

**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**  
**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**  
**CƠ SỞ TẠI THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN 2**

---o0o---



**BÁO CÁO GIỮA KỲ & CUỐI KỲ**  
**TÀI LIỆU HƯỚNG DẪN THIẾT LẬP VÀ TRIỂN KHAI PHẦN MỀM**

**Môn học:** Chuyên đề Hệ thống thông tin

**Giảng viên hướng dẫn:** Lê Hà Thanh

**Lớp:** D20CQCNPM01-N

**Sinh viên thực hiện:**

Văn Tổ Hữu

N20DCCN026

*TP.HCM, ngày 20 tháng 06 năm 2024*

# MỤC LỤC

I.	GIỚI THIỆU TÀI LIỆU .....	2
II.	TỔNG QUAN WEBSITE.....	2
III.	YÊU CẦU HỆ THỐNG.....	2
IV.	HƯỚNG DẪN CÀI ĐẶT VÀ SỬ DỤNG .....	3
4.1.	Clone repository về máy tính cá nhân và cấu hình .....	3
4.2.	Triển khai cài đặt microservice với Kubernetes .....	4
4.3.	Triển khai cài đặt CMS (Reactjs) .....	8
4.4.	Triển khai cài đặt giao diện web movie (Reactjs) .....	10
4.5.	Triển khai CI/CD .....	12
V.	TÀI LIỆU THAM KHẢO.....	25

## **I.GIỚI THIỆU TÀI LIỆU**

- Tài liệu này dùng để:
  - Triển khai cài đặt website sử dụng kỹ thuật Kubernetes cho microservice.
  - Triển khai cài đặt CMS (Movie cms) và cài đặt Frontend (Movie web).
  - Triển khai cài đặt Backend.
  - Triển khai sử dụng CI/CD bằng Github Webhook và Jenkins.
- Phạm vi: tài liệu này hướng dẫn triển khai ở môi trường phát triển (Development Environment) và môi trường kiểm thử (Testing Environment).

## **II. TỔNG QUAN WEBSITE**

- Các công nghệ sử dụng:
  - OS-level virtualization: Docker Desktop
  - Reverse proxy server: Ingress NGINX
  - Content Management System (CMS): Reactjs
  - Database: SQL Server

## **III. YÊU CẦU HỆ THỐNG**

- Yêu cầu phần cứng:
  - Hệ điều hành: Windows 10/11 64-bit.
  - Dung lượng RAM: tối thiểu 4GB.
  - Dung lượng ổ cứng: tối thiểu 10GB.
  - Kích hoạt ảo hóa phần cứng (hardware virtualization) trong BIOS.
  - Bật được tính năng WSL 2 trên Windows.
- Yêu cầu phần mềm:
  - Cài đặt sẵn Docker Desktop, Git.
  - Cài đặt sẵn IDE hoặc TextEditor như Visual Studio Code, Notepad++, ...
  - Phần mềm quản lý mã nguồn như Git, ...

