CAPSTONE PROJECT

PROJECT TITLE

MOVIE LISTING

TEAM - 3:

- IMMINNI SARITHA 20P31A1218
- SUBHA SARANYA BHIMALA 21P35A1201
- NEKKANTI SAI CHAITANYA 20P31A1239
- TEJASWINI THAMBABATHULA 20P31A1233
- KAKI SHIVA SATYA NARAYANA 21A95A0208
- CHIRUKOTI LIKITH 21P35A1202
- KOPPULA RAVINDRA REDDY 20MH1A0590
- NEETIPALLI SIVA GANESH 20MH1A0541

- # PROJECT OVERVIEW

  You are provided with a “Movie listing” website which uses ReactJS as frontend, NodeJS as backend and MongoDB as database. Users can upload movie details where it uses the local storage to store the images. You are required to deploy this entire website into the cloud infrastructure (AWS) with proper scaling.

  Use AWS S3 for storing images. You can use the multer-s3 library.

  Replace local database with Atlas MongoDB cloud infrastructure to take the database into the cloud.

  Deploy Backend in EC2 instance and attach Elastic IP to this instance. Deployment should be done using Docker.

  Modify the Frontend code to be able to fetch data from Backend. Finally, deploy frontend using docker into EC2 instance.

  Create Load balancer and attach load balancer to be able to properly scale the website traffic.

  Use DNS to point to this IP.

  Create a proper AWS deployment diagram and suggest methods to improve it.

  Host docker images into AWS ECR/Docker hub.

