```
provider "aws" {
  region = "us-west-2"
}

# Define an Application Load Balancer
resource "aws_lb" "example" {
  name               = "example-lb"
  internal           = false
  load_balancer_type = "application"
  security_groups    = [aws_security_group.lb_sg.id]
  subnets            = [aws_subnet.example1.id, aws_subnet.example2.id]

  enable_deletion_protection = false
}

# Define a target group for the ALB
resource "aws_lb_target_group" "example" {
  name     = "example-tg"
  port     = 80
  protocol = "HTTP"
  vpc_id   = aws_vpc.example.id

  health_check {
    enabled = true
    path    = "/"
    protocol = "HTTP"
```

```
    matcher = "200"
  }
}

# Launch Template
resource "aws_launch_template" "example" {
  name_prefix   = "example-lt-"
  image_id      = "ami-1234567890abcdef0"  # Replace with a valid AMI ID
  instance_type = "t2.micro"

  # Add other configurations as necessary, such as key name, security groups, etc.
}

# Auto Scaling Group
resource "aws_autoscaling_group" "example" {
  launch_template {
    id      = aws_launch_template.example.id
    version = "$Latest"
  }

  min_size         = 1
  max_size         = 10
  desired_capacity = 2
  vpc_zone_identifier = [aws_subnet.example1.id, aws_subnet.example2.id]
  target_group_arns   = [aws_lb_target_group.example.arn]
}
```

```
# CloudWatch Alarm for Scaling Out
resource "aws_cloudwatch_metric_alarm" "high_cpu" {
  alarm_name          = "high-cpu-usage"
  comparison_operator = "GreaterThanThreshold"
  evaluation_periods  = 2
  metric_name         = "CPUUtilization"
  namespace           = "AWS/EC2"
  period              = 300
  statistic           = "Average"
  threshold           = 75
  alarm_actions       = [aws_autoscaling_policy.scale_out.arn]

  dimensions = {
    AutoScalingGroupName = aws_autoscaling_group.example.name
  }
}

# Scaling Policy for Out
resource "aws_autoscaling_policy" "scale_out" {
  name                   = "scale-out"
  scaling_adjustment     = 1
  adjustment_type        = "ChangeInCapacity"
  autoscaling_group_name = aws_autoscaling_group.example.name
  cooldown               = 300
}
```

```
# CloudWatch Alarm for Scaling In
resource "aws_cloudwatch_metric_alarm" "low_cpu" {
  alarm_name          = "low-cpu-usage"
  comparison_operator = "LessThanThreshold"
  evaluation_periods  = 2
  metric_name         = "CPUUtilization"
  namespace           = "AWS/EC2"
  period              = 300
  statistic           = "Average"
  threshold           = 25
  alarm_actions       = [aws_autoscaling_policy.scale_in.arn]

  dimensions = {
    AutoScalingGroupName = aws_autoscaling_group.example.name
  }
}

# Scaling Policy for In
resource "aws_autoscaling_policy" "scale_in" {
  name                   = "scale-in"
  scaling_adjustment     = -1
  adjustment_type        = "ChangeInCapacity"
  autoscaling_group_name = aws_autoscaling_group.example.name
  cooldown               = 300
}
```