**Install the various components including Jenkins and modify GitHub repository.**

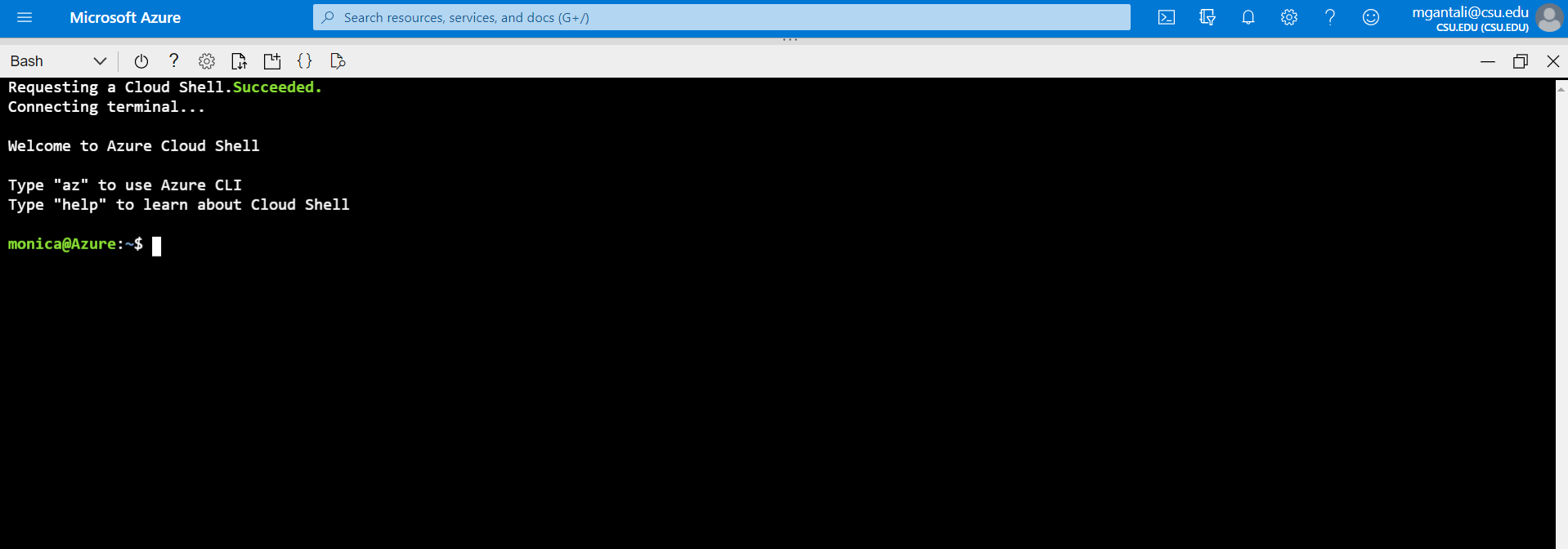
**Create a virtual machine**

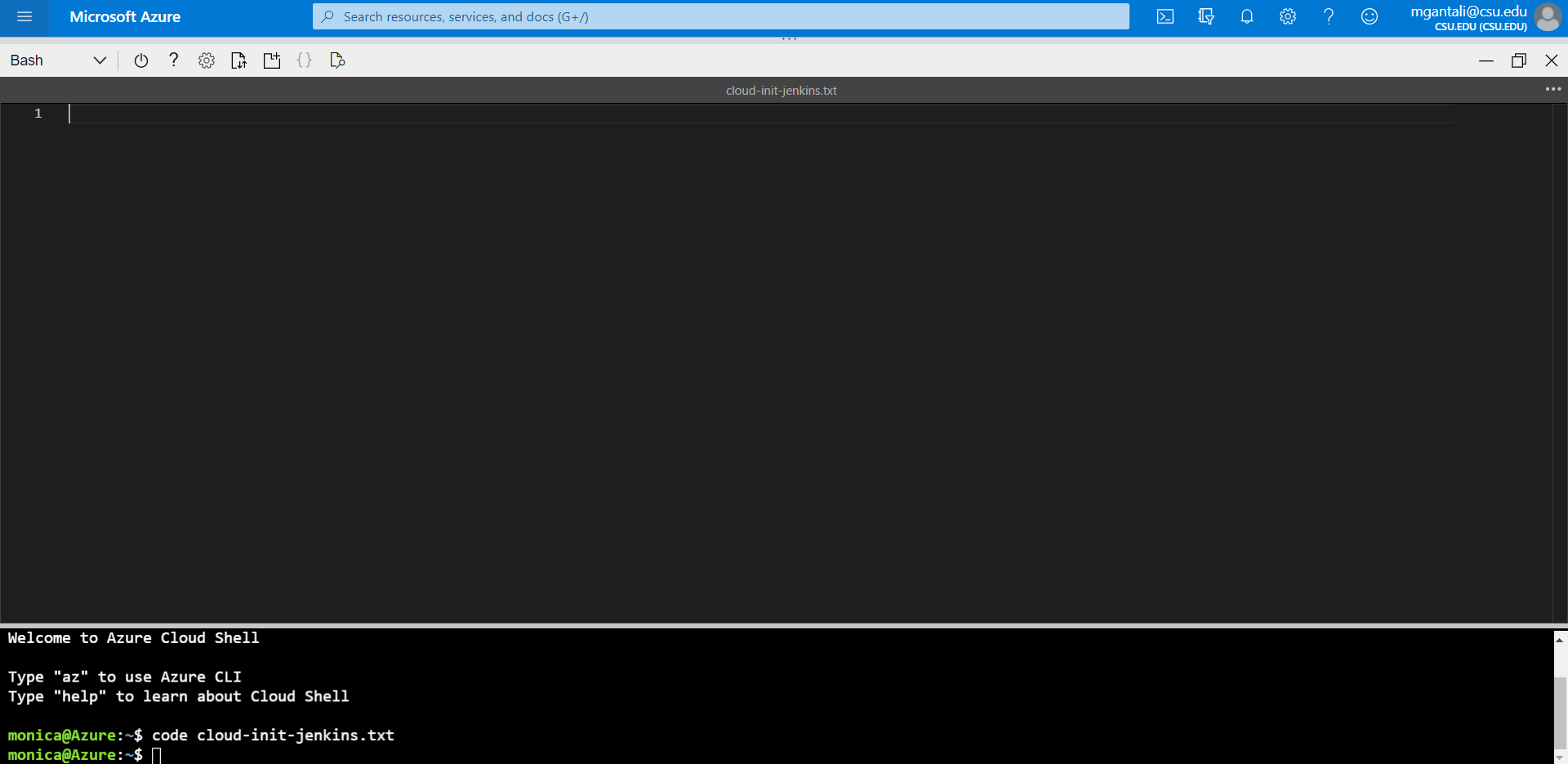
1. Sign in to the Azure portal.

