

**Module 3 Lab: Git Basics**

Template Version: 2.0

**Estimated Time**

30 minutes

**Objectives**

At the end of this lab, you will be able to:

* Push your web app to VSTS with Git.
* Understand Git basics.

**Logon Information**

Please use the Azure Pass provided to you for this lab. Your VM credentials are listed for later use:

* VM Username: super
* VM Password: P@ssw0rd123!

Table of Contents

[Module 3 Lab: Git Basics 3](#_Toc523396238)

[Exercise 1: Push your code to VSTS 3](#_Toc523396239)

[Exercise 2: Making a simple change in Visual Studio – Centralized Workflow 8](#_Toc523396240)

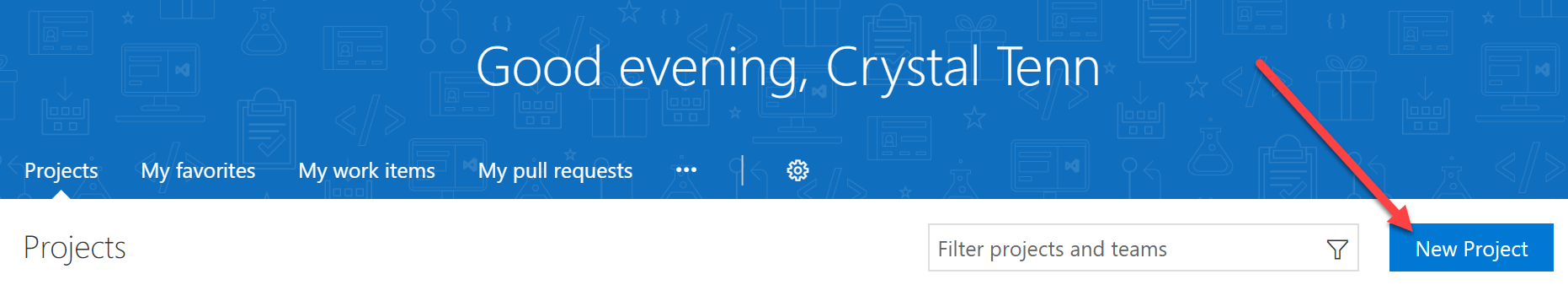
[**Exercise 3: Making a change in Visual Studio – Feature Branch Workflow & Pull Requests** 9](#_Toc523396241)

