

Special Topics: GitHub Workflows, Class Libraries

Introduction

In this lesson, I will introduce you to several special topics that are useful skills to have when working on large development projects in teams.

Software development is often performed in teams with multiple people working on the same project and editing the same code. Oftentimes this develop can occur asynchronous with people working on different tasks at different times but still working on the same project with a single objective.

You will learn how to leverage GitHub features such as **issues** and **branches** to perform asynchronous group work. You will also practice writing class libraries and unit testing in lesson. Both those capabilities indispensable tools in large software development projects with teams of developers.

Asynchronous Group Work

Asynchronous group work or collaboration allows teams of software developers to carry out their responsibilities remotely and at a time that is convenient for each member. You can work on a task at a certain time, remotely while your teammate or group member works on a different task at a different time in a different location.

In software development, GitHub offers features to simplify and strengthen asynchronous group work. You will be able to work on a single software project in a team, yet still be able to work on your specific tasks independently without relying on a specific group member. The GitHub features you will learn about are issues and branches.

GitHub Branches

Branching in GitHub is a powerful feature that allows developers to work on different parts of a project independently without affecting the main codebase. Here's an overview and explanation of its key concepts and advantages:

What is a Branch?

A branch is essentially a parallel version of a repository. By default, most GitHub repositories have a branch called `main` (or `master` in older versions). When developers create a branch, they make a copy of the codebase, allowing them to work on new features, bug fixes, or experiments without modifying the main branch.

Key Concepts:

1. **Creating a Branch:** A new branch is created from a specific point in the main branch (or any other branch). This copy can then be worked on independently.
2. **Switching Between Branches:** Developers can switch between branches to work on different tasks or features without losing the state of their code in the other branches.
3. **Merging Branches:** After work is completed in a branch, the changes can be merged back into the main branch (or another branch). This ensures that the new feature or fix becomes part of the main project.
4. **Pull Requests:** GitHub uses pull requests to facilitate code review and collaboration. A pull request is created to propose merging changes from one branch into another (usually from a feature branch into the main branch). This allows other team members to review, comment, and approve the changes.

Advantages of Branching in GitHub:

1. **Isolation of Work:** Each branch acts as an isolated environment where developers can work on new features or bug fixes without interfering with the main branch or each other's work.
2. **Collaboration in Teams:** In the context of asynchronous group work, multiple developers can work on different branches at the same time. They don't need to wait for others to finish their tasks, improving the team's overall productivity.
3. **Safe Experimentation:** Branching allows for experimentation without fear of breaking the main project. If the changes in a branch don't work out, that branch can be discarded without affecting the rest of the project.

4. **Code Reviews and Quality Control:** Through pull requests, teams can enforce code reviews, ensuring that all changes are carefully evaluated and tested before being merged into the main branch.
5. **Version Control:** Branching makes it easier to track different versions of features or fixes, allowing teams to switch between versions if needed.

Branching is particularly useful for asynchronous group work, where developers can work on different aspects of a project without directly interacting in real-time but still contribute to the same codebase.

Class Libraries

A **Class Library** in C# is a collection of reusable classes and methods that can be used by other applications. It is typically packaged into a `.dll` (Dynamic Link Library) file, which allows multiple applications to share the same functionality without needing to copy the code across projects. Class libraries are commonly used to organize code into modular components.

Benefits of Class Libraries:

1. **Code Reusability:** Class libraries promote code reuse across multiple projects. Once you've developed a class library, it can be referenced in any number of applications, reducing redundancy and ensuring consistency across projects.
2. **Maintenance:** Updating a class library in one place will automatically propagate the changes to all projects that reference it. This ensures centralized control and easier maintenance of common functionality.
3. **Testing:** Class libraries can be tested and debugged separately from the main application. This modular testing leads to more reliable code and helps to isolate bugs.
4. **Asynchronous Development:** In asynchronous projects, especially those with distributed teams, breaking down the project into independent modules or components is beneficial to parallel development. Since class libraries are self-contained and can be developed independently, multiple team members can work on different class libraries at the same time. Once complete, these libraries can be integrated into the main project.

Class libraries in C# offer a way to modularize, reuse, and organize code. By creating class libraries, developers can share common functionality across projects, making development faster and maintenance easier. They provide a clean separation of concerns and help reduce redundancy across applications.

Class libraries are also a powerful tool in asynchronous software development because they allow teams to work in parallel on independent modules.

Unit Testing

In this lesson, you will create class libraries and you will test your code in those class libraries with unit tests. I have written the unit tests. You will just run them against your code.

Unit testing is a process in software development where individual units or components of the code (usually methods or functions) are tested in isolation to ensure they work as expected. In C#, this is typically done using testing frameworks such as **MSTest**. A unit is the smallest testable part of an application, often a single method or property.

Unit tests in C# and any other programming language are an essential part of large development projects. Unit tests can verify the correctness of small, isolated parts of code. They improve code quality by catching bugs early. By writing and using unit tests, developers can create more reliable and maintainable software.

Learn More

- [Tutorial: Create a .NET class library using Visual Studio Code](#) ↗
- [Tutorial: Test a .NET class library using Visual Studio Code](#) ↗

Lab 11 Assignment

Complete Lab 11 by completing the tasks below. You will receive an email from me inviting you to a private GitHub repository. You and one other classmate will be working together to complete two independent tasks on this codebase within the same GitHub repository.

One of you (teammate 1) will create a Shipping class library to calculate shipping costs of certain orders. The other (teammate 2) will create a Tax class library to calculate sales tax costs of the same orders. **In the email from me, you will be told which teammate will perform which task. As you are following the lab instructions, make sure you follow the instructions for your task.**

You will use branching and issues in GitHub to perform these tasks independently and asynchronously. You will build independent class libraries and unit test them so that you can work on a team but still work independently and asynchronously.

Combined, you and your teammate will add both features to a single C# ASP.NET project. **You will be graded individually on your own code and your own contributions.** You can interact with your teammate as much or as little as you like. Your work will have no impact on their work and vice versa.

The **learning objectives** of this assignment are:

- Gain experience in asynchronous software development teams.
- Practice creating branches in GitHub.
- Develop the ability to build class libraries in C#.
- Understand how to run existing unit tests in C#.
- Become proficient in creating pull requests in GitHub.
- Learn the process of merging branches in GitHub through pull requests.

Task 0: Complete Pre-Assignment Survey

As part of research and continuous improvement of this course, I am interested in

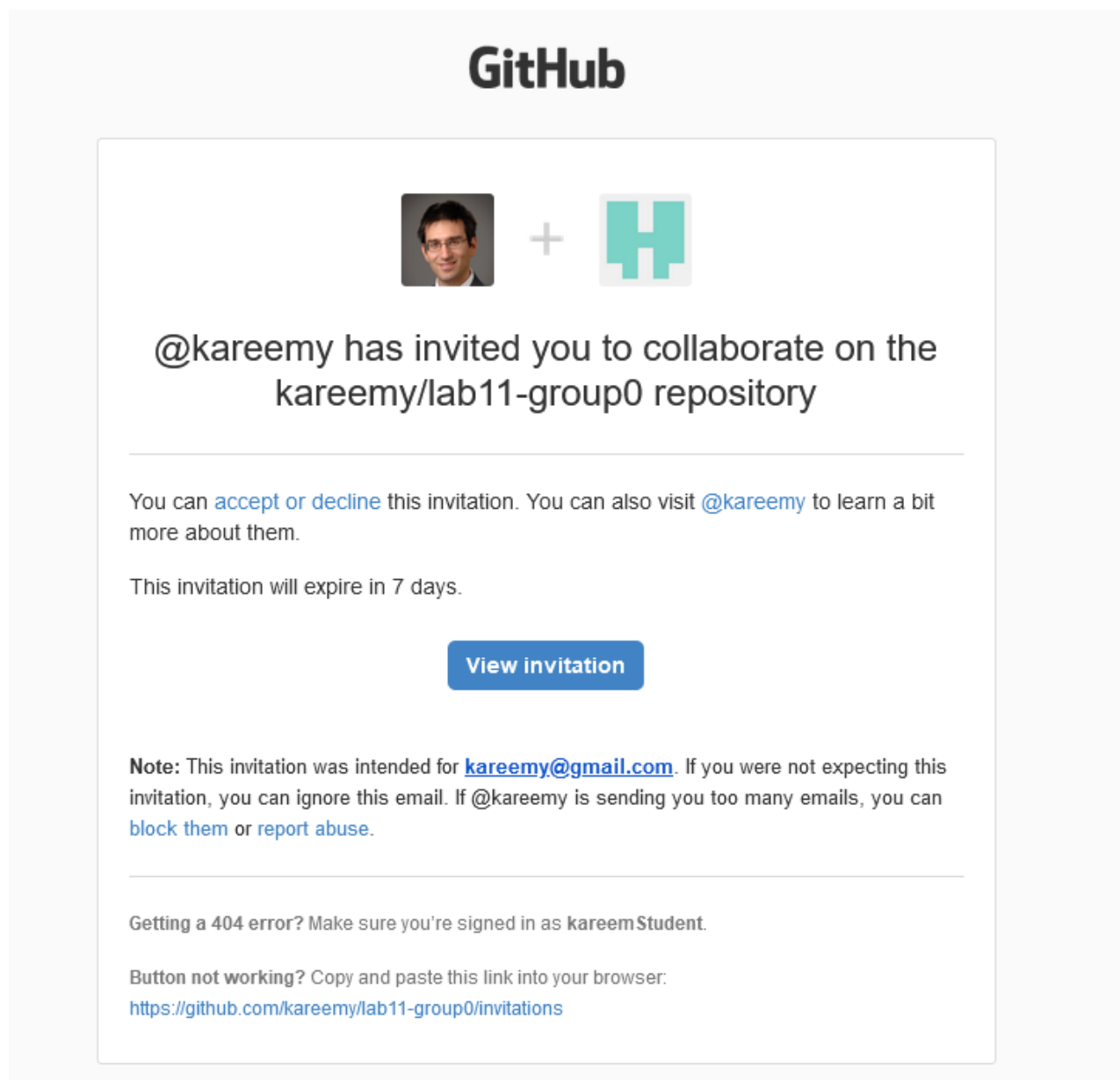
your thoughts on group assignments in courses and your understanding of certain software development techniques before completing this lab assignment. Please complete this pre-assignment survey before starting on the assignment.

