**Deploying of Terraform code using Azure DevOps CI/CD pipeline**

**(Using Self Hosted Agent machine)**

Here we build the infrastructure in the azure cloud faster & effcetively by deploying the terraform code in then Azure pipelines.

And here we create our own Agent machine (self hosted agent machine) manually in azure portal, to perform all the related tasks/jobs in it, like Terraform install, init, validate, plan, and apply in the Release.

This Self hosted agent machine is under our control, we have to maintain and configure it, in order to perform CI/CD pipelines more effectively.

Let’s perform the above process practically.

Step1: Create a separate organization (my-second-organization02) and within it create a project (my-second-project).

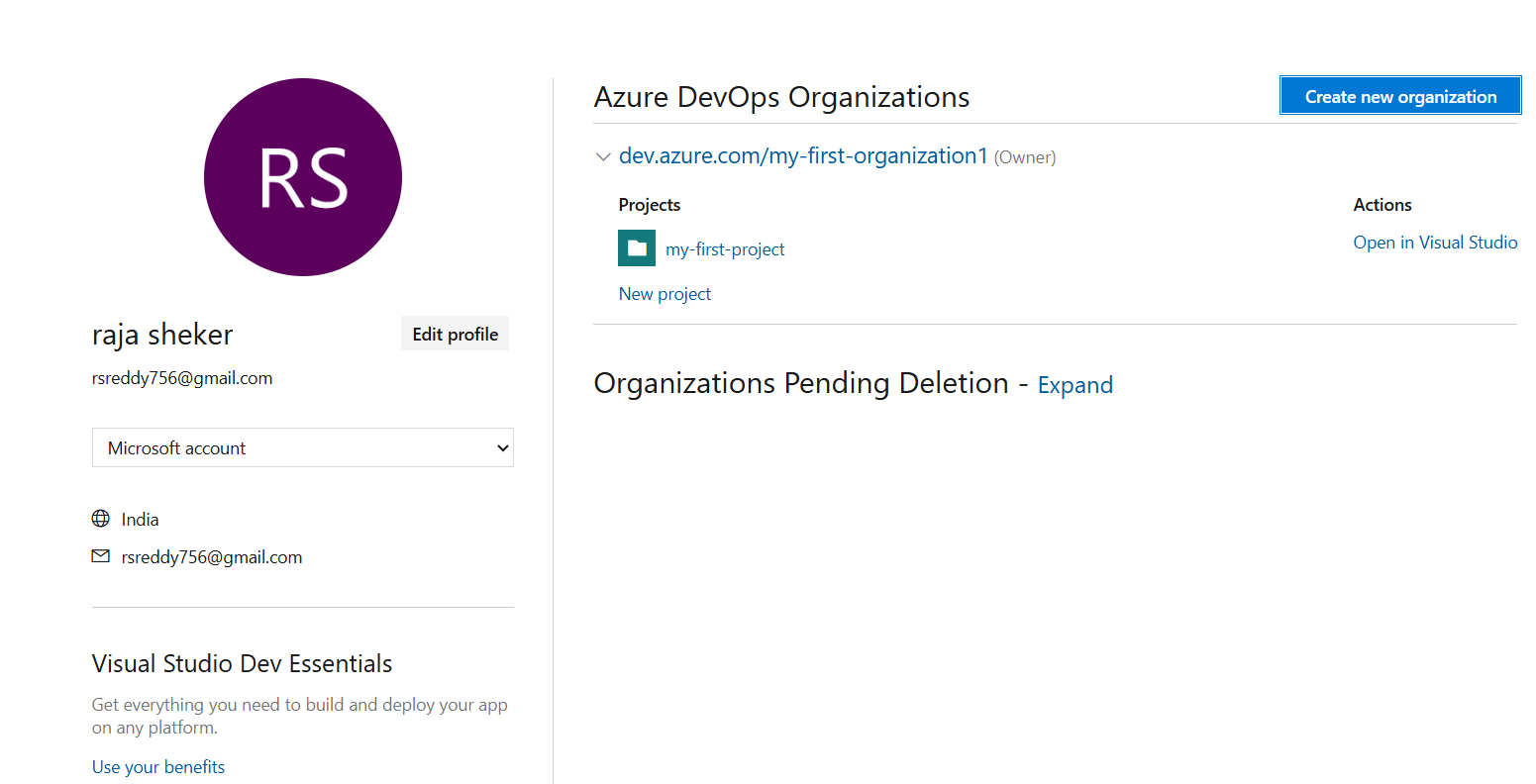


Fig: Click on “**Create New Organization”.**

Creation of New organization with name (my-second-organization) and project name (my-second-project).

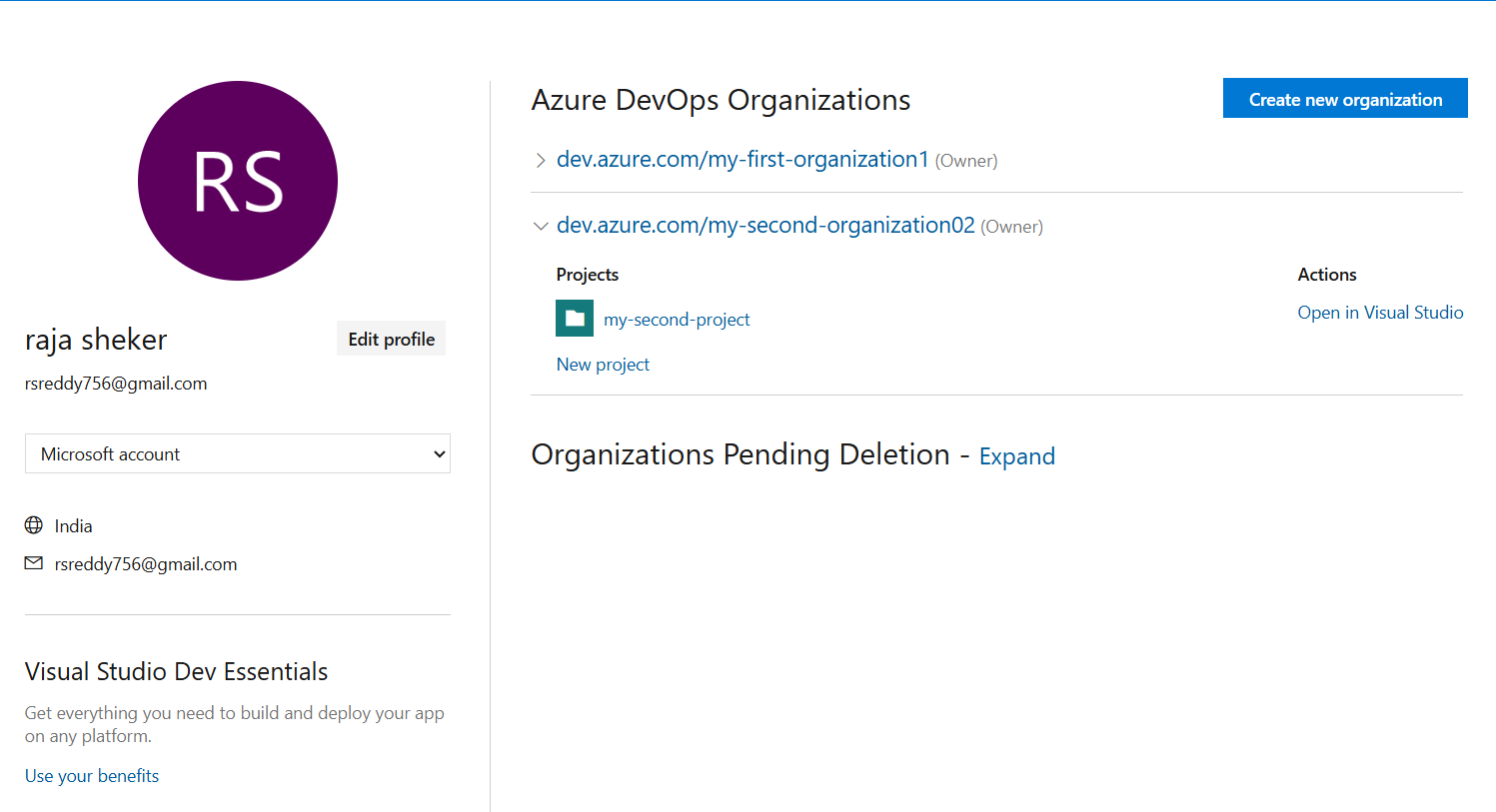


Fig: Organization (my-second-organization) with single project (my-second-project).

**Step2:** Push the Terraform code (VM creation) to the Azure repository form local machine.

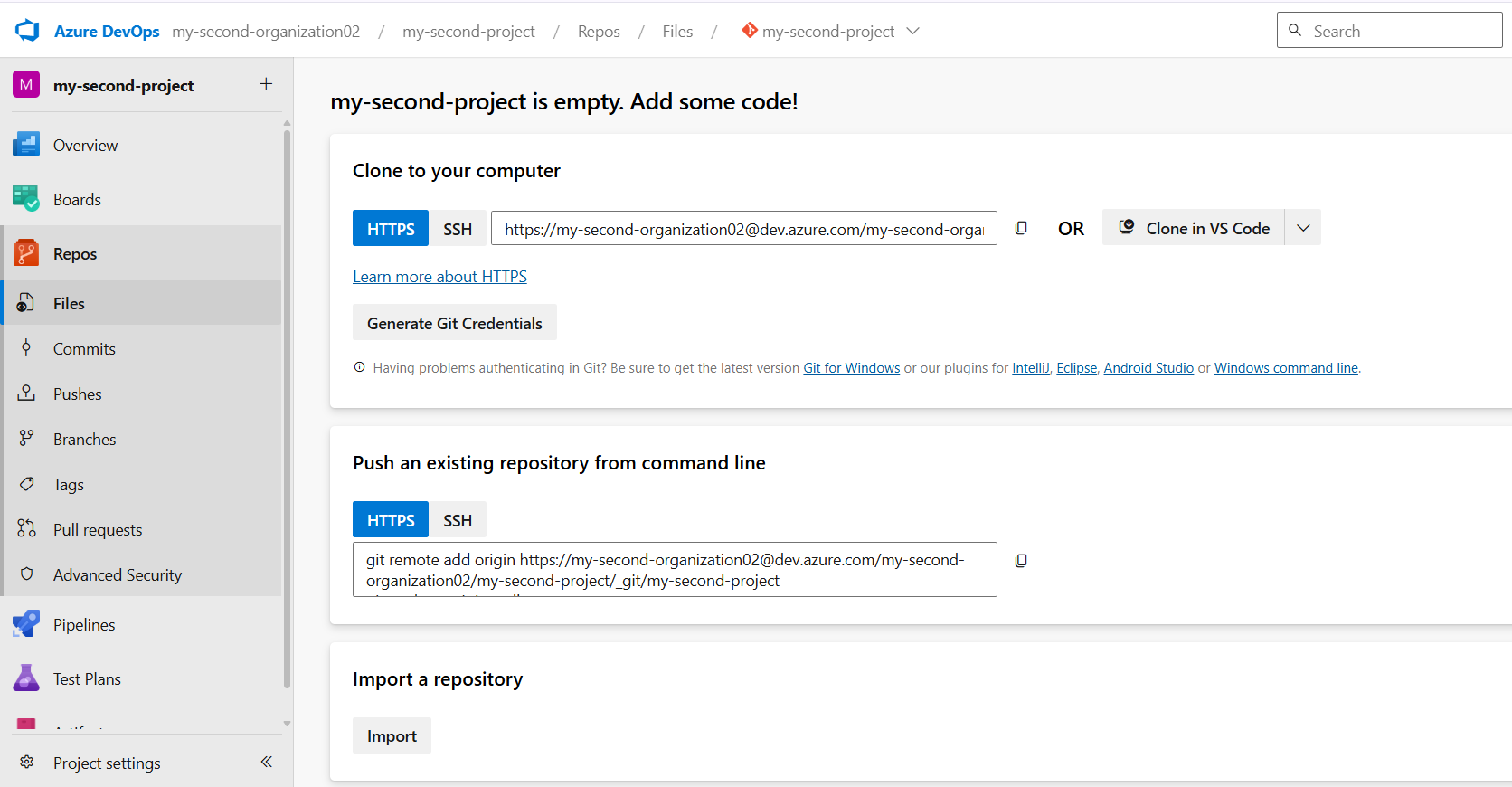
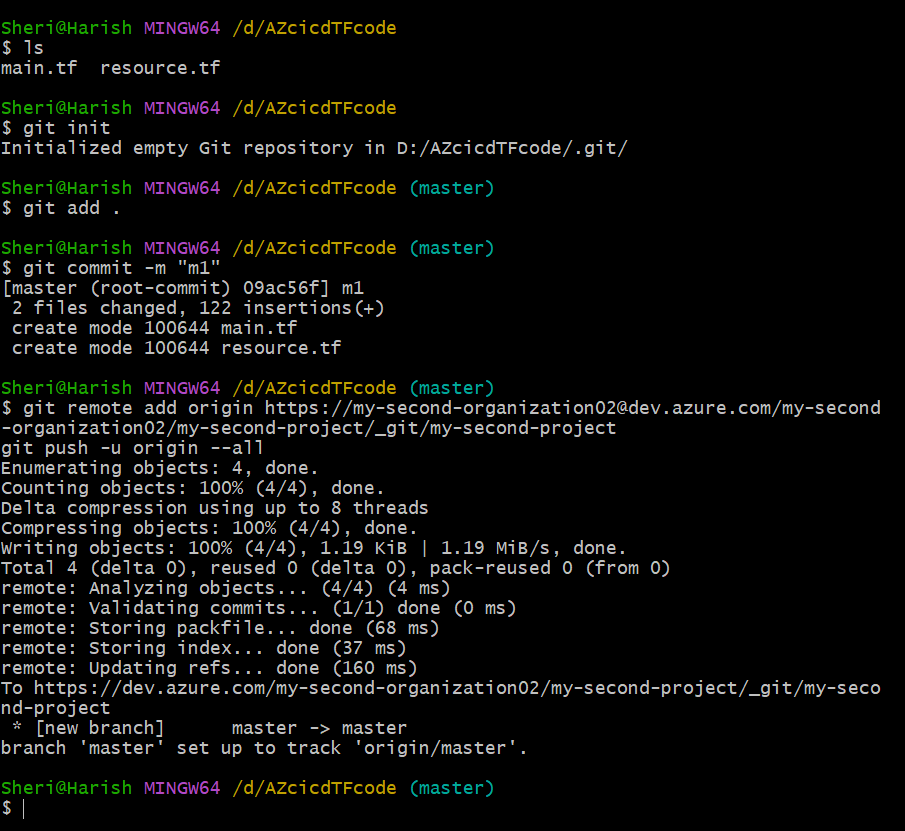


