

# DevOps Fundamentals

---

Lab 02: DevTest – Source Control Management

Lab Manual

## **Conditions and Terms of Use**

### **Microsoft Confidential**

This training package is proprietary and confidential and is intended only for uses described in the training materials. Content and software is provided to you under a Non-Disclosure Agreement and cannot be distributed. Copying or disclosing all or any portion of the content and/or software included in such packages is strictly prohibited.

The contents of this package are for informational and training purposes only and are provided "as is" without warranty of any kind, whether express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

Training package content, including URLs and other Internet Web site references, is subject to change without notice. Because Microsoft must respond to changing market conditions, the content should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

## Copyright and Trademarks

© 2017 Microsoft Corporation. All rights reserved.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

For more information, see Use of Microsoft Copyrighted Content at

<http://www.microsoft.com/en-us/legal/intellectualproperty/Permissions/default.aspx>

DirectX, Hyper-V, Internet Explorer, Microsoft, Outlook, OneDrive, SQL Server, Windows, Microsoft Azure, Windows PowerShell, Windows Server, Windows Vista, and Zune are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other Microsoft products mentioned herein may be either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. All other trademarks are property of their respective owners

## Contents

<b>LAB 2: SOURCE CONTROL MANAGEMENT .....</b>	<b>6</b>
EXERCISE 1: CREATE A SAMPLE PROJECT .....	7
EXERCISE 2: CREATE A REMOTE TOPIC BRANCH.....	14
EXERCISE 3: CREATE A LOCAL TOPIC BRANCH .....	16
EXERCISE 4: COMPLETE THE WORK ITEM.....	19
EXERCISE 5: CREATE A PULL REQUEST.....	23
EXERCISE 6: APPROVE THE PULL REQUEST AND CLEAN UP LOCAL BRANCHES .....	25
EXERCISE 7: ADD DOCKER SUPPORT .....	29

## Lab 2: Source Control Management

### Introduction

In this lab, you will understand the basics of source control management using Git and AZURE DEVOPS. Also, you will start to build the foundation of a continuous integration and continuous delivery strategy.

### Objectives

After completing this lab, you will be able to:

- Create the initial code base for your project.
- Create topic branches.
- Complete Azure Boards work items.
- Create and approve pull requests.

### Prerequisites

None

### Estimated Time to Complete This Lab

90 minutes

### For More Information

**Source Control with AZURE DEVOPS:** <https://docs.microsoft.com/en-us/Azure/DevOps/user-guide/source-control>

**Git:** <https://git-scm.com/>

# Exercise 1: Create a sample project

## Introduction

In this exercise, you will create a sample project.

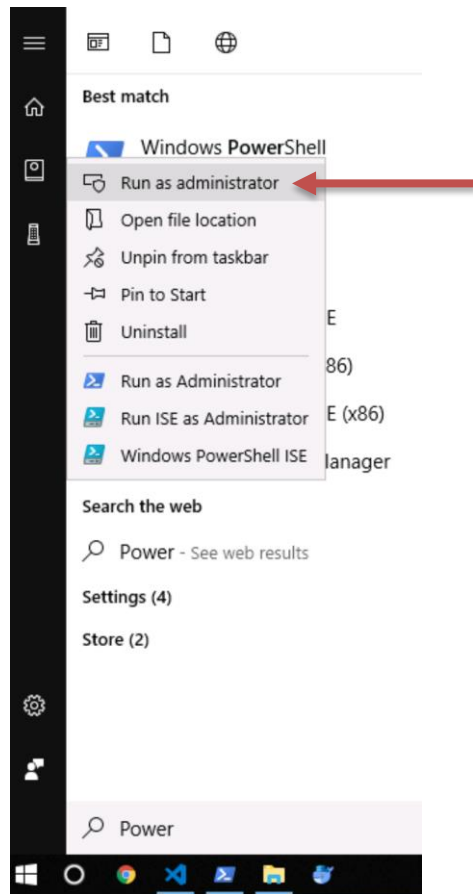
## Objectives

After completing this lab, you will be able to:

- Create a sample project with a Git repository

## Tasks

1. Open a PowerShell window in Administrator mode:



2. Once the PowerShell window is open type the following command:

`Set-ExecutionPolicy RemoteSigned`

When asked to confirm the execution of the command, type “A”.

```
PS C:\Users\devopsadmin> Set-ExecutionPolicy RemoteSigned
Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose
you to the security risks described in the about_Execution_Policies help topic at
https://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): A
```

3. If this is the first time you're running git on this development machine run these two commands.

**(Use the same email address you use to authenticate to your Azure DevOps instance):**

```
git config --global user.email "you@example.com"
```

```
git config --global user.name "Your Name"
```

4. Follow the following steps to create a folder, where we will initiate a git repository for the sample project:

```
cd / ; mkdir hol ; cd hol ; mkdir ModernApiAppSolution; cd ModernApiAppSolution
```

```
git init
```

```
mkdir ModernApiApp ; cd ModernApiApp
```

```
PS C:\> cd / ; mkdir hol ; cd hol ; mkdir ModernApiAppSolution; cd ModernApiAppSolution

Directory: C:\

Mode                LastWriteTime         Length Name
----                -
d-----          9/27/2018   7:09 PM         hol

Directory: C:\hol

Mode                LastWriteTime         Length Name
----                -
d-----          9/27/2018   7:09 PM      ModernApiAppSolution

PS C:\hol\ModernApiAppSolution> git init
Initialized empty Git repository in C:/hol/ModernApiAppSolution/.git/
PS C:\hol\ModernApiAppSolution> mkdir ModernApiApp ; cd ModernApiApp

Directory: C:\hol\ModernApiAppSolution

Mode                LastWriteTime         Length Name
----                -
d-----          9/27/2018   7:10 PM         ModernApiApp

PS C:\hol\ModernApiAppSolution\ModernApiApp>
```

Use the following command to create a new .Net core Web Api project:

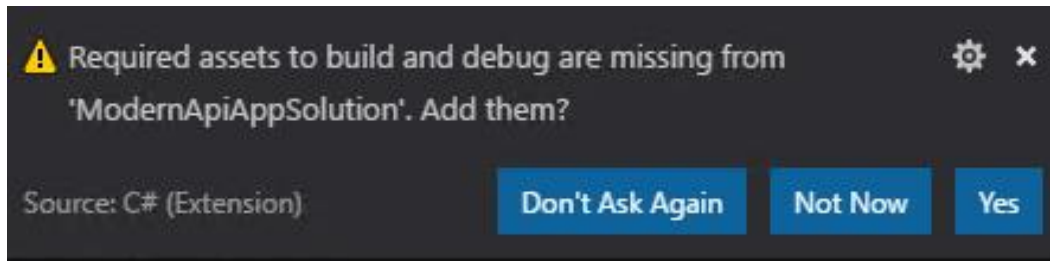
```
dotnet new webapi
```

5. Use the following command to open the project in Visual Studio Code (feel free to use another code editor)

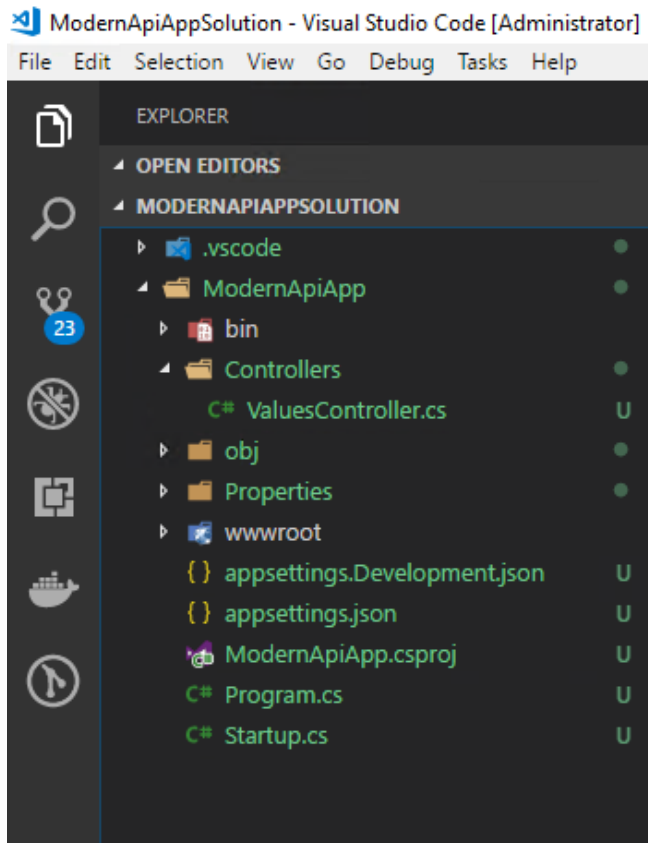
```
cd ..
```

```
Code .
```

- VS Code might ask you to add some “assets” required for building and debugging the solution project, click “Yes”:

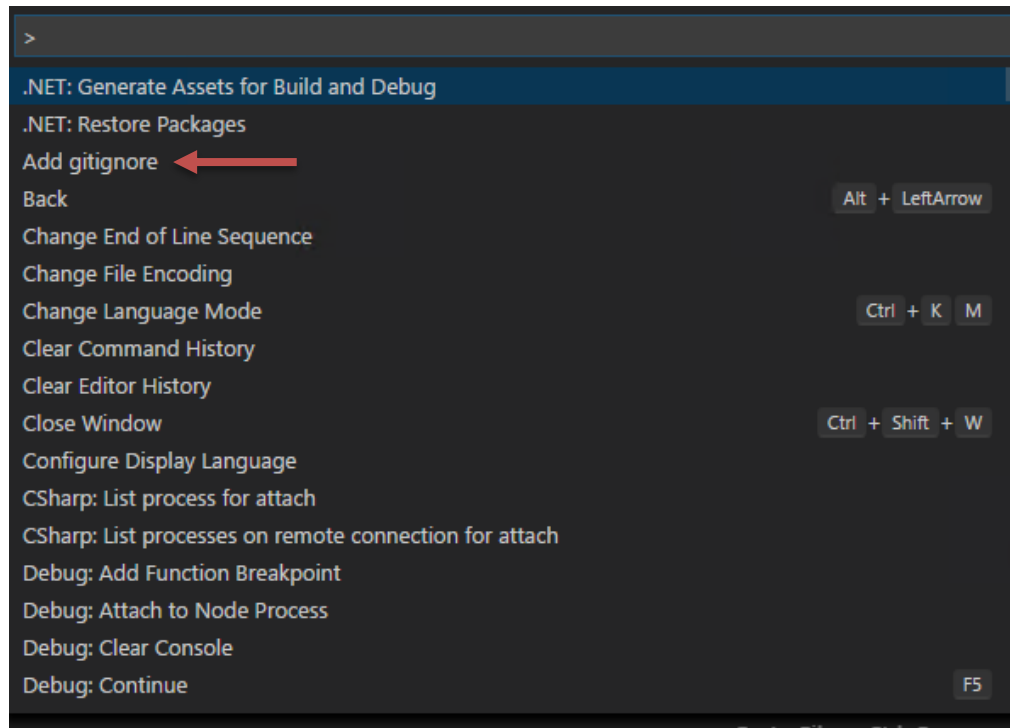


- Expand the “ModernApiApp” folder in VS Code to see all the files in your project:

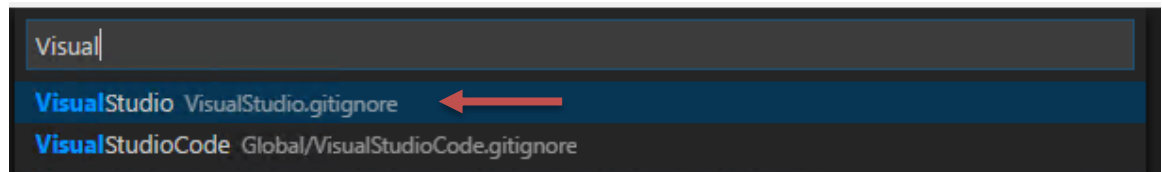


- Press “F1” to launch the VS code commands window:

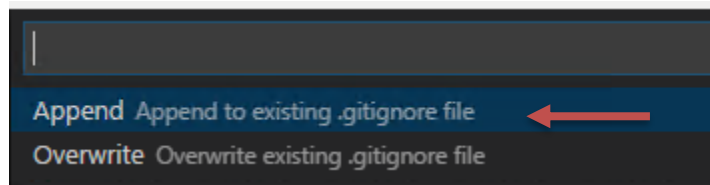




9. Type “Add gitignore” then choose “VisualStudio”:

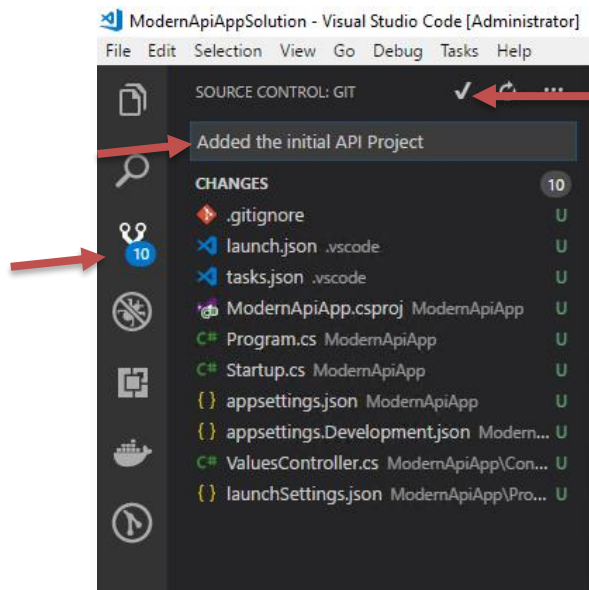


10. Repeat the same step and choose “VisualStudioCode”, then choose “Append”:



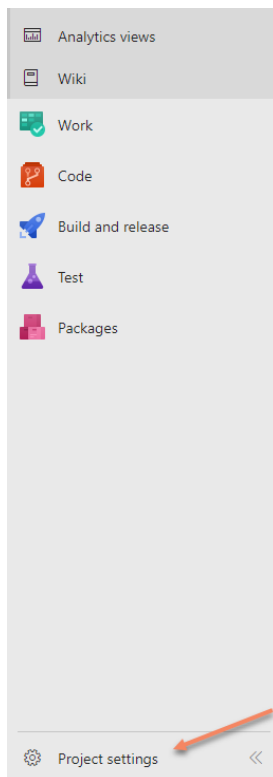
11. Save all your files.

12. Go to the integrated source control management window in VS Code, type a message “Added the initial api project” then commit your code:

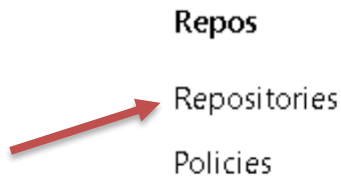


We will now publish this repository to a git repository in the same Azure DevOps Services project you created during Lab 01.

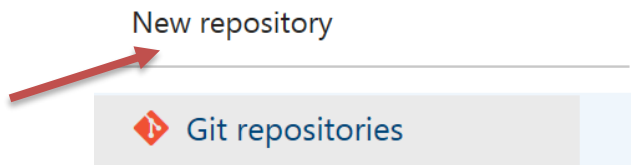
13. From your web browser, go to Azure DevOps Services “[Project Settings](#)” for the “[hol](#)” project that you created in Lab 01:



14. Scroll down to the “Repos” section, then click on Repositories:



15. Click on the “New repository” button:



16. Name the repository “ModernApiRepository”, then click “Create”:

Create a new repository ×

Type

Git ▼

Repository name \*

ModernApiRepository ×

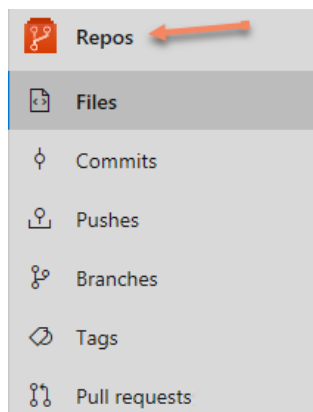
☐ Add a README to describe your repository

Add a .gitignore:

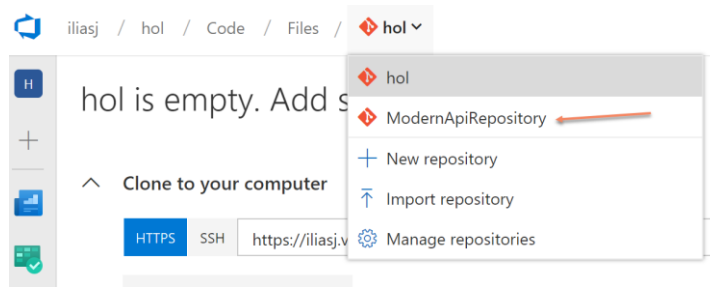
None ▼

Create Cancel

17. Go to the “Repos” hub in Azure DevOps Services:



18. Switch to the “Files” screen to show the “ModernApiRepository” files:



19. Copy the commands to push your local code to the Azure DevOps Service repository:

or push an existing repository from command line

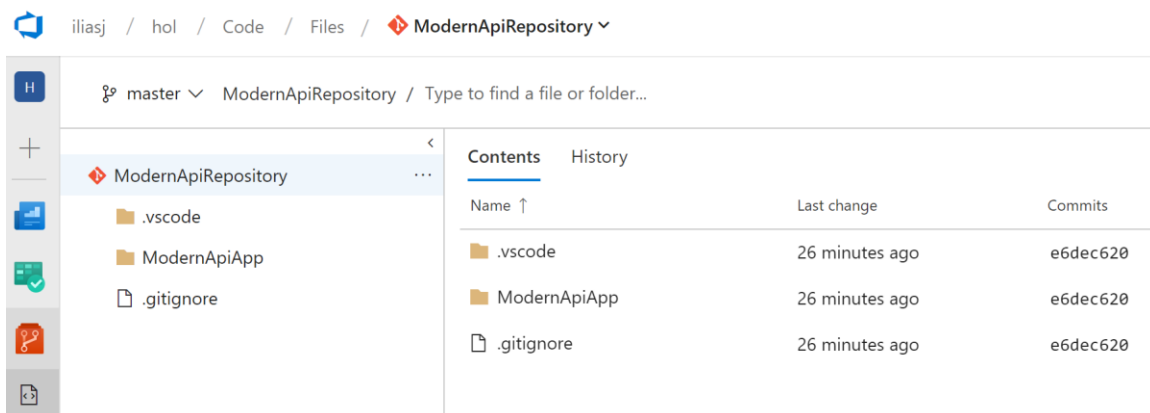


20. Go back to the PowerShell window and use the following commands. **Make sure you are positioned at the solution root folder (ModernApiAppSolution) paste the commands you copied from AZURE DEVOPS:**

```
git remote add origin
https://[yourinstance].visualstudio.com/hol/_git/ModernApiRepository

git push -u origin --all
```

21. From your web browser, go to the “Repos – Files” screen in Azure DevOps Services. Refresh the page and you should be able to see all the files you just pushed to the remote repository:



## Exercise 2: Create a remote topic branch

### Introduction

This exercise and the remaining ones in this lab will help us focus on a process that allows parallel development and team collaboration, ensures quality, and embraces continuous integration and continuous delivery.