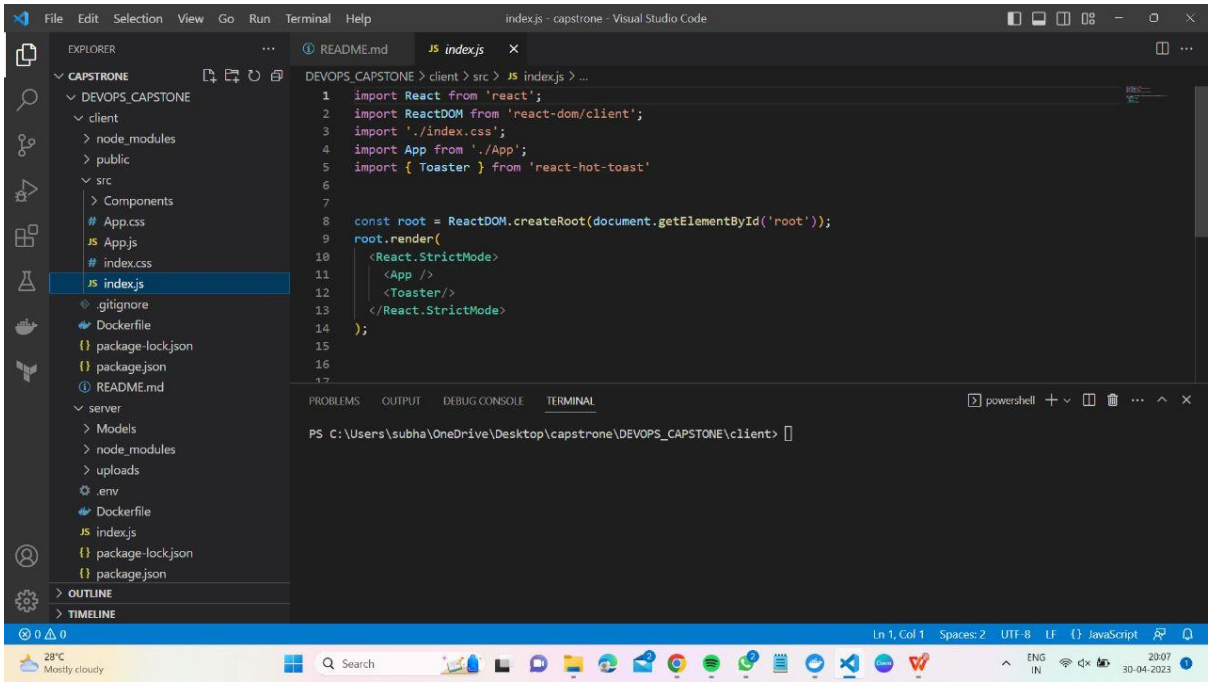
- # TECHNOLOGIES USED

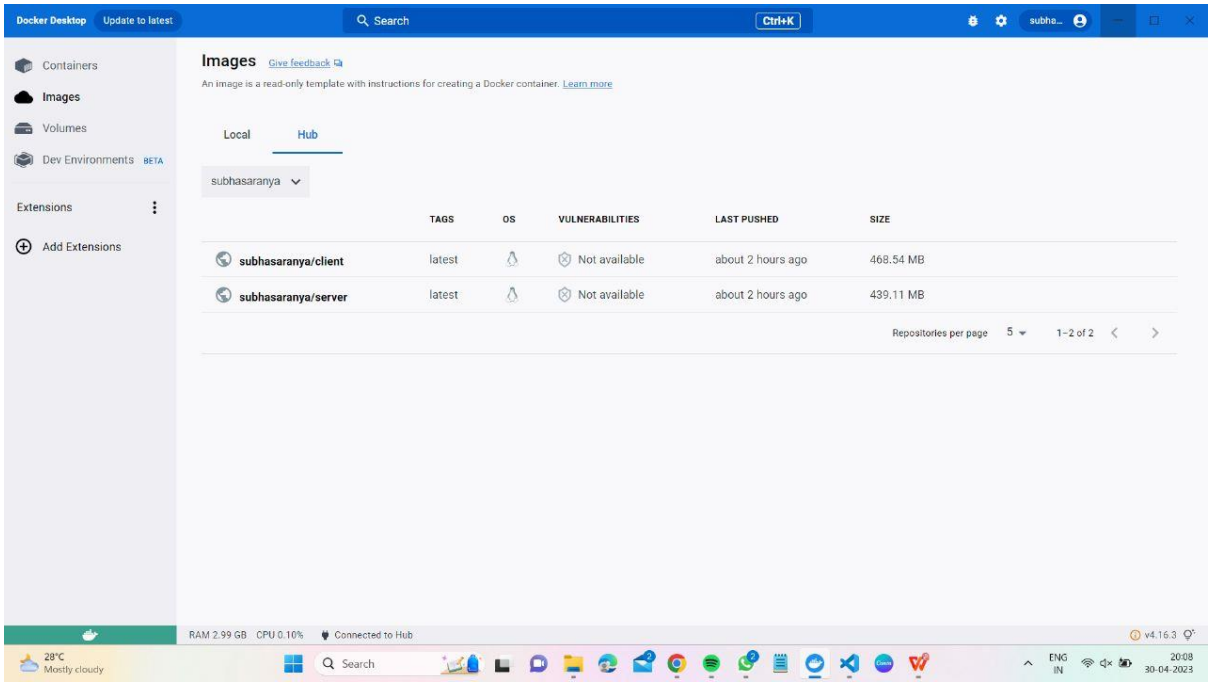
  ReactJS: The React.js framework is an open-source JavaScript framework and library developed by Facebook. It’s used for building interactive user interfaces and web applications quickly and efficiently with significantly less code than you would with vanilla JavaScript. In our project.

