Creating a Web Site @ AWS

(A Manual Way / An Automated Way)

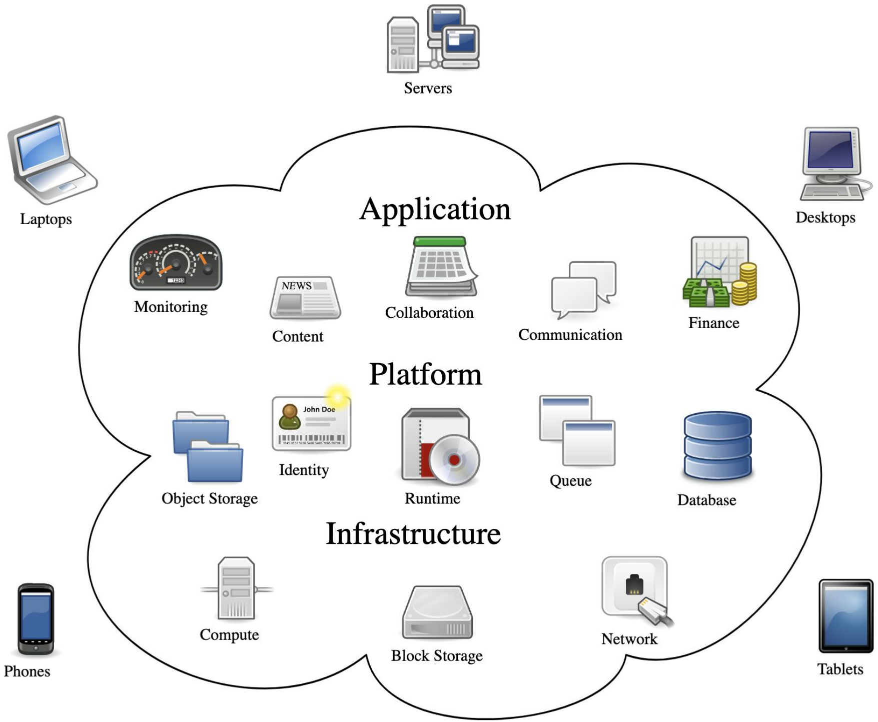
2021 Hour Of Code @ SHS GWC

What is the cloud?

Virtually all of us use it every day, even if we don't know it.

Apple, Google, Facebook, Twitter, Amazon, You Tube, Instagram, Wikipedia, Netflix, Zoom, Tiktok, LinkedIn, Pinterest, etc…

Use cloud services….



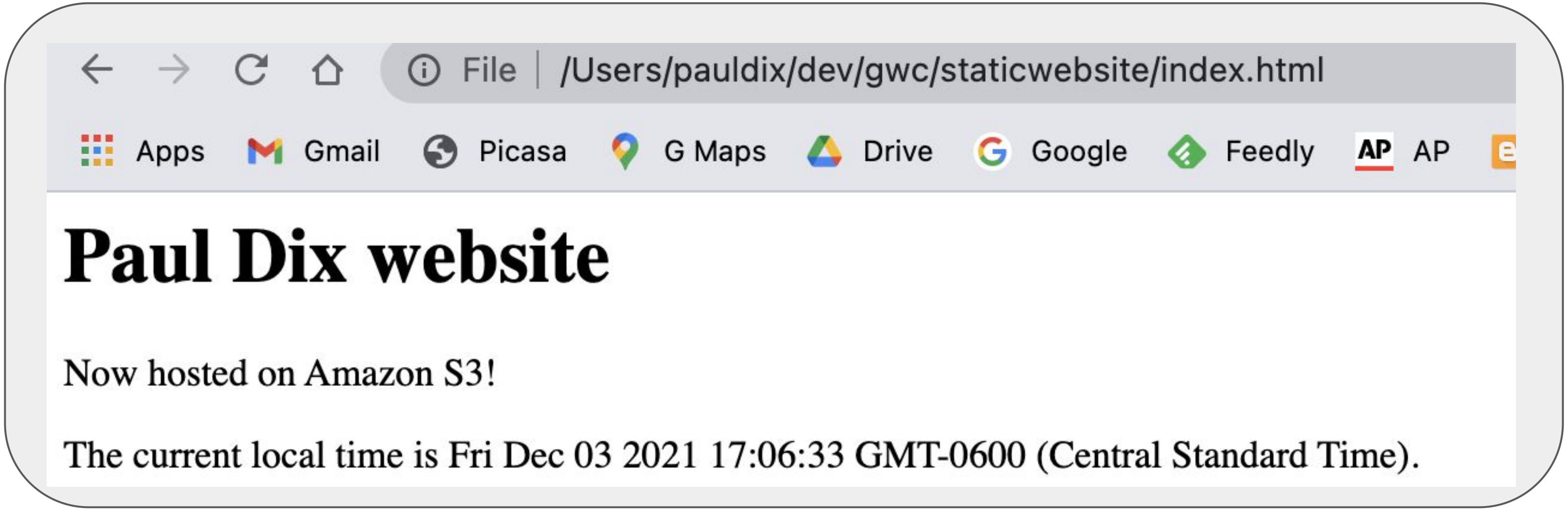
What is a Cloud Provider?

amazon web services™
Google Cloud Platform
DigitalOcean
Microsoft Azure
vmware
openstack
Pivotal Cloud Foundry
IBM Cloud
ORACLE CLOUD
Alibaba Cloud

A company that offers the various services (Application/Platform/Infrastructure) listed on the previous slide

What will we cover today?

Let’s make a simple web site using the cloud provider AWS (Amazon Web Services)



Paul Dix website

Now hosted on Amazon S3!

The current local time is Fri Dec 03 2021 17:06:33 GMT-0600 (Central Standard Time).

How will we do that?

