**Lab 12 - Azure DevOps**



**Part 1- Azure DevOps Sign up Instruction**

Sign up with a personal Microsoft account.

1. Select start free link for Azure Devops:

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

图形用户界面, 文本, 应用程序

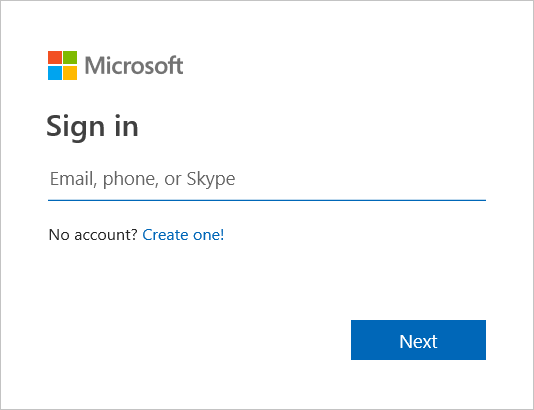
描述已自动生成

2. Enter your email address, phone number, or Skype ID for your Microsoft account. If you're a Visual Studio subscriber and you get Azure DevOps as a benefit, use the Microsoft account associated with your subscription. Select **Next**.

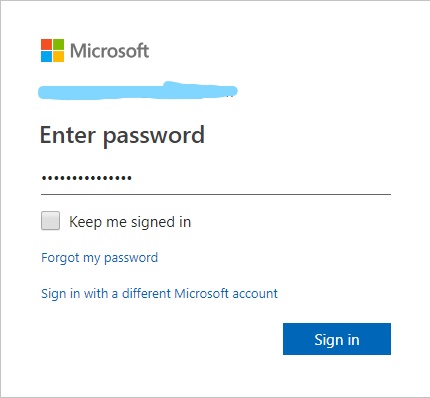
If you don't have a Microsoft account, choose **Create one**.

Note:

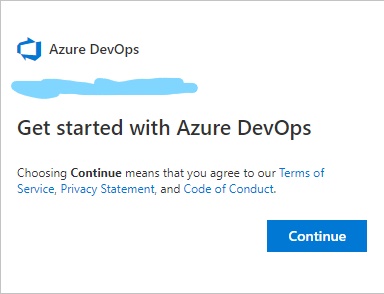
Use your Fordham email to create a new account.



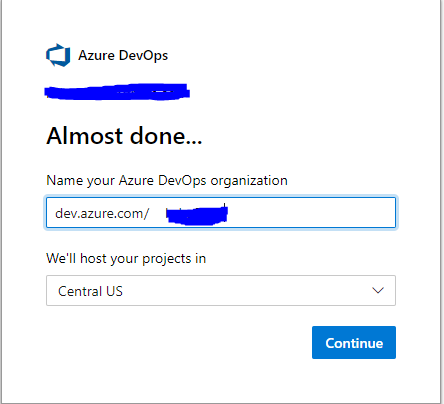
3. Enter your password and select **Sign in**.



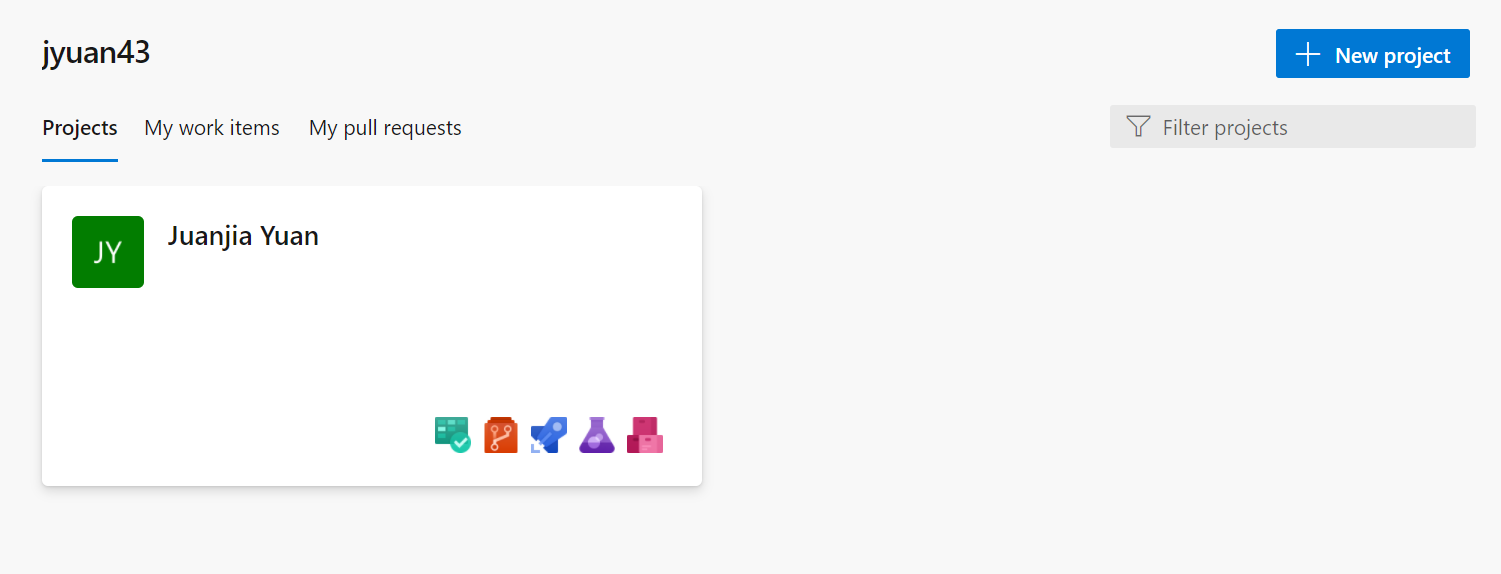
4. To get started with Azure DevOps, select **Continue**.