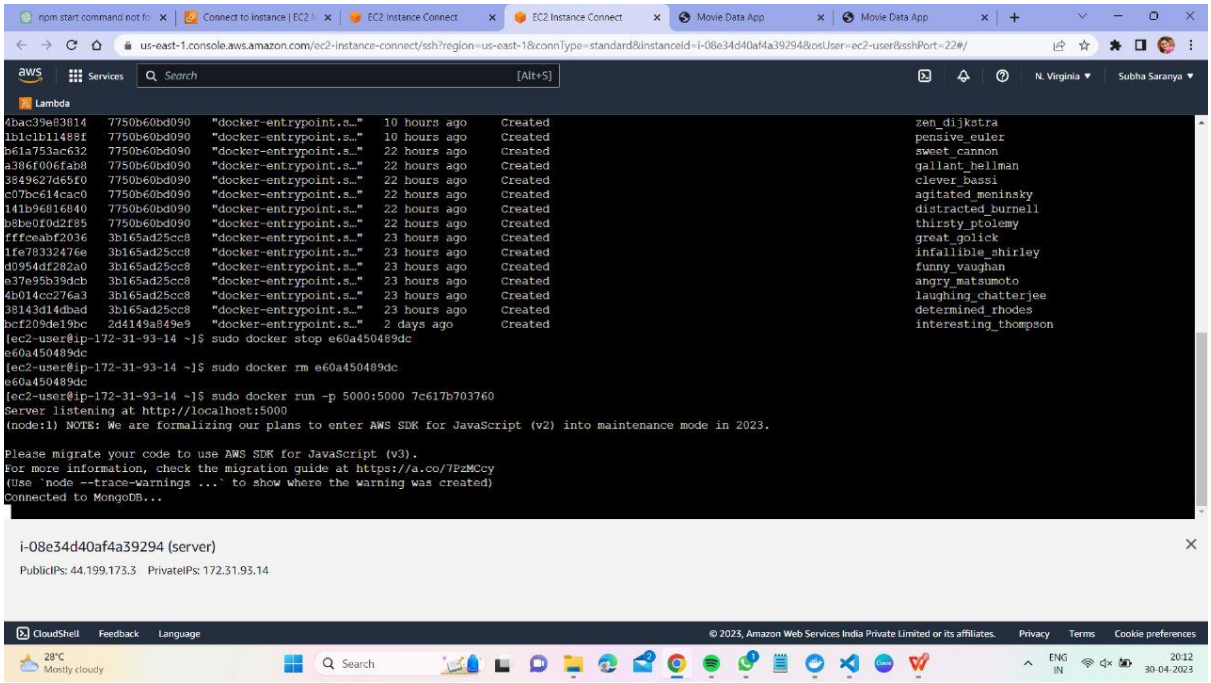
  NodeJS: If we talk about any application then the part with which the user is interacting is basically the frontend of our website whereas there are many things which happen in the background or in the backend of our website. Basically, there are three parts of any application, one is Frontend with which the users are interacting, then comes to the backend server and backend database. For backend servers we can use NodeJS.

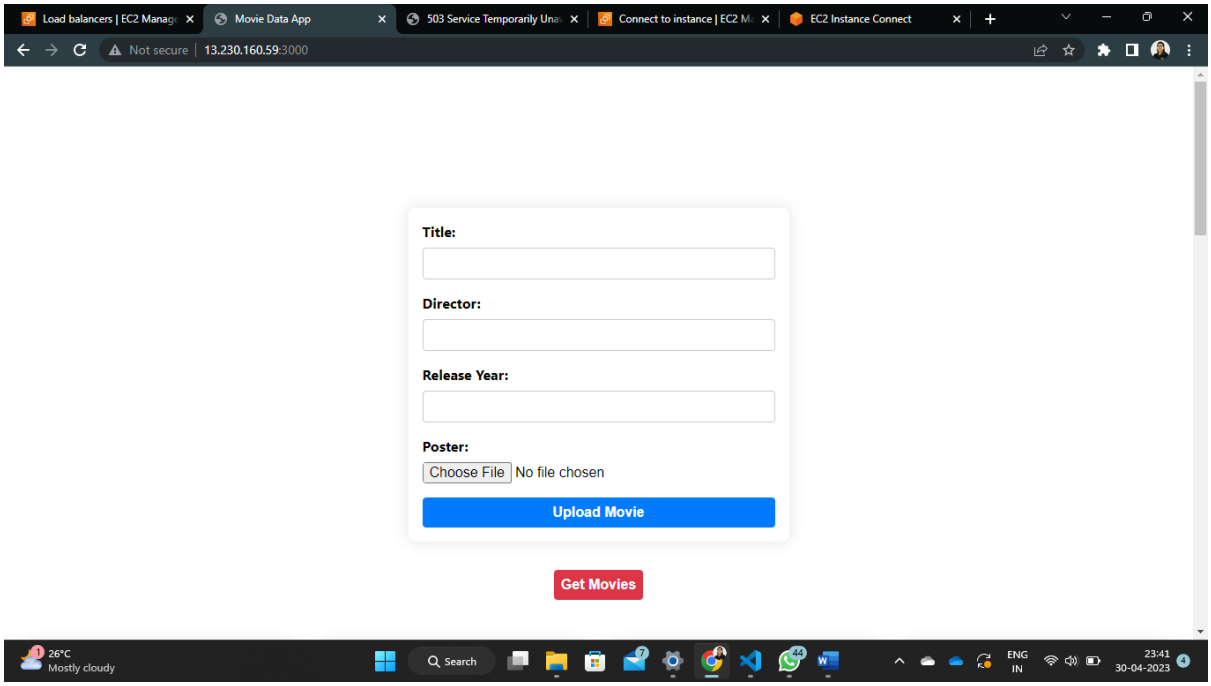
  MongoDB: MongoDB is a document database used to build highly available and scalable internet applications. With its flexible schema approach, it’s popular with development teams using agile methodologies. Offering drivers for all major programming languages, MongoDB allows you to immediately start building your application without spending time configuring a database.

