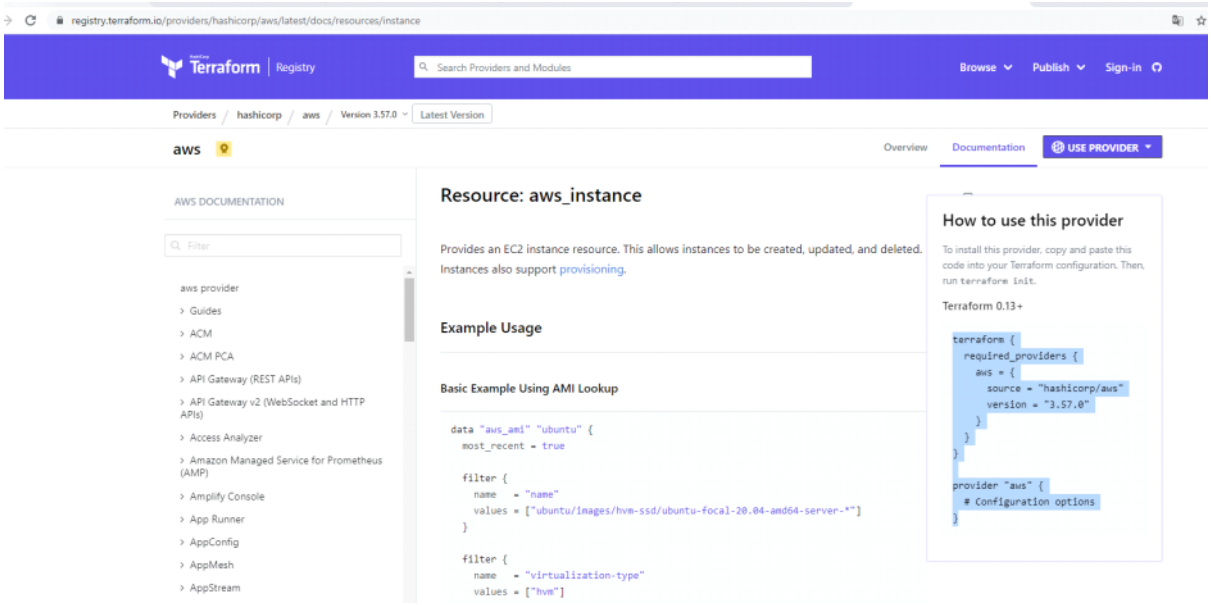
# Terraform - 2

Thursday, 2 September 2021 20.00

[Graphviz Online (dreampuf.github.io)](https://dreampuf.github.io/)

**Ne yaptigimizi kisaca özetlersek**

- **Googledan aws instance Terraform yazip karsimiza cikan adresten gerekli argumanlari aldik**
- **3 ayri blok olusturduk bunlkardan ilki terraform idi**
- **Sonraki provider idi ne calistiracagimizi ve hangi regionda calistiracagimizi yaziyorduk**
- **Son olanda resource yani ousturdugumuz kaynaklar di**
- **Kaynaklarimizi da yazdiktan sxonra terraform init comutunu girdik**
- **Komutu girdigiizde .terraform isimli dosya geldi**
- **Daha sonra terraform plan dedik ve gelen plani terraform apply komutu ile onayladik**




```
main.tf
terraform-aws > main.tf > resource "aws_s3_bucket" "tf-s3"
1  terraform {
2    required_providers {
3      aws = {
4        source = "hashicorp/aws"
5        version = "3.57.0"
6      }
7    }
8  }
9
10 provider "aws" {
11   # Configuration options
12   region = "us-east-1"
13 }
14
15 resource "aws_instance" "tf-ec2" {
16     ami = "ami-087c17d1fe0178315"
17     instance_type = "t2.micro"
18     key_name = "ramiz"
19
20     tags = {
21         "Name" = "created-by-tf"
22     }
23 }
24
25 resource "aws_s3_bucket" "tf-s3" {
26     bucket = "ramiz-tf-test-bucket"
27     acl = "private"
28 }
```

2. **Hansona basliyoruz**

**Main tf deki bazi seyleri variable kullanarak almaya calisacagiz**

Variable yani degisken olarak adlandiriliyor Bu file de bir cok kisi file kullanmak isteyebilir farkli bir sey kurmak isterse variablelerde kucuk degisiklikler ile islemini kolayca yapmis olur

# Hands-on Terraform-02 : Terraform Variables, Conditionals, Loops, Data Sources.

Purpose of the this hands-on training is to give students the knowledge of variables, conditionals, loops and data sources in Terraform.

