DEVOPS CAPSTONE PROJECT

Deploy entire website into the cloud infrastructure (AWS) with proper scaling.

Member's:

20A91A04J2

20A91A04O4

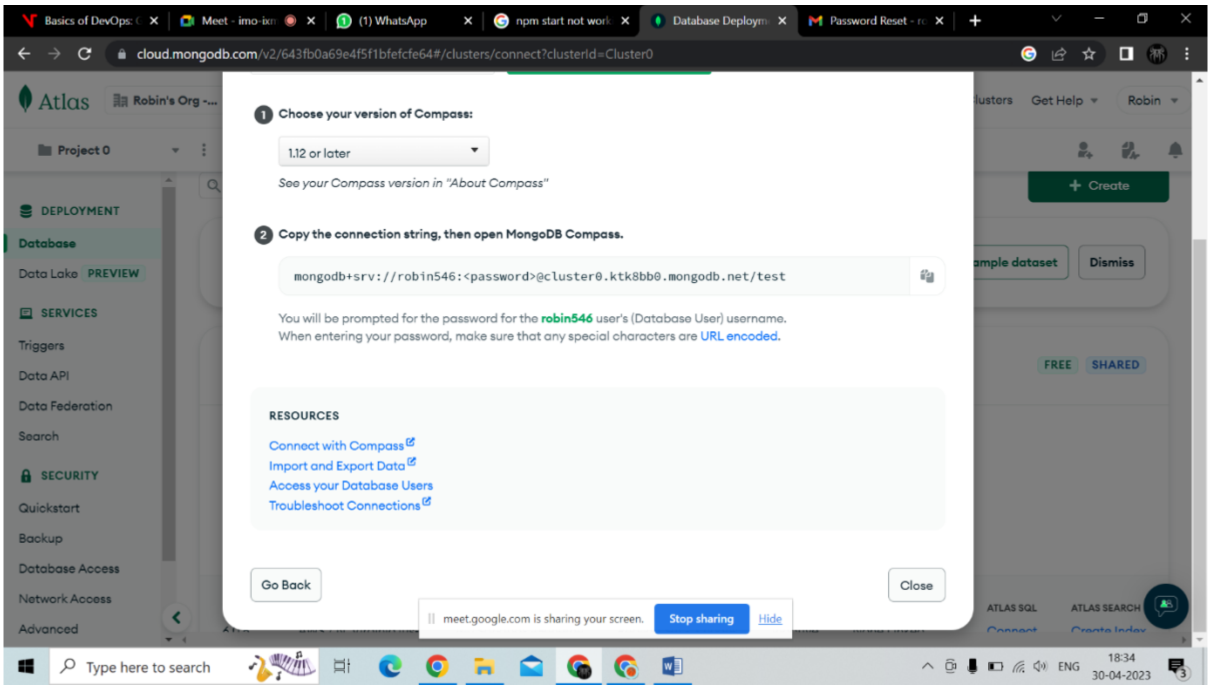
20A91A0345

Set up an AWS account: The first step is to create an AWS account and sign in to the AWS Management Console.

Create an EC2 instance: Launch an EC2 instance with an appropriate operating system (Linux or Windows) and an instance type that meets the website's resource requirements. Choose a VPC, subnet, and security group for the instance.

Install NodeJS and MongoDB: After launching the instance, install NodeJS and MongoDB on the instance.

Configure MongoDB: Configure MongoDB by creating a user and setting up the database.



cloud.mongodb.com/v2/643fb0a69e4f5f1bfefcfe64#/clusters/connect?clusterId=Cluster0

1 Choose your version of Compass:

1.12 or later

See your Compass version in "About Compass"

2 Copy the connection string, then open MongoDB Compass.

```
mongodb+srv://robin546:<password>@cluster0.ktk8bb0.mongodb.net/test
```

You will be prompted for the password for the **robin546** user's (Database User) username.
When entering your password, make sure that any special characters are URL encoded.

RESOURCES

Connect with Compass
Import and Export Data
Access your Database Users
Troubleshoot Connections

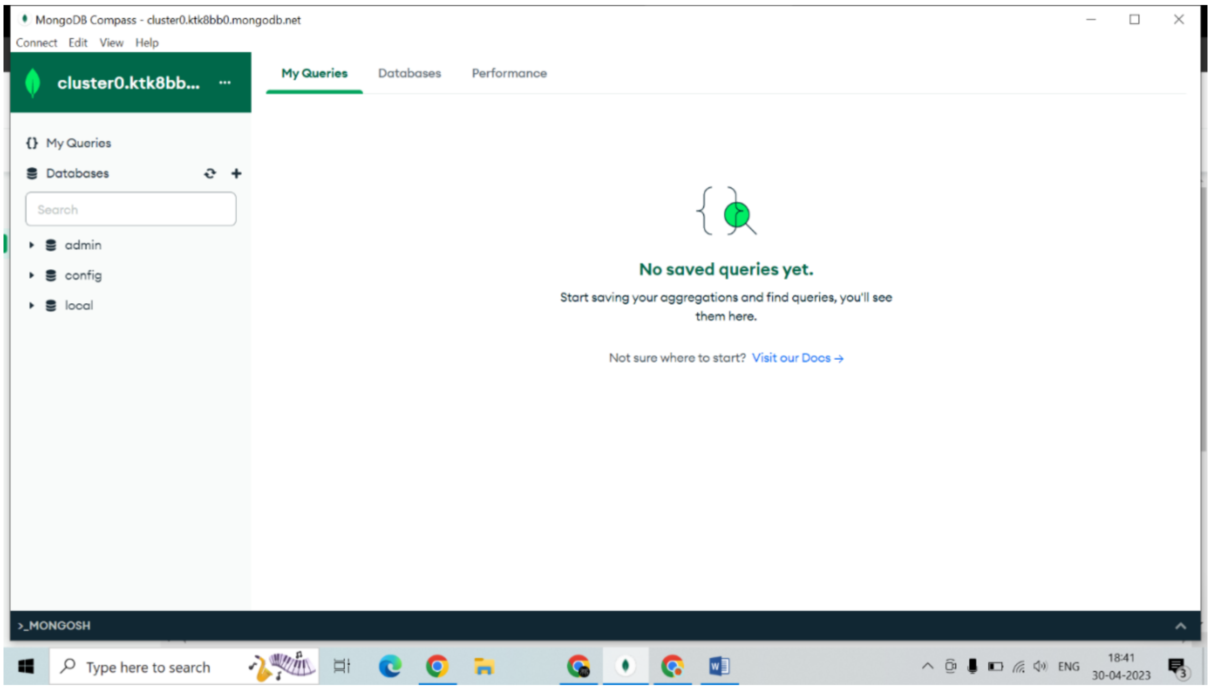
meet.google.com is sharing your screen.

Clone the code repository: Clone the code repository to the EC2 instance.

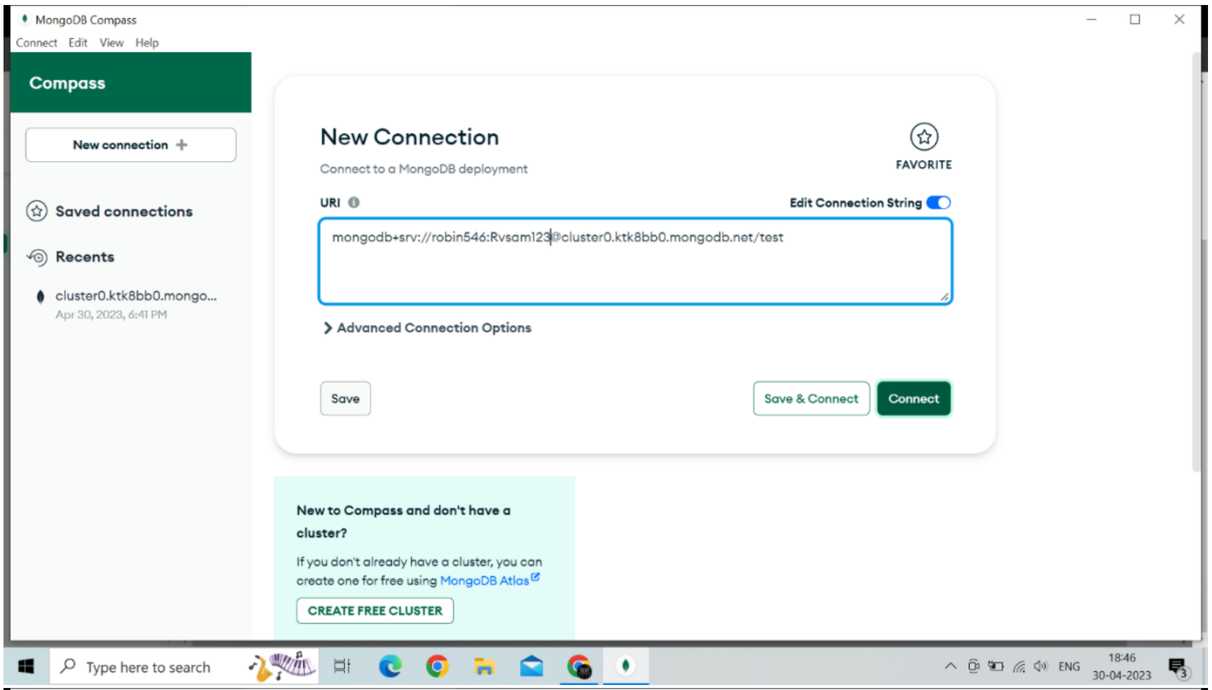
Install dependencies: Install all the necessary dependencies required by the project.

Build and start the project: Build and start the project by running the appropriate commands.

Configure Nginx: Install and configure Nginx as a reverse proxy server to handle incoming requests.



Create an S3 bucket: Create an S3 bucket to store the images uploaded by users.



New Connection

Connect to a MongoDB deployment

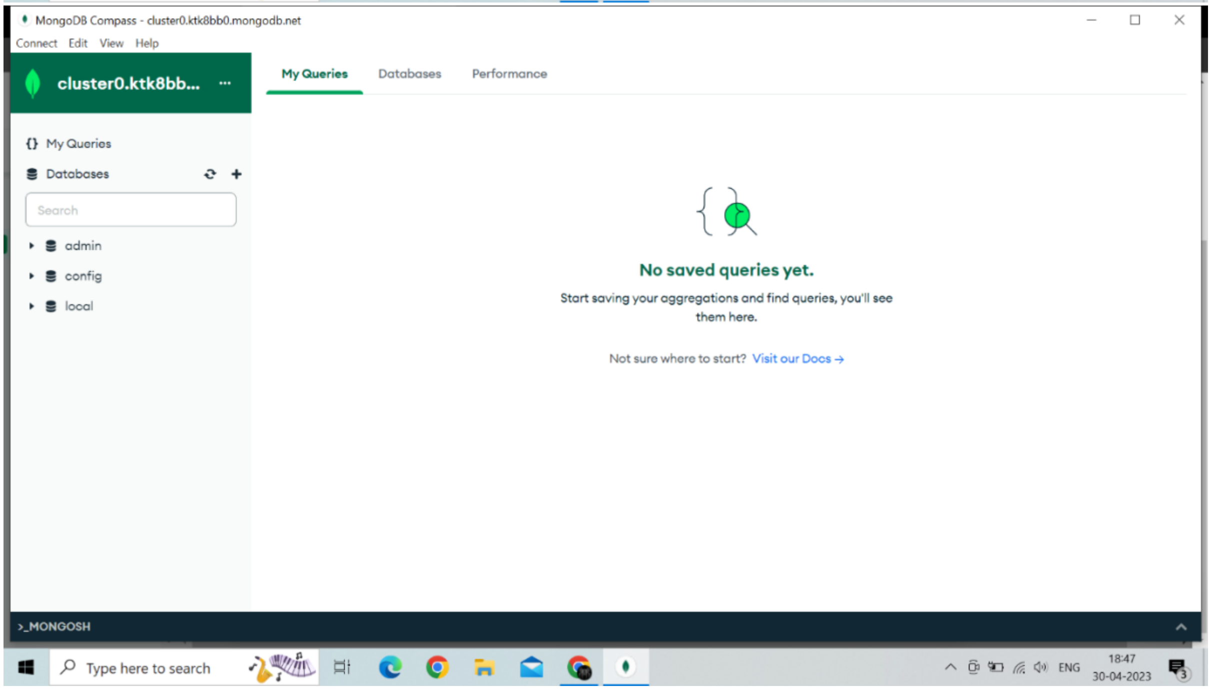
URI

Edit Connection String

```
mongodb+srv://robin546:Rvsam123@cluster0.ktk8bb0.mongodb.net/test
```

New to Compass and don't have a cluster?

