ec2-user@ip-172-31-25-29# project-202-abilalzengin:$ terraform plan

data.aws\_vpc.default\_vpc: Reading...

data.aws\_vpc.default\_vpc: Read complete after 0s [id=vpc-03eb016e3c5d8e3d4]

data.aws\_subnets.subnets: Reading...

data.aws\_subnets.subnets: Read complete after 0s [id=us-east-1]

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

symbols:

+ create

<= read (data resources)

Terraform will perform the following actions:

# data.template\_file.userdata will be read during apply

# (config refers to values not yet known)

<= data "template\_file" "userdata" {

+ id = (known after apply)

+ rendered = (known after apply)

+ template = <<-EOT

#! /bin/bash

yum update -y

yum install python3 -y

pip3 install flask

pip3 install flask\_mysql

cd /home/ec2-user && mkdir phonebook

echo ${rds-endpoint} > /home/ec2-user/phonebook/dbserver.endpoint

TOKEN="GHSAT0AAAAAABZ7LGITQ5XGSO46T6ERSOCIY3SNXVQ"

FOLDER="https://$TOKEN@raw.githubusercontent.com/abilalzengin/project-202-abilalzengin/main"

curl -s --create-dirs -o "/home/ec2-user/templates/index.html" -L "$FOLDER"/templates/index.html

curl -s --create-dirs -o "/home/ec2-user/templates/add-update.html" -L "$FOLDER"/templates/add-update.html

curl -s --create-dirs -o "/home/ec2-user/templates/delete.html" -L "$FOLDER"/templates/delete.html

curl -s --create-dirs -o "/home/ec2-user/app.py" -L "$FOLDER"/phonebook-app.py

python3 /home/ec2-user/app.py

EOT

+ vars = {

+ "rds-endpoint" = (known after apply)

}

}

# aws\_autoscaling\_attachment.bilal-attachment will be created

+ resource "aws\_autoscaling\_attachment" "bilal-attachment" {

+ autoscaling\_group\_name = (known after apply)

+ id = (known after apply)

+ lb\_target\_group\_arn = (known after apply)

}

# aws\_autoscaling\_group.bilal-asg will be created

+ resource "aws\_autoscaling\_group" "bilal-asg" {

+ arn = (known after apply)

+ availability\_zones = (known after apply)

+ default\_cooldown = (known after apply)

+ desired\_capacity = 2

+ force\_delete = true

+ force\_delete\_warm\_pool = false

+ health\_check\_grace\_period = 200

+ health\_check\_type = "ELB"

+ id = (known after apply)

+ max\_size = 3

+ metrics\_granularity = "1Minute"

+ min\_size = 1

+ name = "project-202-asg"

+ name\_prefix = (known after apply)

+ protect\_from\_scale\_in = false

+ service\_linked\_role\_arn = (known after apply)

+ vpc\_zone\_identifier = [

+ "subnet-06604349d7c79e76c",

+ "subnet-06653399011b0fd3e",

+ "subnet-09a979887365939a6",

+ "subnet-0e611e26ead9cc676",

+ "subnet-0fa7e978ecc058067",

+ "subnet-0fcedd72b85c19067",

]

+ wait\_for\_capacity\_timeout = "10m"

+ launch\_template {

+ id = (known after apply)

+ name = (known after apply)

+ version = "1"

}

+ tag {

+ key = "name"

+ propagate\_at\_launch = true

+ value = "bilal-1"

}

}

# aws\_autoscaling\_policy.scale\_down will be created

+ resource "aws\_autoscaling\_policy" "scale\_down" {

+ adjustment\_type = "ChangeInCapacity"

+ arn = (known after apply)

+ autoscaling\_group\_name = "project-202-asg"

+ cooldown = 120

+ enabled = true

+ id = (known after apply)

+ metric\_aggregation\_type = (known after apply)

+ name = "bilal-scale-down"

+ policy\_type = "SimpleScaling"

