

Azure DevOps

Creating an end to end Pipeline using Microsoft azure DevOps

Abstract

To get a basic idea about Azure DevOps and its benefits.

Contents

Azure DevOps CI/CD Pipeline	3
DevOps	3
DevOps Stages	5
A) Code Commit and Version Control	5
B) Code Build	5
C) Units and Integration Tests	5
D) Code Quality Checks	5
E) Artifacts Upload	5
F) Artifacts Download.....	5
E) Deployment	5
G) Notifications Configuration.....	5
H) Maintenance	5
Problem Statement.....	5
Why DevOps?	5
Azure DevOps	6
1. Azure Pipelines.....	6
2. Azure Artifacts.....	6
3. Azure Repos	6
4. Azure Test Plans.....	6
Tasks Used In Azure Build Pipeline –.....	7
Tasks Used in Azure Release Pipeline –.....	7
Creating WebApps for Application Deployment using Azure Portal.	8
Building CI/CD Pipeline with Azure Devops.....	11
Azure Repos.....	14
Build Pipeline	15
Azure Artifacts	23
Release Pipeline	29
Adding Notification services.....	35

Azure DevOps CI/CD Pipeline

DevOps

DevOps is the blending of the terms development and operations, meant to represent a collaborative or shared approach to the tasks performed by a company's application development and IT operations teams. The term DevOps is used in several ways. In its most broad meaning, DevOps is an operational philosophy that promotes better communication between these teams -- and others. In its most narrow interpretation, DevOps describes the adoption of automation and programmable software development and infrastructure deployment and maintenance. The term may also label a culture that strategically looks at the entire software delivery chain, overseeing shared services and promoting the use of new development tools and best practices.

While DevOps is not a technology, there are tools commonly used in DevOps environments. These include continuous integration and continuous delivery or continuous deployment tools, with an emphasis on task automation. Other products to support DevOps include real-time monitoring and incident response systems as well as collaboration platforms.

A DevOps approach can coexist with Agile software development; IT service management frameworks, such as IT Infrastructure Library; project management directives, such as lean and six sigma; and other strategies to execute IT projects to meet business needs. While it is typically associated with Agile development, the two methodologies do not need to be used in concert.

Patrick Debois, a software development consultant, is credited with coming up with the term DevOps in 2009 by naming a conference DevOps Days. The concept of DevOps was then popularized with the book The Phoenix Project in 2013. The Phoenix Project is a novel about DevOps, using a fictional narrative as an example to help IT managers understand the concepts and benefits of collaboration and shared technologies. It illustrates the endemic problems in IT that DevOps solves, such as division and mistrust between developers and IT admins, lack of automation leading to long lead times and errors, and misalignment of business requirements and projects to serve them.

Cloud computing and software-defined infrastructures, microservices, containers and automation are commonly implemented concurrently with DevOps methodologies.

DevOps vs. Waterfall development

In Waterfall development, developers test new code in an isolated environment for quality assurance (QA) and -- if requirements are met -- release the code to operations for use in production, usually in a bundle with other releases so that operations can tightly control the process. The operations team deploys the program and maintains it from that point on. Waterfall approaches usually engender a long time between software releases and because developer and operations teams work separately and the development team is not always aware of operational roadblocks that might prevent code from working as anticipated.

The DevOps approach seeks to meld application development and deployment into a more streamlined process that aligns development, QA and IT operations team efforts with fewer gates and more continuous flow. This approach also shifts some

of the operations team's responsibilities onto the development team to facilitate continuous development, continuous integration, continuous delivery and continuous monitoring processes. It is a way to tear down the metaphorical silos that isolate steps in the software delivery process to release code faster and more often.

DevOps and Agile development

Agile is a software development approach focused on incremental and rapid cycles of code creation and delivery referred to as sprints. Each sprint iterates upon the last, creating a high level of flexibility, as changes and scope and direction can be evaluated between each cycle. However, it is possible for the original vision of a project to be lost through this cycle.

DevOps arose from the success in the speed of development in Agile, when it became clear that there is a lack of communication between development and operations teams that put a significant hindrance on speed and flexibility of code delivery to users.

As Agile development became more efficient, it became clear having development and operations work apart from each other was inefficient. Before DevOps, development and operations teams worked in a way where those who wrote the code had separate objectives and leadership from those who deployed and supported the code. With DevOps and Agile, communication between development and operations ensures code can be managed by both teams fluently without miscommunications or confusion. DevOps does not have an official framework, nor does it consider speed as a core focus (rather speed in development is achieved through what DevOps focuses on). Agile relies on the agile manifesto and often is formalized with a framework, such as Scrum.

DevOps Stages

A) Code Commit and Version Control

B) Code Build

C) Units and Integration Tests

D) Code Quality Checks

E) Artifacts Upload

F) Artifacts Download

E) Deployment

G) Notifications Configuration

H) Maintenance

Problem Statement

Build an end to end CI/CD pipeline to build, test and deploy .NET web projects.

Why DevOps?

1. Shorter Development cycles, faster Innovation.
2. Reduced Deployment failures, rollbacks and Time to recover.
3. Improved communication and collaboration.
4. Increased Efficiencies.
5. Reduced costs and IT Headcount.

Azure DevOps

Azure DevOps captures over 15 years of investment and learnings in providing tools to support software development teams. In the last month, over 80,000 internal Microsoft users and thousands of our customers, in teams both small and large, used these services to ship products to you.

Each Azure DevOps service is open and extensible. They work great for any type of application regardless of the framework, platform, or cloud. You can use them together for a full DevOps solution or with other services. If you want to use Azure Pipelines to build and test a Node service from a repo in GitHub and deploy it to a container in AWS, go for it. Azure DevOps supports both public and private cloud configurations. Run them in our cloud or in your own data center. No need to purchase different licenses.

[Azure DevOps](#) includes:

1. Azure Pipelines

CI/CD that works with any language, platform, and cloud. Connect to GitHub or any Git repository and deploy continuously.

Powerful work tracking with Kanban boards, backlogs, team dashboards, and custom reporting.

2. Azure Artifacts

Powerful work tracking with Kanban boards, backlogs, team dashboards, and custom reporting.

Maven, npm, and NuGet package feeds from public and private sources.

3. Azure Repos

Unlimited cloud-hosted private Git repos for your project. Collaborative pull requests, advanced file management, and more.

4. Azure Test Plans

All in one planned and exploratory testing solution.

Tasks Used In Azure Build Pipeline –

1. **Azure Repos** – It is Azure DevOps native repository for SCM.
2. **Hosted Agent** – A job is a logical grouping of tasks that defines the runtime target on which the tasks will execute. An agent job executes tasks on an agent in an agent pool. These jobs are executed by the agent either Hosted or Private.
3. **Nuget Tool Installer** – Acquires a specific version of NuGet from the internet or the tools cache and adds it to the PATH. Use this task to change the version of NuGet used in the NuGet tasks.
4. **NuGet Restore** – Restore, pack, or push NuGet packages, or run a NuGet command. Supports NuGet.org and authenticated feeds like Package Management and MyGet. Uses NuGet.exe and works with .NET Framework apps. For .NET Core and .NET Standard apps, use the .NET Core task.
5. **Prepare Analysis on SonarCloud** – Prepare SonarCloud analysis configuration
6. **Build Solution** - Build with MSBuild and set the Visual Studio version property.
7. **NuGet pack** – Restore, pack, or push NuGet packages, or run a NuGet command. Supports NuGet.org and authenticated feeds like Package Management and MyGet. Uses NuGet.exe and works with .NET Framework apps. For .NET Core and .NET Standard apps, use the .NET Core task.
8. **NuGet push** – Restore, pack, or push NuGet packages, or run a NuGet command. Supports NuGet.org and authenticated feeds like Package Management and MyGet. Uses NuGet.exe and works with .NET Framework apps. For .NET Core and .NET Standard apps, use the .NET Core task.
9. **Test Assemblies** – Run unit and functional tests (Selenium, Appium, Coded UI test, etc.) using the Visual Studio Test (VsTest) runner. Test frameworks that have a Visual Studio test adapter such as MsTest, xUnit, NUnit, Chutzpah (for JavaScript tests using QUnit, Mocha and Jasmine), etc. can be run. Tests can be distributed on multiple agents using this task
10. **Run code analysis** – Run scanner and upload the results to the SonarCloud server.
11. **Publish quality gate result** – Publish SonarCloud's Quality Gate result on the Azure Pipelines build result. To be used after the actual analysis.
12. **Publish symbols path** – Index your source code and publish symbols to a file share or Azure Artifacts Symbol Server
13. **Publish Artifact** - Publish build artifacts to Azure Pipelines/TFS or a file share.

Tasks Used in Azure Release Pipeline –

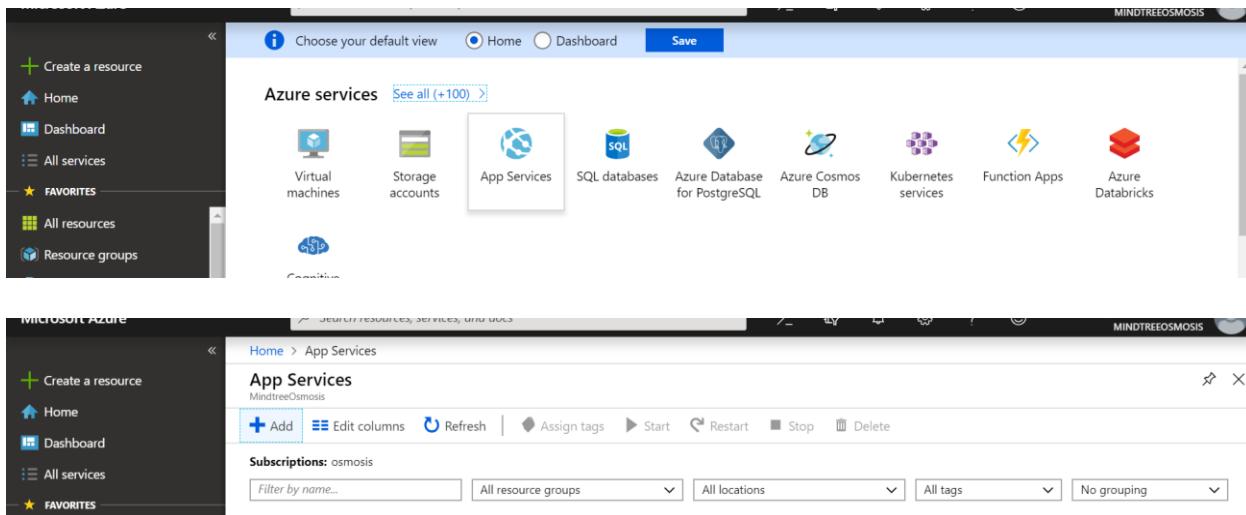
14. **Deploy Azure App Service** - Update Azure App Services on Windows, Web App on Linux with built-in images or Docker containers, ASP.NET, .NET Core, PHP, Python or Node.js based Web applications, Function Apps on Windows or Linux with Docker Containers, Mobile Apps, API applications, Web Jobs using Web Deploy / Kudu REST APIs.

Creating WebApps for Application Deployment using Azure Portal.

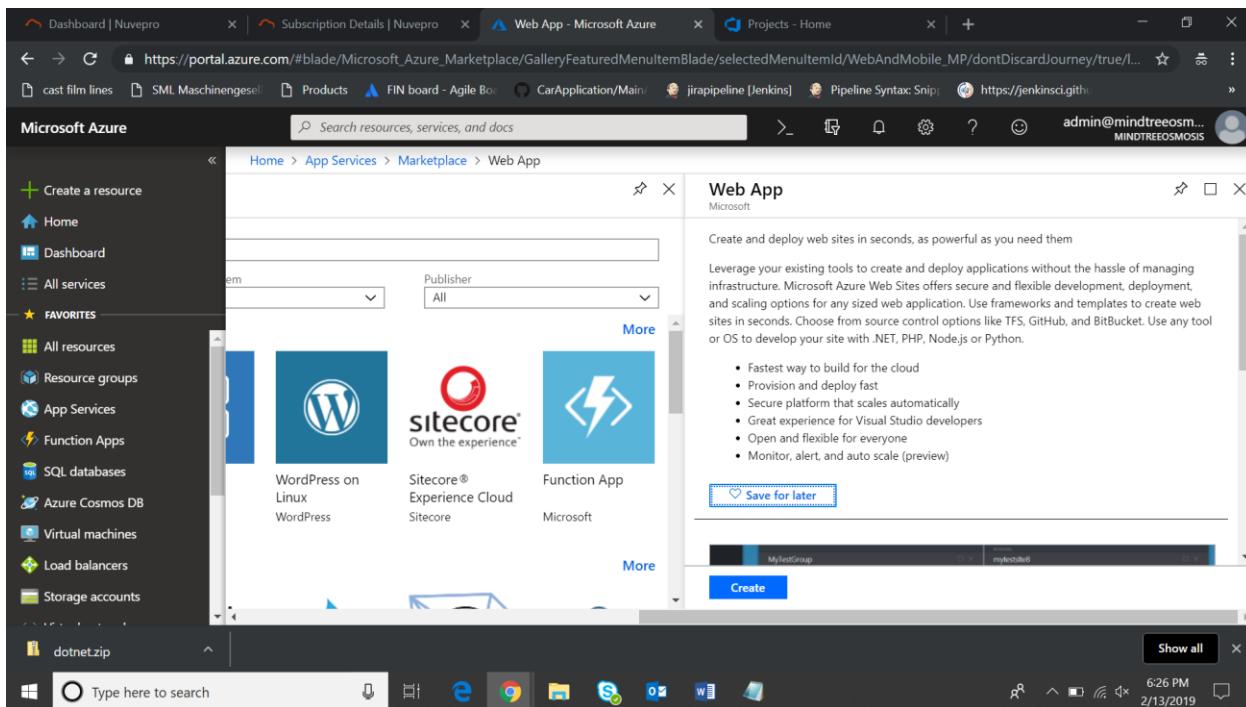
1. Login into Azure Portal.

<https://azure.microsoft.com/en-in/features/azure-portal/>

2. Click on “App services” and click “Add”.

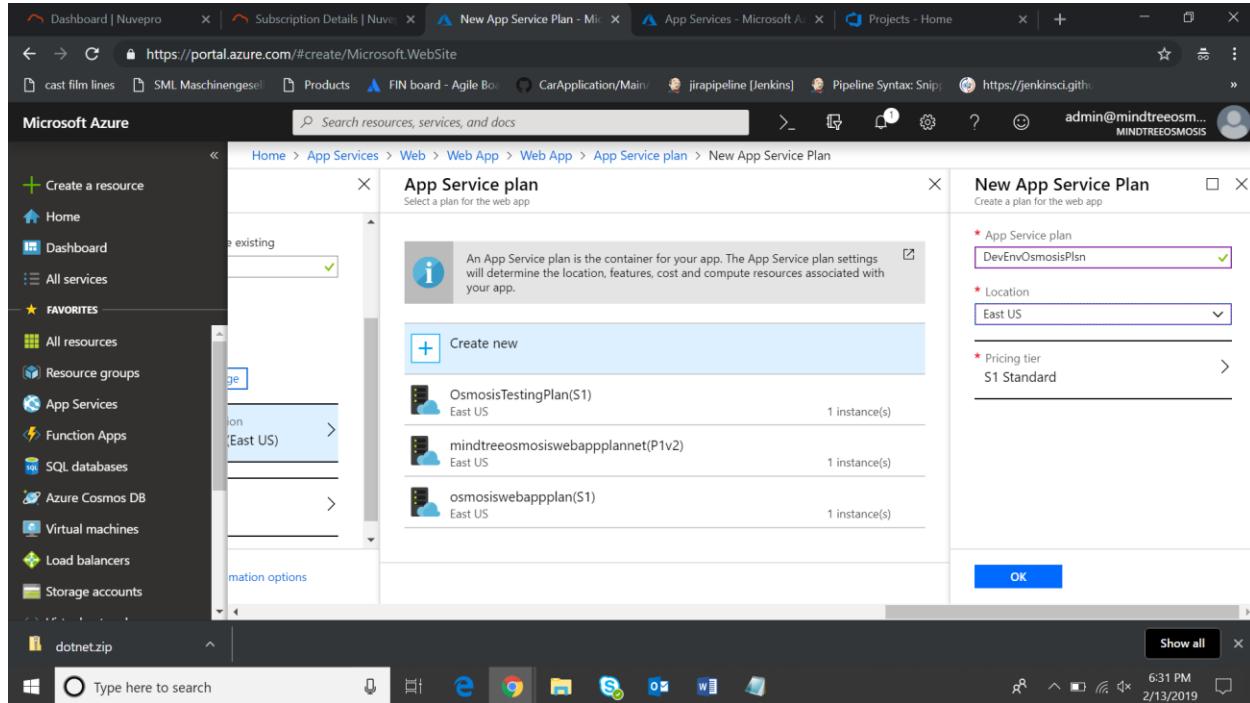


3. Select “WebApp” and click on “Create”.

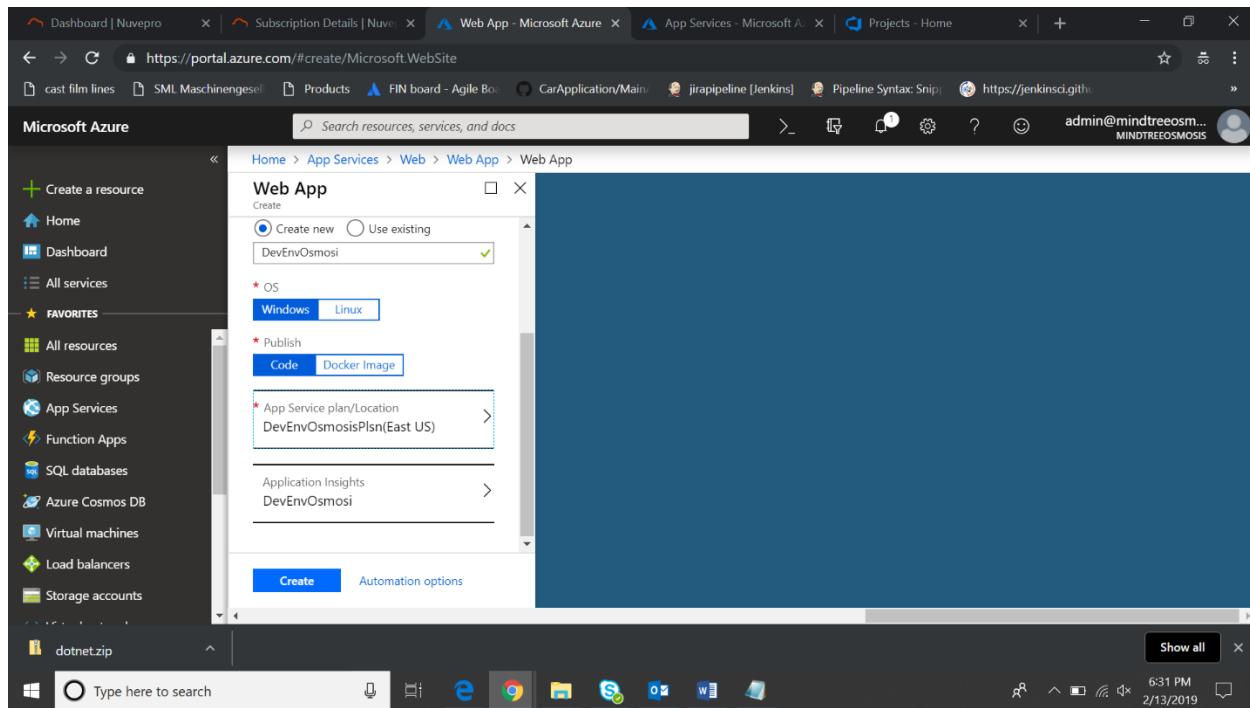


4. Enter App Name, Subscription is already chosen by default, Resource Group is also created and has the name same as the App Name.
5. OS should be set to “Windows” and Publish to “Code”.

6. Click on **App Service Plan**.
7. Click on “**Create New**” → Add a name → Location to “**East US**” → Pricing Tier to “**Standard S1**”.

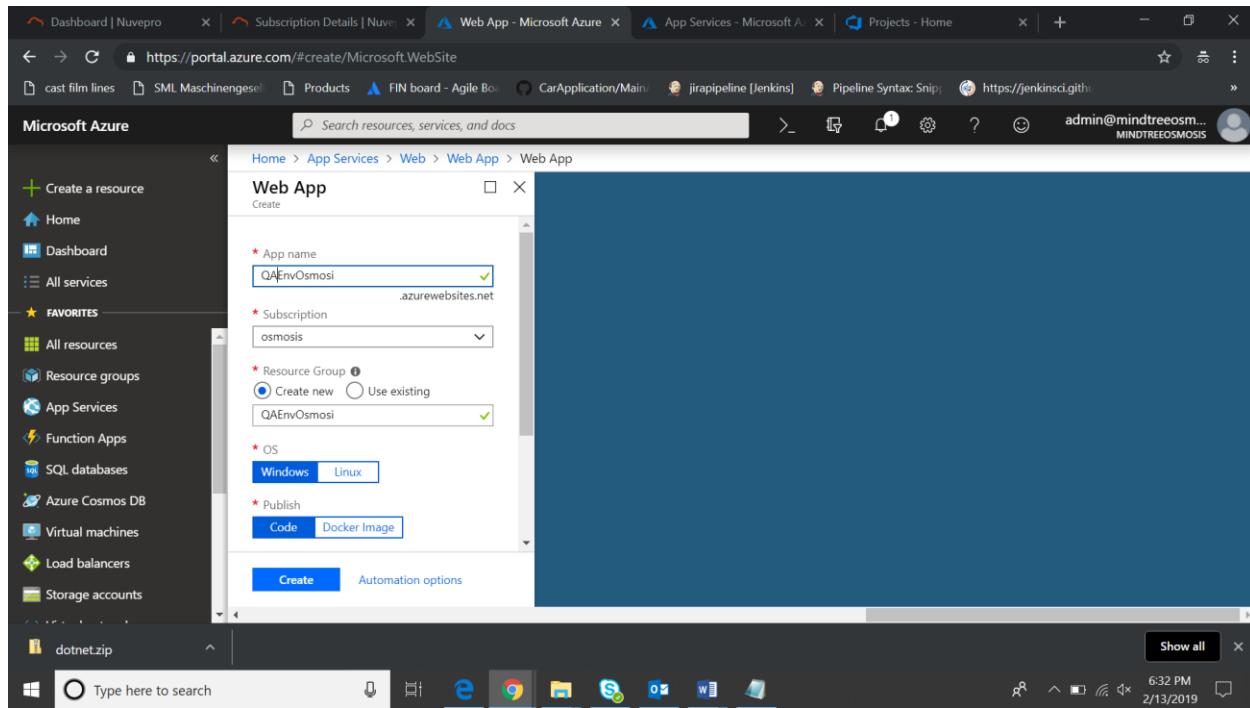


8. Click on “**Create**”.



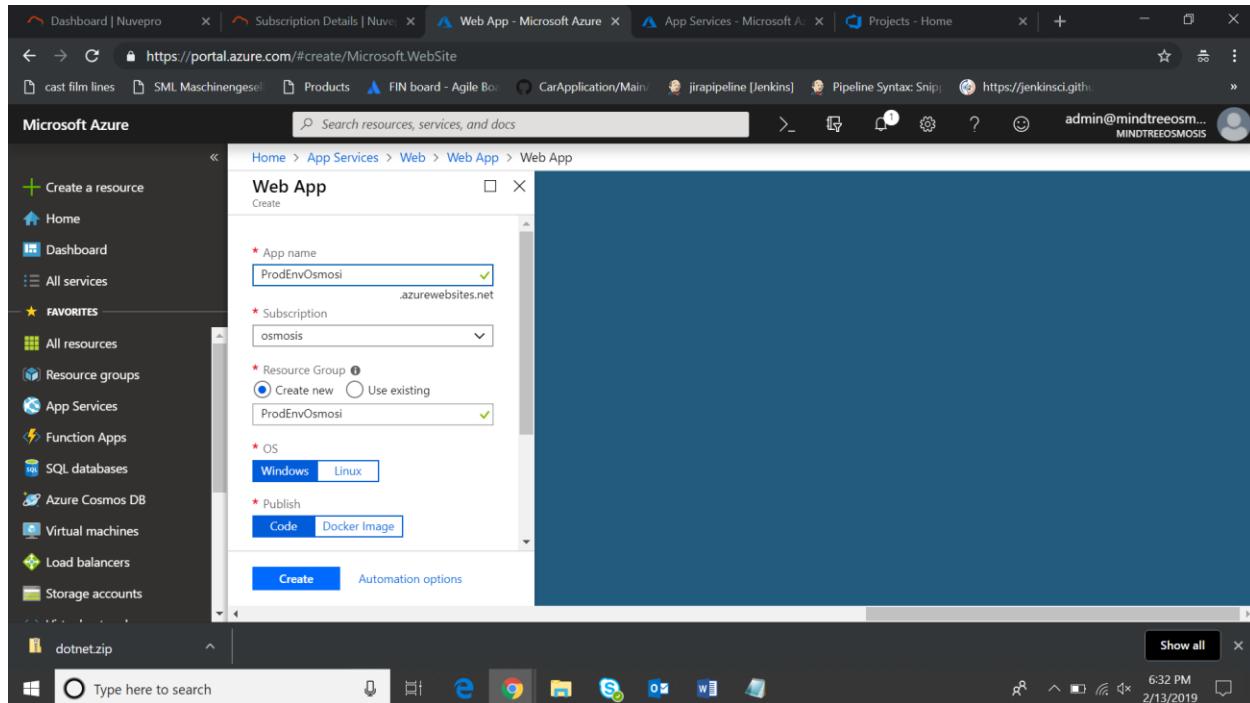
9. Create one for QA and Production Environments also following the above steps.

QA Environment Web APP



The screenshot shows the Microsoft Azure portal interface. The left sidebar is open, showing 'App Services' as the selected service. The main content area is titled 'Web App' and is in 'Create' mode. The 'App name' field is filled with 'QAEnvOsmosi'. Other fields include 'Subscription' (osmosis), 'Resource Group' (Create new, 'QAEnvOsmosi'), 'OS' (Windows), and 'Publish' (Code, Docker Image). The 'Create' button is visible at the bottom. The status bar at the bottom right shows the date and time: 6:32 PM 2/13/2019.

Production Environment Web App



The screenshot shows the Microsoft Azure portal interface, similar to the previous one but with a different app name. The left sidebar is open, showing 'App Services' as the selected service. The main content area is titled 'Web App' and is in 'Create' mode. The 'App name' field is filled with 'ProdEnvOsmosi'. Other fields include 'Subscription' (osmosis), 'Resource Group' (Create new, 'ProdEnvOsmosi'), 'OS' (Windows), and 'Publish' (Code, Docker Image). The 'Create' button is visible at the bottom. The status bar at the bottom right shows the date and time: 6:32 PM 2/13/2019.

10. On the side bar Click on **App Services**. If the deployments are over then you should be able to see the webapps you just created.

Building CI/CD Pipeline with Azure DevOps.

