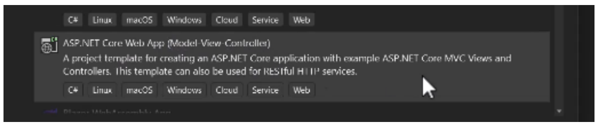
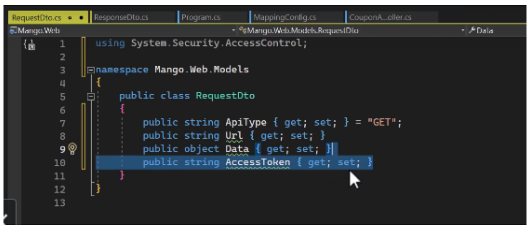
Section 3- Coupons API Part 2

- **Setting Up the Project**
  - Creating the Web Project
    - Folder for Front-End Application
    - Adding a New Project
    - Choosing ASP.NET Core Web App with MVC



ASP.NET Core Web App (Model-View-Controller)
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.

    - Naming the Project: Mango.Web
    - No Authentication Type
  - Consuming the Coupon API
    - Types of Requests: GET, POST, PUT, DELETE
    - URL and Data Specification
    - Creating Request and Response DTOs
      - Copying Response DTO from Coupon API
      - Changing Namespace to Mango.Web.Models
      - Creating Request DTO

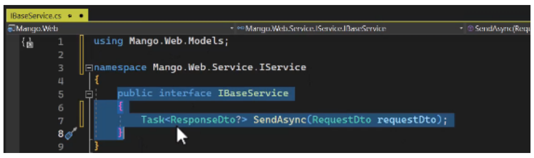


```
using System.Security.AccessControl;

namespace Mango.Web.Models
{
    public class RequestDto
    {
        public string ApiType { get; set; } = "GET";
        public string Url { get; set; }
        public object Data { get; set; }
        public string AccessToken { get; set; }
    }
}
```

        - API Type (Enum)
        - URL
        - Data
        - Access Token
- **Creating Enums and Utility Folder**
  - Enum for API Type
    - Creating a New Folder: Utility
    - Adding Enum: APIType (GET, POST, PUT, DELETE)

Creating Service Interface

- Interface for Base Service
  - Creating Folder: Service
  - Adding Interface: IBaseService

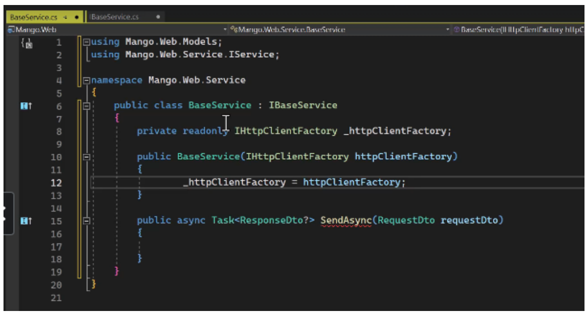


```
using Mango.Web.Models;

namespace Mango.Web.Service.IService
{
    public interface IBaseService
    {
        Task<ResponseDto?> SendAsync(RequestDto requestDto);
    }
}
```

  - Method: SendAsync
    - Parameter: Request DTO
    - Return Type: Task<ResponseDTO>

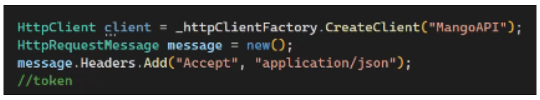
Implementing Base Service

- Class for Base Service
  - Implementing IBaseService
  - Using HTTP Client Factory


BaseService.cs
IBaseService.cs
Mango.Web
Mango.Web.Service.BaseService
BaseService(IHttpClientFactory httpC

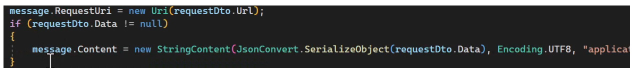
```
using Mango.Web.Models;
using Mango.Web.Service.IService;

namespace Mango.Web.Service
{
    public class BaseService : IBaseService
    {
        private readonly IHttpClientFactory _httpClientFactory;

        public BaseService(IHttpClientFactory httpClientFactory)
        {
            _httpClientFactory = httpClientFactory;
        }

        public async Task<ResponseDto?> SendAsync(RequestDto requestDto)
        {

        }
    }
}
```

  - Configuring HTTP Client
  - Setting Message Headers



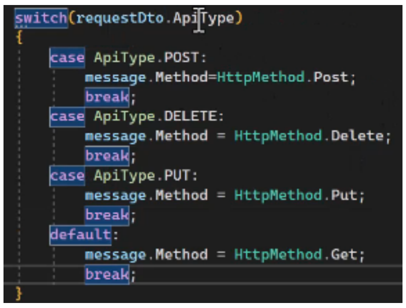
```
HttpClient client = _httpClientFactory.CreateClient("MangoAPI");
HttpRequestMessage message = new();
message.Headers.Add("Accept", "application/json");
//token
```

  - Serializing Data for POST/PUT Requests



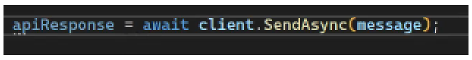
```
message.RequestUri = new Uri(requestDto.Url);
if (requestDto.Data != null)
{
    message.Content = new StringContent(JsonConvert.SerializeObject(requestDto.Data), Encoding.UTF8, "applica
}
```

