BÀI TẬP JENKINS

1.Các thao tác cơ bản

Bài Tập 1: Tạo Job Đơn Giản Trên Jenkins

I.Mục tiêu bài tập

- Làm quen với quy trình tạo và cấu hình một job trong Jenkins.
- Thực hành chạy một job đơn giản và kiểm tra kết quả.
- Hiểu cách Jenkins sử dụng các bước build và console output để báo cáo kết quả job.

II.Nội dung bài tập

1.Chuẩn bị môi trường: Đăng nhập vào Jenkins bằng tài khoản Admin.

2.Tạo job đầu tiên trong Jenkins

2.1.Khởi tạo job

- Bước 1: Trên giao diện chính của Jenkins, chọn **New Item** để tạo một item mới.
- Bước 2: Đặt tên cho job (ví dụ: Job dau tien) và chọn **Freestyle Project**.
- Bước 3: Nhấn OK để tạo job.

2.2.Cấu hình job

Chuyển đến phần cấu hình của job và thực hiện các thiết lập cơ bản:

- Description: Thêm mô tả cho job (ví dụ: “Đây là job đầu tiên của tôi trong Jenkins”).
- Discard Old Builds: Bật tính năng này để Jenkins tự động xóa các build cũ, giúp tiết kiệm tài nguyên.

2.3.Thiết lập bước build cho job

- Trong phần cấu hình Build, chọn Add build step và chọn Execute shell
- (Linux)
- Thêm câu lệnh sau để in ra màn hình thông báo “Hello I’m [ten_cua_ban], I come from HCMUS!”:

  echo "Hello I’m Mr Teo, I come from HCMUS!"

3.Chạy job và kiểm tra kết quả

3.1.Chạy job: Sau khi lưu cấu hình, trên giao diện job, nhấn **Build Now** để chạy job lần đầu tiên.

3.2.Kiểm tra kết quả

- Sau khi job hoàn thành, nhấn vào số build ở mục **Build History** để mở chi tiết của job đó.
- Chọn **Console Output** để xem chi tiết quá trình chạy của job.
- **Kết quả**: Dòng lệnh echo " Hello I’m Mr Teo, I come from HCMUS!" sẽ hiển thị thông báo “Hello I’m Mr Teo, I come from HCMUS!” trong console output.

4.Thực hành thêm các bước mở rộng: thiết lập trigger tự động cho job

- Trong phần cấu hình Build Triggers, chọn Build periodically và đặt thời gian chạy định kỳ, ví dụ: 2 * * * * để job chạy mỗi 2 phút.
- Lưu lại và kiểm tra lịch trình chạy tự động của job.

5.Kiểm tra và kết quả

- Đảm bảo job hiển thị chính xác thông báo “Hello I’m Mr Teo, I come from HCMUS!” và thông điệp từ biến MESSAGE.
- Đánh giá quá trình thiết lập trigger tự động để job chạy đúng theo lịch định kỳ.

Bài tập 2: Mở rộng từ bài tập 1

Thay vì chạy lệnh echo, các bạn tạo ra job mới thay đổi với việc kết hợp nhiều câu lệnh trên Ubuntu như các lệnh kiểm tra hệ thống: uname -a, cat /etc/*release, netstat -tpln

Bài Tập 3: Cài Đặt Plugin Trên Jenkins

I.Mục tiêu bài tập

- Hiểu quy trình cài đặt plugin trên Jenkins.
- Làm quen với kho plugin và tìm kiếm các plugin hữu ích.
- Thực hành cài đặt và cấu hình các plugin Pipeline stage view

II.Nội dung bài tập

1.Truy cập trang quản lý plugin

- Đăng nhập vào Jenkins bằng tài khoản Admin.
- Trên giao diện Jenkins, chọn Manage Jenkins > Plugins để vào kho plugin của Jenkins.

2.Tìm hiểu các tab trong kho plugin

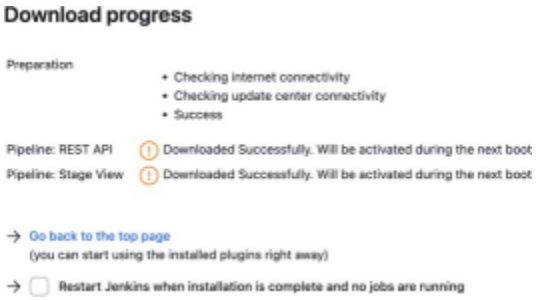
- Updates: Danh sách các plugin có bản cập nhật mới.
- Available: Các plugin có sẵn để cài đặt trên Jenkins.
- Installed: Danh sách plugin đã cài đặt trên Jenkins.
- Advanced: Cấu hình cập nhật plugin và cho phép tải plugin từ nguồn ngoài.

