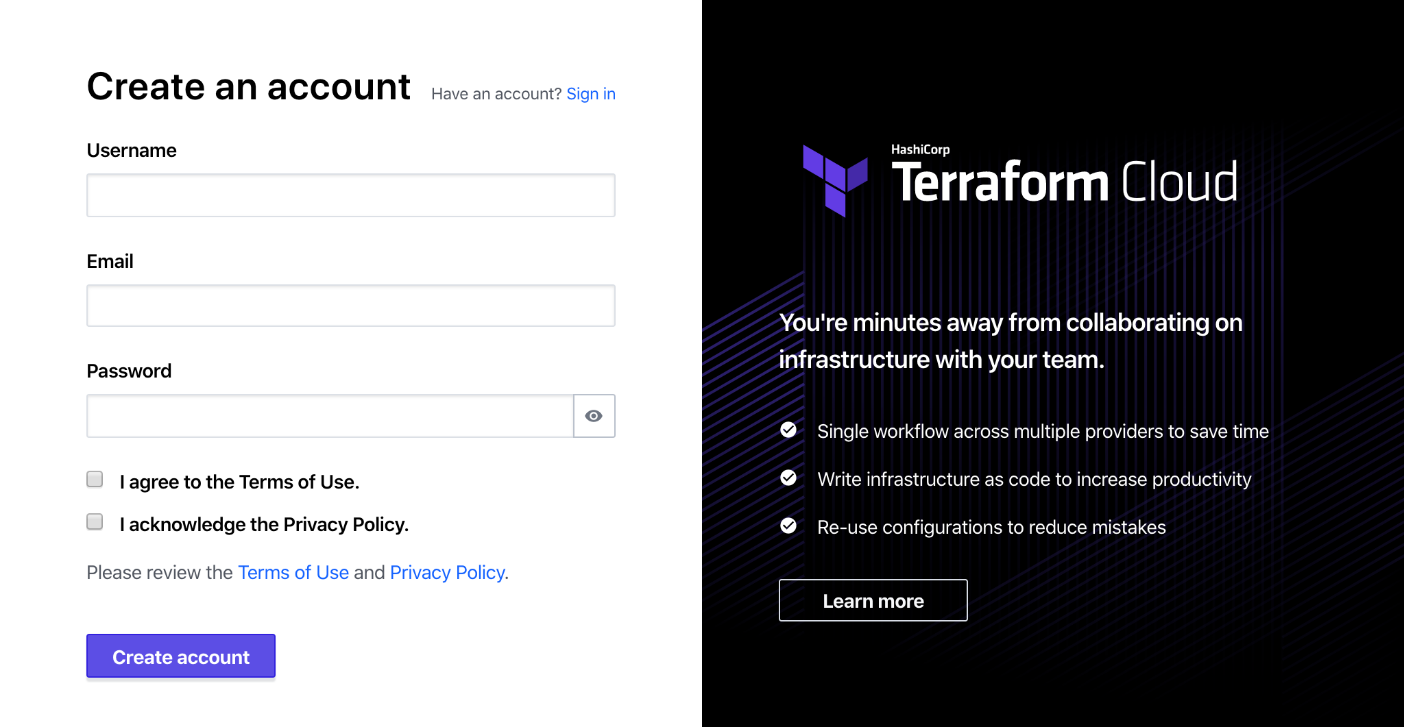
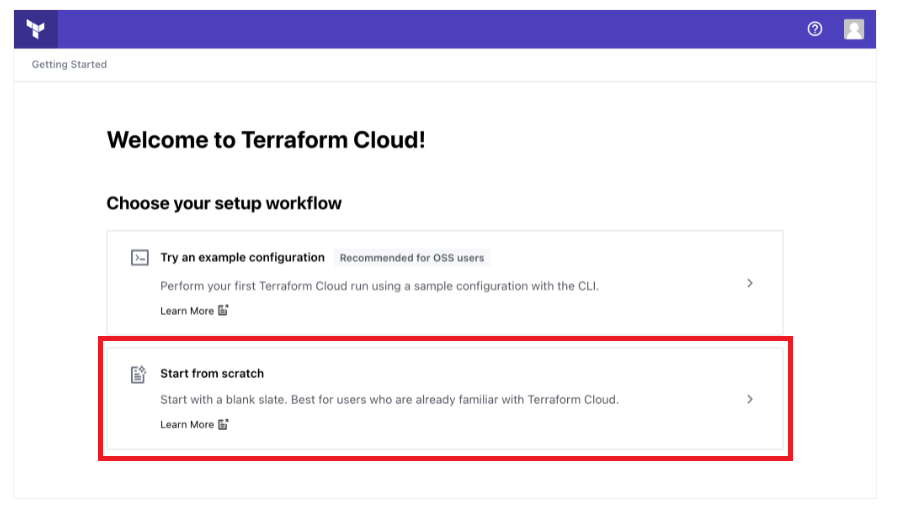
**Creating Infrastructure with Terraform Cloud**