Completing this survey is worth 10 points and you can complete it at this link:

<https://forms.gle/FKwsyU4tKVj17zV48> ↗

Task 1: Accept Invite to GitHub Repository

Check your WT email address for an invitation to the **lab11-groupX** (where X is your group number) GitHub repository. Accept the invitation. The email should look like this -

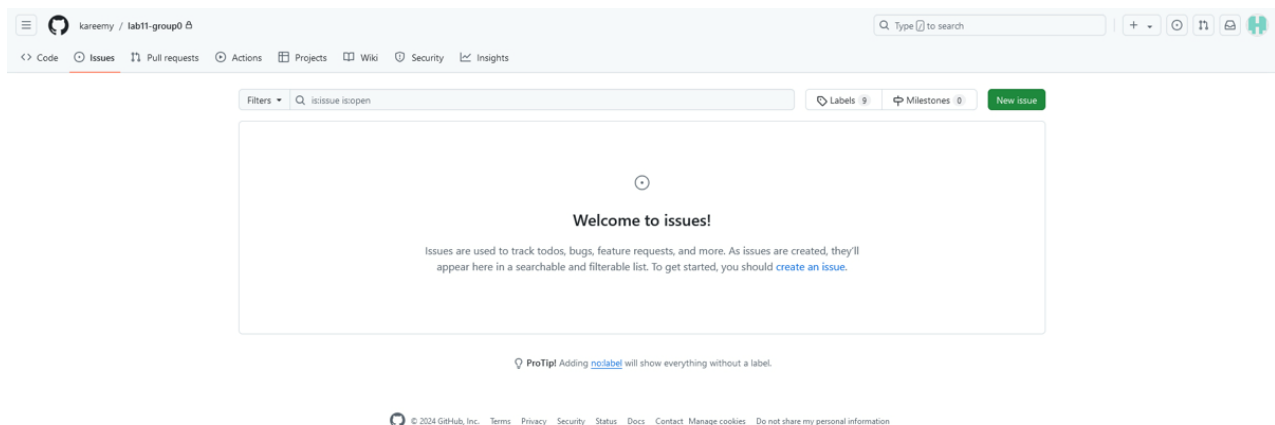




Task 2: Create an Issue in GitHub

Once you accept the invite, you should be taken to the GitHub repository with all the starter code. Within GitHub, you need to create an issue for your particular programming task - either **Calculate shipping costs** or **Calculate sales tax**.

1. Within GitHub, click on the **Issues** tab -



2. Click **New issue**. If you are tasked with calculating shipping cost, click the Shipping tab below and Submit the new issue based on the image. If you are tasked with calculating tax, click the Tax tab below and Submit the new issue based on that image.

Shipping



Add a title

Calculate shipping costs

Add a description

Write Preview

H B I

Add support to calculate shipping costs for each order -

1. Create a ShippingLibrary class library
2. Within the class library, write code to calculate shipping costs based on order zip code
3. Test code with built-in unit tests
4. Modify ReviewOrder page model (in OrderApp) to display shipping costs

Markdown is supported

Paste, drop, or click to add files

Submit new issue

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Assignees

No one—[assign yourself](#)

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Shows branches and pull requests linked to this issue.

Helpful resources

[GitHub Community Guidelines](#)

Tax



Add a title

Calculate sales tax

Add a description

Write Preview

H B I

Add support to calculate Tax for each order -

1. Create a TaxLibrary class library
2. Within the class library, write code to calculate sales tax based on the order zip code
3. Test code with built-in unit tests
4. Modify ReviewOrder page (in OrderApp) to display tax

Markdown is supported

Paste, drop, or click to add files

Submit new issue

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Assignees

No one—[assign yourself](#)

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Shows branches and pull requests linked to this issue.

Helpful resources

[GitHub Community Guidelines](#)

3. **Assign the new issue to yourself.** Click assign yourself under Assignees on the right side of the Issues page.

Task 3: Branch in GitHub

1. **Create a new branch for your issue.** There are several ways to create a branch in GitHub. I recommend click create a branch within the issue window itself -

Calculate shipping costs #1

Edit New issue

Open kareemStudent opened this issue 4 minutes ago · 0 comments

kareemStudent commented 4 minutes ago

Add support to calculate shipping costs for each order -

1. Create a ShippingLibrary class library
2. Within the class library, write code to calculate shipping costs based on order zip code
3. Test code with built-in unit tests
4. Modify ReviewOrder page model (in OrderApp) to display shipping costs

kareemStudent self-assigned this 4 minutes ago

Add a comment

Write Preview

Add your comment here...

Markdown is supported Paste, drop, or click to add files

Close issue Comment

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Assignees
kareemStudent

Labels
None yet

Projects
None yet

Milestone
No milestone

Development
Create a branch for this issue or link a pull request.

Notifications
Unsubscribe

You're receiving notifications because you authored the thread.

1 participant

Lock conversation
Pin issue
Transfer issue

2. This will open a new window. Give your branch a name. For shipping costs, call it **1-calculate-shipping-costs**. For tax costs, call it **2-calculate-sales-tax**. Click **Create branch**.

Create a branch for this issue

Branch name

1-calculate-shipping-costs

Repository destination

kareemy/lab11-group0

Change branch source

What's next?

☐ Open in codespace

☒ Checkout locally

☐ Open branch with GitHub Desktop

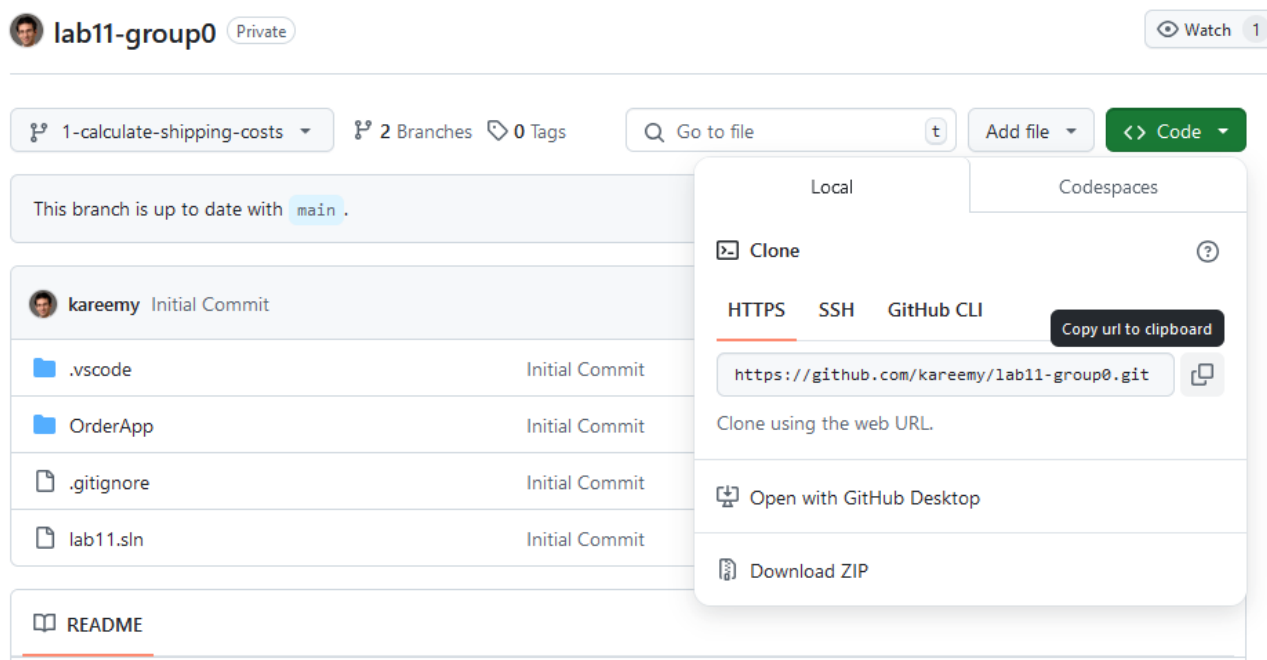
Beta Share feedback

Create branch


Task 4: Clone Repository

After you create your branch, you can clone the repository in VS Code. This will be done the same way as previous assignments.

