

Ingredients Implementation

GROUP 4

Mustafa

Eric

Cao

MODELS:

1. **Ingredient**
2. **IIngredientRepo**
3. **EIIngredientRepo**
4. **RecipeViewModel**
5. **Updated Recipe**
6. **Updated ApplicationDbContext**
7. **Updated SeedData**

Ingredient:

Included RecipeID to associate with Recipe

The screenshot displays the Visual Studio IDE with the `Ingredient.cs` file open in the editor. The file is part of the `RecipeBook.Models` namespace and defines the `Ingredient` class. The class includes properties for `IngredientID`, `RecipeID`, `Name`, `Unit`, and `Amount`, each with appropriate data annotations for validation. The `Solution Explorer` on the right shows the project structure, including the `Ingredient` folder and its associated views.

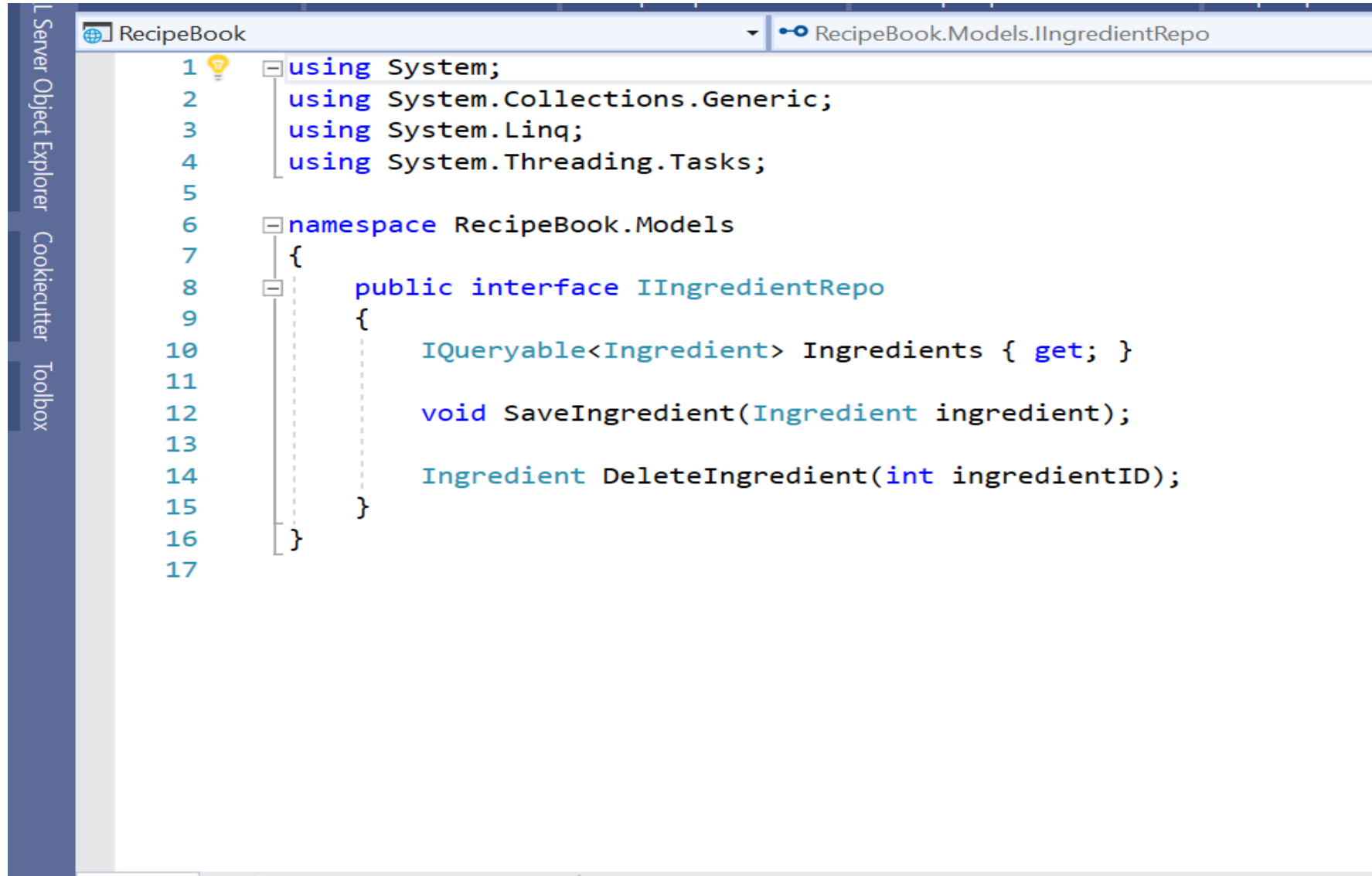
```
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel.DataAnnotations;
4 using System.Linq;
5 using System.Threading.Tasks;
6
7 namespace RecipeBook.Models
8 {
9     public class Ingredient
10    {
11        public int IngredientID { get; set; }
12        public int RecipeID { get; set; }
13        [Required(ErrorMessage = "Please enter valid name")]
14        public string Name { get; set; }
15        [Required(ErrorMessage = "Please enter valid Unit of Measure")]
16        public string Unit { get; set; }
17        [Required]
18        [Range(0.25, double.MaxValue,
19            ErrorMessage = "Please enter valid Amount")]
20        public double Amount { get; set; }
21    }
22 }
23
24
```

