

Lab Instructions: Provisioning Database Configuration in AWS using Terraform.

Introduction

In this lab, you will learn how to use Terraform to provision a simple database configuration in AWS (Amazon Web Services). Terraform is an infrastructure code tool that allows you to define and provision infrastructure resources in a declarative configuration language.

Prerequisites

Before starting this lab, make sure you have the following:

1. An AWS Academy lab account with appropriate permissions to create EC2 instances.
2. Terraform installed on your local machine. You can download Terraform from the official website.

Task

Your task is to write Terraform code to provision the following configuration in AWS in a fresh configuration.

1. Provision an RDS resource with a postgres engine as we discussed in the class

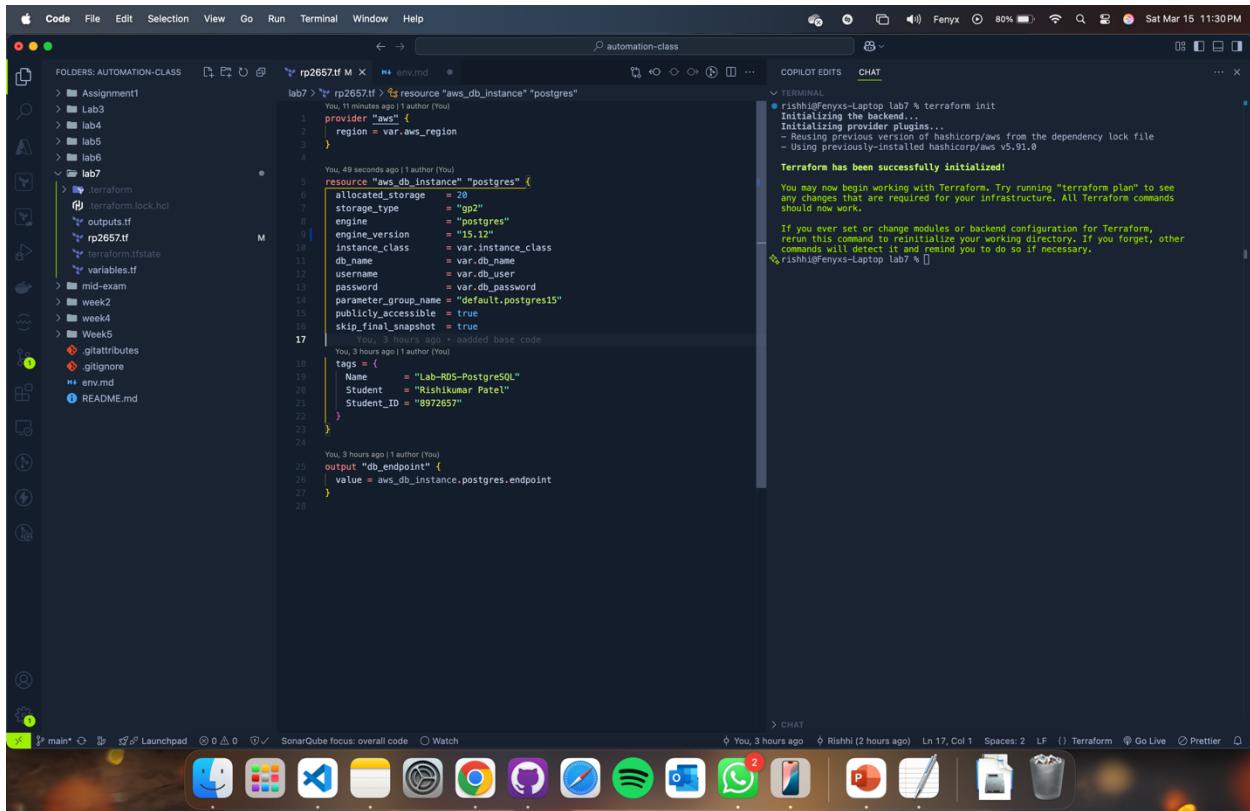
Deliverables

Submit your terraform code file, screenshots of your init, plan, apply commands and a screenshot of your provisioned resource in AWS. Also login to the provisioned resource from the aws console and attach screenshots.