## Learning Outcomes

At the end of the this hands-on training, students will be able to;
- Use variables, conditionals, loops and data sources with Terraform

### Variables

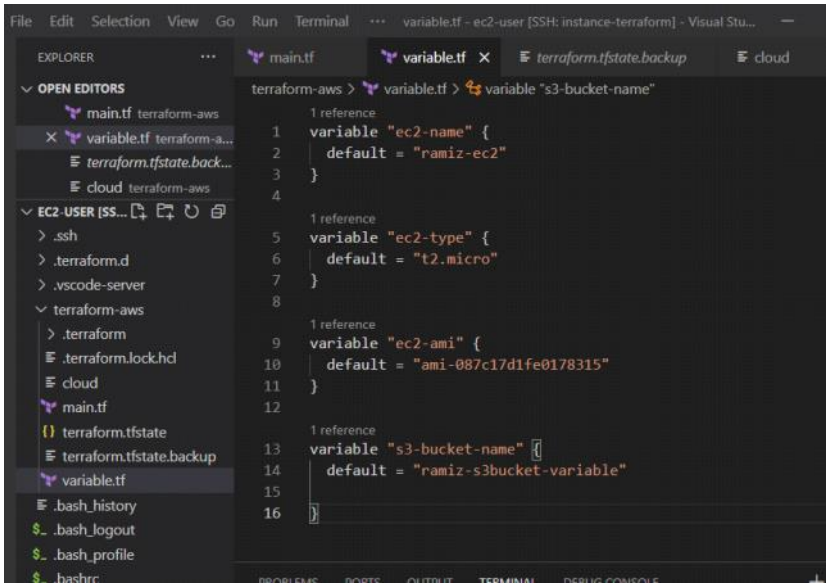
- Make the changes in the `main.tf` file.

```bash
provider "aws" {
  region  = "us-east-1"
}
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "3.38.0"
    }
  }
}
variable "ec2-name" {
  default = "oliver-ec2"
}
variable "ec2-type" {
  default = "t2.micro"
}
variable "ec2-ami" {
  default = "ami-0742b4e673072066f"
}
resource "aws_instance" "tf-ec2" {
  ami           = var.ec2-ami
  instance_type = var.ec2-type
  key_name      = "mk"
  tags = {
    Name = "${var.ec2-name}-💻🎯🚀"
  }
}
variable "s3-bucket-name" {
  default = "oliver-s3-bucket-variable-addwhateveryouwant"
}
resource "aws_s3_bucket" "tf-s3" {
  bucket = var.s3-bucket-name
  acl    = "private"
}
output "tf-example-public_ip" {
  value = aws_instance.tf-ec2.public_ip
}
output "tf-example-private-ip" {
  value = aws_instance.tf-ec2.private_ip
}
output "tf-example-s3" {
  value = aws_s3_bucket.tf-s3[*]
}
```

```bash
terraform apply
```

Yeni bir tf uzantili variable isimli bir file olusturduk

- Create a file name `variables.tf`. Take the variables from `main.tf` file and paste into "variables.tf".




```
output "tf-example-s3" {
  value = aws_s3_bucket.tf-s3[*]
```

Bu komut ile s3 un ana etributleri geliyor

Ve asagidaki komutlari uyguluyiruz

```bash
terraform validate     # file dogru mu
terraform fmt          # formatladik
terraform apply        # ve onayladik
```