- Handling Different API Types



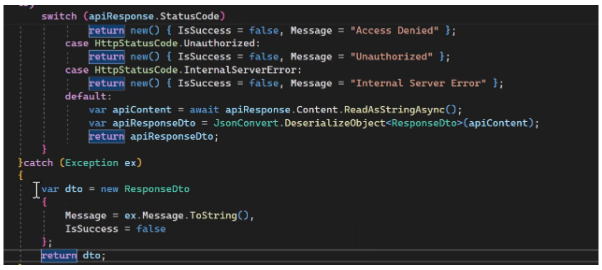
```
switch(requestDto.ApiType)
{
    case ApiType.POST:
        message.Method=HttpMethod.Post;
        break;
    case ApiType.DELETE:
        message.Method = HttpMethod.Delete;
        break;
    case ApiType.PUT:
        message.Method = HttpMethod.Put;
        break;
    default:
        message.Method = HttpMethod.Get;
        break;
}
```

- Sending Async Request



```
apiResponse = await client.SendAsync(message);
```

- Deserializing Response



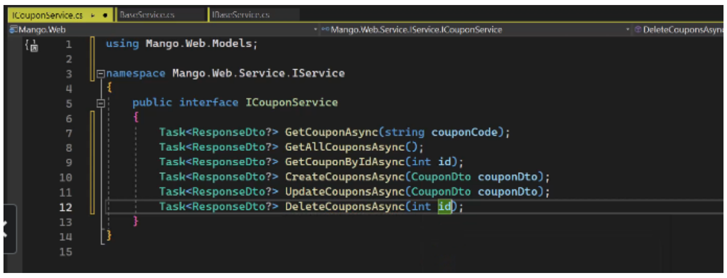
```
    switch (apiResponse.StatusCode)
            return new() { IsSuccess = false, Message = "Access Denied" };
        case HttpStatusCode.Unauthorized:
            return new() { IsSuccess = false, Message = "Unauthorized" };
        case HttpStatusCode.InternalServerError:
            return new() { IsSuccess = false, Message = "Internal Server Error" };
        default:
            var apiContent = await apiResponse.Content.ReadAsStringAsync();
            var apiResponseDto = JsonConvert.DeserializeObject<ResponseDto>(apiContent);
            return apiResponseDto;
    }
}catch (Exception ex)
{
    var dto = new ResponseDto
    {
        Message = ex.Message.ToString(),
        IsSuccess = false
    };
    return dto;
```

- Error Handling

Creating Coupon Service

Interface for Coupon Service

- Creating Folder: IService
- Adding Interface: ICouponService
- Methods: GetCoupon, GetAllCoupons, GetCouponById, CreateCoupon, UpdateCoupon, DeleteCoupon

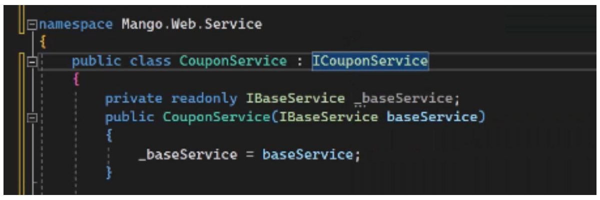


```
using Mango.Web.Models;

namespace Mango.Web.Service.IService
{
    public interface ICouponService
    {
        Task<ResponseDto?> GetCouponAsync(string couponCode);
        Task<ResponseDto?> GetAllCouponsAsync();
        Task<ResponseDto?> GetCouponByIdAsync(int id);
        Task<ResponseDto?> CreateCouponsAsync(CouponDto couponDto);
        Task<ResponseDto?> UpdateCouponsAsync(CouponDto couponDto);
        Task<ResponseDto?> DeleteCouponsAsync(int id);
    }
}
```

Implementing Coupon Service

- Class for Coupon Service

- Dependency Injection for Base Service



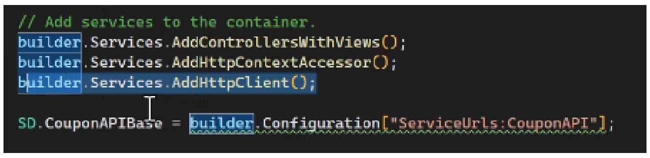
```csharp
namespace Mango.Web.Service
{
    public class CouponService : ICouponService
    {
        private readonly IBaseService _baseService;
        public CouponService(IBaseService baseService)
        {
            _baseService = baseService;
        }
```

- Implementing Interface Methods

Configuring Dependency Injection

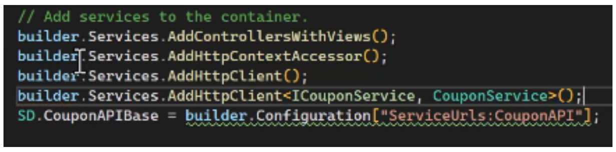
Registering Services in Program.cs

- Adding HTTP Client Factory



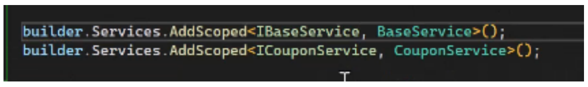
```csharp
// Add services to the container.
builder.Services.AddControllersWithViews();
builder.Services.AddHttpContextAccessor();
builder.Services.AddHttpClient();

SD.CouponAPIBase = builder.Configuration["ServiceUrls:CouponAPI"];
```

- Configuring Coupon Service



```csharp
// Add services to the container.
builder.Services.AddControllersWithViews();
builder.Services.AddHttpContextAccessor();
builder.Services.AddHttpClient();
builder.Services.AddHttpClient<ICouponService, CouponService>();
SD.CouponAPIBase = builder.Configuration["ServiceUrls:CouponAPI"];
```

- Registering Base Service and Coupon Service

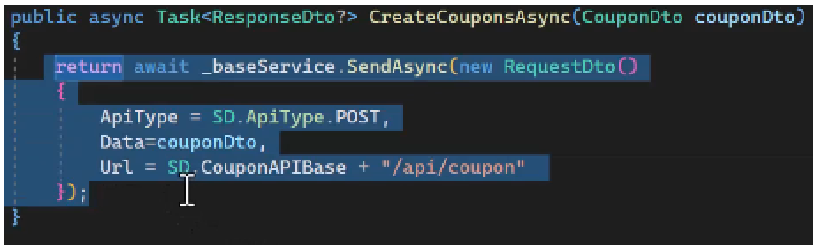


```csharp
builder.Services.AddScoped<IBaseService, BaseService>();
builder.Services.AddScoped<ICouponService, CouponService>();
```