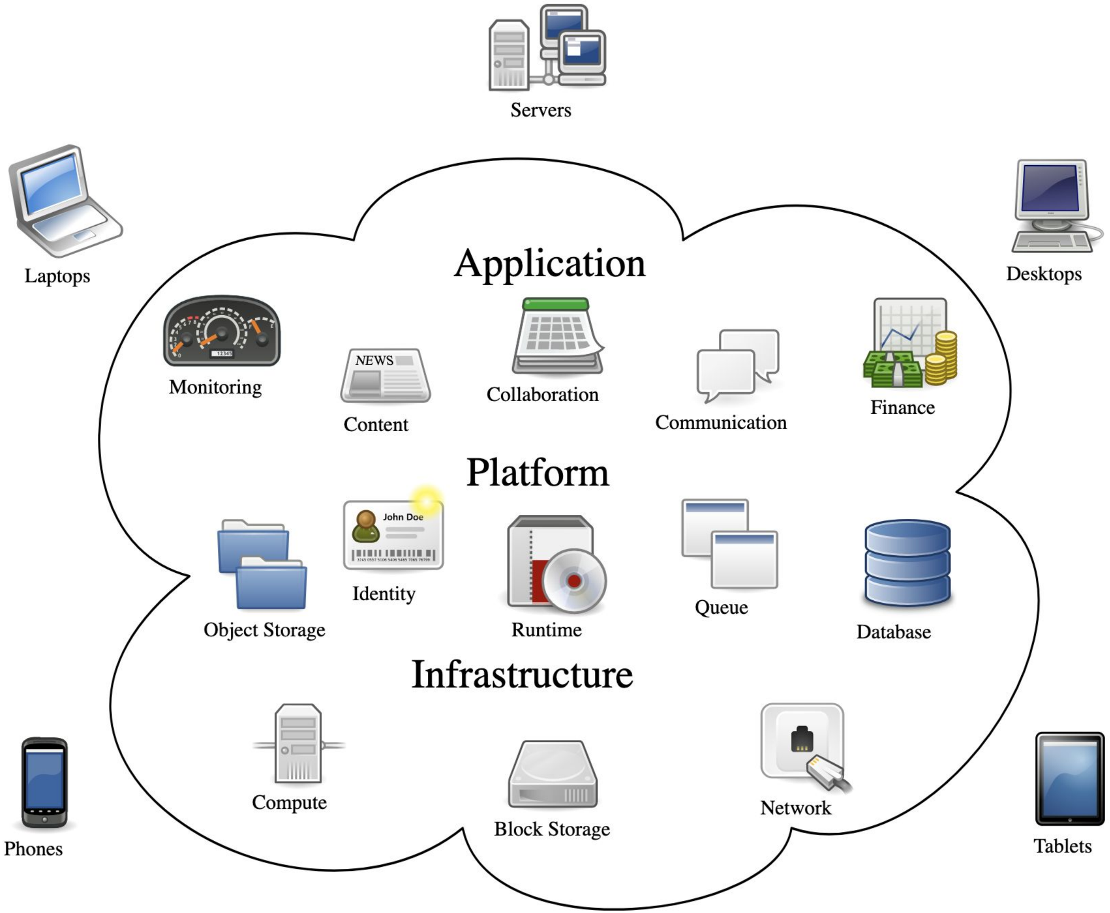
- We will use an HTML (Hypertext Mark Up) file for our page.
- Here is what the code looks like for that.
- You are already provided this code, so you don’t need to type it in.
- If you are already familiar with HTML you can add in additional items to the page as you see fit.

```
<html
xmlns="http://www.w3.org/1999/xhtml" >
<head>
   <title>Paul Dix Home Page</title>
</head>
<body>
 <h1>Paul Dix website</h1>
 <p>Now hosted on Amazon S3!</p>
 <p>
   The current local time is
   <script
type="text/javascript">document.write(
new Date() );</script>.
   </p>
</body>
</html>
```

What services will we use?

***Object Storage** - a.k.a. A place to store different types of files*

***Content Application** - A way to do a website with files*

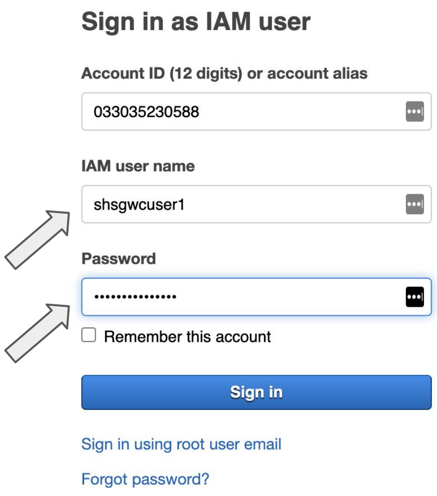


Steps we will follow

1. Login to the AWS Console
2. Configure an S3 bucket for static website hosting
3. Configure it to make it accessible from the web
4. Create the HTML File (test it locally first)
5. Upload the HTML file to the bucket
6. Test

Login to the AWS Console

Press on the provided link and login with the provided ID and password.
https://033035230588.signin.aws.amazon.com/console



Sign in as IAM user

Account ID (12 digits) or account alias: 033035230588

IAM user name: shsgwcuser1

Password: ....

☐ Remember this account

Sign in

Sign in using root user email

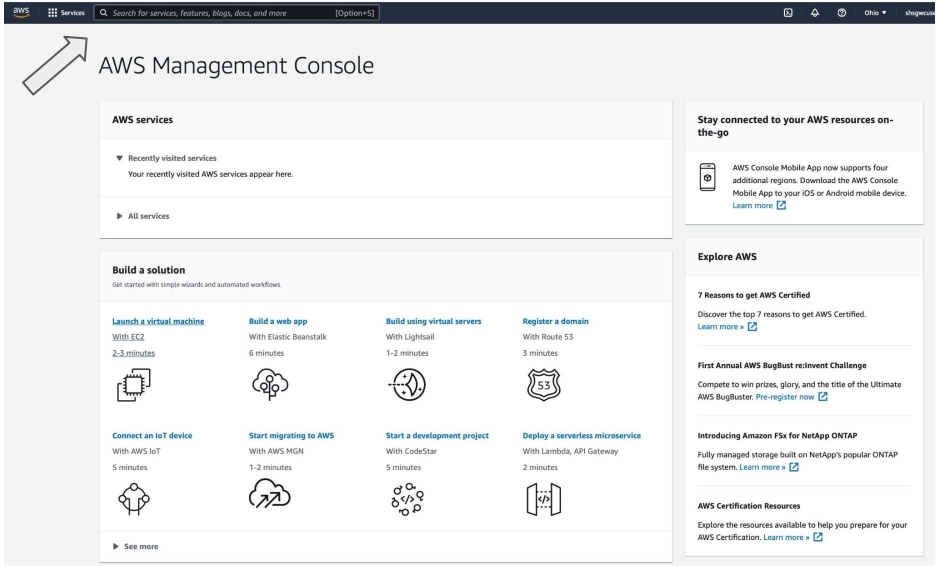
Forgot password?

