Blogging Website

(Deployment on AWS)

Name: Shammi Kumar

Class: CSE-B , 7th Sem

Roll no : 09920802722

Trainer: Saurabh Dwivedi

Assignments	AWS Services used
Assignment -1	EC2, EBS
Assignment-2	EC2, S3, SNS
Assignment-3	RDS, DynamoDB, CloudWatch
Assignment-4	EC2, ELB, Auto-Scaling, Cloud Watch
Assignment-5	Lambda, API Gateway, S3, DynamoDB

Project Report: BlogVerse

Project Overview

BlogVerse is a full-stack, responsive blogging platform that enables users to create, read, update, and delete blog posts with rich media content. Designed to mimic real-world blogging systems (like Medium or Dev.to), it supports text formatting, media uploads (images, videos, PDFs), and public/private publishing options. The platform focuses on user experience, scalability, and modern web architecture, making it suitable for personal blogging, tech publications, or educational content sharing.

Key Features

- **Rich Blog Post Creation**
  Users can create blog posts with formatted text, embedded images, videos, and attached PDFs, supporting various content types for flexible publishing.
- **Media Upload Support (AWS S3)**
  Upload and preview images, videos, and documents via AWS S3 for scalable and reliable media storage.
- **Secure Authentication System**
  User registration and login with JWT-based authentication and encrypted password storage using bcrypt.
- **Public & Private Post Toggle**
  Authors can control the visibility of each blog post with a single switch to publish publicly or keep it private.
- **Like System with Count**
  Logged-in users can like/unlike posts, with dynamic like counts displayed on each post.
- **User Profiles**
  Each user has a profile page listing their posts and showing personal info like name, email, and total likes.

- **Fully Responsive UI**
  The app is optimized for mobile, tablet, and desktop devices with modern styling using Tailwind CSS.
- **Dockerized Deployment on AWS**
  The project runs inside Docker containers orchestrated with Docker Compose, deployed on an AWS EC2 instance with static file serving via NGINX.

Technologies Used

Frontend

- **React.js:** JavaScript library used to build the dynamic and interactive user interface (UI) of the blogging platform.
- **Vite:** A fast frontend build tool that provides lightning-fast hot reload and optimized production builds.
- **Redux Toolkit**: For managing global state (authentication, post data, likes, etc.) in a scalable and maintainable way.
- **Tailwind CSS**: Utility-first CSS framework used to design responsive, clean, and mobile-friendly UI.
- **React Router DOM**: For client-side navigation and routing across pages.

Backend

- **Node.js**: JavaScript runtime environment used for writing the server-side logic.
- **Express.js**: A minimal and flexible Node.js framework for building RESTful APIs and handling routing, middleware, and HTTP requests.
- **MongoDB:** NoSQL database used to store user data, posts, media URLs, and other structured documents.
- **Mongoose:** ODM (Object Data Modeling) library that provides schema-based modeling and queries for MongoDB.

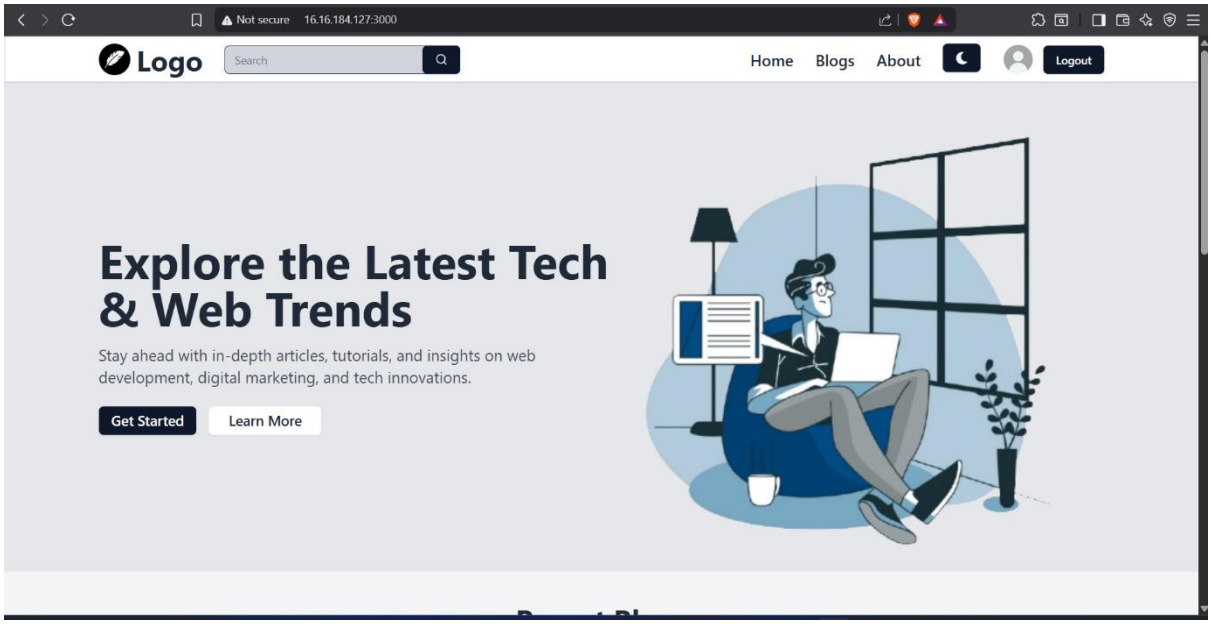
AWS Services

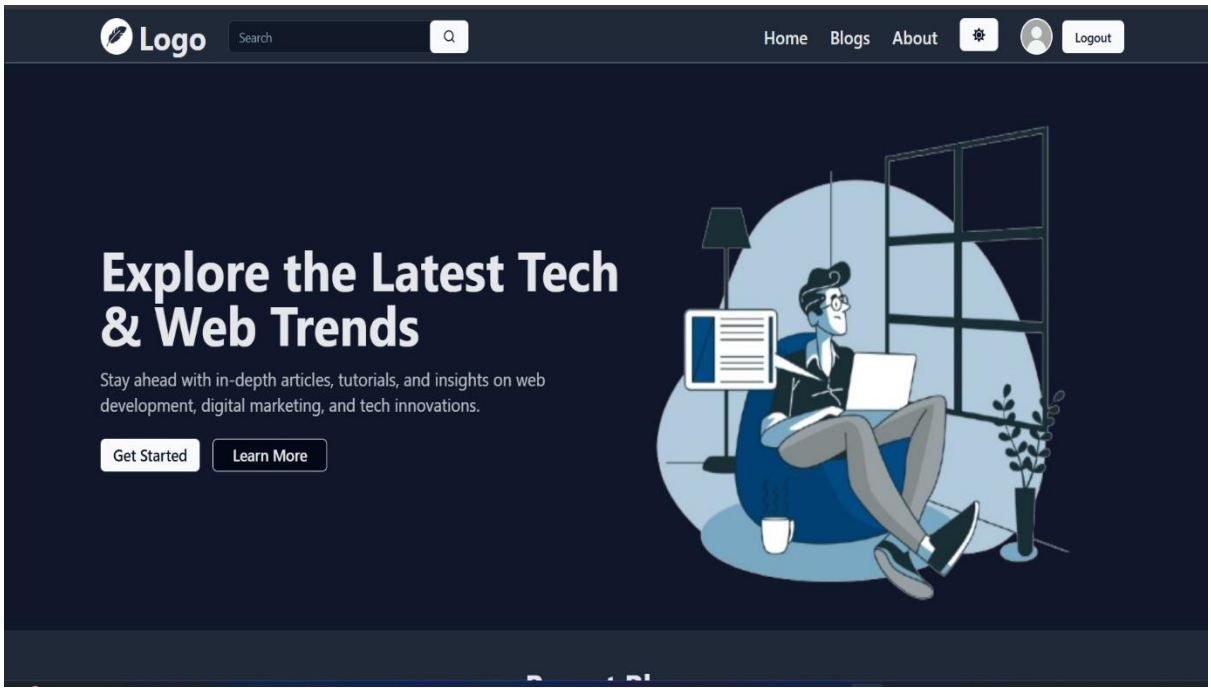
- **AWS EC2**: Virtual Linux server used to host the entire application stack (frontend, backend, database).
- **AWS S3**: Cloud object storage service used for uploading and storing images, videos, and PDF files.
- **AWS SNS** : Simple Notification Service used for sending alerts or report notifications (optional feature).
- **AWS Cloudwatch** : AWS CloudWatch is a monitoring and observability service that provides real-time visibility into your application's performance, logs, and system metrics.

