**Automate EC2 Deployment on AWS using Terraform & Jenkins | DevOps CI/CD Pipeline**

**Description**

We’ll automate **AWS EC2** deployment using **Terraform** and **Jenkins**. By the end of this session, you’ll learn how to set up a **Jenkins** **pipeline** that provisions an EC2 instance using Terraform. Let’s get started!

**Topic Covered:**

1. AWS IAM User creation and attaching policy.
2. Writing Terraform code to deploy an EC2 instance.
3. Configuring a Jenkins pipeline to automate Terraform.
4. Fixing **AWSCompromisedKeyQuarantineV3** policy issues.

**Prerequisites**

1. Terraform Installed on System.
2. Jenkins Installed on System.
3. GitHub Repository for Terraform Code

**Step 1: Set Up GitHub Repository**

1. Create a new GitHub repository (e.g., terraform-ec2-deployment)
2. Copy the GitHub repository URL for later use in Jenkins

**Step 2: Create IAM User for Terraform**

1. Log in to AWS Console → Go to IAM
2. Click "Users" → "Add User"
3. Enter User Name (e.g., terraform-user)
4. Select Access Key – Programmatic access → Next
5. Attach Policies:
   1. AmazonEC2FullAccess
   2. IAMFullAccess (if needed)
   3. AdministratorAccess (optional)
6. Click "Create User" → Download Access Key & Secret Key

Terraform can now deploy AWS resourcesusing this IAM user.

**Step 3: Create Terraform Configuration for EC2**

1. Open a code editor and create a file main.tf
2. Add the following Terraform code:

provider "aws" {

  region     = "eu-west-2"

  access\_key = "AKIAWIJIUVESLBBDSJGQ"

  secret\_key = "fGahxiy9MJwpAY87PbpIDTnG0it/GC1Qq2qrTfWh"

}

resource "aws\_instance" "ServerA" {

  ami           = "ami-0fd24edaae77b2388"

  instance\_type = "t2.micro"

  tags = {

    Name = "Terraform-EC2"

  }

}

1. Commit and push this file to **GitHub Repo.**

**Step 4: Create a Jenkins Pipeline Job**

1. Go to **Jenkins Dashboard → New Item**
2. Enter **Job Name** → Select **Pipeline** → Click **OK**
3. Scroll to the **Pipeline** section → Select **Pipeline Script**
4. Paste the following pipeline script:

pipeline {

agent any

stages {

stage('Checkout') {

steps {

git branch: 'main', url: ' https://github.com/rajandubey/DevOps\_With\_AWS.git'

}

}

stage('Terraform Init') {

steps {

bat 'terraform init'

}

}

stage('Terraform Plan') {

steps {

bat 'terraform plan'

}

}

stage('Terraform Apply') {

steps {

bat 'terraform apply -auto-approve'

}

}

}

}

1. Click **Save**

**Step 5: Run the Jenkins Pipeline**

1. Click Build Now
2. Monitor the logs for any errors
3. Once successful, check the AWS Console

**Step 6: Verify EC2 Instance in AWS**

1. Go to AWS Console → EC2 Dashboard
2. Click on Instances
3. Find the instance named Jenkins-Terraform-EC2

🎉 **Congratulations! You’ve successfully deployed an EC2 instance using Terraform and Jenkins.**