AZUBI
AFRICA

AzubiAfrica, AWS Cloud Training Program

Individual Project

Title: Cloud Formation and S3 bucket

By: Yidnekachew Tefera Gebresilasei

Advisor: Samrawit Ayalew

Ethiopia(Jan 3, 2024)

DECLARATION

I hereby declare that the project entitled **Cloud Formation and S3 Bucket.** is submitted to **Azubi Africa** under the supervision and guidance of **Ms. Samrawit Ayalew**.
The matter embodied in this work is original and has not been submitted for the award of any other degree.

Introduction

I am actively engaged in the deployment of this project on Amazon Web Service (Aws), designed to express the knowledge that I have and I am on the way to have.

IaC:Infrastructure as code, also referred to as IaC, is an IT practice that codifies and manages underlying IT infrastructure as software. The purpose of infrastructure as code is to enable developers or operations teams to automatically manage, monitor and provision resources, rather than manually configure discrete hardware devices and operating systems. Infrastructure as code is sometimes referred to as programmable or software-defined infrastructure.
Some Example of IaC are AWS CloudFormation and Terraform.

AWS CloudForamtion: AWS CloudFormation is a service that helps you model and set up your AWS resources so that you can spend less time managing those resources and more time focusing on your applications that run in AWS. You create a template that describes all the AWS resources that you want (like Amazon EC2 instances or Amazon RDS DB instances), and CloudFormation takes care of provisioning and configuring those resources for you. You don't need to individually create and configure AWS resources and figure out what's dependent on what; CloudFormation handles that. The following scenarios demonstrate how CloudFormation can help.

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.

On this scenario we are going to be using AWS CloudFormation for creating an S3 Bucket and perform some kind of functionality using AWS CloudFormation Script.

Objectives

- Creating an S3 Bucket and adding some functionality using AWS CloudFormation Template.
- Uploading simple html file into S3 Bucket.

1. The AWS CloudFormation Template

The following Image shows the script that I used to perform the task.