1. **Create an account**

1.1 Visit <https://app.terraform.io/signup/account> and follow the prompts to create a free Terraform Cloud account.

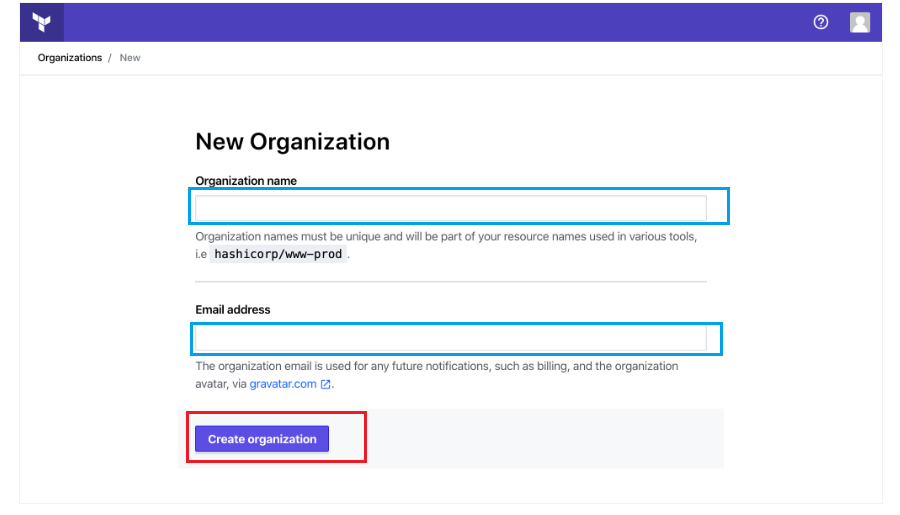


1.2 When you sign up, you will receive an email asking you to confirm your email address. Confirm your email address before moving on. When you click the link to confirm your email address, the Terraform Cloud UI will ask which setup workflow you would like use. Select **Start from scratch**.



**2.Create an organization**

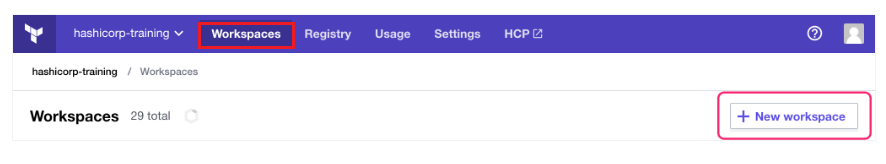
2.1 The next screen will prompt you to create a new organization. Your organization is free, and the members you add will be able to collaborate on your workspaces and share private modules.

Enter an organization name and email address. You can use the same email address that you used for your account.

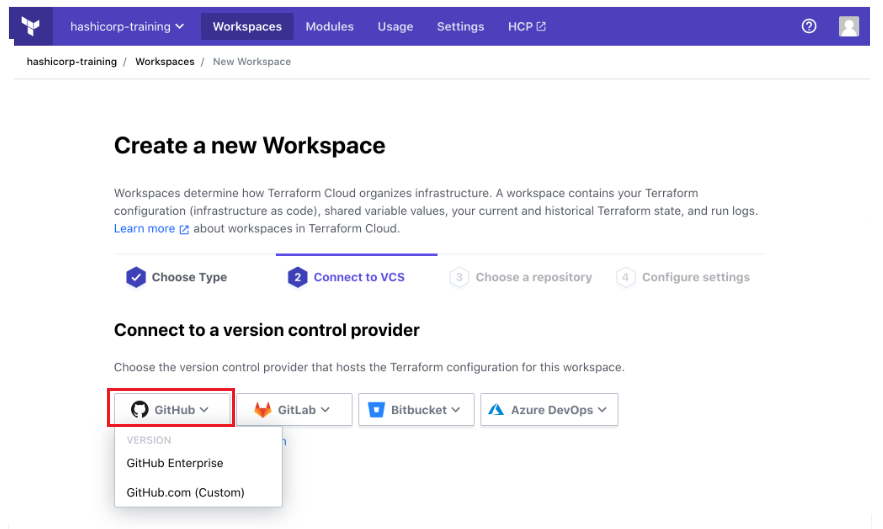
2.2 Then click **Create organization**. Terraform Cloud will prompt you to create a new workspace.

