

**PROG 8870 -- Final Project: Deploying AWS Infrastructure with Terraform and CloudFormation**

**Title:** AWS Infrastructure Automation with Terraform and CloudFormation

**Name:** Suman Kumari Jakhar

**Prof:** Vikas Vattikonda

Overview

In this project, students will utilize Terraform and CloudFormation to create a scalable AWS infrastructure. The project will demonstrate a multi-service environment including S3 Buckets, EC2 Instances, and RDS Database Instances, applying Infrastructure as Code (IaC) best practices across different tools.

Project Scope

You are required to complete deployment tasks across six modules using Terraform and CloudFormation, ensuring:

- Private and secure AWS storage and compute resources.
- Automated and modular infrastructure.
- Proper documentation, reusability, and dynamic configuration.
- A live demo showcasing resource provisioning.

Tasks and Deliverables:

1. S3 Bucket Setup

- Using Terraform:

- o Create 4 Private S3 Buckets without public access.
- o Enable versioning on all buckets (Bonus Challenge).

```
PS C:\Assignments\Finals\Final_Project_Deploying_AWS_Infrastructure\terraform\s3> terraform plan
aws_s3_bucket.buckets["student-proj-bucket-b"]: Refreshing state... [id=student-proj-bucket-b]
aws_s3_bucket.buckets["student-proj-bucket-d"]: Refreshing state... [id=student-proj-bucket-d]
aws_s3_bucket.buckets["student-proj-bucket-c"]: Refreshing state... [id=student-proj-bucket-c]
aws_s3_bucket.buckets["student-proj-bucket-a"]: Refreshing state... [id=student-proj-bucket-a]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_s3_bucket.buckets["student-proj-bucket-a"] will be created
+ resource "aws_s3_bucket" "buckets" {
  + acceleration_status      = (known after apply)
  + acl                      = (known after apply)
  + arn                     = (known after apply)
  + bucket                  = "student-proj-bucket-a"
  + bucket_domain_name      = (known after apply)
  + bucket_prefix           = (known after apply)
  + bucket_region           = (known after apply)
  + bucket_regional_domain_name = (known after apply)
  + force_destroy           = false
  + hosted_zone_id          = (known after apply)
  + id                      = (known after apply)
  + object_lock_enabled      = false
  + policy                  = (known after apply)
  + region                  = "us-east-1"
  + request_payer            = (known after apply)
  + tags_all                = (known after apply)
  + website_domain          = (known after apply)
  + website_endpoint        = (known after apply)

  + cors_rule (known after apply)

  + grant (known after apply)

  + lifecycle_rule (known after apply)

  + logging (known after apply)

  + object_lock_configuration (known after apply)

  + replication_configuration (known after apply)
```

```
+ mfa_delete = (known after apply)
+ status      = "Enabled"
}
}

# aws_s3_bucket_versioning.versioning["student-proj-bucket-b"] will be created
+ resource "aws_s3_bucket_versioning" "versioning" {
+   bucket = (known after apply)
+   id      = (known after apply)
+   region  = "us-east-1"

+   versioning_configuration {
+     mfa_delete = (known after apply)
+     status      = "Enabled"
+   }
}

# aws_s3_bucket_versioning.versioning["student-proj-bucket-c"] will be created
+ resource "aws_s3_bucket_versioning" "versioning" {
+   bucket = (known after apply)
+   id      = (known after apply)
+   region  = "us-east-1"

+   versioning_configuration {
+     mfa_delete = (known after apply)
+     status      = "Enabled"
+   }
}

# aws_s3_bucket_versioning.versioning["student-proj-bucket-d"] will be created
+ resource "aws_s3_bucket_versioning" "versioning" {
+   bucket = (known after apply)
+   id      = (known after apply)
+   region  = "us-east-1"

+   versioning_configuration {
+     mfa_delete = (known after apply)
+     status      = "Enabled"
+   }
}

}

Plan: 16 to add, 0 to change, 0 to destroy.
```

aws

Search

[Alt+S]

United States (N. Virginia)

Account ID: 7950-9629-6861

voclabs/user3807391-Suman\_Jakhar

Amazon S3

Buckets

General purpose buckets

Directory buckets

General purpose buckets (4)

Info

Copy ARN

Empty

Delete

Create bucket

Buckets are containers for data stored in S3.

Find buckets by name

1

Name	AWS Region	Creation date
student-proj-bucket-a	US East (N. Virginia) us-east-1	August 12, 2025, 11:50:59 (UTC-04:00)
student-proj-bucket-b	US East (N. Virginia) us-east-1	August 12, 2025, 11:50:59 (UTC-04:00)
student-proj-bucket-c	US East (N. Virginia) us-east-1	August 12, 2025, 11:50:59 (UTC-04:00)
student-proj-bucket-d	US East (N. Virginia) us-east-1	August 12, 2025, 11:50:59 (UTC-04:00)

Account snapshot

Updated daily

View dashboard

Storage Lens provides visibility into storage usage and activity trends.

External access summary - new

Updated daily

Info

External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.

