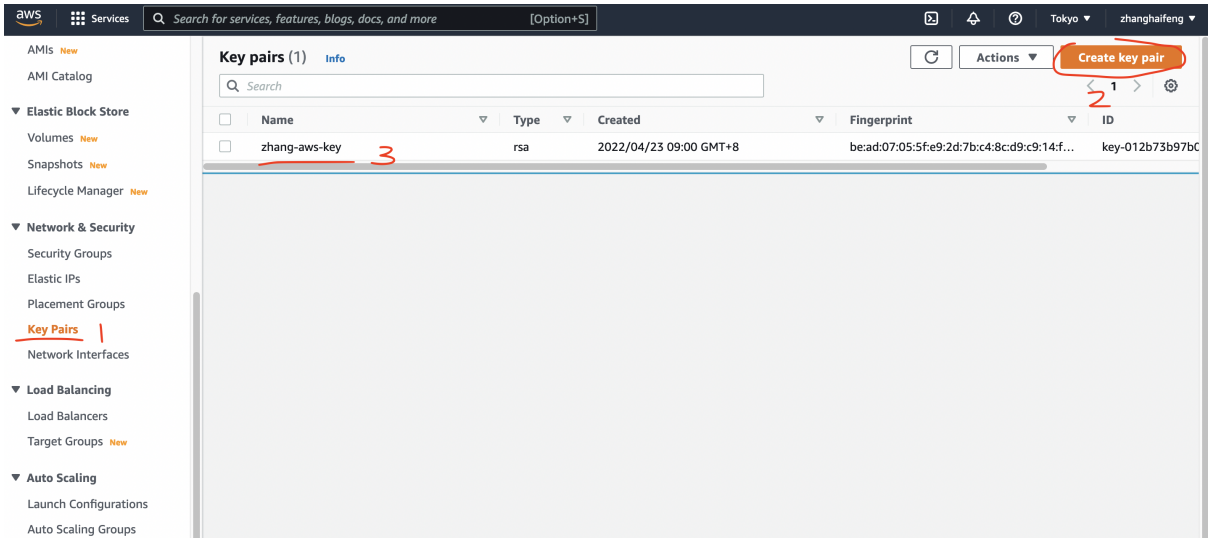
create ec2 instance 2

> 1. Learn about Terraform `Input Variable` basics
>
> - AWS Region
> - Instance Type
> - Key Name
>
> 1. Define `Security Groups` and Associate them as a `List item` to AWS EC2 Instance
>
> - vpc-ssh
> - vpc-web
>
> 1. Learn about Terraform `Output Values`
>
> - Public IP
> - Public DNS
>
> 1. Get latest EC2 AMI ID Using `Terraform Datasources` concept
> 2. We are also going to use existing EC2 Key pair `terraform-key`
> 3. Use all the above to create an EC2 Instance in default VPC

1，做一个keypair，用来远程登录ec2

就是 zhang-aws-key.cer 文件



2，做一个variable.tf 文件，里面存放一些变量

Define Input Variables in Terraform

```
# AWS Region
variable "aws_region" {
  description = "Region in which AWS Resources to be created"
  type = string
  default = "ap-northeast-1"
}

# AWS EC2 Instance Type
variable "instance_type" {
  description = "EC2 Instance Type"
  type = string
  default = "t2.micro"
}

# AWS EC2 Instance Key Pair
variable "instance_keypair" {
  description = "AWS EC2 Key pair that need to be associated with EC2 Instance"
  type = string
  default = "zhang-aws-key"
}
```

3，terraform-setting.tf 文件导入变量

```
terraform {
  //required_version = "~> 0.14"   # which means any version equal & above 0.14 like 0.15, 0.16 etc and < 1.xx
  required_version = "~> 1.2.2"     # which means any version equal & above 1.2.2 like 1.2.3, 1.2.4 etc and < 1.3.xx
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.31.0"
    }
  }
}

# "aws" 与 required_providers里的aws 同名
# profile = "default" 可以不写，它指的是cat $HOME/.aws/credentials 里设定的默认的access key
provider "aws"{
  region  = var.aws_region
  profile = "default"
}
```