Implementing the Coupon Service

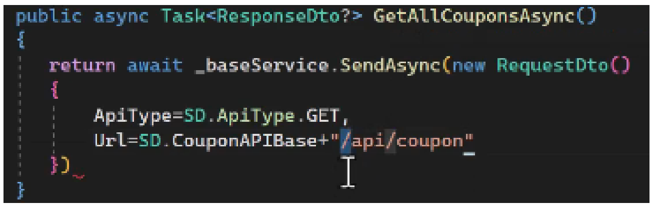
Methods in Coupon Service

- Create Coupon



```csharp
public async Task<ResponseDto?> CreateCouponsAsync(CouponDto couponDto)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.POST,
        Data=couponDto,
        Url = SD.CouponAPIBase + "/api/coupon"
    });
}
```

- Get All Coupons
  - Adding Asynchronous Keyword
  - Configuring Request DTO

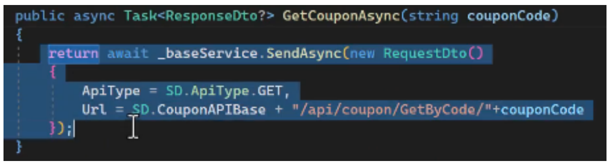


```csharp
public async Task<ResponseDto?> GetAllCouponsAsync()
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType=SD.ApiType.GET,
        Url=SD.CouponAPIBase+"/api/coupon"
    });
}
```

- API Type: GET
- URL: Coupon API Base + API/Coupon
- Adding Forward Slash in App Settings

- Awaiting Base Service SendAsync

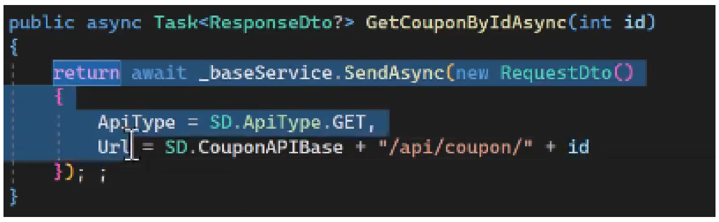
- Get Coupon By Code



```
public async Task<ResponseDto?> GetCouponAsync(string couponCode)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.GET,
        Url = SD.CouponAPIBase + "/api/coupon/GetByCode/"+couponCode
    });
}
```

  - Passing Coupon Code
  - Custom Name: GetByCode

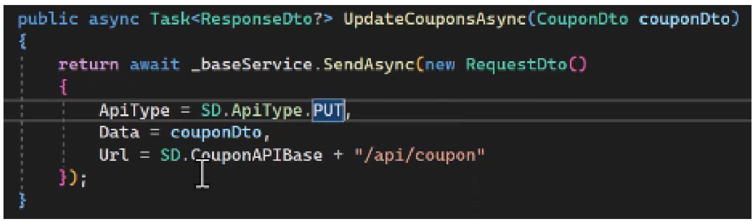
- Get Coupon By ID



```
public async Task<ResponseDto?> GetCouponByIdAsync(int id)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.GET,
        Url = SD.CouponAPIBase + "/api/coupon/" + id
    }); ;
}
```

  - Passing Coupon ID

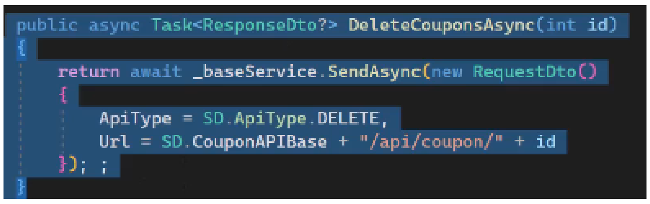
- Update Coupon



```
public async Task<ResponseDto?> UpdateCouponsAsync(CouponDto couponDto)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.PUT,
        Data = couponDto,
        Url = SD.CouponAPIBase + "/api/coupon"
    });
}
```

  - API Type: PUT
  - Data: CouponDTO

- Delete Coupon

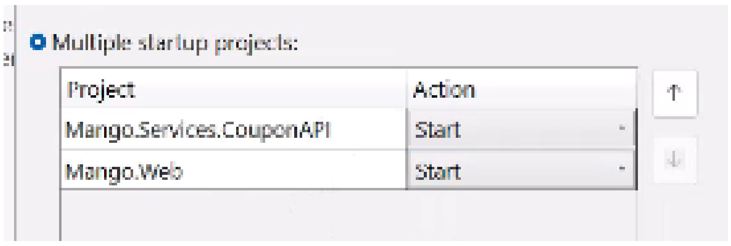


```
public async Task<ResponseDto?> DeleteCouponsAsync(int id)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.DELETE,
        Url = SD.CouponAPIBase + "/api/coupon/" + id
    }); ;
}
```

  - API Type: DELETE
  - URL: Coupon API Base + API/Coupon + ID

- CRUD Operations
  - Implementing All Endpoints

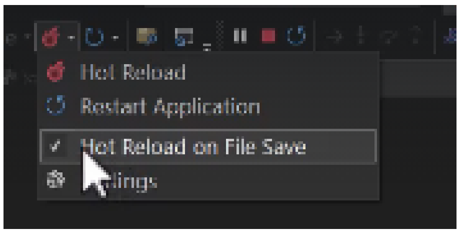
- Running Multiple Projects



Multiple startup projects:

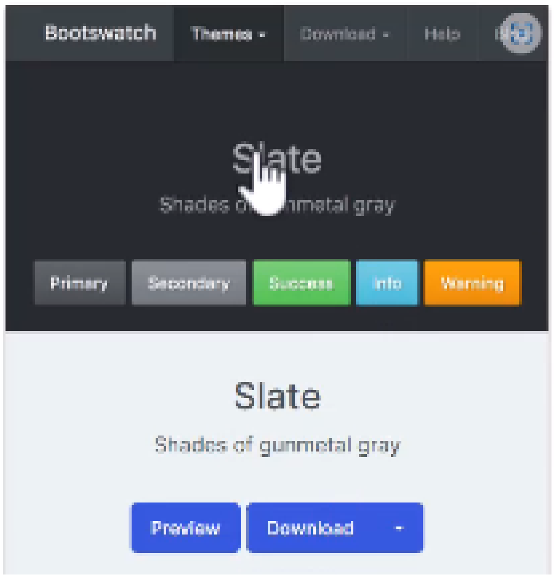
Project	Action
Mango.Services.CouponAPI	Start
Mango.Web	Start

  - Configuring Startup Projects
  - Enabling Hot Reload



Configuring the Front-End Application

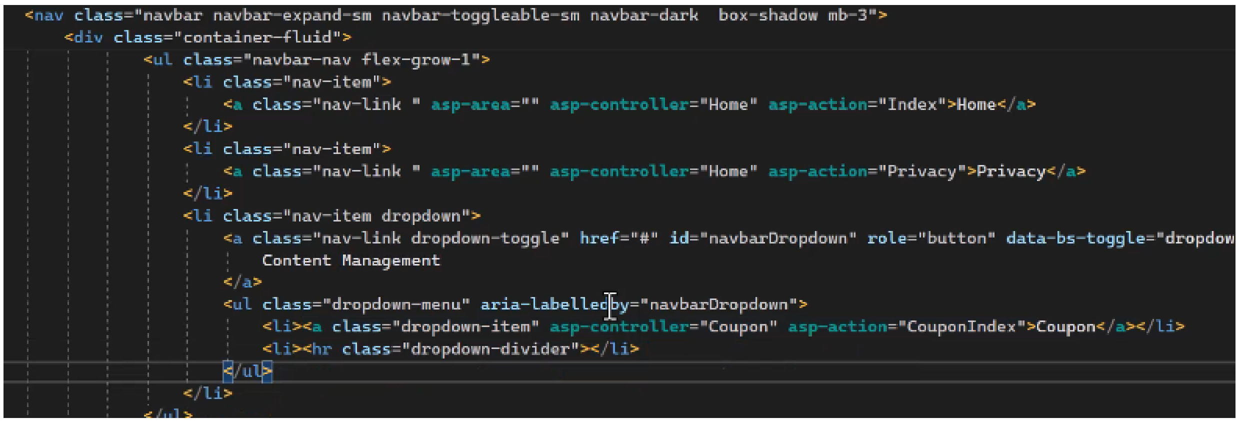
- Changing Bootstrap Theme vis bootwatch.com
  - Using Bootswatch Theme: Slate



  - Overwriting Bootstrap CSS
  - Updating _Layout to Use Non-Minified CSS
- Customizing Header and Footer
  - Adding Dropdown to Navbar
  - Removing Custom CSS for Anchor Tags
  - Making Header Darker
  - Modifying Footer

- Removing Border Top
- Adding "Made with Love by Dotnet Mastery"

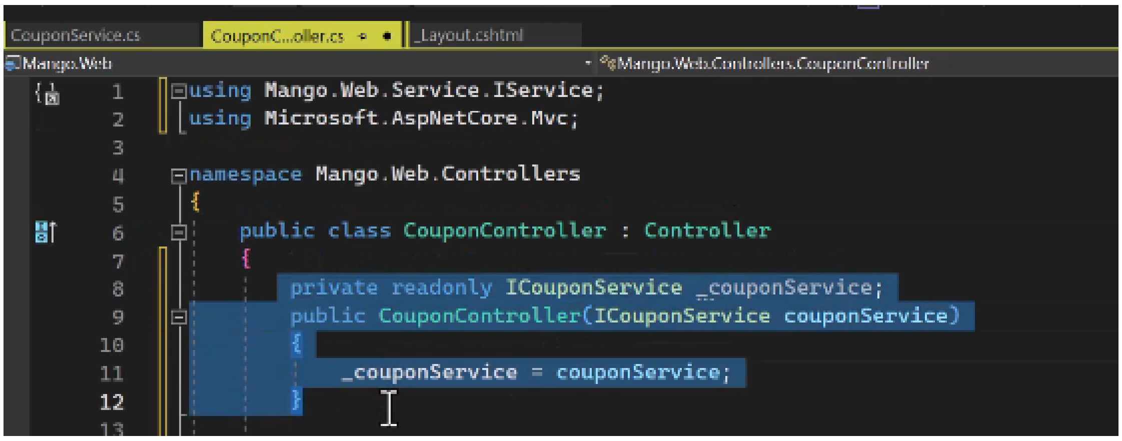
- Adding Bootstrap Icons
  - Including Bootstrap Icons CDN
  - Adding Heart Icon to Footer
- Link Coupon Controller to Nav



```
<nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-dark  box-shadow mb-3">
    <div class="container-fluid">
            <ul class="navbar-nav flex-grow-1">
                <li class="nav-item">
                    <a class="nav-link " asp-area="" asp-controller="Home" asp-action="Index">Home</a>
                </li>
                <li class="nav-item">
                    <a class="nav-link " asp-area="" asp-controller="Home" asp-action="Privacy">Privacy</a>
                </li>
                <li class="nav-item dropdown">
                    <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-bs-toggle="dropdow
                        Content Management
                    </a>
                    <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
                        <li><a class="dropdown-item" asp-controller="Coupon" asp-action="CouponIndex">Coupon</a></li>
                        <li><hr class="dropdown-divider"></li>
                    </ul>
                </li>
            </ul>
```

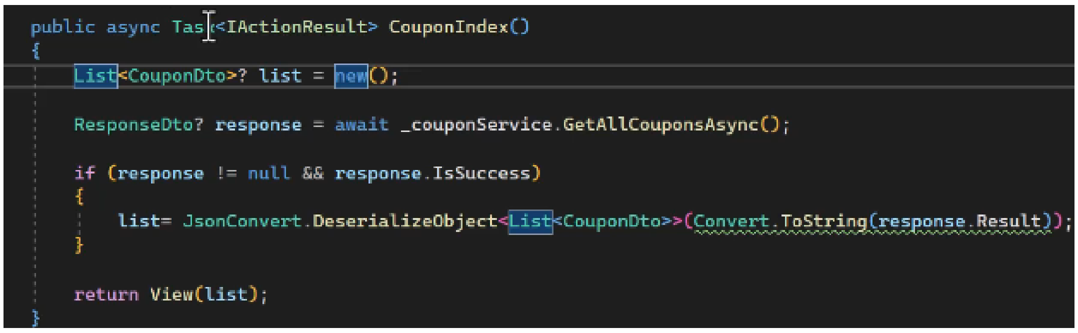
Creating Coupon Controller

- Adding Controller
  - Creating Folder: Controllers
  - Adding CouponController
  - Dependency Injection for Coupon Service



```
using Mango.Web.Service.IService;
using Microsoft.AspNetCore.Mvc;

namespace Mango.Web.Controllers
{
    public class CouponController : Controller
    {
        private readonly ICouponService _couponService;
        public CouponController(ICouponService couponService)
        {
            _couponService = couponService;
        }
```

- Index Action Method

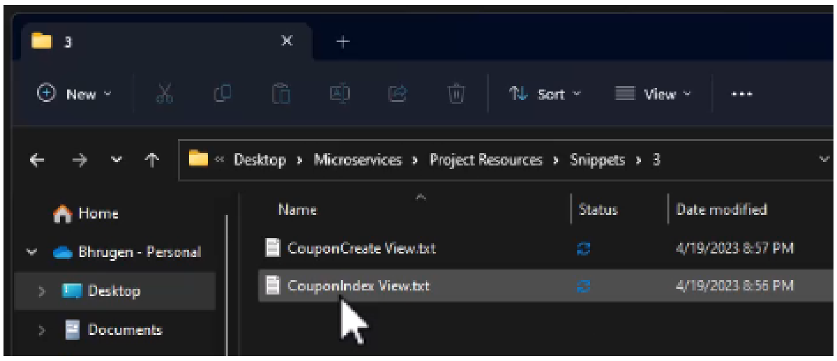


```
public async Task<IActionResult> CouponIndex()
{
    List<CouponDto>? list = new();

    ResponseDto? response = await _couponService.GetAllCouponsAsync();

    if (response != null && response.IsSuccess)
    {
        list= JsonConvert.DeserializeObject<List<CouponDto>>(Convert.ToString(response.Result));
    }

    return View(list);
}
```

  - Displaying All Coupons
  - Deserializing Response
  - Returning View with List of CouponDTO

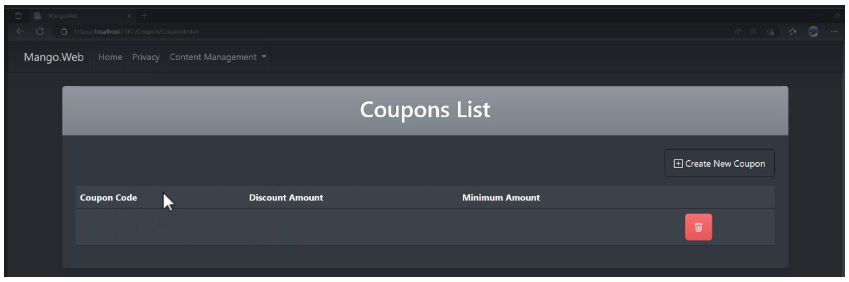
Creating Views

- Coupon Index View
  - Creating View for Coupon Index



Name	Status	Date modified
CouponCreate View.txt		4/19/2023 8:57 PM
CouponIndex View.txt		4/19/2023 8:56 PM

