You can configure your Visual Studio Team Services team projects to automatically build and deploy to Azure web apps or cloud services. (For information on how to set up a continuous build and deploy system using an *on-premises* Team Foundation Server, see Continuous Delivery for Cloud Services in Azure.)

This tutorial assumes you have Visual Studio 2013 and the Azure SDK installed. If you don't already have Visual Studio 2013, download it by choosing the **Get started for free** link at www.visualstudio.com. Install the Azure SDK from here.

> **Note:**
>
> You need an Visual Studio Team Services account to complete this tutorial: You can open a Visual Studio Team Services account for free.

To set up a cloud service to automatically build and deploy to Azure by using Visual Studio Team Services, follow these steps.

1: Create a team project

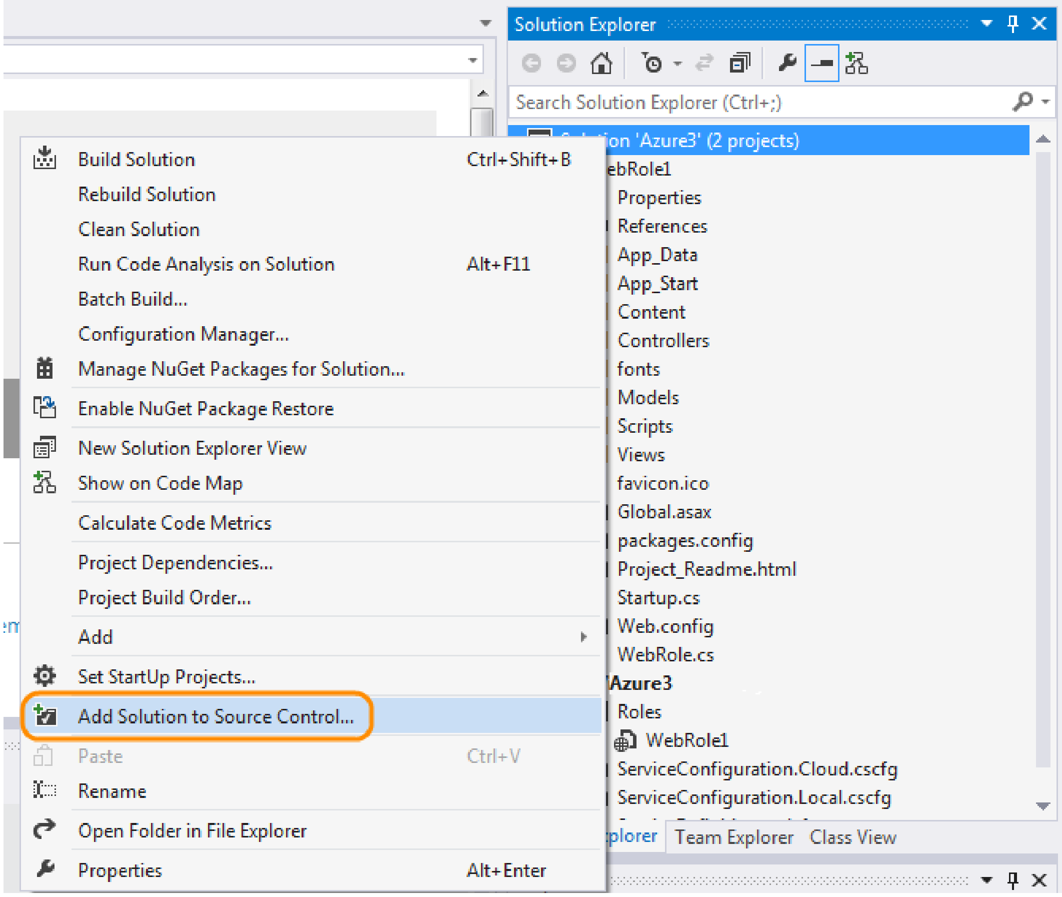
Follow the instructions here to create your team project and link it to Visual Studio. This walkthrough assumes you are using Team Foundation Version Control (TFVC) as your source control solution. If you want to use Git for version control, see the Git version of this walkthrough.

2: Check in a project to source control

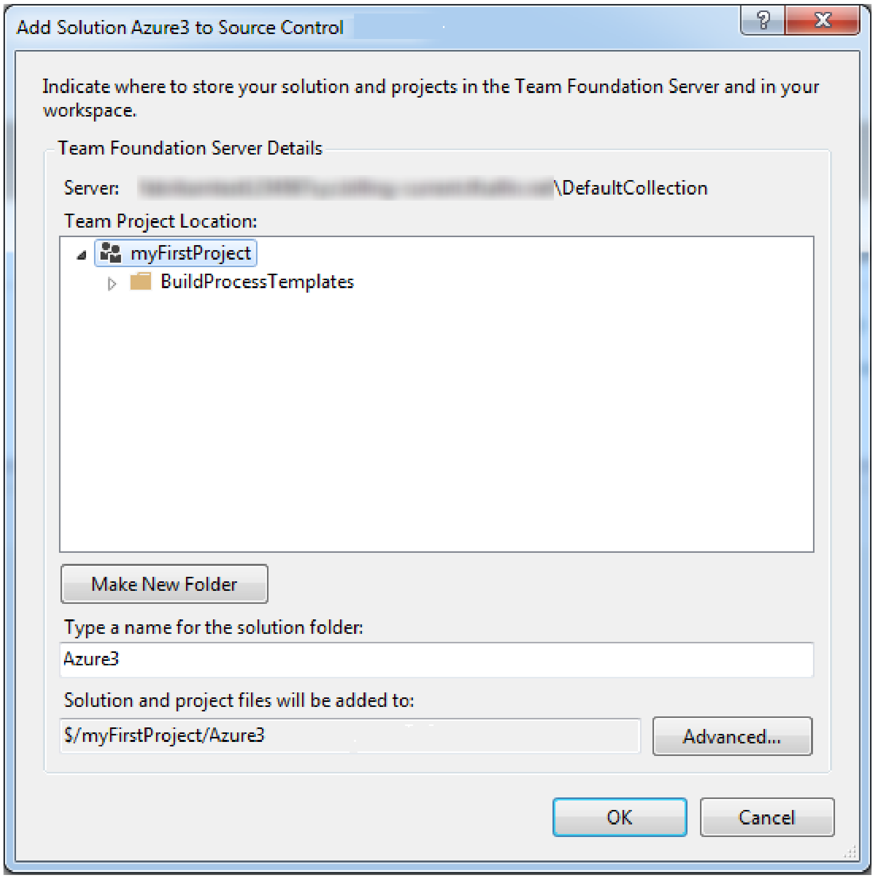
1. In Visual Studio, open the solution you want to deploy, or create a new one. You can deploy a web app or a cloud service (Azure Application) by following the steps in this walkthrough. If you want to create a new solution, create a new Azure Cloud Service project, or a new ASP.NET MVC project. Make sure that the project targets .NET Framework 4 or 4.5, and if you are creating a cloud service project, add an ASP.NET MVC web role and a worker role, and choose Internet application for the web role. When prompted, choose **Internet Application**. If you want to create a web app, choose the ASP.NET Web Application project template, and then choose MVC. See Create an ASP.NET web app in Azure App Service.

   > **Note:**
   >
   > Visual Studio Team Services only support CI deployments of Visual Studio Web Applications at this time. Web Site projects are out of scope.

2. Open the context menu for the solution, and choose **Add Solution to Source Control**.



3. Accept or change the defaults and choose the **OK** button. Once the process completes, source control icons appear in **Solution Explorer**.



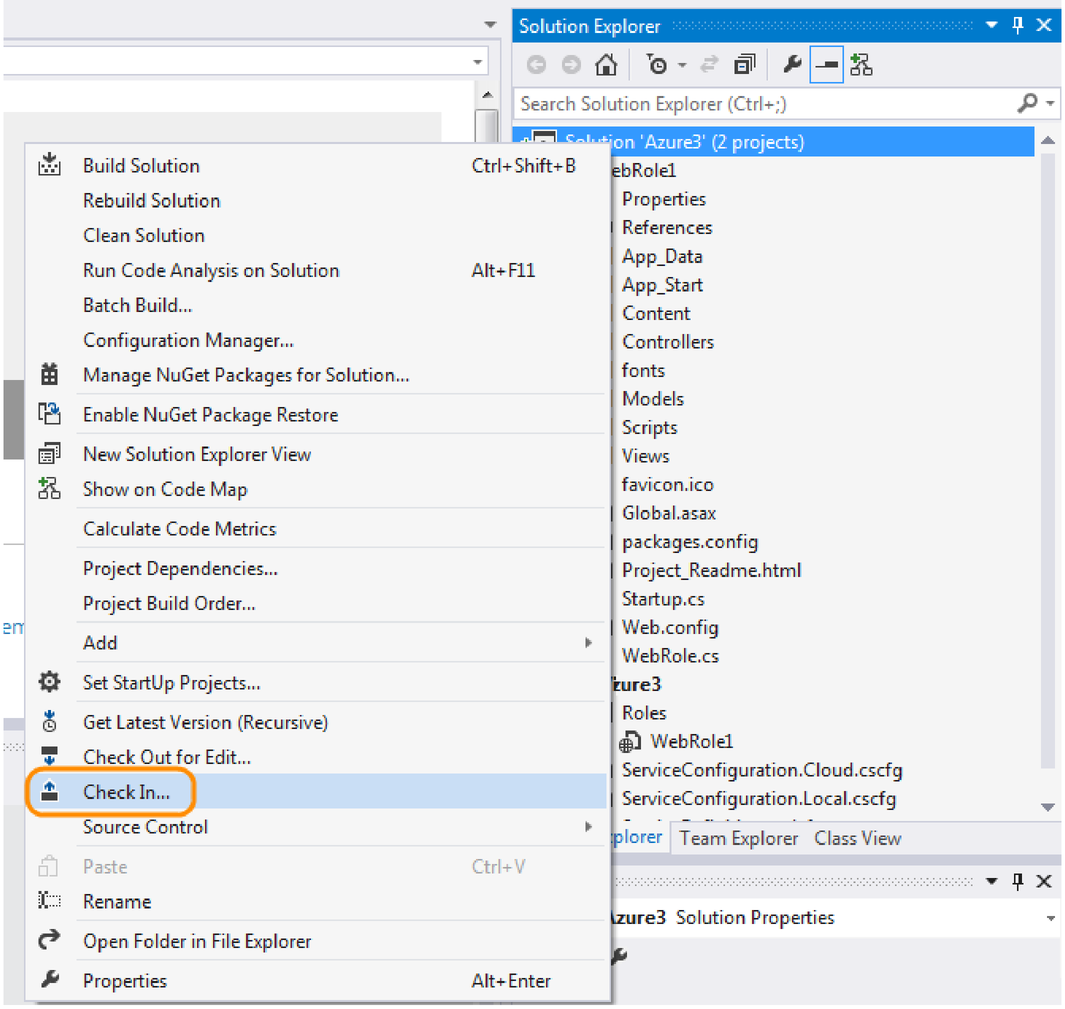
Indicate where to store your solution and projects in the Team Foundation Server and in your workspace.

Server: [redacted]\DefaultCollection

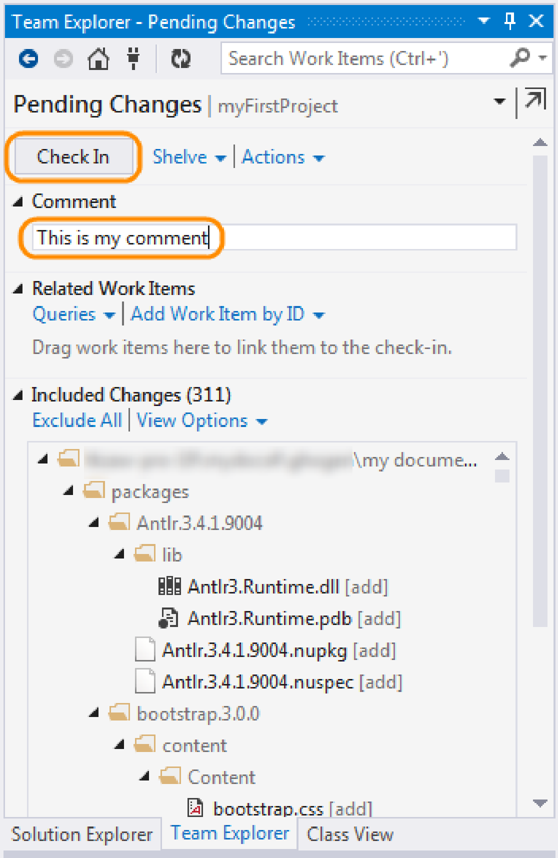
Type a name for the solution folder: Azure3

Solution and project files will be added to: $/myFirstProject/Azure3

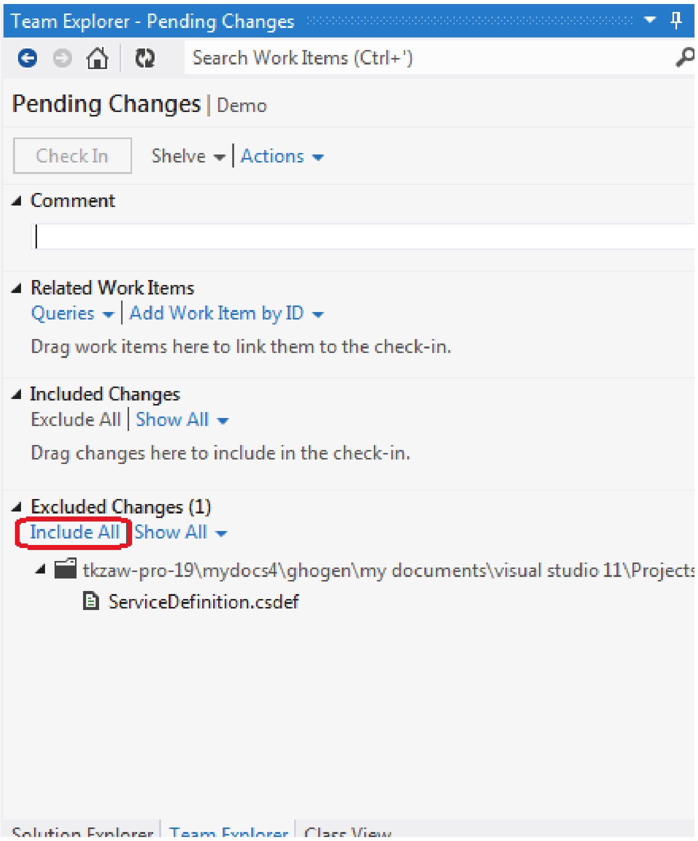
4. Open the shortcut menu for the solution, and choose **Check In**.



