

When Azure Cloud and Azure DevOps are in different login accounts, follow the below steps:

User1 – Azure Cloud Access – Azure App Service for Deployment is created here

User2 – Azure DevOps Portal Access – Azure Pipeline and Release Pipeline are created here

Login to Azure portal with **User1** and go to the subscription details -> Access control (IAM) and add **User2**.

Home > Free Trial

Free Trial | Access control (IAM) ...

Subscription

Search

+ Add Download role assignments Edit columns Refresh Remove Got feedback?

3 4000

Search by name or email Type: All Role: All Scope: All scopes Group by: Role

3 items (1 Users, 2 Service Principals)

Name	Type	Role	Scope	Condition
Contributor				
azure-cli-2022-11-25	App	Contributor	This resource	None
vspradeepk-vspradee	App	Contributor	This resource	None
Owner				
vspradeepk (Guest) vspradeepk@gmail...	User	Owner	This resource	None

Add the Role Assignments for **User2** and save the changes.

Home > appservice123456 | Access control (IAM) > vspradeepk

vspradeepk | Assigned roles ...

User

Diagnose and solve problems

Manage

- Profile
- Assigned roles
- Roles and administrators
- Administrative units
- Groups
- Applications
- License

+ Add assignments Remove assignments Refresh Got feedback?

Administrative roles

Administrative roles can be used to grant access to Azure AD and other Microsoft services. [Learn more](#)

Search by name or description Add filters

Role	Description	Resource Name	Resource Type	Assignment Path
Application Admin	Can create and manage all aspe...	Directory	Organization	Direct
Azure DevOps Ac	Can manage Azure DevOps org...	Directory	Organization	Direct
Global Administr	Can manage all aspects of Azur...	Directory	Organization	Direct
Groups Administ	Members of this role can create...	Directory	Organization	Direct

Accept the invitation received to **User2** email address.

Now we will be able to create resources in Azure Portal with **User2** login.

Login to Azure DevOps with **User2** and create the azure-pipeline.yaml and Release Pipeline.

Release Pipeline:

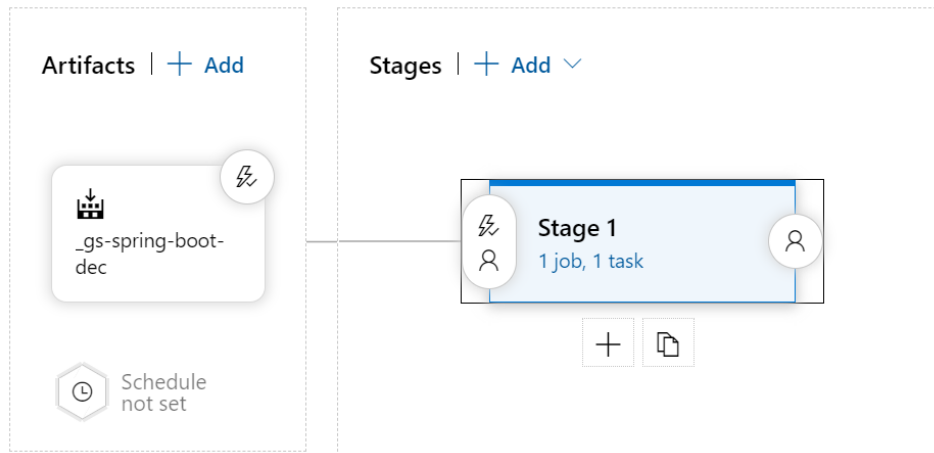
Create a new Release Pipeline and under Artifacts section configure the build pipeline created in the Azure Repos as below:

The screenshot shows the Azure DevOps Release Pipeline configuration interface. The top navigation bar includes 'All pipelines > gs-spring-boot-dec' and buttons for 'Save', 'Create release', and a menu icon. Below the navigation bar are tabs for 'Pipeline', 'Tasks', 'Variables', 'Retention', 'Options', and 'History'. The main area is divided into 'Artifacts' and 'Stages' sections. The 'Artifacts' section shows a build pipeline artifact named '_gs-spring-boot-dec' with a 'Schedule not set' icon. The 'Stages' section shows a stage named 'gs-spring-boot-dec' with a 'Schedule not set' icon. The configuration panel on the right is titled 'Artifact' and shows the following settings: Project: vspradeepk, Source (build pipeline): gs-spring-boot-dec, Default version: Latest, and Source alias: _gs-spring-boot-dec.