# **3.Create a Workspace:**

# 3.1 Navigate to the **Workspaces** page from the main menu and click **+ New workspace.**



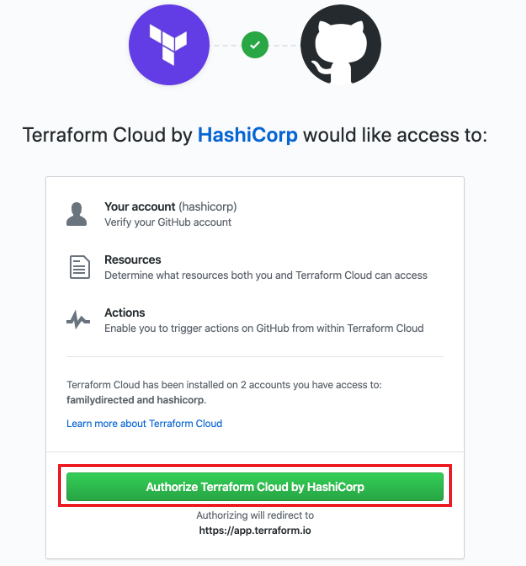
3.2 On the **New Workspace** page, select the **Version control workflow** from the workflow options screen. On the **Connect to VCS** screen, press the **GitHub** button and then choose **GitHub.com** from the drop-down to continue.

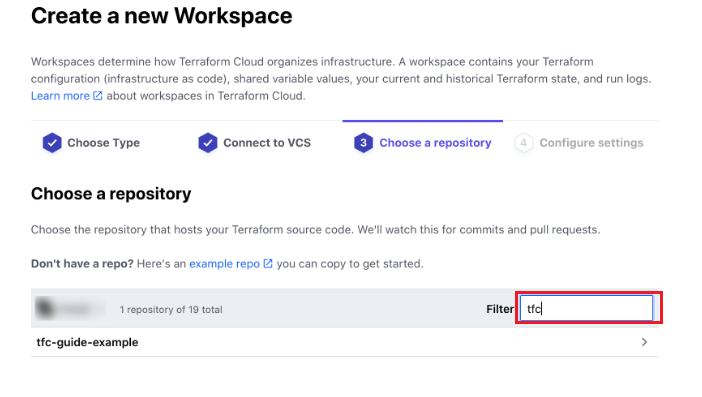


3.3 A new window will open asking you to authorize Terraform Cloud to use your GitHub account. Click the green **Authorize** button to connect Terraform Cloud to your GitHub account.

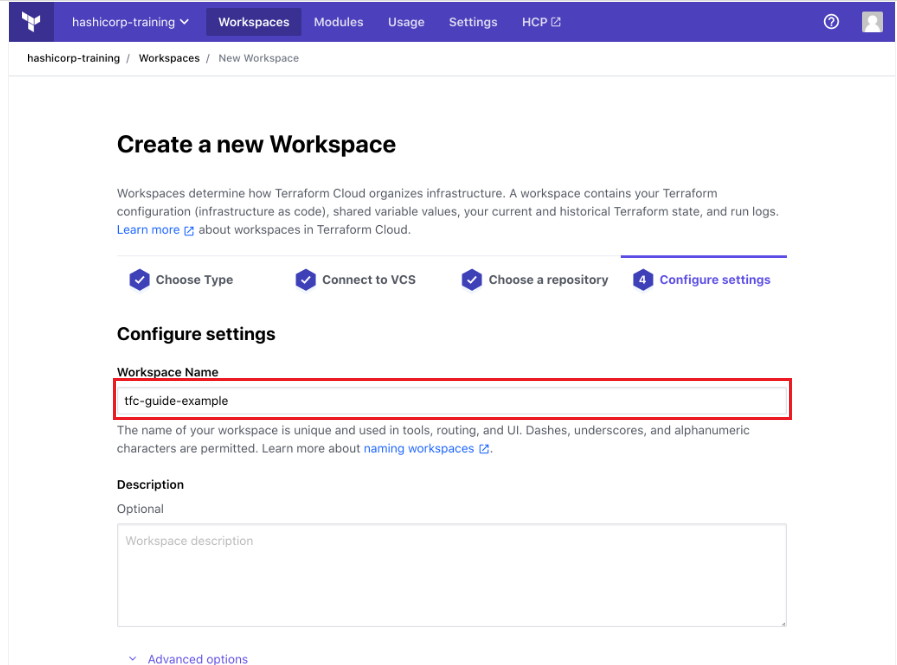
You may be asked to install Terraform Cloud for your GitHub account or organization. If prompted, select your account or organization to install Terraform Cloud.

