



T Tej Mandaliya

Building a Scalable Three-Tier Architecture on AWS with Terraform

```
terraform-three-tier-architecture/  
├─ modules/  
│   ├── vpc/  
│   │   ├── main.tf  
│   │   ├── variables.tf  
│   │   └─ outputs.tf  
│   ├── ec2/  
│   │   ├── main.tf  
│   │   ├── variables.tf  
│   │   └─ outputs.tf  
│   ├── s3/  
│   │   ├── main.tf  
│   │   ├── variables.tf  
│   │   └─ outputs.tf  
│   ├── securitygroup/  
│   │   ├── main.tf  
│   │   ├── variables.tf  
│   │   └─ outputs.tf  
│   └─ rds/  
│       ├── main.tf  
│       ├── variables.tf  
│       └─ outputs.tf  
├─ env/  
│   ├── dev/  
│   │   ├── main.tf  
│   │   └─ dev.tfvars  
│   └─ prod/  
│       ├── main.tf  
│       └─ prod.tfvars  
├─ providers.tf  
├─ main.tf  
└─ README.md
```



Introduction:

This document provides a comprehensive overview of the infrastructure setup and configuration deployed using **Terraform**. The infrastructure includes resources such as EC2 instances, RDS databases, and S3 buckets, all provisioned in a cloud environment (AWS in this case). The goal of this project is to automate and manage cloud resources efficiently and reproducibly using **Infrastructure as Code (IaC)** principles.

In this project, Terraform is used to manage the lifecycle of AWS resources, allowing for easy provisioning, management, and decommissioning of cloud resources based on the configuration.

Objective:

The main objective of this project is to:

- **Automate infrastructure provisioning:** Deploy and manage cloud resources using Terraform.
- **Ensure environment consistency:** Use workspace-specific configurations to maintain different environments (like production, staging, development).
- **Simplify resource management:** Allow dynamic configuration of resources like EC2 instances, RDS databases, and S3 buckets based on defined variables.
- **Implement best practices:** Adopt industry-standard practices for AWS infrastructure deployment.

Technologies used:

- **AWS (Amazon Web Services):** Cloud computing platform used to deploy and manage infrastructure resources.
- **Terraform:** An open-source Infrastructure as Code (IaC) tool that allows you to define and provision data center infrastructure using a declarative configuration language.



Tej Mandaliya

How I Set Up

First Step: Setup and Configuration

I have Installed terraform

Terraform is installed on your local machine. You can download

<https://developer.hashicorp.com/terraform/install>

AWS CLI is installed and configured with appropriate access credentials.

Follow the steps here to configure AWS CLI: [AWS CLI Configuration](#).

[AWS CLI Installation](#)

[AWS CLI Configuration](#)

Check :

```
C:\Users\tejma\Downloads>terraform --version
Terraform v1.10.4
on windows_386

Your version of Terraform is out of date! The latest version
is 1.10.5. You can update by downloading from https://www.terraform.io/downloads.html

C:\Users\tejma\Downloads>aws --version
aws-cli/2.22.20 Python/3.12.6 Windows/11 exe/AMD64

C:\Users\tejma\Downloads>aws configure
AWS Access Key ID [*****HVG6]:
AWS Secret Access Key [*****3RuW]:
Default region name [ap-south-1]:
Default output format [json]:

C:\Users\tejma\Downloads>aws s3 ls
2025-01-31 13:43:30 my-dev-terraform-bucket-dev
2025-01-31 13:13:18 my-dev-terraform-bucket-prod
```

Second Step : Creating Modules

2.1 This project follows a modular approach, making it scalable and reusable for multiple environments.

1VPC Module (modules/vpc/)

- main.tf → Defines the AWS VPC, subnets, internet gateway, and route tables.
- variables.tf → Declares input variables like CIDR blocks, subnet ranges, and availability zones.
- outputs.tf → Outputs important attributes like VPC ID and Subnet IDs for reference in other modules.

- **GitHub:** [VPC Code](#)
- ✨ *Purpose:* Provides network isolation and communication between different tiers.

2 EC2 Module (modules/ec2/)

- main.tf → Defines EC2 instances with appropriate AMI, instance type, security groups, and key pairs.
- variables.tf → Defines parameters like instance size, AMI ID, SSH key, and security group IDs.
- outputs.tf → Returns instance public IP, private IP, and instance ID.
- GitHub: [EC2 code](#)

✨ *Purpose:* Hosts the application tier by provisioning EC2 instances.