3.Cài đặt Plugin

3.1.Tìm kiếm Plugin

- Trong tab Available, sử dụng ô tìm kiếm và nhập Pipeline stage view.

- Chọn plugin nhấn Install without restart để cài đặt ngay lập tức (không yêu cầu khởi động lại Jenkins).



- Nếu cài đặt xong Jenkins bắt buộc phải restart thì các bạn tick chọn Restart…
- Sau khi cài đặt xong, kiểm tra tab Installed để đảm bảo plugin đã được cài đặt thành công.

Bài Tập 4: Tạo và Phân Quyền User Trên Jenkins

I.Mục tiêu bài tập

- Hướng dẫn cách tạo user trong Jenkins và phân quyền cho từng user.
- Hiểu cách quản lý quyền truy cập để đảm bảo bảo mật và tổ chức phân quyền hiệu quả.
- Kiểm tra quyền truy cập của các user khác nhau và thực hành các thao tác quản lý user cơ bản.

Yêu cầu:

- Tạo 2 user developer và viewer với cụ thể quyền hạn như sau:
- Developer: sẽ có quyền tạo, chỉnh sửa và chạy job
- Viewer: chỉ có quyền xem và không thể thay đổi hoặc chạy bất kỳ job nào

II.Nội dung bài tập

1.Cấu hình bảo mật cho Jenkins:

- Đăng nhập vào Jenkins với tài khoản Admin.
- Truy cập Manage Jenkins > Security.

- Trong phần Authorization, chọn Matrix-based security để thiết lập phân quyền chi tiết cho từng user và từng chức năng của Jenkins.

2.Tạo user:

- Truy cập Manage Jenkins > Manage Users để tạo các user.
- Nhấn Create User để hoàn tất.

3.Phân quyền cho các user

- Quay lại Security và vào phần Matrix-based security.
- Nhấn Add user và nhập tên user.
- Gán các quyền cần thiết cho user developer như sau:
    - Overall: Chọn Read
    - Job: Chọn Read, Build, Workspace, và Configure.
    - View: Chọn Read.
    - Run: Chọn pdate, Delete.
- Gán quyền hạn chế cho viewer như sau:
    - Overall: Chọn Read.
    - Job: Chọn Read.
    - View: Chọn Read.
- Lưu lại cấu hình. User viewer chỉ có quyền xem và không thể thay đổi hoặc chạy bất kỳ job nào.

4.Kiểm tra quyền truy cập của các user

- Đăng xuất khỏi Jenkins và đăng nhập bằng tài khoản developer, kiểm tra xem user có thể:
    - Tạo và cấu hình các job mới.
    - Chạy các job hiện tại và xem console output.
    - Thực hiện các thao tác chỉnh sửa trên job.
- Đăng xuất và đăng nhập lại với tài khoản viewer, kiểm tra xem user có thể:
    - Chỉ xem các job hiện có.
    - Không thể chỉnh sửa, tạo mới hoặc chạy job.

5.Thực hành điều chỉnh quyền cho user

- Thay đổi quyền của user ‘developer’
    - Đăng nhập lại với tài khoản Admin.
    - Trong Matrix-based security, thay đổi quyền của developer để không cho phép chỉnh sửa job (bỏ chọn quyền Configure trong Job).
    - Đăng xuất và đăng nhập với user developer để kiểm tra thay đổi quyền.

- Thêm quyền mới cho user ‘viewer’

    - Trở lại tài khoản Admin và cho phép viewer có quyền Build trong phần Job.
    - Đăng nhập với tài khoản viewer và kiểm tra xem user có thể chạy các job hiện tại hay không.
- Kiểm tra và đánh giá phân quyền
    - Đảm bảo rằng các quyền đã được thiết lập đúng theo yêu cầu của bài tập.
    - Đánh giá xem các quyền khác nhau ảnh hưởng như thế nào đến khả năng truy cập và thực thi công việc trên Jenkins.

