            <div class="container-fluid">
                <a class="navbar-brand" asp-area="" asp-controller="CategoryAd" asp-action="Index">OnlineAdz</a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target=".navbar-collapse" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="navbar-collapse collapse d-sm-inline-flex justify-content-between">
                    <ul class="navbar-nav flex-grow-1">
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Home</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="CategoryAd" asp-action="Index">Ad Category</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="UserAd" asp-action="Index">Users and Ads</a>
                        </li>
                    </ul>
                </div>
            </div>
        </header>
        <div class="container">
            <div class="row">
                <div class="col-3">
                    <div>
                        <h2>Category Ad</h2>
                        <p>This is a sample category ad page. It displays ads from different categories. You can search by category or user interest.</p>
                    </div>
                </div>
                <div class="col-9">
                    <div>
                        <h3>Recent Ads</h3>
                        <table border="1">
                            <thead>
                                <tr>
                                    <th>Category</th>
                                    <th>User Interest</th>
                                    <th>Ad Title</th>
                                    <th>Ad Description</th>
                                </tr>
                            </thead>
                            <tbody>
                                <tr>
                                    <td>Electronics</td>
                                    <td>Tech Enthusiast</td>
                                    <td>Smartphones</td>
                                    <td>The latest smartphones from major brands. Find the best deals here!</td>
                                </tr>
                                <tr>
                                    <td>Clothing</td>
                                    <td>Fashionista</td>
                                    <td>Designer Clothing</td>
                                    <td>Find the latest fashion trends and designer clothing at great prices!</td>
                                </tr>
                                <tr>
                                    <td>Sports Equipment</td>
                                    <td>Athlete</td>
                                    <td>Golf Clubs</td>
                                    <td>Find the best golf clubs and accessories for your game!</td>
                                </tr>
                                <tr>
                                    <td>Food & Beverage</td>
                                    <td>Foodie</td>
                                    <td>Italian Restaurant</td>
                                    <td>Find the best Italian restaurant deals in town!</td>
                                </tr>
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </div>
    </body>
</html>
```

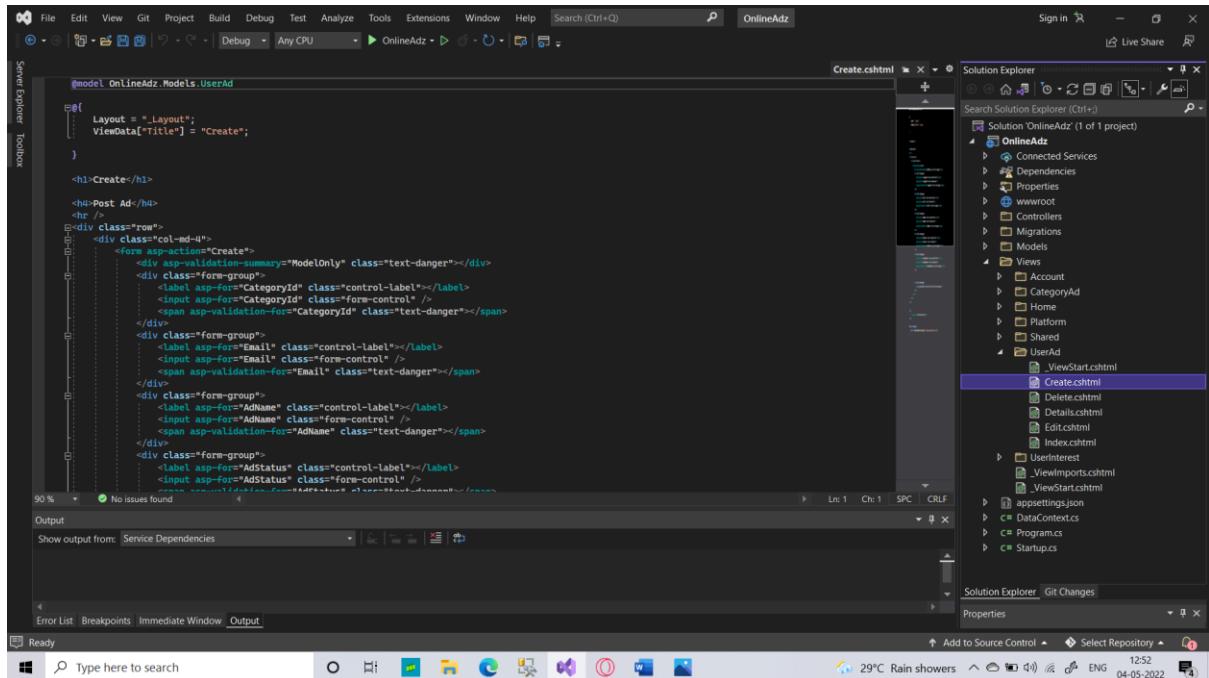
_Layout2:

The screenshot shows the Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search (Ctrl+Q). The title bar says "OnlineAdz". The main area displays the code for `_Layout2.cshtml`. The code is identical to `_Layout1.cshtml` except for the background image URL, which is set to `~/css/Images/nine.jpg`. The Solution Explorer on the right shows the project structure for "OnlineAdz" with files like `Startup.cs`, `Program.cs`, and various controller and model files.

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <title>ViewData["Title"] - OnlineAdz</title>
        <link rel="stylesheet" href="/lib/bootstrap/dist/css/bootstrap.min.css" />
        <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
        <link rel="stylesheet" href="~/OnlineAdz.styles.css" asp-append-version="true" />
    </head>
    <body style="background-image:url('~/css/Images/nine.jpg');">
        <header>
            <div class="container-fluid">
                <a class="navbar-brand" asp-area="" asp-controller="UserAd" asp-action="Index">OnlineAdz</a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target=".navbar-collapse" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="navbar-collapse collapse d-sm-inline-flex justify-content-between">
                    <ul class="navbar-nav flex-grow-1">
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Home</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="UserAd" asp-action="Index">Users and Ads</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="Platform" asp-action="Index">About Ads</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="UserInterest" asp-action="Index">User Interest</a>
                        </li>
                    </ul>
                </div>
            </div>
        </header>
        <div class="container">
            <div class="row">
                <div class="col-3">
                    <div>
                        <h2>Category Ad</h2>
                        <p>This is a sample category ad page. It displays ads from different categories. You can search by category or user interest.</p>
                    </div>
                </div>
                <div class="col-9">
                    <div>
                        <h3>Recent Ads</h3>
                        <table border="1">
                            <thead>
                                <tr>
                                    <th>Category</th>
                                    <th>User Interest</th>
                                    <th>Ad Title</th>
                                    <th>Ad Description</th>
                                </tr>
                            </thead>
                            <tbody>
                                <tr>
                                    <td>Electronics</td>
                                    <td>Tech Enthusiast</td>
                                    <td>Smartphones</td>
                                    <td>The latest smartphones from major brands. Find the best deals here!</td>
                                </tr>
                                <tr>
                                    <td>Clothing</td>
                                    <td>Fashionista</td>
                                    <td>Designer Clothing</td>
                                    <td>Find the latest fashion trends and designer clothing at great prices!</td>
                                </tr>
                                <tr>
                                    <td>Sports Equipment</td>
                                    <td>Athlete</td>
                                    <td>Golf Clubs</td>
                                    <td>Find the best golf clubs and accessories for your game!</td>
                                </tr>
                                <tr>
                                    <td>Food & Beverage</td>
                                    <td>Foodie</td>
                                    <td>Italian Restaurant</td>
                                    <td>Find the best Italian restaurant deals in town!</td>
                                </tr>
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </div>
    </body>
</html>
```

Views: UserAd:

Create:



The screenshot shows the Visual Studio IDE interface with the 'Create.cshtml' file open in the main editor. The code is a standard ASP.NET MVC form for creating a 'UserAd'. It includes fields for CategoryId, Email, AdName, AdStatus, and UserInterest, each with validation messages. The Solution Explorer on the right shows the project structure, including the 'Views/UserAd' folder which contains the 'Create.cshtml' file.

```
@model OnlineAdz.Models.UserAd

@{
    Layout = "~/Layout";
    ViewData["Title"] = "Create";
}



# Create



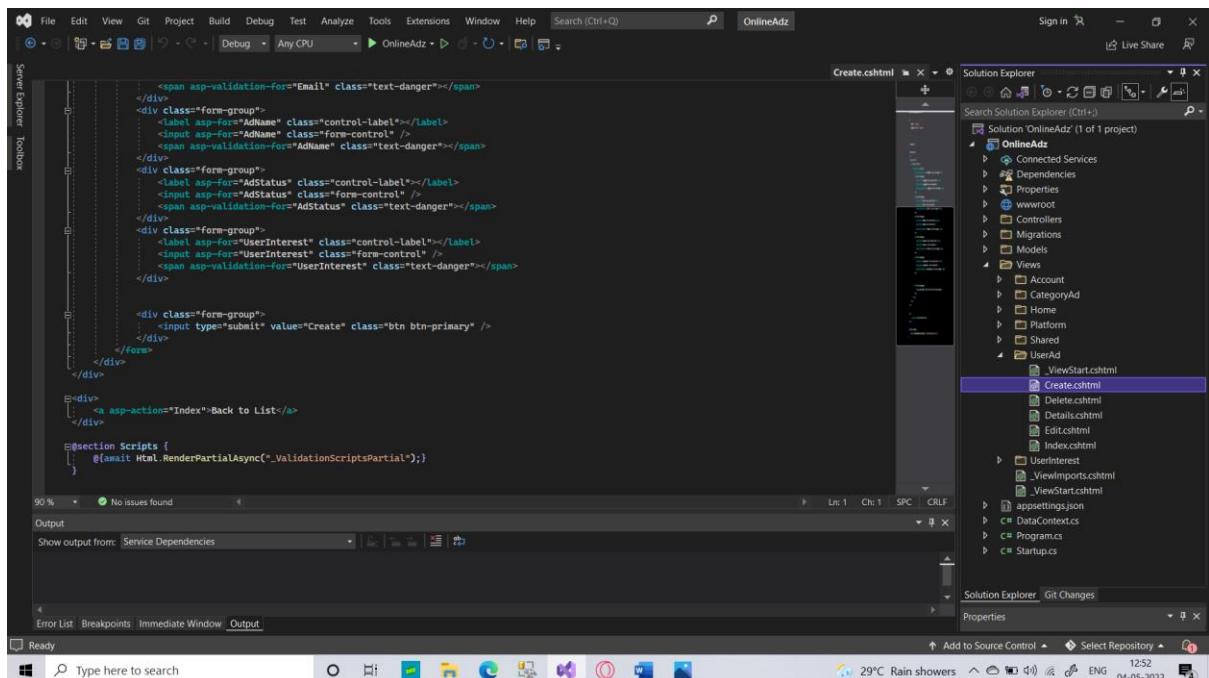
---



<form action="Create">
    <div asp-validation-summary="ModelOnly" class="text-danger"></div>
    <div class="form-group">
        <label asp-for="CategoryId" class="control-label"></label>
        <input asp-for="CategoryId" class="form-control" />
        <span asp-validation-for="CategoryId" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Email" class="control-label"></label>
        <input asp-for="Email" class="form-control" />
        <span asp-validation-for="Email" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="AdName" class="control-label"></label>
        <input asp-for="AdName" class="form-control" />
        <span asp-validation-for="AdName" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="AdStatus" class="control-label"></label>
        <input asp-for="AdStatus" class="form-control" />
        <span asp-validation-for="AdStatus" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="UserInterest" class="control-label"></label>
        <input asp-for="UserInterest" class="form-control" />
        <span asp-validation-for="UserInterest" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="Create" class="btn btn-primary" />
    </div>
</form>


<div>
    <a href="#">Back to List</a>
</div>

@section Scripts {
    @await Html.RenderPartialAsync("_ValidationScriptsPartial");
}
```



This screenshot shows the same 'Create.cshtml' view as above, but with additional code at the bottom. It includes a section for scripts and a partial view for validation scripts. The rest of the code is identical to the first screenshot.

```
@model OnlineAdz.Models.UserAd

@{
    Layout = "~/Layout";
    ViewData["Title"] = "Create";
}



# Create



---

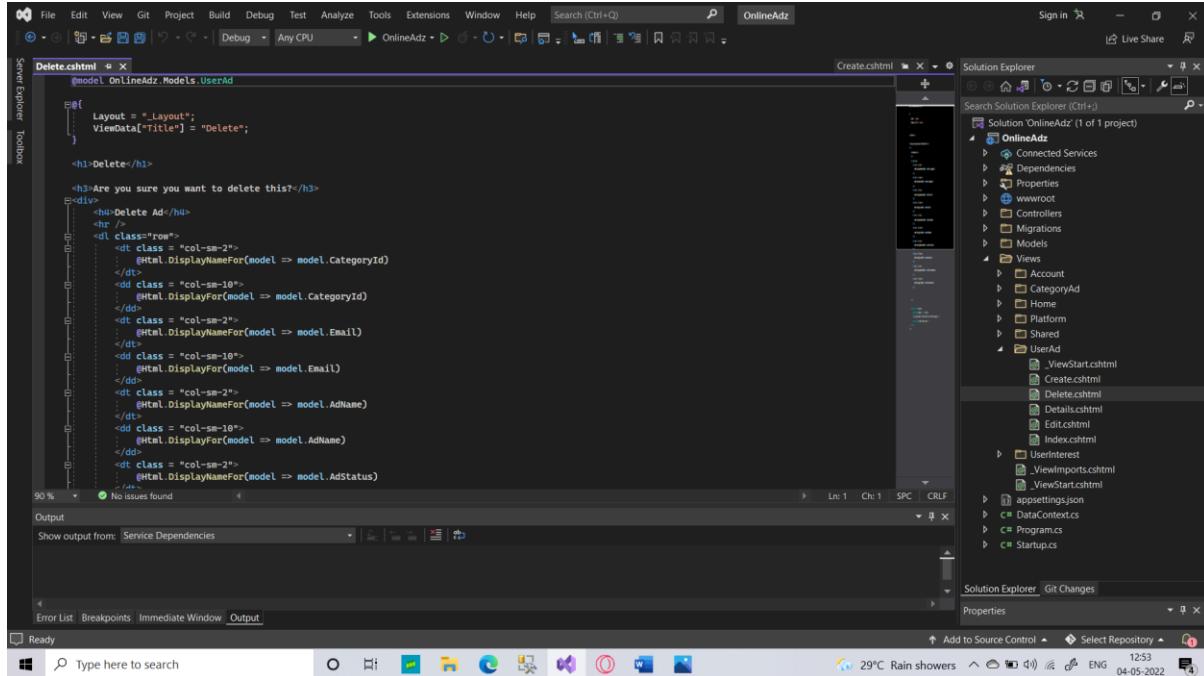


<form action="Create">
    <div asp-validation-summary="ModelOnly" class="text-danger"></div>
    <div class="form-group">
        <label asp-for="CategoryId" class="control-label"></label>
        <input asp-for="CategoryId" class="form-control" />
        <span asp-validation-for="CategoryId" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Email" class="control-label"></label>
        <input asp-for="Email" class="form-control" />
        <span asp-validation-for="Email" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="AdName" class="control-label"></label>
        <input asp-for="AdName" class="form-control" />
        <span asp-validation-for="AdName" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="AdStatus" class="control-label"></label>
        <input asp-for="AdStatus" class="form-control" />
        <span asp-validation-for="AdStatus" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="UserInterest" class="control-label"></label>
        <input asp-for="UserInterest" class="form-control" />
        <span asp-validation-for="UserInterest" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="Create" class="btn btn-primary" />
    </div>
</form>


<div>
    <a href="#">Back to List</a>
</div>

@section Scripts {
    @await Html.RenderPartialAsync("_ValidationScriptsPartial");
}
```

Delete:



The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q) and OnlineAdz.
- Solution Explorer:** Shows the project structure for "OnlineAdz".
- Code Editor:** The current file is "Delete.cshtml" under the "Views/UserAd" folder. The code is as follows:

```
@model OnlineAdz.Models.UserAd

@{
    Layout = "_Layout";
    ViewData["Title"] = "Delete";
}



# Delete



Are you sure you want to delete this?



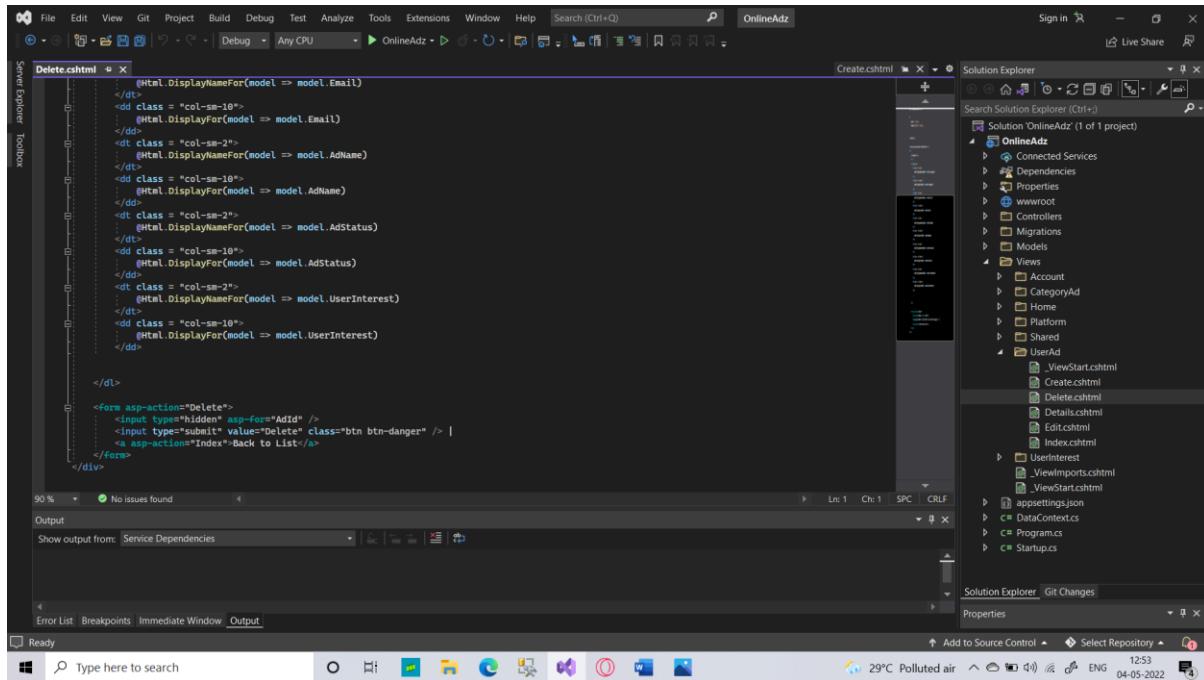
---



|                                                                         |                                                                                          |
|-------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
| <input asp-for="AdId" type="hidden" value="1"/>                         | <input class="btn btn-danger" data-bbox="298 534 393 568" type="submit" value="Delete"/> |
| <input data-bbox="298 601 457 635" type="button" value="Back to List"/> |                                                                                          |


