Terraform 2

1. Launch an ASG in AWS and do Rolling Deployment with change in User Data in LaunchConfig using terraform

```
jay@Jay-Patel:learn3 $ tree
.
├── main.tf
├── terraform.tfstate
└── userdata.sh

0 directories, 3 files
jay@Jay-Patel:learn3 $
```

```
provider "aws" {
  profile    = "default"
  region     = "us-east-1"
}
resource "aws_launch_template" "LT1" {
  name = "LT-1"
  image_id = "ami-0d57c35552646fea8"
  instance_initiated_shutdown_behavior = "terminate"
  instance_type = "t2.micro"
  key_name = "Jay"
  vpc_security_group_ids = ["sg-00e4aad25f3645693"]
  user_data = filebase64("userdata.sh")
  lifecycle {
        create_before_destroy = true
        }
}

resource "aws_autoscaling_group" "Jay-ASG" {
  name                      = "Jay-ASG"
  max_size                  = 2
  min_size                  = 1
  health_check_grace_period = 300
  health_check_type         = "EC2"
  desired_capacity          = 1
  force_delete              = true
  availability_zones        = ["us-east-1c"]
  vpc_zone_identifier       = ["subnet-0fa3917512684d821"]

  launch_template {
    id      = aws_launch_template.LT1.id
    version = "$Latest"
  }

}
~
~
```

```
Plan: 2 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_launch_template.LT1: Creating...
aws_launch_template.LT1: Creation complete after 4s [id=lt-01110dbabe81de783]
aws_autoscaling_group.Jay-ASG: Creating...
aws_autoscaling_group.Jay-ASG: Still creating... [10s elapsed]
aws_autoscaling_group.Jay-ASG: Still creating... [20s elapsed]
aws_autoscaling_group.Jay-ASG: Still creating... [30s elapsed]
aws_autoscaling_group.Jay-ASG: Still creating... [40s elapsed]
aws_autoscaling_group.Jay-ASG: Still creating... [50s elapsed]
aws_autoscaling_group.Jay-ASG: Creation complete after 52s [id=Jay-ASG]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

Name	Launch Configuration /	Instances	Desired	Min	Max	Availability Zones	Default Cooldown	Health C
Jay-ASG	LT-1	1	1	1	2	us-east-1c	300	300 Termin

Auto Scaling Group: Jay-ASG

Details | Activity History | Scaling Policies | **Instances** | Monitoring | Notifications | Tags | Scheduled Actions | Lifecycle Hooks

Actions

Filter: Any Health Status | Any Lifecycle State | Filter instances...

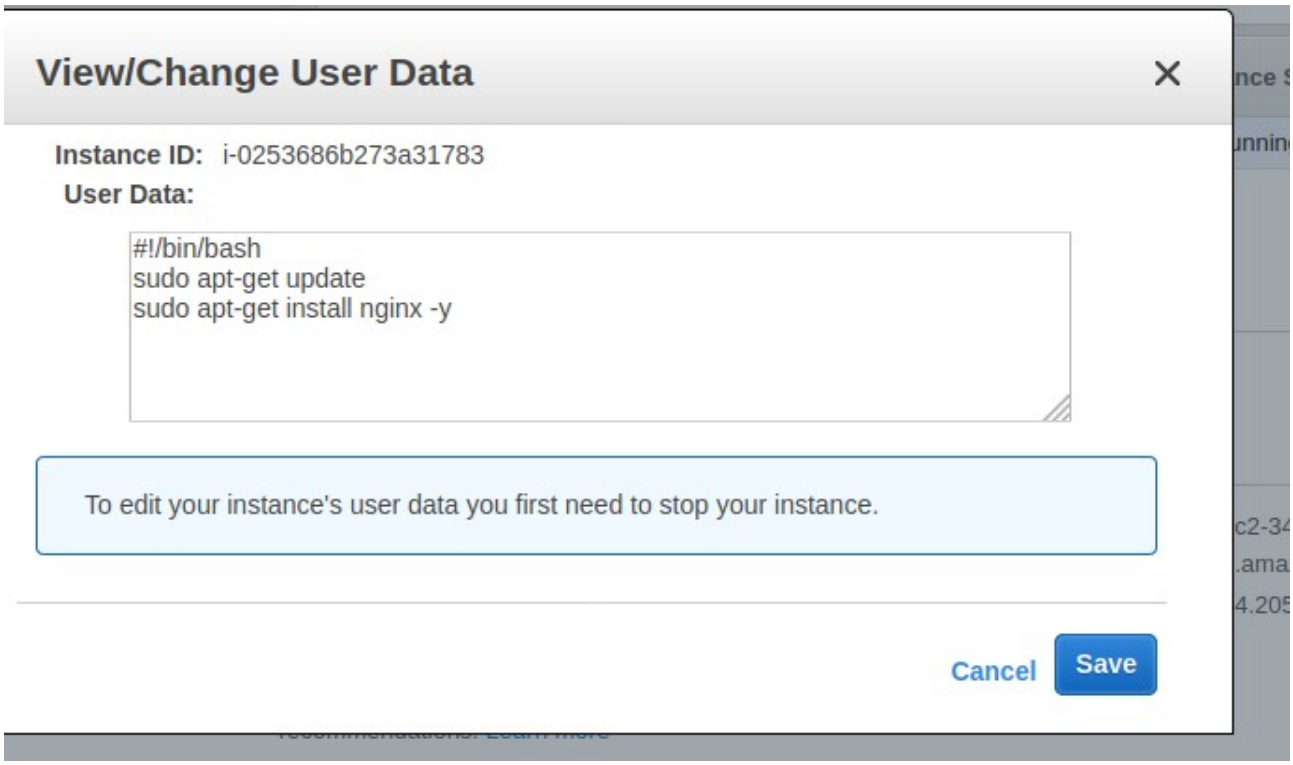
1 to 1 of 1 Instances

Instance ID	Lifecycle	Launch Configuration / Template	Availability Zone	Health Status	Protected from
i-0e6b3498f556ce494	InService	LT-1	us-east-1c	Healthy	

2. Deploy a sample nginx/tomcat/react service on it.