Fig: Azure repository within the project (my-second-project).



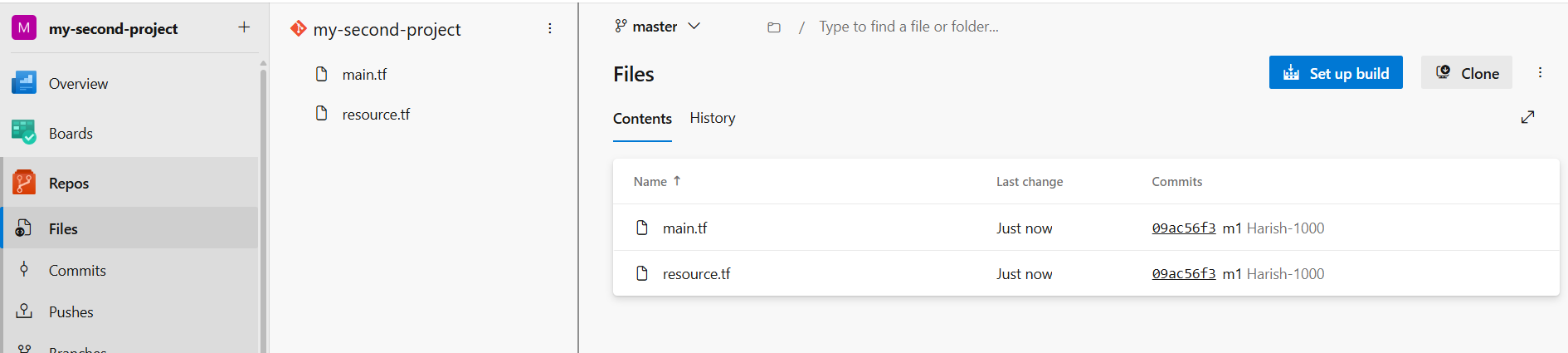
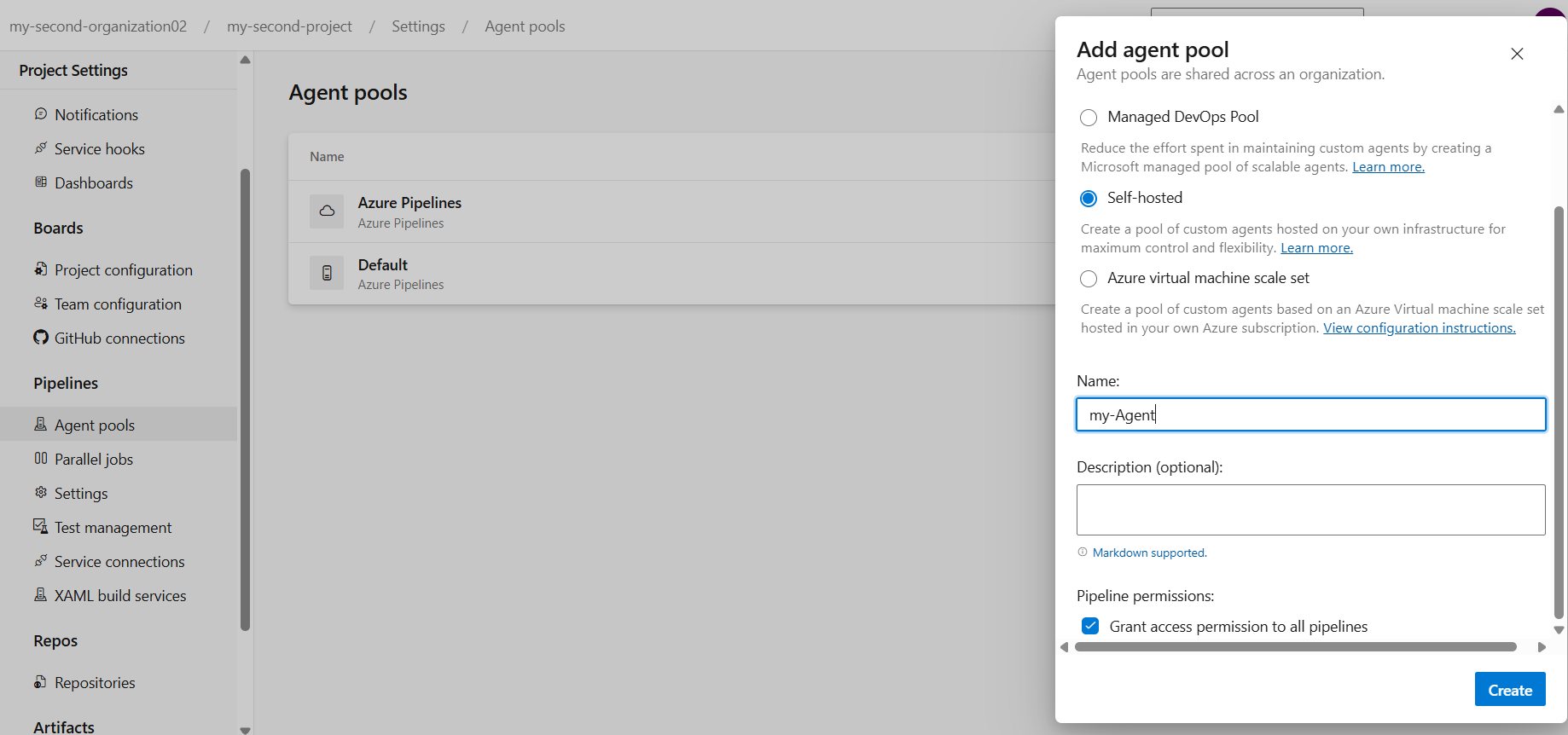


Fig: Local Terraform code files are pushed successfully to the Azure repository (my-second-project).

**Step3:** Create a Virtual machine manually in azure portal which is used as self hosted Agent machine.

Before adding self hosted Agent machine to the Azure pipelines we have to create an Agent pool and then configure it. For This

**Go To🡪 project settings🡪Agent pool🡪Add pool🡪Self-hosted🡪Configure Name🡪enable “Grant access permission to all pipelines”.**

****

**Fig:** Creating of Agent Pool.

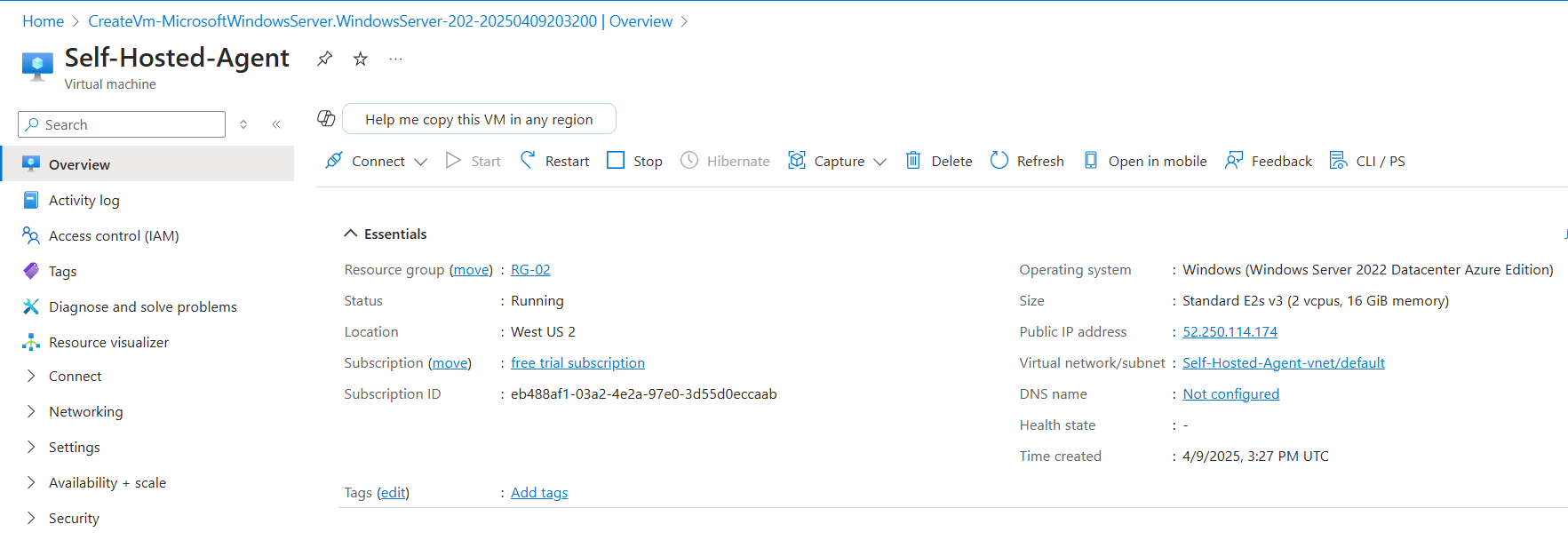
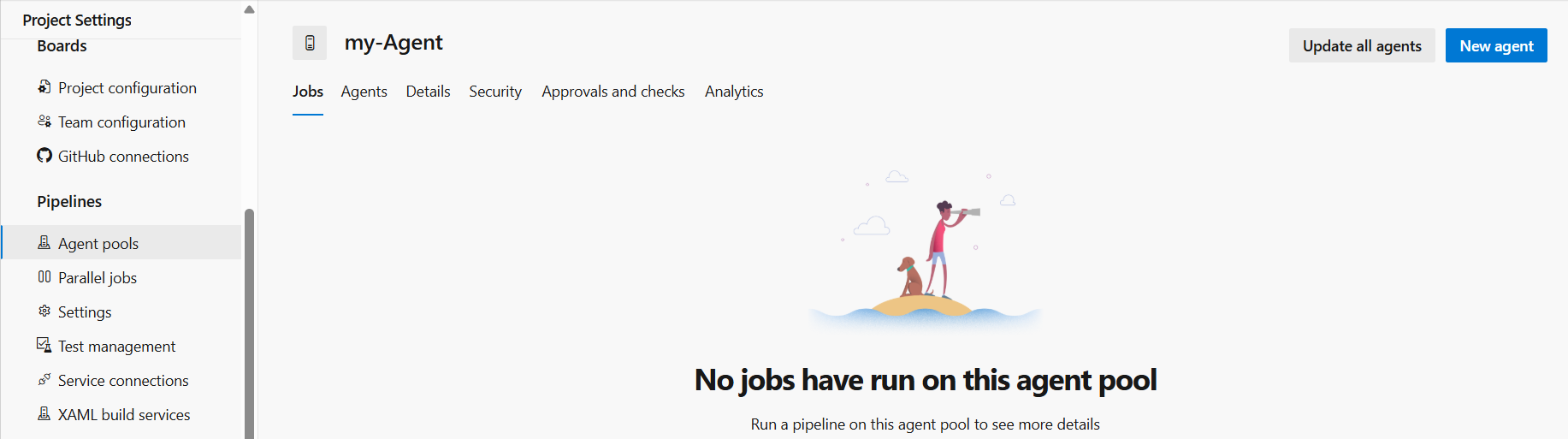


Fig: Windows virtual machine is created successfully.