2. Open Azure Cloud Shell and - if not done already - switch to **Bash**.

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







4) Paste the following code into the new file:

**#cloud-config**

**package\_upgrade: true**

**runcmd:**

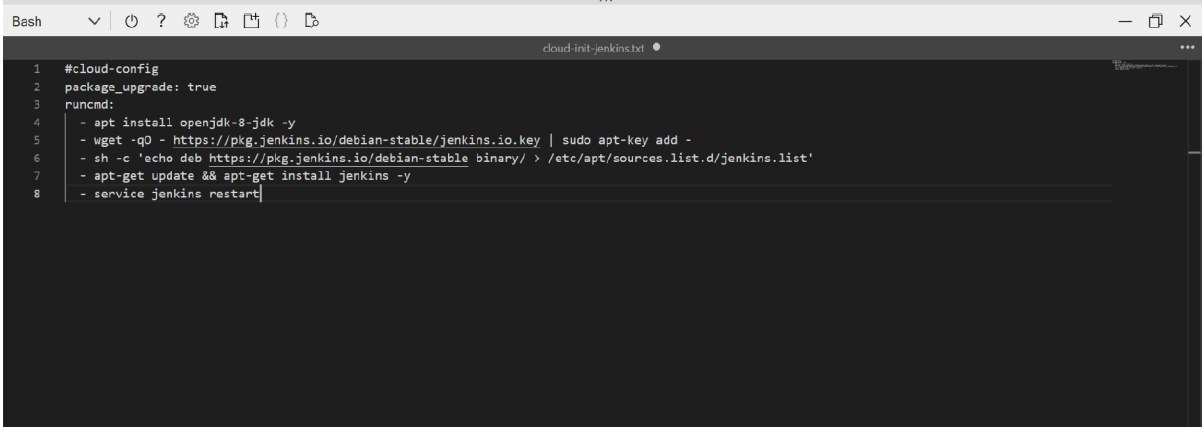
**- apt install openjdk-8-jdk -y**

**- 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**



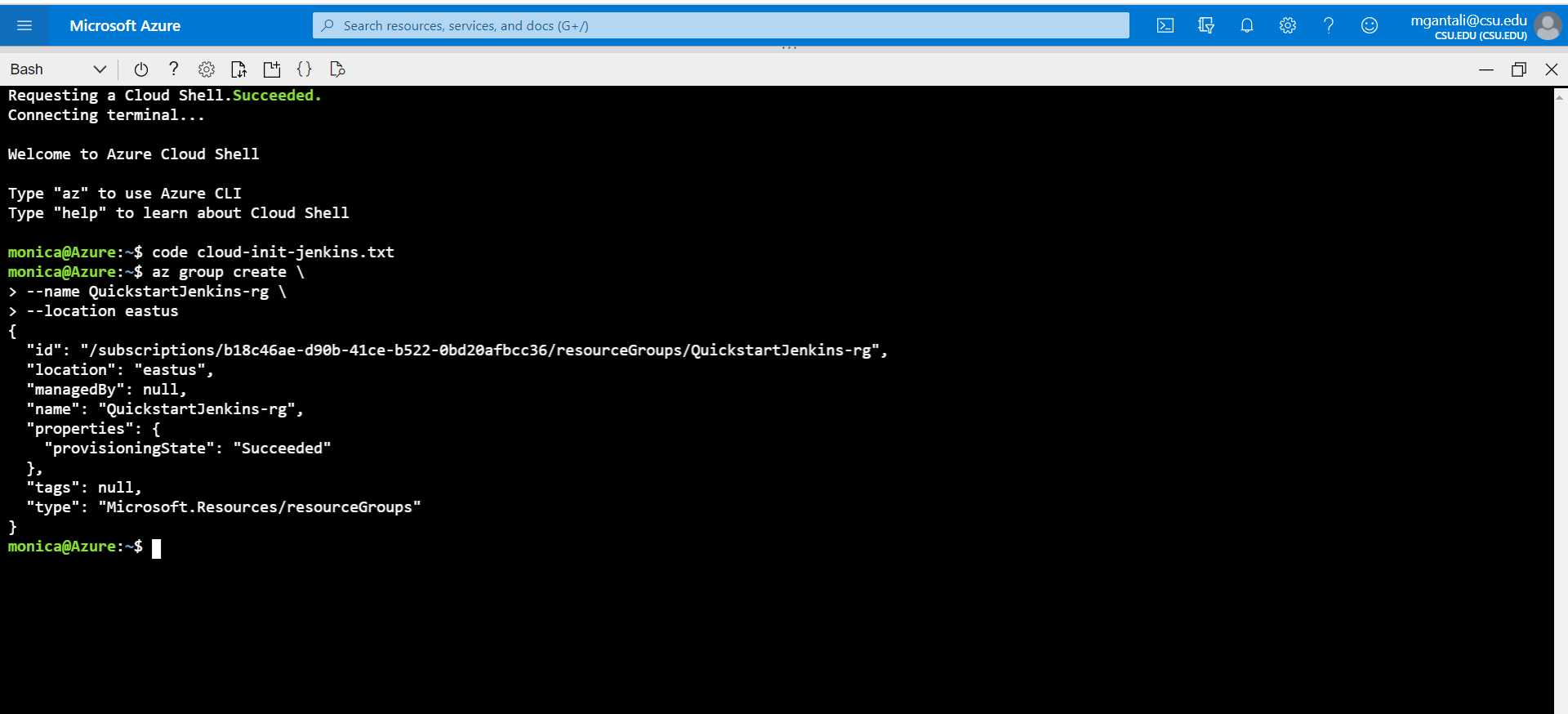
5) Save the file (**<Ctrl>S**) and exit the editor (**<Ctrl>Q**).