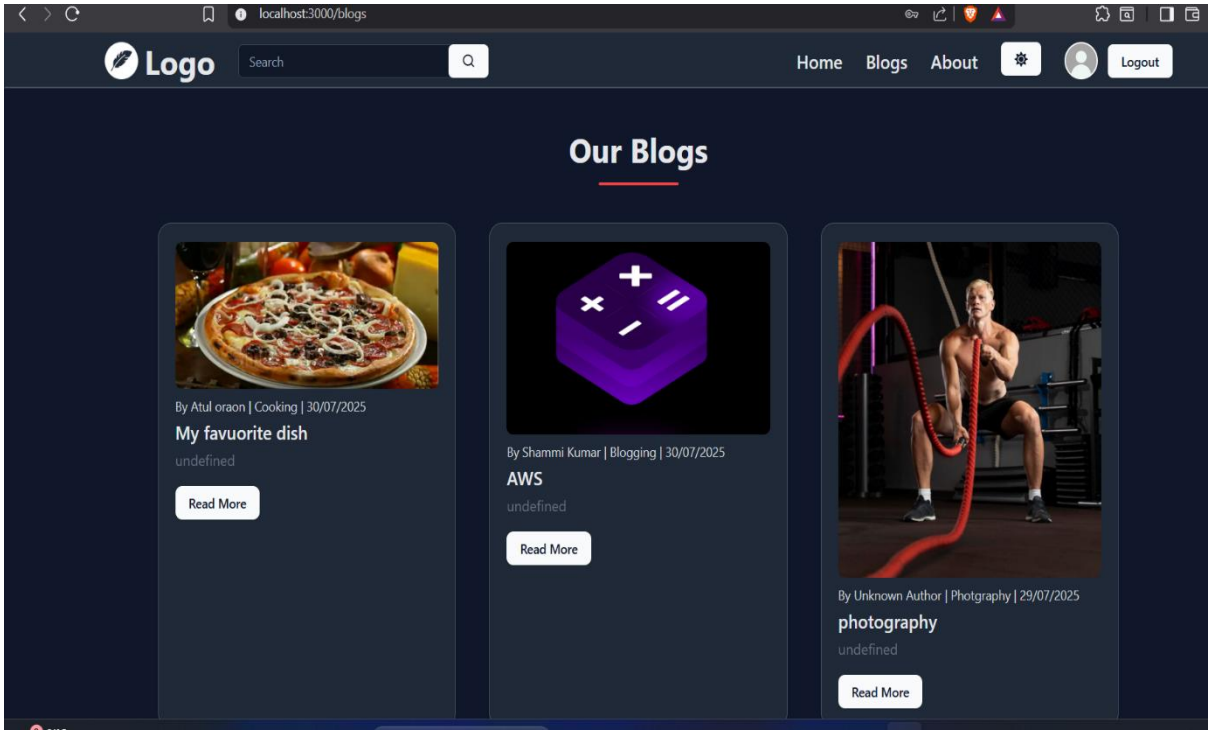
Implementation Steps:

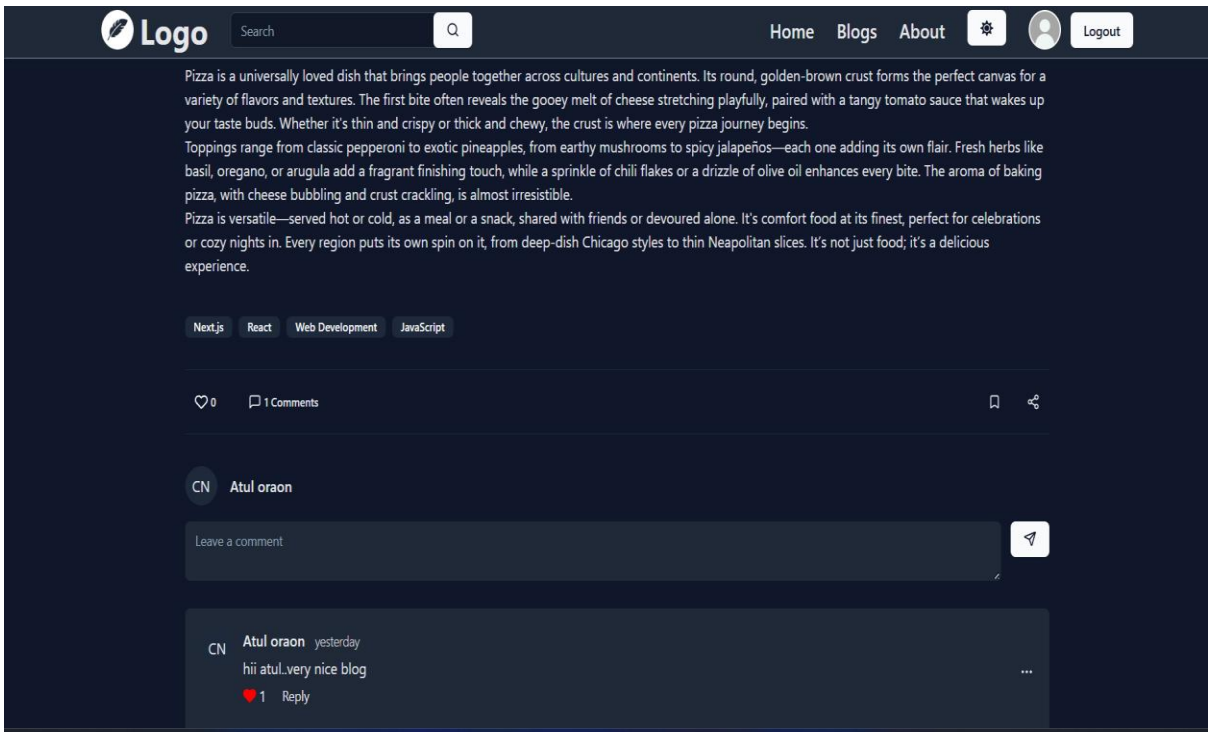
1. Created a React project using Vite with JavaScript support

