Hybrid Devops Demo – Setup Guide

|  |  |
| --- | --- |
| Version | Author |
| 1.0 draft | Benjamin Kirnich |
|  |  |
|  |  |

Contents

[Preconditions 2](#_Toc515292422)

[Azurestack 2](#_Toc515292423)

[VSTS Account 3](#_Toc515292424)

[Build Agent 3](#_Toc515292425)

[Azure 3](#_Toc515292426)

[Buildfile Location 4](#_Toc515292427)

[Setup 5](#_Toc515292428)

[VSTS Project 5](#_Toc515292429)

[Import Code Repository 6](#_Toc515292430)

[Create Build Definition 7](#_Toc515292431)

[Create Release Definition 8](#_Toc515292432)

[Testing 13](#_Toc515292433)

[Preparing the demo 16](#_Toc515292434)

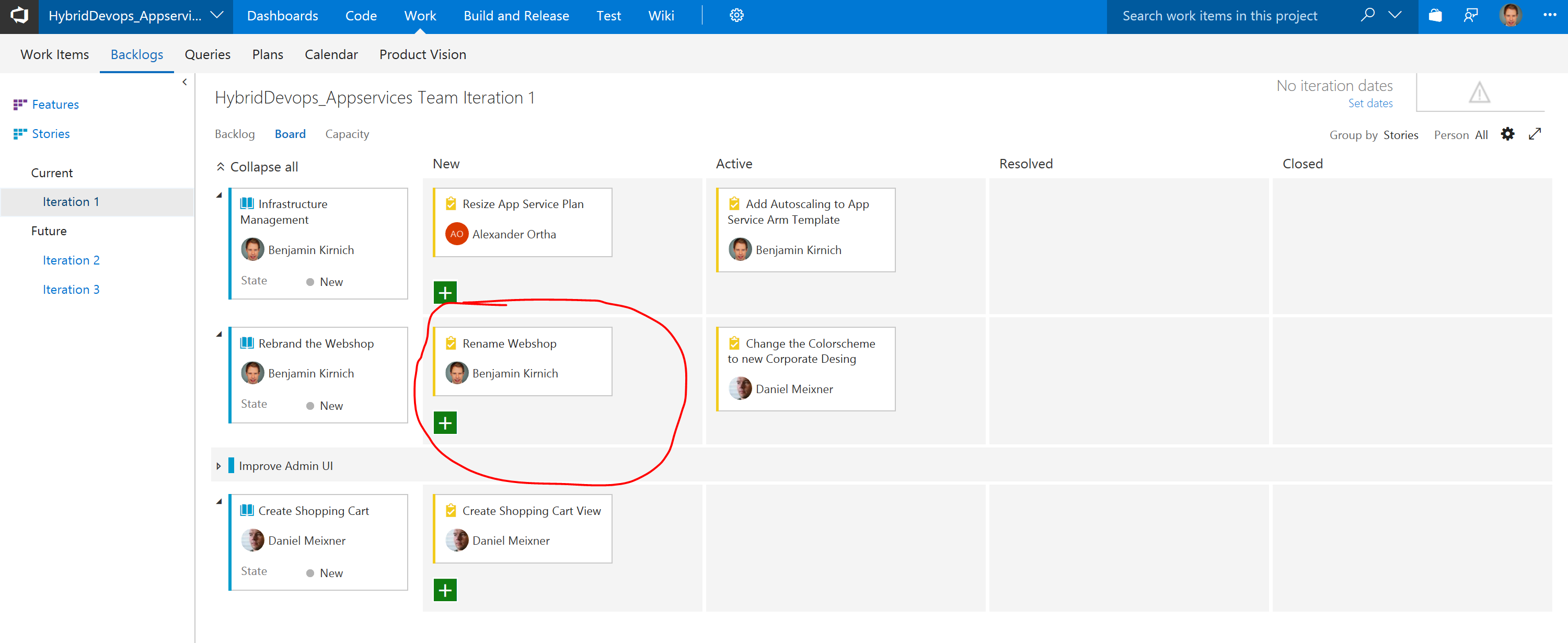
[Backlog and work items 16](#_Toc515292435)

[VSTS Dashboard 17](#_Toc515292436)

# Preparation

## VSTS Demo Task

Open your VSTS Project an ensure you have a new “ Rename Webshop Task” you can use for the Demo.



Delete the completed one from the last demo and create a new one you can use.

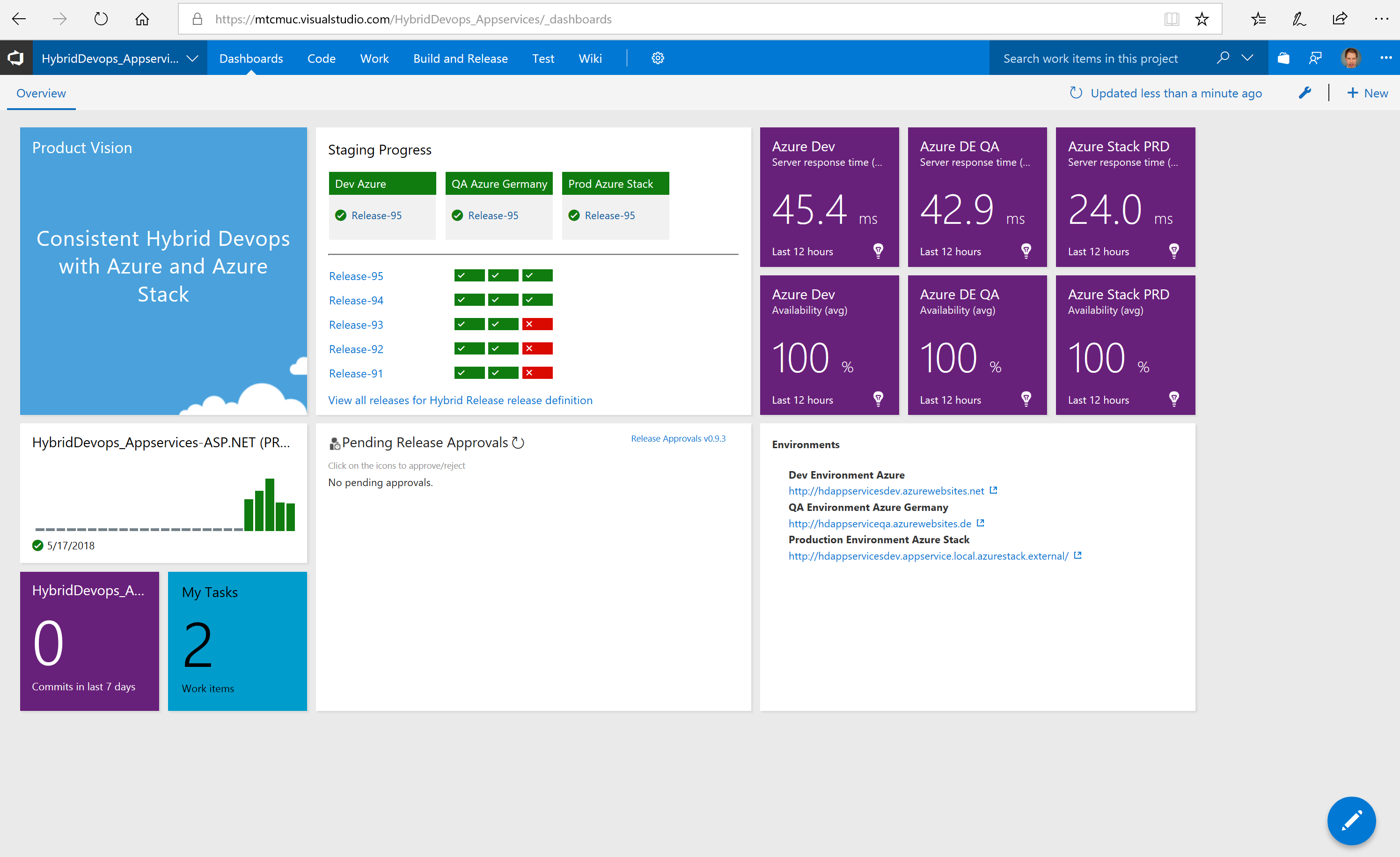
## Open Azurestack Tenant Portal

Open Azurestack Tenant Portal and navigate to the resource group holding your production demo environment.

