Cloud Computing Report

2024

Prepared by
KEERTHI TR

Table of Contents

Summary

AWS (Amazon Web Services) is a comprehensive cloud computing platform offered by Amazon. It provides a wide range of services that allow individuals, businesses, and organizations to build, deploy, and manage applications and infrastructure without the need for physical hardware.

Common AWS Use Cases:

- Web Hosting: Host websites and web applications.
- Storage: Backup and store data securely in the cloud.
- Application Deployment: Run applications without managing the underlying infrastructure.
- Big Data: Process and analyze large datasets using AWS analytics tools.
- Machine Learning: Build and train machine learning models using AWS services like SageMaker.

Key Features of AWS:

1. Cloud Computing:
   - Instead of managing physical servers and data centers, AWS offers virtualized resources that can be accessed over the internet. These resources can scale as needed, offering flexibility and cost savings.
2. Global Infrastructure:
   - AWS operates data centers around the world, providing users with the ability to deploy applications and services globally with low-latency access.
3. Wide Range of Services:
   - Compute: Services like Amazon EC2 (virtual servers) for running applications.
   - Storage: Services like Amazon S3 (object storage) for storing data, backups, and hosting static websites.
   - Databases: Managed database services like Amazon RDS and DynamoDB.
   - Networking: Tools for managing network connections, including VPC (Virtual Private Cloud).
   - Security: Services to ensure secure access and encryption of data.
4. Pay-As-You-Go:
   - AWS offers a pricing model where users only pay for the resources they use. This allows for efficient cost management as users can scale their resources up or down depending on demand.

Creating AWS Account

Goal:

To create a cloud account on AWS and log in.

Steps:

1. Sign Up for AWS:
   - Navigate to aws.amazon.com and click "Create an AWS Account."
   - Fill in the required details (email, password) and choose the Free Tier to avoid costs.
   - Complete account setup by providing billing information and verifying your identity.
2. Log in to AWS:
   - After successfully creating an account, log in to the AWS Management Console using your credentials.

Develop or Clone a Small Web Application

Goal:

To create or clone a simple static website using HTML.

Options:

1. Custom Coding:
   - Create a simple HTML file (e.g., index.html) with basic website content (e.g., "Welcome to My Website" with an image and text).
2. Cloning a Website:
   - Use tools like HTTrack or Darcy Ripper to clone an existing static website from the internet.
   - Customize or review the cloned website files if needed.

Upload the Static Website to AWS S3

Goal:

To upload the website files (HTML, CSS, images) to an S3 bucket on AWS.

Steps:

1. Create an S3 Bucket:
    - Go to the S3 service in the AWS Management Console.
    - Click on "Create Bucket."
    - Choose a globally unique bucket name (e.g., my-static-website).
    - Choose a region (e.g., us-east-1).
    - Disable Block Public Access if you want the website to be public.
    - Leave other settings as default and click Create Bucket.
2. Upload Files:
    - Select the created bucket and go to the Objects tab.
    - Click Upload and select the website files (index.html, images, CSS, etc.) from your computer.
    - Once the files are uploaded, your content will be stored in the S3 bucket.

Configure the Bucket for Static Website Hosting

Goal:

To enable the S3 bucket for static website hosting and configure it to be publicly accessible.

Steps:

1. Enable Static Website Hosting:
   - In the Properties tab of the bucket, scroll down to the Static Website Hosting section.
   - Click Edit and enable Static Website Hosting.
   - Enter index.html as the Index Document.
   - Enter error.html (or any other error page) as the Error Document.
   - Click Save Changes.

Set Permissions:

- **Navigate to the Permissions tab.**
- **Edit the Bucket Policy to allow public read access by entering the following JSON policy:**

json

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

- Replace my-static-website with your actual bucket name. This will allow anyone to view the files.
- Make Files Public:
- Ensure that the individual files (e.g., index.html) are set to public by selecting each file, clicking Actions, and then choosing Make Public.

Make the Website Accessible to the Public

Steps:

1. Get the Website URL:
    - Go to the Properties tab and look at the Static Website Hosting section.
    - Copy the endpoint URL provided (e.g., http://my-static-website.s3-website-us-east-1.amazonaws.com).

2. Test the Website:
    - Open the endpoint URL in a browser.
    - You should see the homepage (index.html) of your website. Test the error document by entering an invalid URL (e.g., http://my-static-website.s3-website-us-east-1.amazonaws.com/nonexistent.html) to verify the custom error page is displayed.

Configure Index Document and Error Document

Goal:

To configure an index and error document for their static website.