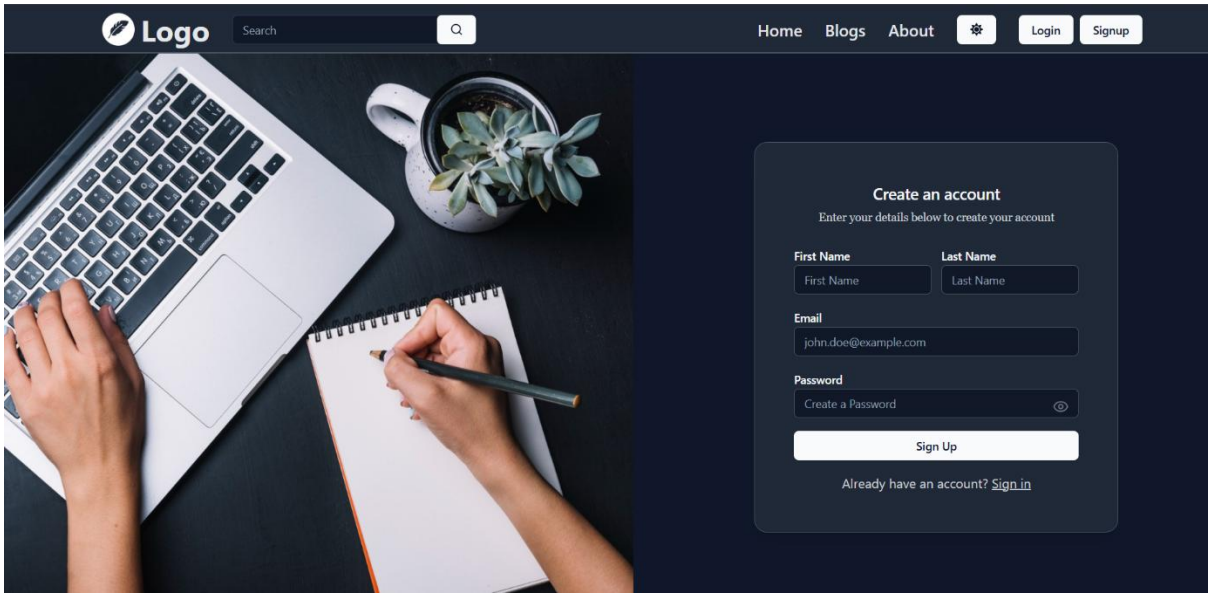
2. **Website Demonstration (Screenshots)**