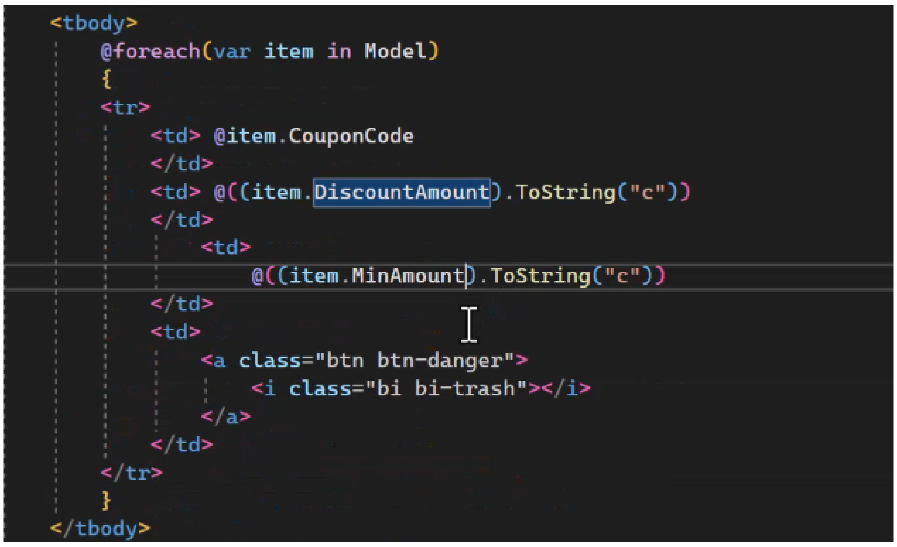
  - Displaying Coupons in a Table



Coupons List

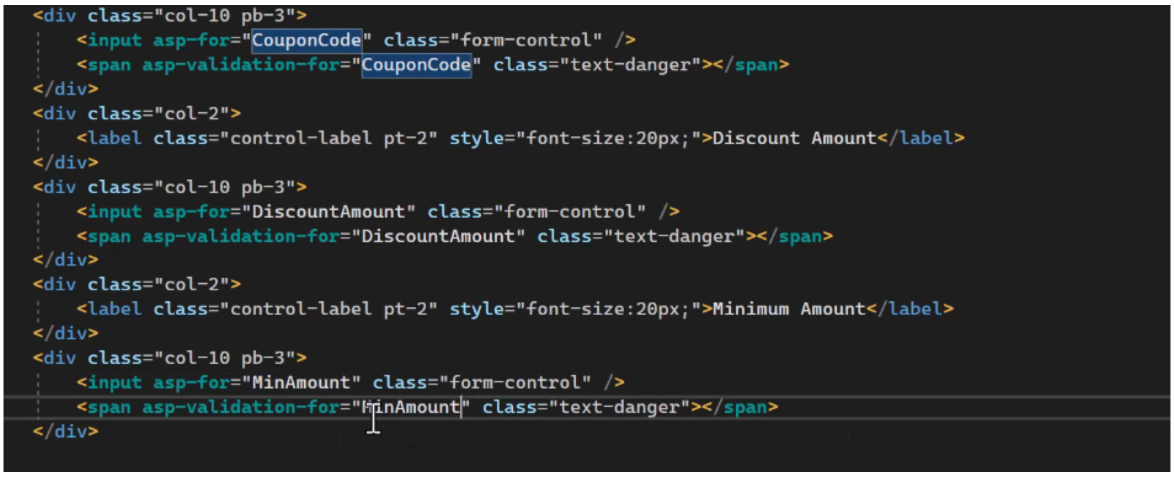
Coupon Code	Discount Amount	Minimum Amount	

- Adding ForEach Loop to Iterate Through Model



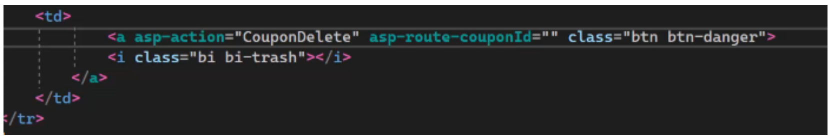
```
<tbody>
    @foreach(var item in Model)
    {
    <tr>
        <td> @item.CouponCode
        </td>
        <td> @((item.DiscountAmount).ToString("c"))
        </td>
            <td>
                @((item.MinAmount).ToString("c"))
        </td>
        <td>
            <a class="btn btn-danger">
                <i class="bi bi-trash"></i>
            </a>
        </td>
    </tr>
    }
</tbody>
```

- Coupon Create View
  - Creating View for Coupon Create
  - Adding Form for Coupon Details
  - Client-Side Validation



```
<div class="col-10 pb-3">
    <input asp-for="CouponCode" class="form-control" />
    <span asp-validation-for="CouponCode" class="text-danger"></span>
</div>
<div class="col-2">
    <label class="control-label pt-2" style="font-size:20px;">Discount Amount</label>
</div>
<div class="col-10 pb-3">
    <input asp-for="DiscountAmount" class="form-control" />
    <span asp-validation-for="DiscountAmount" class="text-danger"></span>
</div>
<div class="col-2">
    <label class="control-label pt-2" style="font-size:20px;">Minimum Amount</label>
</div>
<div class="col-10 pb-3">
    <input asp-for="MinAmount" class="form-control" />
    <span asp-validation-for="MinAmount" class="text-danger"></span>
</div>
```

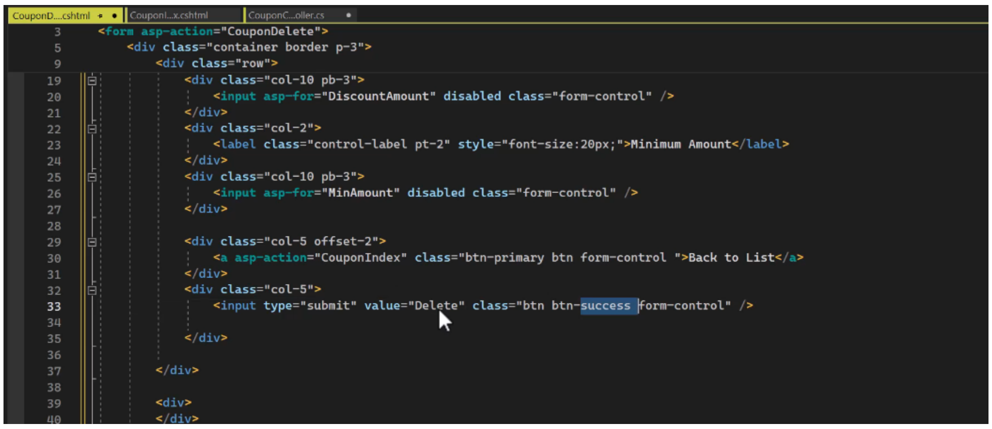
- Coupon Delete View
  - Creating View for Coupon Delete



```
    <td>
            <a asp-action="CouponDelete" asp-route-couponId="" class="btn btn-danger">
            <i class="bi bi-trash"></i>
        </a>
    </td>
</tr>
```

