

Home / AWS / Guided Lab / Securely access S3 images using Amazon CloudFront - Terraform

# Securely access S3 images using Amazon CloudFront - Terraform

Level: Intermediate

Amazon S3    Amazon CloudFront    Amazon Web Services    Terraform



0h 59m 47s left



End Lab

Open Console



K ▾

User Name ⓘ

Whiz\_User\_80425.58941718



Password ⓘ

2bd2a02f-a202-4b5e-a397-9d57b6e7968f



Access Key ⓘ

AKIAZVVUXNLP62A5C5PJ



Secret Key ⓘ

KIHl435C3TQhxmJ+yqT70FS/Qq/OdqBG9y8qzS0Z






Lab Resources

- 1. [Download Zip](#)

Support Documents


- 1. [FAQs and Troubleshooting](#)

## Need help?

-  How to use Hands on Lab
-  Troubleshooting Lab
-  FAQs

[Submit Feedback](#)[Share](#)[Lab Overview](#)[Lab Steps](#)[Lab Validation](#)

 Cloud Architect, Cloud Developer, Cloud Network Engineer

 Storage, Networking, Infrastructure

# Lab Steps

## Task 1: Sign in to AWS Management Console

1. Click on the **Open Console** button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,
  - Leave the Account ID as default. Never edit/remove the 12 digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
  - Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign in** button
3. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1**.

## Task 2: Setup Visual Studio Code

In this task, we are going to set up Visual Studio Code as the code editor for working with Terraform. It provides instructions on opening Visual Studio Code, navigating to the desired folder, and preparing the terminal for executing Terraform commands.

1. Open the visual studio code.
2. If you have already installed and using Visual studio code, open a new window.

3. A new window will open a new file and release notes page (only if you have installed or updated Visual Studio Code recently). Close the Release notes tab.
4. Open Terminal by selecting View from the Menu bar and choose Terminal.
5. It may take up to 2 minutes to open the terminal window.
6. Once the terminal is ready, let us navigate to the Desktop.

```
cd Desktop
```



7. Create a new folder by running the below command.

```
mkdir task_10096
```



8. Change your present working directory to use the newly created folder by running the below command:

```
cd task_10096
```



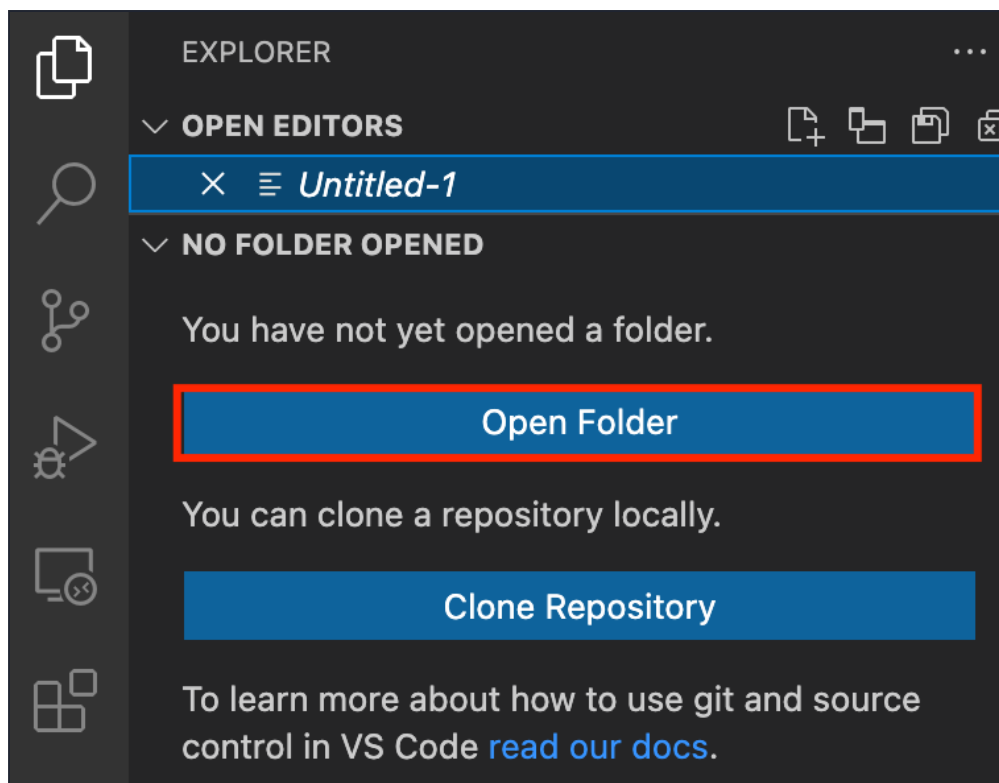
9. Get the location of the present working directory by running the below command:

```
pwd
```