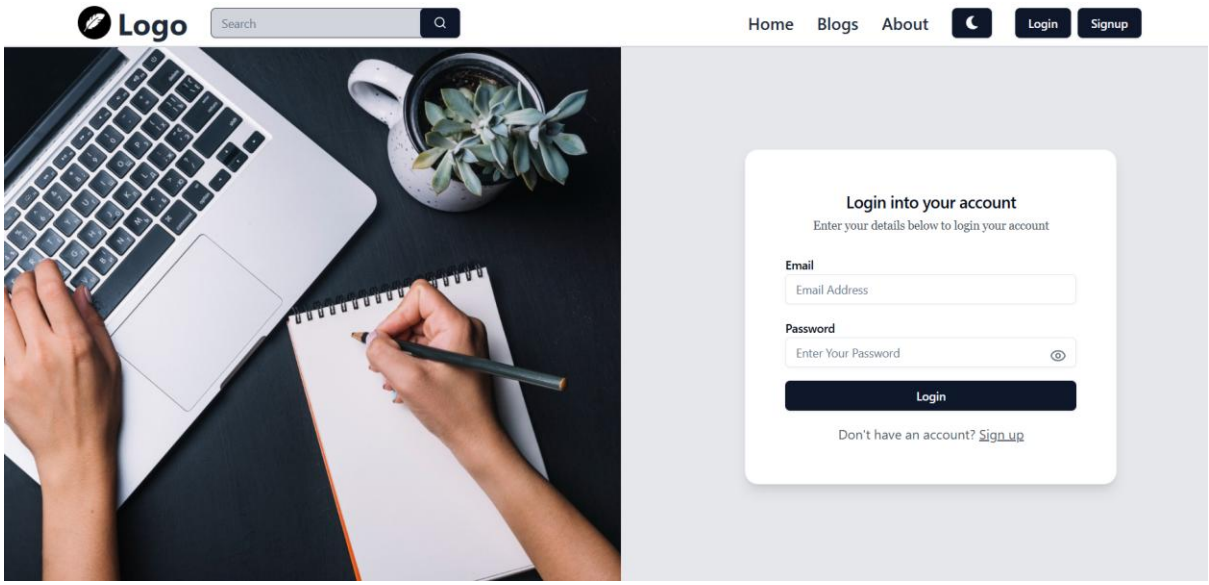


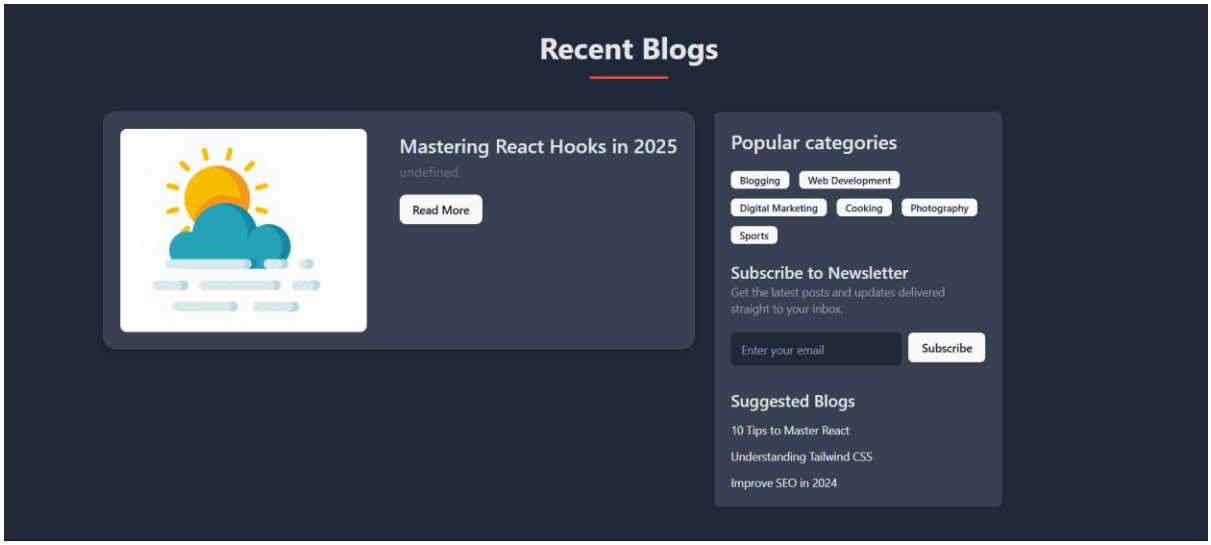


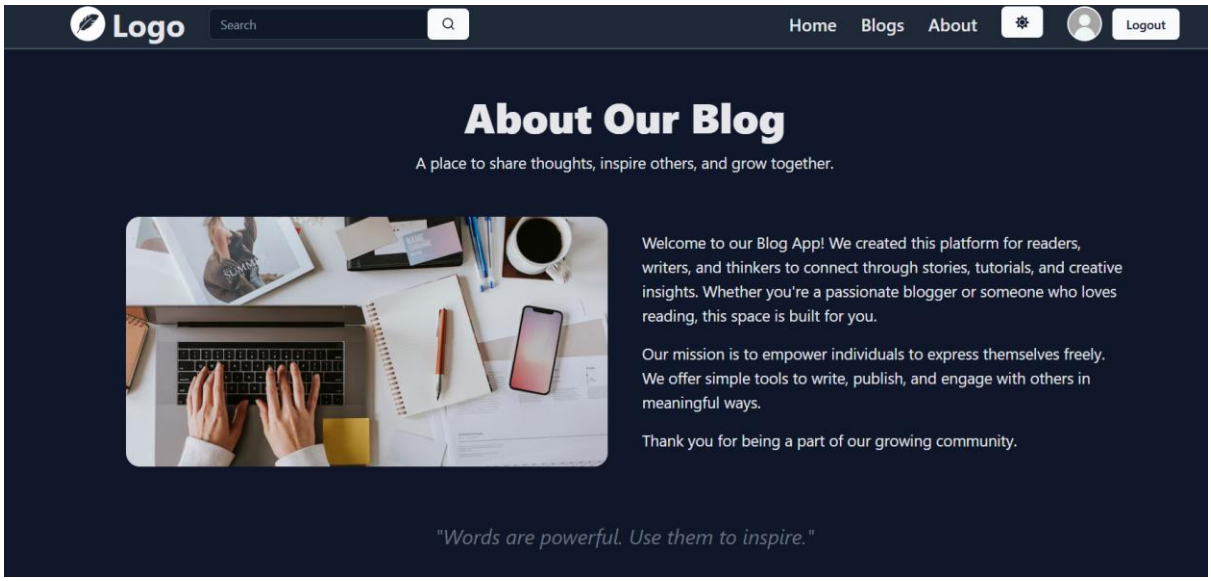

localhost:3000/blogs
Logo
Search
Home
Blogs
About
Logout
Our Blogs
By Atul oraon | Cooking | 30/07/2025
My favuorite dish
undefined
Read More
By Shammi Kumar | Blogging | 30/07/2025
AWS
undefined
Read More
By Unknown Author | Photgraphy | 29/07/2025
photography
undefined
Read More

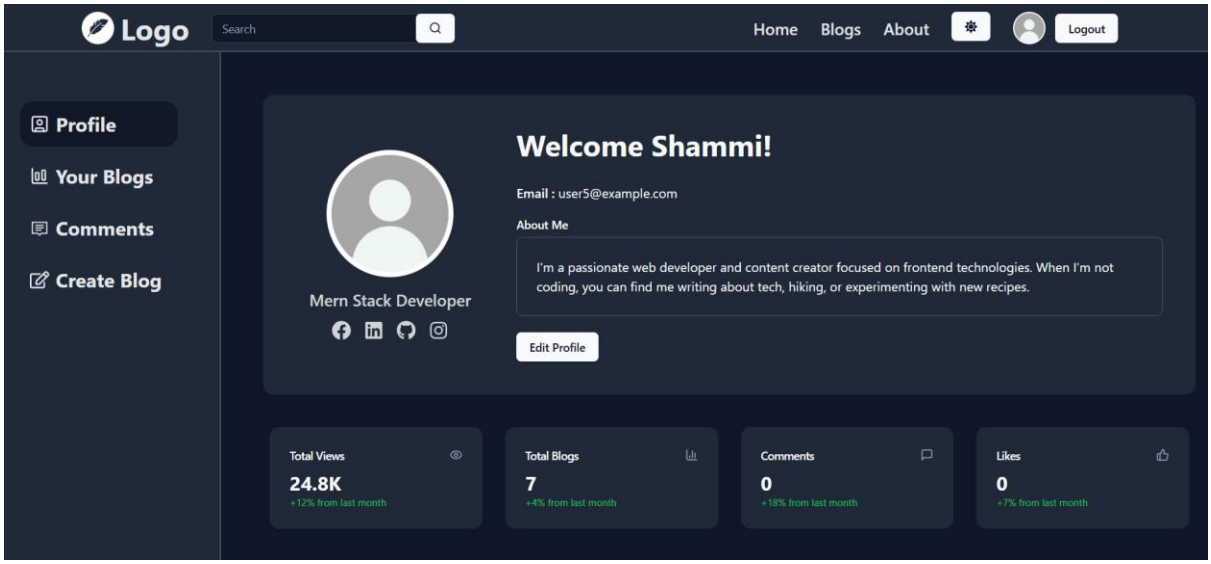

Logo
Search
Home
Blogs
About
Logout
Pizza is a universally loved dish that brings people together across cultures and continents. Its round, golden-brown crust forms the perfect canvas for a variety of flavors and textures. The first bite often reveals the gooey melt of cheese stretching playfully, paired with a tangy tomato sauce that wakes up your taste buds. Whether it's thin and crispy or thick and chewy, the crust is where every pizza journey begins.
Toppings range from classic pepperoni to exotic pineapples, from earthy mushrooms to spicy jalapeños—each one adding its own flair. Fresh herbs like basil, oregano, or arugula add a fragrant finishing touch, while a sprinkle of chili flakes or a drizzle of olive oil enhances every bite. The aroma of baking pizza, with cheese bubbling and crust crackling, is almost irresistible.
Pizza is versatile—served hot or cold, as a meal or a snack, shared with friends or devoured alone. It's comfort food at its finest, perfect for celebrations or cozy nights in. Every region puts its own spin on it, from deep-dish Chicago styles to thin Neapolitan slices. It's not just food; it's a delicious experience.
Next.js
React
Web Development
JavaScript
0
1 Comments
CN
Atul oraon
Leave a comment
CN
Atul oraon yesterday
hii atul..very nice blog
1
Reply


Logo
Search
Home
Blogs
About
Login
Signup
Create an account
Enter your details below to create your account
First Name
Last Name
First Name
Last Name
Email
john.doe@example.com
Password
Create a Password
Sign Up
Already have an account? Sign in