  - Displaying Coupon Details

- Disabling Fields



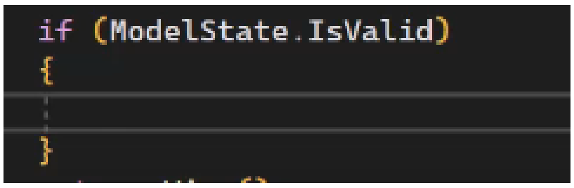
CouponD....cshtml | CouponI...x.cshtml | CouponC...oller.cs

```
<form asp-action="CouponDelete">
    <div class="container border p-3">
        <div class="row">
            <div class="col-10 pb-3">
                <input asp-for="DiscountAmount" disabled class="form-control" />
            </div>
            <div class="col-2">
                <label class="control-label pt-2" style="font-size:20px;">Minimum Amount</label>
            </div>
            <div class="col-10 pb-3">
                <input asp-for="MinAmount" disabled class="form-control" />
            </div>

            <div class="col-5 offset-2">
                <a asp-action="CouponIndex" class="btn-primary btn form-control ">Back to List</a>
            </div>
            <div class="col-5">
                <input type="submit" value="Delete" class="btn btn-success form-control" />

            </div>

        </div>

        <div>
        </div>
```

Implementing CRUD Operations

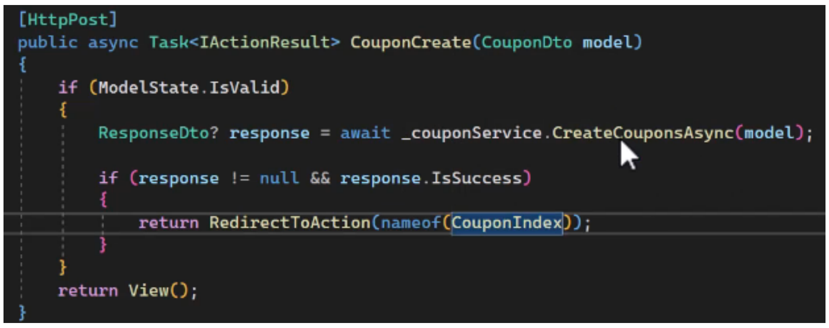
Create Coupon

- Post Action Method for Coupon Create
- Server-Side Validation



```
if (ModelState.IsValid)
{

}
```

- Invoking CreateCouponAsync

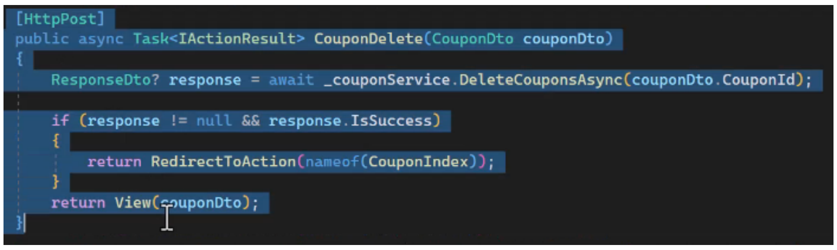


```
[HttpPost]
public async Task<IActionResult> CouponCreate(CouponDto model)
{
    if (ModelState.IsValid)
    {
        ResponseDto? response = await _couponService.CreateCouponsAsync(model);

        if (response != null && response.IsSuccess)
        {
            return RedirectToAction(nameof(CouponIndex));
        }
    }
    return View();
}
```

- Redirecting to Index Action

Delete Coupon

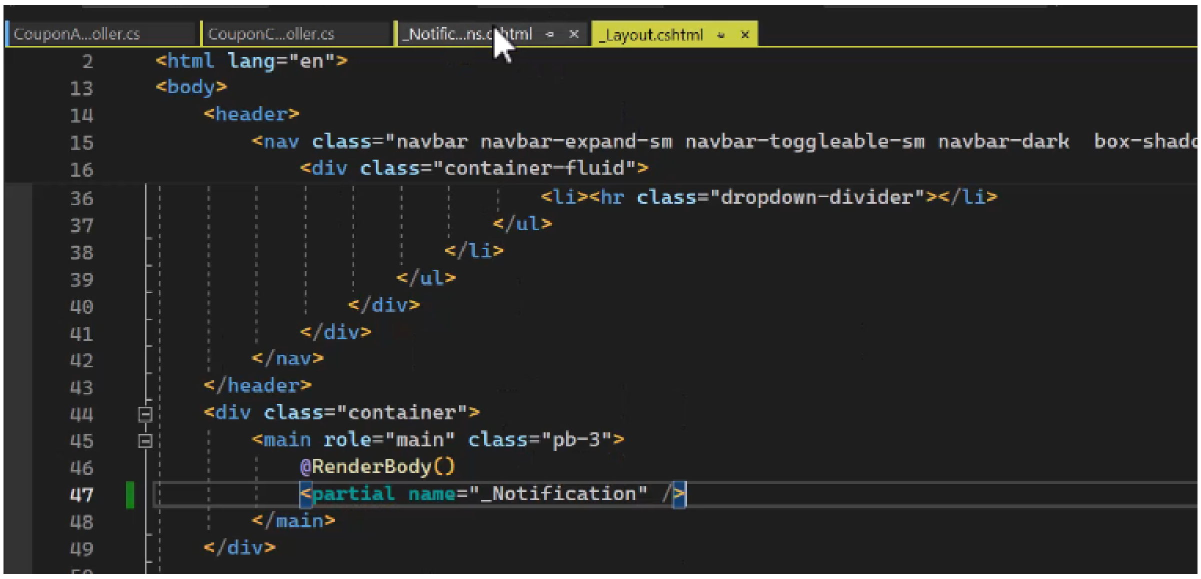
- Post Action Method for Coupon Delete
- Invoking DeleteCouponAsync



```
[HttpPost]
public async Task<IActionResult> CouponDelete(CouponDto couponDto)
{
    ResponseDto? response = await _couponService.DeleteCouponsAsync(couponDto.CouponId);

    if (response != null && response.IsSuccess)
    {
        return RedirectToAction(nameof(CouponIndex));
    }
    return View(couponDto);
}
```

