

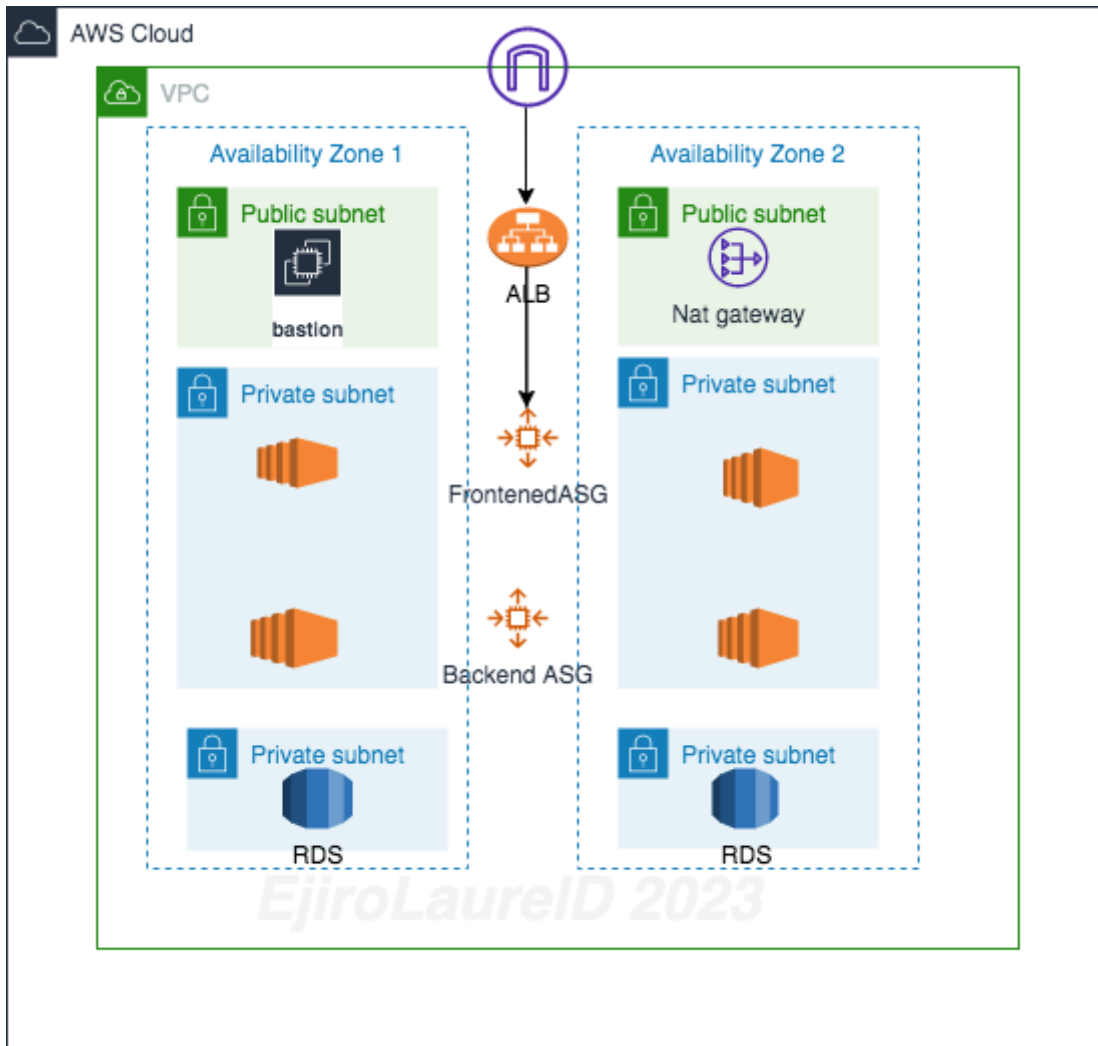
Three-Tier Architecture Deployment on AWS with Terraform

A three-tier architecture consisting of a Web tier, Application tier and a Database tier in private subnets with Autoscaling for the web and application tier and a load balancer. A Bastion Host and Nat gateway provisioned to allow ssh access to the instances and access to the internet.

Terraform modules were used to make the process easily repeatable and reusable.

This deployment will create a scalable, secure and highly available infrastructure that separates the different layers ensuring they are all communicating with each other. The architecture includes an Amazon Virtual Private Cloud (VPC), Elastic Load Balancer (ELB), Auto Scaling Group (ASG), and a Relational Database(RDS).