## IV. HƯỚNG DẪN CÀI ĐẶT VÀ SỬ DỤNG

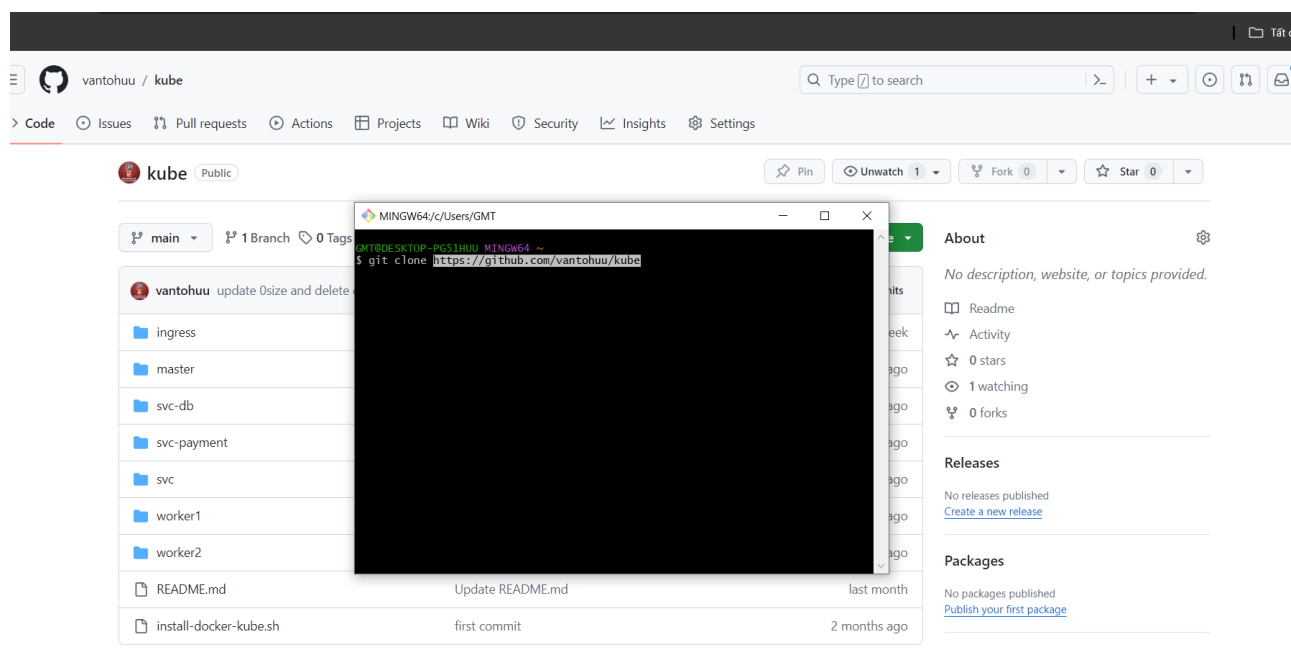
### 4.1. Clone repository về máy tính cá nhân và cấu hình

#### Source code gồm 5 phần

- Main service | Link: <https://github.com/vantohuu/SOA-Backend>
- Payment service | Link: <https://github.com/vantohuu/payment-service.git>
- Cms Movie | Link: <https://github.com/lucvan02/web-xem-phim-admin>
- Web Movie | Link: <https://github.com/lucvan02/web-xem-phim>
- Deploy kubernetes | Link: <https://github.com/vantohuu/kube>

