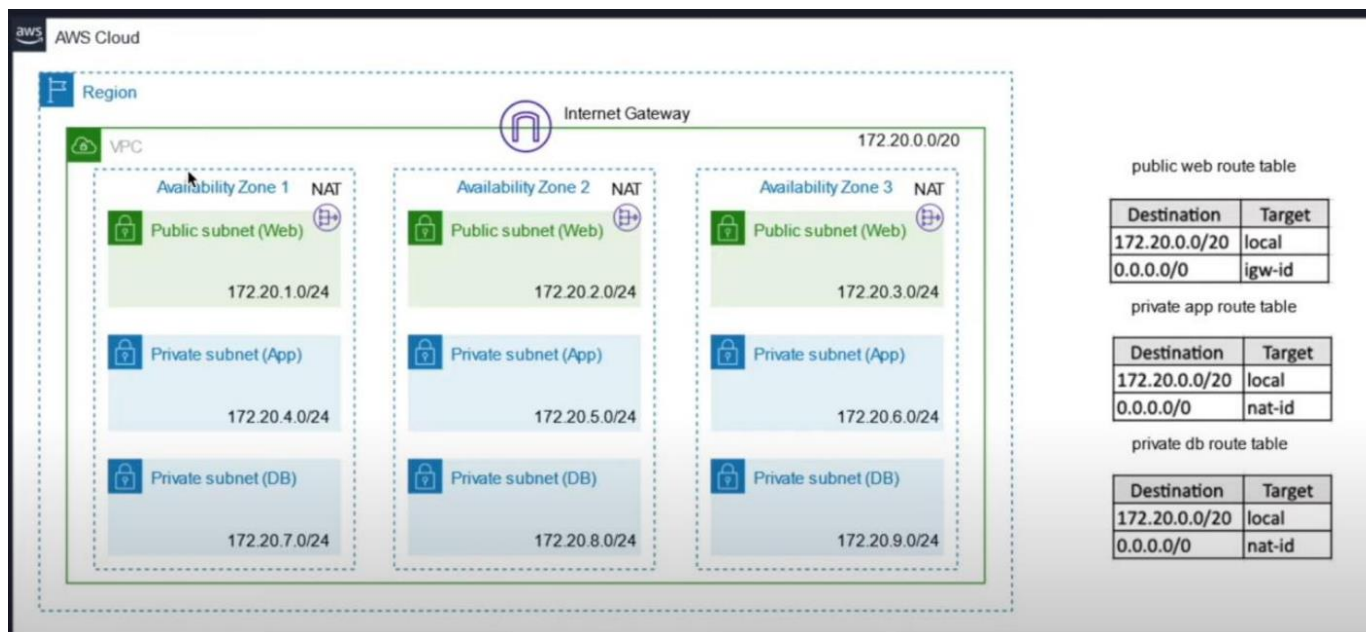


3-TIER ARCHITECTURE PROJECT USING TERRAFORM

The aim of this project is to use terraform infrastructure as a code (IAC) to design and deploy a highly secure and scalable three-tier application infrastructure on Amazon Web Services (AWS) for an online pharmaceutical store. The primary objective is to establish a robust Virtual Private Cloud (VPC) that ensures the web application's continuous operation while adhering to stringent healthcare sector privacy and security standards.

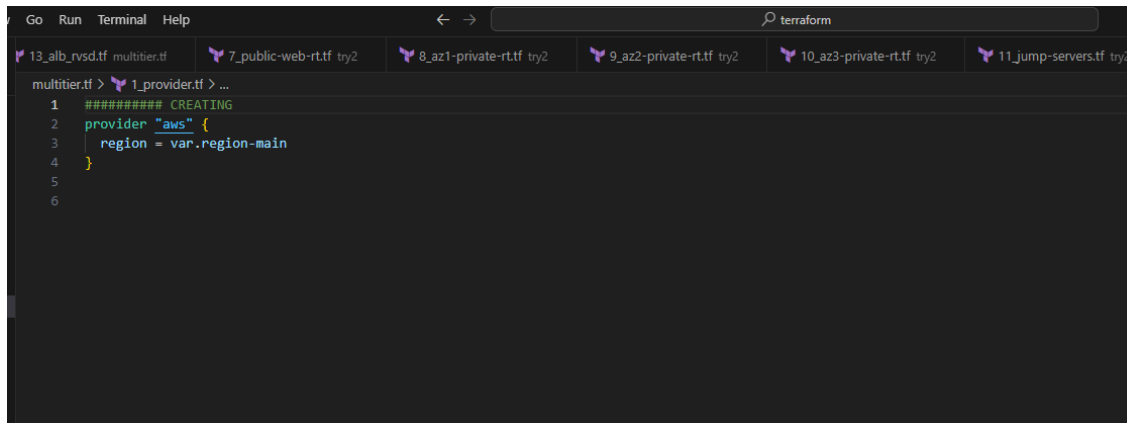
The infrastructure required the creation of distinct network layers, including public subnets for internet-facing components and private subnets for the application and database layers. Key components of this architecture included an Internet Gateway (IGW), a Network Address Translation (NAT) Gateway, and an Application Load Balancer (ALB), all strategically positioned across multiple Availability Zones (AZs) to guarantee high availability and fault tolerance.

As part of the deployment process, the project involved setting up EC2 instances within these subnets, with a public-facing jump server in the public subnet and application and database servers in the private subnets. The application servers were bootstrapped with a lampstack script, excluding the database setup, as an RDS instance was employed to meet the database needs.



1. CREATED A PROVISIONER: 1_provider.tf

The provider block configures Terraform to interact with AWS, specifying the region where all resources will be provisioned. By setting the region to “var.region”-main, this block ensures that all resources are deployed in the designated AWS region, aligning with the project's geographical and compliance requirements.

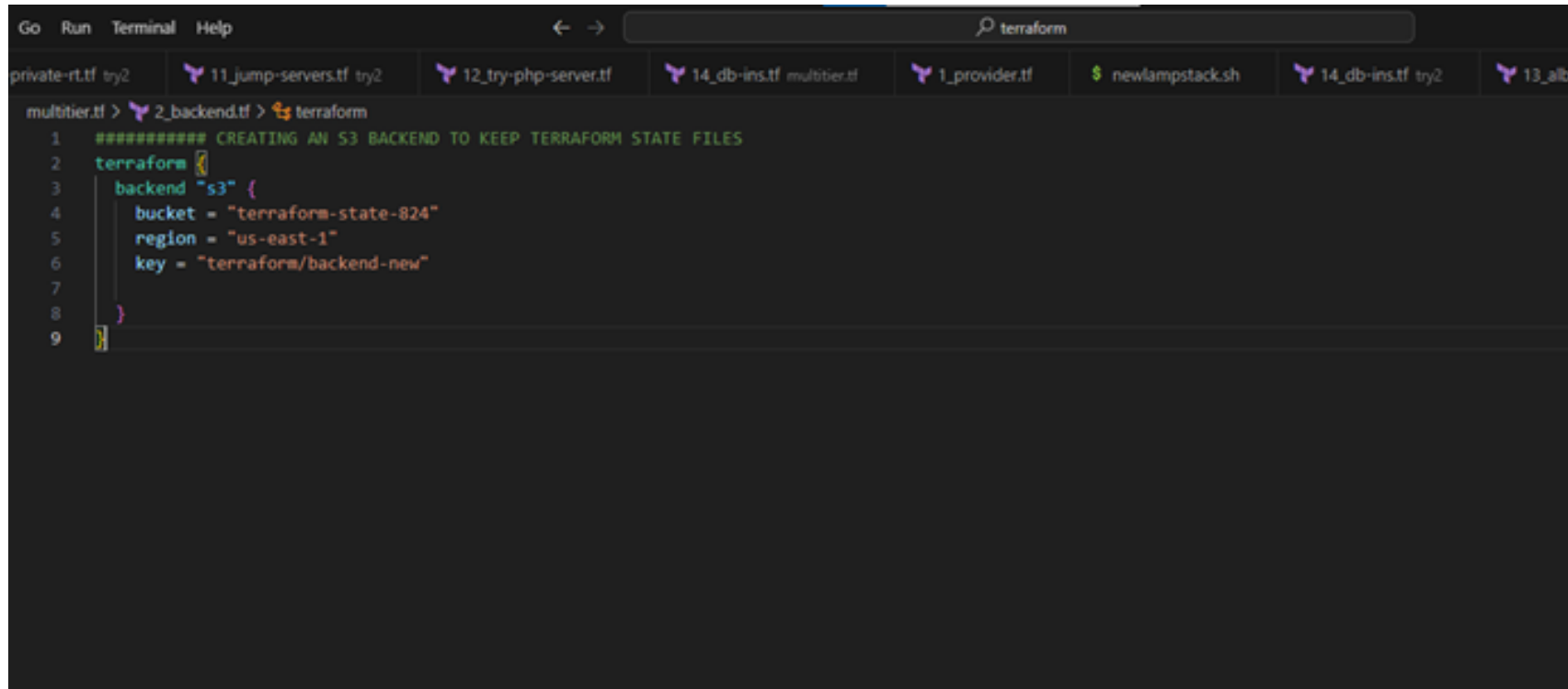


The screenshot shows a code editor with a dark theme. At the top, there is a menu bar with 'Go', 'Run', 'Terminal', and 'Help'. Below the menu bar, there is a search bar with the text 'terraform'. The editor displays a file named '1_provider.tf' with the following content:

```
1 ##### CREATING
2 provider "aws" {
3     region = var.region-main
4 }
5
6
```

2. CREATING S3 BACKEND TO STORE TERRAFORM STATE FILES – 2 *backend.tf*

The S3 backend is crucial for storing the Terraform state files, which maintain the mapping between the resources in the configuration and the real-world resources in AWS. By storing these state files in an S3 bucket, the Terraform environment is set up for collaboration, allowing multiple team members to access the state information in a secure and scalable manner. This setup also facilitates the use of versioning to manage changes to the infrastructure over time.