5. In the **Pending Changes** area of **Team Explorer**, type a comment for the check-in and choose the **Check In** button.



Note the options to include or exclude specific changes when you check in. If desired changes are excluded, choose the **Include All** link.



Team Explorer - Pending Changes

Search Work Items (Ctrl+')

Pending Changes | Demo

Check In Shelve | Actions

Comment

Related Work Items

Queries | Add Work Item by ID

Drag work items here to link them to the check-in.

Included Changes

Exclude All | Show All

Drag changes here to include in the check-in.

Excluded Changes (1)

Include All | Show All

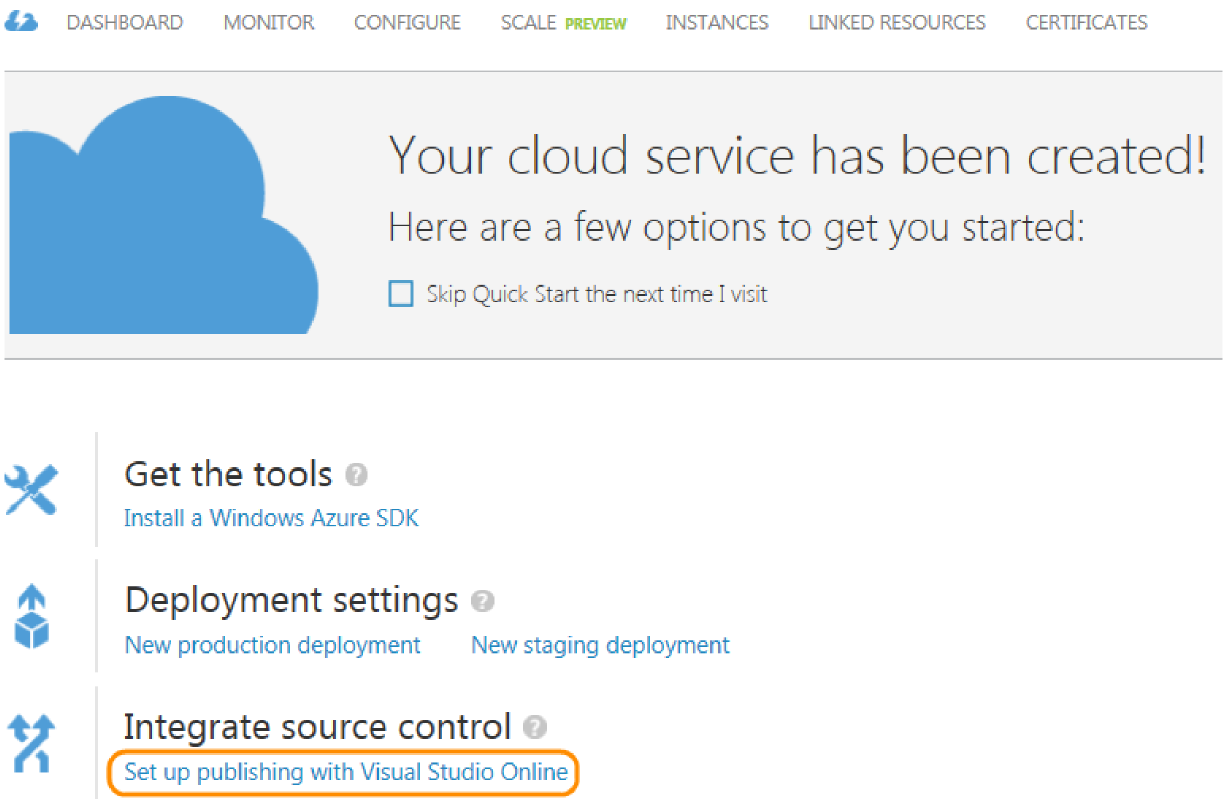
tkzaw-pro-19\mydocs4\ghogen\my documents\visual studio 11\Projects

ServiceDefinition.csdef

Solution Explorer | Team Explorer | Class View

3: Connect the project to Azure

1. Now that you have a VS Team Services team project with some source code in it, you are ready to connect your team project to Azure. In theAzure classic portal, select your cloud service or web app, or create a new one by choosing the + icon at the bottom left and choosing **Cloud Service** or **Web App** and then **Quick Create**. Choose the **Set up publishing with Visual Studio Team Services** link.



DASHBOARD MONITOR CONFIGURE SCALE PREVIEW INSTANCES LINKED RESOURCES CERTIFICATES

Your cloud service has been created!

Here are a few options to get you started:

□ Skip Quick Start the next time I visit

Get the tools

Install a Windows Azure SDK

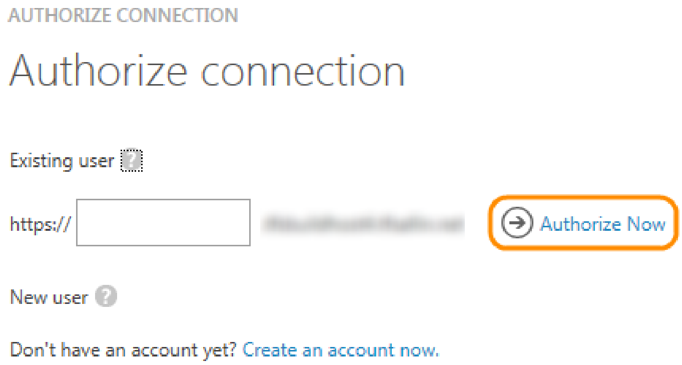
Deployment settings

New production deployment New staging deployment

Integrate source control

Set up publishing with Visual Studio Online

2. In the wizard, type the name of your Visual Studio Team Services account in the textbox and click the **Authorize Now** link. You might be asked to sign in.



AUTHORIZE CONNECTION

Authorize connection

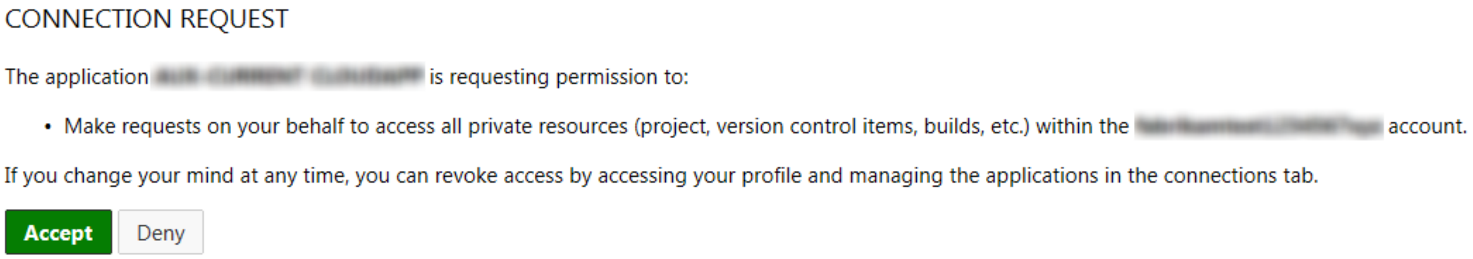
Existing user

https:// [redacted] Authorize Now

New user

Don't have an account yet? Create an account now.

3. In the **Connection Request** pop-up dialog, choose the **Accept** button to authorize Azure to configure your team project in VS Team Services.



CONNECTION REQUEST

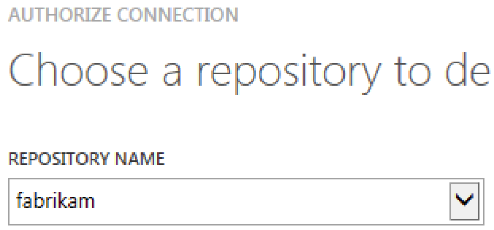
The application [redacted] is requesting permission to:

- Make requests on your behalf to access all private resources (project, version control items, builds, etc.) within the [redacted] account.

If you change your mind at any time, you can revoke access by accessing your profile and managing the applications in the connections tab.

Accept Deny

4. When authorization succeeds, you see a dropdown containing a list of your Visual Studio Team Services team projects. Choose the name of team project that you created in the previous steps, and then choose the wizard's checkmark button.



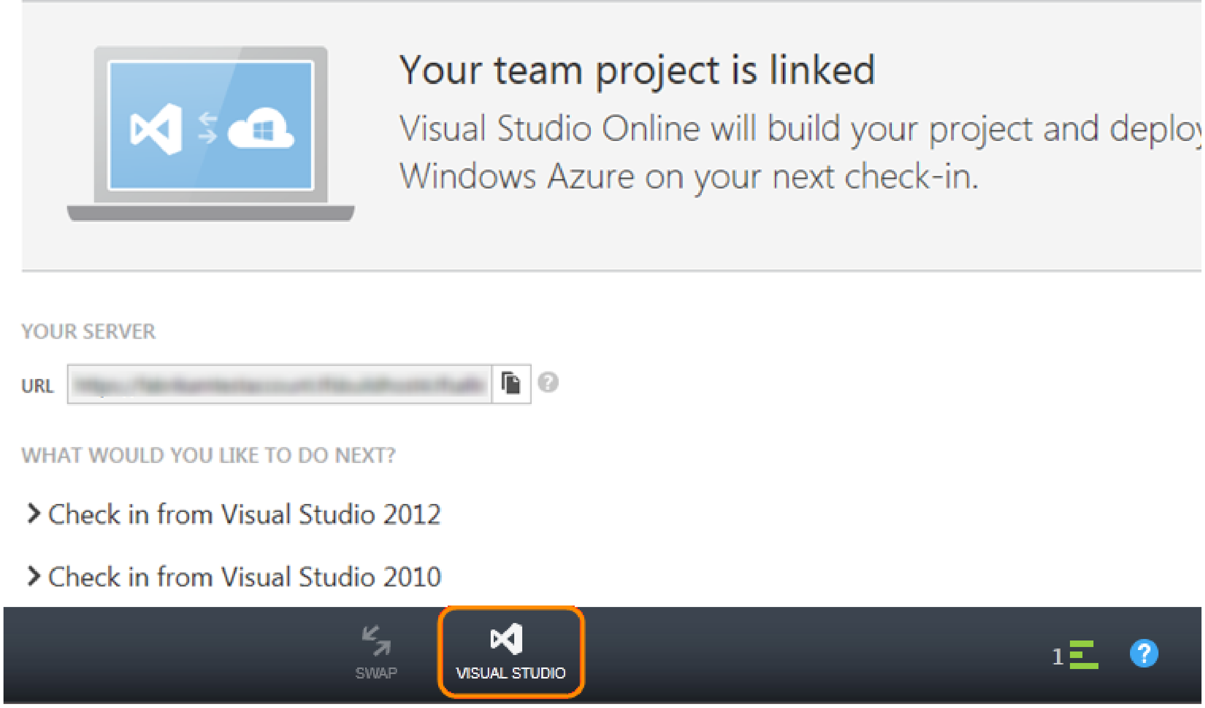
AUTHORIZE CONNECTION

Choose a repository to deploy

REPOSITORY NAME

fabrikam

5. After your project is linked, you will see some instructions for checking in changes to your Visual Studio Team Services team project. On your next check-in, Visual Studio Team Services will build and deploy your project to Azure. Try this now by clicking the **Check In from Visual Studio** link, and then the **Launch Visual Studio** link (or the equivalent**Visual Studio** button at the bottom of the portal screen).



Your team project is linked

Visual Studio Online will build your project and deploy
Windows Azure on your next check-in.

YOUR SERVER

URL [redacted]

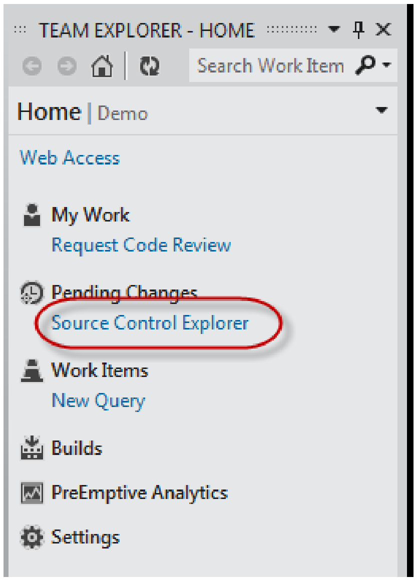
WHAT WOULD YOU LIKE TO DO NEXT?

