VPC + EC2 + Node.js + CloudWatch Alarm using AWS CloudFormation

 Abhishek Bhagat

 Abhishek Bhagat

💡 Introducing Today's Project!

What is AWS CloudFormation?

AWS CloudFormation is an Infrastructure as Code (IaC) service that lets you model and provision AWS resources in a safe, repeatable manner. You write templates in YAML or JSON that define infrastructure components like VPCs, EC2 instances, security groups, and more.

🛠️ How I Used AWS Services in This Project

- **AWS CloudFormation**: For provisioning all infrastructure as code, including the VPC, Subnet, EC2, Security Group, and CloudWatch Alarm.
- **Amazon EC2**: To host a simple Node.js application.
- **CloudWatch**: To monitor EC2 instance metrics and raise an alarm when CPU utilization exceeds 80%.
- **IAM**: To ensure EC2 and CloudFormation have appropriate permissions.
- **KeyPair**: Created manually via the AWS Console for SSH access to EC2.
- **SSM Parameter Store**: Used for dynamic AMI resolution.

😮 One Thing I Didn't Expect...

Even though the app was listening on port 3000 internally, external traffic wasn't reaching it. It required either updating the app to listen on port 80 or using iptables to forward port 80 to 3000. Also, ensuring the app starts as the ec2-user with proper ownership and privileges was crucial.

⏱️ This Project Took Me...

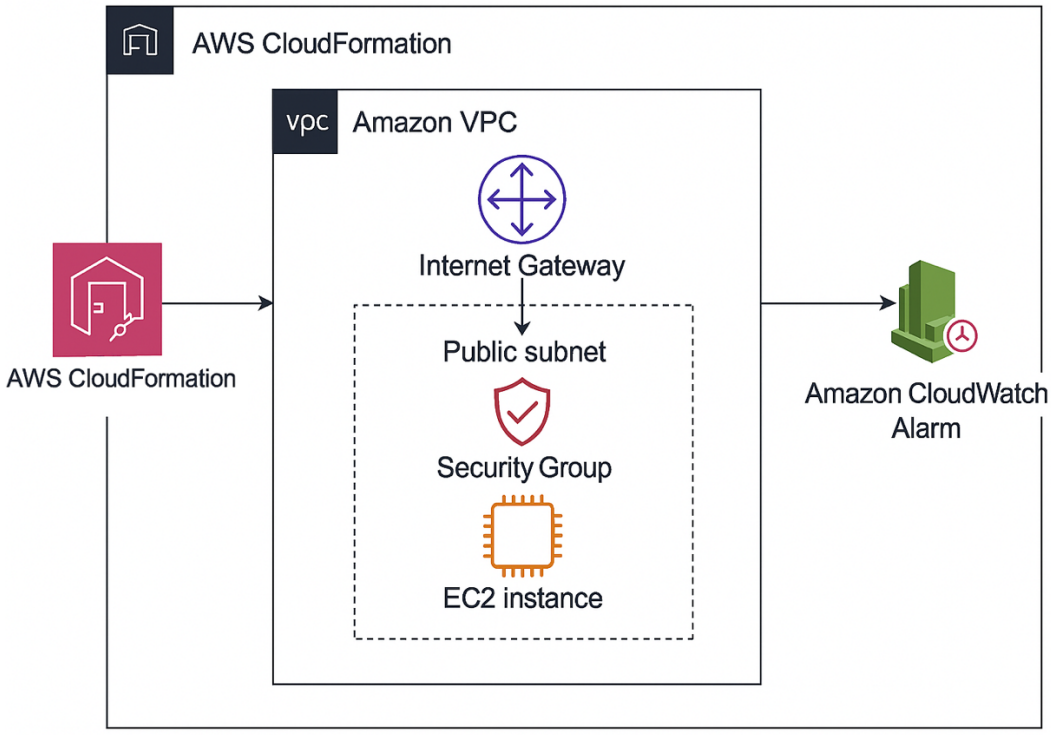
~1 hour to plan, code, deploy, troubleshoot, and verify.