Bài tập mở rộng:

- Tạo thêm một user với quyền quản lý pipeline (có quyền tạo và chạy pipeline nhưng không thể thay đổi các job khác).
- Tìm hiểu và thực hành thiết lập xác thực qua LDAP hoặc Active Directory cho Jenkins để quản lý user tập trung.

2.Thực hành Pipeline:

Bài Tập 1: Tạo Pipeline Jenkins

I.Mục tiêu bài tập

- Hiểu cách tạo và cấu hình một pipeline cơ bản trong Jenkins.
- Thực hành sử dụng Jenkinsfile để định nghĩa các giai đoạn trong pipeline.
- Thực hiện các bước build, test, và deploy đơn giản trên Jenkins.

II.Nội dung bài tập

1.Chuẩn bị môi trường

- Đăng nhập vào Jenkins với tài khoản Admin.
- Đảm bảo đã có sẵn một repository Git chứa mã nguồn ứng dụng mẫu để thực hành pipeline.

2.Tạo một Pipeline Job trong Jenkins

2.1.Khởi tạo Pipeline Job

- Trên giao diện chính của Jenkins, chọn New Item để tạo một item mới.
- Đặt tên cho job (ví dụ: bt1-pipeline) và chọn Pipeline.
- Nhấn OK để tạo job.

2.2.Cấu hình Pipeline Job

- Trong phần cấu hình của job, đi đến mục Pipeline và nhập script như bên dưới: (Code trên Moodle)

```
pipeline {
      agent any
```

```
        stages {
        stage('Build') {
                steps {
                        echo "Building.."
                        sh '''
                        echo "doing build stuff.."
                        '''
                }
        }
        stage('Test') {
                steps {
                        echo "Testing.."
                        sh '''
                        echo "doing test stuff..
                        '''
                }
        }
        stage('Deliver') {
                steps {
                        echo 'Deliver....'
                        sh '''
                        echo "doing delivery stuff.."
                        '''
                }
        }
        }
}
```

- Giải thích các phần:
- agent any: Cho phép pipeline chạy trên bất kỳ Agent nào có sẵn.
- stages: Gồm các giai đoạn chính của pipeline, như Build, Test, và Deploy.
- steps: Các lệnh cụ thể trong mỗi giai đoạn.

3.Chạy Pipeline và kiểm tra kết quả

3.1.Thực hiện pipeline: Quay lại Jenkins, vào job vừa tạo và nhấn Build Now để chạy pipeline lần đầu tiên.

3.2.Kiểm tra kết quả từng giai đoạn

Sau khi pipeline chạy, vào từng build trong phần Build History và chọn Console Output để kiểm tra kết quả của từng giai đoạn.

Đảm bảo rằng:

- Build Stage: Ứng dụng được build thành công.
- Test Stage: Các kiểm thử chạy thành công.
- Deploy Stage: Ứng dụng được deploy thành công

Bài Tập 2: Tạo Pipeline Jenkins chạy Jenkins file từ Github

I.Mục tiêu bài tập

- Hiểu cách thiết lập kết nối giữa Jenkins và GitHub.
- Thực hành tạo và quản lý thông tin đăng nhập (credentials) trong Jenkins.
- Thiết lập chạy script pipeline từ Jenkinsfile
- Thiết lập trigger để Jenkins tự động build khi có thay đổi trên repository GitHub.

II.Nội dung bài tập

1.Chuẩn bị:

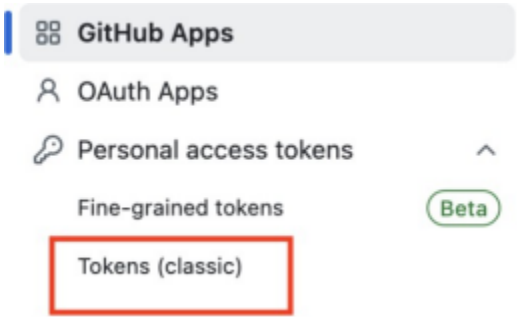
- Đăng nhập vào Jenkins với tài khoản Admin.
- Tạo một repository trên GitHub mà bạn muốn Jenkins kết nối để pull Jenkinsfile. Ví dụ trong bài tập này tôi tạo Repository bt2-pipeline.
    - Tạo file Jenkinsfile với nội dung script như bài tập 1

2.Tạo Personal Access Token (PAT) trên GitHub

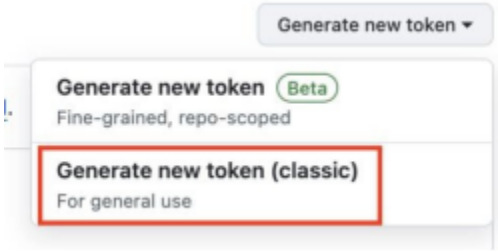
2.1.Truy cập GitHub: Đăng nhập vào tài khoản GitHub của bạn và truy cập Settings.