3 S3 Module (modules/s3/)

- main.tf → Creates S3 buckets.
- variables.tf → Defines bucket names, and storage settings.
- outputs.tf → Outputs the bucket name.
- GitHub: [S3 code](#)

✨ *Purpose:* Provides object storage for logs, backups, or static content.

4 Security Group Module (modules/security group/)

- main.tf → Defines security groups with rules for inbound/outbound traffic.
- variables.tf → Specifies allowed IP ranges, protocols, and ports.
- outputs.tf → Outputs security group IDs.
- GitHub: [Security Code](#)

✨ *Purpose:* Controls access between different architecture layers securely.

5 RDS Module (modules/rds/)

- main.tf → Defines RDS instances (MySQL/PostgreSQL) with multi-AZ deployment.
- variables.tf → Defines DB engine type, version, storage, and credentials.
- outputs.tf → Outputs RDS endpoint and DB name.
- GitHub: [RDS code](#)

✨ *Purpose:* Provides a managed relational database for the backend layer.

Environments (env/)

The env/ directory contains configurations for different environments:

- Development (env/dev/)
 - Uses smaller instance types and non-production settings.
 - dev.tfvars contains environment-specific variable values.
- Production (env/prod/)
 - Uses larger instances and enables high availability.
 - prod.tfvars contains environment-specific variable values.
- GitHub: [Env-terra-code](#)

✧ *Purpose:* Enables separation of environments with different infrastructure configurations.

providers.tf

- Defines the AWS provider and specifies the Terraform backend (S3, DynamoDB, etc.).

✧ *Purpose:* Ensures Terraform interacts with AWS correctly.

Deployment Steps

Step 1: Initialize Terraform

- **terraform init**

This downloads the required provider plugins and initializes the backend.

Step 2: Select or Create a Workspace

- **terraform workspace new dev # Create workspace for Dev**
- **terraform workspace select dev # Switch to Dev workspace**

For production:

- **terraform workspace new prod # Create workspace for Prod**
- **terraform workspace select prod # Switch to Prod workspace**

Step 3: Plan Infrastructure

- **terraform plan -var-file=dev.tfvars**

This shows the execution plan for the development environment.

- **terraform plan -var-file=prod.tfvars**

This shows the execution plan for the production environment

Step 3: Apply Changes

- **terraform apply -var-file=dev.tfvars -auto-approve**

Apply for Development it will start creating all modules.

- `terraform apply -var-file=prod.tfvars`

Apply for Production it will start creating all modules.

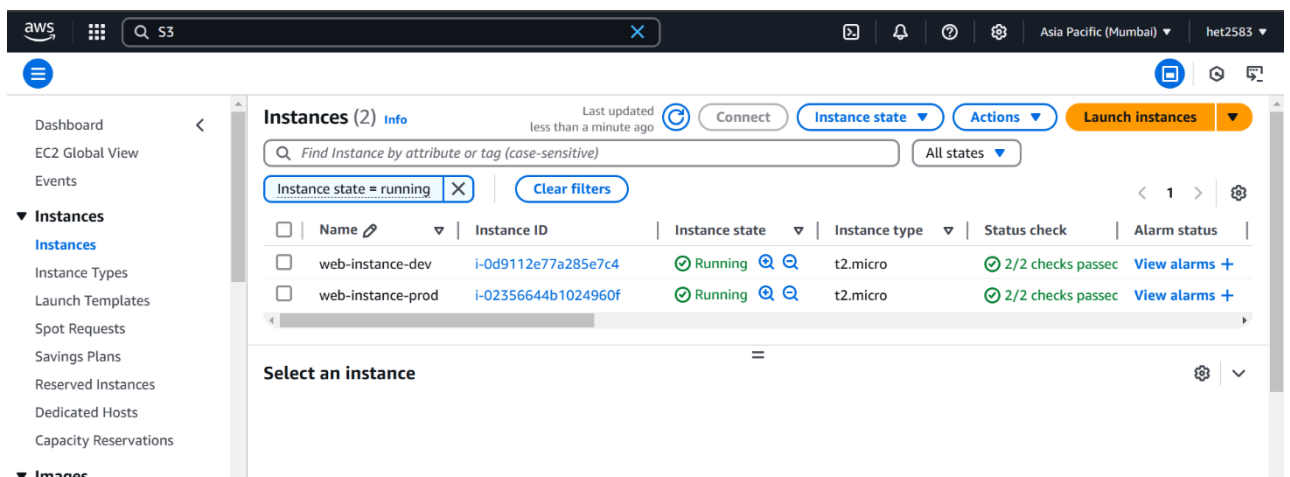
Step 4: Destroy Infrastructure (if needed)

- `terraform destroy -var-file="dev.tfvars"`
- `terraform destroy -var-file="prod.tfvars"`

Destroy command will delete all resource created using terraform command.