### Objectives

After completing this lab, you will be able to:

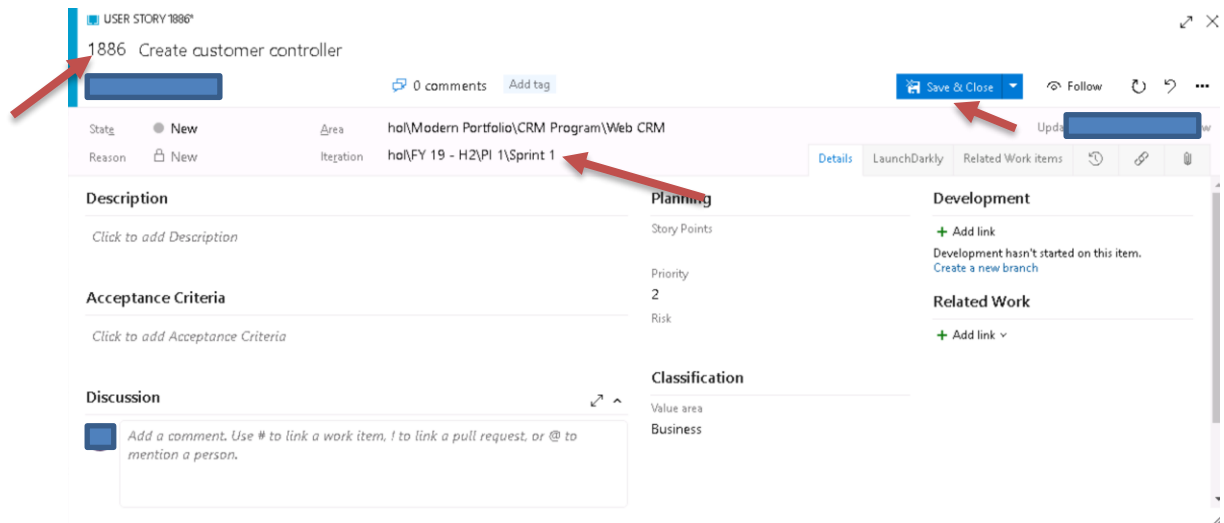
- Create a remote topic branch and link it to an AZURE DEVOPS work item

### Prerequisites

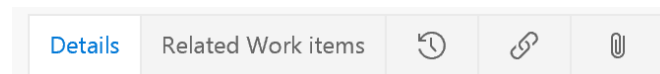
None

### Tasks

1. From the “[Web CRM Team](#)” board in Azure DevOps, create a new story:
  - “[Create Customer Controller](#)”
  - Set its iteration to “[Sprint 1](#)”
  - Assign it to yourself



- From the work item form click on the “[Create a new branch](#)” link:



Details Related Work items

## Development

+ Add link

Development hasn't started on this item.

[Create a new branch](#)

- Name the topic branch using the following convention “[topics/tb-123](#)” **where the suffix number refers to the work item Id**. This makes it easy to identify what the source work item for any branch is:

## Create a branch

Name

topics/tb-1886

Based on

ModernApiRepository

master

Work items to link

Search work items by ID or title

1886 Create customer controller  
Updated 3 minutes ago, New

Create branch

Cancel

## Exercise 3: Create a local topic branch

### Introduction

### Objectives

After completing this lab, you will be able to:

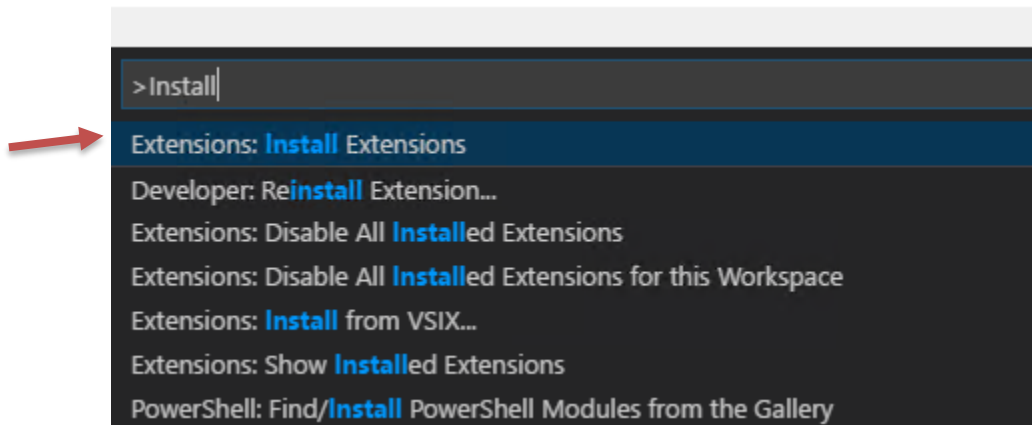
- Create a local topic branch

### Prerequisites

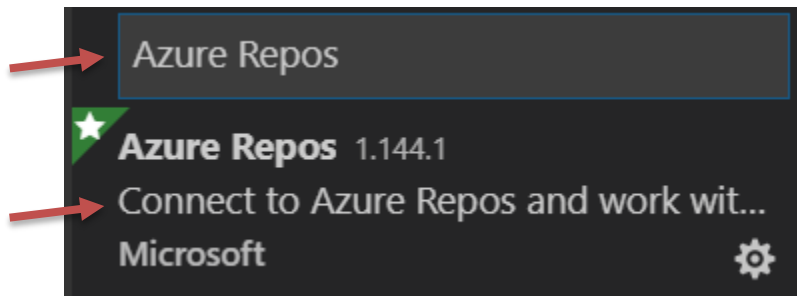
None

### Tasks

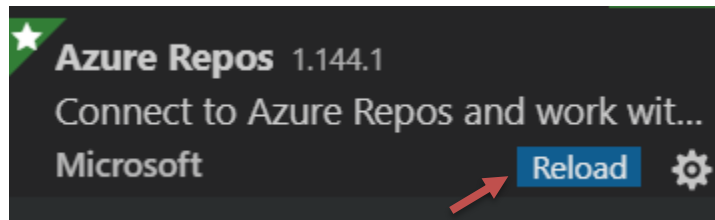
1. From VS Code Use the “F1” to launch the VS Code command palette, and type “Install”, then select “**Extensions: Install Extensions**”. If this extension is already installed on your machine you can skip to step 4.



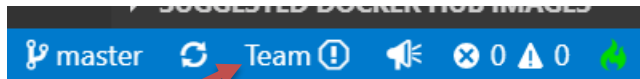
2. In the extensions search bar type “Azure Repos” then install the first extension:



3. Reload VS Code:

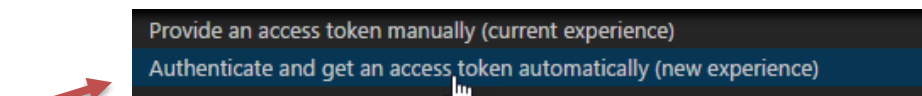


4. Click on the “Team” icon at the Visual Studio Notification tray:



5. Select the following option to authenticate to AZURE DEVOPS from VS Code.

(In case you are not getting the authentication prompt, make sure you close any open notification in VS Code first) :



6. Follow the prompts to get authenticated:

## Device Login

Enter the code that you received from the application on your device

AWSRGNTB

## Visual Studio Team Services

Click Cancel if this isn't the application you were trying to sign in to on your device.

