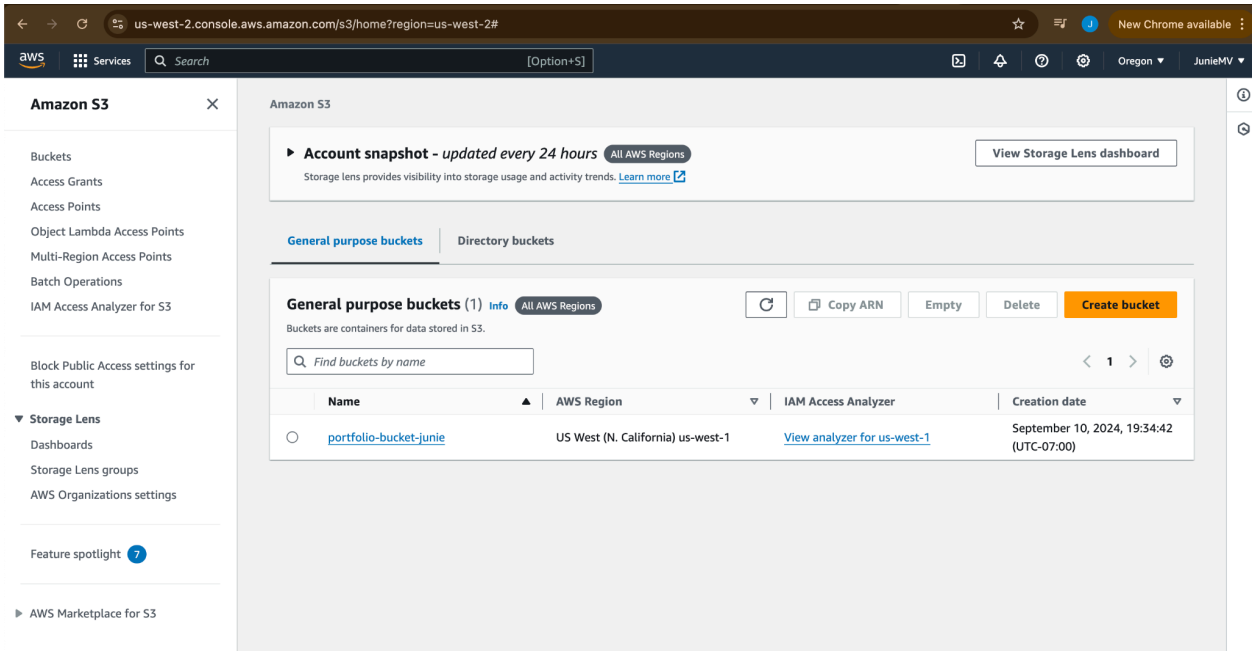
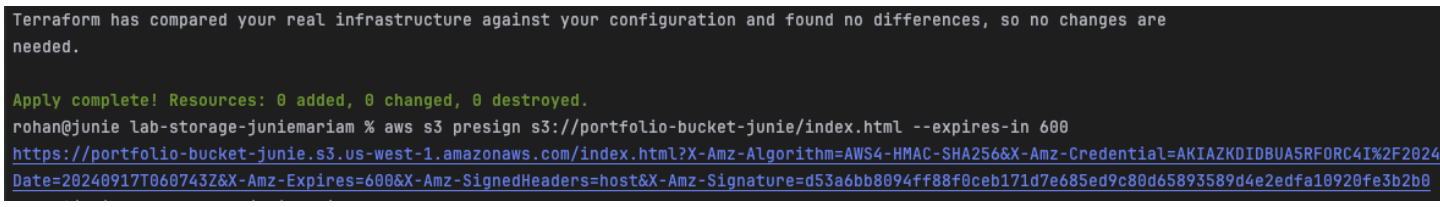
Screenshots of Lab-Storage

