3 TIER ARCHITECTURE DEPLOYMENT TERRAFORM+AWS

Free 30 DAYS DevOps + AWS Bootcamp4 [DNA]

- BY Praveen Singampalli founder of Heydevops
- Linkedin - https://www.linkedin.com/in/praveen-singampalli/
- Instagram - https://www.instagram.com/singam4devops/
- Join telegram group for documents - https://t.me/+UIfRjeavrYgwMWY1
- Download HeyDevOps App on IOS/Android

HEY DEVOPS

Register for Upcoming 100% Job Ready PRIME BATCH5.0

UPCOMING 100% Job ready | 12+ Projects | PRIME 5.0 : https://heydevops.in/course-details

Early Bird Offer Code "HEYDEVOPS" [70% DISCOUNT VALID FOR NEXT FEW HOURS] 😃

Join the telegram group for documents - https://t.me/+UIfRjeavrYgwMWY1

PRICE VS ADDONS FOR PRIME 5.0

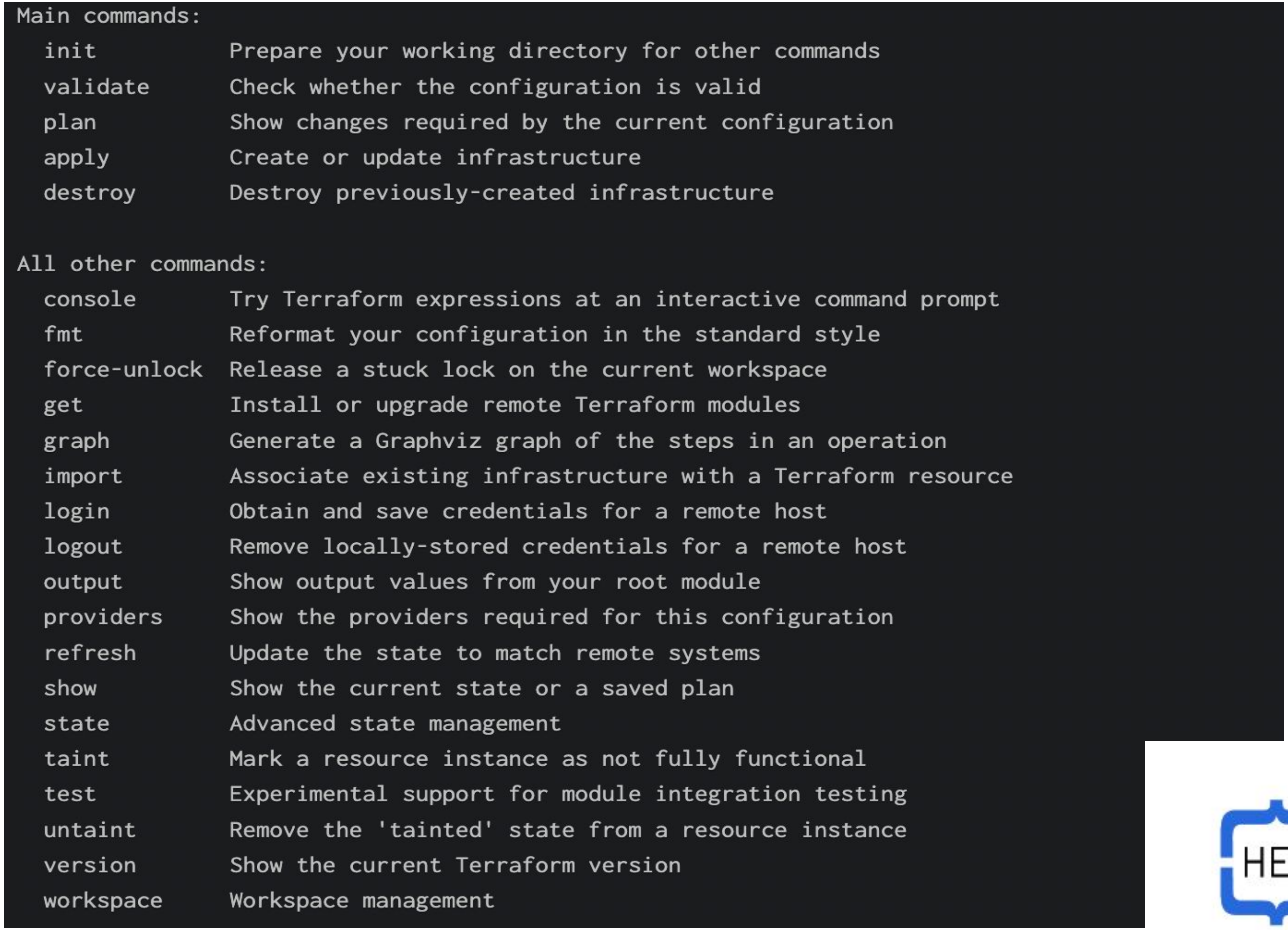
✅ Starting 04th Nov 2023
✅ 12+ Industry based projects
✅ Download the app from the Android play store
✅ Live Sessions / Recordings available
✅ 2 months (8+ weeks)
✅ Sat-Sun live classes 9 AM - 12 PM
✅ Certificate after course completion
✅ EXTRA DOUBT CLEARING SESSIONS
✅ EXTRA PROJECTS WITH DOCUMENTS
✅ JOB SUPPORT VIA FEW EXTRA SESSIONS AFTER COMPLETION OF COURSE
✅ JOB SUPPORT VIA RESUME TO HR DIRECT CONNECT
✅ BONUS: RESUME REVIEWS / MOCK INTERVIEWS / LIVE FEEDBACK SHARE TO STUDENTS / INTEWISE DISCUSSION