1. Within the Code tab on the GitHub web page, Click the **Code** button. Copy the HTTPS web URL to the clipboard.



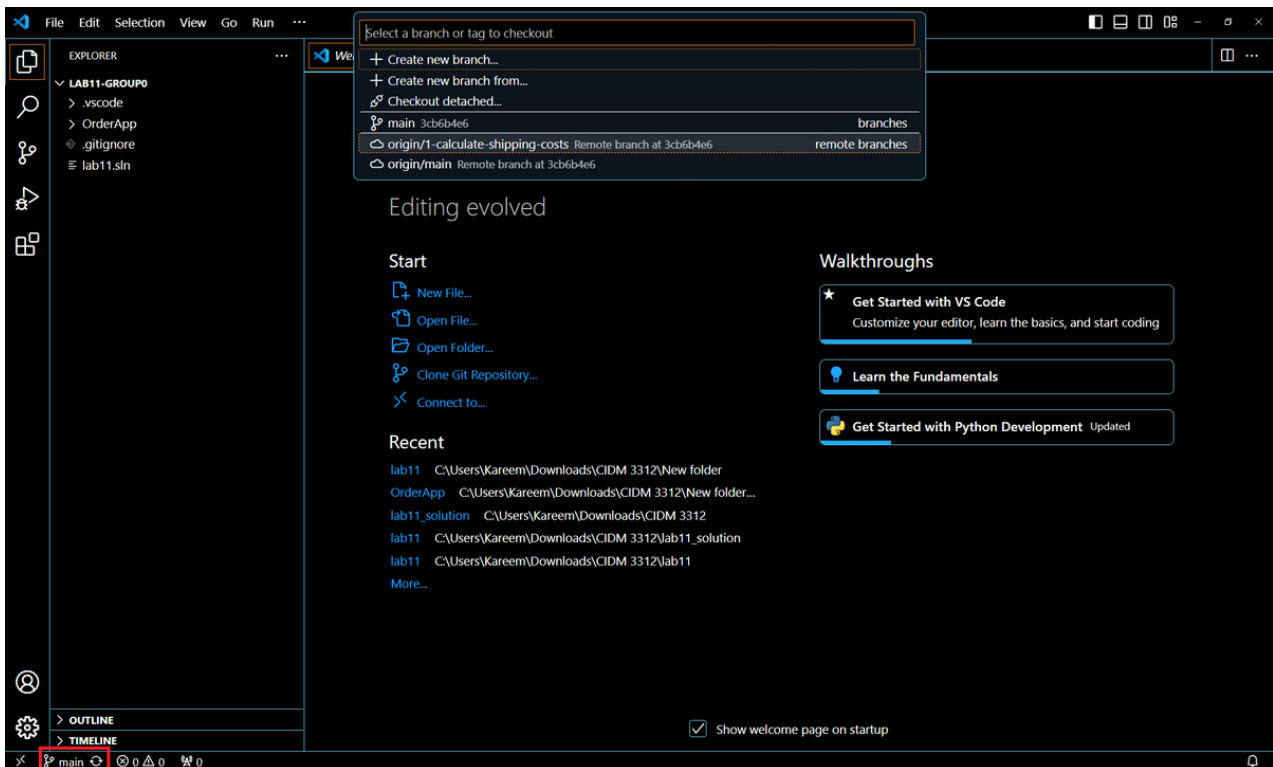
2. Within Visual Studio Code, open the **Command Palette** (View ⇒ Command Palette or Ctrl+Shift+P)
3. Type **git: clone** and press enter.
4. Paste the repository link here and press enter. When prompted, open the repository.

 VS Code may pop up a warning message that says there were errors restoring the **ShippingTest** or **TaxTest** projects. You can ignore those warnings for now. These test classes refer to code you have not written yet. Once you complete the assignment, the errors will resolve.

Task 5: Checkout Your Branch

Now the repository should be open in VS Code. By default, you will be working on the main branch. You need to checkout your branch to start working on your GitHub Branch.

1. Checkout your branch through the VS Code GUI. On the bottom status bar of VS Code, click on the main icon (main indicates you are working on the main branch).



2. In the textbox that opens up, select your branch. For shipping, it should be **origin/1-calculate-shipping-costs**. For tax, it should be **origin/2-calculate-sales-tax**.
3. Another way to checkout your branch is through the terminal with the `git checkout` command. If you did not check out your branch already, type the following at the terminal where branchName is the name of your specific branch:

```
git checkout branchName
```

4. Verify the correct branch. At the terminal., type `git branch` to see branch you are working on. The branch you are working on should have an asterisk and be highlighted in green. Make sure it is **1-calculate-shipping-costs** or **2-calculate-sales-tax**.

```

    TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE  PORTS

● PS C:\Users\Kareem\Downloads\CIDM 3312\testing\lab11-group0> git branch
  * 1-calculate-shipping-costs
    main
○ PS C:\Users\Kareem\Downloads\CIDM 3312\testing\lab11-group0> 

```

Task 6: Make Class Library

1. To solve your specific task, you will create a new class library. All the code you write will be in that class library. Create a new class library by typing the following at the terminal:

ShippingLibrary

```
dotnet new classlib -o ShippingLibrary
```

TaxLibrary

```
dotnet new classlib -o TaxLibrary
```

2. Add your class library to the main solution (sln) file.

ShippingLibrary

```
dotnet sln add ShippingLibrary\ShippingLibrary.csproj
```

TaxLibrary

```
dotnet sln add TaxLibrary\TaxLibrary.csproj
```

3. Add a reference to your class library in the **OrderApp** project. This ensures that

OrderApp can access your class library.

ShippingLibrary

```
dotnet add OrderApp/OrderApp.csproj reference ShippingLibrary/  
ShippingLibrary.csproj
```

TaxLibrary

```
dotnet add OrderApp/OrderApp.csproj reference TaxLibrary/  
TaxLibrary.csproj
```

Task 7: Write Code in Class Library

1. Creating a class library should have made a new folder (either **ShippingLibrary/** or **TaxLibrary/**). Within that folder there is a **Class1.cs** file. That file is where you will write your code.
2. Replace the contents of **Class1.cs** with the following code for your particular task:

ShippingLibrary

```
namespace ShippingLibrary;

public static class ShippingClass
{
    public static decimal CalculateShippingCost(string zipCode)
    {
        // Ensure the zipcode is valid (numeric and 5 digits long)
        if (string.IsNullOrEmpty(zipCode) || zipCode.Length !=
5)
        {
            throw new ArgumentException("Invalid Zip Code");
        }

        // Convert the zipcode to an integer for comparison
        int zipCodeNumber = int.Parse(zipCode);

        // Shipping cost logic based on zip code ranges
        if (zipCodeNumber >= 10000 && zipCodeNumber <= 19999)
        {
            return 5.99m; // Example shipping cost for this range
        }
        else if (zipCodeNumber >= 20000 && zipCodeNumber <= 29999)
        {
            return 7.99m;
        }
        else if (zipCodeNumber >= 30000 && zipCodeNumber <= 39999)
        {
            return 9.99m;
        }
        else if (zipCodeNumber >= 40000 && zipCodeNumber <= 49999)
        {
            return 11.99m;
        }
        else if (zipCodeNumber >= 50000 && zipCodeNumber <= 59999)
        {
            return 13.99m;
        }
        else
        {
            // Default or high-cost shipping for other areas
            return 15.99m;
        }
    }
}
```

TaxLibrary

```
namespace TaxLibrary;

public static class TaxClass
{
    public static decimal CalculateSalesTax(string zipCode)
    {
        if (string.IsNullOrEmpty(zipCode) || zipCode.Length != 5)
        {
            throw new ArgumentException("Invalid Zip Code.");
        }

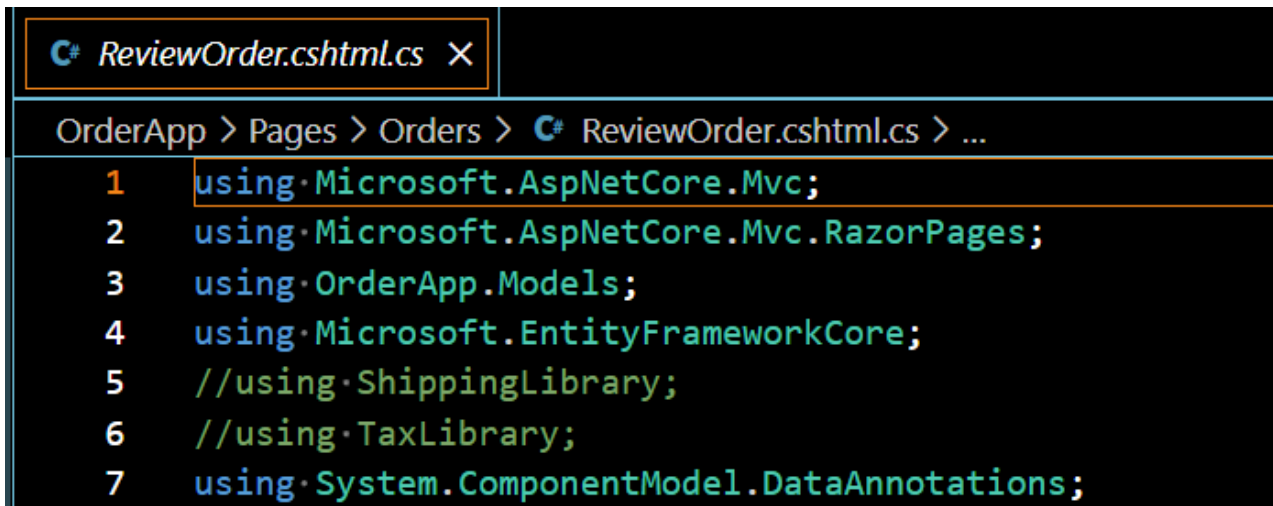
        // Convert zip code to integer for easier range checks
        int zip = int.Parse(zipCode);

        // Define sales tax rates based on zip code ranges
        if (zip >= 10000 && zip < 20000)
        {
            return 0.0875M; // Example: New York area
        }
        else if (zip >= 90000 && zip < 100000)
        {
            return 0.0925M; // Example: California area
        }
        else if (zip >= 70000 && zip < 80000)
        {
            return 0.0445M; // Example: Louisiana area
        }
        else if (zip >= 60000 && zip < 70000)
        {
            return 0.0625M; // Example: Illinois area
        }
        else if (zip >= 30000 && zip < 40000)
        {
            return 0.07M; // Example: Georgia area
        }
        else
        {
            return 0.05M; // Default tax rate for other regions
        }
    }
}
```