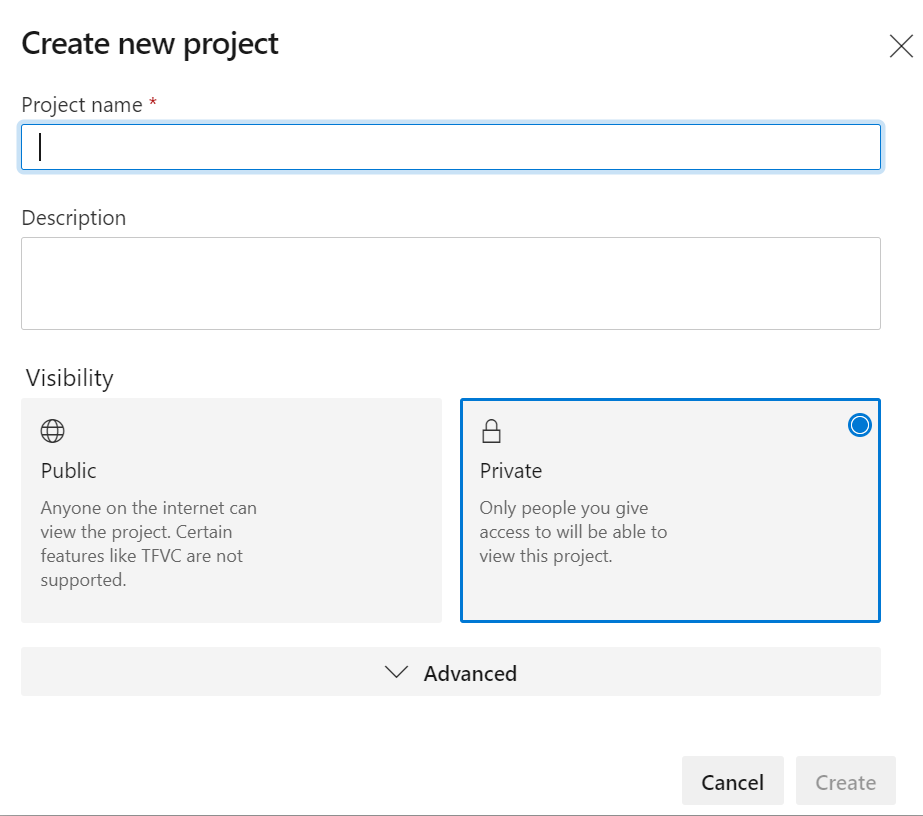
5. Create your Azure DevOps organization, and select **Continue**.

