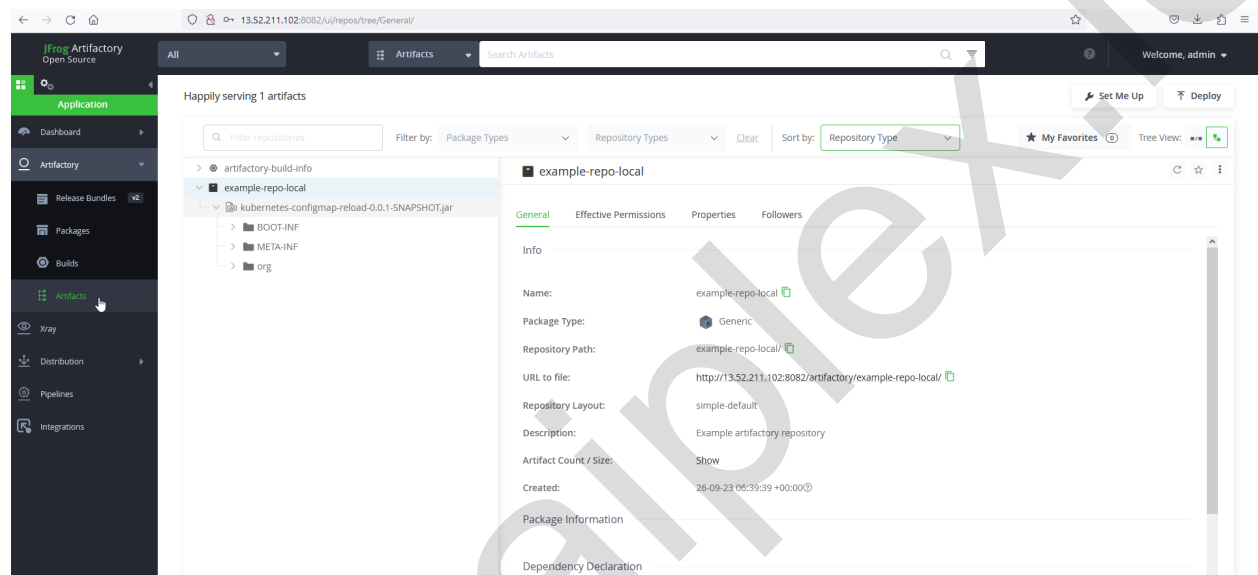



```
[INFO] Installing /home/ubuntu/Java_app_3.0/pom.xml to /home/ubuntu/.m2/repository/co
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 21.912 s
[INFO] Finished at: 2023-09-26T06:50:35Z
[INFO] -----

Maven build completed successfully.

PUT request was successful!
```

JAR File Pushed to JFrog



Part2 – Groovy Pipeline Script Stage to Integrate the Jfrog in the CI Pipeline

Modified Python Code

```
#!/usr/bin/env python3

import requests
import subprocess

def jfrogUpload():
    # Define the URL, file path, and authentication credentials
    url = 'http://44.219.155.115:8082/artifactory/example-repo-local/kubernetes-configmap-reload-0.0.1-SNAPSHOT.jar'
    file_path = '/var/lib/jenkins/workspace/HW2/target/kubernetes-configmap-reload-0.0.1-SNAPSHOT.jar'
    username = 'admin'
    password = 'U1123456' # Replace 'your_password' with the actual password

    # Send the PUT request with authentication and file upload
    with open(file_path, 'rb') as file:
        response = requests.put(url, auth=(username, password), data=file)

    # Check the response status code
    if response.status_code == 201:
        print("\nPUT request was successful!")
    else:
        print(f"PUT request failed with status code {response.status_code}")
        print("Response content:")
        print(response.text)

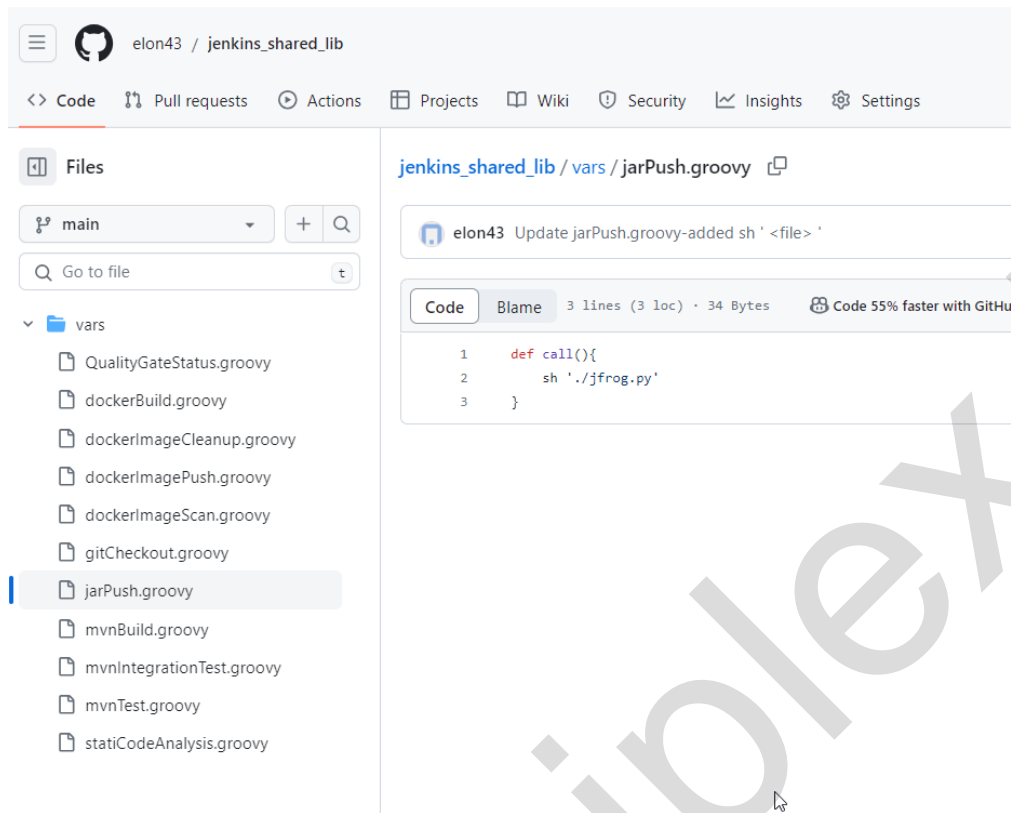
def mvnBuild():
    # Define the Maven command
    maven_command = "mvn clean install -DskipTests"

    # Run the Maven command as a subprocess
    try:
        subprocess.run(maven_command, check=True, text=True, shell=True)
        print("\nMaven build completed successfully.")
    except subprocess.CalledProcessError as e:
        print(f"Error: Maven build failed with exit code {e.returncode}")

def main():
    # mvnBuild()
    jfrogUpload()

#####
if __name__ == "__main__":
    main()
```