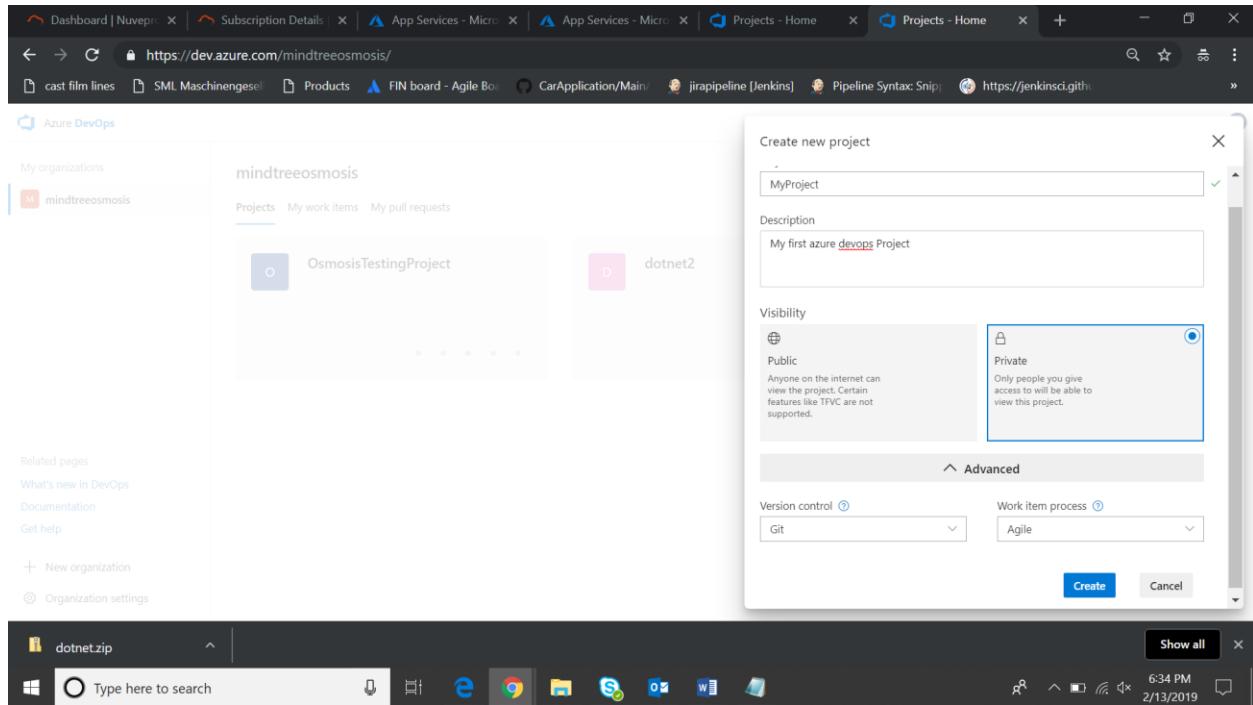
11. Login into Azure DevOps.

<https://azure.microsoft.com/en-in/services/devops/>

Note: Click on “Start Free” if you are a first time user.

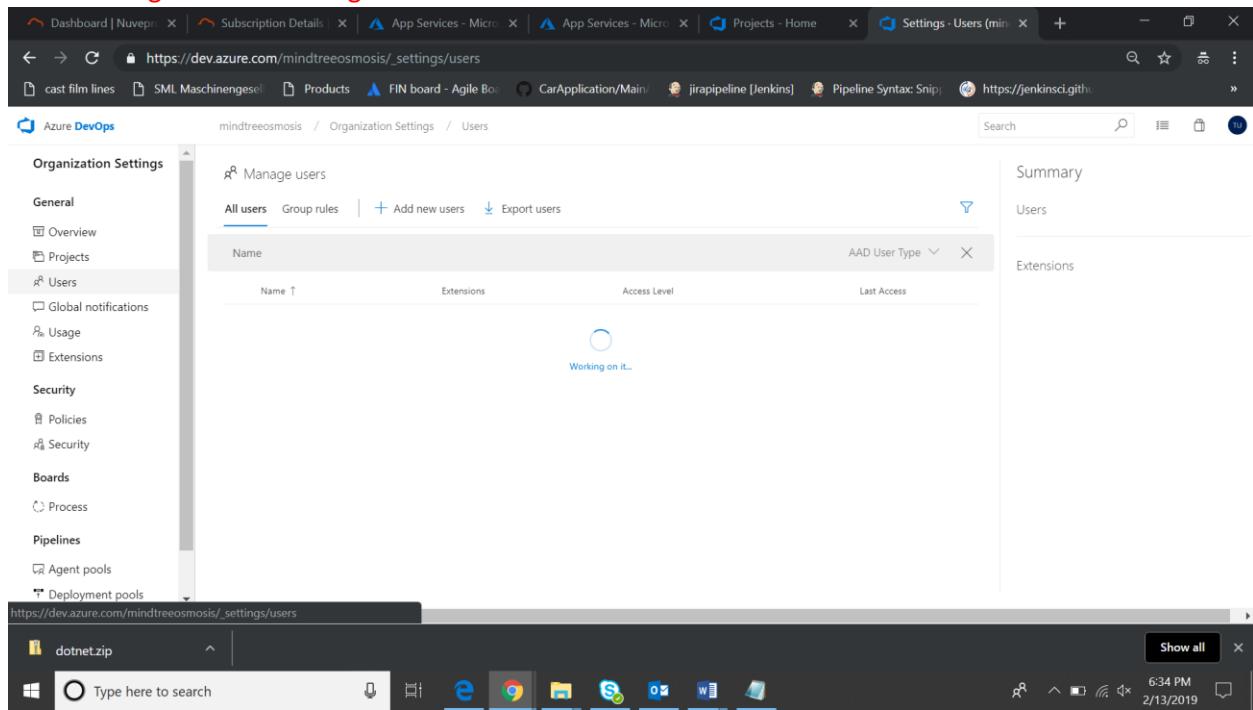
12. Click on “Create Project”.

13. Enter the Project Name → Description (Optional) → Visibility to “Private” → Click “Advanced” → Ensure Version Control is “Git” and the other option is “Agile”



14. The project is created and now we have to add user who can use and make changes to this project.

15. Click on “Organization Settings” → Click on “Users” → Click on “Add new User”.



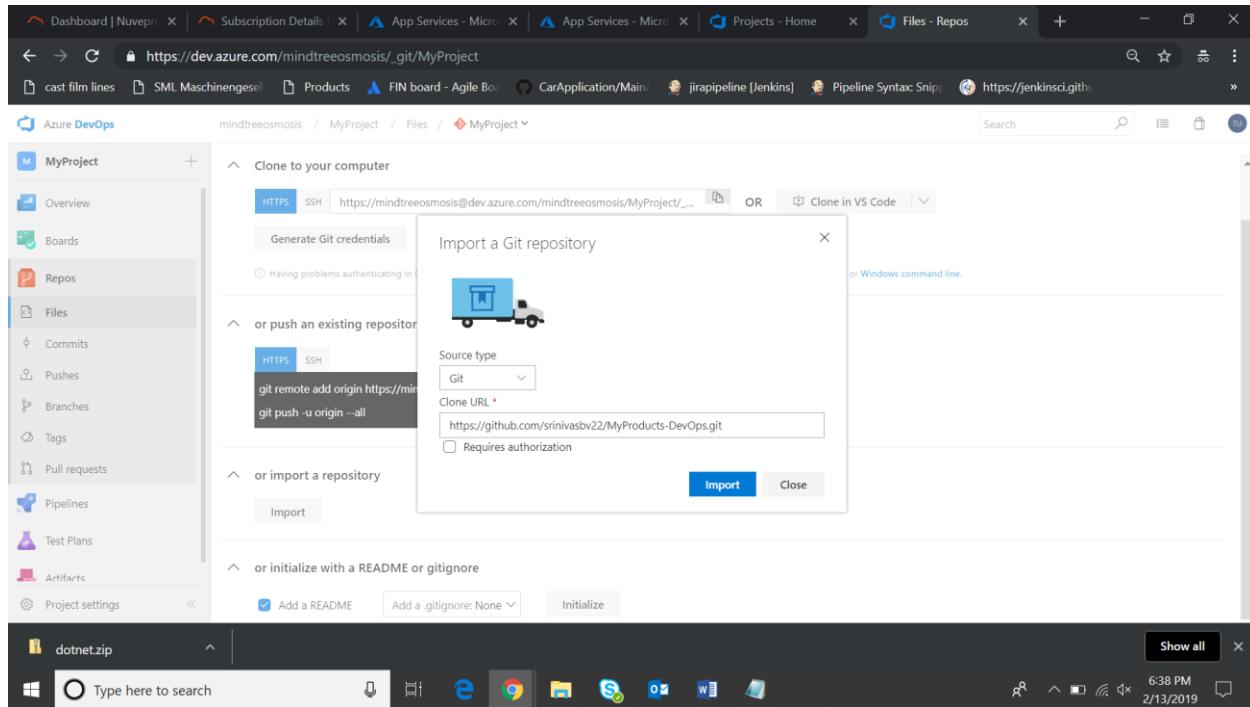
16. Enter the user name (“email id” or “mid”) → Set access level to “**Basic**” → Depending on your needs you can assign projects (When you click on the drop down in projects all the projects available in that account will be displayed) → Set Azure DevOps Groups to “**Project Contributors**” → Check “**Send Email Invites**” → Click **Add**.

After Adding the Screen looks like.

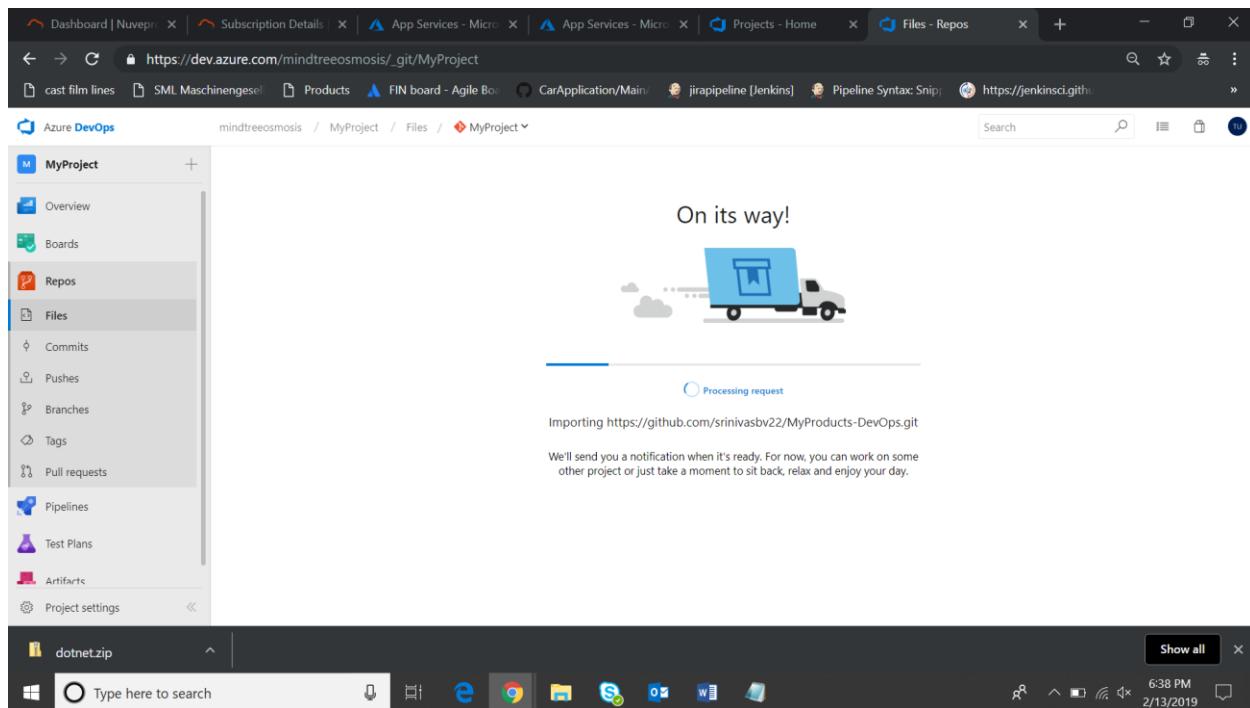
17. Click on **Azure DevOps** and Select your Project.

Azure Repos.

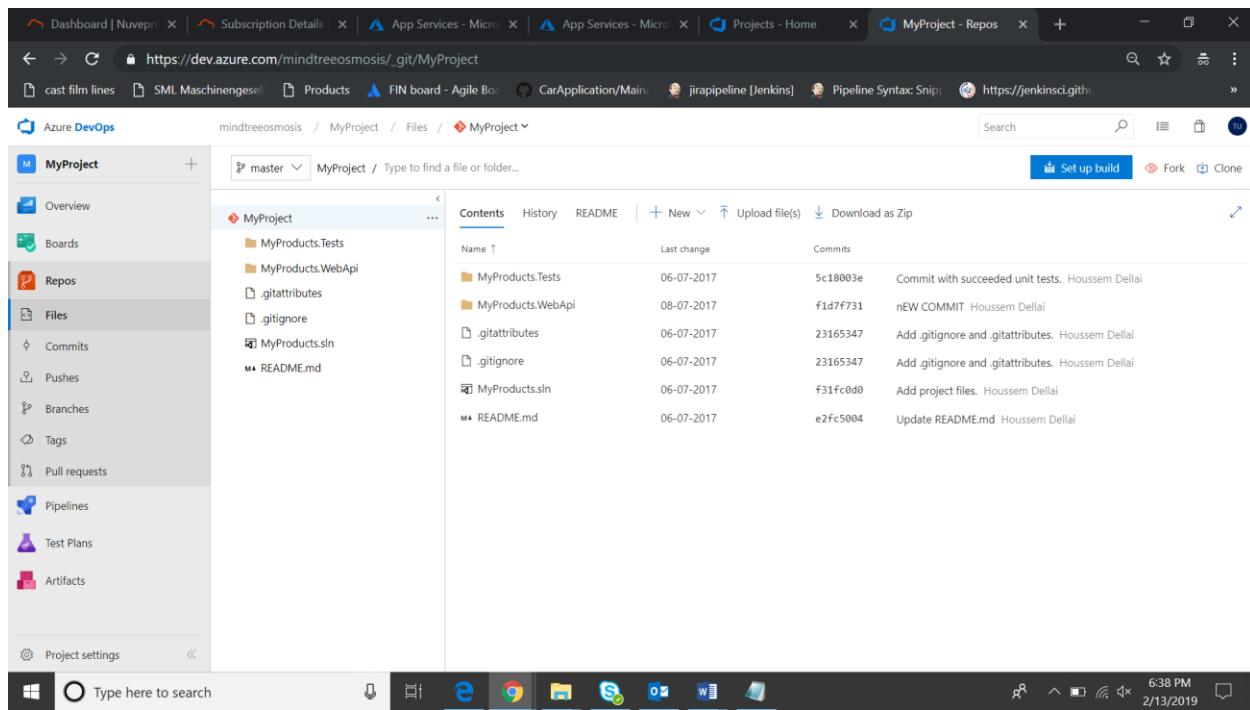
18. In the left bullet bar click on “**Repos**”.
19. Click on “**import**”.
20. Set source type to “**Git**” and add the Project Git Repository URL.
21. Click on “**Import**”.



It takes a while to load the code as shown below.



22. By clicking on “**File**” under “**Repos**” you will be able to see you imported project.

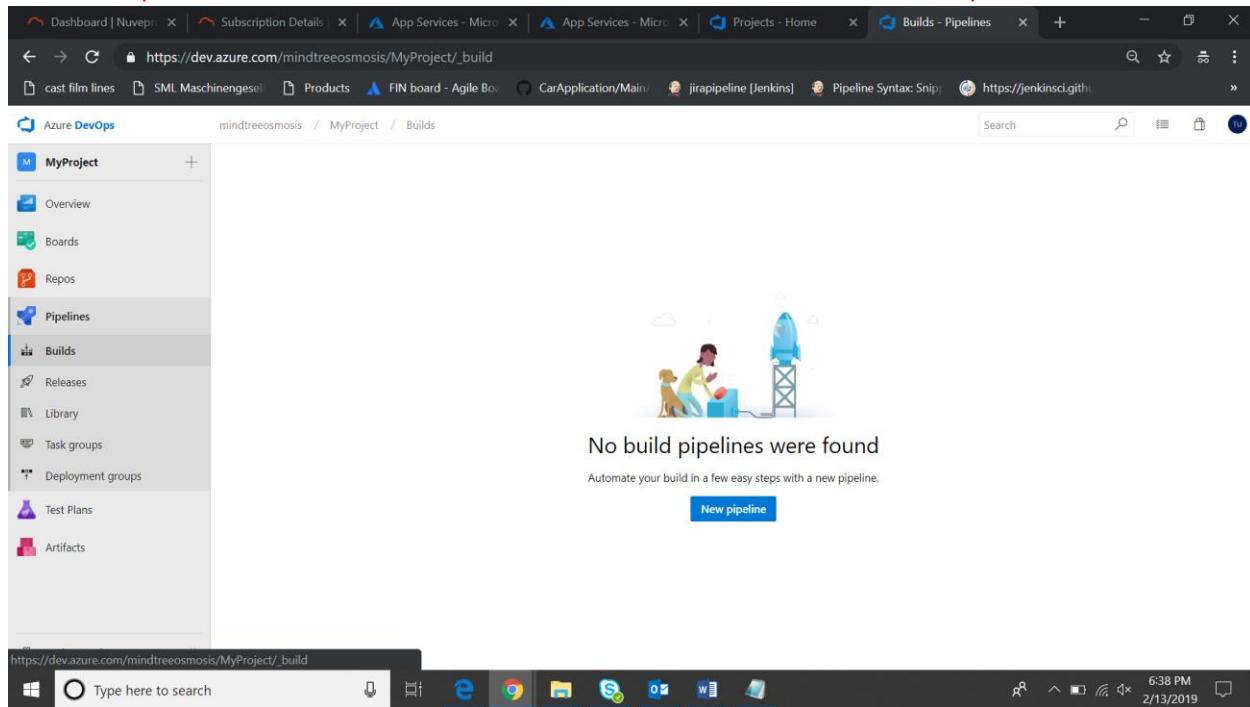


Azure DevOps interface showing the 'Files' section for 'MyProject'. The 'master' branch is selected. The list of files includes:

Name	Last change	Commits
MyProducts.Tests	06-07-2017	5c18003e Commit with succeeded unit tests. Houssem Dellai
MyProducts.WebApi	08-07-2017	f1d7f731 nEW COMMIT Houssem Dellai
.gitattributes	06-07-2017	23165347 Add .gitignore and .gitattributes. Houssem Dellai
.gitignore	06-07-2017	23165347 Add .gitignore and .gitattributes. Houssem Dellai
MyProducts.sln	06-07-2017	f31fc0d0 Add project files. Houssem Dellai
README.md	06-07-2017	e2Fc5004 Update README.md Houssem Dellai

Build Pipeline

23. Click on “Pipelines” on the left Bullet bar → Click On “Builds” → Click on “New Pipeline”.



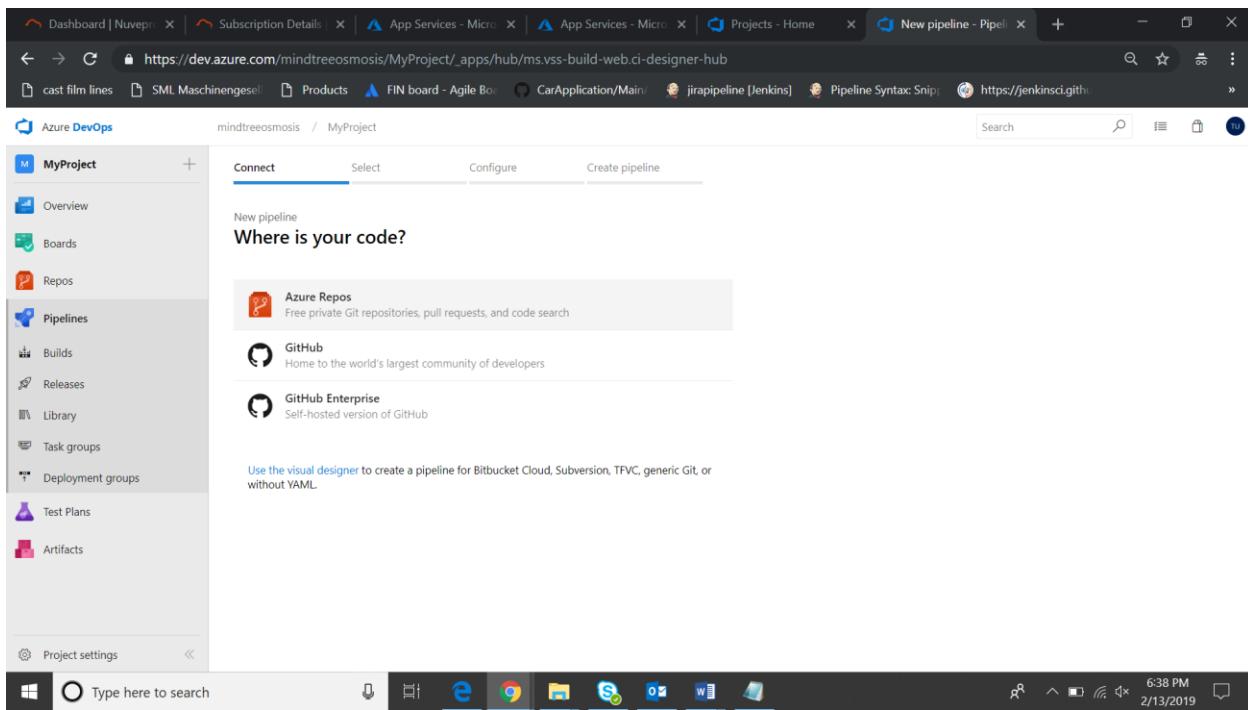
Azure DevOps interface showing the 'Builds' section for 'MyProject'. The 'Builds' section is selected in the left sidebar. The central area displays:

No build pipelines were found

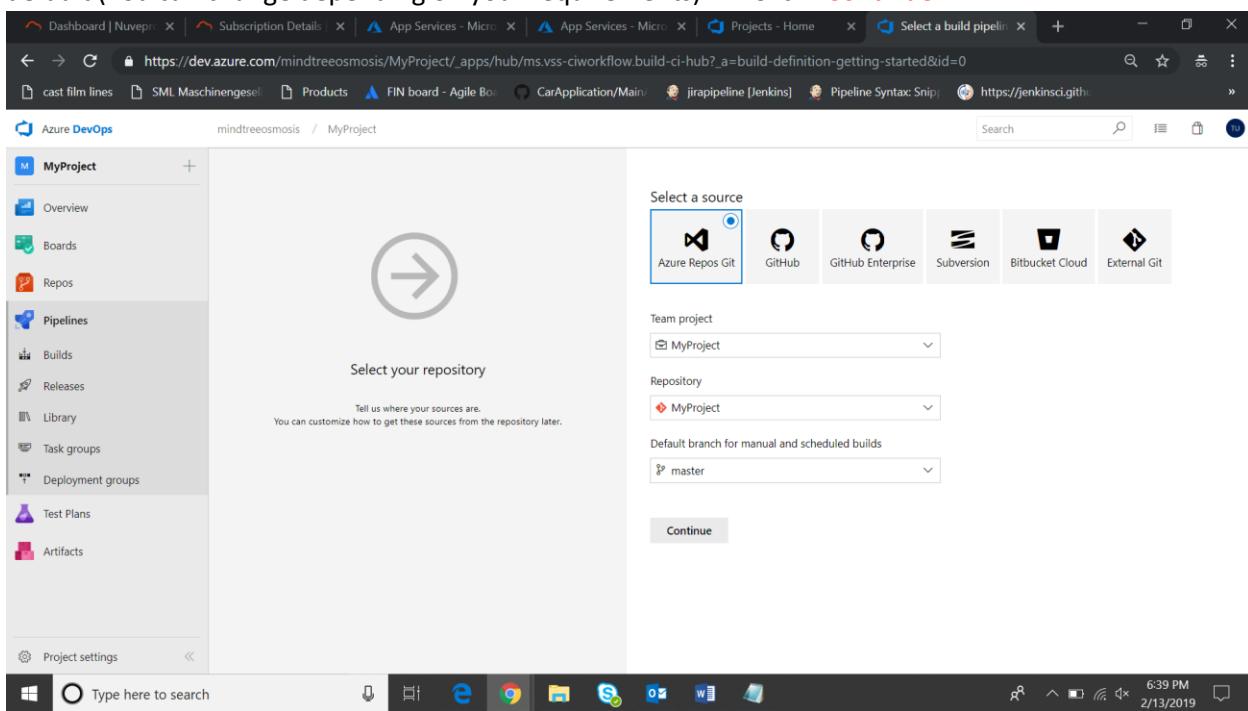
Automate your build in a few easy steps with a new pipeline.

New pipeline

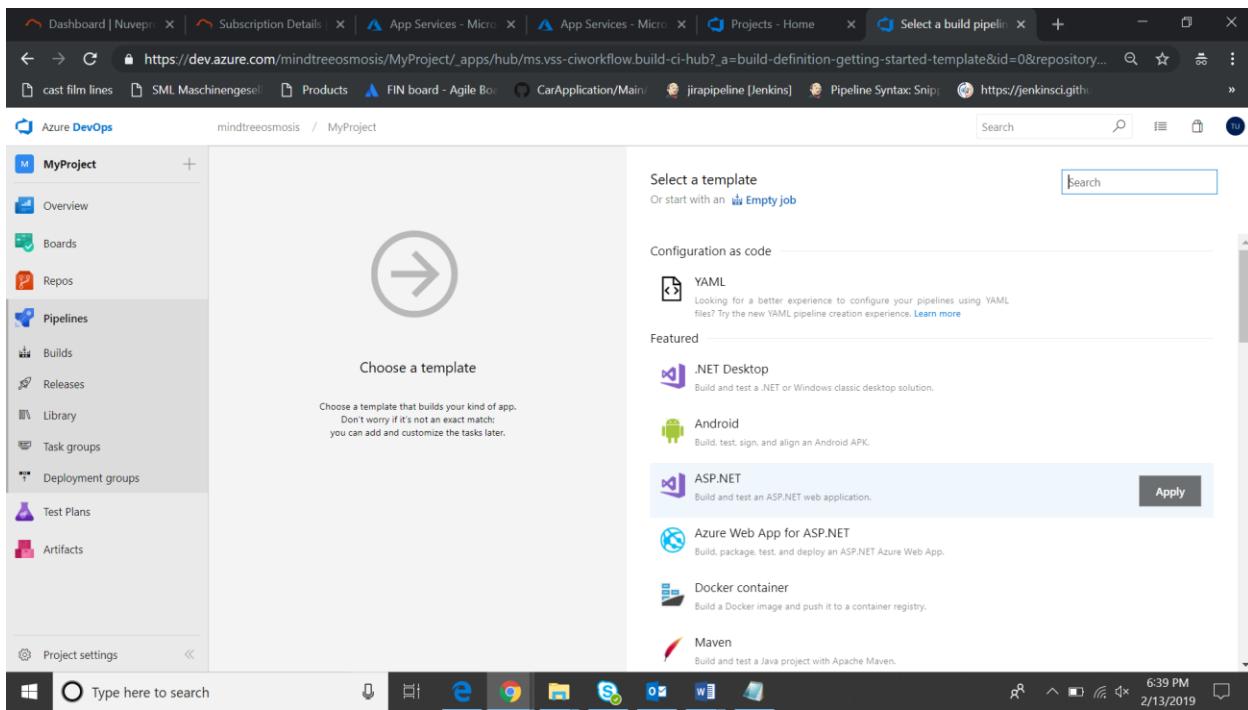
24. Click on “Use the Visual Designer”.



