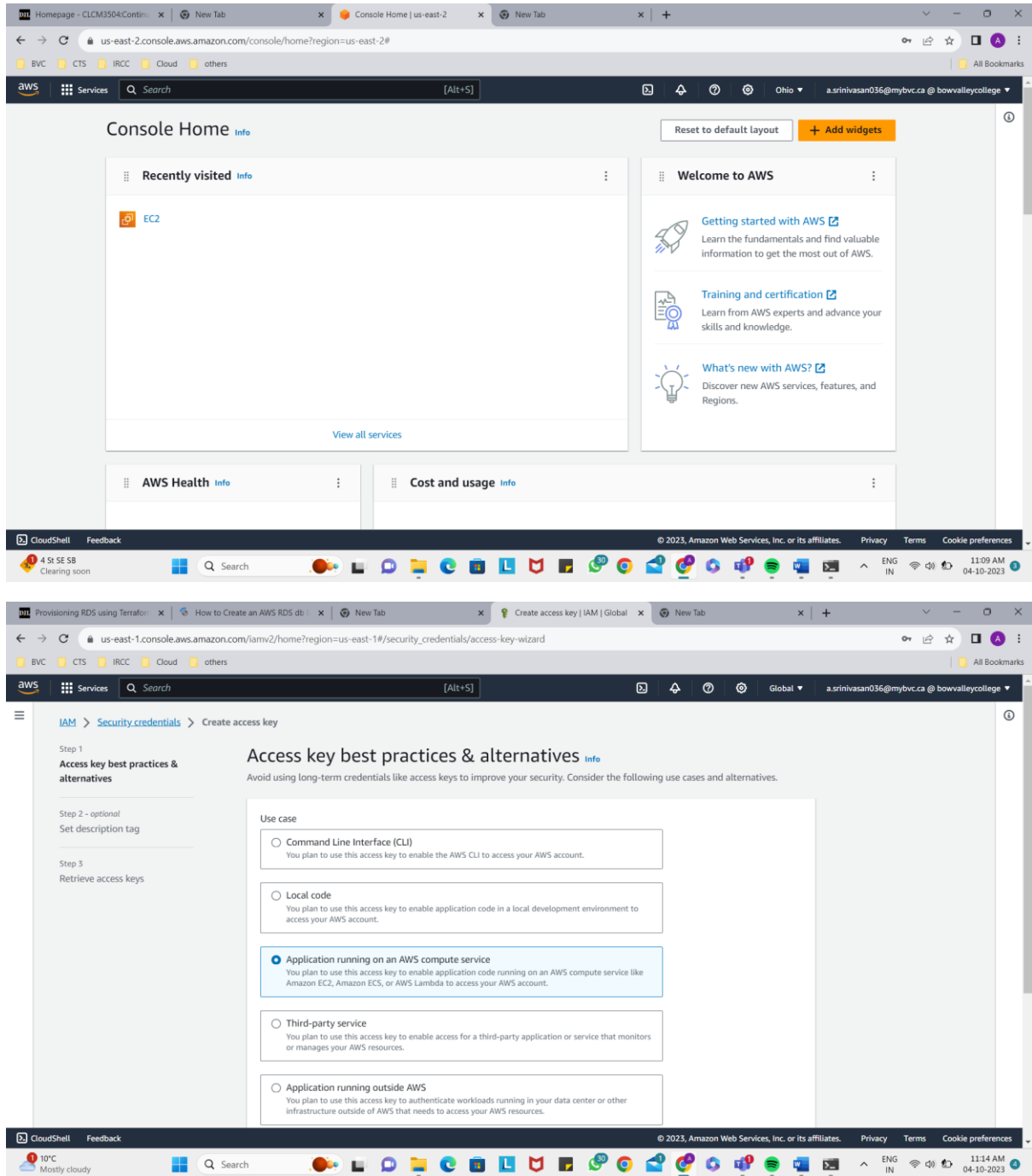