+ scaling\_adjustment = -1

}

# aws\_autoscaling\_policy.scale\_up will be created

+ resource "aws\_autoscaling\_policy" "scale\_up" {

+ adjustment\_type = "ChangeInCapacity"

+ arn = (known after apply)

+ autoscaling\_group\_name = "project-202-asg"

+ cooldown = 120

+ enabled = true

+ id = (known after apply)

+ metric\_aggregation\_type = (known after apply)

+ name = "bilal-scale-up"

+ policy\_type = "SimpleScaling"

+ scaling\_adjustment = 1

}

# aws\_cloudwatch\_metric\_alarm.bilal-scale-down will be created

+ resource "aws\_cloudwatch\_metric\_alarm" "bilal-scale-down" {

+ actions\_enabled = true

+ alarm\_actions = (known after apply)

+ alarm\_description = "scale-down-for-CPU"

+ alarm\_name = "bilal-scale\_down"

+ arn = (known after apply)

+ comparison\_operator = "LessThanOrEqualToThreshold"

+ dimensions = {

+ "autoscaling\_group\_name" = "project-202-asg"

}

+ evaluate\_low\_sample\_count\_percentiles = (known after apply)

+ evaluation\_periods = 3

+ id = (known after apply)

+ metric\_name = "CPUUtilization"

+ namespace = "AWS/EC2"

+ period = 60

+ statistic = "Average"

+ tags\_all = (known after apply)

+ threshold = 20

+ treat\_missing\_data = "missing"

}

# aws\_cloudwatch\_metric\_alarm.bilal-scale-up will be created

+ resource "aws\_cloudwatch\_metric\_alarm" "bilal-scale-up" {

+ actions\_enabled = true

+ alarm\_actions = (known after apply)

+ alarm\_description = "scale-up-for-CPU"

+ alarm\_name = "bilal-scale\_up"

+ arn = (known after apply)

+ comparison\_operator = "GreaterThanOrEqualToThreshold"

+ dimensions = {

+ "autoscaling\_group\_name" = "project-202-asg"

}

+ evaluate\_low\_sample\_count\_percentiles = (known after apply)

+ evaluation\_periods = 3

+ id = (known after apply)

+ metric\_name = "CPUUtilization"

+ namespace = "AWS/EC2"

+ period = 60

+ statistic = "Average"

+ tags\_all = (known after apply)

+ threshold = 55

+ treat\_missing\_data = "missing"

}

# aws\_db\_instance.bilal-db will be created

+ resource "aws\_db\_instance" "bilal-db" {

+ address = (known after apply)

+ allocated\_storage = 10

+ apply\_immediately = (known after apply)

+ arn = (known after apply)

+ auto\_minor\_version\_upgrade = true

+ availability\_zone = (known after apply)

+ backup\_retention\_period = (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 = "phonebook"

+ db\_subnet\_group\_name = (known after apply)

+ delete\_automated\_backups = true

+ endpoint = (known after apply)

+ engine = "mysql"

+ engine\_version = "8.0.25"

+ 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.t2.micro"

+ kms\_key\_id = (known after apply)

+ latest\_restorable\_time = (known after apply)

+ license\_model = (known after apply)

+ maintenance\_window = (known after apply)

+ monitoring\_interval = 0

+ monitoring\_role\_arn = (known after apply)

+ multi\_az = false

+ name = (known after apply)

+ nchar\_character\_set\_name = (known after apply)

+ network\_type = (known after apply)

+ option\_group\_name = (known after apply)

+ parameter\_group\_name = (known after apply)

+ password = (sensitive value)

+ performance\_insights\_enabled = false

+ performance\_insights\_kms\_key\_id = (known after apply)

+ performance\_insights\_retention\_period = (known after apply)

+ port = (known after apply)

+ publicly\_accessible = false

+ replica\_mode = (known after apply)

+ replicas = (known after apply)

+ resource\_id = (known after apply)

+ skip\_final\_snapshot = true

+ snapshot\_identifier = (known after apply)

