FACULDADE ENGENHARIA

Departamento de Informática

Integrating Jenkins with Django

Why Jenkins?

- Automation: Automates repetitive tasks, reducing manual intervention.
- Scalability: Distributed build architecture with master-slave configuration.
- Extensibility: Over 1,500 plugins available for integration.
- Flexibility: Supports different coding languages, deployment environments, and testing frameworks.

Jenkins Architecture

- Jenkins Master: Orchestrates jobs and communicates with agents (or nodes).
- Jenkins Agents (Nodes): Executes jobs, distributed across various machines.
- Plugins: Extend Jenkins functionality (e.g., Git, Docker, pipeline plugins).

Jenkins Pipeline

- Pipelines automate build, test, and deployment processes.
- Defined using a *Jenkinsfile*, stored in version control.
- Declarative Pipeline: Structured, more user-friendly.
- Scripted Pipeline: More flexible but complex.

Benefits of Jenkins

- Efficient CI/CD: Streamlines the process from code commit to production deployment.
- Fast Feedback: Instant notifications of build and test failures.
- Improved Collaboration: Encourages frequent integration, quicker release cycles.

Step-by-Step: Jenkins with Django

- Step 1: Install Jenkins.
  - Use the package manager for your OS (e.g., apt for Ubuntu).

    sudo apt update

    sudo apt install openjdk-11-jre

    wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -

    sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

    sudo apt update

    sudo apt install jenkins

  - Start Jenkins

    sudo systemctl start jenkins

    sudo systemctl enable jenkins

Step-by-Step: Jenkins with Django

- Step 2: Install Plugins for Django Integration.
  - Git Plugin for cloning repositories.
  - Pipeline Plugin to create pipelines as code.
  - Python Plugin to run Django-related commands.

- Step 3: Set up a New Job for Django.
  - Create a New Pipeline Job:Open Jenkins, click on New Item, select Pipeline.
  - Name it and click OK.
  - Configure Git Repository.

Step-by-Step: Jenkins with Django

- Step 4: Write a Jenkins Pipeline for Django.

Example for Jenkinsfile (Do not forget to Commit this file to your Git repositor)

```
pipeline {
    agent any

    stages {
        stage('Clone Repository') {
            steps {
                git branch: 'main', url: 'https://github.com/username/django-project.git'
            }
        }
        stage('Install Dependencies') {
            steps {
                sh 'pip install -r requirements.txt'
            }
        }
```

Step-by-Step: Jenkins with Django

```
stage('Run Tests') {
            steps {
                sh 'python manage.py test'
            }
        }
        stage('Deploy') {
            steps {
                // Add your deployment script here (e.g., using Docker, or SSH to the server)
            }
        }
    }
}
```

Step-by-Step: Jenkins with Django

- Step 5: Build and Monitor.
  - Go to your Jenkins job and click Build Now.
  - Monitor the pipeline process through the Blue Ocean interface for easy visualization.
  - Check logs for any issues and ensure that the Django tests and deployment steps run successfully.


HOW
WHY
WHERE
WHEN
WHERE
WHY
WHERE
WHO
WHEN
WHAT
WHAT
WHY
WHEN
WHAT
HOW
WHEN
WHO
WHERE
WHY
HOW