4，securitygroups.tf 文件，创建security groups

ingress就是inbound

engress就是 outbound

cidr_blocks = ["0.0.0.0/0"] 是允许所有ip

以下代码建立了 SSH Traffic，Web Traffic 两个 安全组

Resource: aws_security_group

```
# Create Security Group - SSH Traffic
resource "aws_security_group" "vpc-ssh" {
  name        = "vpc-ssh"
  description = "Dev VPC SSH"
  //vpc_id    = aws_vpc.main.id  //vpc_id是可以省略的，这时使用的是aws提供的默认vpc
```

```
  ingress {
    description = "Allow Port 22"
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  egress {
    description = "Allow all ip and ports outboun"
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

# Create Security Group - Web Traffic
resource "aws_security_group" "vpc-web" {
  name        = "vpc-web"
  description = "Dev VPC web"

  ingress {
    description = "Allow Port 80"
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  ingress {
    description = "Allow Port 443"
    from_port   = 443
    to_port     = 443
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  egress {
    description = "Allow all ip and ports outbound"
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
```

5，**ami-datasource.tf - Define Get Latest AMI ID for Amazon Linux2 OS**

AWS AMI

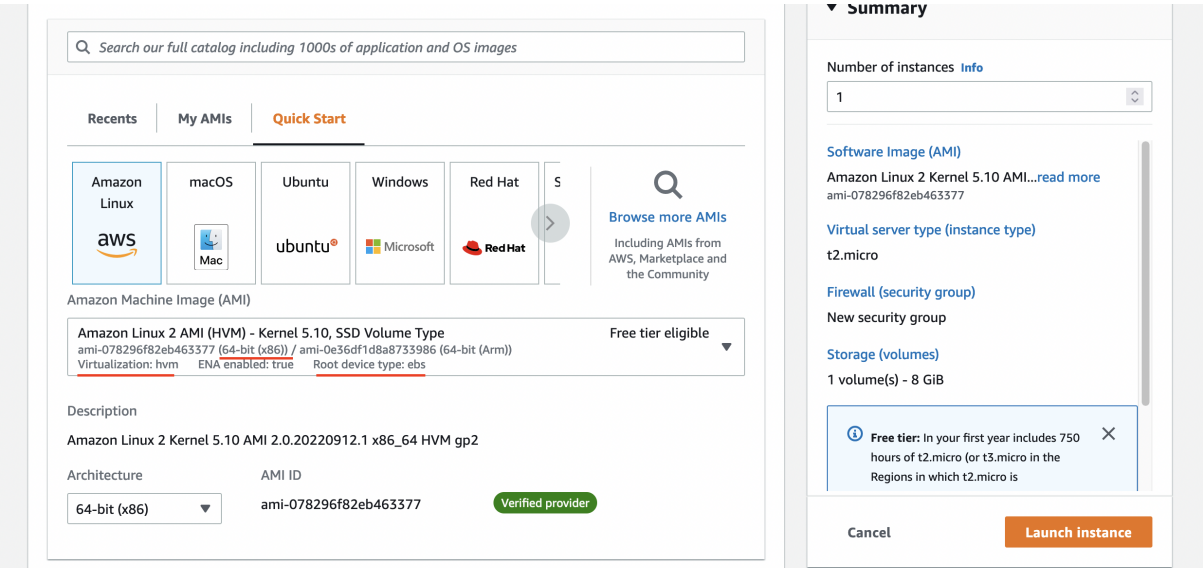
とはインスタンスを起動するのに必要なOSやボリュームの情報などのテンプレートのこと

