**GitHub Repo:** <https://github.com/TwinkleM97/IAC-Final-Project-8894858>

**Student: Twinkle Mishra  
Student ID: 8894858  
Instructor: Prof. Vikas Vattikonda**

**PROG 8870 - Final Project**

**DEPLOYING AWS INFRASTRUCTURE WITH TERRAFORM ANDCLOUDFORMATION**

# 1. Overview

This project provisions a small AWS environment in two ways Terraform and CloudFormation to practice IaC best practices. The environment includes private S3 buckets, an EC2 instance in a custom VPC, and a MySQL RDS instance. The work emphasizes parameterization, reproducibility, and a clean teardown.

# 2. What Gets Deployed

## 2.1 Terraform

* VPC (10.20.0.0/16) with two public subnets, Internet Gateway, and route table
* Security Group allowing SSH (22) from a single /32 CIDR
* EC2 (Amazon Linux 2023) with public IP (AMI via SSM parameter)
* Four private S3 buckets, Public Access Block enabled, Versioning enabled
* MySQL RDS (db.t3. micro) in dedicated DB subnet group (private)

## 2.2 CloudFormation

* S3: three private buckets with PublicAccessBlock (+ optional versioning)
* EC2: VPC + IGW + route table + subnet + instance; outputs public IP
* RDS: public MySQL (demo only, per spec) with SG allowing 3306

# 3. How to Reproduce

*Run these from the repository root. Region used: us-east-1.*

## 3.1 Environment Variables

export AWS\_REGION="us-east-1"

export AWS\_DEFAULT\_REGION="$AWS\_REGION"

export KEY\_NAME="twinkle-key” #existing EC2 key pair name

export SSH\_CIDR="YOUR.PUBLIC.IP/32" # e.g 20.42.11.16/32

export DB\_USER="adminuser"

set +H

export DB\_PASS='StaYAlive\_2025!' # example; use your own strong password

set -H

aws sts get-caller-identity --region "$AWS\_REGION"

## 3.2 Terraform Path

terraform -chdir=terraform init

terraform -chdir=terraform plan

terraform -chdir=terraform apply -auto-approve

terraform -chdir=terraform output

EC2\_IP="$(terraform -chdir=terraform output -raw ec2\_public\_ip)"

echo "EC2\_IP=$EC2\_IP"

## 3.3 CloudFormation Path

# S3

aws cloudformation deploy \

--stack-name cfn-s3-8894858 \

--template-file cloudformation/s3.yaml \

--parameter-overrides BucketPrefix=twinkle-8894858 Suffix=demo EnableVersioning=true \

--capabilities CAPABILITY\_NAMED\_IAM \

--region "$AWS\_REGION"

# EC2

aws cloudformation deploy \

--stack-name cfn-ec2-8894858 \

--template-file cloudformation/ec2.yaml \

--parameter-overrides KeyName="$KEY\_NAME" SSHCidr="$SSH\_CIDR" \

--capabilities CAPABILITY\_NAMED\_IAM \

--region "$AWS\_REGION"

# RDS (public)

set +H

aws cloudformation deploy \

--stack-name cfn-rds-8894858 \

--template-file cloudformation/rds.yaml \

--parameter-overrides DBName=proj8870db MasterUsername="$DB\_USER" MasterPassword="$DB\_PASS" \

--capabilities CAPABILITY\_NAMED\_IAM \

--region "$AWS\_REGION"

set -H

## 3.4 Verify RDS from EC2

# From your local shell:

ssh -o StrictHostKeyChecking=accept-new -i "${KEY\_NAME}.pem" ec2-user@"$EC2\_IP"

# *On the EC2 host:*

mysql --version || sudo dnf install -y mariadb105

# *Get the public CFN RDS endpoint (from your local shell)*

CFN\_ENDPOINT="$(aws rds describe-db-instances \

--query "DBInstances[?PubliclyAccessible==`true`].Endpoint.Address | [0]" \

--output text --region "$AWS\_REGION")"

# *Back on EC2 shell, run:*

mysql --connect-timeout=5 --protocol=TCP \

-h "$CFN\_ENDPOINT" -u "$DB\_USER" -p \

-e "SELECT VERSION() AS mysql\_version, NOW() AS server\_time;"

# 4. Evidence (Screenshots in repo /images)

## 1. AWS Identity

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: AWS CLI is authenticated to the correct account; confirms User ID, Account and ARN.* **(file: 00\_sts\_identity.png)**

## 2. Terraform Init

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: Initializes providers and backend; ready to run plan/apply.* **(file: 01\_tf\_init.png)**

## 3. Terraform Plan (Resources)

A screenshot of a computer program

AI-generated content may be incorrect.