Steps:

- Ensure the correct documents are specified in the Static Website Hosting settings.
- The Index Document should be set to index.html, and the Error Document should be set to error.html.
- These settings determine how the website behaves when accessed by users or when an error occurs.

Analyze Monitoring and Logging

Goal: To enable logging and monitoring to analyze website activity.

Steps:

- Enable S3 Access Logs:
- In the Properties tab of the S3 bucket, enable Server Access Logging.
- Specify another S3 bucket to store the logs of all access requests to your website.
- Use AWS CloudWatch:
- Set up CloudWatch Metrics to monitor S3 usage.
- CloudWatch can track request rates, latency, and errors.
- This provides insights into how the website is being accessed and whether there are any performance issues.

Clean Up Resources

Goal:

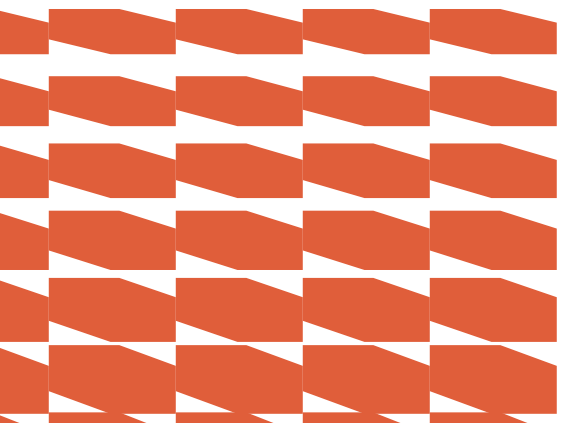
To ensure that all resources are cleaned up to avoid unnecessary charges after the project is completed.

Steps:

1. Delete S3 Bucket:
    - Go to the S3 dashboard, select the bucket, and delete it.
    - Ensure all contents inside the bucket are deleted as well.
2. Disable Logging and Monitoring:
    - Turn off CloudWatch metrics and remove the S3 access logging setup.
    - This will ensure you're not incurring extra costs for unused resources.

Conclusion

We has now successfully created and hosted a static website on AWS S3, configured public access, enabled monitoring, and cleaned up after the project. These steps provide an understanding of cloud hosting, static website deployment, and monitoring practices on AWS.



POC :