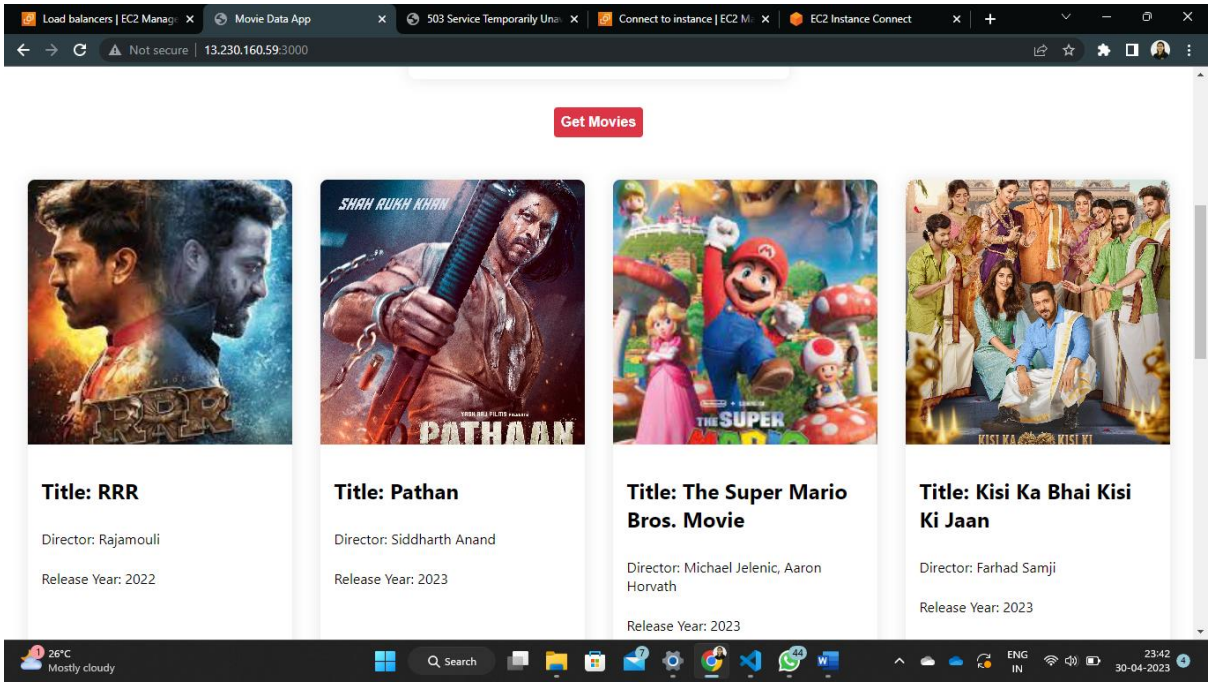
PROOF OF CONCEPT






npm start command not fo
Connect to instance | EC2
EC2 Instance Connect
EC2 Instance Connect
Movie Data App
Movie Data App
us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-08e34d40af4a39294&osUser=ec2-user&sshPort=22#/
aws
Services
Search
[Alt+S]
N. Virginia
Subha Saranya
Lambda
4bac39e83814 7750b60bd090 "docker-entrypoint.s…" 10 hours ago Created zen_dijkstra
1b1c1b11488f 7750b60bd090 "docker-entrypoint.s…" 10 hours ago Created pensive_euler
b61a753ac632 7750b60bd090 "docker-entrypoint.s…" 22 hours ago Created sweet_cannon
a386f006fab8 7750b60bd090 "docker-entrypoint.s…" 22 hours ago Created gallant_hellman
3849627d65f0 7750b60bd090 "docker-entrypoint.s…" 22 hours ago Created clever_bassi
c07bc614cac0 7750b60bd090 "docker-entrypoint.s…" 22 hours ago Created agitated_meninsky
141b96816840 7750b60bd090 "docker-entrypoint.s…" 22 hours ago Created distracted_burnell
b8be0f0d2f85 7750b60bd090 "docker-entrypoint.s…" 22 hours ago Created thirsty_ptolemy
fffceabf2036 3b165ad25cc8 "docker-entrypoint.s…" 23 hours ago Created great_golick
1fe78332476e 3b165ad25cc8 "docker-entrypoint.s…" 23 hours ago Created infallible_shirley
d0954df282a0 3b165ad25cc8 "docker-entrypoint.s…" 23 hours ago Created funny_vaughan
e37e95b39dcb 3b165ad25cc8 "docker-entrypoint.s…" 23 hours ago Created angry_matsumoto
4b014cc276a3 3b165ad25cc8 "docker-entrypoint.s…" 23 hours ago Created laughing_chatterjee
38143d14dbad 3b165ad25cc8 "docker-entrypoint.s…" 23 hours ago Created determined_rhodes
bcf209de19bc 2d4149a849e9 "docker-entrypoint.s…" 2 days ago Created interesting_thompson
[ec2-user@ip-172-31-93-14 ~]$ sudo docker stop e60a450489dc
e60a450489dc
[ec2-user@ip-172-31-93-14 ~]$ sudo docker rm e60a450489dc
e60a450489dc
[ec2-user@ip-172-31-93-14 ~]$ sudo docker run -p 5000:5000 7c617b703760