If the release pipeline needs to be triggered once the build pipeline is completed, enable continuous deployment trigger.

The screenshot shows the Azure DevOps Release Pipeline configuration interface. The top navigation bar includes 'All pipelines > gs-spring-boot-dec' and buttons for 'Save', 'Create release', and a menu icon. Below the navigation bar are tabs for 'Pipeline', 'Tasks', 'Variables', 'Retention', 'Options', and 'History'. The main area is divided into 'Artifacts' and 'Stages' sections. The 'Artifacts' section shows a build pipeline artifact named '_gs-spring-boot-dec' with a 'Schedule not set' icon. The 'Stages' section shows a stage named 'gs-spring-boot-dec' with a 'Schedule not set' icon. The configuration panel on the right is titled 'Continuous deployment trigger' and shows the following settings: Build: _gs-spring-boot-dec, Enabled: Yes (toggle switch is on), Build branch filters: No filters added. (with a '+ Add' button), and Pull request trigger: Build: gs-spring-boot-dec.

Click Stages and add a new task:



Configure the Agent to run the Release Pipeline:

Pipeline **Tasks** ▾ Variables Retention Options History

Stage 1
Deployment process

Agent job
Run on agent

Agent job ⓘ [Remove](#)

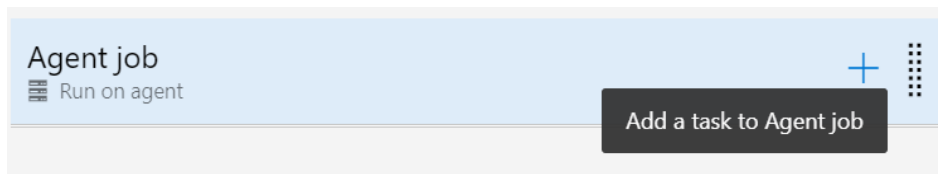
Display name *
Agent job

Agent selection ^

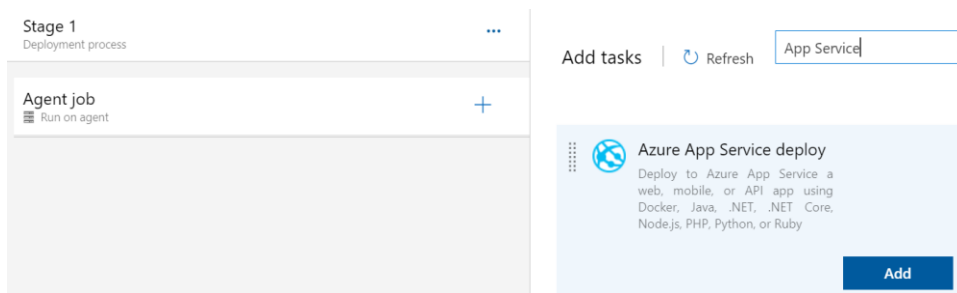
Agent pool ⓘ | [Pool information](#) | [Manage](#) ↗
Azure Pipelines

Agent Specification *
ubuntu-22.04

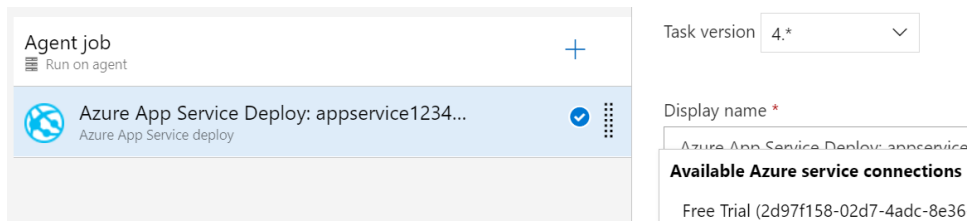
Add a task to Agent to perform deployment to Azure App Service:



Search for Azure App Service Deploy Task and Add:

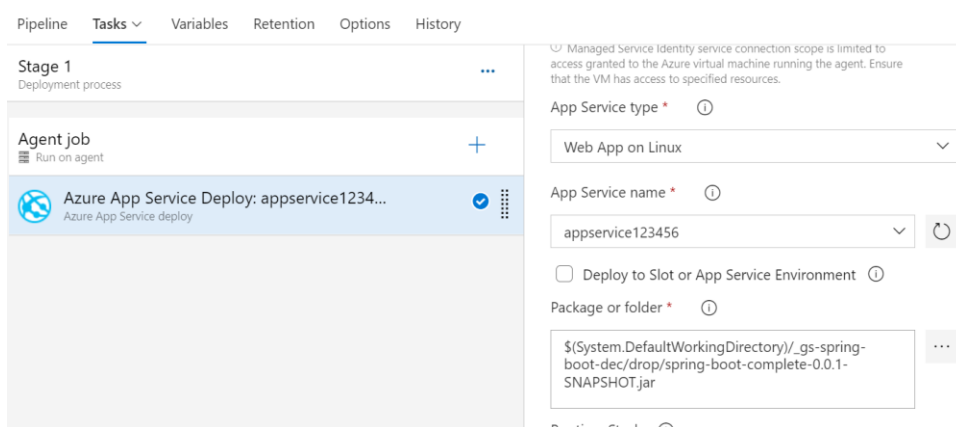


Click on the Azure App Service Deploy Task and under Azure Subscription you will be able to see the Available Azure Service Connection which has the subscription of **User1** in which **User2** was added.



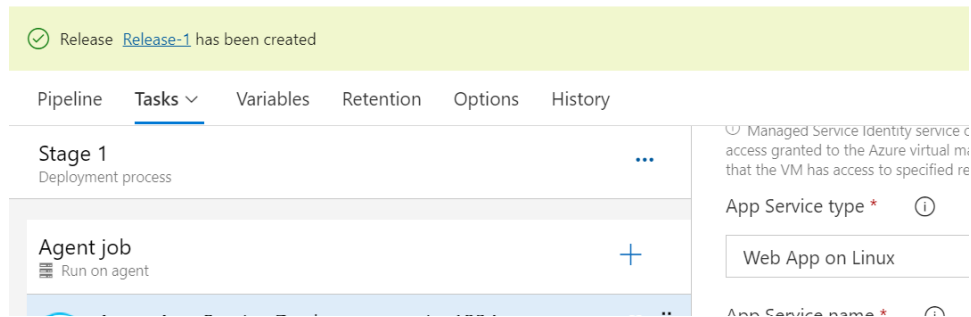
Once you select, you will be able to see the App Services Created under the subscription.

Choose the App Service and under Package choose the artifact to be deployed.

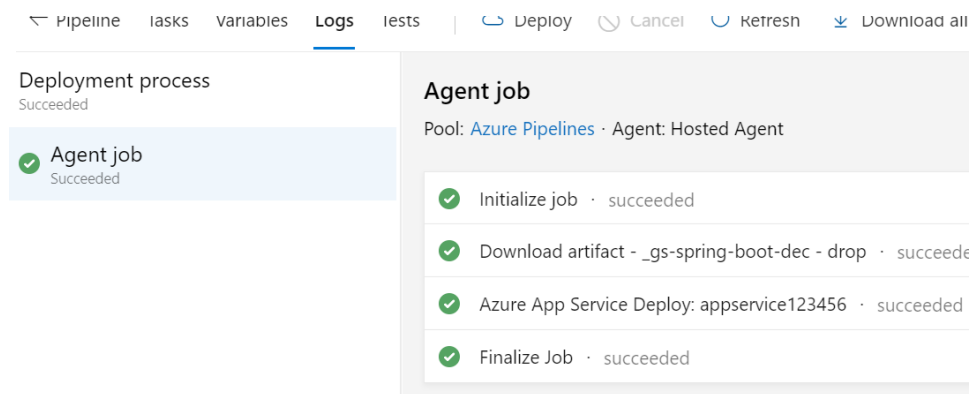


Save the pipeline and click Create Release for manual deployment to App Service.