```

The code editor shows syntax highlighting and a code completion dropdown.



The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q) and OnlineAdz.
- Solution Explorer:** Shows the project structure for "OnlineAdz".
- Code Editor:** The current file is "Delete.cshtml" under the "Views/UserAd" folder. The code has been modified to include additional fields and logic:

```
@model OnlineAdz.Models.UserAd



|                                                                         |                                                                                          |
|-------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
| <input asp-for="AdId" type="hidden" value="1"/>                         | <input class="btn btn-danger" data-bbox="298 534 393 568" type="submit" value="Delete"/> |
| <input data-bbox="298 601 457 635" type="button" value="Back to List"/> |                                                                                          |


```

The code editor shows syntax highlighting and a code completion dropdown.

Details:

The screenshot shows the Visual Studio IDE interface. The main window displays the `Details.cshtml` file content:

```
@model OnlineAdz.Models.UserAd
@{
    Layout = "_Layout";
    ViewData["Title"] = "Details";
}


# Details



## Ad Details



---



Html.DisplayNameFor(model => model.CategoryId)


Html.DisplayFor(model => model.CategoryId)


Html.DisplayNameFor(model => model.Email)


Html.DisplayFor(model => model.Email)


Html.DisplayNameFor(model => model.AdName)


Html.DisplayFor(model => model.AdName)


Html.DisplayNameFor(model => model.AdStatus)


Html.DisplayFor(model => model.AdStatus)



The Solution Explorer on the right shows the project structure for OnlineAdz:



- Connected Services
- Dependencies
- Properties
- wwwroot
- Controllers
- Migrations
- Models
- Views
  - Account
  - CategoryAd
  - Home
  - Platform
  - Shared
  - UserAd
    - _ViewStart.cshtml
    - Create.cshtml
    - Delete.cshtml
    - Details.cshtml
    - Edit.cshtml
    - Index.cshtml
  - UserInterest
    - _ViewImports.cshtml
    - _ViewStart.cshtml
  - appsettings.json
  - DataContext.cs
  - Program.cs
  - Startup.cs

```

The screenshot shows the Visual Studio IDE interface with the `Details.cshtml` file content updated to include navigation links:

```
<dt class="col-sm-2">Html.DisplayNameFor(model => model.Email)</dt>
<dd class="col-sm-10">Html.DisplayFor(model => model.Email)</dd>
<dt class="col-sm-2">Html.DisplayNameFor(model => model.AdName)</dt>
<dd class="col-sm-10">Html.DisplayFor(model => model.AdName)</dd>
<dt class="col-sm-2">Html.DisplayNameFor(model => model.AdStatus)</dt>
<dd class="col-sm-10">Html.DisplayFor(model => model.AdStatus)</dd>
<dt class="col-sm-2">Html.DisplayNameFor(model => model.UserInterest)</dt>
<dd class="col-sm-10">Html.DisplayFor(model => model.UserInterest)</dd>

```

```
</div>
</div>
<a href="#" asp-action="Edit" asp-route-id="@Model?.AdId">Edit</a> | 
<a href="#" asp-action="Index">Back to List</a>
```

The Solution Explorer on the right shows the project structure for `OnlineAdz`:

- Connected Services
- Dependencies
- Properties
- wwwroot
- Controllers
- Migrations
- Models
- Views
 - Account
 - CategoryAd
 - Home
 - Platform
 - Shared
 - UserAd
 - _ViewStart.cshtml
 - Create.cshtml
 - Delete.cshtml
 - Details.cshtml
 - Edit.cshtml
 - Index.cshtml
 - UserInterest
 - _ViewImports.cshtml
 - _ViewStart.cshtml
 - appsettings.json
 - DataContext.cs
 - Program.cs
 - Startup.cs

Edit:

The screenshot shows the Visual Studio IDE interface with the following details:

- Title Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Search (Ctrl+Q), OnlineAdz.
- Solution Explorer:** Shows the project structure for "OnlineAdz" with files like Connected Services, Dependencies, Properties, wwwroot, Controllers, Migrations, Models, Views, and UserAd.
- Code Editor:** Displays the "Edit.cshtml" view for the "UserAd" model. The code includes form validation for CategoryId, Email, AdName, and AdStatus.
- Status Bar:** Shows the date (04-05-2022) and time (12:54).

The screenshot shows the Visual Studio IDE interface with the following details:

- Title Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, Search (Ctrl+Q), OnlineAdz.
- Solution Explorer:** Shows the project structure for "OnlineAdz" with files like Connected Services, Dependencies, Properties, wwwroot, Controllers, Migrations, Models, Views, and UserAd.
- Code Editor:** Displays the "Edit.cshtml" view for the "UserAd" model. The code includes form validation for CategoryId, Email, AdName, and AdStatus, and a "Save" button.
- Output:** Shows the addition of a back link: `[Back to List](#)`.
- Status Bar:** Shows the date (04-05-2022) and time (12:54).

Index:

The screenshot shows the Visual Studio IDE interface. The main window displays the `Index.cshtml` file content:

```
@model IEnumerable<OnlineAdz.Models.UserAd>
@{
    // Layout = "~/Views/Shared/_Layout2.cshtml";
    ViewData["Title"] = "Index";
}



# User



Create New



@using (Html.BeginForm("Index", "UserAd", FormMethod.Get))
    {
        <b>Search :</b>
        @Html.RadioButton("option", "Email")<text>Email </text>
        @Html.RadioButton("option", "AdName")<text>AdName</text><br />
        @Html.TextBox("search")<input type="submit" name="submit" value="search" />
    }



| AdId | AdName  | Email            | Category   | Status |
|------|---------|------------------|------------|--------|
| 1    | Test Ad | test@example.com | Category A | Active |


```

The Solution Explorer on the right shows the project structure for `OnlineAdz`:

- Connected Services
- Dependencies
- Properties
- wwwroot
- Controllers
- Migrations
- Models
- Views
 - Account
 - CategoryAd
 - Home
 - Platform
 - Shared
 - UserAd
 - _ViewStart.cshtml
 - Create.cshtml
 - Delete.cshtml
 - Details.cshtml
 - Edit.cshtml
 - Index.cshtml
 - UserInterest
 - _ViewImports.cshtml
 - _ViewStart.cshtml
 - appsettings.json
 - DataContext.cs
 - Program.cs
 - Startup.cs

The screenshot shows the Visual Studio IDE interface. The main window displays the `Index.cshtml` file content, which has been modified to include additional columns in the table header:

```
@model IEnumerable<OnlineAdz.Models.UserAd>
@{
    // Layout = "~/Views/Shared/_Layout2.cshtml";
    ViewData["Title"] = "Index";
}



# User



Create New



@using (Html.BeginForm("Index", "UserAd", FormMethod.Get))
    {
        <b>Search :</b>
        @Html.RadioButton("option", "Email")<text>Email </text>
        @Html.RadioButton("option", "AdName")<text>AdName</text><br />
        @Html.TextBox("search")<input type="submit" name="submit" value="search" />
    }



| AdId | AdName  | Email            | Category   | Status | User Interest |
|------|---------|------------------|------------|--------|---------------|
| 1    | Test Ad | test@example.com | Category A | Active | High          |


```

The Solution Explorer on the right shows the project structure for `OnlineAdz`:

- Connected Services
- Dependencies
- Properties
- wwwroot
- Controllers
- Migrations
- Models
- Views
 - Account
 - CategoryAd
 - Home
 - Platform
 - Shared
 - UserAd
 - _ViewStart.cshtml
 - Create.cshtml
 - Delete.cshtml
 - Details.cshtml
 - Edit.cshtml
 - Index.cshtml
 - UserInterest
 - _ViewImports.cshtml
 - _ViewStart.cshtml
 - appsettings.json
 - DataContext.cs
 - Program.cs
 - Startup.cs

The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q).
- Solution Explorer:** Shows the solution "OnlineAdz" with one project "OnlineAdz".
- Code Editor:** Displays the "Index.cshtml" file content. The code includes a table structure with columns for Email, Category Id, and Ad Name, and links for Edit, Details, and Delete actions.
- Output Window:** Shows "No issues found".
- Taskbar:** Includes icons for Error List, Breakpoints, Immediate Window, and Output.
- System Tray:** Shows battery status, network connection, and system date/time.