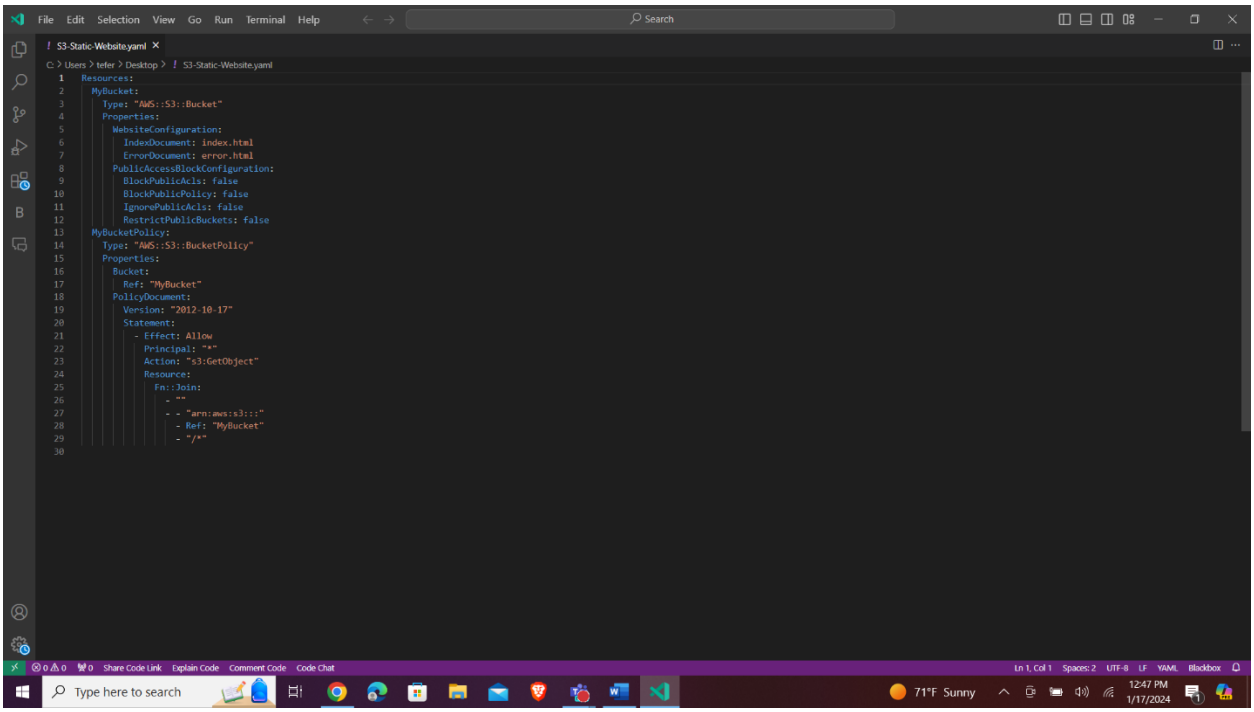


Figure 1

1.1 Code Explanation

The code we used is the extension type ".YAML" .

A. The first 3 line defines the Bucket name and the type of AWS Service and what service we are going to use what type of service we are going to use inside that service.

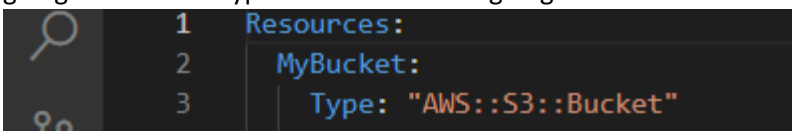


Figure 2

"MyBucket": used to define the name.
"Type": used to define the type of service we are going to use.

B. The code that is used for enabling the static web hosting and the default "IndexDocument" and "ErrorDocument".

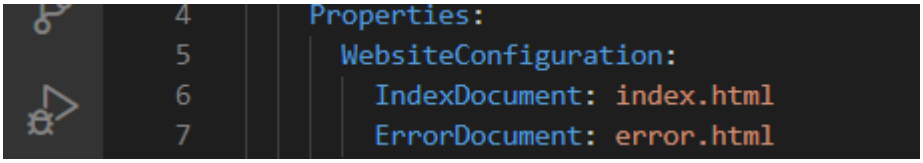


Figure 3

C. The following is the code I used for enabling public access.

```
PublicAccessBlockConfiguration:
  BlockPublicAcls: false
  BlockPublicPolicy: false
  IgnorePublicAcls: false
  RestrictPublicBuckets: false
```

Figure 4

D. The following is the Bucket policy.

```
13  MyBucketPolicy:
14    Type: "AWS::S3::BucketPolicy"
15    Properties:
16      Bucket:
17        Ref: "MyBucket"
18      PolicyDocument:
19        Version: "2012-10-17"
20        Statement:
21          - Effect: Allow
22            Principal: "*"
23            Action: "s3:GetObject"
24            Resource:
25              Fn::Join:
26                - ""
27                - - "arn:aws:s3:::"
28                  - Ref: "MyBucket"
29                  - "/*"
```

Figure 5

“Ref”: it is referring to the bucket name.

2. Creating AWS CloudFormation Stack.

On this scenario we are going to create a CloudFormation Stack .

2.1 Uploading our Template and Uploading the html file.

Since I created the template on my local machine I will be uploading it to my AWS cloud formation and create the stack.

i. Accessing the CloudForamtion:
Services->Management and Governance->CloudFormation