Continue

Cancel

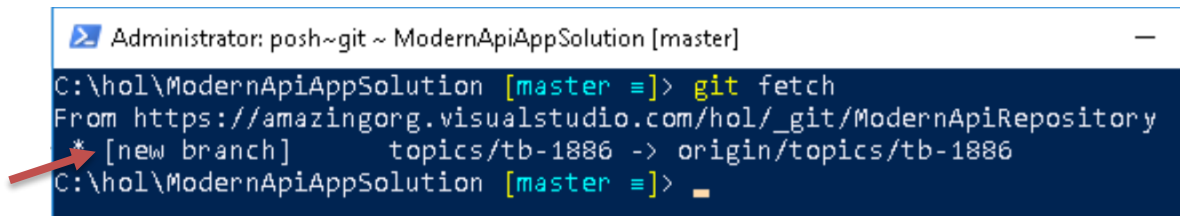
## Visual Studio Team Services

You have signed in to the Visual Studio Team Services application on your device. You may now close this window.



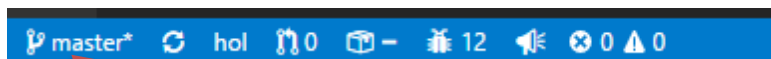
7. From the PowerShell window you have been using, use the following command to update your local git repository:

`git fetch`

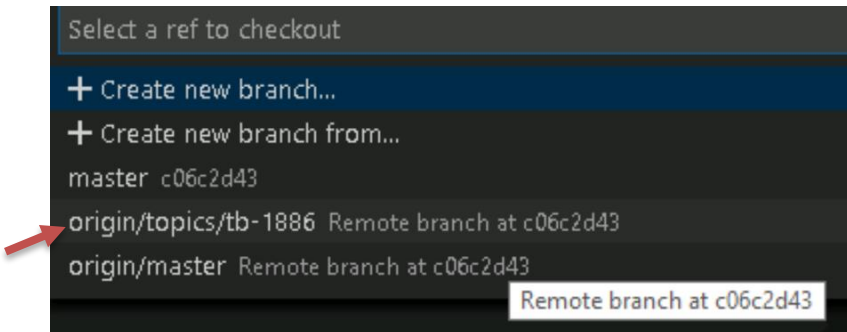


```
Administrator: posh~git ~ ModernApiAppSolution [master]
C:\hol\ModernApiAppSolution [master =>]> git fetch
From https://amazingorg.visualstudio.com/hol/_git/ModernApiRepository
* [new branch]      topics/tb-1886 -> origin/topics/tb-1886
C:\hol\ModernApiAppSolution [master =>]>
```

8. Click on the branch selector in the VS Code taskbar:



This will show you the remote and local branches that are currently available to you.



9. Click on the remote topic branch. This will create an equivalent local branch:



## Exercise 4: Complete the work item

### Introduction

### Objectives

After completing this lab, you will be able to:

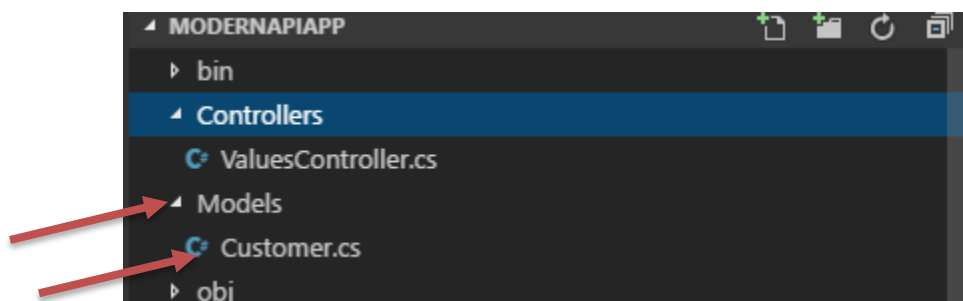
- Complete a work item

### Prerequisites

None

### Tasks

1. From VS Code, add a folder to “[ModernApiApp](#)” project and name it “[Models](#)” at the root of the ModernApiApp folder.
2. In that folder create a new C# class and name is “[Customer](#)” (Customer.cs)

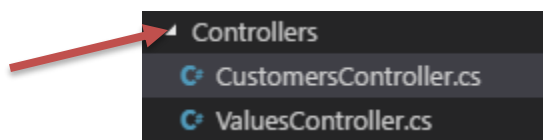


3. Copy the following code to the “[Customer](#)” class:

```
namespace ModernApiApp.Models
{
    public class Customer
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Address { get; set; }
    }
}
```

(the code file is available in the provided labs folder under “[\Labs\Lab 02](#)”)

4. Under the “[Controllers](#)” folder add a new class “[CustomersController.cs](#)”



5. Copy the following code to the “CustomersController” class:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Mvc;
using ModernApiApp.Models;

namespace ModernApiApp.Controllers
{
    [Route("api/[controller]")]
    public class CustomersController : ControllerBase
    {
        // GET api/customers
        [HttpGet]
        public IEnumerable<Customer> Get()
        {
            var customers = new List<Customer>();
            for (int i = 0; i < 10; i++)
            {
                var customer = new Customer {Id= i+1, Name =
                $"Customer {i+1}" , Address = $"{(i+1) * 100} Main St" };
                customers.Add(customer);
            }
            return customers.ToArray();
        }
    }
}
```

(the code is available in the provided labs folder under “\Labs\Lab 02”)

6. Save all the changes.
7. Switch to the PowerShell window and run the following two commands:

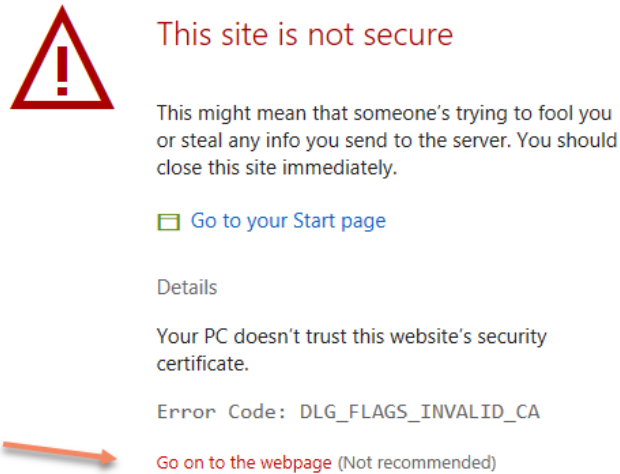
```
cd ModernApiApp
```