#### Tạo một thư mục trống

Thực hiện lệnh: `git clone <link source>` để clone remote repository về máy tính

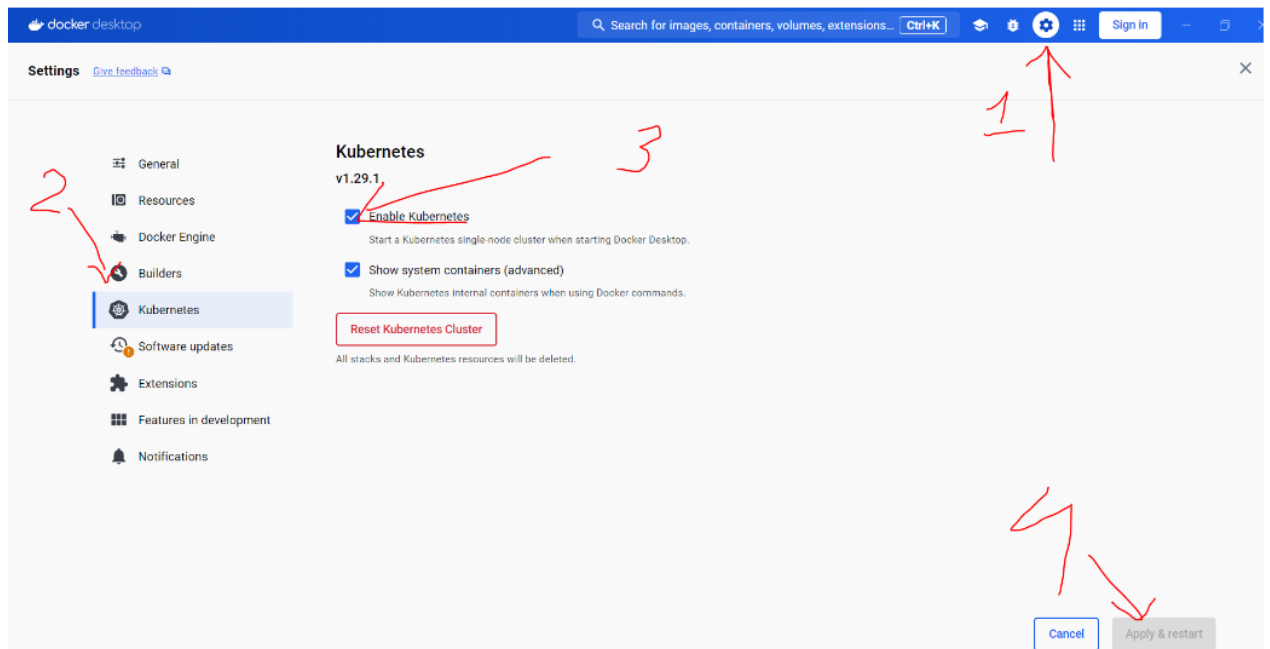


Hình 1. Giao diện phần mềm mẫu lệnh clone

## 4.2. Triển khai cài đặt microservice với Kubernetes

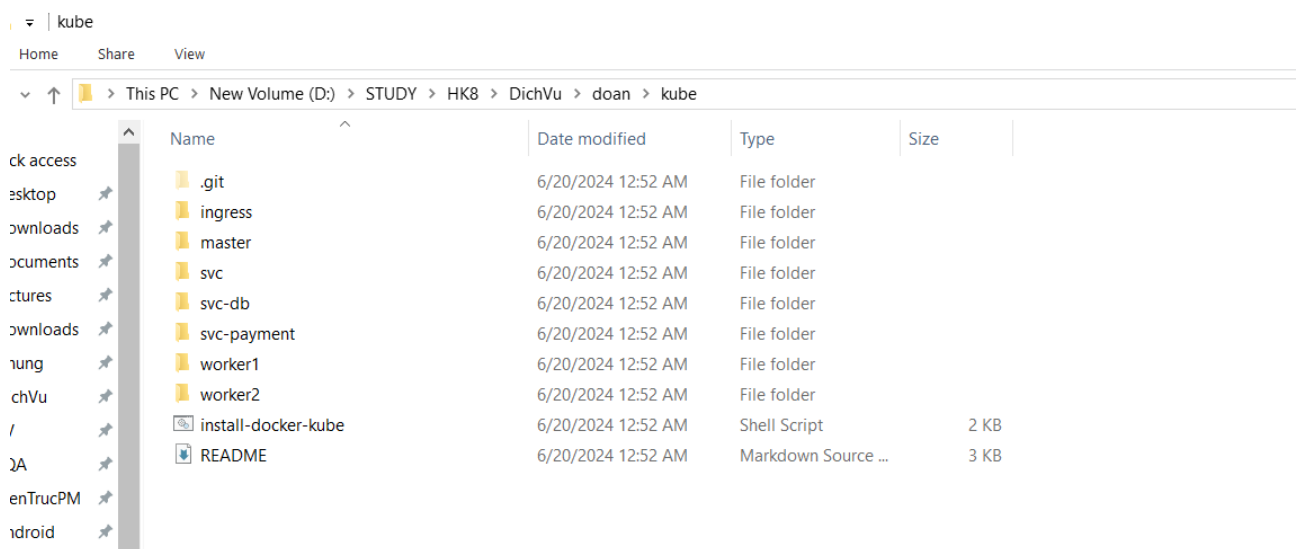
### Các bước triển khai

#### B1: Bật kubernetes trong docker



#### B2: Clone code về máy, trở vào thư mục kube

Link: <https://github.com/vantohuu/kube>



#### B3: Triển khai ingress nginx controller (Reverse proxy)

Tạo controller trong ingress-nginx

Thực hiện lệnh

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.3.0/deploy/static/provider/cloud/deploy.yaml
```

```
namespace/ingress-nginx created
serviceaccount/ingress-nginx created
serviceaccount/ingress-nginx-admission created
role.rbac.authorization.k8s.io/ingress-nginx created
role.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrole.rbac.authorization.k8s.io/ingress-nginx created
clusterrole.rbac.authorization.k8s.io/ingress-nginx-admission created
rolebinding.rbac.authorization.k8s.io/ingress-nginx created
rolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
configmap/ingress-nginx-controller created
service/ingress-nginx-controller created
service/ingress-nginx-controller-admission created
deployment.apps/ingress-nginx-controller created
job.batch/ingress-nginx-admission-create created
job.batch/ingress-nginx-admission-patch created
ingressclass.networking.k8s.io/nginx created
validatingwebhookconfiguration.admissionregistration.k8s.io/ingress-nginx-admission created
```

Kiểm tra Ingress controller pod đang chạy