- Comment the variable of `ec2-name` with "ctrl+k+c" in vscode. (comment out = ctrl+k+u) Then make the changes in the `main.tf` file.

```bash
locals {
  instance-name = "oliver-local-name"
}
resource "aws_instance" "tf-ec2" {
  ami           = var.ec2-ami
  instance_type = var.ec2-type
  key_name      = "mk"
  tags = {
    Name = "${local.instance-name}-come from locals"
  }
}
```

- ```variable lara benzeyen ancak bize özel bir durumda lokal var file icinde sirketin ismini kullanacaksiniz bunu da ayri bur yere degil de locals yarazak yapa biliyoruz

- Main .tf dosyamiza gidip locals degiskenimizi tanimladik lokals olarak instance name icin bir tanimlama yaptik Daha sonra tag e gittik ve ---  local. Yukarda tanimladigimiz ismi yazdik "${local.---- }"----- tirnak icerisinde oldugu icin yani string oldugu icin bu sekilde bir uygulama yaptik

- Ve daha sonra instance miza gelerek bu degiskenni tanimladik

Ayrica variable file icerisne gelip ec2 ile ilgil verdigimiz satiri ctrl + k + c ile yorum satir haline getirdik

Ve degiikligi kaydetmek icin terraform aopply yaptik

A100 satirlik bir filemiz olabilr isimizi kolaylastirmak maksadiyla degisiklik yapmayi

kolaylastiriyor cok defa yapilan degisiklikler icin bulunmaz kaftan

```
main.tf   variable.tf   terraform.tfstate.backup   cloud
terraform-aws > main.tf > resource "aws_instance" "tf-ec2" > tags > N
14
15  locals {
      1 reference
16    instance-name = "ramazan-ec2-name"
17  }
18
    2 references
19  resource "aws_instance" "tf-ec2" {
20    ami           = var.ec2-ami
21    instance_type = var.ec2-type
22    key_name      = "ramiz"
23
24    tags = {
25      "Name" = "${local.instance-name}-come from locals "
26    }
27  }
28
```

```
main.tf   variable.tf   terraform
terraform-aws > variable.tf > ...
1  # variable "ec2-name" {
2  #   default = "ramiz-ec2"
3  # }
4
   1 reference
5  variable "ec2-type" {
6    default = "t2.micro"
7  }
8
   1 reference
```

- A `local` value assigns a name to an expression, so you can use it multiple times within a module without repeating it.
- Run the command `terraform plan`
```bash
terraform plan
```
- Run the command `terraform apply` again. Check the EC2 instance's Name tag column.
```bash
terraform apply
```
- Go to the `variables.tf` file and comment the s3 bucket name variable's default value.
```tf
variable "s3-bucket-name" {
#   default     = "oliver-new-s3-bucket-addwhateveryouwant"
}
```

```variable tanimlama yöntemleri ;

1. Daha sonra .variable file mizda bir satirimizi komut satiri yaptik ve sonrasinda terraform plan dedik

```
main.tf   variable.tf   terraform.tfstate.backup
terraform-aws > variable.tf > variable "s3-bucket-name"
8
   1 reference
9  variable "ec2-ami" {
10   default = "ami-087c17d1fe0178315"
11 }
12
   1 reference
13 variable "s3-bucket-name" {
14 #   default = "ramiz-s3bucket-variable"
15
16 }
```