Logo
Search
Home
Blogs
About
Login
Signup
Login into your account
Enter your details below to login your account
Email
Email Address
Password
Enter Your Password
Login
Don't have an account? Sign up

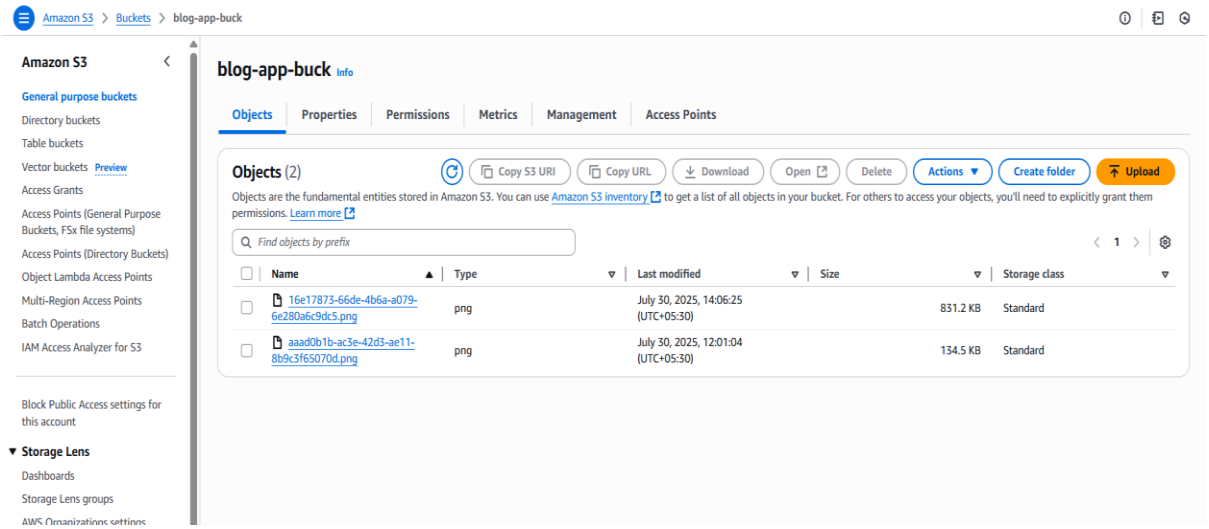

Recent Blogs
Mastering React Hooks in 2025
undefined
Read More
Popular categories
Blogging
Web Development
Digital Marketing
Cooking
Photography
Sports
Subscribe to Newsletter
Get the latest posts and updates delivered straight to your inbox.
Enter your email
Subscribe
Suggested Blogs
10 Tips to Master React
Understanding Tailwind CSS
Improve SEO in 2024


Logo
Search
Home
Blogs
About
Logout
About Our Blog
A place to share thoughts, inspire others, and grow together.
Welcome to our Blog App! We created this platform for readers, writers, and thinkers to connect through stories, tutorials, and creative insights. Whether you're a passionate blogger or someone who loves reading, this space is built for you.
Our mission is to empower individuals to express themselves freely. We offer simple tools to write, publish, and engage with others in meaningful ways.
Thank you for being a part of our growing community.
"Words are powerful. Use them to inspire."

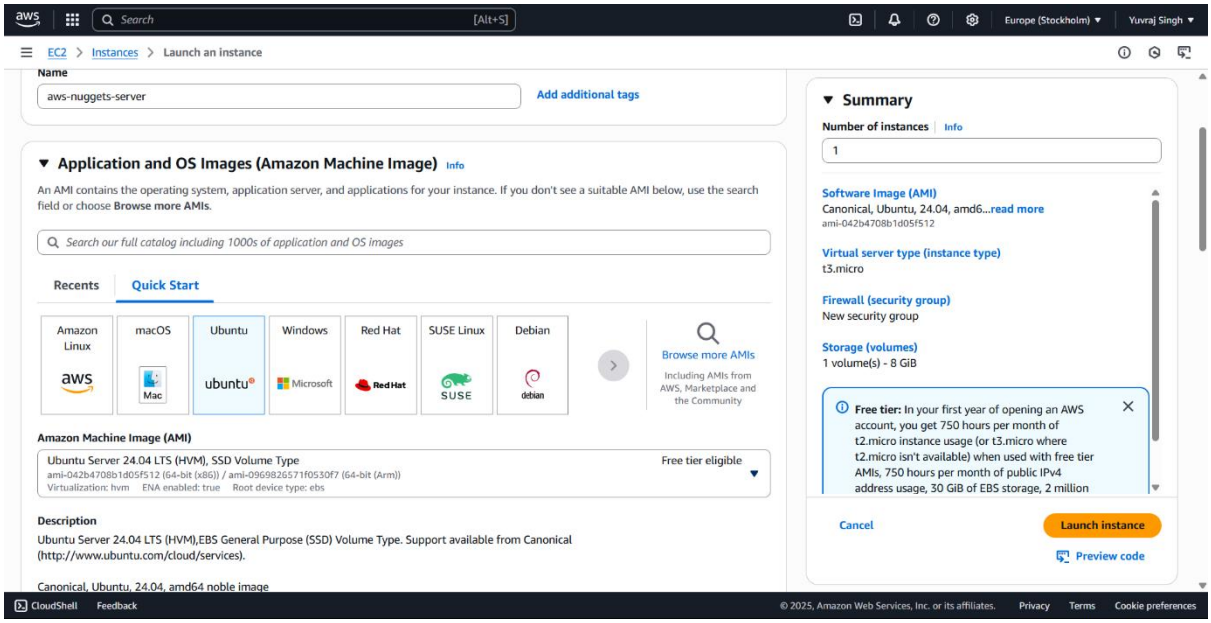

Logo
Search
Home
Blogs
About
Logout
Profile
Your Blogs
Comments
Create Blog
Mern Stack Developer
Welcome Shammi!
Email : user5@example.com
About Me
I'm a passionate web developer and content creator focused on frontend technologies. When I'm not coding, you can find me writing about tech, hiking, or experimenting with new recipes.
Edit Profile
Total Views
24.8K
+12% from last month
Total Blogs
7
+4% from last month
Comments
0
+18% from last month
Likes
0
+7% from last month