Views: UserInterest:

Index:

The screenshot shows the Visual Studio IDE interface with the following details:

- File Menu:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search (Ctrl+Q).
- Solution Explorer:** Shows the solution "OnlineAdz" with one project "OnlineAdz".
- Code Editor:** Displays the "Index.cshtml" file content for the "UserInterest" model. It shows a table with columns for Email, Category Id, and Ad Name, each with a corresponding text input field.
- Output Window:** Shows "No issues found".
- Taskbar:** Includes icons for Error List, Breakpoints, Immediate Window, and Output.
- System Tray:** Shows battery status, network connection, and system date/time.

The screenshot shows the Visual Studio IDE interface. The main window displays the code for the `Delete.cshtml` view, which contains HTML and C# code for a table form. The code includes fields for Category ID, Ad Name, and Ad Desc, each with a text input field. A submit button is also present. The Solution Explorer on the right shows the project structure for "OnlineAdz" with files like `Index.cshtml`, `_ViewStart.cshtml`, and `Startup.cs`.

```
<tr>
    <td><h5>Category Id:</h5></td>
    <td>@Html.TextBoxFor(m => m.CategoryId)</td>
</tr>
<tr>
    <td><h5>Ad Name:</h5></td>
    <td>@Html.TextBoxFor(m => m.AdName)</td>
</tr>
<tr>
    <td><h5>Ad Desc.:</h5></td>
    <td>@Html.TextBoxFor(m => m.AdDesc)</td>
</tr>
<tr>
    <td></td>
    <td><input type="submit" value="Submit" style="background-color:blue;color:white;border:none;" /></td>
</tr>
</table>
}
</body>
```

Views:ViewStart:

The screenshot shows the Visual Studio IDE interface. The main window displays the code for the `_ViewStart.cshtml` file, which contains a single line of code: `Layout = "_Layout";`. The Solution Explorer on the right shows the project structure for "OnlineAdz" with files like `Index.cshtml`, `_ViewImports.cshtml`, and `Startup.cs`.

```
Layout = "_Layout";
```

Appsettings.json:

The screenshot shows the Visual Studio interface with the 'appsettings.json' file open in the code editor. The file contains configuration settings for a database connection and logging levels. The Solution Explorer on the right shows the project structure, including files like 'Delete.cshtml', 'Index.cshtml', 'Program.cs', and 'Startup.cs'. The status bar at the bottom indicates the date as 04-05-2022.

```
{
  "ConnectionStrings": {
    "WindowsAuthentication": true,
    "DefaultConnection": "Data Source=.;Database=OnlineAdz; uid=sa;pwd=12345;"
  },
  "Logging": {
    "LogLevel": {
      "Default": "Warning"
    }
  },
  "AllowedHosts": "*"
}
```

DataContext:

The screenshot shows the Visual Studio interface with the 'DataContext.cs' file open in the code editor. The file defines a custom 'DataContext' class that inherits from 'IdentityDbContext<IdentityUser, IdentityRole, string, IdentityUserClaim<string>, IdentityUserRole<string>>'. It includes a constructor that takes 'DbContextOptions<DataContext>' options and overrides the 'OnModelCreating' method to configure the 'IdentityUser' entity. The Solution Explorer on the right shows the project structure, including files like 'Delete.cshtml', 'Index.cshtml', 'Program.cs', and 'Startup.cs'. The status bar at the bottom indicates the date as 04-05-2022.

```
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;

namespace OnlineAdz
{
  public class DataContext : IdentityDbContext<IdentityUser, IdentityRole, string, IdentityUserClaim<string>, IdentityUserRole<string>>
  {
    public DataContext(DbContextOptions<DataContext> options)
      : base(options)
    {
    }

    protected override void OnModelCreating(ModelBuilder builder)
    {
      base.OnModelCreating(builder);
      builder.Entity<IdentityUser>();
    }
  }
}
```

Program:

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and Search (Ctrl+Q). The title bar displays "OnlineAdz". The left sidebar has "Server Explorer" and "Toolbox" tabs. The main area features a code editor with "Program.cs" open, containing C# code for a .NET Core application. To the right is the "Solution Explorer" window, which lists the project structure for "OnlineAdz" (1 of 1 project). The "Views" node under "Account" contains files like "_ViewStart.cshtml", "Index.cshtml", "ViewImports.cshtml", and "_ViewStart.cshtml". Other nodes include "Connected Services", "Dependencies", "Properties", "wwwroot", "Controllers", "Migrations", "Models", "Views", "Account", "CategoryAd", "Home", "Platform", "Shared", "UserAd", "UserInterest", "appsettings.json", "DataContext.cs", "Programs", and "Startup.cs". The bottom navigation bar includes tabs for Error List, Breakpoints, Immediate Window, Output, and a "Ready" status message. The status bar at the bottom right shows "AWL -5.00%" and the date "04-05-2022".

StartUp:

```
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.HttpsPolicy;
using Microsoft.AspNetCore.Identity;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using OnlineAdz.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
namespace OnlineAdz
{
    public class Startup
    {
        public Startup(IConfiguration configuration, IWebHostEnvironment env)
        {
            Configuration = configuration;
            environment = env;
            var builder = new ConfigurationBuilder()
                .SetBasePath(env.ContentRootPath)
                .AddJsonFile("appsettings.json", optional: true, reloadOnChange: true)
                .AddJsonFile($"appsettings.{env.EnvironmentName}.json", optional: true)
                .AddEnvironmentVariables();
            Configuration = builder.Build();
        }
        public IConfiguration Configuration { get; }
        public IWebHostEnvironment Environment { get; }
    }
}
```