This windows machine can be used a self-hosted agent machine by configure some of instruction in order work effectively with azure pipelines.

The instruction are given in below figure.



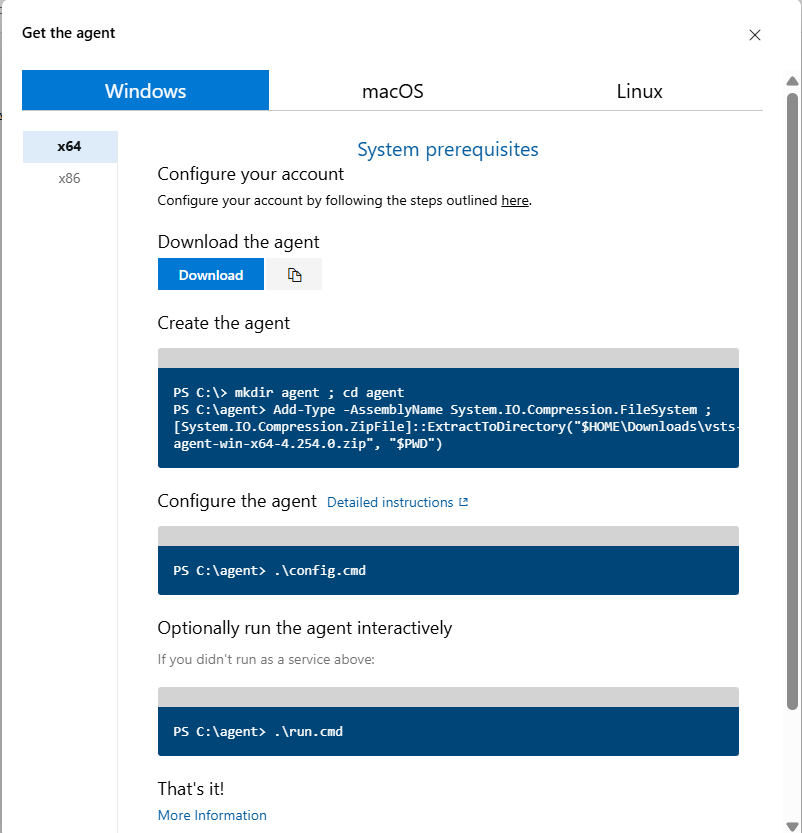
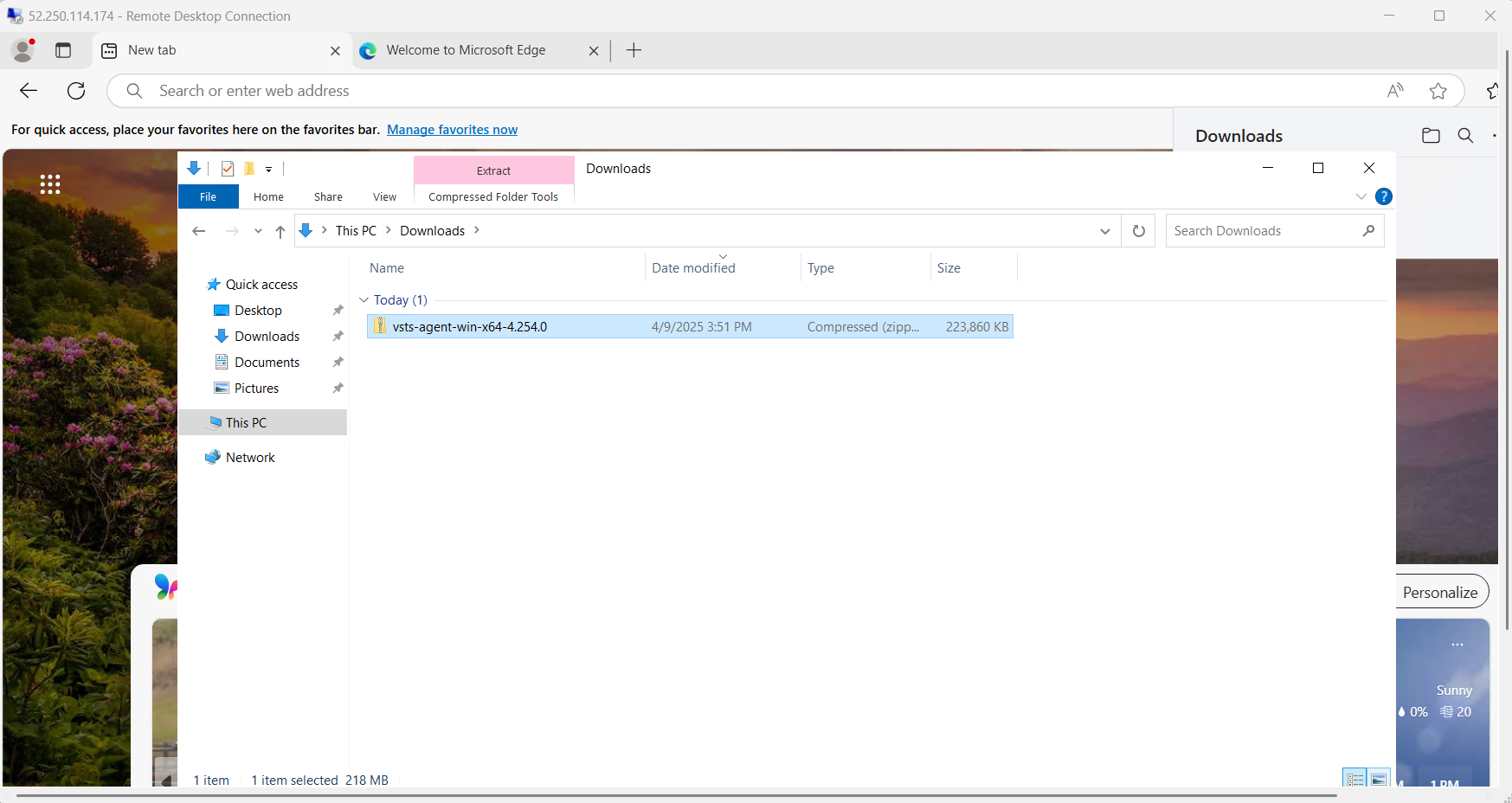


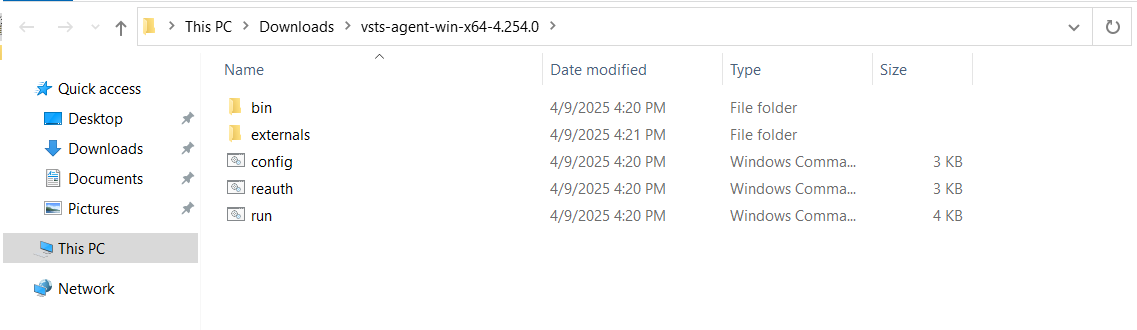
Fig: The instruction which are used to configure for self hosted windows agent machine.

**Step4:** Configure the above instruction in the Windows machine by login into it.

**Case1:** Copy the Download URL and brows in the windows agent machine of Microsoft edge and download the Zip file.



**Case2:** Now extract the Downloaded Zip file.



**Case3:** Create a Directory with name **“agent”** and copy all the files & folders of extracted folder to the agent directory.

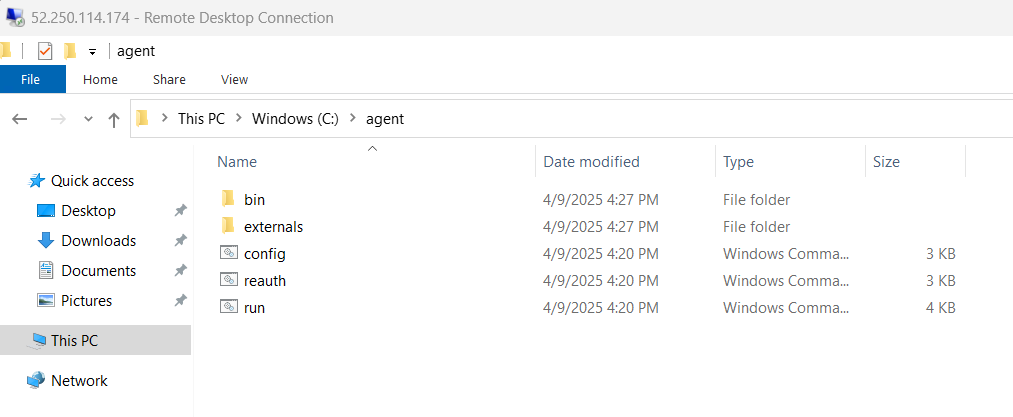


Fig: Extracted files & folders are copied to the agent folder.

In above case 1, 2, 3 are done manually instead of doing it through CLI (PowerShell) as show in below figure is CLI command.

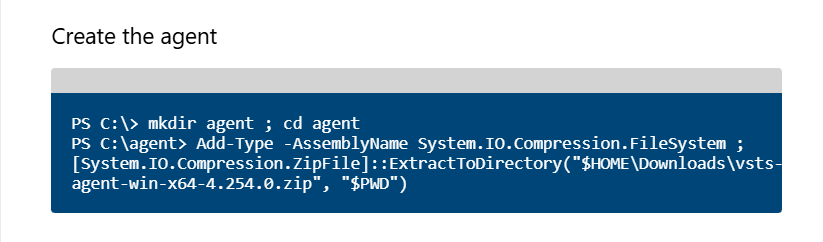
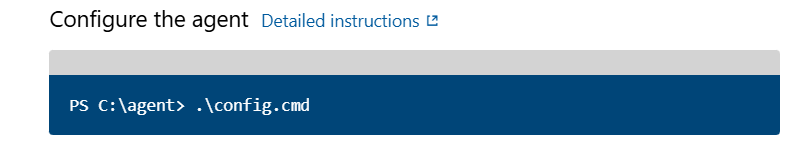
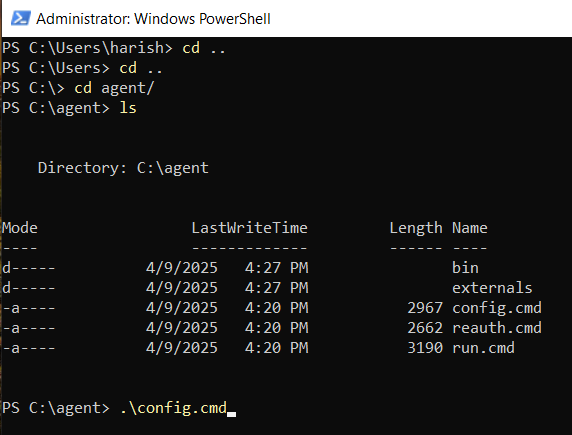


Fig: Creation of agent directory and extracting the Zip folder within the agent directory using PowerShell. (We can do it manual also creating of agent directory and copying the extracted files in it.)

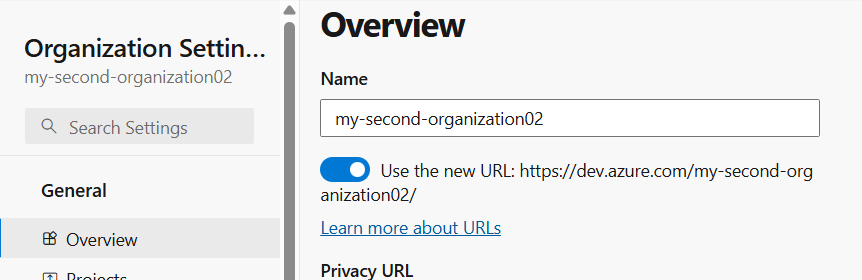
**Case4:** Configure the agent using the command **“.\config.cmd”** in PowerShell of Windows VM.





After pressing enter it will asks for server URL. For server URL

Go TO 🡪 organization settings🡪Overview🡪copy the URL.