I created seven login accounts. Each of you take a different account.
Password is **shs.gwc+pass?** where ? is the same number your ID ends with.
Example **shsgwcuser1** has a password of **shs.gwc+pass1**

shsgwcuser1 **shsgwcuser2**	**shsgwcuser3** **shsgwcuser4**	**shsgwcuser5** **shsgwcuser6**	**shsgwcuser7**

Navigate to the S3 Service

In the upper left enter “S3” to filter the list of services down. Select S3 from the list of presented services…..



AWS Management Console

AWS services

▼ Recently visited services

Your recently visited AWS services appear here.

▶ All services

Build a solution

Get started with simple wizards and automated workflows.

Launch a virtual machine With EC2 2-3 minutes	Build a web app With Elastic Beanstalk 6 minutes	Build using virtual servers With Lightsail 1-2 minutes	Register a domain With Route 53 3 minutes
Connect an IoT device With AWS IoT 5 minutes	Start migrating to AWS With AWS MGN 1-2 minutes	Start a development project With CodeStar 5 minutes	Deploy a serverless microservice With Lambda, API Gateway 2 minutes

▶ See more

Stay connected to your AWS resources on-the-go

AWS Console Mobile App now supports four additional regions. Download the AWS Console Mobile App to your iOS or Android mobile device. Learn more

Explore AWS

7 Reasons to get AWS Certified

Discover the top 7 reasons to get AWS Certified. Learn more »

First Annual AWS BugBust re:Invent Challenge

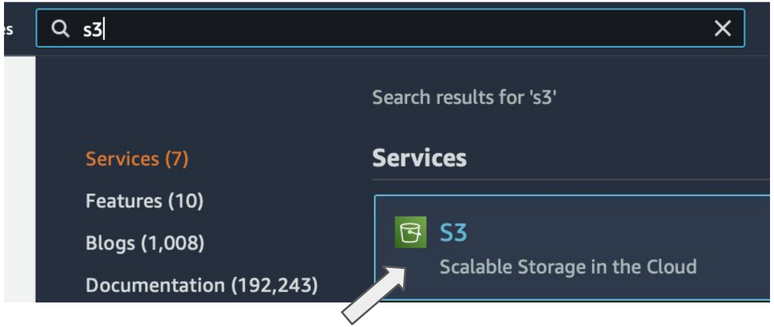
Compete to win prizes, glory, and the title of the Ultimate AWS BugBuster. Pre-register now

Introducing Amazon FSx for NetApp ONTAP

Fully managed storage built on NetApp's popular ONTAP file system. Learn more »

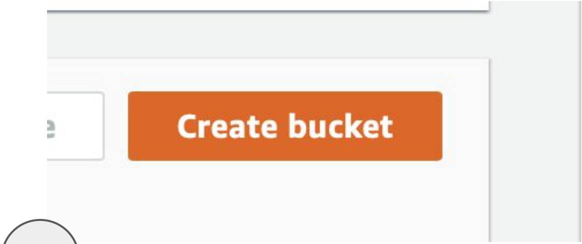
AWS Certification Resources

Explore the resources available to help you prepare for your AWS Certification. Learn more »



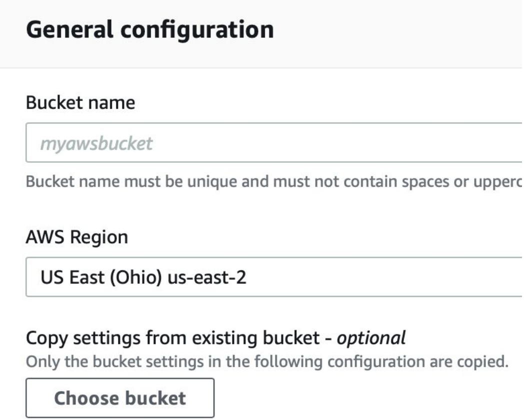
Create an S3 bucket

In the upper left enter “S3” to filter the list of services down. Select S3 from the list of presented services…..



1 Press “Create Bucket”

2 Enter in a bucket name. Follow this naming standard. **shsgwc53211bucket?** Where ? is the same number you used with your user id. Example **shsgwcuser1** uses **shsgwc53211bucket1** as the name of their bucket.


General configuration

Bucket name

myawsbucket

Bucket name must be unique and must not contain spaces or upperc

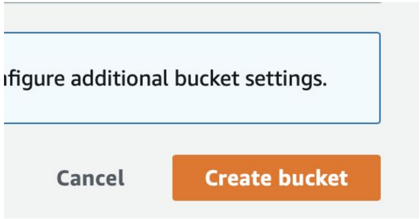
AWS Region

US East (Ohio) us-east-2

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

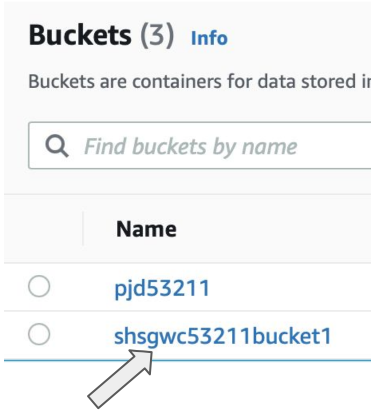
3 Take all defaults on the remainder of the page. Press the “Create Bucket” at the bottom of the page.


ıfigure additional bucket settings.

Enable Hosting

Select your bucket to set additional properties. Make sure to select the bucket that you created (1,2,3,4,5,6,7,8, etc.…)

1



Buckets (3) Info

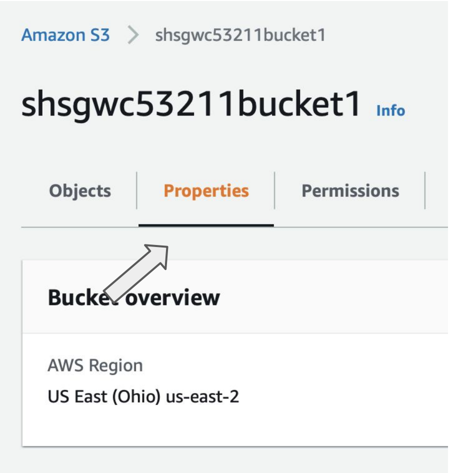
Buckets are containers for data stored i