```
kubectl get pods --namespace ingress-nginx
```

NAME	READY	STATUS	RESTARTS	AGE
ingress-nginx-admission-create--1-2blhw	0/1	Completed	0	10m
ingress-nginx-admission-patch--1-m9sv7	0/1	Completed	0	10m
ingress-nginx-controller-55dcf56b68-pd4nb	1/1	Running	0	10m

Kiểm tra Nginx Ingress controller đã được gán public IP hay chưa

```
kubectl get service ingress-nginx-controller --namespace=ingress-nginx
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
ingress-nginx-controller	LoadBalancer	10.0.132.242	20.117.254.250	80:31170/TCP,443:32018/TCP	16m

#### B4: Triển khai các service lên namespace ingress-nginx vừa triển khai xong

- *Triển khai soa service*

- Vào thư mục svc bằng cách: cd svc
- Triển khai pod
  - kubectl apply -f 3.pods.yaml --namespace ingress-nginx (nếu muốn xóa cái vừa triển khai thì thay apply thành delete)

■

- 

■

- 



- *Triển khai payment service*

- Vào thư mục svc bằng cách: `cd ../svc-payment`

- Triển khai pod
  - `kubectl apply -f 3.pods.yaml --namespace ingress-nginx` (nếu muốn xóa cái vừa triển khai thì thay apply thành delete)

```
apiVersion: v1
kind: Pod
metadata:
  name: soa-payment
  labels:
    app: soa-payment
spec:
  containers:
    - name: soa-payment
      image: huu2412002/soa-payment:v01
      resources:
        limits:
          memory: "512Mi"
          cpu: "500m"
      ports:
        - containerPort: 8081
    - name: redis
      image: redis
      resources:
        limits:
          memory: "300Mi"
          cpu: "200m"
      ports:
        - containerPort: 6379
```

- Triển khai service
  - `kubectl apply -f soapayment.svc.yaml --namespace ingress-nginx` (nếu muốn xóa cái vừa triển khai thì thay apply thành delete)

```
apiVersion: v1
kind: Service
metadata:
  name: svc-soa-payment
spec:
  clusterIP: 10.110.64.36
  selector:
    app: soa-payment
  type: ClusterIP
  ports:
    - name: port2
      port: 80
      targetPort: 8081
```

- ***Đưa các services lên ingress***

- Vào thư mục svc bằng cách: `cd ../ingress`
  - `kubectl apply -f nginx.ingress.yaml --namespace ingress-nginx` (nếu muốn xóa cái vừa triển khai thì thay apply thành delete)



```

apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: soa-ingress-ingress
  annotations:
    nginx.ingress.kubernetes.io/use-regex: "true"
    nginx.ingress.kubernetes.io/auth-snippet: |
      proxy_set_header Authorization $http_authorization;
    nginx.ingress.kubernetes.io/configuration-snippet: |
      add_header Access-Control-Allow-Methods "PUT, GET, POST, OPTIONS, DELETE";
      add_header Access-Control-Allow-Credentials true;
    nginx.ingress.kubernetes.io/enable-cors: "true"
    nginx.ingress.kubernetes.io/cors-allow-methods: "PUT, GET, POST, OPTIONS, DELETE"
    nginx.ingress.kubernetes.io/proxy-body-size: "0"
spec:
  ingressClassName: nginx
  rules:
    - http:
        paths:
          - path: /api/payment/(.*)
            pathType: Prefix
            backend:
              service:
                name: svc-soa-payment
                port:
                  number: 80
          - path: /(.* )
            pathType: Prefix
            backend:
              service:
                name: svc-soa-main
                port:
                  number: 80

