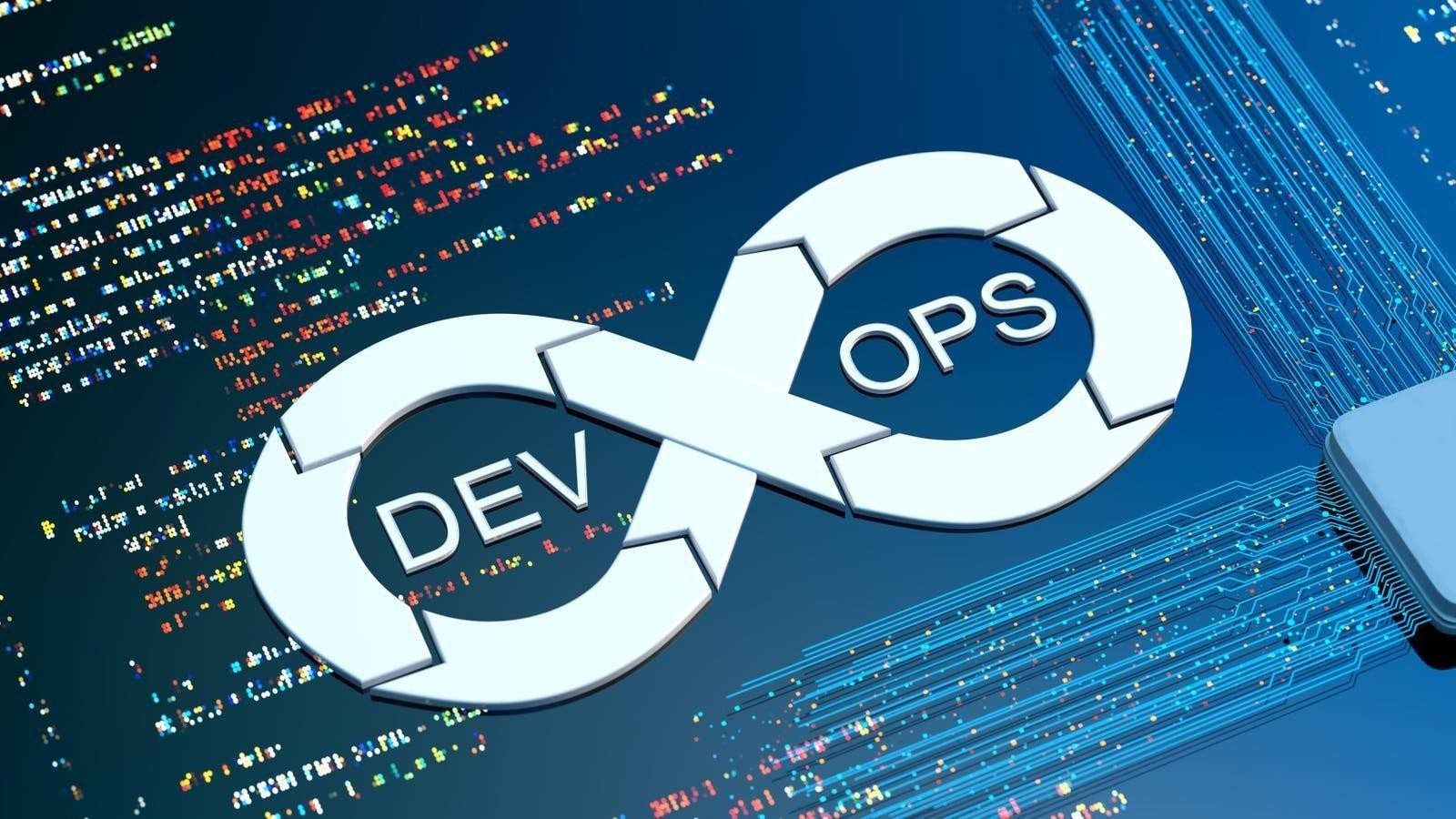
DEVOPS PROJECT REPORT



**Project Title: Configure the Jenkins platform to send personal email alerts and develop a**

**Git-based infrastructure management with Terraform Project Members:**

|  |  |
| --- | --- |
| **Batch No** | 7 |
| 2200032598 | D.AMRUTHA |
| 2200080014 | MORLA PALLAVI |
| 2200030362 | HARIKA |
| 2200031272 | VISHNU VARDHAN |