› Check in from Visual Studio 2012

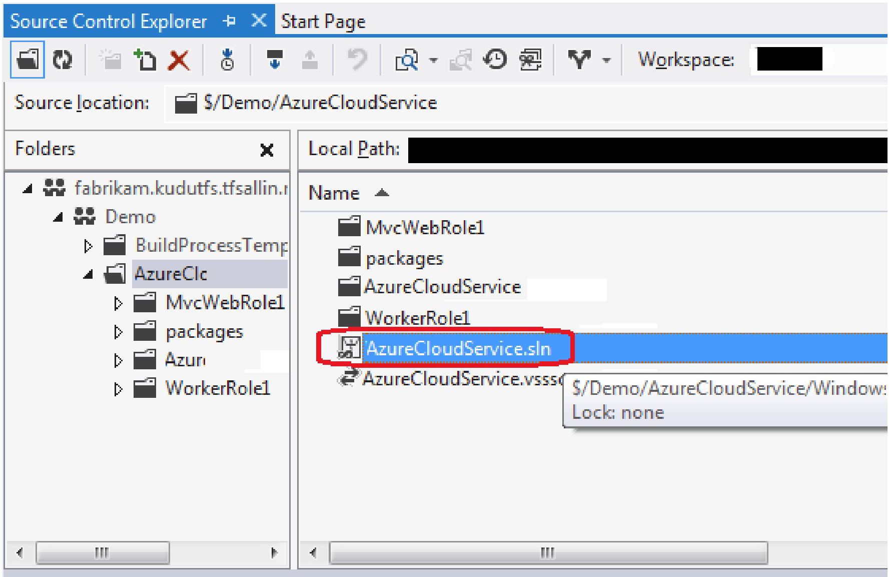
› Check in from Visual Studio 2010

4: Trigger a rebuild and redeploy your project

1. In Visual Studio's **Team Explorer**, choose the **Source Control Explorer**link.



2. Navigate to your solution file and open it.

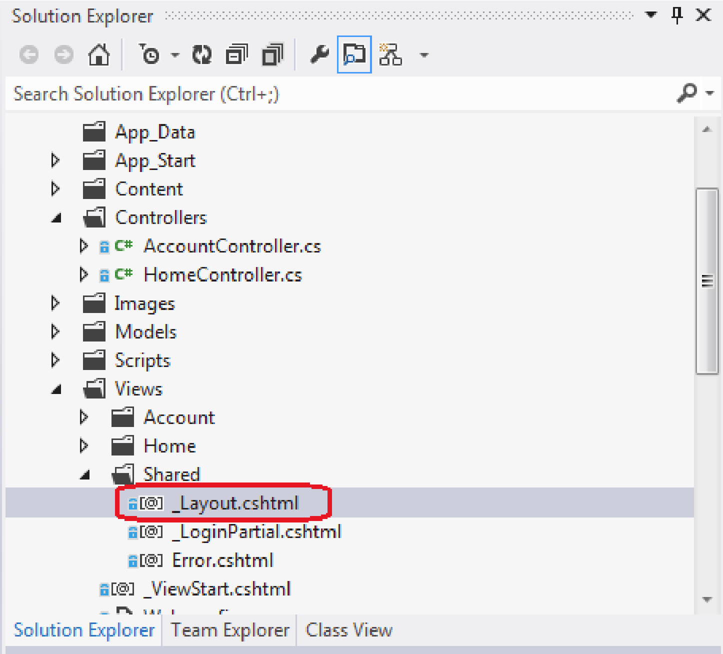


Source location: $/Demo/AzureCloudService

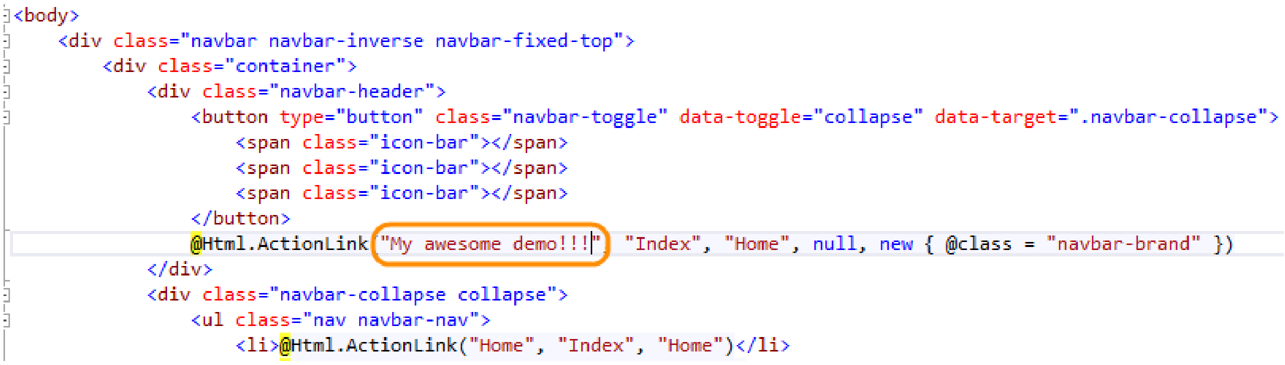
Workspace: [redacted]

Local Path: [redacted]

3. In **Solution Explorer**, open up a file and change it. For example, change the file `_Layout.cshtml` under the Views\Shared folder in an MVC web role.

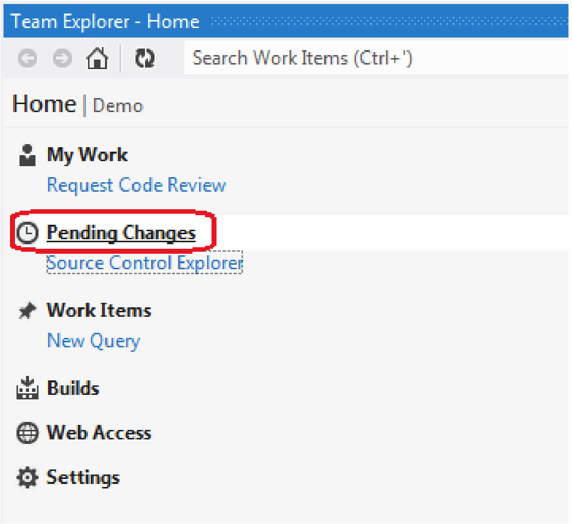


4. Edit the logo for the site and choose **Ctrl+S** to save the file.

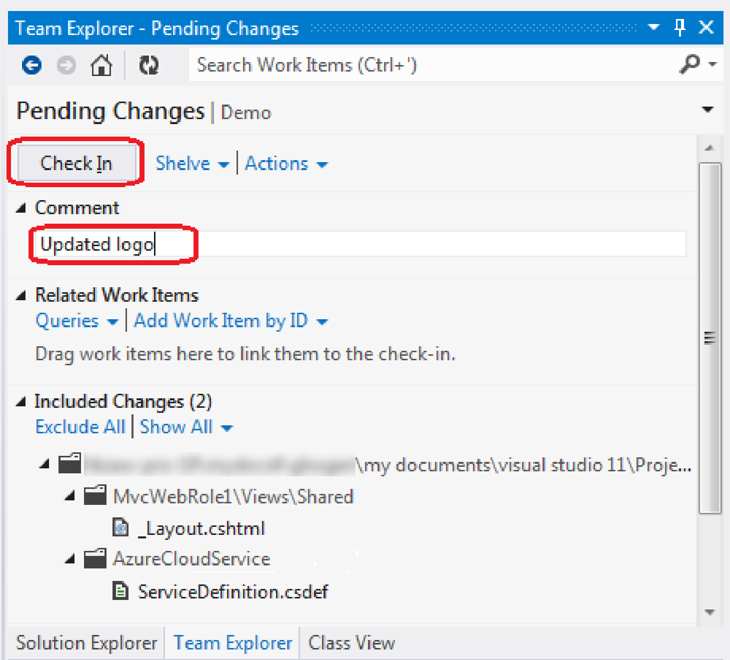


```
<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                @Html.ActionLink("My awesome demo!!!", "Index", "Home", null, new { @class = "navbar-brand" })
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav">
                    <li>@Html.ActionLink("Home", "Index", "Home")</li>
```

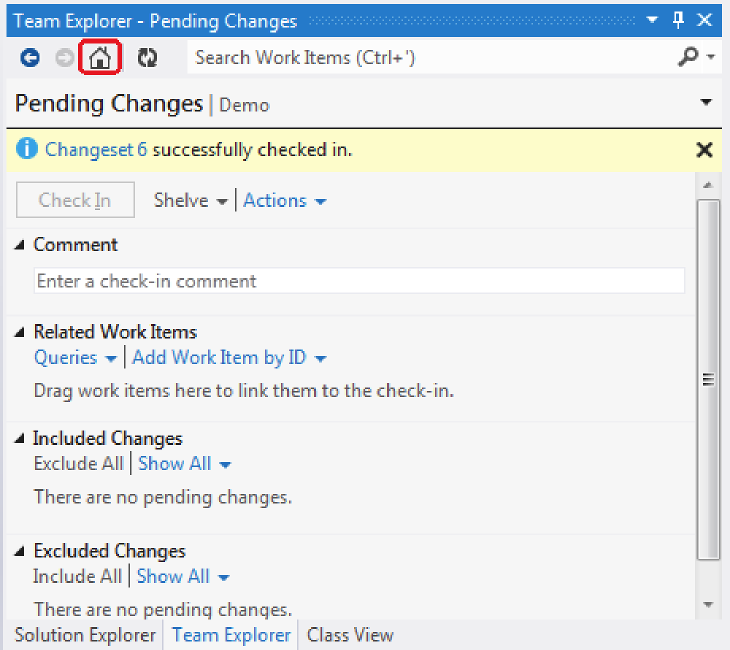
5. In **Team Explorer**, choose the **Pending Changes** link.



6. Enter a comment and then choose the **Check In** button.

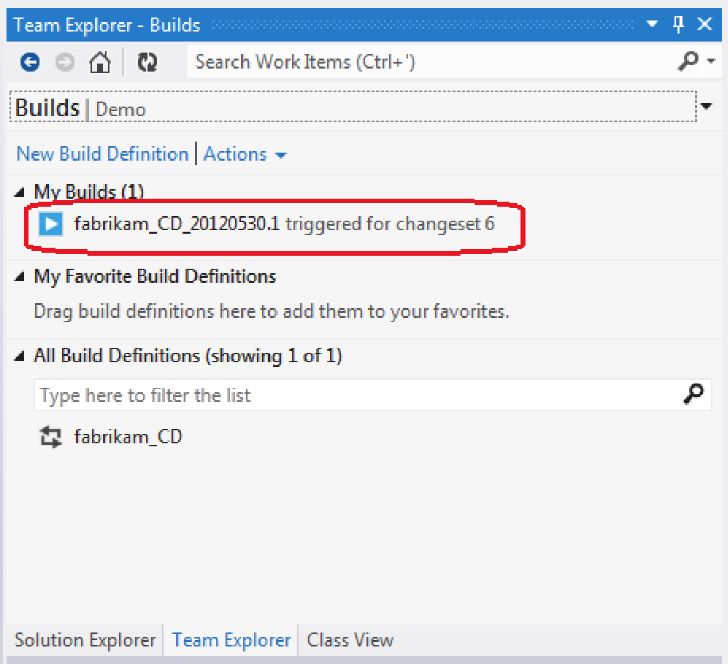


7. Choose the **Home** button to return to the **Team Explorer** home page.



8. Choose the **Builds** link to view the builds in progress.

Team Explorer shows that a build has been triggered for your check-in.



Team Explorer - Builds

Search Work Items (Ctrl+')

Builds | Demo

New Build Definition | Actions

My Builds (1)

fabrikam_CD_20120530.1 triggered for changeset 6

My Favorite Build Definitions

Drag build definitions here to add them to your favorites.

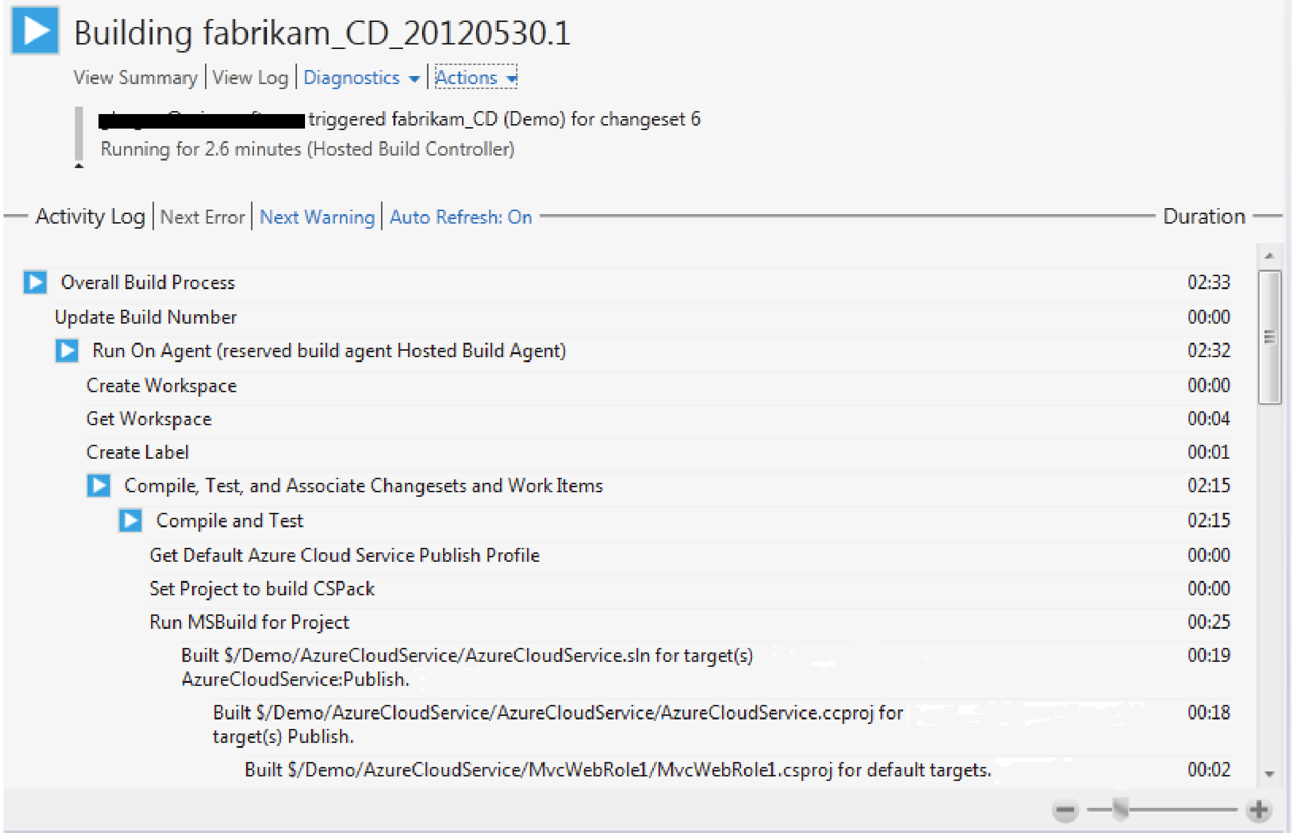
All Build Definitions (showing 1 of 1)