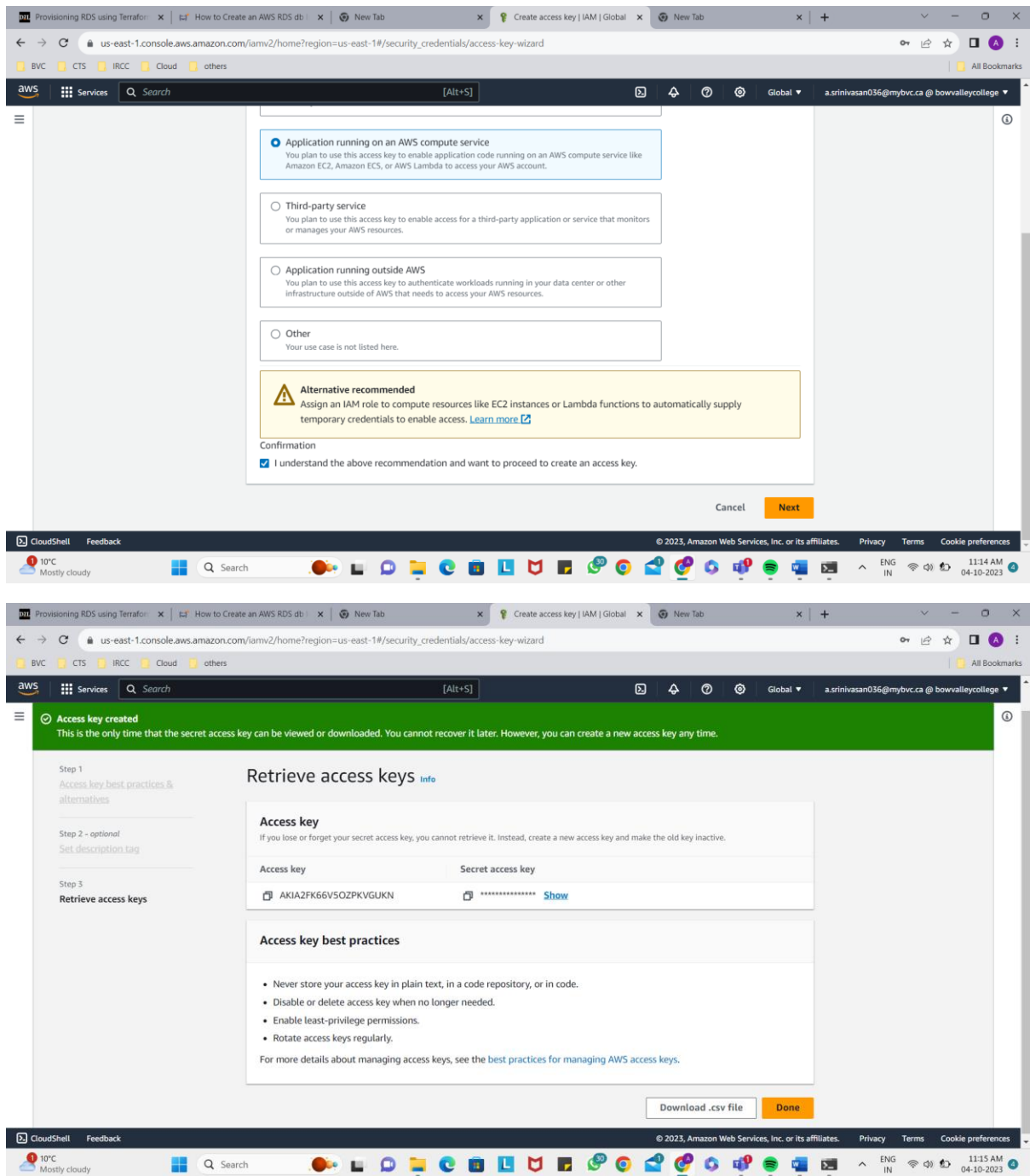