- The Web tier will have a bastion host and NAT gateway provisioned in the public subnets. The bastion host will serve as our access point to the underlying infrastructure. The NAT Gateway will allow our private subnets to communicate with the internet while maintaining a level of security by hiding the private instances' private IP addresses from the public internet.
- In the Application tier, we will create an internet facing load balancer to direct internet traffic to an autoscaling group in the private subnets, along with a backend autoscaling group for our backend application. We will create a script to install the apache webserver in the frontend, and a script to install Node.js in the backend.
- In the Database tier, we will have another layer of private subnets hosting a MySQL database which will eventually be accessed using Node.js..



I have provided a step-by-step guide to deploying this architecture on Amazon Web Services (AWS) using Terraform.

Prerequisites

Before you begin, ensure that you have the following prerequisites:

1. AWS account credentials (access key ID and secret access key).
2. Terraform installed on your local machine. You can download Terraform from the official website: <https://www.terraform.io/downloads.html>.
3. Basic knowledge of AWS services such as EC2, VPC, ELB, ASG, and RDS.
4. Familiarity with the basics of Terraform, including how to write Terraform configuration files (`.tf`).

Steps

Follow these step-by-step instructions to deploy a three-tier architecture on AWS using Terraform:

Step 1: Clone the Repository

1. Open a terminal or command prompt on your local machine.

2. Clone the repository containing the Terraform configuration files:

```
git clone https://github.com/3tierweb.git
```

3. Change into the project directory:

```
cd your-repo-directory
```

Step 2: Configure AWS Credentials

1. Open the AWS Management Console in your web browser.
2. Navigate to the **IAM** service.
3. Create a new IAM user or use an existing one.
4. Assign the necessary permissions to the IAM user, such as [AmazonEC2FullAccess](#), [AmazonRDSFullAccess](#), [AmazonVPCFullAccess](#), and [ElasticLoadBalancingFullAccess](#).
5. Generate an access key ID and secret access key for the IAM user.
6. Configure the AWS CLI with the IAM user credentials using the following command:

```
aws configure
```

Enter the access key ID and secret access key when prompted, and optionally set the default region.

Step 3: Configure S3 bucket for state file storage

1. Sign in to your AWS account.
2. Open the Amazon S3 service.
3. Click "Create Bucket" and configure basic settings like name and region.
4. Optionally, enable features like versioning, logging, and encryption.
5. Review settings and click "Create bucket."

Step 4: Configure Terraform Variables

1. Open the project directory in a text editor.
2. Locate the Terraform configuration file named `terraform.tfvars`.
3. Modify the values of the variables according to your requirements.
 - **dbuser**: Set the username for the database.
 - **dbpassword**: Set the password for the database.
 - **db_name**: Set the name of the database. Do not forget to gitignore your .tfvars file

Step 5: Initialize Terraform

1. In the terminal or command prompt, navigate to the project directory., cd to the root directory 'terraform'
2. Run the following command to fix any syntax issue

```
terraform fmt
```

3. Run the following command to initialize Terraform and download the required providers:

```
terraform init
```

Step 6: Review and Validate the Configuration

1. Run the following command to review the changes that Terraform will make:

```
terraform plan
```

Review the output to ensure that the planned infrastructure matches your expectations.

Step 7: Deploy the Infrastructure

1. Run the following command to deploy the infrastructure:

```
terraform apply
```

Terraform will show you a summary of the changes that will be made. Type **yes** to confirm and start the deployment.

2. Wait for Terraform to provision the infrastructure. This process may take several minutes.

Step 8: Access the Application

1. After the deployment is complete, Terraform will output the DNS name of the ELB.
2. Copy the DNS name and paste it into your web browser.
3. If everything is set up correctly, you should see the application running.

Step 9: Destroy the Infrastructure (Optional)

If you want to tear down the infrastructure and remove all resources created by Terraform, you can follow these steps:

1. In the terminal or command prompt, navigate to the project directory.

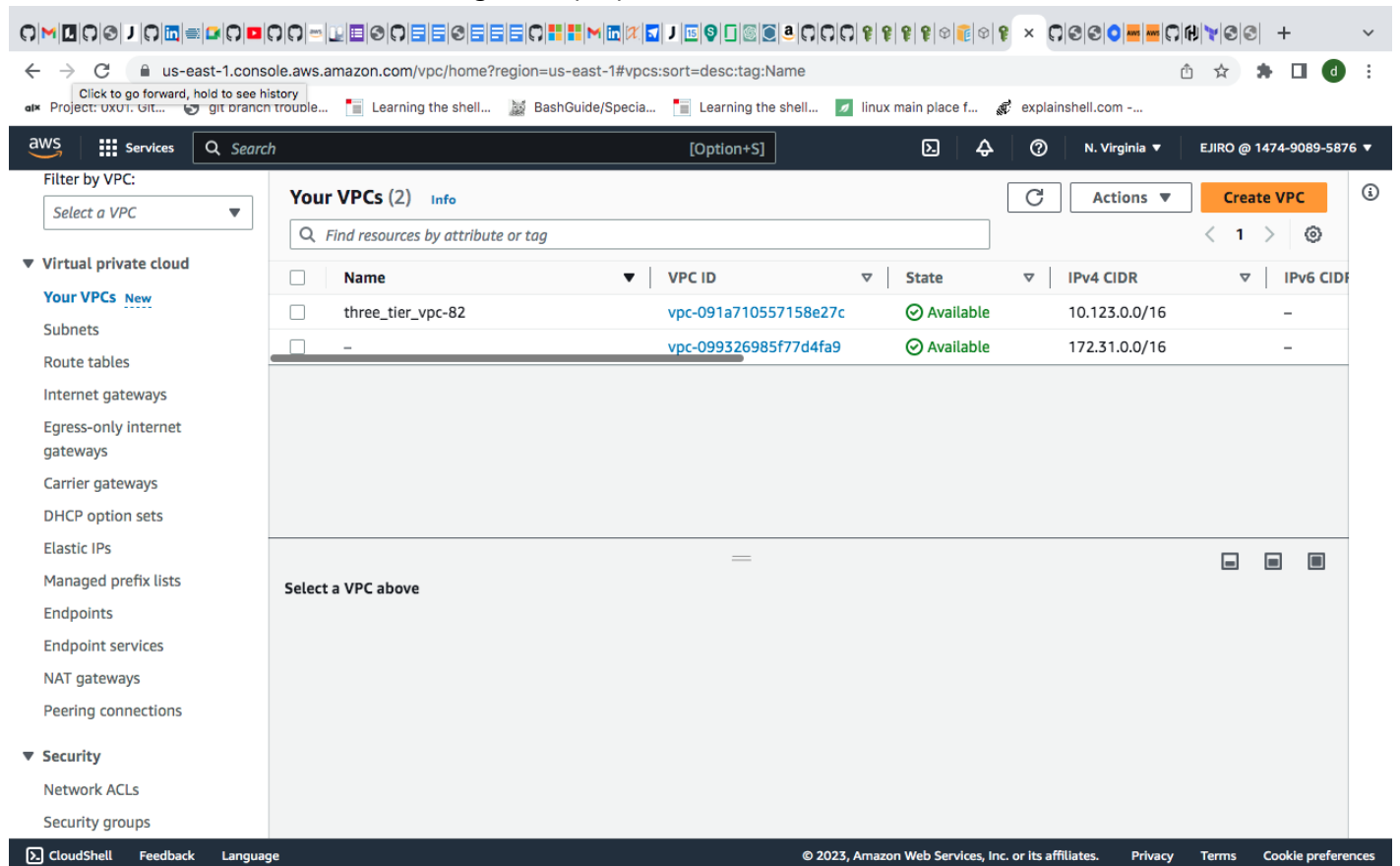
2. Run the following command to destroy the infrastructure:

```
terraform destroy
```

Type **yes** to confirm the destruction.

Step 10: Confirm Infrastructure

If you go into your AWS console, you should be able to see the VPC and subnets, internet gateway, route tables and associations, EC2 instances running in the proper locations, load balancers, and RDS database.



The screenshot shows the AWS Management Console interface. The top navigation bar includes the AWS logo, a search bar, and account details (N. Virginia, EJIRO @ 1474-9089-5876). The left sidebar is expanded to show the 'Virtual private cloud' section, with 'Your VPCs' selected. The main content area displays 'Your VPCs (2)' with a table listing the VPCs.

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR
three_tier_vpc-82	vpc-091a710557158e27c	Available	10.123.0.0/16	-
-	vpc-099326985f77d4fa9	Available	172.31.0.0/16	-

Below the table, there is a section titled 'Select a VPC above' with three small icons.

Project: 0x01. Git...git branch trouble...Learning the shell... BashGuide/Specia... Learning the shell...linux main place f...explainshell.com -...

ServicesSearch[Option+S]

N. VirginiaEJIRO @ 1474-9089-5876

VPC dashboard

EC2 Global View

Filter by VPC:
Select a VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Carrier gateways

DHCP option sets

Elastic IPs

Managed prefix lists

Endpoints

Endpoint services

NAT gateways

Peering connections

Subnets (1/12)