**Instructor Name:** Dr. Prasanth Yalla

**Course Coordinator:** Ch. Lavanya Susanna

**Course Title:** Cloud DevOps

**Course Code:** 22SDCI05R

# Table of Contents

|  |  |  |
| --- | --- | --- |
| **Sno** | **Content** | **Page No** |
| 1. | Introduction | 3 |
| 2. | Project Overview | 4 |
| 3. | Key Technology and Tools used in Project | 5 |
| 4. | Setting Up the Environment | 6 |
| 5. | Workflow Overview | 8 |
| 6. | Conclusion | 9 |
| 7. | References | 12 |

**1.INTRODUCTION:**

This project explores two essential DevOps practices: configuring Jenkins for email alert notifications through pipelines and using Terraform to provision cloud infrastructure via Git-based source code versioning. These activities form a foundation for continuous integration (CI) and Infrastructure as Code (IaC).

1. **Project Overview:**

The project is divided into two independent tasks:

Task 1: Configure Jenkins to send personal email alerts upon pipeline execution.

Task 2: Create a Git-based Terraform project in VS Code to provision an AWS EC2 instance. These tasks were implemented separately but align with DevOps automation and cloud provisioning principles.

1. **Key Technology and Tools used in Project:**

* 1. Jenkins - Automation server used for creating and executing pipelines.
  2. Email Extension Plugin - Sends email notifications for pipeline results.
  3. Terraform - Infrastructure as Code tool to provision EC2 instances.
  4. AWS EC2 - Virtual server hosted in Amazon Web Services.
  5. Git - Version control system used for managing Terraform scripts.
  6. VS Code - Code editor used for developing Terraform configurations.

1. **Software and Hardware Requirements**

**Task 1**: Jenkins Email Notification Setup

* 1. Launch Jenkins in browser at http://localhost:8080.
  2. Install Email Extension Plugin from Jenkins Plugin Manager.
  3. Go to Manage Jenkins > Configure System and add email settings:
     + SMTP Server: smtp.gmail.com o Port: 587 o Use TLS: Checked
     + Authentication: Enabled with App Passwords
  4. Create a new pipeline job with the following script: pipeline { agent any stages { stage('Notify') { steps {

echo 'Sending notification email...'

}

} } post { success {

mail to: 'amruthadarbha01@gmail.com', subject: "Build Successful",

body: "The Jenkins pipeline executed successfully."

}

failure {

mail to: 'amruthadarbha01@gmail.com', subject: "Build Failed",

body: "The Jenkins pipeline has failed."

}

}

}

**Task 2**: Terraform AWS EC2 Setup using Git and VS Code

1. Create a new project folder and initialize Git repository.
2. Write main.tf with the following Terraform configuration:
3. provider "aws" {

region = "us-east-1"

}

resource "aws\_instance" "example" { ami = "ami-

0c55b159cbfafe1f0"

instance\_type = "t2.micro"

tags = {

} Name = "Terraform-EC2"

}

1. Run the following commands in terminal:

terraform initterraform apply -auto-approve

1. EC2 instance is launched and visible on AWS Console.

**5 .Workflow Overview**

**Task 1:** Jenkins Email Notification Workflow

Open Jenkins in browser

↓

Install Email Extension Plugin

↓

Configure SMTP settings with Gmail

↓

Create a Pipeline Job

↓

Run Pipeline → Email Sent to User

(Success/Failure)

**Task 2**: Terraform Infrastructure Workflow

Open VS Code and write Terraform script

↓

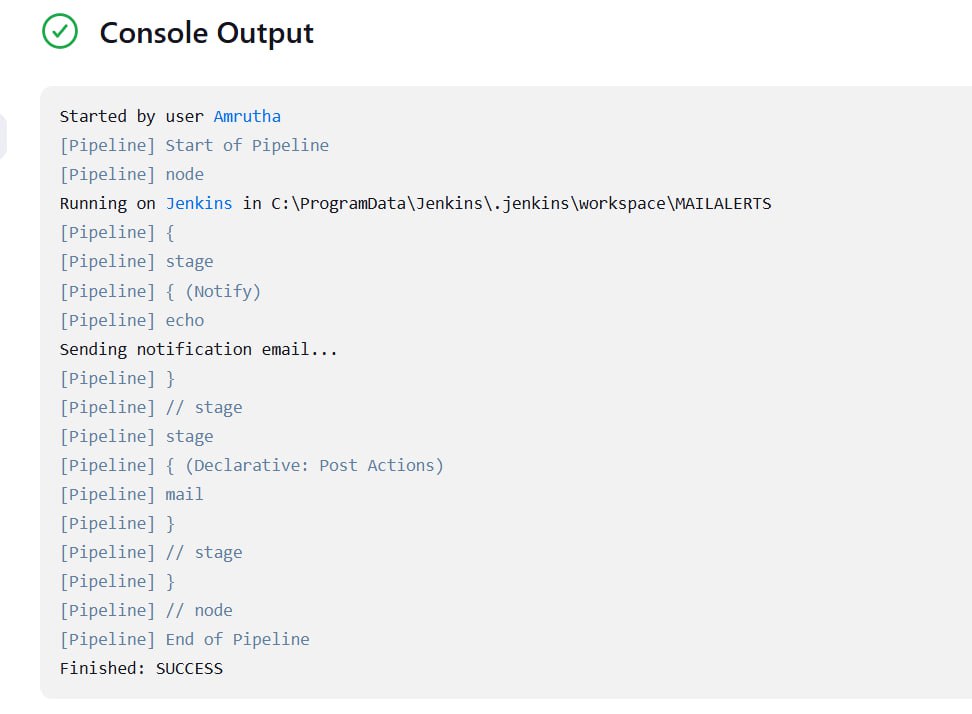
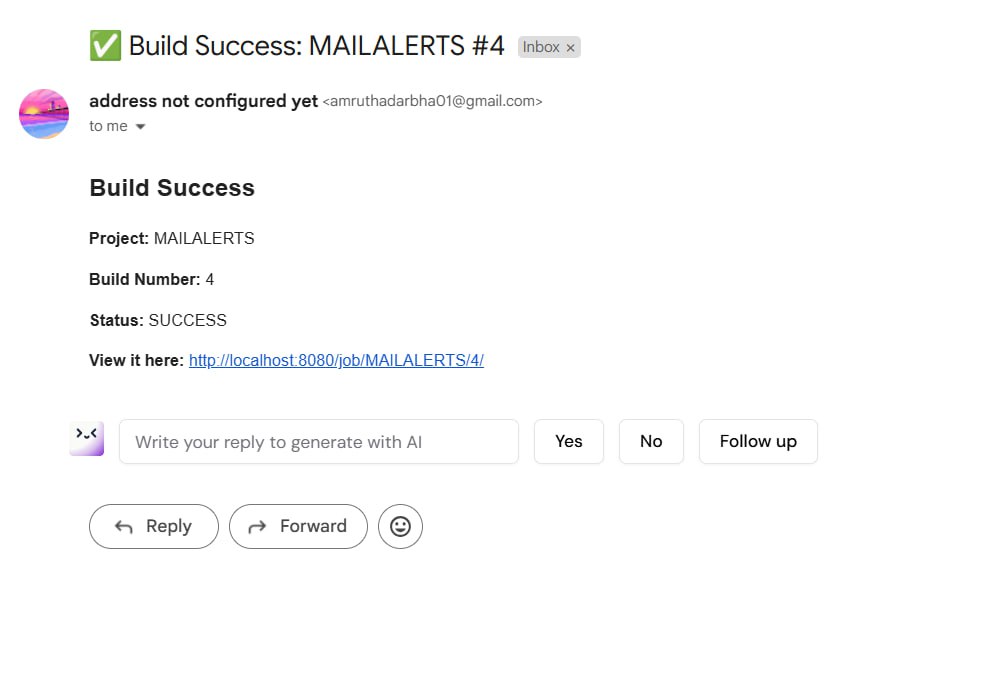
Initialize and Apply Terraform via Terminal

↓

AWS EC2 instance is provisioned

**6. Conclusion**

This project demonstrates two foundational DevOps practices. Jenkins automation enables timely email alerts for pipeline executions, improving team communication and response time. Terraform, combined with Git, enables version-controlled infrastructure provisioning. This separation of automation and infrastructure showcases the modularity of DevOps and prepares for more complex workflows in the future.



**7.References**

1. Jenkins Documentation - https://www.jenkins.io/doc/
2. Email Extension Plugin - https://plugins.jenkins.io/email-ext/
3. Terraform Documentation - https://developer.hashicorp.com/terraform/docs
4. AWS EC2 - https://aws.amazon.com/ec2/
5. Visual Studio Code - https://code.visualstudio.com/