```

**Kiểm tra những cái vừa triển khai: kubectl get all --namespace ingress-nginx**

```

PS C:\Users\GMT> kubectl get all --namespace ingress-nginx
NAME                                     READY   STATUS    RESTARTS   AGE
pod/ingress-nginx-admission-create-lxhfv 0/1     Completed 0           16d
pod/ingress-nginx-admission-patch-6hcgz   0/1     Completed 0           16d
pod/ingress-nginx-controller-7f49ccf564-k5lgh 1/1     Running   48 (11m ago) 16d
pod/soa-main                              2/2     Running   38 (34m ago) 13d
pod/soa-payment                           2/2     Running   32 (11m ago) 12d

NAME                                     TYPE          CLUSTER-IP      EXTERNAL-IP      PORT(S)                                     AGE
service/ingress-nginx-controller         LoadBalancer  10.111.119.61    localhost        80:32323/TCP,443:30120/TCP                16d
service/ingress-nginx-controller-admission ClusterIP      10.102.143.94    <none>           443/TCP                                         16d
service/svc-soa-main                     ClusterIP      10.107.244.72    <none>           80/TCP                                         13d
service/svc-soa-payment                   ClusterIP      10.110.64.36     <none>           80/TCP                                         12d

NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/ingress-nginx-controller 1/1     1             1           16d

NAME                                     DESIRED   CURRENT   READY   AGE
replicaset.apps/ingress-nginx-controller-7f49ccf564 1         1         1       16d

NAME                                     REFERENCE          TARGETS          MINPODS   MAXPODS   REPLICAS   AGE
horizontalpodautoscaler.autoscaling/soamain-scaler  ReplicaSet/soa-main <unknown>/50%    1         2         0          13d

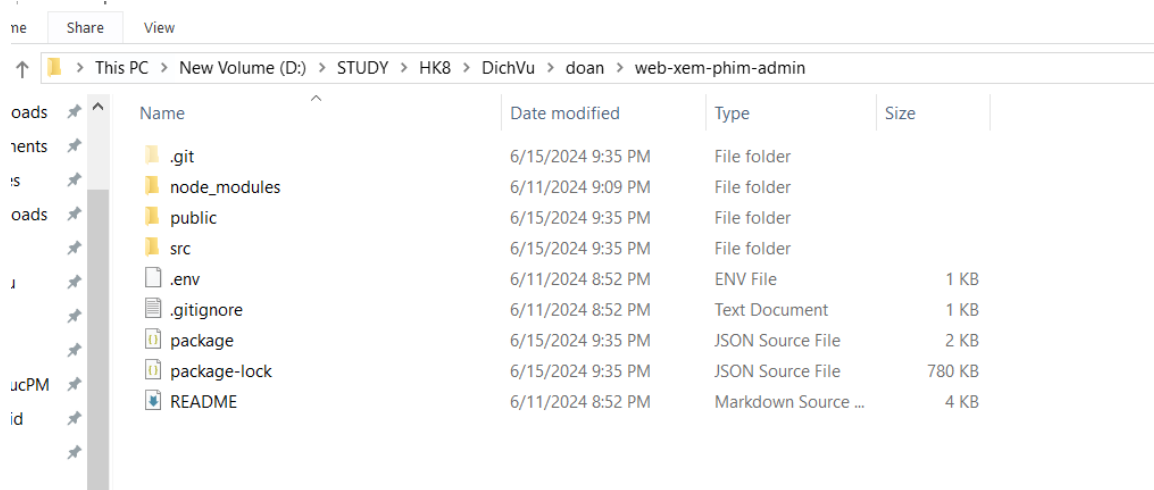
NAME                                     COMPLETIONS   DURATION   AGE
job.batch/ingress-nginx-admission-create 1/1           72s        16d
job.batch/ingress-nginx-admission-patch  1/1           68s        16d
PS C:\Users\GMT>

```

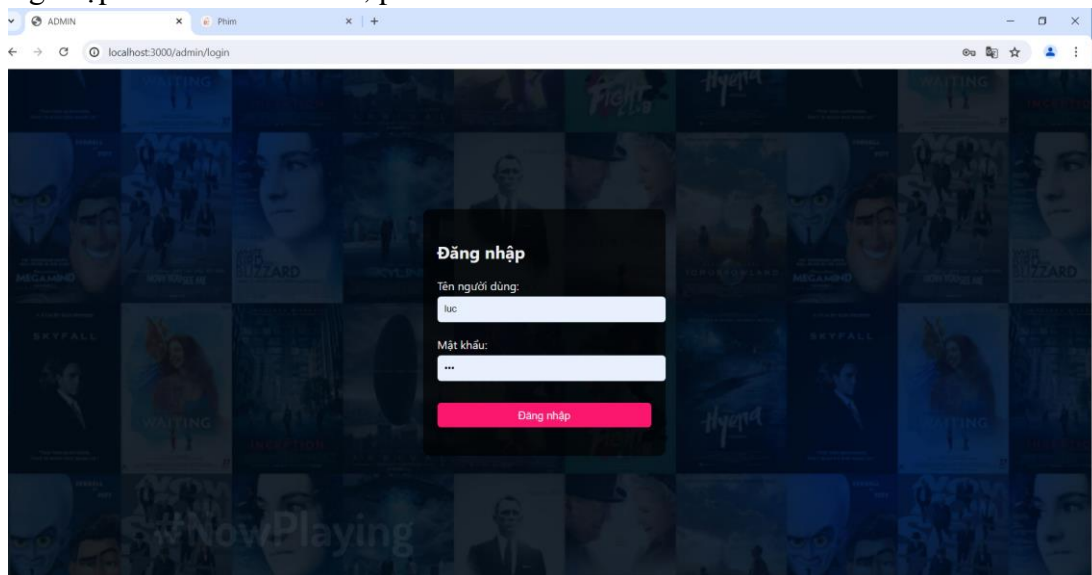
### 4.3. Triển khai cài đặt CMS (Reactjs)

