```
provider "aws" {
  region     = "us-east-1"
  secret_key = "h+ZAe2PIoPUDro01A/yVm3L9FQMKOeCjkwPQN93M"
  access_key = "your access key"
  token      = "Your token"
}



# S3 Bucket
resource "aws_s3_bucket" "s3urvashi" {
  bucket = "s3urvashi-unique-name"  # Change this to a unique bucket name
}

# S3 Bucket Versioning
resource "aws_s3_bucket_versioning" "versioning" {
  bucket = aws_s3_bucket.s3urvashi.id

  versioning_configuration {
    status = "Enabled"  # Enable versioning
  }
}

# Block Public Access Settings
resource "aws_s3_bucket_public_access_block" "s3urvashi_block_public_access" {
  bucket                  = aws_s3_bucket.s3urvashi.id
```

```
  block_public_acls       = true
  ignore_public_acls      = true
  block_public_policy     = true
  restrict_public_buckets = true
}

# SQS Queue
resource "aws_sqs_queue" "sqsurvashi" {
  name = "sqsurvashi"  # Name of the SQS queue
}

# IAM Role for Lambda execution
resource "aws_iam_role" "lambda_exec" {
  name               = "lambda_exec_role"
  assume_role_policy = jsonencode({
    Version   = "2012-10-17",
    Statement = [{
      Action = "sts:AssumeRole",
      Effect = "Allow",
      Principal = {
        Service = "lambda.amazonaws.com"
      }
    }]
  })
}
```

```
# IAM Role Policy for Lambda (grant permissions to interact with S3 and SQS)
resource "aws_iam_role_policy" "lambda_exec_policy" {
  name = "lambda_exec_policy"
  role = aws_iam_role.lambda_exec.id

  policy = jsonencode({
    Version   = "2012-10-17",
    Statement = [
      {
        Action   = ["sqs:SendMessage"],
        Effect   = "Allow",
        Resource = aws_sqs_queue.sqsurvashi.arn
      },
      {
        Action   = ["s3:GetObject"],
        Effect   = "Allow",
        Resource = "${aws_s3_bucket.s3urvashi.arn}/*"
      },
      {
        Action   = ["s3:PutObject", "s3:DeleteObject"],  # Grant permissions to put and delete objects
        Effect   = "Allow",
        Resource = "${aws_s3_bucket.s3urvashi.arn}/*"
      }
    ]
  })
}
```

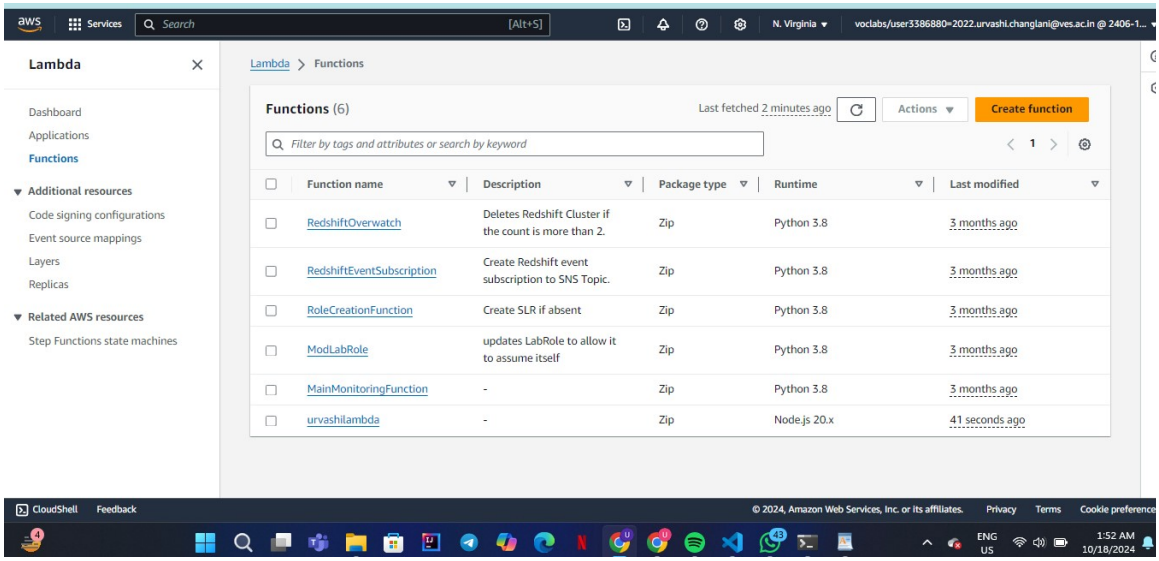
```
# Lambda Function
resource "aws_lambda_function" "urvashilambda" {
  function_name = "urvashilambda"         # Name of the Lambda function
  role          = aws_iam_role.lambda_exec.arn  # Role assigned to Lambda
  handler       = "index.handler"         # Adjust the handler as needed
  runtime       = "nodejs14.x"            # Specify the runtime environment
  timeout       = 10                      # Set timeout in seconds
  filename      = "lambda.zip"            # Path to the Lambda zip file

  environment {
    variables = {
      QUEUE_URL = aws_sqs_queue.sqsurvashi.id # Pass the SQS queue URL to Lambda
    }
  }
}

# S3 Bucket Notification to trigger Lambda on object creation
resource "aws_s3_bucket_notification" "s3_notification" {
  bucket = aws_s3_bucket.s3urvashi.id

  lambda_function {
    lambda_function_arn = aws_lambda_function.urvashilambda.arn
    events              = ["s3:ObjectCreated:*"]  # Trigger on object creation events
  }
}
```