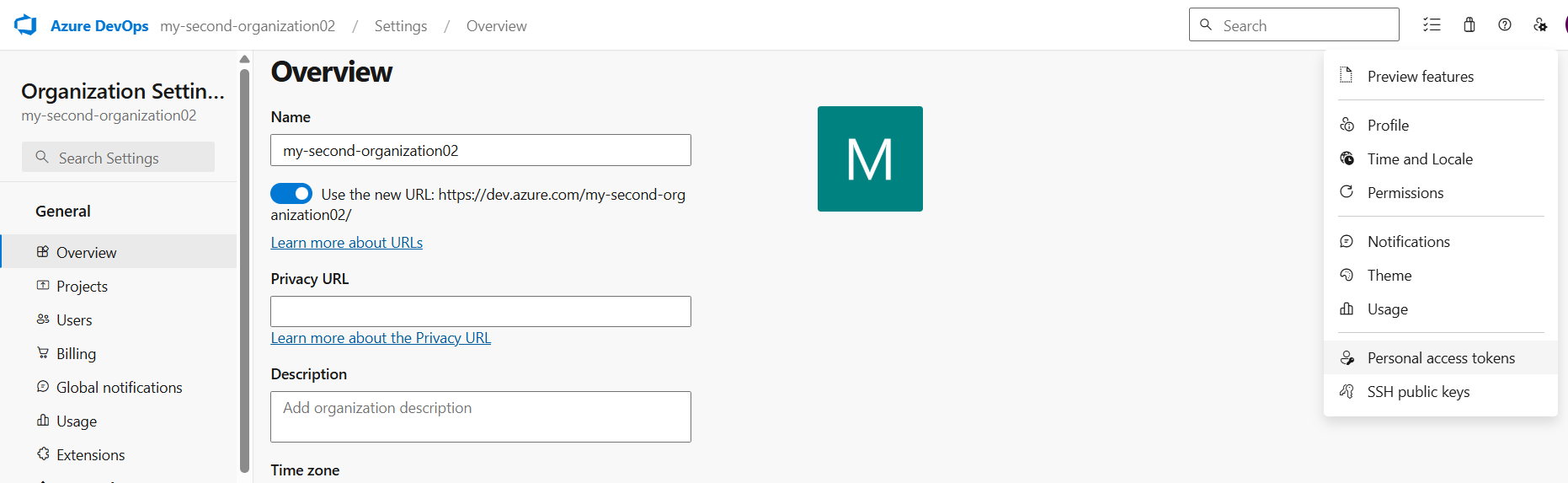


**Fig:** Server URL.



After pressing enter then it will asks for the personal access token. For PAT

Go To 🡪organization settings🡪User settings🡪Personal access tokens🡪New Token🡪create.



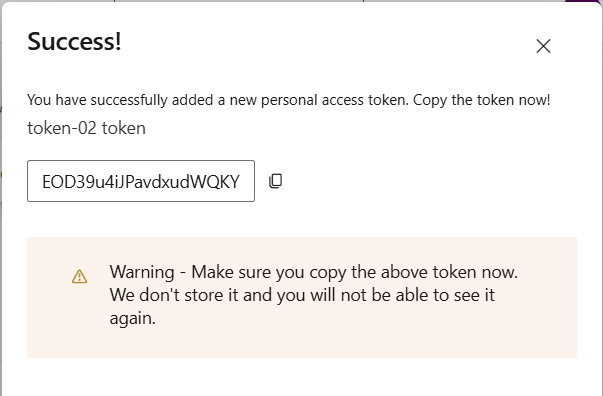
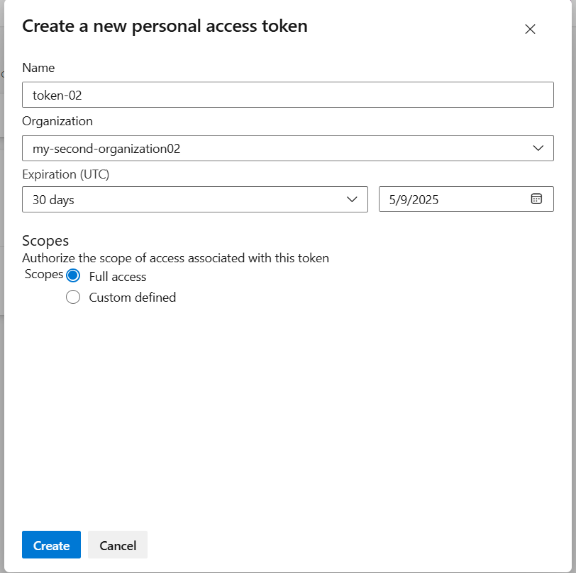
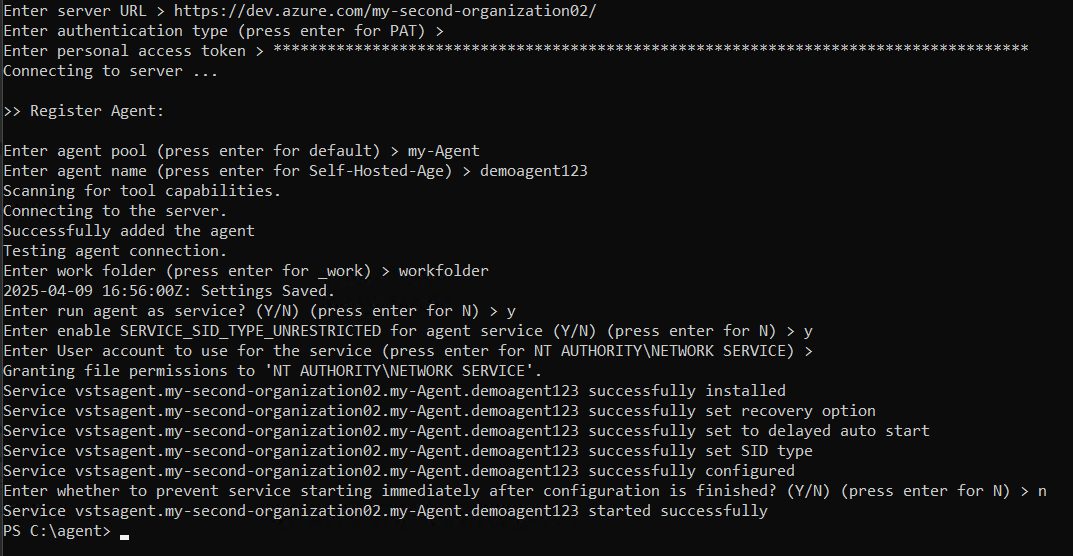


Fig: Creation of Token Fig: Copy the Token.



In above figure after entering the Server URL press enter, then it will asks for the personal access token (PAT) and create the token and past it and press enter,

Then it will again ask for the Agent pool name (**my-agent**) which is created under the organization (my-second-organization02) and enter the name and press enter,

Then it will asks to, type the Agent Name, we can give any name **(demoagent123)** to it after that it will connect to the server and test the agent machine (Windows VM).

Then it will asks for the work folder, give any folder name (workfolder).

Enter run agent as service? (Y/N) (press enter for N) > y

Enter enable SERVICE\_SID\_TYPE\_UNRESTRICTED for agent service (Y/N) (press enter for N) > y

Enter User account to use for the service (press enter for NT AUTHORITY\NETWORK SERVICE) > (press Enter)

Enter whether to prevent service starting immediately after configuration is finished? (Y/N) (press enter for N) > n

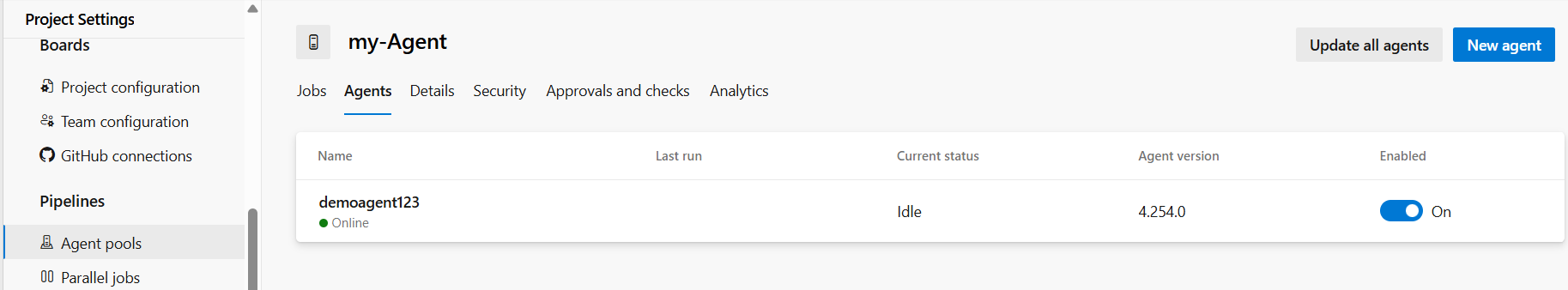


Fig: Self hosted Windows Agent machine is up and running.

By using the above self hosted agent machine (demoagent123) we can run or execute the Agent jobs/tasks in it using CI/CD pipelines.

**Note1:** While working with the self hosted agent we no need to rise the parallelism request for CI/CD pipeline.

**Note2:** For any application self hosted agent machine is recommended in real world.

**Step5:** Now perform the continuous integration (CI) using self hosted agent machine.

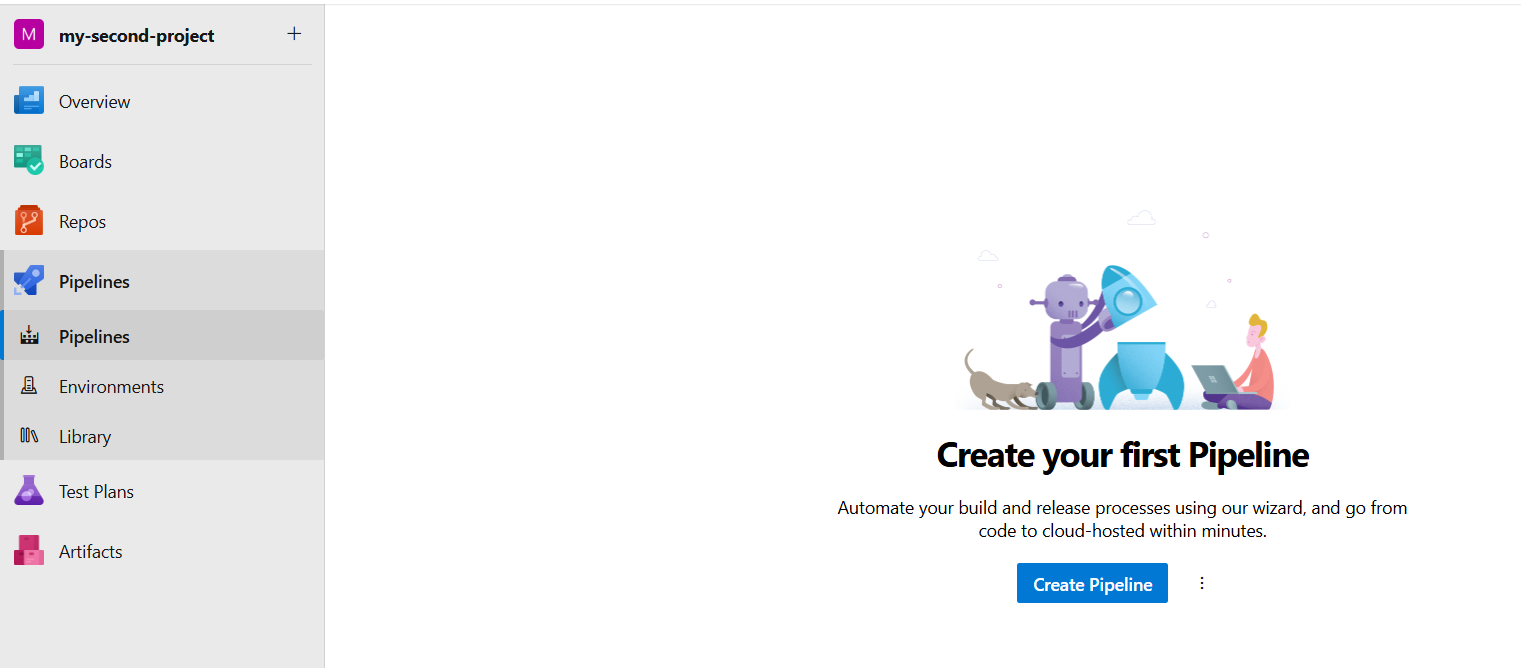


Fig: Creation of pipeline.