# Demo

## Step 1.

Start on your VSTS Project Dashboard and explain the scene. You are a developer and you are working on a web application Project that uses three staging environments across Azure and Azurestack on premise.



Explain the Application Setup. The app is a ASP.net web application hosted on Azure app services and is using Azure SQL db as database platform. There is not a single piece of infrastructure to manage across all three staging environments. Also Application Insights is used across the cloud an on premise to monitor the application.

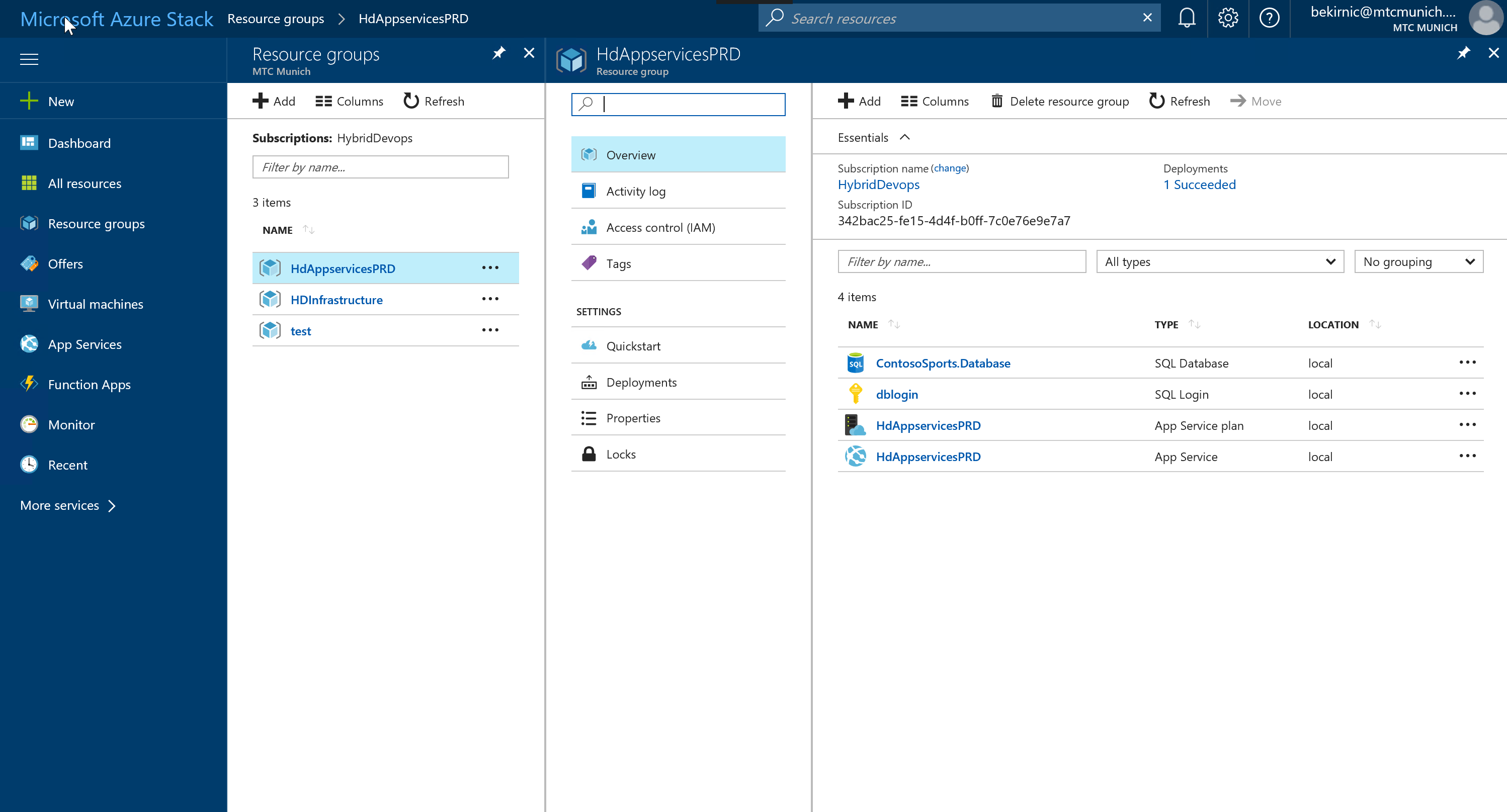
## Step2.

Open one of your environment and show the Application. The Application is just a little fantasy football ticket and merchandising shop. Mention the Webshop Title because this is the Item we are going to change as example for a feature change.



## Step 3.

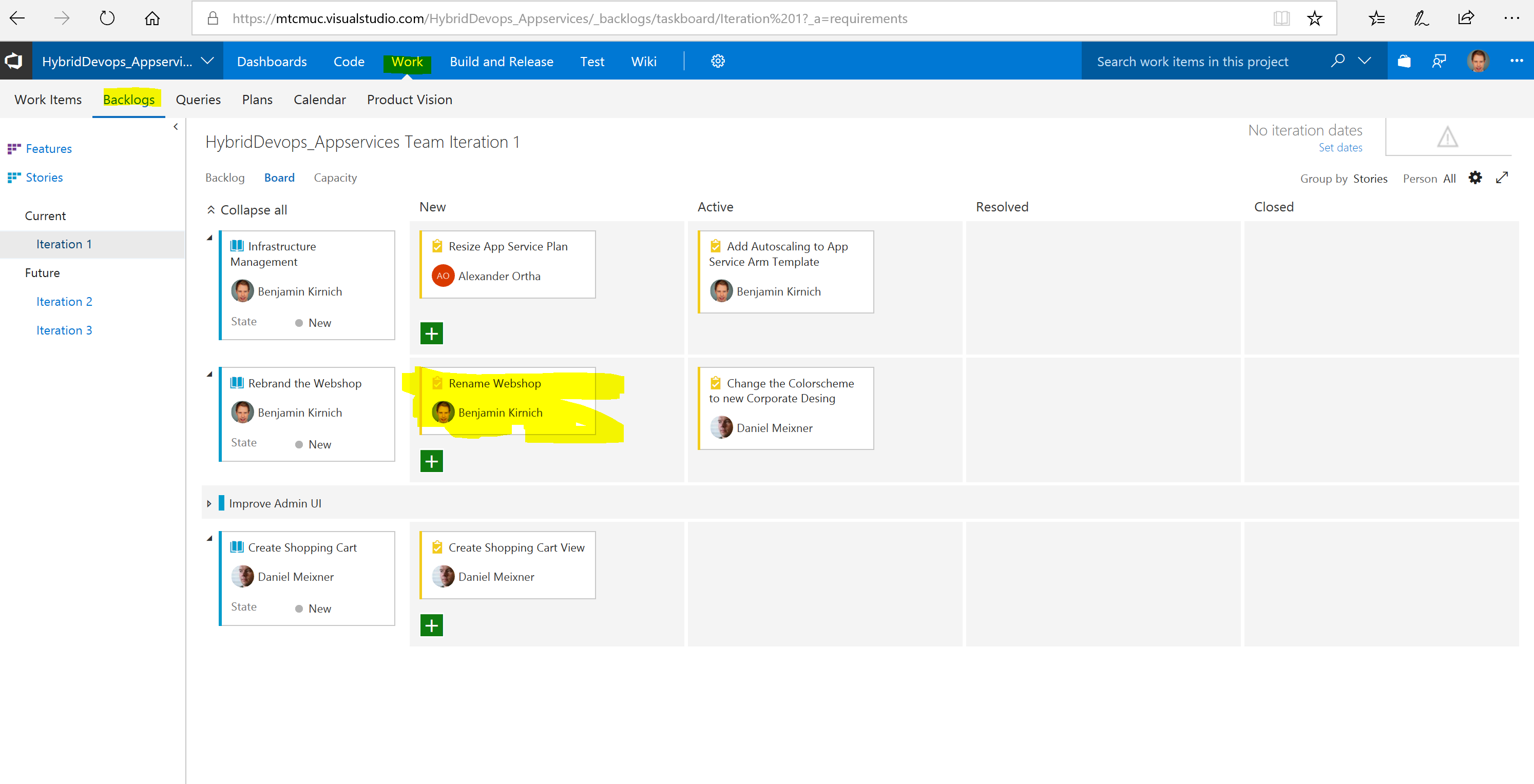
Open your Azure stack customer tenant portal and show the resource group. Shortly explain the used components.