Type here to filter the list

fabrikam_CD

Solution Explorer | Team Explorer | Class View

9. Double-click the name of the build in progress to view a detailed log as the build progresses.



Building fabrikam_CD_20120530.1

View Summary | View Log | Diagnostics | Actions

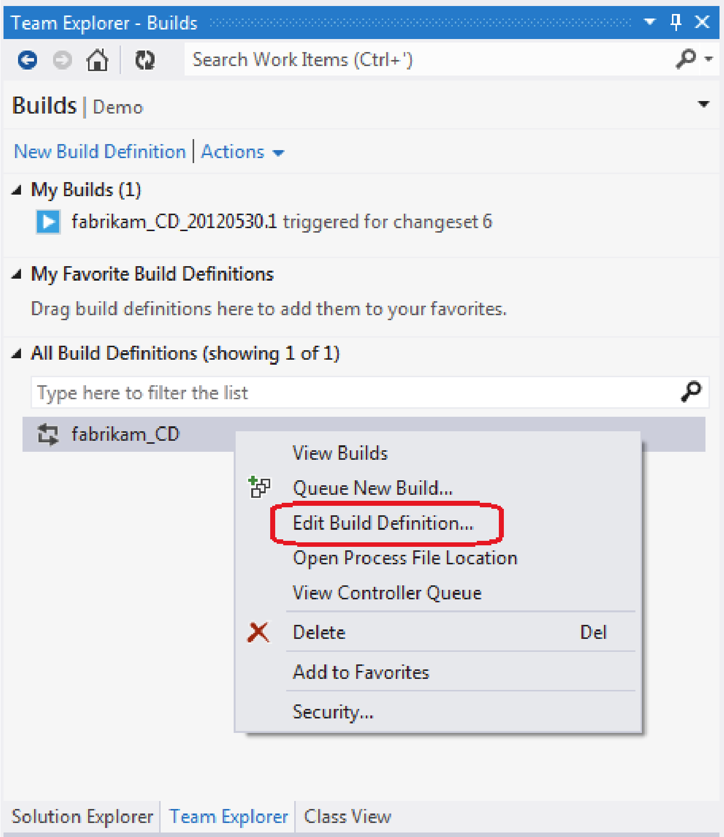
[redacted] triggered fabrikam_CD (Demo) for changeset 6
Running for 2.6 minutes (Hosted Build Controller)

Activity Log | Next Error | Next Warning | Auto Refresh: On

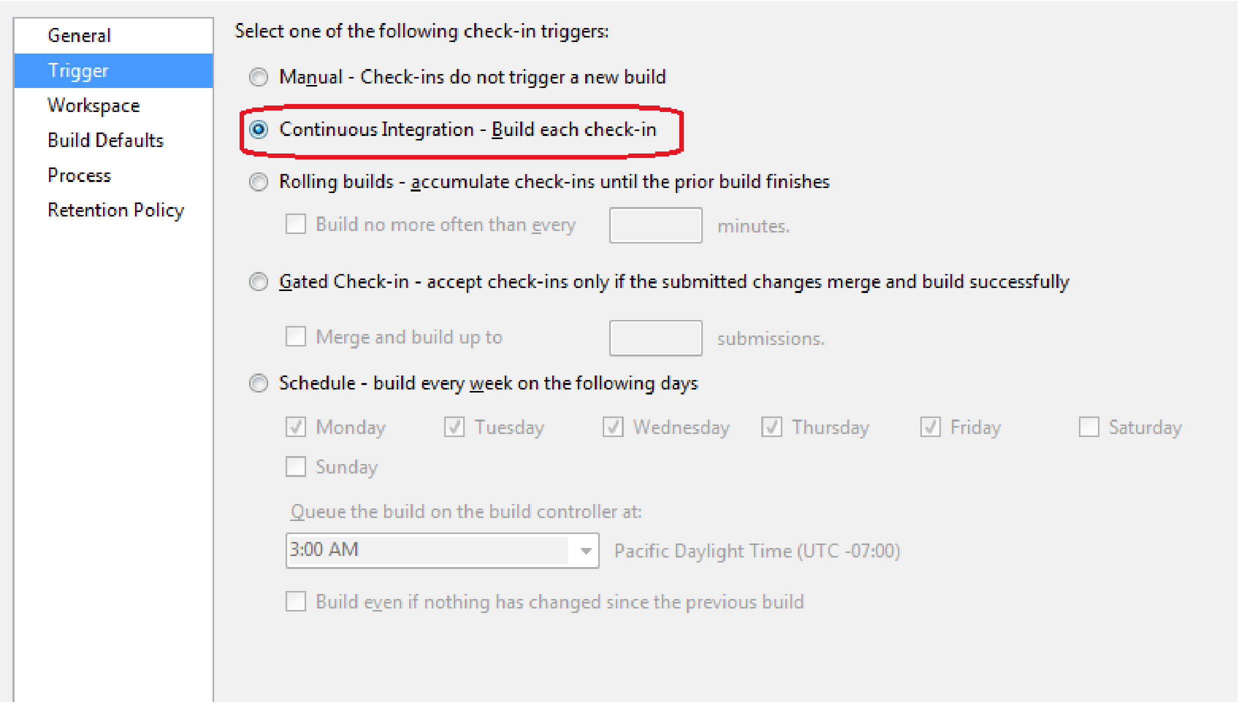
```
Overall Build Process                                                                  02:33
 Update Build Number                                                                   00:00
  Run On Agent (reserved build agent Hosted Build Agent)                               02:32
   Create Workspace                                                                    00:00
   Get Workspace                                                                       00:04
   Create Label                                                                        00:01
   Compile, Test, and Associate Changesets and Work Items                              02:15
    Compile and Test                                                                   02:15
     Get Default Azure Cloud Service Publish Profile                                   00:00
     Set Project to build CSPack                                                       00:00
     Run MSBuild for Project                                                           00:25
      Built $/Demo/AzureCloudService/AzureCloudService.sln for target(s)               00:19
      AzureCloudService:Publish.
       Built $/Demo/AzureCloudService/AzureCloudService/AzureCloudService.ccproj for   00:18
       target(s) Publish.
        Built $/Demo/AzureCloudService/MvcWebRole1/MvcWebRole1.csproj for default targets.  00:02
```

Duration

10. While the build is in-progress, take a look at the build definition that was created when you linked TFS to Azure by using the wizard. Open the shortcut menu for the build definition and choose **Edit Build Definition**.


Team Explorer - Builds
Search Work Items (Ctrl+')
Builds | Demo
New Build Definition | Actions
My Builds (1)
fabrikam_CD_20120530.1 triggered for changeset 6
My Favorite Build Definitions
Drag build definitions here to add them to your favorites.
All Build Definitions (showing 1 of 1)
Type here to filter the list
fabrikam_CD
View Builds
Queue New Build...
Edit Build Definition...
Open Process File Location
View Controller Queue
Delete Del
Add to Favorites
Security...
Solution Explorer
Team Explorer
Class View

In the **Trigger** tab, you will see that the build definition is set to build on every check-in by default.



Select one of the following check-in triggers:

○ Manual - Check-ins do not trigger a new build

● Continuous Integration - Build each check-in

○ Rolling builds - accumulate check-ins until the prior build finishes

□ Build no more often than every ____ minutes.

○ Gated Check-in - accept check-ins only if the submitted changes merge and build successfully

□ Merge and build up to ____ submissions.

○ Schedule - build every week on the following days

☑ Monday ☑ Tuesday ☑ Wednesday ☑ Thursday ☑ Friday □ Saturday

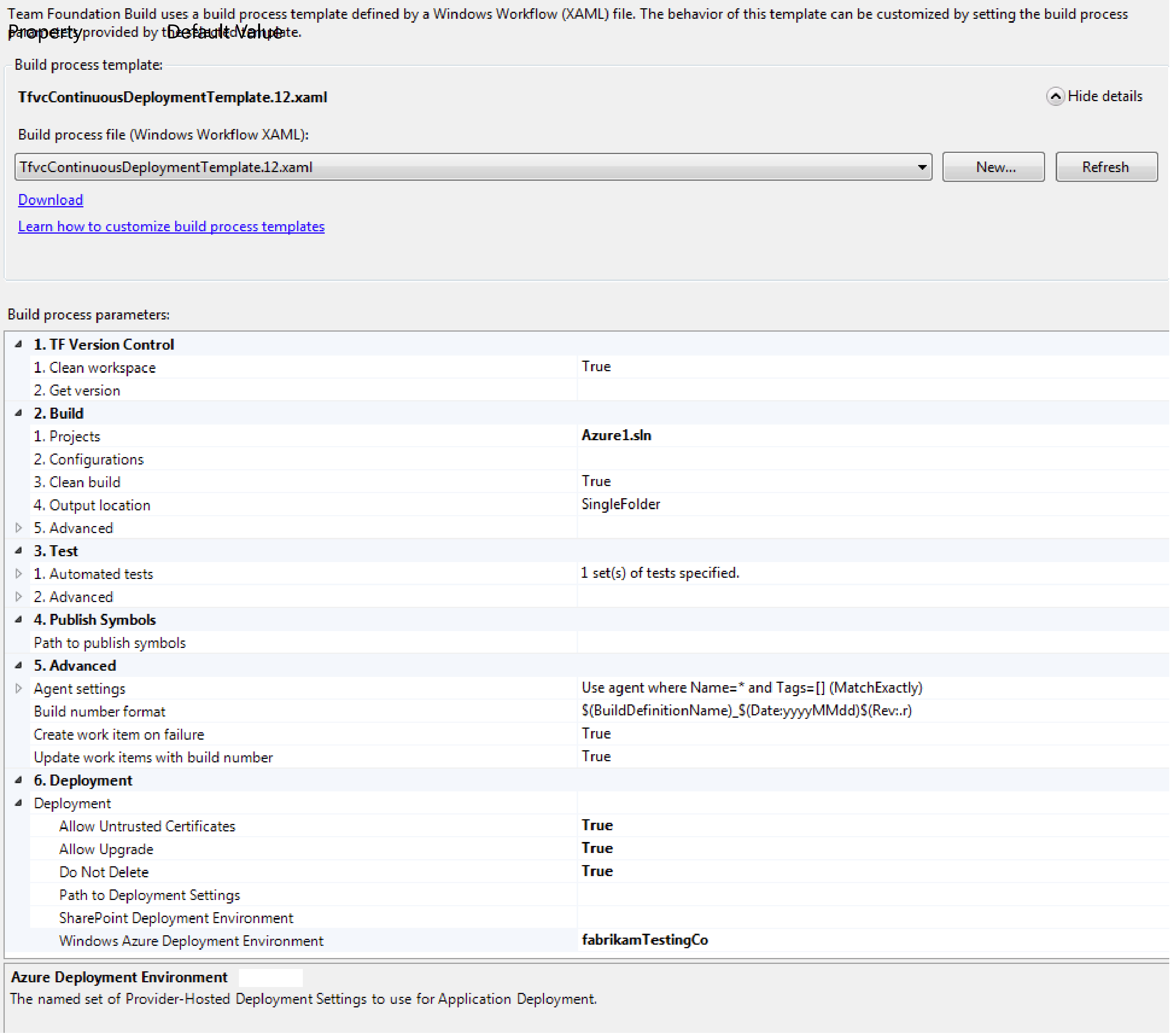
□ Sunday

Queue the build on the build controller at:

3:00 AM Pacific Daylight Time (UTC -07:00)

□ Build even if nothing has changed since the previous build

In the **Process** tab, you can see the deployment environment is set to the name of your cloud service or web app. If you are working with web apps, the properties you see will be different from those shown here.



Team Foundation Build uses a build process template defined by a Windows Workflow (XAML) file. The behavior of this template can be customized by setting the build process parameters provided by the selected template.

Build process template:

TfvcContinuousDeploymentTemplate.12.xaml

Build process file (Windows Workflow XAML):

TfvcContinuousDeploymentTemplate.12.xaml

Build process parameters:

1. TF Version Control	
1. Clean workspace	True
2. Get version	
2. Build	
1. Projects	**Azure1.sln**
2. Configurations	
3. Clean build	True
4. Output location	SingleFolder
5. Advanced	
3. Test	
1. Automated tests	1 set(s) of tests specified.
2. Advanced	
4. Publish Symbols	
Path to publish symbols	
5. Advanced	
Agent settings	Use agent where Name=* and Tags=[] (MatchExactly)
Build number format	$(BuildDefinitionName)_$(Date:yyyyMMdd)$(Rev:.r)
Create work item on failure	True
Update work items with build number	True
6. Deployment	
Deployment	
Allow Untrusted Certificates	**True**
Allow Upgrade	**True**
Do Not Delete	**True**
Path to Deployment Settings	
SharePoint Deployment Environment	
Windows Azure Deployment Environment	**fabrikamTestingCo**

Azure Deployment Environment

The named set of Provider-Hosted Deployment Settings to use for Application Deployment.

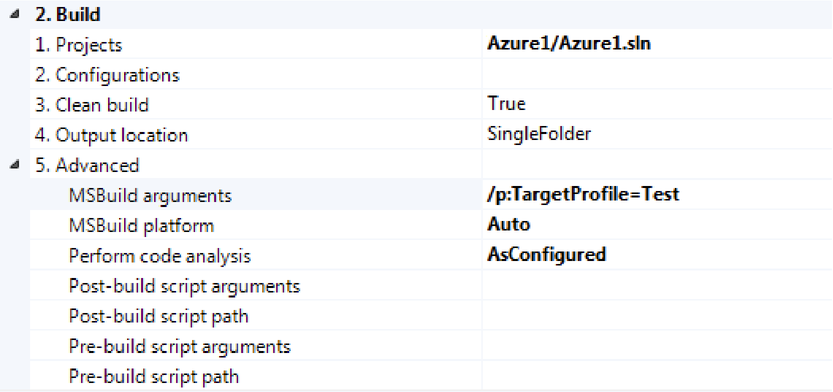
11. Specify values for the properties if you want different values than the defaults. The properties for Azure publishing are in the **Deployment**section.

    The following table shows the available properties in the **Deployment**section:

Property	Default Value
Allow Untrusted Certificates	If false, SSL certificates must be signed by a root authority.
Allow Upgrade	Allows the deployment to update an existing deployment instead of creating a new one. Preserves the IP address.
Do Not Delete	If true, do not overwrite an existing unrelated deployment (upgrade is allowed).

