* Terraform init – to initialize the cloud provider
* terraform providers -> to check the cloud provider
* export AWS\_ACCESS\_KEY\_ID=”accesskey”(for linux)

set AWS\_ACCESS\_KEY\_ID=”accessley”(for windows)

* set AWS\_SECRET\_ACCESS\_KEY=”secret\_key”
* terraform plan
* terraform apply
* terraform apply -refresh=false
* cls -> in windows to clear screen
* terraform apply -target="aws\_iam\_user.my\_iam\_user"---> whn u want to update a particular resource, not all the resources
* terraform validate -> to check whether the terraform code is valid or not
* terraform show -> it shows human readable output of current state

..................

# STATE

# DESIRED - KNOWN - ACTUAL

# DESIRED -> in main.tf wahtever is mentioned that is a desired state. i would want a s3 bucket, i would want 5 vms

# KNOWN - this state is the result of previous execution and it is stored at terraform.tfstate, which is a kind of back up

# ACTUAL - this is the actual state in aws

# so, when we run terraform apply, terraform checks that desired state and actual state is same or not.

# based on that it performs the addition, deletion all these

.....................................

.gitignore => when we want to commit our terraform projects in any version control system like git, we would exclude all terraform state files and /terraform files. Because they can contain sensitive information. Better to store them in s3 bucket.

..............................................................................

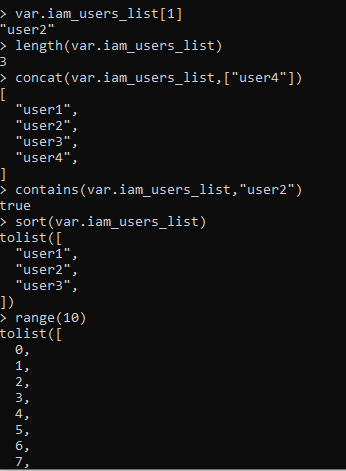
Variables – what made the terraform code dynamic

**project1** – creating multiple aws resources(like s3, IAM user) using terraform

**project2** – creating multiple IAM user, and check different options to declare variable

**project3** - use list to create Multiple IAM users, then all the functions for list – length, sort, range,

----if we use count, our deletions and updations for the list is based on index

----if we use for\_each, then deletions and updations are based on values of the list

**project4** – maps. Creating multiple user using map, map of maps

---- terraform fmt -> it would format the code with spaces

---- terraform has 4 states – desired state, known state(terraform state),actual state

---- terraform state stores the mapping between terraform object and the actual cloud object

---- EC2 – a virtual server(a server in cloud)

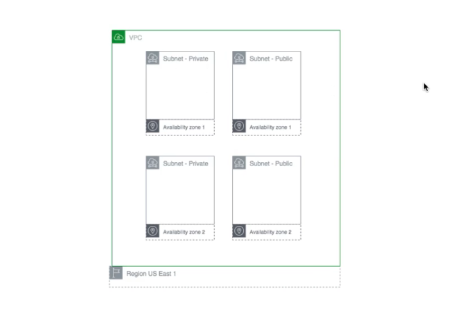
what is a virtual server?

So virtual servers are basically servers in the cloud which we make use of. And in AWS the virtual servers are called EC2, Elastic Compute Cloud.

choose the region,

choose AMI(An AMI is a template that contains the software configuration(os,application server, and applications...) requird to launch your instance.

vpc – is a kind of your own network in cloud. Which has multiple public and private subnets in it. Public subnets where the instances are accessible from outside world. In private subnet only the resources within the vpc can access the resources in private subnet-like database server. In all the subnets there can be multiple ec2 instances. Security groups are configured on these ec2 instances to control inbound and outbound traffic.



**project5** – creating a security group in terraform,

creating secure access key in aws console,

then creating ec2 instance .

Deploy our 1st web server in terraform in automated way.

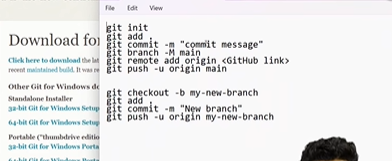
---- In terraform whatever resource we create thats immutable. We cant add or change anything of that resource. We need to destroy the resource and then create again with the changes.

------ linux commands, network(like CIDR) & os, cloud, docker, kubernetes, ci/cd pipeline(jenkins/gitlab),iac(terraform)

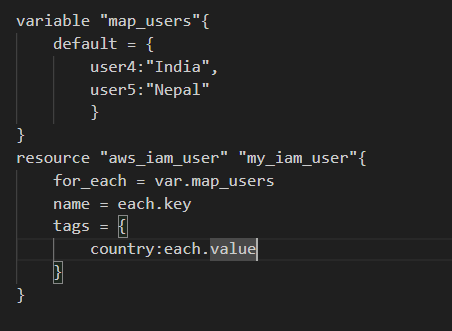
**project6** – creating multiple ec2 instances and creating load balancer on top of them

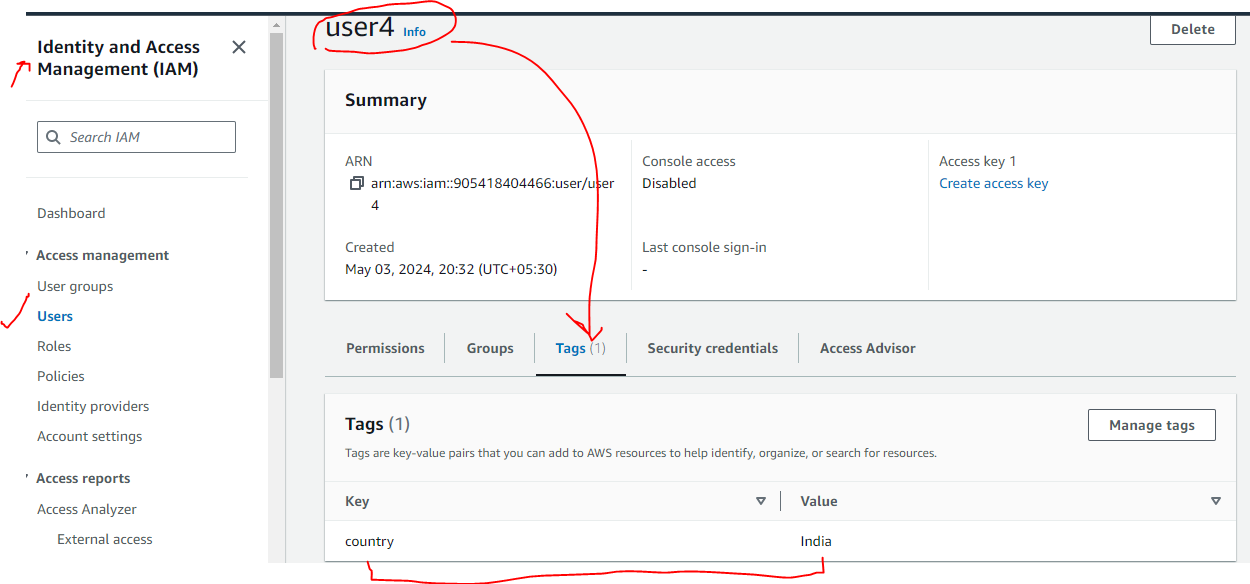
**project7** – terraform project to store remote state. Store project state in s3 bucket.

------ There are multiple environment/workspaces – dev, QA and stage environment



In the below set up, we can see for map implementation whats the change in aws





................................

Map functions:-

|  |  |
| --- | --- |
|  |  |
|  |  |

.............

in aws virtual servers are called ec2 instances.

Information needed to create ec2 instances:-

1. Region

2. AMI id

3. instance type

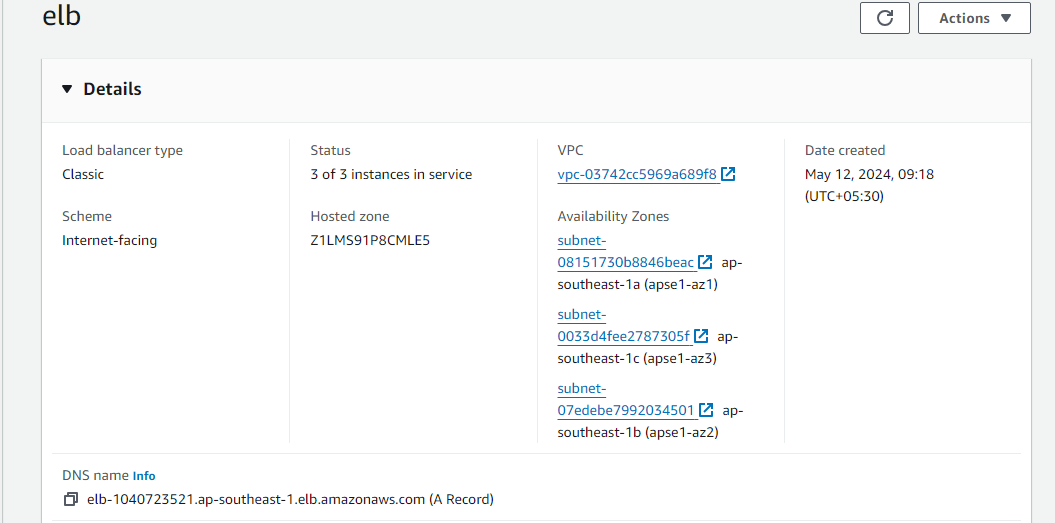
4. vpc id

5. security group

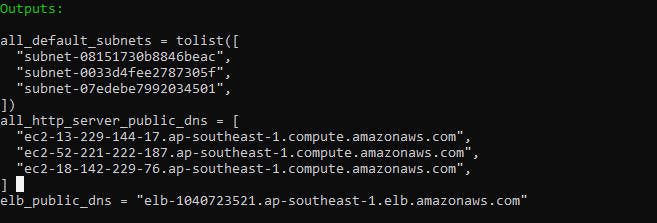
Data providers:-

.......................

Creating multiple ec2 instances and put them behind our load balancers



|  |
| --- |
|  |
|  |
|  |

...........................