Module 3 Lab: Git Basics

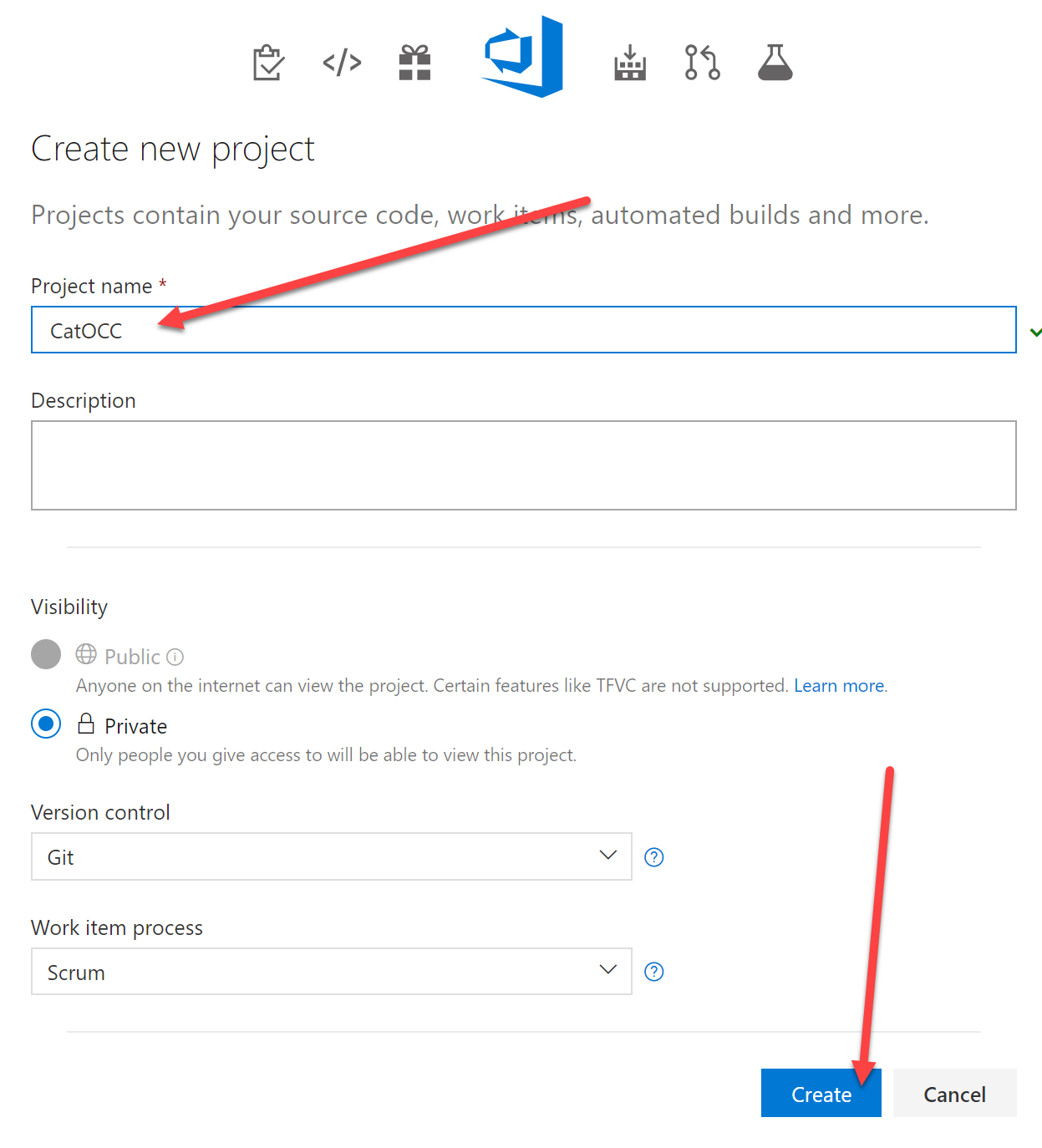
Exercise 1: Push your code to VSTS

Tasks

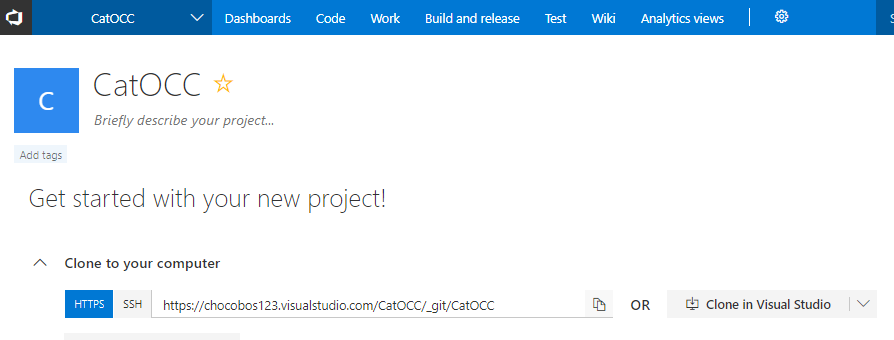
1. Login to VSTS
2. RDP into your VM.
3. Open a Chrome browser in your VM.
4. Navigate to initials-occ.visualstudio.com
5. Sign in if needed.
6. Click New Project.



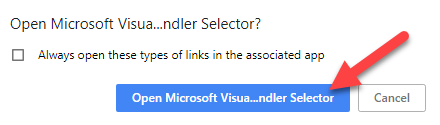
1. Name it initials-OCC
2. Leave all other settings as default, Git and Scrum. Click Create.



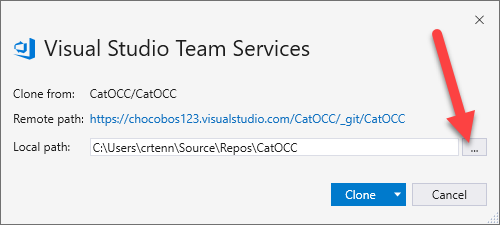
1. Push your code using Visual Studio
2. After the Create step, you will see a landing page like below. Click Clone in Visual Studio.



