

HƯỚNG DẪN TRIỂN KHAI CI/CD VỚI DOCKER & GITHUB ACTIONS

1. MỤC TIÊU BÀI THỰC HÀNH

Sau khi hoàn thành bài này, sinh viên có thể:

- Hiểu khái niệm **CI/CD** và vai trò của Docker trong quy trình triển khai phần mềm
- Tạo **Dockerfile** cho một ứng dụng Node.js tối giản
- Thiết lập **GitHub Actions workflow** để tự động build & push Docker image
- Làm quen với **Secrets** trong GitHub phục vụ CI/CD

2. TỔNG QUAN KIẾN TRÚC CI/CD

Quy trình tổng quát:

Developer → git push → GitHub



GitHub Actions (CI/CD)



Build Docker Image



Push lên Docker Hub



(Tuỳ chọn) Deploy lên Server

Trong bài này:

- CI: Build & push Docker image
- CD: Minh họa deploy (chưa yêu cầu server thật)

3. YÊU CẦU CHUẨN BỊ

- Tài khoản GitHub
- Tài khoản Docker Hub: <https://hub.docker.com/>

4. CẤU TRÚC PROJECT MẪU

Project đơn giản gồm các file sau: <https://github.com/lamdb/it4409-ex4-cicd-docker.git>

```
ex4-cicd-docker/  
├── index.js  
├── package.json  
├── package-lock.json  
├── Dockerfile  
├── .gitignore  
└── .github/  
    ├── workflows/  
    └── cicd-docker.yml
```

4.1. File index.js

```
const express = require("express");  
const app = express();  
  
app.get("/", async (req, res) => {  
    res.status(200).send("Xin chào bạn");  
});  
  
// Start server  
app.listen(3000, () => {  
    console.log("Server running on http://localhost:3000");  
});
```

4.2. File package.json

```
{  
  "name": "ex4-cicd-docker",  
  "version": "1.0.0",  
  "description": "",  
  "main": "index.js",  
  "scripts": {  
    "test": "echo \"Error: no test specified\" && exit 1",  
    "start": "node index.js"
```

```

    },
    "keywords": [],
    "author": "",
    "license": "ISC",
    "type": "commonjs",
    "dependencies": {
      "express": "^5.2.1"
    }
  }
}

```

4.3. Dockerfile đơn giản

```

FROM node:18-alpine
WORKDIR /app
COPY package*.json ./
RUN npm install
COPY . .
EXPOSE 3000
CMD ["node", "index.js"]

```

4.4. Thiết lập GitHub Actions workflow

- Vị trí bắt buộc **.github/workflows/**. Ví dụ: **.github/workflows/cicd-docker.yml**
- Nội dung file workflow **.yml**

```

name: CI/CD and Docker

on:
  push:
    branches:
      - main

jobs:
  build-and-push:
    name: Build & Push Docker Image
    runs-on: ubuntu-latest

    steps:
      # 1. Lấy source code từ GitHub
      - name: Checkout source code
        uses: actions/checkout@v4

      # 2. Chuẩn bị môi trường build Docker
      - name: Set up Docker Buildx
        uses: docker/setup-buildx-action@v3

      # 3. Đăng nhập Docker Hub (dùng GitHub Secrets)
      - name: Login to Docker Hub

```

```

    uses: docker/login-action@v3
    with:
      username: ${ secrets.DOCKERHUB_USERNAME }}
      password: ${ secrets.DOCKERHUB_TOKEN }}

# 4. Build Docker image và push lên Docker Hub, lưu ý trường
tags: có dạng dockerhub-username/image-name:tag
# Ví dụ nếu tên tài khoản dockerhub của bạn là lamdb => đặt là
lamdb/it4409-sample:staging chẳng hạn

- name: Build and push Docker image
  uses: docker/build-push-action@v5
  with:
    context: .
    file: Dockerfile
    push: true
    tags: lamdb/it4409-sample:staging

deploy:
  name: Deploy to Server
  runs-on: ubuntu-latest
  needs: build-and-push

  steps:
    # 5. SSH vào server và deploy container, phần này không chạy
    được do không có server cụ thể
    - name: Deploy via SSH
      uses: appleboy/ssh-action@v1.0.3
      with:
        host: ${ secrets.SERVER_HOST }}
        username: ${ secrets.SERVER_USER }}
        key: ${ secrets.SERVER_SSH_KEY }}
        script: |
          cd /opt/express-app
          docker compose pull
          docker compose down
          docker compose up -d