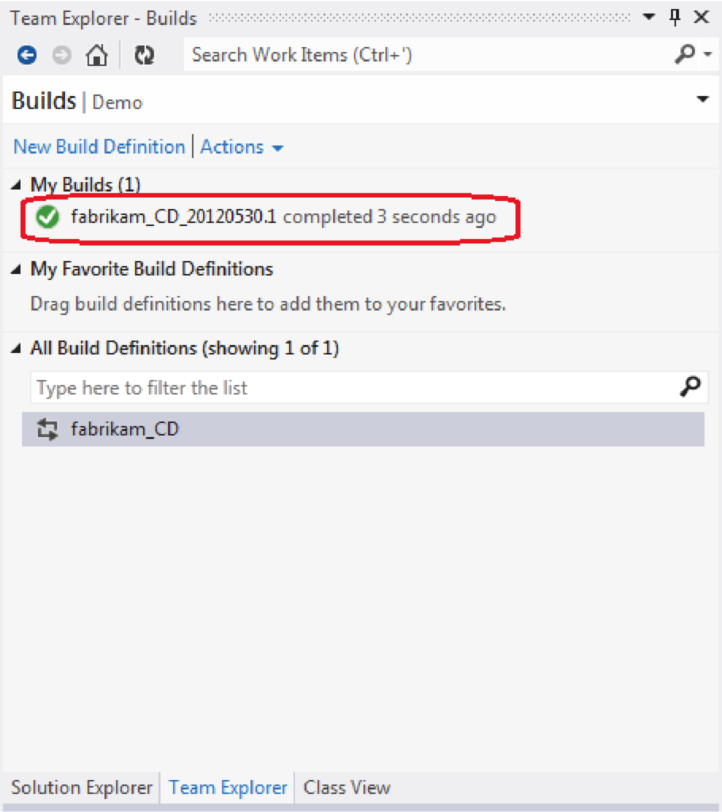
Property	Default Value
Path to Deployment Settings	The path to your .pubxml file for a web app, relative to the root folder of the repo. Ignored for cloud services.
Sharepoint Deployment Environment	The same as the service name.
Azure Deployment Environment	The web app or cloud service name.

12. If you are using multiple service configurations (.cscfg files), you can specify the desired service configuration in the **Build, Advanced, MSBuild arguments** setting. For example, to use ServiceConfiguration.Test.cscfg, set MSBuild arguments line option `/p:TargetProfile=Test` .

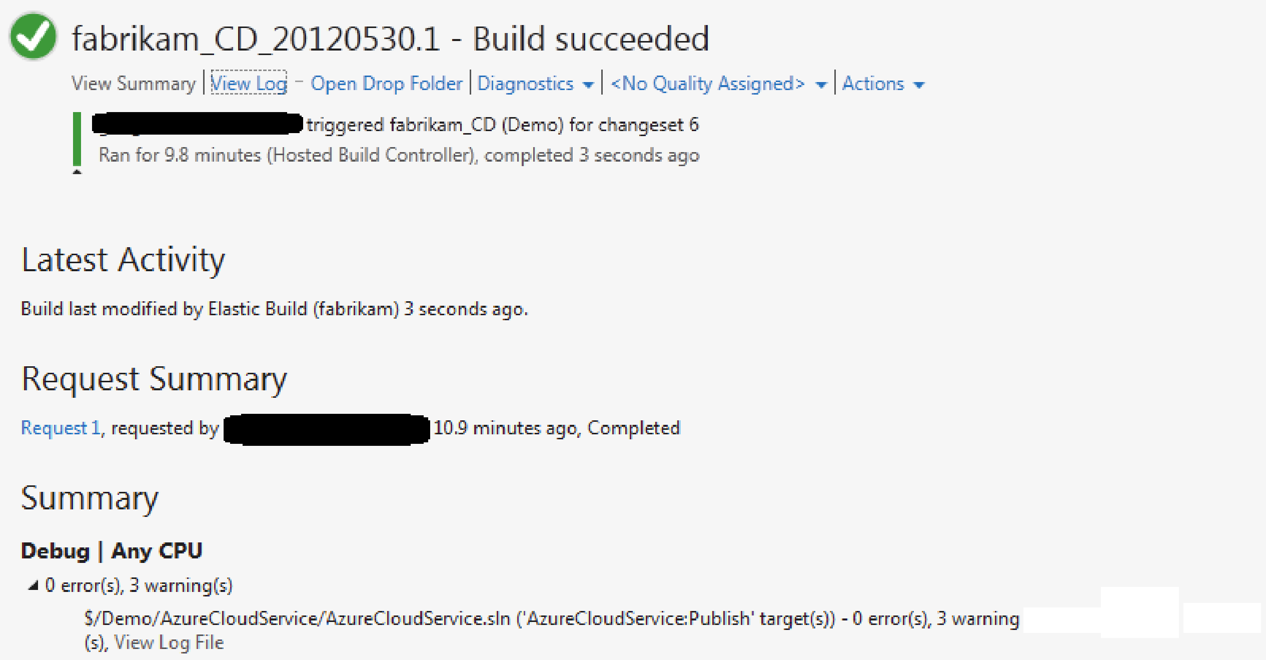


2. Build	
1. Projects	**Azure1/Azure1.sln**
2. Configurations	
3. Clean build	True
4. Output location	SingleFolder
5. Advanced	
MSBuild arguments	**/p:TargetProfile=Test**
MSBuild platform	**Auto**
Perform code analysis	**AsConfigured**
Post-build script arguments	
Post-build script path	
Pre-build script arguments	
Pre-build script path	

By this time, your build should be completed successfully.


Team Explorer - Builds
Search Work Items (Ctrl+')
Builds | Demo
New Build Definition | Actions
My Builds (1)
fabrikam_CD_20120530.1 completed 3 seconds ago
My Favorite Build Definitions
Drag build definitions here to add them to your favorites.
All Build Definitions (showing 1 of 1)
Type here to filter the list
fabrikam_CD
Solution Explorer
Team Explorer
Class View

13. If you double-click the build name, Visual Studio shows a **Build Summary**, including any test results from associated unit test projects.



fabrikam_CD_20120530.1 - Build succeeded

View Summary | View Log - Open Drop Folder | Diagnostics ▾ | <No Quality Assigned> ▾ | Actions ▾

[redacted] triggered fabrikam_CD (Demo) for changeset 6
Ran for 9.8 minutes (Hosted Build Controller), completed 3 seconds ago

Latest Activity

Build last modified by Elastic Build (fabrikam) 3 seconds ago.

Request Summary

Request 1, requested by [redacted] 10.9 minutes ago, Completed

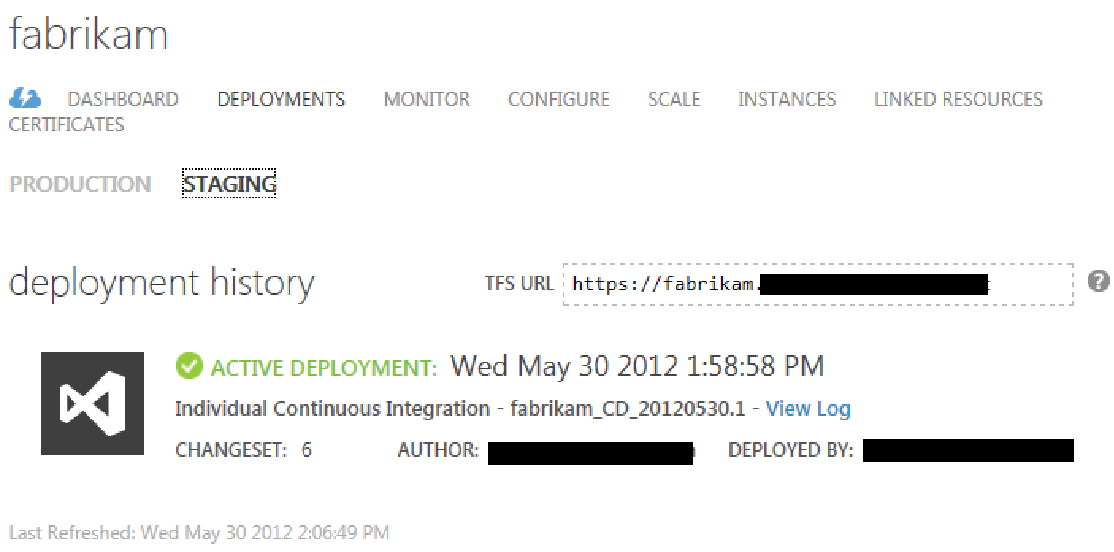
Summary

Debug | Any CPU

◢ 0 error(s), 3 warning(s)

$/Demo/AzureCloudService/AzureCloudService.sln ('AzureCloudService:Publish' target(s)) - 0 error(s), 3 warning (s), View Log File

14. In the Azure classic portal, you can view the associated deployment on the **Deployments** tab when the staging environment is selected.



fabrikam

DASHBOARD DEPLOYMENTS MONITOR CONFIGURE SCALE INSTANCES LINKED RESOURCES CERTIFICATES

PRODUCTION STAGING

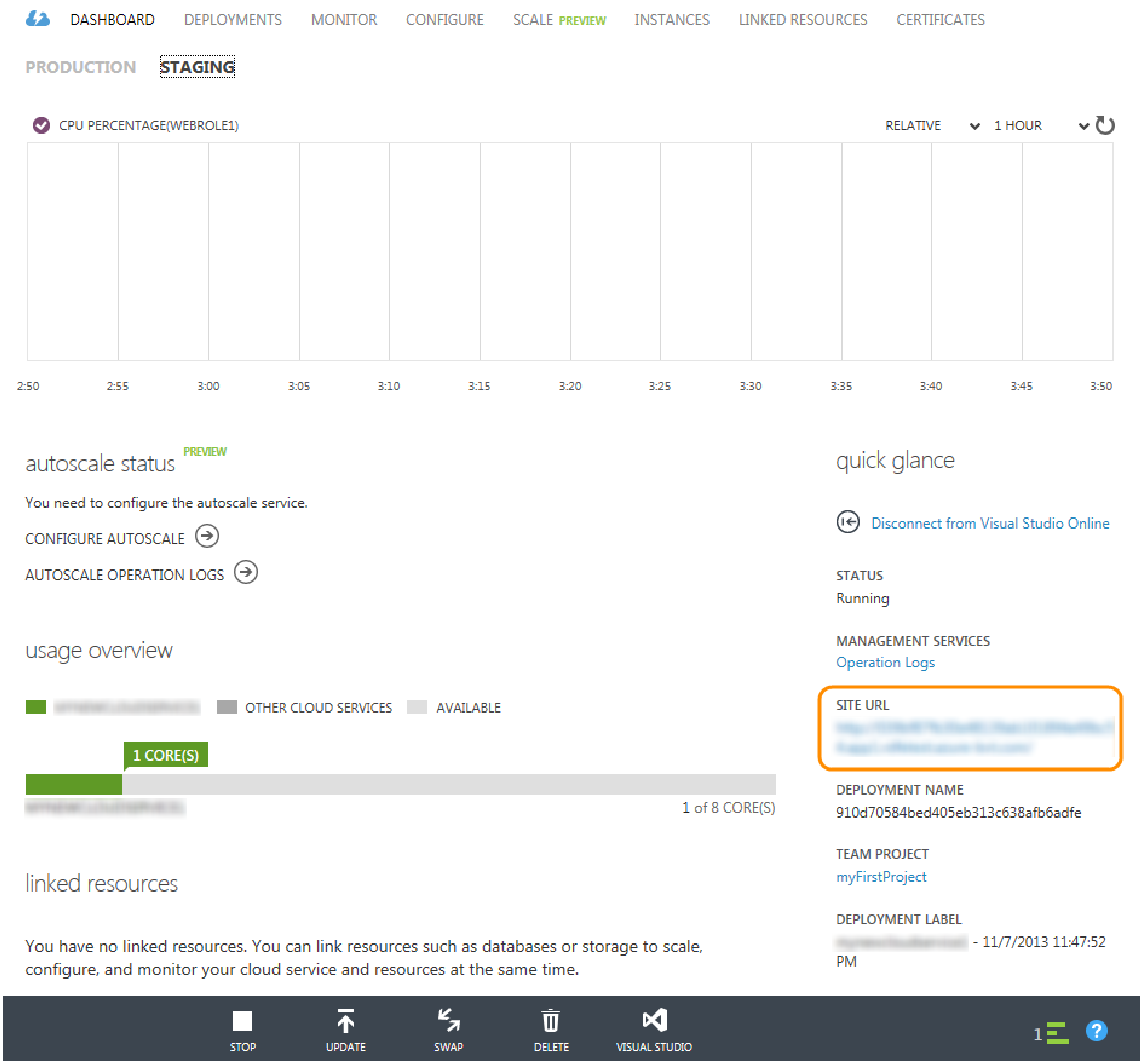
deployment history

TFS URL https://fabrikam.[redacted]

ACTIVE DEPLOYMENT: Wed May 30 2012 1:58:58 PM
Individual Continuous Integration - fabrikam_CD_20120530.1 - View Log
CHANGESET: 6 AUTHOR: [redacted] DEPLOYED BY: [redacted]

Last Refreshed: Wed May 30 2012 2:06:49 PM

15. Browse to your site's URL. For a web app, just click the **Browse** button on the command bar. For a cloud service, choose the URL in the **Quick Glance** section of the **Dashboard** page that shows the Staging environment for a cloud service. Deployments from continuous integration for cloud services are published to the Staging environment by default. You can change this by setting the **Alternate Cloud Service Environment** property to **Production**. This screenshot shows where the site URL is on the cloud service's dashboard page.



autoscale status PREVIEW

You need to configure the autoscale service.