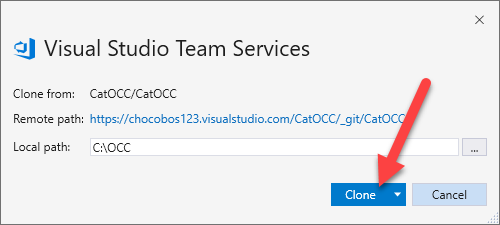
1. Hit Open Microsoft Visual Studio… Selector



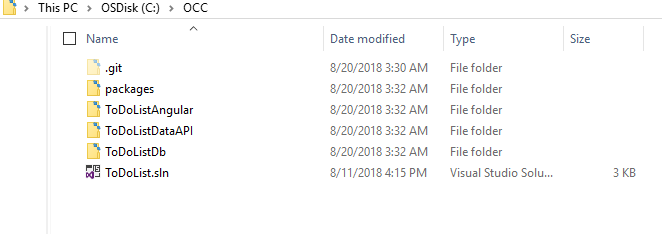
1. Wait a couple of minutes for Visual Studio to open and for a modal to pop-up like below. Click the “3 dots” to change the local path.



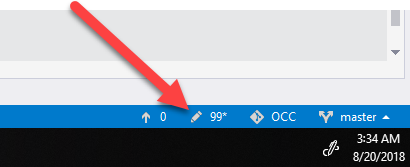
1. Add a new OCC folder under the C:\ drive. Change the local path to the C:\OCC drive. Click Clone.



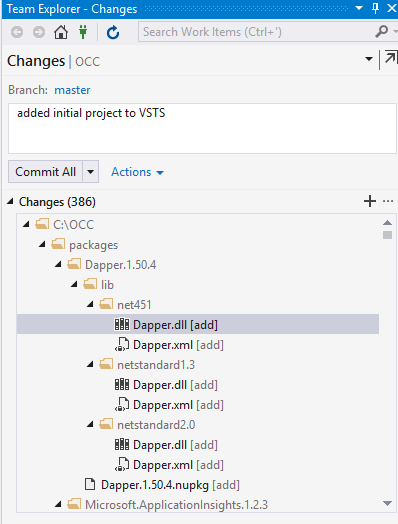
1. Copy all the files from the C:\labs\day1 folder into the C:\OCC folder.



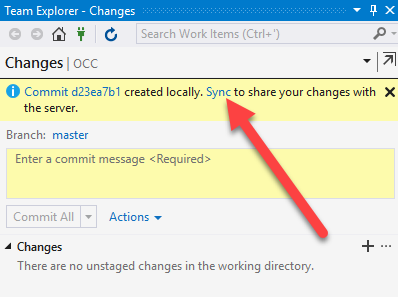
1. Click on the pencil icon on the bottom of Visual Studio with a # next to it.



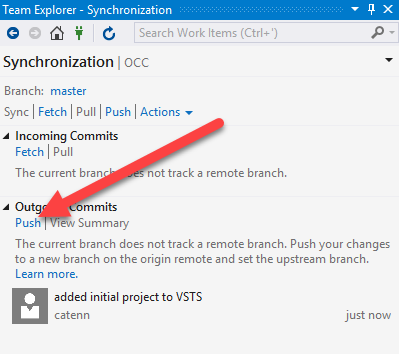
1. Write a commit message: added initial project to VSTS