Terrafor s3 bucket in ne oldugunu bilmiyor ve soruyor
```

```
main.tf  ×  variables.tf
terraform-aws > main.tf > locals
28
    2 references
29  resource "aws_s3_bucket" "tf-s3" {
30    bucket = var.s3-bucket-name
31    acl    = "private"
32  }
33
34  output "tf-example-public-ip" {
35    value = aws_instance.tf-ec2.public_ip
36  }
37
38  output "tf-example-s3-meta" {
39    value = aws_s3_bucket.tf-s3.region
40  }
PROBLEMS   OUTPUT   TERMINAL   ...   terraform
[ec2-user@ip-172-31-85-232 terraform-aws]$ terraform plan
var.s3-bucket-name
  Enter a value:
```

2. Ayrica terraform plan -var="variable icerisinde tanimladigimiz isim = vermek istedigumiz isim "

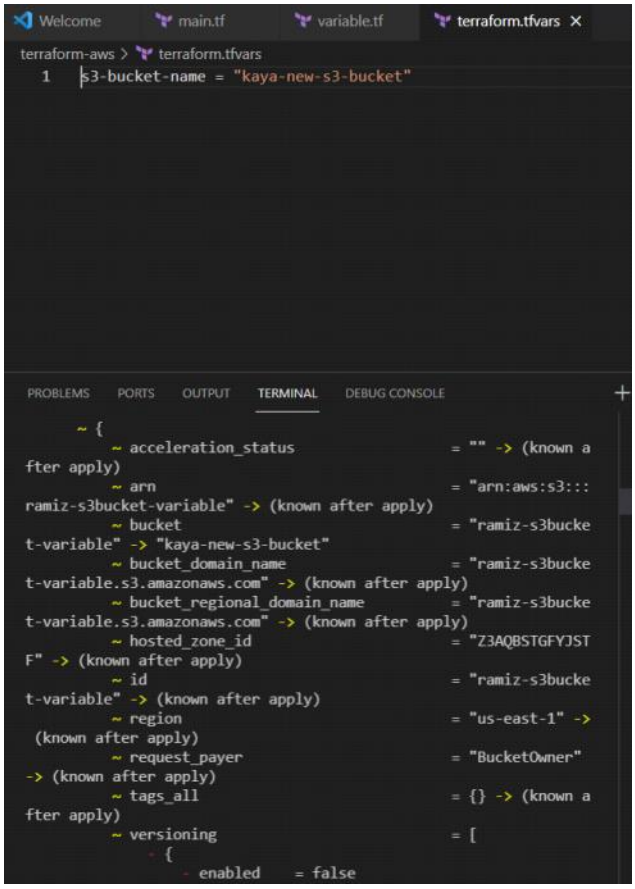
```
12
    1 reference
13  variable "s3-bucket-name" {
14  #   default = "oliver-s3-bucket-variable-addwhateveryouwant-newest"
15  }
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   terraform
[ec2-user@ip-172-31-85-232 terraform-aws]$ terraform plan -var="s3-bucket-name=oliver-new-s3-buck
et-2"
```

3. Uncu yontem ramazan.tfvars isimli bir file olusturduk
daha sonra degisken in icine isimlendirmek istedigimiz resource yazdik
Ve terraform plan --var-file= "hangi file dan aldigimizi belirtiyoruz."

```
File  Edit  Selection  View  Go  Run  ...   ramazan.tfvars - ec2-user [SSH: instance-terraform] - Visual S...
EXPLORER   main.tf   variable.tf   ramazan.tfvars ×   terraform.tfstate.bac
OPEN EDITORS                          terraform-aws > ramazan.tfvars
  main.tf terraform-aws                 1  s3-bucket-name = ramazan-new-s3-bucket
  variable.tf terraform-a...
  ramazan.tfvars terraf...
  terraform.tfstate.back...
  cloud terraform-aws
EC2-USER [SSH: INSTANCE-TER...        PROBLEMS   PORTS   OUTPUT   TERMINAL   DEBUG CONSOLE
  .ssh                                [ec2-user@ip-172-31-21-63 terraform-aws]$ terraform plan --var-f
  .terraform.d                        ile= "ramazan.tfvars"
  .vscode-server
  terraform-aws
    .terraform
    .terraform.lock.hcl
    cloud
    main.tf
    ramazan.tfvars
    terraform.tfstate
    terraform.tfstate.backup
    variable.tf
```

Daha sonra apply komutunu uygulamamizi bekler

Ayriva dosyamizin ismini terraform .tf olarak degistiriisek ve terraform plan yaparsak gidip main dosyasdaki default u almaz burda tanimladigimiz degiskene öncelik verir.



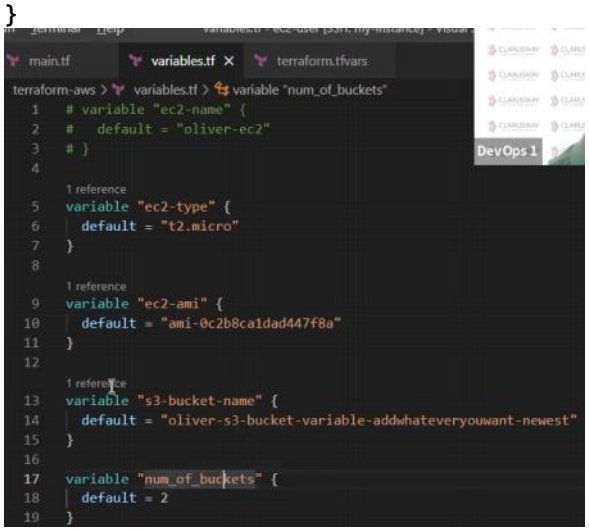