**Note:** Enable the classic editor. The "classic editor" in Azure DevOps primarily refers to the **user interface for creating and managing build and release pipelines without writing YAML code.** While YAML pipelines are now the recommended and default approach.

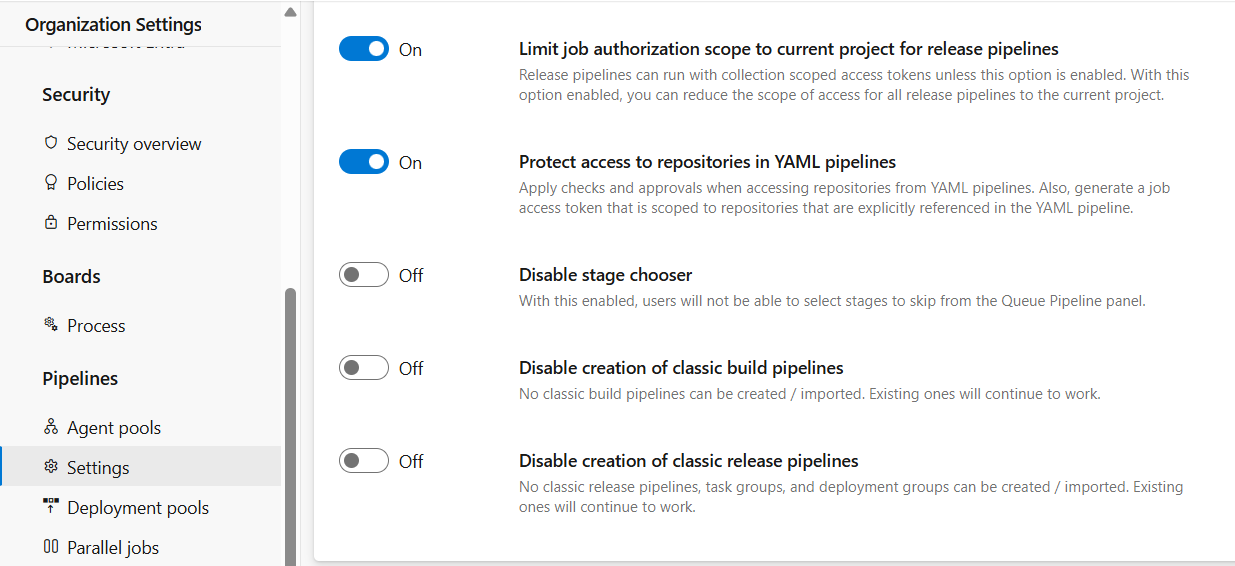


Fig: Enabling of classic editor to build and release pipeline without writing YAML file.

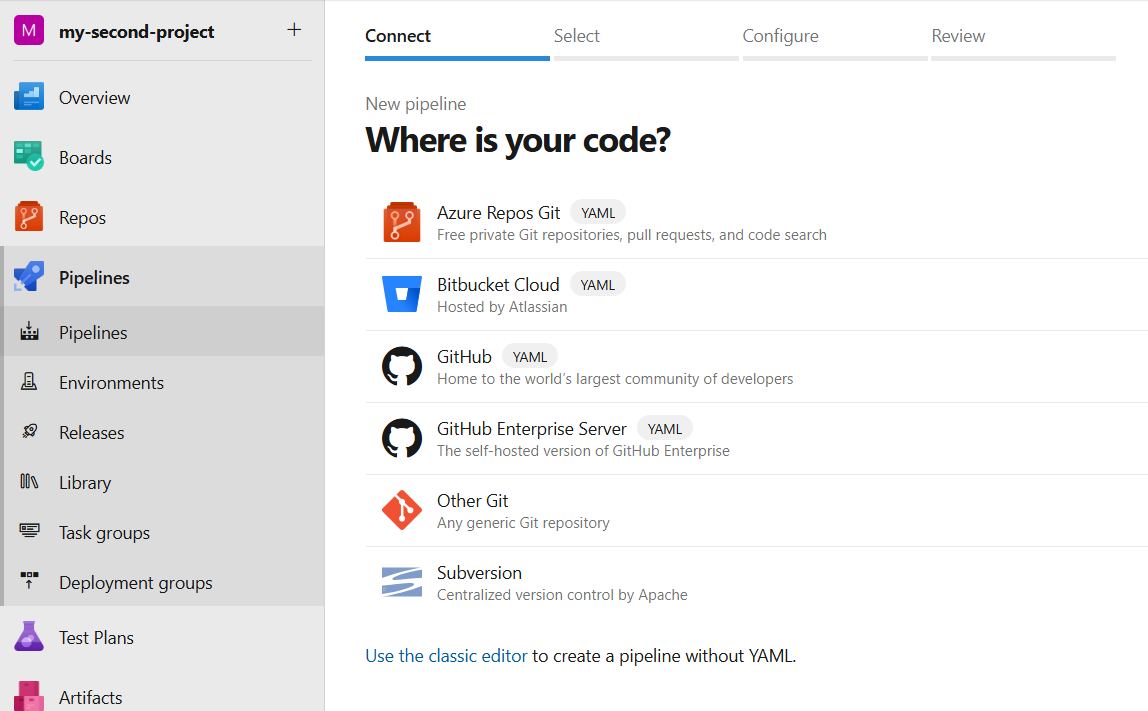


Fig: Classic editor is enabled successfully.

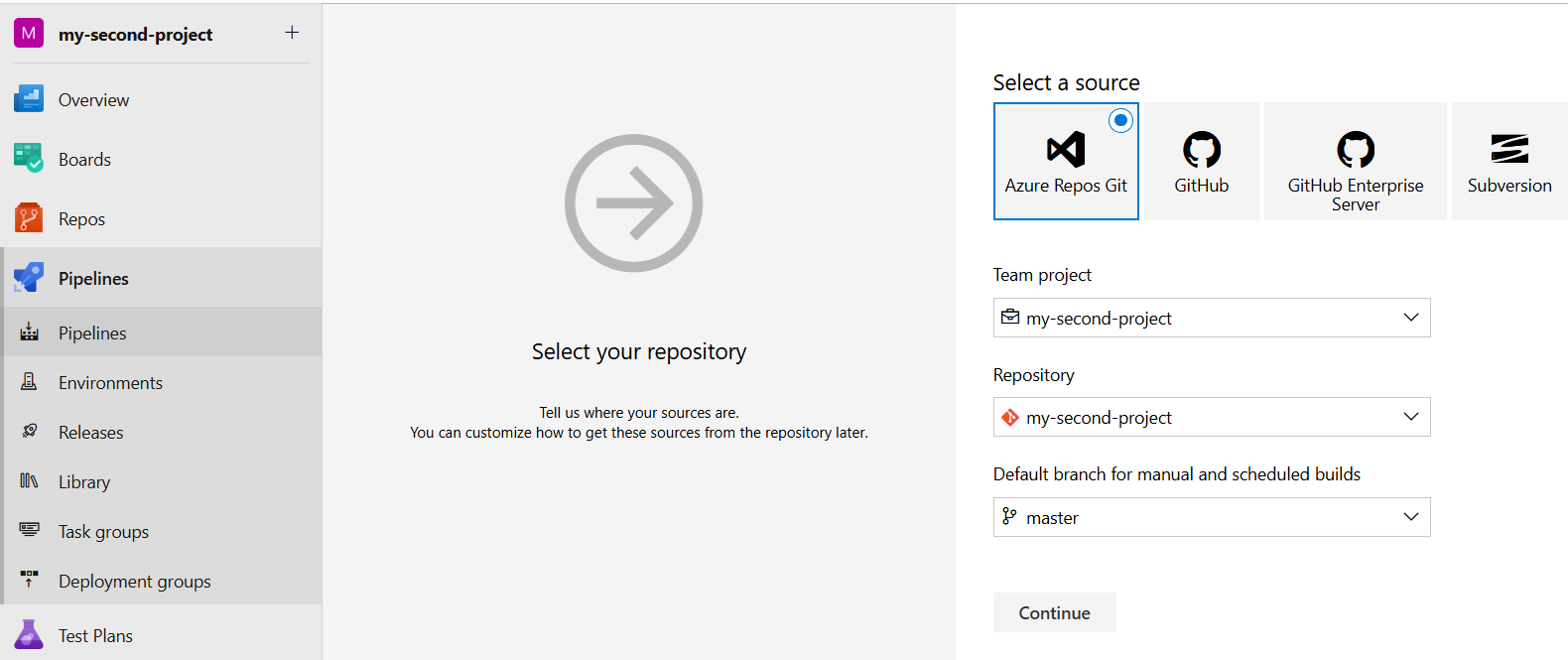


Fig: Selecting of Our application repository.

After clicking on the “**Continue”** then again click on the **“Empty job”** and then select our agent pool **(my-agent).**

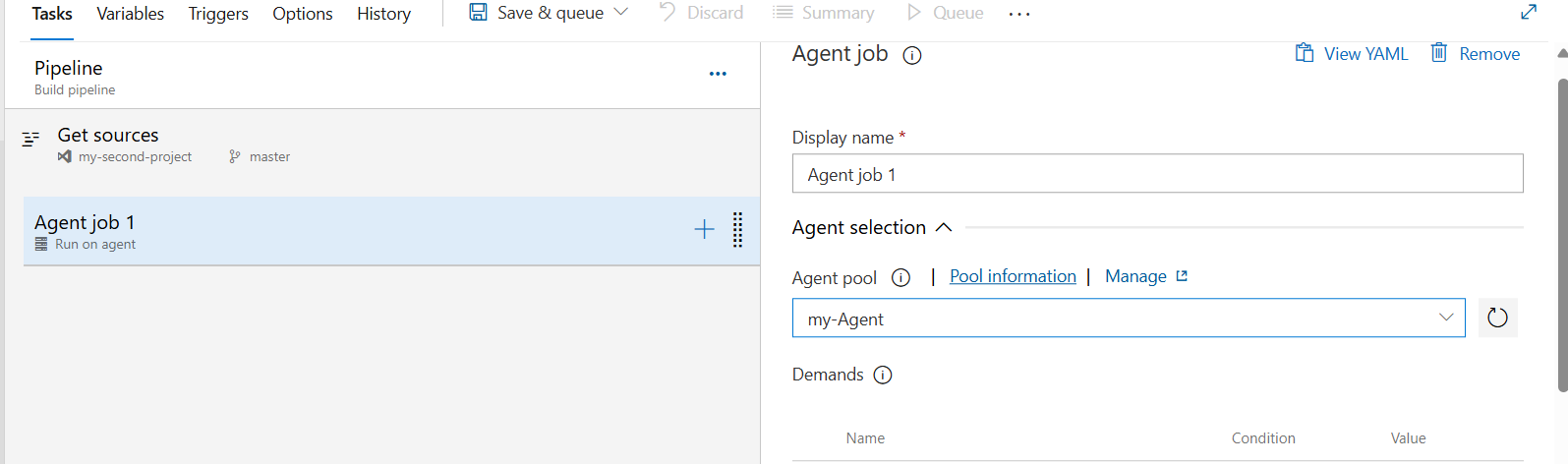


Fig: Adding of self hosted agent machine pool (my-agent).