Create AWS RDS using Terraform



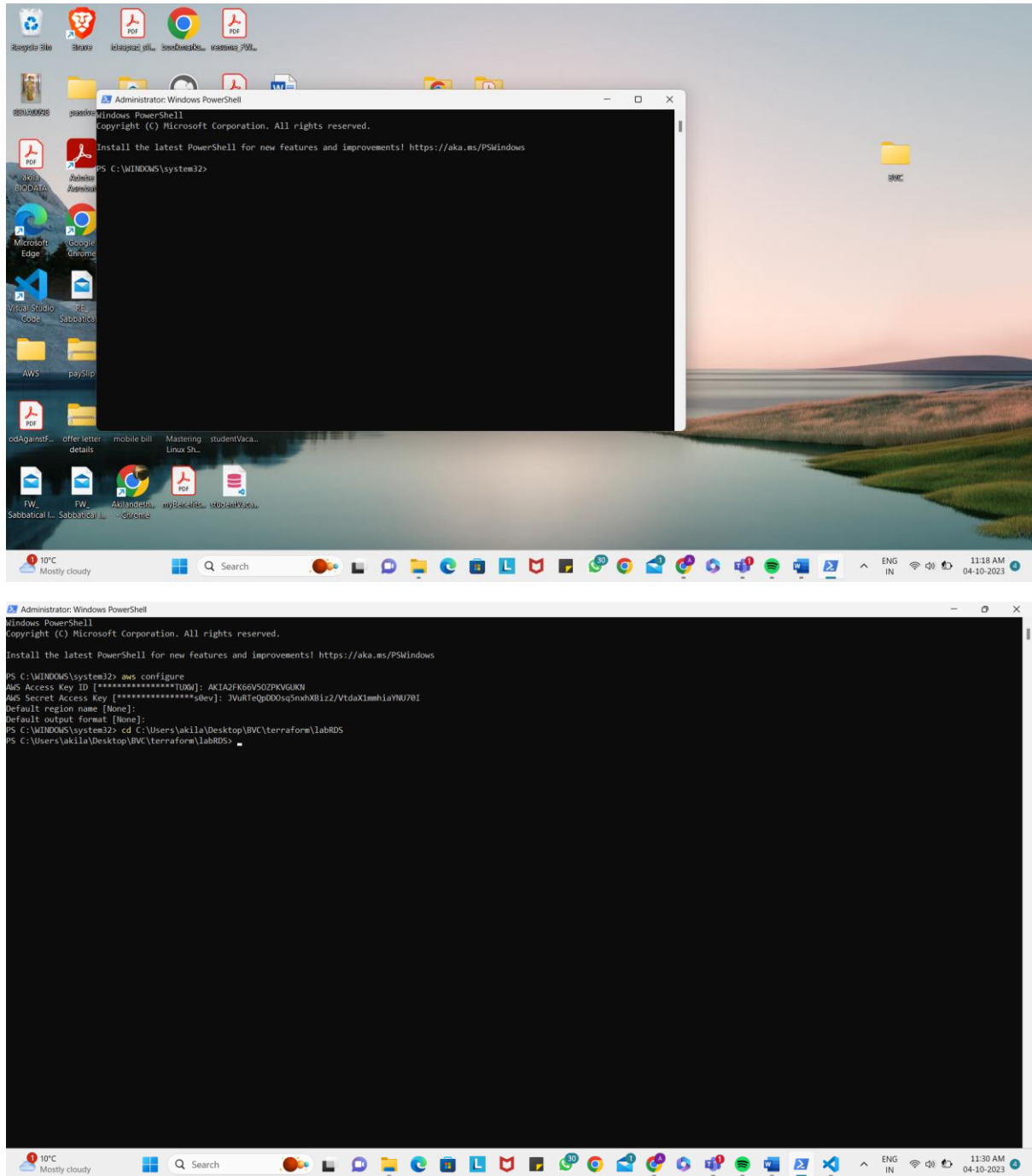
Create AWS RDS using Terraform



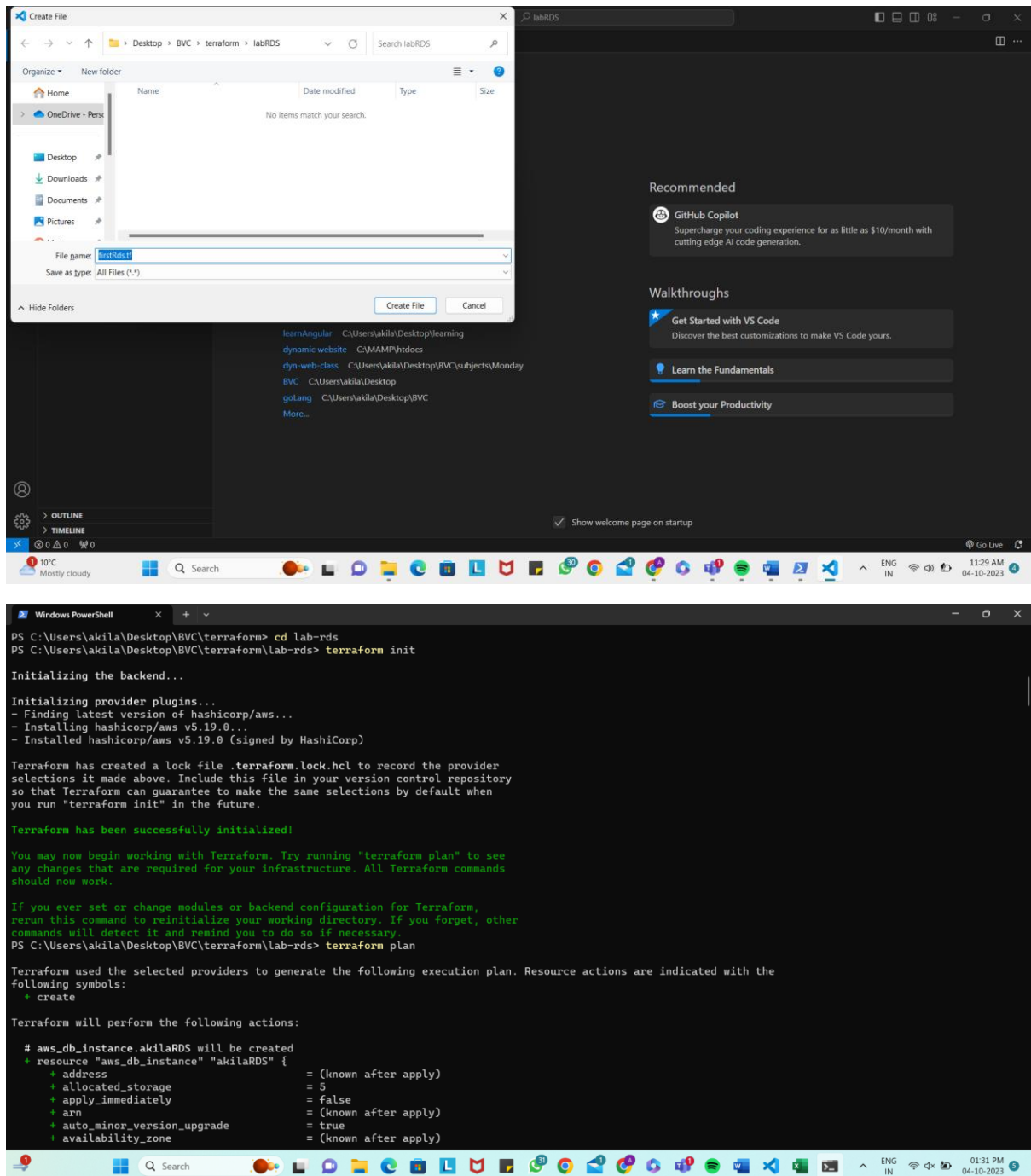
AKIA2FK66V5OTN7AHGG7

PS/ZyM0yywcnQWfNAAhFHTWfBgBdd9akcvZDzZbo

Create AWS RDS using Terraform



Create AWS RDS using Terraform



```
Windows PowerShell
PS C:\Users\akila\Desktop\BVC\terraform> cd lab-rds
PS C:\Users\akila\Desktop\BVC\terraform\lab-rds> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.19.0...
- Installed hashicorp/aws v5.19.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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.
PS C:\Users\akila\Desktop\BVC\terraform\lab-rds> 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.akilaRDS will be created
+ resource "aws_db_instance" "akilaRDS" {
  + address                               = (known after apply)
  + allocated_storage                     = 5
  + apply_immediately                     = false
  + arn                                   = (known after apply)
  + auto_minor_version_upgrade            = true
  + availability_zone                     = (known after apply)
```

Create AWS RDS using Terraform

```
Windows PowerShell
PS C:\Users\akila\Desktop\BVC\terraform\lab-rds> 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.akilaRDS will be created
+ resource "aws_db_instance" "akilaRDS" {
  + address                = (known after apply)
  + allocated_storage      = 5
  + 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
  + db_name                 = "akiladb"
  + db_subnet_group_name   = (known after apply)
  + delete_automated_backups = true
  + endpoint               = (known after apply)
  + engine                  = "mysql"
  + engine_version          = "8.0.33"
  + engine_version_actual   = (known after apply)
  + hosted_zone_id          = (known after apply)
  + id                     = (known after apply)
  + identifier              = "akila"
  + identifier_prefix       = (known after apply)
  + instance_class           = "db.t3.micro"
  + iops                    = (known after apply)
  + kms_key_id              = (known after apply)
  + latest_restorable_time  = (known after apply)
  + license_model            = (known after apply)
  + listener_endpoint       = (known after apply)