Instructions

1. **Initialize Terraform:** Open a terminal window and navigate to the directory where you have your Terraform configuration file (e.g., main.tf). Run the following command to initialize Terraform:

```
terraform init
```



The screenshot shows a Mac OS X desktop environment. In the center is a terminal window titled 'rp2657.tf M'. The terminal output shows the process of initializing Terraform:

```
You, 11 minutes ago | author (You)
provider "aws" {
  region = var.aws_region
}

You, 49 seconds ago | author (You)
resource "aws_db_instance" "postgres" {
  allocated_storage = 20
  storage_type     = "gp2"
  engine           = "postgres"
  engine_version   = "15.12"
  instance_class   = var.instance_class
  db_name          = var.db_name
  username          = var.db_user
  password          = var.db_password
  parameter_group_name = "default.postgres15"
  publicly_accessible = true
  skip_final_snapshot = true
}

You, 3 hours ago | author (You)
tags = {
  Name      = "Lab-RDS-PostgreSQL"
  Student   = "Rishikumar Patel"
  Student_ID = "8972657"
}

You, 3 hours ago | author (You)
output "db_endpoint" {
  value = aws_db_instance.postgres.endpoint
}

rishihs@enys-Laptop lab7 % terraform init
initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.91.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

The terminal window is part of a larger interface with a sidebar containing project files like 'Assignment1', 'lab3', 'lab4', 'lab5', 'lab6', and 'lab7'. The status bar at the bottom shows the date and time as 'Sat Mar 15 11:30PM'.

2. **Write Terraform Configuration:** Create a new file (e.g., main.tf) and write Terraform configuration code to define the AWS resources needed for the server.

3. Provision Infrastructure: Once you have written the Terraform configuration, run the following command to preview the changes that Terraform will make:

terraform plan

The screenshot shows a Mac desktop environment. In the center is a terminal window titled 'automation-class' with the command 'terraform plan' running. The output of the plan is displayed, showing the actions Terraform will perform to create an AWS PostgreSQL database instance. To the left of the terminal is a file browser showing the directory structure of the Terraform workspace, including files like 'rp2657.tf', 'env.md', and 'README.md'. The desktop dock at the bottom contains various application icons.

```
You, 11 minutes ago | 1 author (You)
resource "aws_db_instance" "postgres" {
  provider = "aws"
  region   = var.aws_region
}

You, 1 minute ago | 1 author (You)
resource "aws_db_instance" "postgres" {
  allocated_storage      = 20
  storage_type           = "gp2"
  engine                 = "postgres"
  engine_version         = "15.12"
  instance_class          = var.instance_class
  db_name                = var.db_name
  username               = var.db_user
  password               = var.db_password
  parameter_group_name   = "default.postgres15"
  publicly_accessible     = true
  skip_final_snapshot     = true
}

You, 3 hours ago | 1 author (You)
tags = [
  { Name      = "Lab-ROS-PostgreSQL"
  Student    = "Rishikumar Patel"
  Student_ID = "8972657" }
]

You, 9 hours ago | 1 author (You)
output "db_endpoint" {
  value = aws_db_instance.postgres.endpoint
}

TERRINARY
rishihi@Fenyx-Laptop lab7 % terraform plan
rishihi@Fenyx-Laptop lab7 % terraform plan

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.postgres will be created
+ resource "aws_db_instance" "postgres" {
  + address                                = (known after apply)
  + allocated_storage                      = 20
  + availability_zone                     = (known after apply)
  + auto_minor_version_upgrade            = true
  + auto_minor_version_upgrade_time       = (known after apply)
  + backup_retention_period               = (known after apply)
  + backup_target                          = (known after apply)
  + backup_window                          = (known after apply)
  + character_set_name                   = (known after apply)
  + copy_tags_to_snapshot                = (known after apply)
  + db_name                               = "mydatabase"
  + dedicated_ip                         = (known after apply)
  + dedicated_log_volume                = false
  + delete_automated_backups            = true
  + domain_fqdn                         = (known after apply)
  + engine                                = "postgres"
  + engine_lifecycle_support            = (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                    = "db.t3.micro"
  + instance_class                       = (known after apply)
  + iops                                  = (known after apply)
  + key_id                                = (known after apply)
  + latest_restorable_time              = (known after apply)
  + license_model                         = (known after apply)
  + maintenance_window                  = (known after apply)
  + master_user_secret                 = (known after apply)
  + master_user_secret_kms_key_id      = (known after apply)
  + master_user_secret_kms_key_id_time = (known after apply)
  + monitoring_role_arn                = (known after apply)
  + multi_az                             = (known after apply)
  + multi_az_character_set_name        = (known after apply)
  + network_type                         = (known after apply)
  + option_group_name                   = (known after apply)
  + parameter_group_name                = "default.postgres15"
  + performance_insights_enabled      = false
  + performance_insights_kms_key_id   = (known after apply)
  + performance_insights_retention_period = (known after apply)
  + publicly_accessible                = true
  + replica_mode                         = (known after apply)
}
```

Review the plan to ensure it matches your expectations.

4. **Apply Changes:** If the plan looks correct, apply the changes to provision the server in AWS:

terraform apply

Type yes when prompted to confirm the action.

The screenshot shows a Mac desktop environment with a code editor open in the foreground. The code editor displays a Terraform configuration file (`variables.tf`) for creating an AWS PostgreSQL database instance. The configuration includes variables for provider, region, allocated storage, storage type, engine version, instance class, db name, username, password, publicly accessible, skip final snapshot, and tags. An output block defines the endpoint of the created database instance. The terminal window to the right shows the execution of the `terraform apply` command, which is creating the database instance. The terminal output includes a list of changes to outputs, a confirmation prompt asking if the user wants to perform the actions, and the final message "Apply complete! Resources: 1 added, 0 changed, 0 destroyed." The status bar at the bottom indicates the user's name is Rishabh, the date is Mar 15, and the time is 11:38 PM.

