1 Create a simple Hello World Flask app (with venv)

Windows CMD / PowerShell commands

```
# Create project folder
mkdir flask-hello
cd flask-hello

# Create virtual environment
python -m venv venv

# Activate venv
.\venv\Scripts\activate

# Install Flask
pip install flask

# Freeze dependencies
pip freeze > requirements.txt
```

---

app.py

```
from flask import Flask

app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello, World from Flask & Jenkins!"

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

---

2 Create `.gitignore` (to avoid committing venv)

```
venv/
__pycache__/
*.pyc
flask.log
```

---

3 Initialize Git & Push to GitHub

```
# Initialize repo
git init
git add .
git commit -m "Initial commit: Hello World Flask"

# Link to GitHub (replace URL)
git remote add origin https://github.com/YOUR_USERNAME/flask-hello.git
git branch -M main
git push -u origin main
```

Now your Hello World Flask app is live in GitHub.

---

4 Setup Jenkins to Auto-Build on Commit

In Jenkins

1. Install GitHub Integration and Pipeline plugins.
2. Create a New Item → Pipeline.
3. In Pipeline from SCM:
   - SCM: Git
   - Repo URL: `https://github.com/YOUR_USERNAME/flask-hello.git`
   - Branch: `*/main`
   - Script Path: `Jenkinsfile`
4. Add GitHub Webhook:
   - Use ngrok to expose Jenkins:

     ```
     ngrok http 8080
     ```

   - Copy HTTPS URL from ngrok (e.g., `https://abc123.ngrok-free.app`)
   - GitHub Repo → Settings → Webhooks → Add Webhook:

     ```
     Payload URL: https://abc123.ngrok-free.app/github-webhook/
     Content type: application/json
     Just the push event
     ```

   - Save.

---

5 Jenkinsfile (Windows, Flask, Auto Deploy)

Create this file in your project root and commit it:

```
pipeline {
    agent any

    tools {
        python 'Python_3.11'   // Must be defined in Jenkins Global Tool Config
    }

    triggers {
        githubPush()
    }

    stages {
        stage('Checkout') {
            steps {
                git branch: 'main', url: 'https://github.com/YOUR_USERNAME/flask-hello.git'
            }
        }

        stage('Install Dependencies') {
            steps {
                bat '''
                python -m venv venv
                call venv\\Scripts\\activate
                pip install --upgrade pip
                pip install -r requirements.txt
                '''
            }
        }

        stage('Run Unit Tests') {
            steps {
                bat '''
                call venv\\Scripts\\activate
                pytest --maxfail=1 --disable-warnings -q || echo "No tests found"
                '''
            }
        }

        stage('Generate Version Document') {
            steps {
                bat '''
                echo Build Version: > version.txt
                git rev-parse --short HEAD >> version.txt
                echo Build Time: >> version.txt
                powershell -Command "Get-Date -Format 'yyyy-MM-dd HH:mm:ss'" >> version.txt
                '''
```

```
                archiveArtifacts artifacts: 'version.txt', followSymlinks: false
            }
        }

        stage('Local Deploy') {
            steps {
                bat '''
                call venv\\Scripts\\activate
                start /B python app.py
                '''
            }
        }
    }

    post {
        success {
            echo "Flask app built, tested, and deployed successfully!"
        }
        failure {
            echo "Build failed!"
        }
    }
}
```

---

6 How it works

- When you `git push` → GitHub sends webhook → Jenkins job triggers.
- Jenkins:
  1. Pulls latest code
  2. Creates venv & installs dependencies
  3. Runs tests (if present)
  4. Creates `version.txt`
  5. Deploys Flask app locally on port 5000
-

> You can see logs in Jenkins or **`flask.log`** if you modify to redirect output.

Got it — we'll make the Deploy stage smart:

1. Check if Flask is already running on port 5000
2. If yes → kill that process
3. Start the new instance in the background

Here's the updated Windows Jenkinsfile:

```
pipeline {
    agent any

    tools {
        python 'Python_3.11'    // Must match name in Jenkins Global Tool Config
    }

    triggers {
        githubPush()
    }

    stages {
        stage('Checkout') {
            steps {
                git branch: 'main', url: 'https://github.com/YOUR_USERNAME/flask-hello.git'
            }
        }

        stage('Install Dependencies') {
            steps {
                bat '''
                python -m venv venv
                call venv\\Scripts\\activate
                pip install --upgrade pip
                pip install -r requirements.txt
                '''
            }
        }

        stage('Run Unit Tests') {
            steps {
                bat '''
                call venv\\Scripts\\activate
                pytest --maxfail=1 --disable-warnings -q || echo "No tests found"
                '''
            }
        }

        stage('Generate Version Document') {
            steps {
                bat '''
                echo Build Version: > version.txt
                git rev-parse --short HEAD >> version.txt
```

```
                echo Build Time: >> version.txt
                powershell -Command "Get-Date -Format 'yyyy-MM-dd HH:mm:ss'" >> version.txt
                '''
                archiveArtifacts artifacts: 'version.txt', followSymlinks: false
            }
        }

        stage('Local Deploy') {
            steps {
                bat '''
                REM ===== Kill old Flask process on port 5000 =====
                for /f "tokens=5" %%a in ('netstat -ano ^| findstr :5000 ^| findstr LISTENIN

                REM ===== Start new Flask app =====
                call venv\\Scripts\\activate
                start /B python app.py
                '''
            }
        }
    }

    post {
        success {
            echo "Flask app built, tested, and deployed successfully!"
        }
        failure {
            echo "Build failed!"
        }
    }
}
```

What Changed in Deploy Stage

- **`netstat -ano | findstr :5000`** Finds any process listening on port 5000
- **`taskkill /F /PID %%a`** Force kills the process ID found
- Then it starts `python app.py` fresh in background.

Extra Tip

If your app takes time to start, Jenkins might exit before Flask is ready. You can add a short delay after starting:

```
timeout /T 3 >nul
```
