VIỆN CÔNG NGHỆ THÔNG TIN T3H

Đào tạo chuyên sâu - Trải nghiệm thực tế

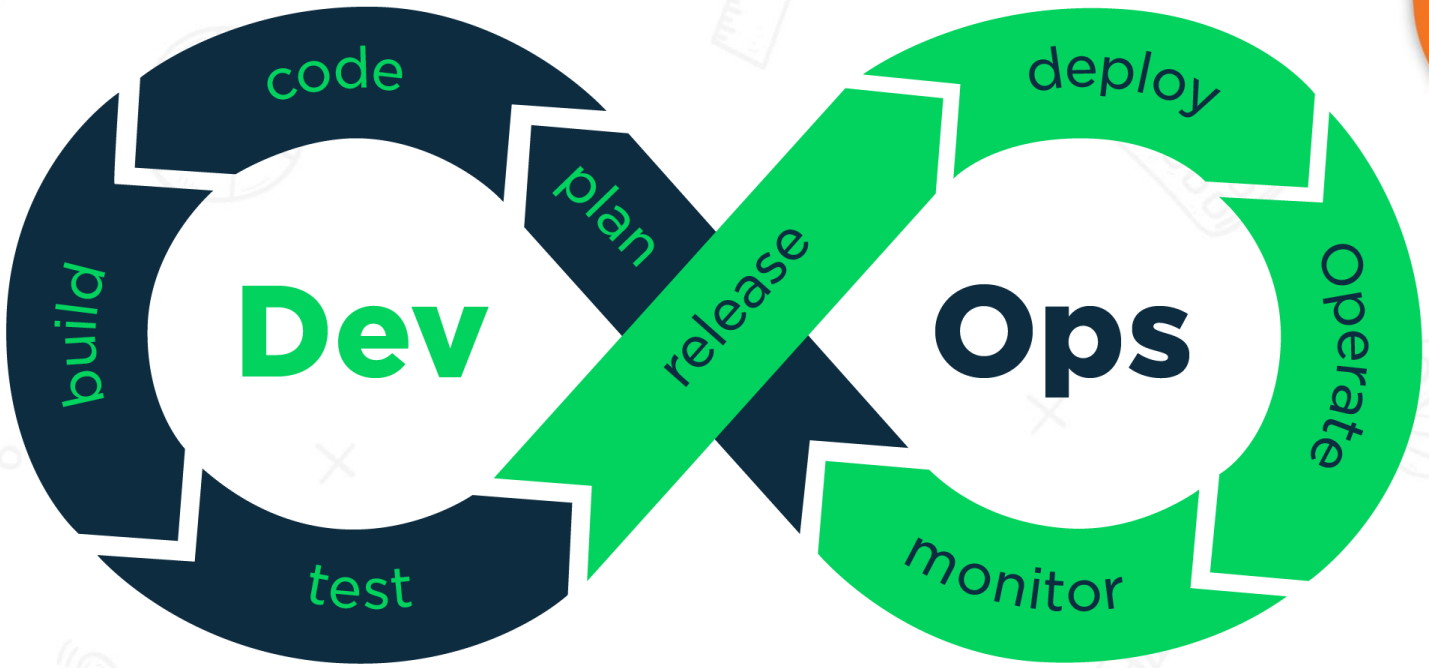
Môn học

DevOps

Giảng viên

Tan Do

0868880797



Nội dung buổi 09

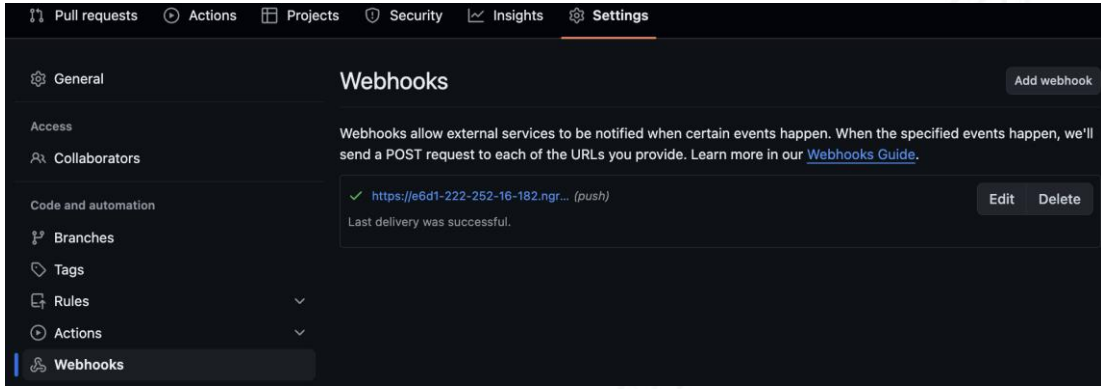
- Tích hợp Jenkins với GitHub
- Tích hợp Jenkins với GitHub pipeline
- Pipeline syntax
- Cài đặt plugin Jenkins
- Docker với Jenkins
  - Build Docker image và push Docker registry
- Gửi Email thông báo tự động