```
You, 3 minutes ago | author (You)
resource "aws_db_instance" "postgres" {
    provider = "aws"
    region = var.aws_region
}

You, 3 minutes ago | author (You)
resource "aws_db_instance" "postgres" {
    allocated_storage = 20
    storage_type = "gp2"
    engine = "postgres"
    engine_version = "15.12"
    instance_class = var.instance_class
    db_name = var.db_name
    username = var.db_user
    password = var.db_password
    parameter_group_name = "default.postgres15"
    publicly_accessible = true
    skip_final_snapshot = true
}

You, 3 hours ago | author (You)
tags = {
    Name = "Lab-RDS-PostgreSQL"
    Student = "Rishikumar Patel"
    Student_ID = "0972657"
}

You, 3 hours ago | author (You)
output "db_endpoint" {
    value = aws_db_instance.postgres.endpoint
}
```

```
Planned: 1 to add, 0 to change, 0 to destroy.
Changes to Outputs:
  + db_endpoint = (known after apply)
  + rds_endpoint = (known after apply)

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes
```

```
aws_db_instance.postgres: Creating...
aws_db_instance.postgres: Still creating... [10s elapsed]
aws_db_instance.postgres: Still creating... [20s elapsed]
aws_db_instance.postgres: Still creating... [30s elapsed]
aws_db_instance.postgres: Still creating... [40s elapsed]
aws_db_instance.postgres: Still creating... [50s elapsed]
aws_db_instance.postgres: Still creating... [60s elapsed]
aws_db_instance.postgres: Still creating... [1m0s elapsed]
aws_db_instance.postgres: Still creating... [1m10s elapsed]
aws_db_instance.postgres: Still creating... [1m20s elapsed]
aws_db_instance.postgres: Still creating... [1m30s elapsed]
aws_db_instance.postgres: Still creating... [1m40s elapsed]
aws_db_instance.postgres: Still creating... [1m50s elapsed]
aws_db_instance.postgres: Still creating... [2m0s elapsed]
aws_db_instance.postgres: Still creating... [2m10s elapsed]
aws_db_instance.postgres: Still creating... [2m20s elapsed]
aws_db_instance.postgres: Still creating... [2m30s elapsed]
aws_db_instance.postgres: Still creating... [2m40s elapsed]
aws_db_instance.postgres: Still creating... [2m50s elapsed]
aws_db_instance.postgres: Still creating... [3m0s elapsed]
aws_db_instance.postgres: Still creating... [3m10s elapsed]
aws_db_instance.postgres: Still creating... [3m20s elapsed]
aws_db_instance.postgres: Still creating... [3m30s elapsed]
aws_db_instance.postgres: Still creating... [3m40s elapsed]
aws_db_instance.postgres: Still creating... [3m50s elapsed]
aws_db_instance.postgres: Still creating... [4m0s elapsed]
aws_db_instance.postgres: Still creating... [4m10s elapsed]
aws_db_instance.postgres: Still creating... [4m20s elapsed]
aws_db_instance.postgres: Still creating... [4m30s elapsed]
aws_db_instance.postgres: Still creating... [4m40s elapsed]
aws_db_instance.postgres: Still creating... [4m50s elapsed]
aws_db_instance.postgres: Still creating... [5m0s elapsed]
aws_db_instance.postgres: Creation complete after 5m18s [id=db-FEXVXOYUK2V6B43XMCFEQABDJI]
```

