

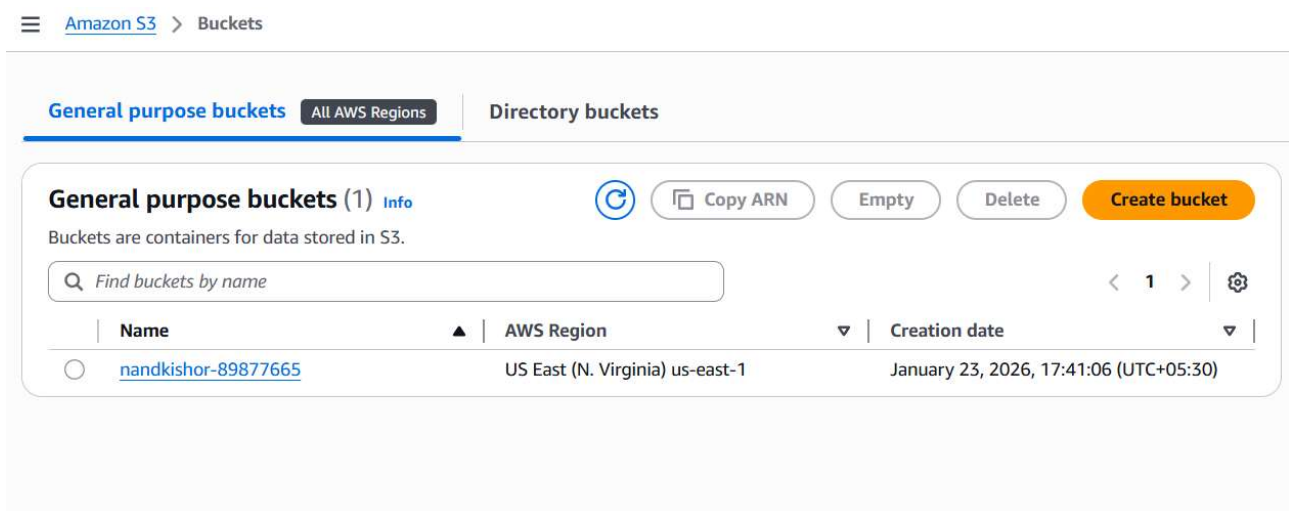
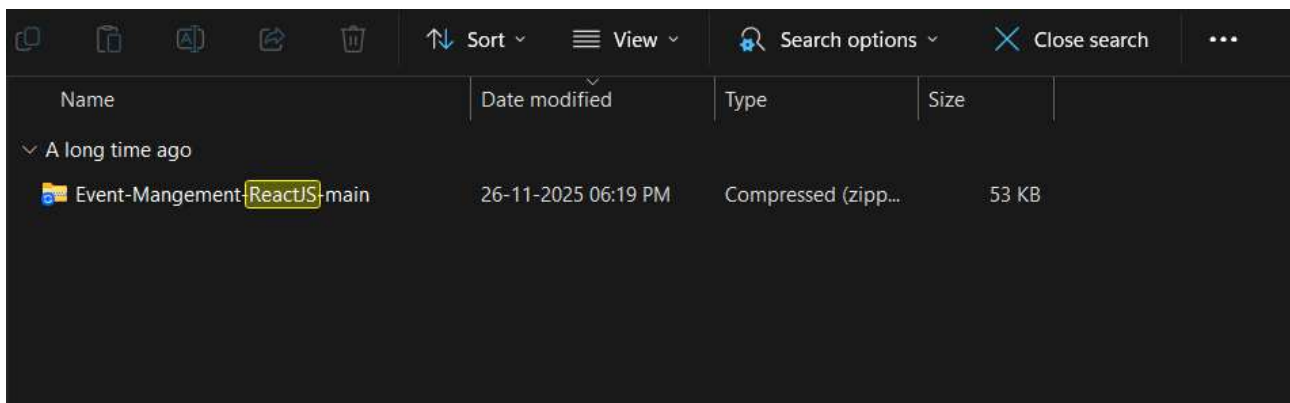
## Deployment of ReactJS Application using AWS EC2, S3 and NGINX

The deployment process involved object storage using Amazon Simple Storage Service (S3), compute resources using Amazon Elastic Compute Cloud (EC2), secure access control via IAM Role, and application hosting using the NGINX web server.