The screenshot shows a code editor with a dark theme. At the top, there is a menu bar with 'Go', 'Run', 'Terminal', and 'Help'. Below the menu bar is a search bar with the text 'terraform'. A tab bar below the search bar contains several tabs: 'private-rt.tf try2', '11_jump-servers.tf try2', '12_try-php-server.tf', '14_db-ins.tf multitier.tf', '1_provider.tf', '\$ newlampstack.sh', '14_db-ins.tf try2', and '13_alb'. The active tab is '14_db-ins.tf multitier.tf'. The main editor area shows the following code:

```
1 ##### CREATING AN S3 BACKEND TO KEEP TERRAFORM STATE FILES
2 terraform {
3     backend "s3" {
4         bucket = "terraform-state-824"
5         region = "us-east-1"
6         key = "terraform/backend-new"
7     }
8 }
9 }
```

3. CREATING A VPC TO ACCOMMODATE THE ARCHITECTURE: 3 vpc.tf

Virtual Private Cloud (VPC) will be used to set up an isolated network environment that provides full control over IP addressing, subnets, routing, and network gateways. This VPC serves as the environment where all other resources in architecture will reside.

```
Go Run Terminal Help terraform
14_db-ins.tf multitier.tf 1_provider.tf newlampstack.sh 14_db-ins.tf try2 13_alb_rvsd.tf try2 2_backend.tf try2 2_backend.tf multitier.tf 3_vpc.tf multitier.tf x 5_igw.tf multitier.tf 4_subnets.tf ...

multitier.tf > 3_vpc.tf > ...
1 ##### CREATING A VPC FOR THE ARCHITECTURE
2 resource "aws_vpc" "multitier-vpc" {
3     cidr_block      = var.vpc-cidr-block
4     instance_tenancy = "default"
5     enable_dns_hostnames = true
6     enable_dns_support = true
7
8     tags = {
9         Name = var.vpc-name
10     }
11 }
12
13
```

VPC > Your VPCs > vpc-09ea485f51149f872

vpc-09ea485f51149f872 / multitier-vpc Actions ▼

Details Info

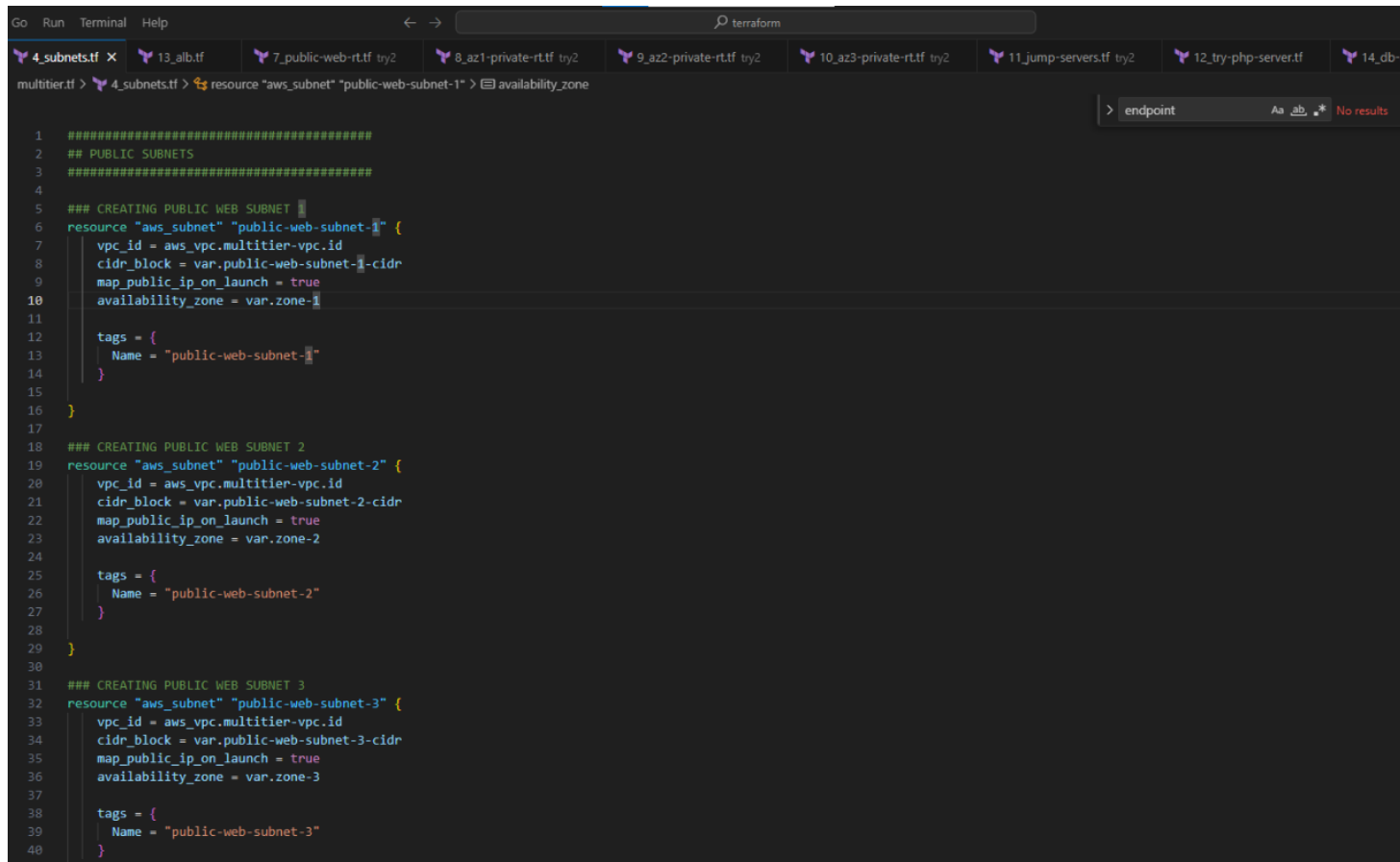
VPC ID vpc-09ea485f51149f872	State Available	DNS hostnames Enabled	DNS resolution Enabled
Tenancy Default	DHCP option set dopt-035edd748e09fd3cc	Main route table rtb-0fd8a5c1639012935	Main network ACL acl-034c63290b9a9bfd5
Default VPC No	IPv4 CIDR 172.20.0.0/20	IPv6 pool -	IPv6 CIDR (Network border group) -
Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 010526276130	