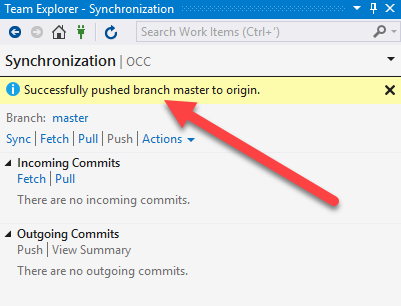
1. Click Commit All.
2. Click Sync



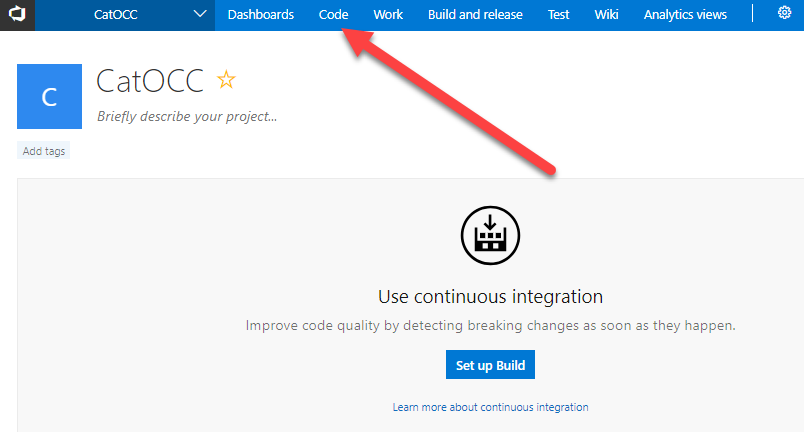
1. Click Push



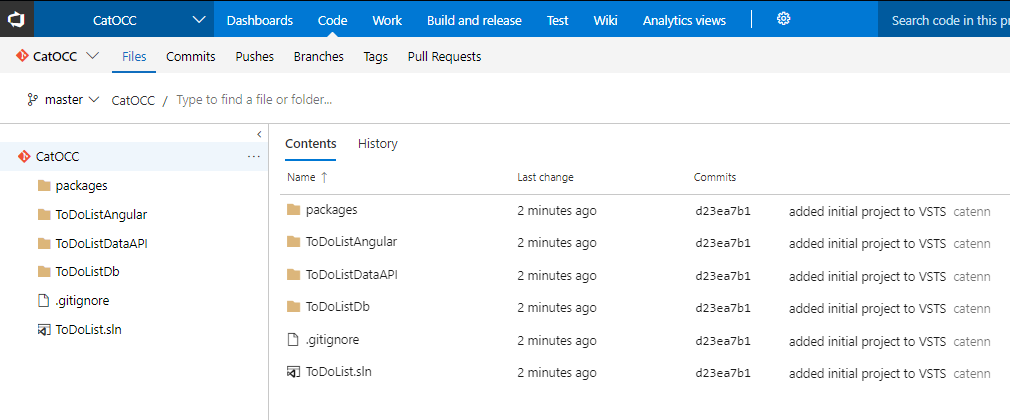
1. Wait a minute for the push to complete, you should see a progress bar then a notification when it is done.



1. Go back to VSTS in Chrome and hit refresh. Click on the Code tab.



1. Click Files. You should see your code.



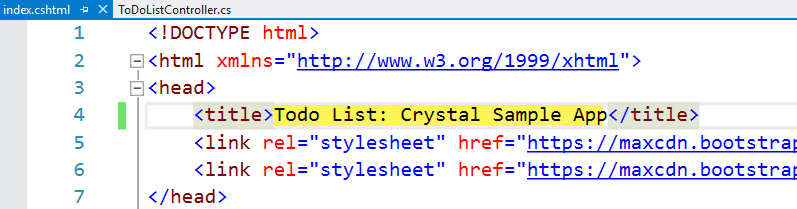
1. Navigate to the Azure Portal at <http://portal.azure.com> and login with your Azure Pass account associated with a personal Outlook email address.

Exercise 1 has been completed

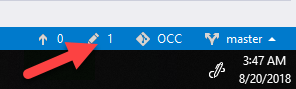
Exercise 2: Making a simple change in Visual Studio – Centralized Workflow

Tasks

1. Change some code!
2. In the Solution Explorer, under the ToDoListAngular project, find the index.cshtml file.
3. Modify the <title> in line 4 to say something custom.



1. Modify the <h1> in line 43 also to say something custom.
2. Click the pencil icon on the bottom of Visual Studio.



1. Double click the index.cshtml file to see a “diff” of your changes:



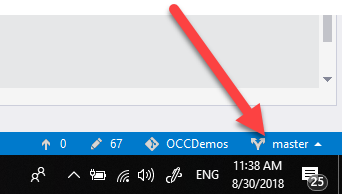
1. Enter a commit message. Hit Commit All.
2. Push your changes.
   1. Note: Sync will do a fetch, pull, then a push in order. Fetch will check what changes have been made to your branch by others. Pull will bring these changes into your branch.
3. Verify that the changes are seen in the VSTS Code Tab.