Solution Explorer:

- RecipeBook
 - RecipeBook.Models
 - Ingredient
 - IngAddToRecipe.csh
 - IngEdit.cshhtml
 - IngList.cshhtml
 - Recipe
 - Create.cshhtml
 - Edit.cshhtml
 - Index.cshhtml
 - List.cshhtml
 - ReviewRecipe.cshhtml
 - View.cshhtml

Bottom Bar: 112 % | No issues found | Ln: 21 Ch: 1 SPC CRLF

IIngredientRepo:



The screenshot shows the Visual Studio IDE with the 'RecipeBook' project selected in the Solution Explorer. The file 'RecipeBook.Models.IIngredientRepo.cs' is open in the editor. The code defines the 'IIngredientRepo' interface within the 'RecipeBook.Models' namespace. The interface includes three methods: 'Ingredients' (returning 'IEnumerable<Ingredient>'), 'SaveIngredient' (returning 'void'), and 'DeleteIngredient' (returning 'Ingredient').

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Threading.Tasks;
5
6  namespace RecipeBook.Models
7  {
8      public interface IIngredientRepo
9      {
10         IQueryable<Ingredient> Ingredients { get; }
11
12         void SaveIngredient(Ingredient ingredient);
13
14         Ingredient DeleteIngredient(int ingredientID);
15     }
16 }
17
```

EFIngredientRepo:

```
RecipeBook
RecipeBook.Models.EFIngredientRepo

7 {
8   public class EFIngredientRepo : IIngredientRepo
9   {
10    private ApplicationDbContext context;
11
12    public EFIngredientRepo(ApplicationDbContext ctx)
13    {
14        context = ctx;
15    }
16
17    public IQueryable<Ingredient> Ingredients => context.Ingredients;
18
19    public void SaveIngredient(Ingredient ingredient)
20    {
21        if (ingredient.IngredientID == 0)
22        {
23            context.Ingredients.Add(ingredient);
24        }
25        else
26        {
27            Ingredient dbEntry = context.Ingredients
28                .FirstOrDefault(r => r.IngredientID == ingredient.IngredientID);
29            if (dbEntry != null)
30            {
31                dbEntry.Name = ingredient.Name;
32                dbEntry.Unit = ingredient.Unit;
33                dbEntry.Amount = ingredient.Amount;
34            }
35        }
36        context.SaveChanges();
37    }
38 }
```

```
RecipeBook
RecipeBook.Models.EFIngredientRepo

38
39 public Ingredient DeleteIngredient(int ingredientID)
40 {
41     Ingredient dbEntry = context.Ingredients
42         .FirstOrDefault(r => r.IngredientID == ingredientID);
43     if (dbEntry != null)
44     {
45         context.Ingredients.Remove(dbEntry);
46         context.SaveChanges();
47     }
48     return dbEntry;
49 }
50
51
52 }
```

RecipeViewModel:

The image shows a Visual Studio IDE with the `RecipeBook` project open. The `RecipeBook.Models.ViewModels.RecipeViewModel` file is selected, showing the following C# code:

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Threading.Tasks;
5
6 namespace RecipeBook.Models.ViewModels
7 {
8     public class RecipeViewModel
9     {
10         public Recipe Recipe { get; set; }
11         public List<Ingredient> Ingredients { get; set; }
12
13         public RecipeViewModel()
14         {
15             Recipe = new Recipe();
16             Ingredients = new List<Ingredient>();
17         }
18     }
19 }
20
21
```