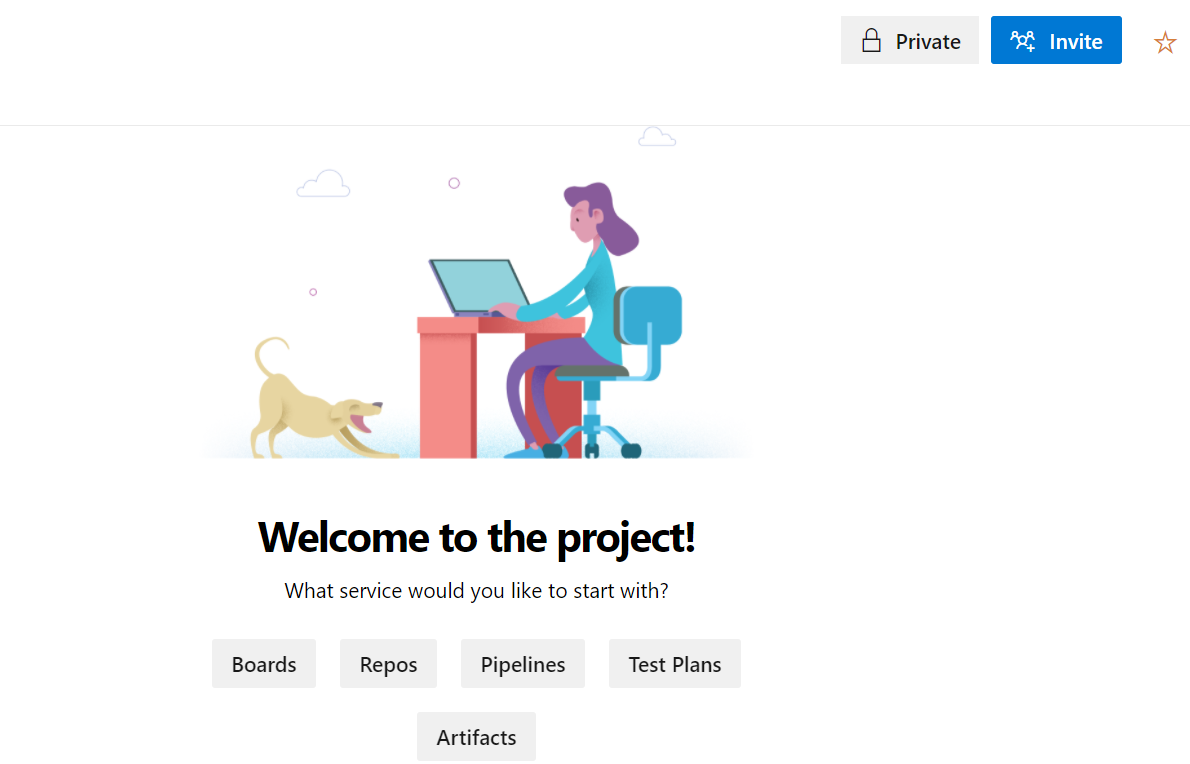


6. After logging in, create your new project.



7. enter project name and description





**Part 2 - Configure Jenkins using Azure CLI**

# Introduction

This handout introduces how to install Jenkins on an Ubuntu Linux VM with the tools and plug-ins configured to work with Azure.

# Jenkins

Jenkins is a popular open-source automation server used to set up continuous integration and delivery (CI/CD) for your software projects. You can host your Jenkins deployment in Azure or extend your existing Jenkins configuration using Azure resources. Jenkins plugins are also available to simplify CI/CD of your applications to Azure.

# Prerequisites

* Azure subscription: If you do not have an Azure subscription, create a free account before you begin.

# Create a virtual machine

1. Sign into the Azure Portal: <https://portal.azure.com/#home>

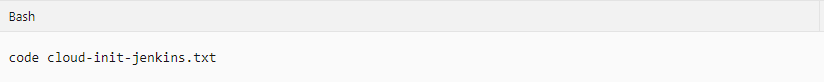
2. Open Azure Cloud Shell and select **Bash:**

Option 1: Select the Cloud Shell icon on the Azure Portal

Icon to launch the Cloud Shell from the Azure portal

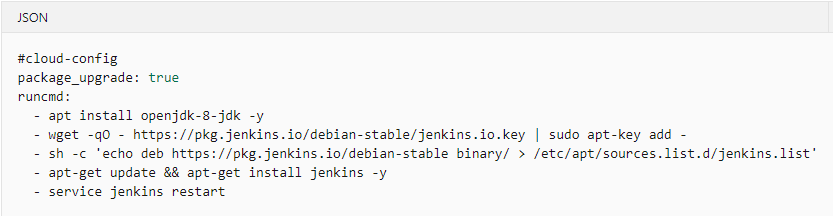
Option 2: Direct link: [https://shell.azure.com](https://shell.azure.com/)

3. Create a file named **cloud-init-jenkins.txt**.



Code: code cloud-init-jenkins.txt

4. Paste the following code into the new file:



Code:

#cloud-config

package\_upgrade: true

runcmd:

- apt install openjdk-8-jdk -y

- wget -qO - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -

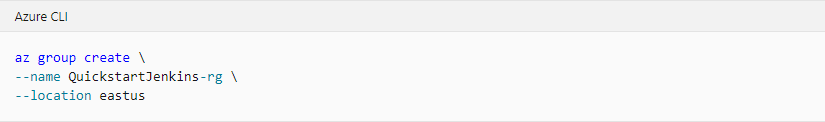
- sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

- apt-get update && apt-get install jenkins -y

- service jenkins restart

5. Save the file (**<Ctrl>S**) and exit the editor (**<Ctrl>Q**).

6. Using **az group create** to create a resource grouping. **Notice**: Based on different environment, **--location** may be replaced by appropriate value, but in most cases, **–location** is good to use.



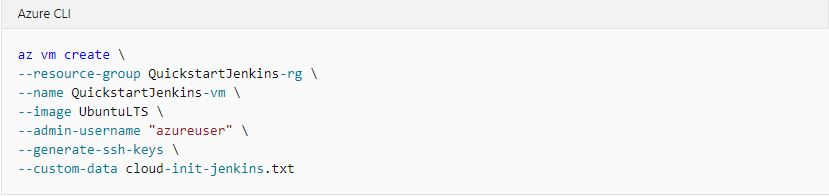
Code:

az group create \

--name QuickstartJenkins-rg \

--location eastus

7. Using **az vm create** to create virtual machine



Code:

az vm create \

--resource-group QuickstartJenkins-rg \

--name QuickstartJenkins-vm \

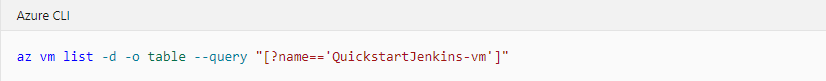
--image UbuntuLTS \

--admin-username "azureuser" \

--generate-ssh-keys \

--custom-data cloud-init-jenkins.txt

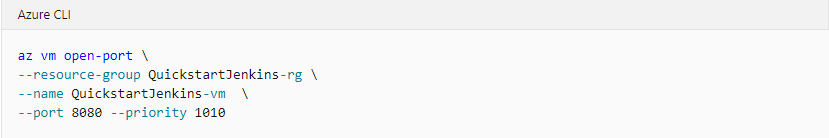
8. Verify the creation (and state) of the new virtual machine using **az vm list**.