You can review the pipeline stages, artifacts and then click Create.



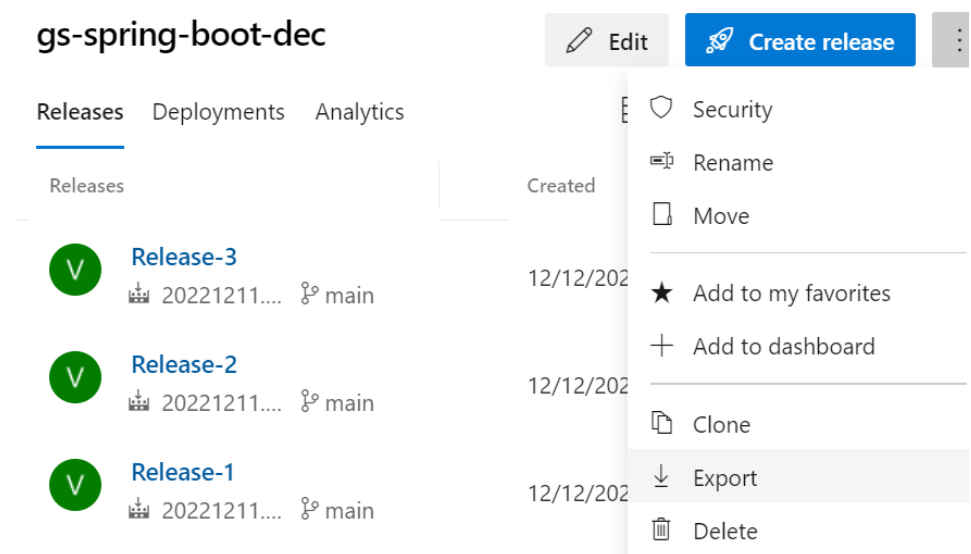
Click on Release-1 and you will see the execution output.



Once the pipeline is successful, you can click the App Service URL and see the result.

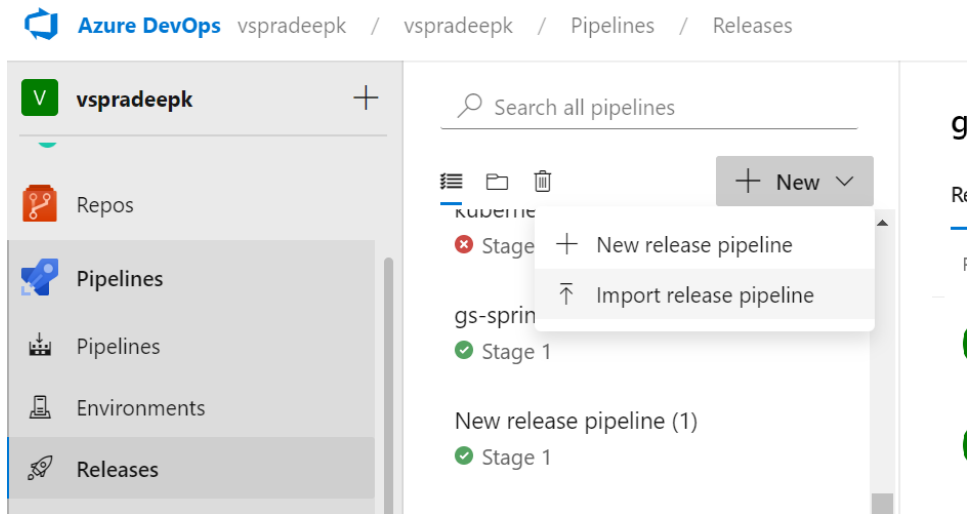
Pipeline:

Release pipeline can be exported to file as below:



Release pipeline can be imported to Azure DevOps as below:

Go to Releases -> New -> Import Release pipeline



Azure Pipeline files can be found in the below link:

[Azure DevOps Pipeline Yaml Files](#)