3. Spend a few minutes reviewing this code and understand what it does. There is

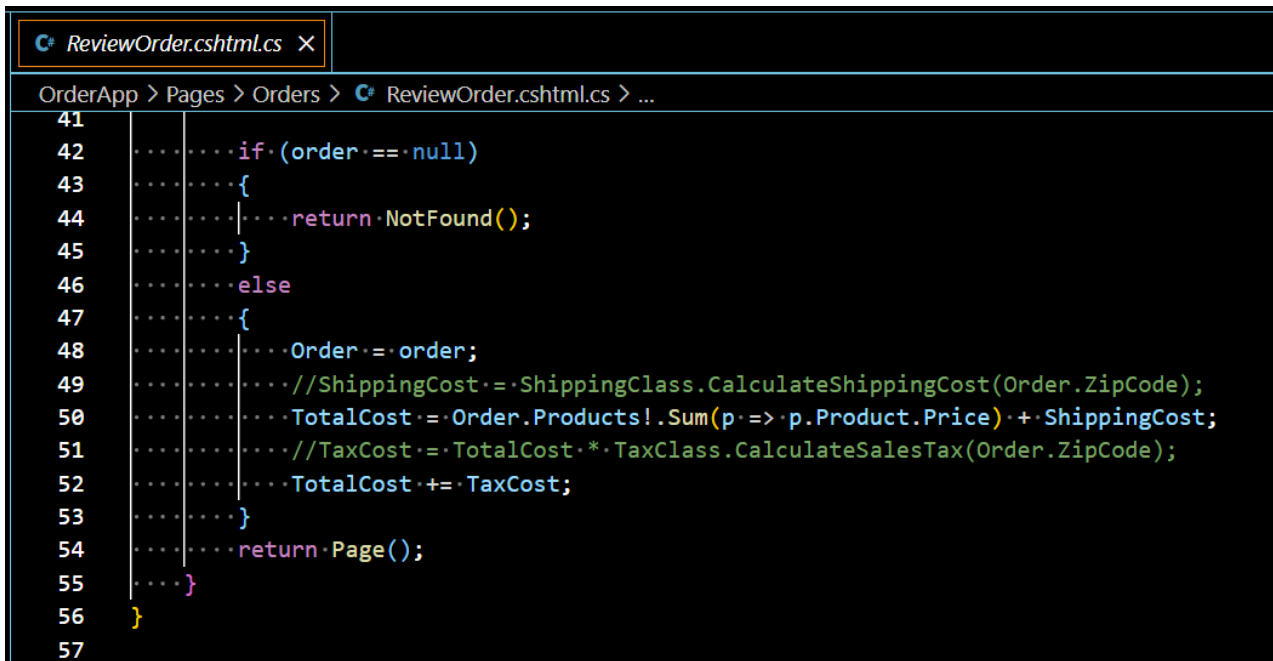
a check to determine that the zip code is valid. If it is invalid, an exception will be thrown. Otherwise, there is a simple if/else if/else block that determines the shipping price or tax percentage based on a range of zip codes.

4. Save your code.
5. Open **OrderApp/Pages/Orders/ReviewOrder.cshtml.cs** and uncomment that access your class library.
6. On lines 5-6 uncomment the using directive for your class:



```
C# ReviewOrder.cshtml.cs X
OrderApp > Pages > Orders > C# ReviewOrder.cshtml.cs > ...
1  using Microsoft.AspNetCore.Mvc;
2  using Microsoft.AspNetCore.Mvc.RazorPages;
3  using OrderApp.Models;
4  using Microsoft.EntityFrameworkCore;
5  //using ShippingLibrary;
6  //using TaxLibrary;
7  using System.ComponentModel.DataAnnotations;
```

7. On line 49 or 51 uncomment the C# statement for your class:



```
C# ReviewOrder.cshtml.cs X
OrderApp > Pages > Orders > C# ReviewOrder.cshtml.cs > ...
41
42  .....if (order == null)
43  .....{
44  .....    return NotFound();
45  .....}
46  .....else
47  .....{
48  .....    Order = order;
49  .....    //ShippingCost = ShippingClass.CalculateShippingCost(Order.ZipCode);
50  .....    TotalCost = Order.Products!.Sum(p => p.Product.Price) + ShippingCost;
51  .....    //TaxCost = TotalCost * TaxClass.CalculateSalesTax(Order.ZipCode);
52  .....    TotalCost += TaxCost;
53  .....}
54  .....return Page();
55  .....}
56  }
57
```

❗ If there is a red squiggly error on any of these lines of code, it is possible that VS Code hasn't updated itself yet to bring in the class library. Run `dotnet build` at the terminal.

The build should fail with 1 error related to the unit test for your teammates section because their code is not in your branch. That error is OK. If there are other errors, debug them. Otherwise the red squiggly errors should go away and you can continue.

Task 8: Test Code

1. I have already written unit tests to test your particular feature. Run either the Shipping or Tax unit tests and ensure that all tests pass. Test your code by running the unit tests at the terminal:

ShippingLibrary

```
dotnet test .\ShippingTest\ShippingTest.csproj
```

TaxLibrary

```
dotnet test .\TaxTest\TaxTest.csproj
```

2. The output should look like this. There are 7 tests and all should pass.

```
Test run for C:\Users\Kareem\Downloads\CIDM 3312\testing\lab11-group0\ShippingTest\bin\Debug\net8.0\ShippingTest.dll (.NETCoreApp,Version=v8.0)
Microsoft (R) Test Execution Command Line Tool Version 17.10.0 (x64)
Copyright (c) Microsoft Corporation. All rights reserved.

Starting test execution, please wait...
A total of 1 test files matched the specified pattern.

Passed! - Failed:    0, Passed:    7, Skipped:    0, Total:    7, Duration: 37 ms - ShippingTest.dll (net8.0)
PS C:\Users\Kareem\Downloads\CIDM 3312\testing\lab11-group0>
```

3. Once all the unit tests pass, run the **OrderApp** project:

```
dotnet run --project .\OrderApp\OrderApp.csproj
```

4. Visit the **/Orders** URL. The web page should look like this -

Index

[Create New](#)

Customer Name	Order Date	Zip Code	
United States Department of Defense	5/15/2024	20301	Review Order
UK Ministry of Defence	6/23/2024	10001	Review Order
Germany Bundeswehr	7/10/2024	10002	Review Order
Google Inc.	3/5/2024	94043	Review Order
Amazon Web Services	1/12/2024	98109	Review Order

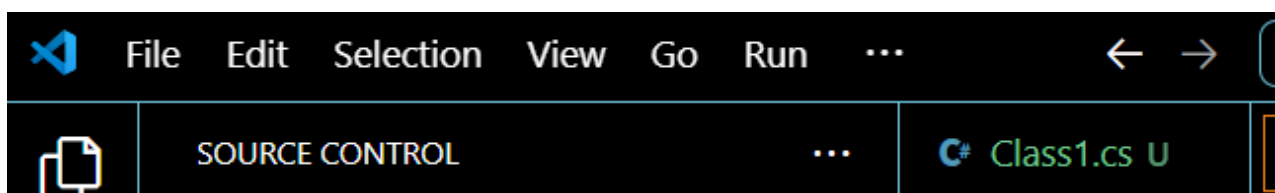
- Click Review Order for one of the orders. The Review Order page should look similar to this. If your teammate has not completed their portion, their portion will indicate \$0.00 for tax or shipping -