Code:

az vm list -d -o table --query "[?name=='QuickstartJenkins-vm']"

9. By default, Jenkins runs on port 8080. Therefore, open port 8080 on the new virtual machine using **az vm open**.



Code:

az vm open-port \

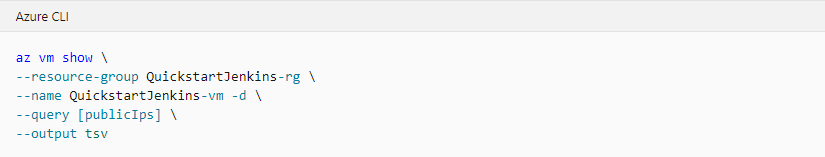
--resource-group QuickstartJenkins-rg \

--name QuickstartJenkins-vm \

--port 8080 --priority 1010

# Configure Jenkins

1. Using az vm show to get the public IP address for the sample virtual machine



Notes: The **--query** parameter limits the output to the public IP addresses for the virtual machine.

Code:

az vm show \

--resource-group QuickstartJenkins-rg \

--name QuickstartJenkins-vm -d \

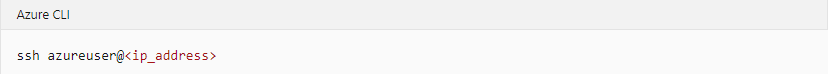
--query [publicIps] \

--output tsv

2. Using the IP address retrieved in the previous step, SSH into the virtual machine. You'll need to confirm the connection request.

**Notes**:

* Upon successful connection, the Cloud Shell prompt includes the username and virtual machine name: azureuser@QuickstartJenkins-vm.

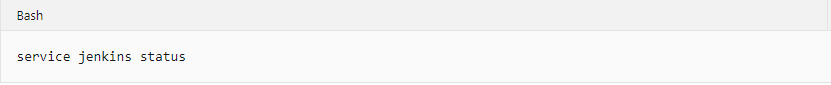


Code: ssh azureuser@<ip\_address>

Example: if your IP address is 55.555.555.555, the code should be

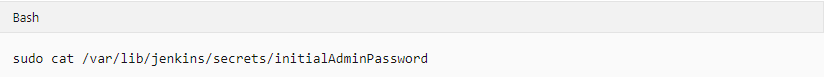
**ssh** [**azureuser@55.555.555.555**](mailto:azureuser@55.555.555.555)

3. Verify that Jenkins is running by getting the status of the Jenkins service.



Code: service jenkins status

4. Get the autogenerated Jenkins password.



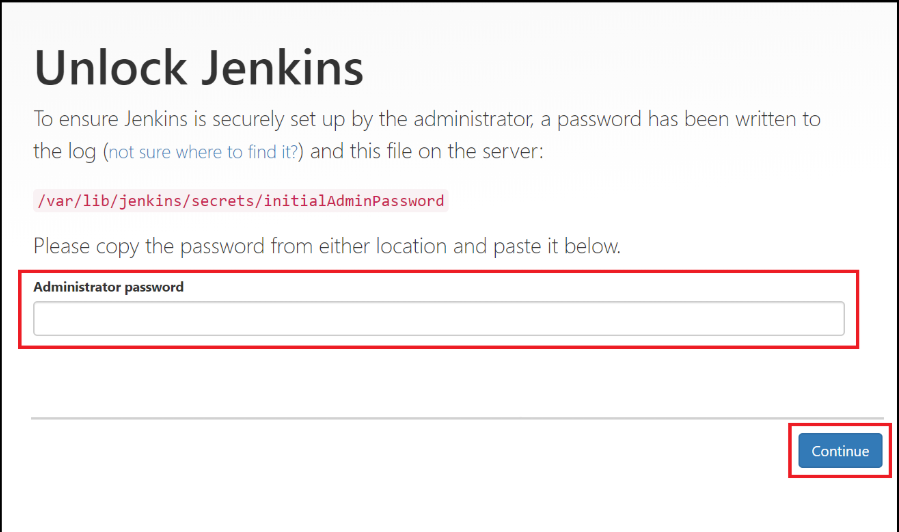
Code: sudo cat /var/lib/jenkins/secrets/initialAdminPassword