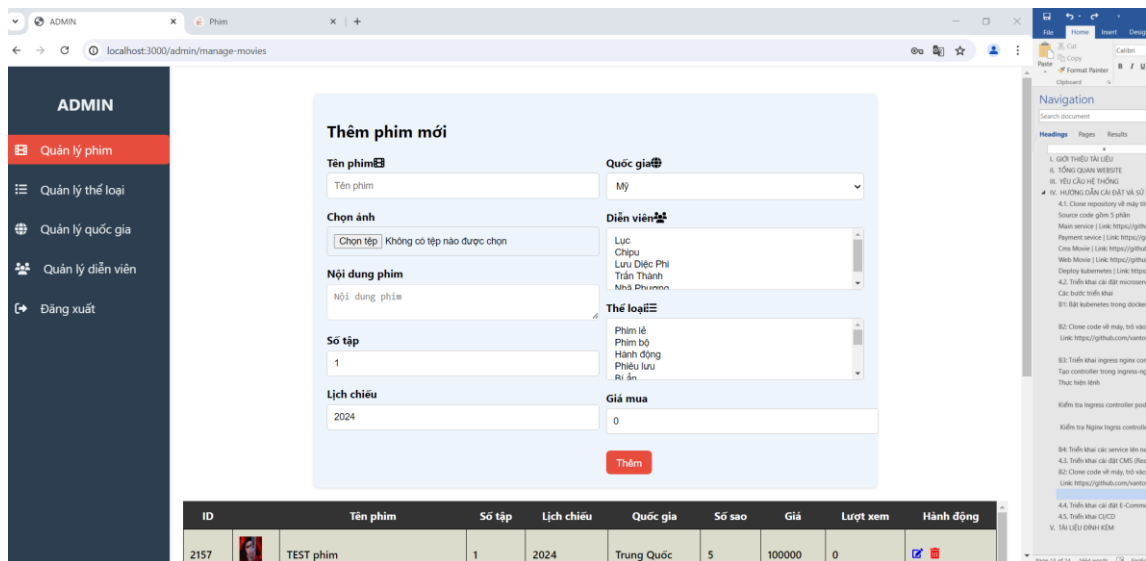
**Clone code về máy, trở vào thư mục web-xem-phim-admin**

Link: : <https://github.com/lucvan02/web-xem-phim-admin>



- Khởi động Visual code chọn project
- + Thực hiện lệnh `npm install` để cài đặt các thư viện
- + Thực hiện lệnh `npm start` chạy chương trình
- + Đăng nhập test: username: luc, pass: 123

