

# Jenkins CI/CD Pipeline for Flask Application

## 1. Project Overview

This project demonstrates the implementation of a Jenkins CI/CD pipeline for a Python Flask web application. The objective was to automate the process of building, testing, and deploying the application using Jenkins, GitHub, and Python tools.

## 2. Tools and Technologies Used

- Jenkins (CI/CD Automation)
- Git & GitHub (Source Code Management)
- Python 3 & Flask (Application)
- Pytest (Testing Framework)
- Linux (Ubuntu)

## 3. Project Structure

The repository is structured as follows:

```
jenkins_CI_CD_pipeline_for_flask_application/
├── Jenkinsfile
├── app.py
├── requirements.txt
├── test_app.py
├── README.md
└── documentation/
    ├── Jenkins_CICD_Overview.md
    ├── Pipeline_Stages.md
    ├── Setup_Prerequisites.md
    └── Screenshots.md
```

## 4. Jenkins Pipeline Explanation

The Jenkins pipeline is defined using a Jenkinsfile and consists of the following stages:

Checkout:

Jenkins pulls the latest source code from the GitHub repository.

Build:

A Python virtual environment is created and application dependencies are installed using pip.

Test:

Unit tests are executed using pytest. The pipeline proceeds only if tests pass successfully.

Deploy:

The Flask application is deployed to a staging environment after successful build and testing.

## 5. GitHub Webhook and Automation

A GitHub webhook is configured to automatically trigger the Jenkins pipeline whenever changes are pushed to the main branch. This ensures continuous integration without manual intervention.

## 6. Screenshots

The screenshot shows a GitHub repository page for 'jenkins\_CI\_CD\_pipeline\_for\_flask\_application'. The main branch is 'main', which has had recent pushes 23 minutes ago. There are 2 branches and 0 tags. The repository is 5 commits ahead of its fork. The commit history lists the following changes:

| Author | Commit Message                   | Time                                    | Commits        |
|--------|----------------------------------|---|----------------|
| Ubuntu | Added CI/CD documentation folder | 8104a0b · 23 minutes ago                | 75 Commits     |
|        | documentation                    | Added CI/CD documentation folder        | 23 minutes ago |
|        | .gitignore                       | UPdate files                            | 2 years ago    |
|        | Jenkinsfile                      | Added Jenkins CI/CD pipeline            | 1 hour ago     |
|        | README.md                        | Updated README with CI/CD documentation | 34 minutes ago |
|        | app.py                           | update                                  | 2 years ago    |
|        | requirements.txt                 | Update requirements.txt                 | 2 years ago    |
|        | test.py                          | update                                  | 2 years ago    |
|        | test_app.py                      | Added basic pytest test                 | 45 minutes ago |

The screenshot shows the Jenkins Stage View for the 'flask-ci-cd' pipeline. The pipeline has four stages: Declarative: Checkout SCM, Build, Test, and Deploy. The Build stage is currently executing, indicated by a progress bar. The Test stage has completed with a duration of 650ms. The Deploy stage has completed with a duration of 269ms. The Declarative: Post Actions stage has completed with a duration of 110ms. The Average stage times are listed as 674ms for Declarative: Checkout SCM, 5s for Build, 650ms for Test, 269ms for Deploy, and 110ms for Declarative: Post Actions.

|                      | Declarative: Checkout SCM | Build | Test  | Deploy | Declarative: Post Actions |
|----------------------|---------------------------|-------|-------|--------|---------------------------|
| Average stage times: | 674ms                     | 5s    | 650ms | 269ms  | 110ms                     |
| #1                   | Jan 06<br>01:39           | 643ms | 3s    | 596ms  | 162ms                     |
| #2                   | Jan 06<br>01:35           | 630ms | 5s    | 599ms  | 346ms                     |
| #3                   | Jan 06<br>01:29           | 616ms | 3s    | 611ms  | 378ms                     |
| #4                   | Jan 06<br>Now             | ---   | ---   | ---    | 109ms                     |

The screenshot shows the Jenkins interface for a build named 'flask-ci-cd' (number 5). The left sidebar has 'Console Output' selected. The main area is titled 'Console Output' and displays the following log output:

```
Started by GitHub push by Rushiargade
Obtained Jenkinsfile from git https://github.com/Rushiargade/jenkins\_CI\_CD\_pipeline\_for\_flask\_application.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/flask-ci-cd
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/flask-ci-cd/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/Rushiargade/jenkins\_CI\_CD\_pipeline\_for\_flask\_application.git #
timeout=10
Fetching upstream changes from https://github.com/Rushiargade/jenkins\_CI\_CD\_pipeline\_for\_flask\_application.git
> git --version # timeout=10
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/Rushiargade/
+ Jenkins CI CD pipeline for flask application git workspace threads/10s/for/execute/pipeline/timeout=10
```

The screenshot shows the GitHub Webhooks settings page. At the top, there's a heading "Webhooks" and a button "Add webhook". Below this, a paragraph explains what webhooks are and how they work. A list contains one item: "✓ http://18.144.12.145:8080/github-w... (push)". To the right of this list are "Edit" and "Delete" buttons. A message "Last delivery was successful." is displayed below the list.

The screenshot shows the Jenkins Pipeline details page for a pipeline named "flask-ci-cd". On the left, there's a sidebar with options like "Configure", "Delete Pipeline", "Full Stage View", "Stages", "Rename", "Pipeline Syntax", and "GitHub Hook Log". The main area displays a summary of build stages. It includes a table with columns: "Declarative: Checkout SCM" (659ms), "Build" (5s), "Test" (689ms), "Deploy" (320ms), and "Declarative: Post Actions" (105ms). Below this table, four individual stage cards are shown, each with a timestamp (Jan 06 01:50, 01:39, 01:35, 01:29), a commit count (1 commit), and a duration (599ms, 643ms, 630ms, 616ms). The Jenkins logo and navigation icons are at the top right.

## 7. Conclusion

This project successfully demonstrates a complete CI/CD workflow using Jenkins for a Flask application. Automation of build, test, and deployment improves reliability, reduces manual effort, and ensures faster delivery of applications following DevOps best practices.