10. Note down the location, as you will open the same in the next steps.
11. Now click on the first icon Explorer present on the left sidebar.
12. Click on the button called Open folder and navigate to the location of folder **task\_10096**





13. (Optional) Click on Authorize button for allowing Visual Studio Code to use the **task\_10096** folder. This will only be asked when you have been using Visual Studio Code for a while as you are allowing a new folder to be accessed by VSC.

14. Visual Studio Code is now ready to use.

### Task 3: Create a variable file

In this task, you will create variable files where you will declare all the global variables with a short description and a default value.

1. To create a variable file, expand the folder **task\_10096** and click on the **New File** icon to add the file.
2. Name the file as **variables.tf** and press **Enter** to save it.
3. **Note:** Don't change the location of the new file, keep it default, i.e. inside the **task\_10096** folder.
4. Paste the below contents in **variables.tf** file.

```
variable "access_key" {  
    description = "Access key to AWS console"  
}  
variable "secret_key" {  
    description = "Secret key to AWS console"  
}  
variable "region" {  
    description = "AWS region"  
}
```



5. In the above content, you are declaring a variable called, `access_key`, `secret_key`, and `region` with a short description of all 3.
6. After pasting the above contents, save the file by pressing **ctrl + S**.
7. Now expand the folder **task\_10096** and click on the **New File** icon to add the file.
8. Name the file as **terraform.tfvars** and press **Enter** to save it.
9. Paste the below content into the **terraform.tfvars** file.

```
region = "us-east-1"
access_key = "<YOUR AWS CONSOLE ACCESS ID>"
secret_key = "<YOUR AWS CONSOLE SECRET KEY>"
```



10. In the above code, you are defining the dynamic values of variables declared earlier.
11. Replace the values of `access_key` and `secret_key` by copying from the lab page.
12. After replacing the values of `access_key` and `secret_key`, save the file by pressing **Ctrl + S**.



```
terraform.tfvars > secret_key
1  region = "us-east-1"
2
3  access_key = "[REDACTED]"
4
5  secret_key = "[REDACTED]"
```

#### Task 4: Create a S3 bucket in main.tf file

In this task, you will create a **main.tf** file where you will add details of the provider and resources.

1. To create a **main.tf** file, expand the folder **task\_10096** and click on the **New File** icon to add the file.
2. Name the file as **main.tf** and press **Enter** to save it.
3. Paste the below content into the **main.tf** file.

```
provider "aws" {
  region      = "${var.region}"
  access_key  = "${var.access_key}"
```



```
secret_key = "${var.secret_key}"
}
```

4. In the above code, you are defining the provider as aws.
5. Next, we want to tell Terraform to create a S3 bucket.
6. Since the bucket name must be unique across all existing bucket names in Amazon S3, let us also creating a random string and an S3 bucket with public access block settings.
7. Paste the below content into the **main.tf** file after the provider.

```
##### Creating a Random String #####
resource "random_string" "random" {
  length = 6
  special = false
  upper = false
}

##### Creating an S3 Bucket #####
resource "aws_s3_bucket" "bucket" {
  bucket = "whizbucket-${random_string.random.result}"
  force_destroy = true
}

##### Creating Bucket Public Access Block #####
resource "aws_s3_bucket_public_access_block" "public_access_block" {
  bucket = aws_s3_bucket.bucket.id

  block_public_acls      = false
  block_public_policy    = false
  ignore_public_acls     = false
  restrict_public_buckets = false
}
```



```

provider "aws" {
  region      = "${var.region}"
  access_key  = "${var.access_key}"
  secret_key  = "${var.secret_key}"
}

##### Creating a Random String #####
resource "random_string" "random" {
  length = 6
  special = false
  upper = false
}

##### Creating an S3 Bucket #####
resource "aws_s3_bucket" "bucket" {
  bucket = "whizbucket-${random_string.random.result}"
  force_destroy = true
}

##### Creating Bucket Public Access Block #####
resource "aws_s3_bucket_public_access_block" "public_access_block" {
  bucket = aws_s3_bucket.bucket.id

  block_public_acls       = false
  block_public_policy     = false
  ignore_public_acls     = false
  restrict_public_buckets = false
}

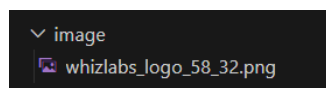
```