5. Using the IP address, open the following URL in a browser: http://<ip\_address>:8080

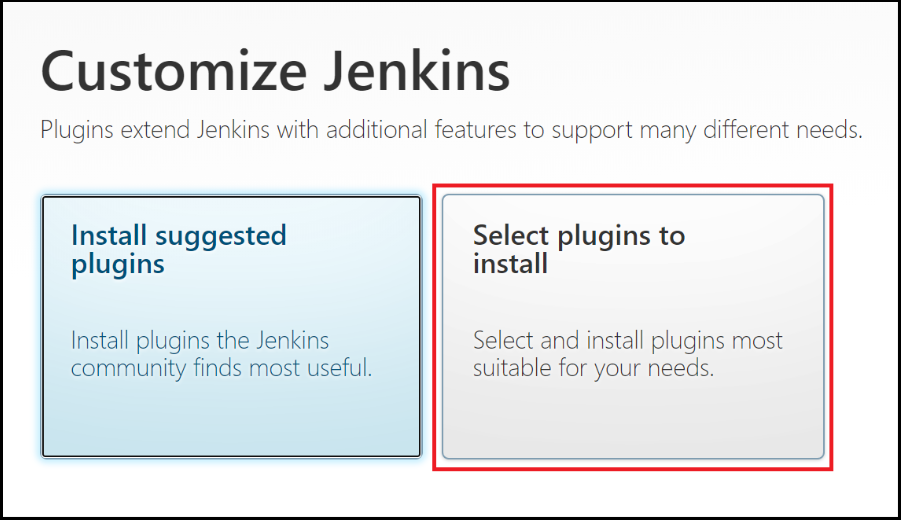
Example: if your IP address is 55.555.555.555, the URL should be

55.555.555.555:8080

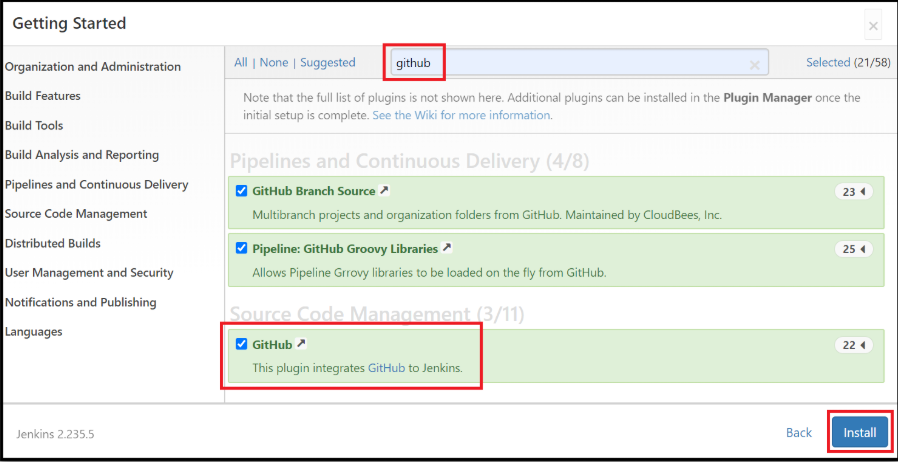
6. Enter the password you retrieved earlier and select **Continue**.



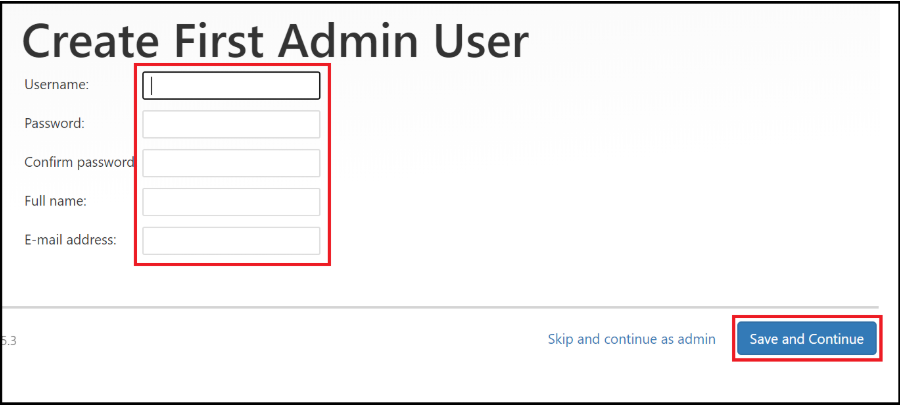
7. Select **Select plugins to install**.



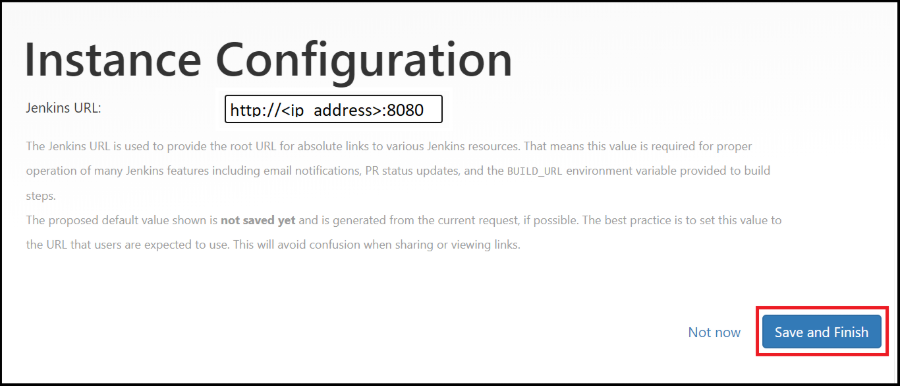
8. In the filter box at the top of the page, enter github. Select the GitHub plugin and select **Install**.