HEY DEVOPS

Project Theory+Handson Complete Video

https://youtube.com/live/UyOso3keZkM?feature=share [Date - Saturday 14 Oct 2023]

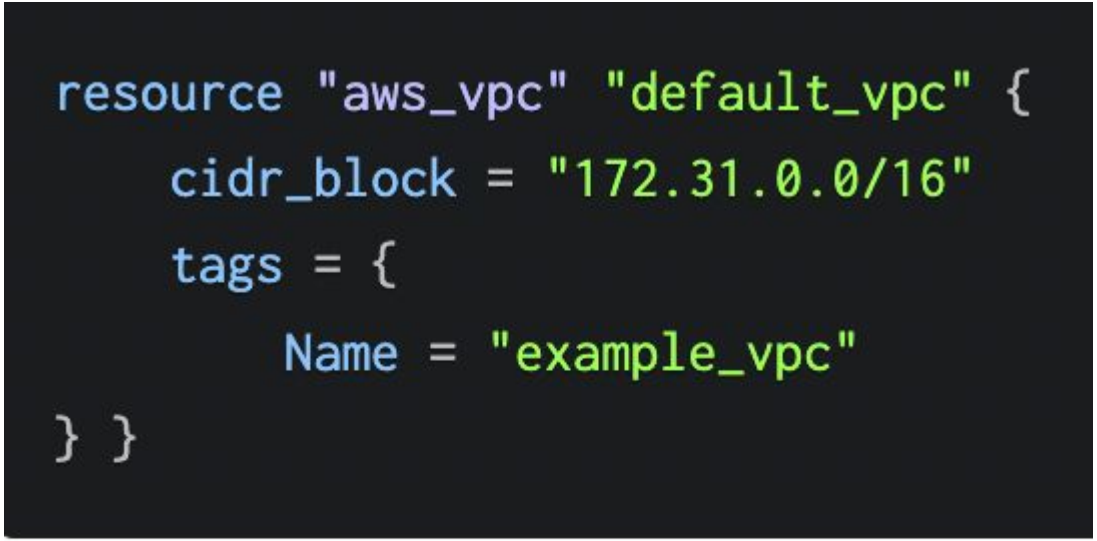
Terraform Commands



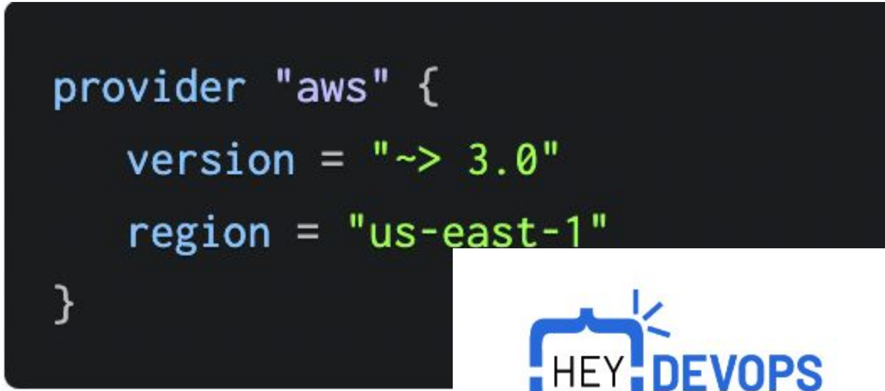
```
Main commands:
  init          Prepare your working directory for other commands
  validate      Check whether the configuration is valid
  plan          Show changes required by the current configuration
  apply         Create or update infrastructure
  destroy       Destroy previously-created infrastructure

All other commands:
  console       Try Terraform expressions at an interactive command prompt
  fmt           Reformat your configuration in the standard style
  force-unlock  Release a stuck lock on the current workspace
  get           Install or upgrade remote Terraform modules
  graph         Generate a Graphviz graph of the steps in an operation
  import        Associate existing infrastructure with a Terraform resource
  login         Obtain and save credentials for a remote host
  logout        Remove locally-stored credentials for a remote host
  output        Show output values from your root module
  providers     Show the providers required for this configuration
  refresh       Update the state to match remote systems
  show          Show the current state or a saved plan
  state         Advanced state management
  taint         Mark a resource instance as not fully functional
  test          Experimental support for module integration testing
  untaint       Remove the 'tainted' state from a resource instance
  version       Show the current Terraform version
  workspace     Workspace management
```