If you don't already have a cluster, you can create one for free using MongoDB Atlas



No saved queries yet.

Start saving your aggregations and find queries, you'll see them here.

Not sure where to start? Visit our Docs →

MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time

The document model maps to the objects in your application code, making data easy to work with

Ad hoc queries, indexing, and real time aggregation provide powerful ways to access and analyze your data

MongoDB is a distributed database at its core, so high availability, horizontal scaling, and geographic distribution are built in and easy to use

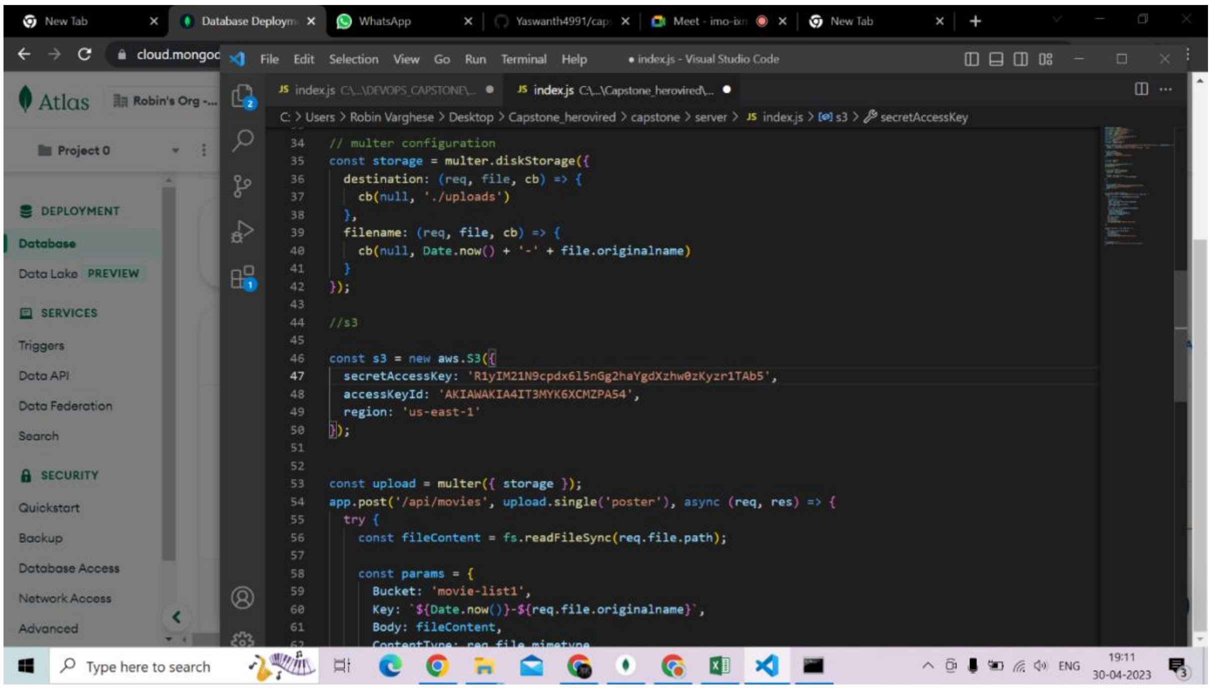
MongoDB is free to use. Versions released prior to October 16, 2018 are published under the AGPL. All versions released after October 16, 2018, including patch fixes for prior versions, are published under the Server Side Public License (SSPL) v1.

MongoDB Atlas is a multi-cloud database service by the same people that build MongoDB. Atlas simplifies deploying and managing your databases while offering the versatility you need to build resilient and performant global applications on the cloud providers of your choice.

MongoDB Atlas is a cloud service by MongoDB. It is built for developers who'd rather spend time building apps than managing databases. This service is available on AWS, Azure, and GCP.

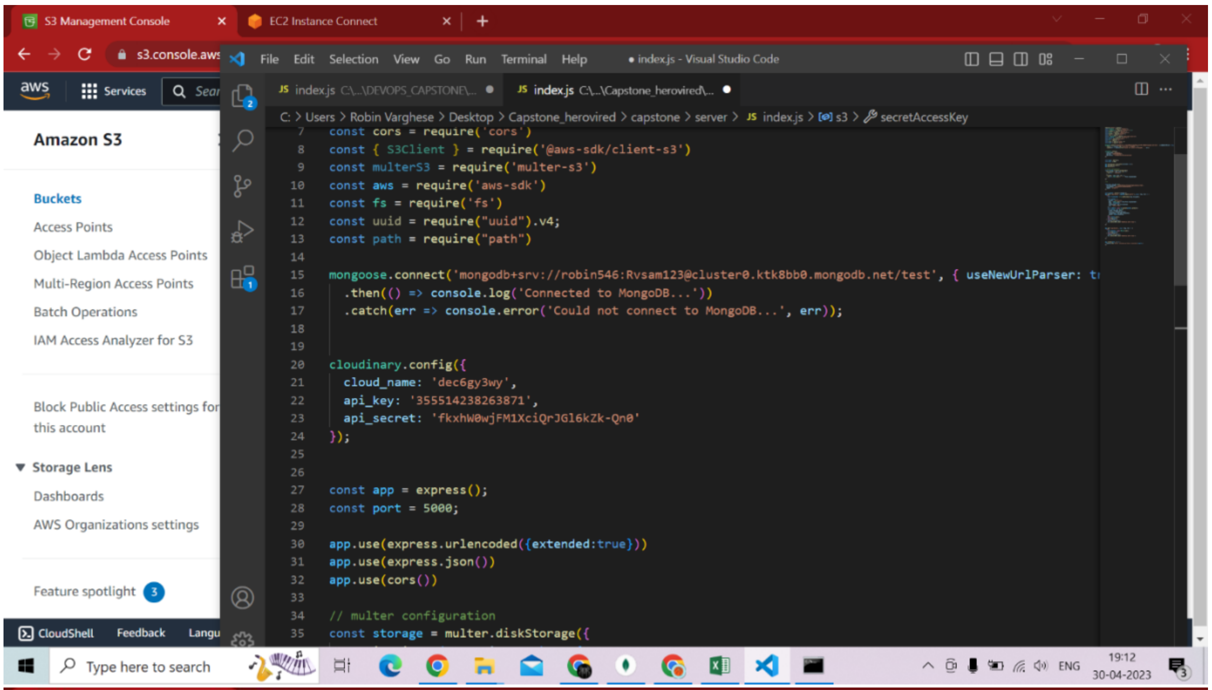
It is the worldwide cloud database service for modern applications that give best-in-class automation and proven practices guarantee availability, scalability, and compliance with the foremost demanding data security and privacy standards. We can use MongoDB's robust ecosystem of drivers, integrations, and tools to create faster and spend less time managing our database.

Enable SSL: Enable SSL to ensure that the website is secure.



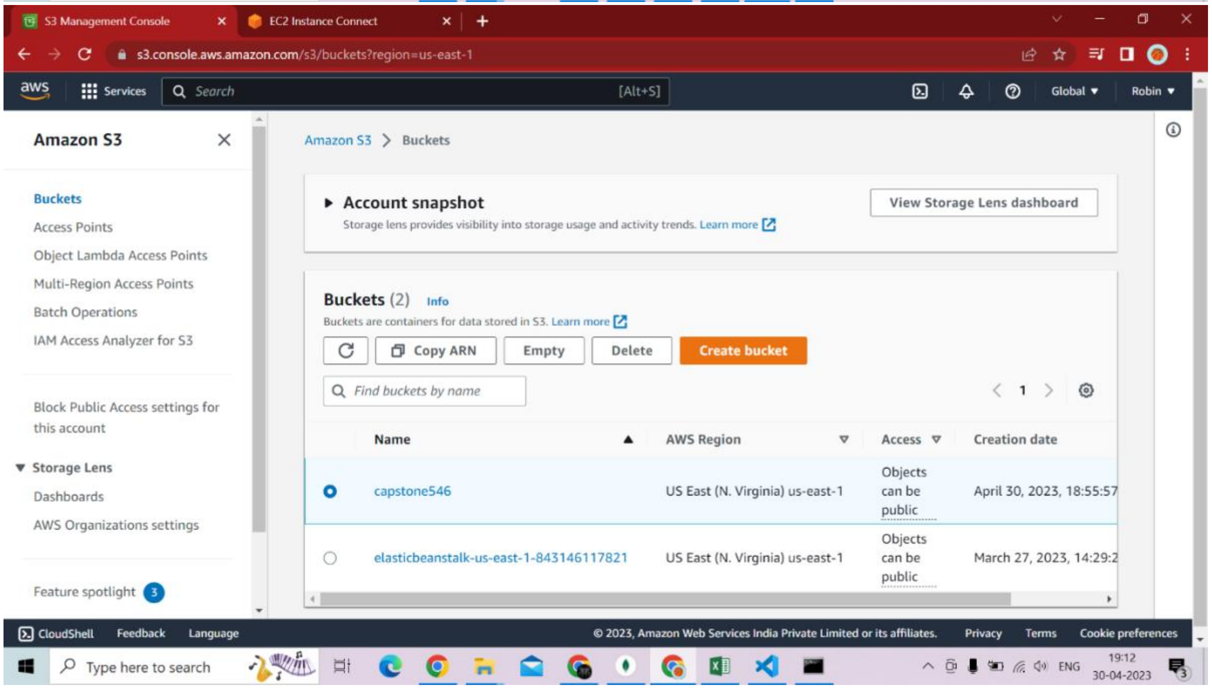
C: > Users > Robin Varghese > Desktop > Capstone_herovired > capstone > server > index.js > s3 > secretAccessKey

```
// multer configuration
const storage = multer.diskStorage({
  destination: (req, file, cb) => {
    cb(null, './uploads')
  },
  filename: (req, file, cb) => {
    cb(null, Date.now() + '-' + file.originalname)
  }
});

//s3

const s3 = new aws.S3({
  secretAccessKey: 'R1yIM21N9cpdx615nGg2haYgdXzhw0zKyzr1TAb5',
  accessKeyId: 'AKIAWAKIA4IT3MYK6XCMZPA54',
  region: 'us-east-1'
});


const upload = multer({ storage });
app.post('/api/movies', upload.single('poster'), async (req, res) => {
  try {
    const fileContent = fs.readFileSync(req.file.path);

    const params = {
      Bucket: 'movie-list1',
      Key: `${Date.now()}-${req.file.originalname}`,
      Body: fileContent,
      ContentType: req.file.mimetype
```



```
const cors = require('cors')
const { S3Client } = require('@aws-sdk/client-s3')
const multerS3 = require('multer-s3')
const aws = require('aws-sdk')
const fs = require('fs')
const uuid = require("uuid").v4;
const path = require("path")

mongoose.connect('mongodb+srv://robin546:Rvsam123@cluster0.ktk8bb0.mongodb.net/test', { useNewUrlParser: t
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));


cloudinary.config({
  cloud_name: 'dec6gy3wy',
  api_key: '355514238263871',
  api_secret: 'fkxhW0wjFM1XciQrJGl6kZk-Qn0'
});


const app = express();
const port = 5000;

app.use(express.urlencoded({extended:true}))
app.use(express.json())
app.use(cors())

// multer configuration
const storage = multer.diskStorage({
```

S3 Management Console
EC2 Instance Connect
index.js - Visual Studio Code
File Edit Selection View Go Run Terminal Help
C: > Users > Robin Varghese > Desktop > Capstone_herovired > capstone > server > index.js > s3 > secretAccessKey
Amazon S3
Buckets
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
AWS Organizations settings
Feature spotlight 3
Type here to search
19:12
30-04-2023



s3.console.aws.amazon.com/s3/buckets?region=us-east-1
Services
Search [Alt+S]
Global
Robin
Amazon S3 > Buckets
Account snapshot
Storage lens provides visibility into storage usage and activity trends. Learn more
View Storage Lens dashboard
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
CloudShell Feedback Language
© 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences

Buckets (2) Info

Buckets are containers for data stored in S3. Learn more

	Name	AWS Region	Access	Creation date
●	capstone546	US East (N. Virginia) us-east-1	Objects can be public	April 30, 2023, 18:55:57
○	elasticbeanstalk-us-east-1-843146117821	US East (N. Virginia) us-east-1	Objects can be public	March 27, 2023, 14:29:2

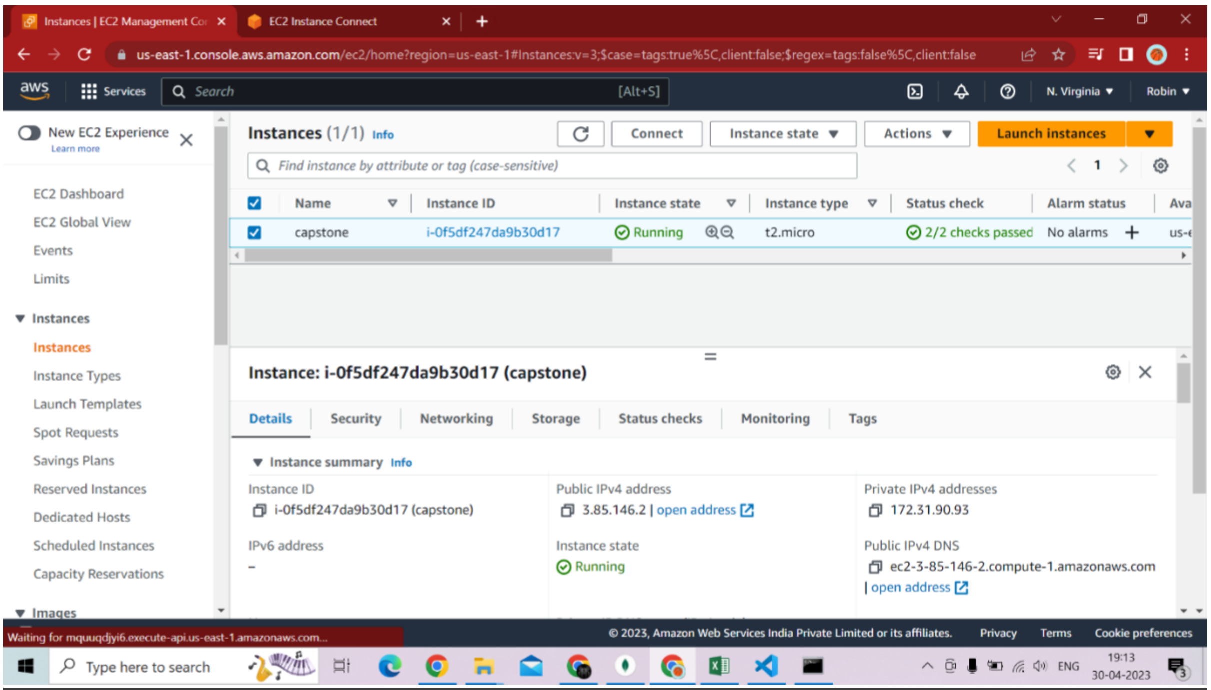
Configure S3 for file storage: Configure the NodeJS application to use the S3 bucket for storing the images.

Set up auto-scaling: Set up auto-scaling to automatically adjust the number of instances based on traffic.

Set up load balancing: Set up load balancing to distribute traffic across the instances.

Set up monitoring and alerts: Set up monitoring and alerts to ensure that the website is running smoothly and to receive notifications if there are any issues.

Test and deploy: Test the website thoroughly and deploy it to the AWS infrastructure.



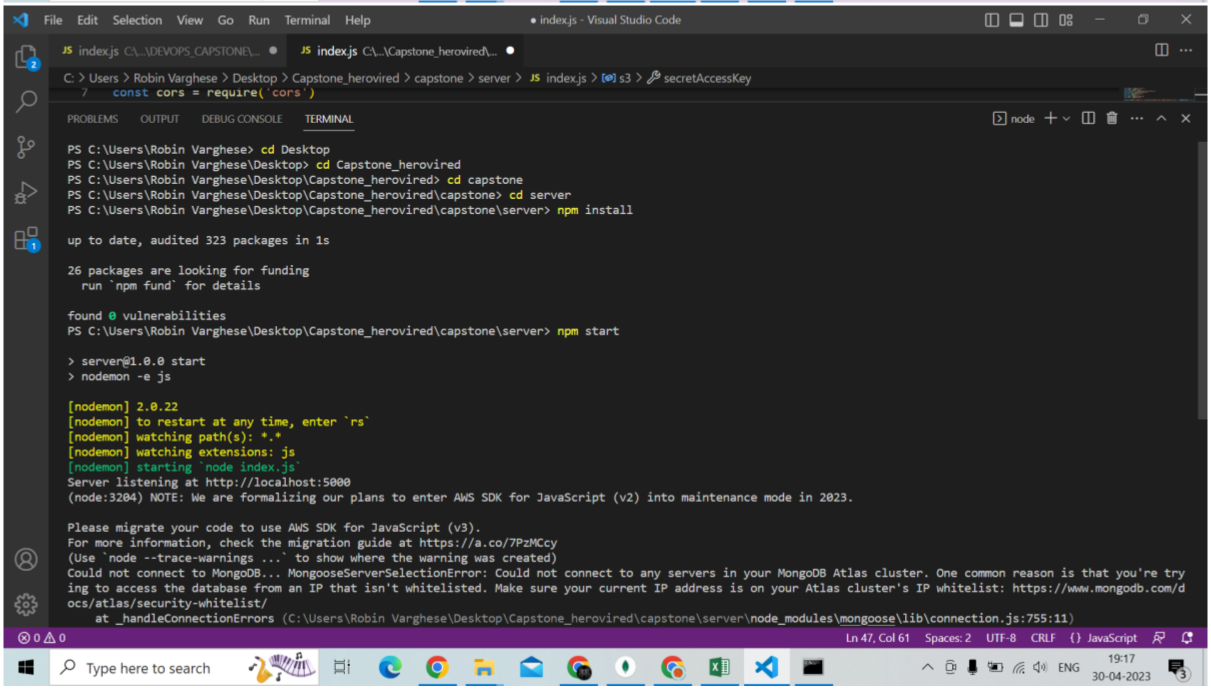
Instances (1/1)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Ava
capstone	i-0f5df247da9b30d17	Running	t2.micro	2/2 checks passed	No alarms	us-e

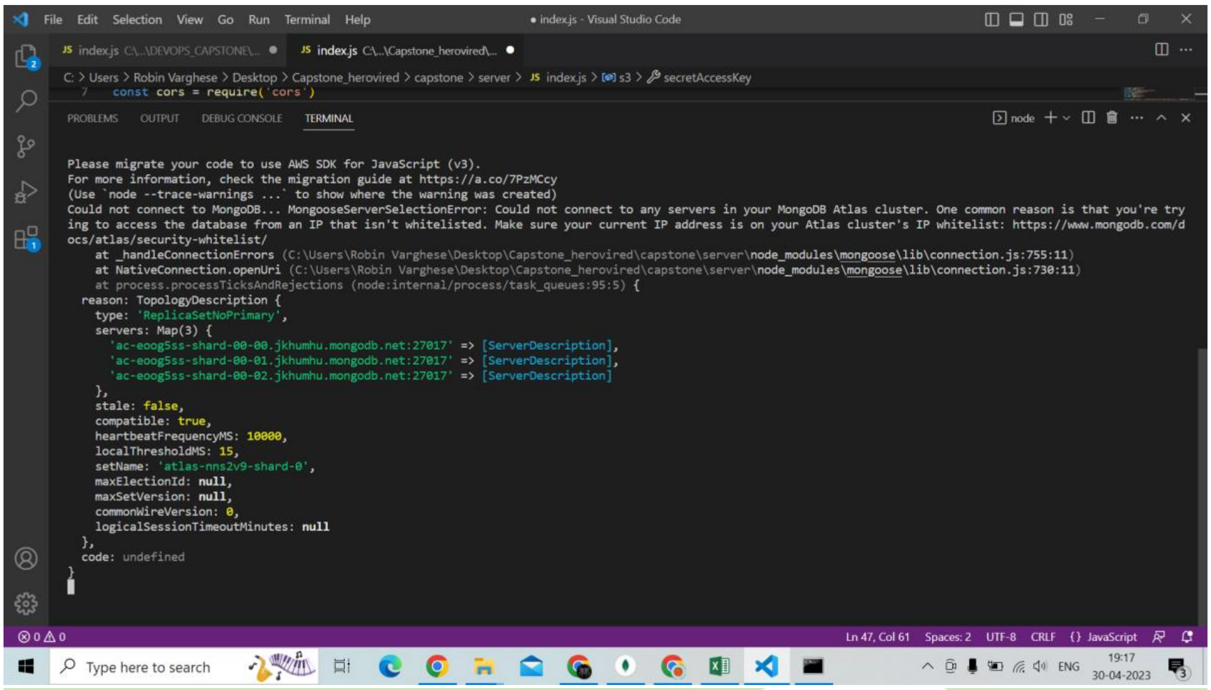
Instance: i-0f5df247da9b30d17 (capstone)

Instance summary

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0f5df247da9b30d17 (capstone)	3.85.146.2 | open address	172.31.90.93
IPv6 address	Instance state	Public IPv4 DNS
–	Running	ec2-3-85-146-2.compute-1.amazonaws.com | open address