```

5. Commit & push code lên GitHub

Trong thư mục project:

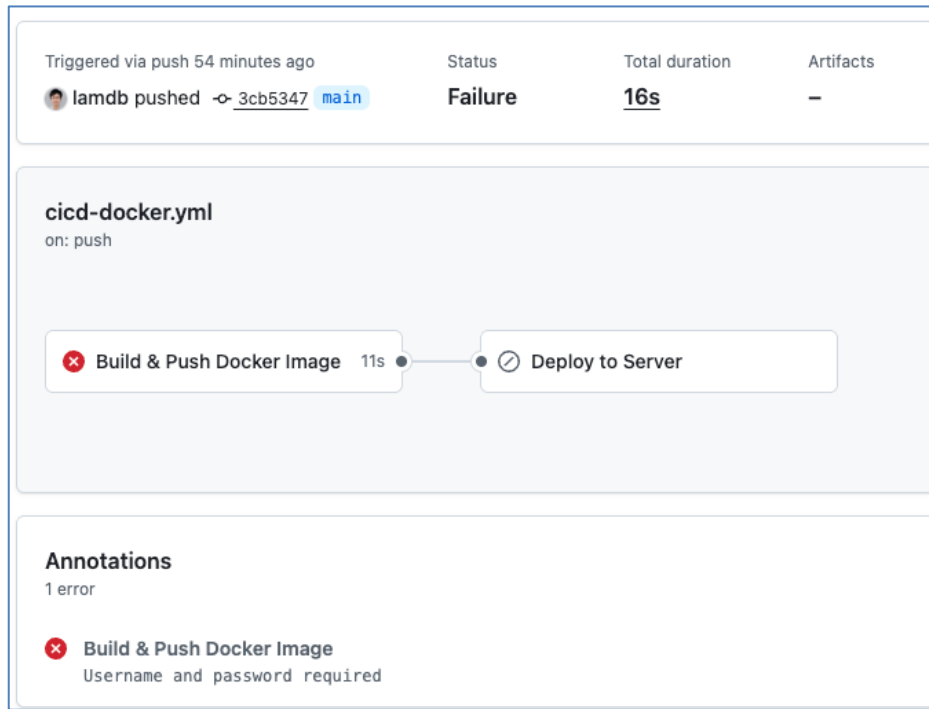
git add .

git commit -m "Add Dockerfile and CI/CD workflow"

git push origin main

Sau khi push:

- Vào tab **Actions** trên GitHub
- Quan sát workflow CI/CD chạy tự động
- Hiện sẽ gặp lỗi do chưa cấu hình các thông tin đăng nhập Dockerhub



6. Thiết lập đăng nhập vào DockerHub

- Tạo access token đăng nhập vào DockerHub
- Tài khoản => Settings => Personal access tokens. Chọn tên mô tả (tùy bạn), ngày hết hạn (tùy bạn) và quyền Read, Write, Delete.
- Copy và lưu lại xâu **access token** được tạo ra.

Create access token

A personal access token is similar to a password except you can have many tokens and revoke access to each one at any time. [Learn more](#)

Access token description

Github-CICD

Expiration date

None

Optional

Access permissions

Read, Write, Delete

Read, Write, Delete tokens allow you to manage your repositories.