Filter subnets

<input type="checkbox"/>	three_tier_public_2	subnet-02f55c18f118390b8	Available	vpc-091a710557158e27c thr...	10.123.11.0/24
<input type="checkbox"/>	-	subnet-0cb7cbcd8f25c828f	Available	vpc-099326985f77d4fa9	172.31.80.0/20
<input type="checkbox"/>	-	subnet-01a6326bb5690d6b2	Available	vpc-099326985f77d4fa9	172.31.64.0/20
<input type="checkbox"/>	-	subnet-0691b59e776672b28	Available	vpc-099326985f77d4fa9	172.31.0.0/20
<input type="checkbox"/>	three_tier_public_1	subnet-0574e04af59997634	Available	vpc-091a710557158e27c thr...	10.123.10.0/24
<input type="checkbox"/>	-	subnet-0530be2263ae4834a	Available	vpc-099326985f77d4fa9	172.31.16.0/20
<input type="checkbox"/>	three_tier_private_2	subnet-08293d28bfbdcac49	Available	vpc-091a710557158e27c thr...	10.123.21.0/24
<input checked="" type="checkbox"/>	three_tier_private_1	subnet-0a2e45a65ba1d44c0	Available	vpc-091a710557158e27c thr...	10.123.20.0/24
<input type="checkbox"/>	three_tier_private_...	subnet-0075a27442f1e8ffc	Available	vpc-091a710557158e27c thr...	10.123.41.0/24
<input type="checkbox"/>	three_tier_private_...	subnet-0dc88387eff19ace	Available	vpc-091a710557158e27c thr...	10.123.40.0/24

subnet-0a2e45a65ba1d44c0 / three_tier_private_1

DetailsFlow logsRoute tableNetwork ACLCIDR reservationsSharingTags

Details

CloudShellFeedbackLanguage

© 2023, Amazon Web Services, Inc. or its affiliates. PrivacyTermsCookie preferences

Project: 0x01. Git...git branch trouble...Learning the shell... BashGuide/Specia... Learning the shell...linux main place f...explainshell.com -...

ServicesSearch[Option+S]

N. VirginiaEJIRO @ 1474-9089-5876

Instances (1/5)

Find instance by attribute or tag (case-sensitive)

Instance state = runningClear filters

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
-	i-042eb86be47bf9eb9	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	-
-	i-0fceb7614305f8d40	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	-
-	i-043107e3f8d626222	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-52-90-213-2
-	i-0662a7271088df9eb	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	-
-	i-02ec07d7a1603f30c	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	-

Instance: i-043107e3f8d626222

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LoadBalancer:loadBalancerArn=arn:aws:elasticloadbalancing:us-east-1:147...

Project: 0x01. Git... git branch trouble... Learning the shell... BashGuide/Specia... Learning the shell... linux main place f... explainshell.com -...

aws Services Search [Option+S]

N. Virginia EJIRO @ 1474-9089-5876

New EC2 Experience Tell us what you think

EC2 Dashboard
EC2 Global View
Events

▼ Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Scheduled Instances
Capacity Reservations

▼ Images
AMIs
AMI Catalog

► Elastic Block Store

CloudShell Feedback Language

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

three-tier-loadbalancer

Details

Load balancer type Application	Status Active	VPC vpc-091a710557158e27c	IP address type IPv4
Scheme Internet-facing	Hosted zone Z35SXDOTRQ7X7K	Availability Zones subnet-02f55c18f118390b8 us-east-1b (use1-az6) subnet-0574e04af59997634 us-east-1a (use1-az4)	Date created July 16, 2023, 08:36 (UTC+01:00)
Load balancer ARN arn:aws:elasticloadbalancing:us-east-1:147490895876:loadbalancer/app/three-tier-loadbalancer/8646b6c56c0df728		DNS name three-tier-loadbalancer-1608768141.us-east-1.elb.amazonaws.com (A Record)	

Listeners and rules Network mapping Security Monitoring Integrations Attributes Tags

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#database:id=three-tier-db;is-cluster=false

Project: 0x01. Git... git branch trouble... Learning the shell... BashGuide/Specia... Learning the shell... linux main place f... explainshell.com -...

aws Services Search [Option+S]

N. Virginia EJIRO @ 1474-9089-5876

three-tier-db

Summary

DB identifier three-tier-db	CPU 3.90%	Status Available	Class db.t2.micro
Role Instance	Current activity 0 Connections	Engine MySQL Community	Region & AZ us-east-1b

Connectivity & security Monitoring Logs & events Configuration Maintenance & backups Tags

Connectivity & security

Endpoint & port Endpoint three-tier-db.cfy4cc8daeq.us-east-1.rds.amazonaws.com	Networking Availability Zone us-east-1b	Security VPC security groups three-tier_rds_sg-sg-0acc367a924e1ce67
--	---	---

CloudShell Feedback Language

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

If we copy the load balancer endpoint we got from our Terraform output, and place it in the search bar, we will see the message we specified in our script for the Apache webserver. If we refresh the page, we should see the IP

address from the other instance in our frontend autoscaling group.

Conclusion

Congratulations! You have successfully deployed a three-tier architecture on AWS using Terraform. This architecture provides a scalable and highly available infrastructure for your applications. Make sure to follow AWS best practices and security recommendations when deploying your production workloads.