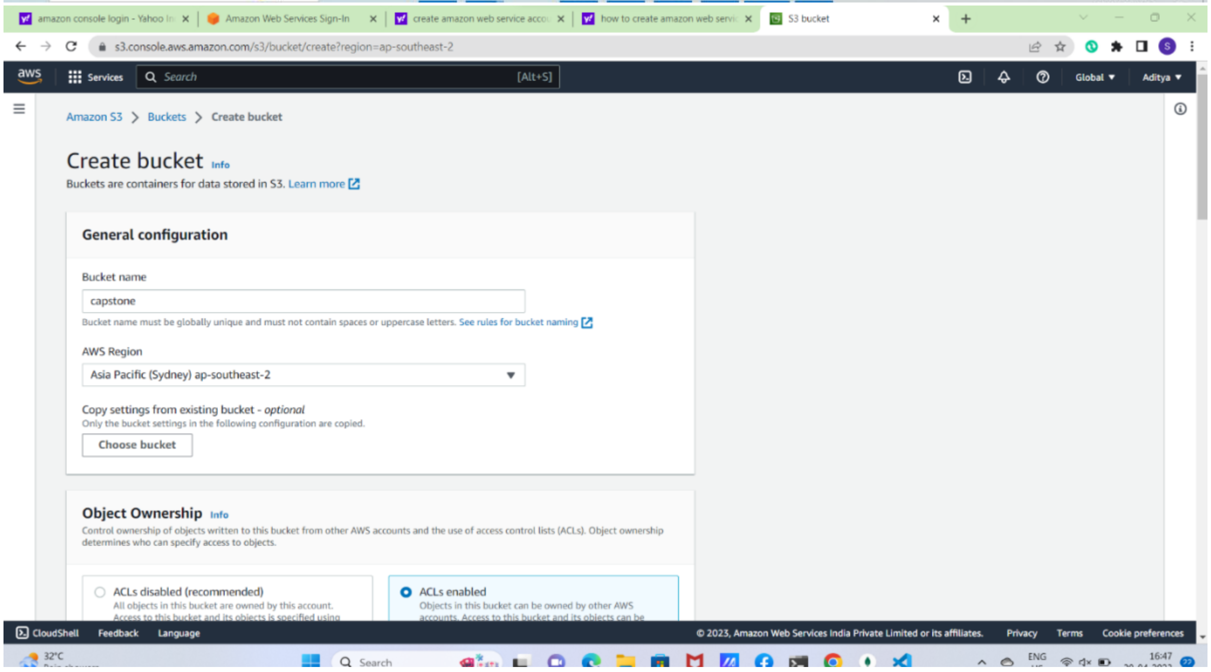


```
PS C:\Users\Robin Varghese> cd Desktop
PS C:\Users\Robin Varghese\Desktop> cd Capstone_herovired
PS C:\Users\Robin Varghese\Desktop\Capstone_herovired> cd capstone
PS C:\Users\Robin Varghese\Desktop\Capstone_herovired\capstone> cd server
PS C:\Users\Robin Varghese\Desktop\Capstone_herovired\capstone\server> npm install

up to date, audited 323 packages in 1s

26 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\Robin Varghese\Desktop\Capstone_herovired\capstone\server> npm start

> server@1.0.0 start
> nodemon -e js

[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js
[nodemon] starting `node index.js`
Server listening at http://localhost:5000
(node:3204) NOTE: We are formalizing our plans to enter AWS SDK for JavaScript (v2) into maintenance mode in 2023.

Please migrate your code to use AWS SDK for JavaScript (v3).
For more information, check the migration guide at https://a.co/7PzMCcy
(Use `node --trace-warnings ...` to show where the warning was created)
Could not connect to MongoDB... MongooseServerSelectionError: Could not connect to any servers in your MongoDB Atlas cluster. One common reason is that you're try
ing to access the database from an IP that isn't whitelisted. Make sure your current IP address is on your Atlas cluster's IP whitelist: https://www.mongodb.com/d
ocs/atlas/security-whitelist/
    at _handleConnectionErrors (C:\Users\Robin Varghese\Desktop\Capstone_herovired\capstone\server\node_modules\mongoose\lib\connection.js:755:11)
```



```
Please migrate your code to use AWS SDK for JavaScript (v3).
For more information, check the migration guide at https://a.co/7PzMCcy
(Use `node --trace-warnings ...` to show where the warning was created)
Could not connect to MongoDB... MongooseServerSelectionError: Could not connect to any servers in your MongoDB Atlas cluster. One common reason is that you're try
ing to access the database from an IP that isn't whitelisted. Make sure your current IP address is on your Atlas cluster's IP whitelist: https://www.mongodb.com/d
ocs/atlas/security-whitelist/
    at _handleConnectionErrors (C:\Users\Robin Varghese\Desktop\Capstone_herovired\capstone\server\node_modules\mongoose\lib\connection.js:755:11)
    at NativeConnection.openUri (C:\Users\Robin Varghese\Desktop\Capstone_herovired\capstone\server\node_modules\mongoose\lib\connection.js:730:11)
    at process.processTicksAndRejections (node:internal/process/task_queues:95:5) {
  reason: TopologyDescription {
    type: 'ReplicaSetNoPrimary',
    servers: Map(3) {
      'ac-eoog5ss-shard-00-00.jkhumhu.mongodb.net:27017' => [ServerDescription],
      'ac-eoog5ss-shard-00-01.jkhumhu.mongodb.net:27017' => [ServerDescription],
      'ac-eoog5ss-shard-00-02.jkhumhu.mongodb.net:27017' => [ServerDescription]
    },
    stale: false,
    compatible: true,
    heartbeatFrequencyMS: 10000,
    localThresholdMS: 15,
    setName: 'atlas-nns2v9-shard-0',
    maxElectionId: null,
    maxSetVersion: null,
    commonWireVersion: 0,
    logicalSessionTimeoutMinutes: null
  },
  code: undefined
}
```

File Edit Selection View Go Run Terminal Help
● index.js - Visual Studio Code
index.js C:\...\DEVOPS_CAPSTONE\...
index.js C:\...\Capstone_herovired\...
C: > Users > Robin Varghese > Desktop > Capstone_herovired > capstone > server > index.js > s3 > secretAccessKey
const cors = require('cors')
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
node
Ln 47, Col 61 Spaces: 2 UTF-8 CRLF {} JavaScript
Type here to search
19:17
30-04-2023



amazon console login - Yahoo In
Amazon Web Services Sign-In
create amazon web service acco
how to create amazon web servi
S3 bucket
s3.console.aws.amazon.com/s3/bucket/create?region=ap-southeast-2
Services
Search [Alt+S]
Global
Aditya
Amazon S3 > Buckets > Create bucket

Create bucket

Buckets are containers for data stored in S3. Learn more

General configuration

Bucket name

capstone

Bucket name must be globally unique and must not contain spaces or uppercase letters. See rules for bucket naming

AWS Region

Asia Pacific (Sydney) ap-southeast-2

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Choose bucket

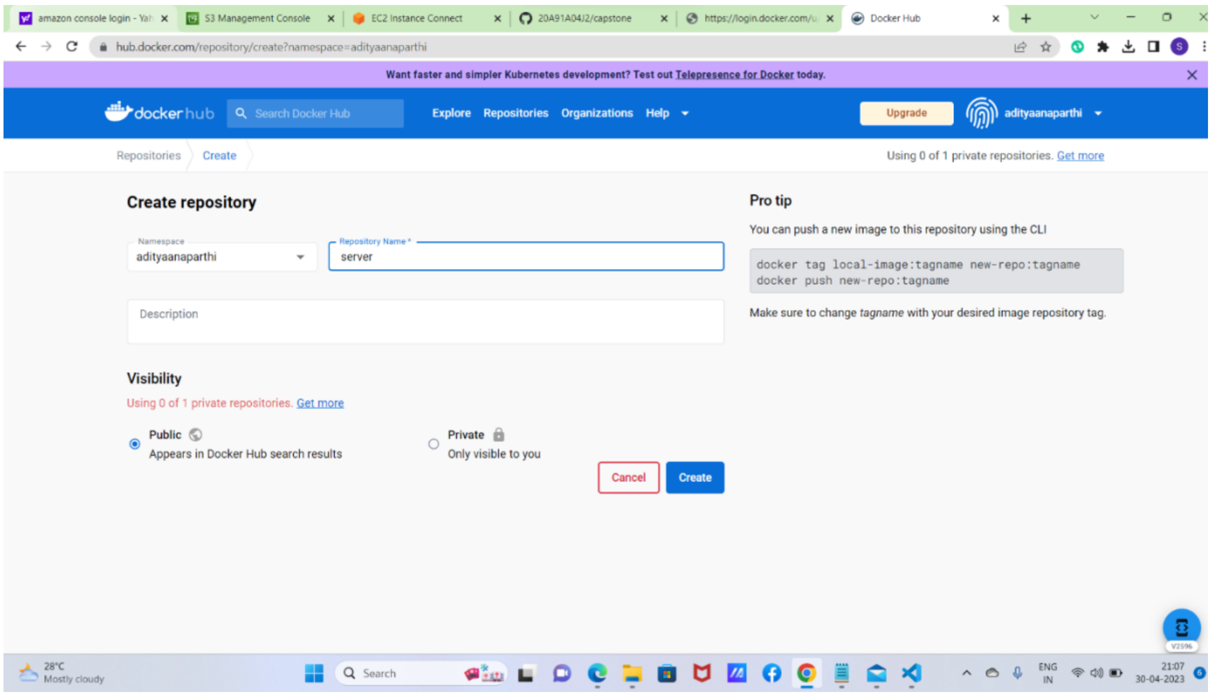
Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

ACLs disabled (recommended)
All objects in this bucket are owned by this account.
Access to this bucket and its objects is specified using

[✓ ACLs enabled]
Objects in this bucket can be owned by other AWS
accounts. Access to this bucket and its objects can be

CloudShell Feedback Language
© 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences
32°C
Search
ENG
16:47


amazon console login - Yah
S3 Management Console
EC2 Instance Connect
20A91A04J2/capstone
https://login.docker.com/u
Docker Hub
hub.docker.com/repository/create?namespace=adityaanaparthi
Want faster and simpler Kubernetes development? Test out Telepresence for Docker today.
docker hub
Search Docker Hub
Explore
Repositories
Organizations
Help
Upgrade
adityaanaparthi
Repositories
Create
Using 0 of 1 private repositories. Get more

Create repository

Namespace: adityaanaparthi

Repository Name *: server

Description

Visibility

Using 0 of 1 private repositories. Get more

Public [✓] — Appears in Docker Hub search results

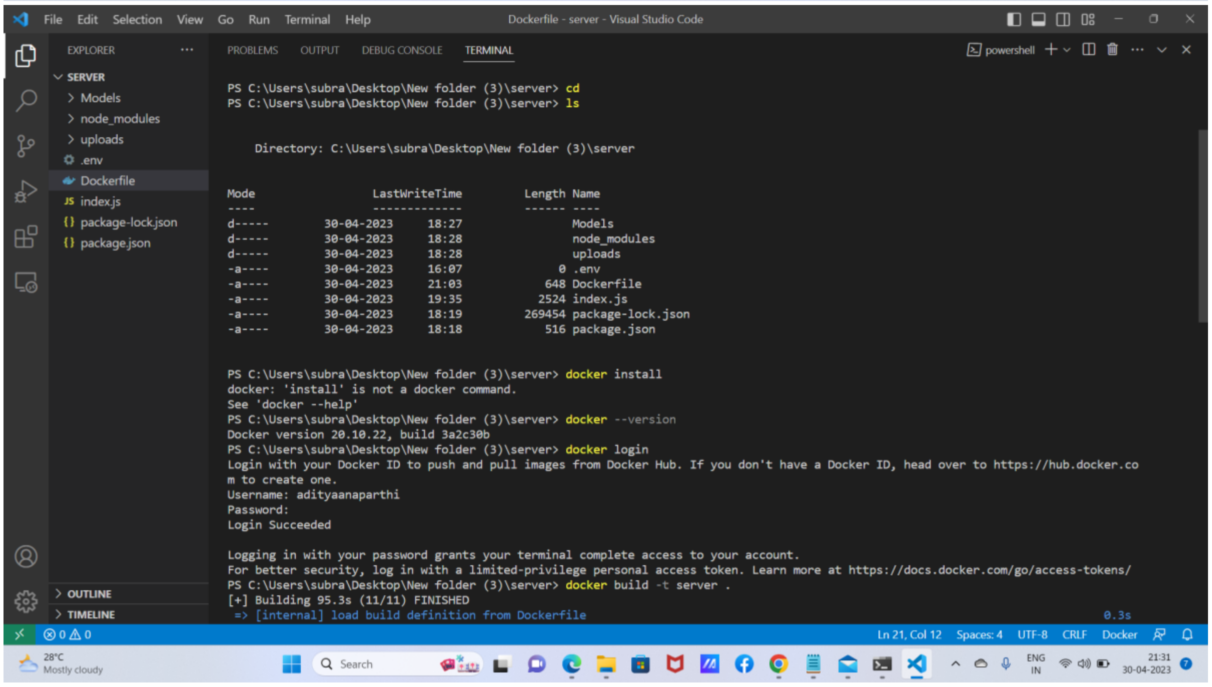
Private — Only visible to you

Pro tip

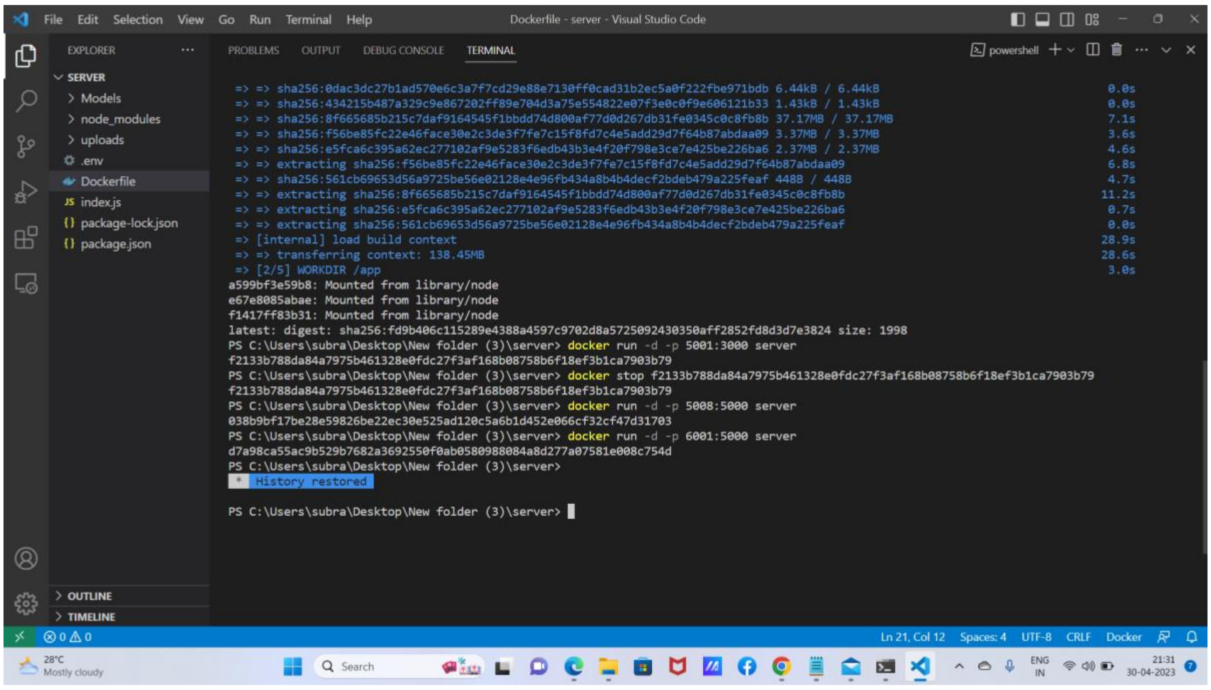
You can push a new image to this repository using the CLI

```
docker tag local-image:tagname new-repo:tagname
docker push new-repo:tagname
```

