

THAM KHẢO NHANH GITLAB CI/CD

Pipelines, jobs, stages, variables, artifacts, environments

Cơ Bản Về Pipeline

Cách Pipeline Hoạt Động

Pipeline Container cấp cao nhất; một pipeline cho mỗi commit/trigger

Stage Nhóm các job chạy song song

Job Tác vụ đơn lẻ (script) trong một stage

Runner Agent thực thi các job

Kích Hoạt Pipeline

Push to branch Tự động (mặc định)

Merge request Qua workflow:rules hoặc only: merge_requests

Schedule CI/CD → Schedules trong cài đặt dự án

API POST /projects/:id/trigger/pipeline

Manual Nút Run Pipeline trong menu CI/CD

.gitlab-ci.yml

Cấu Hình Tối Giản

```
stages: [build, test, deploy]
```

```
build-job:
```

```
  stage: build
```

```
  script: echo "Compiling..."
```

Từ Khóa Toàn Cục

stages Định nghĩa thứ tự stage

default Giá trị mặc định cho tất cả job

variables Biến CI/CD toàn cục

workflow Kiểm soát khi nào pipeline được tạo

include Import file YAML bên ngoài

Include Templates

```
include:
```

```
  - template: Auto-DevOps.gitlab-ci.yml
```

```
  - local: .ci/lint.yml
```

```
  - project: 'group/shared-ci'
```

```
    file: '/templates/deploy.yml'
```

Jobs

Định Nghĩa Job

```
test-unit:
```

```
  stage: test
```

```
  image: node:20
```

```
  script:
```

```
    - npm ci
```

```
    - npm test
```

Từ Khóa Job

script Lệnh shell cần chạy (bắt buộc)

before_script Lệnh chạy trước script chính

after_script Lệnh chạy sau (kể cả khi thất bại)

image Docker image cho job

rules Điều kiện để đưa job vào

needs Phụ thuộc DAG (bỏ qua thứ tự stage)

allow_failure Pipeline tiếp tục nếu job thất bại

retry Số lần tự động thử lại (0-2)

timeout Thời gian tối đa cho job

Rules

```
deploy:
```

```
  rules:
```

```
    - if: '$CI_COMMIT_BRANCH == "main"'
```

```
      when: manual
```

```
    - if: '$CI_PIPELINE_SOURCE == "merge_request_event"'
```

```
      when: never
```

```
    - when: on_success
```

Stages

Thứ Tự Stage

```
stages:
```

```
  - lint
```

```
  - build
```

```
  - test
```

```
  - deploy
```

Stage Mặc Định

pre Luôn chạy đầu tiên

build Stage mặc định 1

test Stage mặc định 2

deploy Stage mặc định 3

post Luôn chạy cuối cùng

DAG với needs

```
test-api:
```

```
  stage: test
```

```
  needs: ["build-api"] # skip waiting for full stage
```

```
test-web:
```

```
  stage: test
```

```
  needs: ["build-web"] # runs as soon as build-web done
```

Variables

Định Nghĩa Biến

```
variables:
```

```
  NODE_ENV: "production"
```

```
  DB_HOST: "postgres"
```

```
job:
```

```
  variables:
```

```
    NODE_ENV: "test" # job-level override
```

Biến Định Sẵn

CI_COMMIT_SHA Hash commit đầy đủ

CI_COMMIT_BRANCH Tên branch

CI_COMMIT_TAG Tên tag (nếu là tag pipeline)

CI_PIPELINE_ID ID pipeline duy nhất

CI_PROJECT_DIR Đường dẫn checkout repo

CI_MERGE_REQUEST_IID Số MR (chỉ cho MR pipeline)

CI_REGISTRY_IMAGE Đường dẫn image container registry

Protected & Masked

Protected Chỉ khả dụng trên branch/tag được bảo vệ

Masked Ẩn trong log job

File Ghi vào file tạm; đường dẫn trong biến

Artifacts

Lưu Artifacts

```
build:
```

```
  script: npm run build
```

```
  artifacts:
```

```
    paths: [dist/]
```

```
    expire_in: 1 week
```

Loại Artifact

paths File/thư mục cần lưu

exclude Mẫu cần bỏ qua

expire_in Tự động xóa sau khoảng thời gian

reports:junit JUnit XML cho tóm tắt test trong MR

reports:coverage_report Trục quan hóa coverage Cobertura

JUnit Report

```
test:
```

```
  script: pytest --junitxml=report.xml
```

```
  artifacts:
```

```
    reports:
```

```
      junit: report.xml
```

Cache

Cache Dependencies

```
test:
```

```
  cache:
```

```
    key: ${CI_COMMIT_REF_SLUG}
```

```
    paths: [node_modules/]
```

```
    script: npm ci && npm test
```

Cache vs Artifacts

Cache Tăng tốc job; không đảm bảo; tái sử dụng cùng key

Artifacts Truyền file giữa jobs/stages; đảm bảo

Cache Policies

pull-push Tải xuống + tải lên (mặc định)

pull Chỉ tải xuống (nhành hơn cho consumer)

push Chỉ tải lên (cho producer)

Environments

Định Nghĩa Environments

```
deploy-staging:
```

```
  stage: deploy
```

```
  environment:
```

```
    name: staging
```

```
    url: https://staging.example.com
```

```
    script: ./deploy.sh staging
```

Tính Năng Environment

name Tên environment (hiển thị trong UI)

url Link đến ứng dụng đã triển khai

on_stop Job chạy khi environment bị dừng

auto_stop_in Tự động dừng sau khoảng thời gian

action: stop Đánh dấu job là hành động dừng

Review Apps

```
review:
```

```
  environment:
```

```
    name: review/${CI_COMMIT_REF_SLUG}
```

```
    url: https://${CI_COMMIT_REF_SLUG}.example.com
```

```
    on_stop: stop-review
```

```
    auto_stop_in: 1 week
```

Docker

Build & Push Image

```
build-image:
```

```
  image: docker:24
```

```
  services: [docker:24-dind]
```

```
  script:
```

```
    - docker login -u $CI_REGISTRY_USER -p $CI_REGISTRY_PASSWORD
```

```
    - docker build -t $CI_REGISTRY_IMAGE:$CI_COMMIT_SHA
```

```
    - docker push $CI_REGISTRY_IMAGE:$CI_COMMIT_SHA
```

Services (Sidecar Containers)

```
test:
```

```
  image: python:3.12
```

```
  services:
```

```
    - postgres:16
```

```
    - redis:7
```

```
  variables:
```

```
    POSTGRES_DB: testdb
```

```
    POSTGRES_PASSWORD: secret
```

Docker-in-Docker

docker:24-dind Image DinD service

DOCKER_TLS_CERTDIR Đặt '/certs' hoặc '' để cấu hình TLS

DOCKER_HOST tcp://docker:2376 (TLS) hoặc :2375

Mẫu Phổ Biến

Monorepo (changes)

```
test-api:
```

```
  rules:
```

```
    - changes: [api/**/*]
```

```
test-web:
```

```
  rules:
```

```
    - changes: [web/**/*]
```

Manual Deploy Gate

```
deploy-prod:
```

```
  stage: deploy
```

```
  when: manual
```

```
  rules:
```

```
    - if: '$CI_COMMIT_BRANCH == "main"'
```

Parallel Matrix

```
test:
```

```
  parallel:
```

```
    matrix:
```

```
      PYTHON: ["3.10", "3.11", "3.12"]
```

```
      DB: ["postgres", "sqlite"]
```

```
    script: tox -e py${PYTHON}-${DB}
```