6) Create a resource group using az group create. You might need to replace the --location parameter with the appropriate value for your environment.

**az group create \**

**--name QuickstartJenkins-rg \**

**--location eastus**



7) Create a virtual machine using az vm create.

**az vm create \**

**--resource-group QuickstartJenkins-rg \**

**--name QuickstartJenkins-vm \**

**--image UbuntuLTS \**

**--admin-username "azureuser" \**

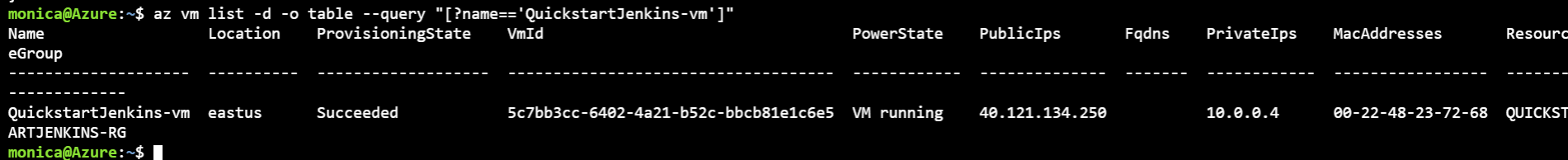
**--generate-ssh-keys \**

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



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

**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.

**az vm open-port \**

**--resource-group QuickstartJenkins-rg \**

**--name QuickstartJenkins-vm \**

**--port 8080 --priority 1010**



**Configure Jenkins**

1) Get the public IP address for the sample virtual machine using az vm show.

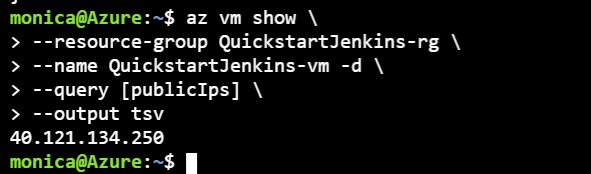
**az vm show \**

**--resource-group QuickstartJenkins-rg \**

**--name QuickstartJenkins-vm -d \**

**--query [publicIps] \**

**--output tsv**



**Notes**:

1) The --query parameter limits the output to the public IP addresses for the virtual machine.

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

**ssh azureuser@<ip\_address>**



**Notes**:

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

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

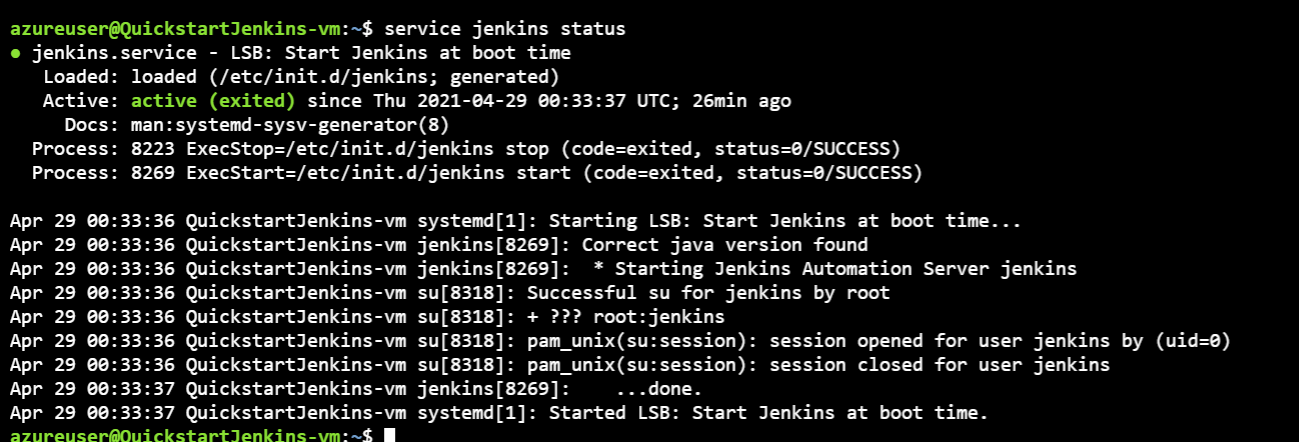
Command - *service jenkins status*

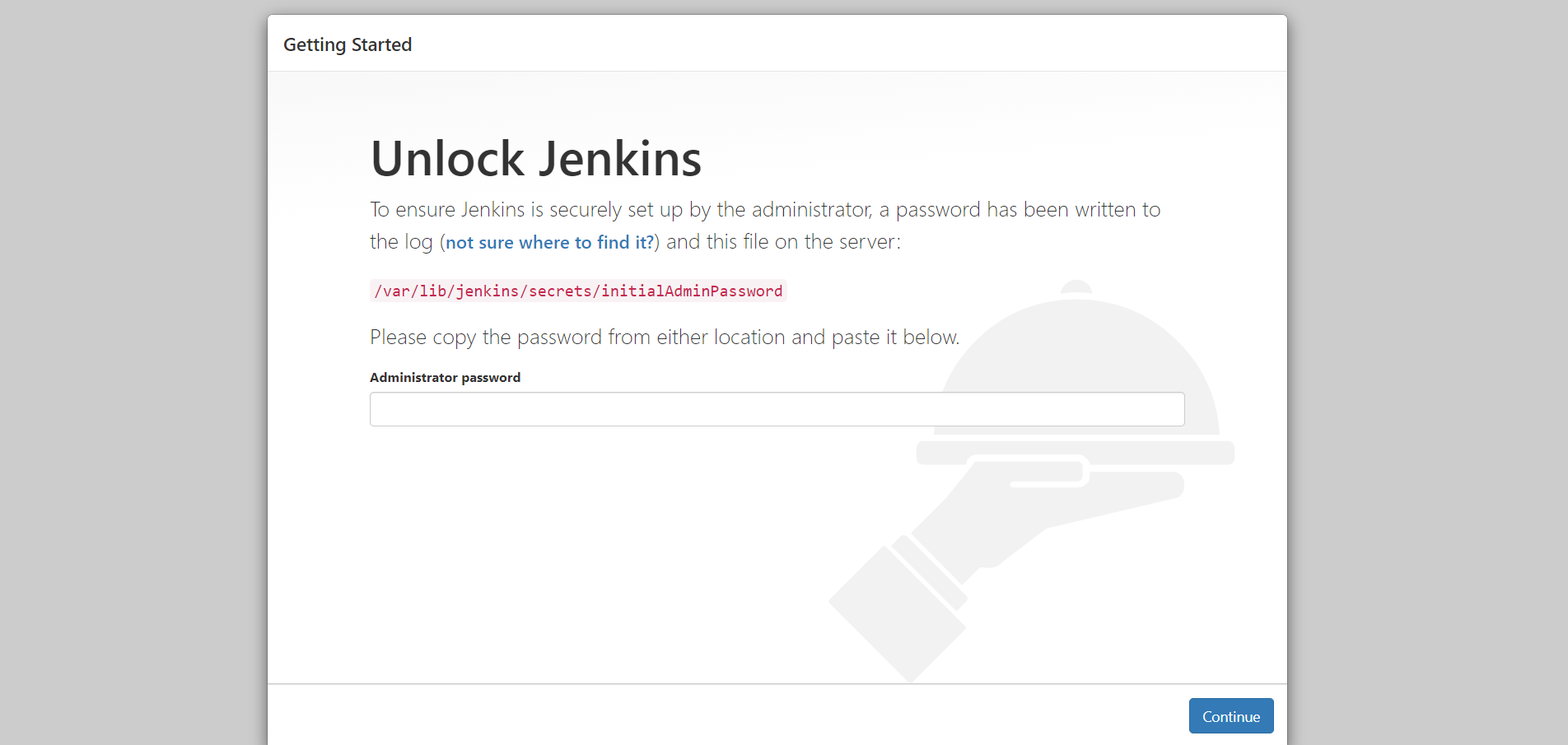
4) Get the auto generated Jenkins password.