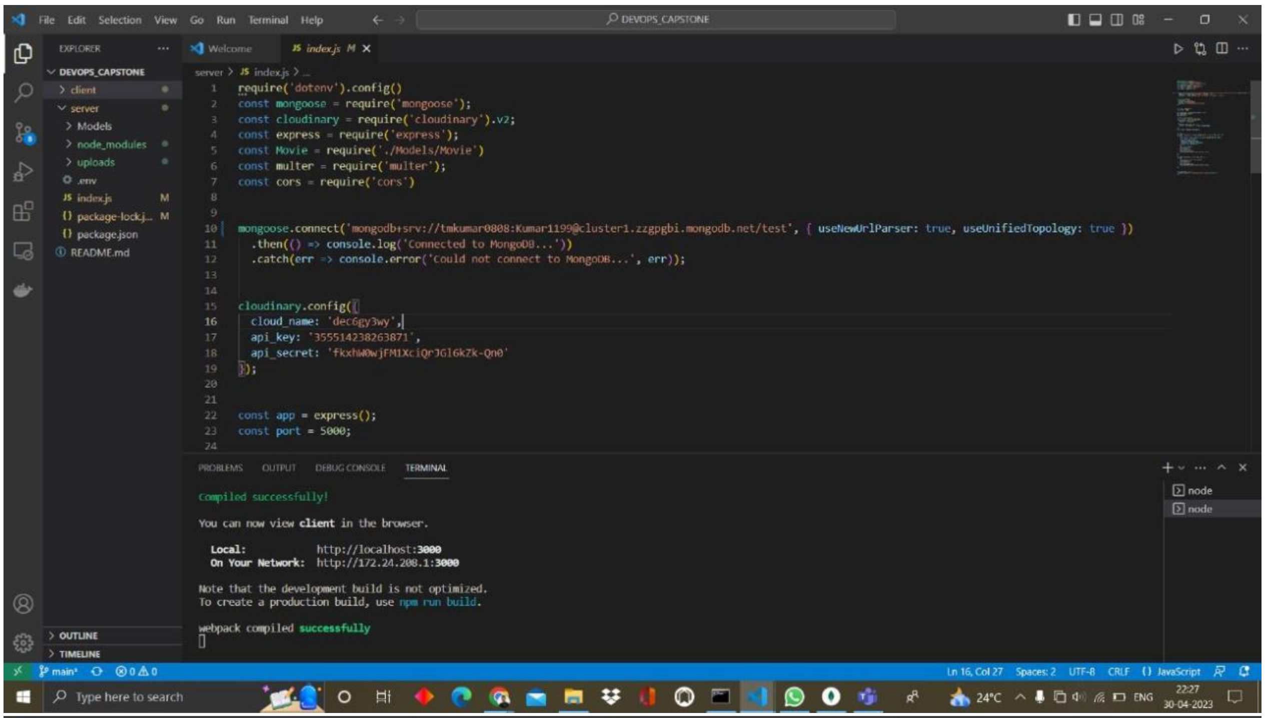
Make sure to change *tagname* with your desired image repository tag.



```
PS C:\Users\subra\Desktop\New folder (3)\server> cd
PS C:\Users\subra\Desktop\New folder (3)\server> ls


    Directory: C:\Users\subra\Desktop\New folder (3)\server


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        30-04-2023     18:27                Models
d-----        30-04-2023     18:28                node_modules
d-----        30-04-2023     18:28                uploads
-a----        30-04-2023     16:07              0 .env
-a----        30-04-2023     21:03            648 Dockerfile
-a----        30-04-2023     19:35           2524 index.js
-a----        30-04-2023     18:19         269454 package-lock.json
-a----        30-04-2023     18:18            516 package.json


PS C:\Users\subra\Desktop\New folder (3)\server> docker install
docker: 'install' is not a docker command.
See 'docker --help'
PS C:\Users\subra\Desktop\New folder (3)\server> docker --version
Docker version 20.10.22, build 3a2c30b
PS C:\Users\subra\Desktop\New folder (3)\server> docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.co
m to create one.
Username: adityaanaparthi
Password:
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/
PS C:\Users\subra\Desktop\New folder (3)\server> docker build -t server .
[+] Building 95.3s (11/11) FINISHED
 => [internal] load build definition from Dockerfile                                                                          0.3s
```

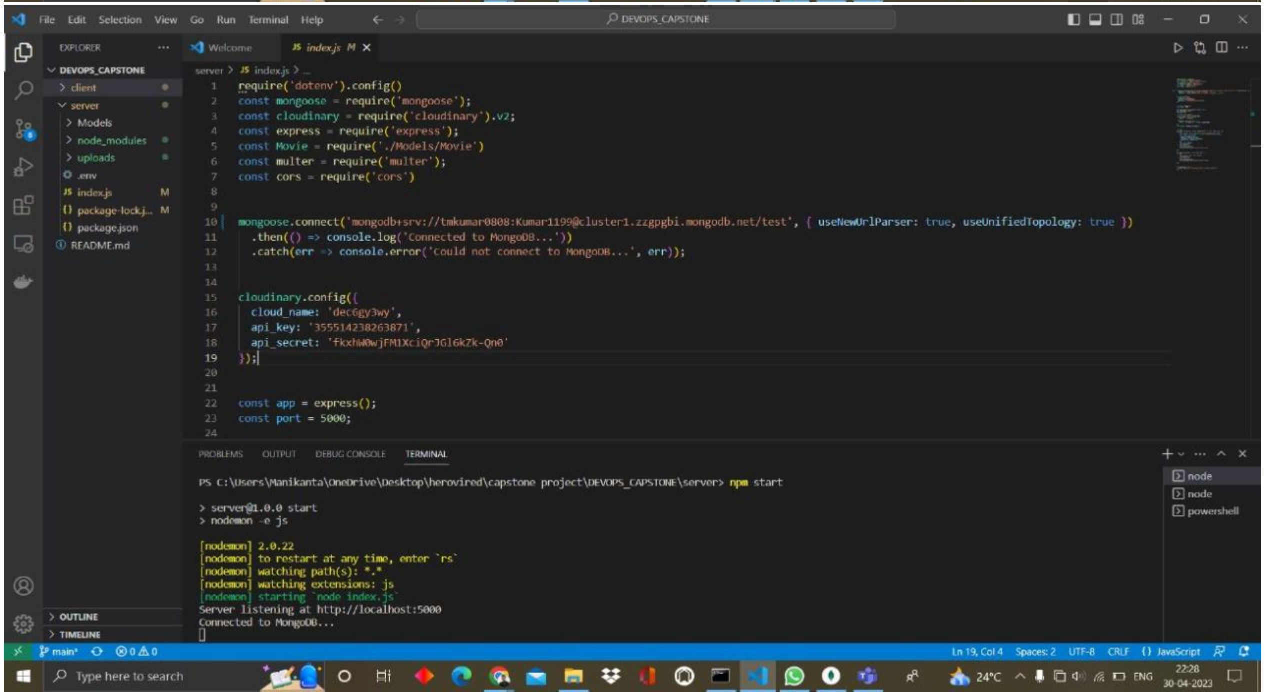


```
 => => sha256:0dac3dc27b1ad570e6c3a7f7cd29e88e7130ff0cad31b2ec5a0f222fbe971bdb 6.44kB / 6.44kB                      0.0s
 => => sha256:434215b487a329c9e867202ff89e704d3a75e554822e07f3e0c0f9e606121b33 1.43kB / 1.43kB                      0.0s
 => => sha256:8f665685b215c7daf9164545f1bbdd74d800af77d0d267db31fe0345c0c8fb8b 37.17MB / 37.17MB                    7.1s
 => => sha256:f56be85fc22e46face30e2c3de3f7fe7c15f8fd7c4e5add29d7f64b87abdaa09 3.37MB / 3.37MB                      3.6s
 => => sha256:e5fca6c395a62ec277102af9e5283f6edb43b3e4f20f798e3ce7e425be226ba6 2.37MB / 2.37MB                      4.6s
 => => extracting sha256:f56be85fc22e46face30e2c3de3f7fe7c15f8fd7c4e5add29d7f64b87abdaa09                           6.8s
 => => sha256:561cb69653d56a9725be56e02128e4e96fb434a8b4b4decf2bdeb479a225feaf 448B / 448B                          4.7s
 => => extracting sha256:8f665685b215c7daf9164545f1bbdd74d800af77d0d267db31fe0345c0c8fb8b                          11.2s
 => => extracting sha256:e5fca6c395a62ec277102af9e5283f6edb43b3e4f20f798e3ce7e425be226ba6                           0.7s
 => => extracting sha256:561cb69653d56a9725be56e02128e4e96fb434a8b4b4decf2bdeb479a225feaf                           0.0s
 => [internal] load build context                                                                                  28.9s
 => => transferring context: 138.45MB                                                                              28.6s
 => [2/5] WORKDIR /app                                                                                              3.0s
a599bf3e59b8: Mounted from library/node
e67e8085abae: Mounted from library/node
f1417ff83b31: Mounted from library/node
latest: digest: sha256:fd9b406c115289e4388a4597c9702d8a5725092430350aff2852fd8d3d7e3824 size: 1998
PS C:\Users\subra\Desktop\New folder (3)\server> docker run -d -p 5001:3000 server
f2133b788da84a7975b461328e0fdc27f3af168b08758b6f18ef3b1ca7903b79
PS C:\Users\subra\Desktop\New folder (3)\server> docker stop f2133b788da84a7975b461328e0fdc27f3af168b08758b6f18ef3b1ca7903b79
f2133b788da84a7975b461328e0fdc27f3af168b08758b6f18ef3b1ca7903b79
PS C:\Users\subra\Desktop\New folder (3)\server> docker run -d -p 5008:5000 server
038b9bf17be28e59826be22ec30e525ad120c5a6b1d452e066cf32cf47d31703
PS C:\Users\subra\Desktop\New folder (3)\server> docker run -d -p 6001:5000 server
d7a98ca55ac9b529b7682a3692550f0ab0580988084a8d277a07581e008c754d
PS C:\Users\subra\Desktop\New folder (3)\server>
 *  History restored

PS C:\Users\subra\Desktop\New folder (3)\server>
```