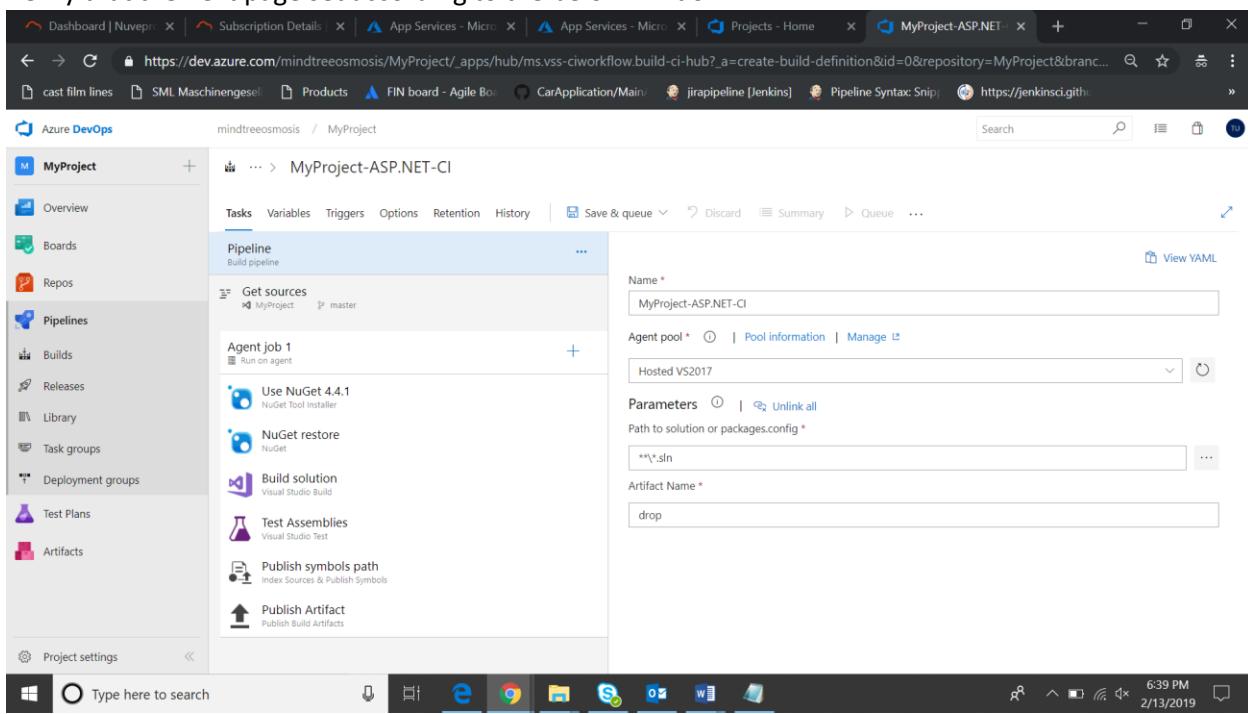
25. Click on “**Azure Repos Git**” → Select your “**Team Project**”, “**Repository**”. The branch will be set to “**Master**” by default (You can change depending on your requirements). → Click “**Continue**”



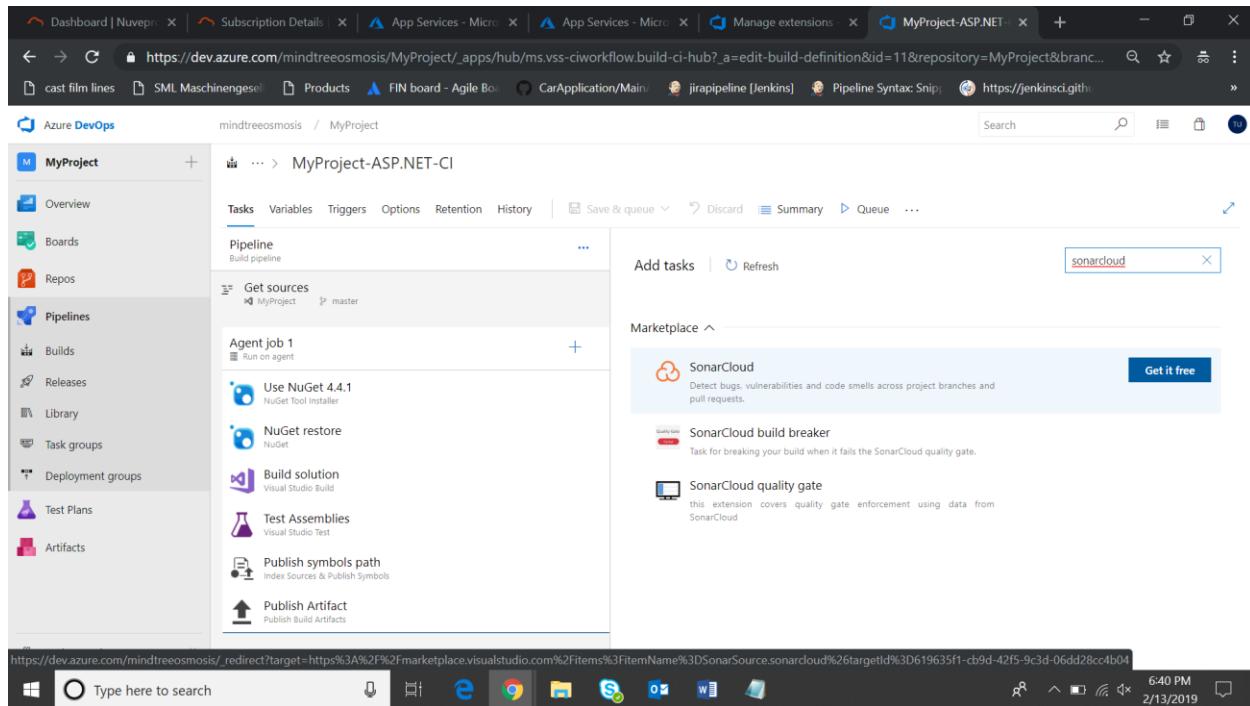
26. Select “**ASP .NET**”.



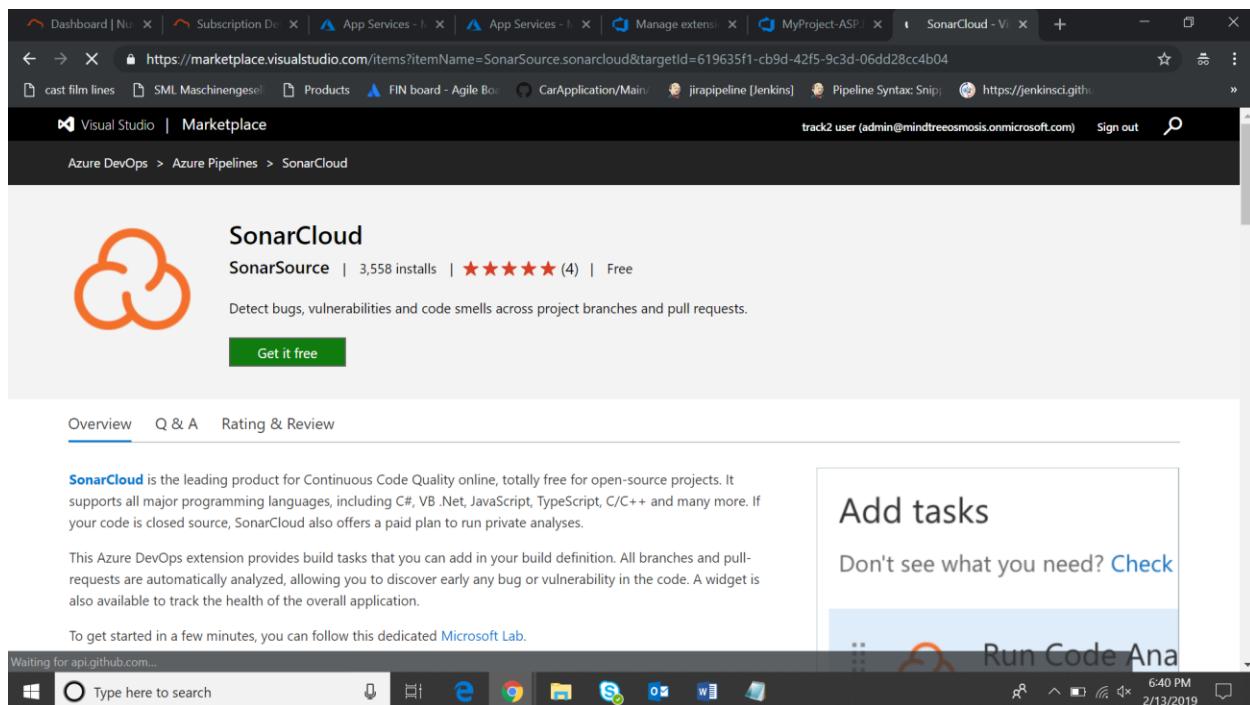
27. Verify that the next page set according to the below window.



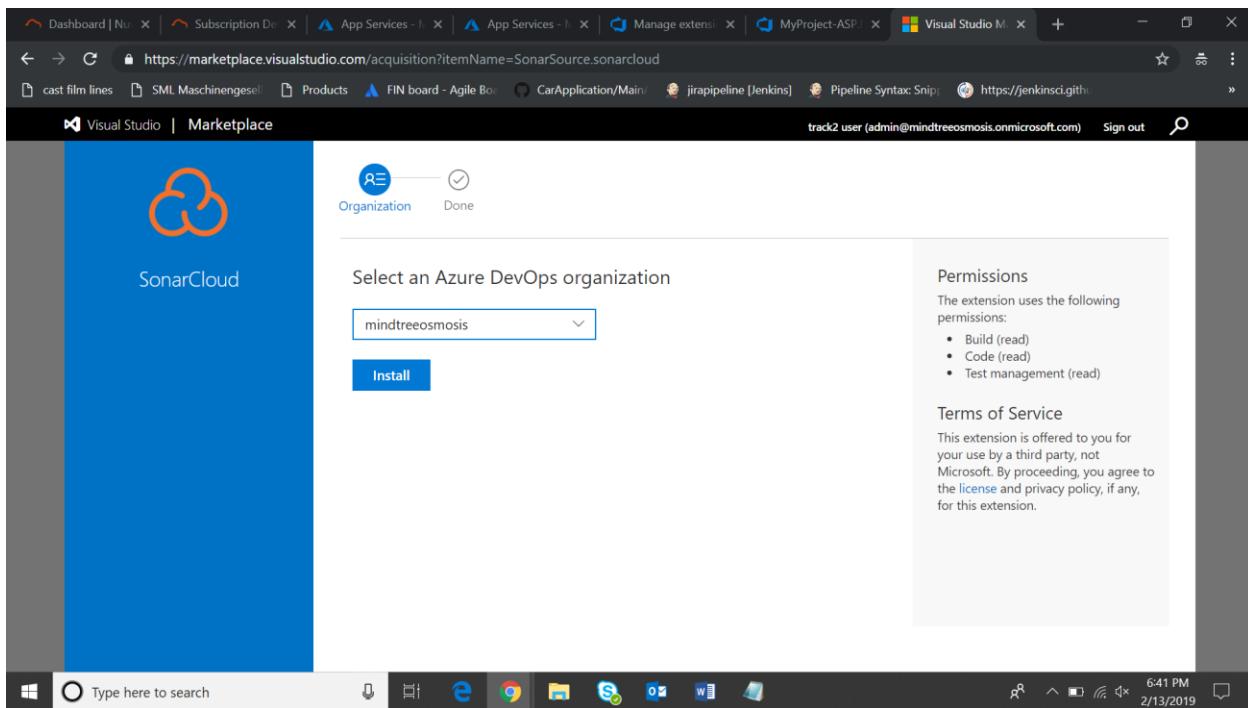
28. Click on the “+” in Agent Job 1 (It is used for adding tasks to the Pipeline) → Search for “SonarCloud”.



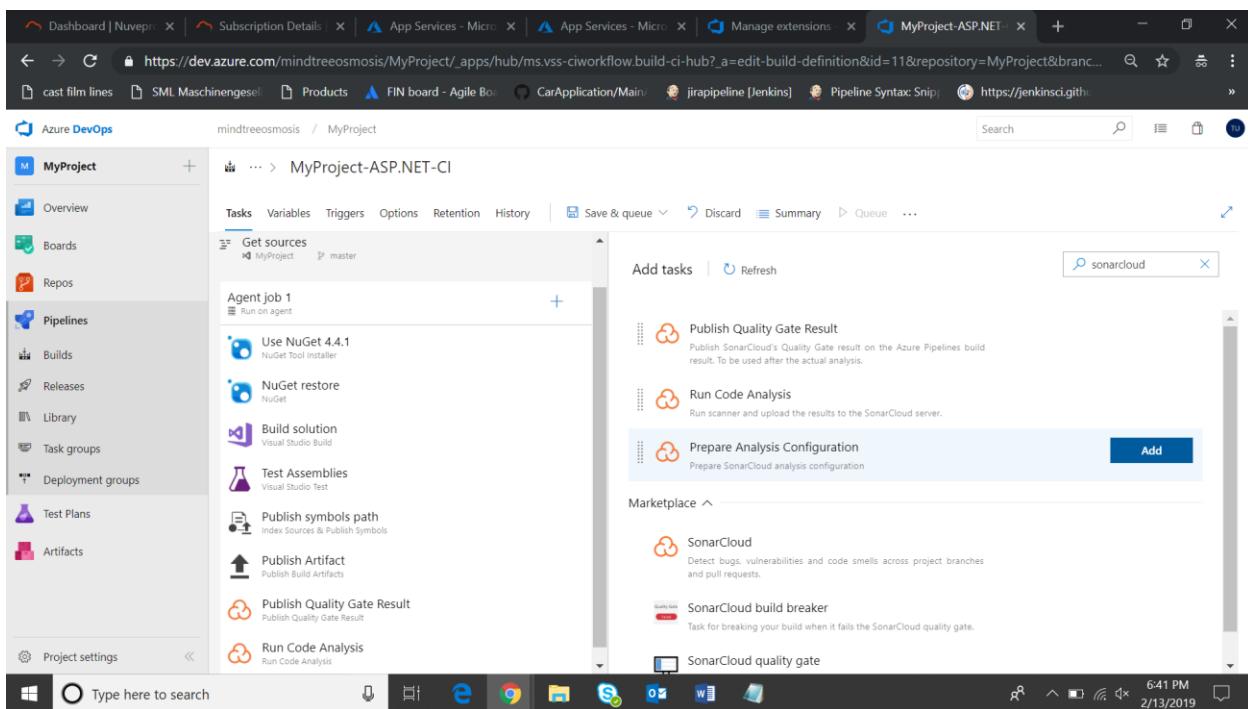
29. Click on “Get it Free” → Click “Get it Free” on the Redirected Page.



30. Click “Install”.



31. Click “Proceed to Organization”.
32. Now add the SonarCloud Tasks → Add “Prepare Analysis Configuration” → Add “Run Code Analysis” → Add “Publish Quality Gate Result”.



33. The arrangement of tasks should be in the order as shown in the window.

The screenshot shows the Azure DevOps Pipeline interface for a project named 'MyProject'. The left sidebar is expanded to show the 'Pipelines' section. A pipeline named 'MyProject-ASP.NET-CI' is selected. The pipeline tasks are listed on the right, with 'Prepare analysis on SonarCloud' highlighted. The configuration pane for this task is open, showing the following settings:

- Display name: Prepare analysis on SonarCloud
- SonarCloud Service Endpoint: (dropdown menu, required)
- Organization: (dropdown menu, required)
- Choose the way to run the analysis:
 - Integrate with MSBuild (selected)
 - Integrate with Maven or Gradle
 - Use standalone scanner
- Project Key: (dropdown menu, required)

34. On the Browser go to “Sonarcloud.io”.

<https://sonarcloud.io/about>

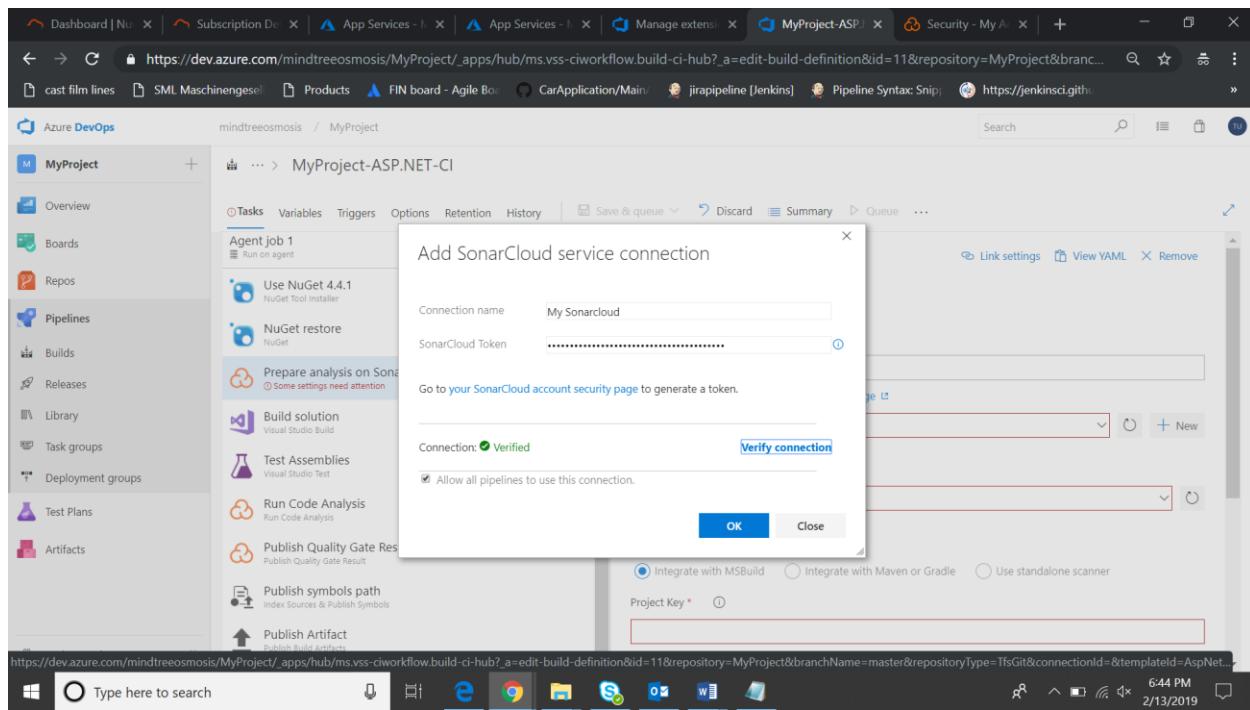
35. Login with “GitHub”.

The screenshot shows the SonarCloud website homepage. The top navigation bar includes links for Dashboard, Subscription Details, App Services, Manage extensions, MyProject-ASP.NET, and SonarCloud. The main content area features a large orange banner with the text 'Clean Code Rockstar Status' and three cartoon characters (a woman, a man with a beard, and a man with a mustache) holding up a foam finger. Below the banner, there are buttons for 'GitHub', 'Bitbucket', and 'Azure DevOps', and a note that says 'Free for Open-Source Projects'.

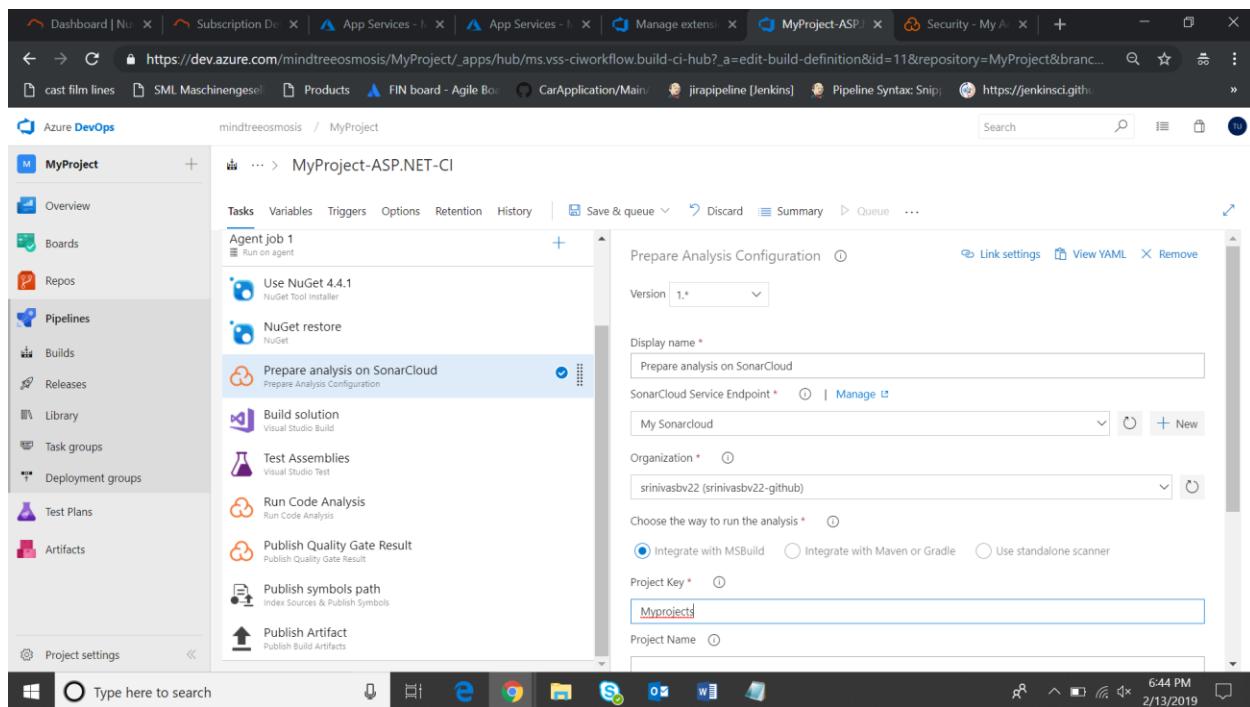
36. Click on “Account” → Click on “My Account”[For account click on right top Photo].

37. Click on “Security” → Give a name for the token → Click “Generate” → Copy the “Access Key”.

38. Get back to your Azure DevOps Project and Click “Prepare Analysis on SonarCloud” → On the right Bullet tab Click on “+ New” → Give the connection a name → Paste the token generated in SonarCloud.io → Click Verify Connection → Click “Ok”.



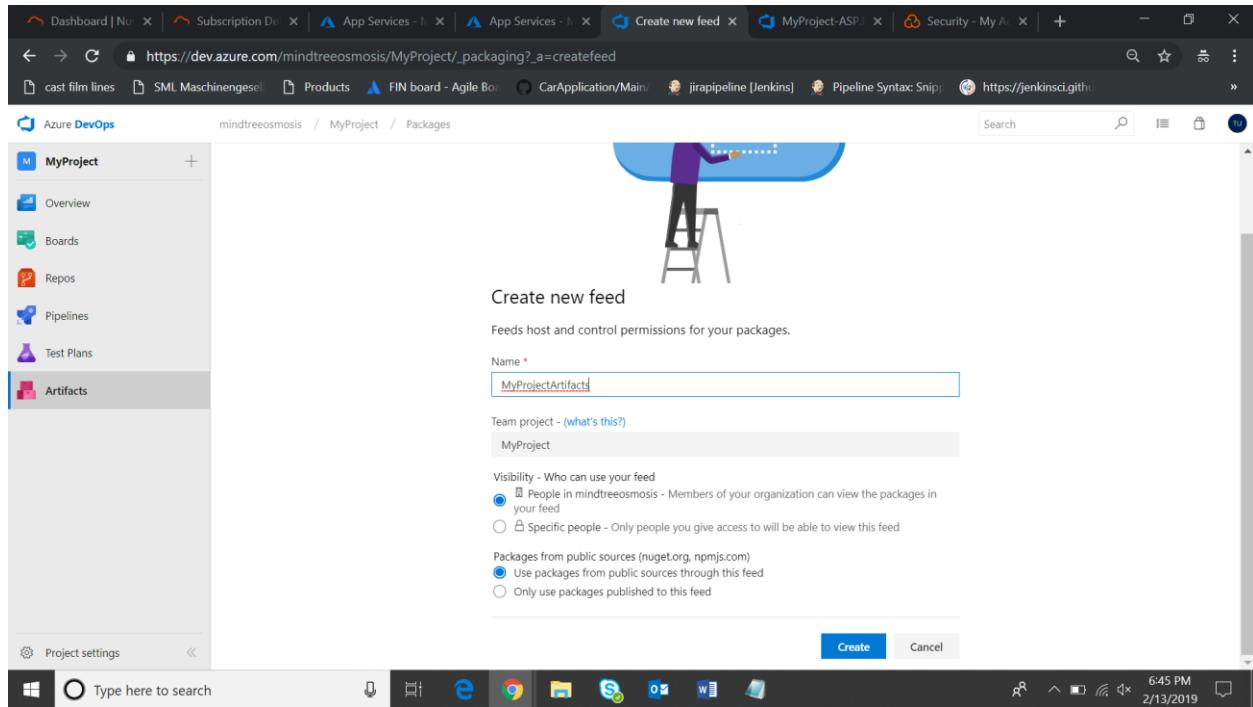
39. Select your “Organization” from the drop down and give the “Project key” a Unique Name.



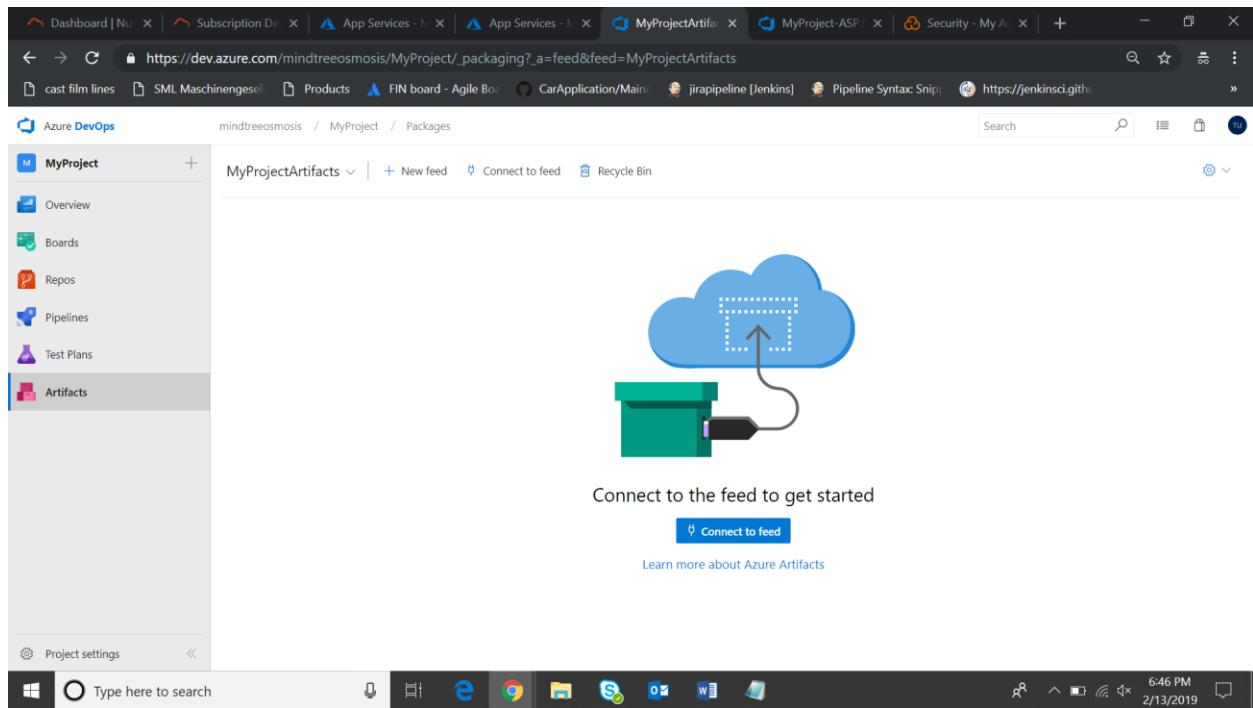
40. Click on “Save” at the top.