Resource map Info

Resource map CIDRs Flow logs Tags Integrations

4. CREATING PUBLIC SUBNETS TO HOST JUMP SERVERS: 4 *subnets.tf*

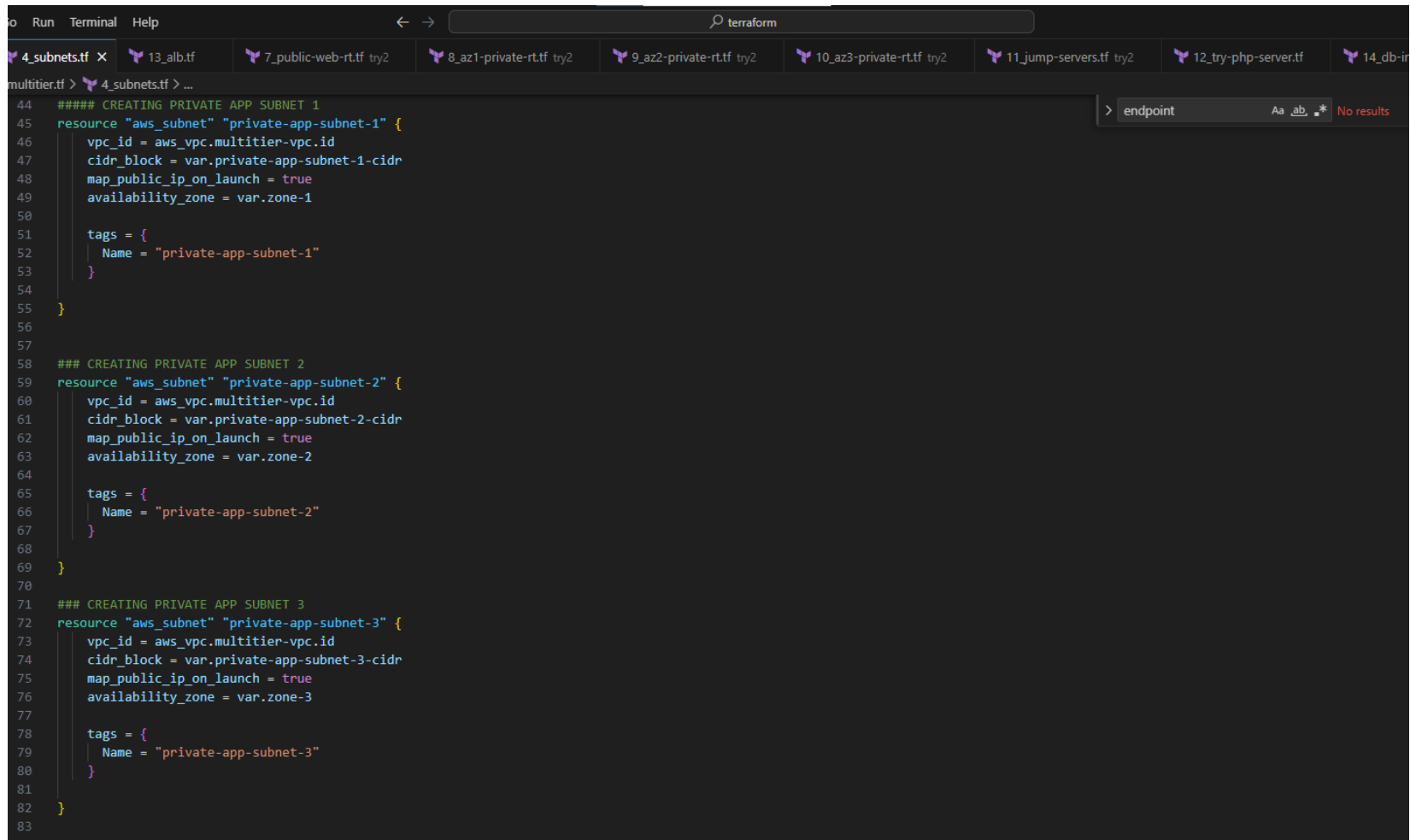
Public subnets are designed to host the jump-servers that need to be accessible from the internet. The jump server acts as a secure gateway through which administrators can access servers in private subnets. By placing the jump server in a public subnet, you allow SSH access to it, while keeping your application and database servers isolated in private subnets.



```
1 #####
2 ## PUBLIC SUBNETS
3 #####
4
5 ### CREATING PUBLIC WEB SUBNET 1
6 resource "aws_subnet" "public-web-subnet-1" {
7   vpc_id = aws_vpc.multitier-vpc.id
8   cidr_block = var.public-web-subnet-1-cidr
9   map_public_ip_on_launch = true
10  availability_zone = var.zone-1
11
12  tags = {
13    Name = "public-web-subnet-1"
14  }
15
16 }
17
18 ### CREATING PUBLIC WEB SUBNET 2
19 resource "aws_subnet" "public-web-subnet-2" {
20   vpc_id = aws_vpc.multitier-vpc.id
21   cidr_block = var.public-web-subnet-2-cidr
22   map_public_ip_on_launch = true
23   availability_zone = var.zone-2
24
25   tags = {
26     Name = "public-web-subnet-2"
27   }
28
29 }
30
31 ### CREATING PUBLIC WEB SUBNET 3
32 resource "aws_subnet" "public-web-subnet-3" {
33   vpc_id = aws_vpc.multitier-vpc.id
34   cidr_block = var.public-web-subnet-3-cidr
35   map_public_ip_on_launch = true
36   availability_zone = var.zone-3
37
38   tags = {
39     Name = "public-web-subnet-3"
40   }
41 }
```

5. CREATING PRIVATE SUBNETS TO HOST PHP AND DB SERVERS: 4 subnet.tf

Private subnets are used to host sensitive components of the application that should not be directly accessible from the internet. In this project, the PHP application server and the database server are placed in private subnets to ensure they are protected from external threats, while still being able to communicate with each other and the public subnet via appropriate routing rules.



The screenshot shows a code editor with a dark theme. The top bar includes a search bar with the text 'terraform'. Below the search bar is a tab bar with several open files: '4_subnets.tf' (active), '13_alb.tf', '7_public-web-rt.tf try2', '8_az1-private-rt.tf try2', '9_az2-private-rt.tf try2', '10_az3-private-rt.tf try2', '11_jump-servers.tf try2', '12_try-php-server.tf', and '14_db-ir'. The main editor area displays the content of '4_subnets.tf', which contains Terraform code for creating three private subnets. The code is as follows:

```
44 ##### CREATING PRIVATE APP SUBNET 1
45 resource "aws_subnet" "private-app-subnet-1" {
46     vpc_id = aws_vpc.multitier-vpc.id
47     cidr_block = var.private-app-subnet-1-cidr
48     map_public_ip_on_launch = true
49     availability_zone = var.zone-1
50
51     tags = {
52         Name = "private-app-subnet-1"
53     }
54 }
55
56
57
58 ### CREATING PRIVATE APP SUBNET 2
59 resource "aws_subnet" "private-app-subnet-2" {
60     vpc_id = aws_vpc.multitier-vpc.id
61     cidr_block = var.private-app-subnet-2-cidr
62     map_public_ip_on_launch = true
63     availability_zone = var.zone-2
64
65     tags = {
66         Name = "private-app-subnet-2"
67     }
68 }
69
70
71 ### CREATING PRIVATE APP SUBNET 3
72 resource "aws_subnet" "private-app-subnet-3" {
73     vpc_id = aws_vpc.multitier-vpc.id
74     cidr_block = var.private-app-subnet-3-cidr
75     map_public_ip_on_launch = true
76     availability_zone = var.zone-3
77
78     tags = {
79         Name = "private-app-subnet-3"
80     }
81 }
82
83
84
```

On the right side of the editor, there is a search bar with the text 'endpoint'. Below the search bar, it says 'No results'.

```
Run Terminal Help
terraform
4_subnets.tf x 13_...
multitier.tf > 4_subnets.tf > resource "aws_subnet" "private-db-subnet-3"

84
85 ### CREATING PRIVATE DB SUBNET 1
86 resource "aws_subnet" "private-db-subnet-1" {
87     vpc_id = aws_vpc.multitier-vpc.id
88     cidr_block = var.private-db-subnet-1-cidr
89     map_public_ip_on_launch = true
90     availability_zone = var.zone-1
91
92     tags = {
93         Name = "private-db-subnet-1"
94     }
95 }
96
97
98 ### CREATING PRIVATE DB SUBNET 2
99 resource "aws_subnet" "private-db-subnet-2" {
100     vpc_id = aws_vpc.multitier-vpc.id
101     cidr_block = var.private-db-subnet-2-cidr
102     map_public_ip_on_launch = true
103     availability_zone = var.zone-2
104
105     tags = {
106         Name = "private-db-subnet-2"
107     }
108 }
109
110
111
112 ### CREATING PRIVATE DB SUBNET 3
113 resource "aws_subnet" "private-db-subnet-3" {
114     vpc_id = aws_vpc.multitier-vpc.id
115     cidr_block = var.private-db-subnet-3-cidr
116     map_public_ip_on_launch = true
117     availability_zone = var.zone-3
118
119     tags = {
120         Name = "private-db-subnet-3"
121     }
122 }
123 }
```