Exercise 2 has been completed

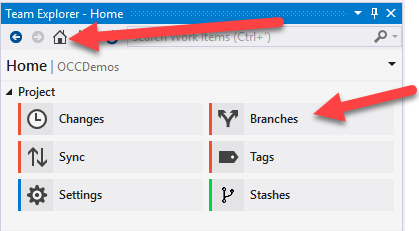
**Exercise 3: Making a change in Visual Studio – Feature Branch Workflow & Pull Requests**

**Tasks**

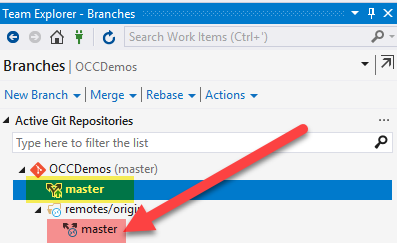
1. **Create a new branch.**
2. You can see which branch you are currently “checked out” to (which branch you are working on now) on the bottom right corner of VS, in this screenshot we are checked out to the master branch:



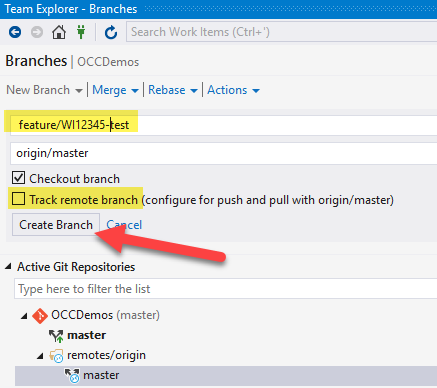
1. On Team Explorer, click Home > Branches.



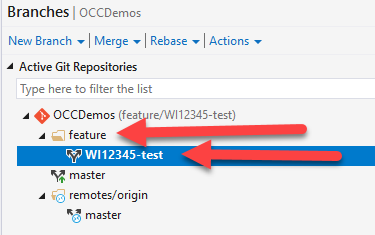
1. Right click on the master branch highlighted in red below.
   * The master branch highlighted in red is from “remote” = somewhere else, “origin” = the URL from VSTS in this instance.
   * The yellow highlighted master branch represents your local copy.
   * The local and remote (VSTS) won’t be in sync unless you explicitly pull from the remote VSTS to your local copy.
   * You should always create a new feature branch off the latest in VSTS (red highlighted) because it will be the most up to date branch that everyone works off of.



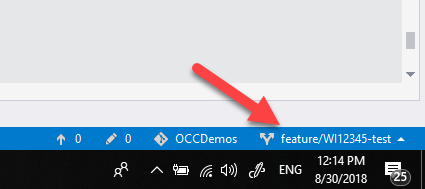
1. On the right click menu, select “New Local Branch From”.
2. Click create new branch. Name your new branch, the typical format is **feature/12345-description** (12345 can be changed to any number that correlated with Work Item #). Make sure “Track remote branch” is unchecked. Click Create Branch.



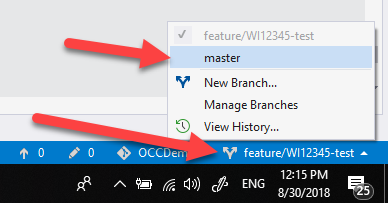
1. Notice, because you put “feature/” in the beginning of the branch name, it will signal to create a new empty directory for other similar branches.



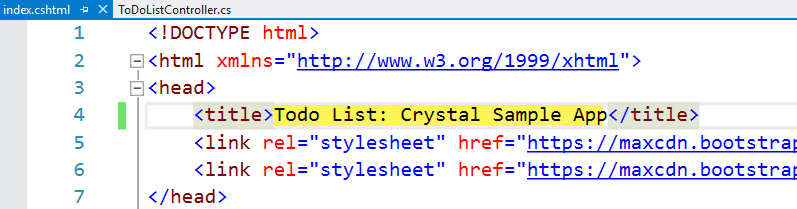
1. Also notice, you will now be “checked out” to your new feature branch automatically. Always check which branch you are in before you do your work.



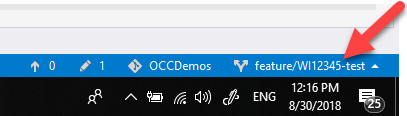
1. If you want to change back and forth between branches, as long as you have no changes or have committed all changes locally, click on the current branch name to open a pop-up menu. Try changing to master branch. Then change back to your feature branch!