**4.Choose a repository**

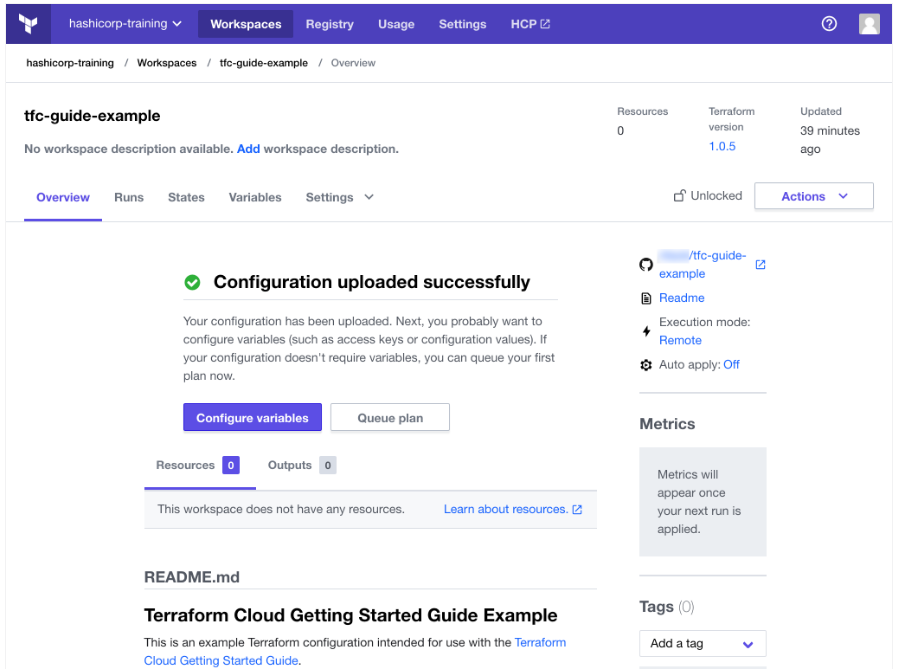
4.1 Next, Terraform will display a list of your GitHub repositories. Choose the repository you forked, called "tfc-guide-example". If you have many GitHub repositories, you may need to filter the list to find the correct one.



4.2 On the next screen, leave the workspace name and **Advanced options** unchanged, and click the purple **Create workspace** button to create the workspace.



It will take a few minutes for Terraform Cloud to connect to your GitHub repository. Once connected, Terraform Cloud will display a notification that your configuration was uploaded successfully.

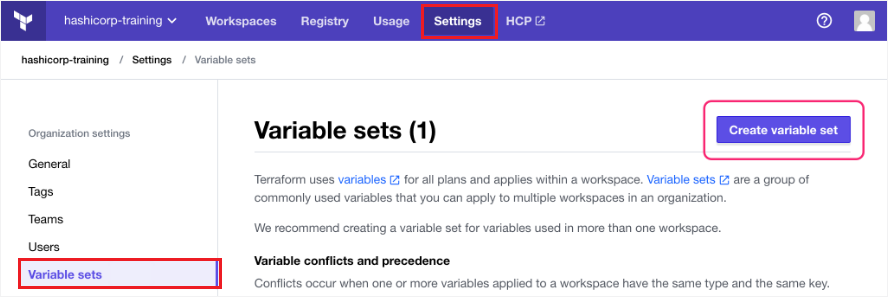


**5.Create a variable set**

You can define both input variables and environment variables in variable sets. Input variables define the values for variables you reference in your configuration, while environment variables typically store provider credentials or modify Terraform’s behavior, such as logging verbosity.

5.1To create a variable set, click on **Settings** in the top menu bar, then **Variable sets** in the left side bar.