Subnets (15) Info

Find resources by attribute or tag

Last updated less than a minute ago

Actions

Create subnet

< 1 >

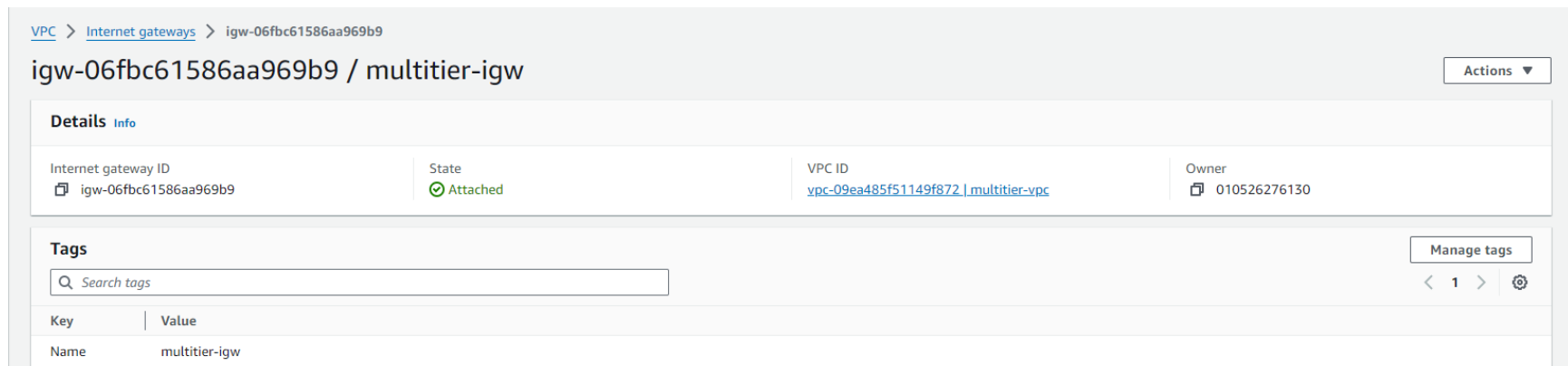
<input type="checkbox"/>	Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR	IPv6 CIDR association ID
<input type="checkbox"/>	public-web-subnet-1	subnet-0cbdff19affedff9	Available	vpc-09ea485f51149f872 multi...	172.20.1.0/24	-	-
<input type="checkbox"/>	public-web-subnet-2	subnet-09d9419cbf38228dd	Available	vpc-09ea485f51149f872 multi...	172.20.2.0/24	-	-
<input type="checkbox"/>	public-web-subnet-3	subnet-012afcdcae043ef8d	Available	vpc-09ea485f51149f872 multi...	172.20.3.0/24	-	-
<input type="checkbox"/>	private-app-subnet-1	subnet-06c6dc501de49b98c	Available	vpc-09ea485f51149f872 multi...	172.20.4.0/24	-	-
<input type="checkbox"/>	private-app-subnet-2	subnet-0511398ee983f5804	Available	vpc-09ea485f51149f872 multi...	172.20.5.0/24	-	-
<input type="checkbox"/>	private-app-subnet-3	subnet-09b00a7a260b32696	Available	vpc-09ea485f51149f872 multi...	172.20.6.0/24	-	-
<input type="checkbox"/>	private-db-subnet-1	subnet-046f0cbf12312a77b	Available	vpc-09ea485f51149f872 multi...	172.20.7.0/24	-	-
<input type="checkbox"/>	private-db-subnet-2	subnet-0ba1229ff7edd5db0	Available	vpc-09ea485f51149f872 multi...	172.20.8.0/24	-	-
<input type="checkbox"/>	private-db-subnet-3	subnet-03d689945786a198b	Available	vpc-09ea485f51149f872 multi...	172.20.9.0/24	-	-

6. CREATING INTERNET GATEWAY TO ALLOW CONNECTION BETWEEN THE INTERNET AND THE VPC: 5 igw.tf

The Internet Gateway (IGW) is attached to the VPC to enable communication between the instances in the VPC and the internet. This is essential for allowing users to access the application hosted in your VPC and for the jump server to communicate with the outside world. The IGW provides a path for internet traffic to reach the public subnet.



```
Go Run Terminal Help
12_try-php-server.tf 14_db-ins.tf multitier.tf 1_provider.tf $ newlampstack.sh 14_db-ins.tf try2 13_alb_rvs
multitier.tf > 5_igw.tf > ...
1 ##### CREATING INTERNET GATEWAY
2 resource "aws_internet_gateway" "multitier-igw" {
3     vpc_id = aws_vpc.multitier-vpc.id
4
5     tags = {
6         Name = "multitier-igw"
7     }
8
9 }
```



VPC > Internet gateways > igw-06fbc61586aa969b9

igw-06fbc61586aa969b9 / multitier-igw

Actions ▾

Details Info

Internet gateway ID igw-06fbc61586aa969b9	State Attached	VPC ID vpc-09ea485f51149f872 multitier-vpc	Owner 010526276130
--	-------------------	---	-----------------------

Tags Manage tags

Search tags

Key	Value
Name	multitier-igw

7. CREATING EIPS AND MAPPING THEM ONTO INDIVIDUAL NAT GATEWAYS: 6-nat-gw:

The NAT Gateways will allow instances in private subnets to connect to the internet while preventing the internet from initiating connections with those instances. By strategically positioning NAT Gateways in each of your public subnets across different Availability Zones (AZs), high availability and fault tolerance is ensured. Each private subnet routes its outbound traffic through the NAT Gateway, Each NAT Gateway is assigned an EIP to ensure consistent outbound IP addresses by mapping an EIP to each NAT Gateway, you ensure that the gateway has a persistent address that does not change even if the NAT Gateway is replaced or restarted.",

```
multitier.tf > 6_nat-gw.tf > ...
1 #####
2 # PUBLIC WEB SUBNET 1 NAT GATEWAY
3 #####
4 resource "aws_eip" "nat-gw-eip1" {
5   domain = "vpc"
6 }
7
8 resource "aws_nat_gateway" "az1-nat-gw" {
9   allocation_id = aws_eip.nat-gw-eip1.id
10  subnet_id = aws_subnet.public-web-subnet-1.id
11  tags = {
12    Name = "az1-nat-gw"
13  }
14  depends_on = [ aws_internet_gateway.multitier-igw ]
15 }
16
17 #####
18 # PUBLIC WEB SUBNET 2 NAT GATEWAY
19 #####
20 resource "aws_eip" "nat-gw-eip2" {
21   domain = "vpc"
22 }
23
24 resource "aws_nat_gateway" "az2-nat-gw" {
25   allocation_id = aws_eip.nat-gw-eip2.id
26   subnet_id = aws_subnet.public-web-subnet-2.id
27   tags = {
28     Name = "az2-nat-gw"
29   }
30   depends_on = [ aws_internet_gateway.multitier-igw ]
31 }
32
33 #####
34 # PUBLIC WEB SUBNET 3 NAT GATEWAY
35 #####
36 resource "aws_eip" "nat-gw-eip3" {
37   domain = "vpc"
38 }
39
40 resource "aws_nat_gateway" "az3-nat-gw" {
41   allocation_id = aws_eip.nat-gw-eip3.id
42   subnet_id = aws_subnet.public-web-subnet-3.id
43   tags = {
44     Name = "az3-nat-gw"
45   }
46   depends_on = [ aws_internet_gateway.multitier-igw ]
47 }
```

8. CREATING 4 ROUTE TABLES TO ROUTE TRAFFIC TO DESIRED DESTINATIONS.

Route tables will control the routing of traffic within the VPC. In this setup, one public route table is associated with the public subnets, which includes a route that directs internet-bound traffic through the Internet Gateway (IGW). This enables resources like the jump server and NAT Gateway in the public subnet to communicate with the internet. Three private route tables, each associated with a different group of two private subnets (PHP and DB) within a specific Availability Zone. Each private route table routes traffic destined for the internet through the corresponding NAT Gateway in the same Availability Zone. " This design ensures that the private instances have internet access while maintaining security and isolation from direct public internet exposure.

PUBLIC WEB ROUT TABLES

```
Go Run Terminal Help
terraform
7_public-web-rt.tf multitier.tf X 8_az1-private-rt.tf multitier.tf 6_nat-gw.tf multitier.tf 9_az2-private-rt.tf multitier.tf 10_az3-private-rt.tf multitier.tf 11_jump-servers.tf multitier.tf 12_php-servers.tf multitier.tf
multitier.tf > 7_public-web-rt.tf > ...
1 #####
2 # PUBLIC WEB ROUTE TABLE
3 #####
4
5 resource "aws_route_table" "public-web-rt" {
6     vpc_id = aws_vpc.multitier-vpc.id
7
8     route {
9         cidr_block = "0.0.0.0/0"
10        gateway_id = aws_internet_gateway.multitier-igw.id
11    }
12
13    tags = {
14        Name = "public-web-rt"
15    }
16 }
17
18 #####
19 # ROUTE TABLE ASSOCIATION FOR PUBLIC SUBNETS
20 #####
21
22 resource "aws_route_table_association" "public-web-subnet-1-rta" {
23     subnet_id      = aws_subnet.public-web-subnet-1.id
24     route_table_id = aws_route_table.public-web-rt.id
25 }
26
27 resource "aws_route_table_association" "public-web-subnet-2-rta" {
28     subnet_id      = aws_subnet.public-web-subnet-2.id
29     route_table_id = aws_route_table.public-web-rt.id
30 }
31
32 resource "aws_route_table_association" "public-web-subnet-3-rta" {
33     subnet_id      = aws_subnet.public-web-subnet-3.id
34     route_table_id = aws_route_table.public-web-rt.id
35 }
36
37
```

rtb-04bd426e391802fb4 / public-web-rt

Actions ▾

Details [Info](#)

Route table ID rtb-04bd426e391802fb4	Main No	Explicit subnet associations 3 subnets	Edge associations -
VPC vpc-09ea485f51149f872 multitier-vpc	Owner ID 010526276130		