Tích hợp Jenkins với GitHub

- Tạo repo trên GitHub. Vd ‘mediplus-lite’ (free-css.com)
- Tạo job trên Jenkins. Vd job2
- Thêm link Jenkins vào GitHub Webhooks
  - GitHub > Repo > Settings > Webhooks
  - Webhooks
    - Payload URL: https://1978-1-55-247-180.ngrok-free.app/github-webhook/
  - Tạo project trên Jenkins
    - Freestyle project
    - SCM > Git
      - Nhập thông tin repo, credentials, branch specifier > cập nhật từ */master thành */main
    - Build Triggers > GitHub hook trigger for GITScm polling
  - Click Save
- Build Now
- Xác nhận kết quả trong Console Output
- Sửa code và push code lên GitHub
- Xác nhận kết quả

Tích hợp Jenkins với GitHub pipeline

- Tạo repo trên GitHub. Vd ‘mediplus-lite’ (free-css.com)
- Thêm file Jenkinsfile (slide sau) vào source code và push code lên GitHub
- Tạo job mới trên Jenkins. Vd job3
- Thêm link Jenkins vào GitHub Webhooks
  - GitHub > Repo > Settings > Webhooks
  - Webhooks
    - Access trên trình duyệt bằng link ngrok để đảm bảo truy cập ok trước khi nhập vào Payload URL
    - Payload URL: https://1978-1-55-247-180.ngrok-free.app/github-webhook/
  - Tạo project trên Jenkins
    - Pipeline project
    - Build Triggers > GitHub hook trigger for GITScm polling
    - Pipeline > Pipeline script from SCM
      - SCM: Git
        - Nhập thông tin repo git
        - Branches to build > Branch specifier > cập nhật từ */master thành */main
        - Script Path: Xác nhận là Jenkinsfile
  - Click Save
- Build Now
- Xác nhận kết quả trong Console Output
- Sửa code và push code lên GitHub
- Xác nhận kết quả



Tích hợp Jenkins với GitHub pipeline

- Xem thêm tại Pipeline Syntax
  - Để generate Pipeline script trên Jenkins
    - Sample step: git: Git
      - Nhập thông tin repo, branch (main), credentials
      - Click Generate Pipeline Script
- Jenkinsfile

```
pipeline {
    agent any
    stages{
        stage('Clone') {
            steps {
                git branch: 'main', credentialsId: 'cred-github', url:
'https://github.com/letdoitnow/mediplus-lite'
            }
        }
    }
}
```

Cấu trúc Jenkinsfile

- pipeline {}: Là top-level khi bắt đầu pipeline code, tất cả các thành phần sections, stages, khai báo docker image , run sh script ..vv đều phải nằm trong cặp thẻ này
- agent
  - Là môi trường để chạy pipeline code, các đoạn shell script vv...
  - Agent có thể hiểu chính jenkins master, slave machine nào đó, hoặc docker image
  - Options:
    - any : Chỉ định cho bất cứ master/slave nào available sẽ được pick để chạy
    - none : Không sử dụng, thay vì đó các stages sẽ phải tường minh chỉ định agent cho chính nó (Xem phần stages ở phía dưới)
    - label : tên của agent (master hoặc slave machine) sẽ dụng để thực thi nhiều stages đã khai báo. (Khác với any ở chỗ - any sẽ tự động pick master/slave machine available)
    - node : Giống như label nhưng có thể thêm nhiều options hơn. Hay nói cách khác option node này là phụ trợ cho label
    - docker : sử dụng docker image cho việc chạy pipeline - https://hub.docker.com/explore/
- post
  - Sẽ được chạy cuối cùng khi kết thúc Pipeline hoặc stage (giống như finally trong Java đó)
  - Nhằm handle kết quả của pipeline hoặc chạy các tác vụ cần chạy sau cùng
- stages
  - Là một sub section nữa bên trong pipeline
  - Một stages có thể hiểu là chứa nhiều stage con
  - Mỗi stage sẽ đóng một vai trò đảm nhiệm khác nhau tùy bài toán. Nó giống như việc viết nhiều method trong 1 class vậy/s
  - Nếu agent là none bạn phải chỉ định agent tương ứng cho các stage (refer section agent ở mục trên)
- steps
  - Là một thành phần nằm bên trong stage con. Nơi đặt các xử lý logic bên trong nó

Cấu trúc Jenkinsfile

```
pipeline {
    agent any

    stages {
        stage('Build') {
            steps {
                sh 'make'
            }
        }
        stage('Test'){
            steps {
                sh 'make check'
                junit 'reports/**/*.xml'
            }
        }
        stage('Deploy') {
            steps {
                sh 'make publish'
            }
        }
    }
}
```