Find buckets by name

	Name
○	pjd53211
○	shsgwc53211bucket1

Select your bucket from the list

2



Amazon S3 > shsgwc53211bucket1

shsgwc53211bucket1 Info

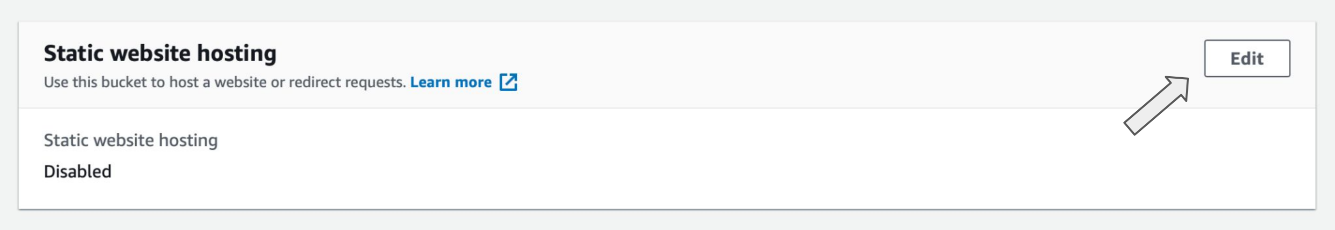
Bucket overview

AWS Region

US East (Ohio) us-east-2

Select the “Properties” tab on next screen

Enable Hosting (2)



Static website hosting

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

Static website hosting

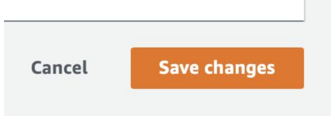
Disabled

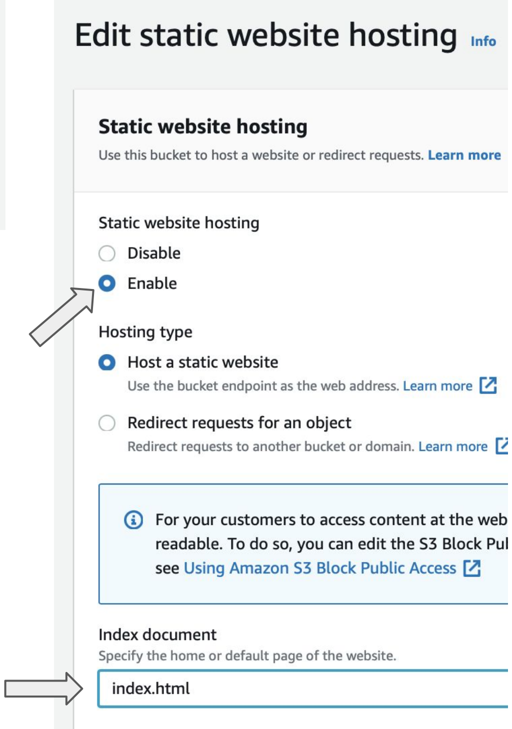
1 Press “Edit” in this section

2 Next screen, select “Enable”

3 Fill in the “Index Document”

3 Press “Save Changes” at bottom of the screen





Edit static website hosting Info

Static website hosting

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

Static website hosting

○ Disable

● Enable

Hosting type

● Host a static website

Use the bucket endpoint as the web address. Learn more

○ Redirect requests for an object

Redirect requests to another bucket or domain. Learn more

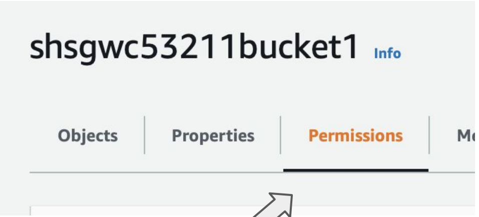
For your customers to access content at the web readable. To do so, you can edit the S3 Block Pul see Using Amazon S3 Block Public Access

Index document

Specify the home or default page of the website.

index.html

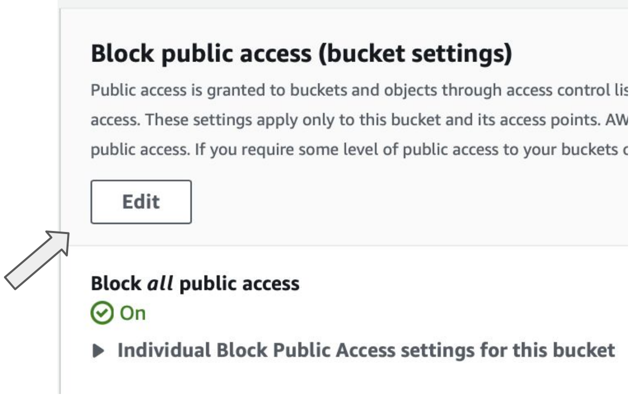
Public Access Settings



shsgwc53211bucket1 Info

Objects | Properties | Permissions | Mo

1 Now select the “Permissions” tab



Block public access (bucket settings)

Public access is granted to buckets and objects through access control lis access. These settings apply only to this bucket and its access points. AW public access. If you require some level of public access to your buckets c

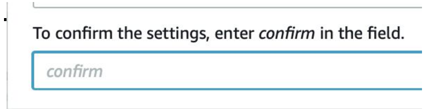
Edit

Block *all* public access

✓ On

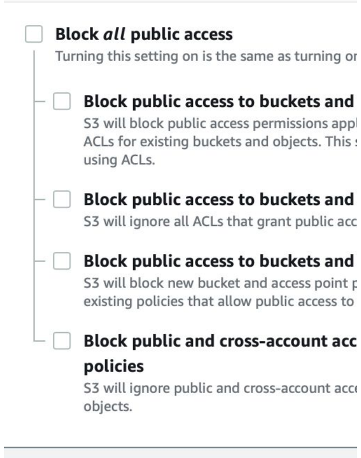
▶ Individual Block Public Access settings for this bucket

2 Uncheck the top box for block all public access.
All child boxes will be unchecked.
Press Save.
Pop up box. Will need to enter the word “confirm” and press confirm button.



To confirm the settings, enter *confirm* in the field.

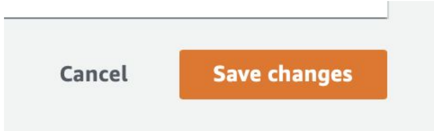
confirm



□ **Block *all* public access**
Turning this setting on is the same as turning or

- □ **Block public access to buckets and**
  S3 will block public access permissions appl ACLs for existing buckets and objects. This s using ACLs.
- □ **Block public access to buckets and**
  S3 will ignore all ACLs that grant public acc
- □ **Block public access to buckets and**
  S3 will block new bucket and access point p existing policies that allow public access to
- □ **Block public and cross-account acc policies**
  S3 will ignore public and cross-account acce objects.