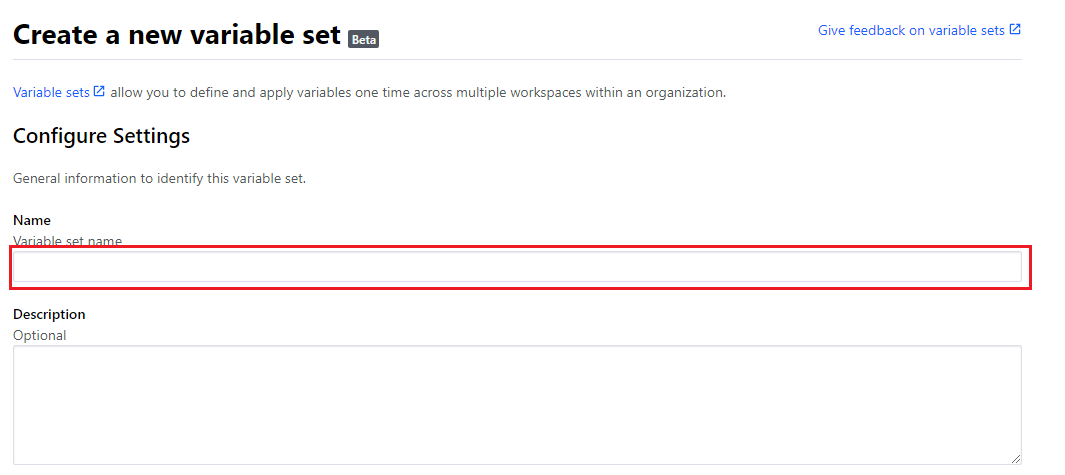
Next, click **Create variable set**.

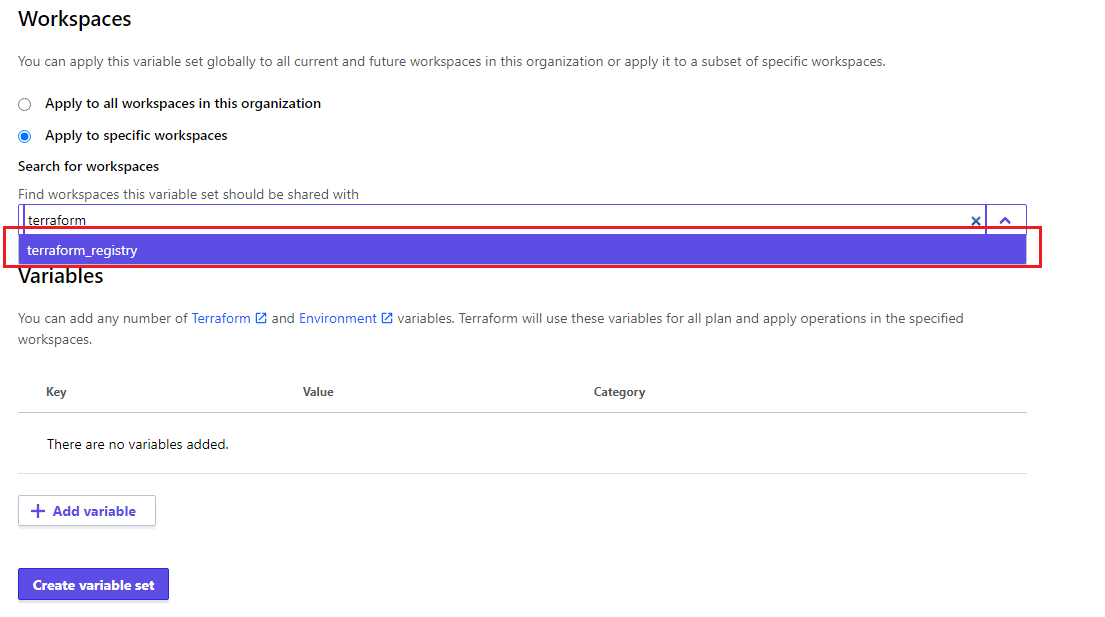


5.2Name the variable set “AWS Credentials”.

You can apply a variable set to all workspaces in your organization, or scope it to specific ones. When using variable sets with credentials, reuse the variable set with care and avoid the global option, since it does not follow least privilege best practices.