✓ Outputs:

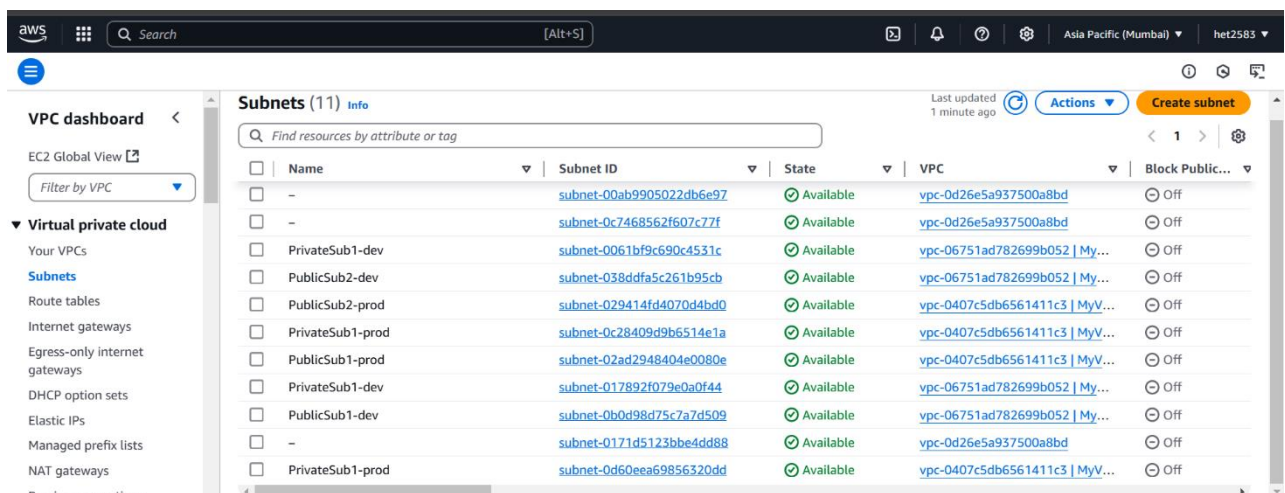
- EC2



The screenshot shows the AWS Management Console for the 'Asia Pacific (Mumbai)' region. The 'Instances' page is active, showing a list of two EC2 instances. The left sidebar contains the navigation menu, and the top bar shows the AWS logo and search bar. The instances listed are 'web-instance-dev' and 'web-instance-prod', both in a 'Running' state.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
web-instance-dev	i-0d9112e77a285e7c4	Running	t2.micro	2/2 checks passed	View alarms
web-instance-prod	i-02356644b1024960f	Running	t2.micro	2/2 checks passed	View alarms

- VPC



The screenshot shows the AWS Management Console for the 'Asia Pacific (Mumbai)' region. The 'VPC dashboard' is active, showing a list of 11 subnets. The left sidebar contains the navigation menu, and the top bar shows the AWS logo and search bar. The subnets listed include 'PublicSub1-dev', 'PublicSub2-dev', 'PublicSub2-prod', 'PrivateSub1-prod', and 'PrivateSub1-dev', all in an 'Available' state.

Name	Subnet ID	State	VPC	Block Public...
-	subnet-00ab9905022db6e97	Available	vpc-0d26e5a937500a8bd	Off
-	subnet-0c7468562f607c77f	Available	vpc-0d26e5a937500a8bd	Off
PrivateSub1-dev	subnet-0061bf9c690c4531c	Available	vpc-06751ad782699b052 My...	Off
PublicSub2-dev	subnet-038ddfa5c261b95cb	Available	vpc-06751ad782699b052 My...	Off
PublicSub2-prod	subnet-029414fd4070d4bd0	Available	vpc-0407c5db6561411c3 MyV...	Off
PrivateSub1-prod	subnet-0c28409d9b6514e1a	Available	vpc-0407c5db6561411c3 MyV...	Off
PublicSub1-prod	subnet-02ad2948404e0080e	Available	vpc-0407c5db6561411c3 MyV...	Off
PrivateSub1-dev	subnet-017892f079e0a0f44	Available	vpc-06751ad782699b052 My...	Off
PublicSub1-dev	subnet-0b0d98d75c7a7d509	Available	vpc-06751ad782699b052 My...	Off
-	subnet-0171d5123bbe4dd88	Available	vpc-0d26e5a937500a8bd	Off
PrivateSub1-prod	subnet-0d60eea69856320dd	Available	vpc-0407c5db6561411c3 MyV...	Off

aws

Search

[Alt+S]