- Routes
- Subnet associations**
- Edge associations
- Route propagation
- Tags

Explicit subnet associations (3)

Edit subnet associations

Find subnet association

< 1 > ⚙

Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
public-web-subnet-1	subnet-0cbdff19afffedff9	172.20.1.0/24	-
public-web-subnet-3	subnet-012afcdfae043ef8d	172.20.3.0/24	-
public-web-subnet-2	subnet-09d9419cbf38228dd	172.20.2.0/24	-

Subnets without explicit associations (0)

Edit subnet associations

The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:

Find subnet association

< 1 > ⚙

Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
--------	-------------	-------------	-------------

Activate Windows
Go to Settings to activate Windows

PRIVATE SUBNETS' ROUTE TABLES: *8_az1-private-rt.tf*

```
multitier.tf > 8_az1-private-rt.tf > ...
1 #####
2 # AZ-1 PRIVATE ROUTE TABLE
3 #####
4
5 resource "aws_route_table" "az1-private-rt" {
6   vpc_id = aws_vpc.multitier-vpc.id
7
8   route {
9     cidr_block = "0.0.0.0/0"
10    gateway_id = aws_nat_gateway.az1-nat-gw.id
11  }
12
13  tags = {
14    Name = "az1-private-rt"
15  }
16 }
17
18 #####
19 # ROUTE TABLE ASSOCIATION FOR AZ1 PRIVATE SUBNETS
20 #####
21
22 resource "aws_route_table_association" "az1-private-app-subnet-1-rta" {
23   subnet_id   = aws_subnet.private-app-subnet-1.id
24   route_table_id = aws_route_table.az1-private-rt.id
25 }
26
27 resource "aws_route_table_association" "az1-private-db-subnet-1-rta" {
28   subnet_id   = aws_subnet.private-db-subnet-1.id
29   route_table_id = aws_route_table.az1-private-rt.id
30 }
31
32
```

VPC > Route tables > rtb-01add9c9e9a3aca0f

rtb-01add9c9e9a3aca0f / az1-private-rt

Actions ▾

Details Info

Route table ID 📄 rtb-01add9c9e9a3aca0f	Main 📄 No	Explicit subnet associations 2 subnets	Edge associations -
VPC vpc-09ea485f51149f872 multitier-vpc	Owner ID 📄 010526276130		

Routes Subnet associations Edge associations Route propagation Tags

Routes (2) Both ▾ Edit routes

🔍 Filter routes

Destination ▾	Target ▾	Status ▾	Propagated ▾
0.0.0.0/0	nat-06a77a0569100cde7	🟢 Active	No
172.20.0.0/20	local	🟢 Active	No

PRIVATE SUBNETS' ROUTE TABLES: 9_az2-private-rt

```
multitier.tf > 9_az2-private-rt.tf > ...
1 #####
2 # AZ-2 PRIVATE ROUTE TABLE
3 #####
4
5 resource "aws_route_table" "az2-private-rt" {
6   vpc_id = aws_vpc.multitier-vpc.id
7
8   route {
9     cidr_block = "0.0.0.0/0"
10    gateway_id = aws_nat_gateway.az2-nat-gw.id
11  }
12
13  tags = {
14    Name = "az2-private-rt"
15  }
16 }
17
18 #####
19 # ROUTE TABLE ASSOCIATION FOR AZ2 PRIVATE SUBNETS
20 #####
21
22 resource "aws_route_table_association" "az2-private-app-subnet-2-rta" {
23   subnet_id = aws_subnet.private-app-subnet-2.id
24   route_table_id = aws_route_table.az2-private-rt.id
25 }
26
27 resource "aws_route_table_association" "az2-private-db-subnet-2-rta" {
28   subnet_id = aws_subnet.private-db-subnet-2.id
29   route_table_id = aws_route_table.az2-private-rt.id
30 }
31
```

VPC > Route tables > rtb-0c5fbbd46b703bbb3

rtb-0c5fbbd46b703bbb3 / az2-private-rt

Actions ▾

Details info

Route table ID rtb-0c5fbbd46b703bbb3	Main No	Explicit subnet associations 2 subnets	Edge associations -
VPC vpc-09ea485f51149f872 multitier-vpc	Owner ID 010526276130		

Routes | **Subnet associations** | Edge associations | Route propagation | Tags

Explicit subnet associations (2) Edit subnet associations

Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
private-db-subnet-2	subnet-0ba1229ff7edd5db0	172.20.8.0/24	-
private-app-subnet-2	subnet-0511398ee983f5804	172.20.5.0/24	-

Subnets without explicit associations (0) Edit subnet associations

The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:

Name ▾	Subnet ID ▾	IPv4 CIDR ▾	IPv6 CIDR ▾
--------	-------------	-------------	-------------

PRIVATE SUBNETS' ROUTE TABLES: *10_az3-private-rt*

```
multitier.tf > 10_az3-private-rt.tf > resource "aws_route_table_association" "az3-private-app-subnet-3-rta" > subnet_id
1 #####
2 # AZ-3 PRIVATE ROUTE TABLE
3 #####
4
5 resource "aws_route_table" "az3-private-rt" {
6   vpc_id = aws_vpc.multitier-vpc.id
7
8   route {
9     cidr_block = "0.0.0.0/0"
10    gateway_id = aws_nat_gateway.az3-nat-gw.id
11  }
12
13  tags = {
14    Name = "az3-private-rt"
15  }
16 }
17
18 #####
19 # ROUTE TABLE ASSOCIATION FOR AZ3 PRIVATE SUBNETS
20 #####
21
22 resource "aws_route_table_association" "az3-private-app-subnet-3-rta" {
23   subnet_id = aws_subnet.private-app-subnet-3.id
24   route_table_id = aws_route_table.az3-private-rt.id
25 }
26
27 resource "aws_route_table_association" "az3-private-db-subnet-3-rta" {
28   subnet_id = aws_subnet.private-db-subnet-3.id
29   route_table_id = aws_route_table.az3-private-rt.id
30 }
```

VPC > Route tables > rtb-0d88a7cd00bda4edd

rtb-0d88a7cd00bda4edd / az3-private-rt

Actions

Details info

Route table ID rtb-0d88a7cd00bda4edd	Main No	Explicit subnet associations 2 subnets	Edge associations -
VPC vpc-09ea485f1149f872 multitier-vpc	Owner ID 010526276130		

Routes | **Subnet associations** | Edge associations | Route propagation | Tags

Explicit subnet associations (2) Edit subnet associations

Find subnet association

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
private-app-subnet-3	subnet-09b00a7a260b32696	172.20.6.0/24	-
private-db-subnet-3	subnet-05d689945786a198b	172.20.9.0/24	-

Subnets without explicit associations (0) Edit subnet associations

The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:

Find subnet association

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
------	-----------	-----------	-----------

