AWS EC2 + Node.js Application Deployment + CloudWatch Monitoring

---

What is AWS?

AWS (Amazon Web Services) is a comprehensive cloud computing platform provided by Amazon. It offers a wide range of on-demand services like computing power, storage, databases, networking, machine learning, and more, over the internet.

Key points in brief:

- **Cloud Platform:** No need for physical servers; resources are virtual.
- **Pay-as-you-go:** You only pay for what you use.
- **Scalable & Flexible:** Easily scale up or down based on demand.
- **Global Infrastructure:** Data centres across the world for reliability.
- **Popular Services:**
    - **EC2** – Virtual servers
    - **S3** – Object storage
    - **RDS** – Managed databases
    - **Lambda** – Serverless computing

In essence, AWS allows individuals and businesses to run applications and store data without maintaining their own physical hardware.

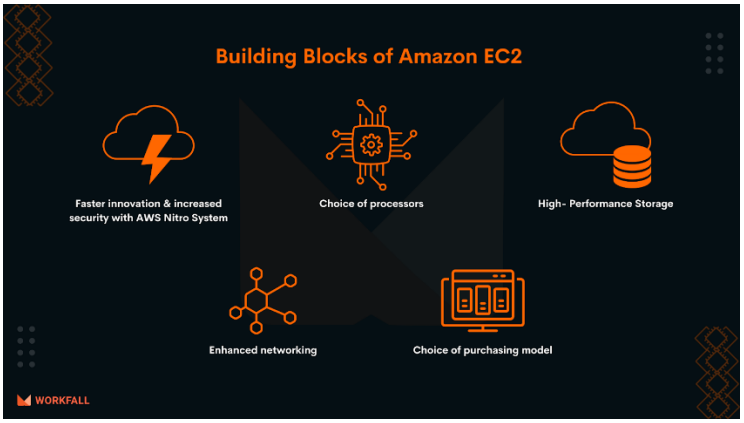
What is EC2?

Amazon EC2 (Elastic Compute Cloud) is a core service of AWS that provides resizable virtual servers in the cloud. It lets you run applications without buying or managing physical hardware.

Key points in brief:

- **Virtual Servers:** You can launch virtual machines called **instances**.
- **Flexible:** Choose CPU, memory, storage, and OS based on your needs.
- **Scalable:** Easily scale up or down depending on traffic or workload.
- **Pay-as-you-go:** Only pay for the compute time you use.
- **Use Cases:** Hosting websites, running applications, batch processing, and more.

In short, **EC2 is like renting a computer in the cloud that you can configure and use anytime.**



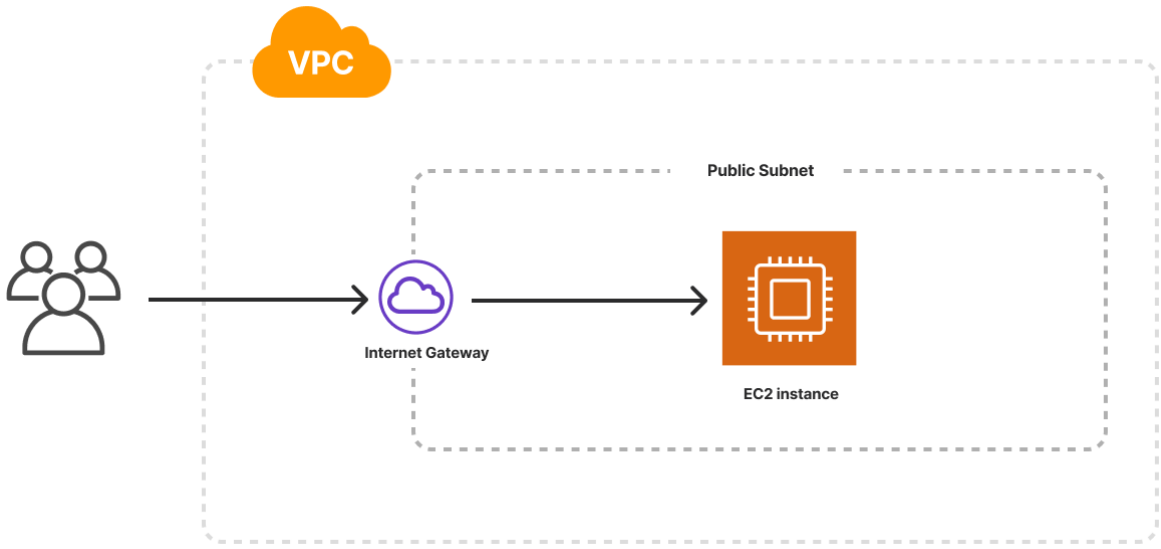
What is VM?