+ status = (known after apply)

+ storage\_type = (known after apply)

+ tags\_all = (known after apply)

+ timezone = (known after apply)

+ username = "admin"

+ vpc\_security\_group\_ids = (known after apply)

}

# aws\_launch\_template.bilal-lt will be created

+ resource "aws\_launch\_template" "bilal-lt" {

+ arn = (known after apply)

+ default\_version = (known after apply)

+ id = (known after apply)

+ image\_id = "ami-09d3b3274b6c5d4aa"

+ instance\_type = "t2.micro"

+ key\_name = "firstkey"

+ latest\_version = (known after apply)

+ name = "project202-lt"

+ name\_prefix = (known after apply)

+ tags\_all = (known after apply)

+ user\_data = (known after apply)

+ vpc\_security\_group\_ids = (known after apply)

+ metadata\_options {

+ http\_endpoint = (known after apply)

+ http\_protocol\_ipv6 = (known after apply)

+ http\_put\_response\_hop\_limit = (known after apply)

+ http\_tokens = (known after apply)

+ instance\_metadata\_tags = (known after apply)

}

+ monitoring {

+ enabled = false

}

}

# aws\_lb.bilal-alb will be created

+ resource "aws\_lb" "bilal-alb" {

+ arn = (known after apply)

+ arn\_suffix = (known after apply)

+ desync\_mitigation\_mode = "defensive"

+ dns\_name = (known after apply)

+ drop\_invalid\_header\_fields = false

+ enable\_deletion\_protection = false

+ enable\_http2 = true

+ enable\_waf\_fail\_open = false

+ id = (known after apply)

+ idle\_timeout = 60

+ internal = (known after apply)

+ ip\_address\_type = (known after apply)

+ load\_balancer\_type = "application"

+ name = "proje-202-alb"

+ preserve\_host\_header = false

+ security\_groups = (known after apply)

+ subnets = [

+ "subnet-06604349d7c79e76c",

+ "subnet-06653399011b0fd3e",

+ "subnet-09a979887365939a6",

+ "subnet-0e611e26ead9cc676",

+ "subnet-0fa7e978ecc058067",

+ "subnet-0fcedd72b85c19067",

]

+ tags\_all = (known after apply)

+ vpc\_id = (known after apply)

+ zone\_id = (known after apply)

+ subnet\_mapping {

+ allocation\_id = (known after apply)

+ ipv6\_address = (known after apply)

+ outpost\_id = (known after apply)

+ private\_ipv4\_address = (known after apply)

+ subnet\_id = (known after apply)

}

}

# aws\_lb\_listener.bilal-listener will be created

+ resource "aws\_lb\_listener" "bilal-listener" {

+ arn = (known after apply)

+ id = (known after apply)

+ load\_balancer\_arn = (known after apply)

+ port = 80

+ protocol = "HTTP"

+ ssl\_policy = (known after apply)

+ tags\_all = (known after apply)

+ default\_action {

+ order = (known after apply)

+ target\_group\_arn = (known after apply)

+ type = "forward"

}

}

# aws\_lb\_target\_group.bilal-target will be created

+ resource "aws\_lb\_target\_group" "bilal-target" {

+ arn = (known after apply)

+ arn\_suffix = (known after apply)

+ connection\_termination = false

+ deregistration\_delay = "300"

+ id = (known after apply)

+ ip\_address\_type = (known after apply)

+ lambda\_multi\_value\_headers\_enabled = false

+ load\_balancing\_algorithm\_type = (known after apply)

+ name = "proje-202-target-group"

+ port = 80

+ preserve\_client\_ip = (known after apply)

+ protocol = "HTTP"

+ protocol\_version = (known after apply)

+ proxy\_protocol\_v2 = false

+ slow\_start = 0

+ tags\_all = (known after apply)

+ target\_type = "instance"

+ vpc\_id = "vpc-03eb016e3c5d8e3d4"