  + maintenance_window      = (known after apply)
}
```

```
Windows PowerShell
Plan: 1 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if
you run "terraform apply" now.
PS C:\Users\akila\Desktop\BVC\terraform\lab-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.akilaRDS will be created
+ resource "aws_db_instance" "akilaRDS" {
  + address                = (known after apply)
  + allocated_storage      = 5
  + 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
  + db_name                 = "akiladb"
  + db_subnet_group_name   = (known after apply)
  + delete_automated_backups = true
  + endpoint               = (known after apply)
  + engine                  = "mysql"
  + engine_version          = "8.0.33"
  + engine_version_actual   = (known after apply)
  + hosted_zone_id          = (known after apply)
  + id                     = (known after apply)
  + identifier              = "akila"
  + identifier_prefix       = (known after apply)
}
```

Create AWS RDS using Terraform

```
Windows PowerShell
+ snapshot_identifier      = (known after apply)
+ status                  = (known after apply)
+ storage_throughput      = (known after apply)
+ storage_type            = (known after apply)
+ tags_all                = (known after apply)
+ timezone                = (known after apply)
+ username                = "akila"
+ vpc_security_group_ids  = (known after apply)
}

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

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.akilaRDS: Creating...
aws_db_instance.akilaRDS: Still creating... [10s elapsed]
aws_db_instance.akilaRDS: Still creating... [20s elapsed]
aws_db_instance.akilaRDS: Still creating... [30s elapsed]
aws_db_instance.akilaRDS: Still creating... [40s elapsed]
aws_db_instance.akilaRDS: Still creating... [50s elapsed]
aws_db_instance.akilaRDS: Still creating... [1m0s elapsed]
aws_db_instance.akilaRDS: Still creating... [1m10s elapsed]
aws_db_instance.akilaRDS: Still creating... [1m20s elapsed]
aws_db_instance.akilaRDS: Still creating... [1m30s elapsed]
aws_db_instance.akilaRDS: Still creating... [1m40s elapsed]