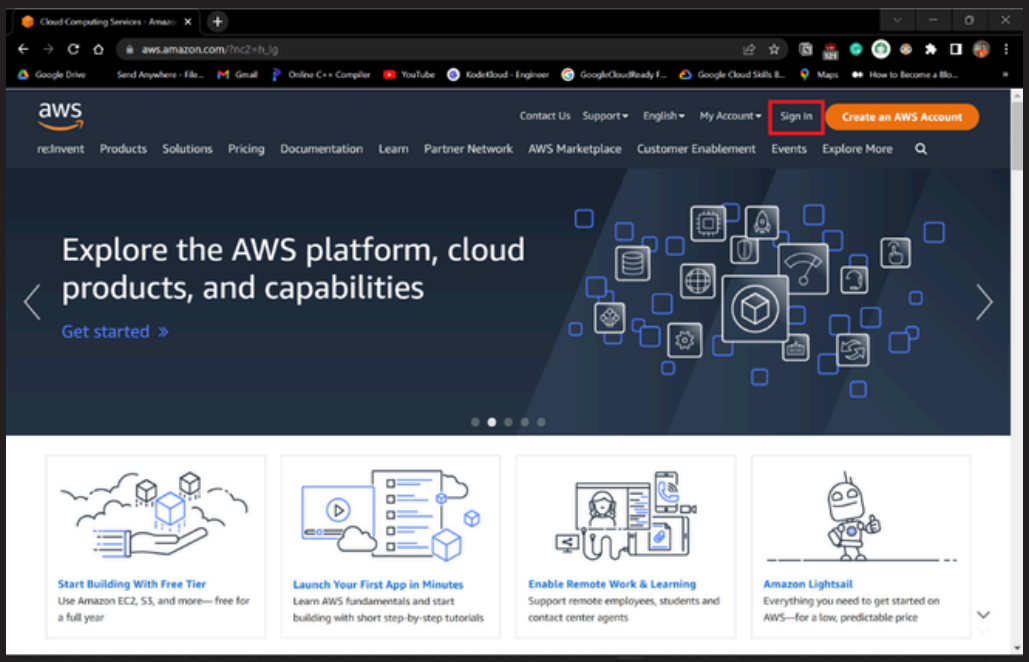


fig 11.1: Creating AWS Account

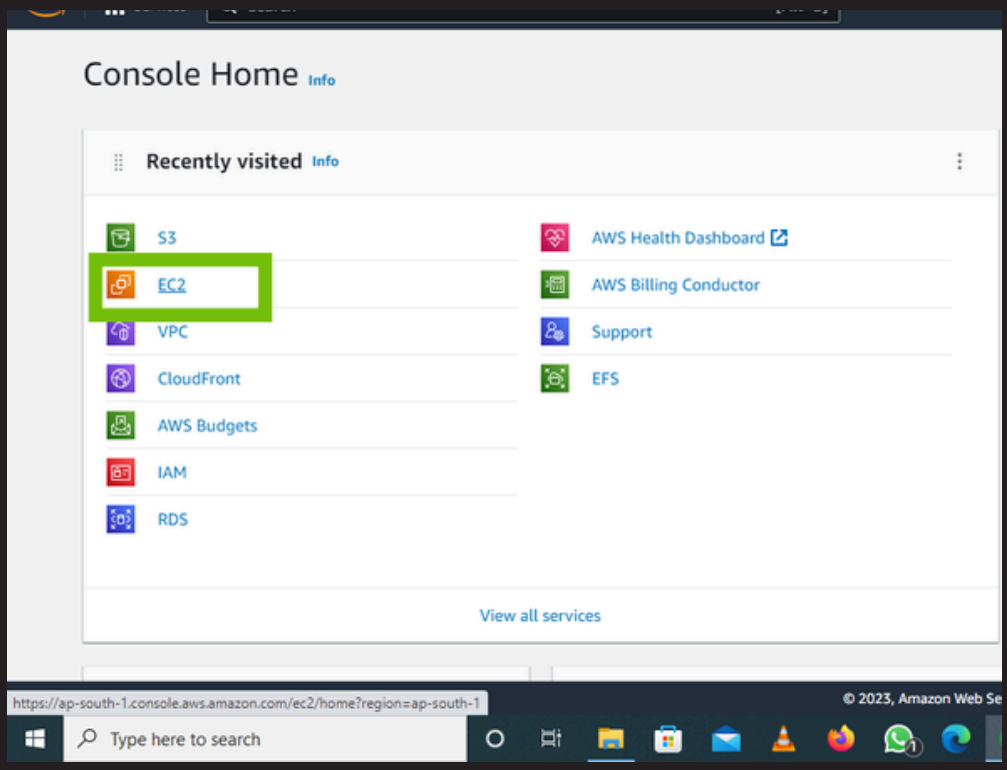


fig 11.2: Launching EC2

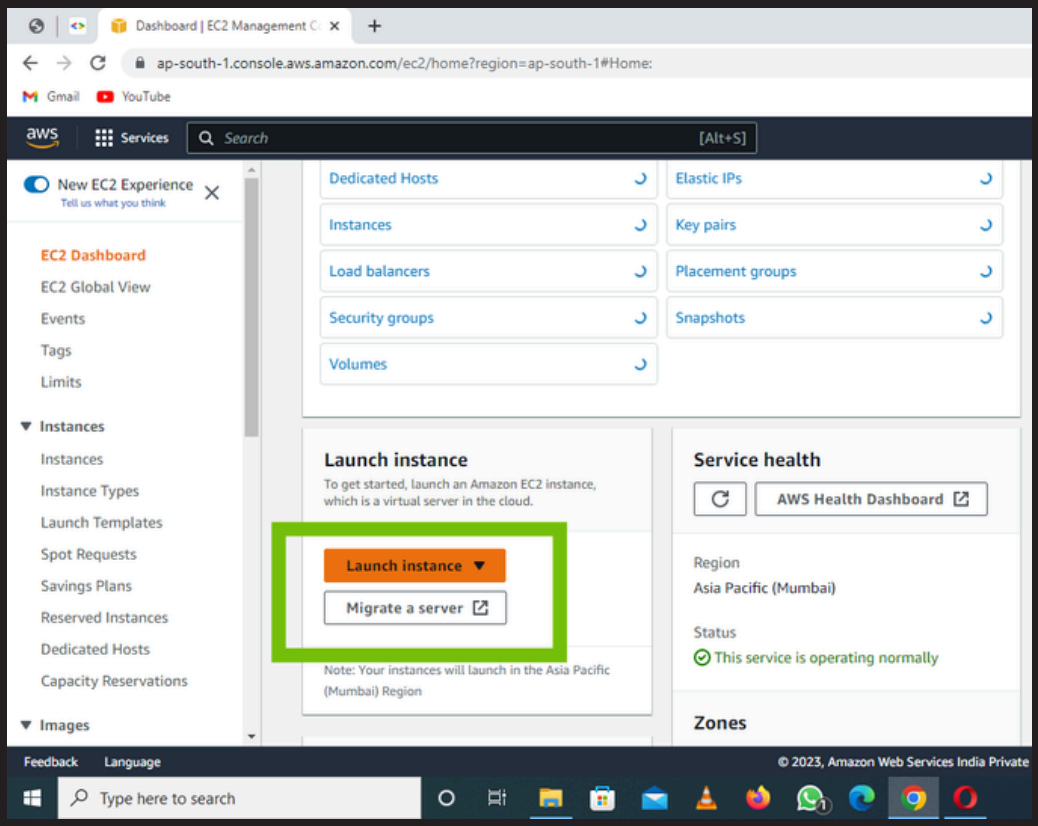