Abhishek Bhagat

📦 Project Steps Breakdown

1. Designed Architecture

- A VPC with a public subnet
- Internet Gateway for outbound internet access
- Route table setup for public subnet routing
- Security Group allowing SSH and HTTP traffic



2. Wrote CloudFormation Template

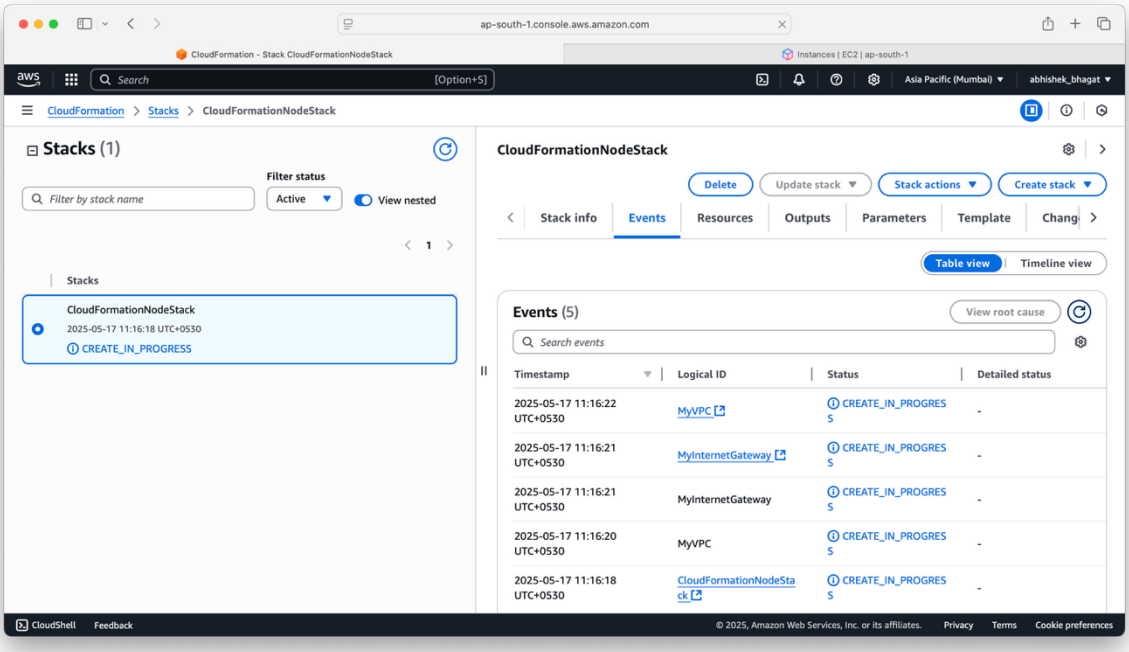
- YAML-based template defining all resources
- Defined EC2 instance with UserData to install Node.js, write app.js, and start the server using pm2
- Used iptables to redirect traffic from port 80 to 3000

3. Created EC2 Key Pair

- Created via AWS Console and used to SSH into instance
  - Go to **EC2 > Key Pairs > Create key pair**
  - Name it something like my-keypair
  - Save the .pem file to your machine

4. Launched Stack

- Used AWS CloudFormation Console to launch the stack
- Verified the EC2 instance, public IP, and security group settings



Abhishek Bhagat

❗ Issues Faced

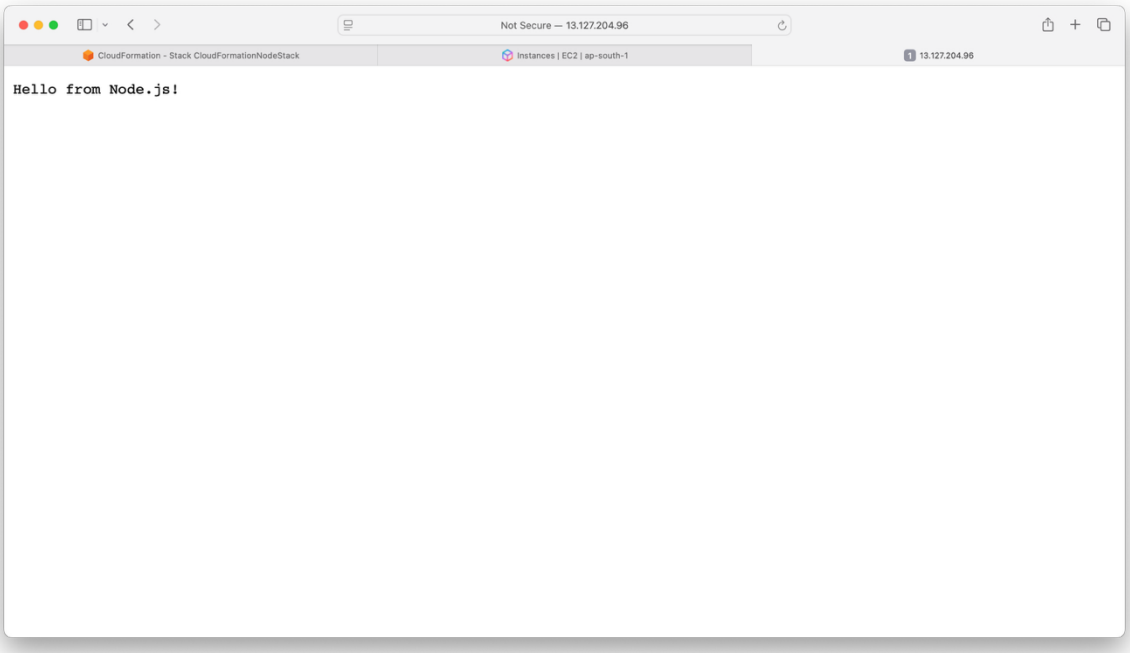
Issue	*Resolution*
Node app not responding on port 80	Used iptables to redirect port 80 to 3000
pm2 not recognized	Ensured Node and pm2 were properly installed under ec2-user
CPU Alarm not triggering	Installed and ran stress tool to generate CPU load