Azure Artifacts

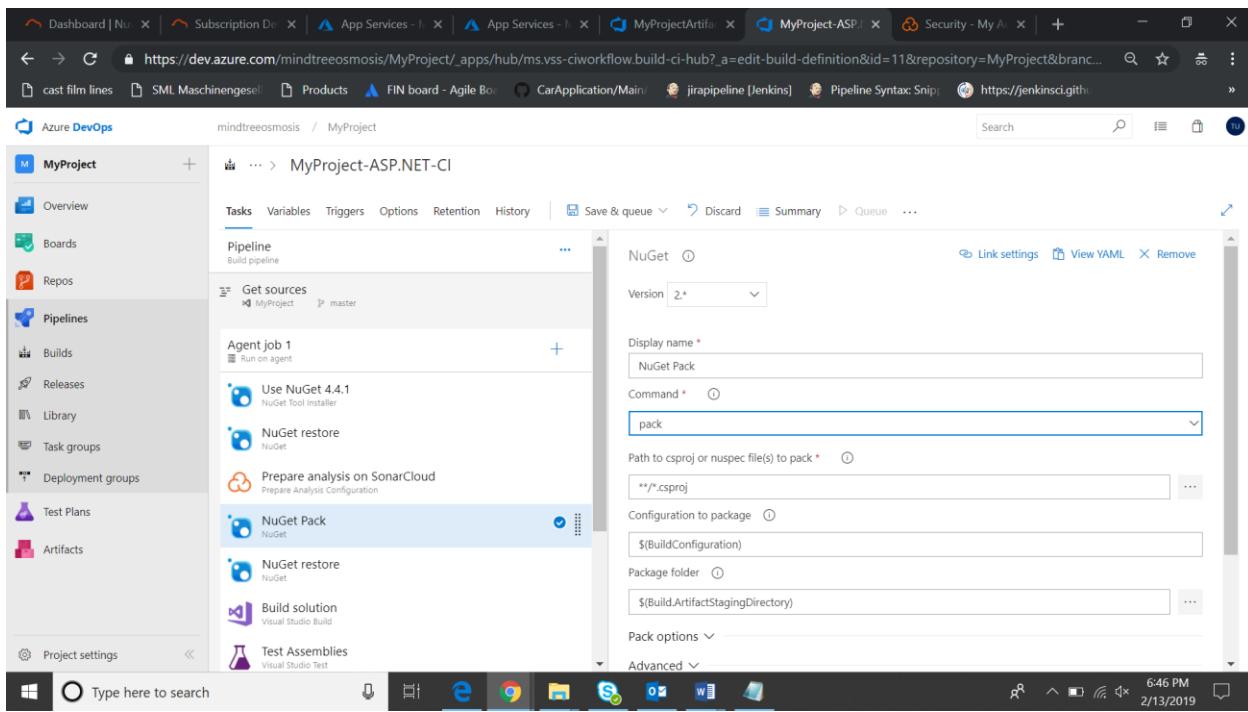
41. Click on “Artifacts” in the right bullet tab → Click on “+ New Feed” → Give the feed a name → Click “Create”.



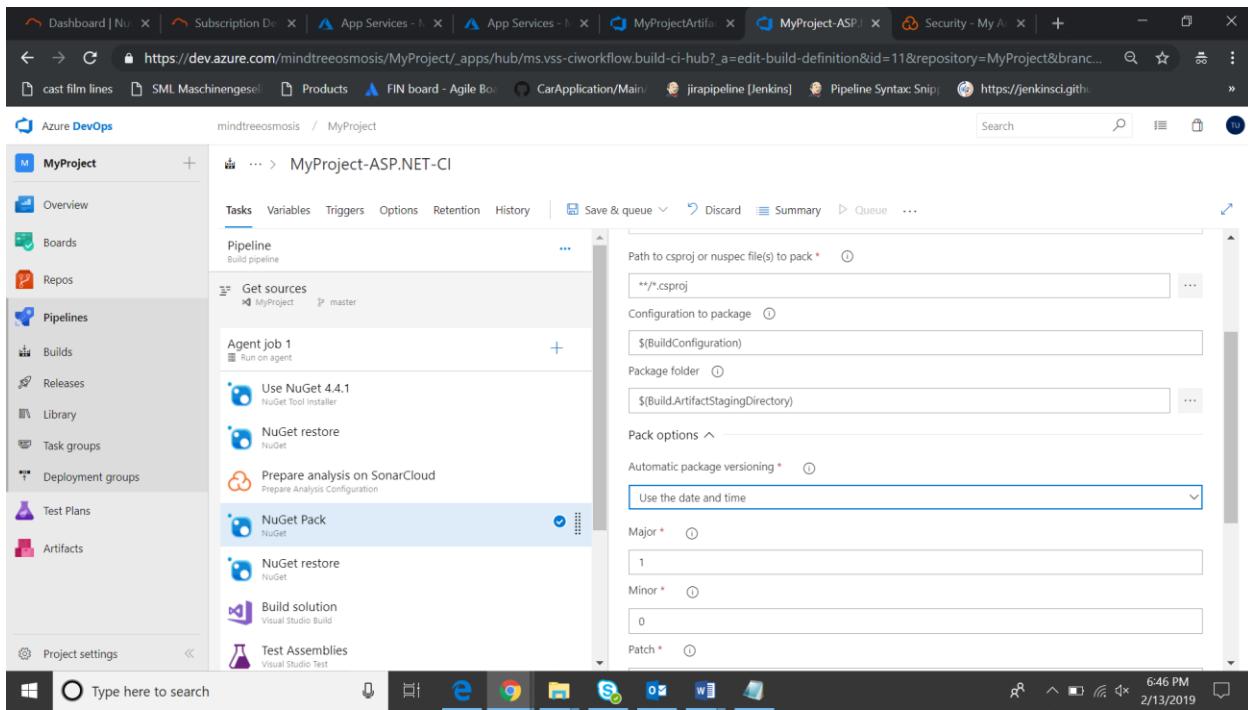
The below screen would be displayed after Create.



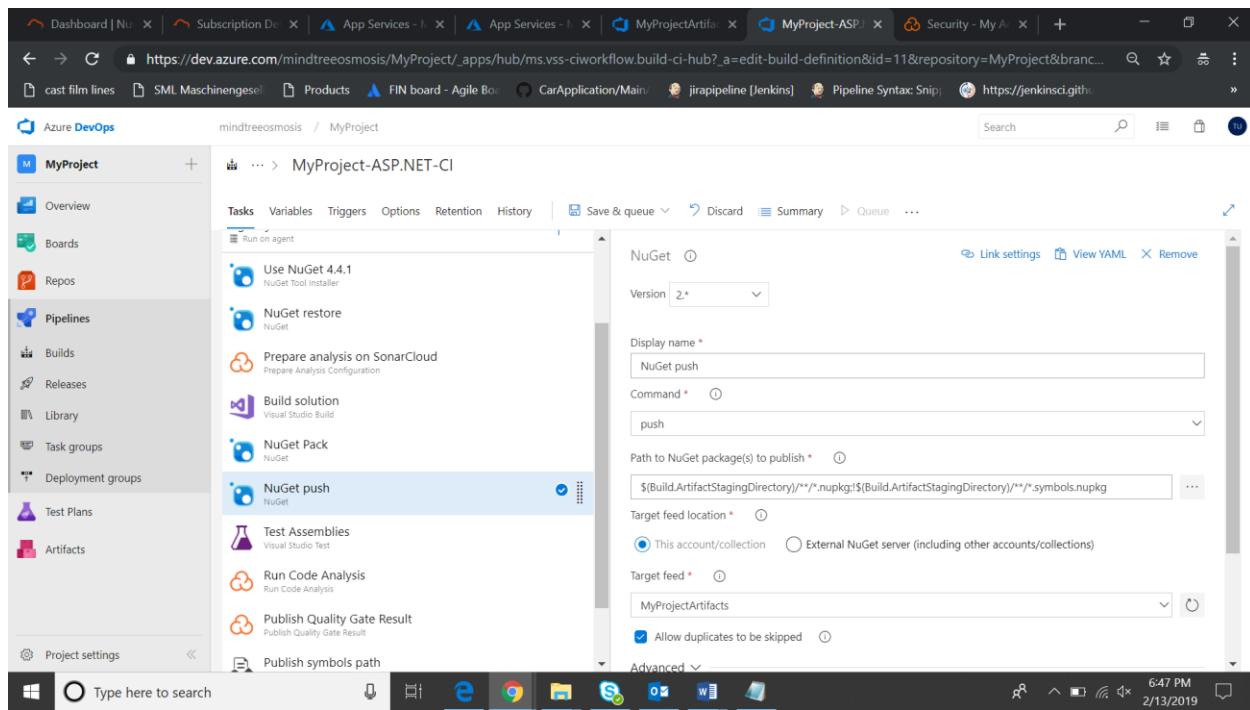
42. Click on “Pipelines” → “Builds” → “Edit” → Click on “+” to add tasks for artifactory Package and Push → Search for “Nuget” → Apply 2 times.



43. Rename the first Nuget Task to “**Nuget Pack**” → Select “**Pack**” for Command → Click on “**Pack Options**” → Select “**Use the date and time**” for Automatic Packaging Version.

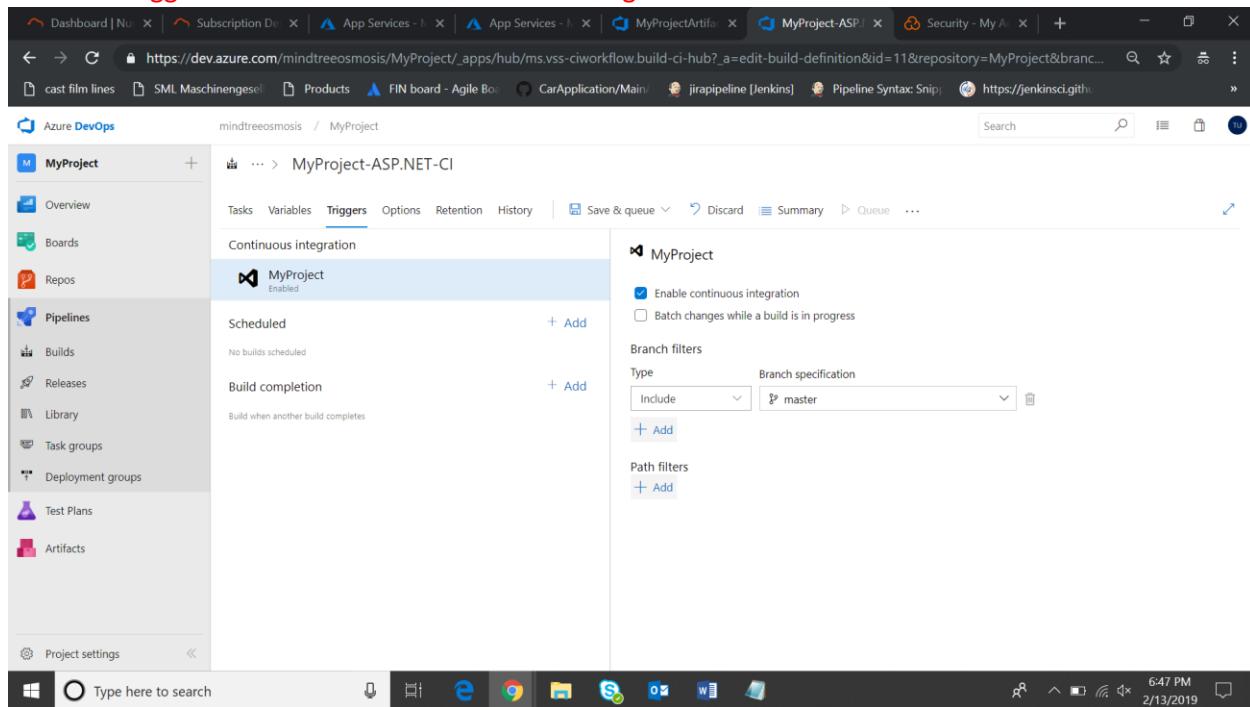


44. Click on the second Nuget and rename it to “**Nuget Push**” → Set the Command to “**Push**” → Select the Target Feed from the dropdown (The name should appear that you have given while creating the Feeds in Artifacts) → Check “**Allow Duplicates to be skipped**”.

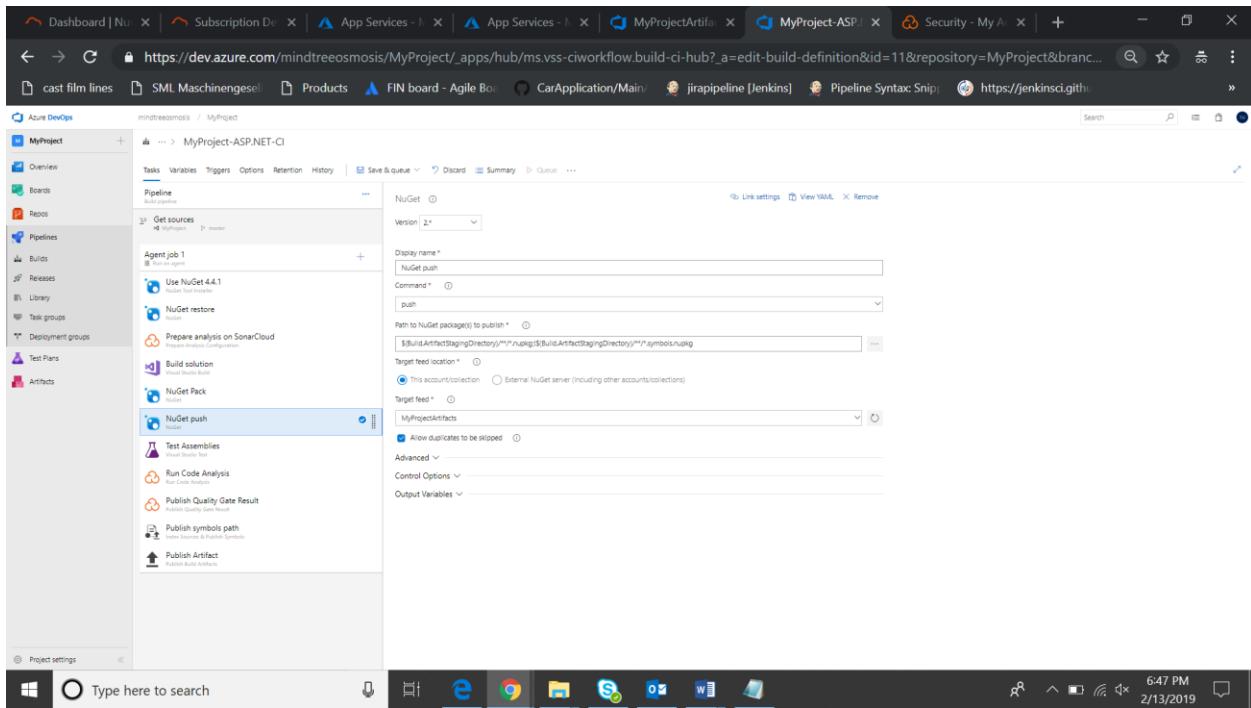


Note – Please make sure that the order of the tasks as in the order shown in the above window.

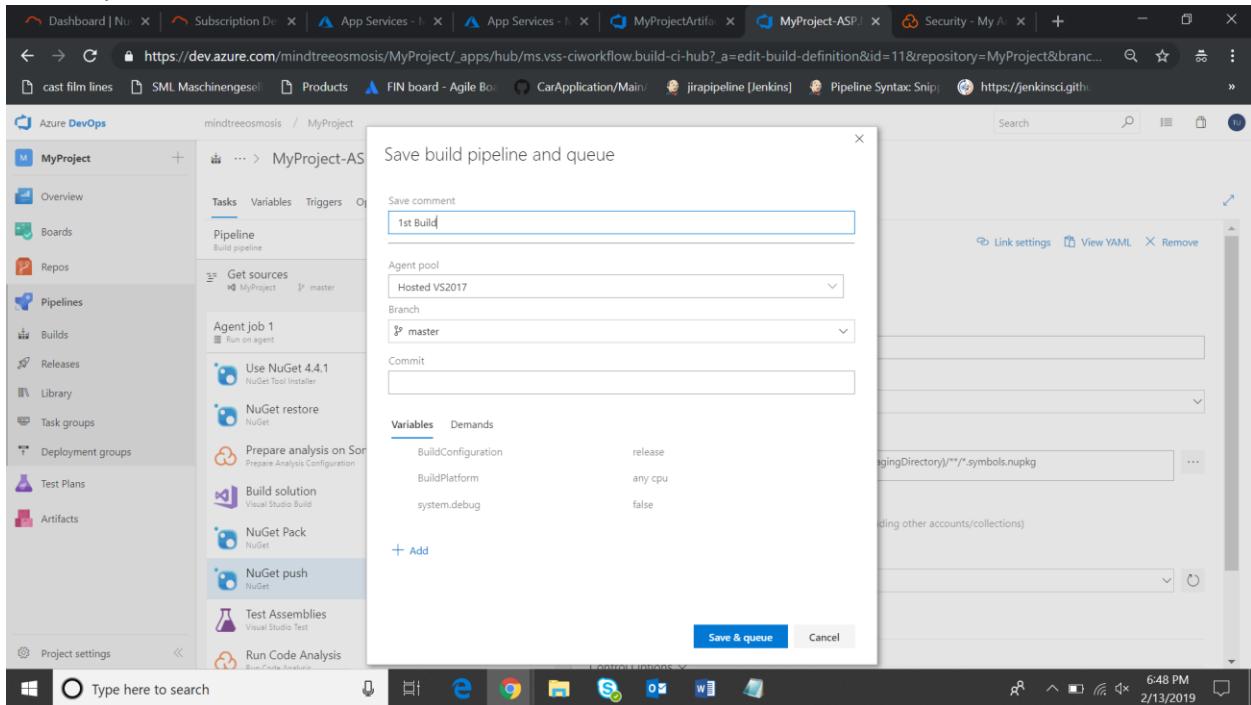
45. Click on “Triggers” -> Check “Enable Continious Integration”.



46. Below window shows the order of the tasks.



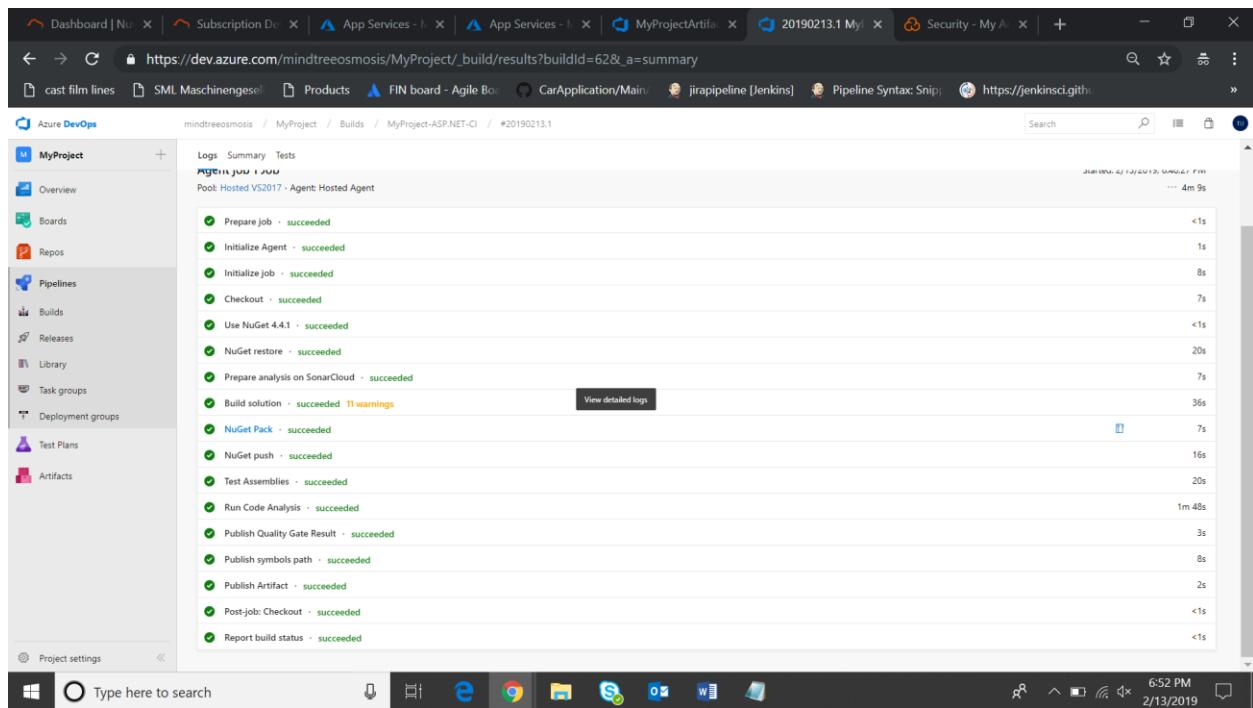
47. Click on “Save and Queue” → This will start a build → In a new Project the first Build has to be initiated manually.



48. The build has started.

49. Tasks assigned are being performed by the agent.[Click on #201902131 to view this]

50. Build has successfully completed with all the below tasks executed completely.



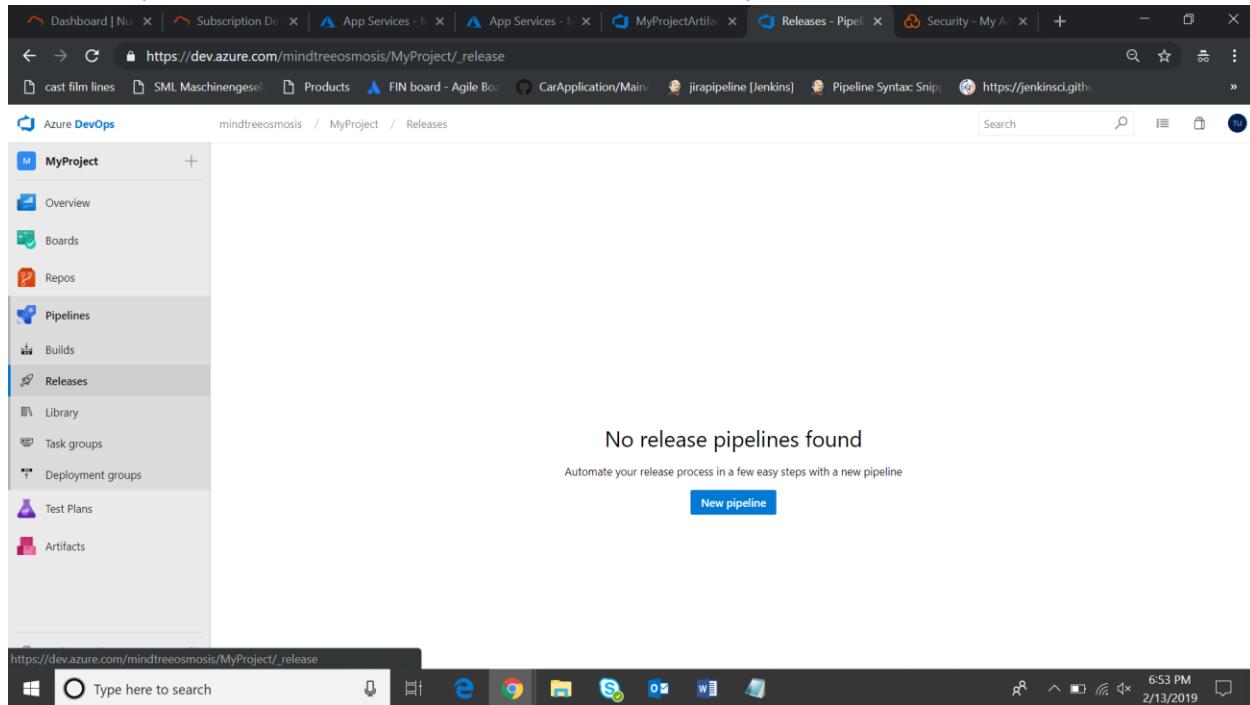
The screenshot shows the Azure DevOps Pipeline results page for a build named #20190213.1. The page displays a summary of the build steps and their execution times. The build was triggered from a Hosted VS2017 agent. The steps listed are:

- Prepare job - succeeded <1s
- Initialize Agent - succeeded 1s
- Initialize job - succeeded 8s
- Checkout - succeeded 7s
- Use NuGet 4.4.1 - succeeded <1s
- NuGet restore - succeeded 20s
- Prepare analysis on SonarCloud - succeeded 7s
- Build solution - succeeded 11 warnings 36s
- NuGet Pack - succeeded 7s
- NuGet push - succeeded 16s
- Test Assemblies - succeeded 20s
- Run Code Analysis - succeeded 1m 48s
- Publish Quality Gate Result - succeeded 3s
- Publish symbols path - succeeded 8s
- Publish Artifact - succeeded 2s
- Post-job: Checkout - succeeded <1s
- Report build status - succeeded <1s

The pipeline summary indicates a total duration of 4m 9s. A 'View detailed logs' button is available for further inspection.

