In this mini lab/lesson we are going to provision an EC2 using Hashicorps's terraform.

Reference: <https://www.terraform.io/docs/providers/aws/index.html>

**What is Terraform?**

Terraform is a tool made by Hashicorp for building, changing, and versioning infrastructure safely and efficiently. Terraform can manage existing and popular service providers ( aws, azure, Google cloud) as well as custom in-house solutions.

You can compare **Terraform** to **Cloudformation** . They are simililar but at the same time have differences.

**Steps to provision**

1. Download the terraform binary file <https://www.terraform.io/downloads.html>

If MAC users have homebrew installed on their machine: Just do: brew install terraform Go to step 5

1. Extract the zip file
2. You will see the terraform binary executable file
3. make sure that the terraform binary is available on the PATH.

For Mac/Linux. On the shell/terminal, go to the folder where terraform binary is extracted

echo $"export PATH=\$PATH:$(pwd)" >> ~/.bash\_profile

source ~/.bash\_profile

For Windows users : follow this to add Terraform to PATH <https://stackoverflow.com/questions/1618280/where-can-i-set-path-to-make-exe-on-windows>

1. make a new directory(can be named anything) and go inside the directory

mkdir terraform-july && cd terraform-july

1. Paste this following code to a file called ec2.tf( can be anything.tf)

**minimal viable configuration**

provider "aws" {

access\_key = "ACCESS\_KEY\_HERE"

secret\_key = "SECRET\_KEY\_HERE"

region = "us-east-1"