```
dotnet build (make sure the application builds correctly)
```

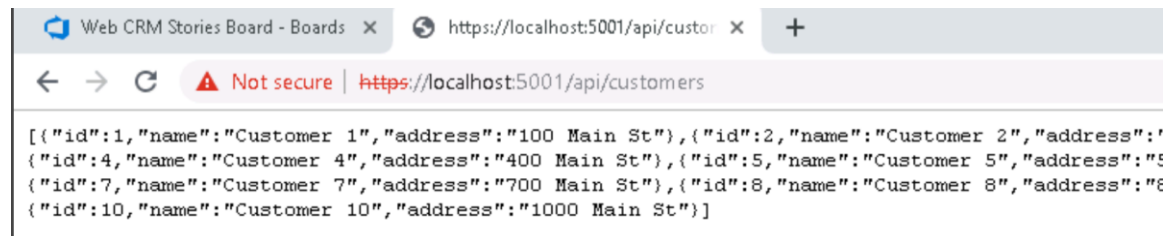
```
dotnet run
```

```
C:\hol\ModernApiAppSolution\ModernApiApp [topics/tb-1886 @ +2 ~0 -0 !]> dotnet run
info: Microsoft.AspNetCore.DataProtection.KeyManagement.XmlKeyManager[0]
      User profile is available. Using 'C:\Users\StudentPC\AppData\Local\ASP.NET
      \DataProtection-Keys' as key repository and Windows DPAPI to encrypt keys at res
      t.
Hosting environment: Development
Content root path: C:\hol\ModernApiAppSolution\ModernApiApp
Now listening on: https://localhost:5001
Now listening on: http://localhost:5000
Application started. Press Ctrl+C to shut down.
```

8. Open your internet browser and go to the following url:  
<https://localhost:5001/api/customers> . You will get a certificate warning since this is a test HTTPs certificate:



9. Click on “Go on to the webpage”. Note that If you are a different browser than Microsoft Edge, it might prompt you differently.
10. You should get the following results:



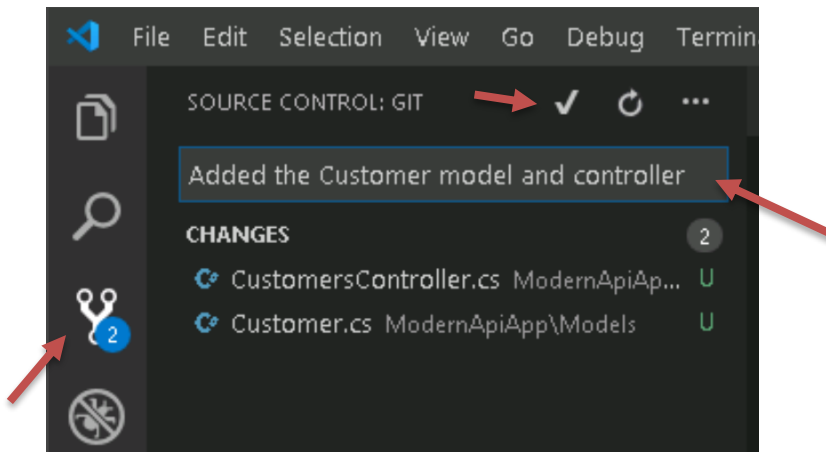
This shows that the HTTP Api you created is correctly working.

11. Stop the execution of the API, by going back to the Powershell window and pressing (CTRL + C):