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

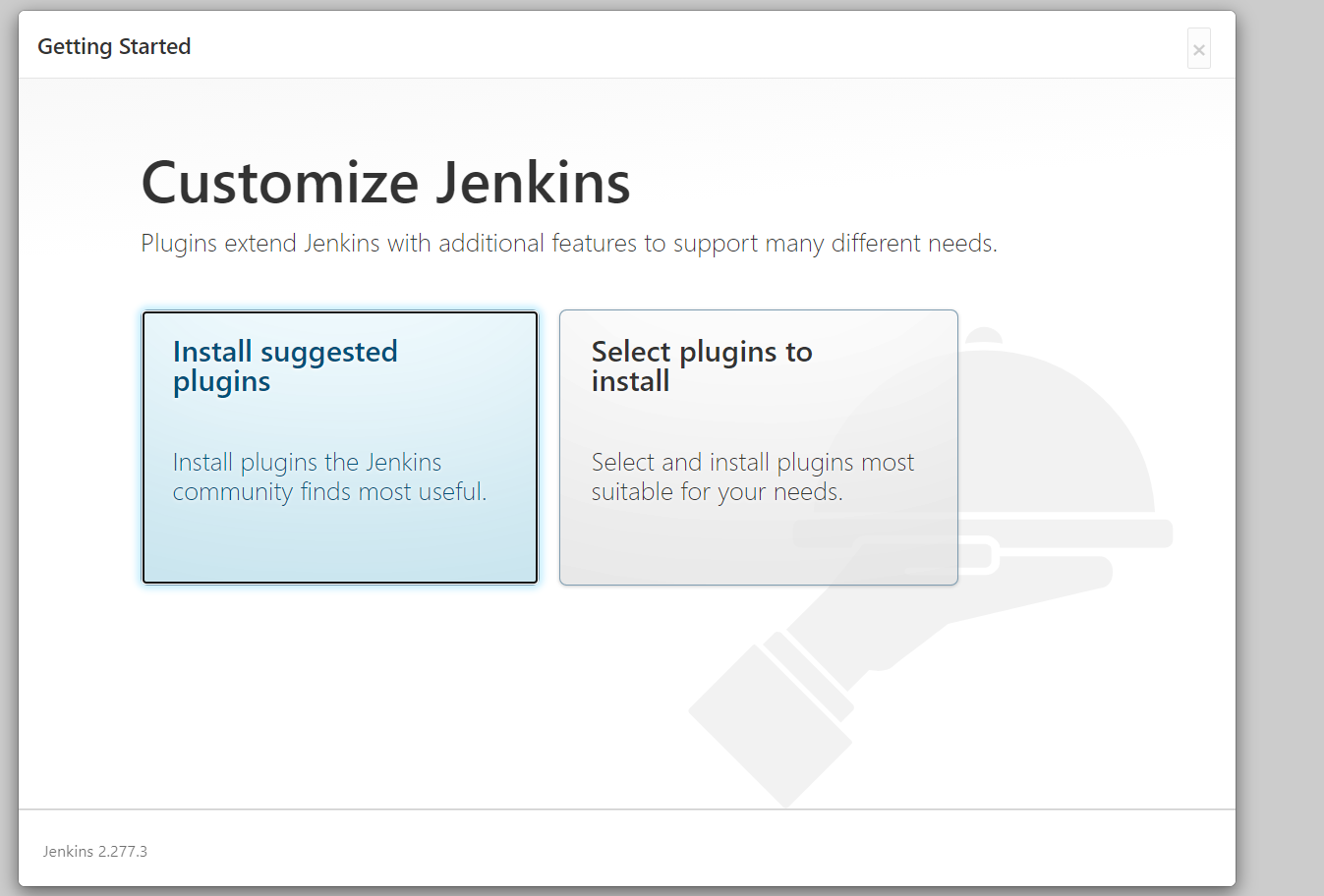
5) Using the IP address, open the following URL in a browser: http://<ip\_address>:8080

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



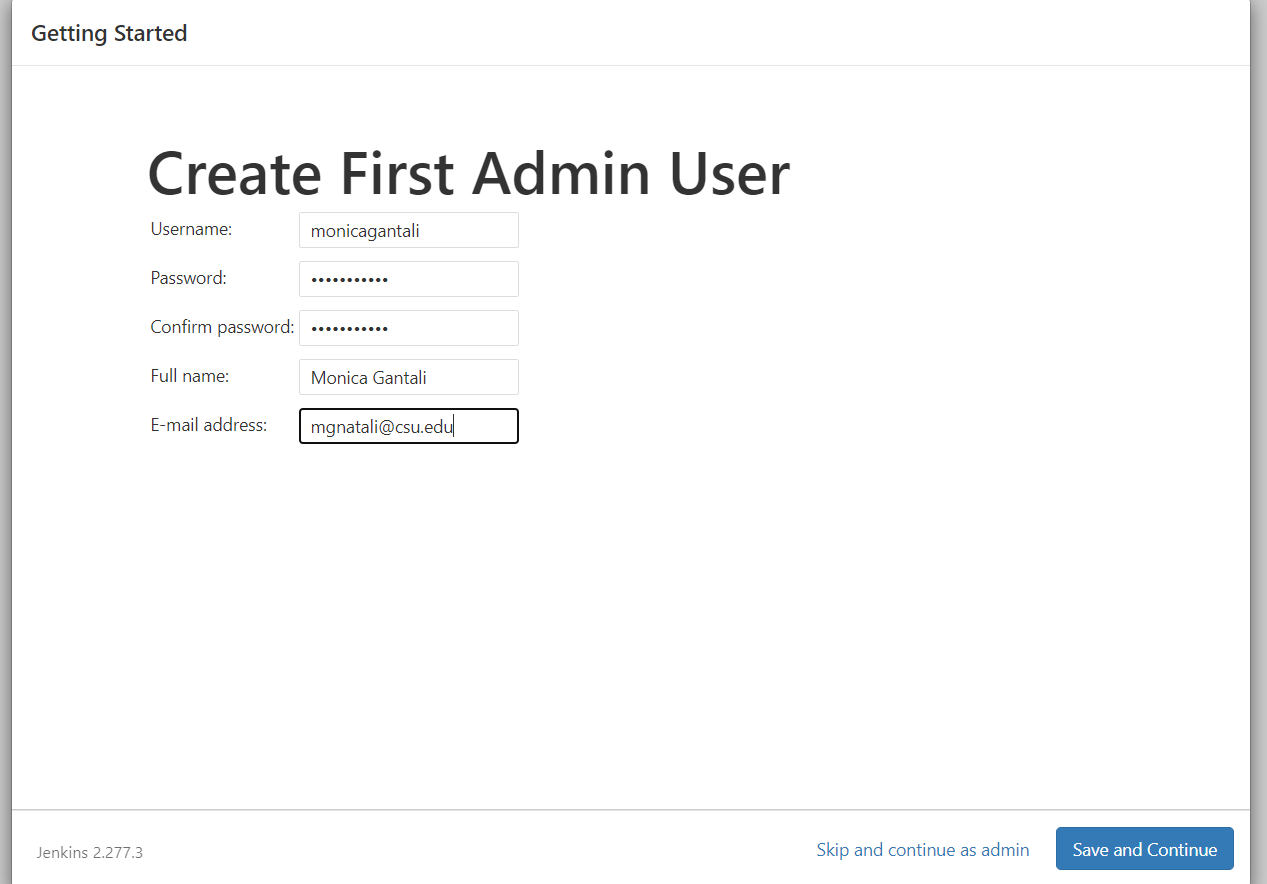


7) Select **Select plugins / install suggested plugins to install**.