}

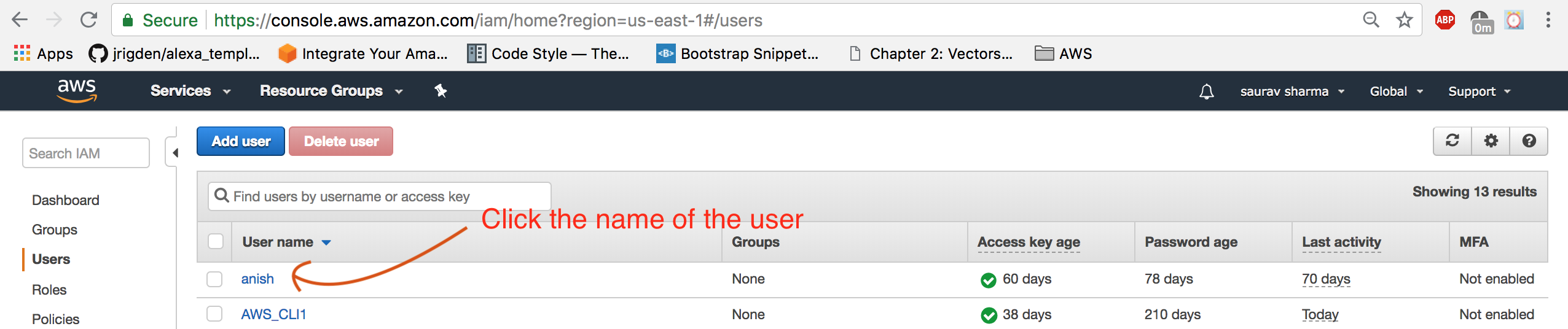
resource "aws\_instance" "example" {

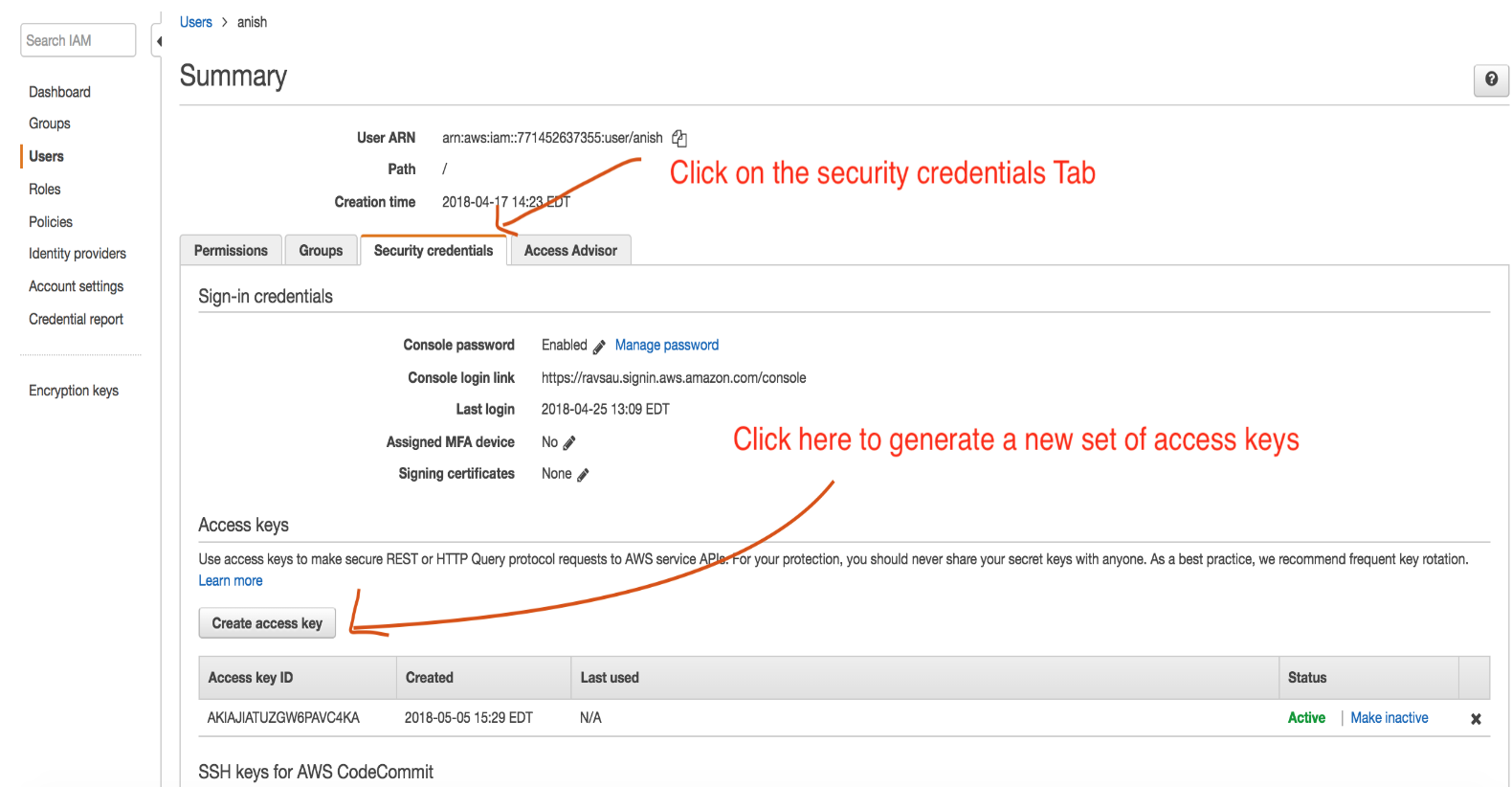
ami = "ami-2757f631"