```
#!/bin/bash
sudo apt-get update
sudo apt-get install nginx -y
~
~
~
~
~
~
~
```

```
        ]
    }

Plan: 0 to add, 1 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_launch_template.LT1: Modifying... [id=lt-01110dbabe81de783]
aws_launch_template.LT1: Modifications complete after 3s [id=lt-01110dbabe81de783]

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



3. Attach a LB and create R53 endpoint pointing to lab, service should be accessible from the endpoint.

```
resource "aws_lb" "Jay-ALB" {
  name               = "Jay-ALB"
  internal           = false
  load_balancer_type = "application"
  security_groups    = ["sg-00e4aad25f3645693"]
  subnets            = ["subnet-0fa3917512684d821","subnet-0cf3ead13dc6
6bff7"]

}

resource "aws_lb_target_group" "Jay" {
  name     = "Jay"
  port     = 80
  protocol = "HTTP"
  vpc_id   = "vpc-fdfaf987"
}

resource "aws_lb_listener" "front_end" {
  load_balancer_arn = aws_lb.Jay-ALB.arn
  port              = "80"
  protocol          = "HTTP"

  default_action {
    type             = "forward"
    target_group_arn = aws_lb_target_group.Jay.arn
  }
}

resource "aws_autoscaling_attachment" "asg_attachment_Jay" {
  autoscaling_group_name = aws_autoscaling_group.Jay-ASG.name
  alb_target_group_arn   = aws_lb_target_group.Jay.arn
}

                                                    44,0-1        62%
```

```
resource "aws_autoscaling_attachment" "asg_attachment_Jay" {
  autoscaling_group_name = aws_autoscaling_group.Jay-ASG.name
  alb_target_group_arn   = aws_lb_target_group.Jay.arn
}


resource "aws_route53_zone" "private" {
  name = "example.com"

  vpc {
    vpc_id = "vpc-fdfaf987"
  }
}

resource "aws_route53_record" "www" {
  zone_id = aws_route53_zone.private.id
  name    = "www.example.com"
  type    = "A"
  alias {
    name                   = aws_lb.Jay-ALB.dns_name
    zone_id                = aws_lb.Jay-ALB.zone_id
    evaluate_target_health = true
  }
}

                                                    87,0-1        Bot
```

jay-alb-1213105581.us-east-1.elb.amazonaws.com

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Domain Name	Type	Record Set Count	Comment	Hosted Zone ID
example.com.	Private	3	Managed by Terraform	Z0305659TKQ5XPUNYTDR