Then delete the resource group. Since we want to show that the needed infrastructure is rolled out with the application if needed.

## Step 4.

In VSTS open the Work 🡪 Backlogs tap and navigate to your Kanban board. Explain your are Developer and after your daily Standup you are going to start your work and start with having a look at your work items.

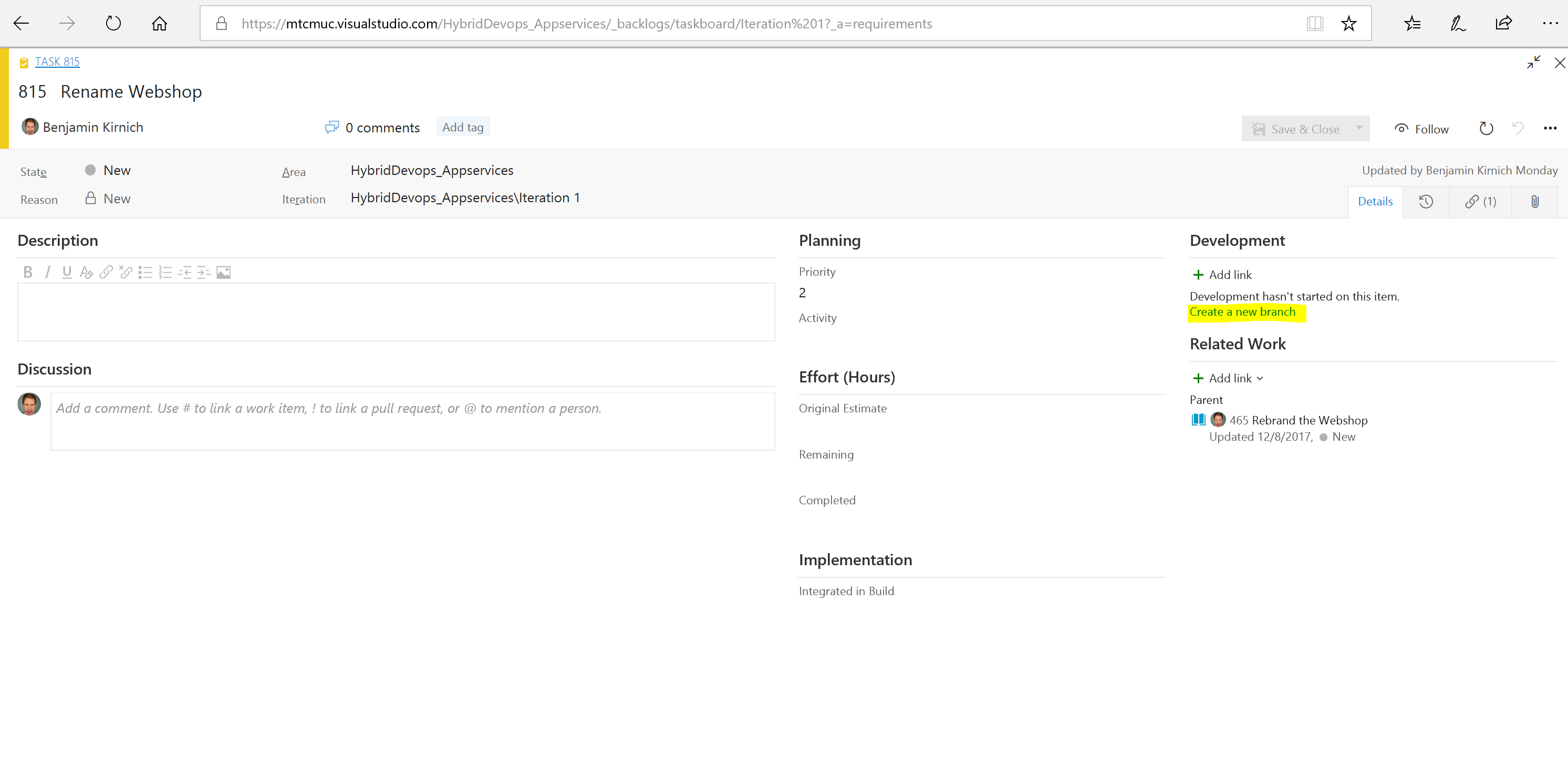


Open your “Rename Webshop Task”.