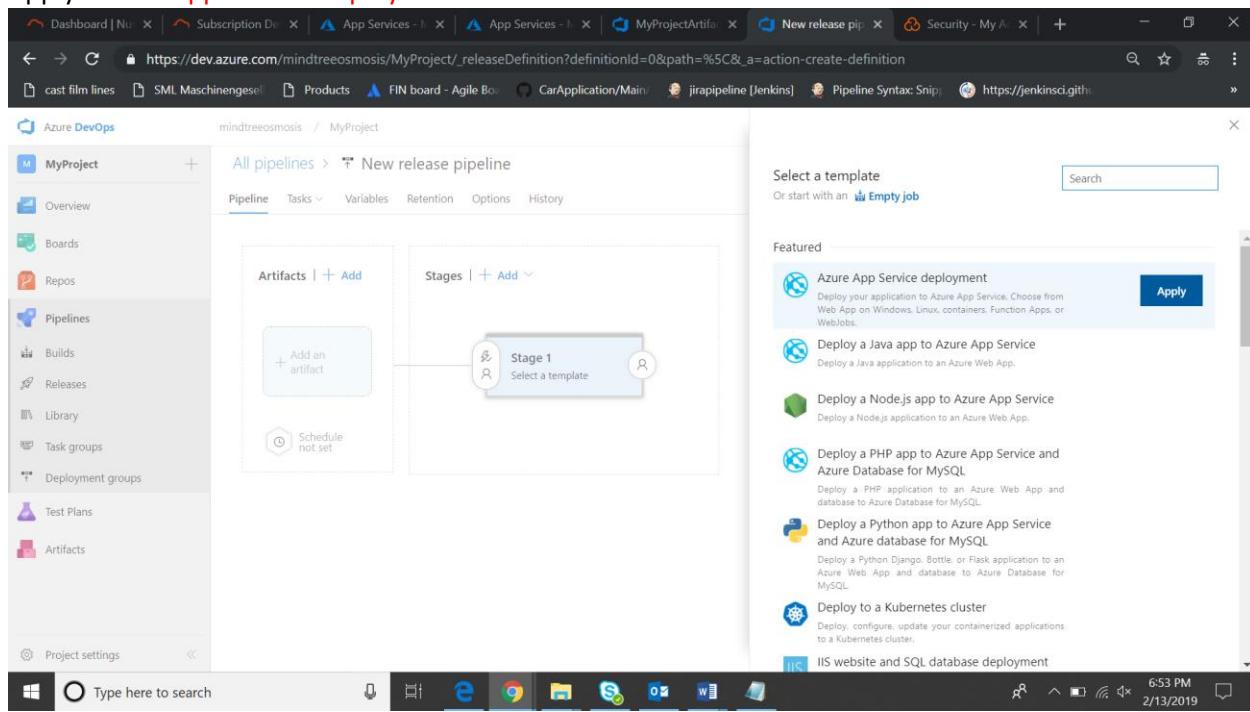
Release Pipeline

51. Click on “Pipelines” → Click on “Releases” → Click on “New Pipeline”.



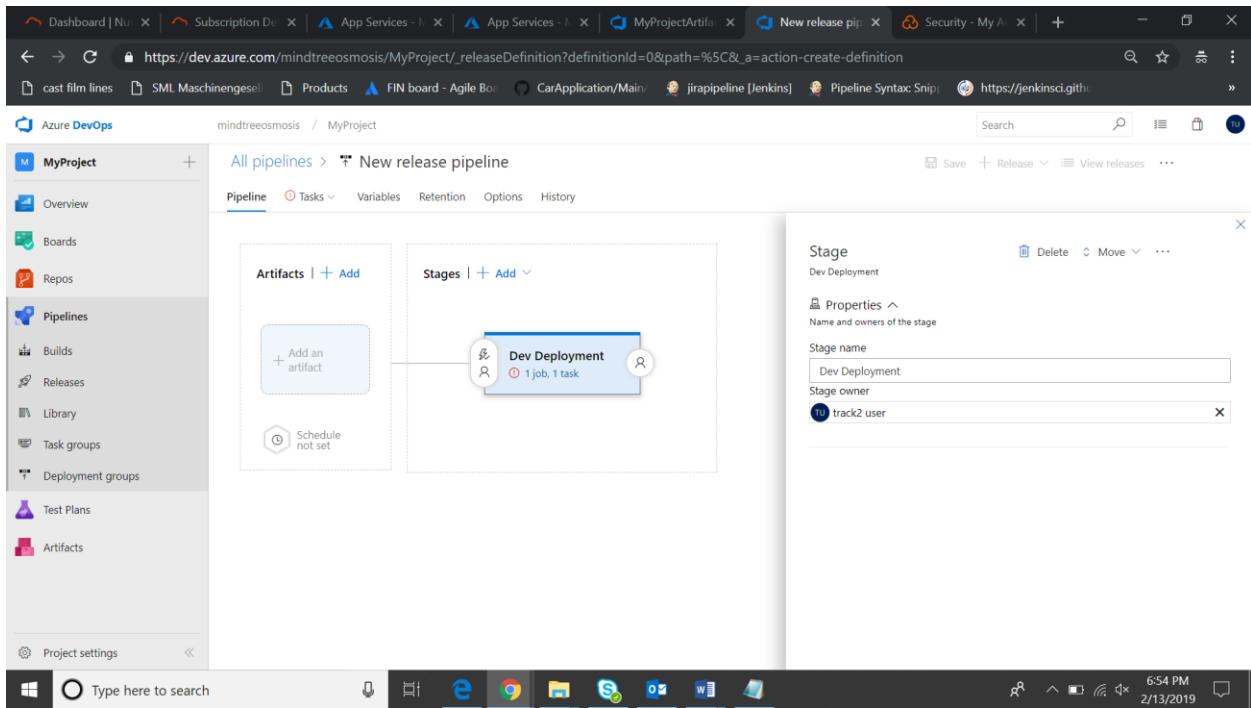
The screenshot shows the Azure DevOps interface for a project named 'MyProject'. The left sidebar has 'Pipelines' selected. The main area displays a message: 'No release pipelines found' with a sub-instruction 'Automate your release process in a few easy steps with a new pipeline'. A prominent blue 'New pipeline' button is centered. The browser address bar shows the URL https://dev.azure.com/mindtreeosmosis/MyProject/_release. The taskbar at the bottom shows various application icons.

52. Apply “Azure App Service Deployment”.

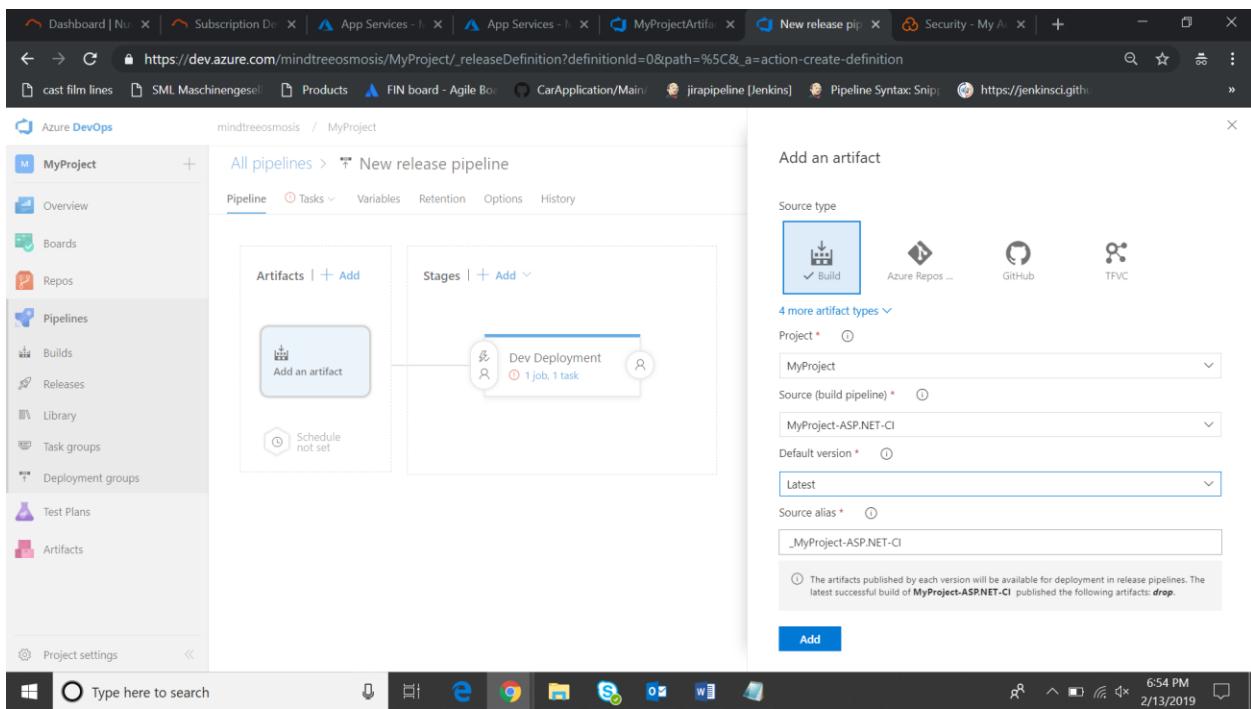


The screenshot shows the 'New release pipeline' creation page. The left sidebar is the same as the previous screenshot. The main area shows the 'Stages' tab selected, with a 'Stage 1' box labeled 'Select a template'. To the right, a 'Select a template' sidebar lists several options under 'Featured', each with a description and an 'Apply' button. The taskbar at the bottom shows various application icons.

53. Name the stage as “Dev Deployment” → And close the right Bullet tab.



54. Click on “Add an Artifact” → Select the source from the drop down → Select the version to be Latest → Click on Add.



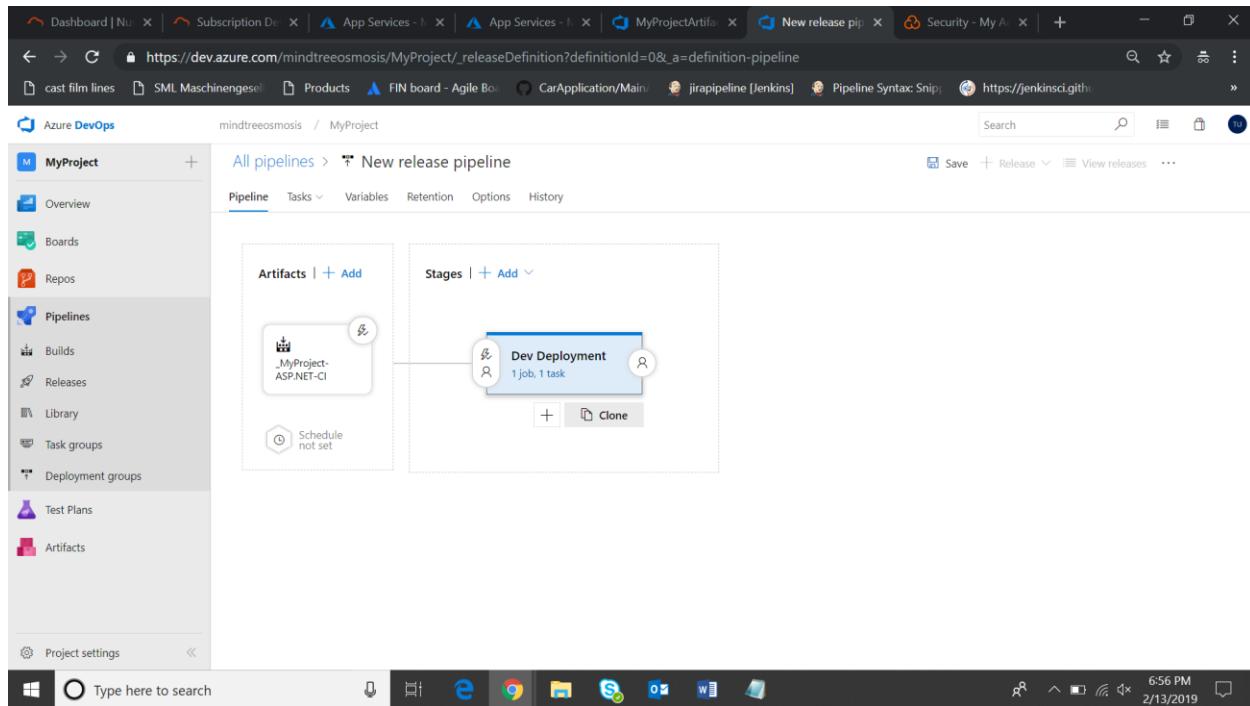
55. Click on the “Lightning Symbol” → Click on “Enable” under Continious Deployment Trigger → Close the bullet tab.

56. In stages under Dev Deployment → Click on “1 job, 1 task” → Select you Azure Subscription from the drop down → Click on “Authorize” → App Type should be set to “Web App on Windows” → Select the App Service Name from the drop down depending on your working environment (Dev Deployment in this case hence DevEnvOsmosis).

Note: The remaining steps are for Manual approval for team managers deployment and Higher authority approval.

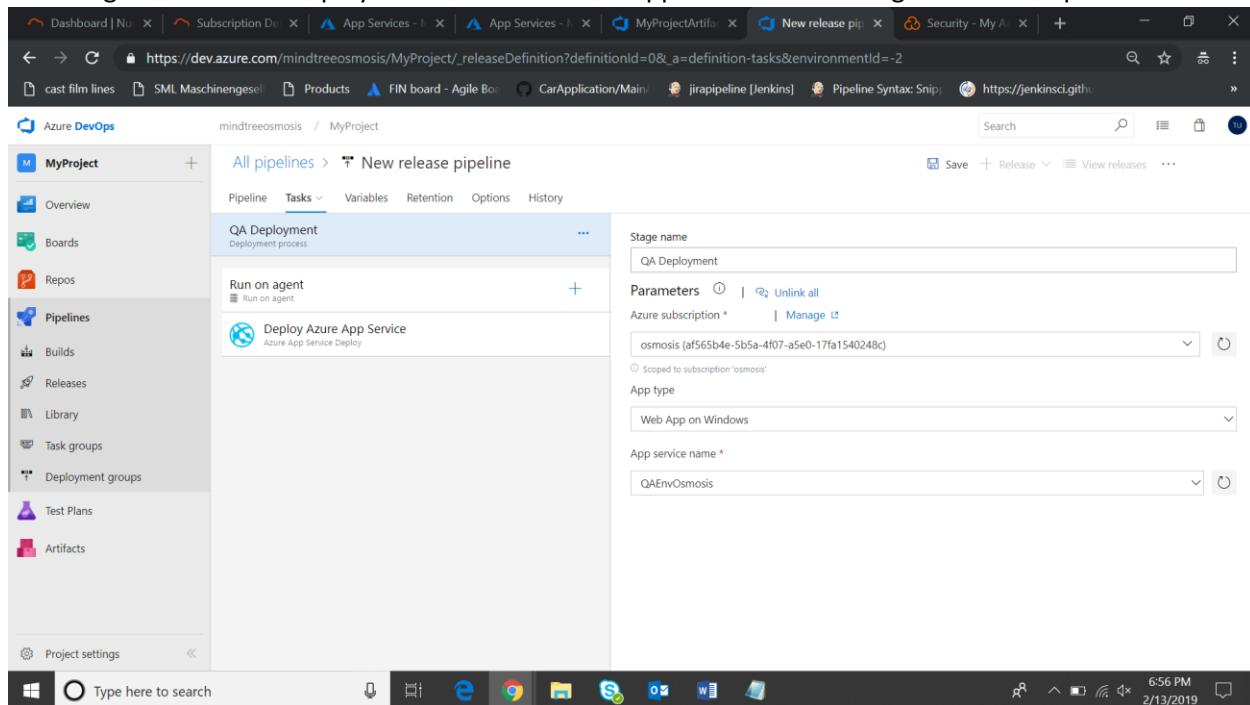
Otherwise continue with step 61.

57. Close the bullet tab → Click on Clone and follow the same steps for other environments.



Click “1 job, 1 task”

Give stage name as QA Deployment like and select “App service name” as given in azure portal.



Dashboard | Nu | Subscription De | App Services - | App Services - | MyProjectArti | New release pip | Security - My | +

cast film lines SML Maschinengesell Products FIN board - Agile Bo CarApplication/Main/ jirapipeline Jenkins Pipeline Syntax: Snip https://jenkinsci.github

Azure DevOps mindtreemosis / MyProject

All pipelines > New release pipeline

Pipeline Tasks Variables Retention Options History

Artifacts | + Add Stages | + Add

MyProject-ASP.NET-CI

Dev Deployment 1 job, 1 task

QA Deployment 1 job, 1 task

Schedule not set

+

Clone

Save Release View releases

Type here to search

6:56 PM 2/13/2019

Click “1 job, 1 task”

Give stage name as QA Deployment like and select “App service name” as given in azure portal.

Dashboard | Nu | Subscription De | App Services - | App Services - | MyProjectArti | New release pip | Security - My | +

cast film lines SML Maschinengesell Products FIN board - Agile Bo CarApplication/Main/ jirapipeline Jenkins Pipeline Syntax: Snip https://jenkinsci.github

Azure DevOps mindtreemosis / MyProject

All pipelines > New release pipeline

Pipeline Tasks Variables Retention Options History

Prod Deployment Deployment process

Run on agent

+

Deploy Azure App Service Azure App Service Deploy

Stage name

Prod Deployment

Parameters

Unlink all

Azure subscription

osmosis (af565b4e-5b5a-4f07-a5e0-17fa1540248c)

Scoped to subscription 'osmosis'

App type

Web App on Windows

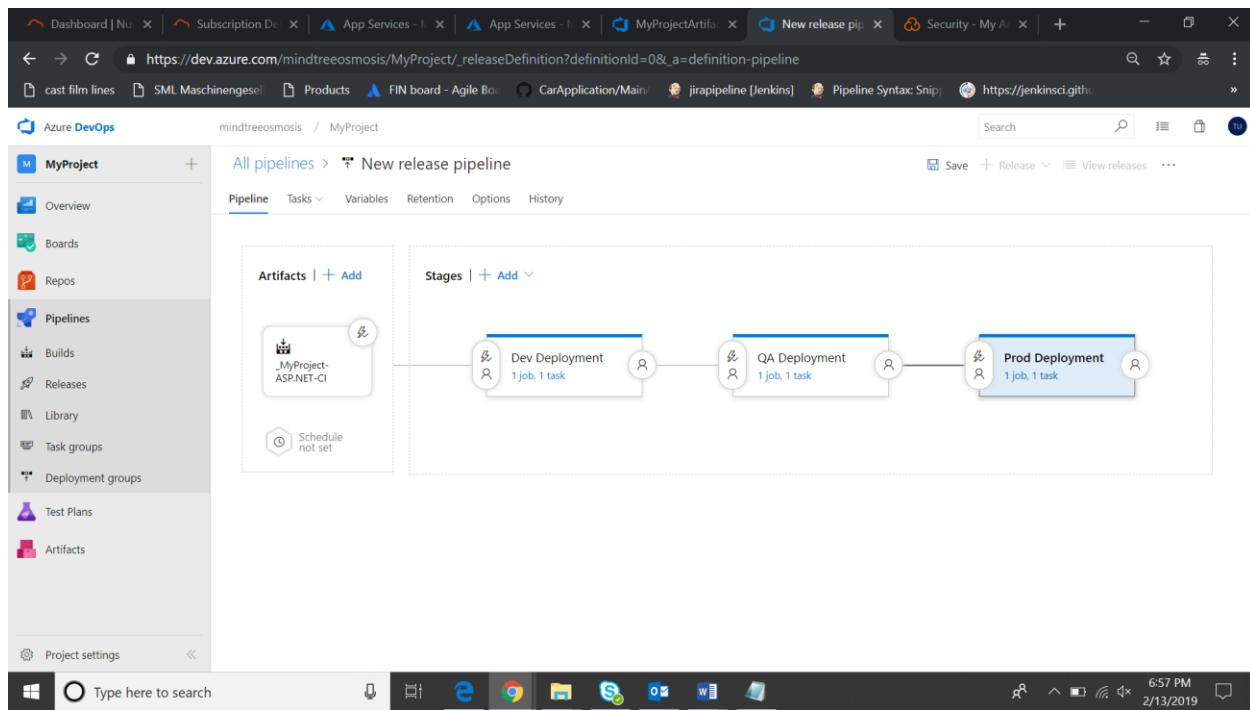
App service name *

ProdEnvOsmosis

Save Release View releases

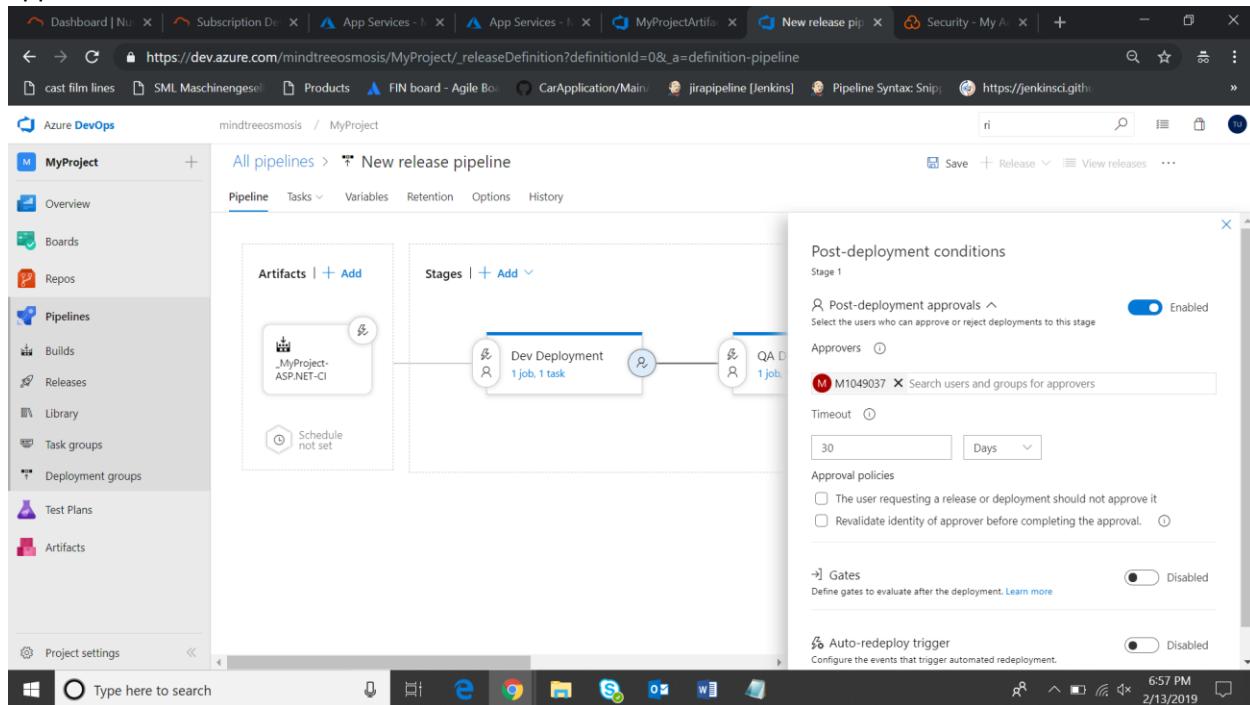
Type here to search

6:57 PM 2/13/2019

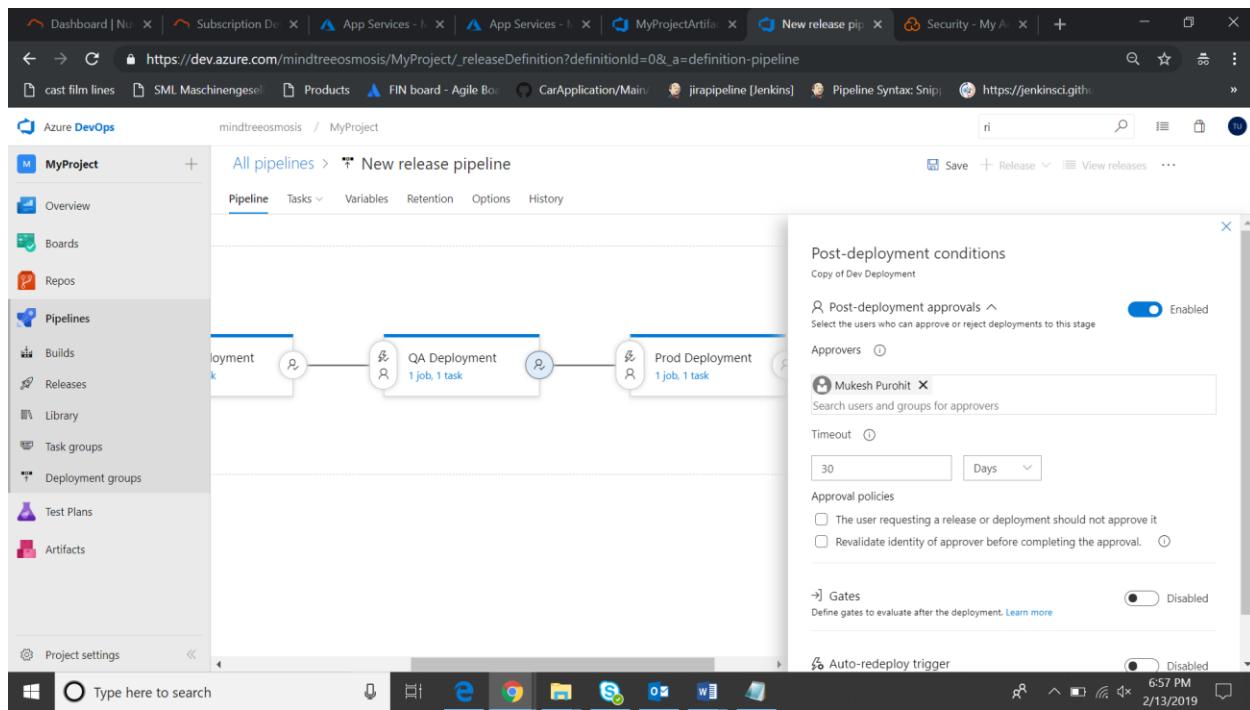


58. The Release pipeline is almost ready.

59. The QA and Production environments should get deployments only after manual approvals hence we need to set Post deployments Conditions → Click on the man icon → Enable Post deployment approvals and add the approver.



60. Similarly for QA to Production.

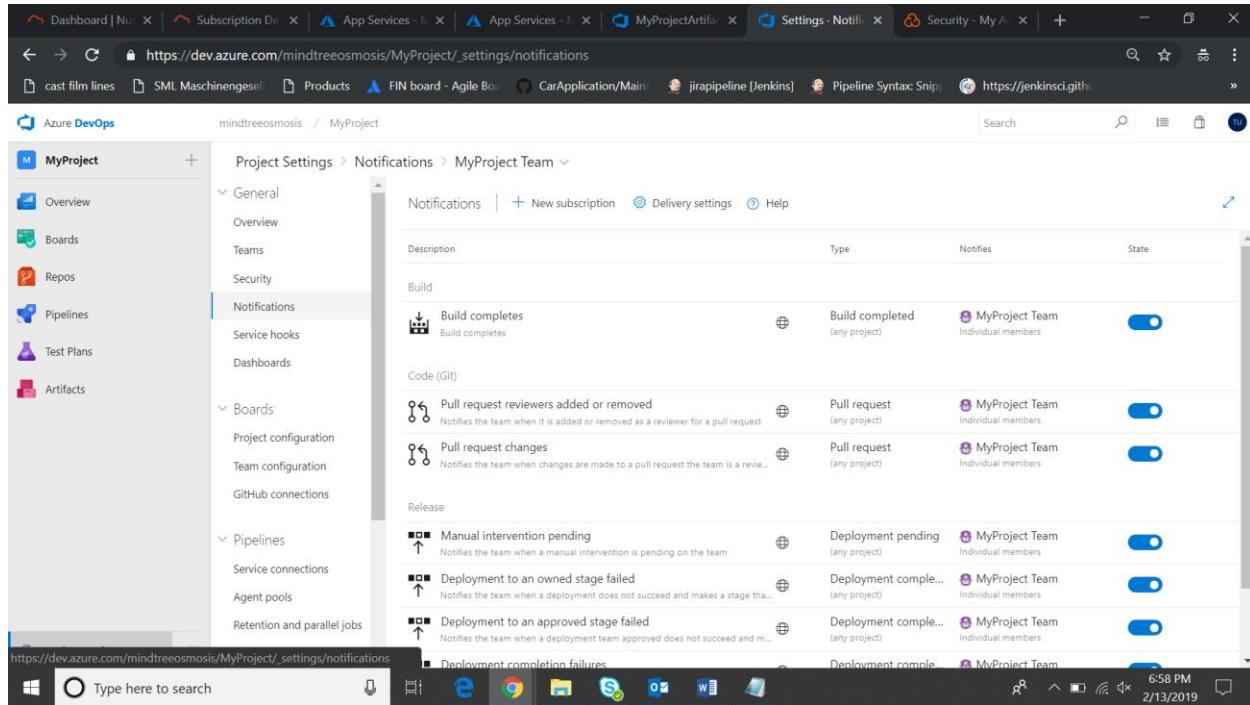


61. Click on “Save”.

Adding Notification services.