fig 11.3: Launching Instance

fig 11.5: Naming the Instance

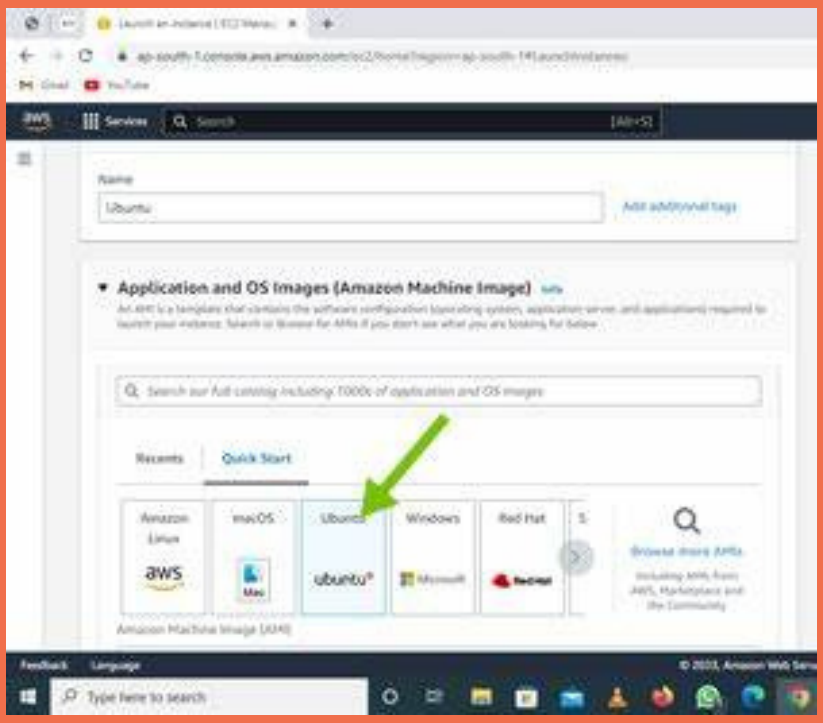


fig 11.5: Selecting the OS

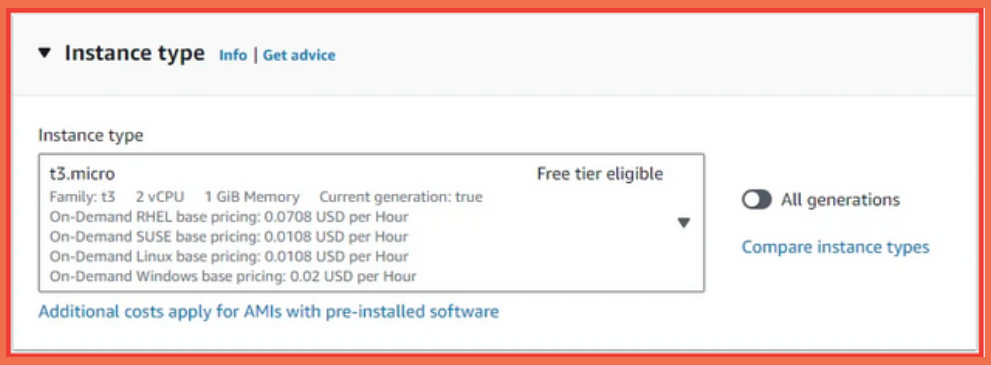