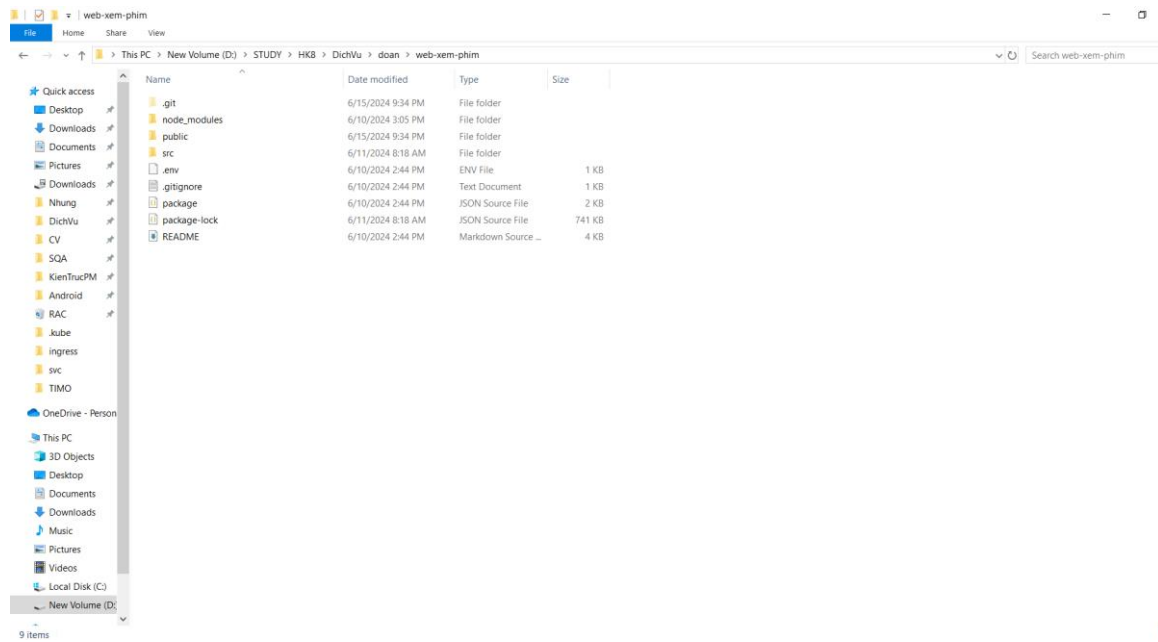




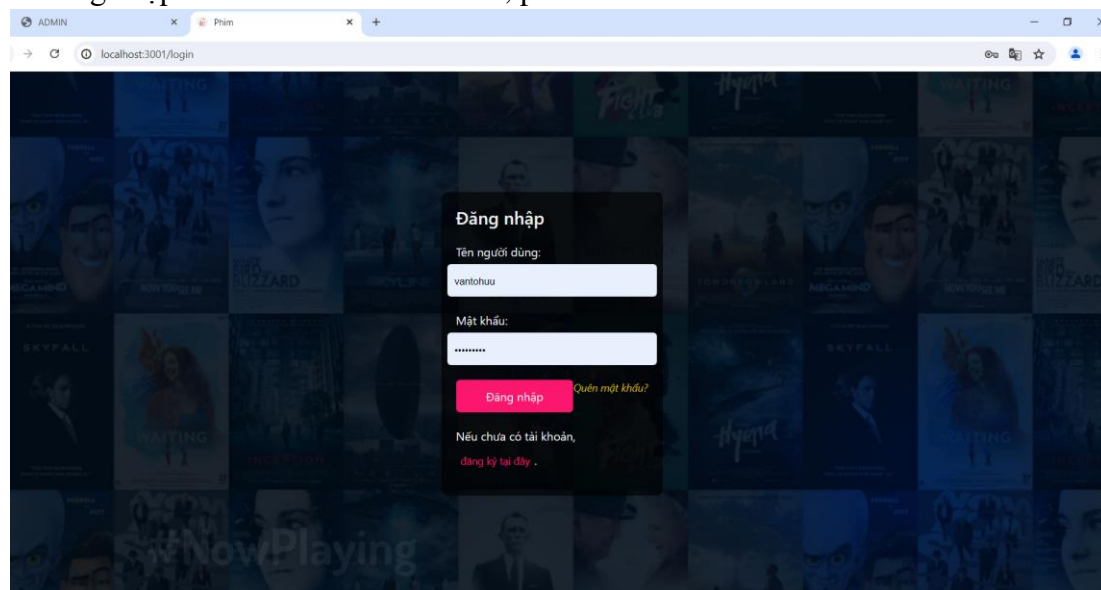
#### 4.4. Triển khai cài đặt giao diện web movie (Reactjs)