*Pointer: Shows planned resources to be created across VPC, EC2, S3, and RDS.* **(file: 02\_tf\_plan.png)**

## 4. Terraform Plan (Summary)

A screenshot of a computer program

AI-generated content may be incorrect.

*Pointer: Plan summary indicating \*\*25 to add\*\* and no changes/destroys.* **(file: 02\_tf\_plan2.png)**

## 5. Terraform Apply (Progress)

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: Apply running; providers creating resources in AWS.* **(file: 03\_tf\_apply.png)**

## 6. Terraform Apply (Success)

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: Apply complete – \*\*25 added\*\*; output variable will provide EC2 public IP.* **(file: 03\_tf\_apply\_success.png)**

## 7. Terraform Outputs



*Pointer: terraform output returns \*\*ec2\_public\_ip\*\* used for SSH connectivity.* **(file: 04\_tf\_outputs.png)**

## 8. Terraform S3 + Versioning

A black screen with white text

AI-generated content may be incorrect.

*Pointer: Lists private S3 buckets created by Terraform and verifies \*\*Versioning: Enabled\*\*.* **(file: 05\_tf\_s3\_versioning.png)**

## 9. EC2 Public IP via AWS CLI

A computer screen with text and numbers

AI-generated content may be incorrect.

*Pointer: Describes the tagged instance and returns its current \*\*PublicIpAddress\*\*.* **(file: 06\_tf\_ec2\_publicip.png)**

## 10. SSH to EC2

A computer screen shot of a program

AI-generated content may be incorrect.

*Pointer: Successful SSH into AL2023 instance; whoami/hostname and Instance Metadata public-ipv4 shown.* **(file: 07\_tf\_ec2\_ssh.png)**

## 11. RDS Connectivity Check

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: Client connects to MySQL on RDS and returns \*\*SELECT VERSION()\*\* and \*\*NOW()\*\*.* **(file: 08\_mysql\_version.png)**

## 12. RDS SELECT from EC2

A screen shot of a computer

AI-generated content may be incorrect.

*Pointer: From EC2, MySQL client connects to RDS endpoint and returns server time.* **(file: 09\_tf\_rds\_query.png)**

## 13. RDS Demo Table

A computer screen shot of a black screen

AI-generated content may be incorrect.

*Pointer: Creates schema + table, inserts a row, and queries data on the RDS instance.* **(file: 10\_tf\_rds\_table\_demo.png)**

## 14. CFN S3 Stack Deploy

A computer screen shot of a program code

AI-generated content may be incorrect.

*Pointer: CloudFormation deploy creates 3 private S3 buckets; versioning confirmed with CLI.* **(file: 11\_cfn\_s3\_deploy\_and\_versioning.png)**

## 15. CFN EC2 Stack Deploy

A computer screen with text

AI-generated content may be incorrect.

*Pointer: CloudFormation deploy for EC2 completes successfully.* **(file: 12\_cfn\_ec2\_deploy.png)**

## 16. CFN EC2 Outputs

A computer screen with a black background

AI-generated content may be incorrect.

*Pointer: CloudFormation \*\*Outputs\*\* tab shows the EC2 public IP exported by the template.* **(file: 13\_cfn\_ec2\_outputs.png)**

## 17. CFN RDS Stack Deploy

A screen shot of a computer

AI-generated content may be incorrect.

*Pointer: CloudFormation deploy for public MySQL RDS completes successfully.* **(file: 14\_cfn\_rds\_deploy.png)**

## 18. CFN RDS Endpoint (CLI)

A screen shot of a computer code

AI-generated content may be incorrect.

*Pointer: CLI resolves the CFN-created RDS endpoint DNS name.* **(file: 15\_cfn\_rds\_endpoint.png)**

## 19. CFN RDS Describe (CLI)

A computer screen with white text

AI-generated content may be incorrect.

*Pointer: Summarizes RDS identifier, status, and endpoint in a table format.* **(file: 16\_cfn\_rds\_query.png)**

## 20. Code: Terraform EC2 + AMI via SSM

A computer screen shot of a program

AI-generated content may be incorrect.

*Pointer: Shows TF using SSM Parameter Store to dynamically pull the latest AL2023 AMI.* **(file: 21\_code\_tf\_ec2\_ssm.png)**

## 21. Code: Terraform S3 Versioning

A computer screen shot of a program

AI-generated content may be incorrect.