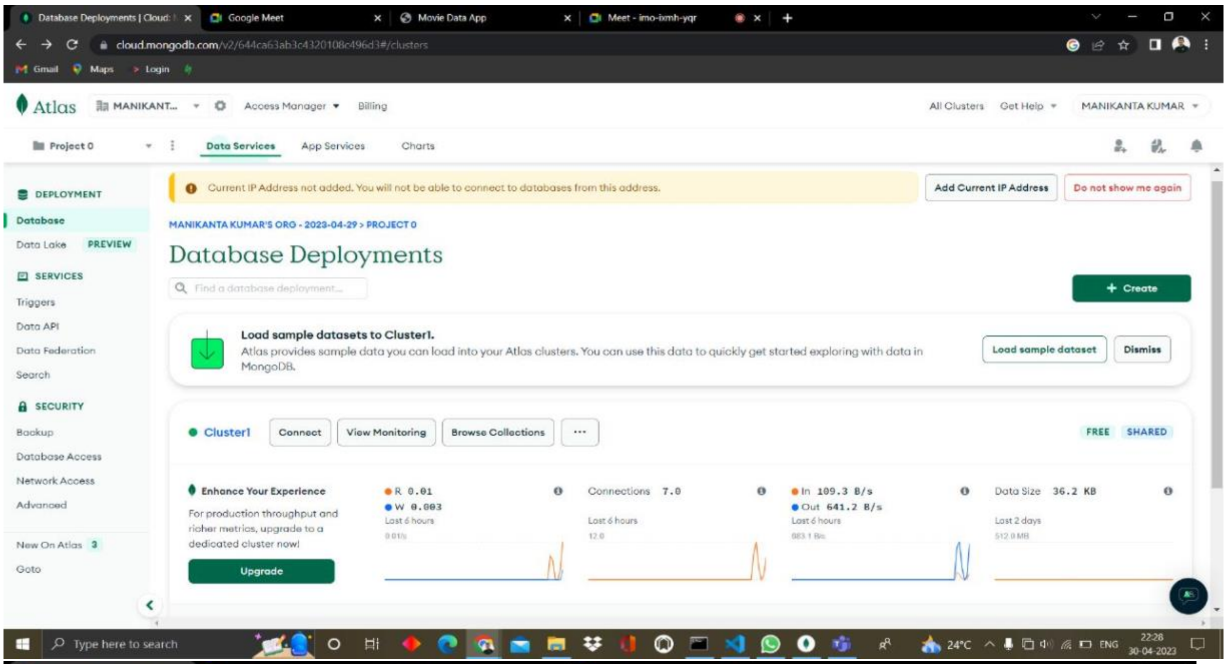
```javascript
require('dotenv').config()
const mongoose = require('mongoose');
const cloudinary = require('cloudinary').v2;
const express = require('express');
const Movie = require('./Models/Movie')
const multer = require('multer');
const cors = require('cors')


mongoose.connect('mongodb+srv://tmkumar0808:Kumar1199@cluster1.zzgpgbi.mongodb.net/test', { useNewUrlParser: true, useUnifiedTopology: true })
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));


cloudinary.config({
  cloud_name: 'dec6gy3wy',
  api_key: '355514238263871',
  api_secret: 'fkxhW0wjFM1XciQrJGl6kZk-Qn0'
});


const app = express();
const port = 5000;
```

```
Compiled successfully!

You can now view client in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://172.24.208.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```



```javascript
require('dotenv').config()
const mongoose = require('mongoose');
const cloudinary = require('cloudinary').v2;
const express = require('express');
const Movie = require('./Models/Movie')
const multer = require('multer');
const cors = require('cors')


mongoose.connect('mongodb+srv://tmkumar0808:Kumar1199@cluster1.zzgpgbi.mongodb.net/test', { useNewUrlParser: true, useUnifiedTopology: true })
  .then(() => console.log('Connected to MongoDB...'))
  .catch(err => console.error('Could not connect to MongoDB...', err));


cloudinary.config({
  cloud_name: 'dec6gy3wy',
  api_key: '355514238263871',
  api_secret: 'fkxhW0wjFM1XciQrJGl6kZk-Qn0'
});


const app = express();
const port = 5000;
```

```
PS C:\Users\Manikanta\OneDrive\Desktop\herovired\capstone project\DEVOPS_CAPSTONE\server> npm start

> server@1.0.0 start
> nodemon -e js

[nodemon] 2.0.22
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js
[nodemon] starting `node index.js`
Server listening at http://localhost:5000
Connected to MongoDB...
```



Current IP Address not added. You will not be able to connect to databases from this address.

MANIKANTA KUMAR'S ORG - 2023-04-29 > PROJECT 0

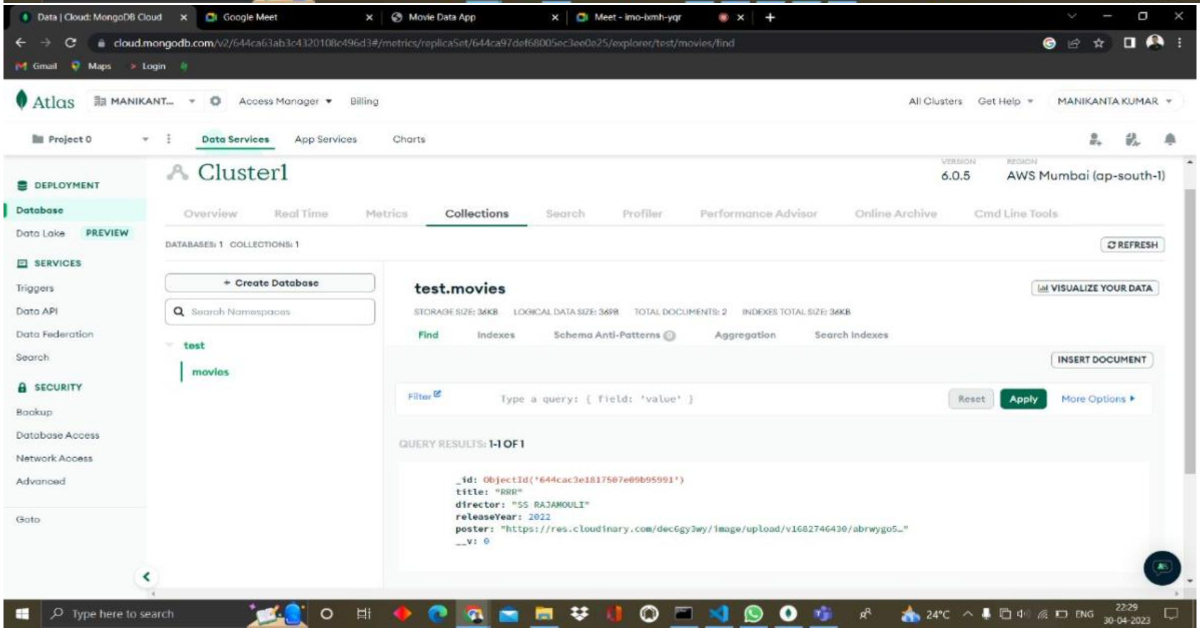
Database Deployments

Load sample datasets to Cluster1.
Atlas provides sample data you can load into your Atlas clusters. You can use this data to quickly get started exploring with data in MongoDB.

Cluster1 FREE SHARED

Enhance Your Experience
For production throughput and richer metrics, upgrade to a dedicated cluster now!

```
R 0.01          Connections 7.0     In 109.3 B/s       Data Size 36.2 KB
W 0.003                             Out 641.2 B/s
Last 6 hours    Last 6 hours        Last 6 hours       Last 2 days
```



Cluster1 VERSION 6.0.5 REGION AWS Mumbai (ap-south-1)

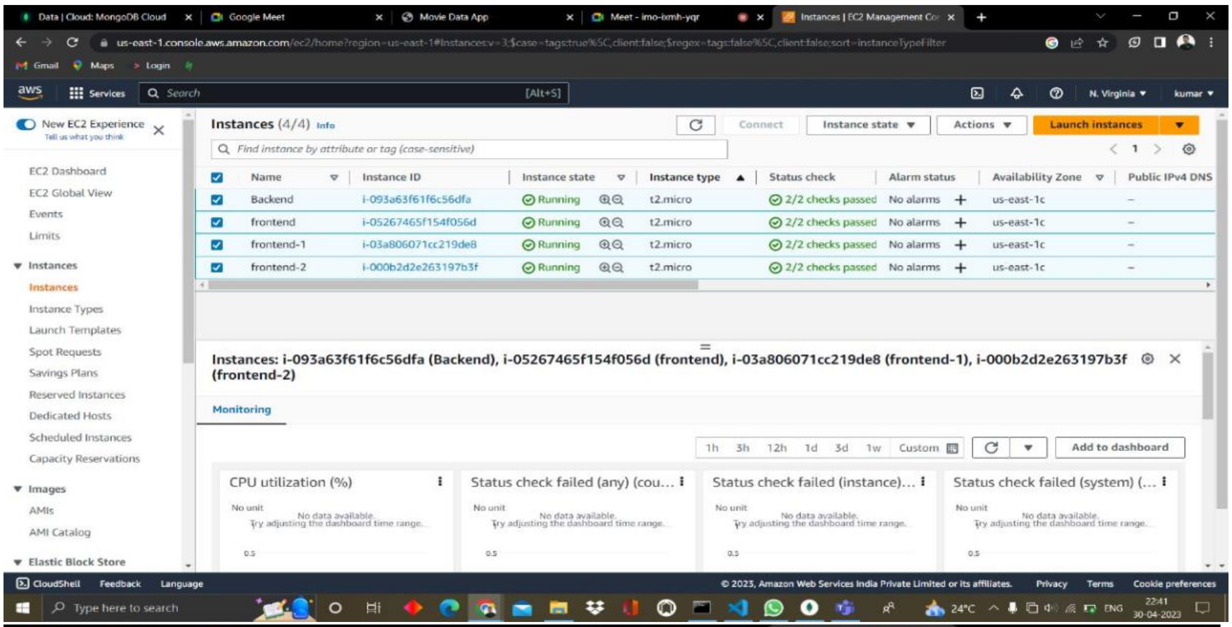
DATABASES: 1 COLLECTIONS: 1

test.movies

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 369B TOTAL DOCUMENTS: 2 INDEXES TOTAL SIZE: 36KB

QUERY RESULTS: 1-1 OF 1