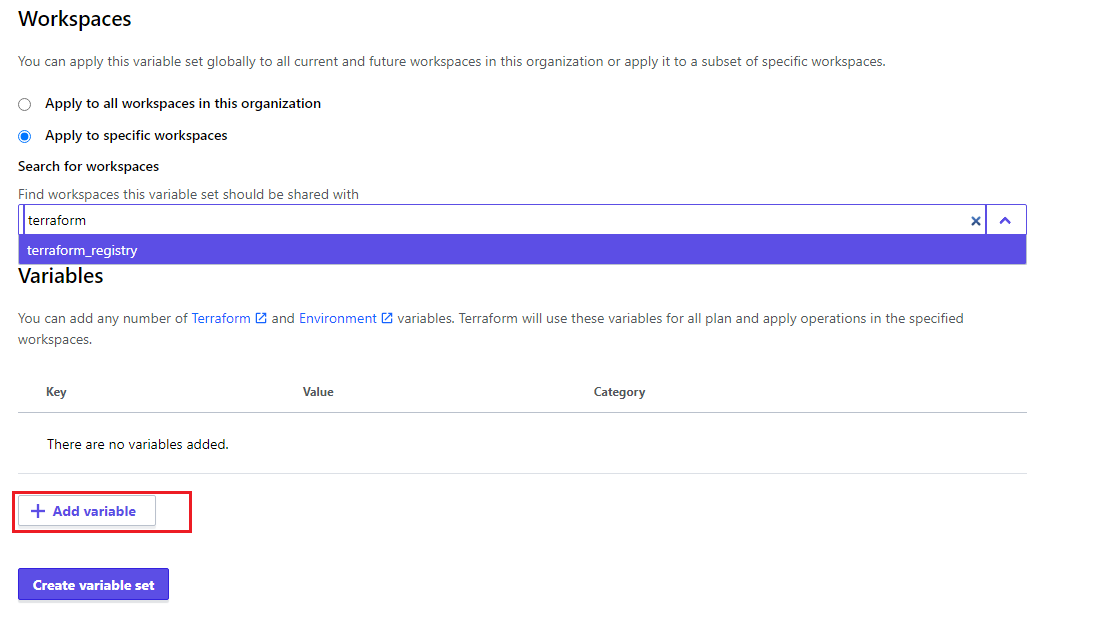
5.3Scroll down to the **Workspaces** section and select **Apply to specific workspaces**. Select the workspace you want to apply the AWS credentials to, in this case the tfc-guide-example workspace.

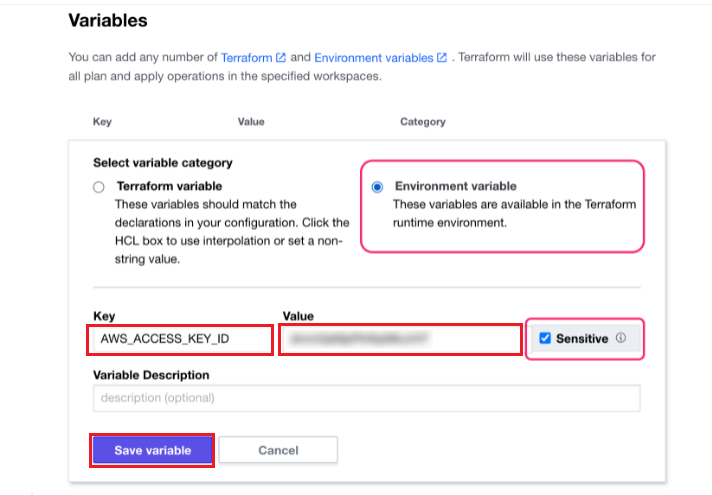




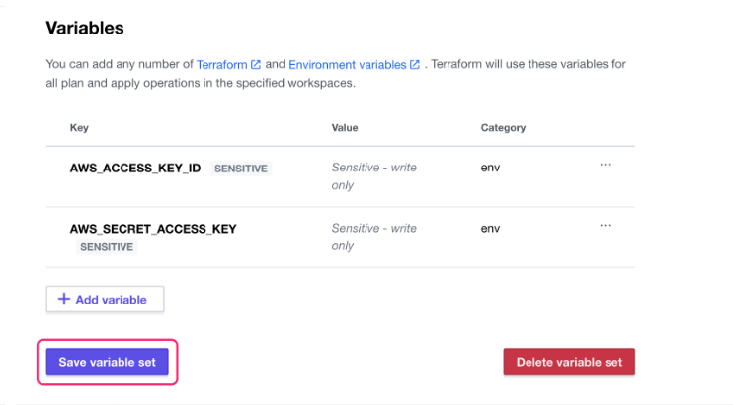
Next, define your AWS credentials as environment variables. Find or create an **Access key ID** and **Secret access key** [in the IAM console](https://console.aws.amazon.com/iam/home?#security_credential).

5.4Click **+Add Variable**. Select the **Environment variable** option. Set the key to AWS\_ACCESS\_KEY\_ID and the value to your AWS **Access Key ID**. Mark it as **Sensitive** and click **Save variable**.