3 All check boxes unselected.



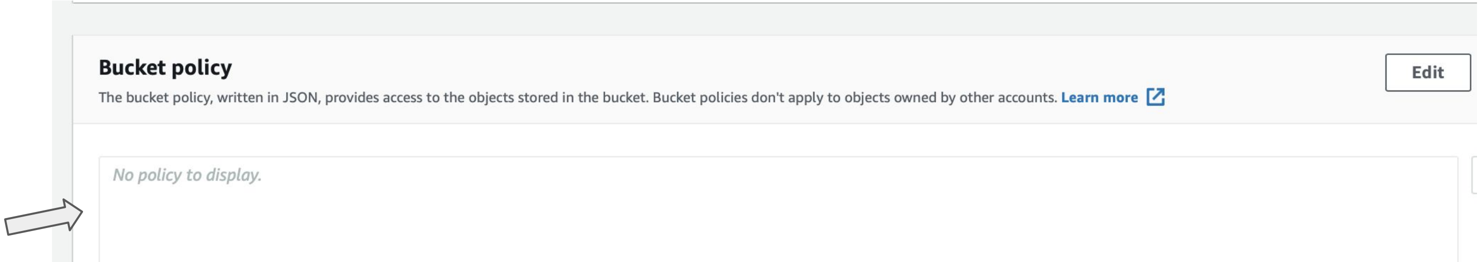
Cancel | Save changes

4 Press “Save Changes”

Set Bucket Policy for Public Access

1

Again on the “Permissions” tab, go to “Bucket Policy”.



Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

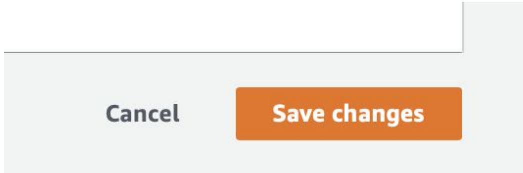
No policy to display.

2

Clear out existing text. Copy and paste in the provided policy statement. **Set the bucket name appropriately!!!**

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::shsgwc53211bucket1/*"
            ]
        }
    ]
}
```

3

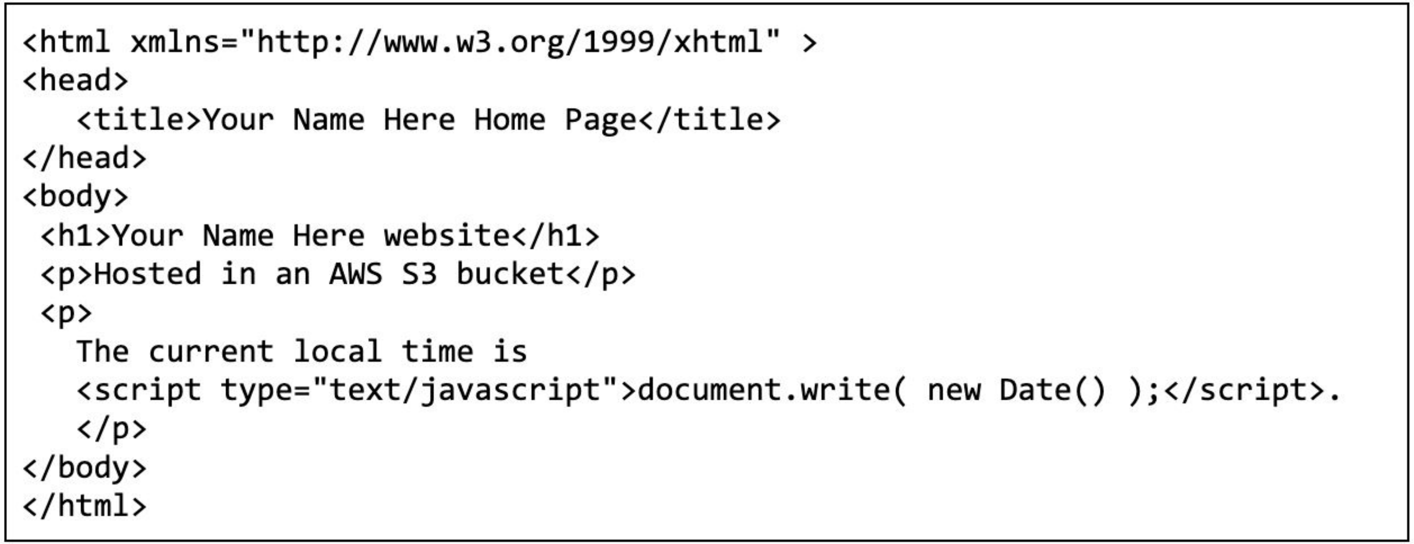


Press “Save Changes”.

Verify policy text is saved and applied in window!!!

Create HTML File on your local machine

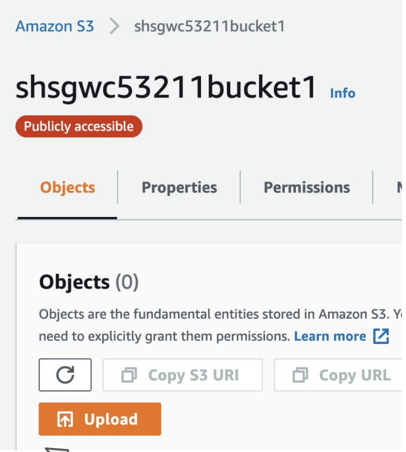
1 Create **index.html** file on your local machine. File must be named exactly that.