fig 11.6: Naming the instance type

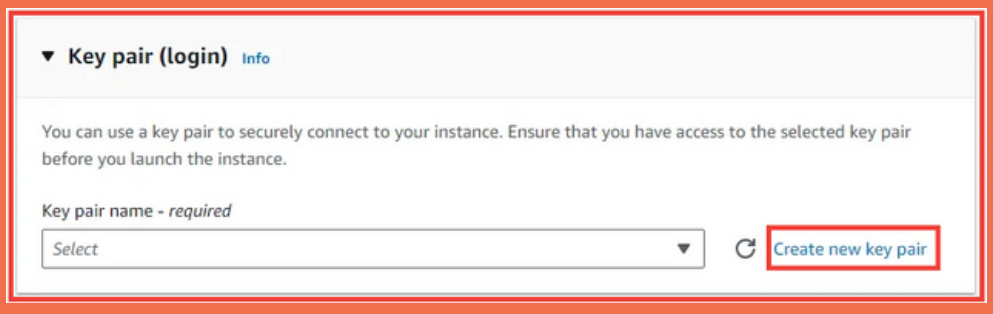


fig 11.7: Creating the key pair

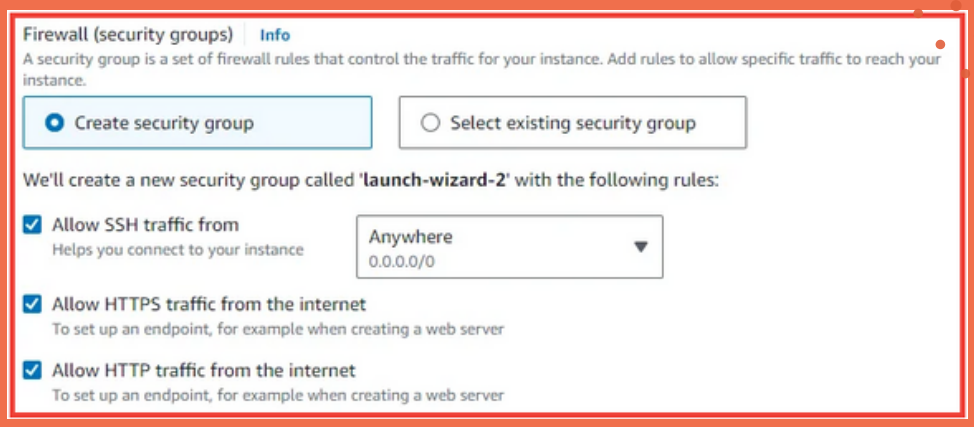


fig 11.8: Key pair

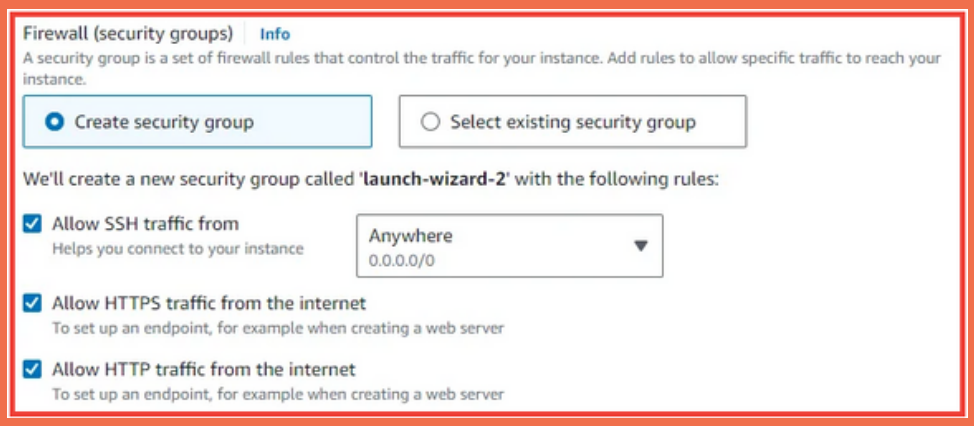


fig 11.9: Creating the security groups

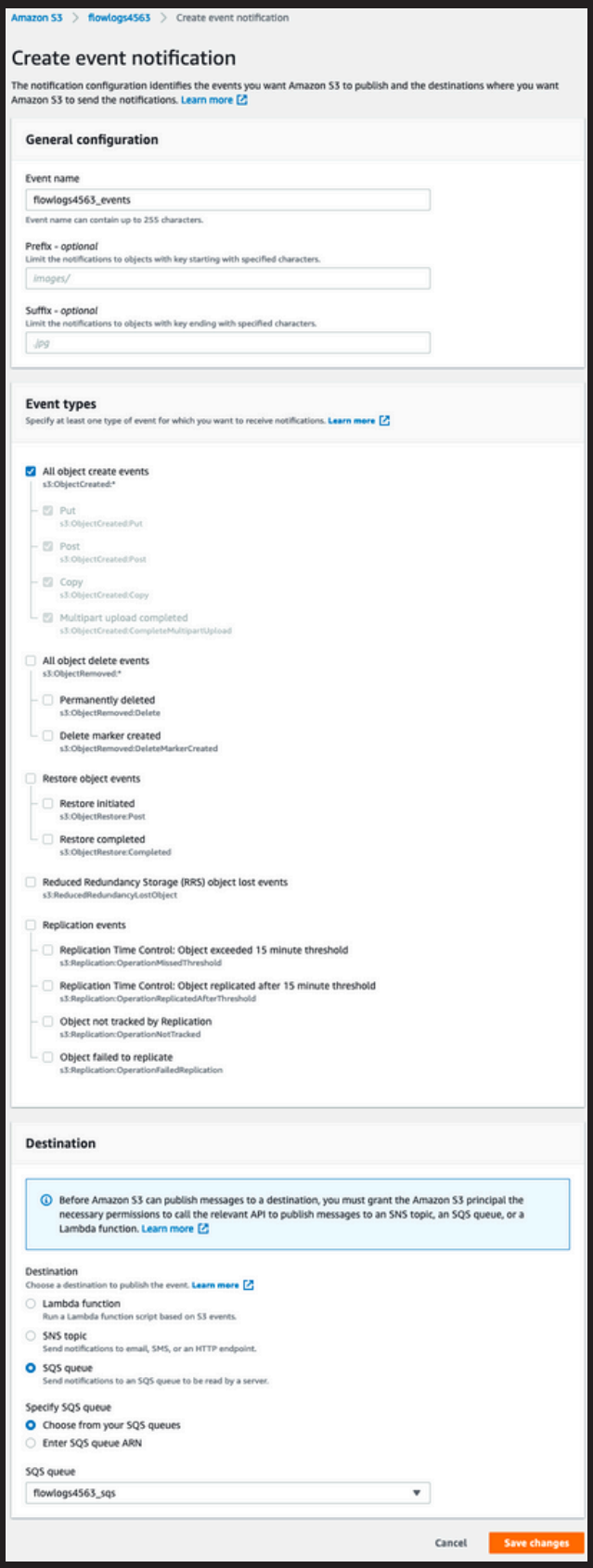


fig 11.10: Triggering the S3 logs

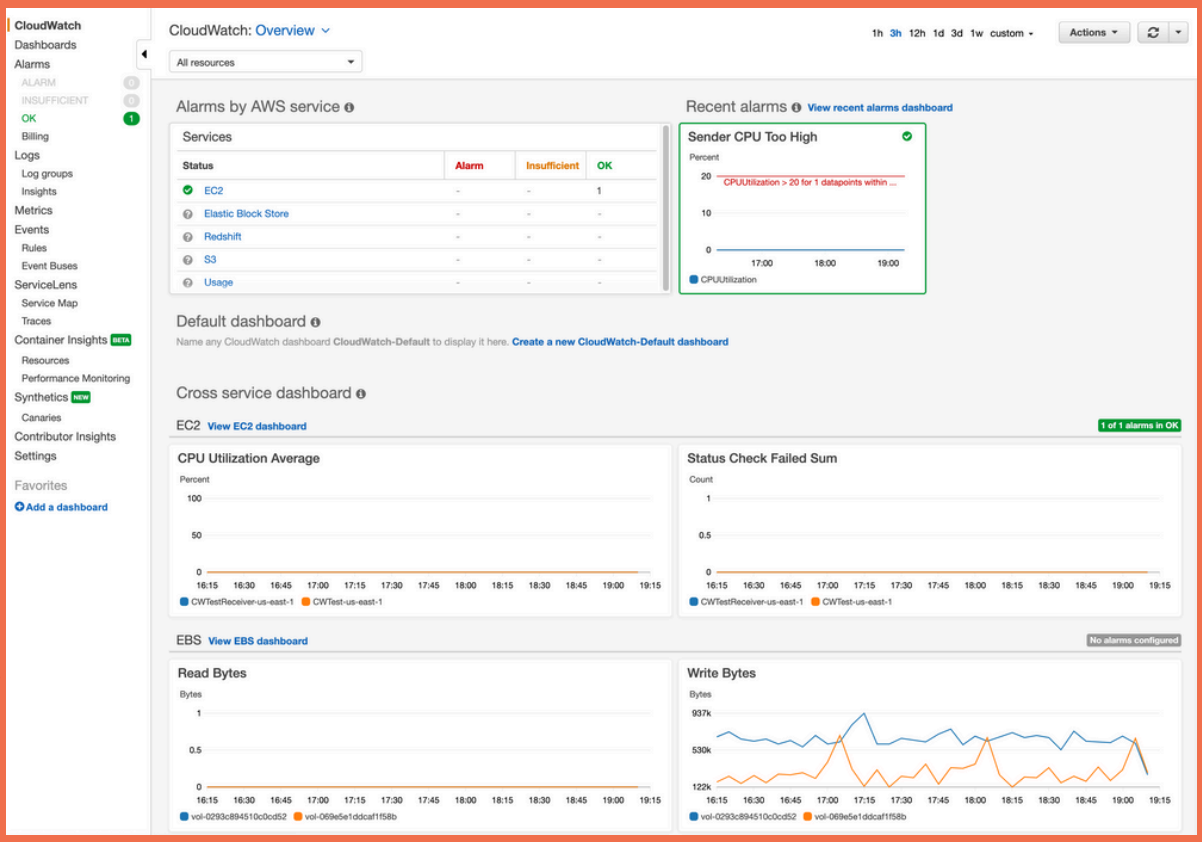


fig 11.11: Triggering the Cloudwatch

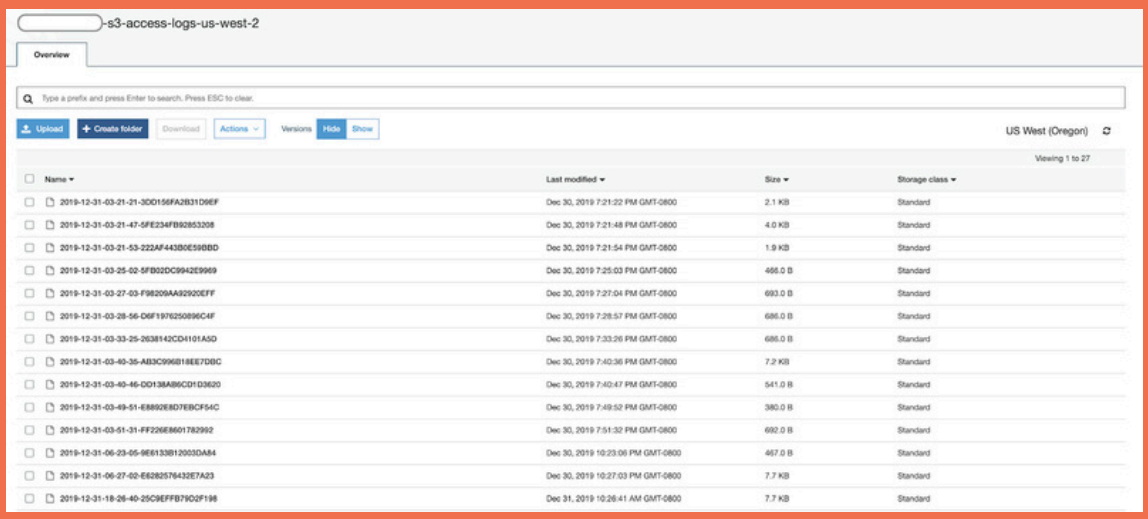


fig 11.12: S3 logs

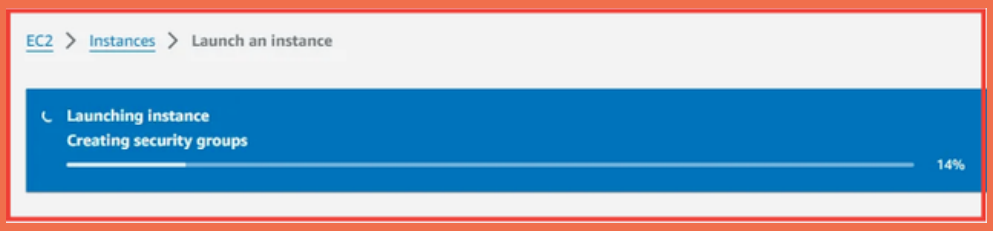


fig 11.13: Launching the instance

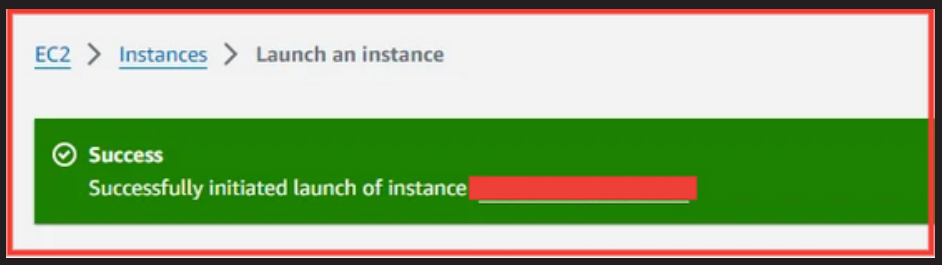


fig 11.14: Successfully launced instance

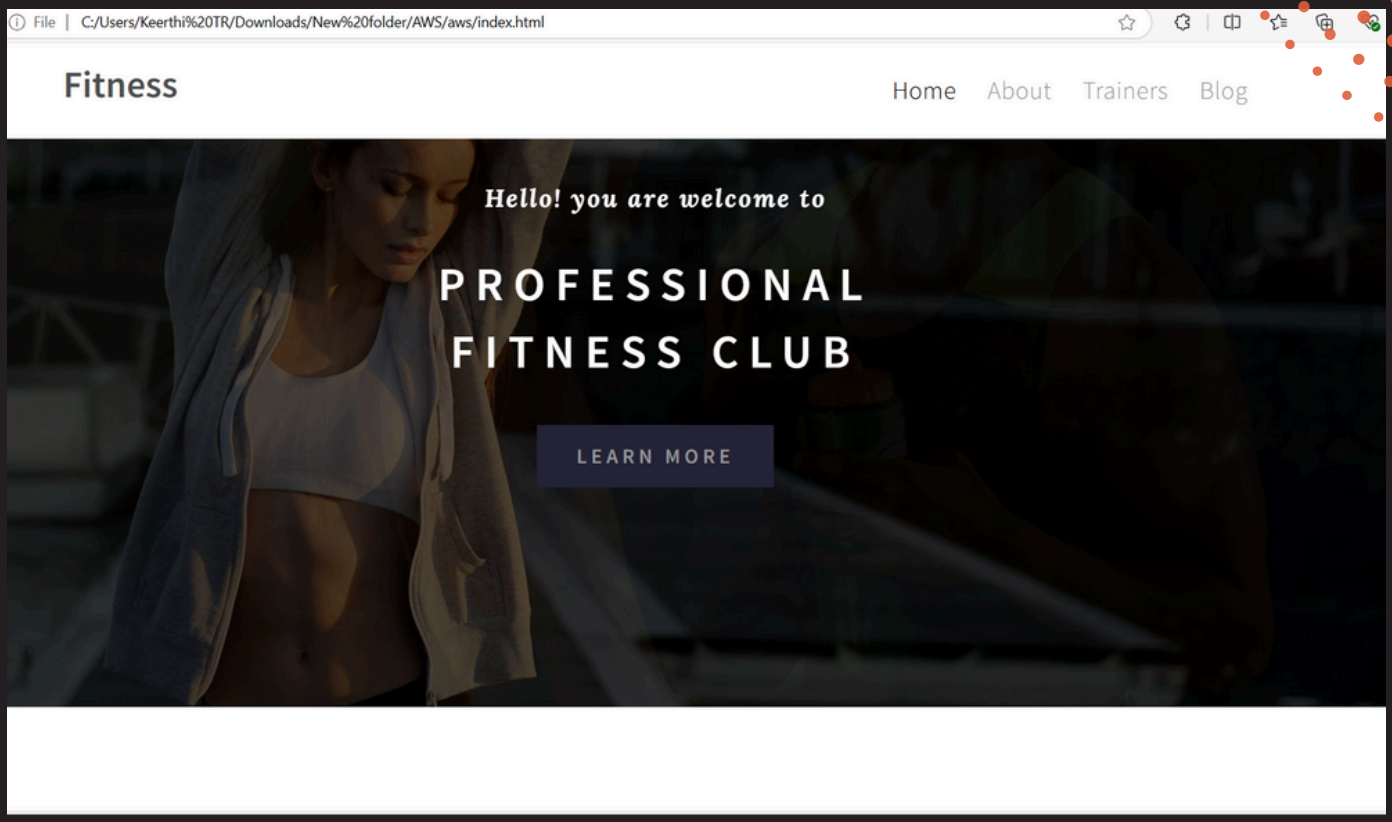


fig 11.15: Website in AWS cloud