Terraform Language



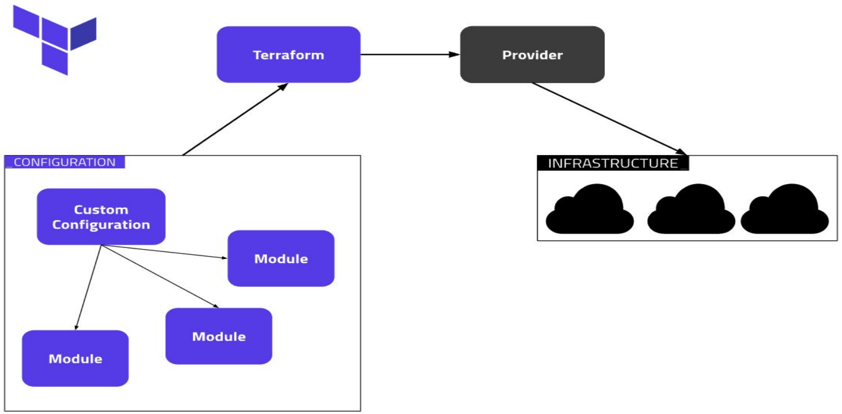
```
resource "aws_vpc" "default_vpc" {
    cidr_block = "172.31.0.0/16"
    tags = {
        Name = "example_vpc"
} }
```



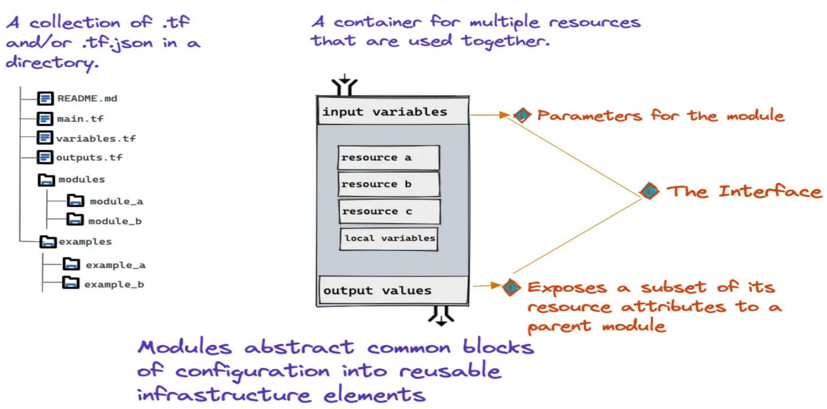
```
provider "aws" {
   version = "~> 3.0"
   region = "us-east-1"
}
```

HEY DEVOPS

Terraform Modules



Modules: Definitions



A collection of .tf and/or .tf.json in a directory.

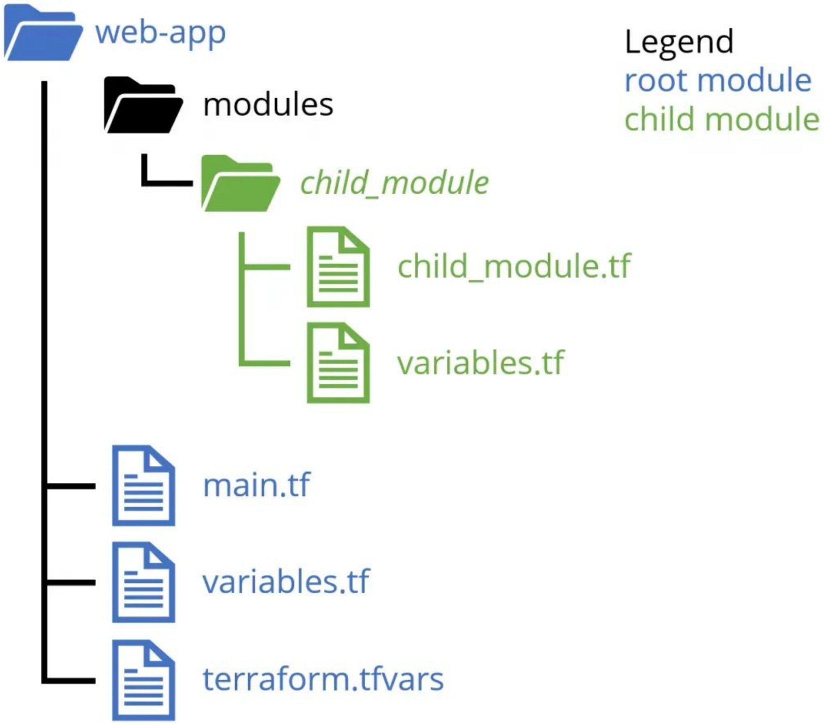
A container for multiple resources that are used together.