```
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

```
Outputs:
```

```
db_endpoint = "terraform-28250316033239052000000001.cd42bopw56sz.us-east-1.rds.amazonaws.c
om:5432"
rds_endpoint = "terraform-28250316033239052000000001.cd42bopw56sz.us-east-1.rds.amazonaws.c
om:5432"
```

You, 13 minutes ago | Rishabh (12 minutes ago) Ln 12, Col 37 (7 selected) Spaces: 2 LF ⌂ Terraform ⌂ Go Live ⌂ Prettier ⌂

5. Access the Server: Once Terraform has finished provisioning the server, you should see the output with the server's public IP address

The screenshot shows the AWS RDS (Relational Database Service) console. On the left, there is a navigation sidebar with the following menu items:

- Dashboard
- Databases** (selected)
- 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 New
- Events
- Event subscriptions
- Recommendations: 0
- Certificate update

The main content area displays a table titled "Databases (1)". The table has one row for the database "terraform-20250316033239052000000001". The columns include DB identifier, Status, Role, Engine, Region ..., Size, Recommendations, and CPU usage (5.69%).

At the top of the main content area, there is a notification bar with the message: "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." It also includes links to "RDS User Guide" and "Aurora User Guide".

Below the main content area, there is a toolbar with buttons for "CloudShell" and "Feedback", and a Mac OS-style dock at the bottom.

In the second screenshot, the same database instance is selected, and the "Connectivity & security" tab is active. The "Summary" section shows the DB identifier as "terraform-20250316033239052000000001", status as "Available", role as "Instance", engine as "PostgreSQL", and region as "us-east-1f".

The "Connectivity & security" section contains three tabs: "Endpoint & port", "Networking", and "Security".

- Endpoint & port:** Shows the endpoint as "terraform-20250316033239052000000001.cd42bopw56sz.us-east-1.rds.amazonaws.com" and port as "5432".
- Networking:** Shows the availability zone as "us-east-1f", VPC as "vpc-08885261ad3cb46b5", subnet group as "default", and subnets: "subnet-049d43503da44e1fa", "subnet-086932ec4bb99776", "subnet-07419397a09f75666", "subnet-00d69ae54ae2e8af1", "subnet-069091f3f15bd35f", and "subnet-043433d40b7de251".
- Security:** Shows VPC security groups as "default (sg-0187adb47cd2c9e6b)" (Active), publicly accessible as "Yes", certificate authority as "rds-ca-rsa2048-g1", and certificate authority date as "May 25, 2061, 19:34 (UTC-04:00)".

6. **Cleanup:** After completing the lab, remember to destroy the resources to avoid incurring unnecessary charges:

terraform destroy

The screenshot shows a Mac desktop with a terminal window open. The terminal is executing a 'terraform destroy' command on a configuration file named 'rp2657.tf'. The configuration file defines an AWS RDS PostgreSQL instance with various parameters like storage size, engine version, and security groups. The terminal output shows the destruction of the resource, including the deletion of the database endpoint and associated AWS resources. A confirmation message asks if the user really wants to destroy all resources, with a note that there is no undo.

```
You, 27 minutes ago | 1 author (You)
resource "aws_db_instance" "postgres" {
  provider = "aws"
  region  = var.aws_region
}

You, 37 minutes ago | 1 author (You)
resource "aws_db_instance" "postgres" {
  allocated_storage    = 20
  storage_type         = "gp2"
  engine               = "postgres"
  engine_version       = "15.12"
  instance_class       = var.instance_class
  db_name              = var.db_name
  username             = var.db_user
  password             = var.db_password
  parameter_group_name = "default.postgres15"
  publicly_accessible   = true
  skip_final_snapshot  = true
}

You, 3 hours ago | 1 author (You)
tags = [
  { Name = "Lab-RDS-PostgreSQL"
    Student = "Rishikumar Patel"
    Student_ID = "8972657" }
]

You, 3 hours ago | 1 author (You)
output "db_endpoint" {
  value = aws_db_instance.postgres.endpoint
}

TERRAFORM STATE
Changes to Outputs:
  - aws_db_instance.postgres.endpoint = "db-FEXVX0UK2V6843MCFFQABDJI"
  - rds_endpoint = "terraform-20250316033239052000000001.cd42bopw56sz.us-east-1.rds.amazonaws.com:5432"
  - aws.com:5432 = null
  - rds_endpoint = "terraform-20250316033239052000000001.cd42bopw56sz.us-east-1.rds.amazonaws.com:5432"
  - aws.com:5432 = null

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: [
```

Type yes when prompted to confirm the destruction of resources.