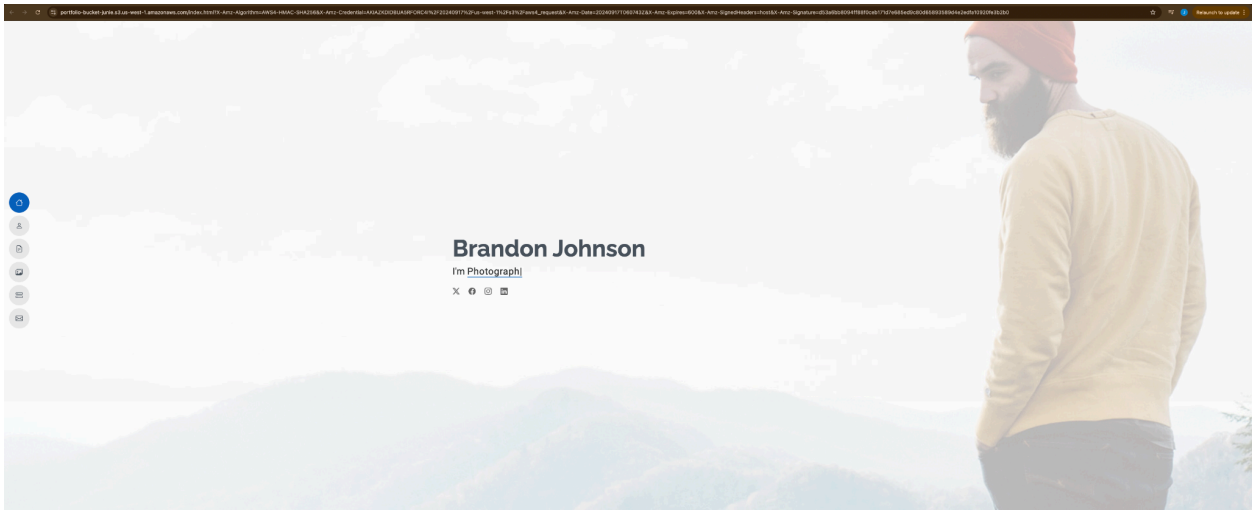
Task 1: Create a New S3 Bucket and Upload a Folder (1 pt)