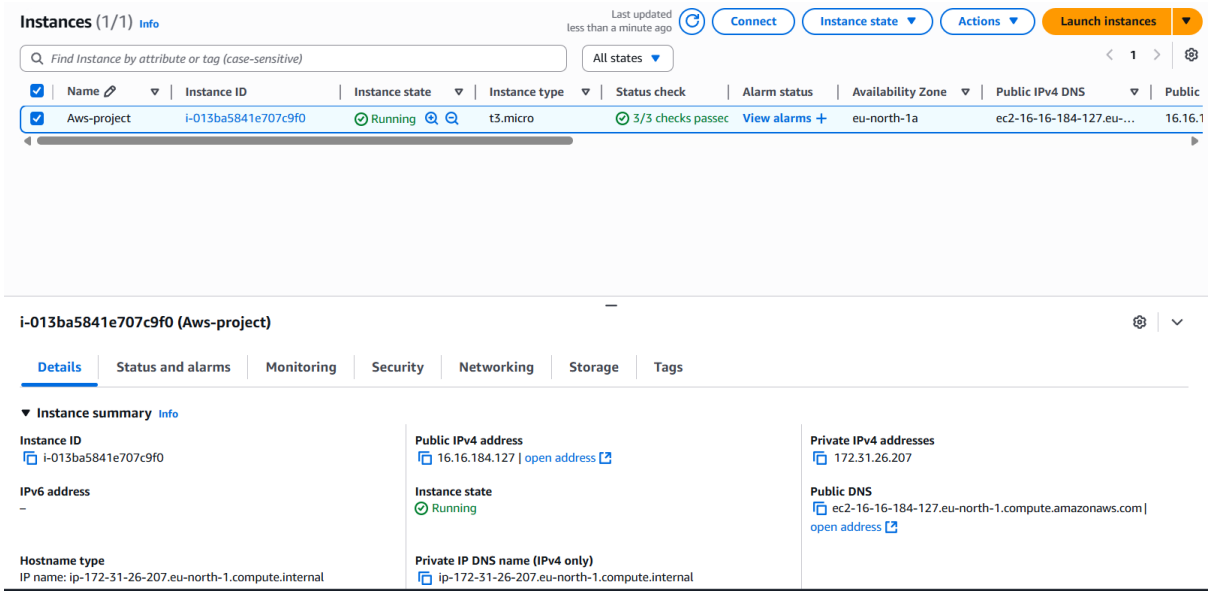
3) AWS Services Used :