Parameters for the module

The Interface

Exposes a subset of its resource attributes to a parent module

Modules abstract common blocks of configuration into reusable infrastructure elements



HEY DEVOPS

AWS Components

Virtual private clouds (VPC)

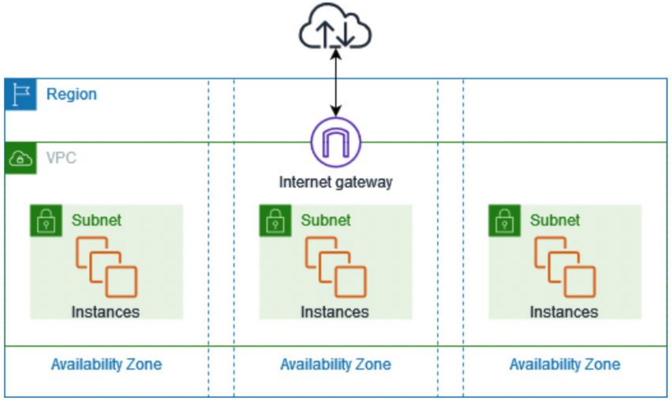
A VPC is a virtual network that closely resembles a traditional network that you'd operate in your own data center. After you create a VPC, you can add subnets.

Subnets

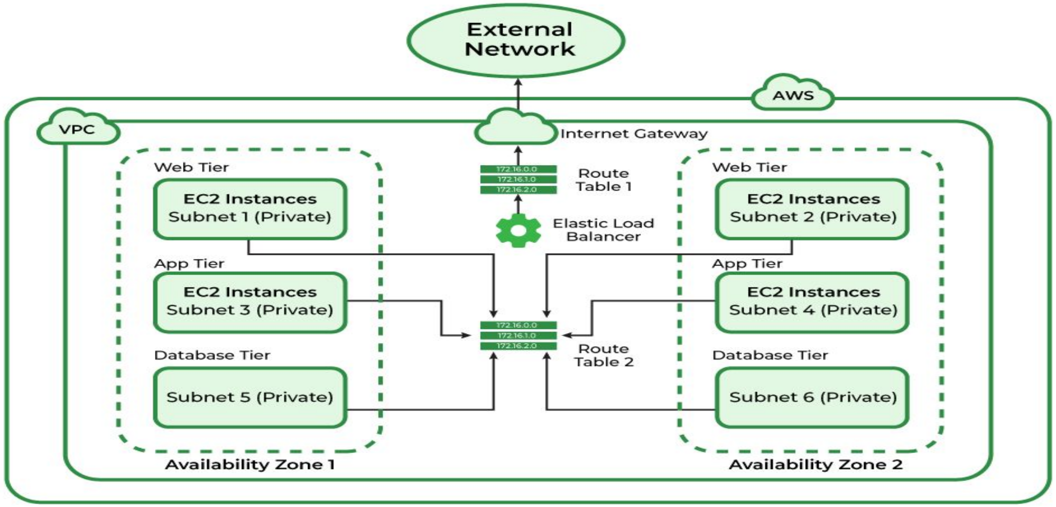
A subnet is a range of IP addresses in your VPC. A subnet must reside in a single Availability Zone. After you add subnets, you can deploy AWS resources in your VPC.