1. Made necessary changes in S3.tf
2. Ran terraform init plan and apply
3. And thus S3 is created.



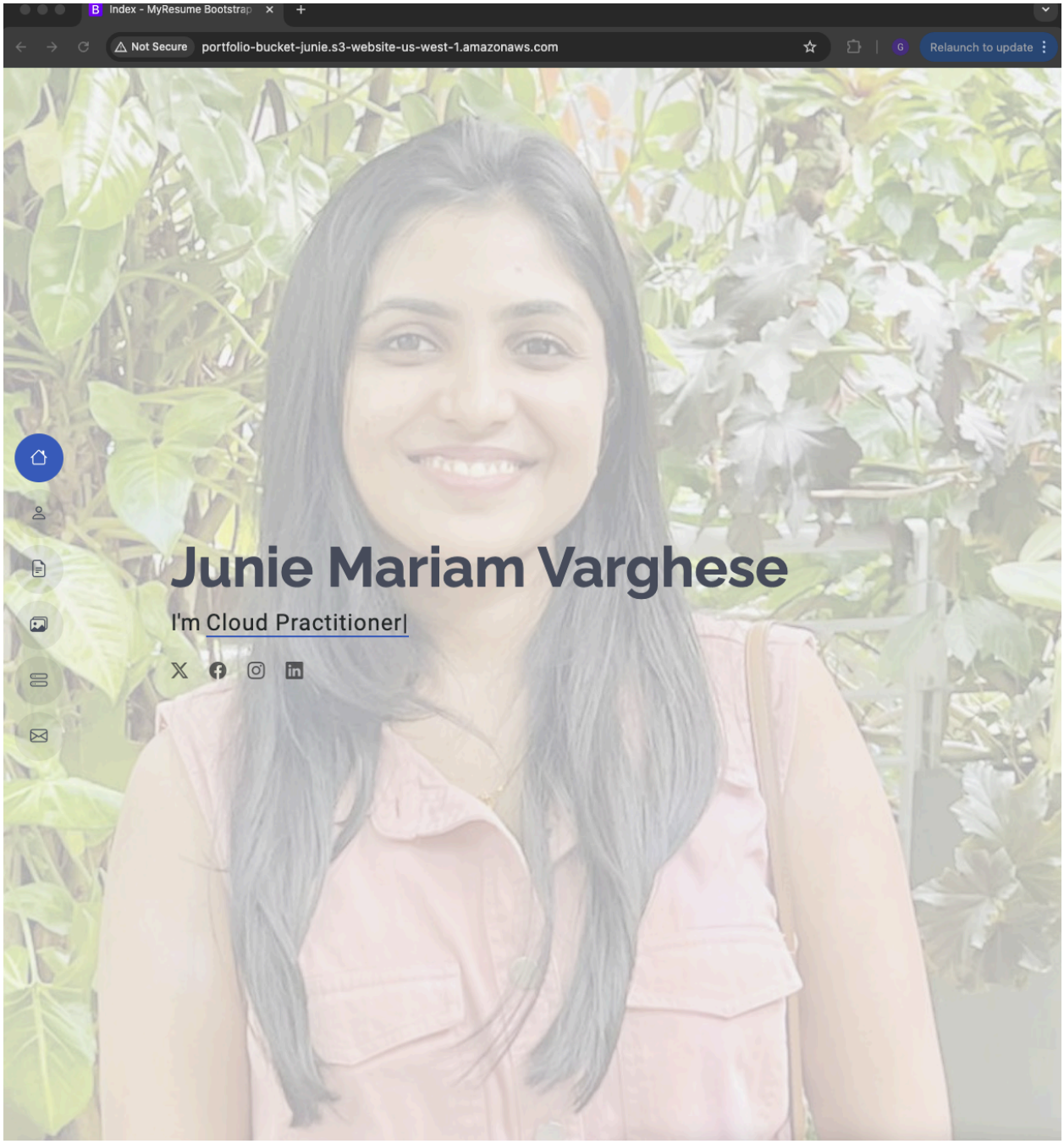
Task 2: Generate and Use a Pre Signed URL (3 pts)





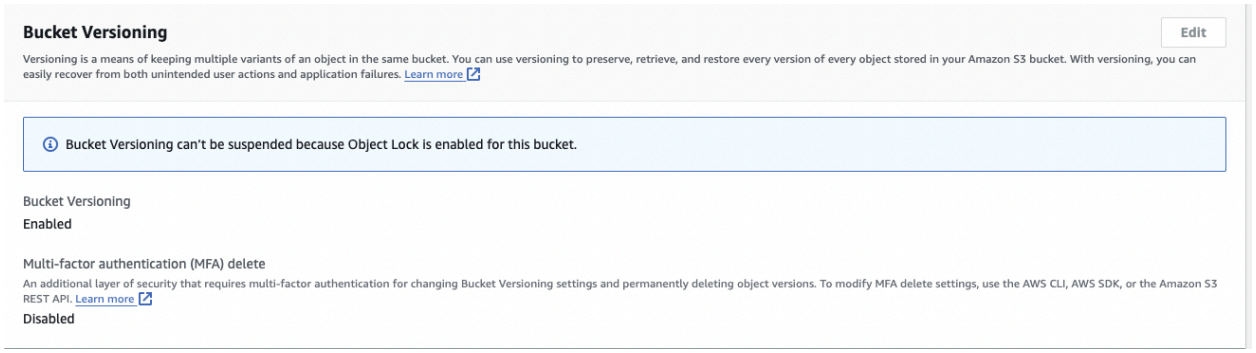
Task 3: Enable Static Website Hosting for an S3 Bucket (1 pt)