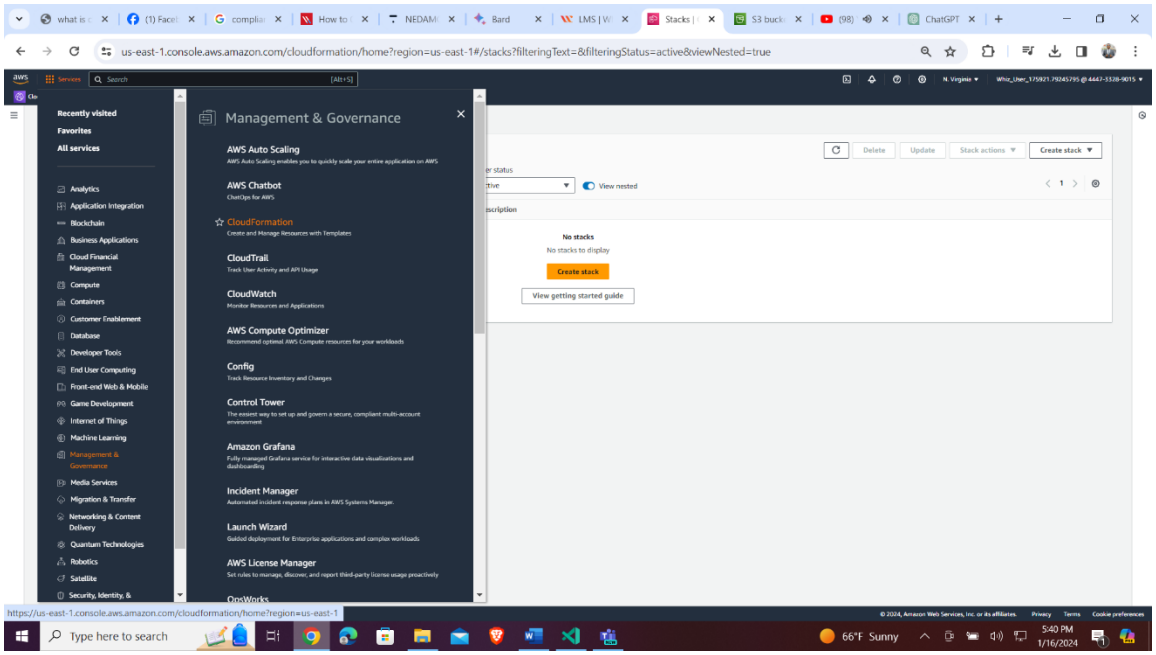


Figure 6

ii. Creating the stack:
Click on “Create stack”

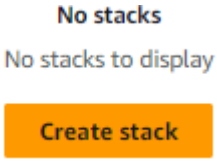
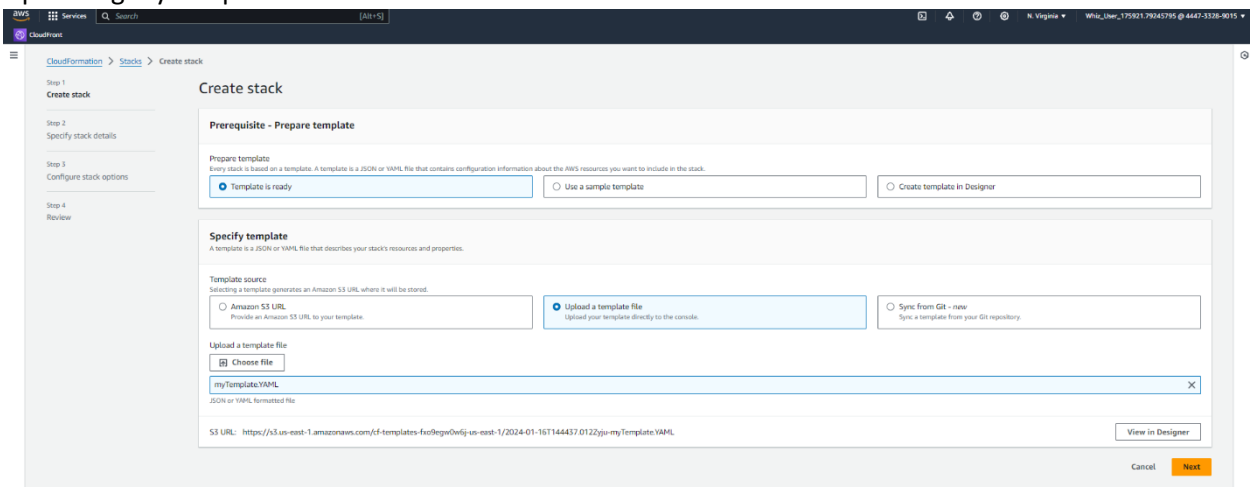