2.2.Tạo Personal Access Token

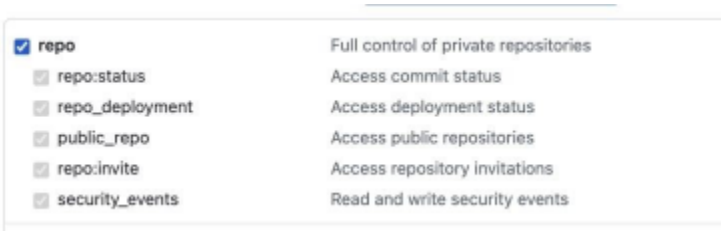
- Vào Developer settings > Personal access tokens > Tokens (classic).



- Chọn Generate new token (Classic) và đặt tên cho token (ví dụ: Jenkins Access Token).



- Tick chọn Repo để có quyền đọc vào tất cả repository.



- Nhấn Generate token và sao chép token vừa tạo (lưu lại token ở nơi an toàn vì sau khi thoát ra cửa sổ này bạn sẽ không xem lại được token id này nữa).

3.Thêm GitHub Credentials vào Jenkins

3.1.Truy cập quản lý thông tin đăng nhập trong Jenkins

Trên giao diện Jenkins, chọn Manage Jenkins > Credentials.

3.2.Thêm GitHub Token vào Jenkins

- Chọn (global) trong Stores scoped to Jenkins, sau đó nhấn Add Credentials.

- Điền thông tin như sau:
  - Kind: Chọn Username and pasword.
  - Username: Nhập username github
  - Password: Dán Personal Access Token (PAT) từ GitHub vào đây.
  - ID: Đặt ID cho token (ví dụ: github-token).
  - Description: (Không bắt buộc) Thêm mô tả như “GitHub PAT for Jenkins”.
- Nhấn Create để lưu lại.

4.Tạo một Pipeline Job để sử dụng GitHub Repository

4.1.Khởi tạo một Pipeline Job

- Trên giao diện chính của Jenkins, chọn New Item để tạo một item mới.
- Đặt tên cho job (ví dụ: bt2-pipeline) và chọn Pipeline.
- Nhấn OK để tạo job.

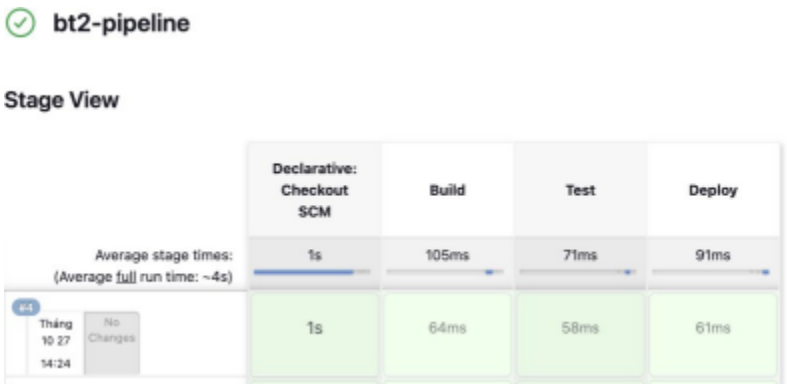
4.2.Cấu hình Pipeline Job để kết nối với GitHub

- Trong phần cấu hình của job, đi đến mục Pipeline và chọn Pipeline script from SCM.
- Chọn loại SCM: Chọn Git.

- Repository URL: Nhập URL của GitHub repository (ví dụ: https://github.com/username/bt2-pipeline.git).
- Credentials: Chọn credentials bạn đã tạo với GitHub Token
- Branch: Chọn nhánh main là nhánh bạn đang sử dụng.
- Script Path: Điền Jenkinsfile nếu bạn sử dụng Jenkinsfile trong repository.
- Nhấn Save để lưu cấu hình job.