### Use Of Amazon S3

- The React application ZIP file was uploaded to S3.
- The EC2 instance downloaded the file securely using an IAM Role.

In this deployment s3 acts as a storage repository. Firstly i created a s3 bucket uploaded my application file in bucket from local storage :



added my application file in bucket which was a zip file :

Amazon S3

**Upload succeeded**  
For more information, see the [Files and folders](#) table.

After you navigate away from this page, the following information is no longer available.

**Summary**

<b>Destination</b> s3://nandkishor-89877665	<b>Succeeded</b> 1 file, 53.0 KB (100.00%)	<b>Failed</b> 0 files, 0 B (0%)
--	---	------------------------------------

**Files and folders** | Configuration

**Files and folders** (1 total, 53.0 KB)

Find by name

Name	Folder	Type	Size	Status	Error
<a href="#">Event-Mangement-ReactJS-mai...</a>	-	application/x-zip-compressed	53.0 KB	Succeeded	-

launching EC2 instance :

EC2 > Instances > Launch an instance

**Launch an instance** Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

**Name and tags** Info

Name  
project007 [Add additional tags](#)

**Application and OS Images (Amazon Machine Image)** Info

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose [Browse more AMIs](#).

Search our full catalog including 1000s of application and OS images

**Recents** | **Quick Start**

Amazon Linux aws	macOS Mac	Ubuntu ubuntu	Windows Microsoft	Red Hat Red Hat	SUSE Linux SUSE	Debian debian
---------------------	--------------	------------------	----------------------	--------------------	--------------------	------------------

[Browse more AMIs](#)  
Including AMIs from AWS, Marketplace and the Community

**Summary**

**Number of instances** Info  
1

**Software Image (AMI)**  
Amazon Linux 2023 AMI 2023.10...[read more](#)  
ami-0c1fe732b5494dc14

**Virtual server type (instance type)**  
t3.micro

**Firewall (security group)**  
New security group

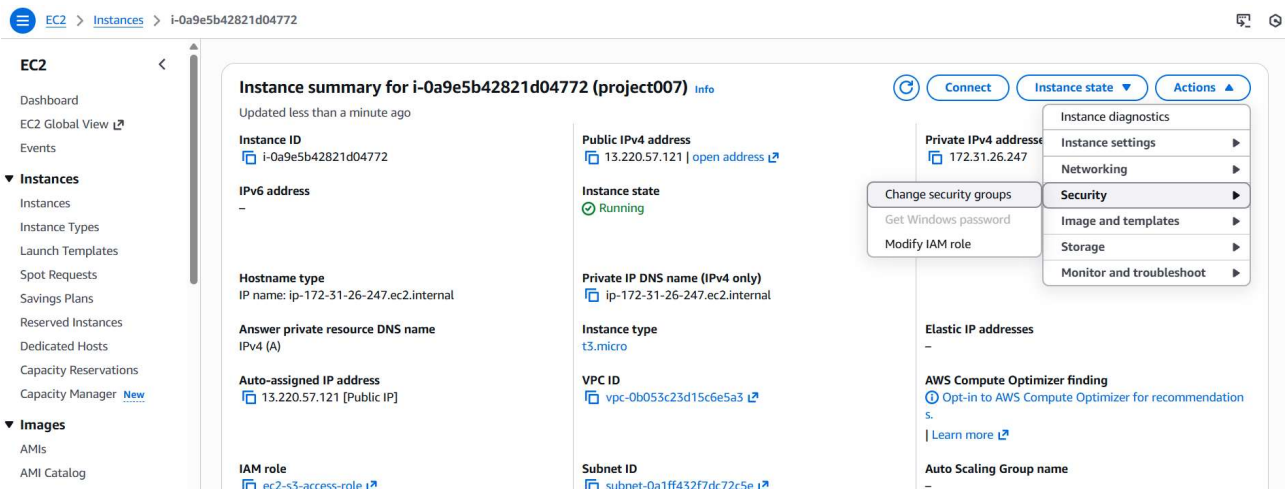
**Storage (volumes)**  
1 volume(s) - 8 GiB