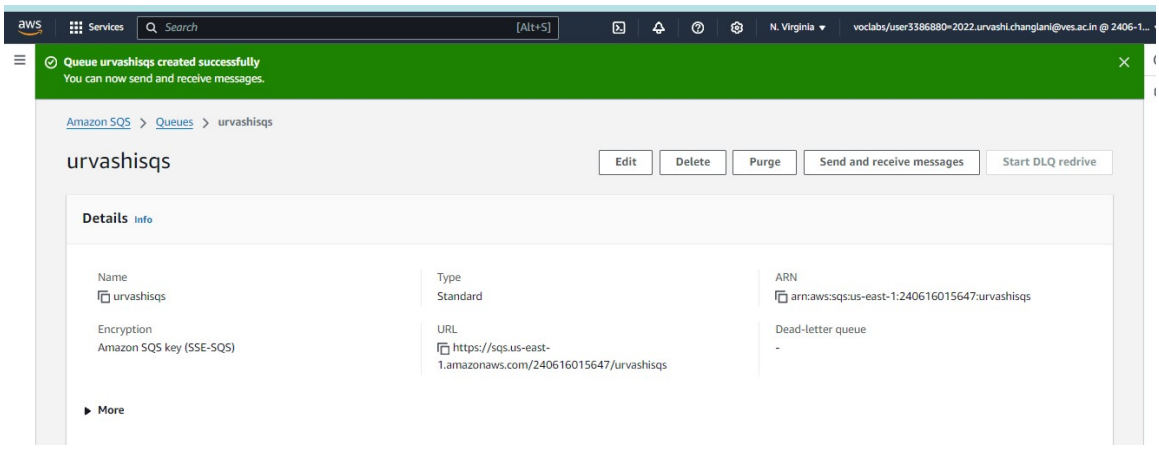
```
# Lambda Permission for S3 to invoke the Lambda function
resource "aws_lambda_permission" "allow_s3" {
 statement_id  = "AllowS3InvokeLambda"      # Unique statement ID
 action        = "lambda:InvokeFunction"    # Specify the action
 function_name = aws_lambda_function.urvashilambda.function_name
 principal     = "s3.amazonaws.com"         # The principal that can invoke the function
 source_arn    = aws_s3_bucket.s3urvashi.arn # Restrict the source ARN to the S3 bucket
}
```