✅ Final Outcome

Successfully deployed a fully functional and monitored EC2 Node.js application — all provisioned via a **single CloudFormation template**. This project demonstrated skills in:

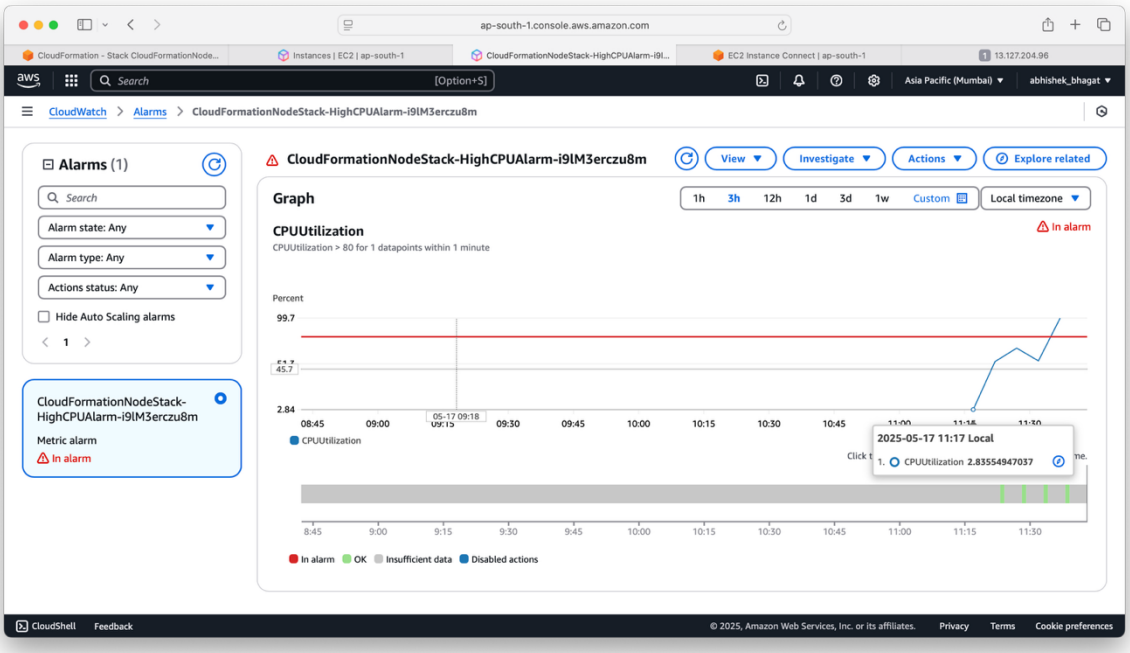
- Infrastructure as Code (IaC)
- Cloud Monitoring (CloudWatch)
- EC2 provisioning and access control
- Automating application setup with UserData

Node.js App Deployment:



Abhishek Bhagat

CloudWatch Alarm After Stress:



📚 What I Learned & Strengthened

- Improved confidence writing complete CloudFormation stacks from scratch.
- Understood practical aspects of UserData scripting and debugging.
- Gained hands-on experience with CloudWatch alarms and stress testing.
- Strengthened my understanding of VPC networking, public subnets, and route tables.