Cancel

Generate

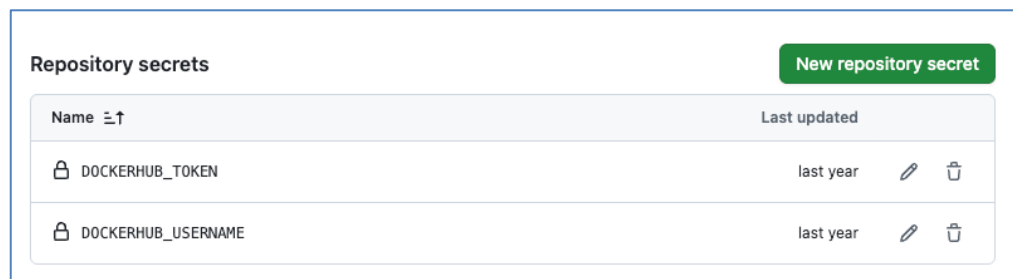
7. Cấu hình GitHub Secrets

- Không được hard-code mật khẩu trong code
- Secrets giúp bảo mật thông tin nhạy cảm

Thực hiện:

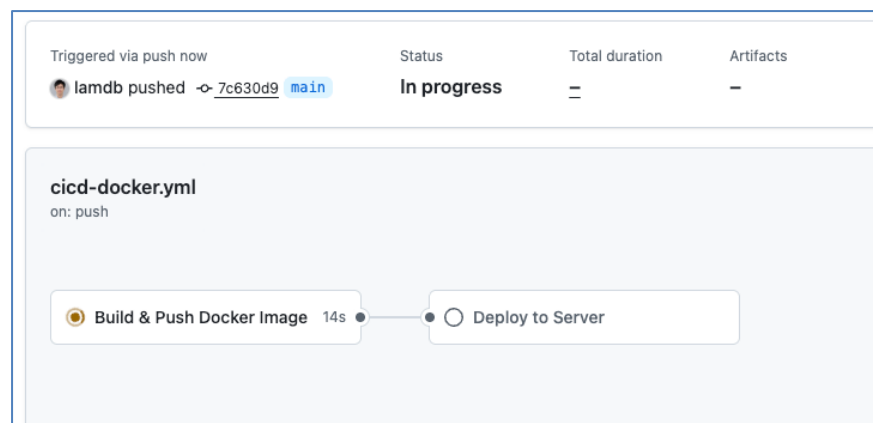
Vào Repository đã tạo trên Github → Settings → Secrets and variables → Actions → Repository secrets. Tạo:

Name	Ý nghĩa	Giá trị
DOCKERHUB_USERNAME	Tên tài khoản Docker Hub	Tên tài khoản DOCKER của bạn (xem ở mục Profile trên DockerHub)
DOCKERHUB_TOKEN	Access token Docker Hub	Xâu token đã tạo ở Bước 6



7. Chạy lại Github Actions

- Tạo một thay đổi trong project trên VSCode để commit lại => kích hoạt workflow
- Quan sát luồng hoạt động của workflow
- Kiểm tra xem trên Dockerhub đã thấy có file image được tạo ra hay chưa



Build & Push Docker Image

succeeded now in 21s

✓ Post Build and push Docker image

- 1 Post job cleanup.
- 2 ▶ Removing temp folder /home/runner/work/_temp/docker-actions-toolkit-55nDHz
- 3 ▶ Post cache

✓ Post Login to Docker Hub

- 1 Post job cleanup.
- 2 `/usr/bin/docker logout docker.io`
- 3 Removing login credentials for docker.io
- 4 ▶ Post cache

✓ Post Set up Docker Buildx

- 1 Post job cleanup.
- 2 ▼ Removing builder
- 3 `/usr/bin/docker buildx rm builder-c922b779-e0d4-494b-a779-ac83890faf1b`
- 4 `builder-c922b779-e0d4-494b-a779-ac83890faf1b` removed
- 5 ▶ Cleaning up certificates
- 6 ▶ Post cache

✓ Post Checkout source code

[Explore](#) / [lamdb](#) / it4409-sample



lamdb/it4409-sample

By [lamdb](#) · Updated 1 minute ago

IMAGE

☆0 ↓0

Overview **Tags**

Sort by

Newest ▼

🔍 Filter tags

TAG

■ [staging](#)

Last pushed 1 minute by [lamdb](#)