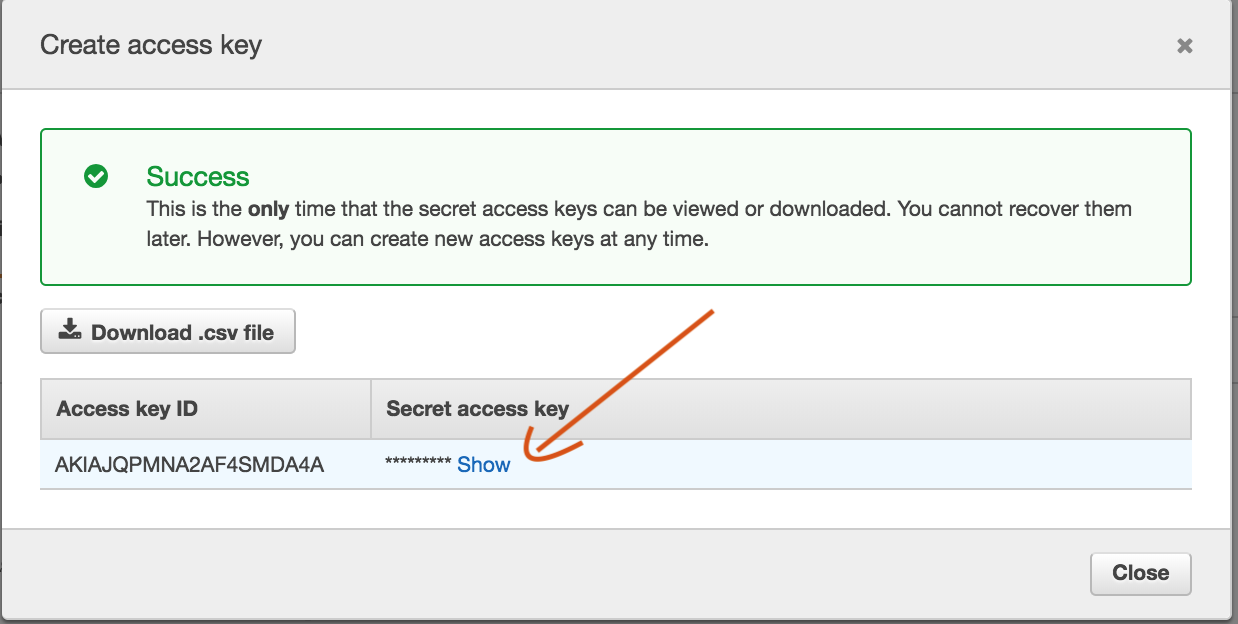
instance\_type = "t2.micro"

}

**Note:**

Replace the access\_key and secret\_access with your AWS IAM user credentials with enough permissions attached. You can go to IAM console on AWS to do this. First, go to the IAM management console [](https://github.com/ravsau/aws-labs/blob/master/images/iam-console.png)

Then Click on the user's name and navigate to the security credentials tab. Click create access keys [](https://github.com/ravsau/aws-labs/blob/master/images/generate-access-keys.png)

Either download the csv file or, click show keys. Now you have both the access\_key and secret\_key required for the terraform code above. [](https://github.com/ravsau/aws-labs/blob/master/images/iam-generated-keys.png)

If you've setup the AWS CLI and have credentials stored , you may skip the credential portion. This is what Hashicorp says "If you simply leave out AWS credentials, Terraform will automatically search for saved API credentials (for example, in ~/.aws/credentials) or IAM instance profile credentials. This option is much cleaner for situations where tf files are checked into source control"

1. initialize the working directory for terraform

terraform init

"The terraform init command is used to initialize a working directory containing Terraform configuration files. This is the first command that should be run after writing a new Terraform configuration or cloning an existing one from version control. It is safe to run this command multiple times."

1. Provision the ec2 with this command

terraform apply

1. Login to the AWS management console and navigate to the EC2 management console. Check if an instance got provisioned
2. From your terminal/command prompt/ shell , destory the resources

terraform destroy

That's it! you installed Terraform and used it to provision an EC2 instance.

Terraform Installation (Linux)

Download Terraform from below link <https://releases.hashicorp.com/terraform/0.13.3/terraform_0.13.3_linux_amd64.zip>

* After Download Terraform directory will generate, Copy Terraform Directory to /usr/local/bin (Terraform Access From any where like global variable)

Ex: sudo mv terraform /usr/local/bin

* Execute the command terraform –v
* Create Main.tf Variable.tf Terraform.tfstate file on your machine