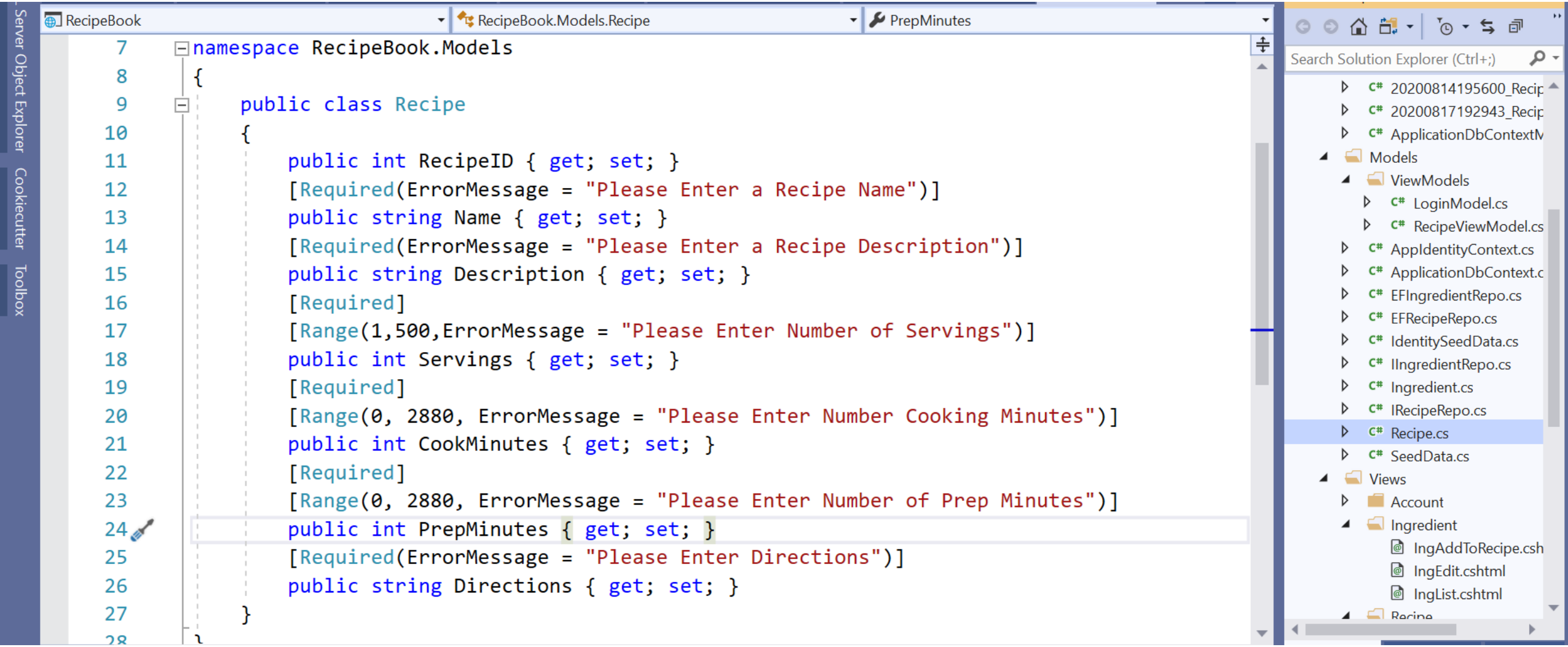
The Solution Explorer on the right shows the project structure:

- 20200814195600_Recip
- 20200817192943_Recip
- ApplicationDbContextV
- Models
 - ViewModels
 - LoginModel.cs
 - RecipeViewModel.cs
- AppIdentityContext.cs
- ApplicationDbContext.c
- EFIngredientRepo.cs
- EFRecipeRepo.cs
- IdentitySeedData.cs
- IIngredientRepo.cs
- Ingredient.cs
- IRecipeRepo.cs
- Recipe.cs
- SeedData.cs
- Views
 - Account
 - Ingredient
 - IngAddToRecipe.csh
 - IngEdit.cshtml
 - IngList.cshtml
 - Recipe

The status bar at the bottom indicates 135% zoom, no issues found, and the file is at line 18, column 9.

Recipe Update:

Removed string Ingredient Property



The screenshot displays the Visual Studio IDE with the RecipeBook.Models.Recipe class open. The class contains several properties with data annotations. The PrepMinutes property is highlighted with a blue selection box. The Solution Explorer on the right shows the project structure, with the Recipe.cs file selected under the Models folder.

```
7 namespace RecipeBook.Models
8 {
9     public class Recipe
10    {
11        public int RecipeID { get; set; }
12        [Required(ErrorMessage = "Please Enter a Recipe Name")]
13        public string Name { get; set; }
14        [Required(ErrorMessage = "Please Enter a Recipe Description")]
15        public string Description { get; set; }
16        [Required]
17        [Range(1,500,ErrorMessage = "Please Enter Number of Servings")]
18        public int Servings { get; set; }
19        [Required]
20        [Range(0, 2880, ErrorMessage = "Please Enter Number Cooking Minutes")]
21        public int CookMinutes { get; set; }
22        [Required]
23        [Range(0, 2880, ErrorMessage = "Please Enter Number of Prep Minutes")]
24        public int PrepMinutes { get; set; }
25        [Required(ErrorMessage = "Please Enter Directions")]
26        public string Directions { get; set; }
27    }
28 }
```

Solution Explorer Structure:

- 20200814195600_Recipe.cs
- 20200817192943_Recipe.cs
- ApplicationDbContext.cs
- Models
 - ViewModels
 - LoginModel.cs
 - RecipeViewModel.cs
 - AppIdentityContext.cs
 - ApplicationDbContext.cs
 - EFIngredientRepo.cs
 - EFRecipeRepo.cs
 - IdentitySeedData.cs
 - IIngredientRepo.cs
 - Ingredient.cs
 - IRecipeRepo.cs
 - Recipe.cs
 - SeedData.cs
- Views
 - Account
 - Ingredient
 - IngAddToRecipe.csh
 - IngEdit.cshhtml
 - IngList.cshhtml
 - Recipe

ApplicationDbContext Update:

Added DbSet for Ingredients

RecipeBook RecipeBook.Models.ApplicationDbContext Recipes

```
1  using Microsoft.EntityFrameworkCore;
2      using System;
3      using System.Collections.Generic;
4      using System.Linq;
5      using System.Threading.Tasks;
6
7  namespace RecipeBook.Models
8  {
9      public class ApplicationDbContext : DbContext
10     {
11         public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options) { }
12
13         public DbSet<Recipe> Recipes { get; set; }
14         public DbSet<Ingredient> Ingredients { get; set; }
15     }
16 }
17
```


SeedData Update:

Added SeedData for Ingredient Context and removed Ingredient SeedData from Recipe Context

```
49
50
RecipeBook
RecipeBook.Models.SeedData
EnsurePopulated
51
13 public static void EnsurePopulated(IApplicationBuilder app)
14 {
15     ApplicationDbContext context = app.ApplicationServices
16         .GetRequiredService<ApplicationDbContext>();
17     context.Database.Migrate();
18
19     if (!context.Recipes.Any())
20     {
21         context.Recipes.AddRange(
22             new Recipe
23             {
24                 Name = "Chicken Fajitas",
25                 Servings = 6,
26                 CookMinutes = 40,
27                 PrepMinutes = 20,
28                 Description = "Seasoned Grilled Chicken and Veg",
29                 Directions = "In a large bowl, whisk together 1/2 cup oil, lime juice, cumin, and red pepper flakes. Season chick
30             },
31             new Recipe
32             {
33                 Name = "Steak Fajitas",
34                 Servings = 6,
35                 CookMinutes = 40,
36                 PrepMinutes = 20,
37                 Description = "Seasoned Grilled Steak and Veg",
38                 Directions = "In a large bowl, whisk together 1/2 cup oil, lime juice, cumin, and red pepper flakes. Season steak
39             },
40             new Recipe
41             {
42                 Name = "Salmon Fajitas",
43                 Servings = 6,
44                 CookMinutes = 40,
45                 PrepMinutes = 20,
46                 Description = "Seasoned Grilled Salmon and Veg",
47                 Directions = "In a large bowl, whisk together 1/2 cup oil, lime juice, cumin, and red pepper flakes. Season salm
48             }
49         );
50     }
51     if (!context.Ingredients.Any())
52     {
53         context.Ingredients.AddRange(
54             new Ingredient { Name = "Olive Oil", Unit = "Cup", Amount = .5, RecipeID = 1 },
55             new Ingredient { Name = "Lime Juice", Unit = "Cup", Amount = .25, RecipeID = 1 },
56             new Ingredient { Name = "Cumin", Unit = "Tsp", Amount = 2, RecipeID = 1 },
57             new Ingredient { Name = "Boneless Salmon", Unit = "Lb", Amount = 1, RecipeID = 1 },
58             new Ingredient { Name = "Kosher Salt", Unit = "Pinch", Amount = 1, RecipeID = 1 },
59             new Ingredient { Name = "Red Pepper Flakes", Unit = "Tsp", Amount = .5, RecipeID = 1 },
60             new Ingredient { Name = "Black Pepper", Unit = "Pinch", Amount = 1, RecipeID = 1 },
61             new Ingredient { Name = "Bell Pepper, thinly sliced", Unit = "Whole", Amount = 2, RecipeID = 1 },
62             new Ingredient { Name = "Large Onion, thinly sliced", Unit = "Whole", Amount = 1, RecipeID = 1 },
63             new Ingredient { Name = "Tortillas", Unit = "Steamed", Amount = 12, RecipeID = 1 },
64             new Ingredient { Name = "Olive Oil", Unit = "Cup", Amount = .5, RecipeID = 2 },
65             new Ingredient { Name = "Lime Juice", Unit = "Cup", Amount = .25, RecipeID = 2 },
66             new Ingredient { Name = "Cumin", Unit = "Tsp", Amount = 2, RecipeID = 2 },
67             new Ingredient { Name = "Thinly Sliced Steak", Unit = "Lb", Amount = 1, RecipeID = 2 },
68             new Ingredient { Name = "Kosher Salt", Unit = "Pinch", Amount = 1, RecipeID = 2 },
69             new Ingredient { Name = "Red Pepper Flakes", Unit = "Tsp", Amount = .5, RecipeID = 2 },
70             new Ingredient { Name = "Black Pepper", Unit = "Pinch", Amount = 1, RecipeID = 2 },
71             new Ingredient { Name = "Bell Pepper, thinly sliced", Unit = "Whole", Amount = 2, RecipeID = 2 },
72             new Ingredient { Name = "Large Onion, thinly sliced", Unit = "Whole", Amount = 1, RecipeID = 2 },
73             new Ingredient { Name = "Tortillas", Unit = "Steamed", Amount = 12, RecipeID = 2 },
74             new Ingredient { Name = "Olive Oil", Unit = "Cup", Amount = .5, RecipeID = 3 },
75             new Ingredient { Name = "Lime Juice", Unit = "Cup", Amount = .25, RecipeID = 3 },
76             new Ingredient { Name = "Cumin", Unit = "Tsp", Amount = 2, RecipeID = 3 },
77             new Ingredient { Name = "Boneless Chicken Breast", Unit = "Lb", Amount = 1, RecipeID = 3 },
78             new Ingredient { Name = "Kosher Salt", Unit = "Pinch", Amount = 1, RecipeID = 3 },
79             new Ingredient { Name = "Red Pepper Flakes", Unit = "Tsp", Amount = .5, RecipeID = 3 },
80             new Ingredient { Name = "Black Pepper", Unit = "Pinch", Amount = 1, RecipeID = 3 },
81             new Ingredient { Name = "Bell Pepper, thinly sliced", Unit = "Whole", Amount = 2, RecipeID = 3 },
82             new Ingredient { Name = "Large Onion, thinly sliced", Unit = "Whole", Amount = 1, RecipeID = 3 },
83             new Ingredient { Name = "Tortillas", Unit = "Steamed", Amount = 12, RecipeID = 3 }
84         );
85     }
86     context.SaveChanges();
87 }
88
```