In AWS, a **VM (Virtual Machine)** is essentially a **software-based computer** that runs on physical hardware in the cloud, allowing you to run applications just like a real computer. In AWS, VMs are called **EC2 instances**.

Key points in brief:

- **Virtualized Server:** It behaves like a full computer with CPU, memory, storage, and network.
- **Runs in the Cloud:** Hosted in AWS data centers, no need for physical hardware.
- **Customizable:** You can choose operating system, size, and resources.
- **Scalable & Flexible:** Start, stop, or resize your VM anytime.
- **Pay-as-you-go:** Only pay for the time your VM is running.

In short, a **VM in AWS is your personal computer in the cloud** that you can use for running apps, websites, databases, or development work without owning physical servers.



What is CloudWatch?

Amazon CloudWatch is a **monitoring and observability service** in AWS that helps you track the performance and health of your cloud resources and applications.

Key points in brief:

- **Monitoring:** Collects metrics like CPU usage, memory, disk I/O, and network activity from AWS resources (EC2, RDS, Lambda, etc.).
- **Logs Management:** Can collect and store logs from applications, servers, and services.
- **Alarms & Notifications:** You can set thresholds to get alerts via SNS if something goes wrong.
- **Visualization:** Provides dashboards to visualize metrics and logs in real-time.
- **Automation:** Can trigger actions automatically (like scaling EC2 instances) based on metrics.

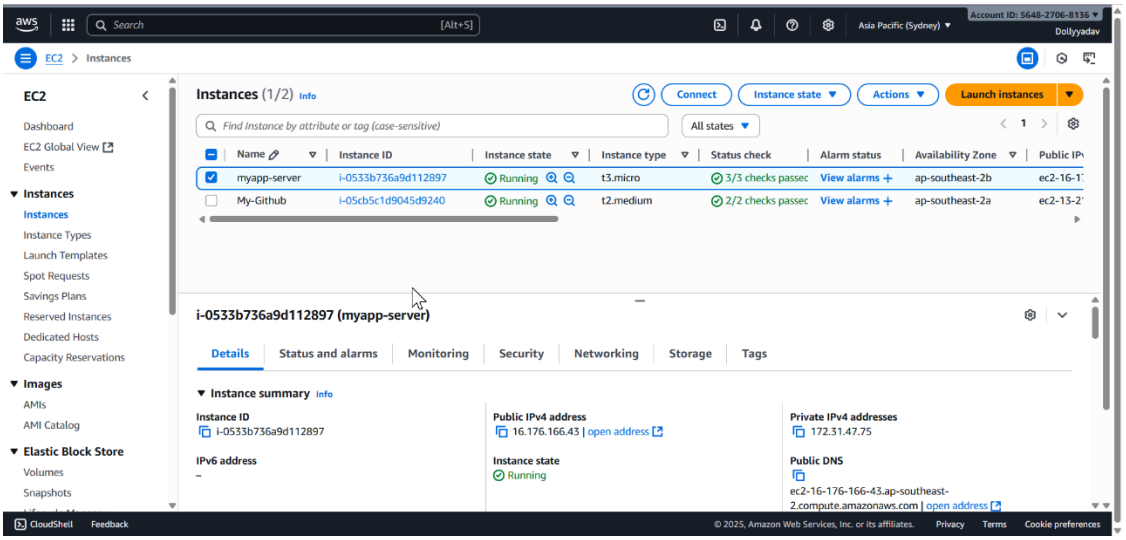
In short: CloudWatch is like a **health monitor for your AWS resources and applications**, helping you detect issues, analyze performance, and automate responses.