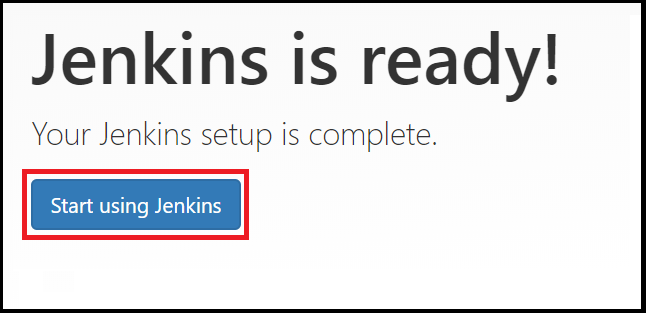
9. Enter the information for the first admin user and select **Save and Continue**.



10. On the **Instance Configuration** page, select **Save and Finish**.

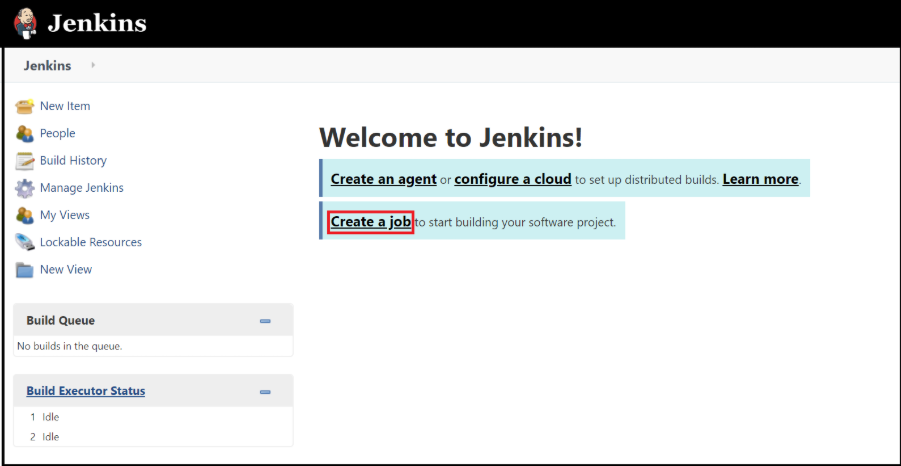


11. Select **Start using Jenkins**.

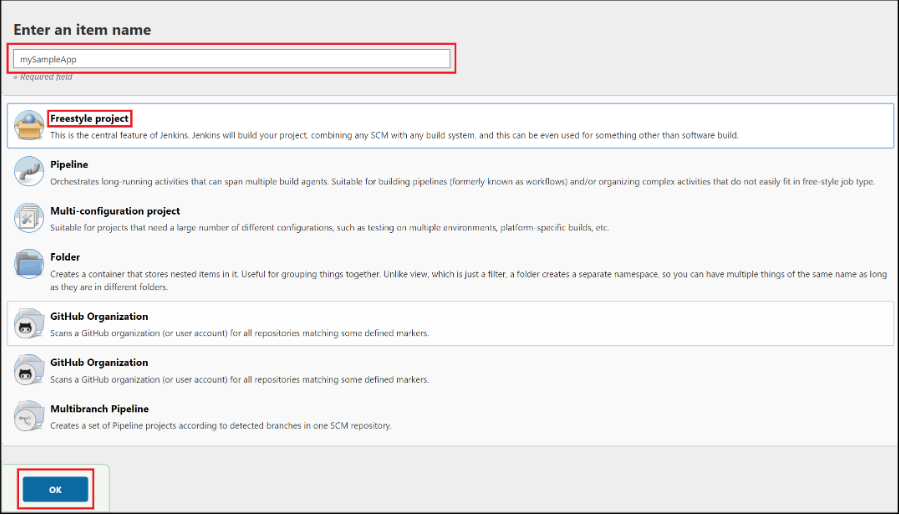


# Create First Job

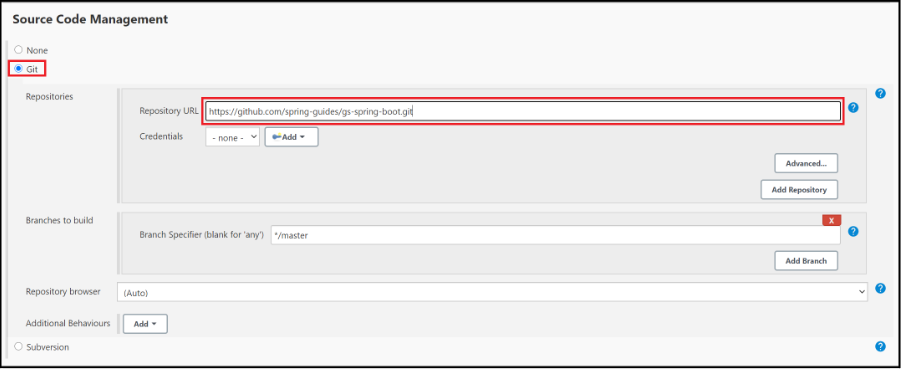
1. On the Jenkins home page, select **Create a job**.