4. Variablize all parameters and pass values as env.tfvars file.

```
Apply complete! Resources: 1 added, 1 changed, 1 destroyed.
jay@Jay-Patel:learn3 $ tree
.
├── env.tfvars
├── main.tf
├── terraform.tfstate
├── terraform.tfstate.backup
├── userdata.sh
└── variables.tf

0 directories, 6 files
jay@Jay-Patel:learn3 $
```

```
jay@Jay-Patel:learn3 $ vim main.tf
jay@Jay-Patel:learn3 $ terraform apply --var-file="env.tfvars"
aws_route53_zone.private: Refreshing state... [id=Z0305659TKQ5XPUNYTDR]
aws_lb_target_group.Jay: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-1:315002452909:targetgroup/Jay/14604366de0276e0]
aws_lb.Jay-ALB: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-1:315002452909:loadbalancer/app/Jay-ALB/8b429b51f52eb288]
aws_launch_template.LT1: Refreshing state... [id=lt-01110dbabe81de783]
aws_autoscaling_group.Jay-ASG: Refreshing state... [id=Jay-ASG]
aws_route53_record.www: Refreshing state... [id=Z0305659TKQ5XPUNYTDR_www.example.com_A]
aws_autoscaling_attachment.asg_attachment_Jay: Refreshing state... [id=Jay-ASG-20200419235250866300000001]
aws_lb_listener.front_end: Refreshing state... [id=arn:aws:elasticloadbalancing:us-east-1:315002452909:listener/app/Jay-ALB/8b429b51f52eb288/03e
73c56ed09a16a]

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  ~ update in-place
+/- create replacement and then destroy

Terraform will perform the following actions:

  # aws_autoscaling_group.Jay-ASG will be updated in-place
  ~ resource "aws_autoscaling_group" "Jay-ASG" {
        arn                     = "arn:aws:autoscaling:us-east-1:315002452909:autoScalingGroup:098f5dfd-2ede-4045-bf35-99e1f8d53c32:autoScalin
gGroupName/Jay-ASG"
        availability_zones      = [
```

```
          + http_tokens                 = (known after apply)
        }
    }

Plan: 1 to add, 1 to change, 1 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_launch_template.LT1: Creating...
aws_launch_template.LT1: Creation complete after 5s [id=lt-033ffd6207dba9c7b]
aws_autoscaling_group.Jay-ASG: Modifying... [id=Jay-ASG]
aws_autoscaling_group.Jay-ASG: Modifications complete after 5s [id=Jay-ASG]
aws_launch_template.LT1: Destroying... [id=lt-01110dbabe81de783]
aws_launch_template.LT1: Destruction complete after 2s

Apply complete! Resources: 1 added, 1 changed, 1 destroyed.
jay@Jay-Patel:learn3 $
```

```
aws_region="us-east-1"
template_name="Jay-LT-1"
ami="ami-0d57c35552646fea8"
instance_type="t2.micro"
key="Jay"
az="us-east-1c"
sg="sg-00e4aad25f3645693"
asg="Jay-ASG"
alb="Jay-ALB"
target_group="Jay"
private_zone="example.com"
record_name="www.example.com"
~
~
~
~
~
~
```

```
        type = string
}
variable "key" {
        description = "Name of the all the resources"
        type = string
}
variable "az" {
        description = "Name of the AMI"
        type = string
}
variable "sg" {
        description = "Count of the all the resources"
        type = string

}
variable  "asg" {
        description = "Name of the all the instance type"
        type = string
}

variable "alb" {
        description = "Name of the all the resources"
        type = string
}
variable "target_group" {
        description = "Name of the AMI"
        type = string
}
variable "private_zone" {
        description = "Count of the all the resources"
        type = string

}
variable "record_name" {
        description = "Name of the all the instance type"
        type = string
}
"variables.tf" 52L, 1187C
```

5. Create ASG from Launch Template and use a mix of on demand and on spot instance type in the ASG. Instance Type for On Demand and Spot should be diferent.
Enable Spot Feature to use multiple instance type if requested instance type is not available.

```
  resource "aws_spot_instance_request" "cheap_worker" {
  ami           = "ami-1234"
  spot_price    = "0.03"
  instance_type = "c4.xlarge"

  tags = {
    Name = "CheapWorker"
     }
  }
instance_types = [
{
        instance_type="t2.micro",
        weighted_capacity = 1 ,

},
{
        instance_type="t3.micro",
        weighted_capacity = 1 ,

},
{
        instance_type="t2.samll",
        weighted_capacity = 2 ,
},
]
}
```

```
resource "aws_spot_fleet_request" "cheap_compute" {
  iam_fleet_role      = "arn:aws:iam::12345678:role/spot-fleet"
  spot_price          = "0.03"
  allocation_strategy = "diversified"
  target_capacity     = 6
  valid_until         = "2019-11-04T20:44:20Z"

  launch_specification {
    instance_type            = "m4.10xlarge"
    ami                      = "ami-1234"
    spot_price               = "2.793"
    placement_tenancy        = "dedicated"
    iam_instance_profile_arn = "${aws_iam_instance_profile.example.arn}"
  }

  launch_specification {
    instance_type            = "m4.4xlarge"
    ami                      = "ami-5678"
    key_name                 = "my-key"
    spot_price               = "1.117"
    iam_instance_profile_arn = "${aws_iam_instance_profile.example.arn}"
    availability_zone        = "us-west-1a"
    subnet_id                = "subnet-1234"
    weighted_capacity        = 35

    root_block_device {
      volume_size = "300"
      volume_type = "gp2"
    }

    tags = {
      Name = "spot-fleet-example"
    }
  }
}
-- INSERT --
```