[Cancel](#) [Launch instance](#) [Preview code](#)

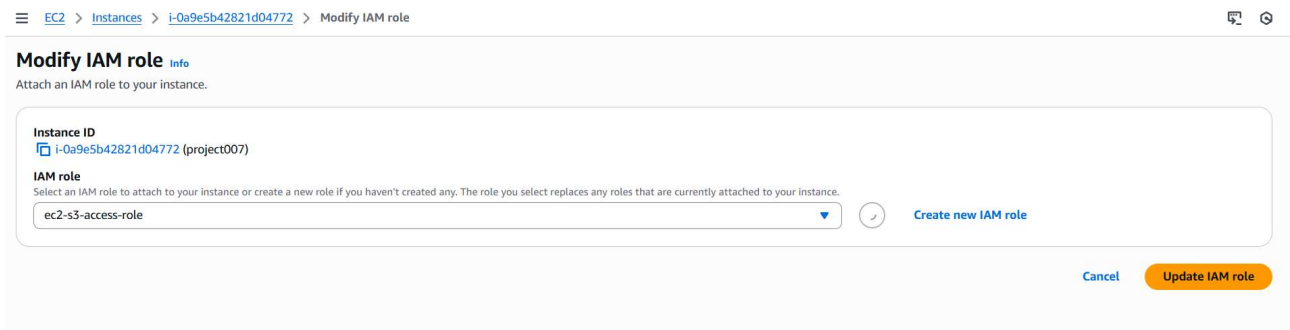
**Success**  
Successfully initiated launch of instance (i-0a9e5b42821d04772)

Nandkishor Gaikwad

create a IAM role so that we will get accesss to s3 and attach it directly to EC2 instance :

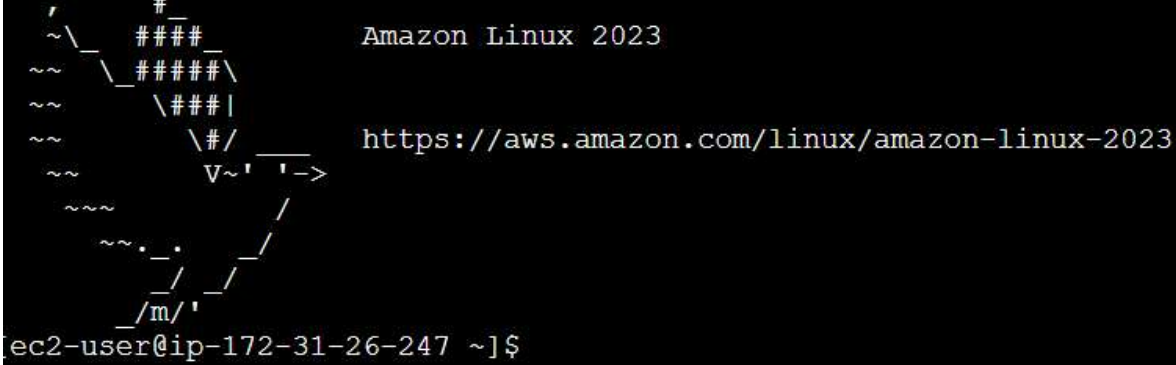


modify IAM role : attach the we cretaed to give full access of S3 to copy application file to EC2