Figure 7

iii. Uploading my template



iv. Viewing in design

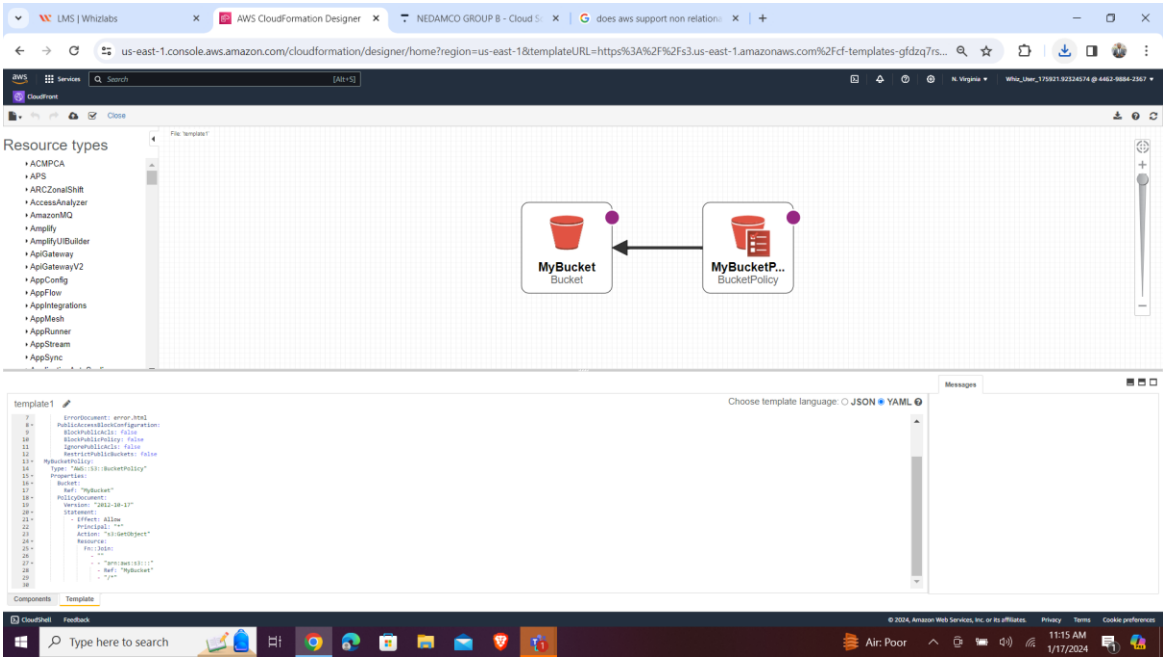


Figure 8

The following is the over all design.

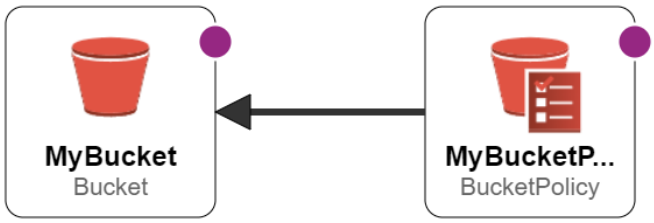


Figure 9

After viewing the detail click on the button named "Next".

v. Name our stack then click on Next.

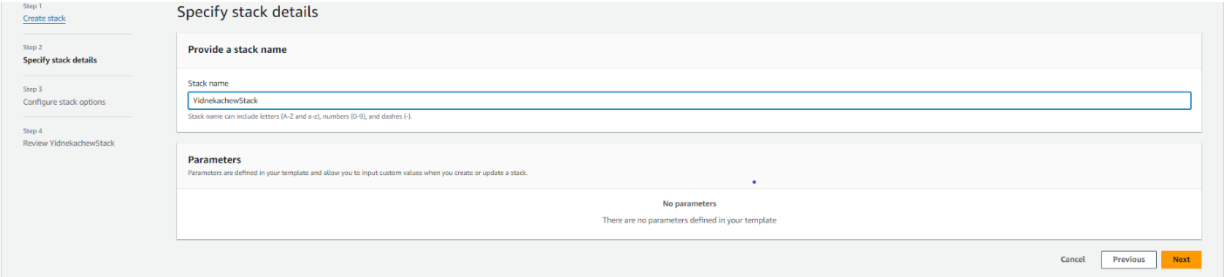


Figure 10

vi. Leave everything as it's default value and submit it.