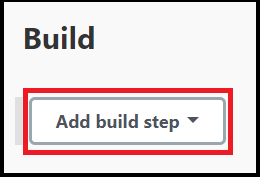
2. Enter a job name of **mySampleApp**, select **Freestyle project**, and select **OK**.



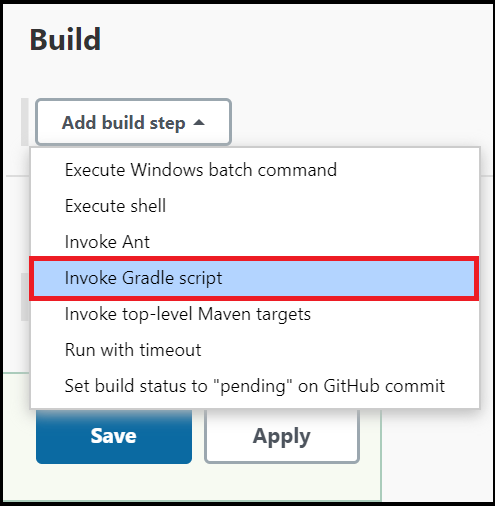
3. Select the **Source Code Management** tab. Enable **Git** and enter the following URL for the **Repository URL** value: <https://github.com/spring-guides/gs-spring-boot.git>



4. Select the **Build** tab, then select **Add build step**



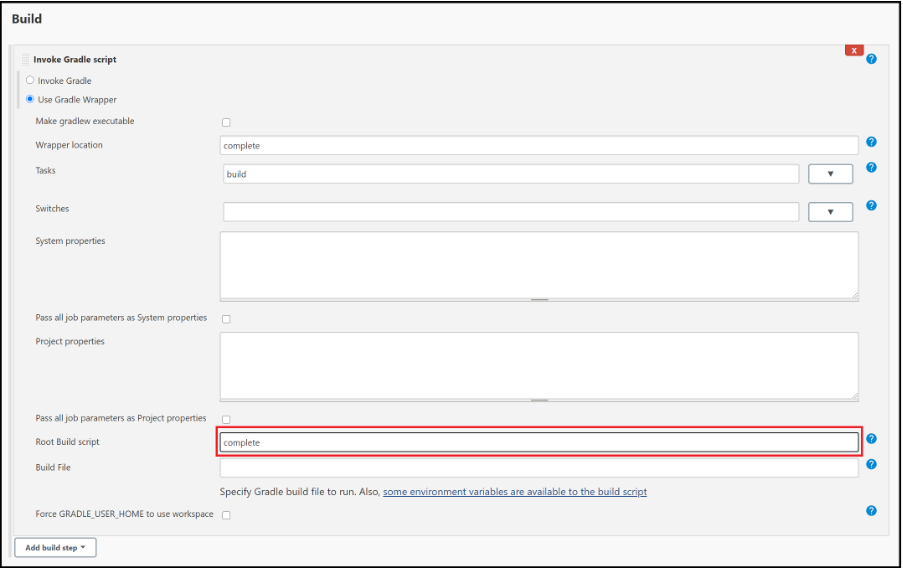
5. From the drop-down menu, select **Invoke Gradle script**.



6. Select **Use Gradle Wrapper**, then enter complete in **Wrapper location** and build for **Tasks**.



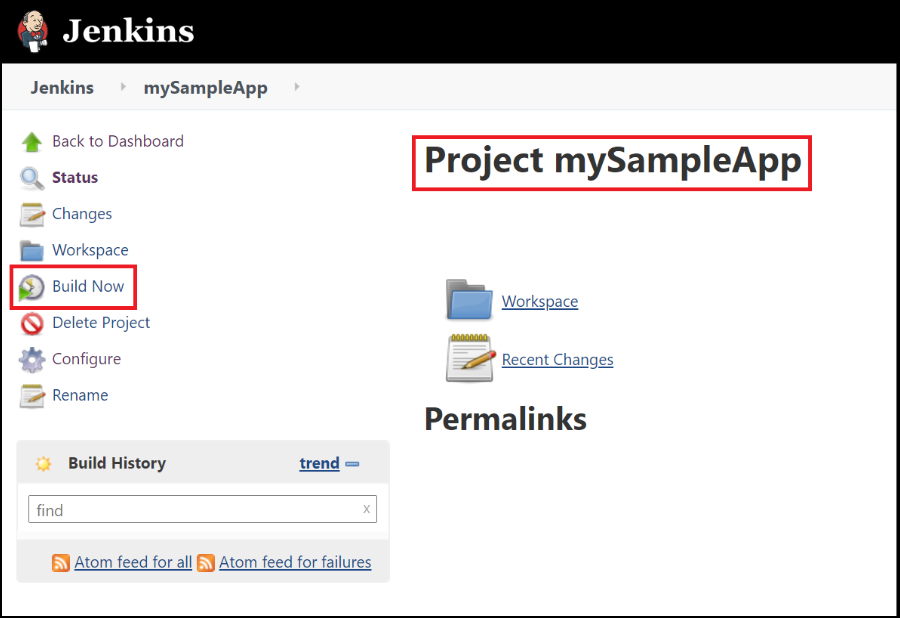
7. Select **Advanced** and enter complete in the **Root Build script** field.