## Step 5.

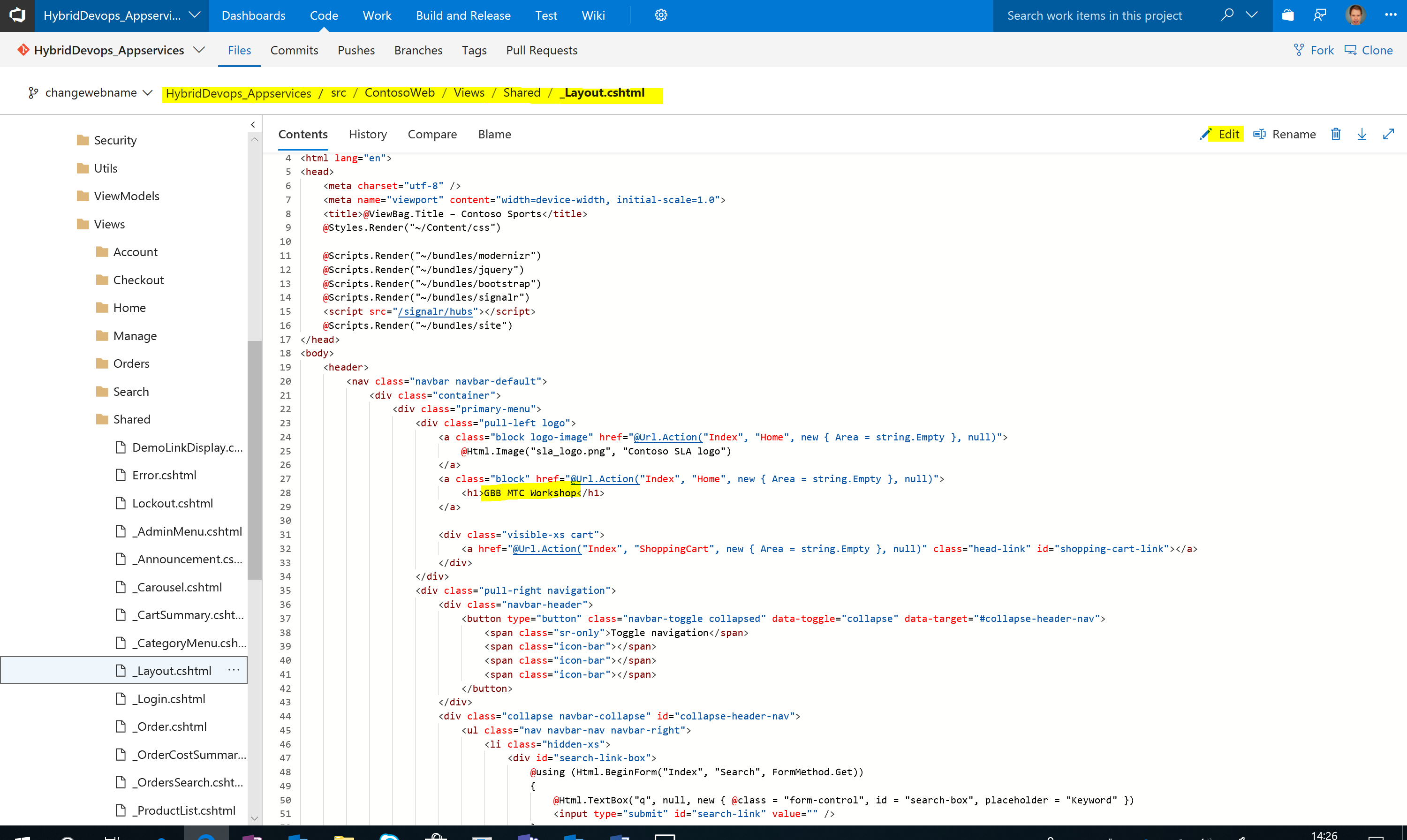
You are going to start on your work item. For the sake of this demo we are going to change the Webshop name as example, in reality this could be adding new features our customers are waiting for or any other change to the application.

Click on create new branch witch is the start of the usual developer workflow to create a new copy of the code to start working on.



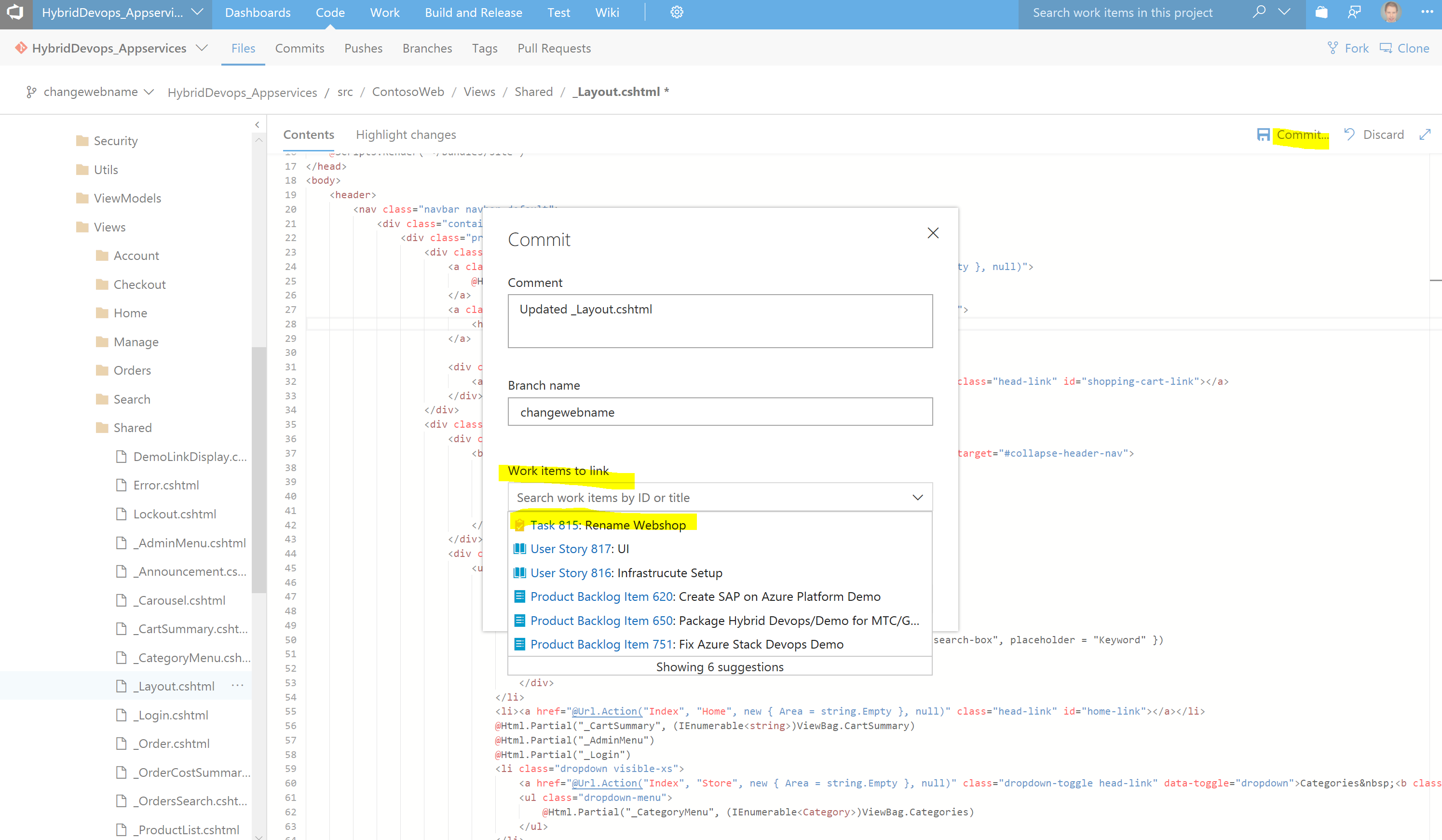
Give the branch any name, and point out that your work items has automatically linked to your branch.

You will be moved to the code view of your new branch. Normally you would start working on the code in your favorite IDE like Visual Studio as a developer for the simplicity of this demo we are using the browser as simple code editor.



Navigate to the \_Layout.cshtml file and klick on edit. Change the title at the marked code line to anything suiting your demo.