Cấu trúc Jenkinsfile

```
pipeline {

    // agent any
    // agent none
    // agent { label 'master' }
    // agent { label 'manhnv-slave' }
    /* agent {
            node {
                label 'manhnv-slave'
                customWorkspace '/path/to/custom/workspace'
            }
        }
    */
    // Hoặc với docker image
    agent { docker 'maven:3-alpine' }
    stages {
        stage('Example Build') {
            steps {
                sh 'mvn clean compile exec:java package -P staging'
            }
        }
    }
}
```

Cấu trúc Jenkinsfile

```
pipeline {
    agent any
    stages {
        stage('Example') {
            steps {
                echo 'Hello World'
            }
        }
    }

    post {
        always {
            echo 'One way or another, I have finished'
            deleteDir() /* clean up our workspace */
        }
        success {
            echo 'I succeeeded!'
        }
        unstable {
            echo 'I am unstable :/'
        }
        failure {
            echo 'I failed :('
        }
        changed {
            echo 'Things were different before...'
        }
    }
}
```

Cấu trúc Jenkinsfile

```
pipeline {
    agent any
    stages {
        stage('Clone') {
            //implement pipeline code
        }
        stage('Build') {
            //implement pipeline code
        }
        stage('Test') {
            //implement pipeline code
        }
    }
}
```

Cấu trúc Jenkinsfile

```
pipeline {
    agent any
    stages {
        stage('Example') {
            steps {
                echo 'Hello World'

                script {
                    def browsers = ['chrome', 'firefox']
                    for (int i = 0; i < browsers.size(); ++i) {
                        echo "Testing the ${browsers[i]} browser"
                    }
                }
            }
        }
    }
}
```

Cài đặt plugin

- Manage Jenkins
  - Plugins
    - Available plugins
      - Docker
      - Docker Pipeline
      - Maven Integration
      - NodeJS
      - GitHub Pull Request Builder
      - GitHub Integration
      - SSH
      - SSH Agent
      - Publish Over SSH
      - …
    - Install

Build Docker image và push Docker registry

- Manage Jenkins
  - Plugins
    - Docker Pipeline
- Generate pipeline script
  - Sample step: With docker registry: Sets up docker registry endpoint
    - Docker registry URL: https://index.docker.io/v1/
    - Registry credentials: Add Jenkins, và cung cấp username/password
      - Vào trang docker hub > User > Account settings > Security > New Access Token
  - Sample step: sh: Shell script
    - Shell script:
      - docker build -t codetheps/web:2.0 .
      - docker push codetheps/web:2.0

Build Docker image và push Docker registry

- Jenkinsfile

```
pipeline {
    agent any
    stages{
        stage('Clone') {
            steps {
                git branch: 'main', credentialsId: 'github', url: 'https://github.com/letdoitnow/mediplus-lite.git'
            }
        }
        stage('Push') {
            steps {
                // This step should not normally be used in your script. Consult the inline help for details.
                withDockerRegistry(credentialsId: 'dockerhub', url: '') { // url default https://index.docker.io/v1/
                    // some block
                    sh label: '', script: 'docker build -t codetheps/web:2.0 .'
                    sh label: '', script: 'docker push codetheps/web:2.0'
                }
            }
        }
    }
}
```

Gửi Email thông báo tự động

- Cấu hình
  - Manage Jenkins
    - System
      - Nhập thông tin mục Email Notification
        - SMTP Server: smtp.gmail.com
        - Advance
          - Use SMTP Authentication (checked)
            - User Name: codetheps@gmail.com
            - Password:
              - Vào Quản lý account gmail
              - Kích hoạt xác thực 2 lớp
              - Về trang chủ, vào lại xác thực 2 lớp và tạo App Password
          - Use SSL
          - Use TLS
        - SMTP Port: 465
        - Check Test configuration by sending test e-mail, nhập email test e-mail
          - Click Test configuration > Xác nhận email nhận được
        - Nhập thông tin Jenkins Location
          - System Admin e-mail address. Vd Jenkins <jenkins@x.com>
          - Click Test configuration > Xác nhận email nhận được
      - Save

Gửi Email thông báo tự động

- Generate pipeline script
  - Sample: mail: Mail
  - Nhập thông tin box mail
    - To, CC, Subject, Body
    - Generate script

Gửi Email thông báo tự động

- Generate pipeline script

```
pipeline {
    agent any
    stages{
        stage('Clone') {
            steps {
                git branch: 'main', credentialsId: 'github', url: 'https://github.com/letdoitnow/mediplus-lite.git'
            }
        }
    }
    post {
        always {
            // sshagent(['ssh']) {
            //     // some block
            //     sh 'ssh -o StrictHostKeyChecking=no -l root ssh-ubuntu touch test.txt' // 172.17.0.2
            // }
            mail bcc: '', body: 'Jenkins Body', cc: '', from: '', replyTo: '', subject: 'Jenkins Subject', to:
'codetheps@gmail.com'
        }
    }
}
```

T3H
IT-Institute

VIỆN CÔNG NGHỆ THÔNG TIN T3H

Đào tạo chuyên sâu - Trải nghiệm thực tế

THANK YOU

www.t3h.edu.vn