Asia Pacific (Mumbai)

het2583

VPC dashboard

EC2 Global View

Filter by VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

DHCP option sets

Elastic IPs

Your VPCs (3)

Info

Last updated less than a minute ago

Actions

Create VPC

Search

1

<input type="checkbox"/>	Name	VPC ID	State	Block Public...	IPv4 CIDR
<input type="checkbox"/>	MyVPC-dev	vpc-06751ad782699b052	Available	Off	10.0.0.0/16
<input type="checkbox"/>	-	vpc-0d26e5a937500a8bd	Available	Off	172.31.0.0/16
<input type="checkbox"/>	MyVPC-prod	vpc-0407c5db6561411c3	Available	Off	10.0.0.0/16

VPC dashboard

EC2 Global View

Filter by VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

Internet gateways (3)

Info

Last updated less than a minute ago

Actions

Create internet gateway

Search

1

<input type="checkbox"/>	Name	Internet gateway ID	State	VPC ID	Owner
<input type="checkbox"/>	MyIGW-prod	igw-0367a47e411a4f32f	Attached	vpc-0407c5db6561411c3 MyVPC-prod	7678:
<input type="checkbox"/>	-	igw-04fa5044e0830d7fa	Attached	vpc-0d26e5a937500a8bd	7678:
<input type="checkbox"/>	MyIGW-dev	igw-05e4bec8845cb1dc8	Attached	vpc-06751ad782699b052 MyVPC-dev	7678:

Route tables (5)

Info

Last updated 4 minutes ago

Actions

Create route table

Find resources by attribute or tag

1

<input type="checkbox"/>	Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
<input type="checkbox"/>	-	rtb-0b9f72e20afd34fce	-	-	Yes	vpc-06751
<input type="checkbox"/>	-	rtb-01ac52852f7a7356d	-	-	Yes	vpc-0407c
<input type="checkbox"/>	Route-prod	rtb-0ef2c571ae79b9c99	2 subnets	-	No	vpc-0407c
<input type="checkbox"/>	-	rtb-0314992797e39b3e4	-	-	Yes	vpc-0d26e
<input type="checkbox"/>	Route-dev	rtb-0fb69ccce96c3675a	2 subnets	-	No	vpc-06751

• S3

Amazon S3

Buckets

Account snapshot - updated every 24 hours

All AWS Regions

View Storage Lens dashboard

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

General purpose buckets

Directory buckets

General purpose buckets (2)

Info

All AWS Regions

Copy ARN

Empty

Delete

Create bucket

Buckets are containers for data stored in S3.

Find buckets by name

1

<input type="radio"/>	Name	AWS Region	IAM Access Analyzer	Creation date
<input type="radio"/>	my-dev-terraform-bucket-dev	Asia Pacific (Mumbai) ap-south-1	View analyzer for ap-south-1	January 31, 2025, 13:43:29 (UTC+05:30)
<input type="radio"/>	my-dev-terraform-bucket-prod	Asia Pacific (Mumbai) ap-south-1	View analyzer for ap-south-1	January 31, 2025, 13:13:17 (UTC+05:30)

• RDS

Amazon RDS > Databases

Amazon RDS

- Dashboard
- Databases**
- Query Editor
- Performance insights
- Snapshots
- Exports in Amazon S3
- Automated backups
- Reserved instances
- Proxies

Consider creating a Blue/Green Deployment to minimize downtime during upgrades

You may want to consider using Amazon RDS Blue/Green Deployments and minimize your downtime during upgrades. A Blue/Green Deployment provides a staging environment for changes to production databases. [RDS User Guide](#) [Aurora User Guide](#)

Notifications 0 0 0 0 3 0

Databases (2) Group resources Modify Actions Restore from S3 Create database

Filter by databases

	DB identifier	Status	Role	Engine	Region ...	Size	Recomr
	mydatabase-dev	Available	Instance	MySQL Co...	ap-south-1a	db.t3.micro	
	mydatabase-prod	Available	Instance	MySQL Co...	ap-south-1a	db.t3.micro	

🔑 Key Features

- ✓ **Modular Approach** → Reusable modules for efficient infrastructure management.
- ✓ **Multi-Environment Support** → Separate configs for dev and prod.
- ✓ **Scalable Architecture** → Easily extendable for additional services.
- ✓ **Security Best Practices** → IAM roles, security groups, and encrypted storage.
- ✓ **Automation Ready** → Can be integrated with CI/CD pipelines.