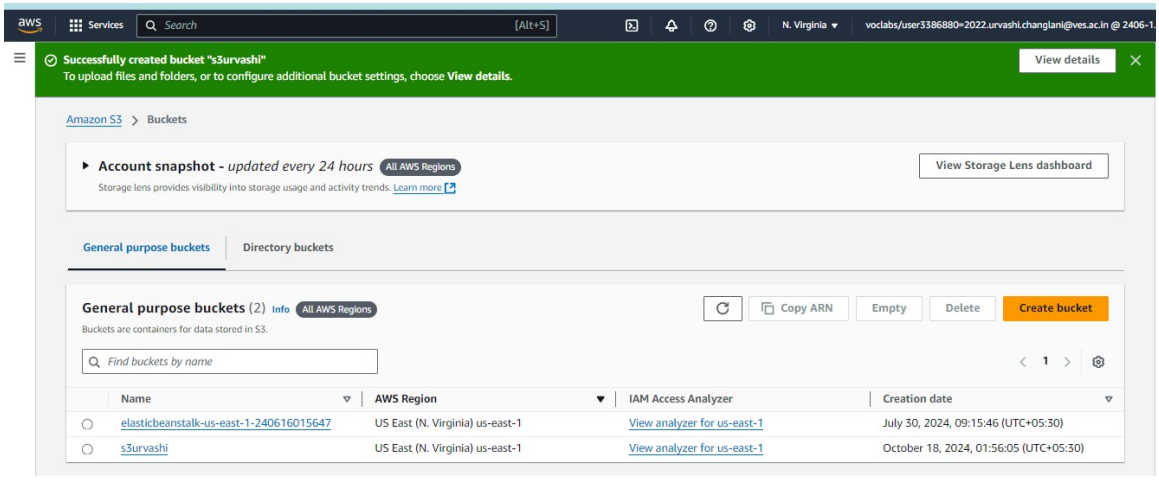
Implementation:

1. Creating Lambda Function

2. Creating Sqs Queue







Performing Terraform commands

1. Terraform init

2. Terraform plan

3. Terraform apply

4.Terraform destroy

```
C:\Users\chang\terraform-aws-s3-sqs-lambda>terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.72.1

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.
```