Storage Lens

Dashboards

aws

Search

[Alt+S]

United States (N. Virginia)

Account ID: 7950-9629-6861

voclabs/user3807391-Suman\_Jakhar

Amazon S3

Buckets

student-proj-bucket-a

General purpose buckets

Directory buckets

student-proj-bucket-a

Info

Objects

Metadata

Properties

Permissions

Metrics

Management

Access Points

Bucket overview

AWS Region

US East (N. Virginia) us-east-1

Amazon Resource Name (ARN)

arn:aws:s3:::student-proj-bucket-a

Creation date

August 12, 2025, 11:50:59 (UTC-04:00)

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. Learn more

Bucket Versioning

Enabled

Multi-factor authentication (MFA) delete

An additional layer of security that requires multi-factor authentication for changing Bucket Versioning settings and permanently deleting object versions. To modify MFA delete settings, use the AWS CLI, AWS SDK, or the Amazon S3 REST API. Learn more

Disabled

Tags (0)

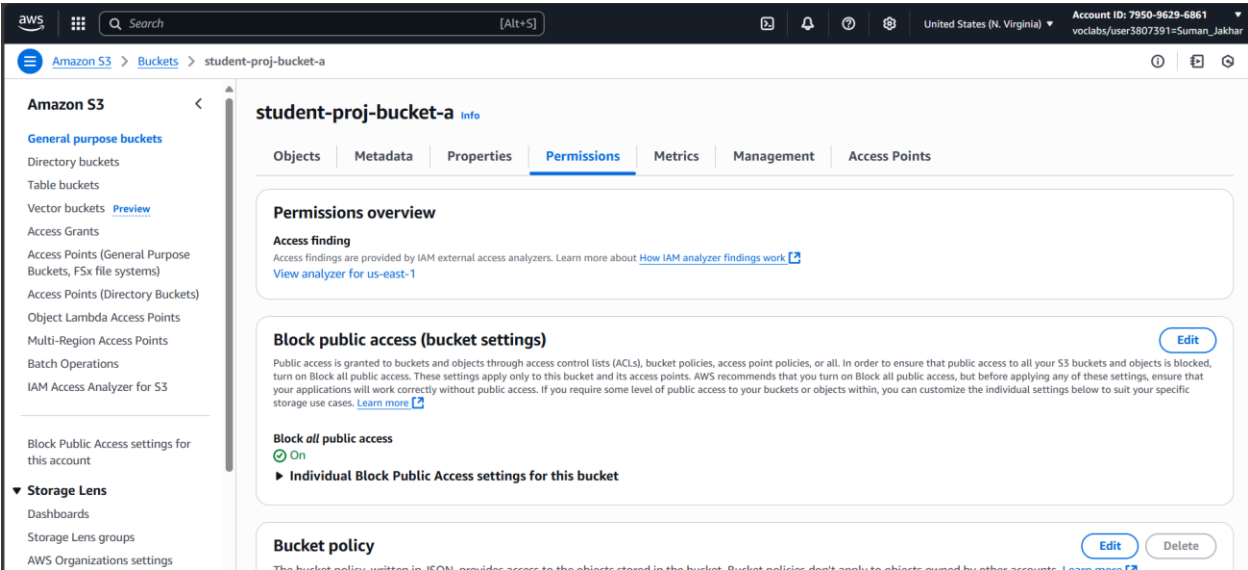
Edit

Storage Lens

Dashboards

Storage Lens groups

AWS Organizations settings



Code Snippet:

Backend.tf

```
main.tf backend.tf X
terraform > s3 > backend.tf
1 terraform {
2   backend "local" {
3     path = "terraform.tfstate"
4   }
5 }
6
```

Variables.tf

```
main.tf variables.tf X
terraform > s3 > variables.tf
1 variable "aws_region" {
2   description = "AWS region to deploy resources"
3   type        = string
4   default     = "us-east-1"
5 }
6
7 variable "bucket_names" {
8   description = "List of S3 bucket names to create"
9   type        = list(string)
10  default     = ["student-proj-bucket-a", "student-proj-bucket-b", "student-proj-bucket-c", "student-proj-bucket-d"]
11 }
12
```

Terraform.tfvars

```
main.tf terraform.tfvars X
terraform > s3 > terraform.tfvars
1 aws_region = "us-east-1"
2 bucket_names = ["student-proj-bucket-a", "student-proj-bucket-b", "student-proj-bucket-c", "student-proj-bucket-d"]
3
```

Main.tf

main.tf ...\ec2main.tf ...\s3 X

terraform > s3 > main.tf

```
1 terraform {
2   required_providers {
3     aws = { source = "hashicorp/aws" }
4   }
5 }
6
7 provider "aws" {
8   region = var.aws_region
9 }
10
11 resource "aws_s3_bucket" "buckets" {
12   for_each      = toset(var.bucket_names)
13   bucket        = each.value
14   object_lock_enabled = false
15
16   lifecycle {
17     ignore_changes = [object_lock_configuration]
18   }
19 }
20
21 resource "aws_s3_bucket_acl" "private" {
22   for_each = aws_s3_bucket.buckets
23   bucket   = each.value.id
24   acl      = "private"
25 }
26
27 resource "aws_s3_bucket_public_access_block" "block" {
28   for_each      = aws_s3_bucket.buckets
29   bucket        = each.value.id
30   block_public_acls       = true
31   block_public_policy     = true
32   ignore_public_acls     = true
33   restrict_public_buckets = true
34 }
35
36 resource "aws_s3_bucket_versioning" "versioning" {
37   for_each = aws_s3_bucket.buckets
38   bucket   = each.value.id
39   versioning_configuration {
40     status = "Enabled"
41   }
42 }
```

- Using CloudFormation:
- o Create 3 Private S3 Buckets with PublicAccessBlockConfiguration.
  - o Enable versioning on all buckets (Bonus Challenge)

aws

Search [Alt+S]

United States (N. Virginia) Account ID: 7950-9629-6861 vodlabs/user3807391=Suman\_Jakhar

CloudFormation > Stacks > S3-Suman

Stacks (3)