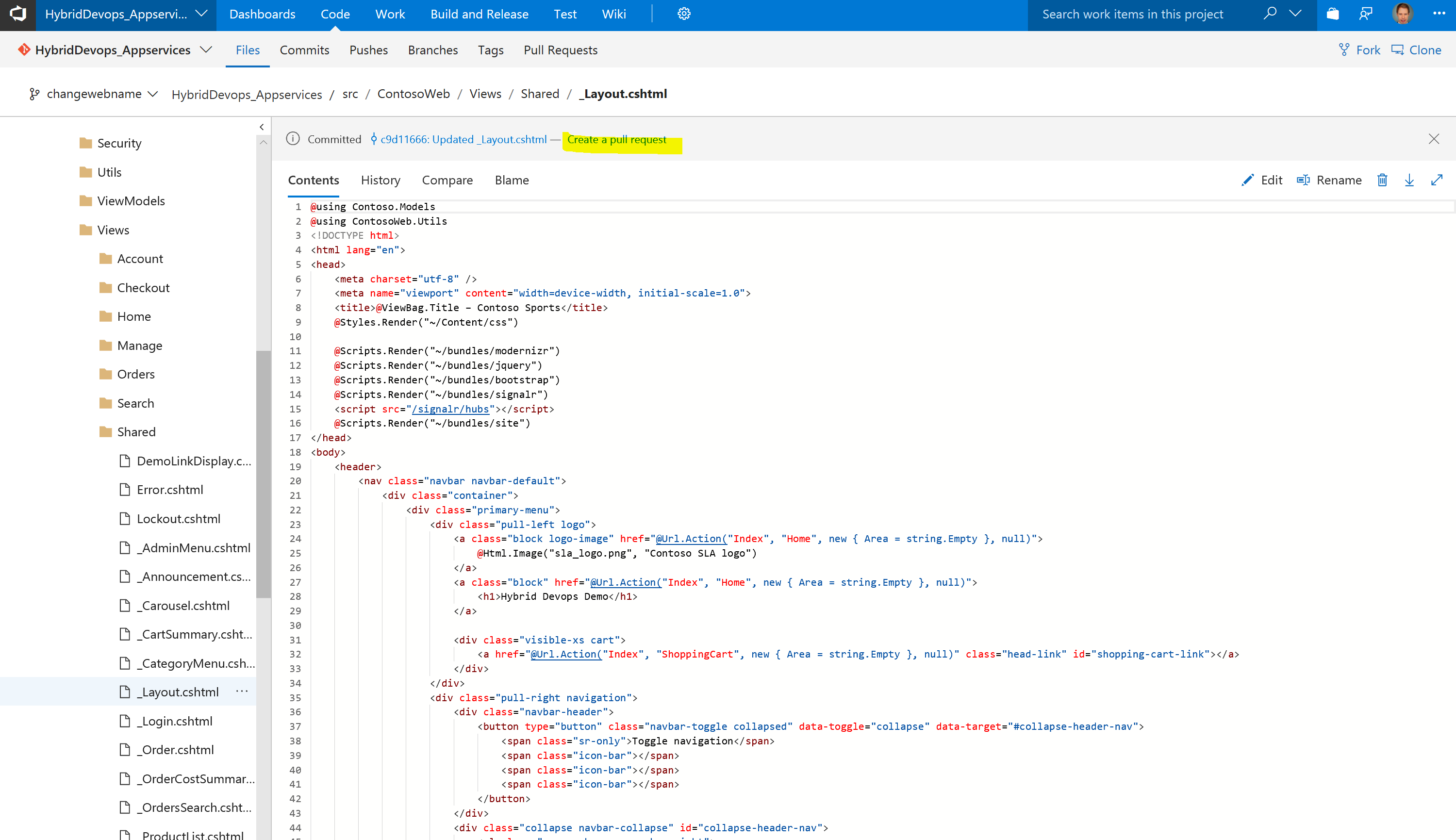
After the change klick on commit and select the work item you have completed.



Then klick on commit again.

## Step 6.

The last Step you have to to as Developer is to create a pull request. That means the project manager gets informed that you have finished the work on your code branch and the associated work items and you want to merge your code changes back to the master repository.

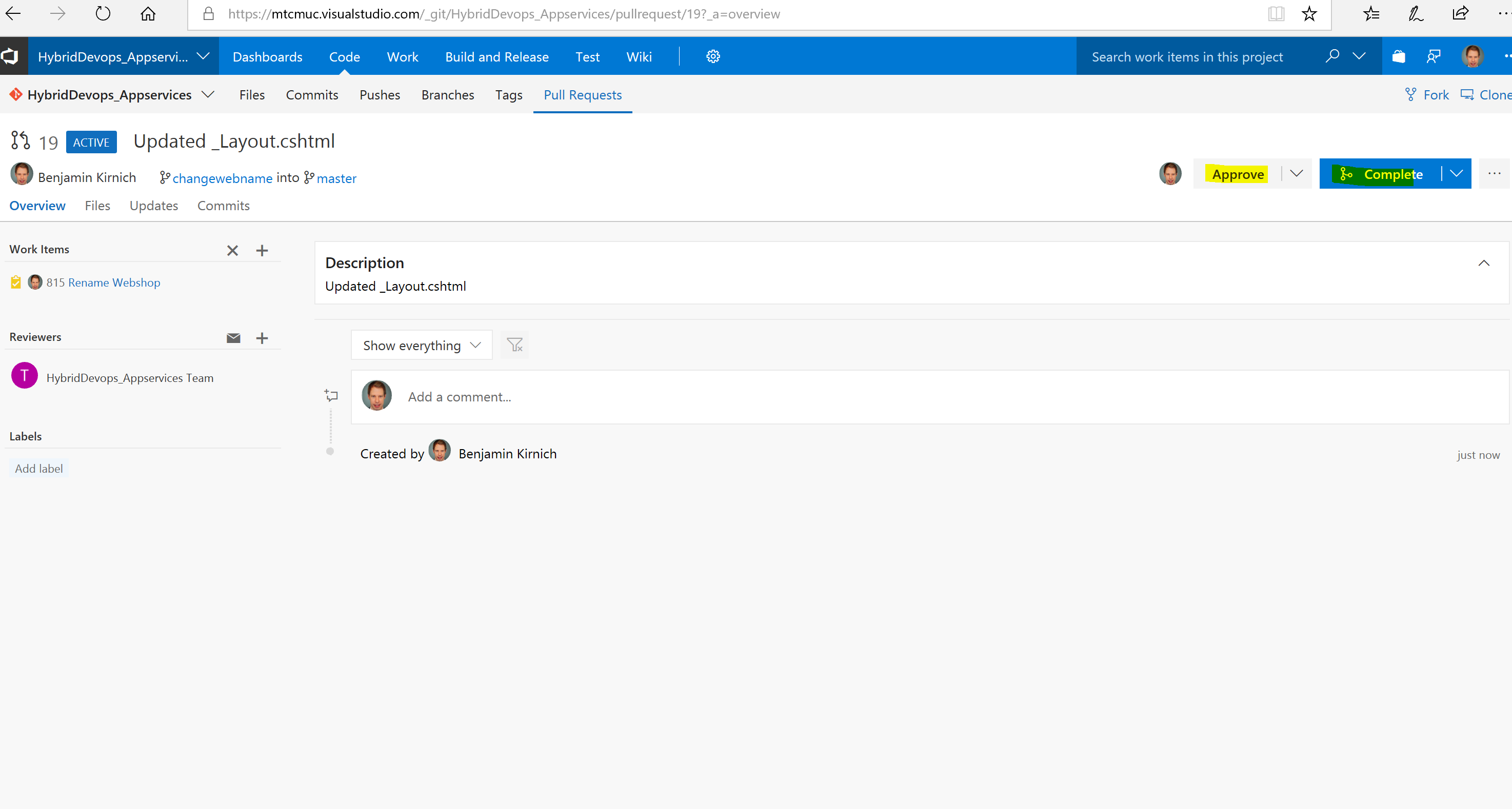


Klick on create pull request.

In the next Window show again the linked work items and code comparison how transparent VSTS is documenting the changes.