```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
   <title>Your Name Here Home Page</title>
</head>
<body>
 <h1>Your Name Here website</h1>
 <p>Hosted in an AWS S3 bucket</p>
 <p>
   The current local time is
   <script type="text/javascript">document.write( new Date() );</script>.
   </p>
</body>
</html>
```

Upload HTML File into S3 bucket

1 Return to S3 page, select bucket, press “Upload”..



Amazon S3 > shsgwc53211bucket1

shsgwc53211bucket1 Info

Publicly accessible

Objects | Properties | Permissions | M

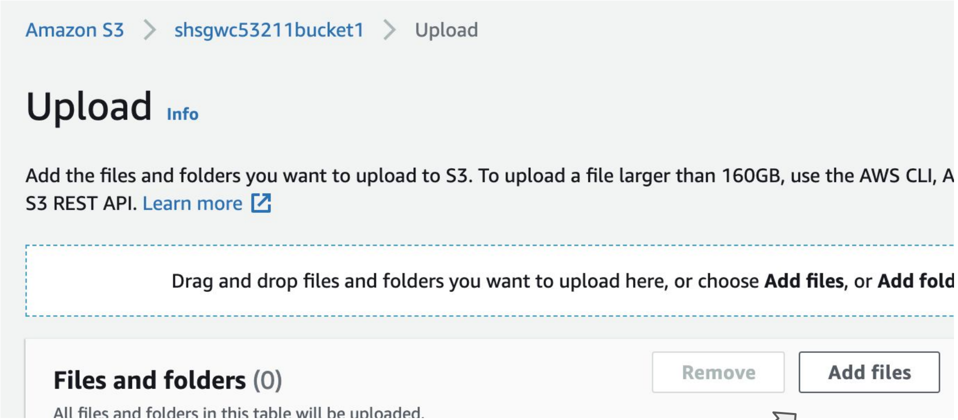
Objects (0)

Objects are the fundamental entities stored in Amazon S3. Y
need to explicitly grant them permissions. Learn more

Copy S3 URI | Copy URL

Upload

2 Press “Add Files”, navigate to your file.



Amazon S3 > shsgwc53211bucket1 > Upload

Upload Info

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, A
S3 REST API. Learn more

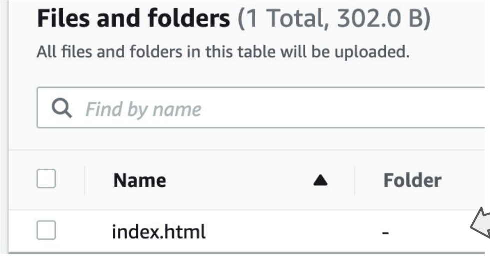
Drag and drop files and folders you want to upload here, or choose **Add files**, or **Add fold**

Files and folders (0)

Remove | Add files

All files and folders in this table will be uploaded.

3 Verify file in list.



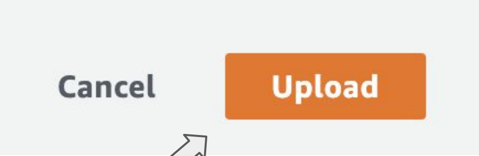
Files and folders (1 Total, 302.0 B)

All files and folders in this table will be uploaded.

Find by name

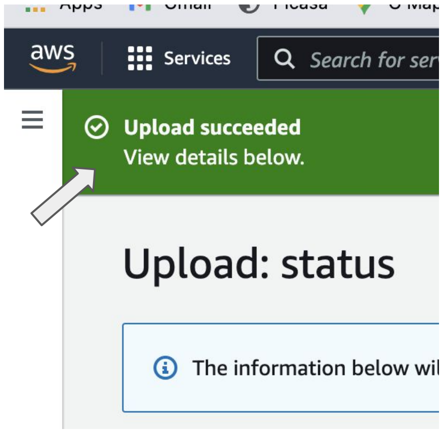
□	Name ▲	Folder
□	index.html	-

4 Press “Upload” to complete.



Verify Steps

1 Look for “Upload succeeded”..

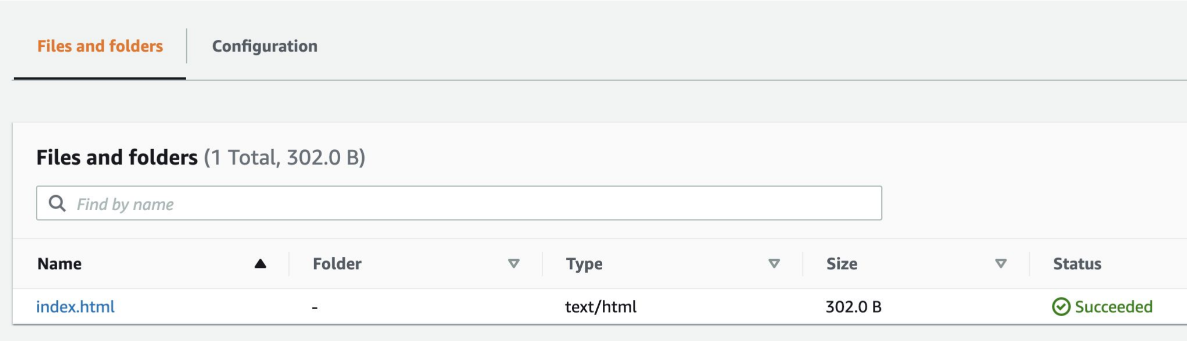


Upload succeeded
View details below.

Upload: status

The information below wil

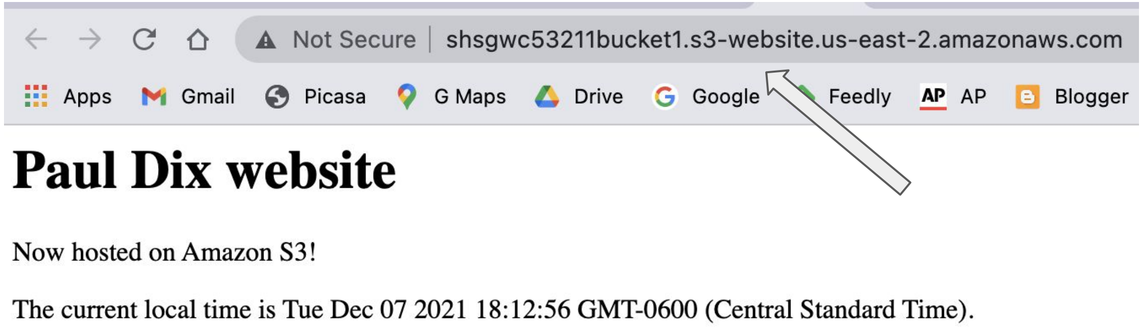
2 Look for “Succeeded” under Files and Folders for the bucket



Files and folders (1 Total, 302.0 B)

Name ▲	Folder ▽	Type ▽	Size ▽	Status
index.html	-	text/html	302.0 B	✓ Succeeded

3 Test out the web site with the link you were given before class.



shsgwc53211bucket1.s3-website.us-east-2.amazonaws.com

Paul Dix website

Now hosted on Amazon S3!

The current local time is Tue Dec 07 2021 18:12:56 GMT-0600 (Central Standard Time).

Wow! That was a lot of clicks!

There’s got to be a better way to do all of those steps

There is!

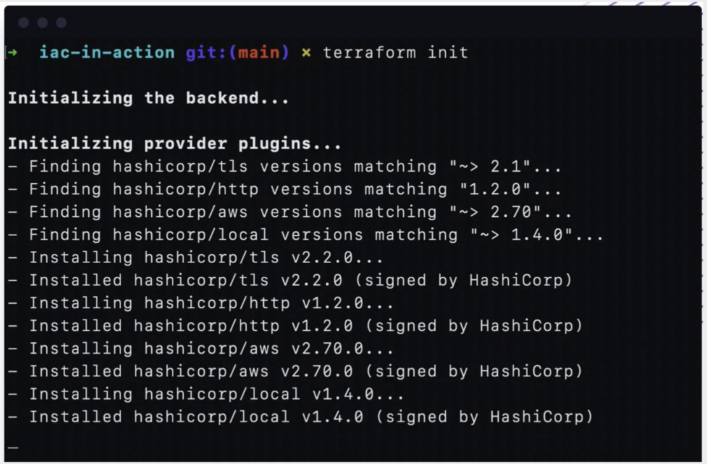
This is the cloud, we can automate just about everything, including all those steps we just did (except for making the HTML file).

Application Code vs. Infrastructure Code

HTML - Web Site Application

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
   <title>Your Name Here Home Page</title>
</head>
<body>
 <h1>Your Name Here website</h1>
 <p>Hosted in an AWS S3 bucket</p>
 <p>
   The current local time is
   <script
type="text/javascript">document.write( new
Date() );</script>.
   </p>
</body>
</html>
```