Filter by stack name

Filter status Active View nested

S3-Suman

2025-08-12 12:10:08 UTC-0400

CREATE\_COMPLETE

EC2-Suman

2025-08-12 12:06:56 UTC-0400

CREATE\_COMPLETE

c167448a43046851111675241w795096296861

2025-08-12 10:09:40 UTC-0400

CREATE\_COMPLETE

S3-Suman

Delete Update stack Stack actions Create stack

Stack info Events Resources Outputs Parameters Template Change set

Overview

Stack ID

arn:aws:cloudformation:us-east-1:795096296861:stack/S3-Suman/d0c5f780-7796-11f0-ab71-0e47978d0a5d

Description

3 private S3 buckets with versioning

Status

CREATE\_COMPLETE

Detailed status

-

Status reason

User Initiated

Root stack

-

Parent stack

-

Created time

2025-08-12 12:10:08 UTC-0400

Updated time

-

Deleted time

-

Drift status

NOT\_CHECKED

Code Snippet:

```
Terminal Help ← → Final_Project_Deploying
! s3.yaml ×
cloudformation > ! s3.yaml
Visualize with Infrastructure Composer
1 AWSTemplateFormatVersion: '2010-09-09'
2 Description: 3 private S3 buckets with versioning
3
4 Resources:
5   Bucket1:
6     Type: AWS::S3::Bucket
7     Properties:
8       VersioningConfiguration: { Status: Enabled }
9       PublicAccessBlockConfiguration:
10        BlockPublicAcls: true
11        BlockPublicPolicy: true
12        IgnorePublicAcls: true
13        RestrictPublicBuckets: true
14
15   Bucket2:
16     Type: AWS::S3::Bucket
17     Properties:
18       VersioningConfiguration: { Status: Enabled }
19       PublicAccessBlockConfiguration:
20        BlockPublicAcls: true
21        BlockPublicPolicy: true
22        IgnorePublicAcls: true
23        RestrictPublicBuckets: true
24
25   Bucket3:
26     Type: AWS::S3::Bucket
27     Properties:
28       VersioningConfiguration: { Status: Enabled }
29       PublicAccessBlockConfiguration:
30        BlockPublicAcls: true
31        BlockPublicPolicy: true
32        IgnorePublicAcls: true
33        RestrictPublicBuckets: true
34
default × Amazon Q BLACKBOX Chat Add Logs CyberCoder Improve Code
```

## 2. VPC and EC2 Instance

### - Using Terraform:

- o Set up an EC2 instance inside a custom VPC.
- o Use dynamic variables for AMI ID and instance type.
- o Enable public IP and allow SSH access (port 22).

```
# aws_instance.ec2 will be created
+ resource "aws_instance" "ec2" {
+   ami                      = "ami-0de716d6197524dd9"
+   arn                      = (known after apply)
+   associate_public_ip_address = true
+   availability_zone        = (known after apply)
+   disable_api_stop        = (known after apply)
+   disable_api_termination = (known after apply)
+   ebs_optimized            = (known after apply)
+   enable_primary_ipv6      = (known after apply)
+   force_destroy            = false
+   get_password_data        = false
+   host_id                  = (known after apply)
+   host_resource_group_arn  = (known after apply)
+   iam_instance_profile     = (known after apply)
+   id                       = (known after apply)
+   instance_initiated_shutdown_behavior = (known after apply)
+   instance_lifecycle       = (known after apply)
+   instance_state           = (known after apply)
+   instance_type            = "t2.micro"
+   ipv6_address_count       = (known after apply)
+   ipv6_addresses           = (known after apply)
+   key_name                 = (known after apply)
+   monitoring               = (known after apply)
```

```
public_ip = "18.212.202.95"
```

VPC > Subnets

subnet-07f8d93e36c13a4c4

VPC dashboard

<

subnet-07f8d93e36c13a4c4 / proj-subnet

Actions

EC2 Global View

Filter by 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

NAT gateways

Peering connections

Route servers

Details

Subnet ID  
subnet-07f8d93e36c13a4c4

IPv4 CIDR  
10.0.1.0/24

Availability Zone  
use1-az4 (us-east-1a)

