JENKINS EXERCISES

NAME:LOKESH

Exercise 1: Freestyle Job - Windows File Transfer Automation 1. Objective: Automate file transfer between two directories on a Windows machine.

2. Steps: o Create a Freestyle Job in Jenkins. o Configure the job to execute a Windows batch command: xcopy C:\source-directory C:\target-directory /E /I /H /Y o Add a post-build action to check the success of the transfer using a custom message.

3. Task: Test the job by placing some files in the source directory and verifying they are transferred to the target directory.

A screenshot of a computer

Description automatically generated

RESULT:

A screenshot of a computer program

Description automatically generated

Exercise 2: Freestyle Job - Build and Deploy a JavaScript Project

1. Objective: Build and deploy a JavaScript-based web project.

2. Steps: o Create a Freestyle Job. o Configure the job to: ▪ Clone a JavaScript repository (e.g., a React project) from GitHub. ▪ Run npm install to install dependencies. ▪ Build the project using npm run build. ▪ Copy the build folder to a deployment directory using a Windows batch command or PowerShell. o Archive the build artifacts for future reference.

3. Task: Verify the deployment by accessing the web application from the deployment directory.

A screenshot of a computer program

Description automatically generated

RESULT:

A screenshot of a computer program

Description automatically generated

Exercise 3: Freestyle Job - Flask Application Deployment

1. Objective: Automate the deployment of a Flask application.

2. Steps: o Create a Freestyle Job. o Configure the job to: ▪ Pull the Flask application repository from GitHub. ▪ Set up a virtual environment using Python: python -m venv venv .\venv\Scripts\activate pip install -r requirements.txt ▪ Start the Flask development server. o Add a post-build action to verify the server is running (e.g., using curl or a similar tool to hit a test endpoint).

3. Task: Test the application deployment by accessing it via localhost

A screenshot of a computer

Description automatically generated

RESULT:

A screenshot of a computer

Description automatically generated

Exercise 4: Pipeline Job - End-to-End CI/CD for a React Application

1. Objective: Implement a CI/CD pipeline for a React web application.

2. Steps: o Create a Pipeline Job. o Write a Jenkinsfile to: ▪ Clone the React project from GitHub. ▪ Install dependencies (npm install). ▪ Run tests (npm test). ▪ Build the project (npm run build). ▪ Deploy the application by copying the build folder to a deployment directory. o Add stages for: ▪ Linting using ESLint. ▪ Post-deployment testing using curl to verify the app is running.

3. Task: Run the pipeline and validate each stage's output



Exercise 5: Pipeline Job - Deploy a Flask Application with Gunicorn

1. Objective: Automate the deployment of a Flask application with Gunicorn on Windows.

2. Steps: o Create a Pipeline Job. o Write a Jenkinsfile to: ▪ Clone a Flask application from GitHub. ▪ Set up a Python virtual environment. ▪ Install required packages using pip install -r requirements.txt. ▪ Configure and start the Gunicorn server: gunicorn -b 127.0.0.1:8000 app:app ▪ Verify the deployment using a curl command in the pipeline. o Add stages for: ▪ Unit tests using pytest. ▪ Post-deployment endpoint checks.

3. Task: Trigger the pipeline and ensure the Flask app is accessible on localhost.

A screenshot of a computer script

Description automatically generated

RESULT:

A screenshot of a computer program

Description automatically generated