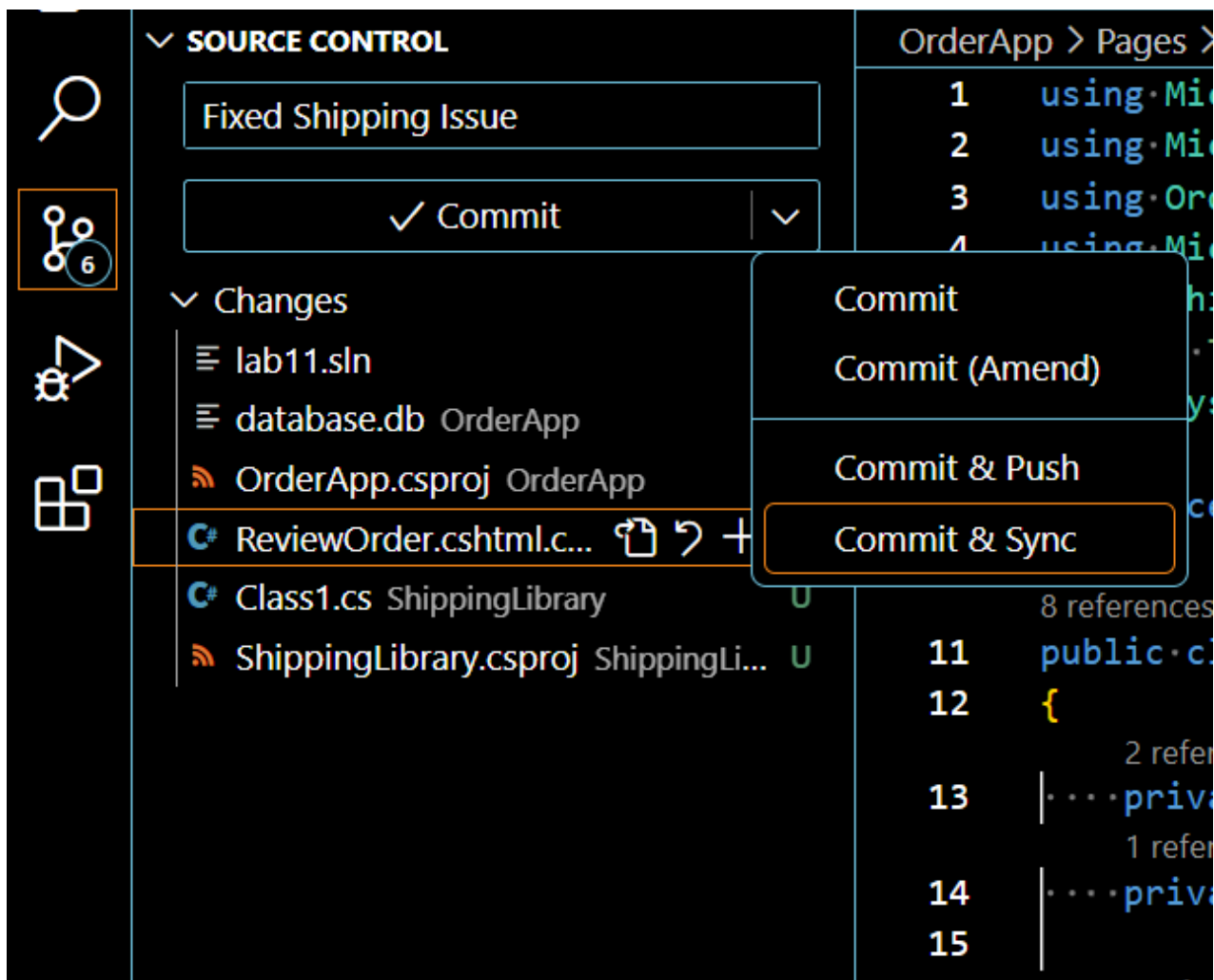
Review Order

Seraphine	\$200.00
PhantomClaw	\$99.95
Shipping (10001)	\$5.99
Tax (10001)	\$26.77
Total (USD)	\$332.71

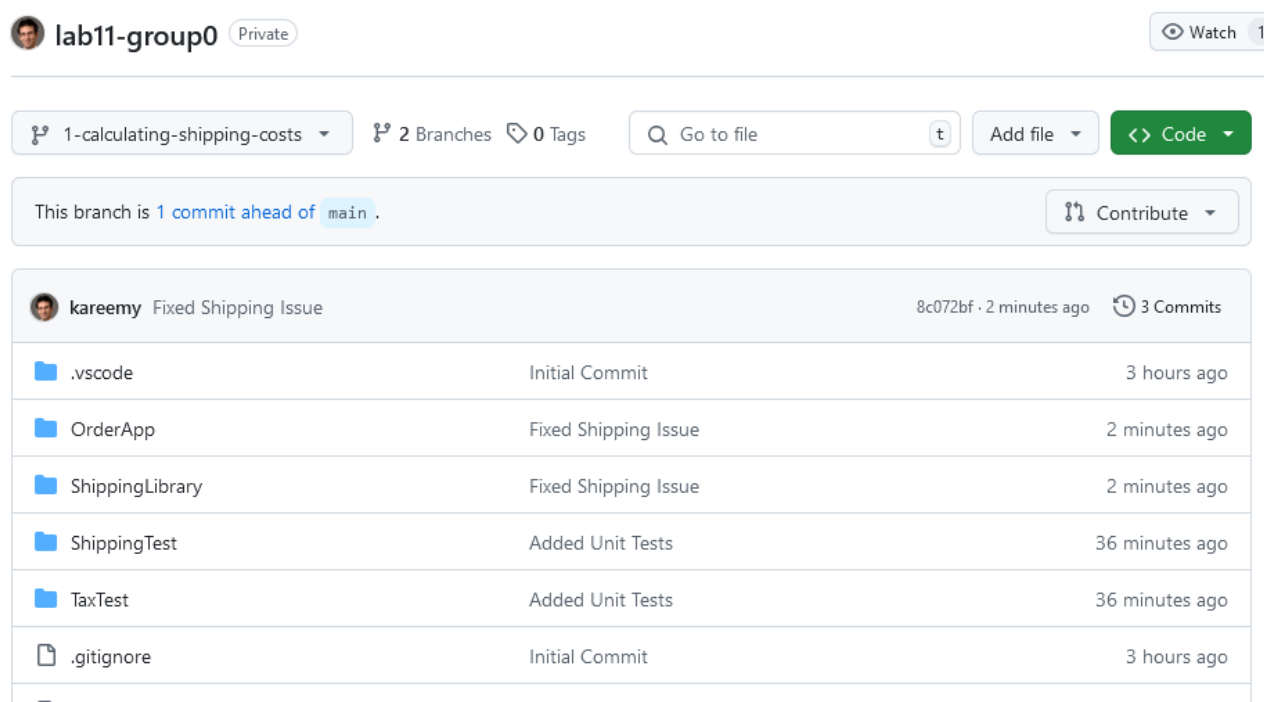
Task 9: Merge Branch on GitHub

- Commit and Sync your code back to GitHub. See this screenshot -





2. Visit GitHub.com, navigate to your repository, and select your branch. GitHub should indicate that your branch is 1 commit ahead of main.



- You can click that link to see the changes you made and also merge your code back into the main branch by creating a Pull request.

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#).

base: main

compare: 1-calculating-shipping-costs

✓ Able to merge. These branches can be automatically merged.

Discuss and review the changes in this comparison with others. [Learn about pull requests](#)

Create pull request

1 commit6 files changed1 contributor

Commits on Sep 14, 2024

Fixed Shipping Issue
kareemy committed 4 minutes ago

Showing 6 changed files with 64 additions and 2 deletions.

SplitUnified

OrderApp/OrderApp.csproj

@@ -20,4 +20,8 @@
20 20 <PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="8.0.5" />
21 21 </ItemGroup>
22 22
23 + <ItemGroup>
24 + <ProjectReference Include="..\ShippingLibrary\ShippingLibrary.csproj" />
25 + </ItemGroup>
26 +
23 27 </Project>

OrderApp/Pages/Orders/ReviewOrder.cshtml.cs

@@ -2,7 +2,7 @@
2 2 using Microsoft.AspNetCore.Mvc.RazorPages;
3 3 using OrderApp.Models;
4 4 using Microsoft.EntityFrameworkCore;
5 - //using ShippingLibrary;
5 + using ShippingLibrary;
6 6 //using TaxLibrary;
7 7 using System.ComponentModel.DataAnnotations;

- Click **Create pull request**. That will open a new window that looks like the below screenshot. Enter a description for your pull request and then click **Create pull request** again.

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#). [Learn more about diff comparisons here](#).

base: main

compare: 1-calculating-shipping-costs

✓ Able to merge. These branches can be automatically merged.

Add a title

Fixed Shipping Issue

Add a description

WritePreview

H B I

Added ~~ShippingLibrary~~ class library, passed all unit tests. Calculating shipping cost works now.

Reviewers

Suggestions

kareemyRequest

Assignees

No one—[assign yourself](#)

Labels

None yet

Projects

None yet

Markdown is supported

Paste, drop, or click to add files

Create pull request

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Milestone

No milestone

Development

Use [Closing keywords](#) in the description to automatically close issues

Helpful resources

[GitHub Community Guidelines](#)

1 commit

6 files changed

1 contributor

Commits on Sep 14, 2024

Fixed Shipping Issue

kareemy

committed 6 minutes ago

8c072bf

Showing 6 changed files with 64 additions and 2 deletions.

Split

Unified

- Finally, click **Merge pull request** to complete the merge between your branch and the main branch. After clicking Merge pull request, click **Confirm merge**.

Fixed Shipping Issue #2

Open

kareemStudent wants to merge 1 commit into `main` from `1-calculating-shipping-costs`

Conversation 0

Commits 1

Checks 0

Files changed 6

+64 -2

kareemStudent commented now

Added ShippingLibrary class library, passed all unit tests. Calculating shipping cost works now.

Fixed Shipping Issue

8c072bf

kareemStudent linked an issue `now` that may be closed by this pull request

Calculating shipping costs #1

Open

This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

Add a comment

Write

Preview

H

B

I

≡

<>

🔗

⋮

⋮

⋮

🔗

@

📎

↩

🗑

Add your comment here...