Network ACL  
acl-08cb11e4d287146a8

Auto-assign customer-owned IPv4 address  
No

IPv6 CIDR reservations  
-

Resource name DNS AAAA record  
Disabled

Subnet ARN  
arn:aws:ec2:us-east-1:795096296861:subnet/subnet-07f8d93e36c13a4c4

Available IPv4 addresses  
250

Network border group  
us-east-1

Default subnet  
No

Customer-owned IPv4 pool  
-

IPv6-only  
No

DNS64  
Disabled

State  
Available

IPv6 CIDR  
-

VPC  
vpc-0a7859ae9287c8287 | proj-vpc

Auto-assign public IPv4 address  
Yes

Outpost ID  
-

Hostname type  
IP name

Owner  
795096296861

Block Public Access  
Off

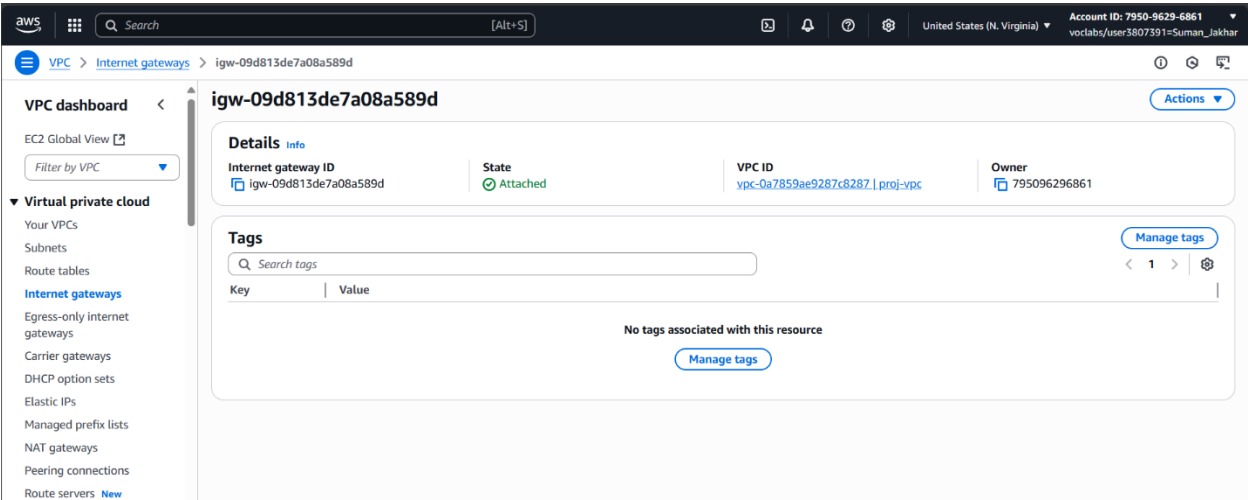
IPv6 CIDR association ID  
-

Route table  
rtb-0f0167e3b4bf8ef40

Auto-assign IPv6 address  
No

IPv4 CIDR reservations  
-

Resource name DNS A record  
Disabled



Code Snippet:

Variables.tf:

```
main.tf | variables.tf X
terraform > ec2 > variables.tf
1  variable "aws_region" {
2    description = "AWS region to deploy resources"
3    type       = string
4    default    = "us-east-1"
5  }
6
7  variable "ami_id" {
8    description = "AMI ID for the EC2 instance"
9    type       = string
10   default    = "ami-0de716d6197524dd9"
11 }
12
13 variable "instance_type" {
14   description = "EC2 instance type"
15   type       = string
16   default    = "t2.micro"
17 }
18
19 variable "vpc_cidr" { default = "10.0.0.0/16" }
20 variable "subnet_cidr" { default = "10.0.1.0/24" }
21
```

Terraform.tfvars

```
main.tf | terraform.tfvars X
terraform > ec2 > terraform.tfvars
1  aws_region      = "us-east-1"
2  ami_id          = "ami-0de716d6197524dd9"
3  instance_type   = "t2.micro"
4
```

Backend.tf



main.tf

backend.tf

terraform > ec2 > backend.tf

1 terraform {

2     backend "local" {

3         path = "terraform.tfstate"

4     }

5 }

6

Main.tf

main.tf

backend.tf

terraform > ec2 > main.tf

1 terraform {

2     required\_providers {

3         aws = { source = "hashicorp/aws" }

4     }

5 }

6

7 provider "aws" {

8     region = var.aws\_region

9 }

10

11 resource "aws\_vpc" "vpc" {

12     cidr\_block = var.vpc\_cidr

13     tags = { Name = "proj-vpc" }

14 }

15

16 resource "aws\_subnet" "subnet" {

17     vpc\_id = aws\_vpc.vpc.id

18     cidr\_block = var.subnet\_cidr

19     map\_public\_ip\_on\_launch = true

20     availability\_zone = "us-east-1a"

21     tags = { Name = "proj-subnet" }

22 }

23

24 resource "aws\_internet\_gateway" "igw" { vpc\_id = aws\_vpc.vpc.id }

25

26 resource "aws\_route\_table" "rt" {

27     vpc\_id = aws\_vpc.vpc.id

28     route {

29         cidr\_block = "0.0.0.0/0"

30         gateway\_id = aws\_internet\_gateway.igw.id

31     }

32 }

33

34 resource "aws\_route\_table\_association" "assoc" {

35     subnet\_id = aws\_subnet.subnet.id

36     route\_table\_id = aws\_route\_table.rt.id

37 }

38

39 resource "aws\_security\_group" "sg" {

40     vpc\_id = aws\_vpc.vpc.id

41

42     ingress {

43         from\_port = 22

44         to\_port = 22

45         protocol = "tcp"

46         cidr\_blocks = ["0.0.0.0/0"]

47     }

48

49     egress {

50         from\_port = 0

51         to\_port = 0

52         protocol = "-1"

53         cidr\_blocks = ["0.0.0.0/0"]

54     }

55 }

56

57 resource "aws\_instance" "ec2" {

58     ami = var.ami\_id

59     instance\_type = var.instance\_type

60     subnet\_id = aws\_subnet.subnet.id

61     vpc\_security\_group\_ids = [aws\_security\_group.sg.id]