Then klick on create.

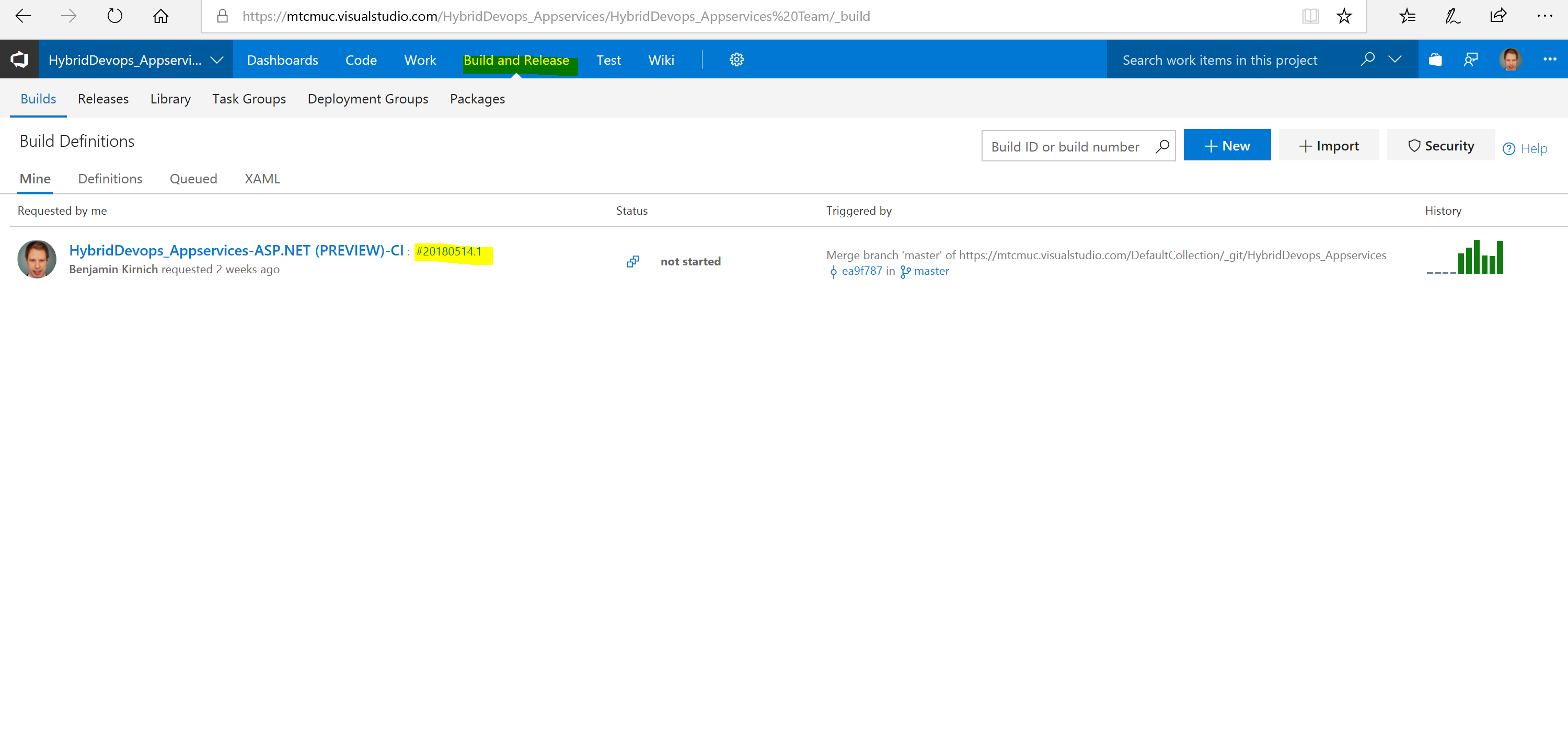
The next Windows is the view of your pull request. Since you are also the project manager in this demo klick on approve and then on complete to merge the code back to the master repository.



## Step 7.

The merge back to master repository will automatically trigger a build since we have configured a fully automated Continuous deployment Pipeline.

Klick on Build an Release 🡪 Build and then on the small Build Number behind the duild definition and then show that the build is currently running and has been kicked of.

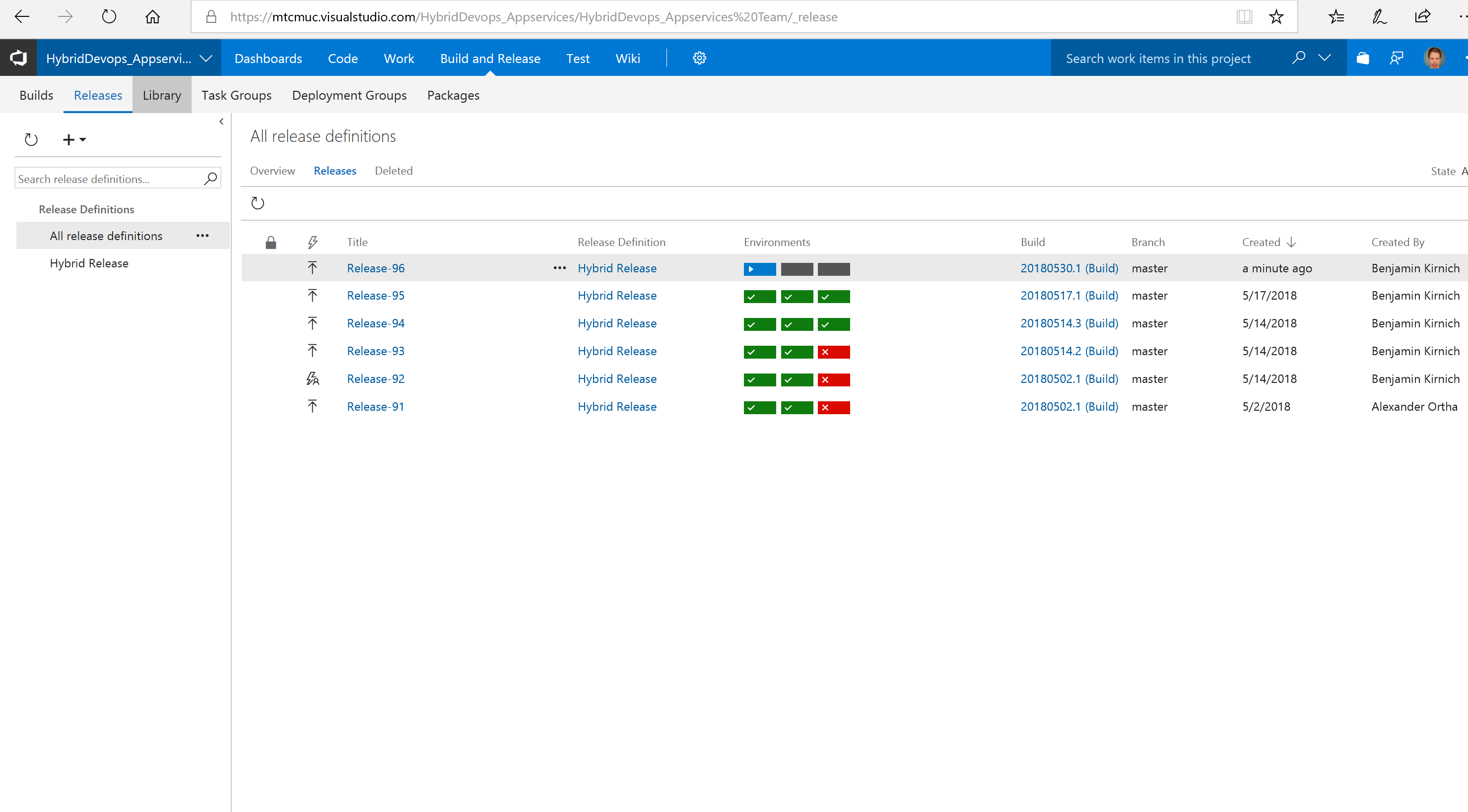


The build will take a minute or so. During the wait time go back to the web page of your Azure stack production website and show that its not reachable anymore since you have deleted the resource group earlier. You can also go back to the Azure stack tenant portal and show that the full resource group including all resources is gone.