Markdown is supported

Paste, drop, or click to add files

Close pull request

Comment

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

ProTip!

Add `.patch` or `.diff` to the end of URLs for Git's plaintext views.

Reviewers

Suggestions

kareemy

Request

Still in progress? [Learn about draft PRs](#)

Assignees

No one—[assign yourself](#)

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

Calculating shipping costs

Notifications

Customize

Unsubscribe

You're receiving notifications because you authored the thread.

2 participants

Lock conversation


⚠️ If your teammate merged their pull request first, you may have conflicts and GitHub will say "Can't automatically merge".

Can't Automatically Merge

If you can't automatically merge, that means your teammate merged first and in this lab, 3 files may have a conflict because both you and your teammate edited those same files. You can still create the pull request and leave it unmerged or you can attempt to resolve the conflicts for extra credit. If you can't automatically merge, GitHub will look like this -

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#).

 base: 2-calculate-sales-tax ← compare: main ✖ Can't automatically merge. Don't worry, you can still create the pull request.

Discuss and review the changes in this comparison with others. [Learn about pull requests](#) Create pull request


2 commits


6 files changed

1 contributor


Commits on Sep 14, 2024


Fixed Shipping Issues

 kareemy committed 5 minutes ago

 d1b6f2d <>

Merge pull request #3 from kareemy/1-calculating-shipping-costs

 kareemy committed 4 minutes ago

Verified  39504c6 <>

Showing 6 changed files with 64 additions and 2 deletions. Split Unified


OrderApp/OrderApp.csproj


```
@@ -20,4 +20,8 @@
20      <PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="8.0.5" />
21      </ItemGroup>
22
23      + <ItemGroup>
24      + <ProjectReference Include="..\ShippingLibrary\ShippingLibrary.csproj" />
25      + </ItemGroup>
26      +
23 27 </Project>
```

Click **Create pull request** anyway even though you can't automatically merge.

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#). [Learn more about diff comparisons here](#).












 base: 2-calculate-sales-tax ← compare: main ✖ Can't automatically merge. Don't worry, you can still create the pull request.

 **Add a title**

Fixed Tax Issue

Add a description

Write Preview

H B I           

Add your description here...

Reviewers

No reviews

Assignees

No one—[assign yourself](#)

Labels

None yet

Projects

None yet

Milestone

No milestone

Markdown is supported

Paste, drop, or click to add files

Development

Use [Closing keywords](#) in the description to automatically close issues

Create pull request



Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Helpful resources


[GitHub Community Guidelines](#)

Enter a title and description for your pull request and click **Create pull request**. If you can't automatically merge pull requests, the next screen will look like this -


Fixed Tax Issue #4


 Open kareemy wants to merge 2 commits into `2-calculate-sales-tax` from `main` 





Conversation 0 Commits 2 Checks 0 Files changed 6



 kareemy commented now ...

No description provided.



 kareemy added 2 commits 6 minutes ago

-   Fixed Shipping Issues d1b6f2d
-   Merge pull request #3 from kareemy/1-calculating-shipping-costs ... Verified 39504c6

  **This branch has conflicts that must be resolved** Resolve conflicts

Use the [web editor](#) or the [command line](#) to resolve conflicts.

Conflicting files

- OrderApp/OrderApp.csproj
- OrderApp/Pages/Orders/ReviewOrder.cshtml.cs
- lab11.sln

Merge pull request ▼ You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

You are all done at this point. The pull request has been made but not merged. If you want extra credit, you can click **Resolve conflicts**. See the Resolve conflicts page at the end of this assignment for help on that topic.

- After you created your pull request, visit the **Code** tab again within the GitHub web page. You should see that your code is now merged into the `main` branch. If conflicts existed, your code may not be merged. That is OK.
- Visit the **Issues** tab. You should notice that your Issue was automatically closed. If conflicts existed, your issue may still be open. That is OK.

8. Your teammate's issue may still be open. They are responsible for their branch and their issue. Your work is complete. Congratulations on completing your portion of this assignment.

Task 10: Complete Post-Assignment Survey

Now that you have completed this lab assignment, please complete the post-assignment survey. This is required and worth 15 points. Complete it at this link:

<https://forms.gle/U5Va2ZHmutjq3N546> ↗

Resolve Merge Conflicts


If your teammate merged their changes before yours, you may notice conflicts while trying to merge your changes. This is because, in a few places, you are both editing the same files. GitHub needs to know how to merge those changes.

You can leave your pull request alone and not resolve conflicts. You will not lose credit if you do not resolve conflicts as it is outside the scope of this assignment. However, resolving code conflicts among development teams is a common occurrence. If you would like to practice resolving conflicts, you can attempt that now. If you successfully resolve the conflicts, both you and your teammate will receive extra credit on this assignment. You can resolve the conflicts on your own, or work with your teammate to walk through the process.

In this assignment, there may be up to 3 files that conflict.

1. If you have not already, click **Resolve conflicts** -

Fixed Tax Issue #4

 Open


kareemy wants to merge 2 commits into `2-calculate-sales-tax` from `main`

Conversation 0

Commits 2


Checks 0


Files changed 6




kareemy commented now

No description provided.

 kareemy added 2 commits 6 minutes ago

 Fixed Shipping Issues

d1b6f2d

 Merge pull request #3 from kareemy/1-calculating-shipping-costs

Verified 39504c6



 **This branch has conflicts that must be resolved**

Use the [web editor](#) or the [command line](#) to resolve conflicts.

Conflicting files

OrderApp/OrderApp.csproj

OrderApp/Pages/Orders/ReviewOrder.cshtml.cs

lab11.sln

Merge pull request

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

Resolve conflicts

2. The Resolve conflicts window should look like this -

Fixed Tax Issue #4
Resolving conflicts between 2-calculate-sales- and main and committing changes → 2-calculate-sales-

3 conflicting files

- OrderApp.csproj
- ReviewOrder.cshtml.cs
- lab11.sln

```
1 <Project Sdk="Microsoft.NET.Sdk.Web">
2
3 <PropertyGroup>
4 <TargetFramework>net8.0</TargetFramework>
5 <Nullable>enable</Nullable>
6 <ImplicitUsings>enable</ImplicitUsings>
7 </PropertyGroup>
8
9 <ItemGroup>
10 <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="8.0.8">
11 <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
12 </PackageReference>
13 <PackageReference Include="Microsoft.EntityFrameworkCore.Sqlite" Version="8.0.8" />
14 <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="8.0.8" />
15 <PackageReference Include="Microsoft.EntityFrameworkCore.Tools" Version="8.0.8" />
16 <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
17 </PackageReference>
18 </ItemGroup>
19 <PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="8.0.5" />
20 </ItemGroup>
21
22 <ItemGroup>
23 <ProjectReference Include="..\2-calculate-sales-tax\TaxLibrary\TaxLibrary.csproj" />
24 <ProjectReference Include="..\ShippingLibrary\ShippingLibrary.csproj" />
25 </ItemGroup>
26 </Project>
```

Conflicting OrderApp.csproj

Highlighted in yellow and red are the conflicting lines of code. Resolve that conflict by modifying the code to bring in both project references -

Fixed Tax Issue #4
Resolving conflicts between 2-calculate-sales- and main and committing changes → 2-calculate-sales-

3 conflicting files

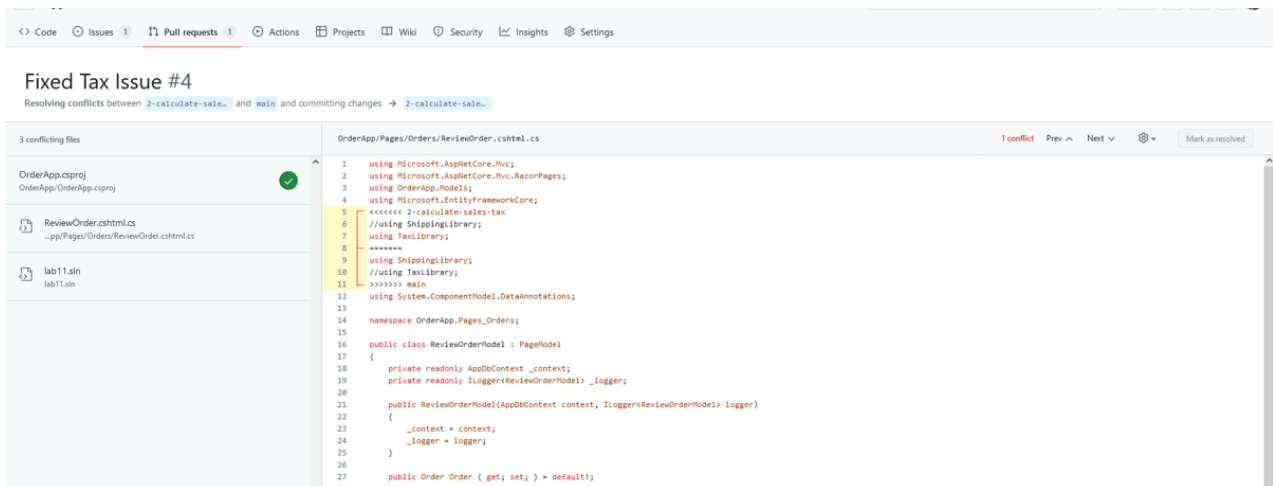
- OrderApp.csproj
- ReviewOrder.cshtml.cs
- lab11.sln

```
1 <Project Sdk="Microsoft.NET.Sdk.Web">
2
3 <PropertyGroup>
4 <TargetFramework>net8.0</TargetFramework>
5 <Nullable>enable</Nullable>
6 <ImplicitUsings>enable</ImplicitUsings>
7 </PropertyGroup>
8
9 <ItemGroup>
10 <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="8.0.8">
11 <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
12 </PackageReference>
13 <PackageReference Include="Microsoft.EntityFrameworkCore.Sqlite" Version="8.0.8" />
14 <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="8.0.8" />
15 <PackageReference Include="Microsoft.EntityFrameworkCore.Tools" Version="8.0.8" />
16 <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
17 </PackageReference>
18 </ItemGroup>
19 <PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="8.0.5" />
20 </ItemGroup>
21
22 <ItemGroup>
23 <ProjectReference Include="..\2-calculate-sales-tax\TaxLibrary\TaxLibrary.csproj" />
24 <ProjectReference Include="..\ShippingLibrary\ShippingLibrary.csproj" />
25 </ItemGroup>
26 </Project>
```

