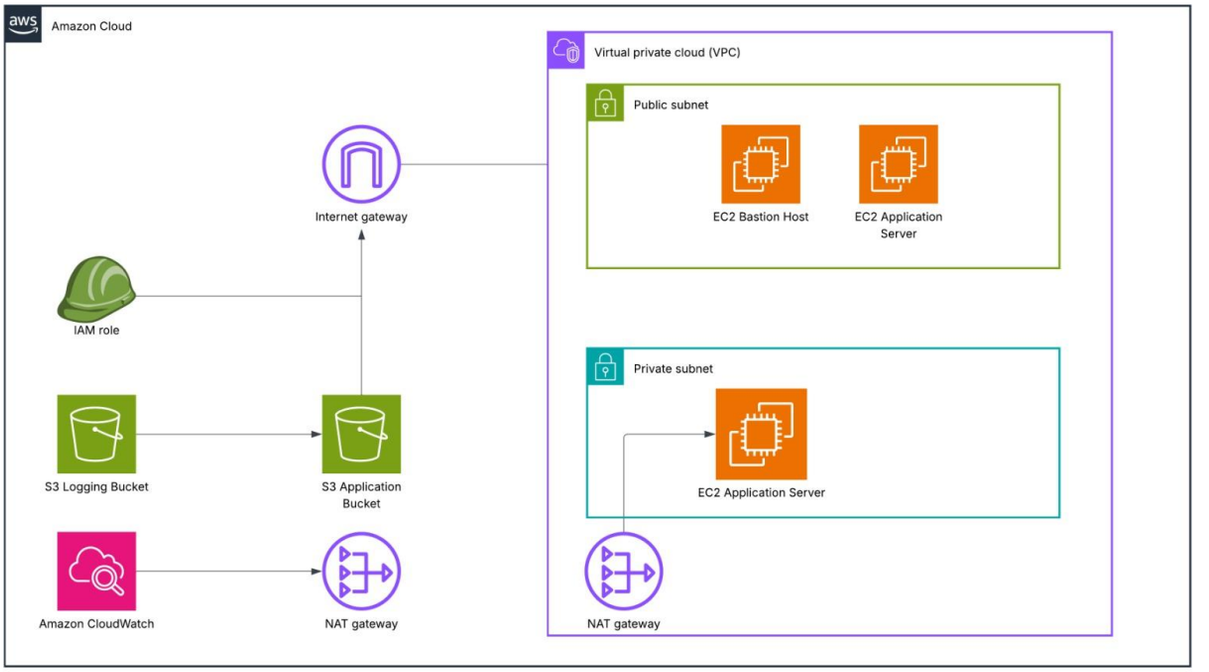
Secure EC2 Hosting in a Custom VPC with IAM Roles, S3 Logging & CloudWatch Monitoring

Project Purpose:
Design and deploy a highly secure, production-ready EC2 environment inside a custom VPC with strict IAM access, S3-based logging, and CloudWatch monitoring — all implemented using the AWS Management Console.



STEP 1 — Create the VPC

1. VPC Console → Create VPC
2. Settings:
    - VPC CIDR: 10.0.0.0/16
    - Name: secure-app-vpc
3. Create.

Public Subnet:

- Name: public-subnet-1
- CIDR: 10.0.1.0/24
- Auto-assign Public IP: Enabled

Private Subnet:

- Name: private-subnet-1
- CIDR: 10.0.2.0/24
- Auto-assign Public IP: Disabled

Create Internet Gateway

- Name: secure-app-igw
- Attach to VPC.

Public Route Table:

- Name: public-rt
- Route: 0.0.0.0/0 → IGW
- Associate with public-subnet-1

Private Route Table:

- Name: private-rt
- Route will be added after NAT Gateway
- Associate with private-subnet-1

STEP 2 — Create NAT Gateway

1. VPC → NAT Gateway
2. Create NAT in Public Subnet
3. Allocate Elastic IP
4. Update private-rt route table:
    - 0.0.0.0/0 → NAT Gateway

STEP 3 — Deploy EC2 Instance [Bastion Host] (public subnet)

Purpose:
You will SSH into your private EC2 through this bastion instead of exposing it publicly.

1. EC2 → Launch Instance
2. Name: bastion-host
3. AMI: Amazon Linux 2
4. Type: t2.micro
5. Subnet: public-subnet-1
6. Auto-assign Public IP: Enable
7. Security Group:
    - Name: sg-bastion
    - Inbound: SSH(22) → My IP only
8. Launch instance

STEP 4 — Deploy Private Application EC2 Server (private subnet)

1. EC2 → Launch instance
2. Name: app-server
3. AMI: Amazon Linux 2
4. Type: t2.micro
5. Subnet: private-subnet-1
6. Auto-assign Public IP: Disable
7. Security Group:
    - Name: sg-app
    - Inbound: SSH(22) → Source: sg-bastion
    - Outbound: Allow All
8. No key pair needed
9. Launch

STEP 5 — IAM Security

Purpose:
This allows the private EC2 instance to upload logs to **S3** and send metrics to **CloudWatch** without access.

1. IAM → Roles → Create Role
2. Select Trusted Entity: EC2
3. Permissions:
    - AmazonS3FullAccess
    - CloudWatchAgentServerPolicy
4. Name: EC2-logging-role
5. Create role
6. Attach role to app-server

STEP 6 — Connect EC2 instance via SSH

- To Send Logs to S3:
  sudo yum update -y
  sudo yum install awscli -y
- To Upload system logs:
  aws s3 cp /var/log/messages s3://app-logs-secure/system/
- To Install CloudWatch Agent on Private EC2:
  sudo yum install amazon-cloudwatch-agent -y
  sudo touch /opt/aws/amazon-cloudwatch-agent/bin/config.json
- To Use default config wizard:
  sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-config-wizard
- To Start agent:
  sudo systemctl start amazon-cloudwatch-agent

STEP 7 — Logging

Creating S3 Logging Bucket

1. S3 → Create bucket
2. Name: app-logs-secure
3. Block Public Access: Enabled
4. Enable Versioning
5. Enable Server-Side Encryption (SSE-S3)

STEP 8 — Monitoring

Cloudwatch Monitoring Setup

Enable Metrics

- CPU Utilization
- Disk Read/Write
- Network In/Out
- Memory (via CloudWatch Agent)

Create CloudWatch Alarms

1. CloudWatch → Alarms → Create alarm
2. Examples:
    - CPUUtilization > 80% for 5 minutes
    - StatusCheckFailed > 0
    - LowDiskSpace Alarm

Create a Dashboard

1. CloudWatch → Dashboards → Create
2. Add widgets:
    - CPU graph
    - Disk usage
    - Network
    - Log streams

STEP 9 — Security Hardening Checks

- Disable SSH on application server from anywhere except bastion
- Ensure no public IP on private EC2
- Enable MFA on IAM account
- Follow least-privilege IAM practices
- S3 bucket public access = Blocked
- Use CloudWatch alarms to detect anomalies