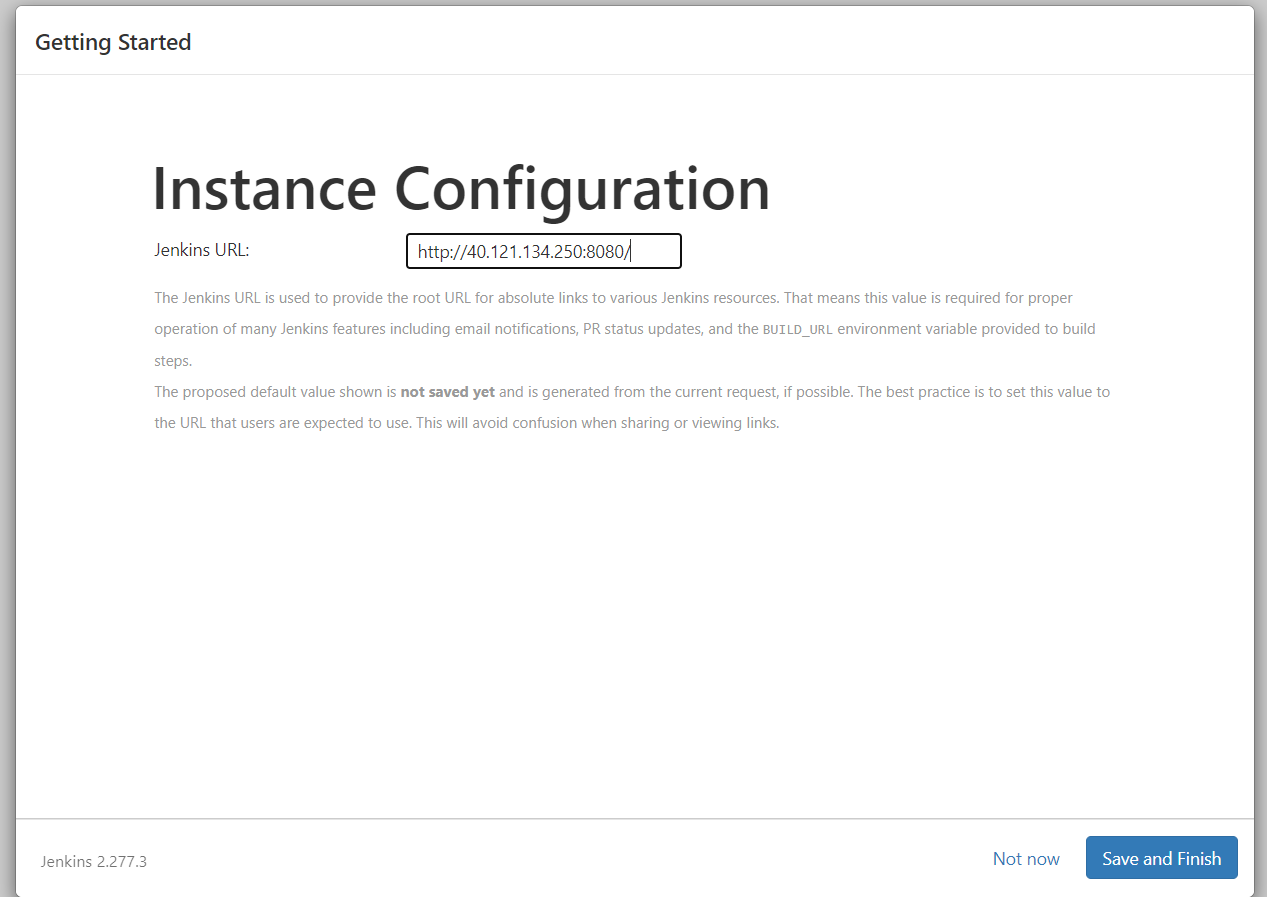
8) Enter the information for the first admin user and select **Save and Continue**.