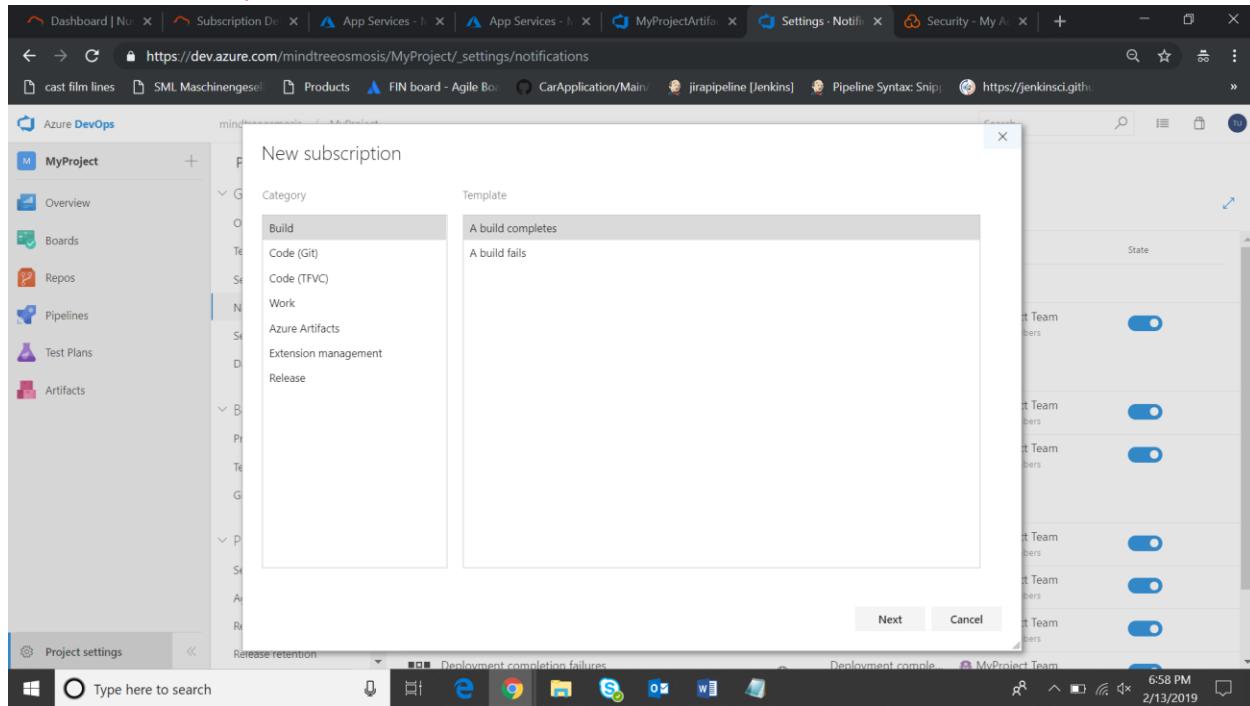
62. Click on “Project Settings”.

63. Click on “Notifications”.

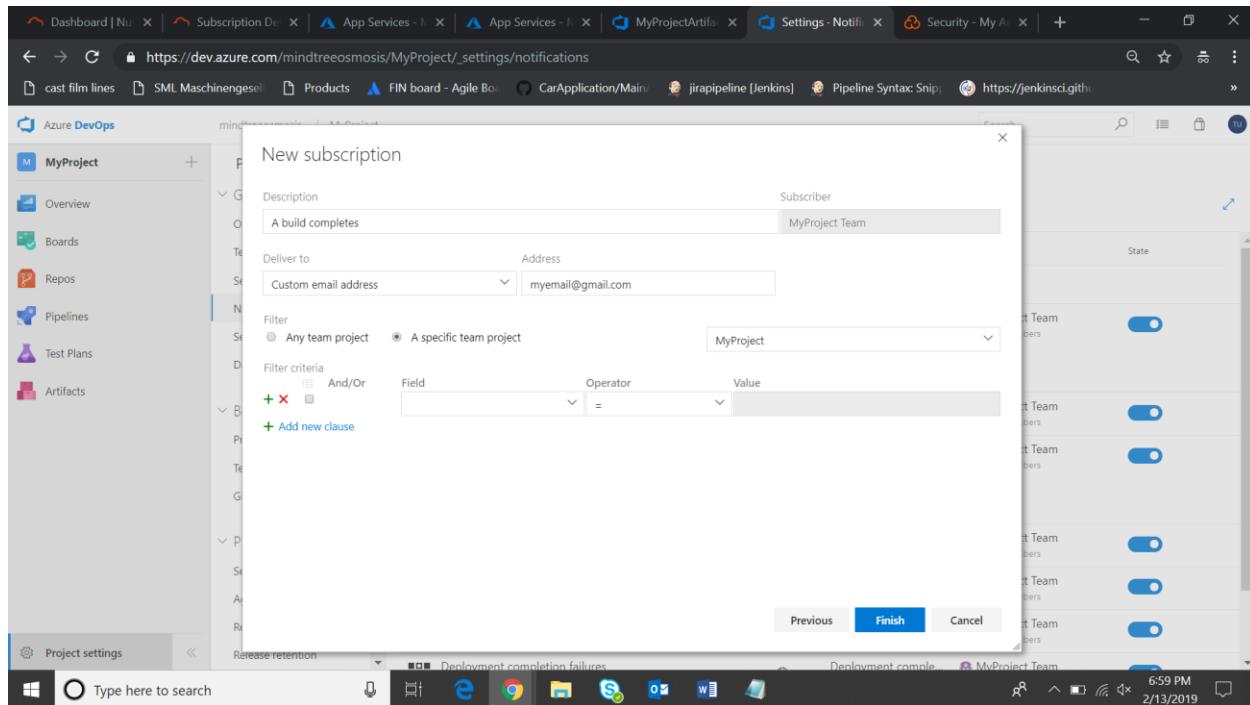


64. Click on “+ New Subscription”.

65. Select “A build Completes” → Click “Next”



66. Add a description → Select “Custom email address” in Deliver to → Type the email address you want to send the notifications to in the Address bar → Click “Finish”.



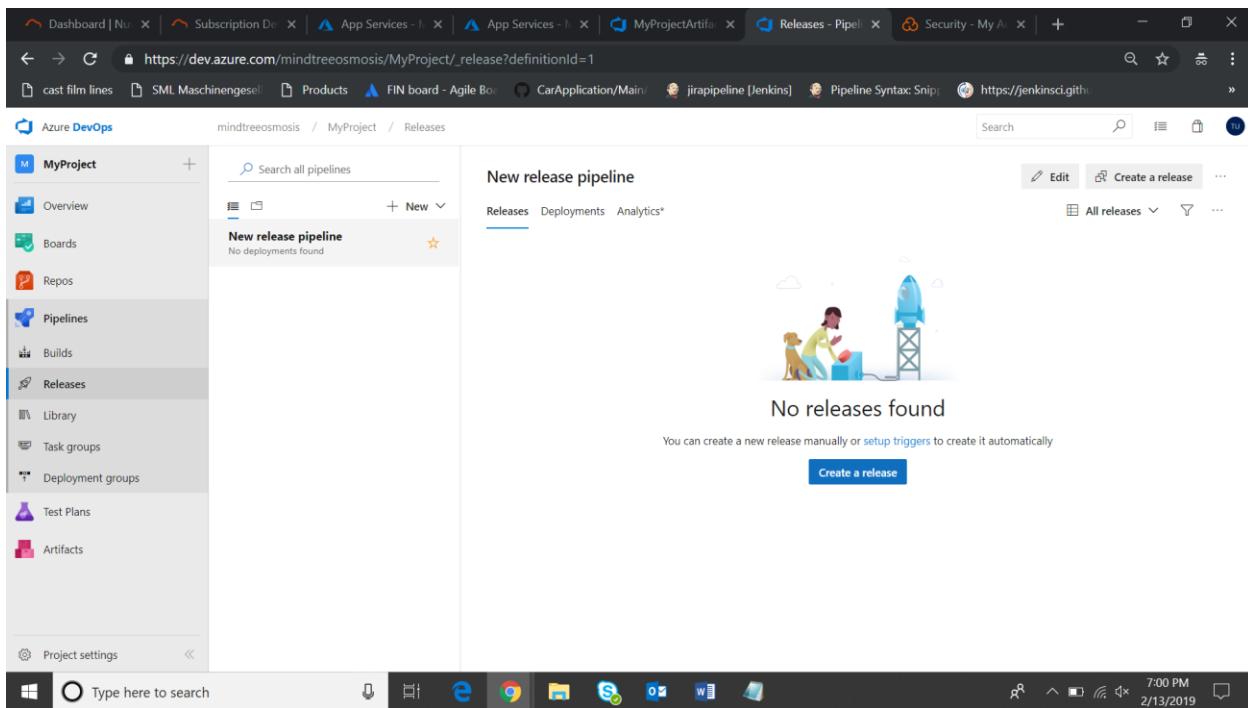
67. As the build has completed the tests written in the project has also been executed. The status of these test cases can be seen by clicking as follows.

68. Click on “Test Plans” on the right bullet bar → Click on “Runs”.

State	Run Id	Title	Completed Date	Build Number	Failed
Completed	50	VSTest Test Run release any cpu	13-02-2019 13:20:33	62	0

69. The build has succeeded and it's time to create a release.

70. Click on “Pipelines” → Click on “Releases” → Click on “Create a release”.



Azure DevOps

mindtreeosmosis / MyProject / Releases

MyProject

Overview Boards Repos Pipelines Builds Releases Library Task groups Deployment groups Test Plans Artifacts Project settings

Search all pipelines

New release pipeline

No releases found

New release pipeline

Releases Deployments Analytics

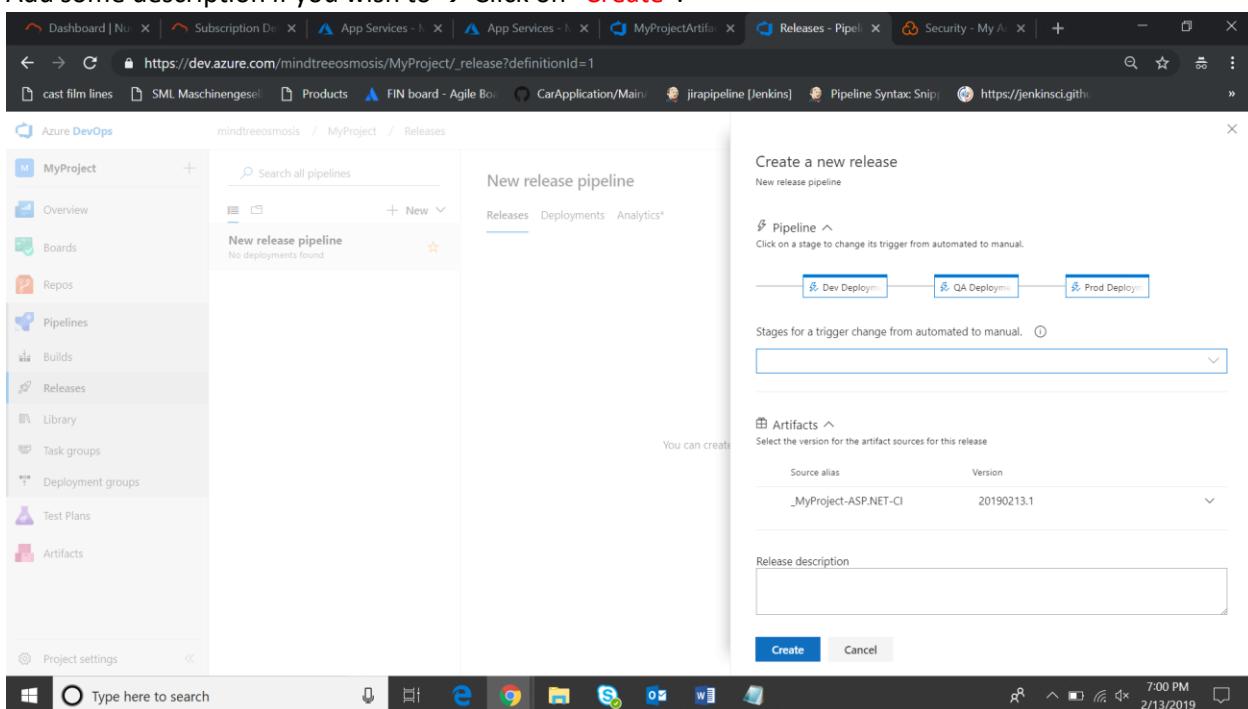
No releases found

You can create a new release manually or [setup triggers](#) to create it automatically

Create a release

7:00 PM 2/13/2019

71. Add some description if you wish to → Click on “Create”.



Dashboard | Nu | Subscription De | App Services - | App Services - | MyProjectArtifa | Releases - Pipe | Security - My A | +

cast film lines SML Maschinengesell Products FIN board - Agile Bo CarApplication/Main/ jirapipeline Jenkins Pipeline Syntax: Snip https://jenkinsci.github

mindtreeosmosis / MyProject / Releases

MyProject

Overview Boards Repos Pipelines Builds Releases Library Task groups Deployment groups Test Plans Artifacts Project settings

Search all pipelines

New release pipeline

Releases Deployments Analytics

Create a new release

New release pipeline

Pipeline

Click on a stage to change its trigger from automated to manual.

Dev Deploy QA Deploy Prod Deploy

Stages for a trigger change from automated to manual.

Artifacts

Select the version for the artifact sources for this release

Source alias Version

_MyProject-ASP.NET-CI 20190213.1

Release description

Create Cancel

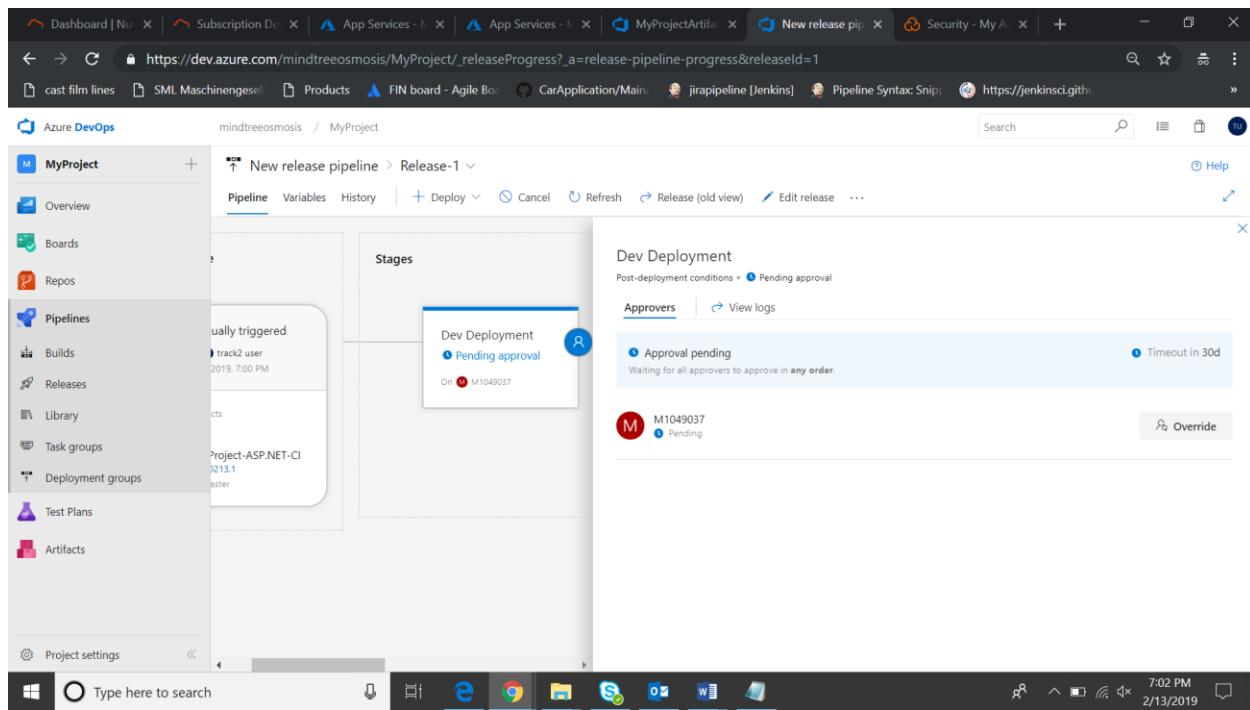
7:00 PM 2/13/2019

72. The first Release has been queued.

73. Click on the release name or stages and it will redirect you to a page where you can view the progress and also give approvals.

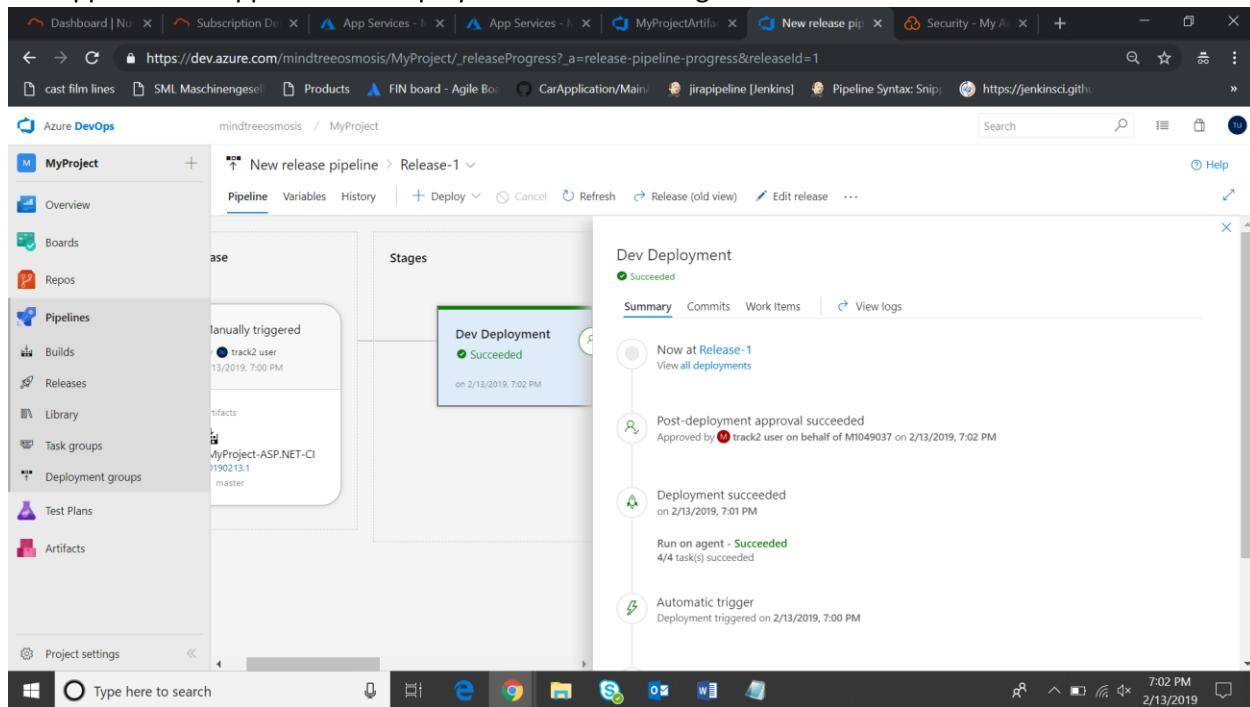
Note: <https://portal.azure.com/#blade/HubsExtension/Resources/resourceType/Microsoft.Web%2Fsites>
 Go to this link -> Select your app service -> Click one the URL to view your application.

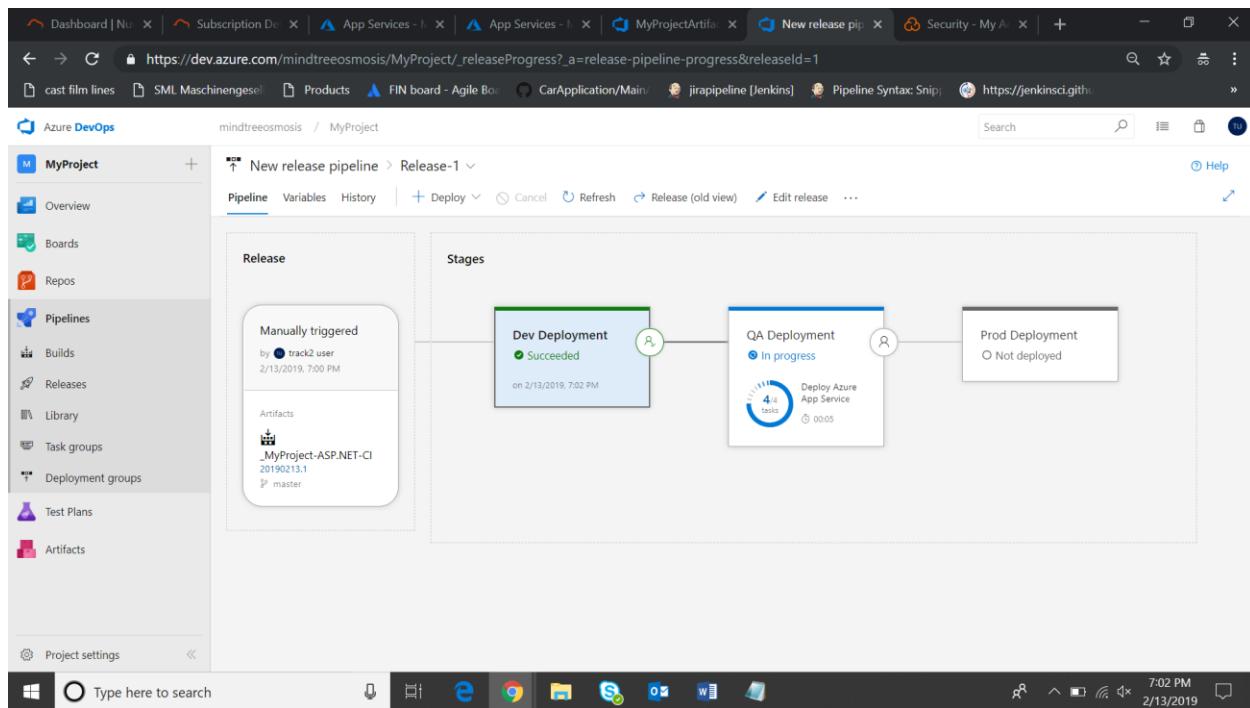
74. So the application has been deployed to the Dev Deployment stage/environment and is awaiting approval to be deployed to the QA stage/environment.



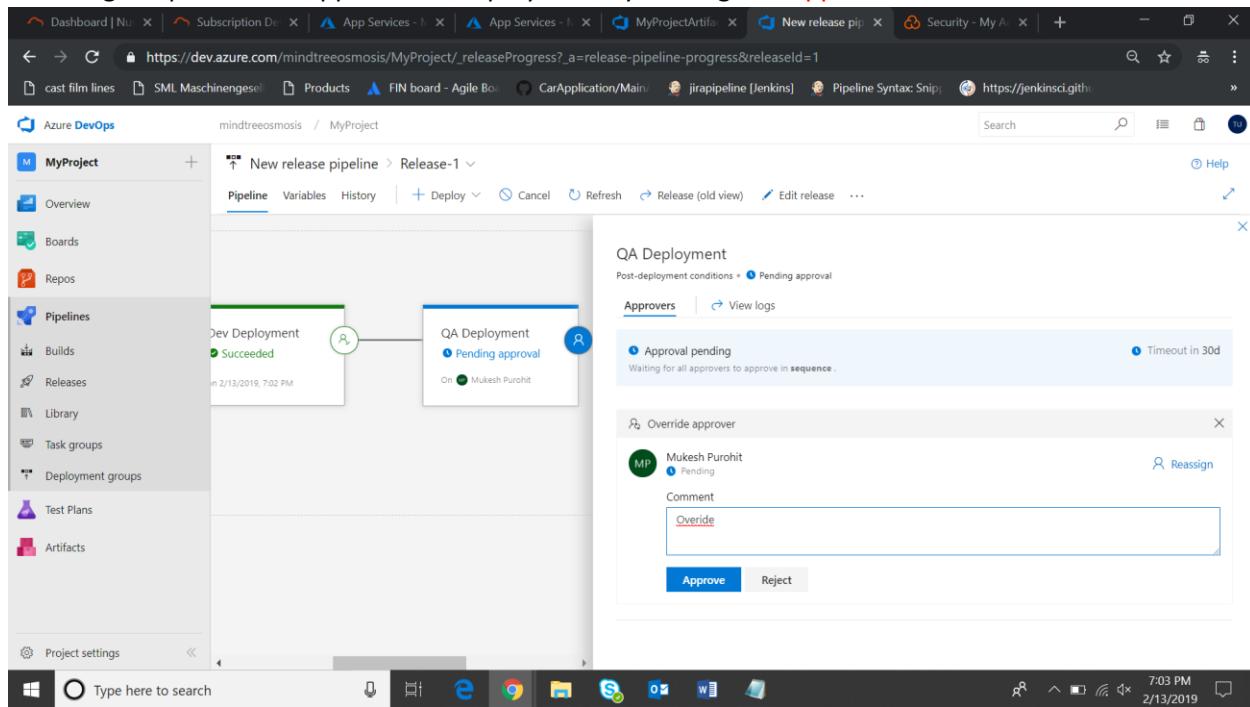
75. The assigned person can approve the QA deployment by clicking on “Approve”.

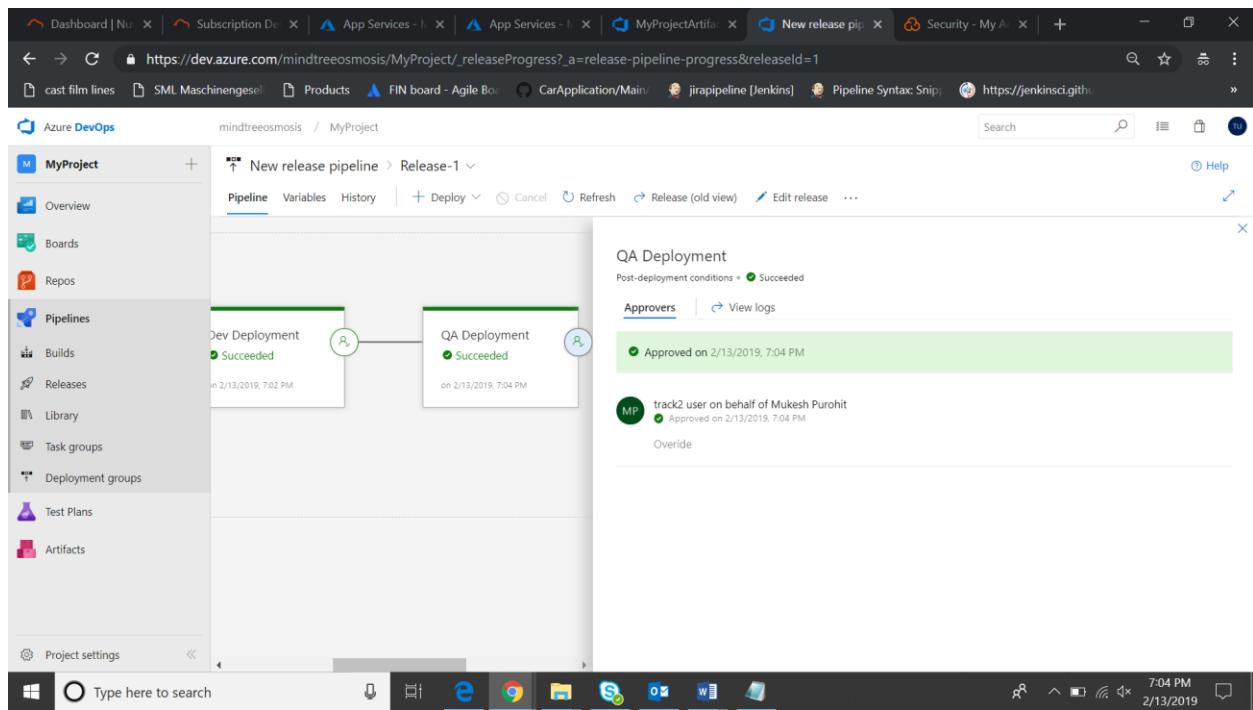
76. The approver has approved the deployment to the QA stage.