Resolved OrderApp.csproj

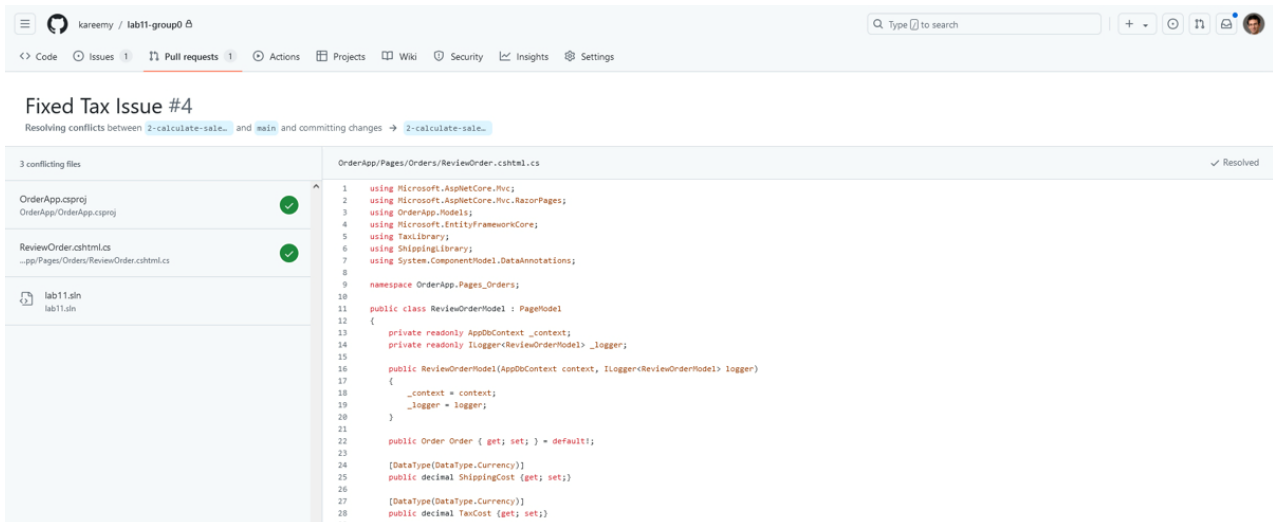
Click **Mark as resolved**.

3. Click **ReviewOrder.cshtml.cs** to resolve that conflict. The conflicting file should look like this -



Conflicting ReviewOrder.cshtml.cs

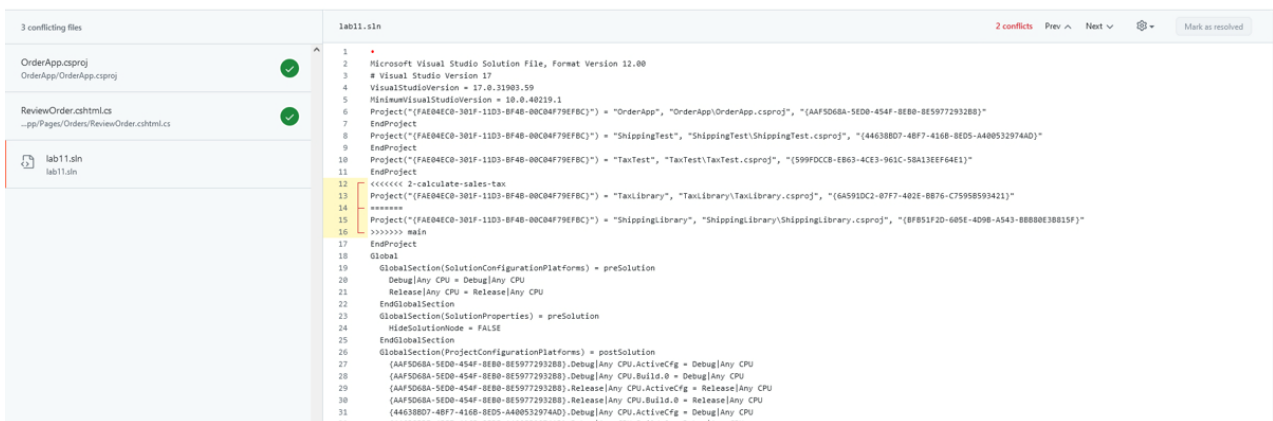
Resolve that conflict by uncommenting both using directives -



Resolved ReviewOrder.cshtml.cs

Click **Mark as resolved**.

4. Click **lab11.sln** to resolve that conflict. The conflicting file should look like this -



```
33 (44638807-48F7-4168-BE05-AA08532974AD).Release[any CPU.ActiveCf = Release[any CPU
34 (44638807-48F7-4168-BE05-AA08532974AD).Release[any CPU.Build.0 = Release[any CPU
35 (599FDCCB-E663-4CE3-961C-SBA13EEF64E1).Debug[any CPU.ActiveCf = Debug[any CPU
36 (599FDCCB-E663-4CE3-961C-SBA13EEF64E1).Debug[any CPU.Build.0 = Debug[any CPU
37 (599FDCCB-E663-4CE3-961C-SBA13EEF64E1).Release[any CPU.ActiveCf = Release[any CPU
38 (599FDCCB-E663-4CE3-961C-SBA13EEF64E1).Release[any CPU.Build.0 = Release[any CPU
39 <<<<<< 2-calculate-sales-tax
40 (6A591DC2-07F7-402E-B876-CT5958593421).Debug[any CPU.ActiveCf = Debug[any CPU
41 (6A591DC2-07F7-402E-B876-CT5958593421).Debug[any CPU.Build.0 = Debug[any CPU
42 (6A591DC2-07F7-402E-B876-CT5958593421).Release[any CPU.ActiveCf = Release[any CPU
43 (6A591DC2-07F7-402E-B876-CT5958593421).Release[any CPU.Build.0 = Release[any CPU
44 <<<<<<<
45 (BF851F2D-605E-4D08-A543-B8880E38815F).Debug[any CPU.ActiveCf = Debug[any CPU
46 (BF851F2D-605E-4D08-A543-B8880E38815F).Debug[any CPU.Build.0 = Debug[any CPU
47 (BF851F2D-605E-4D08-A543-B8880E38815F).Release[any CPU.ActiveCf = Release[any CPU
48 (BF851F2D-605E-4D08-A543-B8880E38815F).Release[any CPU.Build.0 = Release[any CPU
49 >>>>>> main
50 EndGlobalSection
51 EndGlobal
52
```

Conflicting lab11.sln

Resolve that conflict by referencing both projects and removing unnecessary code below -

3 conflicting files

OrderApp.csproj
OrderApp\OrderApp.csproj

ReviewOrder.cshtml.cs
..pp\Pages\Orders\ReviewOrder.cshtml.cs

lab11.sln
lab11.sln

lab11.sln

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

Microsoft Visual Studio Solution File, Format Version 12.00

Visual Studio Version 17

VisualStudioVersion = 17.0.31903.59

MinimumVisualStudioVersion = 10.0.40219.1

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "OrderApp", "OrderApp\OrderApp.csproj", "{AAF5068A-5ED0-454F-BE80-BE59772932B8}"

EndProject

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "ShippingTest", "ShippingTest\ShippingTest.csproj", "{44638807-48F7-4168-BE05-AA08532974AD}"

EndProject

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "TaxTest", "TaxTest\TaxTest.csproj", "{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}"

EndProject

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "TaxLibrary", "TaxLibrary\TaxLibrary.csproj", "{6A591DC2-07F7-402E-B876-CT5958593421}"

EndProject

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "ShippingLibrary", "ShippingLibrary\ShippingLibrary.csproj", "{BF851F2D-605E-4D08-A543-B8880E38815F}"

EndProject

Global

GlobalSection(SolutionConfigurationPlatforms) = preSolution

Debug|any CPU = Debug|any CPU

Release|any CPU = Release|any CPU

EndGlobalSection

GlobalSection(SolutionProperties) = preSolution

HideSolutionNode = FALSE

EndGlobalSection

GlobalSection(ProjectConfigurationPlatforms) = postSolution

{AAF5068A-5ED0-454F-BE80-BE59772932B8}.Debug|any CPU.ActiveCf = Debug|any CPU

{AAF5068A-5ED0-454F-BE80-BE59772932B8}.Debug|any CPU.Build.0 = Debug|any CPU

{AAF5068A-5ED0-454F-BE80-BE59772932B8}.Release|any CPU.ActiveCf = Release|any CPU