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



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

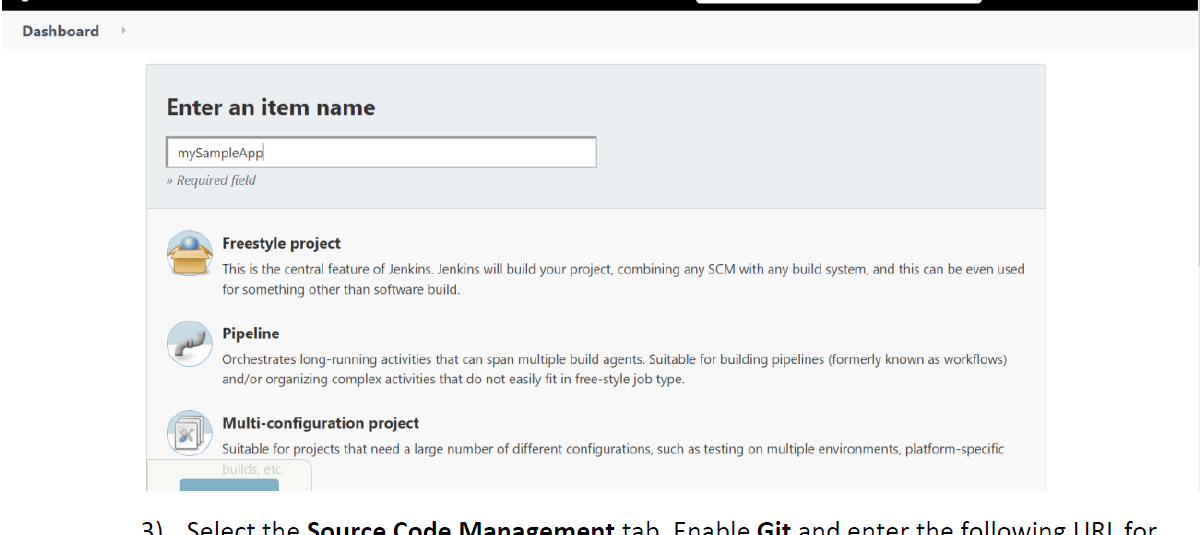
**Create**



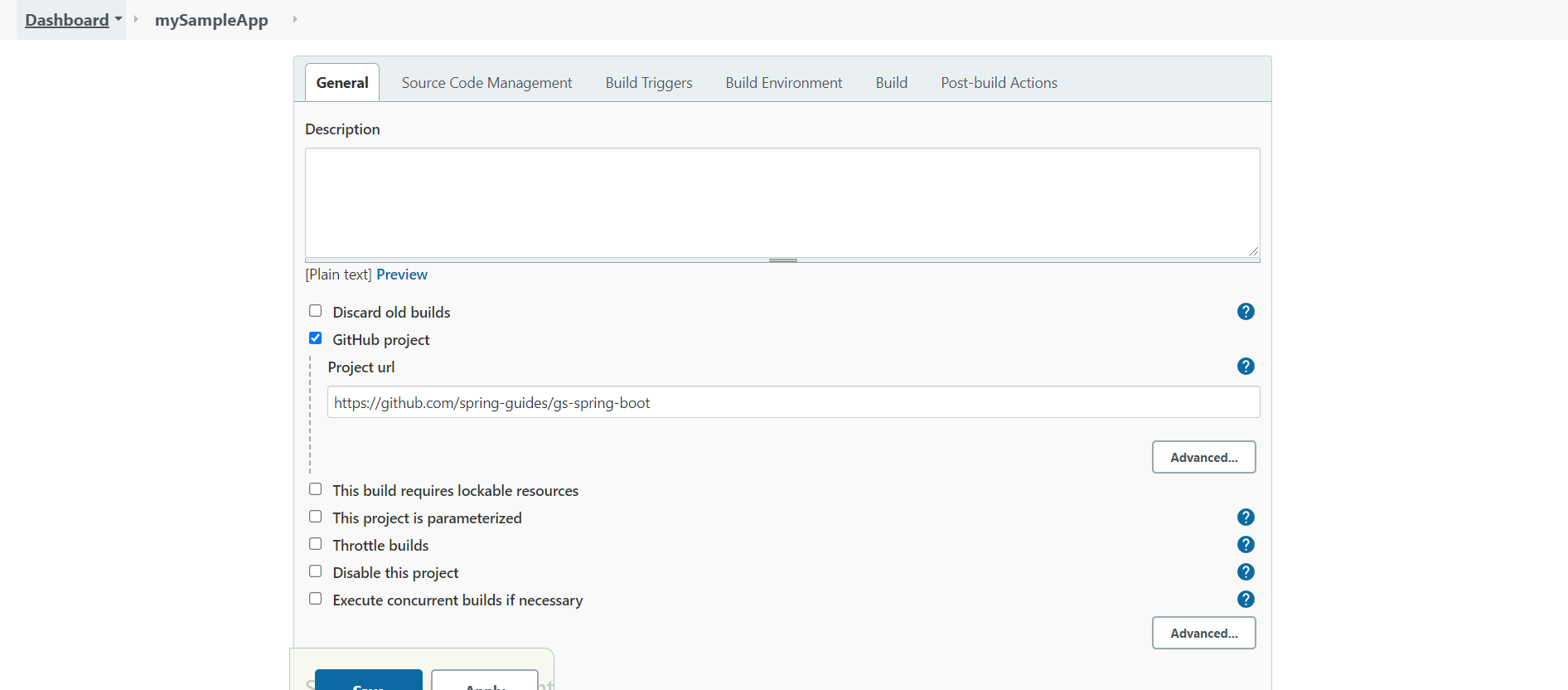
**Create your first job**

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**



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

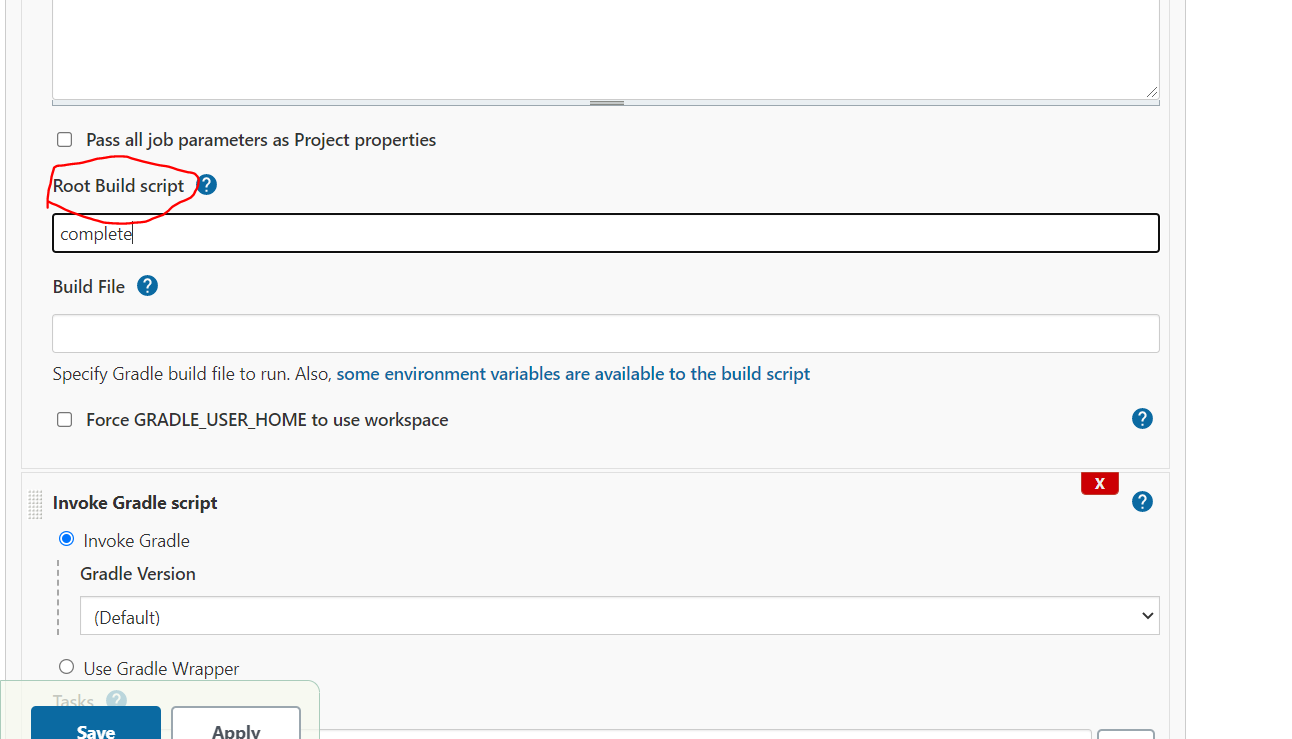