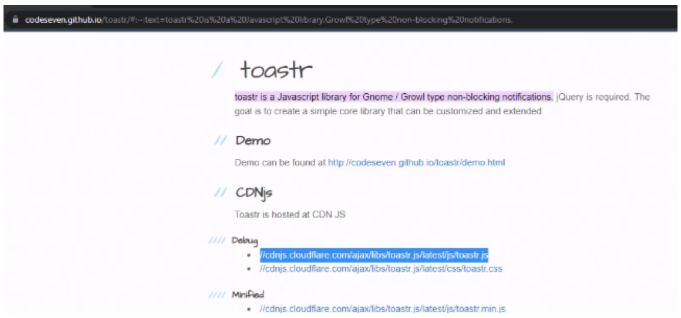
- Redirecting to Index Action

Configuring Toaster Notifications



```html
<html lang="en">
<body>
    <header>
        <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-dark  box-shado
            <div class="container-fluid">
                                        <li><hr class="dropdown-divider"></li>
                                    </ul>
                                </li>
                            </ul>
                        </div>
                    </div>
                </nav>
    </header>
    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
            <partial name="_Notification" />
        </main>
    </div>
```

Adding Toaster CDN



codeseven.github.io/toastr/#:~:text=toastr%20is%20a%20Javascript%20library.Growl%20type%20non-blocking%20notifications.

/ toastr

toastr is a Javascript library for Gnome / Growl type non-blocking notifications. jQuery is required. The goal is to create a simple core library that can be customized and extended.

// Demo

Demo can be found at http://codeseven.github.io/toastr/demo.html

// CDNjs

Toastr is hosted at CDN JS

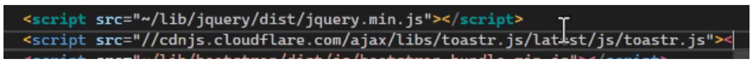
//// Debug

- //cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/js/toastr.js
- //cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.css

//// Minified

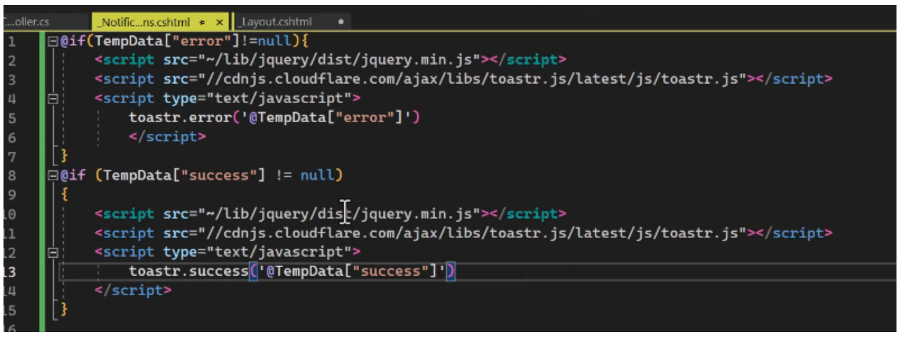
- //cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/js/toastr.min.js

- Adding Script and Stylesheet in _Layout



```html
<script src="~/lib/jquery/dist/jquery.min.js"></script>
<script src="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/js/toastr.js"><
```

Creating Partial View for Notifications

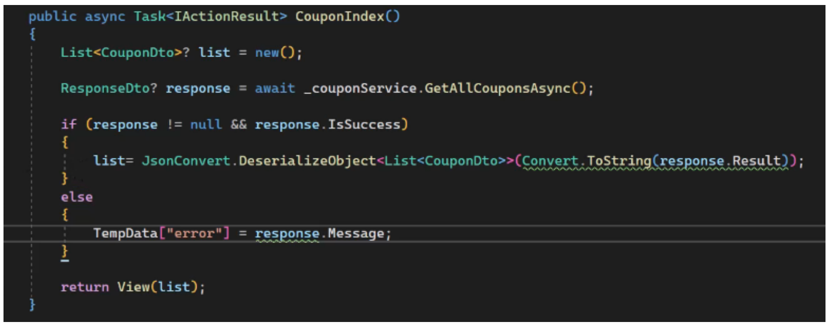


```html
@if(TempData["error"]!=null){
    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/js/toastr.js"></script>
    <script type="text/javascript">
        toastr.error('@TempData["error"]')
        </script>
}
@if (TempData["success"] != null)
{
    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="//cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/js/toastr.js"></script>
    <script type="text/javascript">
        toastr.success('@TempData["success"]')
    </script>
}
```

- Displaying Toaster Notifications for Success and Error

Setting TempData for Notifications

- Adding TempData in Coupon Controller



```
public async Task<IActionResult> CouponIndex()
{
    List<CouponDto>? list = new();

    ResponseDto? response = await _couponService.GetAllCouponsAsync();

    if (response != null && response.IsSuccess)
    {
        list= JsonConvert.DeserializeObject<List<CouponDto>>(Convert.ToString(response.Result));
    }
    else
    {
        TempData["error"] = response.Message;
    }

    return View(list);
}
```

Debugging and Testing

- Debugging Issues
  - Adding Debugging Points
  - Examining Responses
- Testing Functionality
  - Running the Application
  - Verifying CRUD Operations
  - Displaying Toaster Notifications