Data Source: aws_ami

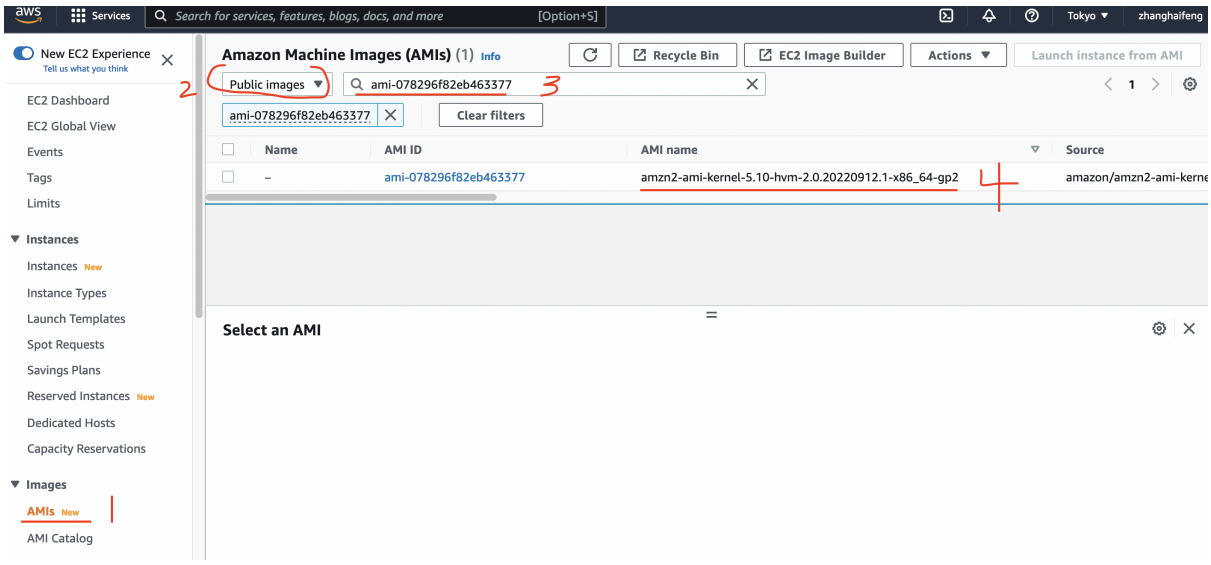
```
# Get latest AMI ID for Amazon Linux2 OS
# Get Latest AWS AMI ID for Amazon2 Linux

data "aws_ami" "amzlinux2" {

  most_recent = true
  owners = [ "amazon" ]

  filter {
    name = "name"
    values = [ "amzn2-ami-kernel-5.10-hvm-*-gp2" ]
  }
```

```
  filter {
    name = "root-device-type"
    values = [ "ebs" ]
  }
  filter {
    name = "virtualization-type"
    values = [ "hvm" ]
  }
  filter {
    name = "architecture"
    values = [ "x86_64" ]
  }
}
```

上面filter里提到的信息在这里



然后复制AMI ID 到AMIS 查找AMI name



6，将前面文件的内容导入ec2-setting.tf

```
resource "aws_instance" "myec2vm" {
  ami = data.aws_ami.amzlinux2.id
  instance_type = var.instance_type
  user_data = file("${path.module}/nginx-install.sh")
  key_name = var.instance_keypair
  vpc_security_group_ids = [aws_security_group.vpc-ssh.id, aws_security_group.vpc-web.id]
  tags = {
    "Name" = "EC2 Demo 2"
  }
}
```