1. **Change some code!**
2. In the Solution Explorer, under the ToDoListAngular project, find the index.cshtml file.
3. Modify the <title> in line 4 to say something custom.



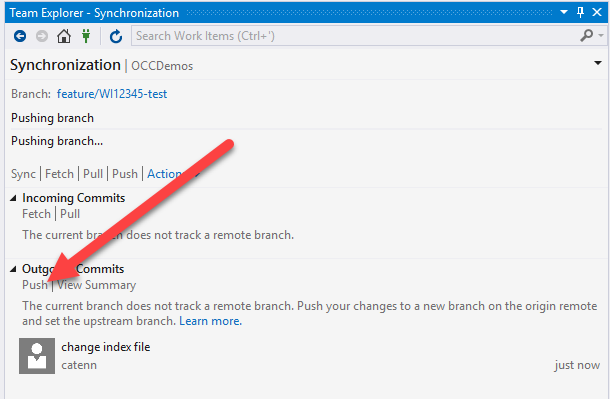
1. Modify the <h1> in line 43 also to say something custom.
2. Click the pencil icon on the bottom of Visual Studio. Note, this time you should be on your feature branch, double check!



1. Double click the index.cshtml file to see a “diff” of your changes:

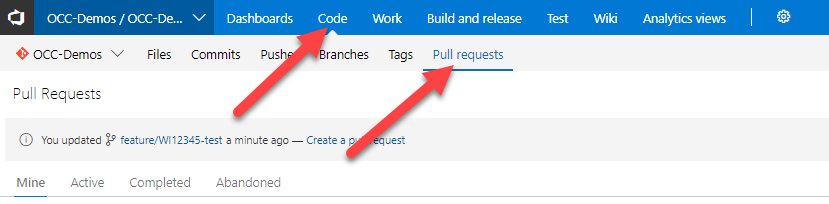


1. Enter a commit message.
2. Hit Commit All.
3. Push your changes. Note, the push will look a little different this time, because you created a new branch. See screenshot below.

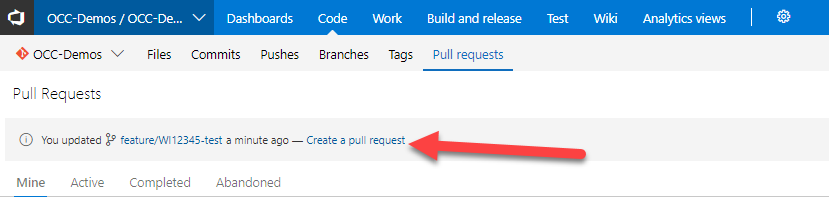


* 1. Note: Sync will do a fetch, pull, then a push in order. Fetch will check what changes have been made to your branch by others. Pull will bring these changes into your branch.

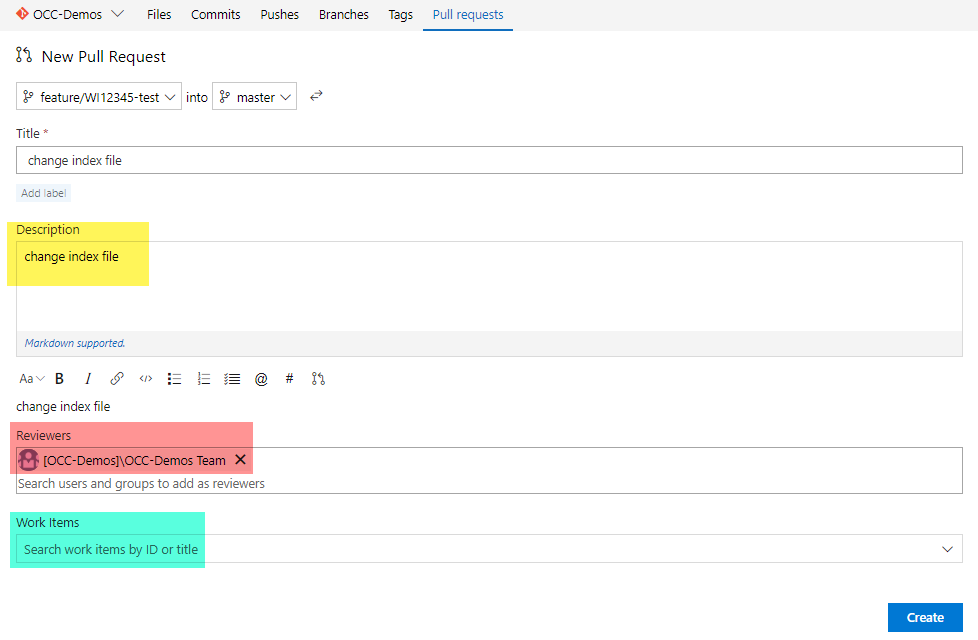
1. **Pull Requests**
2. Go to VSTS > Code > Pull Requests.