```
_id: ObjectId('644cac3e1817507e09b95991')
title: "RRR"
director: "SS RAJAMOULI"
releaseYear: 2022
poster: "https://res.cloudinary.com/dec6gy3wy/image/upload/v1682746430/abrwygo5…"
__v: 0
```



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:v=3;$case=tags:true%5C,client:false;$regex=tags:false%5C,client:false;sort=instanceTypeFilter

Instances (4/4) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Backend	i-093a63f61f6c56dfa	Running	t2.micro	2/2 checks passed	No alarms	us-east-1c	–
frontend	i-05267465f154f056d	Running	t2.micro	2/2 checks passed	No alarms	us-east-1c	–
frontend-1	i-03a806071cc219de8	Running	t2.micro	2/2 checks passed	No alarms	us-east-1c	–
frontend-2	i-000b2d2e263197b3f	Running	t2.micro	2/2 checks passed	No alarms	us-east-1c	–

Instances: i-093a63f61f6c56dfa (Backend), i-05267465f154f056d (frontend), i-03a806071cc219de8 (frontend-1), i-000b2d2e263197b3f (frontend-2)

Monitoring

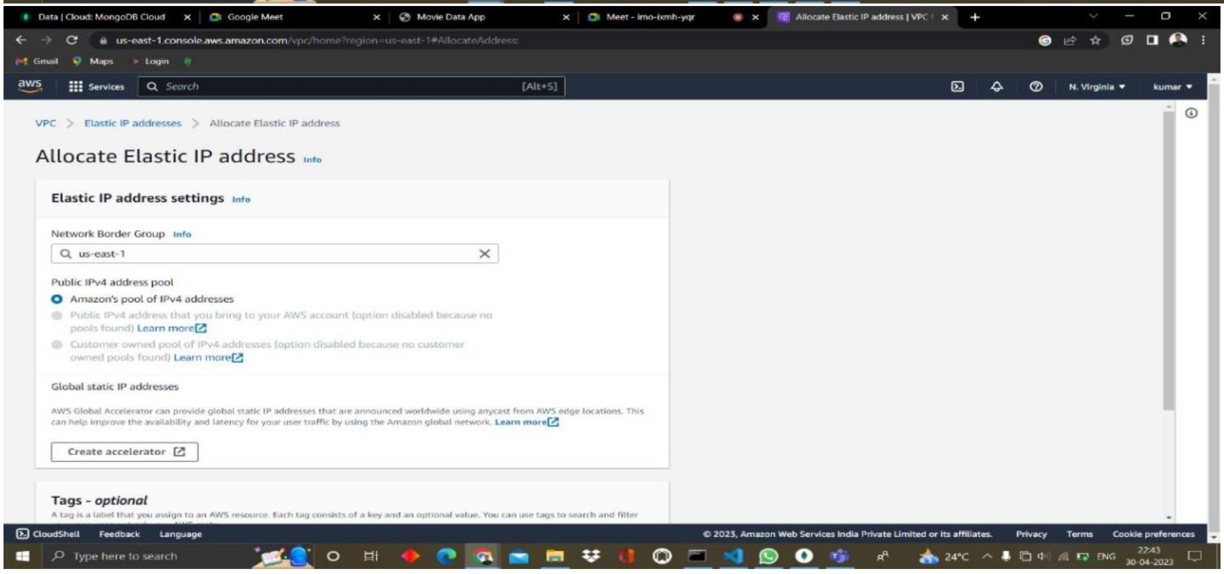
CPU utilization (%) — No data available.

Status check failed (any) (cou… — No data available.

Status check failed (instance)… — No data available.

Status check failed (system) (… — No data available.

© 2023, Amazon Web Services India Private Limited or its affiliates.



us-east-1.console.aws.amazon.com/vpc/home?region=us-east-1#AllocateAddress:

VPC > Elastic IP addresses > Allocate Elastic IP address

Allocate Elastic IP address

Elastic IP address settings

Network Border Group: us-east-1

Public IPv4 address pool

- [✓ Amazon's pool of IPv4 addresses]
- Public IPv4 address that you bring to your AWS account (option disabled because no pools found) Learn more
- Customer owned pool of IPv4 addresses (option disabled because no customer owned pools found) Learn more

Global static IP addresses

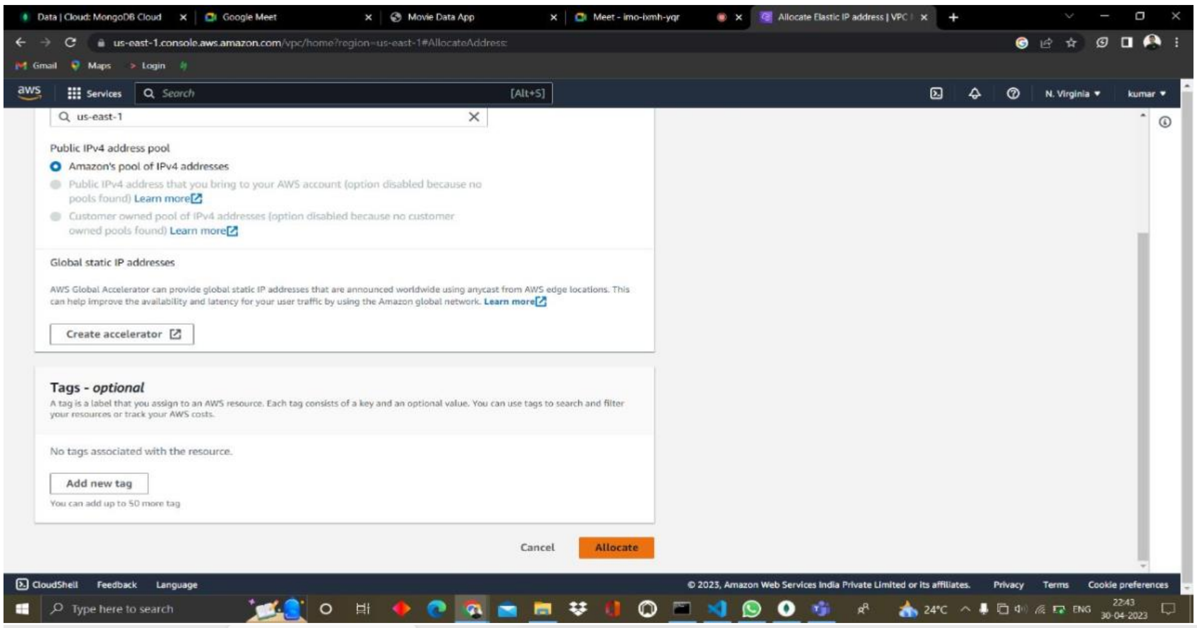
AWS Global Accelerator can provide global static IP addresses that are announced worldwide using anycast from AWS edge locations. This can help improve the availability and latency for your user traffic by using the Amazon global network. Learn more

Create accelerator

Tags - *optional*

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter

© 2023, Amazon Web Services India Private Limited or its affiliates.



us-east-1

Public IPv4 address pool

- Amazon's pool of IPv4 addresses
- Public IPv4 address that you bring to your AWS account (option disabled because no pools found) Learn more
- Customer owned pool of IPv4 addresses (option disabled because no customer owned pools found) Learn more

Global static IP addresses

AWS Global Accelerator can provide global static IP addresses that are announced worldwide using anycast from AWS edge locations. This can help improve the availability and latency for your user traffic by using the Amazon global network. Learn more

Create accelerator

Tags - *optional*

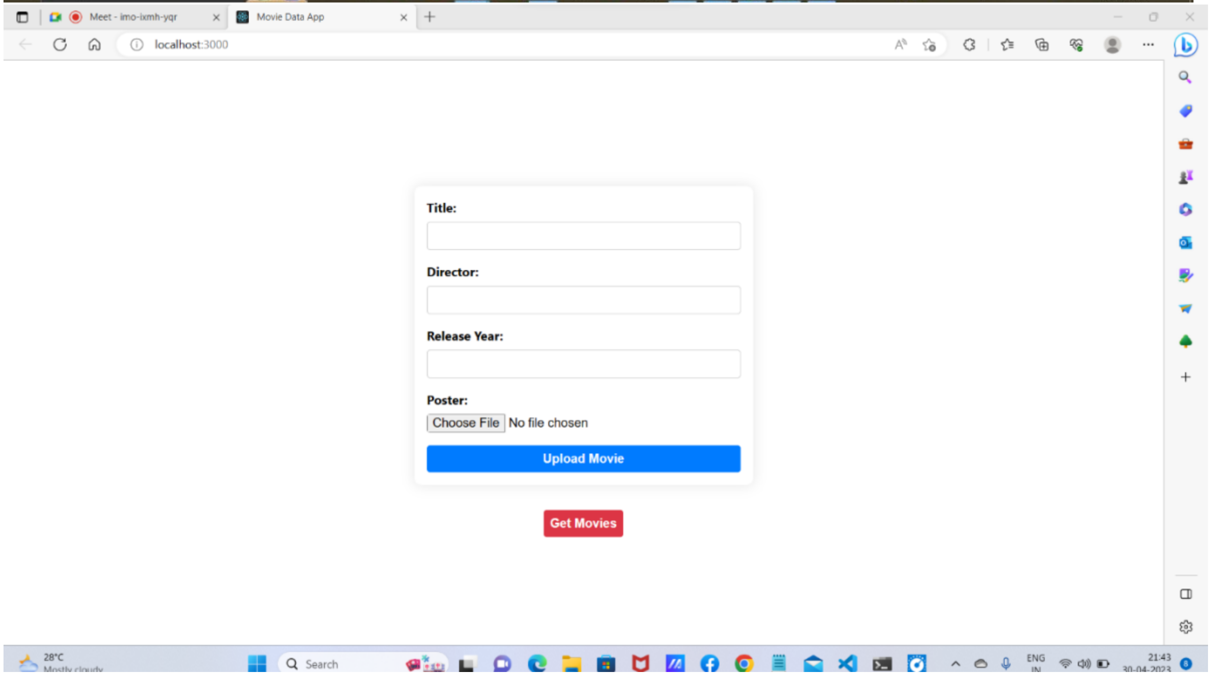
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

Add new tag

You can add up to 50 more tag

Cancel Allocate



Use AWS S3 for storing images: Replace local storage with AWS S3 storage for storing images. Use the multer-s3 library to upload and retrieve images from S3.

Replace local database with Atlas MongoDB cloud infrastructure: Use Atlas MongoDB to host the database in the cloud. Migrate the data from the local MongoDB to Atlas MongoDB.

Deploy Backend in EC2 instance and attach Elastic IP: Use Docker to containerize the backend application and deploy it on an EC2 instance. Use Elastic IP to assign a static IP address to the instance.

Modify Frontend code to fetch data from Backend: Update the frontend code to fetch data from the backend API instead of local storage.