9. CREATING JUMP SERVERS : 11 *jump-servers.tf*

The jump server serves as an intermediary device which will be used to connect to other servers within the private subnets. By creating and configuring the jump server in a public subnet, a secure point of access is provided to manage the private infrastructure, reducing the need to expose sensitive components directly to the internet. To ensure secure and controlled access, a dedicated security group was created specifically for the jump servers, allowing inbound SSH traffic on port 22 and HTTP traffic on port 80 from any IP address. This setup ensures that any other administrator, can securely access the jump server while also enabling basic web traffic if necessary. The security group was configured to permit outbound traffic on all ports and protocols, allowing the jump server to communicate freely with other resources and the internet. The first jump servers were provisioned in each availability zone, configuring it with the appropriate security group, AMI, instance type, and key pair for secure SSH access. I then deployed additional jump servers in the remaining availability zones to ensure redundancy and high availability across the infrastructure.

```
multitier.tf > 11_jump-servers.tf > resource "aws_vpc_security_group_egress_rule" "allow_all_out_traffic_js"
1 ##### CREATING SECURITY GROUP FOR JUMPSERVER
2 resource "aws_security_group" "jump-server-sg" {
3     name           = "jump-server-sg"
4     description    = "jump-server-sg"
5     vpc_id         = aws_vpc.multipier-vpc.id
6
7     tags = {
8         Name = "jump-server-sg"
9     }
10 }
11
12 ##### ALLOWING INBOUND SSH TRAFFIC FROM EVERYWHERE TO THE JUMPSERVER
13 resource "aws_vpc_security_group_ingress_rule" "allow_ssh_js" {
14     security_group_id = aws_security_group.jump-server-sg.id
15     cidr_ipv4         = "0.0.0.0/0"
16     from_port         = 22
17     ip_protocol       = "tcp"
18     to_port           = 22
19 }
20
21 ##### ALLOWING INBOUND HTTP TRAFFIC FROM EVERYWHERE TO JUMP SERVER
22 resource "aws_vpc_security_group_ingress_rule" "allow_http_js" {
23     security_group_id = aws_security_group.jump-server-sg.id
24     cidr_ipv4         = "0.0.0.0/0"
25     from_port         = 80
26     ip_protocol       = "tcp"
27     to_port           = 80
28 }
29
30 ##### ALLOWING OUTBOUND TRAFFIC FROM JUMP SERVER TO EVERYWHERE ON ALL PORTS AND PROTOCOLS
31 resource "aws_vpc_security_group_egress_rule" "allow_all_out_traffic_js" {
32     security_group_id = aws_security_group.jump-server-sg.id
```

```

11_jump-server.tf multiback x 12_php-server.tf 12_php-server.tf 4_subnets.tf 13_alb.tf 7_public-web-rttf try2 8_a21-private-rttf try2 9_a22-private-rttf try2 10_a23-private-rttf try2 11_jump-server.tf try2
multitier.tf > 11_jump-server.tf > resource "aws_instance" "jump-server-2"
28
29
30 ##### ALLOWING OUTBOUND TRAFFIC FROM JUMP SERVER TO EVERYWHERE ON ALL PORTS AND PROTOCOLS
31 resource "aws_vpc_security_group_egress_rule" "allow_all_out_traffic_js" {
32   security_group_id = aws_security_group_jump-server-sg.id
33   cidr_ipv4         = "0.0.0.0/0"
34   ip_protocol       = "-1"
35 }
36
37 ##### CREATING 1ST JUMP SERVER IN AZ1
38 resource "aws_instance" "jump-server-1" {
39   ami           = var.ec2-instance-ami
40   instance_type = var.ec2-instance-type
41   key_name      = var.ec2-key-name
42   vpc_security_group_ids = [aws_security_group_jump-server-sg.id]
43   subnet_id     = aws_subnet_public-web-subnet-1.id
44   availability_zone = var.zone-1
45
46   tags = {
47     Name = "jump-server-1"
48   }
49 }
50
51 ##### CREATING 2ND JUMP SERVER IN AZ2
52 resource "aws_instance" "jump-server-2" {
53   ami           = var.ec2-instance-ami
54   instance_type = var.ec2-instance-type
55   key_name      = var.ec2-key-name
56   vpc_security_group_ids = [aws_security_group_jump-server-sg.id]
57   subnet_id     = aws_subnet_public-web-subnet-2.id
58   availability_zone = var.zone-2
59
60   tags = {
61     Name = "jump-server-2"
62   }
63 }
64
65 ##### CREATING 3RD JUMP SERVER IN AZ3
66 resource "aws_instance" "jump-server-3" {
67   ami           = var.ec2-instance-ami
68   instance_type = var.ec2-instance-type
69   key_name      = var.ec2-key-name
70   vpc_security_group_ids = [aws_security_group_jump-server-sg.id]
71   subnet_id     = aws_subnet_public-web-subnet-3.id
72   availability_zone = var.zone-3
73
74   tags = {
75     Name = "jump-server-3"
76   }
77 }
78
79

```

EC2 > Security Groups > sg-04a551b81453005d1 - jump-server-sg

sg-04a551b81453005d1 - jump-server-sg Actions ▾

Details

Security group name jump-server-sg	Security group ID sg-04a551b81453005d1	Description jump-server-sg	VPC ID vpc-02ea485f51149f872
Owner 010526276130	Inbound rules count 2 Permission entries	Outbound rules count 1 Permission entry	

Inbound rules

Outbound rules

Tags

Inbound rules (2)

🔄

Manage tags

Edit inbound rules

< 1 > ⚙

<input type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range	Source
<input type="checkbox"/>	-	sgr-01be1dfed6c9fec08	IPv4	HTTP	TCP	80	0.0.0.0/0
<input type="checkbox"/>	-	sgr-053a0575497898...	IPv4	SSH	TCP	22	0.0.0.0/0

Instances (1/3) Info

All states ▾

jump X

Clear filters

Refresh

Connect

Instance state ▾

Actions ▾

Launch instances ▾

< 1 >

⚙

	Name ↗ ▾	Instance ID	Instance state ▾	Instance type ▾	Status check	Alarm status	Availability Zone ▾	Public IPv4 DNS ▾	Public IPv4 .
<input checked="" type="checkbox"/>	jump-server-1	i-0c04d1d625c8666c2	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1a	ec2-54-161-7-227.com...	54.161.7.227
<input type="checkbox"/>	jump-server-2	i-0a95c64dfca5776a9	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1b	ec2-44-200-56-63.com...	44.200.56.63
<input type="checkbox"/>	jump-server-3	i-02c8449dbf352ecb0	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c	ec2-54-173-36-142.co...	54.173.36.142

i-0c04d1d625c8666c2 (jump-server-1)

⚙ X

Details

Status and alarms

Monitoring

Security

Networking

Storage

Tags

▼ Instance summary Info

Instance ID

i-0c04d1d625c8666c2 (jump-server-1)

IPv6 address

-

Hostname type

IP name: ip-172-20-1-34.ec2.internal

Answer private resource DNS name

-

Public IPv4 address

54.161.7.227 | open address

Instance state

Running

Private IP DNS name (IPv4 only)

ip-172-20-1-34.ec2.internal

Instance type

t2.micro

Private IPv4 addresses

172.20.1.34

Public IPv4 DNS

ec2-54-161-7-227.compute-1.amazonaws.com | open address

Elastic IP addresses

-

Activate Windows

Go to Settings to activate Windows.

10. CREATING PHP SERVER: 12_php-server.tf

The PHP server hosts the application logic of the e-commerce platform and is crucial for serving dynamic content to users. By deploying the PHP server in a private subnet, it is shielded from direct internet exposure, ensuring enhanced security while still being able to handle incoming requests routed through the Application Load Balancer (ALB). To ensure secure and controlled access, a dedicated security group was created specifically for the PHP servers, allowing inbound HTTP traffic from the ALB security group and any IP address, as well as SSH traffic from the jump server. This setup ensures that administrators can securely manage the PHP server while also enabling it to communicate with other essential components. The security group was configured to permit outbound traffic on all ports and protocols, allowing the PHP server to freely interact with other resources and the internet.

The PHP servers were provisioned in each availability zone, configured with the appropriate security group, AMI, instance type, and private IP address to maintain security within the private subnet. A key pair was created and associated with the servers for secure SSH access. A file provisioner was used to transfer the LAMP stack script to the server, followed by a remote-exec provisioner that executed the script to install and configure the necessary software components, including the integration of the database server's endpoint into the application configuration. This setup was replicated across the remaining availability zones to ensure redundancy and high availability throughout the infrastructure.

```
multiboot> 12_php-server.tf > resource "aws_instance" "php-server-1" > provisioner "file" > destination
1 ##### CREATING KEY PAIR FOR INSTANCES
2 resource "aws_key_pair" "terra-key" {
3   key_name   = var.ec2.key_name
4   public_key = file("terra-key.pub")
5 }
6
7 ##### CREATING SECURITY GROUP FOR PHP-SERVERS
8 resource "aws_security_group" "php-server-sg" {
9   name        = "php-server-sg"
10  description = "php-server-sg"
11  vpc_id      = aws_vpc.multitier-vpc.id
12
13
14 ##### ALLOWING INBOUND HTTP TRAFFIC FROM ALB SECURITY GROUP
15 ingress {
16   from_port = 80
17   to_port   = 80
18   protocol  = "tcp"
19   security_groups = [ aws_security_group.alb-sg.id ]
20 }
21
22 ##### ALLOWING INBOUND HTTP TRAFFIC FROM EVERYWHERE
23 ingress {
24   cidr_blocks = [ "0.0.0.0/0" ]
25   from_port   = 80
26   to_port     = 80
27   protocol    = "tcp"
28 }
29
30 ##### ALLOWING INBOUND SSH TRAFFIC FROM JUMP SERVER
31 ingress {
32   security_groups = [ aws_security_group.jump-server-sg.id ]
33   from_port       = 22
34   to_port         = 22
35   protocol        = "tcp"
36 }
37
38 ##### ALLOWING INBOUND HTTP TRAFFIC FROM ALB
39 egress {
40   cidr_blocks = [ "0.0.0.0/0" ]
41   from_port   = 0
42   to_port     = 0
43   protocol    = "-1"
44 }
45
46 tags = {
47   Name = "php-server-sg"
48 }
49
```

EC2 > Security Groups > sg-03861aa786fb6da33 - php-server-sg

sg-03861aa786fb6da33 - php-server-sg

Actions

Details

Security group name
php-server-sg

Security group ID
sg-03861aa786fb6da33

Description
php-server-sg

VPC ID
vpc-09ea485f51149f872

Owner
010526276130

Inbound rules count
3 Permission entries

Outbound rules count
1 Permission entry

Inbound rules

Outbound rules

Tags

Inbound rules (3)

Manage tags

Edit inbound rules

Search

	Name	Security group rule...	IP version	Type	Protocol	Port range	Source
<input type="checkbox"/>	-	sgr-0956faae3533182d	-	HTTP	TCP	80	sg-0f3ac5ca0c972
<input type="checkbox"/>	-	sgr-05d2bc8e85260cf21	IPv4	HTTP	TCP	80	0.0.0.0/0
<input type="checkbox"/>	-	sgr-098ff3304daf2699d	-	SSH	TCP	22	sg-04a551b81453