Terraform - Code to build your Web Site



```
➜  iac-in-action git:(main) ✗ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/tls versions matching "~> 2.1"...
- Finding hashicorp/http versions matching "1.2.0"...
- Finding hashicorp/aws versions matching "~> 2.70"...
- Finding hashicorp/local versions matching "~> 1.4.0"...
- Installing hashicorp/tls v2.2.0...
- Installed hashicorp/tls v2.2.0 (signed by HashiCorp)
- Installing hashicorp/http v1.2.0...
- Installed hashicorp/http v1.2.0 (signed by HashiCorp)
- Installing hashicorp/aws v2.70.0...
- Installed hashicorp/aws v2.70.0 (signed by HashiCorp)
- Installing hashicorp/local v1.4.0...
- Installed hashicorp/local v1.4.0 (signed by HashiCorp)
_
```

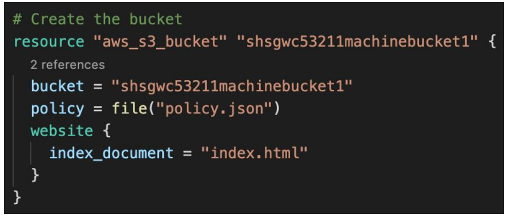
Terraform

The Terraform Language

Terraform uses **declarative configuration** to describe the desired final state. Once a user invokes Terraform on a given resource, Terraform will perform actions on the user's behalf to accomplish the **desired state**.

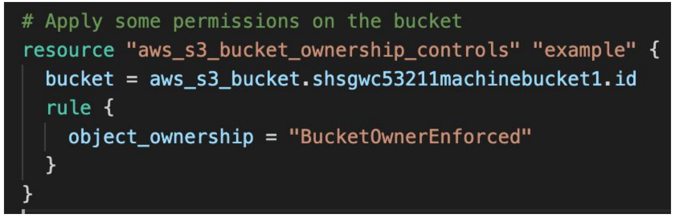
Three different types of resources are declared. S3 buckets, permissions on a bucket and the file…
Terraform will determine the order to build these resources and then build them to achieve desired state...

Create the bucket...



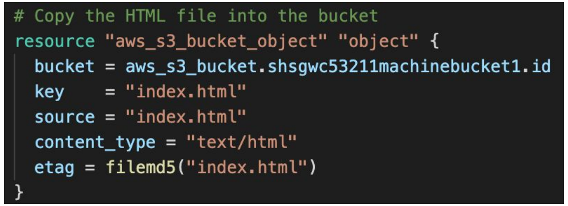
```
# Create the bucket
resource "aws_s3_bucket" "shsgwc53211machinebucket1" {
  2 references
  bucket = "shsgwc53211machinebucket1"
  policy = file("policy.json")
  website {
    index_document = "index.html"
  }
}
```

Apply permissions to the bucket...



```
# Apply some permissions on the bucket
resource "aws_s3_bucket_ownership_controls" "example" {
  bucket = aws_s3_bucket.shsgwc53211machinebucket1.id
  rule {
    object_ownership = "BucketOwnerEnforced"
  }
}
```

Upload file to the bucket...



```
# Copy the HTML file into the bucket
resource "aws_s3_bucket_object" "object" {
  bucket = aws_s3_bucket.shsgwc53211machinebucket1.id
  key    = "index.html"
  source = "index.html"
  content_type = "text/html"
  etag = filemd5("index.html")
}
```

The Terraform Language

Procedural Programming involves writing down a list of instructions to tell the computer what it should do step-by-step to finish the task at hand.

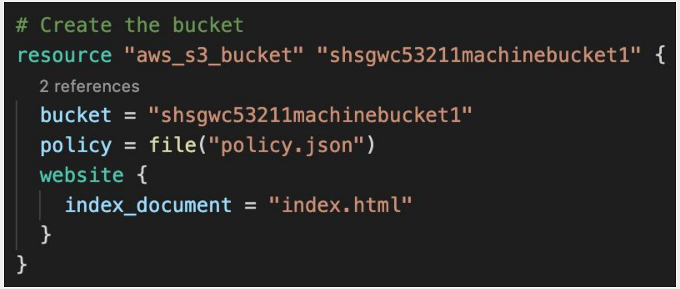
Procedural code is the one that directly instructs a device on how to finish a task in logical steps.

Here is some python code (a function) that lists S32 buckets.
We can see a request, a response and returning that response to the code that called this function.

```
   ## Commands / Actions
   def list_all_buckets(self):
       request = self.create_request("LIST_ALL_BUCKETS")
       response = self.send_request(request)
       response["list"] = getListFromXml(response["data"],
"Bucket")
       return response
```

Terraform Code Dependencies

Create the bucket...