Step 8.

The build should now be done and the release of the dev environment should already be running since the build was successfully. Normally we would run a lot of test of the build as well as after all deployments to test our applications. We have skipped this because auf wait times for this demo but the process would be exactly the same.

Go back to your Dashboard or to the release tab and show that the deployment to the dev environment is already running.



This will also take a minute or so.

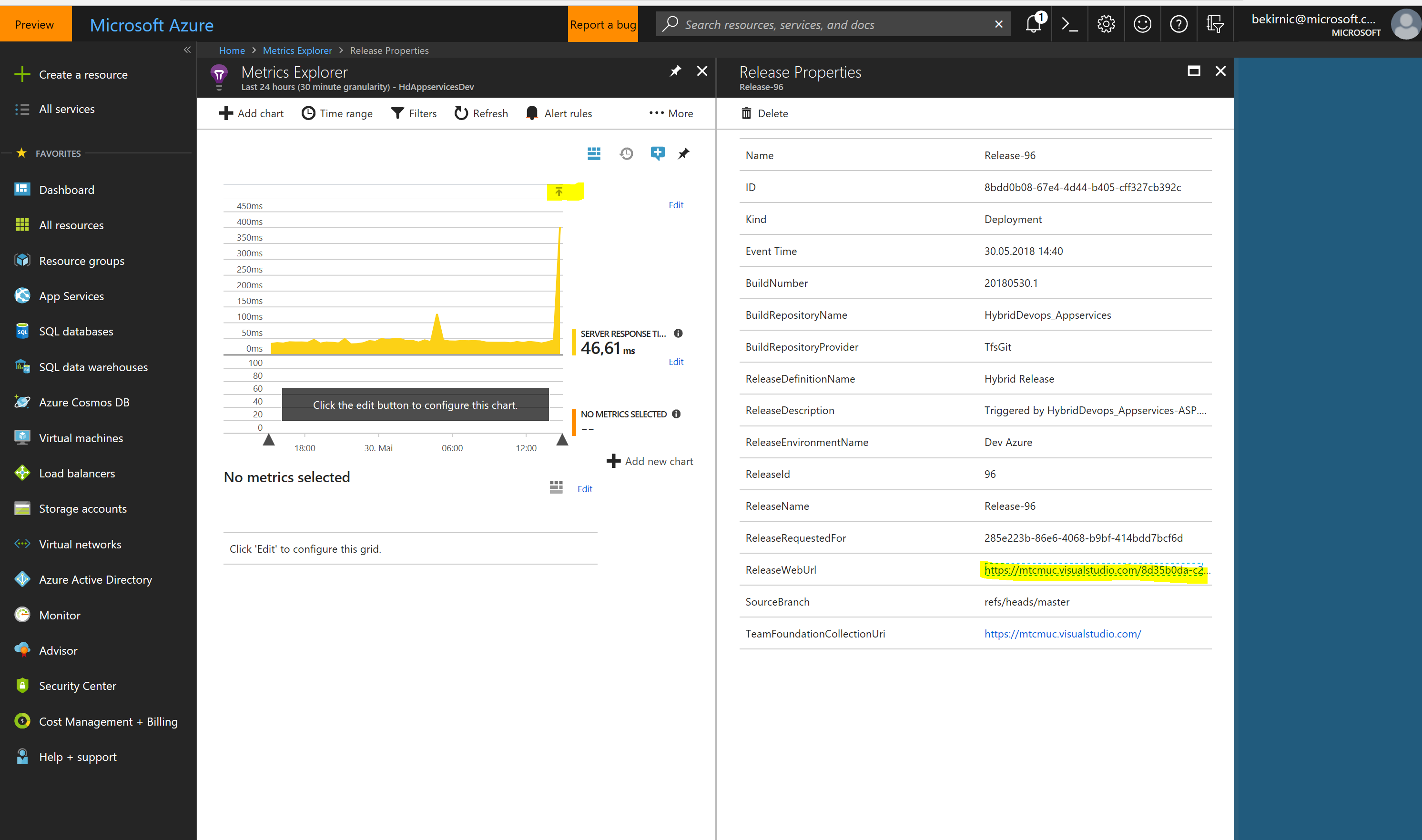
## Step 9.

Wait until the deployment of your dev environment is finished. When it is finished open the application in the browser and show that the changed title is now online in the dev environment.

Also show afterwards via the dashboard or release tab that the deployment to the QA environment has automatically been started.

During the QA deployment takes place we want to show how VSTS and Application Insights can bring some transparency to your deployments.

Klick on one of the Application Insights widgets of the DEV environment on your dashboard. Otherwise you have to manually open the Application Insights Instance of the DEV environment in the Azure portal and navigate to the metrics explorer.



If you do this the first time it can be necessary to edit the chart, so edit it and just choose server response time as metric.

Sometimes during or after new release your application can behave differently or unexpectedly.

The Set release annotation task in your release definition has set a marker in Application Insights when the new release is deployed to the environment.

Show the mark above the response time spike and click on it because we are curious why our application suddenly has such a response spike.

In the opening window we can see that Release – X has been deployed to the environment at this time. And now we would like to know what has changed in that release and is maybe forcing the change in behavior.

Click on the ReleaseWebUrl Link.

You get back to the Relese Task in VSTS. Klick on work items and show that you have full transparency who worked on which task in the fresh deployed release.