Activate Windows

```

multitier.tf > resource "aws_instance" "php-server-1" > provider "file" > destination
56 resource "aws_instance" "php-server-1" {
57   key_name          = var.ec2_key_name
58   vpc_security_group_ids = [aws_security_group.php-server-sg.id]
59   subnet_id         = aws_subnet.private-app-subnet-1.id
60   availability_zone   = var.zone-1
61   associate_public_ip_address = false
62   tags = {
63     Name = "php-server-1"
64   }
65   ##### CREATING FILE PROVISIONER TO TRANSFER LAMPSTACK FILE TO PHP SERVER
66   provisioner "file" {
67     source      = "lampstack.sh"
68     destination = "/tmp/lampstack.sh"
69     on_failure  = continue
70     connection {
71       type        = "ssh"
72       user        = "ec2-user"
73       host        = self.private_ip
74       private_key = file("terra-key")
75       bastion_host = aws_instance.jump-server-1.public_ip
76       bastion_user = "ec2-user"
77       bastion_private_key = file("terra-key")
78     }
79   }
80   ##### CREATING A REMOTE-EXEC PROVISIONER TO EXECUTE LAMPSTACK SCRIPT AND INSERT ENDPOINT OF DB SERVER
81   provisioner "remote-exec" {
82     inline = [
83       "chmod u+x /tmp/lampstack.sh",
84       "sudo /tmp/lampstack.sh",
85       "db_endpoint=$(data.aws_db_instance.db_instance.endpoint)",
86       "sudo sed -i 's/db_endpoint/'$db_endpoint'/'g' /var/www/html/config.sample.inc.php",
87       "sudo mv /var/www/html/config.sample.inc.php /var/www/html/config.inc.php "
88     ]
89     connection {
90       type        = "ssh"
91       user        = "ec2-user"
92       host        = self.private_ip
93       private_key = file("terra-key")
94       bastion_host = aws_instance.jump-server-1.public_ip
95       bastion_user = "ec2-user"
96       bastion_private_key = file("terra-key")
97     }
98   }
99 }

```

Instances (1/3)
Info

Find Instance by attribute or tag (case-sensitive)
Running

php
Clear filters

1

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
<input checked="" type="checkbox"/>	php-server-1	i-042ddfdbbaad0ee2a	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	-	-
<input type="checkbox"/>	php-server-3	i-0fa5804e213615f7f	Running	t2.micro	2/2 checks passed	View alarms	us-east-1c	-	-
<input type="checkbox"/>	php-server-2	i-043847de540f9bf27	Running	t2.micro	2/2 checks passed	View alarms	us-east-1b	-	-

i-042ddfdbbaad0ee2a (php-server-1)

Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags

Instance summary
Info

Instance ID
i-042ddfdbbaad0ee2a (php-server-1)

IPV6 address
-

Hostname type
IP name: ip-172-20-4-115.ec2.internal

Answer private resource DNS name
-

Public IPv4 address
-

Instance state
Running

Private IP DNS name (IPv4 only)
ip-172-20-4-115.ec2.internal

Instance type
t2.micro

Private IPv4 addresses
172.20.4.115

Public IPv4 DNS
-

Elastic IP addresses
-

Activate Windows
Go to Settings to activate Windows.

11. Creating ALB : *13_alb.tf*

The Application Load Balancer (ALB) will be used for distributing incoming web traffic across multiple targets, ensuring that the application remains highly available and can handle varying levels of traffic efficiently. The ALB is configured to distribute traffic to the PHP servers that host the application.

To secure the ALB, a dedicated security group was created, which allows inbound HTTP traffic on port 80 from any IP address. This setup ensures that the ALB can receive traffic from the internet and distribute it to the appropriate backend servers. The security group also permits outbound traffic on all ports and protocols, allowing the ALB to communicate freely with the registered targets, such as the PHP servers.

A target group was created to manage the PHP servers that the ALB will forward traffic to. Each PHP server was registered as a target in this group, enabling the ALB to distribute incoming requests evenly across the available servers, enhancing load distribution and fault tolerance.

The ALB itself was provisioned across multiple public subnets, ensuring high availability by distributing traffic across different availability zones. An HTTP listener was configured for the ALB, which listens for incoming traffic on port 80 and forwards it to the registered PHP servers via the target group. This configuration ensures that incoming requests are efficiently managed and routed to the appropriate resources within the infrastructure.

multitier.tf > 13_alb.tf > resource "aws_lb_listener" "http" > default_action > target_group_arn

```
1  ##### CREATING ALB SECURITY GROUP
2  resource "aws_security_group" "alb-sg" {
3    name           = "alb-sg"
4    description    = "security_group_for_alb"
5    vpc_id         = aws_vpc.multitier-vpc.id
6
7    tags = {
8      Name = "alb-sg"
9    }
10
11  }
12
13
14  ##### ALLOWING INBOUND HTTP TRAFFIC FROM THE INTERNET TO THE ALB
15  resource "aws_vpc_security_group_ingress_rule" "allow_http_alb" {
16    security_group_id = aws_security_group.alb-sg.id
17    cidr_ipv4         = "0.0.0.0/0"
18    from_port         = 80
19    ip_protocol        = "tcp"
20    to_port            = 80
21  }
22
23  ##### ALLOWING OUTBOUND TRAFFIC FROM THE ALB TO EVERYWHERE
24  resource "aws_vpc_security_group_egress_rule" "allow_all_out_traffic_alb" {
25    security_group_id = aws_security_group.alb-sg.id
26    cidr_ipv4         = "0.0.0.0/0"
27    ip_protocol        = "-1"
28  }
29
30  ##### CREATING TARGET GROUP FOR ALB
31  resource "aws_lb_target_group" "alb-tg" {
32    name           = "alb-tg"
33    target_type    = "instance"
34    port           = 80
35    protocol        = "HTTP"
36    vpc_id         = aws_vpc.multitier-vpc.id
37  }
38
39
40  #####
41  # REGISTERING TARGETS
42  #####
43
44  ##### REGISTERING PHP SERVER 1
45  resource "aws_lb_target_group_attachment" "register-php-server-1" {
46    target_group_arn = aws_lb_target_group.alb-tg.arn
47    target_id        = aws_instance.php-server-1.id
48    port             = 80
49  }
```

Activat
Go to Se

Ln 87, Col 54

```

43
44 ##### REGISTERING PHP SERVER 1
45 resource "aws_lb_target_group_attachment" "register-php-server-1" {
46   target_group_arn = aws_lb_target_group.alb-tg.arn
47   target_id        = aws_instance.php-server-1.id
48   port             = 80
49 }
50
51 ##### REGISTERING PHP SERVER 2
52 resource "aws_lb_target_group_attachment" "register-php-server-2" {
53   target_group_arn = aws_lb_target_group.alb-tg.arn
54   target_id        = aws_instance.php-server-2.id
55   port             = 80
56 }
57
58 ##### REGISTERING PHP SERVER 3
59 resource "aws_lb_target_group_attachment" "register-php-server-3" {
60   target_group_arn = aws_lb_target_group.alb-tg.arn
61   target_id        = aws_instance.php-server-3.id
62   port             = 80
63 }
64
65 ##### CREATING ALB
66 resource "aws_lb" "multitier-alb" {
67   name           = "multitier-alb"
68   internal       = false
69   load_balancer_type = "application"
70   security_groups = [aws_security_group.alb-sg.id]
71   subnets       = [
72     aws_subnet.public-web-subnet-1.id,
73     aws_subnet.public-web-subnet-2.id,
74     aws_subnet.public-web-subnet-3.id
75   ]
76   enable_deletion_protection = false
77 }
78
79 ##### CREATING ALB LISTENER FOR HTTP
80 resource "aws_lb_listener" "http" {
81   load_balancer_arn = aws_lb.multitier-alb.arn
82   port              = 80
83   protocol           = "HTTP"
84
85   default_action {
86     type = "forward"
87     target_group_arn = aws_lb_target_group.alb-tg.arn
88   }
89 }
90

```

multitier-alb



Actions ▼

▼ Details

Load balancer type Application	Status ✔ Active	VPC vpc-09ea485f51149f872	Load balancer IP address type IPv4
Scheme Internet-facing	Hosted zone Z35SXDOTRQ7X7K	Availability Zones subnet-0cbdff19afffedff9 us-east-1a (use1-az6) subnet-012afcdfae043ef8d us-east-1c (use1-az2) subnet-09d9419cbf38228dd us-east-1b (use1-az1)	Date created August 10, 2024, 00:31 (UTC+00:00)
Load balancer ARN arn:aws:elasticloadbalancing:us-east-1:010526276130:loadbalancer/app/multitier-alb/9c28606b28c61a68		DNS name Info multitier-alb-819012581.us-east-1.elb.amazonaws.com (A Record)	

- Listeners and rules
- Network mapping
- Resource map - new
- Security
- Monitoring
- Integrations
- Attributes
- Tags

Listeners and rules (1) [Info](#)

Manage rules ▼

Manage listener ▼

Add listener

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Q Filter listeners

< 1 > ⚙

<input type="checkbox"/>	Protocol:Port ▼	Default action ▼	Rules ▼	ARN ▼	Security policy ▼	Default SSL/TLS certificate ▼	mTLS ▼
<input type="checkbox"/>	HTTP:80	Forward to target group <ul style="list-style-type: none">alb-tg: 1 (100%)Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable

Activate Windows
Go to Settings to activate Windows.