Before starting, you need to understand EC2, virtual machines, Node.js deployment, and CloudWatch, as these are essential for setting up an EC2 instance, deploying a Node.js application, configuring the CloudWatch agent to monitor the VM and application, and verifying CPU, memory, disk metrics, as well as application logs.

Step 1: Launch EC2 Instance

1. **Login to AWS Console** → EC2 → Launch Instances.
2. Choose **Ubuntu 22.04 LTS** AMI.
3. Select **t2.micro** (free tier).
4. Configure instance details (default settings).
5. Add storage (default 8GB).
6. Add **Security Group**:
7. Port 22 → SSH → Source 0.0.0.0/0
8. Port 80 → HTTP → Source 0.0.0.0/0
9. Port 443 → HTTPS → Source 0.0.0.0/0
10. Launch and **download key pair**.

Now your VM is created successfully.



Step 2: Connect to the VM

1. Go to **EC2 → Instances → Select your instance**.
2. Click **Connect** (top right).
3. Choose **EC2 Instance Connect (browser-based SSH)**.
4. Click **Connect** → A terminal opens in your browser.

Now you are inside the VM without using your laptop terminal. Here you can use the CLI interface to enter the commands.

Step 3: Install Application (Simple Node.js App)

Inside the EC2 terminal do the following steps

1. Update system:
   - ➔ sudo apt update && sudo apt upgrade -y
   - ➔ This command updates package information and upgrades all installed packages automatically.

2. Install Node.js:
   - ➔ curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash –
   - ➔ **-f :** Fail silently on server errors (don't output HTML error pages).
   - ➔ **-s:** Silent mode — hides progress and error messages.
   - ➔ **-S:** Shows errors even when in silent mode.
   - ➔ **-L:** Follows redirects if the URL points to a different location.
   - ➔ **-E:** Preserves the current user's environment variables while running the command as root.
   - ➔ This command downloads and runs the Node.js 20.x setup script with root privileges.
   - ➔ sudo apt install -y nodejs git
   - ➔ Installs Node.js and Git automatically without prompts.

3. Create app folder:
   ➔ mkdir myapp && cd myapp
   ➔ This command **creates a folder named myapp and navigates into it**.

4. Create app file:
   ➔ vim app.js
   ➔ This command **creates the file app.js in the Vim editor** for editing.

5. Paste this code inside editor:

```
const http = require('http');
const server = http.createServer((req, res) => {
res.writeHead(200, {'Content-Type':'text/plain'});
res.end("Hello from AWS VM!\n");
});
server.listen(80, () => console.log("App running on port 80"));
```

6. Run app:
   sudo node app.js
7. Open browser
   http://<EC2-Public-IP>
   You should see: **Hello from AWS VM!** (As per my application you will see this)

Step 4: Keep App Running Forever

1. Stop app (CTRL+C).
2. Install PM2 (process manager):
   sudo npm install -g pm2
   ➔ **PM2** is a process manager for Node.js applications
   ➔ -g installs it **globally** so it can be used anywhere on the system.

3. Start app with PM2:
   pm2 start app.js
   ➔ Launches your Node.js application as a background process.
   ➔ You don't need to keep the terminal open for the app to run.

   pm2 startup system
   ➔ Configures PM2 to **auto-start your app on system boot** using systemd.
   pm2 save
   ➔ Saves running processes so PM2 can auto-restore them on reboot.

   In short: We use PM2 to run the Node.js app in the background, keep it running after crashes, and ensure it starts automatically on server reboot.

Step 5: Enable Monitoring (VM & App)

A. CloudWatch Agent Installation

- CloudWatch by default gets EC2 CPU metrics, but not memory/disk — the CloudWatch Agent collects those extra system metrics and can forward **log files** from your VM to CloudWatch Logs.
- We install the agent on the VM and give that VM an IAM role (with `CloudWatchAgentServerPolicy`) so the agent is authorized to push metrics & logs to your AWS account and also add (AmazonSSMManagedInstanceCore).