aws_db_instance.akilaRDS: Still creating... [1m50s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m0s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m10s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m20s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m30s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m40s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m50s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m0s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m10s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m20s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m30s elapsed]

aws_db_instance.akilaRDS: Still creating... [1m40s elapsed]
aws_db_instance.akilaRDS: Still creating... [1m50s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m0s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m10s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m20s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m30s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m40s elapsed]
aws_db_instance.akilaRDS: Still creating... [2m50s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m0s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m10s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m20s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m30s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m40s elapsed]
aws_db_instance.akilaRDS: Still creating... [3m50s elapsed]
aws_db_instance.akilaRDS: Still creating... [4m0s elapsed]
aws_db_instance.akilaRDS: Still creating... [4m10s elapsed]
aws_db_instance.akilaRDS: Still creating... [4m20s elapsed]
aws_db_instance.akilaRDS: Still creating... [4m30s elapsed]
aws_db_instance.akilaRDS: Still creating... [4m40s elapsed]
aws_db_instance.akilaRDS: Still creating... [4m50s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m0s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m10s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m20s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m30s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m40s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m50s elapsed]
aws_db_instance.akilaRDS: Still creating... [6m0s elapsed]
aws_db_instance.akilaRDS: Still creating... [6m10s elapsed]
aws_db_instance.akilaRDS: Creation complete after 6m17s [id=db-KIROR532NIBQKFOA4QEXFWGDHQ]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\akila\Desktop\BVC\terraform\lab-rds>
```