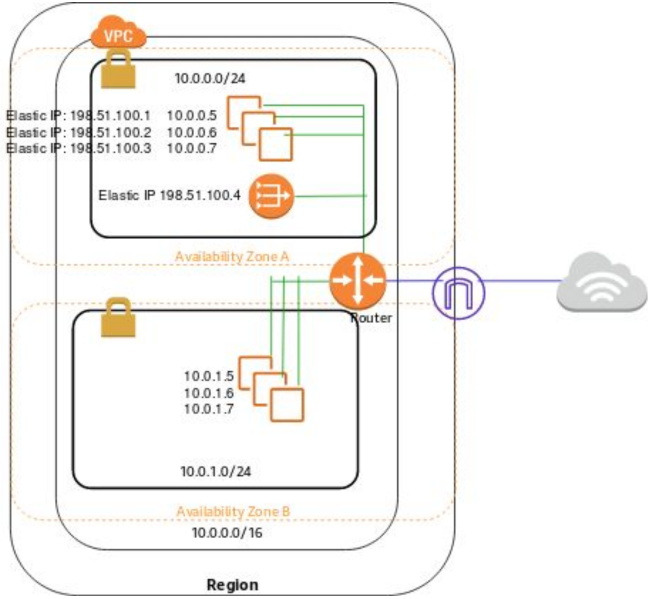
Gateways and endpoints

A gateway connects your VPC to another network. For example, use an internet gateway to connect your VPC to the internet. Use a VPC endpoint to connect to AWS services privately, without the use of an internet gateway or NAT device.



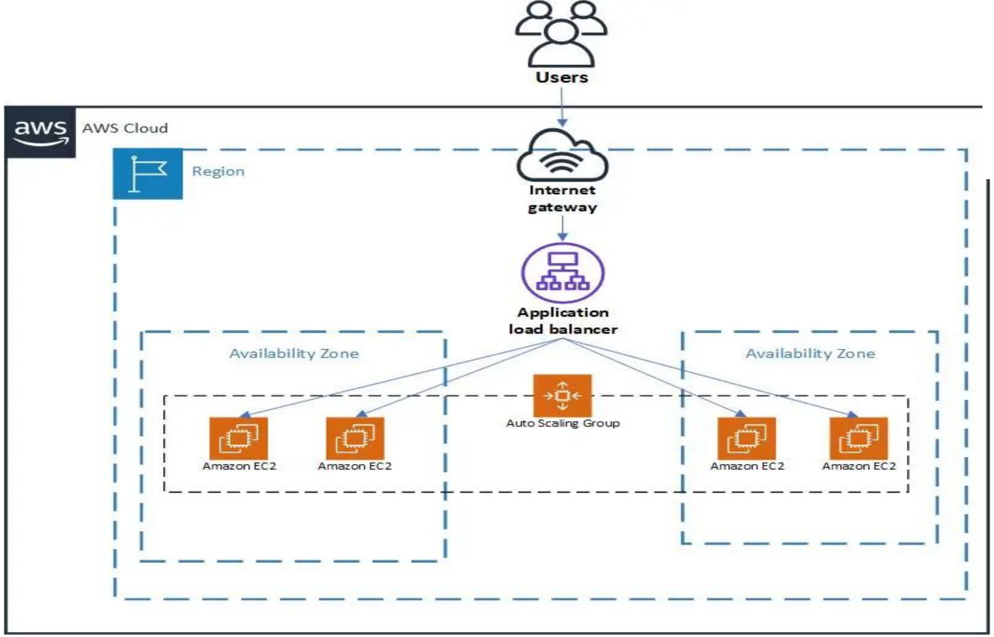
AWS Subnets

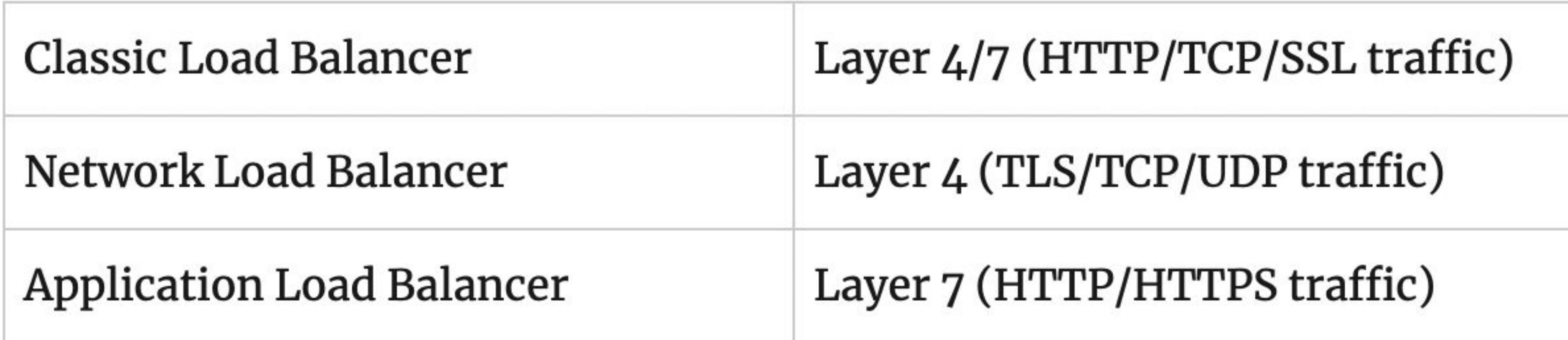

External Network
AWS
VPC
Internet Gateway
172.16.0.0
172.16.1.0
172.16.2.0
Route Table 1
Elastic Load Balancer
Web Tier
EC2 Instances Subnet 1 (Private)
App Tier
EC2 Instances Subnet 3 (Private)
Database Tier
Subnet 5 (Private)
Availability Zone 1
Web Tier
EC2 Instances Subnet 2 (Private)
App Tier
EC2 Instances Subnet 4 (Private)
Database Tier
Subnet 6 (Private)
Availability Zone 2
172.16.0.0
172.16.1.0
172.16.2.0
Route Table 2


VPC
10.0.0.0/24
Elastic IP: 198.51.100.1 10.0.0.5
Elastic IP: 198.51.100.2 10.0.0.6
Elastic IP: 198.51.100.3 10.0.0.7
Elastic IP 198.51.100.4
Availability Zone A
Router
10.0.1.5
10.0.1.6
10.0.1.7
10.0.1.0/24
Availability Zone B
10.0.0.0/16
Region

ASG and LB

HEY DEVOPS



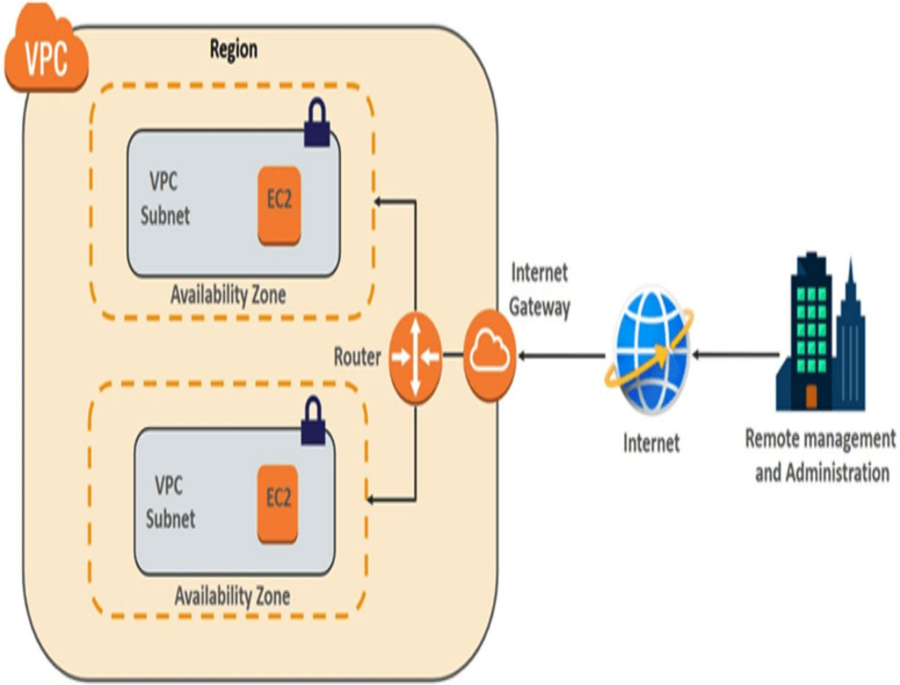


Classic Load Balancer	Layer 4/7 (HTTP/TCP/SSL traffic)
Network Load Balancer	Layer 4 (TLS/TCP/UDP traffic)
Application Load Balancer	Layer 7 (HTTP/HTTPS traffic)

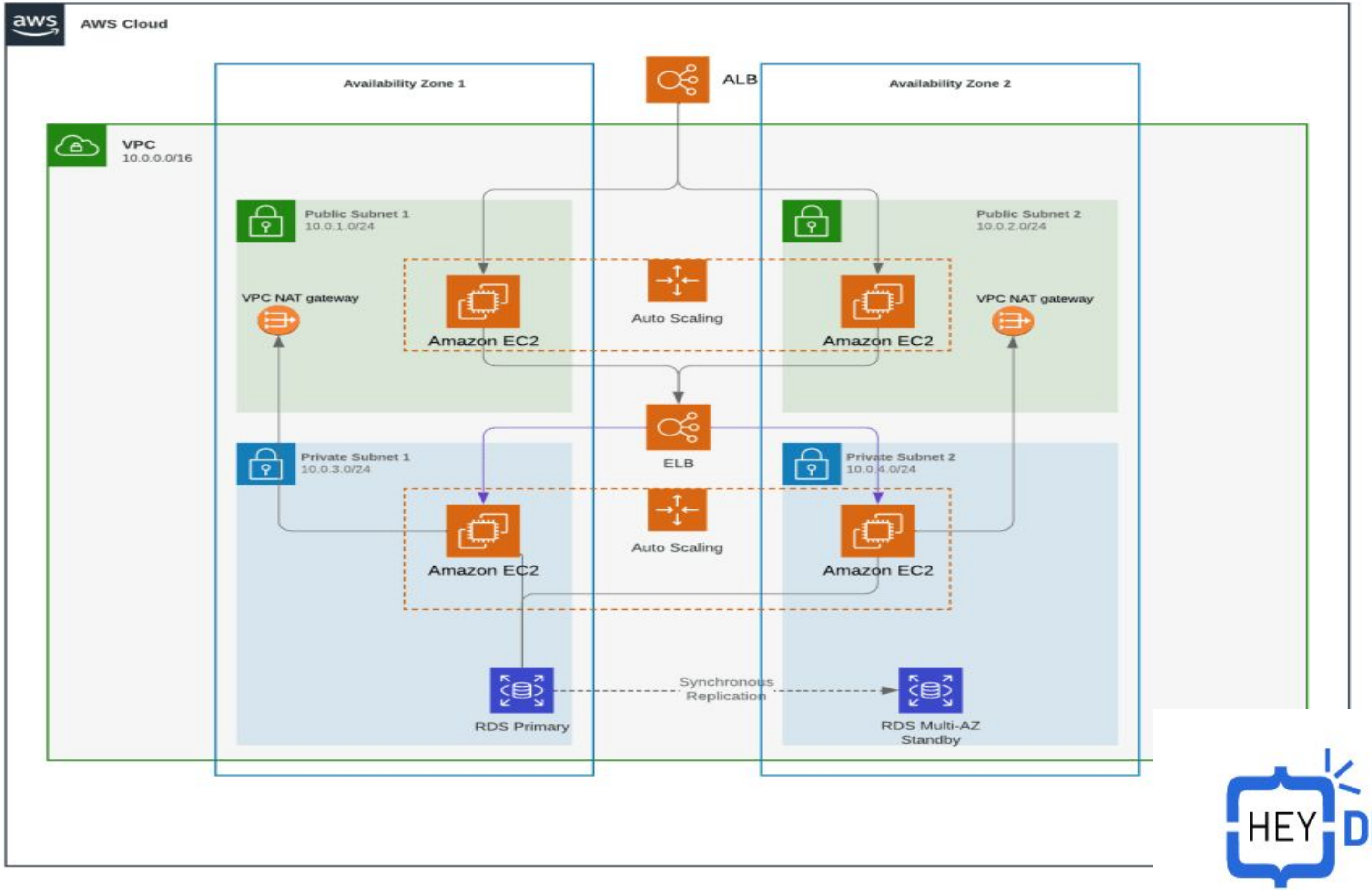
Three-tier architecture in a VPC

The majority of today's applications are designed with a three-tier architecture comprised of the following interconnected tiers:

> The web or presentation tier, which takes requests from web browsers and presents information created by, or stored within, the other layers to end users.
>
> The application tier, which houses the business logic and is where most processing takes place.
>
> The database tier, comprised of database servers that store the data processed in the application tier.



AWS 3 TIER ARCHITECTURE FOR PROJECT

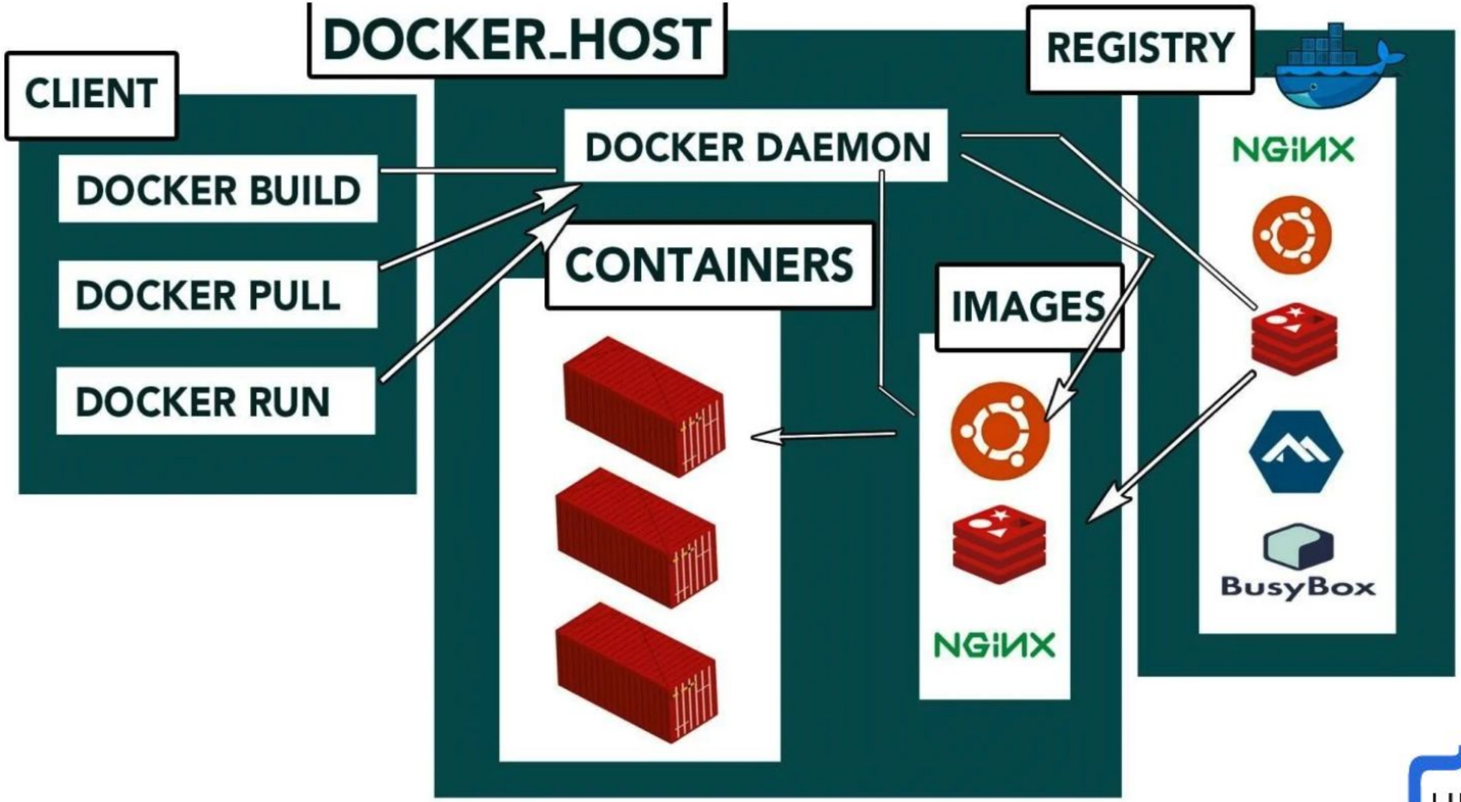


HEY DEVOPS

Components of AWS

- 1 VPC
- 2 Public subnets
- 2 Private subnets
- 2 Autoscaling groups
- 5 Security Groups
- 2 Load Balancers, (one private, one public)
- 2 Private EC2 instances (representing our application tier)
- 2 Public EC2 instances (representing our presentation tier)
- 2 Nat Gateways (so private instances can connect to the internet)
- 2 Elastic IP addresses, one for each Nat gateway
- 1 rds instance

Docker



HEY DEVOPS

HandsOn

1) Clone the code - https://github.com/heydevopsorg/terrform_threetierarch.git
2) Create the AWS account - https://aws.amazon.com/console/
3) Install Docker and terraform on windows
   https://docs.docker.com/desktop/install/windows-install/
   https://developer.hashicorp.com/terraform/tutorials/aws-get-started/install-cli

4) Execute the linux command to give permission
   chmod +x setup-ecrs.sh
5) Run this on terminal to create the ECR repo and to create images in local and send those to ECR- ./setup-ecrs.sh
6) Go to terraform folder -
   terraform init
   terraform plan
   terraform apply

Testing the Application via LB

1) Hit the **Front end** load balancer-
front-end-lb-**********.us-east-1.elb.amazonaws.com/

2) Request to the **presentation layer**, which forwards the requests to the application layer (via the internal Load Balancer) that finally creates a table called users, and adds 2 users in the table
front-end-lb-**********.us-east-1.elb.amazonaws.com/init

3) To view the **users table** you can call

```
front-end-lb-**********.us-east-1.elb.amazonaws.com/users
```

```
4)  Delete the Architecture
    terraform delete
```