62     associate\_public\_ip\_address = true

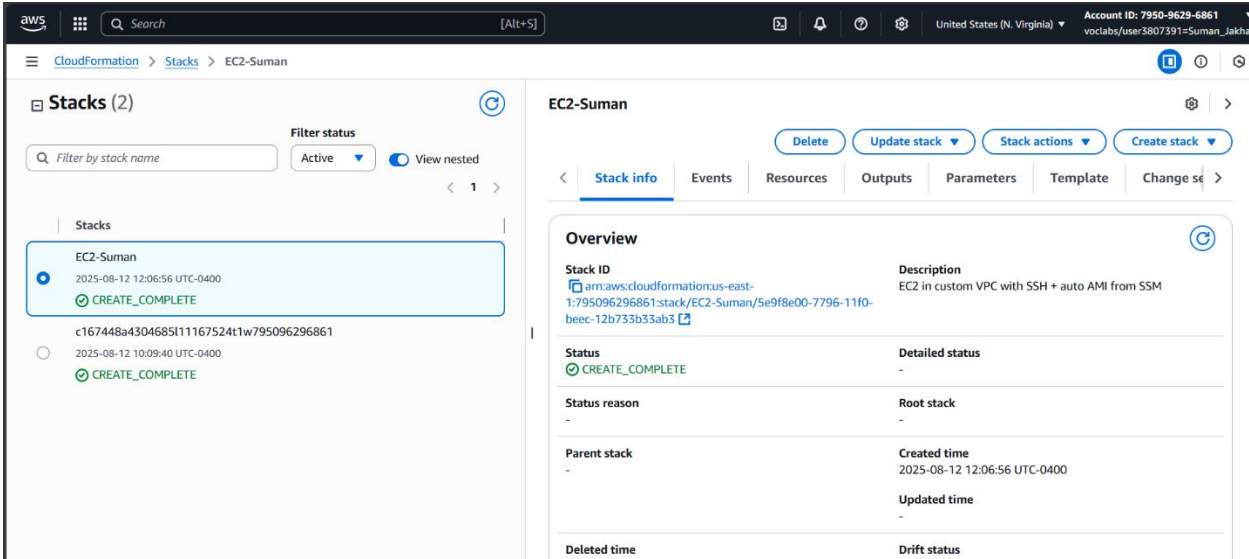
63 }

64

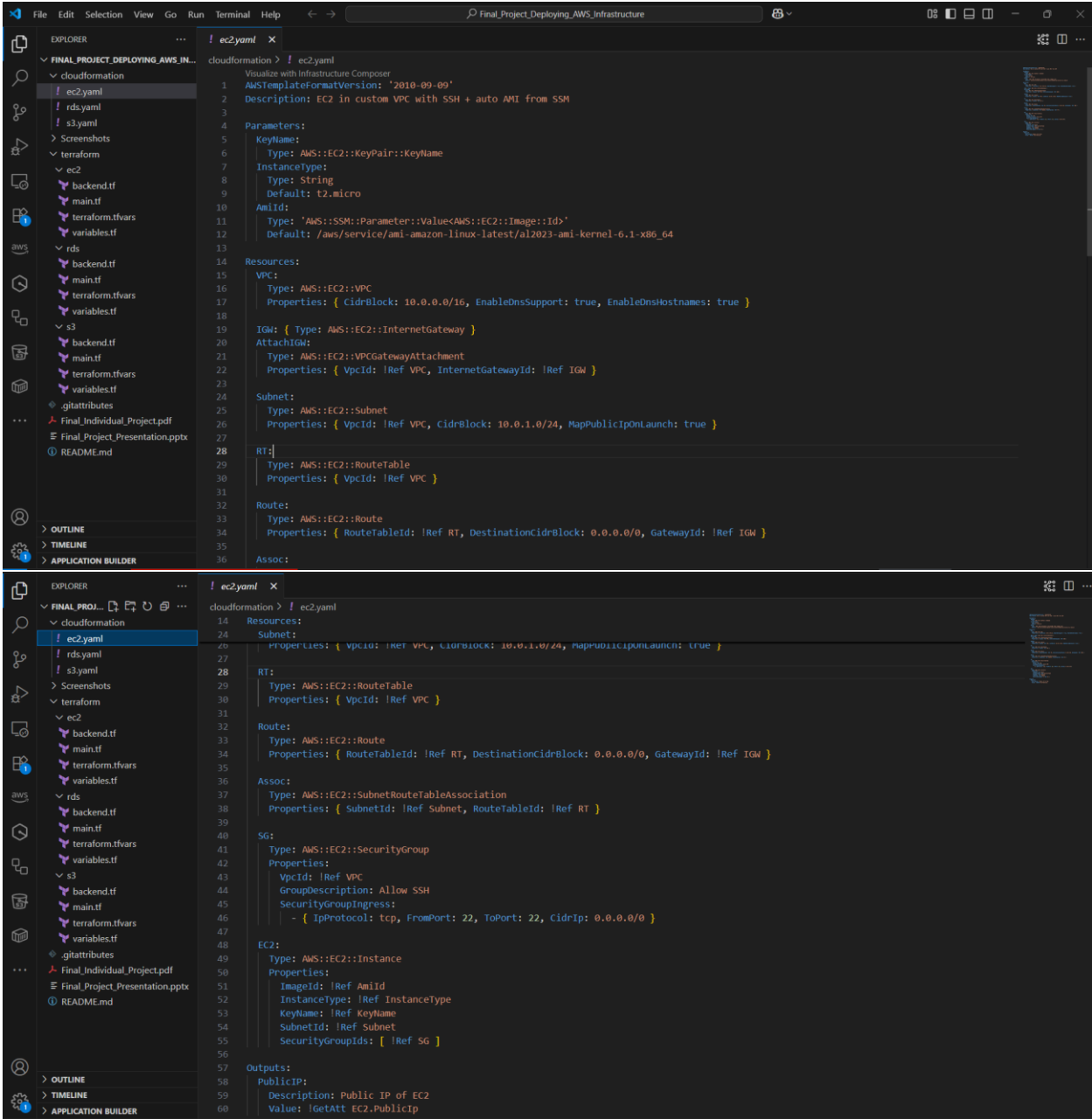
65 output "public\_ip" { value = aws\_instance.ec2.public\_ip }

- Using CloudFormation:

- o Deploy EC2 Instance with YAML-based configuration.
- o Attach necessary networking components (IGW, Route Tables).
- o Output EC2 Public IP as part of the CloudFormation outputs.