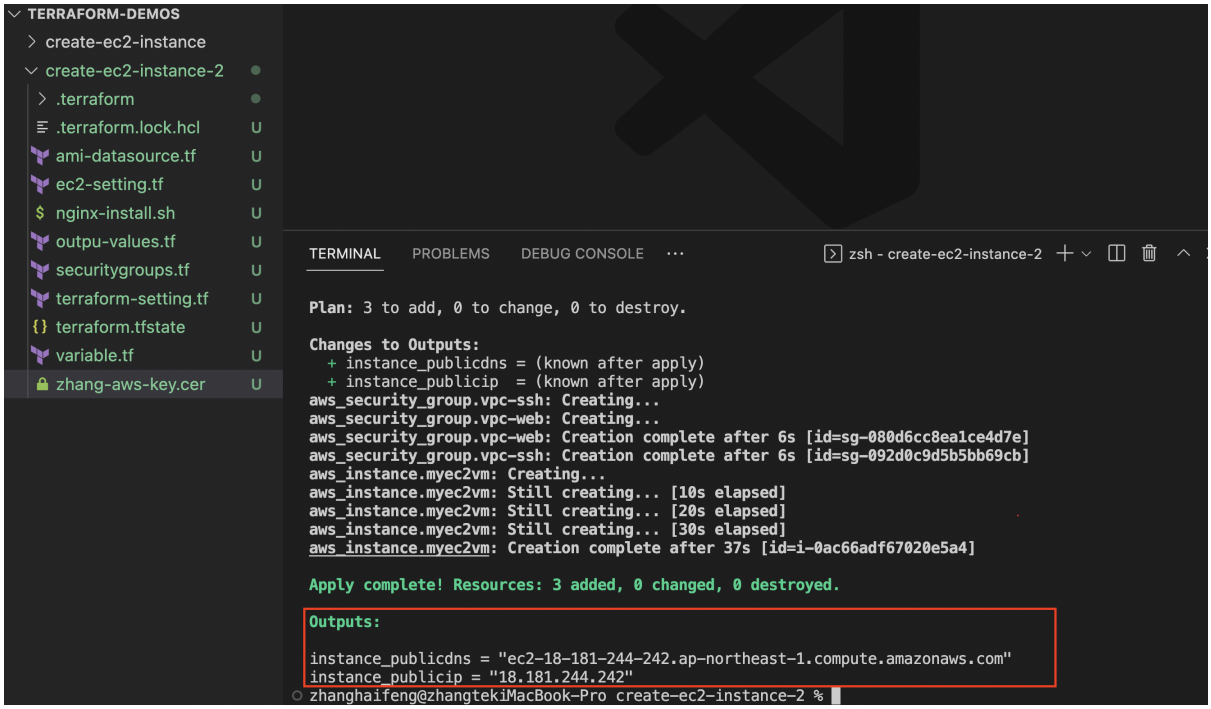
7，建立o**utpu-values.tf 文件**

Define Output Values

```
# Terraform Output Values
output "instance_publicip" {
  description = "EC2 Instance Public IP"
  value = aws_instance.myec2vm.public_ip
}

output "instance_publicdns" {
  description = "EC2 Instance Public DNS"
  value = aws_instance.myec2vm.public_dns
}
```

8，create-ec2-instance-2文件夹里执行命令

```
terraform init     // 自动根据terraform-setting.tf  生成 [.terraform]跟[.terraform.lock.hcl] 两个文件

terraform validate // 验证配置是否正确

terraform plan     // 列出将要做的事，配置是什么

terraform apply  -auto-approve

// 执行这条命令后，ec2实例建立、控制台会打印出下图的内容，outpu-values.tf 文件定义的内容
```



9，验证是否正确建立

```
1，看是否能远程登录
ssh -i zhang-aws-key.cer ec2-user@18.181.244.242

2，检查网址
http://18.181.244.242/index.html
http://18.181.244.242/app1/index.html
http://18.181.244.242/app1/metadata.html
```

10，终结ec2实例,aws管理画面的实例跟security groups会被删除

```
# Terraform Destroy
terraform destroy -auto-approve

# Clean-Up Files
rm -rf .terraform*
rm -rf terraform.tfstate*
```

*github link

https://github.com/no744634936/terraform/tree/main/2-create-ec2-instance