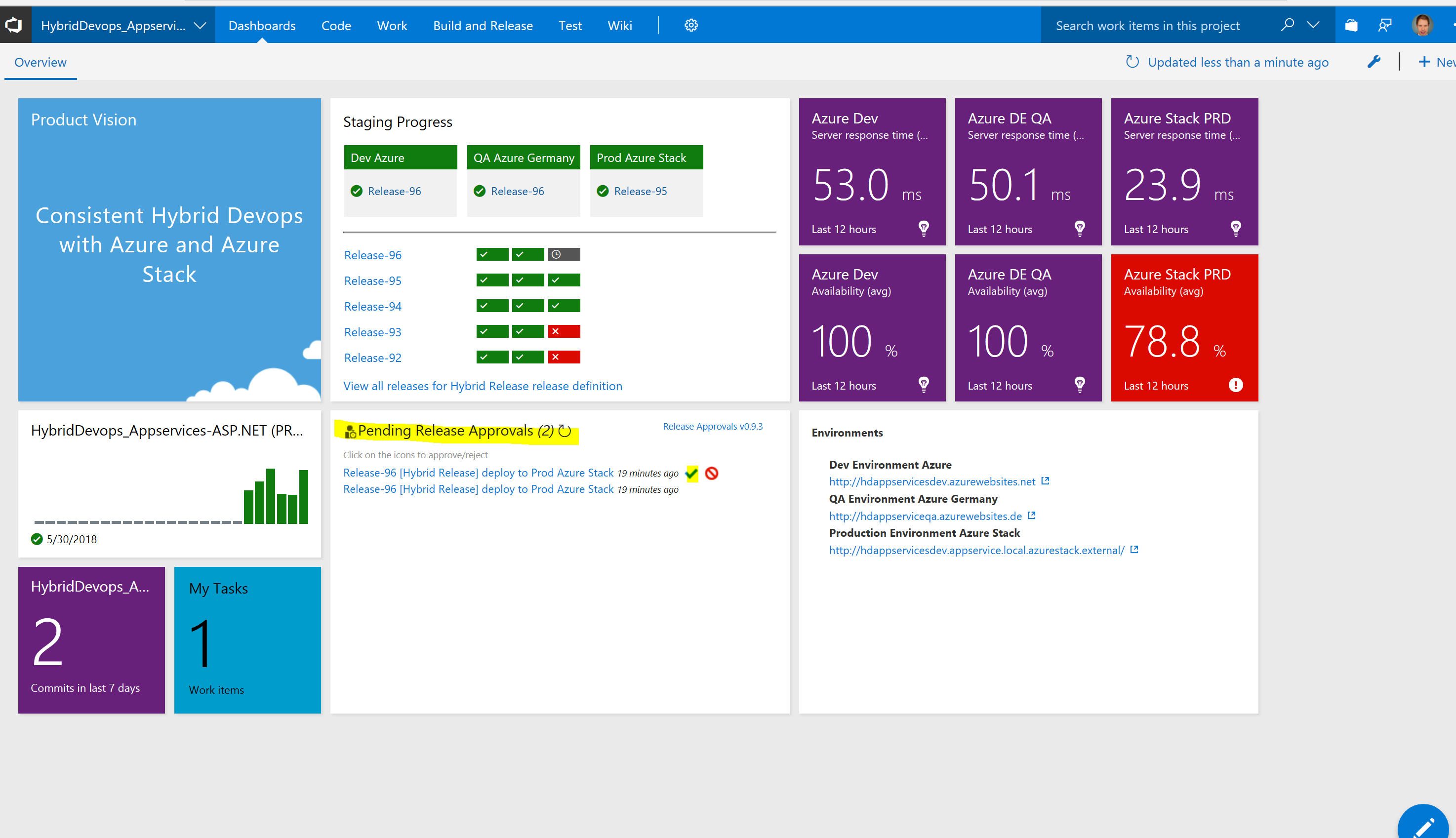
Klick on commits and then on one of the items in the list. You get the code dif comparison what lines of code exactly where changed and how. So you can always relate code changes to the monitoring numbers.

## Step 9.

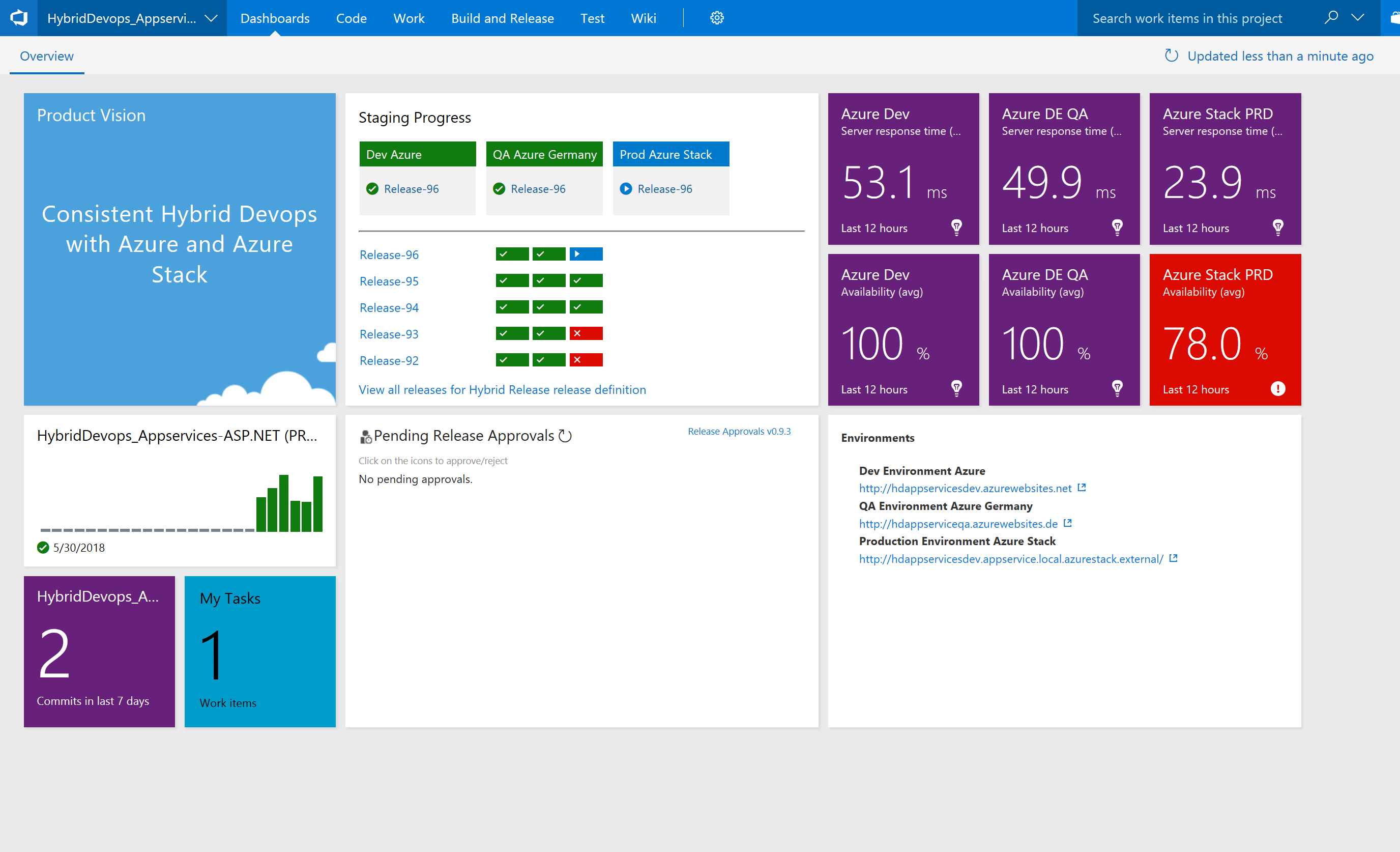
Your QA environment should already be deployed. Open it and show the changed title.

To demonstrate the control possibilities of the VSTS release pipeline we have added an approval step before something gets deployed to the production environment.

Go back to your dashboard. If you are using the release approval widgets which comes from the extension gallery you can approve the task right from your dashboard. Otherwise you must drill into the release within the release tab.



Approve the release to the production environment and reload the dashboard.



Your production environment is now in progress. Its also worth to mention that the project team is aware of the problem with the production environment since the application insights widget makes it visible directly on the projects dashboard

## Step 10.

Wait until the production deployment has finished. You have to refresh the dashboard from time to time.

Open the application in the browser when finished. This can take some extra time since asp does a compile at first load and the environment was completely deleted before.

Show that the changed title was rolled out to the Azure stack environment including all necessary services for the environment.

Open your Azure stack tenant portal and refresh it. Show that the resource group including all services is back there.

Congratulations you now have demonstrated how to stage an application change fully automated through three staging environments in three clouds in a hybrid scenario within minutes. And that all with end to end documentation.