```bash
terraform plan
```
- You can define variables with `-var` command
```bash
terraform plan -var="s3-bucket-name=oliver-new-s3-bucket-2"
```
- Create a file name `oliver.tfvars`. Add the followings.
```bash
s3-bucket-name = "oliver-s3-bucket-newest"
```
- Run the command belov.
```bash
terraform plan --var-file="oliver.tfvars"
```
- Go to the `variables.tf` file and comment out the s3 bucket name variable's default value..
```tf
variable "s3-bucket-name" {
  default     = "oliver-new-s3-bucket"
}
```
- Run terraform apply --var-file="oliver.tfvars" command.
```bash
terraform apply --var-file="oliver.tfvars"
```
- Run terraform apply command.
```bash
terraform apply
```
- Change the name of oliver.tfvars to terraform.tfvars.
- Run terraform apply command.

### Conditionals and Loops

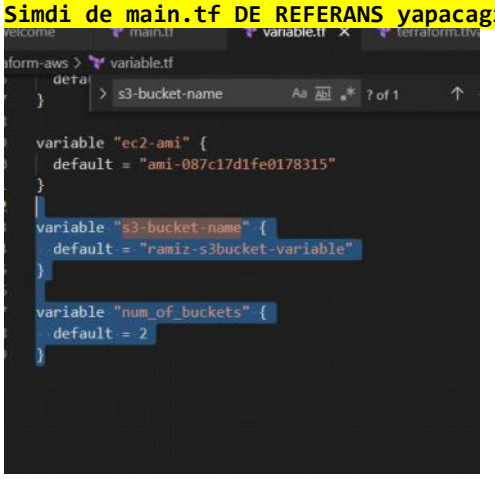
- Count and count.index  ------count ifadesi ile birden fazla instansce kurabiliyoruz

- Go to the `variables.tf` file and create a new variable.

```bash
variable "num_of_buckets" {
  default = 2
}
```




**Simdi de main.tf DE REFERANS yapacagiz**

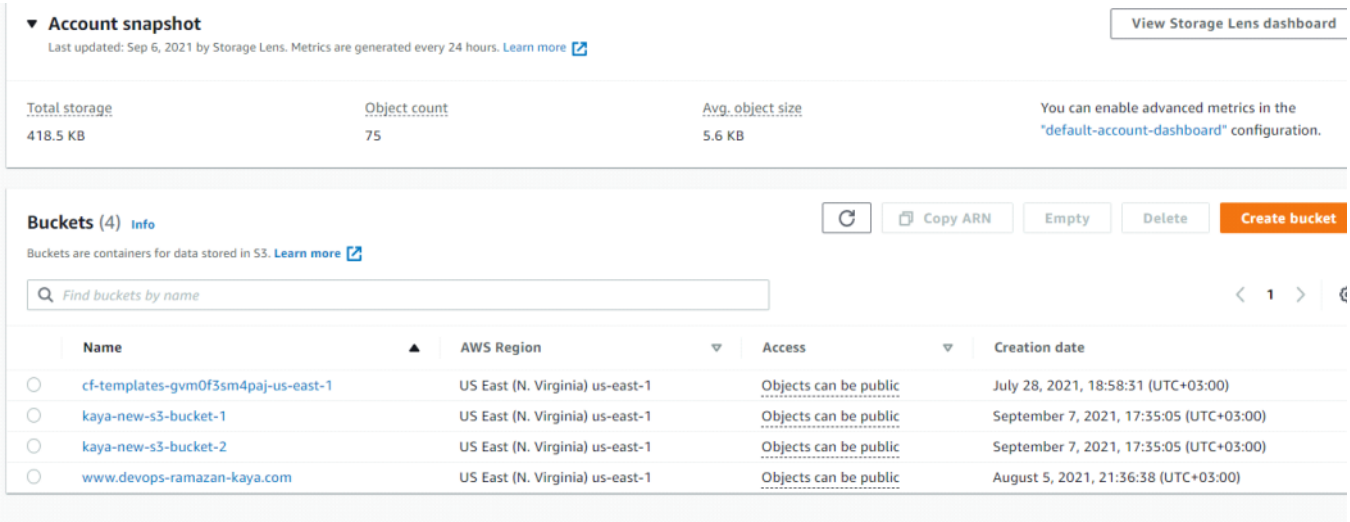




**Bucket = tirnak icerisinde kullandigimiz icin "${}" seklinde kullaniyoruz ilk olarak variablede verdigimiz degiskene yani s3-bucket-name var ile aliyoruz. Son ra count ile indexleme yapiyoruz ve her bierine uniq isimler ver,esi gerektigi icin +1 ile destekliyoruz**

```
- Go to the `main.tf` file, make the changes in order.
```bash
resource "aws_s3_bucket" "tf-s3" {
  bucket = "${var.s3-bucket-name}-${count.index}"
  acl    = "private"
  count = var.num_of_buckets
}
```
```bash
terraform plan
```
```

```
```bash
terraform apply
```
- Check the S3 buckets from console.
- Conditional Expressions.