Create AWS RDS using Terraform

The image displays the AWS RDS console and a Windows PowerShell terminal window, illustrating the creation and management of an Amazon RDS instance using Terraform.

AWS RDS Console:

- The console shows the "Databases (5)" list. The instances listed are:

DB identifier	Status	Role	Engine	Region & AZ	Size	Actions
akila	Available	Instance	MySQL Community	us-east-1c	db.t3.micro	-
example	Available	Instance	MySQL Community	us-east-1b	db.t2.micro	4 Actions
myrds-juju	Available	Instance	MySQL Community	us-east-1a	db.t3.micro	4 Actions
terraform-20231004173832127800000001	Available	Instance	MySQL Community	us-east-1a	db.t3.micro	4 Actions
terraform-20231004173844841500000001	Available	Instance	MySQL Community	us-east-1d	db.t3.micro	4 Actions

Windows PowerShell:

```
aws_db_instance.akilaRDS: Still creating... [5m10s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m20s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m30s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m40s elapsed]
aws_db_instance.akilaRDS: Still creating... [5m50s elapsed]
aws_db_instance.akilaRDS: Still creating... [6m0s elapsed]
aws_db_instance.akilaRDS: Still creating... [6m10s elapsed]
aws_db_instance.akilaRDS: Creation complete after 6m17s [id=db-KI0R532NIBQKFOA4QEXFWGDH0]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\akila\Desktop\BVC\terraform\lab-rds> terraform destroy
aws_db_instance.akilaRDS: Refreshing state... [id=db-KI0R532NIBQKFOA4QEXFWGDH0]

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

Terraform will perform the following actions:

# aws_db_instance.akilaRDS will be destroyed
- resource "aws_db_instance" "akilaRDS" {
  - address = "akila.czptxhzjxjrt.us-east-1.rds.amazonaws.com" -> null
  - allocated_storage = 5 -> null
  - apply_immediately = false -> null
  - arn = "arn:aws:rds:us-east-1:698668199773:db:akila" -> null
  - auto_minor_version_upgrade = true -> null
  - availability_zone = "us-east-1c" -> null
  - backup_retention_period = 0 -> null
  - backup_target = "region" -> null
  - backup_window = "04:28-04:58" -> null
  - ca_cert_identifier = "rds-ca-2019" -> null
  - copy_tags_to_snapshot = false -> null
  - customer_owned_ip_enabled = false -> null
  - db_name = "akiladb" -> null
  - db_subnet_group_name = "default" -> null
  - delete_automated_backups = true -> null
  - deletion_protection = false -> null
  - enabled_cloudwatch_logs_exports = [] -> null
  - endpoint = "akila.czptxhzjxjrt.us-east-1.rds.amazonaws.com:3306" -> null
  - engine = "mysql" -> null
  - engine_version = "5.7" -> null
}
```

Create AWS RDS using Terraform

The image shows a Windows PowerShell terminal window and the AWS Management Console. The terminal window displays the output of a Terraform command to destroy AWS RDS instances. The output shows that 1 instance was destroyed successfully. The AWS Management Console shows the 'Databases' page, which is empty, indicating that the instances have been successfully destroyed.