*Pointer: TF module loop creating private buckets and setting bucket versioning dynamically.* **(file: 22\_code\_tf\_s3\_versioning.png)**

## 22. Code: CFN EC2 YAML

A screenshot of a computer program

AI-generated content may be incorrect.

*Pointer: YAML template for VPC + EC2 with parameters (KeyName, SSHCidr) and outputs.* **(file: 23\_code\_cfn\_ec2\_yaml.png)**

## 23. Code: CFN RDS YAML

A screenshot of a computer program

AI-generated content may be incorrect.

*Pointer: YAML template creating its own networking (VPC, IGW, subnets) and a public RDS.* **(file: 24\_code\_cfn\_rds\_yaml.png)**

## 24. Code: CFN S3 YAML

A screenshot of a computer program

AI-generated content may be incorrect.

*Pointer: YAML template creating 3 private S3 buckets with PublicAccessBlockConfiguration.* **(file: 25\_code\_cfn\_s3\_yaml.png)**

## 25. Code: Terraform Variables

A screen shot of a computer program

AI-generated content may be incorrect.

*Pointer: Centralized variables (region, ids, db creds, CIDRs) supporting dynamic configuration.* **(file: 26\_code\_tf\_variables.png)**

## 26. AWS Console: S3 (TF bucket versioning)

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: S3 console showing \*\*Bucket Versioning: Enabled\*\* on Terraform-created bucket.* **(file: 27\_console\_s3\_tf\_bucket\_versioning.png)**

## 27. AWS Console: S3 (CFN buckets list)

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: S3 console listing the three CloudFormation-created buckets.* **(file: 28\_console\_s3\_cfn\_buckets\_list.png)**

## 28. AWS Console: EC2 instance details

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: EC2 console verifies instance running state and \*\*Public IPv4\*\*.* **(file: 29\_console\_ec2\_instance\_details.png)**

## 29. AWS Console: SG inbound SSH

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: Security Group allows \*\*TCP/22\*\* only from your public \*\*/32\*\*.* **(file: 30\_console\_sg\_inbound\_ssh.png)**

## 30. AWS Console: VPC subnets

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: Shows public subnets used for EC2/RDS per the templates.* **(file: 31\_console\_vpc\_subnets.png)**

## 31. AWS Console: Route table

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: Public route table has \*\*0.0.0.0/0 → Internet Gateway\*\*, confirming internet egress.* **(file: 32\_console\_vpc\_route\_table.png)**

## 32. AWS Console: RDS (Terraform instance)

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: RDS created by Terraform – engine, class, AZ and connectivity settings.* **(file: 33\_console\_rds\_tf\_instance.png)**

## 33. AWS Console: RDS (CFN instance)

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: RDS created by CloudFormation – status \*\*Available\*\*, endpoint in Connectivity & security.* **(file: 34\_console\_rds\_cfn\_instance.png)**

## 34. AWS Console: CloudFormation stacks

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: All three stacks show \*\*CREATE\_COMPLETE\*\*.* **(file: 35\_console\_cfn\_stacks.png)**

## 35. AWS Console: CFN EC2 Outputs

A screenshot of a computer

AI-generated content may be incorrect.

*Pointer: Outputs tab highlights the \*\*PublicIP\*\* output value.* **(file: 36\_console\_cfn\_ec2\_outputs.png)**

# 5. Best Practices & Dynamic Configuration

* No hardcoded secrets in git; use environment variables or local terraform.tfvars (example file provided).
* Parameterization throughout: Terraform variables and CloudFormation Parameters.
* S3 hardened with PublicAccessBlock; versioning enabled as a toggle.
* Security groups restrict SSH to a single /32 CIDR.
* Outputs exposed for demo convenience (e.g EC2 public IP).
* Backend configured as local per project requirements.

# 6. Cleanup (Avoid Costs)

terraform -chdir=terraform destroy -auto-approve

aws cloudformation delete-stack --stack-name cfn-rds-8894858 --region "$AWS\_REGION"

aws cloudformation delete-stack --stack-name cfn-ec2-8894858 --region "$AWS\_REGION"

aws cloudformation delete-stack --stack-name cfn-s3-8894858 --region "$AWS\_REGION"

aws cloudformation wait stack-delete-complete --stack-name cfn-s3-8894858 --region "$AWS\_REGION" || true

# 8. Challenges & Resolutions

* CFN RDS public access required IGW and public subnets; added IGW + correct routing.
* JMESPath quoting for endpoint selection: used backticks around booleans and index pipe.
* Amazon Linux 2023 MySQL client: installed `mariadb105` when `mysql` wasn’t present.
* GitHub repo hygiene: removed large binaries and key material; used `git filter-repo` to purge history and force-pushed.