Server listening at http://localhost:5000
(node:1) 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)
Connected to MongoDB...
i-08e34d40af4a39294 (server)
PublicIPs: 44.199.173.3 PrivateIPs: 172.31.93.14
CloudShell
Feedback
Language
© 2023, Amazon Web Services India Private Limited or its affiliates.
Privacy
Terms
Cookie preferences
28°C
Mostly cloudy
Search
ENG
IN
20:12
30-04-2023


Load balancers | EC2 Manag
Movie Data App
503 Service Temporarily Una
Connect to instance | EC2 M
EC2 Instance Connect
Not secure | 13.230.160.59:3000
Title:
Director:
Release Year:
Poster:
Choose File
No file chosen
Upload Movie
Get Movies
26°C
Mostly cloudy
Search
ENG
IN
23:41
30-04-2023



- **LINK OF THE PROJECT**

  http://13.230.160.59:3000

- **CONCLUSION**

  In conclusion, the deployment of the frontend website for movie listing using AWS Cloud, Docker, and MongoDB has been a success. The use of these technologies has made the deployment process much more efficient, streamlined, and scalable.

  AWS Cloud provided a flexible and scalable infrastructure that allowed us to easily deploy our website and scale it up or down based on demand. Docker enabled us to create a containerized environment for our application, making it easier to manage dependencies and deploy the application across different environments.

  MongoDB provided a reliable and scalable database solution that allowed us to store and retrieve movie information in real-time. It also allowed us to easily scale our database as our application grew.

  Overall, the deployment process was smooth, and we were able to launch our website successfully. The use of these technologies has made it possible to create a highly scalable and reliable movie listing website. We have also gained valuable experience in deploying applications on the cloud and containerized environments, which will be useful in future projects.