8. Save the file by pressing **Ctrl + S**.

## Task 5: Upload an image file in s3 bucket in main.tf file

In this task we are going to upload an image file in S3 bucket

1. Firstly you will create a new folder to place the image file.
2. To create a folder, click on the **Create folder icon** present on the left side panel under your folder called **task\_10096**
3. Name the folder as **image**
4. Now click on this link to [download the link](#) file to your local, extract the zip file.
5. Place the **whizlabs\_logo\_58\_32.png** image file inside the newly created folder called **image**



6. Now Add another block just below the s3 bucket creation code in **main.tf** file, this block of code will upload the image files present under image folder to the S3 bucket.



```
# Upload an object
resource "aws_s3_object" "object" {
  bucket = aws_s3_bucket.bucket.id
  key     = "whizlabs_logo_58_32.png"
  source  = "image/whizlabs_logo_58_32.png"
  etag    = filemd5("image/whizlabs_logo_58_32.png")
}
```

7. Save the file by pressing **Ctrl + S**.

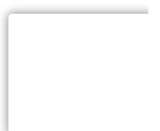
## Task 6: Create a S3 bucket policy in main.tf file

In this task we are going to create a S3 bucket policy to make the objects publicly accessible

1. To Create a bucket policy add another block just below the upload object code in **main.tf** file



```
#Creating Bucket Policy
resource "aws_s3_bucket_policy" "public_read_access" {
  bucket = aws_s3_bucket.bucket.id
  policy = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:list*",
        "s3:get*",
        "s3:putobject"
      ],
      "Resource": [
        "${aws_s3_bucket.bucket.arn}",
        "${aws_s3_bucket.bucket.arn}/*"
      ]
    }
  ]
}
EOF
}
```





```
# Upload an object
resource "aws_s3_object" "object" {
  bucket = aws_s3_bucket.bucket.id
  key    = "whizlabs.png"
  source = "image/whizlabs.png"
  etag   = filemd5("image/whizlabs.png")
}

#Creating Bucket Policy
resource "aws_s3_bucket_policy" "public_read_access" {
  bucket = aws_s3_bucket.bucket.id
  policy = <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:list*",
        "s3:get*",
        "s3:putobject"
      ],
      "Resource": [
        "${aws_s3_bucket.bucket.arn}",
        "${aws_s3_bucket.bucket.arn}/*"
      ]
    }
  ]
}
  ]
}
EOF
}
```

2. Save the file by pressing **Ctrl + S**.

## Task 7: Create a CloudFront Distribution in main.tf file

In this task, you will create a CloudFront Distribution pointed to Amazon S3 bucket origin

1. To create a CloudFront Distribution add another block just below the bucket policy creation code in **main.tf** file

```
# Create Cloudfront distribution
locals {
  s3_origin_id = "myS3Origin"
}

resource "aws_cloudfront_distribution" "s3_distribution" {
  origin {
```



```

    domain_name = aws_s3_bucket.bucket.bucket_regional_domain_name
    origin_id    = local.s3_origin_id
  }
  enabled          = true
  is_ipv6_enabled = true
  comment          = "whiz-cloudfront"
  default_cache_behavior {
    allowed_methods =
["DELETE", "GET", "HEAD", "OPTIONS", "PATCH", "POST", "PUT"]
    cached_methods = ["GET", "HEAD"]
    target_origin_id = local.s3_origin_id
    forwarded_values {
      query_string = false
      cookies {
        forward = "none"
      }
    }
  }
  viewer_protocol_policy = "allow-all"
  min_ttl                = 0
  default_ttl            = 10
  max_ttl                = 20
}
price_class = "PriceClass_200"
restrictions {
  geo_restriction {
    restriction_type = "none"
  }
}
viewer_certificate {
  cloudfront_default_certificate = true
}
}

```

2. Save the file by pressing **Ctrl + S**.

## Task 8: Create an Output file

In this task, you will create an **output.tf** file where you will add details of the provider and resources.