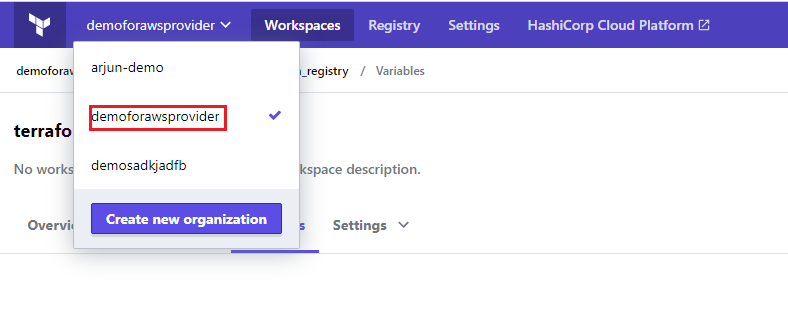




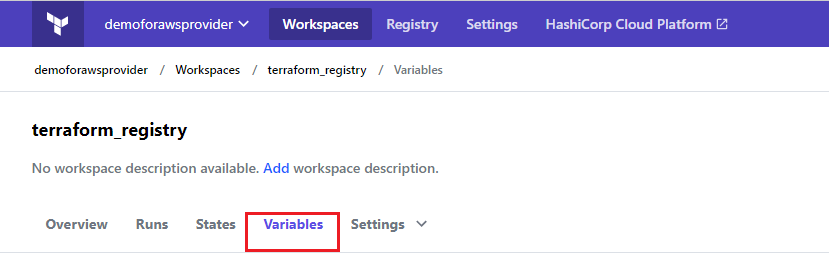
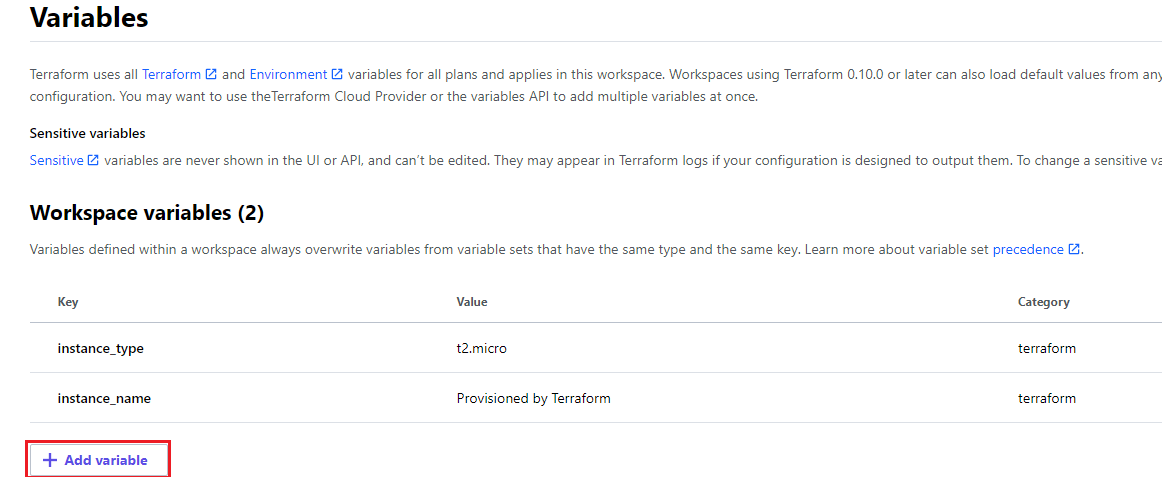
5.5 Now, click **+ Add Variable** again. Define another environment variable named AWS\_SECRET\_ACCESS\_KEY and set it to your AWS **Secret access key**. Mark it as **Sensitive** and click **Save variable**. Now click **Save variable set**.