Now add the Tasks in CI like Copy files to the source folder (**workfolder\1\s)** and from source folder to the artifacts folder $(build.artifactstagingdirectory)**=(workfolder\1\a).** And **publish the artifacts.**

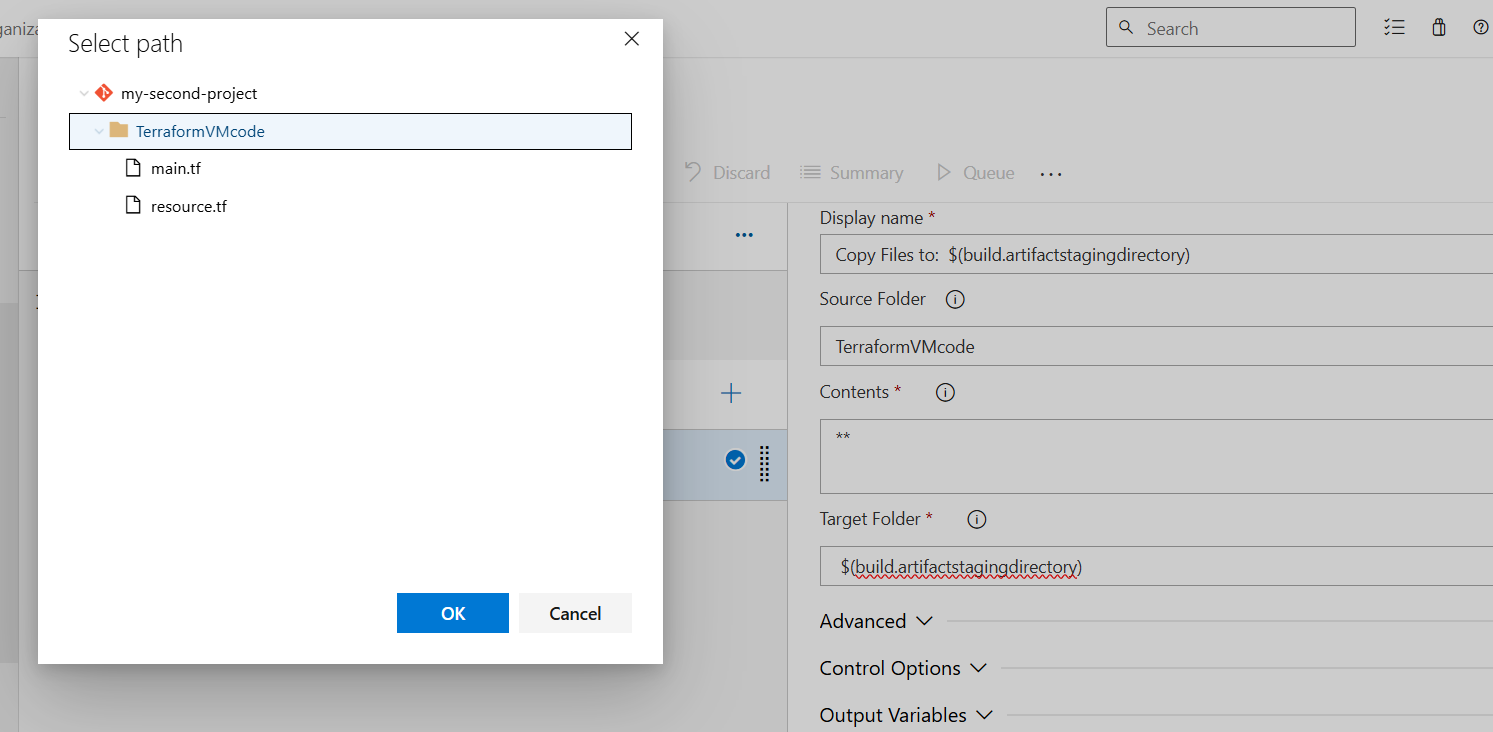
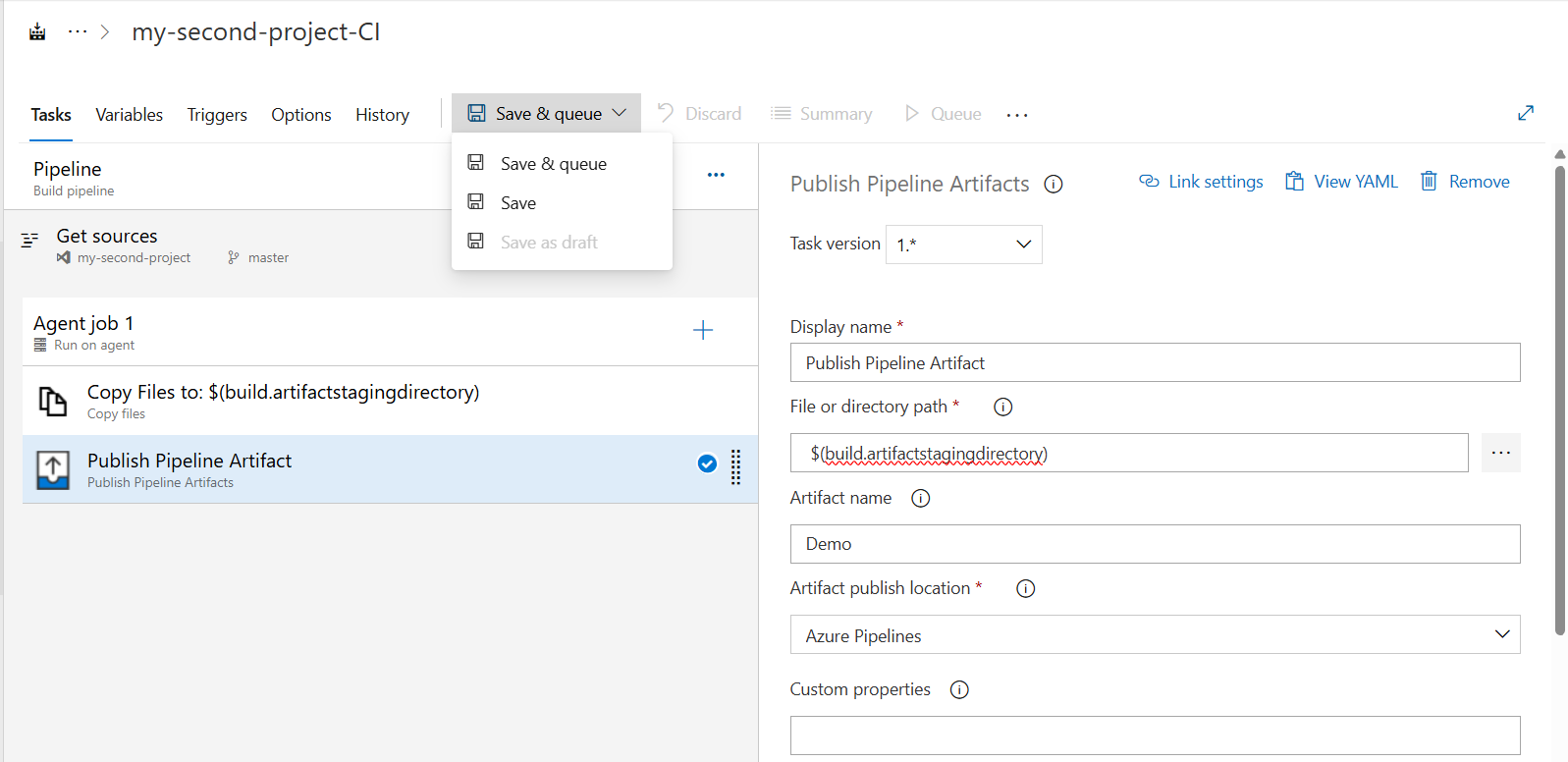


Fig: Copying of files from source to artifacts folder.

Artifacts directory

“\*\*” Copy all files

This tasks says that it copy’s all the files and folder from the source directory (TerraformVMcode) (workfolder\1\s) to the artifacts directory $(build.artifactstagingdirectory) (workfolder\1\a).



**Fig:** Adding task to publish the artifacts.

After publishing the artifacts do **Save & queue,** and **save and run** it. Then see what happened in the **workfolder** which is created in the agent directory of Windows self hosted agent machine (demoagent123).

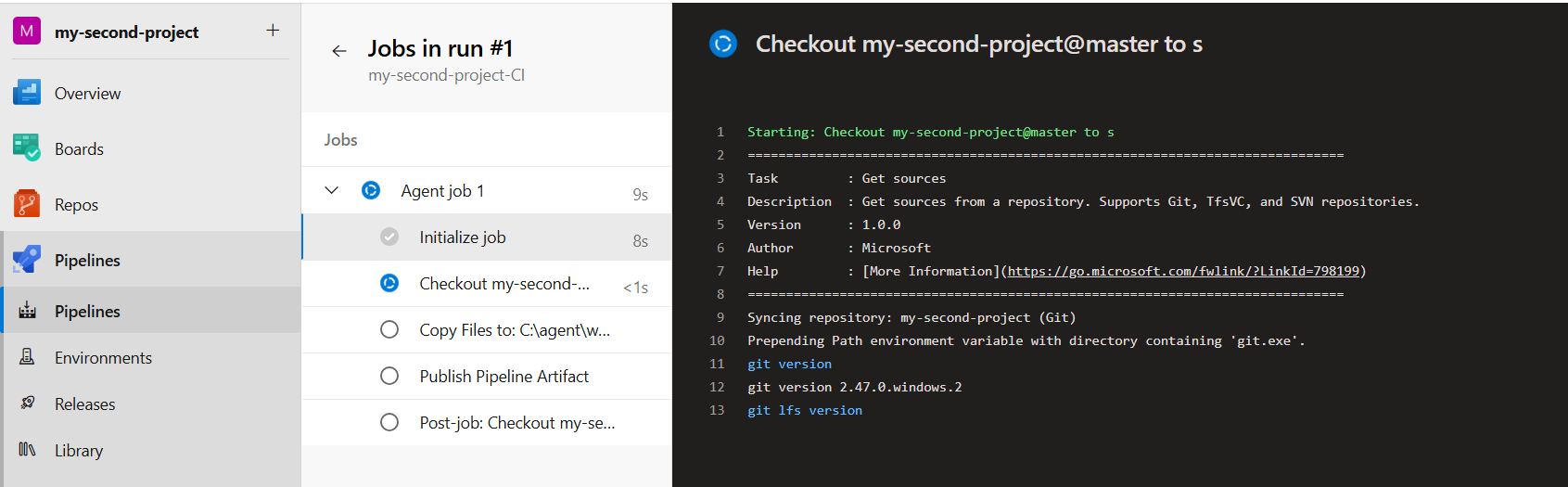


Fig: Pipeline is executing.

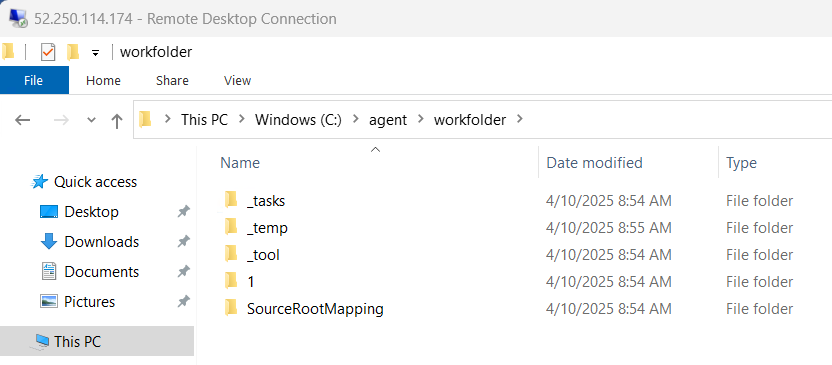
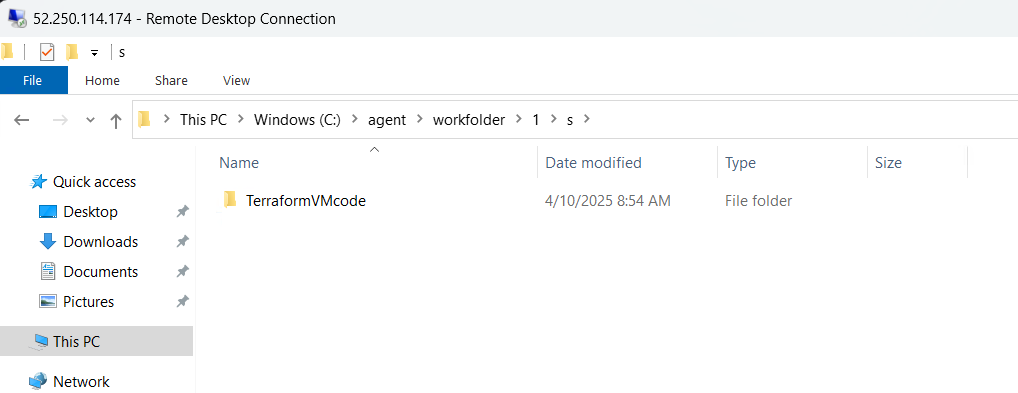


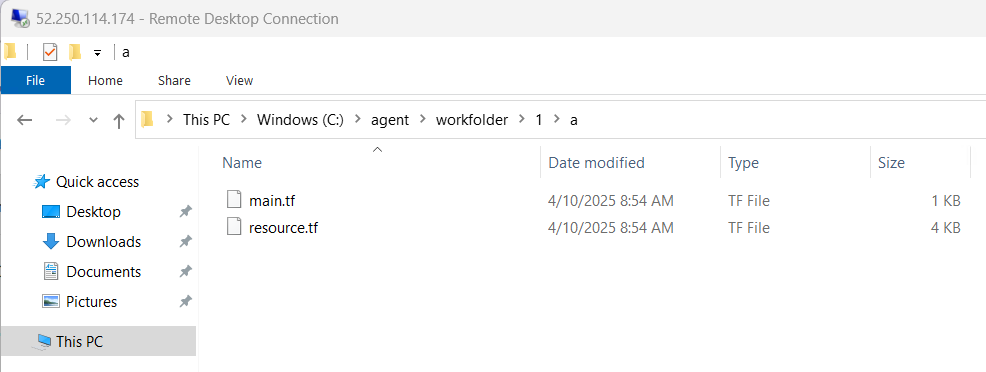
Fig: Workfilder.

Automatically the above folders like \_tasks, \_temp, \_tool, 1, and sourceRootMapping are created.