Create an IAM Role for CloudWatch Agent

1. From the search bar at the top, type IAM → click IAM.
2. click Roles.
3. Click the orange Create role button.
4. Trusted entity type → select AWS service.
5. Use case → choose EC2 → click Next.

On the **Permissions policies** page:

6. In the search bar, type: CloudWatchAgentServerPolicy.
7. Tick the checkbox next to it
8. Click Next.
9. Enter a role name (I have named it EC2CloudWatchRole).
10. Click Create role.
11. Now IAM is created.

Attach the IAM Role to Your EC2 Instance

1. In the AWS Console, go to EC2.
2. In the left menu, click Instances.
3. Select the checkbox next to your running instance.
4. On the top menu, click Actions → Security → Modify IAM role.
5. In the dropdown, select the role you just created (EC2CloudWatchRole).
6. Click Update IAM role.

Now your EC2 instance has permission to send metrics & logs to CloudWatch.

On the EC2 VM — install & configure CloudWatch Agent

(Do these in the EC2 terminal / EC2 Instance Connect.)

A. Update and install agent package
sudo apt update -y
sudo apt upgrade -y

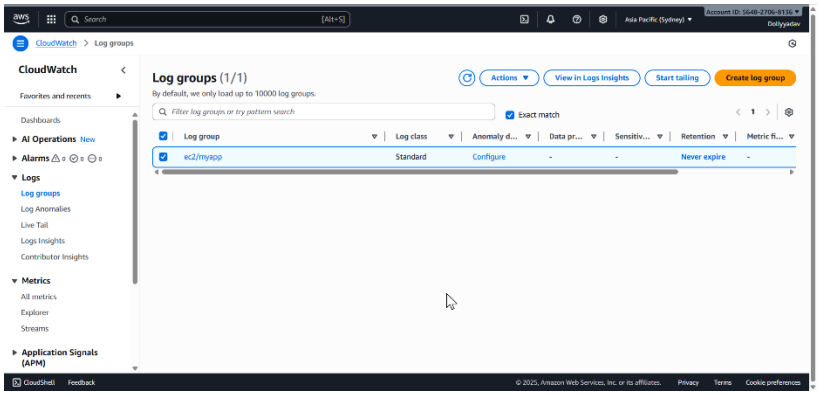
wget https://s3.amazonaws.com/amazoncloudwatch-agent/ubuntu/amd64/latest/amazon-cloudwatch-agent.deb
sudo dpkg -i ./amazon-cloudwatch-agent.deb || sudo apt -f install -y

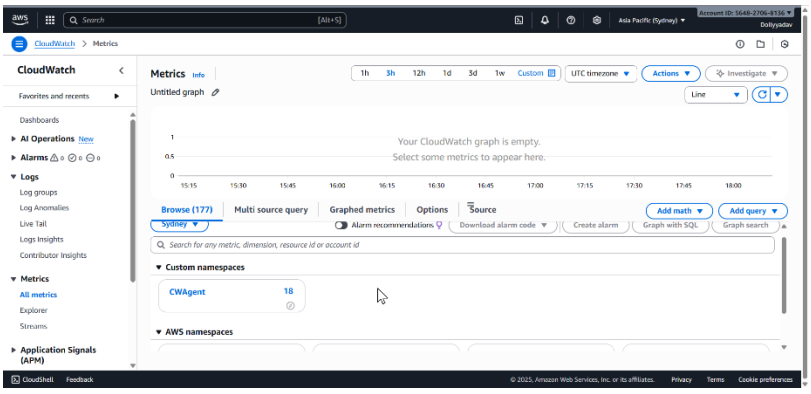
B. Make sure your app writes to a file (so agent can collect logs)
# create log file and set permissions (example path /var/log/myapp.log)
sudo touch /var/log/myapp.log
sudo chown ubuntu:ubuntu /var/log/myapp.log
sudo chmod 664 /var/log/myapp.log

run your app redirecting stdout/stderr to that file (or use PM2/systemd)
cd ~/myapp
sudo nohup node app.js >> /var/log/myapp.log 2>&1 &

C. Run the interactive config wizard
sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-config-wizard
Why: wizard builds the JSON config telling the agent what metrics and log files to push.