1. To create an **output.tf** file, expand the folder **task\_10096** and click on the **New File** icon to add the file.
2. Name the file as **output.tf** and press **Enter** to save it.
3. Paste the below content into the **output.tf** file.

```

output "s3-bucket-name" {
  value = aws_s3_bucket.bucket.id
}
output "cloudfront_domain_name" {
  value = aws_cloudfront_distribution.s3_distribution.domain_name
}

```



```
}
```

4. In the above code, we will extract the s3 bucket name and cloudfront domain name to confirm that they are created.

## Task 9: Confirm the installation of Terraform by checking the version

In this task, we are going to ensure that Terraform is properly installed on the local machine.

1. In the Visual Studio Code, open Terminal by selecting **View** from the Menu bar and choose **Terminal**.
2. If you are not in the newly created folder change your present working directory by running the below command.

```
cd task_10096
```



3. To confirm the installation of Terraform, run the below command to check the version:

```
terraform version
```



4. If you are getting output as command not found: terraform, this means that terraform is not installed on your system, To install terraform follow the official guide link provided in the Prerequisite section above.

## Task 10: Apply terraform configurations

In this task, we are going to apply the Terraform configurations and create the defined resources.

1. Initialize Terraform by running the below command,

```
terraform init
```



**Note:** terraform init will check for all the plugin dependencies and download them if required, this will be used for creating a deployment plan



```
- Finding latest version of hashicorp/aws...
- Installing hashicorp/random v3.4.3...
- Installed hashicorp/random v3.4.3 (signed by HashiCorp)
- Installing hashicorp/aws v4.39.0...
- Installed hashicorp/aws v4.39.0 (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.

2. To generate the action plans run the below command,

```
terraform plan
```



3. To create all the resources declared in main.tf configuration file, run the below command,

```
terraform apply
```



4. Approve the creation of all the resources by entering **yes**.

```
Plan: 6 to add, 0 to change, 0 to destroy.
```

```
Changes to Outputs:
```

```
+ cloudfront_domain_name = (known after apply)
+ s3-bucket-name          = (known after apply)
```

```
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
```

5. Note: This process will take around 5-10 minutes.

6. Id's of all the resources created by terraform will be visible there.



Apply complete! Resources: 6 added, 0 changed, 0 destroyed.

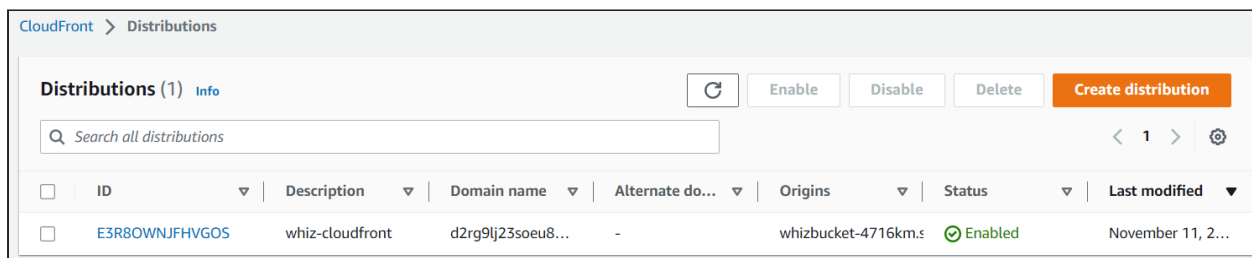
### Outputs:

```
cloudfront_domain_name = "d2g1ody44rf8xb.cloudfront.net"
s3-bucket-name = "whizbucket-qcrxs3"
```

## Task 11: Check the resources in AWS Console

In this task, we are going to verify that the resources have been successfully created in the AWS Management Console.

1. Make sure you are in the **US East (N. Virginia) us-east-1** Region.
2. Navigate to **S3** by clicking on **Services** on the top, then click on **S3** in the **Storage** section.
3. Click on **Buckets** in left navigation menu and Select the bucket **Whizlabs<RANDOM\_STRING>** You can see object uploaded in bucket
4. To Check Cloudfront Distribution Creation Select **Cloudfront** under **Networking and Content Delivery** section.
5. You can see Cloudfront Distribution Created



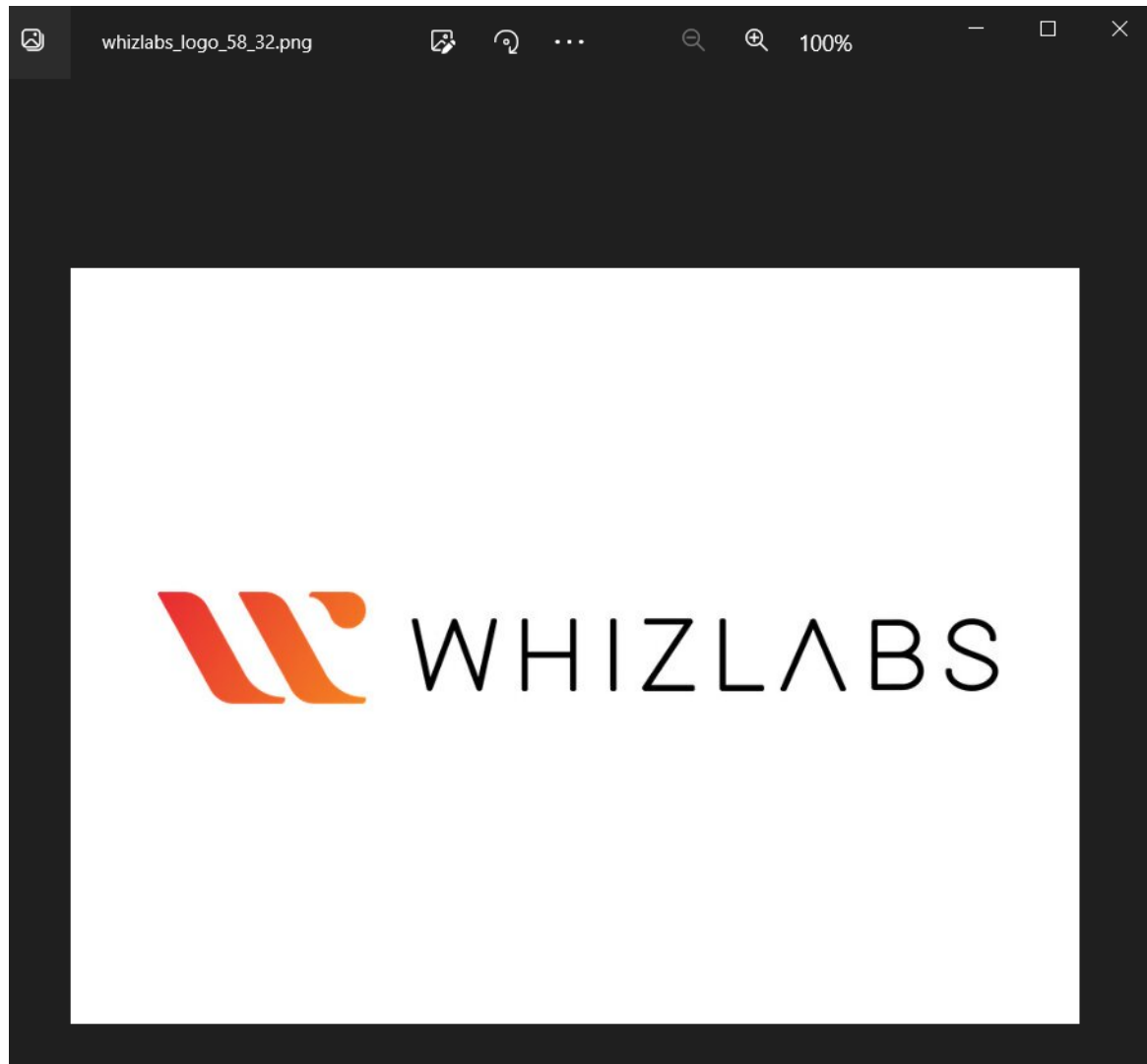
CloudFront > Distributions								
Distributions (1) <a href="#">Info</a>								
<input type="text" value="Search all distributions"/> <span>&lt; 1 &gt;</span>								
<input type="checkbox"/>	ID	Description	Domain name	Alternate do...	Origins	Status	Last modified	
<input type="checkbox"/>	E3R8OWNJFHVGO5	whiz-cloudfront	d2rg9lj23soeu8...	-	whizbucket-4716km.s	Enabled	November 11, 2...	

## Task 12: Accessing Image through CloudFront

Amazon CloudFront is pointed to Amazon S3 bucket origin

1. Click on CloudFront Distribution created and copy the Distribution domain name
2. For testing your distribution, copy your domain name and append your image name after the domain name.
  - Example: [https://d1jptzlydefk0d.cloudfront.net/whizlabs\\_logo\\_58\\_32.png](https://d1jptzlydefk0d.cloudfront.net/whizlabs_logo_58_32.png)
3. Open the CloudFront URL in a new tab. You can see the downloaded image.
4. You can see how much faster the CloudFront URL image loads as compared to the S3 URL. When end users request an object using a CloudFront domain name, they are automatically routed to the nearest edge location for high-performance delivery of your content.

## 5. Open the Downloaded image



## Task 13: Validation of the Lab

1. Once the lab steps are completed, please click on the **Validation** button on the left side panel.
2. This will validate the resources in the AWS account and displays whether you have completed this lab successfully or not.
3. Sample output :

The screenshot shows the Whizlabs lab interface. At the top, there's a navigation bar with 'WHIZLABS', 'Lab Library', 'Cloud Sandboxes', and 'My Activity'. Below this, a blue header contains the lab title 'Securely access S3 images using Amazon CloudFront - Terraform' and its level 'Intermediate'. There are tags for 'Amazon S3', 'Amazon CloudFront', 'Amazon Web Services', and 'Terraform'. The main content area has tabs for 'Lab Overview', 'Lab Steps', and 'Lab Validation'. Under 'Lab Validation', there's a section 'Check your Validation' with instructions on what to do if checks fail. A green button 'Validate My Lab' is present. Below that, a 'Lab Overall Status' section shows a green checkmark and the word 'Passed' with a 100% progress bar.

## Task 14: Delete AWS Resources

1. To delete the resources, open Terminal again.
2. Run the below command to delete all the resources.

```
terraform destroy
```



3. Enter **yes** to confirm the deletion. You can see the **Destroy complete!** message.

```
aws_cloudfront_distribution.s3_distribution: Still destroying... [id=E15KC8C4HHZ43I, 3m0s elapsed]
aws_cloudfront_distribution.s3_distribution: Destruction complete after 3m9s
aws_s3_bucket.bucket: Destroying... [id=whizbucket-qcrxs3]
aws_s3_bucket.bucket: Destruction complete after 3s
random_string.random: Destroying... [id=qcrxs3]
random_string.random: Destruction complete after 0s

Destroy complete! Resources: 6 destroyed.
```

## Do You Know?

The extensive network of CloudFront edge locations enables faster content delivery to users worldwide by caching and serving static and dynamic content from the edge locations closest to them. This reduces the distance and network hops between users and the origin server, resulting in reduced latency and improved performance.

## Completion and Conclusion

- You have successfully set up the Visual Studio Code editor.
- You have successfully created variables.tf and terraform.tfvars files.
- You have successfully created a S3 bucket in the main.tf file.
- You have successfully uploaded an image file S3 bucket in the main.tf file
- You have successfully created a S3 bucket policy in the main.tf file
- You have successfully created a CloudFront Distribution in the main.tf file
- You have successfully created Output.tf file
- You have successfully executed the terraform configuration commands to create the resources.
- You have successfully checked all the resources created by opening the Console.
- You have successfully published an image through CloudFront.
- You have deleted all the resources.

## End Lab

1. Sign out of AWS Account.
2. You have successfully completed the lab.
3. Once you have completed the steps, click on **End Lab** from your whizlabs dashboard.

[About Us](#) [Subscription](#) [Instructions and Guidelines](#) [FAQ's](#) [Contact Us](#)



© 2023, Whizlabs Software Pvt. Ltd.

