# Deploying Apache Airflow on AWS EC2 with Terraform and Docker Compose

---

## Overview

This document outlines the step-by-step process to deploy Apache Airflow on an AWS EC2 instance using Terraform. The setup automatically provisions an EC2 instance, a new RDS Postgres database, an S3 bucket, a new key pair (used automatically), and installs Airflow using Docker Compose. Outputs include the EC2 Public IP, RDS Endpoint, and EC2 DNS.

## Part 1: Running Terraform to Provision Infrastructure

### Step 1: Clone the Terraform Project

Make sure you have the project containing the Terraform code locally.

```
git clone <repo-url>
cd <terraform-project-directory>
```

### Step 2: Initialize Terraform

```
terraform init
```

### Step 3: Review Terraform Plan

```
terraform plan -var-file=variables.json
```

### Step 4: Apply the Terraform Configuration

```
terraform apply -var-file=variables.json
```

You will be prompted to confirm. Type `yes`.

Terraform will perform the following:

- Create a new EC2 instance
- Create a new RDS database
- Create an S3 bucket
- Create a new Instance profile
- Create a new Security Group
- Uses the SG Created for EC2 to the RDS Created
- Generate a new key pair and uses automatically
- SSH into EC2 and install Docker, Docker Compose

- Sync available DAGs from the configured S3 bucket
- Deploy Airflow with Docker Compose
At the end, Terraform will output:

- EC2 Public IP
- EC2 Public DNS
- RDS Endpoint
- S3 Bucket Name

## Part 2: Accessing Apache Airflow UI

### Step 5: Open the Airflow Web UI

Navigate to the URL using the public IP or DNS:

```
http://<ec2-public-ip>:8080
http://<ec2-public-dns>:8080
```

### Step 6: Default Login Credentials

- Username: admin
- Password: admin

## Part 3: Destroying Infrastructure (Optional)

### Step 7: Teardown Resources

```
terraform destroy
```

This will delete the EC2, RDS, S3, and the key pair.

## Notes

- No manual SSH access or installation is required—Terraform automates everything.
- Ensure port 8080 is open in the security group to access the Airflow UI.
- Store Terraform output values for reference.
- Logs and DAGs are synced from the configured S3 bucket to the EC2 instance before Airflow is started.

***End of Document***