D. Start and enable the agent with the saved config
- loads the JSON and runs the agent; enable makes it start on reboot.

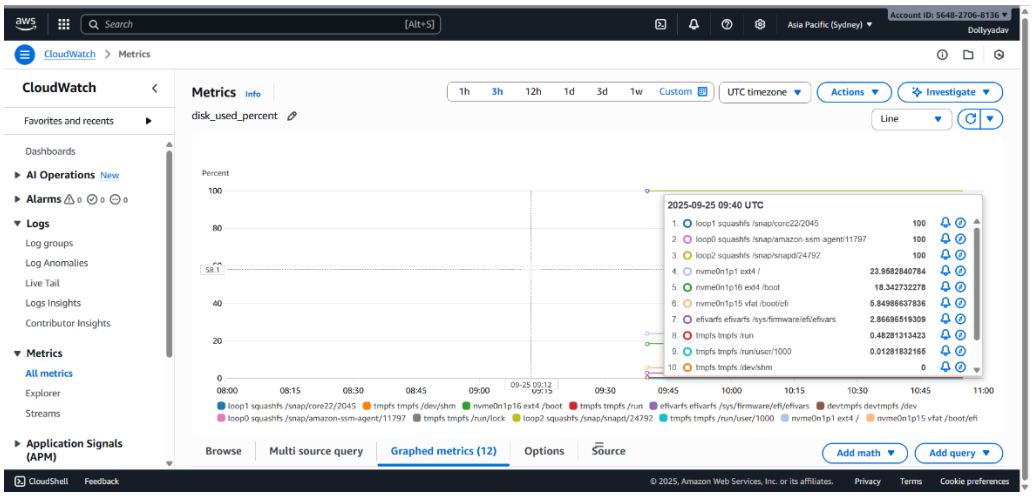


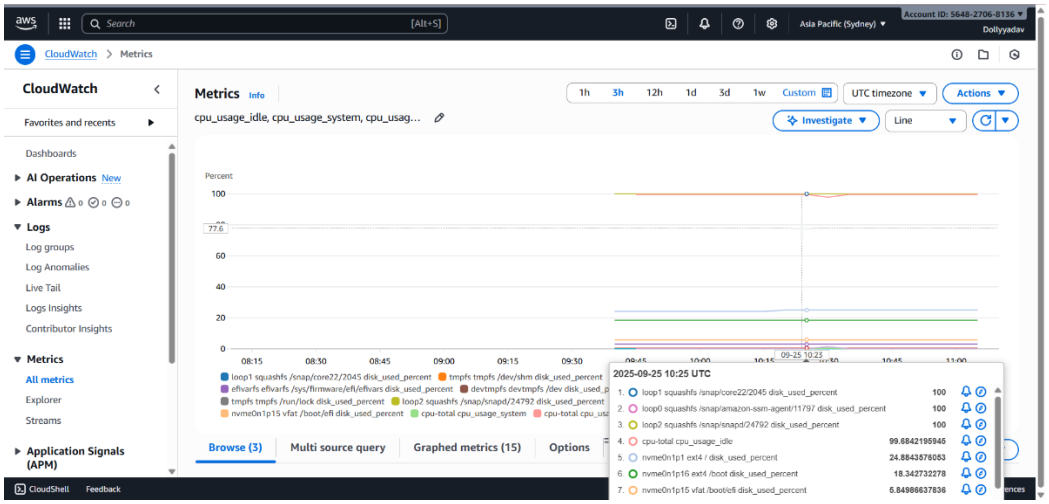


Step 6: View Metrics & Logs

1. In AWS Console → Search **CloudWatch**.
2. Go to **Metrics → All metrics → CWAgent**.
3. You'll see CPU, Memory, Disk usage.
4. Go to **Logs → Log groups**.
5. Check your myapp logs streaming from the VM

Metric Monitoring:





Log Monitoring: