Home / AWS / Guided Lab / Host a S3 Static Website using Terraform

Host a S3 Static Website using Terraform

Level: **Fundamental**

Amazon S3 Amazon Web Services Terraform

0h 58m 42s left

End Lab

Open Console

Validation

Lab Credentials

User Name

Whiz_User_80425.31451955

Password

f4e0c9f6-e9b4-4f4b-8569-3228a70171d0

Access Key

AKIA4U7CK74DJ4KUO7N7

Secret Key

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

Lab Resources

1. Download Zip
2. Link

Support Documents

2. Labs - Instructions and Guidelines

Need help?

How to use Hands on Lab

Troubleshooting Lab

FAQs

Submit Feedback

Share

Lab Overview | Lab Steps | Lab Validation

Cloud Architect

Storage, Serverless, 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

1. Create a folder using File Explorer on your local machine's desktop with the name **task_10002_s3**

2. Open the Visual Studio Code application. If you haven't installed the VSC software, download and install using the software using the link provided in the Prerequisite section above.

3. If you have already installed and using Visual Studio Code, Open a New Window by clicking **File** from the menu bar present on the top and select **New Window**.

4. And, now select **Open** present under the **File** menu and navigate to the desktop location. Select the Folder task_10002_s3.

5. 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**.

6. Visual Studio Code is now ready to use.

Task 3: Create a variables file

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

1. To create variables files, click on the **File** from the menu bar and choose **New file**

2. Press Ctrl + S to save the new file as **variables.tf** and click on the **Save** button after entering the file name.

3. **Note:** Don't change the location of the new file, keep it default, i.e., inside the **task_10002_s3** folder**.**

4. Paste the below contents in **variables.tf** file:

```
# required for AWS
variable "access_key" {
    description = "Access key to AWS console"
}
variable "secret_key" {
    description = "Secret key to AWS console"
}
variable "region" {
    description = "Region of AWS VPC"
}
```

5. In the above content, you are declaring 3 variables called access_key, secret_key, and region.

6. After pasting the above contents, save the file by pressing Ctrl + S.

7. Now create the **terraform.tfvars** file by selecting **New file** present under **File** in the menu bar.

8. Name the file by pressing Ctrl + S and enter **terraform.tfvars**

9. Paste the below content into **terraform.tfvars** file

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

10. Replace the values of **access_key** and **secret_key** by copying the AWS Access Key ID and Secret Access Key ID provided in Whizlabs Labs console.
11. After replacing the values of access_key and secret_key, save the file by pressing Ctrl + S.

Task 4: Create an S3 bucket and its components in main.tf file

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

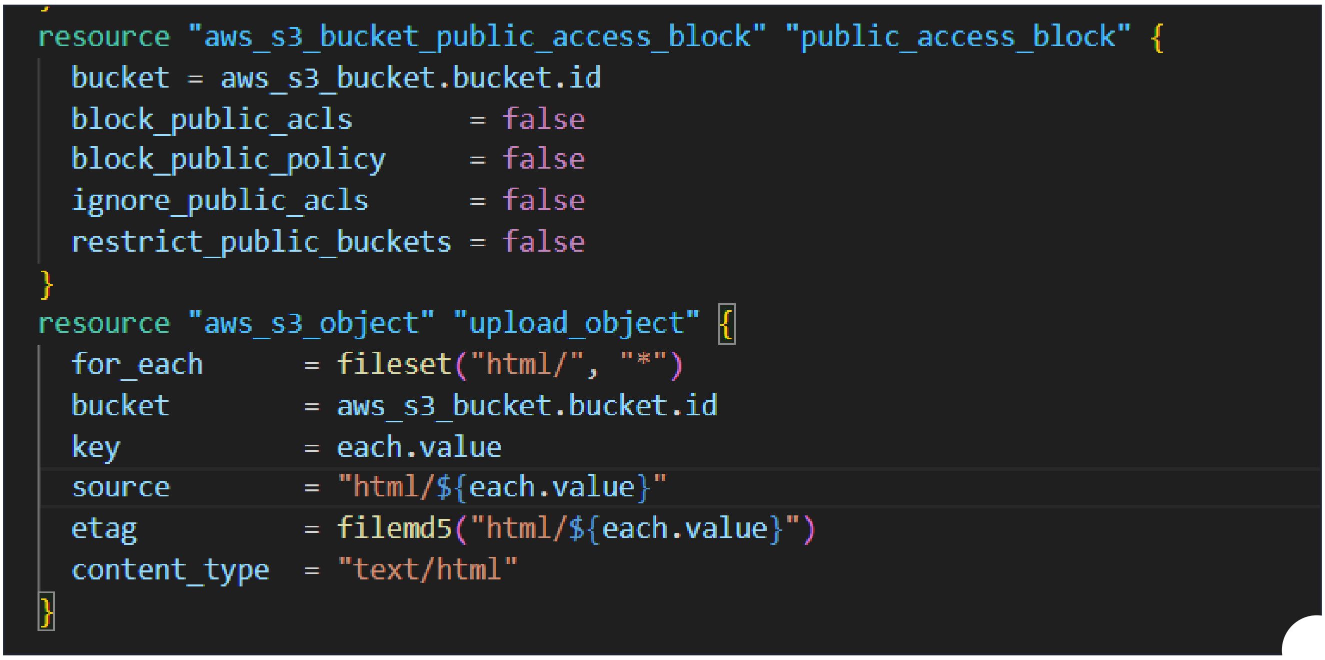
1. To create **main.tf** file, click on the **File** from the menu bar and choose **New file**
2. Press Ctrl + S to save the new file as **main.tf** and click on the **Save** button after entering the file name.
3. Paste the below content into **main.tf** file.

```
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
}
resource "aws_s3_bucket_website_configuration" "blog" {
  bucket = aws_s3_bucket.bucket.id
  index_document {
    suffix = "index.html"
  }
  error_document {
    key = "error.html"
  }
}
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
}
```

4. In the above code, you are defining the provider as **AWS** and provisioning AWS resources, including an S3 bucket, by configuring the AWS provider and using a random string resource to generate a unique bucket name. It also sets up the S3 bucket for website hosting and configures public access settings.

5. Add another block of just below the s3 bucket public access code, this **block of code will upload all the files present under HTML folder to the S3 bucket.**

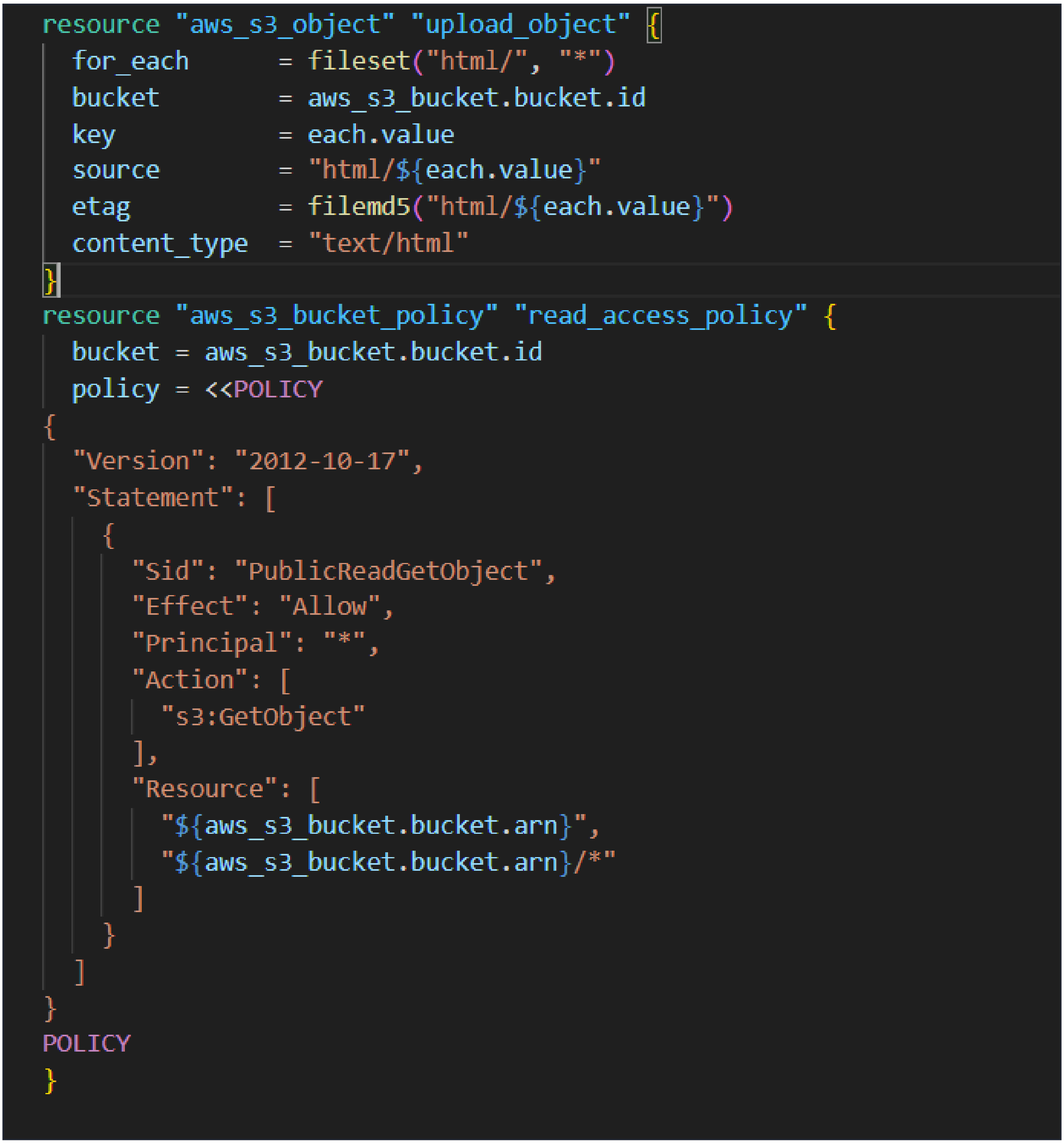
```
resource "aws_s3_object" "upload_object" {
  for_each      = fileset("html/", "*")
  bucket        = aws_s3_bucket.bucket.id
  key           = each.value
  source        = "html/${each.value}"
  etag          = filemd5("html/${each.value}")
  content_type  = "text/html"
}
```



```
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
}
resource "aws_s3_object" "upload_object" {
  for_each      = fileset("html/", "*")
  bucket        = aws_s3_bucket.bucket.id
  key           = each.value
  source        = "html/${each.value}"
  etag          = filemd5("html/${each.value}")
  content_type  = "text/html"
}
```

5. Finally, make the bucket public by adding the bucket policy by adding another block of code to the **main.tf** file.

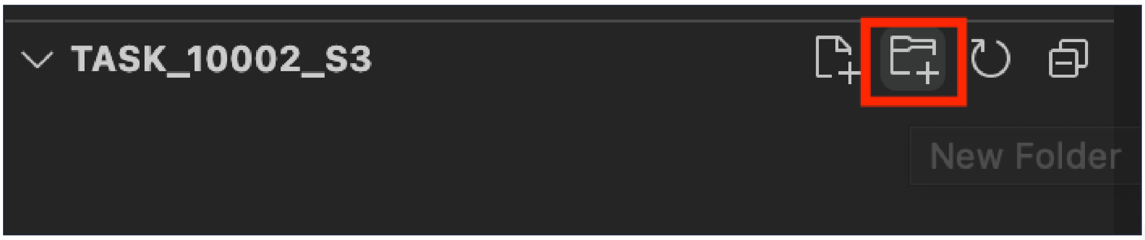
```
resource "aws_s3_bucket_policy" "read_access_policy" {
  bucket = aws_s3_bucket.bucket.id
  policy = <<POLICY
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "${aws_s3_bucket.bucket.arn}",
        "${aws_s3_bucket.bucket.arn}/*"
      ]
    }
  ]
}
POLICY
}
```



```
resource "aws_s3_object" "upload_object" {
  for_each      = fileset("html/", "*")
  bucket        = aws_s3_bucket.bucket.id
  key           = each.value
  source        = "html/${each.value}"
  etag          = filemd5("html/${each.value}")
  content_type  = "text/html"
}
resource "aws_s3_bucket_policy" "read_access_policy" {
  bucket = aws_s3_bucket.bucket.id
  policy = <<POLICY
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "${aws_s3_bucket.bucket.arn}",
        "${aws_s3_bucket.bucket.arn}/*"
      ]
    }
  ]
}
POLICY
}
```

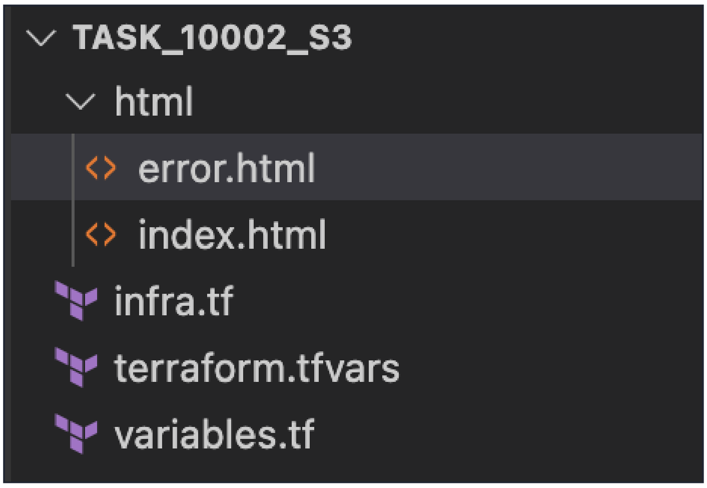
6. **Before moving to the next step, save the file using Ctrl + S.**

7. Next, you will create a new folder to place the index.html and error.html files. To create a folder, click on the **Create folder icon** present on the left side panel under your folder called **task_10002_s3**.



8. Name this folder as **HTML**.

9. Now click on this Link to download file to your local, extract the zip file.
10. Place the **index.html** and **error.html** files inside the newly created folder called **HTML**
11. Or you can manually create an index.html and error.html file inside the **HTML** folder and paste the contents of downloaded files.
12. By now, your files and folder structure should look something like this. As we have created **main.tf**, **terraform.tfvars** and **variables.tf** in the root directory of the folder and index.html and error.html inside the HTML folder.



```
TASK_10002_S3
  html
    error.html
    index.html
  infra.tf
  terraform.tfvars
  variables.tf
```

13. Create an **outputs.tf** file required for displaying the output as website endpoint.
14. To create **outputs.tf** file, click on the **File** from the menu bar and choose **New file**
15. Press Ctrl + S to save the new file as **outputs.tf** and click on the **Save** button after entering the file name.
16. Paste the below content into **outputs.tf** file.

```
output "s3_bucket_id" {
  value = aws_s3_bucket_website_configuration.blog.website_endpoint
}
```

17. Save the file by pressing Ctrl + S.

Task 5: Confirm the installation of Terraform by checking the version

1. Open the Terminal by clicking on the **View** from the menu bar and choosing **Terminal**
2. To confirm the installation of Terraform, run the below command to check the version:

```
terraform version
```

3. 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 6: Apply terraform configurations

1. Initialize Terraform by running the below command:

```
terraform init
```

2. **Note:** terraform init will check for all the plugin dependencies and download them if required, this will be used for creating a deployment plan.
3. To generate the action plans, run the below command:

```
terraform plan
```

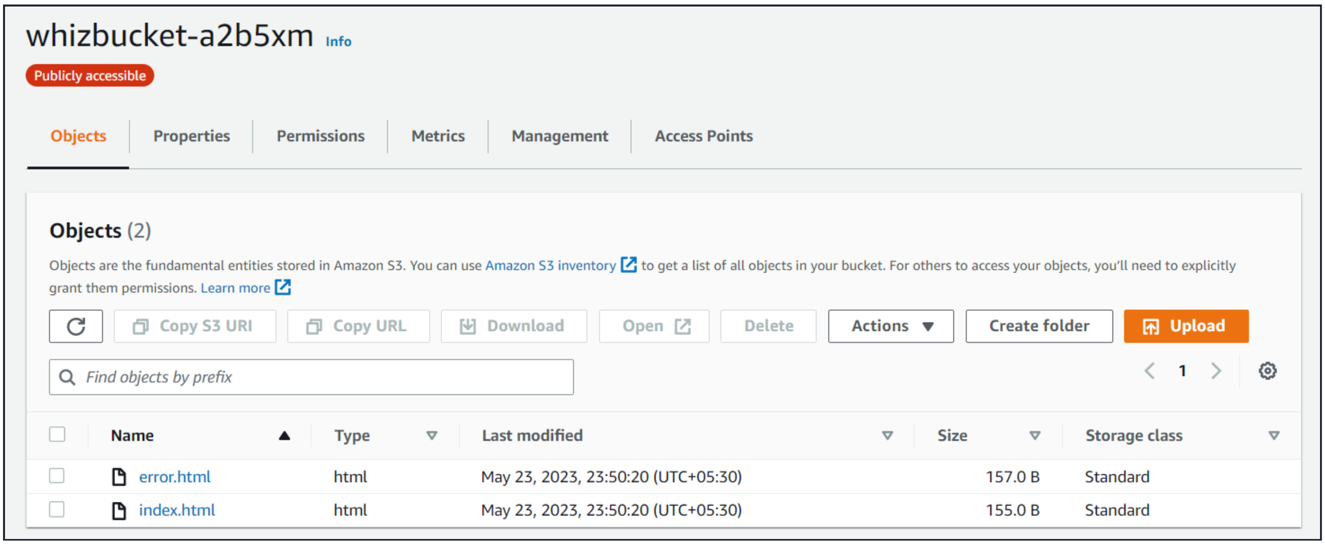
4. Review the whole generated plan.
5. To create all the resources declared in **main.tf** configuration file, run the below command:

```
terraform apply
```

6. You will be able to see the resources which will be created, approve the creation of all the resources by entering **yes.**
7. It may take up to 2 minutes for the **terraform apply** command to create the resources.
8. Ids of all the resources created by terraform will be visible there.
9. Optionally, you can note down the IDs of all the resources.

Task 7: Check the resources in AWS Console

1. Navigate to AWS Management Console page in your browser.
2. Make sure you are in the **US East (N. Virginia) us-east-1** Region.
3. Navigate to the **Services** menu at the top. Click on **S3** in the **Storage** section.
4. On the home page of the S3 bucket, click on the bucket you created.
5. After you open the bucket, two files are present, index.html and error.html



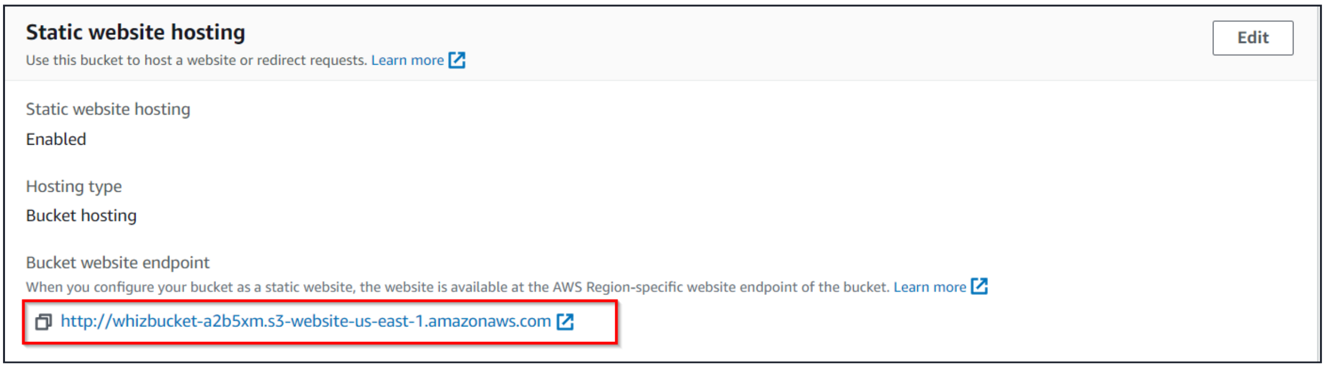
Objects (2)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
error.html	html	May 23, 2023, 23:50:20 (UTC+05:30)	157.0 B	Standard
index.html	html	May 23, 2023, 23:50:20 (UTC+05:30)	155.0 B	Standard

6. Switch to the properties tab and scroll to the end of this page to find Static website hosting options.

7. Copy the Bucket website endpoint and paste into the new tab of your web browser, with /index.html at the end.



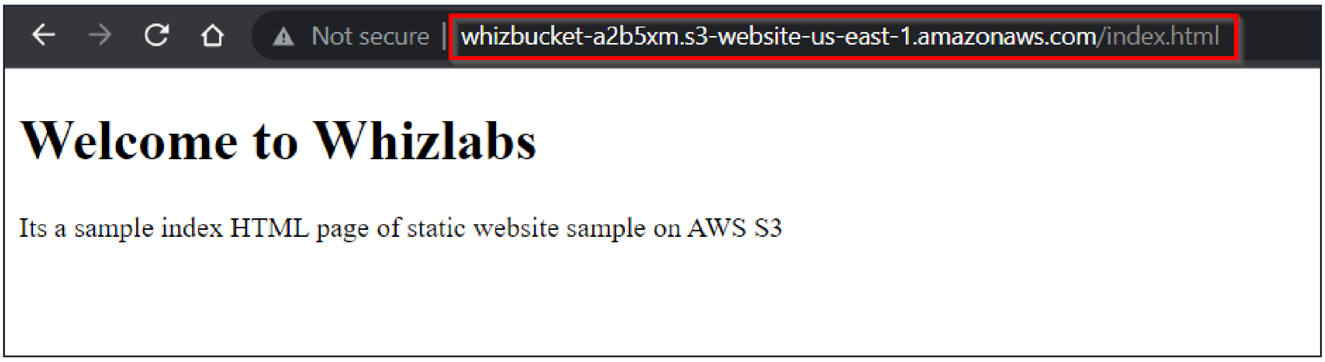
Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Static website hosting
Enabled

Hosting type
Bucket hosting

Bucket website endpoint
When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. Learn more
http://whizbucket-a2b5xm.s3-website-us-east-1.amazonaws.com

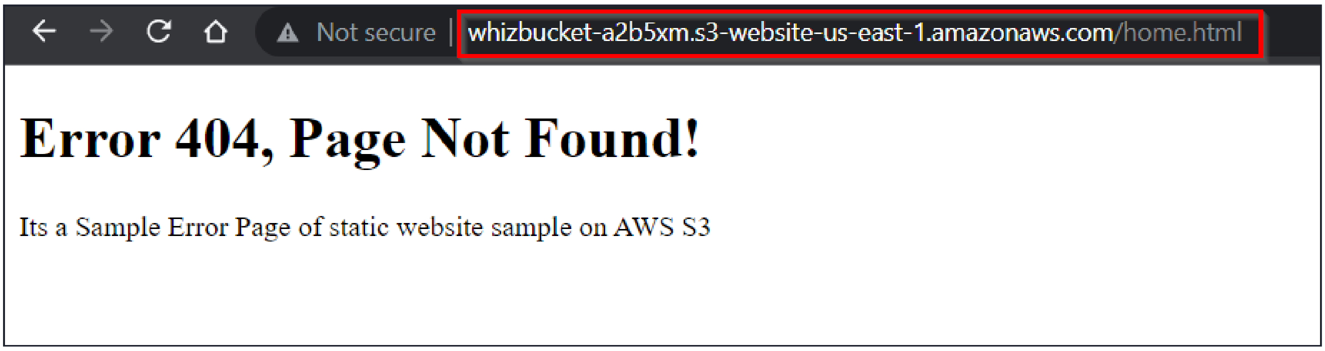


whizbucket-a2b5xm.s3-website-us-east-1.amazonaws.com/index.html

Welcome to Whizlabs

Its a sample index HTML page of static website sample on AWS S3

6. For testing error.html feature is working or not, replace index.html with home.html. Now you should see the contents of error.html


Not secure
whizbucket-a2b5xm.s3-website-us-east-1.amazonaws.com/home.html

Error 404, Page Not Found!

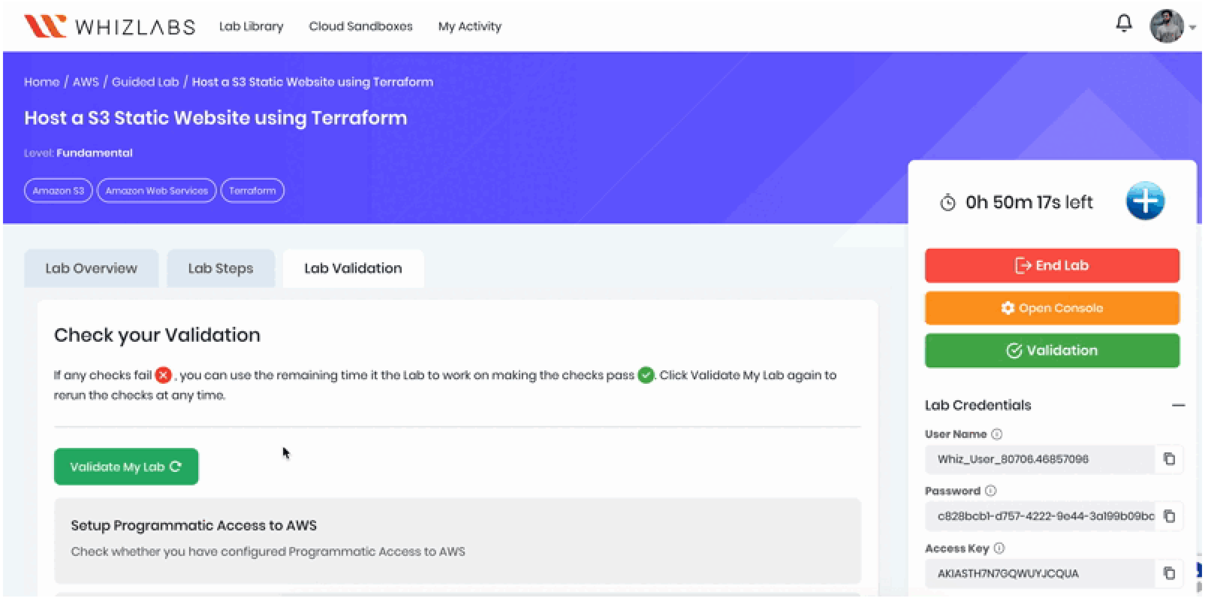
Its a Sample Error Page of static website sample on AWS S3

Do You Know?

S3 automatically scales to accommodate high volumes of web traffic without any additional configuration or intervention. This means that even if your static website experiences a sudden surge in traffic, S3 can handle it effortlessly without any impact on performance. This scalability and reliability make S3 an excellent choice for hosting static websites with high availability requirements.

Task 8: 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 :


WHIZLABS
Lab Library
Cloud Sandboxes
My Activity
Home / AWS / Guided Lab / Host a S3 Static Website using Terraform

Host a S3 Static Website using Terraform

Level: Fundamental

Check your Validation

If any checks fail ✗, you can use the remaining time it the Lab to work on making the checks pass ✓. Click Validate My Lab again to rerun the checks at any time.

Setup Programmatic Access to AWS

Check whether you have configured Programmatic Access to AWS

0h 50m 17s left

Lab Credentials

User Name: Whiz_User_80706.46857096

Password: c828bcbl-d757-4222-9e44-3a199b09bc

Access Key: AKIASTH7N7GQWUYJCQUA

Task 9: 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.

Completion and Conclusion

- You have successfully done the setup of the Visual Studio Code editor.
- You have successfully created **variables.tf** and **terraform.tfvars** file
- You have successfully created a **main.tf** file.
- You have successfully executed the terraform configuration commands to create the resources.
- You have successfully checked all the resources are created by opening the Console.
- You have successfully 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 lab console and wait till the process gets completed.