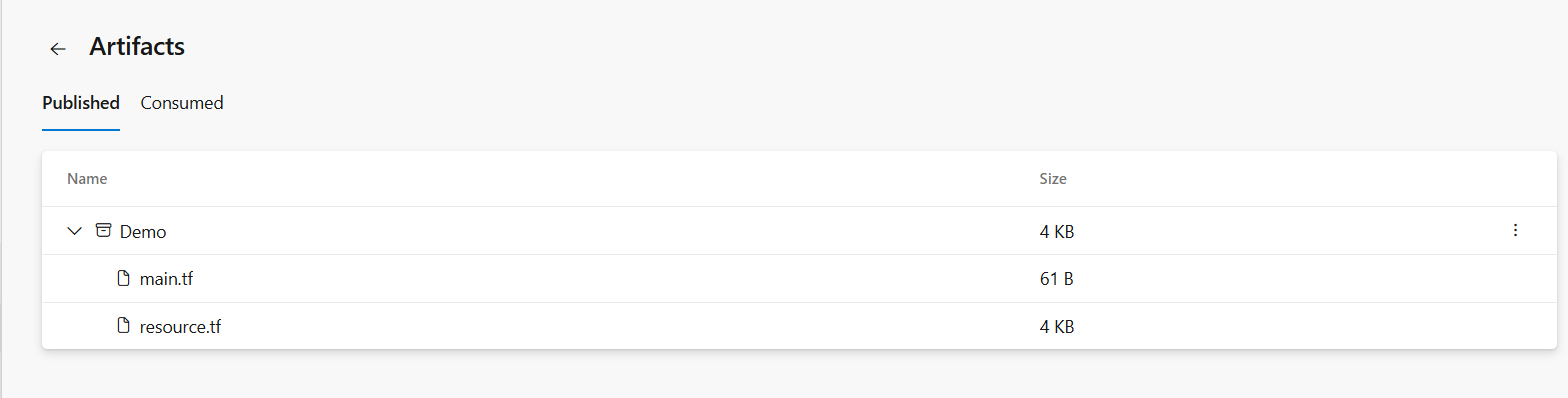
Now go to🡪 “1” Folder🡪 “s” Folder (Here the terraform code all files are copied as shown below figure).

**Fig:** Source folder.

Again the Terraform code related files (main.tf and resource.tf) files are copied successfully from source folder (s) to the artifacts folder (a) as shown in below figure.



**Fig:** artifacts folder.



**Fig:** artifacts are published successfully.

**Step6:** Now do the Continuous delivery/deployment (Release).

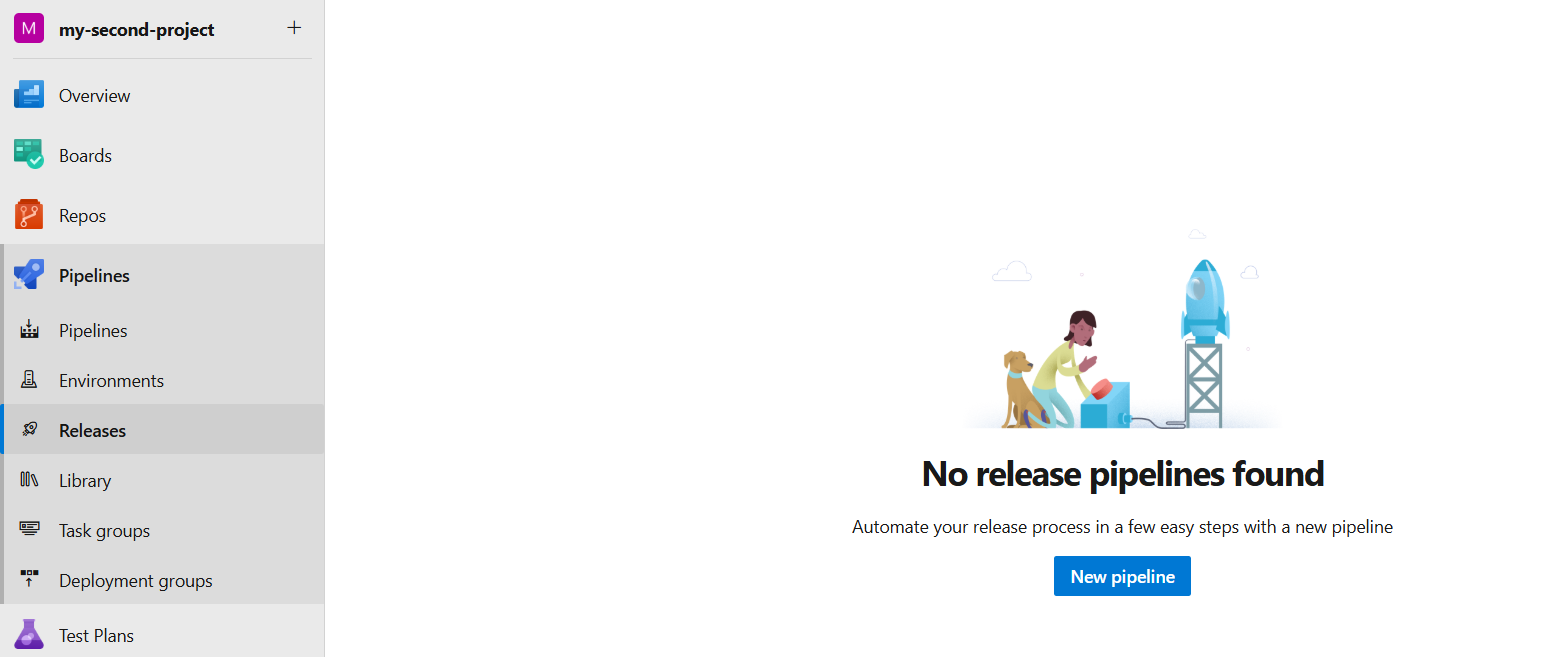


Fig: Release (CD).

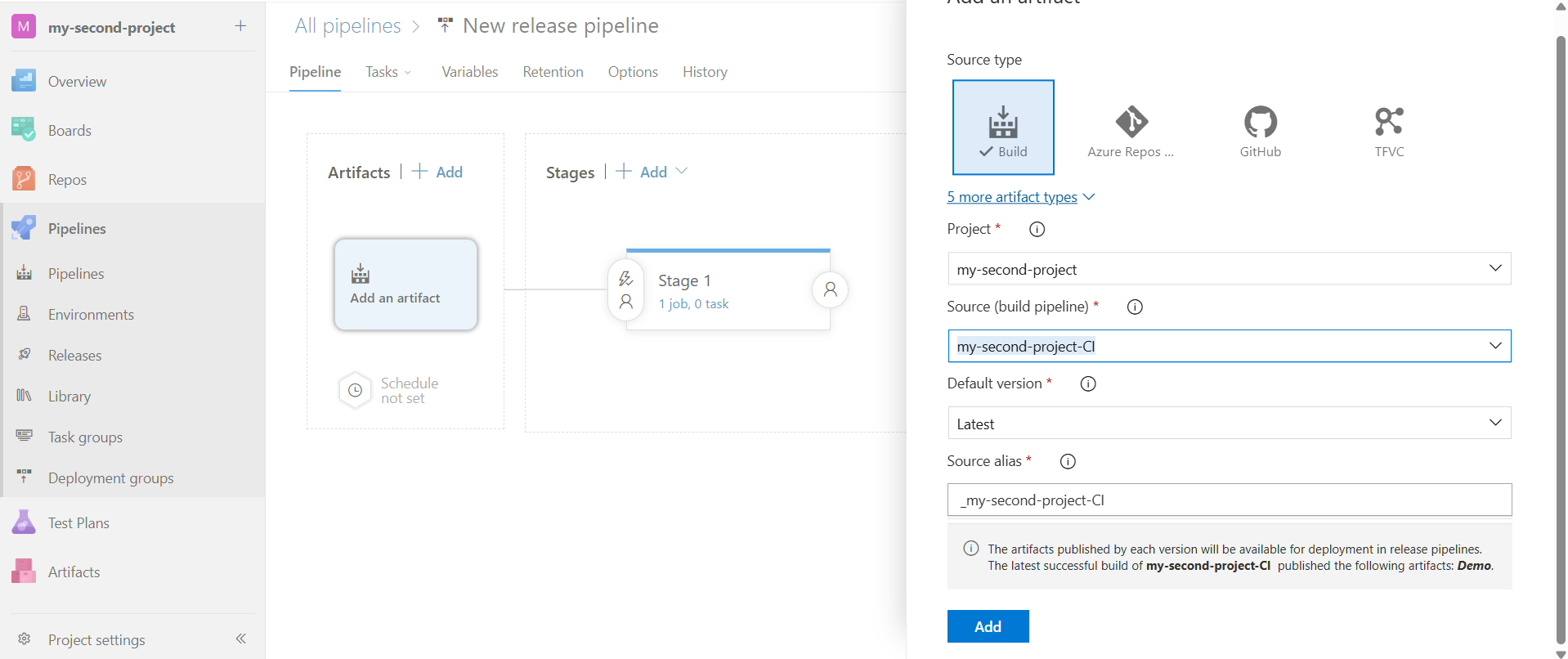


Fig: Adding of published artifacts (my-second-project-CI).

Then now add the stage, within the stage add jobs and tasks like Terraform init, validate, plan, and apply by selecting the self hosted agent pool (my-agent).

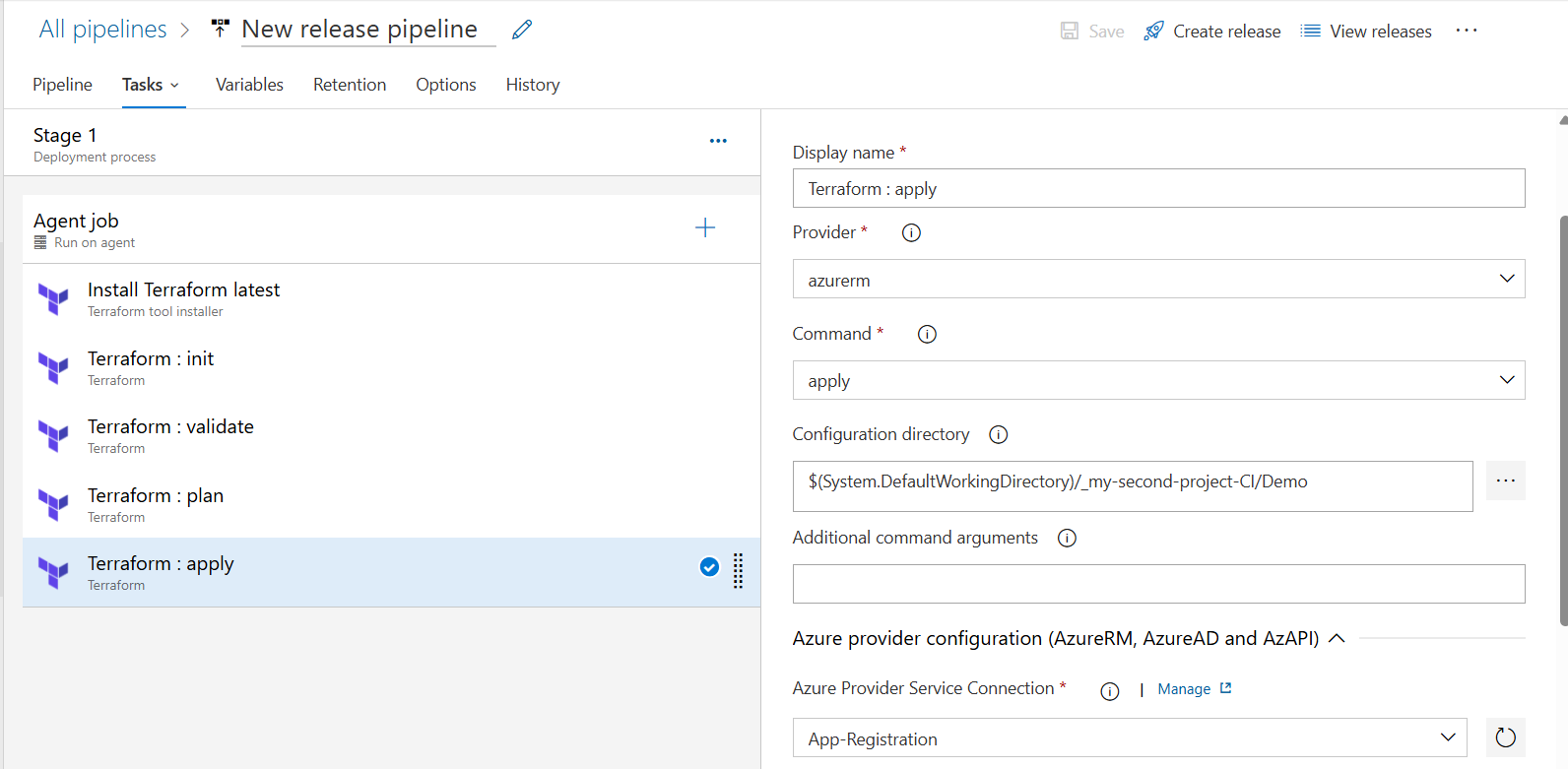
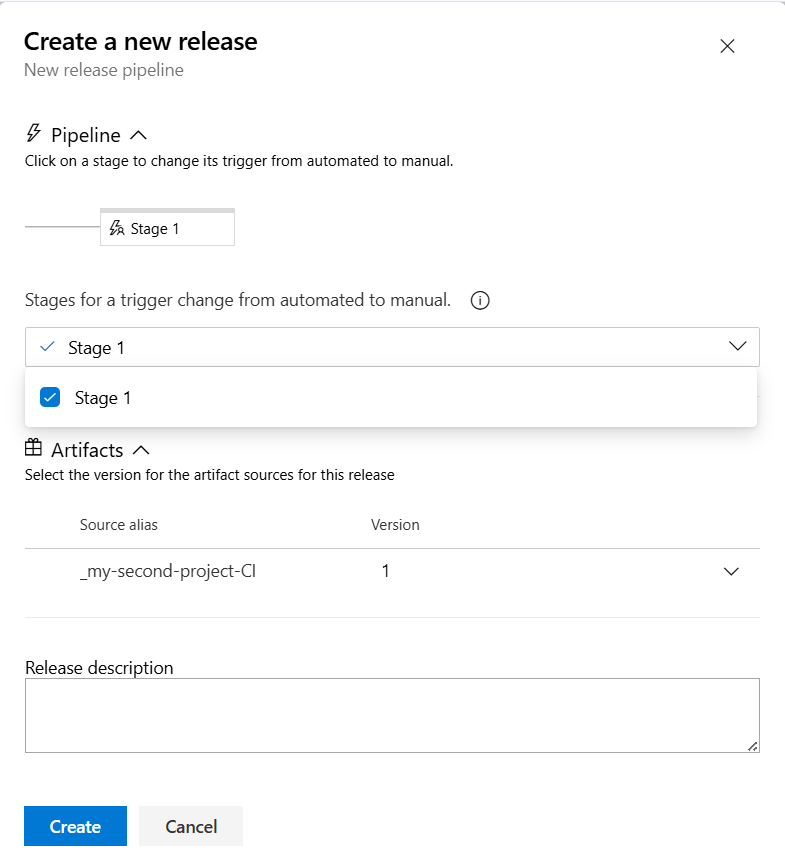