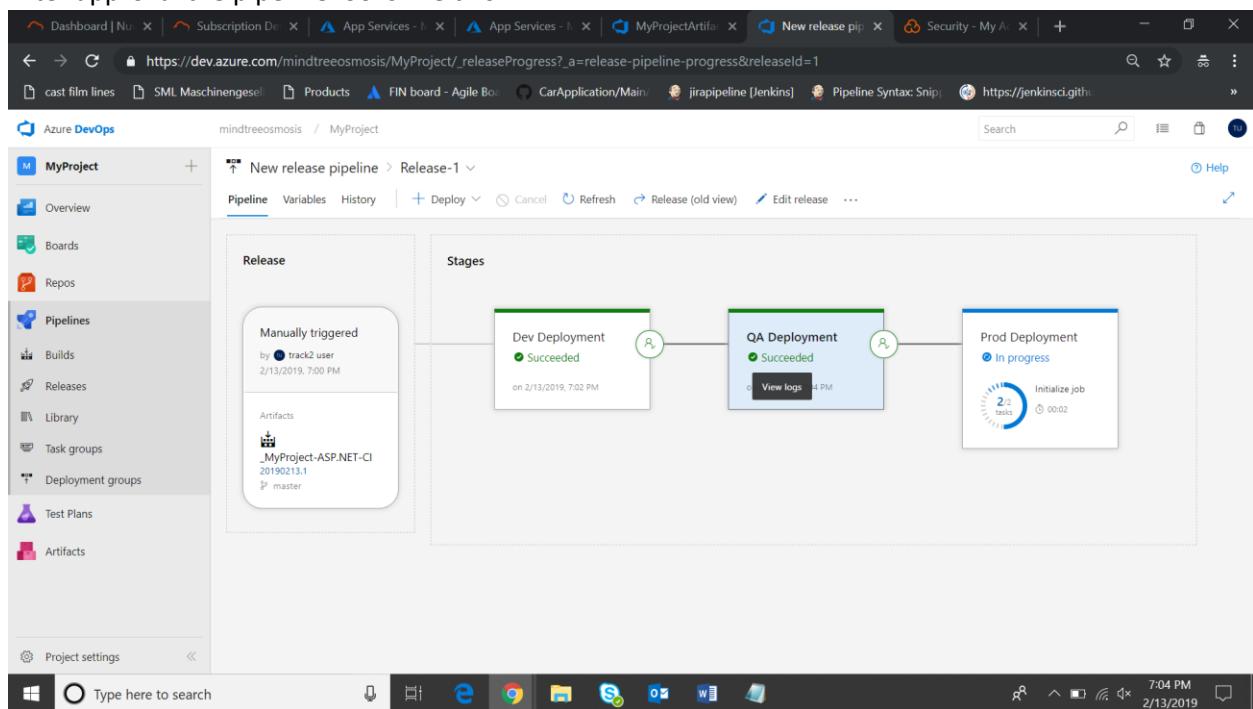


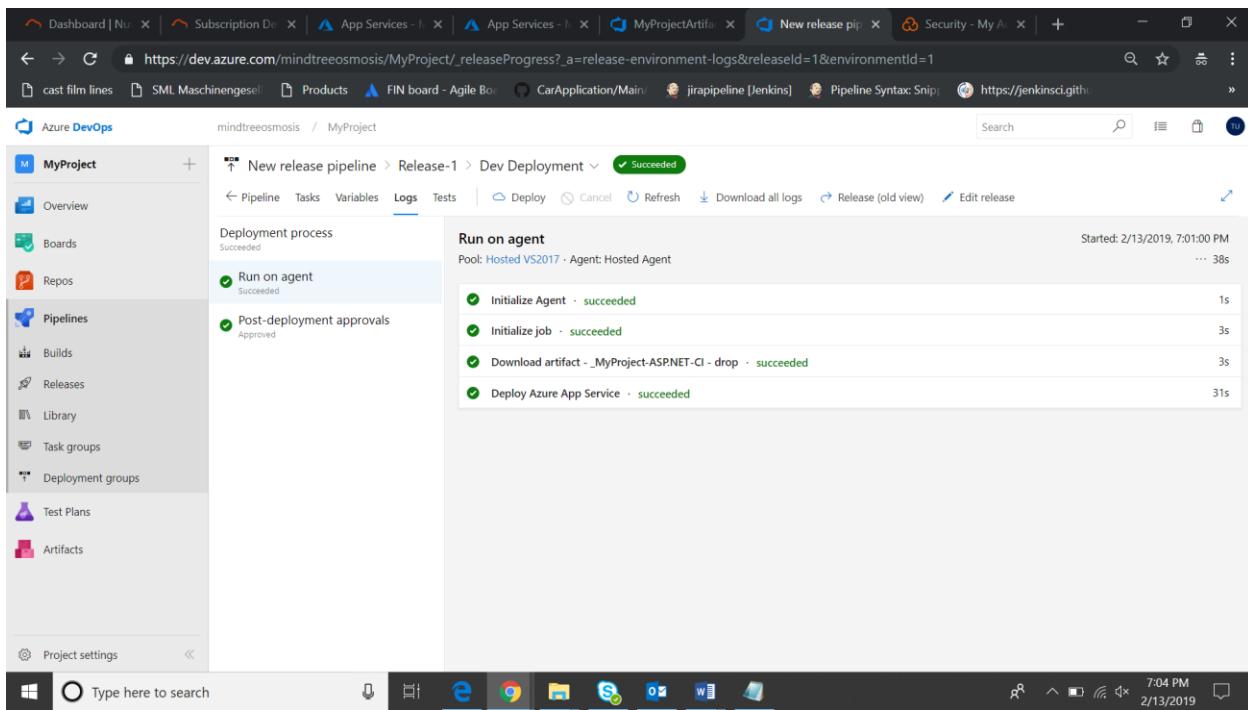
77. Has been deployed in the QA stage and is awaiting approval in for the Production stage/environment.
 78. The assigned person can approve the deployment by clicking on “Approve”.





After approval the pipeline looks like this.



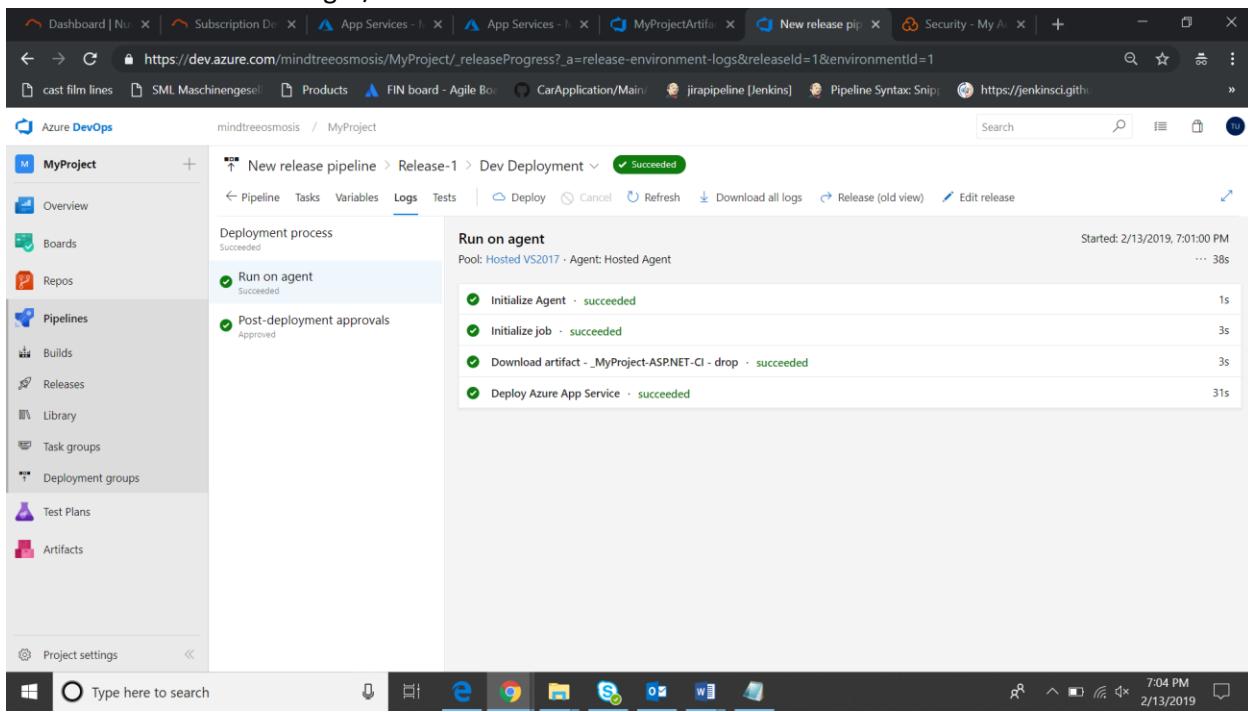


The screenshot shows the Azure DevOps interface for a release pipeline. The left sidebar is titled 'MyProject' and includes options like Overview, Boards, Repos, Pipelines, Builds, Releases, Library, Task groups, Deployment groups, Test Plans, and Artifacts. The 'Pipelines' option is selected. The main content area shows a 'New release pipeline > Release-1 > Dev Deployment' step with a green 'Succeeded' status. Below this, the 'Deployment process' section shows 'Run on agent' and 'Post-deployment approvals' both in 'Succeeded' status. The 'Logs' tab is selected, displaying a 'Run on agent' section with the following tasks and their outcomes:

Task	Status	Time
Initialize Agent	succeeded	1s
Initialize job	succeeded	3s
Download artifact - _MyProject-ASP.NET-CI - drop	succeeded	3s
Deploy Azure App Service	succeeded	31s

The status bar at the bottom indicates the task was started at 2/13/2019, 7:01:00 PM and completed at 7:04 PM on 2/13/2019.

79. Release to all the three stages/environments have succeeded.

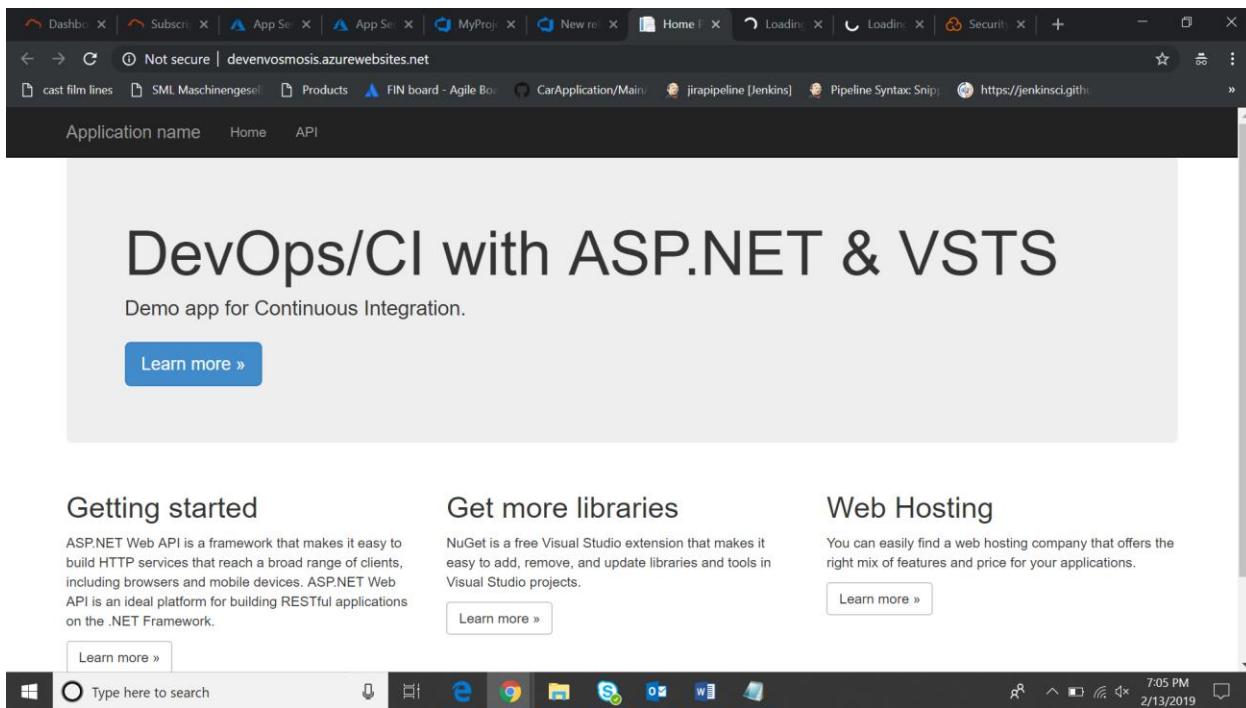


The screenshot is identical to the one above, showing the Azure DevOps interface for a release pipeline. The left sidebar is titled 'MyProject' and includes options like Overview, Boards, Repos, Pipelines, Builds, Releases, Library, Task groups, Deployment groups, Test Plans, and Artifacts. The 'Pipelines' option is selected. The main content area shows a 'New release pipeline > Release-1 > Dev Deployment' step with a green 'Succeeded' status. Below this, the 'Deployment process' section shows 'Run on agent' and 'Post-deployment approvals' both in 'Succeeded' status. The 'Logs' tab is selected, displaying a 'Run on agent' section with the following tasks and their outcomes:

Task	Status	Time
Initialize Agent	succeeded	1s
Initialize job	succeeded	3s
Download artifact - _MyProject-ASP.NET-CI - drop	succeeded	3s
Deploy Azure App Service	succeeded	31s

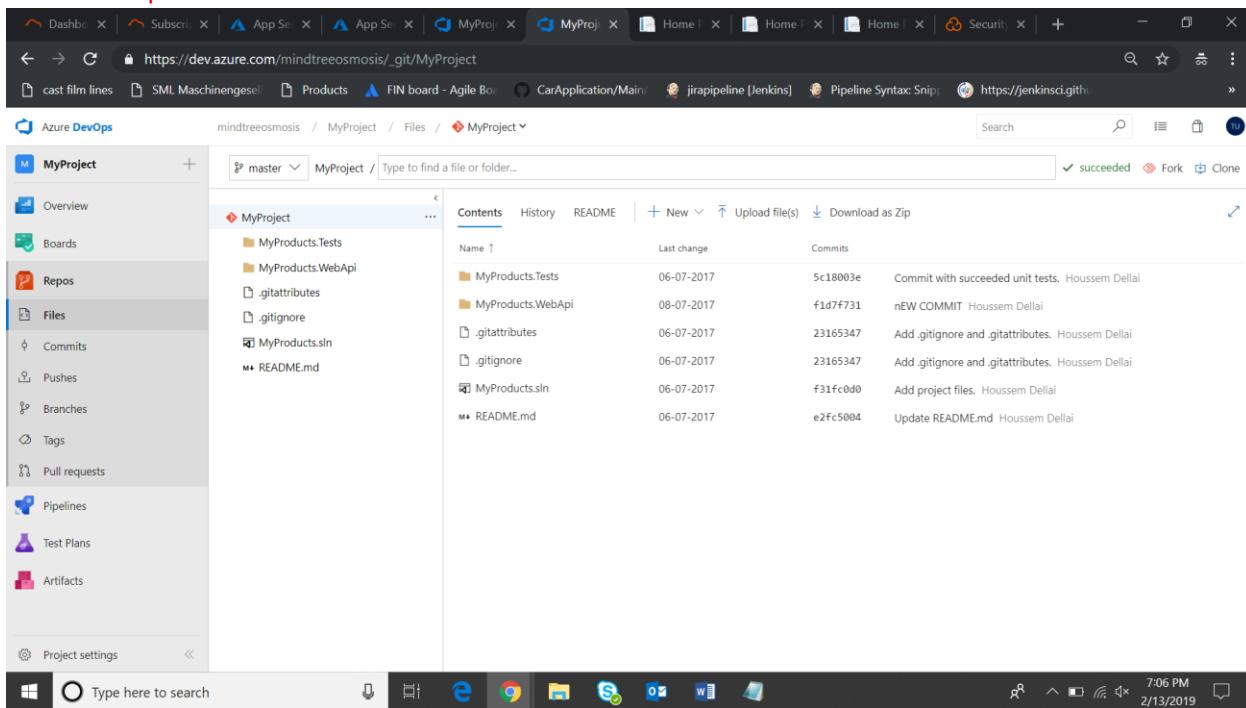
The status bar at the bottom indicates the task was started at 2/13/2019, 7:01:00 PM and completed at 7:04 PM on 2/13/2019.

80. Click on Deploy Azure App Service and scroll down to the bottom of the logs page, you can find the url which can be pasted on the browser to see the applications.

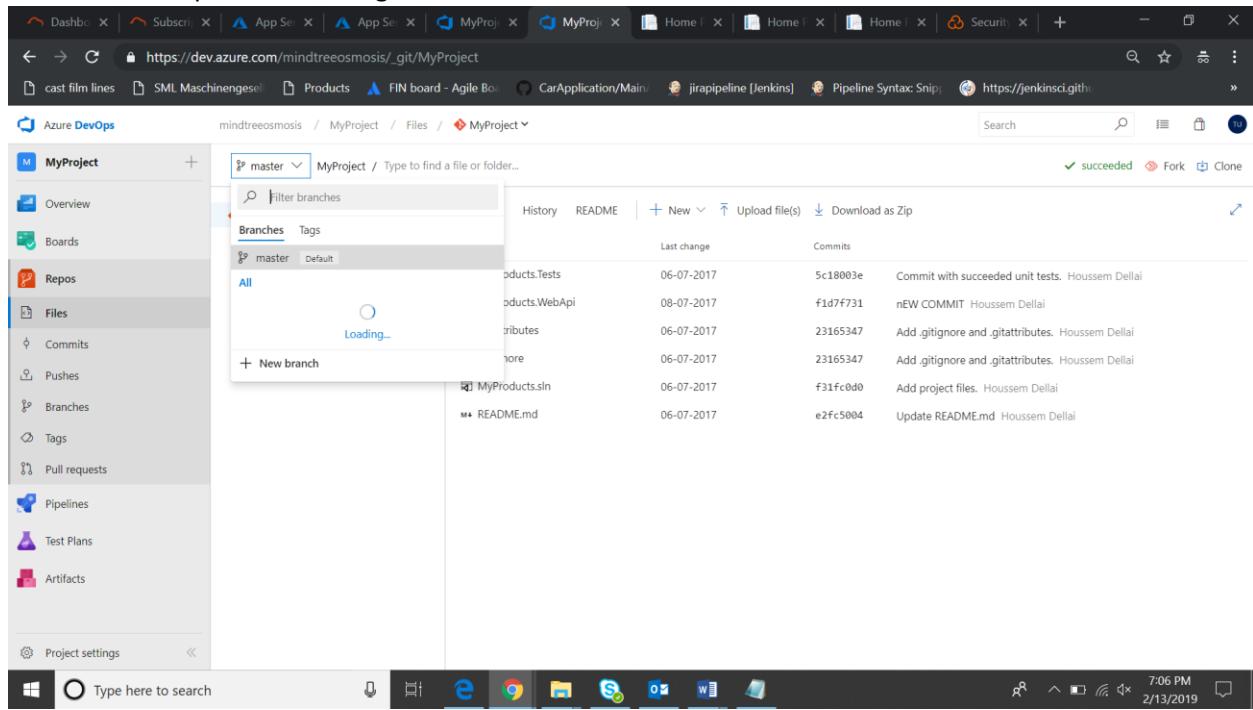


Azure Boards → Boards are used to assign tasks to developers and testers so that they can update or add new features to existing projects. The tasks assigned are done by creating a different branch that is a mirror image of the master.

81. Creating the branch to assign some user with a task. To be performed by the user on this created branch. His work is only merged with the master after various checks and approvals by the assigner.
82. Click on “**Repos**” → Click on “**Files**”.

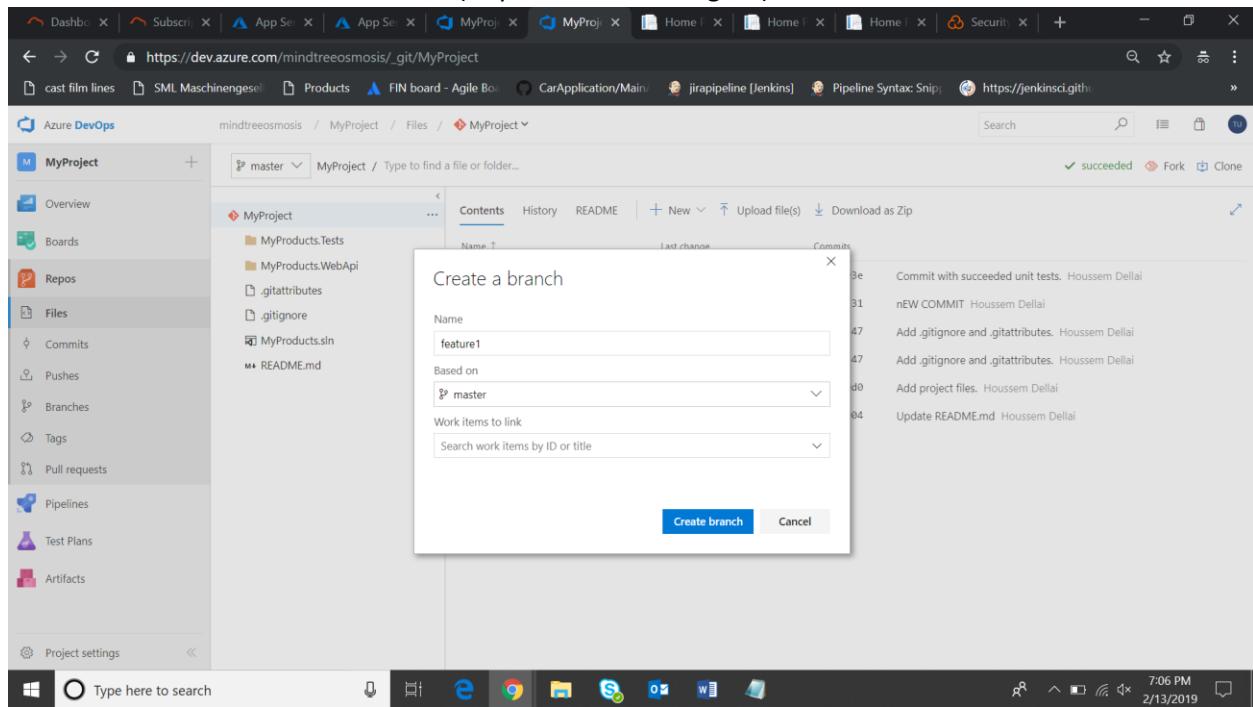


83. Click on the drop down showing “master” branch → Click on “New Branch”

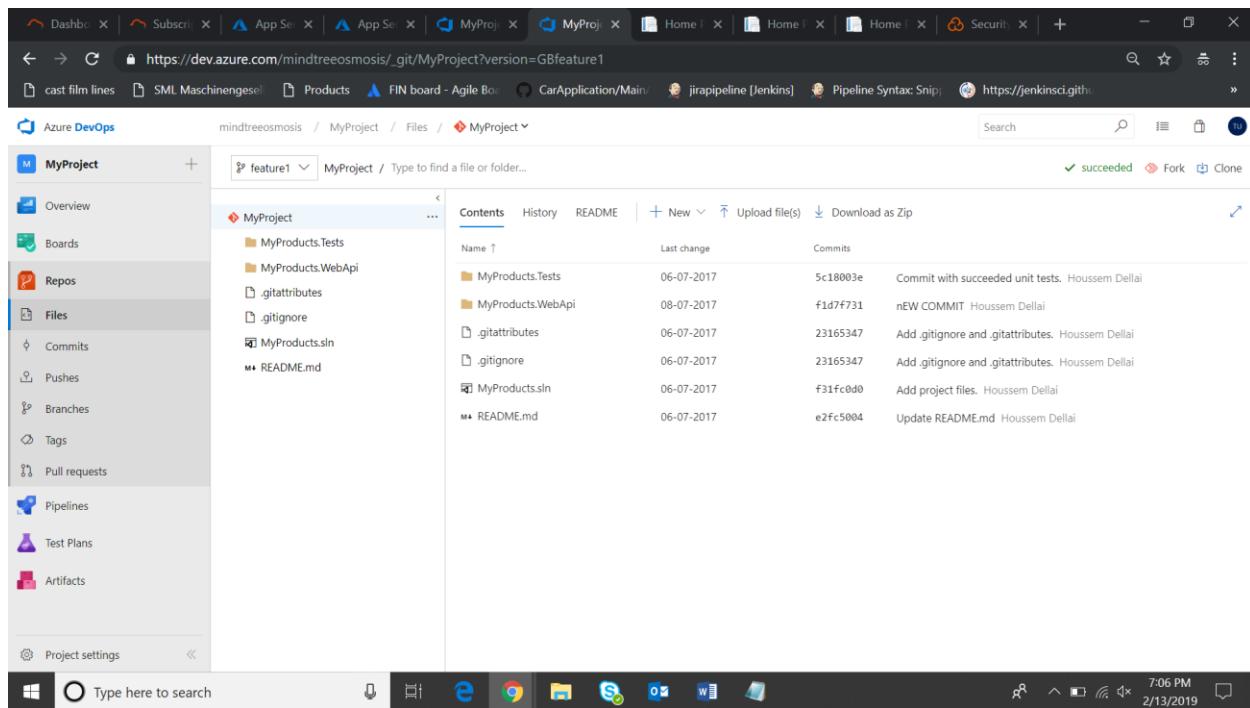


The screenshot shows the Azure DevOps interface for a project named 'MyProject'. The left sidebar is visible with various options like Overview, Boards, Repos, Files, Commits, Pushes, Branches, Tags, Pull requests, Pipelines, Test Plans, and Artifacts. The 'Files' option is selected. In the main content area, there is a dropdown menu for the current branch, which is set to 'master'. Below this dropdown, there is a link labeled '+ New branch'. The main table lists several files and their details, including 'MyProducts.sln' and 'README.md'. The status bar at the bottom shows the date and time as 2/13/2019 7:06 PM.