CONTROLLERS:

1. **IngredientController**
2. **RecipeController Update**

IngredientController:

Created CRUD controller for viewing Ingredient list, Deleting single Ingredient, Editing Ingredient and Adding Single Ingredient to a Recipe

```
RecipeBook | RecipeBook.Controllers.IngredientController | 43
11 [Authorize]
12 public class IngredientController : Controller
13 {
14     private IRecipeRepo repository;
15     private IIngredientRepo ingRepository;
16     public IngredientController(IRecipeRepo repo, IIngredientRepo ingRepo)
17     {
18         repository = repo;
19         ingRepository = ingRepo;
20     }
21
22     public IActionResult Index() => View("IngList", ingRepository.Ingredients.OrderBy(i => i.RecipeID));
23
24     public IActionResult IngEdit(int ingredientId)
25     {
26         ViewBag.InputType = "Edit";
27         return View(ingRepository.Ingredients.FirstOrDefault(i => i.IngredientID == ingredientId));
28     }
29
30     [HttpPost]
31     public IActionResult IngEdit(Ingredient ingredient)
32     {
33         if (ModelState.IsValid)
34         {
35             ingRepository.SaveIngredient(ingredient);
36             TempData["message"] = $"{ingredient.Name} has been updated in your RecipeBook";
37             return RedirectToAction("Index");
38         }
39         else
40         {
41             return View(ingredient);
42         }
43     }
44
45     public IActionResult IngAddToRecipe(int RecipeID) => View();
46     [HttpPost]
47     public IActionResult IngAddToRecipe(Ingredient ingredient, int RecipeID)
48     {
49         ingredient.RecipeID = RecipeID;
50         Recipe recipe = repository.Recipes.FirstOrDefault(r => r.RecipeID == ingredient.RecipeID);
51         ingRepository.SaveIngredient(ingredient);
52         TempData["message"] = $"{ingredient.Name} has been added to your Recipe";
53         return RedirectToAction("View", "Recipe", recipe);
54     }
55
56     [HttpPost]
57     public IActionResult Delete(int ingredientId)
58     {
59         Ingredient deletedIngredient = ingRepository.DeleteIngredient(ingredientId);
60         if (deletedIngredient != null)
61         {
62             TempData["message"] = $"{deletedIngredient.Name} was deleted from your RecipeBook";
63         }
64         return RedirectToAction("Index");
65     }
66
67     public IActionResult Create(int ingredientId)
68     {
69         ViewBag.InputType = "Add New";
70         return View("IngEdit", new Ingredient());
71     }
72 }
73
74
```

RecipeController Update:

```
[Authorize]
public class RecipeController : Controller
{
    private IRecipeRepo repository;
    private IIngredientRepo ingRepository;
    public RecipeController(IRecipeRepo repo, IIngredientRepo ingRepo)
    {
        repository = repo;
        ingRepository = ingRepo;
    }

    public ViewResult View(int recipeId)
    {
        RecipeViewModel vm = new RecipeViewModel();
        vm.Recipe = repository.Recipes.FirstOrDefault(r => r.RecipeID == recipeId);
        vm.Ingredients = ingRepository.Ingredients.Where(i => i.RecipeID == recipeId)
            .OrderBy(i => i.Name).ToList();

        return View(vm);
    }

    RecipeViewModel recipeViewModel = new RecipeViewModel();
    recipeViewModel.Recipe = repository.Recipes.FirstOrDefault(r => r.RecipeID == recipeId);
    recipeViewModel.Ingredients = ingRepository.Ingredients.Where(i => i.RecipeID == recipeId).ToList();
    ViewBag.InputType = "Edit";
    return View(recipeViewModel);
}

[HttpPost]
public IActionResult Edit(RecipeViewModel recipeViewModel)
{
    if (ModelState.IsValid)
    {
        repository.AddRecipe(recipeViewModel.Recipe);
        foreach (Ingredient i in recipeViewModel.Ingredients)
        {
            ingRepository.SaveIngredient(i);
        }
        TempData["message"] = $"{recipeViewModel.Recipe.Name} has been updated in your RecipeBook";
        return RedirectToAction("List");
    }
    else
    {
        return View(recipeViewModel);
    }
}
```

Added IIngredientRepo access. Implemented RecipeViewModel in all Recipe CRUDs except list. Deleting Recipe will delete its Ingredients too.

```
[HttpPost]
public IActionResult Delete(int recipeId)
{
    Recipe deletedRecipe = repository.DeleteRecipe(recipeId);
    List<int> deletables = new List<int>();
    foreach (Ingredient ing in ingRepository.Ingredients.Where(i => i.RecipeID == recipeId))
    {
        deletables.Add(ing.IngredientID);
    }
    foreach (int ingID in deletables)
    {
        Ingredient deleteIngredient = ingRepository.DeleteIngredient(ingID);
    }
    if (deletedRecipe != null)
    {
        TempData["message"] = $"{deletedRecipe.Name} was deleted from your RecipeBook";
    }
    return RedirectToAction("List");
}

public ViewResult Create()
{
    RecipeViewModel recipeViewModel = new RecipeViewModel();
    for (int i = 0; i < 5; i++)
    {
        recipeViewModel.Ingredients.Add(new Ingredient { RecipeID=recipeViewModel.Recipe.RecipeID });
    }
    ViewBag.InputType = "Add New";
    return View(recipeViewModel);
}

[HttpPost]
public IActionResult Create(RecipeViewModel recipeViewModel)
{
    if (ModelState.IsValid)
    {
        repository.AddRecipe(recipeViewModel.Recipe);
        foreach (Ingredient i in recipeViewModel.Ingredients)
        {
            i.RecipeID = repository.Recipes.FirstOrDefault(r => r.Name == recipeViewModel.Recipe.Name).RecipeID;
            ingRepository.SaveIngredient(i);
        }
        TempData["message"] = $"{recipeViewModel.Recipe.Name} has been added to your RecipeBook";
        return RedirectToAction("List");
    }
    else
    {
        return View(recipeViewModel);
    }
}
```

VIEWS:

- 1. Ingredient/**
 - 1. ListIngredient**
 - 2. EditIngredient**
 - 3. AddIngredientToRecipe**
- 2. Recipe/**
 - 1. Updated CreateRecipe**
 - 2. Updated EditRecipe**
 - 3. Updated ViewRecipe**
- 3. Shared/**
 - 1. FullRecipeInput (Partial)**

Ingredient/List:

```
1  @model IQueryable<Ingredient>
2  @{
3      ViewData["Title"] = "IngList";
4  }
5
6  <div class="container">
7      <div class="row" id="h1Format">
8          <h1 class="label-info col-sm-8" id="h1Format"><b>Your Ingredient List</b></h1>
9          <a asp-action="Create" class="btn btn-success col-sm-3 addBtn glyphicon-plus">Add New Ingredient</a>
10     </div>
11
12     <table class="table table-sm table-striped table-bordered text-left bg-warning table-responsive">
13         <tr>
14             <th colspan="2">Ingredient Name</th>
15             <th colspan="3">Action</th>
16         </tr>
17
18         @foreach (var i in Model)
19         {
20             <tr class="listTbl">
21                 <td colspan="2"><a class="navbar-link">@i.RecipeID @i.Amount @i.Unit @i.Name</a></td>
22                 <td>
23                     <form asp-action="Delete" method="post">
24                         <a class="btn btn-default btn-warning center-block" asp-action="IngEdit" asp-route-IngredientID=@i.IngredientID>Edit</a>
25                         <input type="hidden" name="IngredientID" value="@i.IngredientID" />
26                         <button type="submit" class="btn btn-default btn-danger center-block" asp-action="Delete">Delete</button>
27                     </form>
28                 </td>
29             </tr>
30         </tr>
31     </table>
32 </div>
```

Ingredient/Edit:

Edit calls partial View for Single Ingredient input

```
@model Ingredient
```

```
@{
```

```
ViewData["Title"] = "Ingredient List";
```

```
}
```

```
<div class="container">
    <h1 class="label-info" id="h1Format"><b>@ViewBag.InputType Ingredient</b></h1>
    <form class="form-group-sm table-bordered bg-warning" asp-action="IngEdit" method="post">
        <input type="hidden" asp-for="IngredientID" />
        @await Html.PartialAsync("IngInput")
        <br />
        <div class="text-center ingFormBtn">
            <button class="btn btn-primary formBtn" type="submit">Save</button>
            <button class="btn btn-primary formBtn" asp-action="Index">Back To List</button>
        </div>
    </form>
</div>
```

```
@model Ingredient
```

```
<div class="form-group-sm col-sm-2">
    <label class="labelled">Amount: </label>
    <input asp-for="Amount" class="form-control" id="Description" />
    <div><span class="input-validation-error" asp-validation-for="Amount"></span></div>
</div>
<div class="form-group-sm col-sm-5">
    <label class="labelled">Unit Of Measure: </label>
    <select asp-for="Unit" class="form-control dropdown" id="Unit">
        <option value="Tsp">Teaspoon Tsp</option>
        <option value="Tbsp">Tablespoon Tbsp</option>
        <option value="Cup">Cup</option>
        <option value="Oz">Ounce Oz</option>
        <option value="Lb">Pound Lb</option>
        <option value="Kg">Kilogram Kg</option>
        <option value="g">Gram g</option>
        <option value="mg">Miligram mg</option>
        <option value="Pinch">Pinch</option>
        <option value="L">Liter L</option>
        <option value="ml">Mililter mL</option>
        <option value="Gal">Gallon</option>
        <option value="Dash">Dash</option>
        <option value="Qt">Quart</option>
        <option value="Whole">Whole</option>
        <option value="Part">Part</option>
    </select>
    <div><span class="input-validation-error" asp-validation-for="Unit"></span></div>
</div>
<div class="form-group-sm col-sm-5">
    <label class="labelled">Ingredient Name: </label>
    <input asp-for="Name" class="form-control" id="Name" />
    <div><span class="input-validation-error" asp-validation-for="Name"></span></div>
</div>
```

No issues found

Ingredient/AddIngredientToRecipe:

AddIngredientToRecipe calls partial View for Single Ingredient input

```
@model Ingredient
```

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

```
<div class="container">  
    <h1 class="label-info" id="h1Format"><b>Add Ingredient To Recipe</b></h1>  
    <form class="form-group-sm table-bordered bg-warning" asp-action="IngAddToRecipe" method="post">  
        <input type="hidden" asp-for="IngredientID" />  
        <input type="hidden" asp-for="RecipeID" />  
        @await Html.PartialAsync("IngInput")  
        <br />  
        <div class="text-center ingFormBtn">  
            <button class="btn btn-primary formBtn" type="submit">Save</button>  
            <button class="btn btn-primary formBtn" asp-controller="Recipe" asp-action="List">Back To List</button>  
        </div>  
    </form>  
</div>
```


Recipe/Create Update:

```
@model RecipeViewModel
@{
    ViewData["Title"] = "Create";
}

<div class="container">
    <h1 class="label-info" id="h1Format"><b>Add Recipe</b></h1>
    <form class="form-group-sm table-bordered bg-warning" asp-action="Create" method="post">
        <input type="hidden" asp-for="Recipe.RecipeID" />
        <div class="form-group-sm">
            <label class="labelled">Recipe Name: </label>
            <div><span asp-validation-for="Recipe.Name"></span></div>
            <input asp-for="Recipe.Name" class="form-control" id="Name" />
        </div>
        <div class="form-group-sm">
            <label class="labelled">Description: </label>
            <div><span asp-validation-for="Recipe.Description"></span></div>
            <input asp-for="Recipe.Description" class="form-control" id="Description" placeholder="A brief description of the recipe" />
        </div>
        <div class="row">
            <div class="form-group-sm col-md-4">
                <label class="labelled">Servings: </label>
                <input asp-for="Recipe.Servings" class="form-control" id="Servings" />
                <div><span asp-validation-for="Recipe.Servings"></span></div>
            </div>
            <div class="form-group-sm col-md-4">
                <label class="labelled">Prep minutes: </label>
                <input asp-for="Recipe.PrepareMinutes" class="form-control" id="PrepMin" />
                <div><span asp-validation-for="Recipe.PrepareMinutes"></span></div>
            </div>
            <div class="form-group-sm col-md-4">
                <label class="labelled">Cook minutes: </label>
                <input asp-for="Recipe.CookMinutes" class="form-control" id="CookMin" />
                <div><span asp-validation-for="Recipe.CookMinutes"></span></div>
            </div>
            <div class="form-group-sm col-md-12 increaseTop">
                <br />
                <label class="labelled">Directions: </label>
                <textarea asp-for="Recipe.Directions" rows="5" class="form-control" id="Directions" placeholder="Enter directions separated by periods."></textarea>
                <div><span asp-validation-for="Recipe.Directions"></span></div>
            </div>
        </div>
    </form>
</div>
```

```
</div>
<div class="row">
    @for (int x = 0; x < Model.Ingredients.Count(); x++)
    {
        @Html.HiddenFor(i => i.Ingredients[x].IngredientID)
        <div class="col-sm-2">
            @Html.LabelFor(i => i.Ingredients[x].Amount, new { @class = "labelled" })
            <h6>@Html.EditorFor(i => i.Ingredients[x].Amount, new { htmlAttributes = new { @class = "form-control" } })</h6>
            @Html.ValidationMessageFor(i => i.Ingredients[x].Amount, "", new { @class = "input-validation-error" })
        </div>
        <div class="col-sm-4">
            @Html.LabelFor(i => i.Ingredients[x].Unit, new { @class = "labelled" })
            <h6>@Html.EditorFor(i => i.Ingredients[x].Unit, new { htmlAttributes = new { @class = "form-control" } })</h6>
            @Html.ValidationMessageFor(i => i.Ingredients[x].Unit, "", new { @class = "input-validation-error" })
        </div>
        <div class="col-sm-6">
            @Html.LabelFor(i => i.Ingredients[x].Name, new { @class = "labelled" })
            <h6>@Html.EditorFor(i => i.Ingredients[x].Name, new { htmlAttributes = new { @class = "form-control" } })</h6>
            @Html.ValidationMessageFor(i => i.Ingredients[x].Name, "", new { @class = "input-validation-error" })
        </div>
    }
</div>
<br />
<div class="text-center">
    <br />
    <button class="btn btn-primary formBtn" type="submit">Add</button>
    <button class="btn btn-primary formBtn" asp-action="List">Back To List</button>
</div>
</form>
</div>
```

Recipe/Edit Update:

Edit calls FullRecipeInput partial view for recipe input

```
@model RecipeViewModel
@{
    ViewData["Title"] = "Edit";
}

<div class="container">
    <h1 class="label-info" id="h1Format"><b>Edit Recipe</b></h1>
    <form class="form-group-sm table-bordered bg-warning" asp-action="Edit" method="post">
        @await Html.PartialAsync("FullRecipeInput")
        <br />
        <div class="text-center">
            <br />
            <button class="btn btn-primary formBtn" type="submit">Update</button>
            <button class="btn btn-primary formBtn" asp-action="List">Back To List</button>
        </div>
    </form>
</div>
```

Recipe/View Update:

```
@model RecipeViewModel
@{
    ViewData["Title"] = "View";
}
```

```
<div id="rOverview" class="container">
    <div class="row label-info" id="h1Format">
        <div class="col-sm-8 pull-sm-4"><h1 class="label-info" id="h1Format"><b>@Model.Recipe.Name</b></h1></div>
        <div class="col-sm-1">
            <a class="btn btn-default btn-warning center-block" asp-action="Edit" asp-route-RecipeID=@Model.Recipe.RecipeID>Edit Recipe</a>
        </div>
    </div>
    <h4 class="label-info" id="h4Format"><b>@Model.Recipe.Description</b></h4>
    <table class="table table-sm table-striped table-bordered">
        <thead>
            <tr id="tHead">
                <th>Servings:</th>
                <th>Prep Time:</th>
                <th>Cook Time:</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <th>@Model.Recipe.Servings</th>
                <th>@Model.Recipe.PrepareMinutes</th>
                <th>@Model.Recipe.CookMinutes</th>
            </tr>
        </tbody>
    </table>
</div>
```

```
</div>
<div class="container">
    <div id="rIngredients" class="mark">
        <h2><b><u>INGREDIENTS</u></b></h2>
        <ul class="list-unstyled">
            @foreach(Ingredient i in Model.Ingredients)
            {
                <li>@i.Amount @i.Unit @i.Name</li>
            }

            <li><a asp-action="IngAddToRecipe" asp-controller="Ingredient" asp-route-RecipeID=@Model.Recipe.RecipeID"
                class="btn btn-success glyphicon-plus">Add New Ingredient</a>
            </li>
        </ul>
    </div>
    <br />
    <div class="container">
        <div id="rDirections" class="mark">
            <h2><b><u>DIRECTIONS</u></b></h2>
            <ul class="list-unstyled">
                @
                {
                    string[] dirList = Model.Recipe.Directions.Split(".");

                    foreach (var line in dirList)
                    {
                        <li>@line</li>
                    }
                }
            </ul>
        </div>
    </div>
```

Shared/FullRecipeInput:

```
@model RecipeViewModel

<input type="hidden" asp-for="Recipe.RecipeID" />
<div class="form-group-sm">
  <label class="labelled">Recipe Name: </label>
  <div><span asp-validation-for="Recipe.Name"></span></div>
  <input asp-for="Recipe.Name" class="form-control" id="Name" />
</div>
<div class="form-group-sm">
  <label class="labelled">Description: </label>
  <div><span asp-validation-for="Recipe.Description"></span></div>
  <input asp-for="Recipe.Description" class="form-control" id="Description" placeholder="A brief description of the recipe" />
</div>
<div class="row">
  <div class="form-group-sm col-md-4">
    <label class="labelled">Servings: </label>
    <input asp-for="Recipe.Servings" class="form-control" id="Servings" />
    <div><span asp-validation-for="Recipe.Servings"></span></div>
  </div>
  <div class="form-group-sm col-md-4">
    <label class="labelled">Prep minutes: </label>
    <input asp-for="Recipe.PrepareMinutes" class="form-control" id="PrepMin" />
    <div><span asp-validation-for="Recipe.PrepareMinutes"></span></div>
  </div>
  <div class="form-group-sm col-md-4">
    <label class="labelled">Cook minutes: </label>
    <input asp-for="Recipe.CookMinutes" class="form-control" id="CookMin" />
    <div><span asp-validation-for="Recipe.CookMinutes"></span></div>
  </div>
  <div class="form-group-sm col-md-12 increaseTop">
    <br />
    <label class="labelled">Directions: </label>
    <textarea asp-for="Recipe.Directions" rows="5" class="form-control" id="Directions" placeholder="Enter directions separated by commas" />
    <div><span asp-validation-for="Recipe.Directions"></span></div>
  </div>
</div>
```

```
<div class="row">
  <div class="col-sm-2 amountIn">
    <label class="labelled">Amount: </label>
  </div>
  <div class="col-sm-4 unitIn">
    <label class="labelled">Unit: </label>
  </div>
  <div class="col-sm-6 nameIn">
    <label class="labelled">Name: </label>
  </div>
  @for (int x = 0; x < Model.Ingredients.Count(); x++)
  {
    @Html.HiddenFor(i => i.Ingredients[x].IngredientID)
    @Html.HiddenFor(i => i.Ingredients[x].RecipeID)
    <div class="col-sm-2 amountIn">
      <h6>@Html.EditorFor(i => i.Ingredients[x].Amount, new { htmlAttributes = new { @class = "form-control" } })</h6>
      @Html.ValidationMessageFor(i => i.Ingredients[x].Amount, "", new { @class = "input-validation-error" })
    </div>
    <div class="col-sm-4 unitIn">
      <h6>@Html.EditorFor(i => i.Ingredients[x].Unit, new { htmlAttributes = new { @class = "form-control" } })</h6>
      @Html.ValidationMessageFor(i => i.Ingredients[x].Unit, "", new { @class = "input-validation-error" })
    </div>
    <div class="col-sm-6 NameIn">
      <h6>@Html.EditorFor(i => i.Ingredients[x].Name, new { htmlAttributes = new { @class = "form-control" } })</h6>
      @Html.ValidationMessageFor(i => i.Ingredients[x].Name, "", new { @class = "input-validation-error" })
    </div>
    <h6>
  }
</div>
```