Fig: Terraform Tasks added in the Release.

Now create a Release:



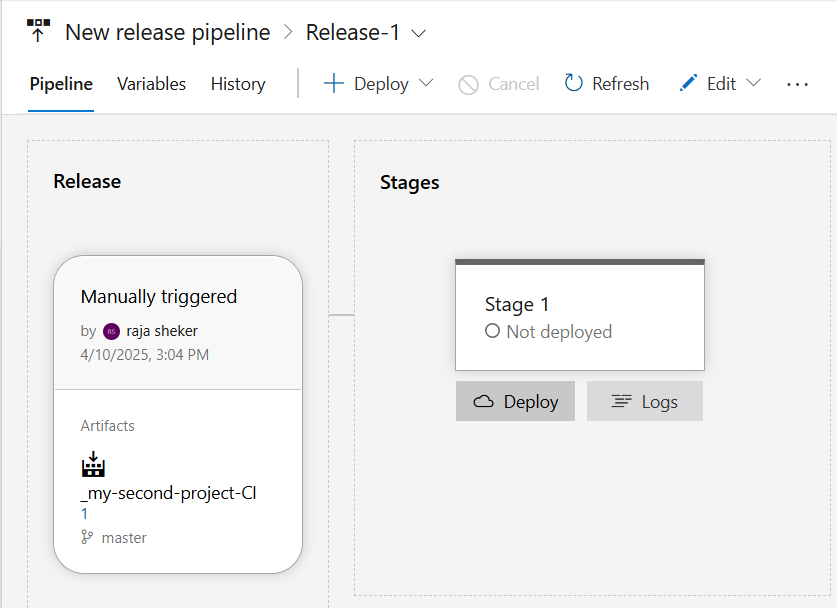


Fig: Deploy the Terraform code.

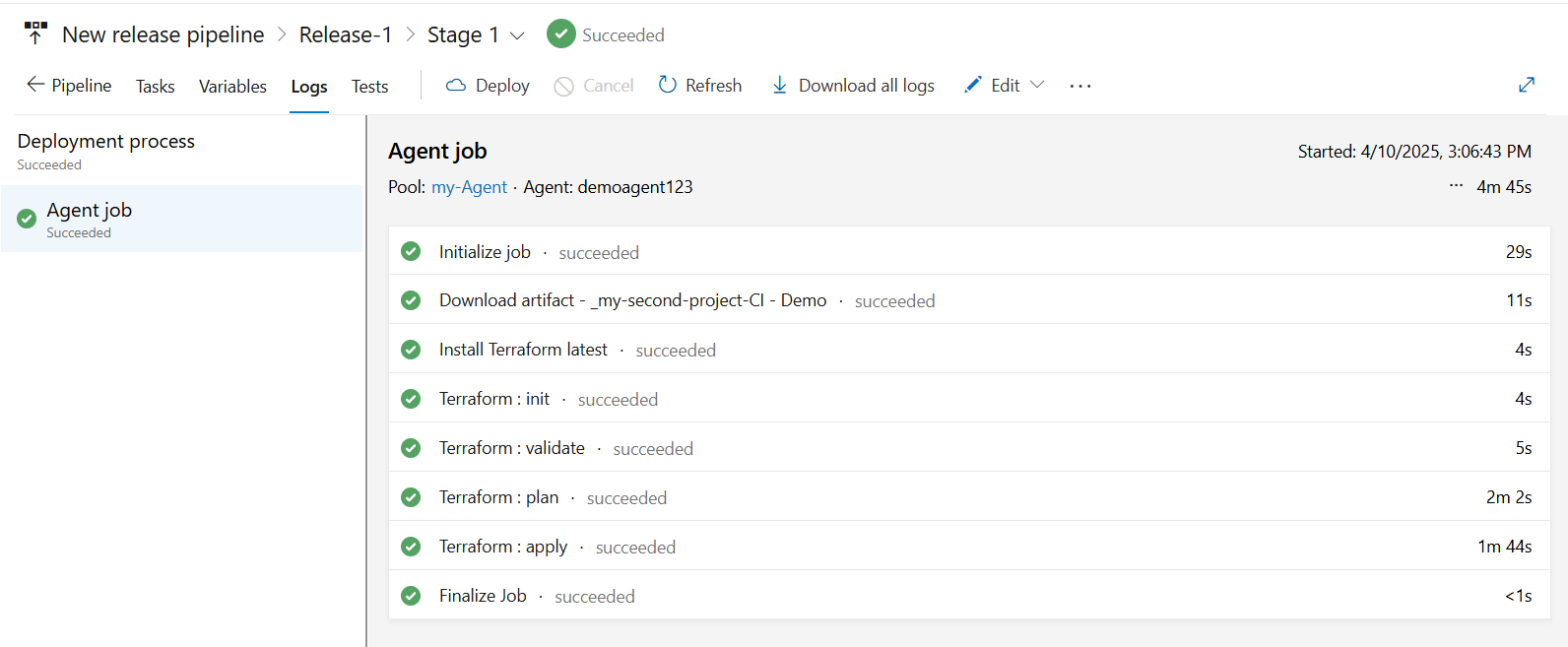
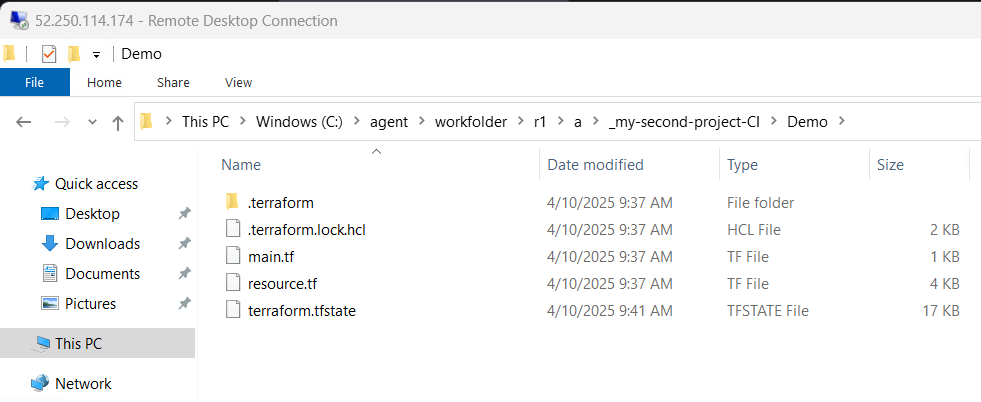


Fig: Deployment is done successfully.

After deployment process is done an **“r1”** (release folder) folder is created in self hosted windows agent machine (demoagent123) as shown in below figure.



After successfully deploying the terraform code of VM creation using Azure pipelines, the infrastructure (VM) is created in the Azure portal as shown below.

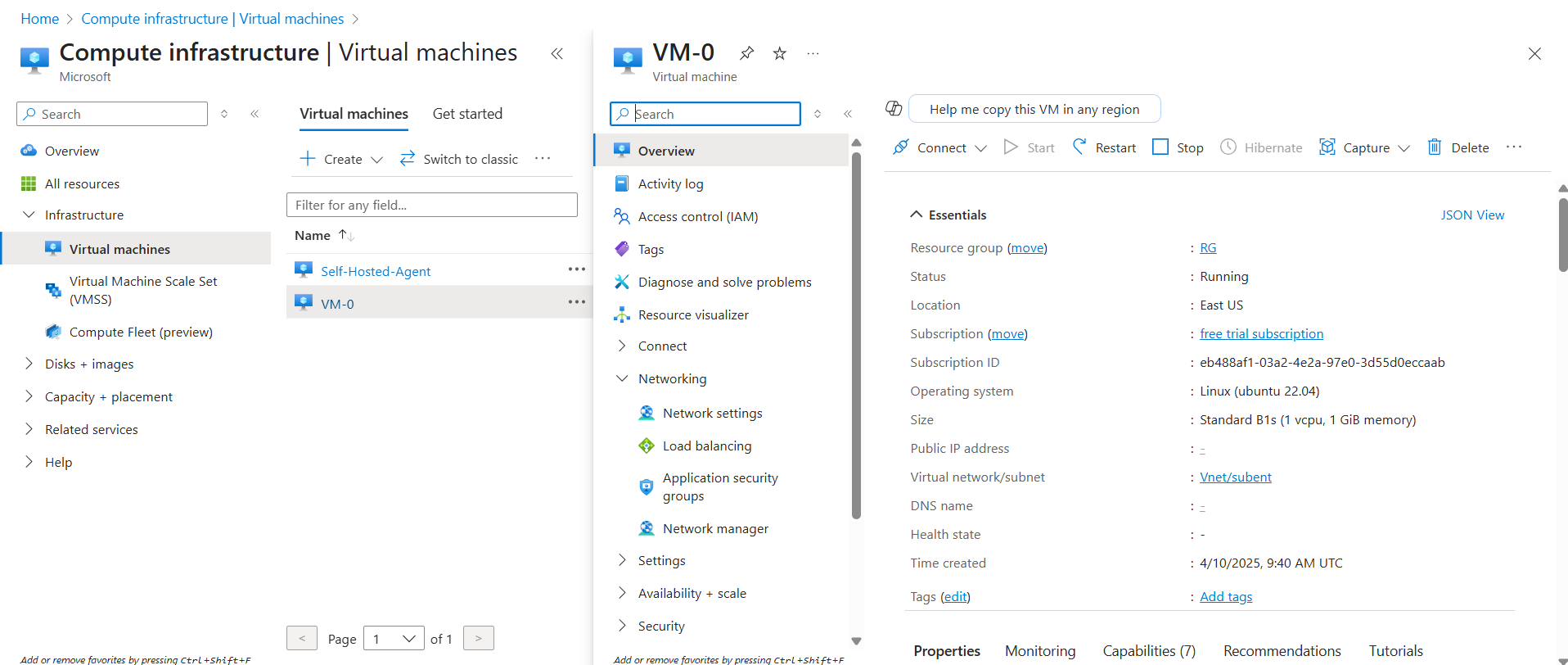


Fig: VM is created successfully using Azure Pipelines.