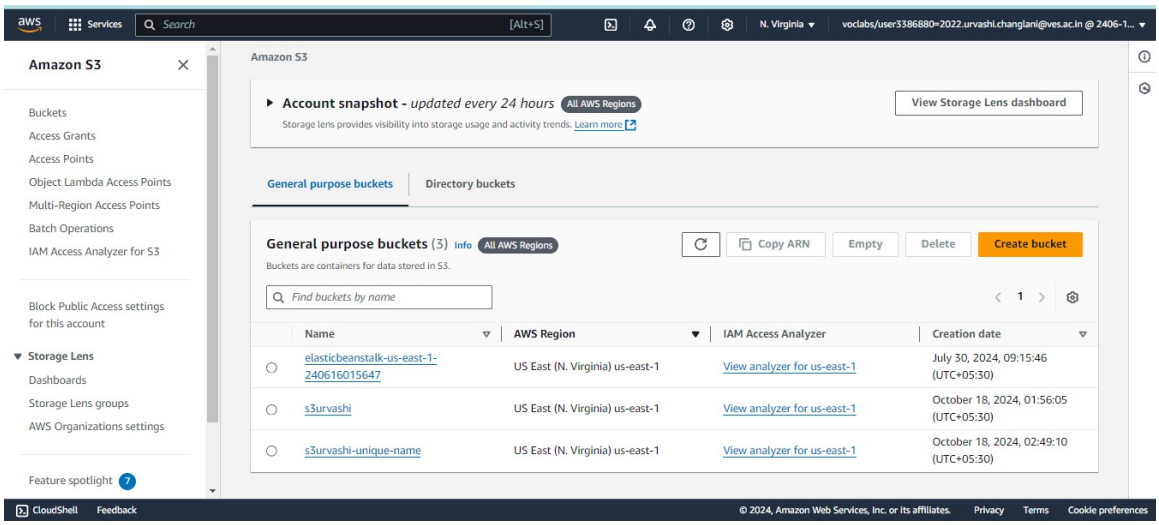
Command Prompt

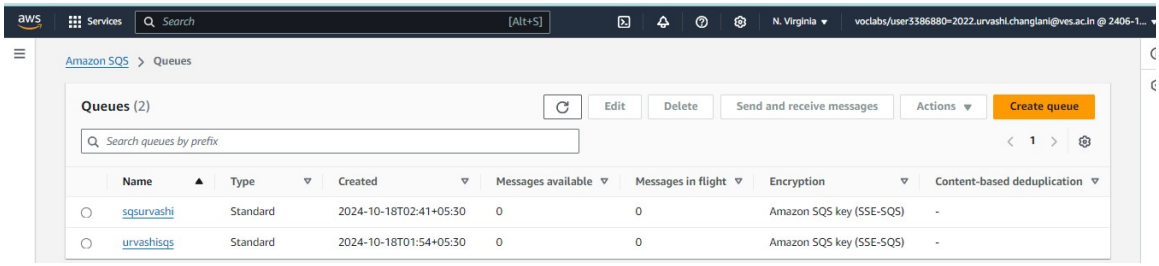
```
C:\Users\chang\terraform-aws-s3-sqs-lambda>terraform plan

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_iam_role.lambda_exec will be created
  + resource "aws_iam_role" "lambda_exec" {
      + arn                   = (known after apply)
      + assume_role_policy    = jsonencode(
            {
              + Statement = [
                  + {
                      + Action    = "sts:AssumeRole"
                      + Effect    = "Allow"
                      + Principal = {
                          + Service = "lambda.amazonaws.com"
                        }
                    },
                ]
              + Version   = "2012-10-17"
            }
        )
      + create_date           = (known after apply)
      + force_detach_policies = false
      + id                    = (known after apply)
      + managed_policy_arns   = (known after apply)
      + max_session_duration  = 3600
      + name                  = "lambda_exec_role"
      + name_prefix           = (known after apply)
      + path                  = "/"
      + tags_all              = (known after apply)
```

```
Command Prompt
      + name_prefix                      = (known after apply)
      + policy                           = (known after apply)
      + receive_wait_time_seconds        = 0
      + redrive_allow_policy             = (known after apply)
      + redrive_policy                   = (known after apply)
      + sqs_managed_sse_enabled          = (known after apply)
      + tags_all                         = (known after apply)
      + url                              = (known after apply)
      + visibility_timeout_seconds       = 30
    }

Plan: 8 to add, 0 to change, 0 to destroy.

  Warning: Argument is deprecated

    with aws_s3_bucket.s3urvashi,
    on main.tf line 9, in resource "aws_s3_bucket" "s3urvashi":
     9: resource "aws_s3_bucket" "s3urvashi" {

  Use the aws_s3_bucket_versioning resource instead

  (and one more similar warning elsewhere)
```

```
C:\Users\chang\terraform-aws-s3-sqs-lambda>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_iam_role.lambda_exec will be created
  + resource "aws_iam_role" "lambda_exec" {
      + arn                 = (known after apply)
      + assume_role_policy  = jsonencode(
            {
              + Statement = [
                  + {
                      + Action    = "sts:AssumeRole"
                      + Effect    = "Allow"
                      + Principal = {
                          + Service = "lambda.amazonaws.com"
                        }
                    },
                ]
              + Version   = "2012-10-17"
            }
        )
      + create_date           = (known after apply)
      + force_detach_policies = false
      + id                    = (known after apply)
      + managed_policy_arns   = (known after apply)
      + max_session_duration  = 3600
      + name                  = "lambda_exec_role"
      + name_prefix           = (known after apply)
      + path                  = "/"
      + tags_all              = (known after apply)
      + unique_id             = (known after apply)
```

```
Plan: 7 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_s3_bucket_public_access_block.s3urvashi_block_public_access: Creating...
aws_s3_bucket_versioning.versioning: Creating...
aws_iam_role.lambda_exec: Creating...
aws_s3_bucket_public_access_block.s3urvashi_block_public_access: Creation complete after 2s [id=s3urvashi-unique-name]
aws_s3_bucket_versioning.versioning: Creation complete after 3s [id=s3urvashi-unique-name]
```






```
C:\Users\chang\terraform-aws-s3-sqs-lambda>terraform destroy
aws_s3_bucket.s3urvashi: Refreshing state... [id=s3urvashi-unique-name]

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

Terraform will perform the following actions:

  # aws_s3_bucket.s3urvashi will be destroyed
  - resource "aws_s3_bucket" "s3urvashi" {
      - arn                         = "arn:aws:s3:::s3urvashi-unique-name" -> null
      - bucket                      = "s3urvashi-unique-name" -> null
      - bucket_domain_name          = "s3urvashi-unique-name.s3.amazonaws.com" -> null
      - bucket_regional_domain_name = "s3urvashi-unique-name.s3.us-east-1.amazonaws.com" -> null
      - force_destroy               = false -> null
      - hosted_zone_id              = "Z3AQBSTGFYJSTF" -> null
      - id                          = "s3urvashi-unique-name" -> null
      - object_lock_enabled         = false -> null
      - region                      = "us-east-1" -> null
      - request_payer               = "BucketOwner" -> null
      - tags                        = {} -> null
      - tags_all                    = {} -> null
        # (3 unchanged attributes hidden)

      - grant {
          - id          = "ea4f5edc59239e73831c849a0766f702ceac6c8e425828059a8cc4e4b677830a" -> null
          - permissions = [
              - "FULL_CONTROL",
            ] -> null
          - type        = "CanonicalUser" -> null
            # (1 unchanged attribute hidden)
        }
```