{AAF5068A-5ED0-454F-BE80-BE59772932B8}.Release|any CPU.Build.0 = Release|any CPU

{44638807-48F7-4168-BE05-AA08532974AD}.Debug|any CPU.ActiveCf = Debug|any CPU

{44638807-48F7-4168-BE05-AA08532974AD}.Debug|any CPU.Build.0 = Debug|any CPU

{44638807-48F7-4168-BE05-AA08532974AD}.Release|any CPU.ActiveCf = Release|any CPU

{44638807-48F7-4168-BE05-AA08532974AD}.Release|any CPU.Build.0 = Release|any CPU

{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}.Debug|any CPU.ActiveCf = Debug|any CPU

{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}.Debug|any CPU.Build.0 = Debug|any CPU

{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}.Release|any CPU.ActiveCf = Release|any CPU

{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}.Release|any CPU.Build.0 = Release|any CPU

{6A591DC2-07F7-402E-B876-CT5958593421}.Debug|any CPU.ActiveCf = Debug|any CPU

{6A591DC2-07F7-402E-B876-CT5958593421}.Debug|any CPU.Build.0 = Debug|any CPU

{6A591DC2-07F7-402E-B876-CT5958593421}.Release|any CPU.ActiveCf = Release|any CPU

{6A591DC2-07F7-402E-B876-CT5958593421}.Release|any CPU.Build.0 = Release|any CPU

EndGlobalSection

EndGlobal

Resolved lab11.sln

Click **Mark as resolved.**

5. Once all three files have been resolved, you can merge the pull request.

kareemy / lab11-group0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Fixed Tax Issue #4

Resolving conflicts between 2-calculate-sale... and main and committing changes → 2-calculate-sale...

This merge commit will be associated with kdana@vtamu.edu. Commit merge

3 conflicting files

OrderApp.csproj
OrderApp\OrderApp.csproj

ReviewOrder.cshtml.cs
..pp\Pages\Orders\ReviewOrder.cshtml.cs

lab11.sln
lab11.sln

lab11.sln

1

2

3

4

5

6

7

8

9

10

11

12

13

14

Microsoft Visual Studio Solution File, Format Version 12.00

Visual Studio Version 17

VisualStudioVersion = 17.0.31903.59

MinimumVisualStudioVersion = 10.0.40219.1

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "OrderApp", "OrderApp\OrderApp.csproj", "{AAF5068A-5ED0-454F-BE80-BE59772932B8}"

EndProject

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "ShippingTest", "ShippingTest\ShippingTest.csproj", "{44638807-48F7-4168-BE05-AA08532974AD}"

EndProject

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "TaxTest", "TaxTest\TaxTest.csproj", "{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}"

EndProject

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "TaxLibrary", "TaxLibrary\TaxLibrary.csproj", "{6A591DC2-07F7-402E-B876-CT5958593421}"

EndProject

Project("FAE04ECB-301F-11D3-BF4B-08C04F79EFBC") = "ShippingLibrary", "ShippingLibrary\ShippingLibrary.csproj", "{BF851F2D-605E-4D08-A543-B8880E38815F}"

EndProject

Global

GlobalSection(SolutionConfigurationPlatforms) = preSolution

Debug|any CPU = Debug|any CPU

Release|any CPU = Release|any CPU

EndGlobalSection

GlobalSection(SolutionProperties) = preSolution

HideSolutionNode = FALSE

EndGlobalSection

GlobalSection(ProjectConfigurationPlatforms) = postSolution

{AAF5068A-5ED0-454F-BE80-BE59772932B8}.Debug|any CPU.ActiveCf = Debug|any CPU

{AAF5068A-5ED0-454F-BE80-BE59772932B8}.Debug|any CPU.Build.0 = Debug|any CPU

{AAF5068A-5ED0-454F-BE80-BE59772932B8}.Release|any CPU.ActiveCf = Release|any CPU

{AAF5068A-5ED0-454F-BE80-BE59772932B8}.Release|any CPU.Build.0 = Release|any CPU

{44638807-48F7-4168-BE05-AA08532974AD}.Debug|any CPU.ActiveCf = Debug|any CPU

{44638807-48F7-4168-BE05-AA08532974AD}.Debug|any CPU.Build.0 = Debug|any CPU

{44638807-48F7-4168-BE05-AA08532974AD}.Release|any CPU.ActiveCf = Release|any CPU

{44638807-48F7-4168-BE05-AA08532974AD}.Release|any CPU.Build.0 = Release|any CPU

{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}.Debug|any CPU.ActiveCf = Debug|any CPU

{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}.Debug|any CPU.Build.0 = Debug|any CPU

{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}.Release|any CPU.ActiveCf = Release|any CPU

{599FDCCB-E663-4CE3-961C-SBA13EEF64E1}.Release|any CPU.Build.0 = Release|any CPU

{6A591DC2-07F7-402E-B876-CT5958593421}.Debug|any CPU.ActiveCf = Debug|any CPU

{6A591DC2-07F7-402E-B876-CT5958593421}.Debug|any CPU.Build.0 = Debug|any CPU

{6A591DC2-07F7-402E-B876-CT5958593421}.Release|any CPU.ActiveCf = Release|any CPU

{6A591DC2-07F7-402E-B876-CT5958593421}.Release|any CPU.Build.0 = Release|any CPU

EndGlobalSection

EndGlobal

```

15 EndProject
16 Global
17 GlobalSection(SolutionConfigurationPlatforms) = preSolution
18 Debug|Any CPU = Debug|Any CPU
19 Release|Any CPU = Release|Any CPU
20 EndGlobalSection
21 GlobalSection(SolutionProperties) = preSolution
22 HideSolutionNode = FALSE
23 EndGlobalSection
24 GlobalSection(ProjectConfigurationPlatforms) = postSolution
25 (AAF5D68A-5ED0-454F-BE80-BE59772932B8).Debug|Any CPU.ActiveCfg = Debug|Any CPU
26 (AAF5D68A-5ED0-454F-BE80-BE59772932B8).Debug|Any CPU.Build.0 = Debug|Any CPU
27 (AAF5D68A-5ED0-454F-BE80-BE59772932B8).Release|Any CPU.ActiveCfg = Release|Any CPU
28 (AAF5D68A-5ED0-454F-BE80-BE59772932B8).Release|Any CPU.Build.0 = Release|Any CPU
29 (44638B07-4B77-416B-BED5-A480532974AD).Debug|Any CPU.ActiveCfg = Debug|Any CPU
30 (44638B07-4B77-416B-BED5-A480532974AD).Debug|Any CPU.Build.0 = Debug|Any CPU
31 (44638B07-4B77-416B-BED5-A480532974AD).Release|Any CPU.ActiveCfg = Release|Any CPU
32 (44638B07-4B77-416B-BED5-A480532974AD).Release|Any CPU.Build.0 = Release|Any CPU
33 (599FDCCB-E863-4CE3-961C-5BA13EEF64E1).Debug|Any CPU.ActiveCfg = Debug|Any CPU
34 (599FDCCB-E863-4CE3-961C-5BA13EEF64E1).Debug|Any CPU.Build.0 = Debug|Any CPU
35 (599FDCCB-E863-4CE3-961C-5BA13EEF64E1).Release|Any CPU.ActiveCfg = Release|Any CPU
36 (599FDCCB-E863-4CE3-961C-5BA13EEF64E1).Release|Any CPU.Build.0 = Release|Any CPU
37 (6A591DC2-87F7-4B2E-BB76-CT5958593421).Debug|Any CPU.ActiveCfg = Debug|Any CPU
38 (6A591DC2-87F7-4B2E-BB76-CT5958593421).Debug|Any CPU.Build.0 = Debug|Any CPU
39 (6A591DC2-87F7-4B2E-BB76-CT5958593421).Release|Any CPU.ActiveCfg = Release|Any CPU
40 (6A591DC2-87F7-4B2E-BB76-CT5958593421).Release|Any CPU.Build.0 = Release|Any CPU
41 EndGlobalSection
42 EndGlobal
43

```

Click **Commit merge**.

- After clicking **Commit merge**, you can click **Merge pull request** and then **Confirm merge**.

Grading Rubric

This is the grading rubric for the Lab 11 Assignment.

Task	Points
Complete Pre-Survey	10
Create an Issue on GitHub	5
Assign the issue to yourself	3
Create a new branch on GitHub	10
Checkout your branch	5
Write all your code in your branch (Do not code directly in the main branch)	5
Create a new class library with <code>dotnet new classlib</code>	5
Add your class library to the main solution file with <code>dotnet sln add</code>	5
Add a reference to your class library in OrderApp	5
Write your code in the class library	5
Uncomment the correct using directive in ReviewOrder.cshtml.cs	3
Uncomment the correct line of code in ReviewOrder.cshtml.cs	3
Test your code with <code>dotnet test</code>	3
Push your changes back to GitHub in the correct branch	3
Create a pull request on GitHub	10
Merge pull request on GitHub or recognize "Can't automatically merge" message	5
Extra Credit: Resolve Conflicts if applicable	+5 points
Complete Post-Survey	15
Total:	100