Code snippet:



3. RDS Instance Deployment

- Using Terraform:

- o Create a MySQL RDS Database with db.t3.micro instance type.
- o Define database name, username, password via input variables.
- o Deploy into a dedicated DB Subnet Group.

```
PS C:\Assignments\Finals\Final_Project_Deploying_AWS_Infrastructure\terraform\rds> terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_db_instance.mysql will be created
+ resource "aws_db_instance" "mysql" {
  + address                               = (known after apply)
  + allocated_storage                     = 20
  + apply_immediately                     = false
  + arn                                   = (known after apply)
  + auto_minor_version_upgrade            = true
  + availability_zone                     = (known after apply)
  + backup_retention_period                = (known after apply)
  + backup_target                         = (known after apply)
  + backup_window                         = (known after apply)
  + ca_cert_identifier                    = (known after apply)
  + character_set_name                     = (known after apply)
  + copy_tags_to_snapshot                 = false
  + database_insights_mode                = (known after apply)
  + db_name                               = "Suman_db"
  + db_subnet_group_name                  = "rds-subnet-group"
  + dedicated_log_volume                  = false
  + delete_automated_backups              = true
  + domain_fqdn                           = (known after apply)
  + endpoint                             = (known after apply)
  + engine                                = "mysql"
  + engine_lifecycle_support              = (known after apply)
  + engine_version                        = (known after apply)
  + engine_version_actual                 = (known after apply)
  + hosted_zone_id                       = (known after apply)
  + id                                    = (known after apply)
  + identifier                            = (known after apply)
  + identifier_prefix                     = (known after apply)
  + instance_class                        = "db.t3.micro"
  + iops                                  = (known after apply)
  + kms_key_id                            = (known after apply)

aws_vpc.rds_vpc: Creating...
aws_vpc.rds_vpc: Creation complete after 2s [id=vpc-0f1f235f68040e1fd]
aws_subnet.db_subnet_b: Creating...
aws_subnet.db_subnet_a: Creating...
aws_subnet.db_subnet_b: Creation complete after 1s [id=subnet-0d62bd4348dbb2abb]
aws_subnet.db_subnet_a: Creation complete after 1s [id=subnet-049dc3c77ae8be4b5]
aws_db_subnet_group.db_group: Creating...
aws_db_subnet_group.db_group: Creation complete after 1s [id=rds-subnet-group]
aws_db_instance.mysql: Creating...
aws_db_instance.mysql: Still creating... [10s elapsed]
aws_db_instance.mysql: Still creating... [20s elapsed]
aws_db_instance.mysql: Still creating... [30s elapsed]
aws_db_instance.mysql: Still creating... [40s elapsed]
aws_db_instance.mysql: Still creating... [50s elapsed]
aws_db_instance.mysql: Still creating... [1m0s elapsed]
aws_db_instance.mysql: Still creating... [1m10s elapsed]
aws_db_instance.mysql: Still creating... [1m20s elapsed]
aws_db_instance.mysql: Still creating... [1m30s elapsed]
aws_db_instance.mysql: Still creating... [1m40s elapsed]
aws_db_instance.mysql: Still creating... [1m50s elapsed]
aws_db_instance.mysql: Still creating... [2m0s elapsed]
aws_db_instance.mysql: Still creating... [2m10s elapsed]
aws_db_instance.mysql: Still creating... [2m20s elapsed]
aws_db_instance.mysql: Still creating... [2m30s elapsed]
aws_db_instance.mysql: Still creating... [2m40s elapsed]
aws_db_instance.mysql: Still creating... [2m50s elapsed]
aws_db_instance.mysql: Still creating... [3m0s elapsed]
aws_db_instance.mysql: Still creating... [3m10s elapsed]
aws_db_instance.mysql: Still creating... [3m20s elapsed]
aws_db_instance.mysql: Still creating... [3m30s elapsed]
aws_db_instance.mysql: Still creating... [3m40s elapsed]
aws_db_instance.mysql: Still creating... [3m50s elapsed]
aws_db_instance.mysql: Still creating... [4m0s elapsed]
aws_db_instance.mysql: Still creating... [4m10s elapsed]
aws_db_instance.mysql: Creation complete after 4m15s [id=db-WA2KIK6WHYTED4AIL7P7S7HTSQ]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.
```

Aurora and RDS

Databases

Query editor

Performance insights

Snapshots

Exports in Amazon S3

Automated backups

Reserved instances

Proxies

Subnet groups

Parameter groups

Option groups

Custom engine versions

Zero-ETL integrations

Events

Event subscriptions

terraform-20250812150616053900000001

Summary

DB identifier  
terraform-20250812150616053900000001

CPU  
3.96%

Status  
Available

Class  
db.t3.micro

Role  
Instance

Current activity  
0 Connections

Engine  
MySQL Community

Region & AZ  
us-east-1a

Recommendations

Connectivity & security

Monitoring

Logs & events

Configuration

Zero-ETL integrations

Maintenance & backups

Data

Connectivity & security

Endpoint & port

Endpoint  
terraform-2025081215061605390000001.c08wqmsvbnbx.us-east-1.rds.amazonaws.com

Port  
3306

Networking

Availability Zone  
us-east-1a

VPC  
vpc-0f1f235f68040e1fd

Subnet group

Security

VPC security groups  
default (sg-0533160d2aa39a6dd)