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the `Startup.cs` file for the `OnlineAdz` project. The code implements cookie policy options and configures the HTTP request pipeline. The Solution Explorer on the right shows the project structure with files like `Index.cshtml`, `ViewImports.cshtml`, and `ViewStart.cshtml`. The status bar at the bottom indicates the current time as 12:59 and the date as 04-05-2022.

```
services.AddControllersWithViews();
services.AddDbContext<DbContext>(
    options => options.UseSqlServer("name=ConnectionString");
    //services.AddIdentity<ApplicationUser, Role>().AddEntityFrameworkStores<DbContext>();
    services.AddIdentity<IdentityUser, IdentityRole>()
        .AddEntityFrameworkStores<DbContext>()
        .AddDefaultTokenProviders();

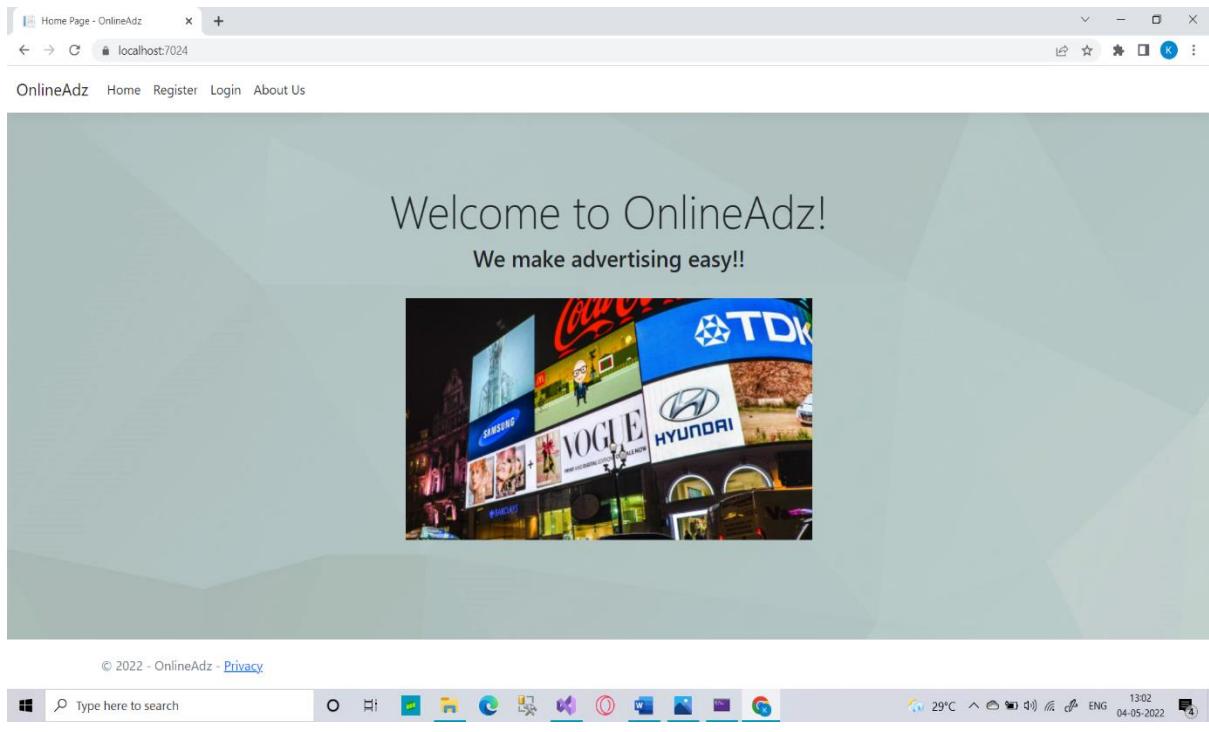
services.ConfigureCookiePolicyOptions > options =>
{
    // This lambda determines whether user consent for non-essential cookies is needed for a given request.
    options.CheckConsentNeeded = context => true;
    options.MinimumSameSitePolicy = SameSiteMode.None;
});