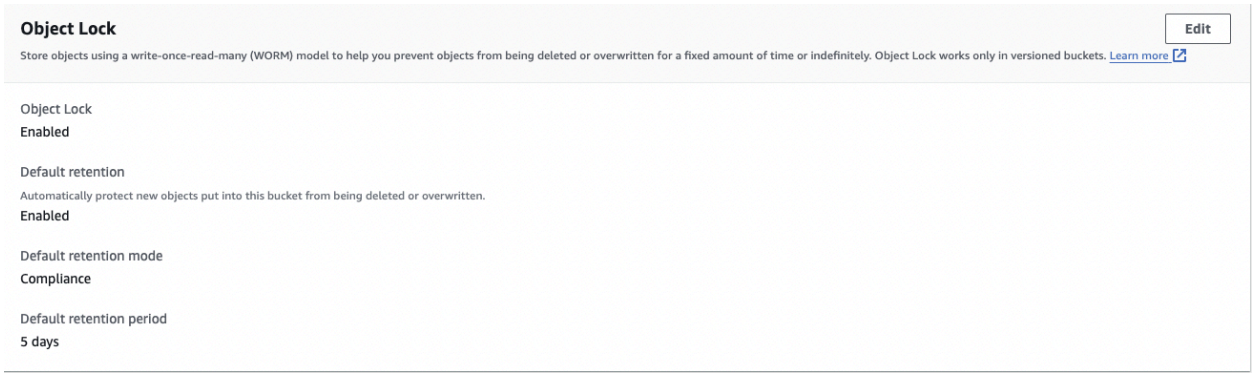
```
Apply complete! Resources: 0 added, 1 changed, 0 destroyed.

Outputs:

endpoint = "portfolio-bucket-junie.s3-website-us-west-1.amazonaws.com"
rohan@junie lab-storage-juniemariam %
```


Index - MyResume Bootstrap
Not Secure
portfolio-bucket-junie.s3-website-us-west-1.amazonaws.com
Relaunch to update
Junie Mariam Varghese
I'm Cloud Practitioner|

Task 4: Enable Versioning, Object Lock, and Lifecycle Management for your S3 Bucket (2 pts)

```
Outputs:

bucket_id = "portfolio-bucket-junie"
endpoint = "portfolio-bucket-junie.s3-website-us-west-1.amazonaws.com"
rohan@junie lab-storage-juniemariam %
```





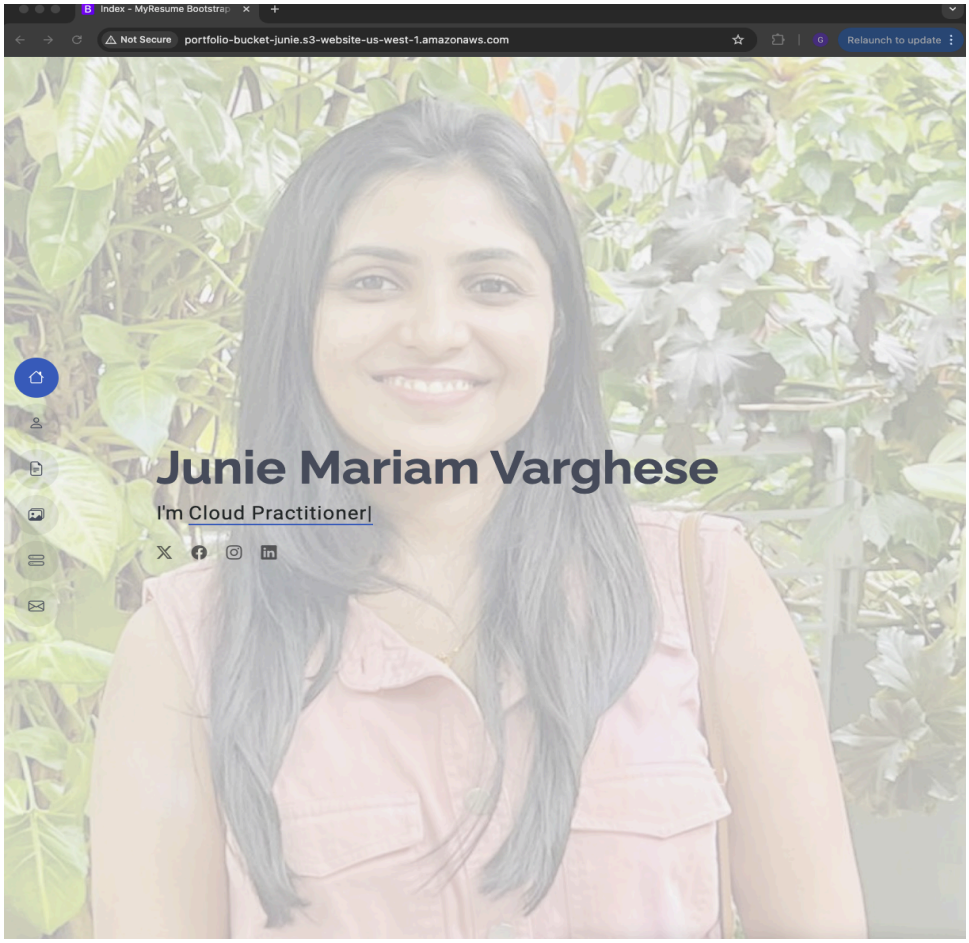
Task 5:Extra Credit:

1. Use the data aws_caller_identity resource to add your account id to your bucket name

```
# Fetch AWS account details
data "aws_caller_identity" "current" {}

output "account_id" {
  value = data.aws_caller_identity.current.account_id
}

output "caller_arn" {
  value = data.aws_caller_identity.current.arn
}

output "caller_user" {
  value = data.aws_caller_identity.current.user_id
}
```

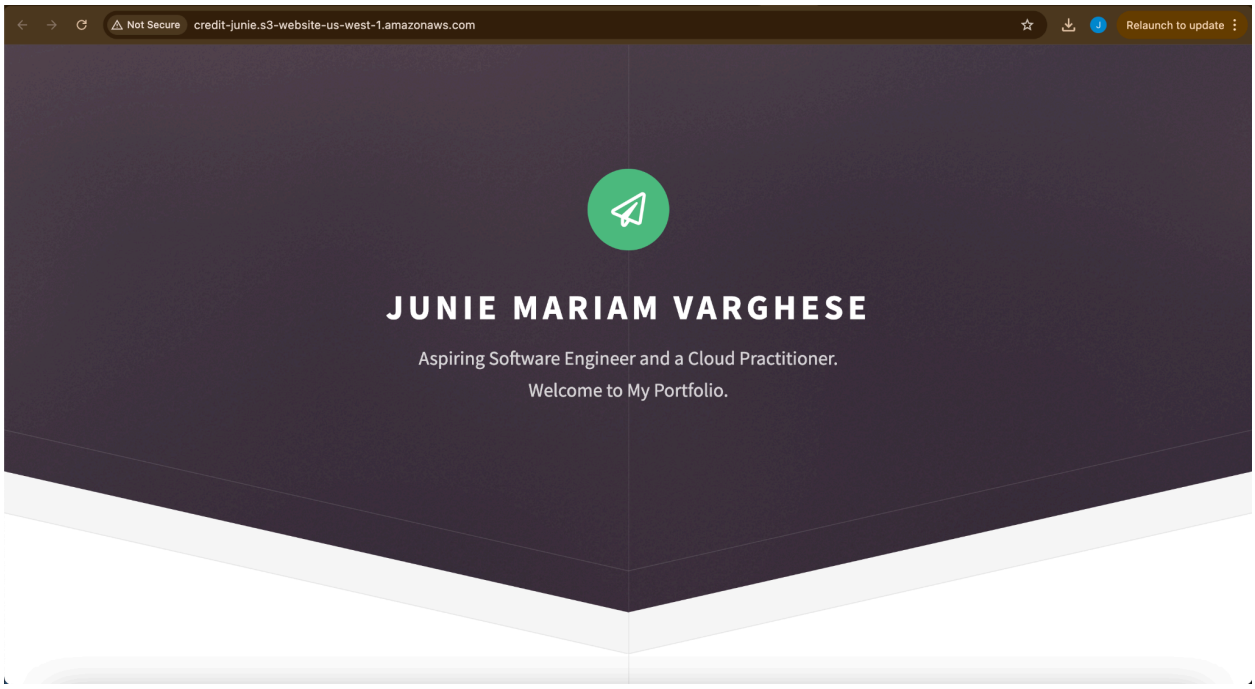
```
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.

Outputs:

account_id = "640168430849"
bucket_id = "portfolio-bucket-junie"
caller_arn = "arn:aws:iam::640168430849:user/JunieMV"
caller_user = "AIDAZKDIDBUA7ATL244R5"
endpoint = "portfolio-bucket-junie.s3-website-us-west-1.amazonaws.com"
```

2. Make it your own by filling out your own portfolio and using your own images.

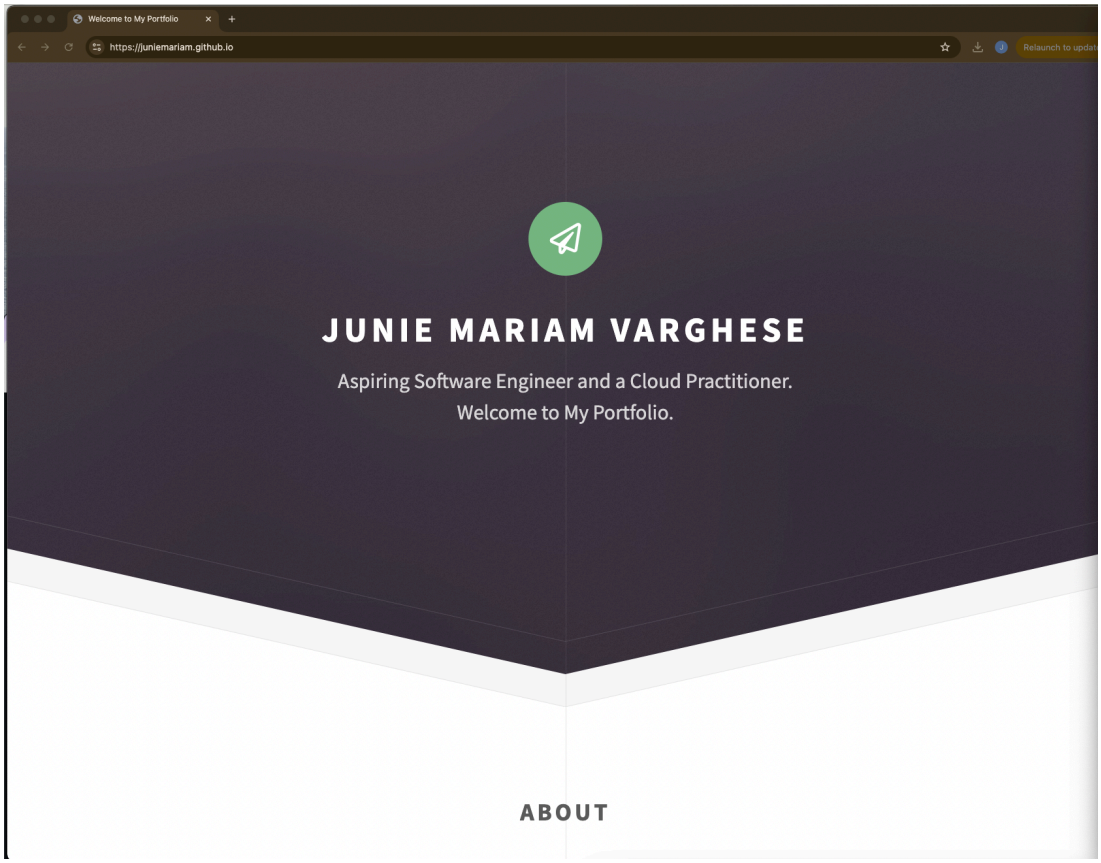


3. Use a different scheme from one of these templates.

   http://credit-junie.s3-website-us-west-1.amazonaws.com



4.https://juniemariam.github.io/



Knowledge Check (3 pts)