Deploy Frontend using Docker into EC2 instance: Containerize the frontend application and deploy it on an EC2 instance using Docker. Ensure that the frontend can communicate with the backend API running on a separate EC2 instance.

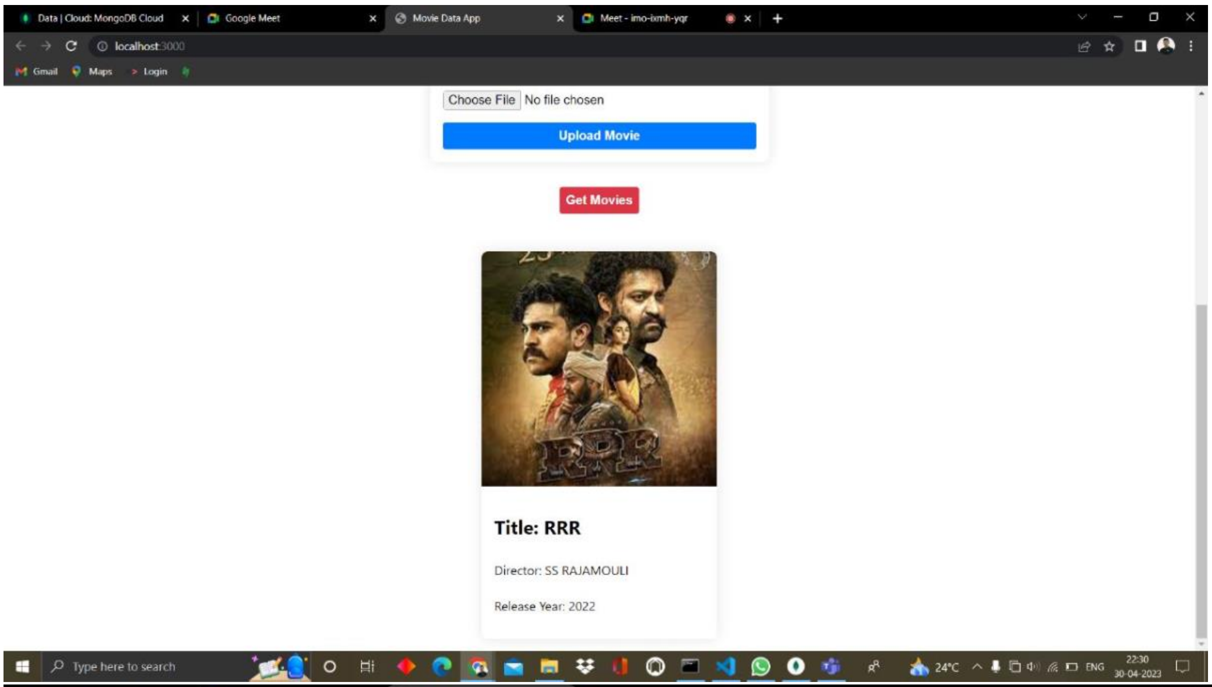
Create Load balancer and attach to scale website traffic: Use an Elastic Load Balancer (ELB) to distribute traffic to multiple EC2 instances running the backend application. Configure Auto Scaling to automatically add or remove EC2 instances based on traffic demand.

Use DNS to point to IP: Use Amazon Route 53 to register a domain name and point it to the Elastic IP address assigned to the load balancer.

Create AWS deployment diagram and suggest methods to improve it: Create a diagram that outlines the different components of the infrastructure, their interactions, and how they are deployed. Suggest methods to improve the architecture, such as using Amazon CloudFront to cache static content, using Amazon RDS to manage the database, and using AWS Lambda to handle serverless functions.

Host docker images into AWS ECR/Docker hub: Store the Docker images in either AWS Elastic Container Registry (ECR) or Docker Hub for easy deployment and management.

By following these steps, the "Movie listing" website can be successfully deployed in the cloud infrastructure (AWS) with proper scaling.



Choose File No file chosen

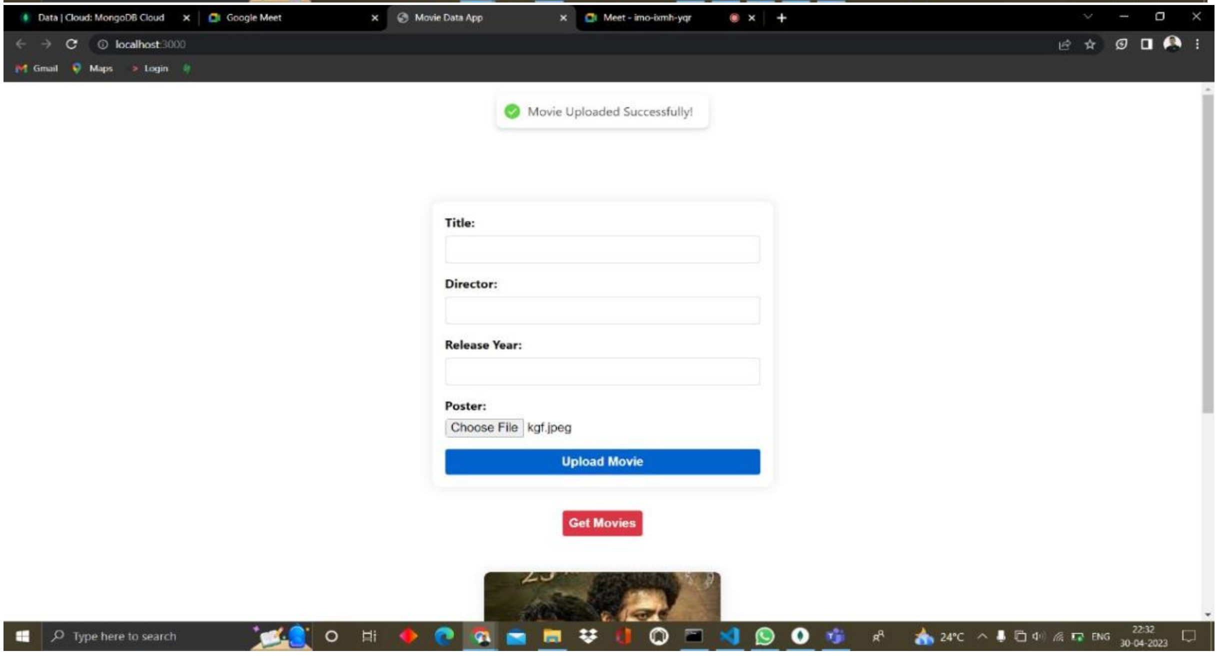
Upload Movie

Get Movies

Title: RRR

Director: SS RAJAMOULI

Release Year: 2022



Movie Uploaded Successfully!

Title:

Director:

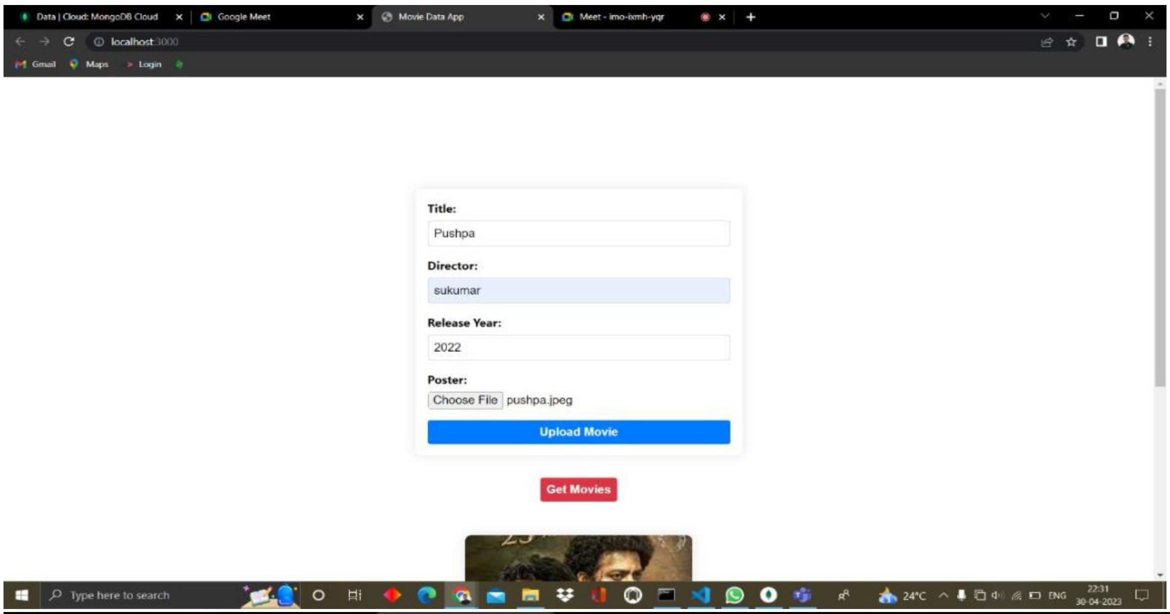
Release Year:

Poster:

Choose File kgf.jpeg

Upload Movie

Get Movies


Title:

Pushpa

Director:

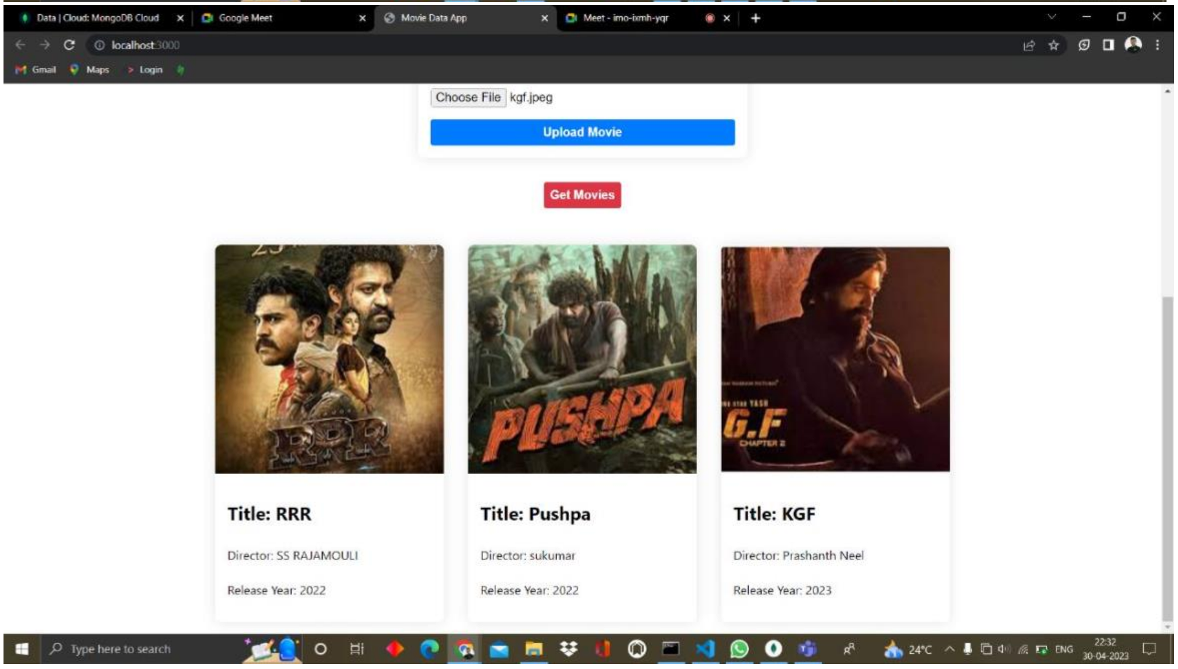
sukumar

Release Year:

2022

Poster:

Choose File pushpa.jpeg


Choose File kgf.jpeg

Title: RRR

Director: SS RAJAMOULI

Release Year: 2022

Title: Pushpa

Director: sukumar

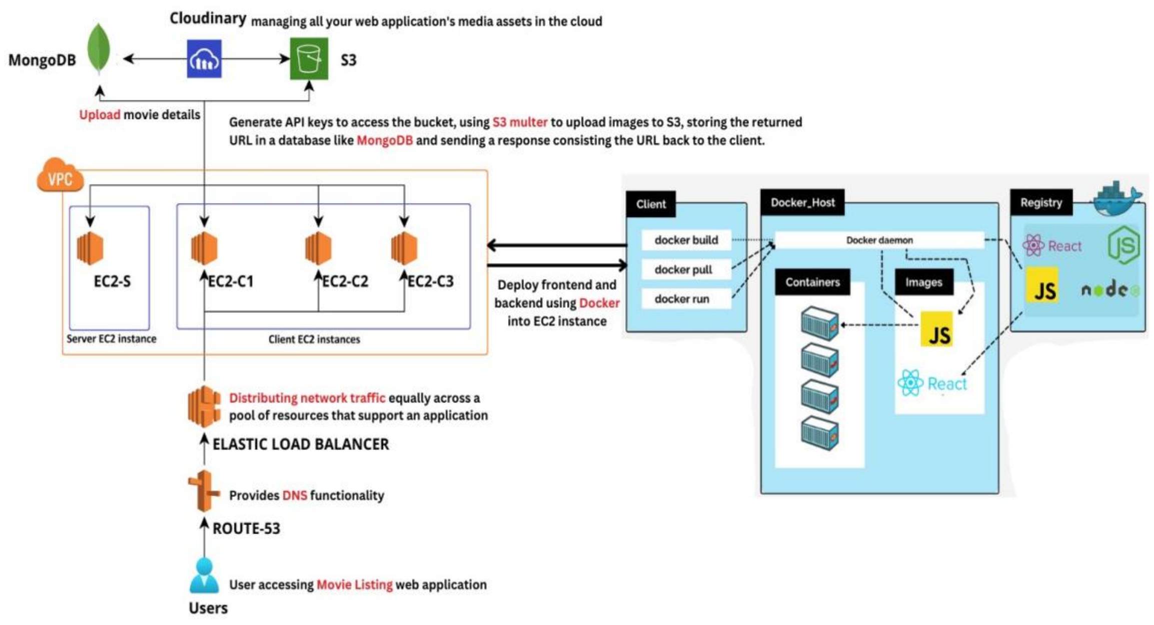
Release Year: 2022

Title: KGF

Director: Prashanth Neel

Release Year: 2023

Deploying diagram:


Cloudinary managing all your web application's media assets in the cloud
MongoDB
S3
Upload movie details
Generate API keys to access the bucket, using S3 multer to upload images to S3, storing the returned URL in a database like MongoDB and sending a response consisting the URL back to the client.
VPC
EC2-S
EC2-C1
EC2-C2
EC2-C3
Server EC2 instance
Client EC2 instances
Deploy frontend and backend using Docker into EC2 instance
Client
docker build
docker pull
docker run
Docker_Host
Docker daemon
Containers
Images
Registry
React
JS
Distributing network traffic equally across a pool of resources that support an application
ELASTIC LOAD BALANCER
Provides DNS functionality
ROUTE-53
User accessing Movie Listing web application
Users

Group contributon:

Connecting the Server with Atlas MongoDB and Creating IAM user and Configuring S3-bucket — **Robin Varghese**

Deploying server on EC2 using Docker and Deploying Containerization of the code using Dockerfile.. — **Aditya Anaparthi**

Containerization and Configuring the Application Code with S3-multer and Creation a target group and Load BalancerPreparing Project Documentation. — **Manikanta Kumar**

Conclusion:

Given Capstone project involves ReactJs, NodeJS, MongoDB, S3-multer and AWS. Our
frontend application allows us to upload movies along with posters, Title, Director name and
Movie released year and also, we have 2 features i.e., upload movies and get movies. We
used S3 to store Posters. We deployed frontend and backend using Docker. User can get
movies and the details of movies