2 adet s3 bucket olustugunu görecegiz

- Go to the `main.tf` file, make the changes in order.
```bash
resource "aws_s3_bucket" "tf-s3" {
  bucket = "${var.s3-bucket-name}-${count.index}"
  acl    = "private"
  # count = var.num_of_buckets
  count = var.num_of_buckets != 0 ? var.num_of_buckets : 3
}
```
```bash
terraform plan
```

```
resource "aws_s3_bucket" "tf-s3" {
  bucket = "${var.s3-bucket-name}-${count.index + 1}"
  acl    = "private"
  # count = var.num_of_buckets
  count = var.num_of_buckets != 0 ? var.num_of_buckets : 3

}
```

Dökumantasyondan bu sekilde alabiliyoruz anlami su var number_of_buckets esit degilse != 0 ? Var.number_of_buckets : 3 tane kur

Bu bir kosullu ifadedir. var.num_of_buckets != 0 ? Var:num_of_buckets : 3
- Functions.

- Eger soldaki sart ifadesi saglaniyorsa yani True ise sagdaki kosulu yerine getir False ise : dan sonra ki kosulu yerine getir.

Builtin Functions dan bu ve benzer seyleri bulabiliriz

- Go to the `variables.tf` file again and add a new variable.
```bash
variable "users" {
  default = ["spring", "micheal", "oliver"]
}
```
- Go to the `main.tf` file make the changes. Change the IAM role and add IAMFullAccess policy User olusturacagimiz icin bu yetkiyi veriyoruz
```

User

Variablelerimiziz tanimladik

```
variable "s3-bucket-name" {
  default = "ramiz-s3bucket-variable"
}

2 references
variable "num_of_buckets" {
  default = 2
}

variable "users" {
  default = ["santino", "michel", "Kayar"]
}
```

```bash
resource "aws_s3_bucket" "tf-s3" {
  # bucket = "var.s3-bucket-name.${count.index}"
  acl = "private"
  # count = var.num_of_buckets
  # count = var.num_of_buckets != 0 ? var.num_of_buckets : 1
  for_each = toset(var.users)
  bucket   = "example-s3-bucket-${each.value}"
}
resource "aws_iam_user" "new_users" {
  for_each = toset(var.users)
  name = each.value
}
output "uppercase_users" {
  value = [for user in var.users : upper(user) if length(user) > 6]
}
`
```

```
  bucket = "${var.s3-bucket-name}-${count.index +1
  acl    = "private"
  # count = var.num_of_buckets
  count = var.num_of_buckets != 0 ? var.num_of_buc
}

resource "aws_iam_users" "new_users" {
  for_each = toset(var.users)
  name = each.value
}
```

for_each = toset(var.users)----set e ceviryoruz
    name = each.value

Bu bir kalip bu sekilde user yazarken kullaniliyor

```bash
terraform apply
```

- Go to the AWS console (IAM and S3) and check the resources.

### Terraform Data Sources

- `Data sources` allow data to be fetched or computed for use elsewhere in Terraform configurati on.
- Go to the `AWS console and create an image` from your EC2. Select your instance and from actio ns click image and templates and then give a name for ami `my-ami` and click create.

# It will take some time. go to the next steps.

- Go to the `variables.tf` file and comment the variable `ec2-ami`.
- Go to the `main.tf` file make the changes in order.

```bash
data "aws_ami" "tf_ami" {
  most_recent      = true
  owners           = ["self"]
  filter {
    name = "virtualization-type"
    values = ["hvm"]
```

```
  }
}
resource "aws_instance" "tf-ec2" {
  ami           = data.aws_ami.tf_ami.id
  instance_type = var.ec2-type
  key_name      = "mk"
  tags = {
    Name = "${local.instance-name}-this is from my-ami"
  }
}
```
```bash
terraform plan
```
```bash
terraform apply
```
- Check EC2 instance's ami id.
- You can see which data sources can be used with a resource in the documentation of terraform. For example EBS snapshot.