Groovy Script



The screenshot shows a GitHub repository for 'elon43 / jenkins_shared_lib'. The 'Files' tab is active, showing a list of files under the 'vars' directory. The file 'jarPush.groovy' is selected. The code view shows the following content:

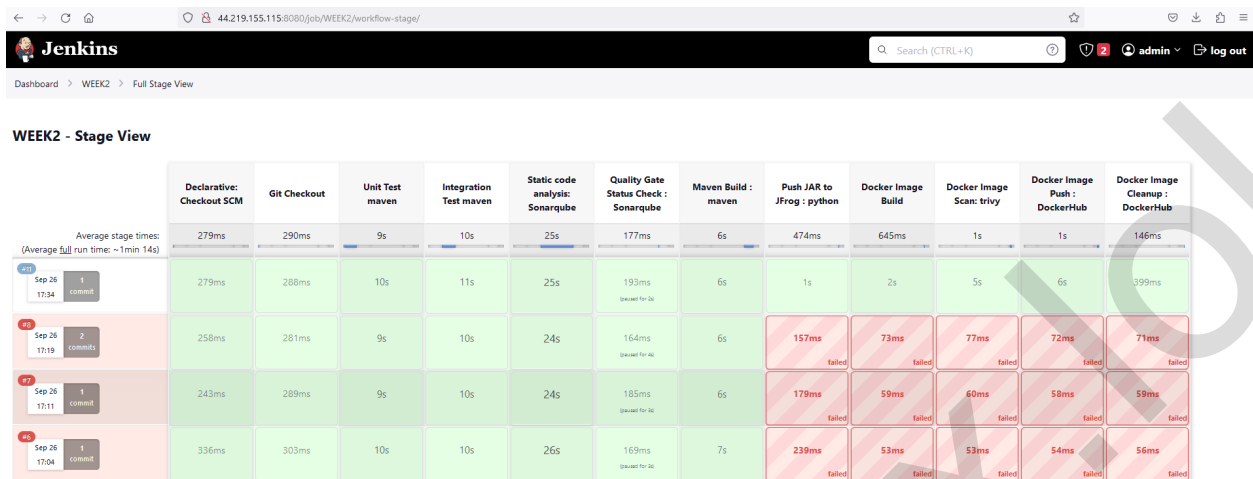
```
1  def call(){
2    sh './jfrog.py'
3  }
```

The repository has 3 lines (3 loc) and 34 Bytes. A badge indicates 'Code 55% faster with GitHub'.

New Stage Added

```
66     stage('Maven Build : maven'){
67         when { expression { params.action == 'create' } }
68         steps{
69             script{
70
71                 mvnBuild()
72             }
73         }
74     }
75     stage('Push JAR to JFrog : python'){
76         when { expression { params.action == 'create' } }
77         steps{
78             script{
79
80                 jarPush()
81             }
82         }
83     }
84     stage('Docker Image Build'){
85         when { expression { params.action == 'create' } }
86         steps{
87             script{
88
89                 dockerBuild("${params.ImageName}", "${params.ImageTag}", "${params.DockerHubUser}")
90             }
91         }
92     }
```

Completed Stage



JAR File Pushed to JFrog