Windows PowerShell Output:

```
Plan: 0 to add, 0 to change, 1 to destroy.
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.akilaRDS: Destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 10s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 20s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 30s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 40s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 50s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 1m0s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 1m10s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 1m20s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 1m30s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 1m40s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 1m50s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 2m0s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 2m10s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 2m20s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 2m30s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 2m40s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 2m50s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 3m0s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 3m10s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 3m20s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 3m30s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 3m40s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 3m50s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 4m0s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 4m11s elapsed]
aws_db_instance.akilaRDS: Still destroying... [id=db-KIROR532NIBQKFOA4QEXFWGDHQ, 4m21s elapsed]
aws_db_instance.akilaRDS: Destruction complete after 4m24s

Destroy complete! Resources: 1 destroyed.
PS C:\Users\akila\Desktop\BVC\terraform\lab-rds>
```

AWS Management Console Output:

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#databases:

Amazon RDS

Introducing Aurora I/O-Optimized
Aurora's I/O-Optimized is a new cluster storage configuration that offers predictable pricing for all applications and improved price-performance, with up to 40% costs savings for I/O-intensive applications.

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](#)

Databases (4)

Group resources Modify Actions Restore from S3 Create database

Filter by databases

DB identifier	Status	Role	Engine	Region & AZ	Size	Actions	CP
example	Available	Instance	MySQL Community	us-east-1b	db.t2.micro	4 Actions	
myrds-juju	Available	Instance	MySQL Community	us-east-1a	db.t3.micro	4 Actions	
terraform-20231004173832127800000001	Available	Instance	MySQL Community	us-east-1a	db.t3.micro	4 Actions	
terraform-20231004173844841500000001	Available	Instance	MySQL Community	us-east-1d	db.t3.micro	4 Actions	

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Create AWS RDS using Terraform

The image displays two screenshots related to creating an AWS RDS instance using Terraform.

Top Screenshot: Terraform Configuration

The Terraform configuration file (`rds.tf`) defines an `aws_db_instance` resource named `akilaRDS`. The configuration includes the following attributes:

```
resource "aws_db_instance" "akilaRDS" {
  provider = "aws"
  region = "us-east-1"
  allocated_storage = 5
  db_name = "akiladb"
  engine = "mysql"
  engine_version = "5.7"
  instance_class = "db.t3.micro"
  identifier = "akila"
  username = "akila"
  password = "akilavasan"
  parameter_group_name = "default.mysql5.7"
  publicly_accessible = true
  skip_final_snapshot = true
}
```

Bottom Screenshot: AWS Management Console

The screenshot shows the AWS Management Console for the `us-east-1` region, specifically the `RDS > Databases` page. A notification banner at the top states: "Aurora's I/O-Optimized is a new cluster storage configuration that offers predictable pricing for all applications and improved price-performance, with up to 40% costs savings for I/O-intensive applications."

A message box prompts the user to "Consider creating a Blue/Green Deployment to minimize downtime during upgrades." Below this, the `Databases (5)` table lists the following instances:

DB identifier	Status	Role	Engine	Region & AZ	Size	Actions
akila	Available	Instance	MySQL Community	us-east-1c	db.t3.micro	-
example	Available	Instance	MySQL Community	us-east-1b	db.t2.micro	4 Actions
myrds-juju	Available	Instance	MySQL Community	us-east-1a	db.t3.micro	4 Actions
terraform-20231004173832127800000001	Available	Instance	MySQL Community	us-east-1a	db.t3.micro	4 Actions
terraform-20231004173844841500000001	Available	Instance	MySQL Community	us-east-1d	db.t3.micro	4 Actions