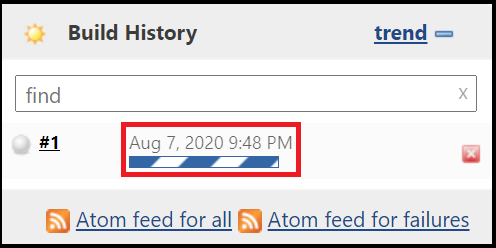
8. Scroll to the bottom of the page and select **Save**.

# Build the sample Java app

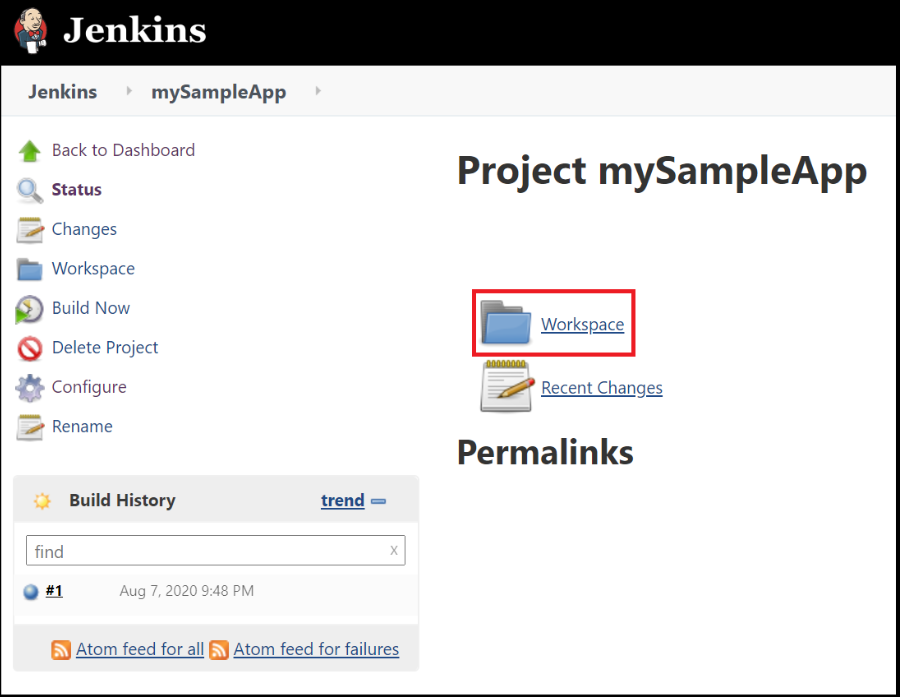
1. When the home page for your project displays, select **Build Now** to compile the code and package the sample app.



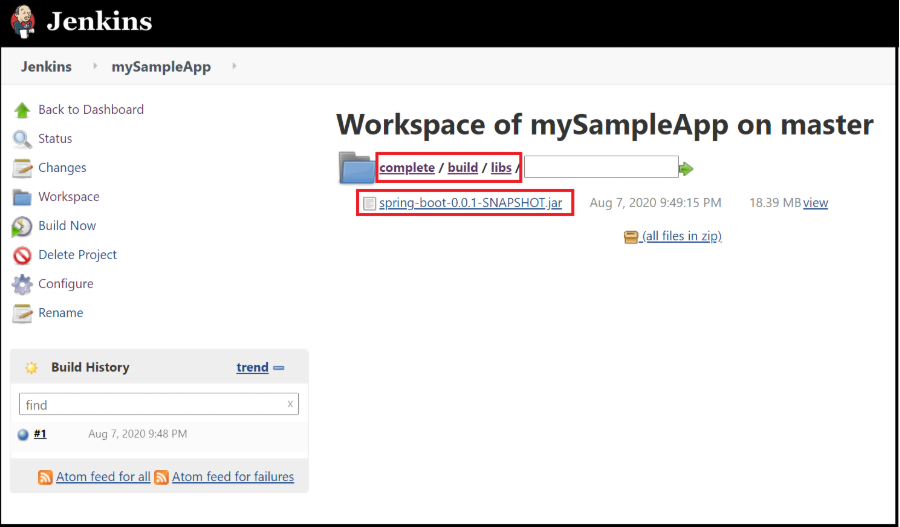
2. A graphic below the **Build History** heading indicates that the job is being built.



3. When the build completes, select the **Workspace** link.



4. Navigate to **complete/build/libs** to see that the **.jar** file was successfully built.



5. Your Jenkins server is now ready to build your own projects in Azure!