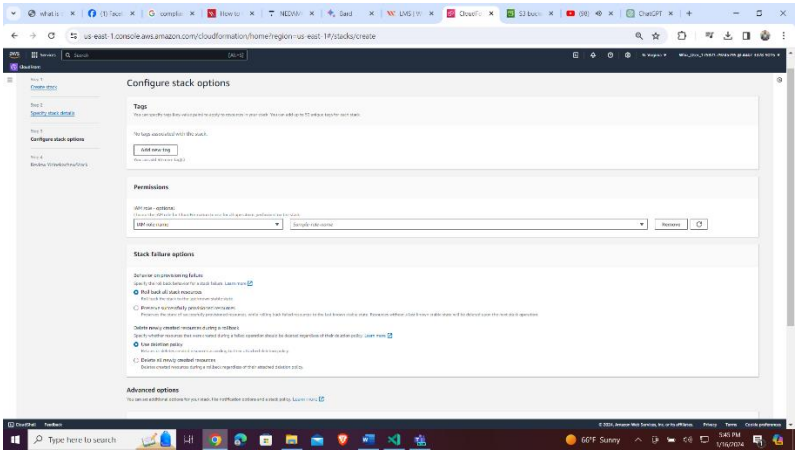


Figure 11

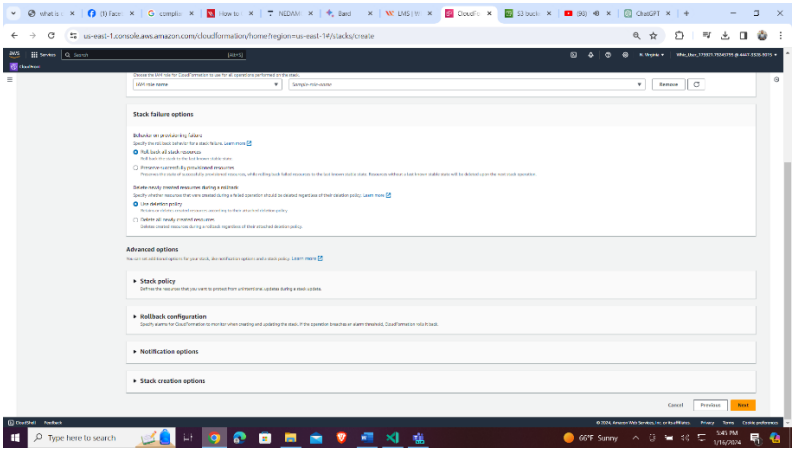


Figure 12

Then click on "Next".