1. Explain how we use a single resource definition in task 1 to upload multiple files.

   You can upload multiple files with a single resource by setting it up to handle all the files in a folder at once. Instead of creating separate entries for each file, you just point the resource to the folder. This way, all the files in that folder get uploaded in one go, making things much simpler and quicker.

```
resource "aws_s3_bucket" "example_bucket" {
  bucket = "my-example-bucket"
}

resource "aws_s3_bucket_object" "upload_files" {
  for_each = fileset("${path.module}/files", "*")

  bucket = aws_s3_bucket.example_bucket.bucket
  key    = each.value
  source = "${path.module}/files/${each.value}"
}
```

2. Explain why the bucket policy is required in task 2 to display the web-page properly.

   The bucket policy is needed to make sure the content in the bucket is publicly accessible for your website. Without it, people won't be able to see your web page because the access to the files would be blocked. The policy opens up the bucket so anyone on the internet can view your site.

3. What are the pros and cons of the resources you used in task 4?

   Versioning

   Advantages:
   - With versioning you can recover more easily from both unintended user actions and application failures.
   - Versioning-enabled buckets can help you recover objects from accidental deletion or overwrite

   Disadvantages:

- Keeping multiple versions of files uses more space, which can get expensive.
- With several versions of the same file, it can get tricky to keep everything organized.

Object Lock

Advantages:

- Prevents changes or deletions, which is helpful if you need to follow certain regulations.

- Your important data stays safe from being accidentally deleted or altered for as long as it's locked.

Disadvantages:

- Keeping locked versions around adds to your storage costs.

- Once a file is locked, it can't be changed or removed, which can limit how you manage your data.

Lifecycle Management

Advantages:

- It automatically moves files to cheaper storage options or deletes them when they're no longer needed, reducing costs.

- Managing large amounts of data becomes easier because it automates how you handle older or less-used files.

Disadvantages:

- You need to carefully set up the rules so they work the way you intend.

- Moving files to cheaper storage options can mean they'll take longer to access when needed.

4. Imagine you are managing an S3 bucket for a company that stores log files generated by their web applications. These log files are frequently accessed during the first 30 days after creation for analysis but are rarely accessed after that period. After 1 year, the logs don't need to be accessible immediately, but should be held for 5 years. Describe how you could use Amazon S3 Lifecycle Management to optimize storage costs for these log files. Include details on the lifecycle rules you would configure and how they would help reduce expenses.

Answer:
To manage your company's log files and save on storage costs, you can use Amazon S3 Lifecycle Management to automate the process. Start by storing the logs in S3 Standard for the first 30 days, which is ideal for frequently accessed data. In one month automatically transition the logs to S3 Intelligent-Tiering, so you only need to pay for what you use.

After a year, move the logs to S3 Glacier or S3 Glacier Deep Archive, which are more affordable options for long-term storage of data that's rarely accessed. Finally, set up a rule to delete the logs after 5 years. This approach saves you money by reducing storage costs over time, automates the management process, and ensures that you're not keeping data longer than necessary.

```
{
  "Rules": [
    {
      "ID": "MoveToIntelligentTiering",
      "Prefix": "logs/",
      "Status": "Enabled",
      "Transitions": [
        {
          "Days": 30,
          "StorageClass": "INTELLIGENT_TIERING"
        }
      ]
    },
    {
      "ID": "MoveToGlacierAfter1Year",
      "Prefix": "logs/",
      "Status": "Enabled",
      "Transitions": [
        {
          "Days": 365,
          "StorageClass": "GLACIER"
        }
      ]
    },
    {
      "ID": "DeleteAfter5Years",
      "Prefix": "logs/",
      "Status": "Enabled",
      "Expiration": {
        "Days": 1825
      }
    }
  ]
}
```