12. CREATING DB INSTANCE: :14_db-server.tf

The database instance is a critical component for storing all data required by the e-commerce platform. An Amazon RDS (Relational Database Service) instance was used, configured with MySQL as the database engine, to ensure the database is secure, highly available, and capable of handling the application's demands.

The database instance was provisioned with 20 GB of allocated storage and configured to use the MySQL 8.0 engine. It was set up with a unique identifier (db-server-1) and configured with secure credentials. The instance was made non-publicly accessible, ensuring that it can only be accessed within the VPC, thereby enhancing its security.

To ensure high availability, the database was deployed in a Multi-AZ (Availability Zone) configuration, which allows the database to automatically failover to a standby instance in another availability zone if an outage occurs. This setup provides resilience and minimizes downtime.

A database subnet group was created, encompassing subnets from three different availability zones. This ensures that the database instance is isolated within the private network, further enhancing its security and availability by distributing it across multiple zones.

The security group for the database server was configured to allow inbound and outbound traffic on port 3306, which is used for MySQL communication. This traffic is restricted to the security group associated with the PHP servers, ensuring that only the application servers can communicate with the database. This configuration secures the database while maintaining efficient communication with the application servers.

multitier.tf > 14_db-ins.tf > resource "aws_db_instance" "db-server-1" > publicly_accessible

```
1  ##### CREATING DB INSTANCE
2  resource "aws_db_instance" "db-server-1" {
3      allocated_storage      = 20
4      db_name                 = "mydb"
5      identifier             = "db-server-1"
6      engine                 = var.db-engine
7      engine_version         = var.engine-version
8      instance_class         = var.db-instance-class
9      username               = var.db-username
10     password               = var.db-username-pswd
11     parameter_group_name    = "default.mysql8.0"
12     skip_final_snapshot     = true
13     db_subnet_group_name    = aws_db_subnet_group.my-db-subnet-group.id
14     vpc_security_group_ids = [ aws_security_group.db-server-sg.id ]
15     multi_az               = true
16     publicly_accessible     = false
17
18     tags = {
19         Name = "db-server-1"
20     }
21 }
22
23
24
25 ##### CREATING DB SUBNET GROUP
26 resource "aws_db_subnet_group" "my-db-subnet-group" {
27     name = "my-db-subnet-group"
28     subnet_ids = [
29         aws_subnet.private-db-subnet-1.id,
30         aws_subnet.private-db-subnet-2.id,
31         aws_subnet.private-db-subnet-3.id
32     ]
33
34     tags = {
35         Name = "multitier-DB-Subnet-Group"
36     }
37 }
38
39 ##### CREATING DB SERVER SECURITY GROUP
40 resource "aws_security_group" "db-server-sg" {
41     name        = "db-server-sg"
42     description = "db-server-sg"
43     vpc_id      = aws_vpc.multitier-vpc.id
44
45     ingress {
46         from_port = 3306
47         to_port   = 3306
48         protocol  = "tcp"
49         security_groups = [ aws_security_group.php-server-sg.id ]
50     }
51 }
```

db-server-1

Modify

Actions ▾

Summary

DB identifier db-server-1	Status ✔ Available	Role Instance	Engine MySQL Community	Recommendations
CPU -	Class db.t3.micro	Current activity	Region & AZ us-east-1a	

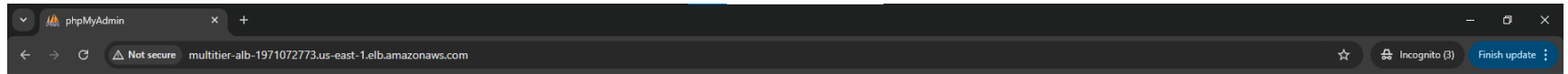
- Connectivity & security
- Monitoring
- Logs & events
- Configuration
- Zero-ETL integrations
- Maintenance & backups
- Tags
- Recommendations

Connectivity & security

Endpoint & port	Networking	Security
Endpoint db-server-1.cvyyqccwirjm.us-east-1.rds.amazonaws.com	Availability Zone us-east-1a	VPC security groups db-server-sg (sg-02f8341ec18e3ef5c) ✔ Active
Port 3306	VPC multitier-vpc (vpc-09ea485f51149f872)	Publicly accessible No
	Subnet group my-db-subnet-group	Certificate authority Info rds-ca-rsa2048-g1
	Subnets subnet-046f0cbf12312a77b subnet-0ba1229ff7edd5db0 subnet-03d689945786a198b	Certificate authority date May 25, 2061, 23:34 (UTC+00:00)
		DB instance certificate expiration date

Activate Windows
Go to Settings to activate Windows.

13. To confirm the application is running the DNS name of the ALB was entered in the web server and logged in with the credentials of the database.




Welcome to phpMyAdmin

Language

English

Log in

Username: pat

Password: *****

Log in

Activate Windows
Go to Settings to activate Windows.

← → ↻ ⚠ Not secure multitier-alb-1971072773.us-east-1.elb.amazonaws.com/index.php?route=/%2F

☆ 🔗 📄 ⓘ ⌵ Finish update

phpMyAdmin

🏠 🗄 📄 📄 📄 📄

Recent Favorites

➕ New

- information_schema
- mydb
- mysql
- performance_schema
- sys

Server: db-server-1.cvyqqcwirjm.us-east-1.rds.amazonaws.com:3306

Databases SQL Status User accounts Export Import Settings Replication Variables Charsets Engines Plugins

General settings

🔑 Change password

📄 Server connection collation: utf8mb4_unicode_ci

🔧 More settings

Appearance settings

🗂 Language English

🎨 Theme pmahomme View all

Database server

- Server: db-server-1.cvyqqcwirjm.us-east-1.rds.amazonaws.com:3306 via TCP/IP
- Server type: MySQL
- Server connection: **SSL is not being used**
- Server version: 8.0.34 - Source distribution
- Protocol version: 10
- User: pat@172.20.4.14
- Server charset: UTF-8 Unicode (utf8mb4)

Web server

- Apache/2.4.61 ()
- Database client version: libmysql - mysqlnd 8.2.19
- PHP extension: mysqli curl mbstring
- PHP version: 8.2.19

phpMyAdmin

- Version information: 5.2.1 (up to date)
- Documentation
- Official Homepage
- Contribute
- Get support
- List of changes
- License

⚠ The configuration file needs a valid key for cookie encryption. A temporary key was automatically generated for you. Please refer to the [documentation](#).

⚠ The phpMyAdmin configuration storage is not completely configured, some extended features have been deactivated. [Find out why.](#)

Or alternately go to 'Operations' tab of any database to set it up there.

Console

Activate Windows
Go to Settings to activate Windows

CHALLENGES ENCOUNTERED

1. Automating Access to the PHP Server through the Jump Server

While developing the Terraform script, I faced a challenge with automating the login process to the PHP server via the jump server. The Terraform documentation didn't readily provide a solution, so I had to research on my own. I eventually found that the connection block could be enhanced with specific attributes, such as *"bastion_host"*, *"bastion_user"*, and *"bastion_private_key"*. By configuring these attributes, I was able to securely connect to the PHP server through the jump server using SSH, which was crucial for maintaining the security of the infrastructure.

2. Automating the Insertion of the Database Endpoint into the PHP Configuration

For the application to function correctly, it was necessary to insert the database endpoint into the PHP application's configuration file located at *"var/www/html/config.inc.php"*. Automating this process within the Terraform script was challenging. I addressed this by capturing the database endpoint as a variable within Terraform and then used basic sed commands in an inline script to automatically replace the placeholder in the configuration file with the actual endpoint. This ensured that the application was correctly configured during deployment.

3. Correcting the Apache Directory Path after LAMP Stack Installation

After executing the LAMP stack installation script, I discovered that the PHP application files were placed inside a phpMyAdmin folder within the *"var/www/html/"* directory. However, Apache was configured to serve files directly from the *"var/www/html/"* directory, causing issues when testing the application via the load balancer. To address this, I added an inline script to the *"remote exec"* provisioner in Terraform, which automatically moved the files from the phpMyAdmin folder back to the root of the *"var/www/html/"* directory. This adjustment allowed the application to become accessible through the web interface.