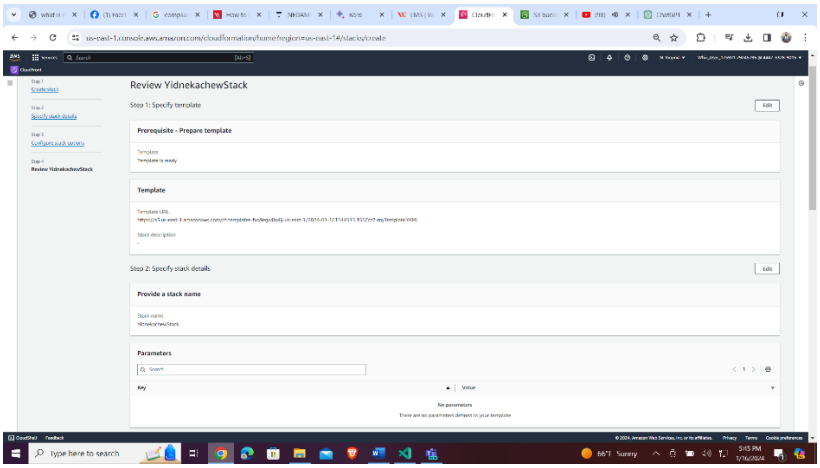


Figure 13

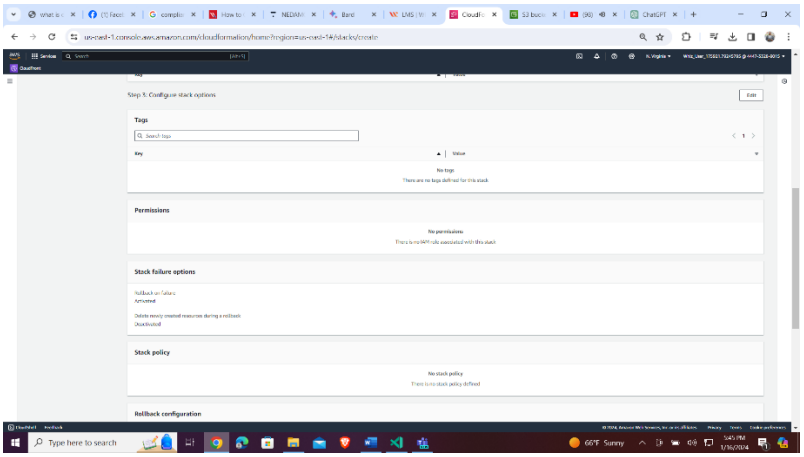


Figure 14

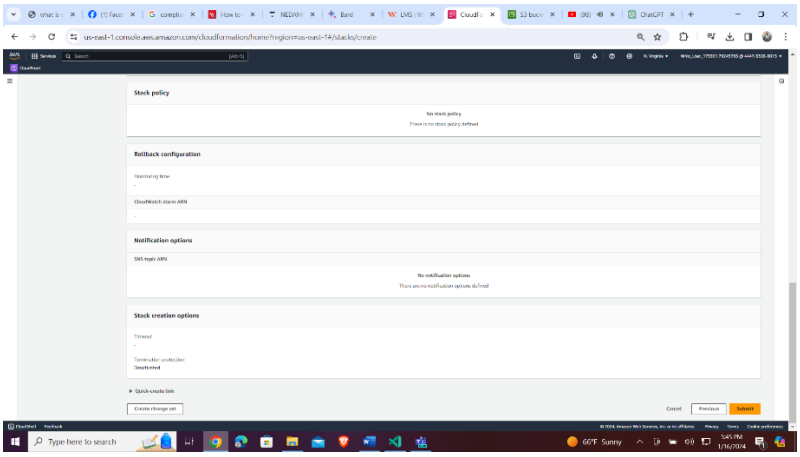
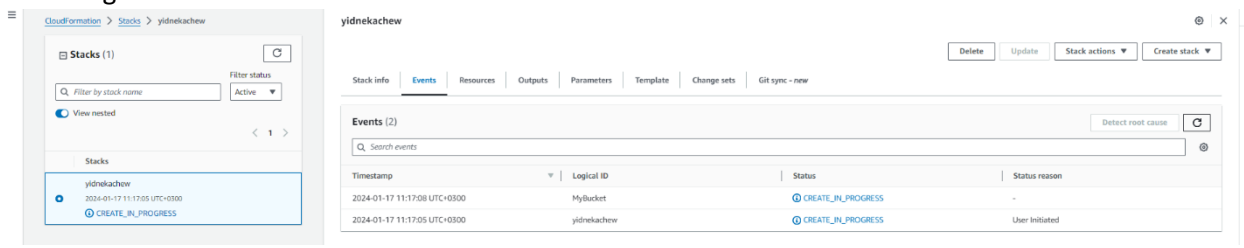


Figure 15

Click on “Submit”.

vii. Accessing the created stack.



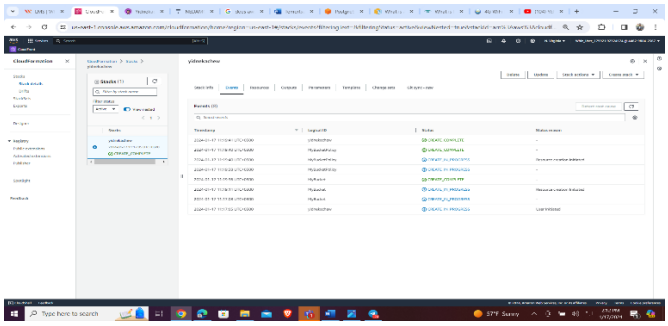


Figure 16

viii. Confirming everything we want to create and implement using AWS CloudFormation Template.

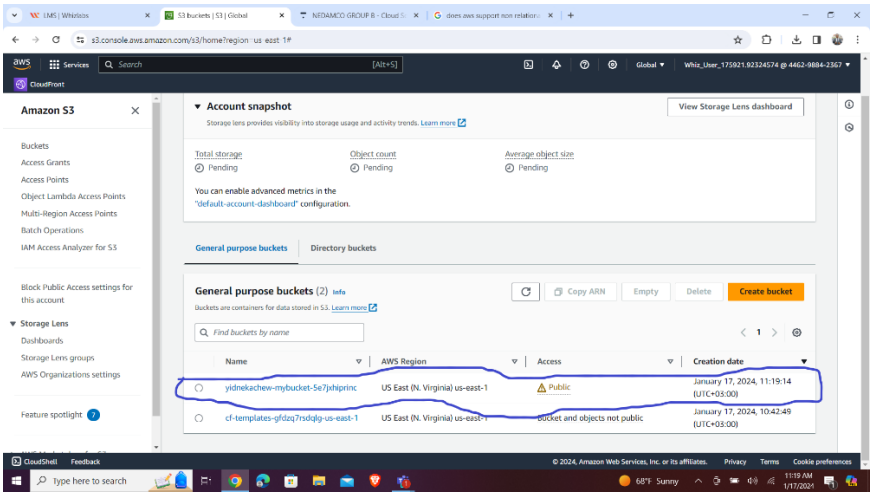


Figure 17

Here our bucket is created and block public access is disabled.

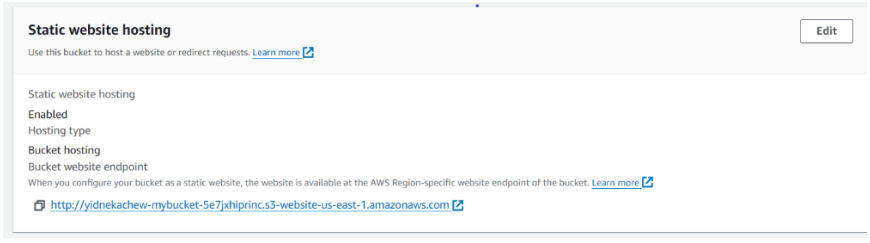


Figure 18

The above image shows the static web hosting is enabled.

ix. Uploading my simple website.

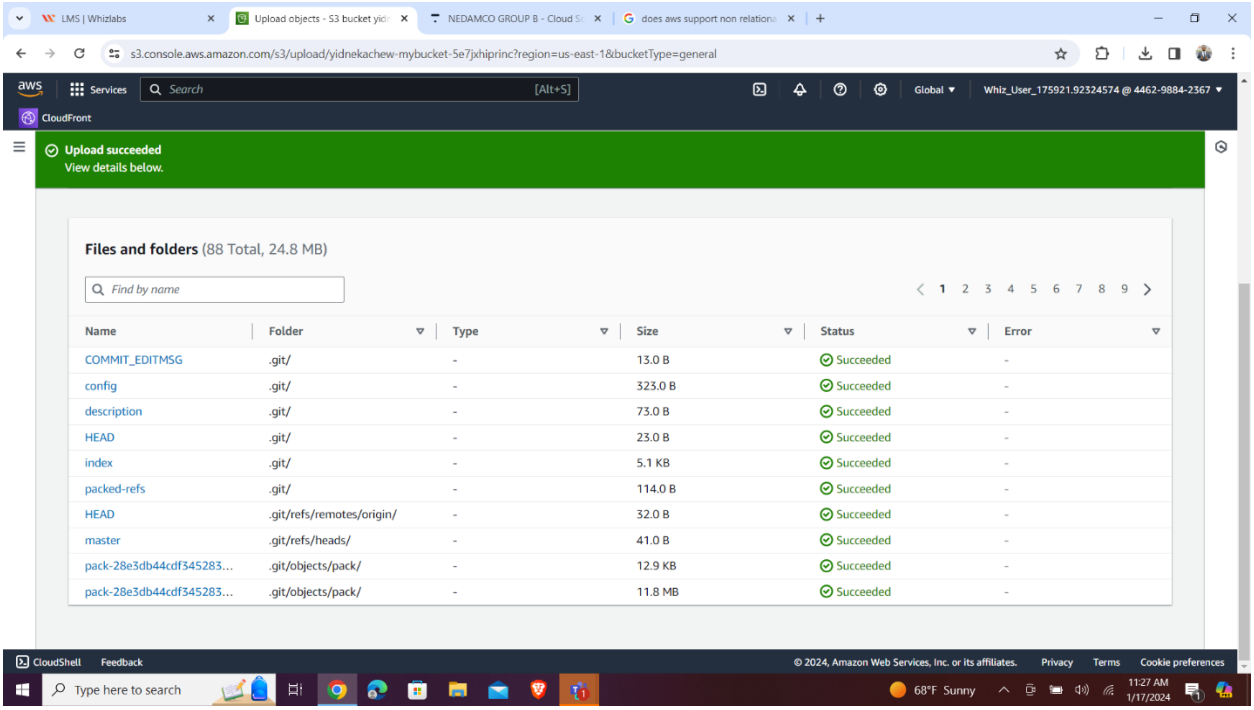


Figure 19

x. Accessing my website using the URL provided by static webhosting.

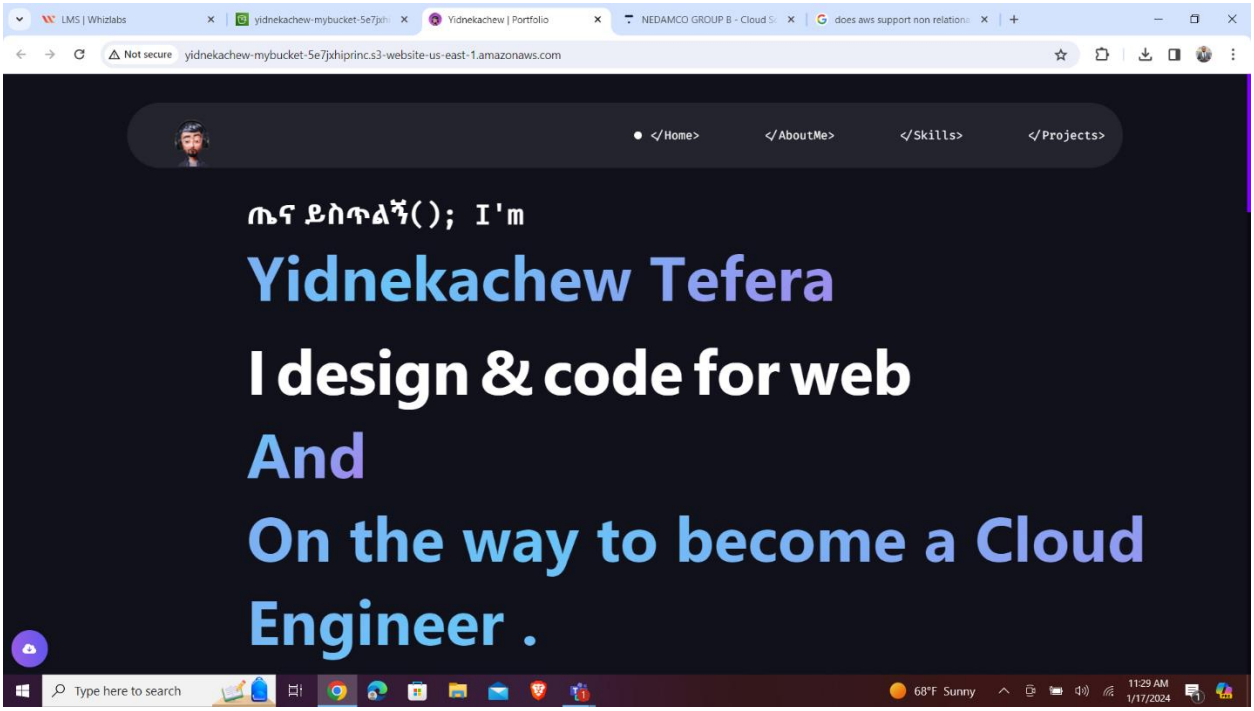


Figure 20

3. Conclusion

CloudFormation is a powerful tool for automating the creation and management of AWS resources, including S3 buckets. Here's a conclusion summarizing its benefits :

1. Infrastructure as code: Define your S3 bucket configuration in a YAML template, enabling version control, sharing, and repeatability.
2. Automation: Create and manage S3 buckets programmatically, eliminating manual configuration and reducing errors.
3. Scalability: Easily scale your S3 infrastructure by deploying multiple stacks or modifying existing templates.
4. Consistency: Ensure consistent configurations across your infrastructure by applying the same template across environments.
5. Integration: Combine S3 creation with other AWS resources within a single template for complex deployments.