services.AddSession(options =>
{
    options.Cookie.SameSite = SameSiteMode.None;
    options.Cookie.SecurePolicy = CookieSecurePolicy.Always;
    options.Cookie.IsEssential = true;
});

// This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    else
    {
        app.UseExceptionHandler("/Error");
        app.UseHsts();
    }
}

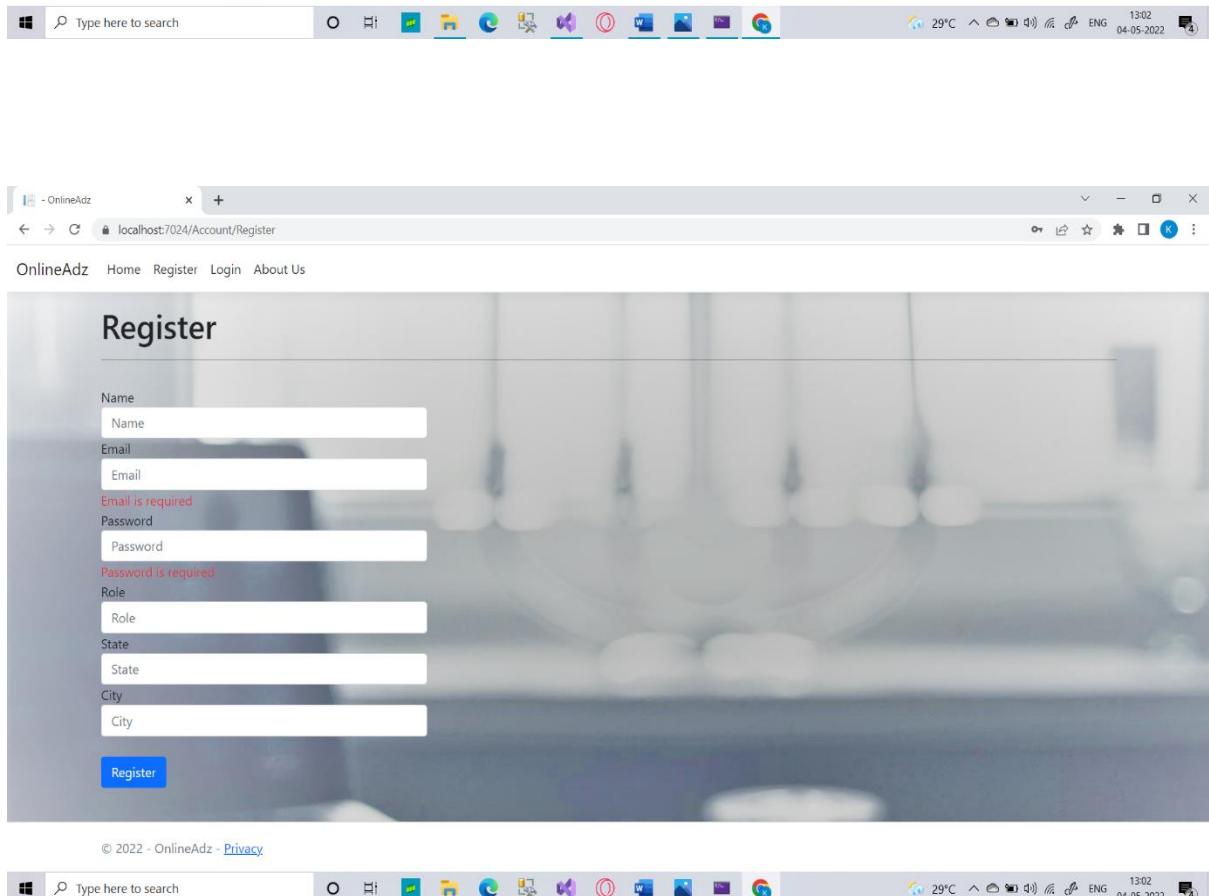
app.UseHttpsRedirection();
app.UseStaticFiles();
app.UseRouting();
app.UseAuthentication();
app.UseAuthorization();
app.UseEndpoints(endpoints =>
{
    endpoints.MapControllerRoute(
        name: "default",
        pattern: "{controller=Home}/{action=Index}/{id?}");
    endpoints.MapRazorPages();
});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


Outputs:

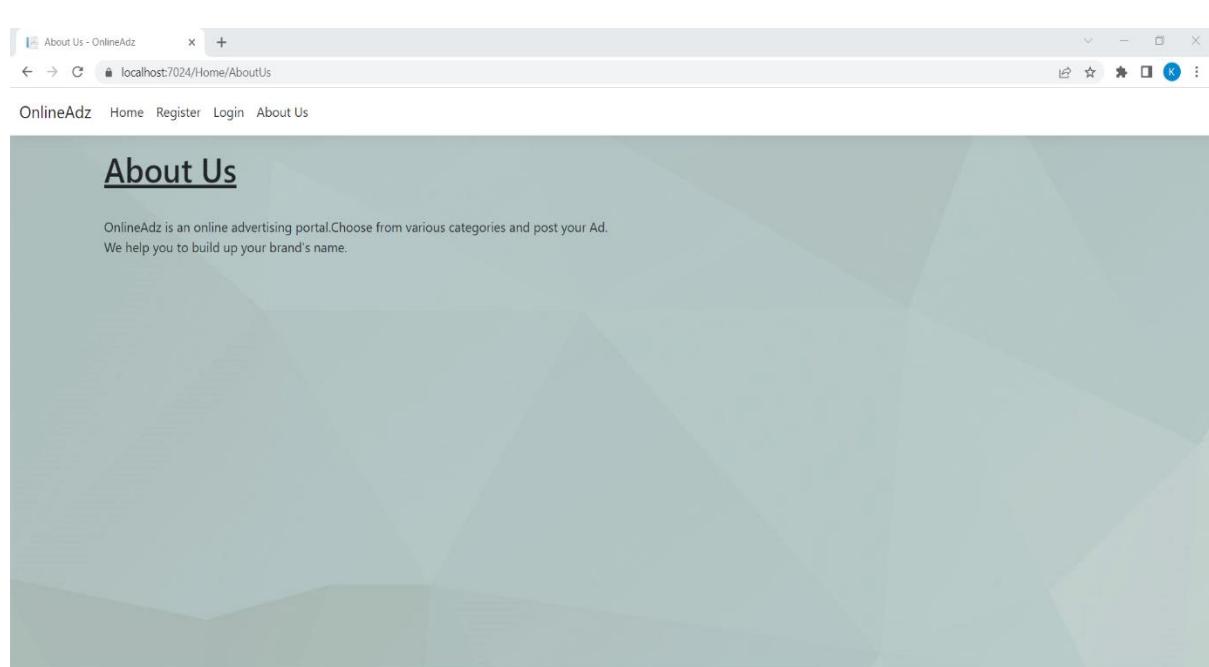
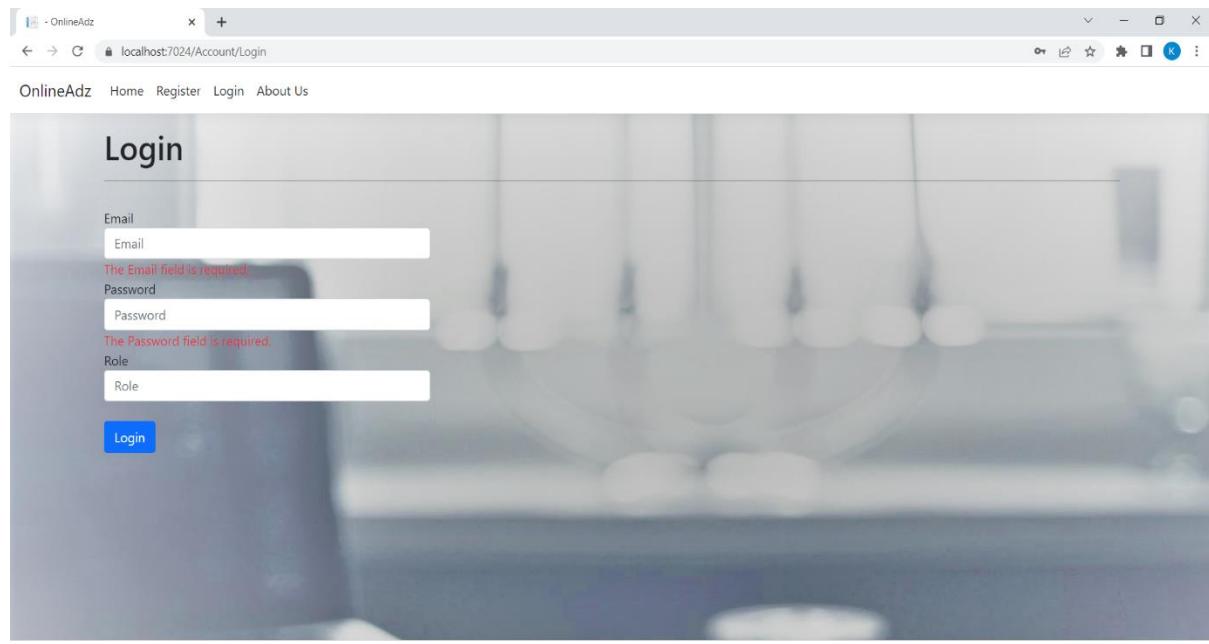
For both Admin and User:

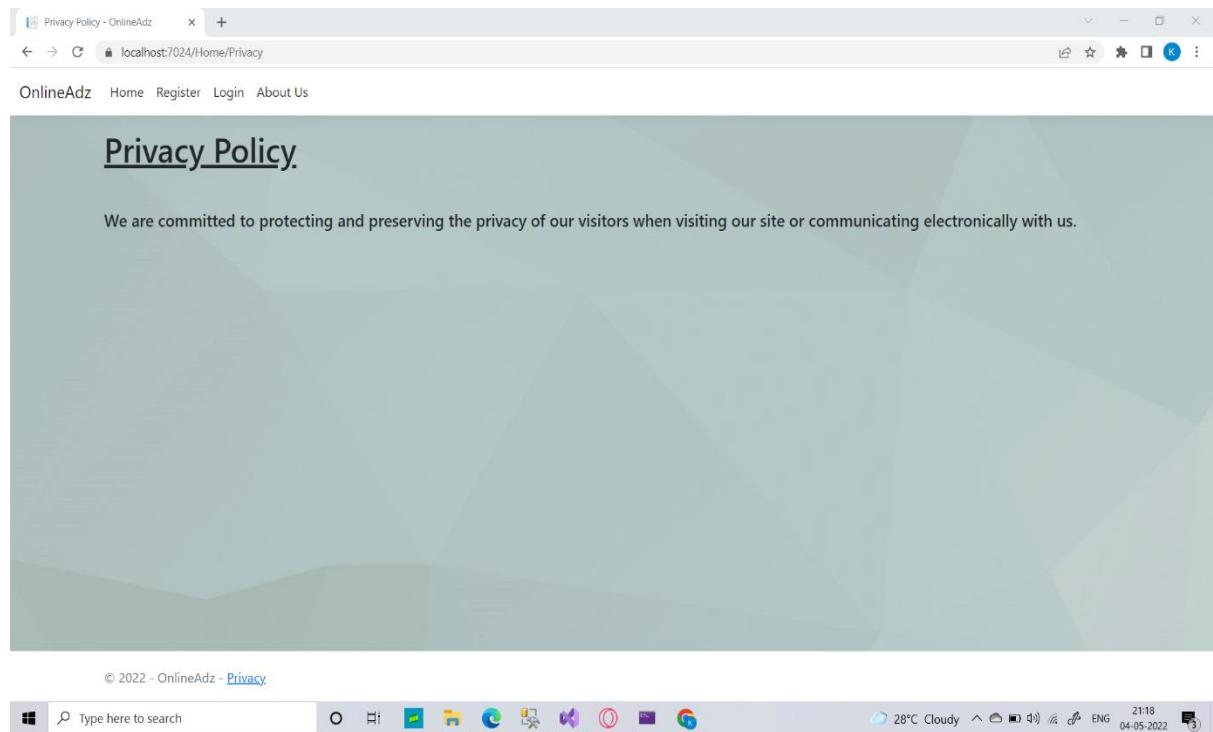


© 2022 - OnlineAdz - [Privacy](#)

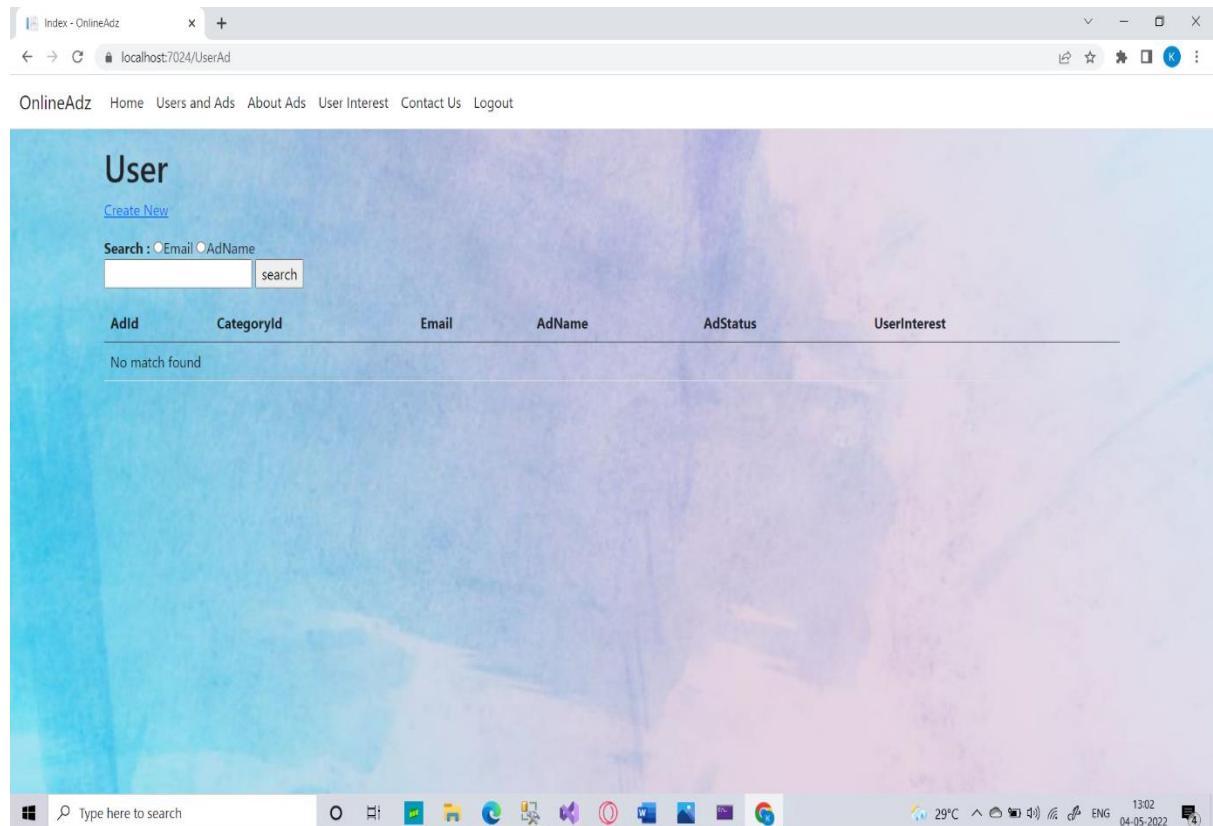


© 2022 - OnlineAdz - [Privacy](#)





For User:



Index - OnlineAdz

localhost:7024/UserAd?option=Email&search=brandon%40gmail.com&submit=search