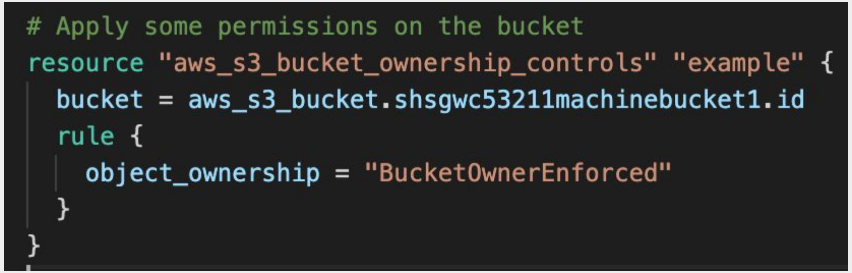


```
# Create the bucket
resource "aws_s3_bucket" "shsgwc53211machinebucket1" {
  2 references
  bucket = "shsgwc53211machinebucket1"
  policy = file("policy.json")
  website {
    index_document = "index.html"
  }
}
```

`aws_s3_bucket_ownership_controls` depends on `aws_s3_bucket`

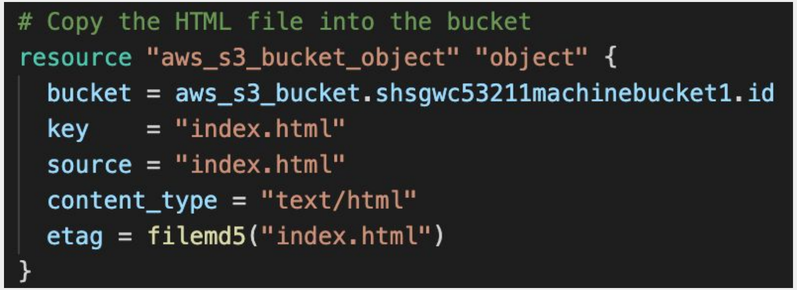
`aws_s3_bucket_object` depends on `aws_s3_bucket`

Apply permissions to the bucket...



```
# Apply some permissions on the bucket
resource "aws_s3_bucket_ownership_controls" "example" {
  bucket = aws_s3_bucket.shsgwc53211machinebucket1.id
  rule {
    object_ownership = "BucketOwnerEnforced"
  }
}
```

Upload file to the bucket...



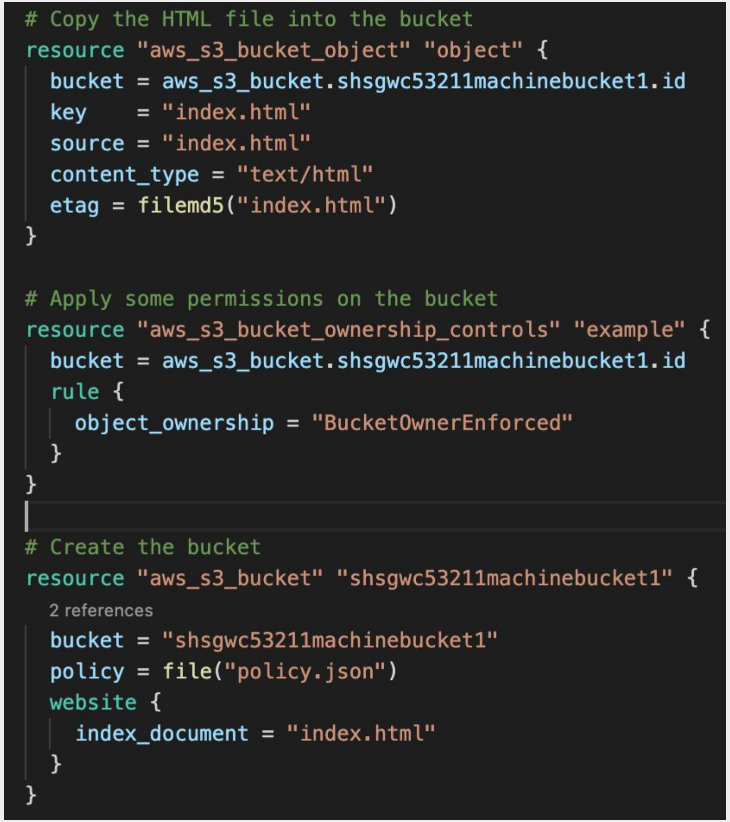
```
# Copy the HTML file into the bucket
resource "aws_s3_bucket_object" "object" {
  bucket = aws_s3_bucket.shsgwc53211machinebucket1.id
  key    = "index.html"
  source = "index.html"
  content_type = "text/html"
  etag = filemd5("index.html")
}
```

The Terraform Language

This code looks out of order.

It doesn’t read top to bottom like a procedural language would.

How do you think Terraform will execute this code?



```
# Copy the HTML file into the bucket
resource "aws_s3_bucket_object" "object" {
  bucket = aws_s3_bucket.shsgwc53211machinebucket1.id
  key    = "index.html"
  source = "index.html"
  content_type = "text/html"
  etag = filemd5("index.html")
}

# Apply some permissions on the bucket
resource "aws_s3_bucket_ownership_controls" "example" {
  bucket = aws_s3_bucket.shsgwc53211machinebucket1.id
  rule {
    object_ownership = "BucketOwnerEnforced"
  }
}

# Create the bucket
resource "aws_s3_bucket" "shsgwc53211machinebucket1" {
  2 references
  bucket = "shsgwc53211machinebucket1"
  policy = file("policy.json")
  website {
    index_document = "index.html"
  }
}
```

Terraform Execution

Terraform uses a file called a “plan” to hold the state of the resources you want to create
The “plan” command creates this plan file.
The “apply” command creates the resources that you specified on AWS.
The “destroy” command deletes the resources.
If you change your resources, terraform will compare what’s in the plan file to your code. When you run “apply” again it will apply your changes to the AWS resources.

Command	**What it does**
`terraform plan -out plan.out`	Creates or updates the plan file from your code.
`terraform apply plan.out`	Applies the plan file to the AWS resources.
`terraform destroy`	Deletes all AWS resources found in the plan file.

Terraform Demonstration

Join the Shorewood Mountain Bike Team

https://www.shorewoodmtb.org