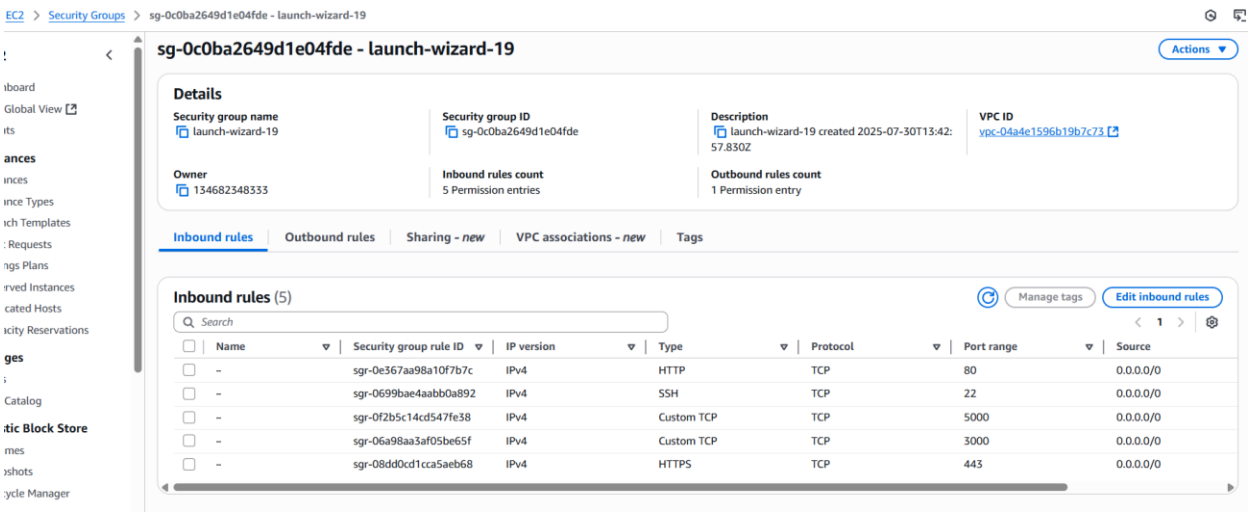
- **AWS S3**



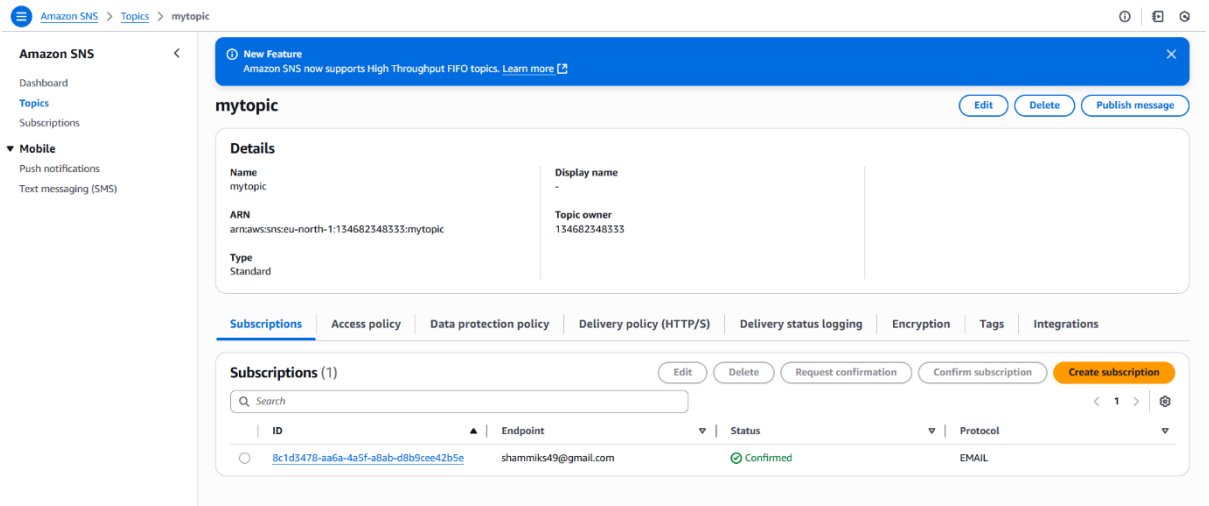
- **AWS EC2**

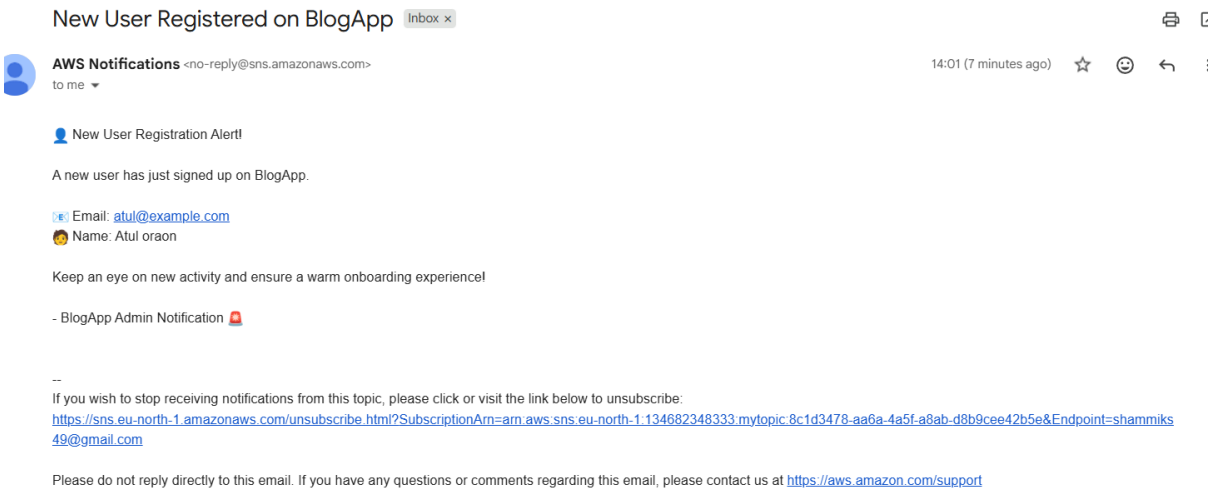



Instances (1/1) Info
Last updated less than a minute ago
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Aws-project
i-013ba5841e707c9f0
Running
t3.micro
3/3 checks passed
View alarms +
eu-north-1a
ec2-16-16-184-127.eu-...
i-013ba5841e707c9f0 (Aws-project)
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Instance summary Info
Instance ID
i-013ba5841e707c9f0
Public IPv4 address
16.16.184.127 | open address
Private IPv4 addresses
172.31.26.207
IPv6 address
–
Instance state
Running
Public DNS
ec2-16-16-184-127.eu-north-1.compute.amazonaws.com | open address
Hostname type
IP name: ip-172-31-26-207.eu-north-1.compute.internal
Private IP DNS name (IPv4 only)
ip-172-31-26-207.eu-north-1.compute.internal


EC2 > Security Groups > sg-0c0ba2649d1e04fde - launch-wizard-19
sg-0c0ba2649d1e04fde - launch-wizard-19
Actions
Details
Security group name
launch-wizard-19
Security group ID
sg-0c0ba2649d1e04fde
Description
launch-wizard-19 created 2025-07-30T13:42:57.830Z
VPC ID
vpc-04a4e1596b19b7c73
Owner
134682348333
Inbound rules count
5 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing - new
VPC associations - new
Tags
Inbound rules (5)
Manage tags
Edit inbound rules
Search
Name
Security group rule ID
IP version
Type
Protocol
Port range
Source
sgr-0e367aa98a10f7b7c IPv4 HTTP TCP 80 0.0.0.0/0
sgr-0699bae4aabb0a892 IPv4 SSH TCP 22 0.0.0.0/0
sgr-0f2b5c14cd547fe38 IPv4 Custom TCP TCP 5000 0.0.0.0/0
sgr-06a98aa3af05be65f IPv4 Custom TCP TCP 3000 0.0.0.0/0
sgr-08dd0cd1cca5aeb68 IPv4 HTTPS TCP 443 0.0.0.0/0

- **AWS SNS**





New User Registered on BlogApp Inbox ×

AWS Notifications <no-reply@sns.amazonaws.com> 14:01 (7 minutes ago)
to me

👤 New User Registration Alert!

A new user has just signed up on BlogApp.

📧 Email: atul@example.com
🧑 Name: Atul oraon

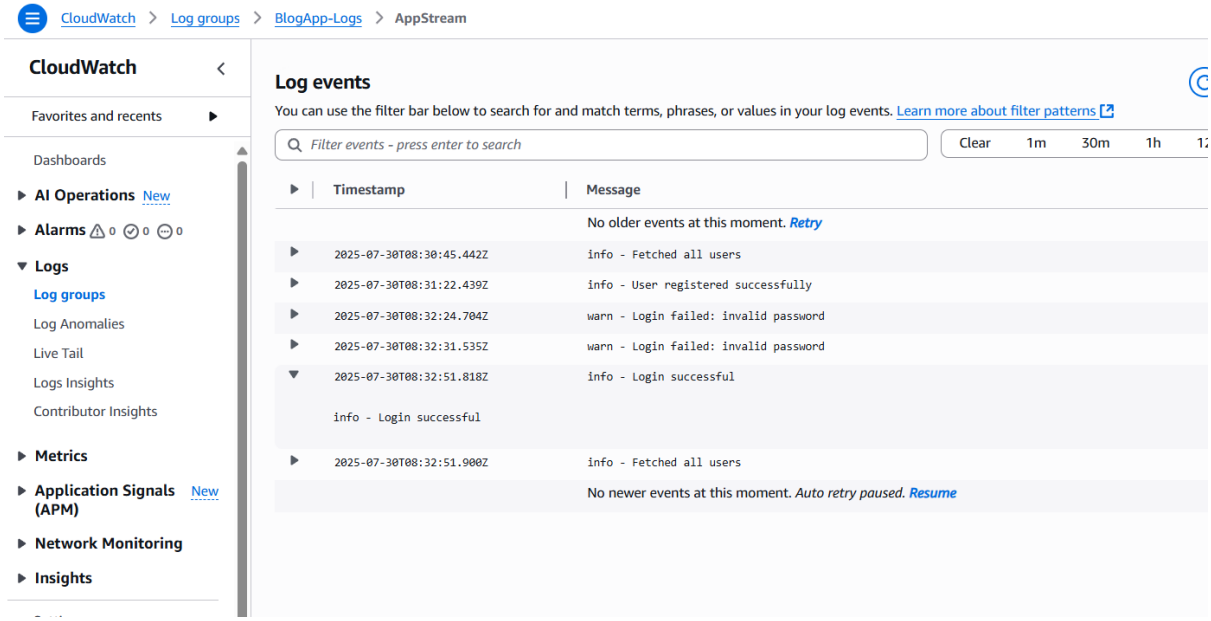
Keep an eye on new activity and ensure a warm onboarding experience!

- BlogApp Admin Notification 🚨

--
If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
https://sns.eu-north-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:eu-north-1:134682348333:mytopic:8c1d3478-aa6a-4a5f-a8ab-d8b9cee42b5e&Endpoint=shammiks49@gmail.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support