OnlineAdz Home Users and Ads About Ads User Interest Contact Us Logout

User

Create New

Search : Email AdName
brandon@gmail.com

AdId	CategoryId	Email	AdName	AdStatus	UserInterest	
4	SM	brandon@gmail.com	Jasper Market	Active	Food and Beverages	Edit Details Delete

Type here to search

28°C 21:35 04-05-2022

Create - OnlineAdz

localhost:7024/UserAd/Create

OnlineAdz Home Register Login About Us

Create

Post Ad

CategoryId

Email

AdName

AdStatus

UserInterest

[Back to List](#)

© 2022 - OnlineAdz - [Privacy](#)

Type here to search

28°C 21:37 04-05-2022

Edit - OnlineAdz

localhost:7024/UserAd/Edit/4

OnlineAdz Home Register Login About Us

Edit

Update Details

CategoryId
SM

Email
brandon@gmail.com

AdName
Jasper Market

AdStatus
Active

UserInterest
Food and Beverages

[Save](#)

[Back to List](#)

© 2022 - OnlineAdz - [Privacy](#)



Details - OnlineAdz

localhost:7024/UserAd/Details/4

OnlineAdz Home Register Login About Us

Details

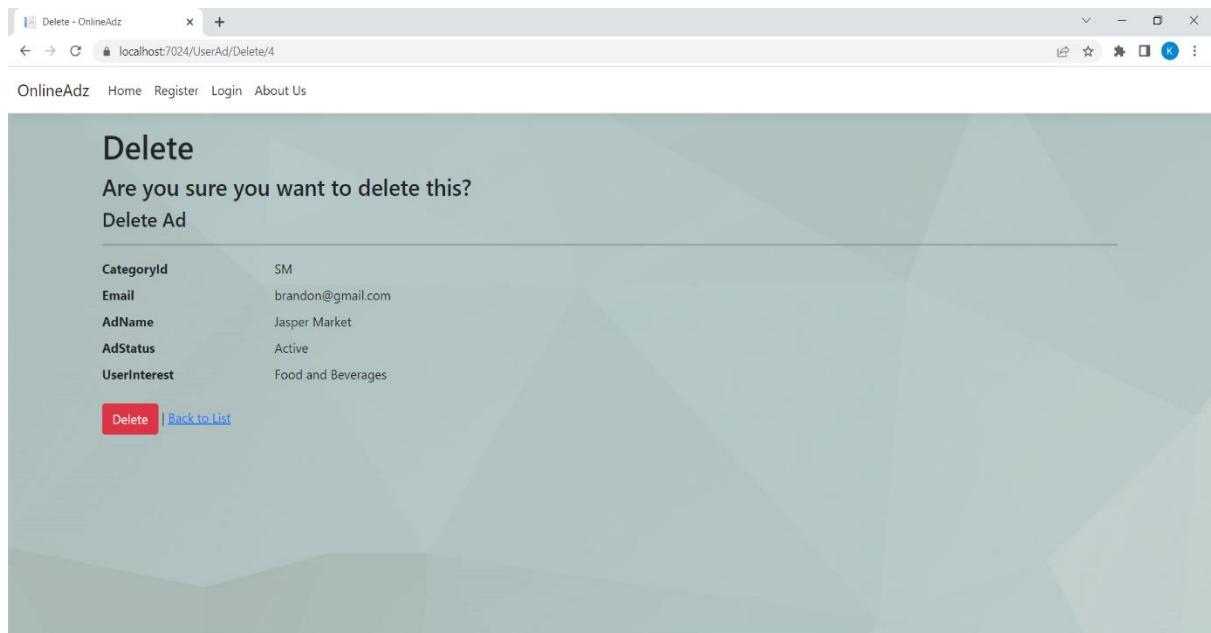
Ad Details

CategoryId	SM
Email	brandon@gmail.com
AdName	Jasper Market
AdStatus	Active
UserInterest	Food and Beverages