CONFIGURE AUTOSCALE

AUTOSCALE OPERATION LOGS

usage overview

🟩 [redacted] ▨ OTHER CLOUD SERVICES □ AVAILABLE

1 CORE(S)

[redacted] 1 of 8 CORE(S)

linked resources

You have no linked resources. You can link resources such as databases or storage to scale, configure, and monitor your cloud service and resources at the same time.

quick glance

Disconnect from Visual Studio Online

STATUS
Running

MANAGEMENT SERVICES
Operation Logs

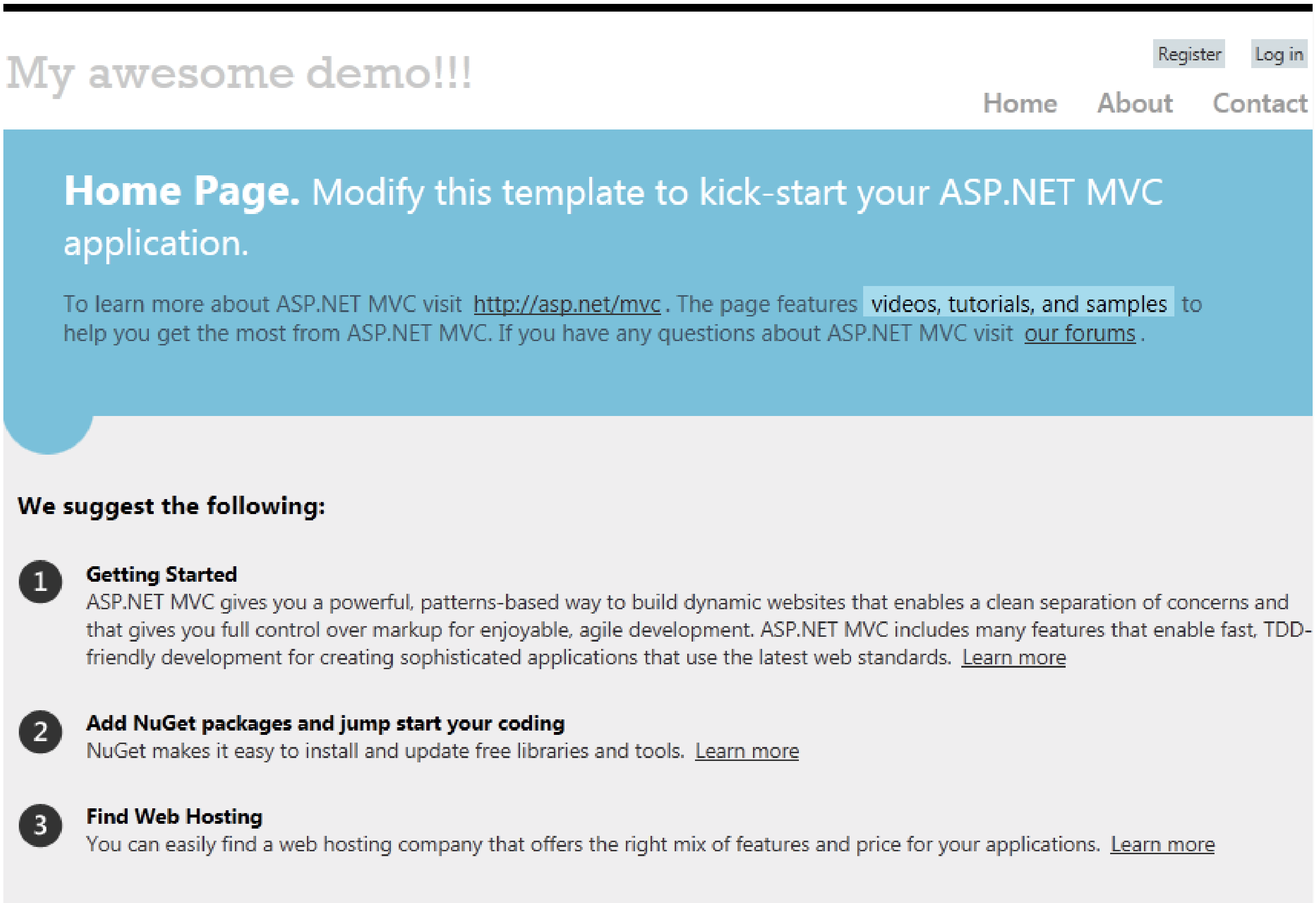
SITE URL
[redacted]

DEPLOYMENT NAME
910d70584bed405eb313c638afb6adfe

TEAM PROJECT
myFirstProject

DEPLOYMENT LABEL
[redacted] - 11/7/2013 11:47:52 PM

A new browser tab will open to reveal your running site.



My awesome demo!!!

Home Page. Modify this template to kick-start your ASP.NET MVC application.

To learn more about ASP.NET MVC visit http://asp.net/mvc. The page features videos, tutorials, and samples to help you get the most from ASP.NET MVC. If you have any questions about ASP.NET MVC visit our forums.

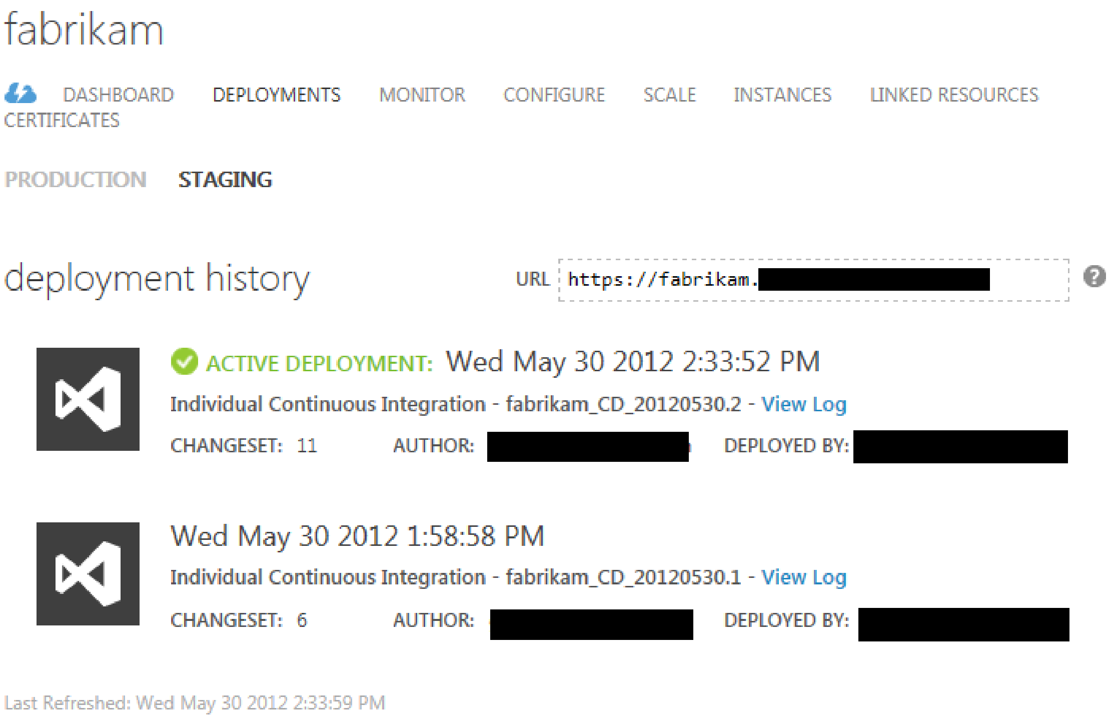
We suggest the following:

1. **Getting Started**
ASP.NET MVC gives you a powerful, patterns-based way to build dynamic websites that enables a clean separation of concerns and that gives you full control over markup for enjoyable, agile development. ASP.NET MVC includes many features that enable fast, TDD-friendly development for creating sophisticated applications that use the latest web standards. Learn more

2. **Add NuGet packages and jump start your coding**
NuGet makes it easy to install and update free libraries and tools. Learn more

3. **Find Web Hosting**
You can easily find a web hosting company that offers the right mix of features and price for your applications. Learn more

For cloud services, if you make other changes to your project, you trigger more builds, and you will accumulate multiple deployments. The latest one marked as Active.



fabrikam

deployment history

URL https://fabrikam.[redacted]

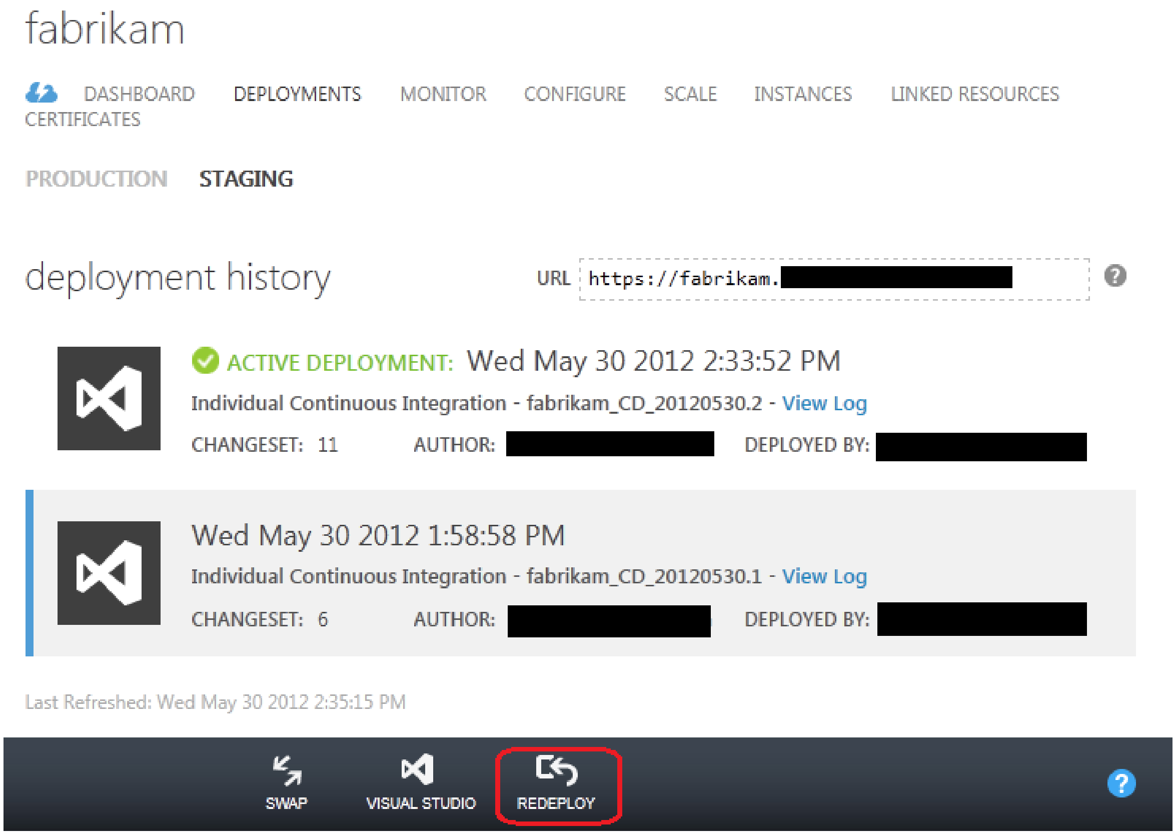
ACTIVE DEPLOYMENT: Wed May 30 2012 2:33:52 PM
Individual Continuous Integration - fabrikam_CD_20120530.2 - View Log
CHANGESET: 11 AUTHOR: [redacted] DEPLOYED BY: [redacted]

Wed May 30 2012 1:58:58 PM
Individual Continuous Integration - fabrikam_CD_20120530.1 - View Log
CHANGESET: 6 AUTHOR: [redacted] DEPLOYED BY: [redacted]

Last Refreshed: Wed May 30 2012 2:33:59 PM

5: Redeploy an earlier build

This step applies to cloud services and is optional. In the Azure classic portal, choose an earlier deployment and then choose the **Redeploy** button to rewind your site to an earlier check-in. Note that this will trigger a new build in TFS and create a new entry in your deployment history.



fabrikam

deployment history

URL https://fabrikam.[redacted]

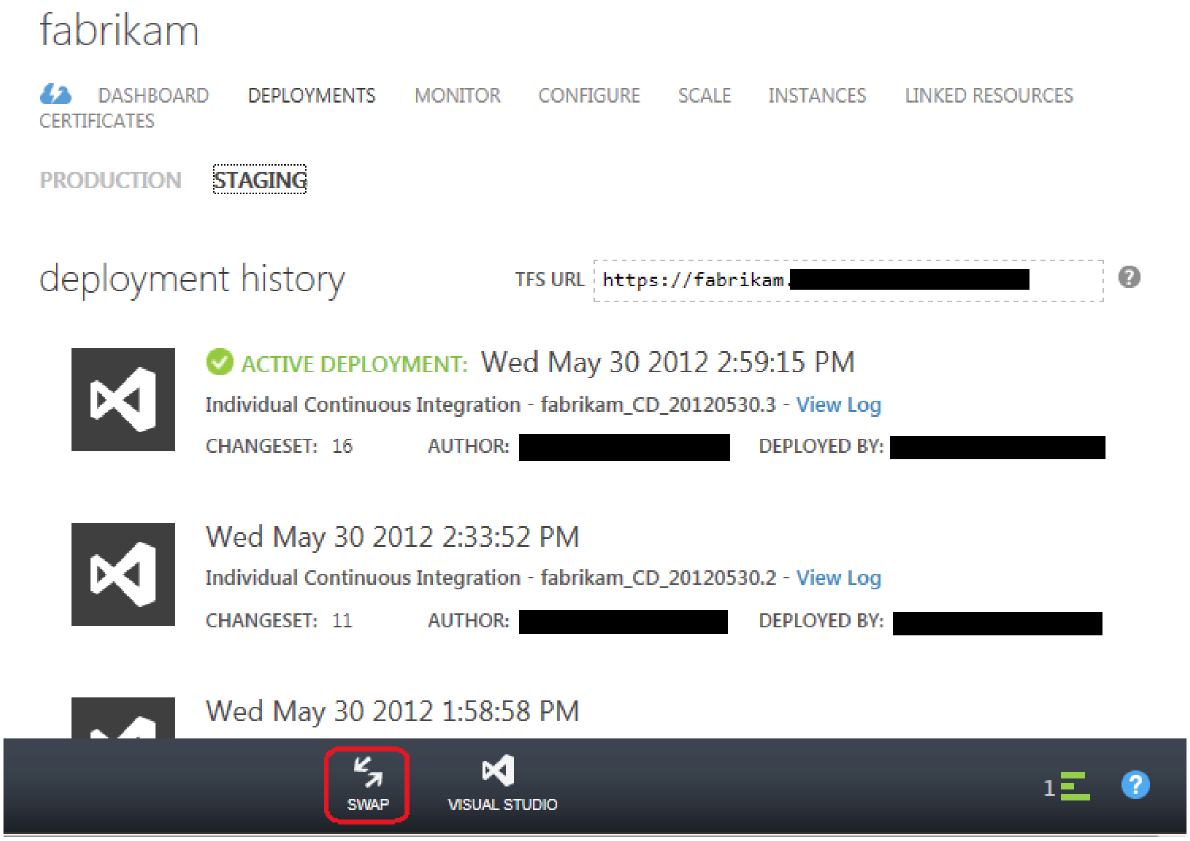
ACTIVE DEPLOYMENT: Wed May 30 2012 2:33:52 PM
Individual Continuous Integration - fabrikam_CD_20120530.2 - View Log
CHANGESET: 11 AUTHOR: [redacted] DEPLOYED BY: [redacted]