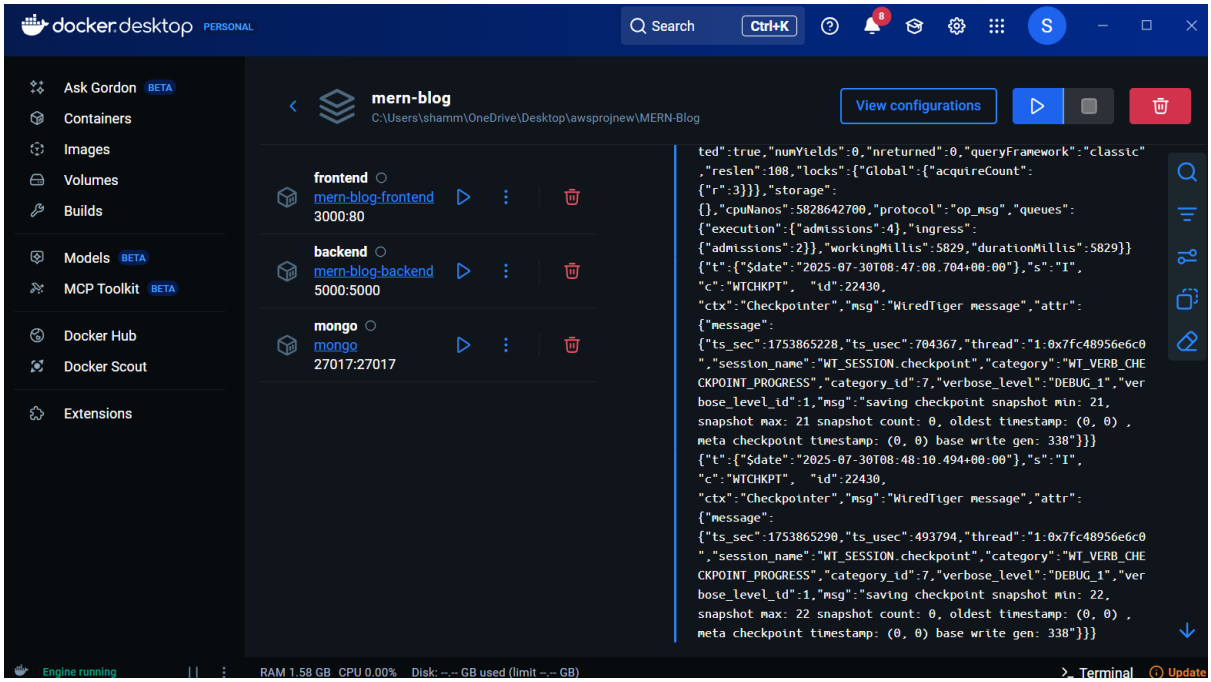
- **AWS Cloudwatch**



4) AWS Linux Commands Used

```
ubuntu@ip-172-31-26-207:~$ history
    1  clear
    2  sudo apt update && sudo apt upgrade -y
    3  sudo apt install docker.io -y
    4  sudo systemctl enable docker
    5  sudo systemctl start docker
    6  sudo usermod -aG docker $USER
    7  newgrp docker
    8  sudo apt install docker-compose -y
    9  git clone https://github.com/shammiks/Aws-project.git
   10  cd Aws-project
   11  cd backend
   12  nano .env
   13  nano .dockerignore
   14  ls
   15  nano .dockerignore
   16  cd ..
   17  cd frontend
   18  nano .dockerignore
   19  cd ..
   20  docker-compose up --build
   21  sudo fallocate -l 2G /swapfile
   22  sudo chmod 600 /swapfile
   23  sudo mkswap /swapfile
   24  sudo swapon /swapfile
   25  free -h
   26  echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
   27  cd Aws-project
   28  docker-compose up --build
   29  df -h
   30  docker container prune -f
   31  docker image prune -a -f
   32  docker volume prune -f
```

5) Docker Desktop Container and Images



6) Code Snippets

```
// Register
export const register = async (req, res) => {
  try {
    const { firstName, lastName, email, password } = req.body;

    if (!firstName || !lastName || !email || !password) {
      logger.warn("Register attempt with missing fields", { email });
      return res.status(400).json({ success: false, message: "All fields are required" });
    }

    const emailRegex = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;
    if (!emailRegex.test(email)) {
      logger.warn("Invalid email format during registration", { email });
      return res.status(400).json({ success: false, message: "Invalid email" });
    }

    if (password.length < 6) {
      logger.warn("Password too short during registration", { email });
      return res.status(400).json({ success: false, message: "Password must be at least 6 characters" });
    }

    const existingUser = await User.findOne({ email });
    if (existingUser) {
      logger.warn("Email already registered", { email });
      return res.status(400).json({ success: false, message: "Email already exists" });
    }

    const hashedPassword = await bcrypt.hash(password, 10);
    const newUser = await User.create({
      firstName,
      lastName,
      email,
      password: hashedPassword
```

```
  });

  // ✅ Send SNS Welcome Message to Admin
  const sns = new SNSClient({ region: process.env.AWS_REGION });
  const snsMessage = `
 New User Registration Alert!

new user has just signed up on BlogApp.

 Email: ${newUser.email}
 Name: ${newUser.firstName} ${newUser.lastName}

eep an eye on new activity and ensure a warm onboarding experience!

BlogApp Admin Notification 🚨
  `;

  await sns.send(new PublishCommand({
    Message: snsMessage,
    TopicArn: process.env.SNS_TOPIC_ARN,
    Subject: "New User Registered on BlogApp"
  }));

  logger.info("User registered successfully", { email });

  return res.status(201).json({
    success: true,
    message: "Account Created Successfully"
  });

} catch (error) {
  logger.error("Registration failed", { error: error.message });
  return res.status(500).json({ success: false, message: "Failed to register" });
}
```

```
export const updateProfile = async (req, res) => {
  try {
    const userId = req.id;
    const {
      firstName, lastName, occupation, bio,
      instagram, facebook, linkedin, github
    } = req.body;

    const file = req.file;
    const user = await User.findById(userId).select("-password");
    if (!user) {
      logger.warn("Profile update failed: user not found", { userId });
      return res.status(404).json({ message: "User not found", success: false });
    }

    if (file) {
      const fileExt = path.extname(file.originalname);
      const uniqueFileName = `avatars/${uuidv4()}${fileExt}`;
      const uploadParams = {
        Bucket: process.env.S3_BUCKET_NAME,
        Key: uniqueFileName,
        Body: file.buffer,
        ContentType: file.mimetype,
        ACL: 'public-read'
      };

      await s3.send(new PutObjectCommand(uploadParams));
      const s3Url = `https://${uploadParams.Bucket}.s3.${process.env.AWS_REGION}.amazonaws.com/${uploadParams.Key}`;
      user.photoUrl = s3Url;
    }
```

7) Conclusion

The blogging website project successfully demonstrates the capabilities of a full-stack MERN (MongoDB, Express.js, React, Node.js) application integrated with modern DevOps practices using Docker and AWS. With essential features such as user authentication, post creation, media uploads, and responsive UI, the platform offers a robust and user-friendly environment for writers and readers alike. The use of AWS services like EC2 and S3 ensures scalability, while Docker ensures consistency across development and production. Overall, the project combines clean architecture, modular code structure, and cloud deployment to create a production-ready blogging system.

---

8) Future Enhancements

To make the platform even more powerful and feature-rich, several enhancements can be considered. Implementing a **commenting system** with nested replies would allow deeper engagement between users. An **admin dashboard** for moderating posts and managing users can improve content quality and control. Integrating **real-time features** like notifications via WebSockets or AWS SNS can increase user activity. Adding **SEO optimization** and server-side rendering can help improve discoverability and performance. Lastly, enabling **social logins** (Google, GitHub) and adding support for **progressive web apps (PWA)** or **native mobile apps** would broaden accessibility and enhance user experience.

Project Submitted by: Shammi Kumar

B.Tech CSE, BPIT, GGSIPU

GitHub : https://github.com/shammiks/Aws-project

Submitted to: Saurabh Dwivedi