84. Give it a name → Based on “master” (Depends on the assigner) → Click “Create Branch”.



The screenshot shows the 'Create a branch' dialog box overlaid on the Azure DevOps interface. The dialog box has fields for 'Name' (set to 'feature1') and 'Based on' (set to 'master'). There is also a 'Work items to link' section with a search bar. At the bottom of the dialog box are 'Create branch' and 'Cancel' buttons. The background shows the same project structure and file list as the previous screenshot, with the 'master' branch selected.



https://dev.azure.com/mindtreeosmosis/_git/MyProject?version=GBfeature1

Azure DevOps mindtreeosmosis / MyProject / Files / MyProject

MyProject

Contents History README

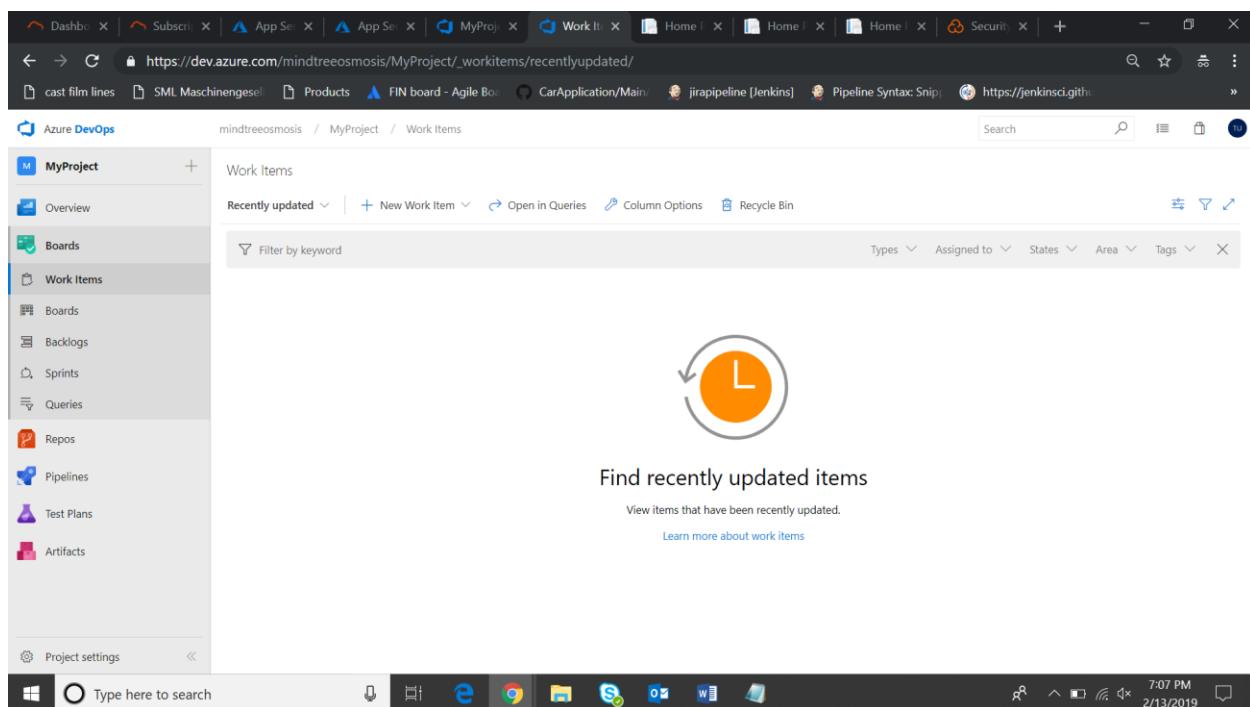
Name Last change Commits

- MyProducts.Tests 06-07-2017 5c18003e Commit with succeeded unit tests. Houssem Dellai
- MyProducts.WebApi 08-07-2017 f1d7f731 nEW COMMIT Houssem Dellai
- .gitattributes 06-07-2017 23165347 Add .gitignore and .gitattributes. Houssem Dellai
- .gitignore 06-07-2017 23165347 Add .gitignore and .gitattributes. Houssem Dellai
- MyProducts.sln 06-07-2017 f31fc0d0 Add project files. Houssem Dellai
- README.md 06-07-2017 e2fc5004 Update README.md Houssem Dellai

Type here to search

7:06 PM 2/13/2019

85. Click on “Boards” on the left bullet panel → Click on “Work Items” → Click on “+ New Work Item”.



https://dev.azure.com/mindtreeosmosis/_workitems/recentlyupdated/

Azure DevOps mindtreeosmosis / MyProject / Work Items

Work Items

Recently updated | + New Work Item | Open in Queries | Column Options | Recycle Bin

Filter by keyword

Types Assigned to States Area Tags

L

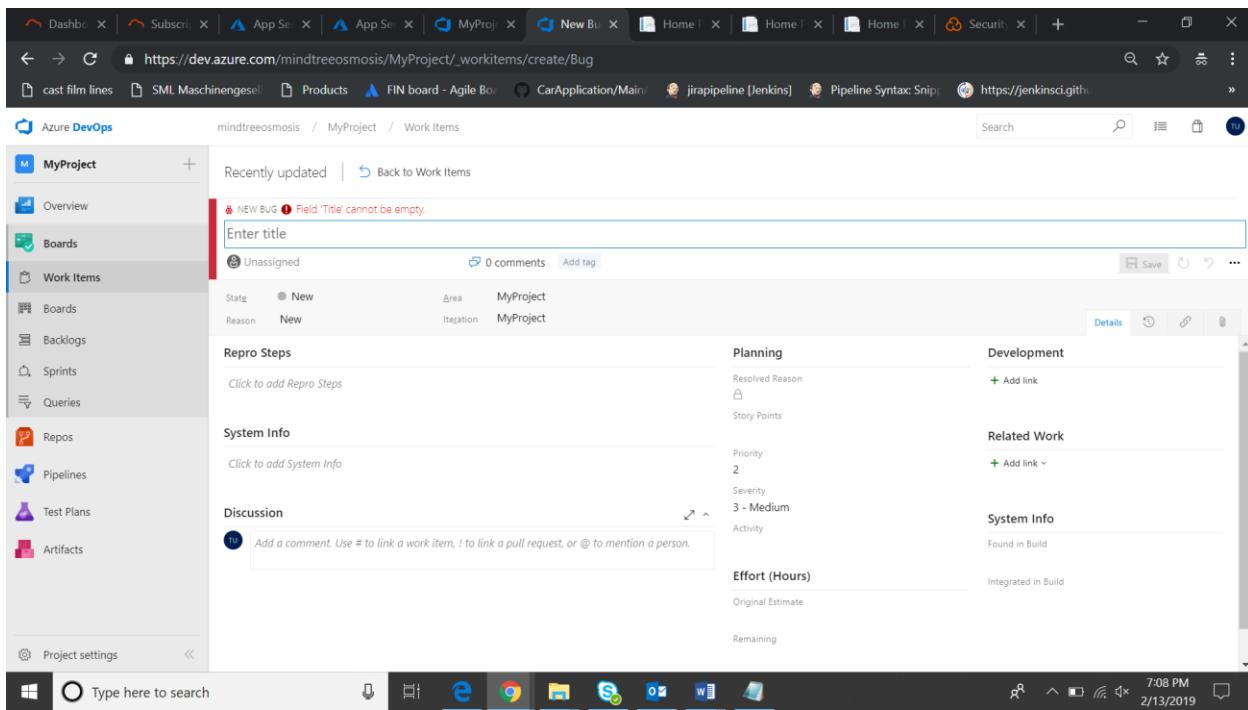
Find recently updated items

View items that have been recently updated.

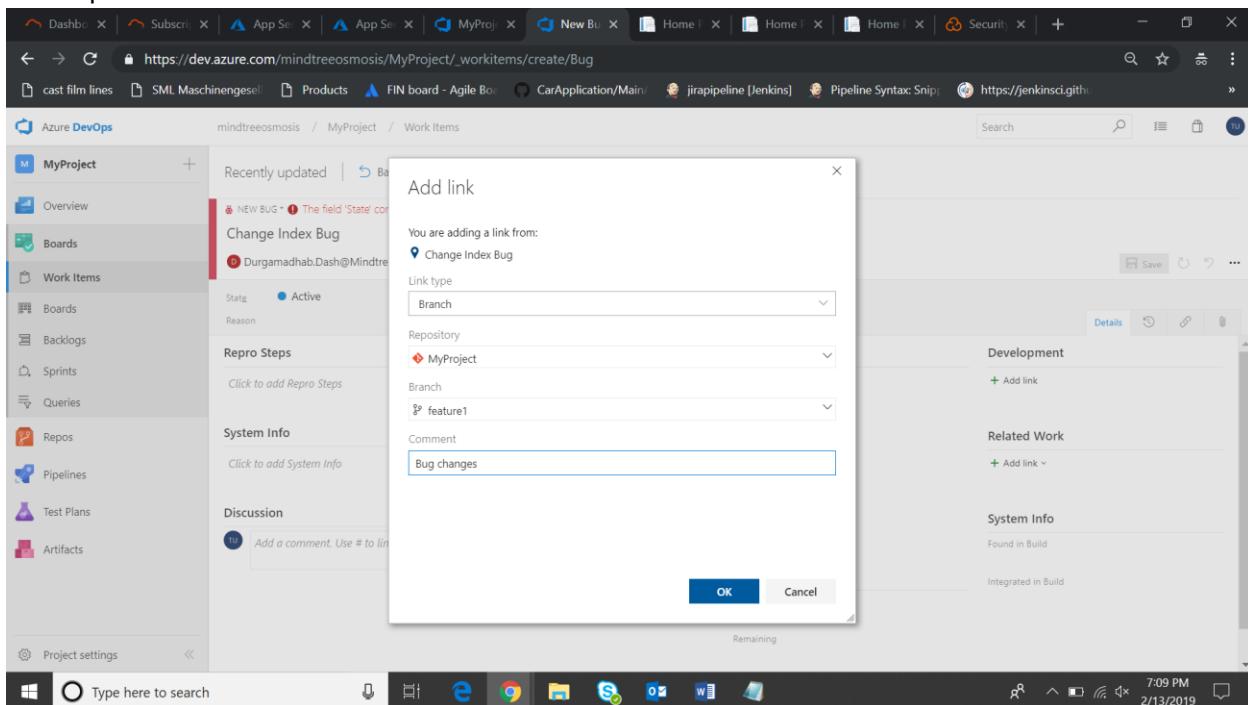
Learn more about work items

Type here to search

7:07 PM 2/13/2019

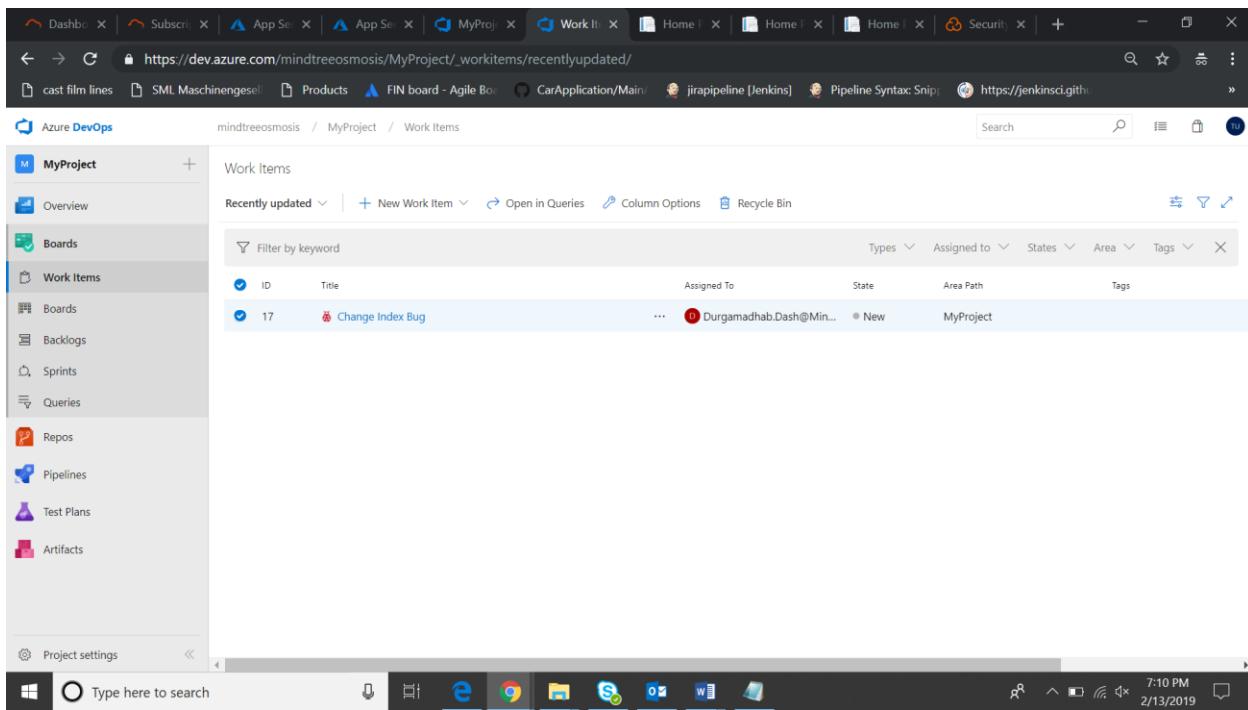


86. Enter the title → Assign the task by giving the name in the below tab → Change the status to “**Active**” → Click on “**+ Add Link**” (This is where the concerned has to visit to perform the job) → Select Link type to “**Branch**” → Select Repository to the one you have imported for this project at the beginning of this documentation → Branch should be set to “**Feature 1**” (where the work has to be performed so that if anything goes wrong it doesn’t mess up the “**master**”).



87. Click “**Save**”.

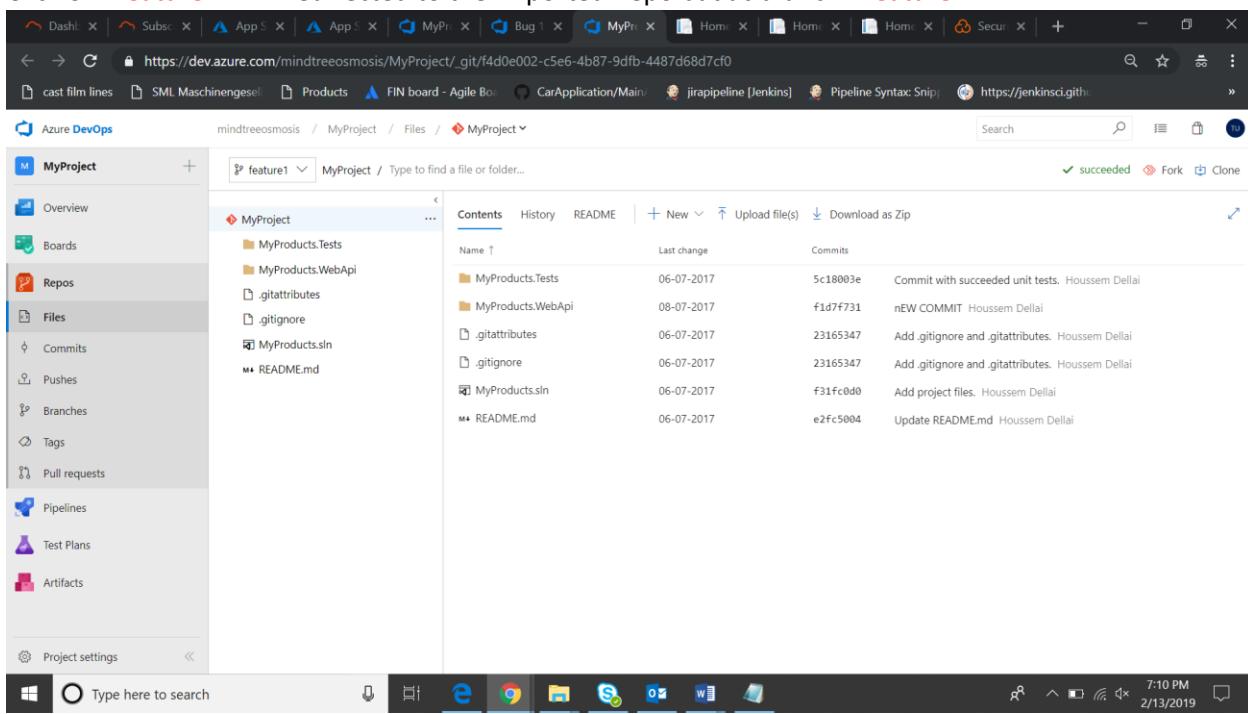
88. The assignee can see the tasks from his account on his board and get into that by clicking on it to perform his task.



The screenshot shows the Azure DevOps interface for the 'MyProject' project. The left sidebar is the navigation menu. The main content area is titled 'Work Items' and shows a table of work items. One work item is listed:

ID	Title	Assigned To	State	Area Path	Tags
17	Change Index Bug	Durgamadhab.Dash@Min...	New	MyProject	

89. Click on “feature 1” → Redirected to the imported Repo but at branch “Feature 1”.

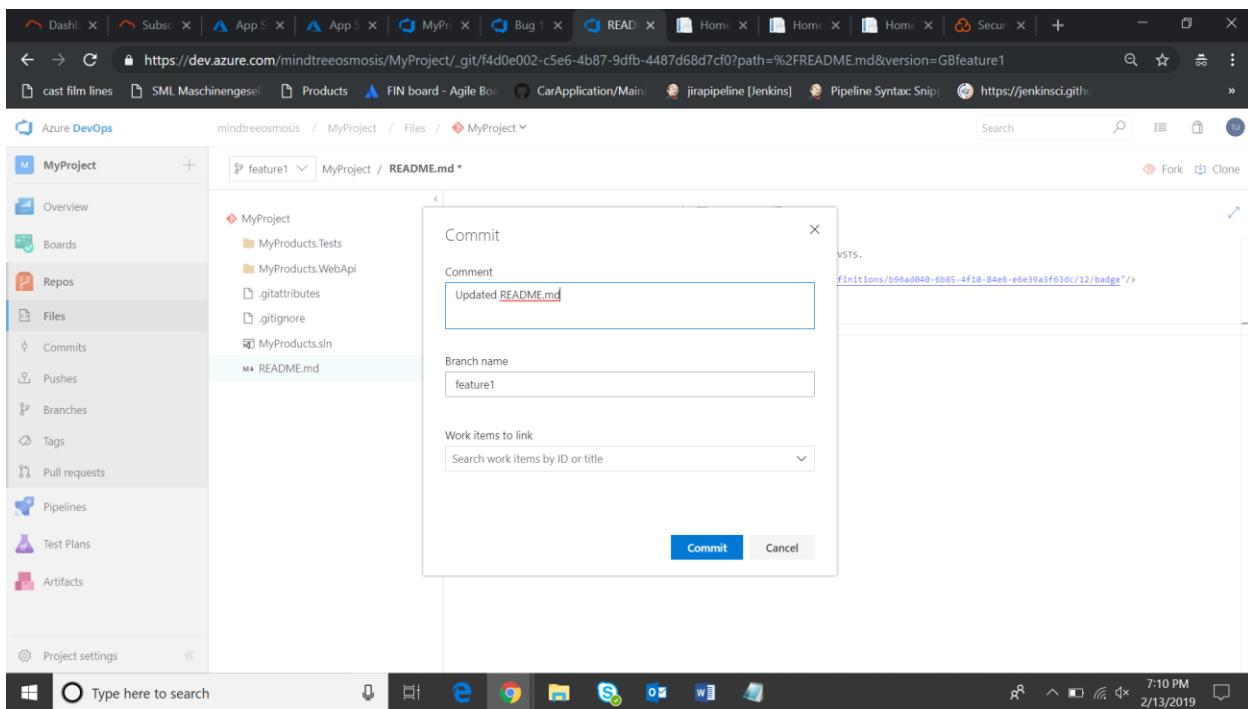
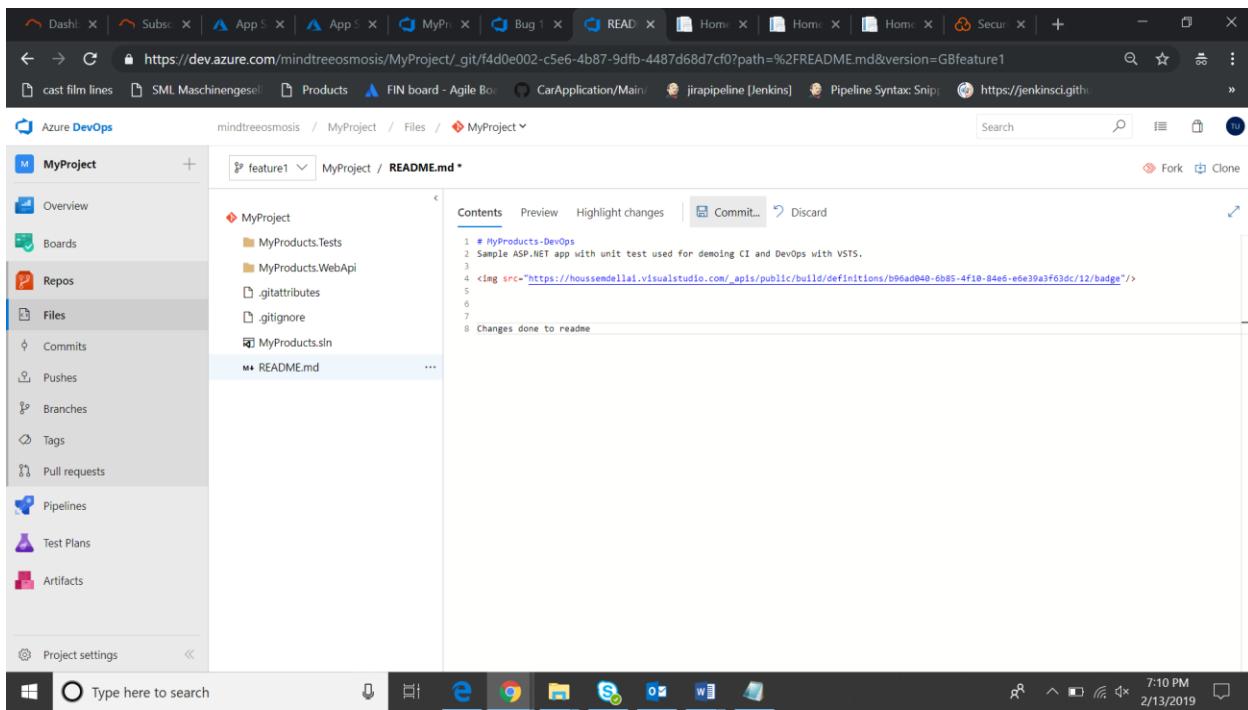


The screenshot shows the Azure DevOps interface for the 'MyProject' project, specifically the 'Files' section. The left sidebar is the navigation menu. The main content area shows a list of files in the 'feature1' branch of the 'MyProject' repository. The files listed are:

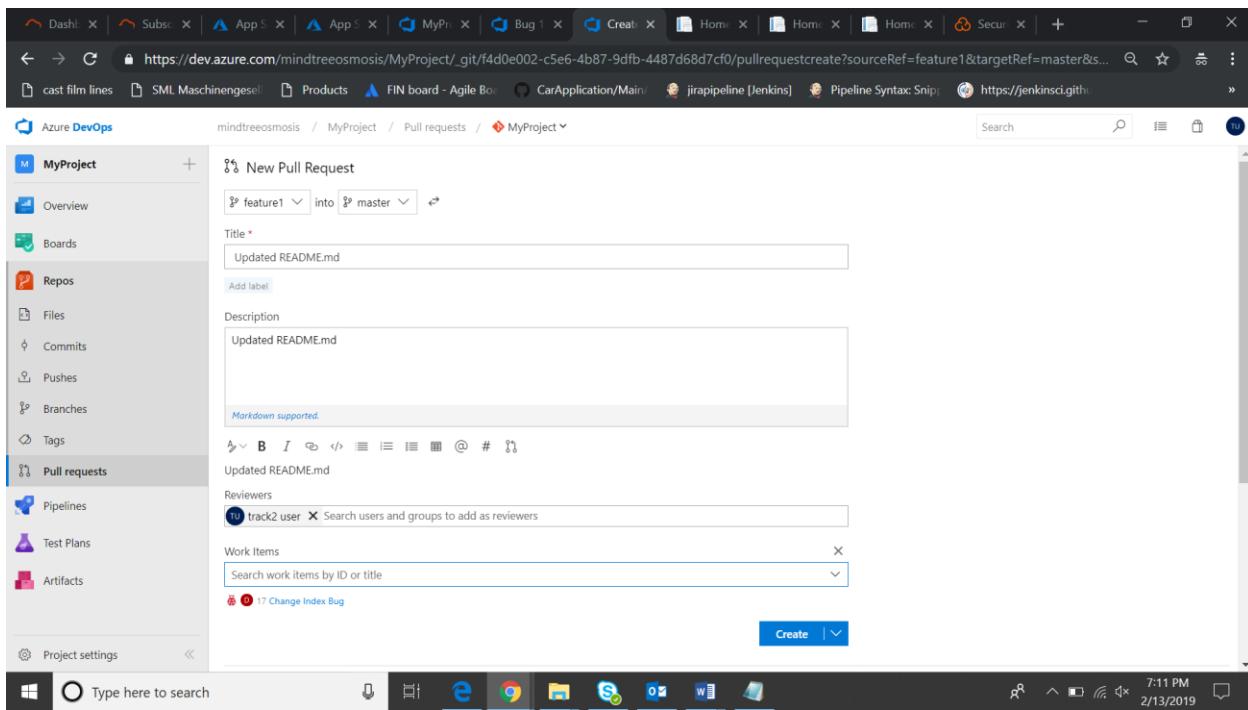
- MyProducts.Tests
- MyProducts.WebApi
- .gitattributes
- .gitignore
- MyProducts.sln
- README.md

Each file entry includes the name, last change date, and a commit message.

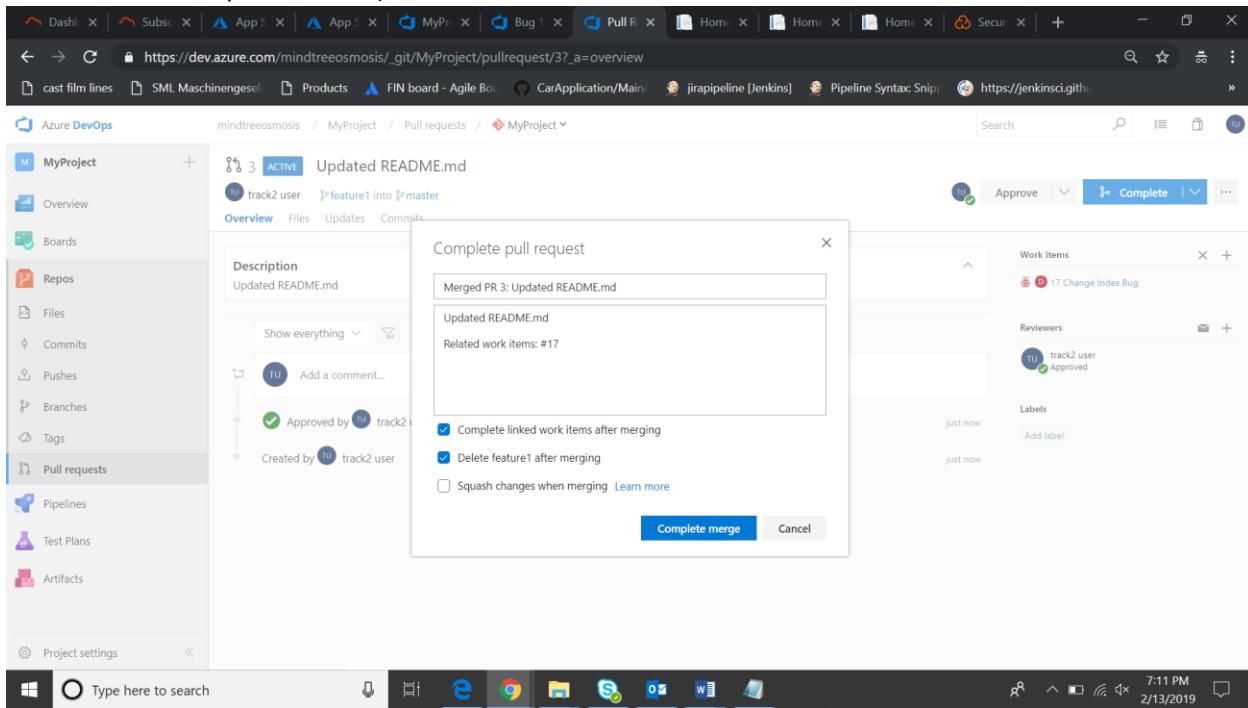
90. Do the necessary changes → Commit.



91. Click on “**Create a pull request**” → Add a title → Add a description → Select Reviewers (The people you want to review your work and give their approval for merging with the master) → Click “**Create**”.



92. The approver can check form his account and Click “**Approve**” if he is satisfied with the work → Click “**Complete**” (to do the final merge with the master) → Click “**Complete Merge**” (This will go ahead and merger your branch “Feature 1” with the master and delete the branch “Feature 1” as it’s purpose has been fulfilled, but you can also choose to keep the branch).



93. The merge has been completed.

94. A new “Build” has been queued automatically as soon as it gets over the releases will also get queued automatically. Deployment to Dev Deployment will happen but deployment to further stages will only happen after approvals.

Thank You