+ health\_check {

+ enabled = true

+ healthy\_threshold = 3

+ interval = 10

+ matcher = (known after apply)

+ path = (known after apply)

+ port = "traffic-port"

+ protocol = "HTTP"

+ timeout = (known after apply)

+ unhealthy\_threshold = 3

}

+ stickiness {

+ cookie\_duration = (known after apply)

+ cookie\_name = (known after apply)

+ enabled = (known after apply)

+ type = (known after apply)

}

+ target\_failover {

+ on\_deregistration = (known after apply)

+ on\_unhealthy = (known after apply)

}

}

# aws\_security\_group.ec2-sec-group will be created

+ resource "aws\_security\_group" "ec2-sec-group" {

+ arn = (known after apply)

+ description = "created-for-ec2"

+ egress = [

+ {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ description = ""

+ from\_port = 0

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "-1"

+ security\_groups = []

+ self = false

+ to\_port = 0

},

]

+ id = (known after apply)

+ ingress = [

+ {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ description = ""

+ from\_port = 22

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "tcp"

+ security\_groups = []

+ self = false

+ to\_port = 22

},

+ {

+ cidr\_blocks = []

+ description = ""

+ from\_port = 80

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "tcp"

+ security\_groups = (known after apply)

+ self = false

+ to\_port = 80

},

]

+ name = "ec2-sec-group"

+ name\_prefix = (known after apply)

+ owner\_id = (known after apply)

+ revoke\_rules\_on\_delete = false

+ tags = {

+ "Name" = "project-202-ec2-sec"

}

+ tags\_all = {

+ "Name" = "project-202-ec2-sec"

}

+ vpc\_id = "vpc-03eb016e3c5d8e3d4"

}

# aws\_security\_group.load-balancer-sec will be created

+ resource "aws\_security\_group" "load-balancer-sec" {

+ arn = (known after apply)

+ description = "for-lb"

+ egress = [

+ {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ description = ""

+ from\_port = 0

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "-1"

+ security\_groups = []

+ self = false

+ to\_port = 0

},

]

+ id = (known after apply)

+ ingress = [

+ {

+ cidr\_blocks = [

+ "0.0.0.0/0",

]

+ description = ""

+ from\_port = 80

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "tcp"

+ security\_groups = []

+ self = false

+ to\_port = 80

},

]

+ name = "load-balancer-sec"

+ name\_prefix = (known after apply)

+ owner\_id = (known after apply)

+ revoke\_rules\_on\_delete = false

+ tags = {

+ "Name" = "project-202-lb-sec"

}

+ tags\_all = {

+ "Name" = "project-202-lb-sec"

}

+ vpc\_id = "vpc-03eb016e3c5d8e3d4"

}

# aws\_security\_group.rds-sec will be created

+ resource "aws\_security\_group" "rds-sec" {

+ arn = (known after apply)

+ description = "for-rds"

+ egress = [

+ {

+ cidr\_blocks = []

+ description = ""

+ from\_port = 0

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "-1"

+ security\_groups = (known after apply)

+ self = false

+ to\_port = 0

},

]

+ id = (known after apply)

+ ingress = [

+ {

+ cidr\_blocks = []

+ description = ""

+ from\_port = 3306

+ ipv6\_cidr\_blocks = []

+ prefix\_list\_ids = []

+ protocol = "tcp"

+ security\_groups = (known after apply)

+ self = false

+ to\_port = 3306

},

]

+ name = "rds-sec-group"

+ name\_prefix = (known after apply)

+ owner\_id = (known after apply)

+ revoke\_rules\_on\_delete = false

+ tags = {

+ "Name" = "project-202-rds-sec"

}

+ tags\_all = {

+ "Name" = "project-202-rds-sec"

}

+ vpc\_id = "vpc-03eb016e3c5d8e3d4"

}

Plan: 14 to add, 0 to change, 0 to destroy.

Changes to Outputs:

+ rds\_endpoint = (known after apply)