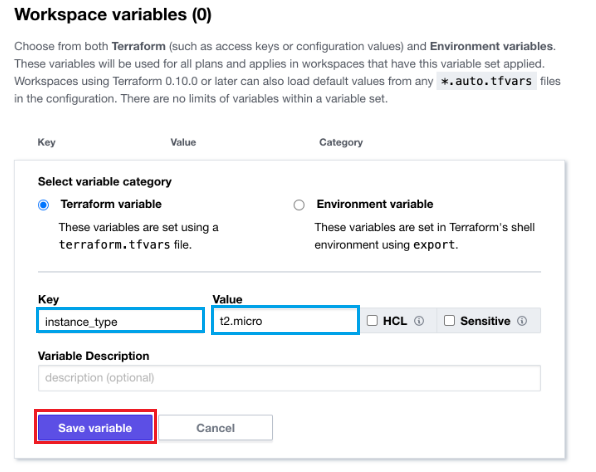


5.6 Next click on our organization name.

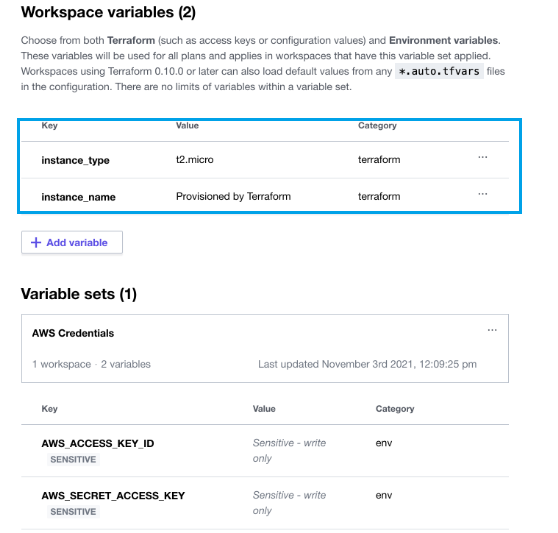


5.7 Find the **Workspace variables** section to set two variables that Terraform will use to configure your instance. Then, click **+ Add variable**.



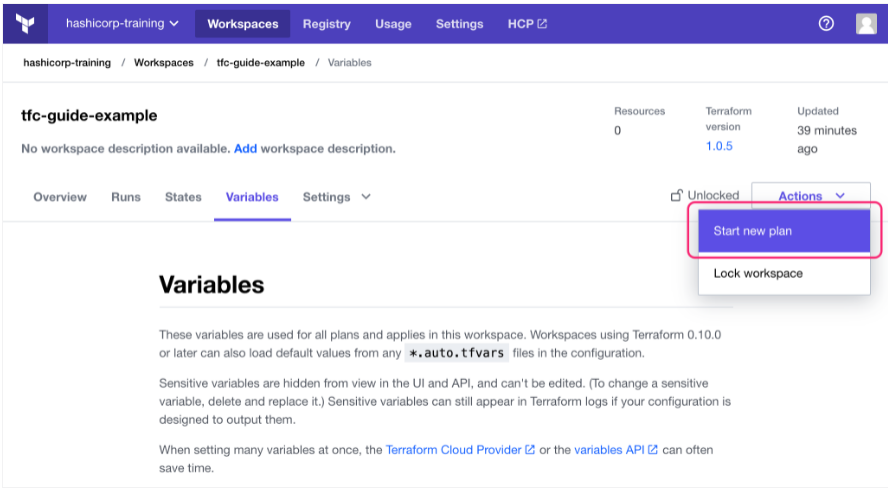
5.8 First, define a Terraform variable by selecting the **Terraform variable** radio button. Set the key to instance\_type and value to t2.micro.

5.9 Then, click **+ Add variable** again. Create another Terraform variable with the key instance\_name and the value Provisioned by Terraform. Click **Save variable**.

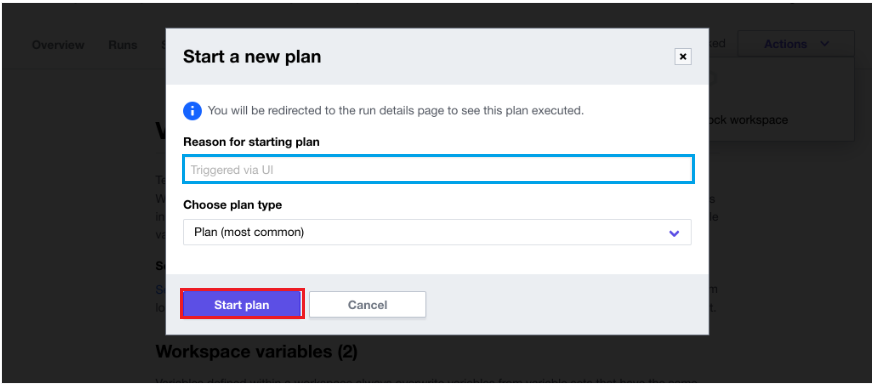


**6.Plan infrastructure changes**

6.1 Now that you have set your Terraform variables, select **Start new plan** from the **Actions** drop down menu.



6.2Choose the **Plan (most common)** option, and provide an optional explanation. Then select the purple **Start plan** button to start a plan. This may take a few minutes.



6.3 This produces an execution plan that describes which actions Terraform will take to change real infrastructure to match the configuration. Terraform will not make any changes until you confirm and apply the plan. This gives you and your team an opportunity to review the planned changes.