```
Application is shutting down...
C:\repos\hol\ModernApiAppSolution\ModernApiApp [master ≡ +2 ~0 -0 !]>
```

12. Save all your changes in VS Code.

13. Commit your changes to the local topic branch.



14. Push your changes to the remote topic branch, by clicking on the “synchronize” arrows:



## Exercise 5: Create a pull request

### Introduction

In order to merge the work that has been done on the topic branch with the master branch, we will use a Pull Request. Pull requests let your team give feedback on changes in topic branches before merging the code into the master branch. Reviewers can step through the proposed changes, leave comments, vote and approve or reject the code.

### Objectives

After completing this lab, you will be able to:

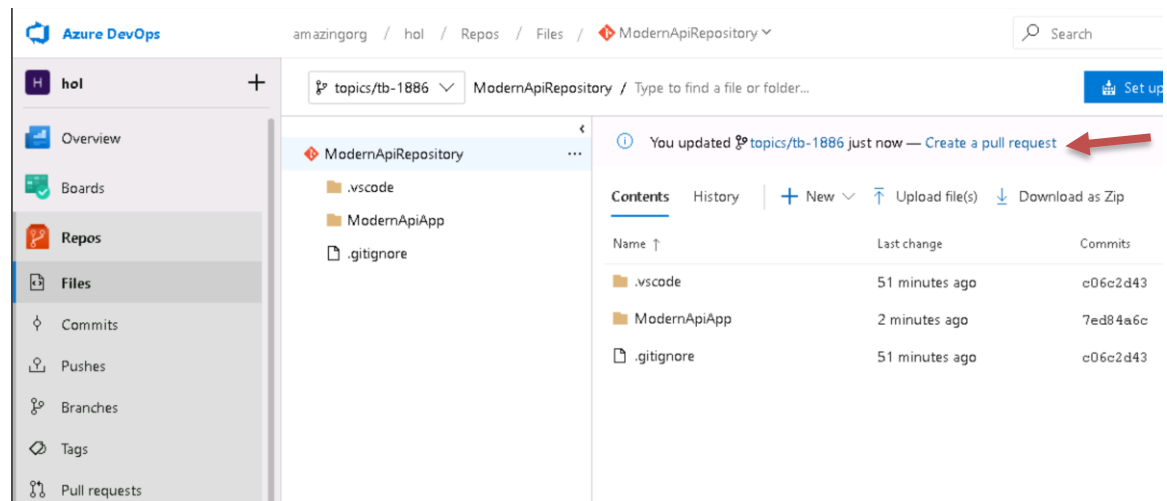
- Create a pull request

### Prerequisites


None






### Tasks

1. Go to “[ModernAPIRepository](#)” files browser in Azure DevOps Repos.
2. Refresh the page if it was already open, then click on the “Create a pull request” link.



3. Enter the pull request information inform the master branch approver that you wish you to merge your code with the master branch:

 New Pull Request

 topics/tb-1886  into  master  

Title \*












Added the Customer model and controller

Add label

Description

Added the Customer model and controller

*Markdown supported.*



Added the Customer model and controller

Reviewers

Search users and groups to add as reviewers

Work Items

Search work items by ID or title

  1886 Create customer controller

4. Click on the “[Create](#)” button to finish the process of creating a pull request.

## Exercise 6: Approve the pull request and clean up local branches

### Introduction

In a real project, you will not be approving your own pull requests. However, in this hands-on lab, we will assume that you are able to do so.

### Objectives

After completing this lab, you will be able to:

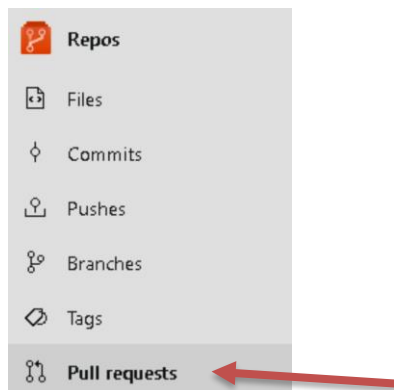
- Approve pull requests in AZURE DEVOPS
- Clean up local branches using git

### Prerequisites

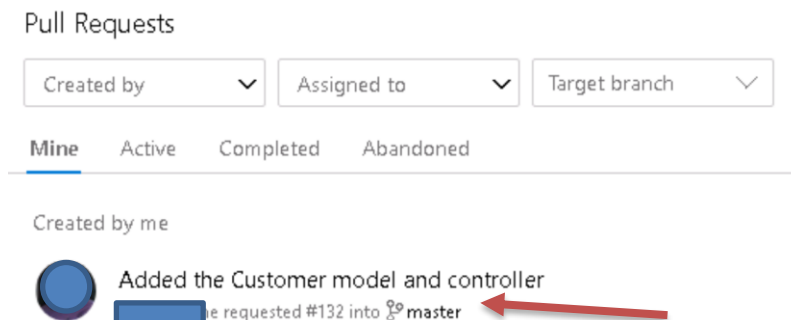
None

### Tasks

1. Go to the “[Pull Requests](#)” list screen in Azure Repos:

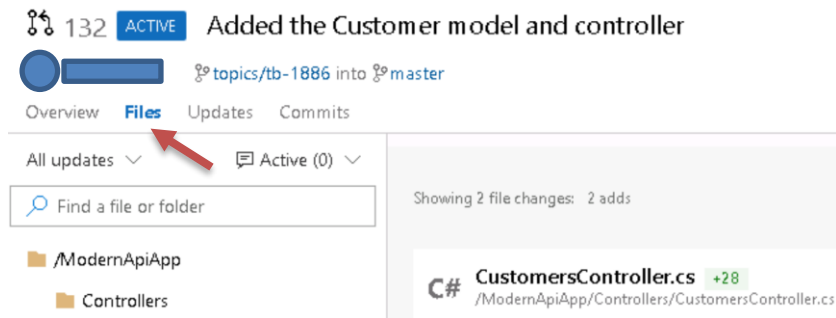


2. Click on the pull requested you created in the previous exercise:

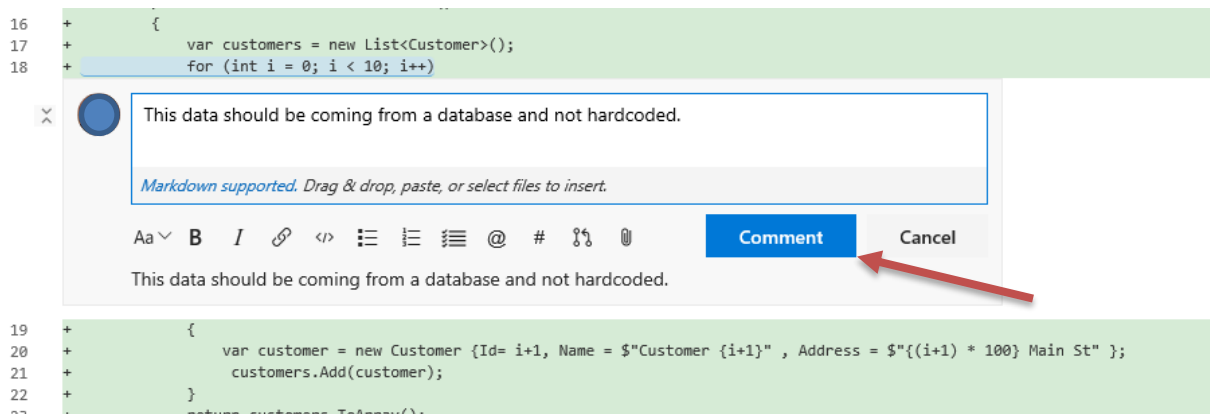




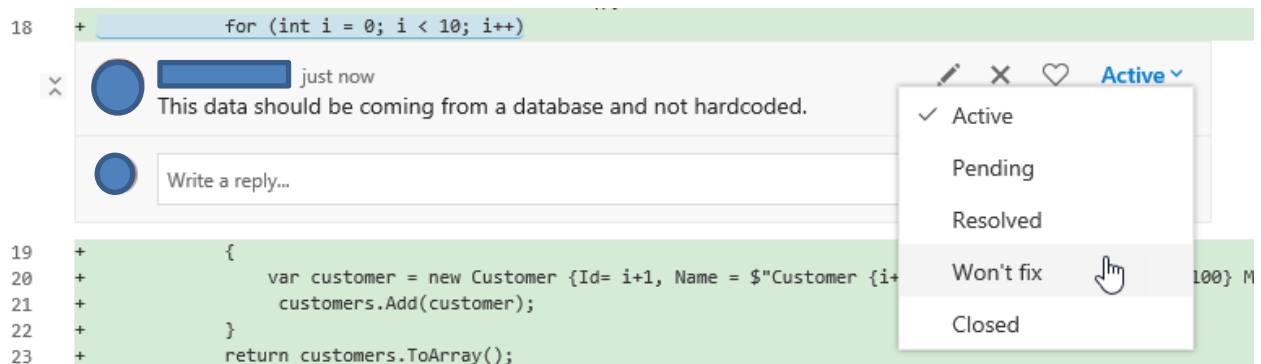
- Click on the “Files” tab:



- Enter a code comment, and then click “Comment”:



The comment you added is now transformed to a task that the pull request submitter can choose to solve or decide not to solve. This communication will be saved with the pull request and can help enhance the quality of the overall code base. For this demo, we will choose not to solve it:



- Approve the pull request and complete it. This will trigger a merge with the master branch.

6. Select the options that allow you to mark the work item as complete and delete the topic branch:

Complete pull request


Merge commit comment

Merged PR 132: Added the Customer model and controller

Added the Customer model and controller  
Related work items: #1886

Merge type

Squash commit



Post-completion options

☒ Complete linked work items after merging

☒ Delete topics/tb-1886 after merging

Complete merge

Cancel

7. Clean up local branches: switch to the local master branch (From the PowerShell window)

`git checkout master`

8. Pull the changes from the remote master branch:

`git pull`

9. Delete the topic branch (use the right branch name):

`git branch -d topics/...`

10. Delete the remote tracking branch:

`git branch -dr origin/topics/...`