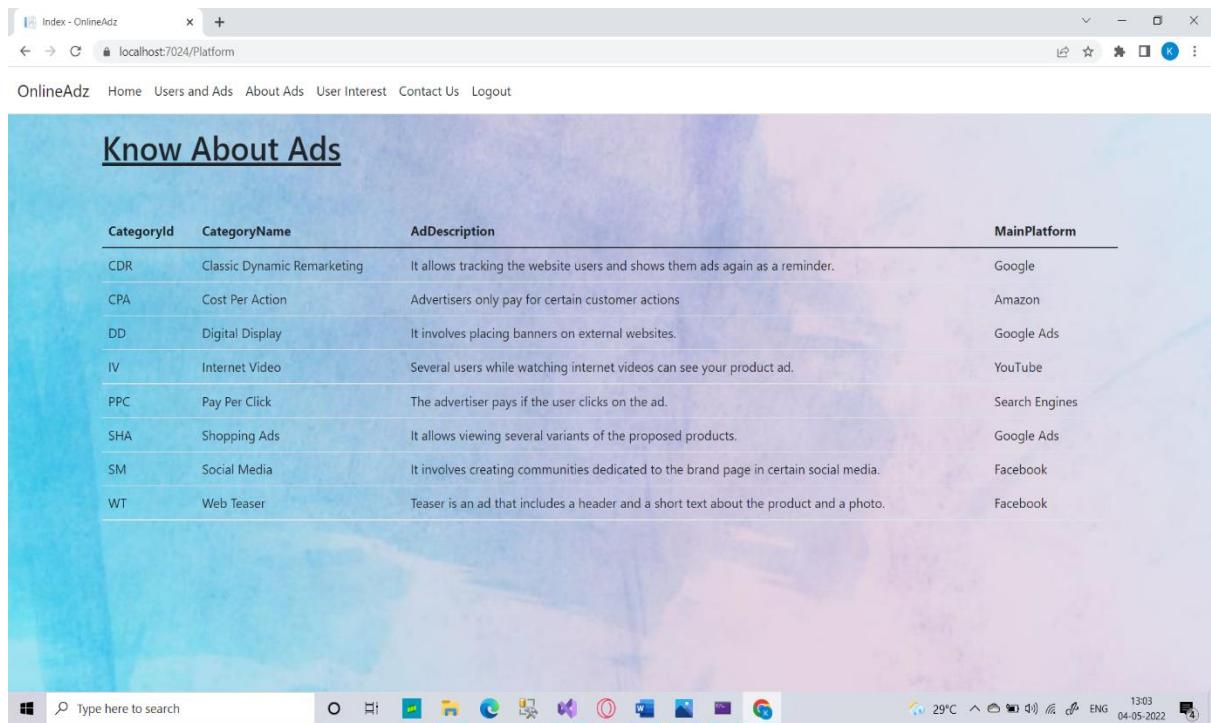
[Edit](#) | [Back to List](#)

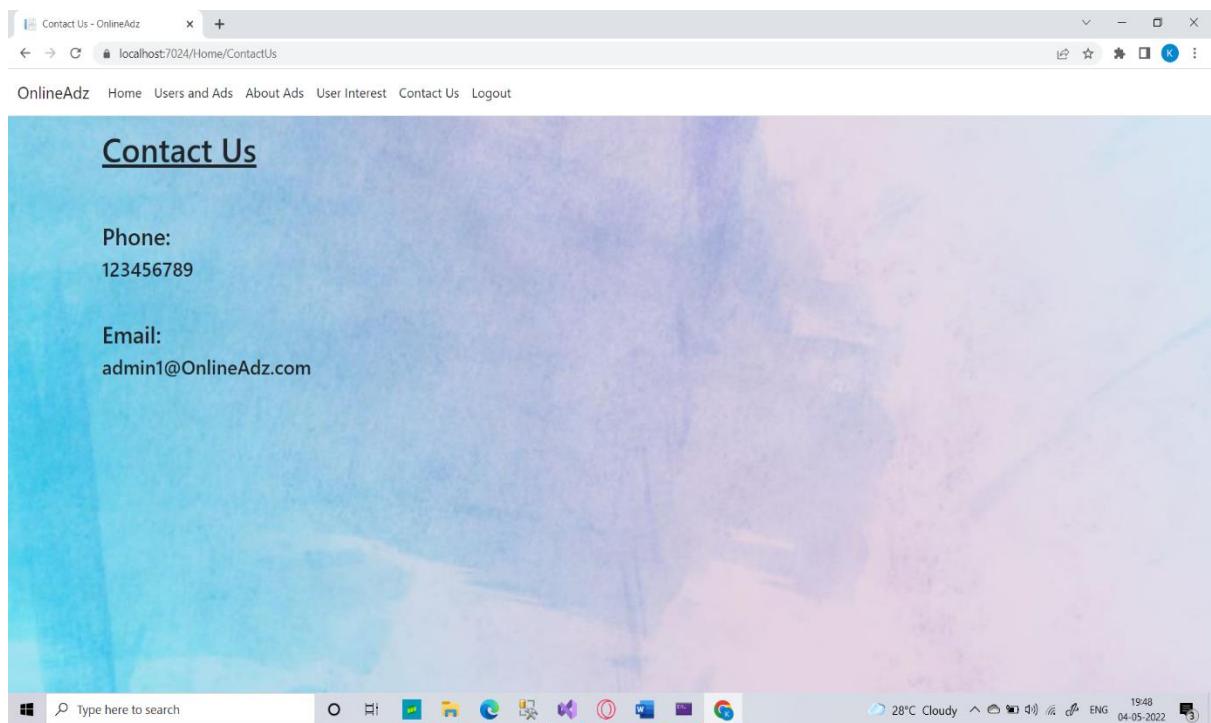
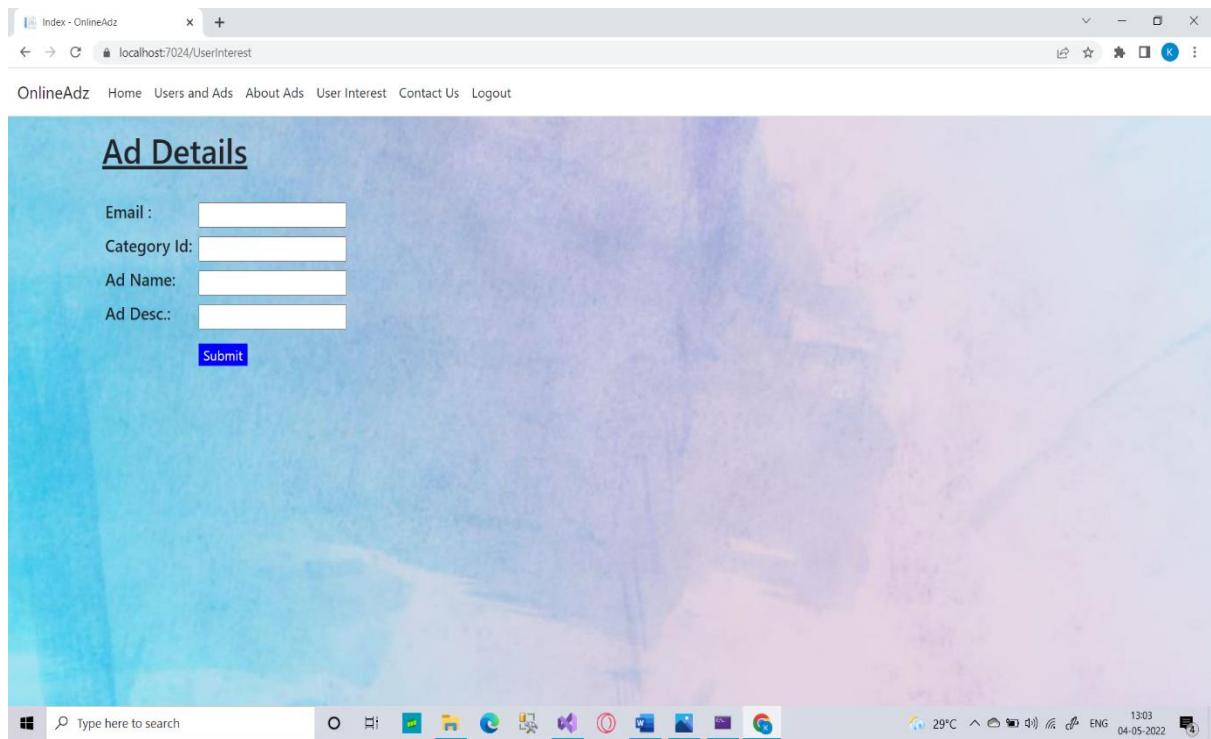
© 2022 - OnlineAdz - [Privacy](#)





© 2022 - OnlineAdz - [Privacy](#)





For Admin:

Index - OnlineAdz

localhost:7024/CategoryAd/Index

OnlineAdz Home Ad Category Users and Ads

Types Of Ads

Create New

CategoryId	CategoryName	
CDR	Classic Dynamic Remarketing	Edit Details Delete
CPA	Cost Per Action	Edit Details Delete
DD	Digital Display	Edit Details Delete
IV	Internet Video	Edit Details Delete
PPC	Pay Per Click	Edit Details Delete
SHA	Shopping Ads	Edit Details Delete
SM	Social Media	Edit Details Delete
WT	Web Teaser	Edit Details Delete

Type here to search

29°C 13:04 04-05-2022

Create - OnlineAdz

localhost:7024/CategoryAd/Create

OnlineAdz Home Register Login About Us

Add new category

Ad Categories

CategoryId

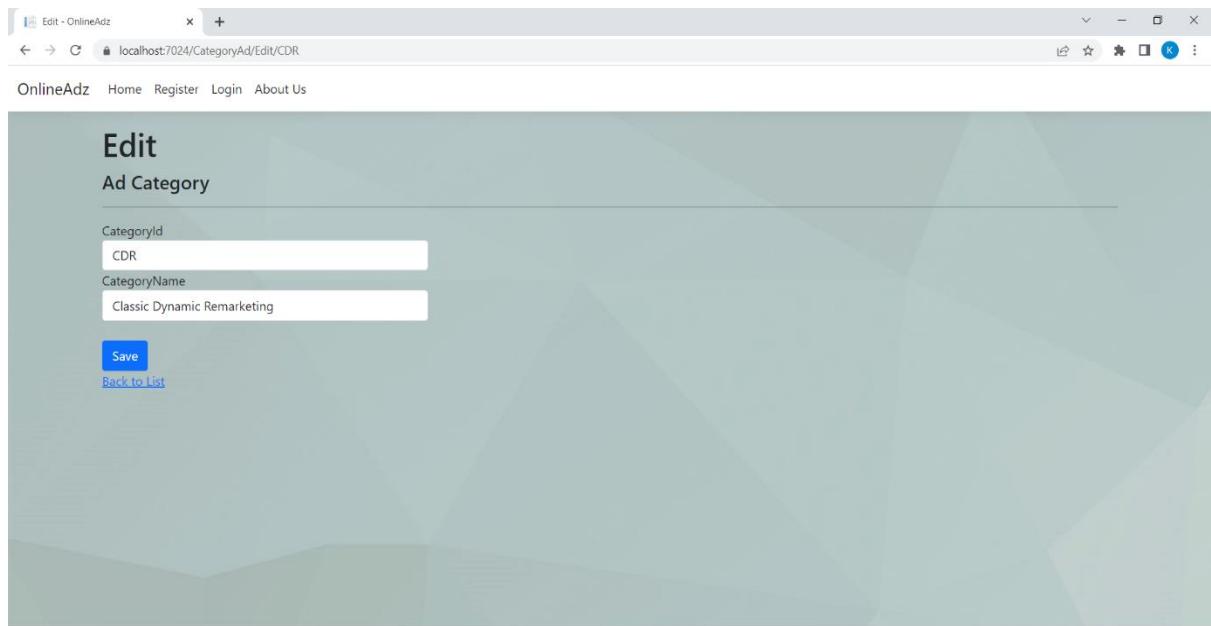
CategoryName

Create

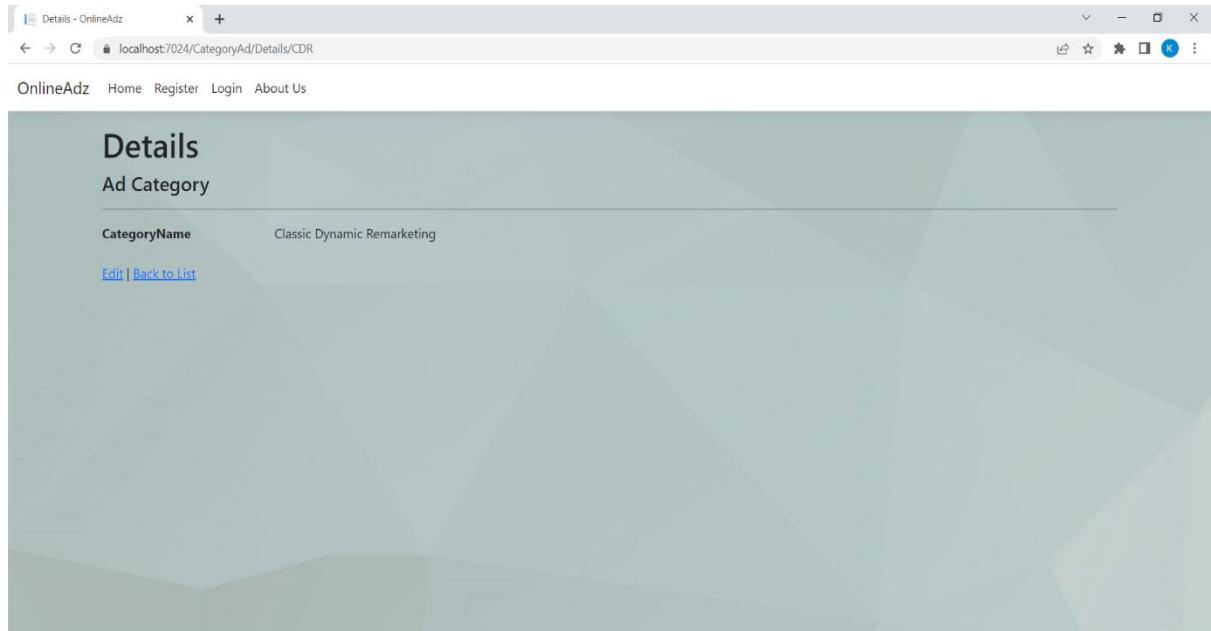
[Back to List](#)

© 2022 - OnlineAdz - [Privacy](#)





© 2022 - OnlineAdz - [Privacy](#)



© 2022 - OnlineAdz - [Privacy](#)