Wed May 30 2012 1:58:58 PM
Individual Continuous Integration - fabrikam_CD_20120530.1 - View Log
CHANGESET: 6 AUTHOR: [redacted] DEPLOYED BY: [redacted]

Last Refreshed: Wed May 30 2012 2:35:15 PM

6: Change the Production deployment

This step applies only to cloud services, not web apps. When you are ready, you can promote the Staging environment to the production environment by choosing the **Swap** button in the Azure classic portal. The newly deployed Staging environment is promoted to Production, and the previous Production environment, if any, becomes a Staging environment. The Active deployment may be different for the Production and Staging environments, but the deployment history of recent builds is the same regardless of environment.



fabrikam

deployment history

TFS URL https://fabrikam.[redacted]

ACTIVE DEPLOYMENT: Wed May 30 2012 2:59:15 PM
Individual Continuous Integration - fabrikam_CD_20120530.3 - View Log
CHANGESET: 16 AUTHOR: [redacted] DEPLOYED BY: [redacted]

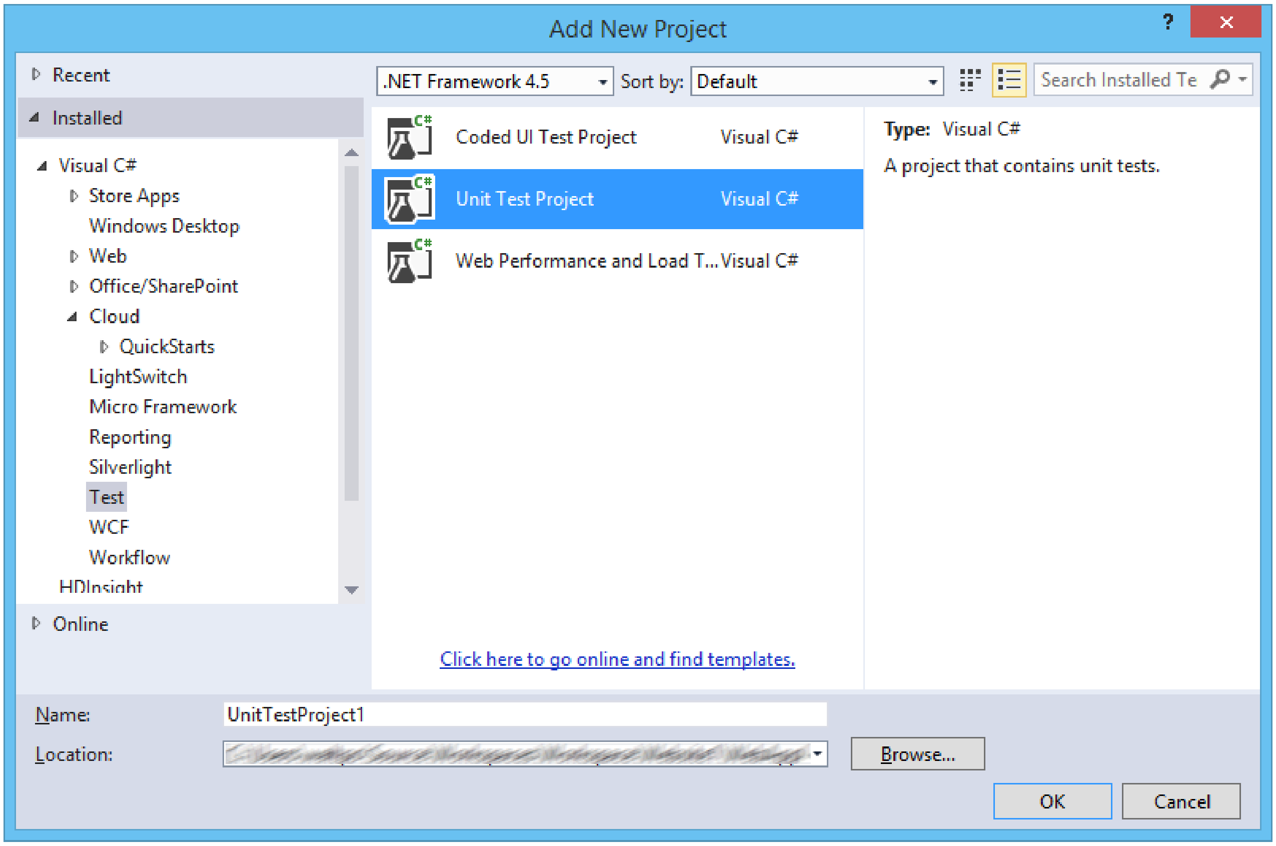
Wed May 30 2012 2:33:52 PM
Individual Continuous Integration - fabrikam_CD_20120530.2 - View Log
CHANGESET: 11 AUTHOR: [redacted] DEPLOYED BY: [redacted]

Wed May 30 2012 1:58:58 PM

7: Run unit tests

This step applies only to web apps, not cloud services. To put a quality gate on your deployment, you can run unit tests and if they fail, you can stop the deployment.

1. In Visual Studio, add a unit test project.



Add New Project

.NET Framework 4.5 Sort by: Default

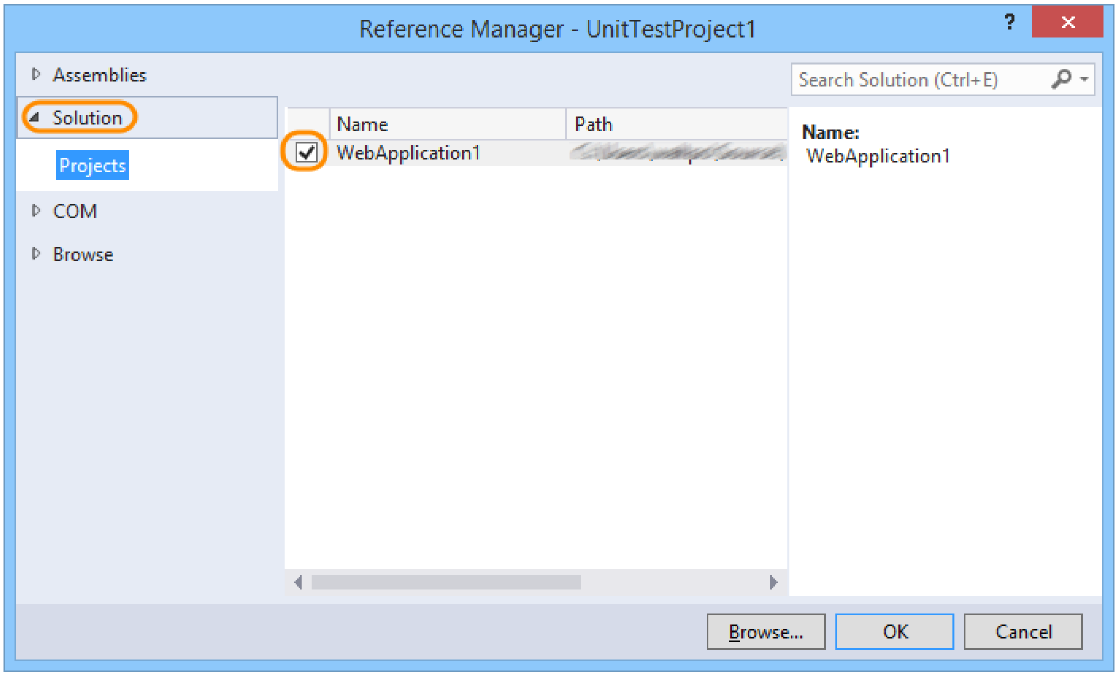
Type: Visual C#

A project that contains unit tests.

Name: UnitTestProject1

Location:

2. Add project references to the project you want to test.



Reference Manager - UnitTestProject1

	Name	Path
☑	WebApplication1	

Name: WebApplication1

3. Add some unit tests. To get started, try a dummy test that will always pass.

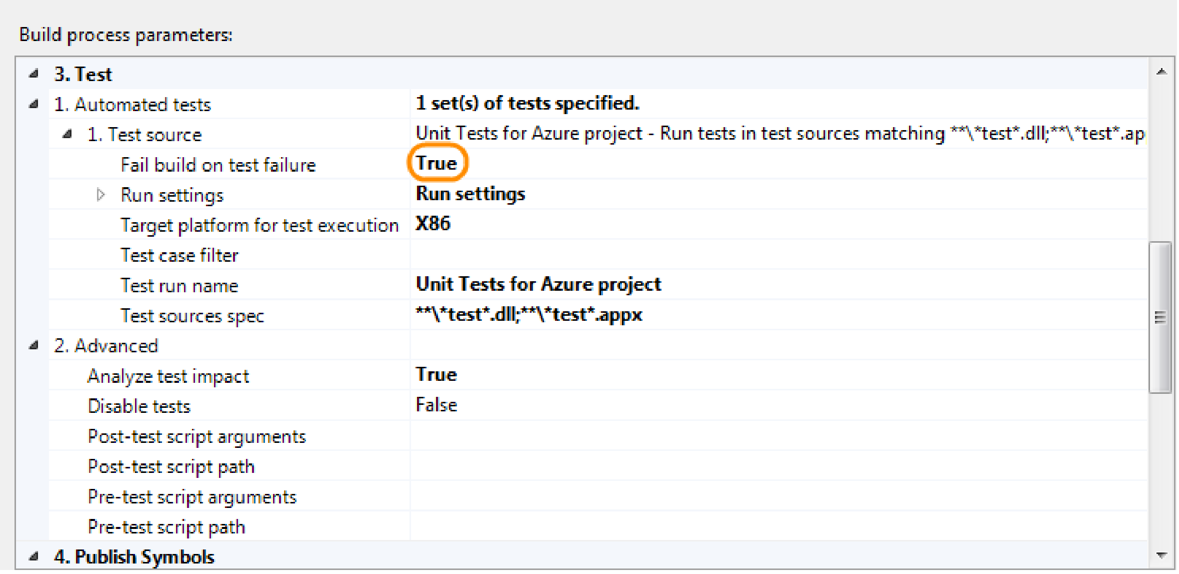
Copy

```
```
using System;
using Microsoft.VisualStudio.TestTools.UnitTesting;
```
```

~~~
namespace UnitTestProject1
{
    [TestClass]
    public class UnitTest1
    {
        [TestMethod]
        [ExpectedException(typeof(NotImplementedException))]
        public void TestMethod1()
        {
            throw new NotImplementedException();
        }
    }
}
```
~~~

4. Edit the build definition, choose the **Process** tab, and expand the **Test**node.

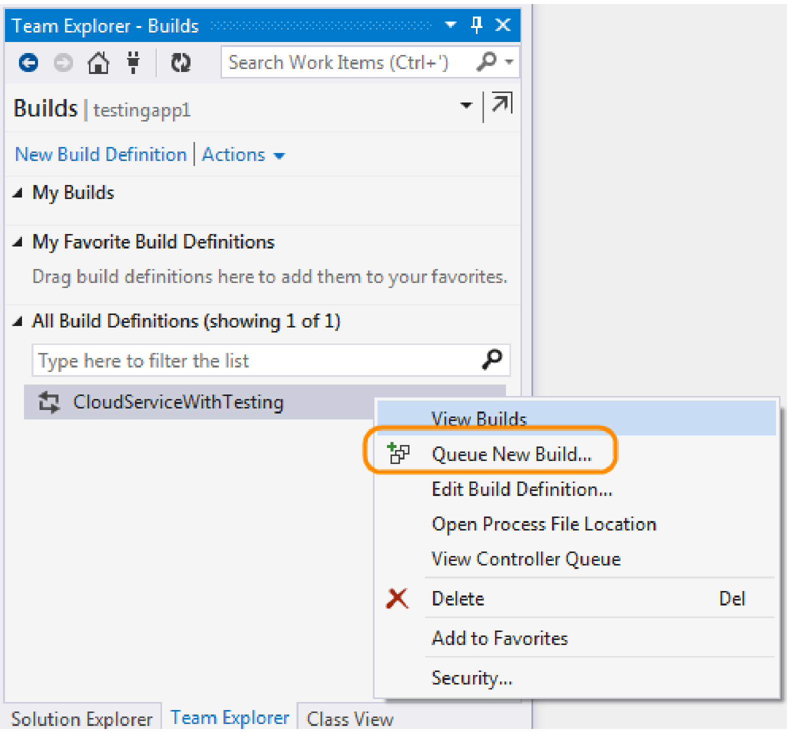
5. Set the **Fail build on test failure** to True. This means that the deployment won't occur unless the tests pass.

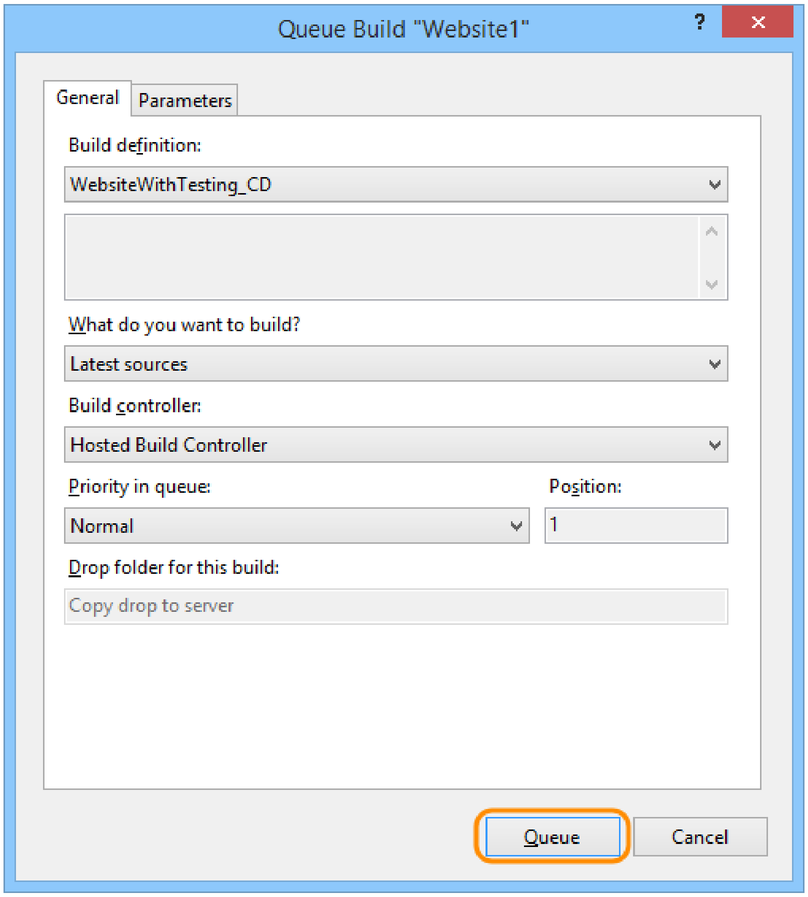


Build process parameters:

3. Test	
1. Automated tests	1 set(s) of tests specified.
1. Test source	Unit Tests for Azure project - Run tests in test sources matching ***test*.dll;***test*.ap
Fail build on test failure	True
Run settings	Run settings
Target platform for test execution	X86
Test case filter	
Test run name	Unit Tests for Azure project
Test sources spec	***test*.dll;***test*.appx
2. Advanced	
Analyze test impact	True
Disable tests	False
Post-test script arguments	
Post-test script path	
Pre-test script arguments	
Pre-test script path	
4. Publish Symbols	

6. Queue a new build.


Team Explorer - Builds
Search Work Items (Ctrl+')
Builds | testingapp1
New Build Definition | Actions
My Builds
My Favorite Build Definitions
Drag build definitions here to add them to your favorites.
All Build Definitions (showing 1 of 1)
Type here to filter the list
CloudServiceWithTesting
View Builds
Queue New Build...
Edit Build Definition...
Open Process File Location
View Controller Queue
Delete
Del
Add to Favorites
Security...
Solution Explorer
Team Explorer
Class View



Queue Build "Website1"

General | Parameters

Build definition: WebsiteWithTesting_CD

What do you want to build? Latest sources

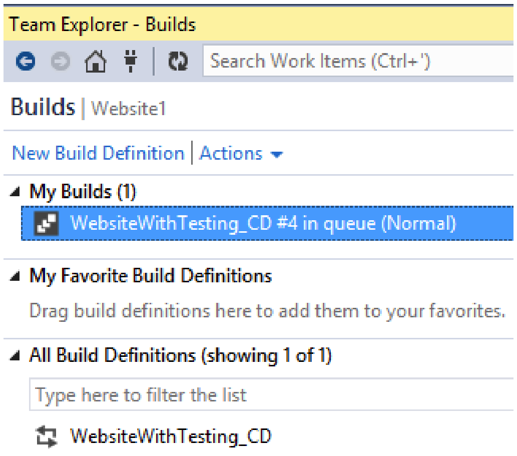
Build controller: Hosted Build Controller

Priority in queue: Normal

Position: 1

Drop folder for this build: Copy drop to server

7. While the build is proceeding, check on its progress.



Team Explorer - Builds

Search Work Items (Ctrl+')

Builds | Website1

New Build Definition | Actions

My Builds (1)

WebsiteWithTesting_CD #4 in queue (Normal)

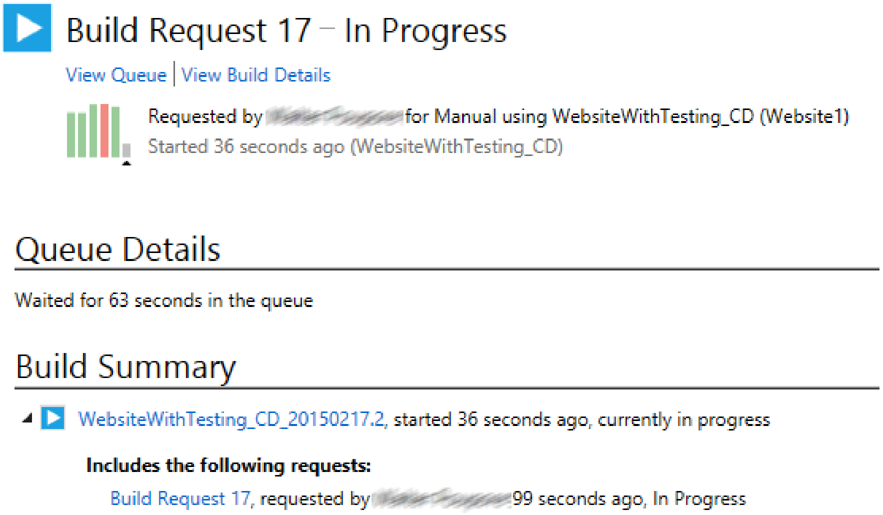
My Favorite Build Definitions

Drag build definitions here to add them to your favorites.

All Build Definitions (showing 1 of 1)

Type here to filter the list

WebsiteWithTesting_CD



Build Request 17 – In Progress

View Queue | View Build Details

Requested by [redacted] for Manual using WebsiteWithTesting_CD (Website1)

Started 36 seconds ago (WebsiteWithTesting_CD)

Queue Details

Waited for 63 seconds in the queue

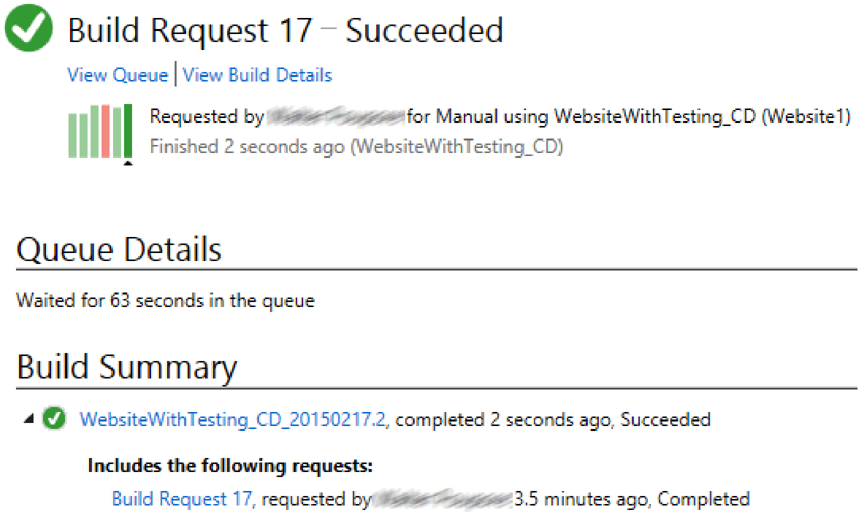
Build Summary

WebsiteWithTesting_CD_20150217.2, started 36 seconds ago, currently in progress

Includes the following requests:

Build Request 17, requested by [redacted] 99 seconds ago, In Progress

8. When the build is done, check the test results.



Build Request 17 – Succeeded

View Queue | View Build Details

Requested by [redacted] for Manual using WebsiteWithTesting_CD (Website1)
Finished 2 seconds ago (WebsiteWithTesting_CD)

Queue Details

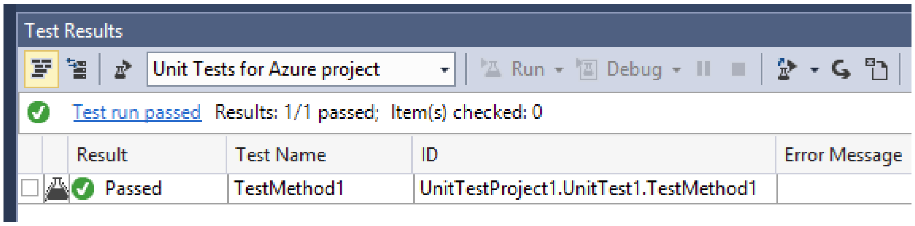
Waited for 63 seconds in the queue

Build Summary

WebsiteWithTesting_CD_20150217.2, completed 2 seconds ago, Succeeded

Includes the following requests:

Build Request 17, requested by [redacted] 3.5 minutes ago, Completed



Test run passed Results: 1/1 passed; Item(s) checked: 0

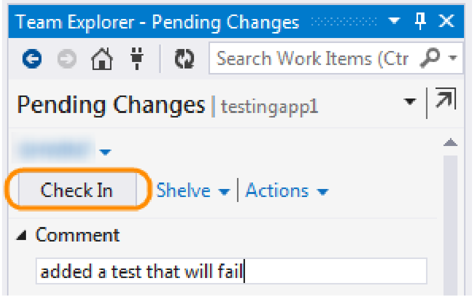
Result	Test Name	ID	Error Message
Passed	TestMethod1	UnitTestProject1.UnitTest1.TestMethod1	

9. Try creating a test that will fail. Add a new test by copying the first one, rename it, and comment out the line of code that states NotImplementedException is an expected exception.

```
```
[TestMethod]
//[ExpectedException(typeof(NotImplementedException))]
public void TestMethod2()
{
    throw new NotImplementedException();
}
```
```

Copy

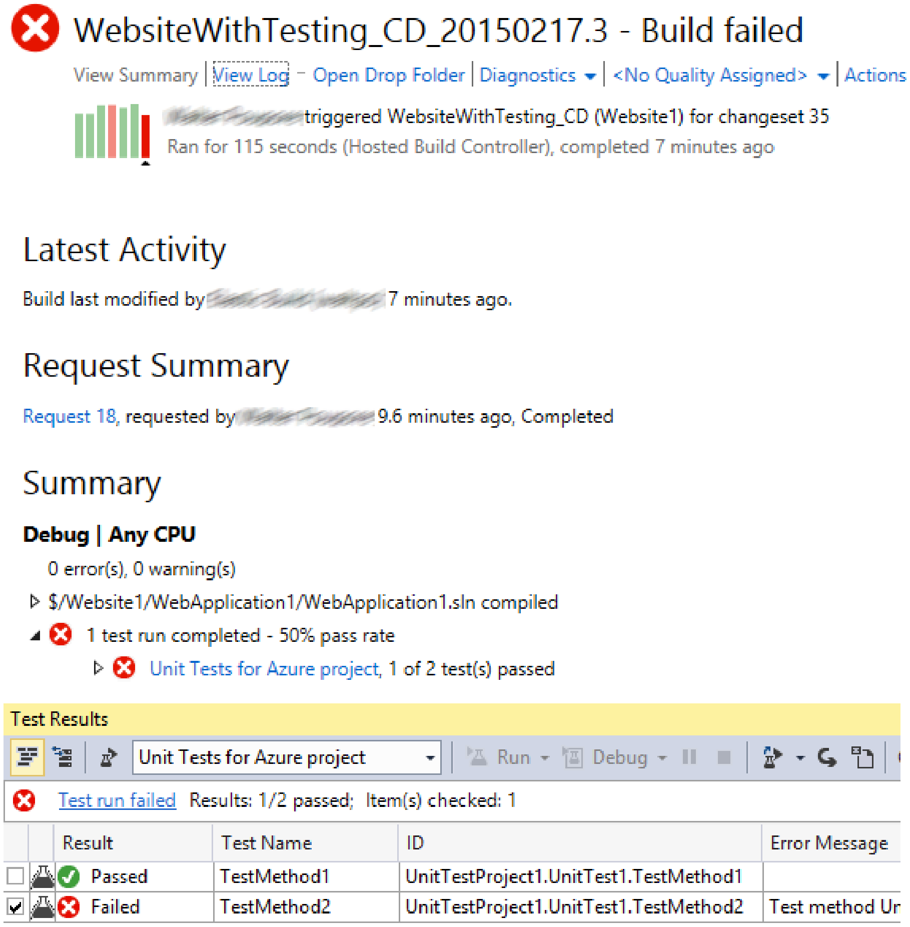
10. Check in the change to queue a new build.



Team Explorer - Pending Changes
Search Work Items (Ctr
Pending Changes | testingapp1
[redacted]
Check In
Shelve
Actions
Comment

added a test that will fail

11. View the test results to see details about the failure.



View Summary | View Log - Open Drop Folder | Diagnostics | <No Quality Assigned> | Actions

WebsiteWithTesting_CD_20150217.3 - Build failed

[redacted] triggered WebsiteWithTesting_CD (Website1) for changeset 35
Ran for 115 seconds (Hosted Build Controller), completed 7 minutes ago

Latest Activity

Build last modified by [redacted] 7 minutes ago.

Request Summary

Request 18, requested by [redacted] 9.6 minutes ago, Completed

Summary

Debug | Any CPU

0 error(s), 0 warning(s)

$/Website1/WebApplication1/WebApplication1.sln compiled

1 test run completed - 50% pass rate

Unit Tests for Azure project, 1 of 2 test(s) passed

Test Results

Test run failed Results: 1/2 passed; Item(s) checked: 1

	Result	Test Name	ID	Error Message
□	✓ Passed	TestMethod1	UnitTestProject1.UnitTest1.TestMethod1	
☑	✗ Failed	TestMethod2	UnitTestProject1.UnitTest1.TestMethod2	Test method Ur

Next steps