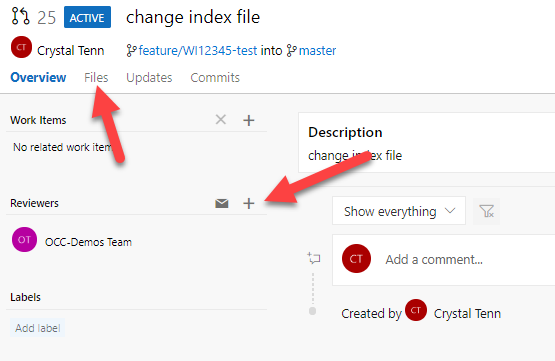
1. You will see a notification, click on the “Create a pull request” text:



1. You will see a pull request form partially filled out from your feature branch to your master branch.
   * In the yellow highlight: You can change the description to explain more of what you did so reviewers are able to understand your work.
   * In the red highlight: You can add different reviewers, you can also setup elsewhere in Settings which reviewers or groups of reviewers should be automatically added per repo.
   * In the blue highlight: You can attach work item IDs of existing work items so that QA will know which work items (tasks, bugs, user stories) that have been completed in a particular build.
   * On the very bottom of the screen when you scroll down, you can verify that all your changes made it in and look correct by viewing the diffs.



1. Click Create.
2. On the newly created Pull Request, you can add/change Reviewers. Now, click on the Files view.

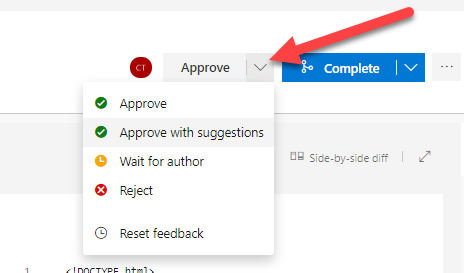


1. Mouseover a line until the little comment icon pops up, and click it (see screenshot below):

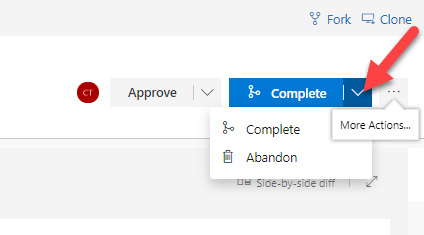


1. Try making comments and resolving them.
2. Click the arrow on the Approve button to view the options, then select one:

* **Approve**: Agree with the proposed changes in the pull request as-is.
* **Approve with suggestions**: Agree with the pull request, but provide optional suggestions to improve the code.
* **Wait for author**: Do not approve the changes, and ask the author to review your comments. The author should let you know when you should re-review the code after they have addressed your concerns.
* **Reject**: The changes aren't acceptable. If you are voting this way, you should leave a comment in the pull request detailing why the changes were rejected.
* **Reset feedback**: Choose **Reset feedback** to remove your vote.



1. Click the down arrow on the Complete button to see the options:
   * Complete: Complete the pull request now and merge the changes to the target branch.
   * Abandon: Choose Abandon to close the pull request without merging the changes.



1. Click Complete.
2. Click Complete Merge on the pop-up, you can leave the defaults for the checkboxes. Completing the merge will merge your feature branch code into the master branch.

* Check **Complete linked work items after merging** to complete any linked work items.
* Check **Delete <branch name> after merging** to delete the source branch from the pull request.
* Check **Squash changes when merging** to [squash merge](https://docs.microsoft.com/en-us/vsts/repos/git/merging-with-squash?view=vsts) your pull request.

*Exercise 3 has been completed*