```
C:\hol\ModernApiAppSolution\ModernApiApp [topics/tb-1886 = +2 ~0 -0 !]> cd ..
C:\hol\ModernApiAppSolution [topics/tb-1886 =]> git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
C:\hol\ModernApiAppSolution [master =]> git pull
remote: Azure Repos
remote: Found 7 objects to send. (10 ms)
Unpacking objects: 100% (7/7), done.
From https://amazingorg.visualstudio.com/hol/_git/ModernApiRepository
   c06c2d4..0fafaec master    -> origin/master
Updating c06c2d4..0fafaec
Fast-forward
 ModernApiApp/Controllers/CustomersController.cs | 27 ++++++
 ModernApiApp/Models/Customer.cs                 |  9 ++++++
 2 files changed, 36 insertions(+)
 create mode 100644 ModernApiApp/Controllers/CustomersController.cs
 create mode 100644 ModernApiApp/Models/Customer.cs
C:\hol\ModernApiAppSolution [master =]> git branch -d topics/tb-1886
warning: deleting branch 'topics/tb-1886' that has been merged to
        'refs/remotes/origin/topics/tb-1886', but not yet merged to HEAD.
Deleted branch topics/tb-1886 (was 7ed84a6).
C:\hol\ModernApiAppSolution [master =]> git branch -dr origin/topics/tb-1886
Deleted remote-tracking branch origin/topics/tb-1886 (was 7ed84a6).
C:\hol\ModernApiAppSolution [master =]> _
```

## Exercise 7: Add docker support

### Introduction

### Objectives

After completing this lab, you will be able to:

- Add docker support

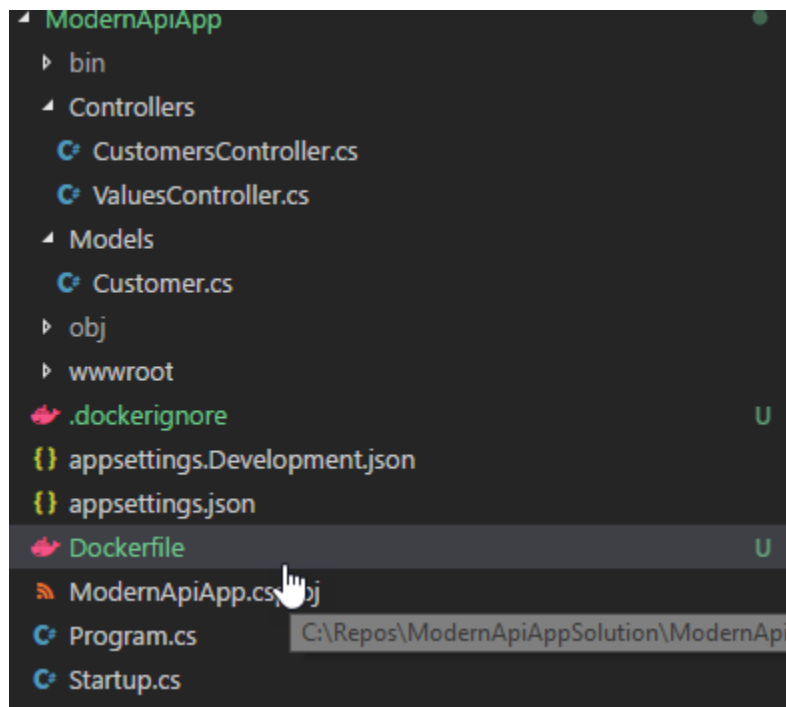
### Prerequisites

None

### Tasks

Without visual assistance, execute the following steps:

- Create a work item and name it: “[Add docker support](#)”
- Add it to the current sprint and commit it.
- Create a topic branch for it using the same convention we followed in the previous exercise.
- From the topic branch:
  - Add a file name “[Dockerfile](#)” to the “[ModernApiApp](#)” project folder



- Copy the following content into it:

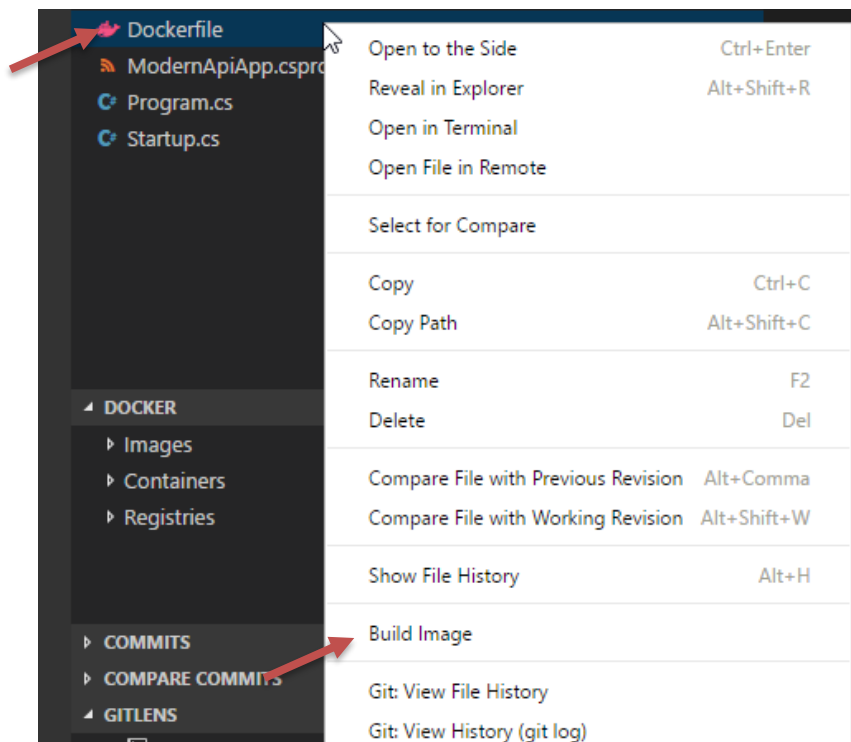
```
FROM mcr.microsoft.com/dotnet/core/aspnet:3.1 AS base
WORKDIR /app
EXPOSE 80
EXPOSE 443

FROM mcr.microsoft.com/dotnet/core/sdk:3.1 AS build
WORKDIR /src
COPY ["ModernApiApp.csproj", "./"]
RUN dotnet restore "./ModernApiApp.csproj"
COPY . .
WORKDIR "/src/."
RUN dotnet build "ModernApiApp.csproj" -c Release -o /app/build

FROM build AS publish
RUN dotnet publish "ModernApiApp.csproj" -c Release -o /app/publish

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "ModernApiApp.dll"]
```

- Use the integrated docker experience to build a docker image. The image will be named “`modernapiapp:latest`”:



- Using a PowerShell window, Run the following command:  
`docker run -d -p 8080:80 --name modernapiapp modernapiapp`
- Open a browser and go to <http://localhost:8080/api/customers> to make sure that your application is returning data from within the docker container.
- Commit the changes to the topic branch.
- Push the changes to the remote topic branch.
- Create and complete a pull request to merge the changes to the master branch.
- Clean up your local branches after you delete the remote topic branch during the pull request.

**\*note if you name your app/folder something different, make sure you use the same names (capital letters included, this changes docker .dll file)**