The screenshot shows a Mac desktop environment with a code editor and a terminal window. The code editor displays two files: `rp2657.tf` and `variables.tf`. The `rp2657.tf` file contains a Terraform configuration for creating an AWS PostgreSQL database instance. The `variables.tf` file defines variables for the database name, engine version, and other parameters. The terminal window shows the output of a `terraform destroy` command, which is prompting the user to confirm the destruction of resources. The desktop dock at the bottom shows various application icons.

```
Code File Edit Selection View Go Run Terminal Window Help
FOLDERS: AUTOMATION-CLASS
> Assignment1
> lab3
> lab4
> lab5
> lab6
> lab7
  > rp2657.tf M
  > variables.tf M
  > env.md
  > .terraform
    > .terraform.lock.hcl
    > outputs.tf
    > rp2657.tf
    > terraform.state
    > terraform.state.backup
  > variables.tf M
  > mid-exam
  > week2
  > week4
  > Week5
    > .gitattributes
    > .gitignore
  > env.md
  > README.md

Code File Edit Selection View Go Run Terminal Window Help
automation-class
lab7 > rp2657.tf > resource "aws_db_instance" "postgres" > username
You, 37 minutes ago | author (You)
1 provider "aws" {
2   region = var.aws_region
3 }
4
5 You, 27 minutes ago | author (You)
6 resource "aws_db_instance" "postgres" {
7   allocated_storage      = 20
8   storage_type          = "gp2"
9   engine                = "postgres"
10  engine_version        = "15.12"
11  instance_class       = var.instance_class
12  db_name               = var.db_name
13  username              = var.db_user
14  password              = var.db_password
15  parameter_group_name = "default.postgres15"
16  publicly_accessible   = true
17  skip_final_snapshot   = true
18
19  tags = {
20     Name      = "Lab-RDS-PostgreSQL"
21     Student   = "Rishikumar Patel"
22     Student_ID = "8972657"
23   }
24
25 You, 3 hours ago | author (You)
26 output "db_endpoint" {
27   value = aws_db_instance.postgres.endpoint
28 }

tags = {
  "Name"      = "Lab-RDS-PostgreSQL"
  "Student"   = "Rishikumar Patel"
  "Student_ID" = "8972657"
}

You, 37 minutes ago | author (You)
resource "aws_db_instance" "postgres" {
  allocated_storage      = 20
  storage_type          = "gp2"
  engine                = "postgres"
  engine_version        = "15.12"
  instance_class       = var.instance_class
  db_name               = var.db_name
  username              = var.db_user
  password              = var.db_password
  parameter_group_name = "default.postgres15"
  publicly_accessible   = true
  skip_final_snapshot   = true
}
tags = {
  "Name"      = "Lab-RDS-PostgreSQL"
  "Student"   = "Rishikumar Patel"
  "Student_ID" = "8972657"
}

tags_all = [
  "Name"      = "Lab-RDS-PostgreSQL"
  "Student"   = "Rishikumar Patel"
  "Student_ID" = "8972657"
]
tags_all = [
  "username" = "fishi8972657" -> null
]
tags_all = [
  "vpc_security_group_ids" = [
    "sp-0187adb47cd2c9a6b",
  ] -> null
]
# (12 unchanged attributes hidden)

Plan: 0 to add, 0 to change, 1 to destroy.

Changes to Outputs:
  - db_endpoint -> null
  - rds_endpoint = "terraform-20250316033239052000000001.cd42bop56sz.us-east-1.rds.amazonaws.com:5432" -> null

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_db_instance.postgres: Destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 10s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 26s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 38s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 40s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 58s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 1m0s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 1m10s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 1m20s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 1m30s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 1m40s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 1m50s elapsed]
aws_db_instance.postgres: Still destroying... [id=db-FEXVXQYU2V6843XMCFEQABDJI, 2m0s elapsed]

Destroy complete! Resources: 1 destroyed.
rishihi@enxys-Laptop lab7 %
```