```bash
terraform destroy
```

```
resource "aws_s3_bucket" "tf-s3" {
  # bucket = "${var.s3-bucket-name}-${count.index +1}"
  acl    = "private"
  # count = var.num_of_buckets
  # count = var.num_of_buckets != 0 ? var.num_of_buckets : 3
  for_each = toset(var.users)
  bucket = "example-s3-bucket-${ecah.value}"

}


resource "aws_iam_users" "new_users" {
  for_each = toset(var.users)
  name = each.value
}

output "upper" {
  value = [for user in var.users :upper(user) if length(user) >6]
}
```

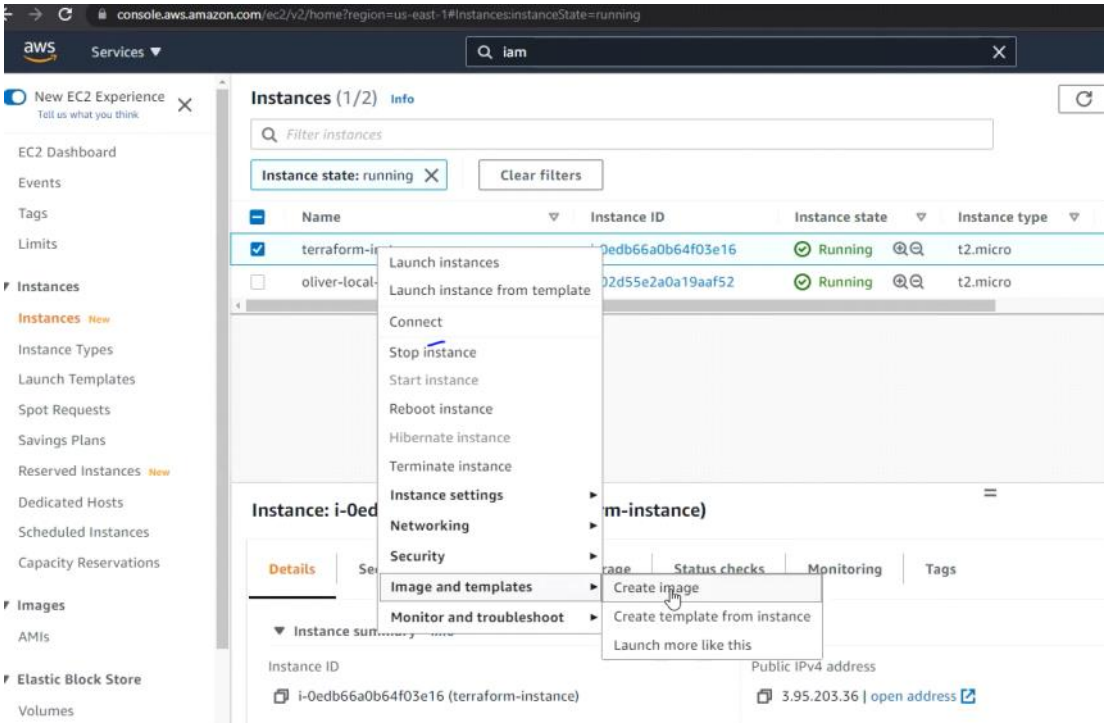
**Variable lerde toplam 3 user atamistik**

**Resorce da for_each ile 3 user olusturduk**

**3 tanede bucket olusturduk**

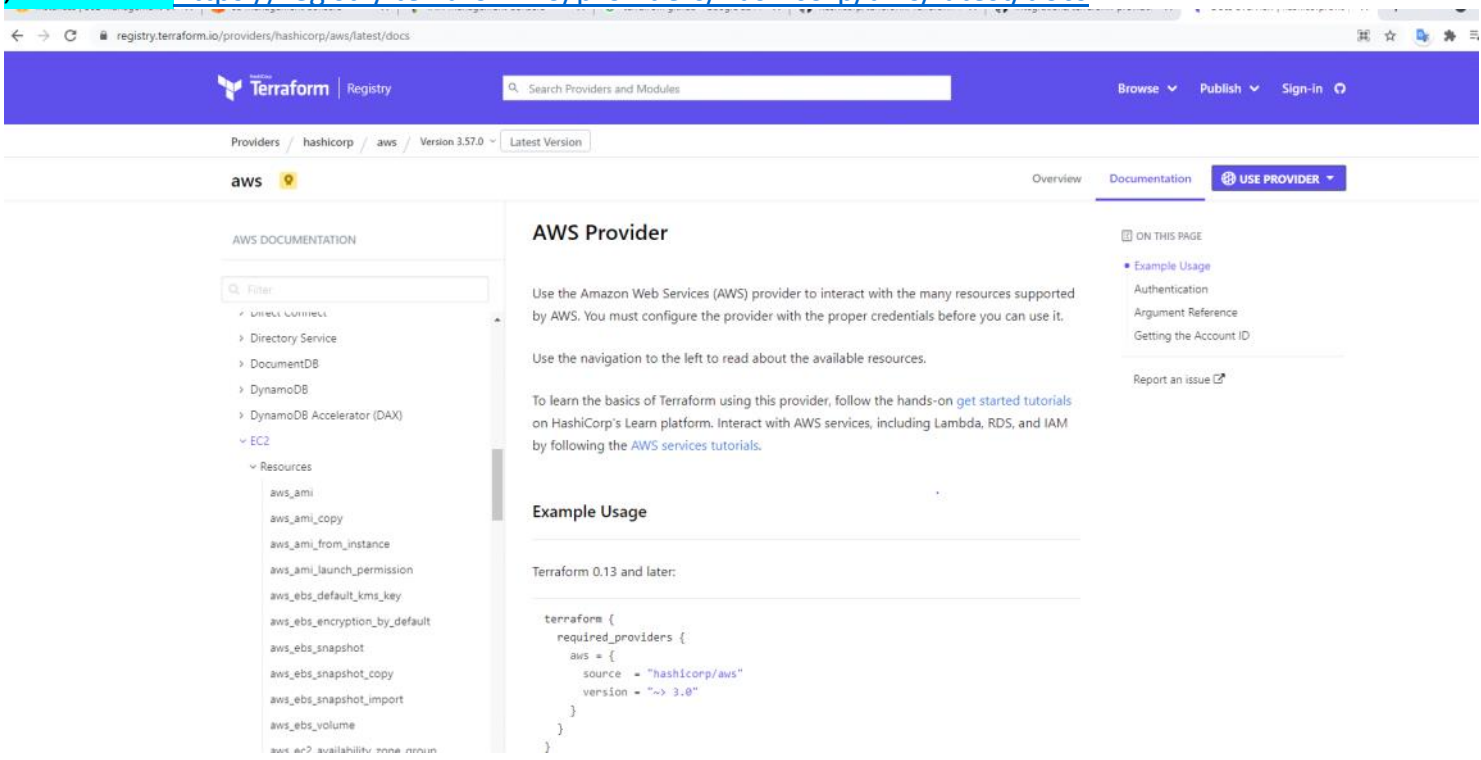
**Output da bir döngu olusturduk for i in de oldugu gibi  isimdeki harf sayisi 6 dan buyuk olalariyazdirmasini istedik**

**Ilk olarak ec2 da image create ediyorruz**




**Bir seyin Datasi var mi cekilebiliyor mu bakmak istersek asagidaki adresten bakabiliyoruz**

; https://registry.terraform.io/providers/hashicorp/aws/latest/docs




**Kisazca komutlarimiiz özetlersek ;**

**Main tf deki bazi seyleri variable kullanarak almaya calisacagiz**

**Variable yani degisken olarak adlandiriliyor Bu file de bir cok kisi file kullanmak isteyebilir farkli bir sey kurmak isterse variablelerde kucuk degisiklikler ile islemini kolayca yapmis olur**

**Dosyanin icerosinden cewkmek icinde ;**

**Var.ulasmak istedigiz resource**

```
resource "aws_s3_bucket" "tf-s3" {
```

```
  bucket = var.s3-bucket-name
  acl    = "private"
}
```

```variable lara benzeyen ancak bize özel bir durumda lokal var file icinde sirketin ismini kullanacaksiniz bunu da ayri bur yere degil de locals yarazak yapa biliyoruz

- A `local`

for_each = toset(var.users)----set e ceviryoruz
   name = each.value

Bu bir kalip bu sekilde user yazarken kullaniliyor

Ilk olarak
```