5.Chạy Pipeline và Kiểm tra Kết Quả

- Quay lại Jenkins, Nhấn build job vừa tạo và kiểm tra xem pipeline đã được tự động chạy chưa.
- Kiểm tra Console Output để xem chi tiết quá trình thực thi và kiểm tra xem có kết nối thành công tới GitHub không.
- Nếu hiển thị như hình bên dưới nghĩa là chạy pipeline thành công:



Bài tập 3: Thiết lập Webhook trên GitHub để tự động trigger job

I.Mục tiêu bài tập

Hiểu cách thiết lập trigger để Jenkins tự động build khi có thay đổi trên repository GitHub.

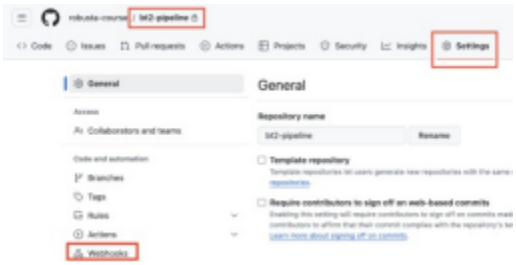
Yêu cầu:

- Thiết lập cấu hình webhook từ Github đến Jenkins
- Tạo pipe mới đặt tên là bt3-pipeline-trigger trong đó cấu hình phần Pipeline như bài tập 2 phần Pipeline
- Cấu hình thêm 1 số phần liên quan đến auto trigger trên Jenkins
- Thay đổi dòng chữ Deploying thành Deploying automation Deploy-staging trong file Jenkinsfile bài tập 2
- Github sẽ tự động phát hiện có commit mới trong repo ở bài tập 2 và build lại.

II.Nội dung bài tập

1.Truy cập phần Webhooks của GitHub Repository

- Đăng nhập vào Github.com
- Truy cập repository trên GitHub, chọn Settings > Webhooks > Add webhook.



2.Cấu hình Webhook để kết nối với Jenkins

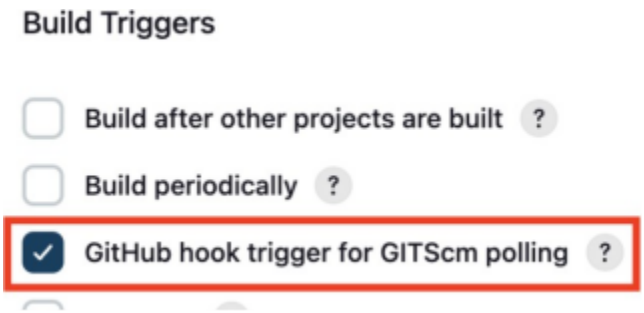
Cần cài đặt **ngrok**

- Payload URL: Nhập địa chỉ URL của Jenkins và thêm /github-webhook/ ở cuối.
- Lưu ý: các bạn cần điền url Jenkins có thể truy cập được từ internet (ví dụ: https://keen-honeybee-tops.ngrok-free.app/github-webhook/).
- Content type: Chọn application/json.
- Secret: Để trống (hoặc bạn có thể thêm một secret nếu cần bảo mật, sau đó cấu hình secret này trong Jenkins).
- Events: Chọn Just the push event để trigger khi có commit mới.

Nhấn Add webhook để hoàn tất thiết lập.

Tạo webhook thành công sẽ như hình ảnh trên.

3.Cấu hình pipeline: như bài tập 2 mục pipeline và thêm các cài đặt Enable Github-trigger như hình:



4.Chạy Pipeline và Kiểm tra Kết Quả

4.1.Kiểm tra hoạt động của Webhook

- Trên GitHub, thực hiện một thay đổi đơn giản trong repository (ví dụ: chỉnh sửa một file và commit).
- GitHub sẽ gửi một thông báo qua Webhook đến Jenkins để khởi động pipeline.

4.2.Kiểm tra kết quả trong Jenkins

- Quay lại Jenkins, vào job vừa tạo và kiểm tra xem pipeline đã được tự động chạy chưa.
- Kiểm tra Console Output để xem chi tiết quá trình thực thi và kiểm tra xem có kết nối thành công tới GitHub không.
- Hình ảnh trên thể hiện Jenkins auto trigger thành công khi có commit mới.