Active

Publicly accessible  
No



```
main.tf variables.tf X
terraform > rds > variables.tf
1  variable "aws_region" {
2      description = "AWS region to deploy resources"
3      type        = string
4      default     = "us-east-1"
5  }
6
7  variable "db_name" {
8      description = "Database name"
9      type        = string
10     default     = "suman_db"
11 }
12
13 variable "db_user" {
14     description = "Database username"
15     type        = string
16     default     = "admin"
17 }
18
19 variable "db_password" {
20     description = "Database password"
21     type        = string
22     default     = "Suman123!"
23     sensitive   = true
24 }
```

Terraform.tfvars

```
main.tf terraform.tfvars X
terraform > rds > terraform.tfvars
1  aws_region = "us-east-1"
2  db_name    = "Suman_db"
3  db_user    = "admin"
4  db_password = "Suman123!"
5  |
```

Main.tf

main.tf ...\ec2main.tf ...\rds

```
terraform > rds > main.tf
1 terraform {
2   required_providers {
3     aws = { source = "hashicorp/aws" }
4   }
5 }
6
7 provider "aws" {
8   region = var.aws_region
9 }
10
11
12 resource "aws_vpc" "rds_vpc" {
13   cidr_block = "10.1.0.0/16"
14 }
15
16 resource "aws_subnet" "db_subnet_a" {
17   vpc_id      = aws_vpc.rds_vpc.id
18   cidr_block   = "10.1.1.0/24"
19   availability_zone = "us-east-1a"
20 }
21
22 resource "aws_subnet" "db_subnet_b" {
23   vpc_id      = aws_vpc.rds_vpc.id
24   cidr_block   = "10.1.2.0/24"
25   availability_zone = "us-east-1b"
26 }
27
28 resource "aws_db_subnet_group" "db_group" {
29   name      = "rds-subnet-group"
30   subnet_ids = [aws_subnet.db_subnet_a.id, aws_subnet.db_subnet_b.id]
31 }
32
33 resource "aws_db_instance" "mysql" {
34   allocated_storage = 20
35   engine            = "mysql"
36   instance_class    = "db.t3.micro"
37   db_name           = var.db_name
38   db_name           = var.db_name
39   username          = var.db_user
40   password          = var.db_password
41   skip_final_snapshot = true
42   db_subnet_group_name = aws_db_subnet_group.db_group.name
43   publicly_accessible = false
44 }
```

- Using CloudFormation:
- o Deploy RDS instance using YAML templates.
  - o Ensure public access is enabled (for this project only).
  - o Configure security groups to allow MySQL traLic on port 3306.

aws

Search

[Alt+S]

United States (N. Virginia)Account ID: 7950-9629-6861voclabs/user3807391=Suman\_Jakhar

CloudFormation > Stacks > RDS-Suman

Stacks (4)

Filter status

Filter by stack name

Active

View nested

Stacks

RDS-Suman

2025-08-12 12:11:58 UTC-0400

CREATE\_COMPLETE

S3-Suman

2025-08-12 12:10:08 UTC-0400

CREATE\_COMPLETE

EC2-Suman

2025-08-12 12:06:56 UTC-0400

CREATE\_COMPLETE

c167448a4304685111167524t1w795096296861

2025-08-12 10:09:40 UTC-0400

CREATE\_COMPLETE

Stack info

Events

Resources

Outputs

Parameters

Template

Change set

Overview

Stack ID

am:aws:cloudformation:us-east-1:795096296861:stack/RDS-Suman/12837ee0-7797-11f0-91e5-0affd20cdf1

Description

-

Status

CREATE\_COMPLETE

Detailed status

-

Status reason

-

Root stack

-

Parent stack

-

Created time

2025-08-12 12:11:58 UTC-0400

Updated time

-

Deleted time

-

Drift status

NOT\_CHECKED

Last drift check time

-

Termination protection

Deactivated

IAM role

Stacks (4)