5) From the drop-down menu, select **Invoke Gradle script**.



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

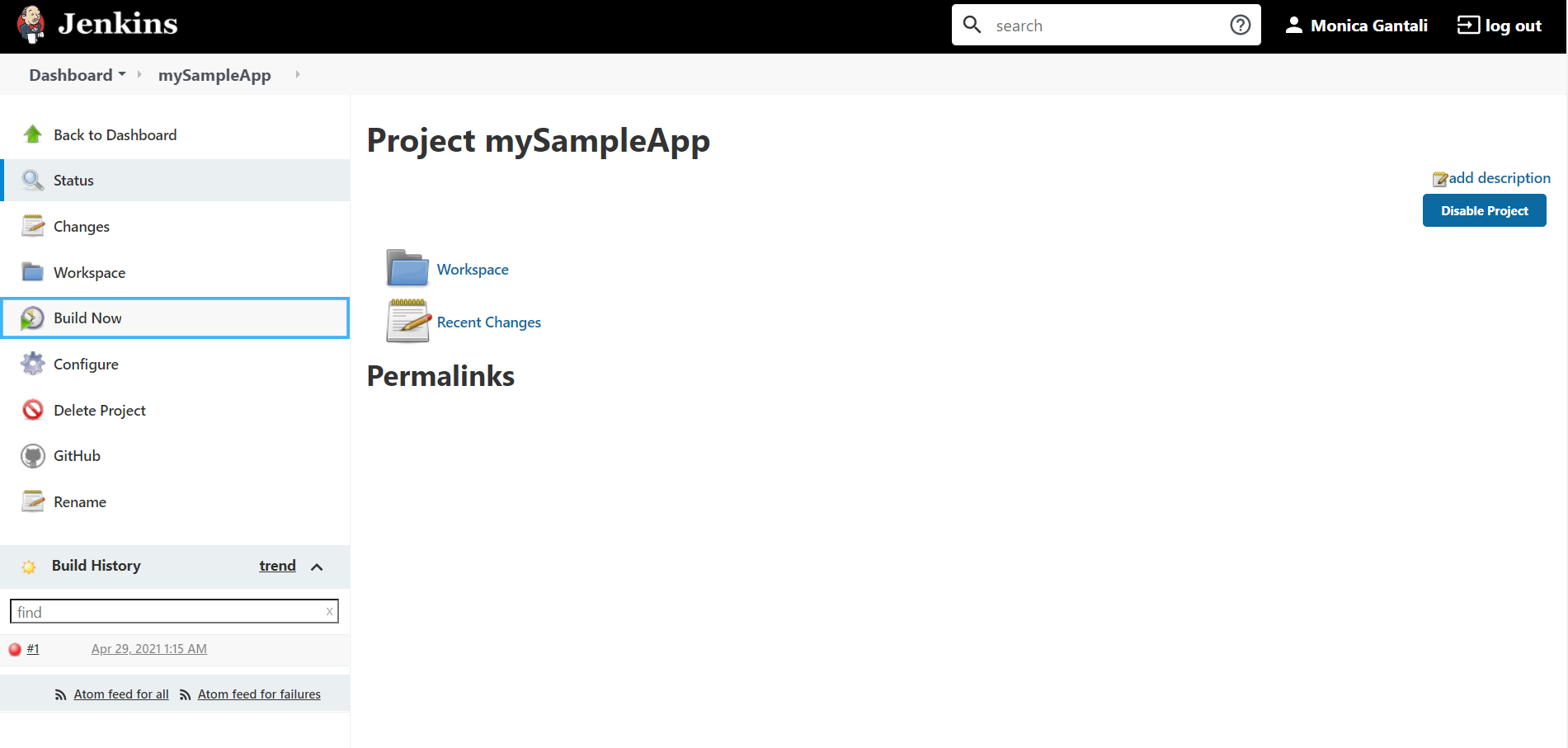


7) Select **Advanced** and enter complete in the **Root Build script** field.



**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) 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.

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

