Student Handout: Introduction to Jenkins

# 1. What is Jenkins?

Definition: Jenkins is an open-source automation server that helps developers build, test, and deploy software using CI/CD (Continuous Integration/Continuous Delivery) practices.

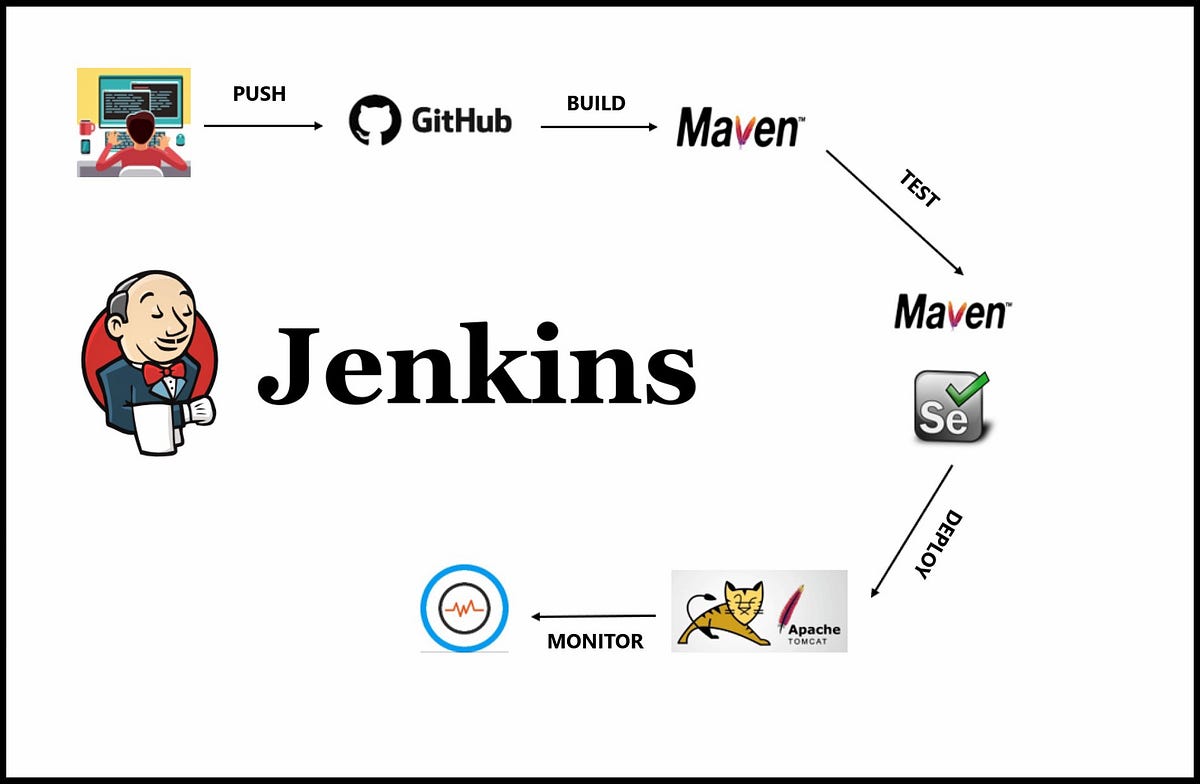
Purpose: Automates repetitive tasks in the software development lifecycle, reducing manual effort and errors.

Key Roles:

- Automation Server: Manages builds, tests, and deployments.

- CI/CD Enabler: Ensures code changes are continuously tested and deployed.

- Extensible Platform: Supports 1800+ plugins for integration with nearly any tool.



# 2. History and Evolution

- 2004 → Hudson project started at Sun Microsystems.

- 2011 → Forked as Jenkins after Oracle acquired Hudson.

- 2016 → Release of Jenkins 2.0 with "Pipeline as Code."

- 2018+ → Cloud-native features and Kubernetes integration.

# 3. Why Jenkins is Popular

1. Open Source & Free – No licensing costs.

2. Extensive Plugin Ecosystem – Works with almost any DevOps tool.

3. Flexibility & Customization – Pipelines as Code.

4. Strong Community – Large user base, frequent updates, active forums.

Impact: Widely used in startups, enterprises, and Fortune 500 companies.

# 4. Jenkins vs. Other CI/CD Tools

| Tool | Strengths | Limitations |

|---------------|---------------------------------------------|--------------------------------------|

| Jenkins | Customizable, huge plugin ecosystem | Needs maintenance, manual server mgmt|

| GitHub Actions| GitHub-native, cloud-hosted, YAML-based | Limited to GitHub, new ecosystem |

| GitLab CI | Great GitLab integration, container-native | Best with GitLab repos only |

| Azure DevOps | Microsoft ecosystem, full ALM suite | Less flexible, commercial pricing |

Jenkins remains a favorite for flexibility and enterprise-grade extensibility.

# 5. Real-World Use Cases

- Netflix → Runs thousands of automated tests daily.

- Amazon → Deploys code to production several times per day.

- Financial Institutions → Integrate security scanning in pipelines.

- Mobile Developers → Automate iOS/Android builds, signing & releases.

# 6. Key Components of Jenkins

1. Jenkins Master → Central controller with web UI.

2. Jenkins Agents → Worker nodes for distributed builds.

3. Pipelines → Code-defined CI/CD workflows (via Jenkinsfile).

4. Plugins → Extend Jenkins to work with cloud, containers, testing tools, etc.

# 7. Getting Started

Installation Options:

- Install on Linux, Windows, macOS.

- Run in Docker:

docker run -p 8080:8080 -p 50000:50000 jenkins/jenkins:lts

- Deploy on Kubernetes for scalability.

Initial Configuration:

- Access at http://localhost:8080

- Unlock Jenkins (admin password from logs).

- Install suggested plugins.

- Create the first admin user.

Create Your First Pipeline (Example Jenkinsfile):

pipeline {

agent any

stages {

stage('Build') { steps { echo 'Building...' } }

stage('Test') { steps { echo 'Testing...' } }

stage('Deploy') { steps { echo 'Deploying...' } }

}

}

# 8. Resources

- Jenkins Official Documentation: https://www.jenkins.io/doc/

- Plugin Index: https://plugins.jenkins.io/

- Pipeline Examples: https://www.jenkins.io/doc/pipeline/examples/

# Key Takeaways

- Jenkins = Automation + CI/CD + Extensibility.

- Still dominant despite newer tools, especially in large enterprises.

- Core strength lies in plugins, flexibility, and pipelines as code.