Delete

Update stack

Stack actions

Create stack

Filter status

Filter by stack name

Active

View nested

Stack name

Status

Created time

Description

RDS-Suman

CREATE\_COMPLETE

2025-08-12 12:11:58 UTC-0400

-

S3-Suman

CREATE\_COMPLETE

2025-08-12 12:10:08 UTC-0400

3 private S3 buckets with versioning

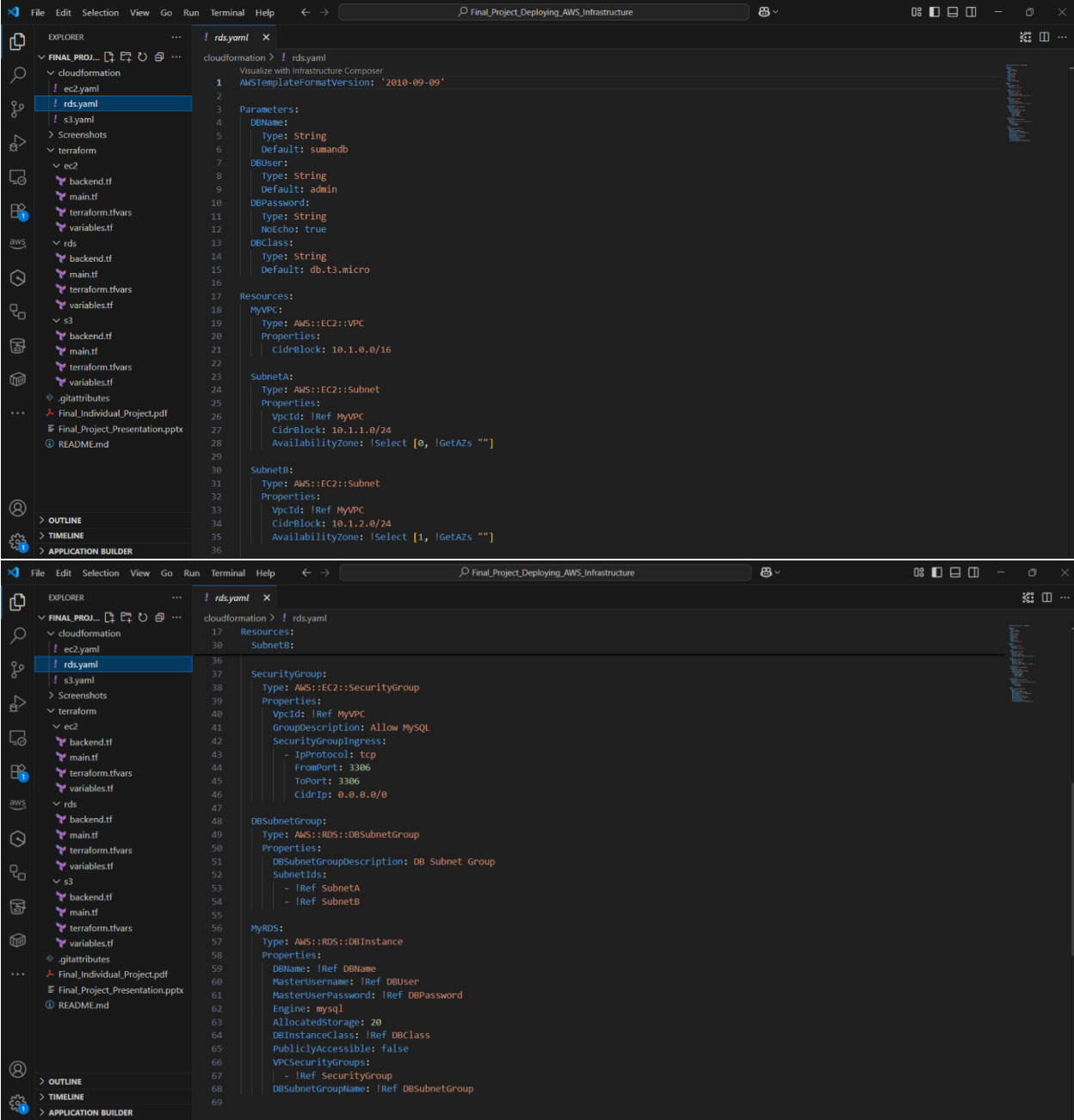
EC2-Suman

CREATE\_COMPLETE

2025-08-12 12:06:56 UTC-0400

EC2 in custom VPC with SSH + auto AMI from SSM

Code Snippet:



4. **Dynamic Configuration** to Avoid hardcoding all values. 

- o Use variables files (variables.tf, .tfvars) and CloudFormation Parameters where applicable.

5. **Backend/State Management**

- o Store Terraform state file locally on your laptop.
- o Use AWS CLI or AWS Console for CloudFormation stack deployment.

6. **GitHub Repository**

- Push your Terraform code and CloudFormation YAML files to a GitHub repository.
- Your repository must include:
  - o main.tf (Terraform configuration)
  - o variables.tf (Variables definition)
  - o terraform.tfvars (Variables values; sensitive data should not be pushed to GitHub)
  - o backend.tf (Backend configuration)
  - o CloudFormation YAML files for S3, EC2, and RDS.
  - o README.md (Documentation)
- Share your GitHub repository URL along with the submission document.

7. **Presentation/Demo**

- Prepare a **5-10 minute presentation** explaining:
  - o Your code structure and implementation.

- The AWS infrastructure you created.
- Key features or challenges you encountered.
- How your Terraform code and CloudFormation ensures modularity and best practices.
- **Live Demo:**
  - Run your Terraform configuration (terraform init, terraform plan, terraform apply).
  - Create and run CloudFormation Stack using YAML's. ○ Show the resources created in the AWS Management Console (e.g., the S3 bucket, EC2 instance, VPC, RDS, etc.).
  - Demonstrate the use of your tfvars file, YAML and backend configuration.

---

**Submission Details • Submission Items: One document containing below items**

1. GitHub repository link.
2. Screenshot(s) showing:
  - S3 Buckets created and versioning enabled.
  - EC2 Instances launched with Public IP.
  - RDS Instances running.
  - Terraform and CloudFormation code snippets.
  - Terraform apply or CloudFormation deployment outputs.
3. **Terraform Files:**
  - main.tf, provider.tf, variables.tf, vars.tfvars
4. **CloudFormation Templates:**
  - YAML files for S3, EC2, and RDS.
5. Clear and concise documentation in the README.md (in GitHub repo)
6. Power Point Presentation(PPT) slides for demo.

---

**Assessment Criteria (Total: 30 points)**

**Weightage in Final grade: 35% 1. Functionality (10**

- points) ○ Are all resources deployed correctly?
2. **Best Practices (5 points)** ○ Use of variables and tfvars files. ○ Proper backend configuration. ○ Dynamic configuration, clean code.
3. **Documentation (5 points)** ○ Is the README.md clear and comprehensive? ○ Can a third party replicate the setup using your documentation? ○ Are comments added in the code?
4. **Presentation/Demo (10 points)**
- Was the presentation well-structured and informative? ○ Did the demo showcase the infrastructure and code effectively?
  - Were challenges and solutions explained clearly during presentation in the class?
  - Was the presentation well-structured and informative?

---

**Resources**

- **Terraform Documentation:** <https://registry.terraform.io/>
- **AWS Free Tier:** <https://aws.amazon.com/free/>
- **GitHub Guides:** <https://guides.github.com/>