Connecting to EC2 Instance : The instance was accessed using SSH

Nandkishor Gaikwad



## Installation of Required Software

The following packages were installed:

- NodeJS
- npm
- unzip
- awscli
- nginx

Prepare environment for building and serving the application.

```
4 yum update -y
5 sudo yum update -y
6 sudo yum install -y unzip awscli docker
7 sudo systemctl start docker
8 sudo systemctl enable docker
```

## Copying File from S3 to EC2

command for it is : `aws s3 cp s3://bucket-name/file.zip /home/ec2-user/`

## Extracting Application

```
12 sudo systemctl status docker
13 clear
14 aws s3 cp s3://nandkishor-89877665/Event-Mangement-ReactJS-main.zip /home/ec2-user/
15 unzip Event-Mangement-ReactJS-main
16 cd Event-Mangement-ReactJS-main/
```

install dependencies using a command : npm install

it creates node\_modules folder

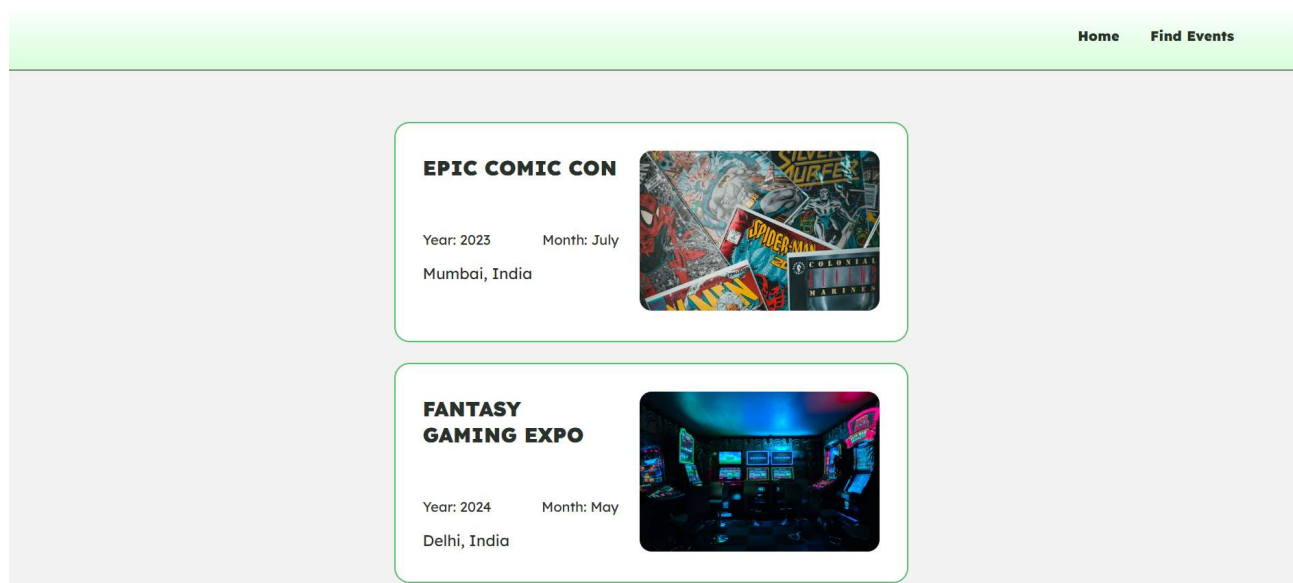
after installing dependencies start nginx service by command : sudo systemctl start nginx and enable it

after enabling, deploy build files to NGINX directory

```
40 sudo rm -rf /usr/share/nginx/html/*
41 sudo cp -r dist/* /usr/nginx/html/
42 sudo cp -r dist/* /usr/share/nginx/html/
```

Now we can access our application with public ip that we will get from instance interface in this process browser sends http request to EC2 processed by NGINX then EC2 serves static react files so that application should run on browser

**Result :**



Application Interface