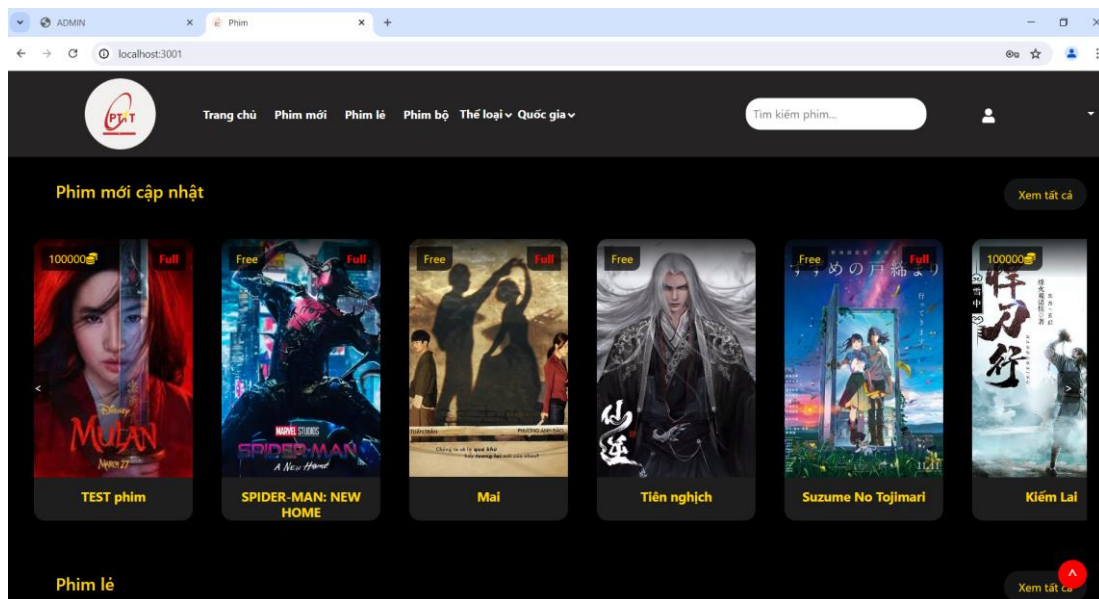
Clone code về máy, trở vào thư mục web-xem-phim

Link: : <https://github.com/lucvan02/web-xem-phim>



- Khởi động Visual code chọn project
- + Thực hiện lệnh npm install để cài đặt các thư viện
- + Thực hiện lệnh npm start chạy chương trình
- + Đăng nhập test: username: vantohuu, pass: 123456789





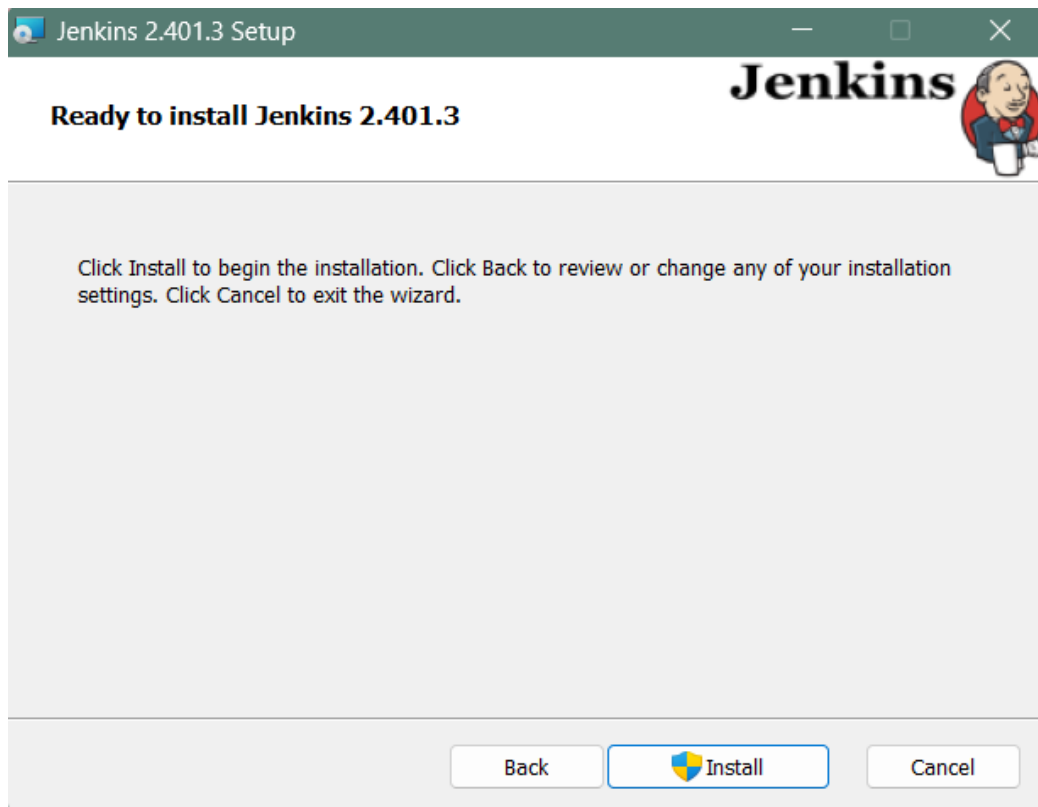
#### 4.5. Triển khai CI/CD

##### Cài đặt Jenkins (Windows)

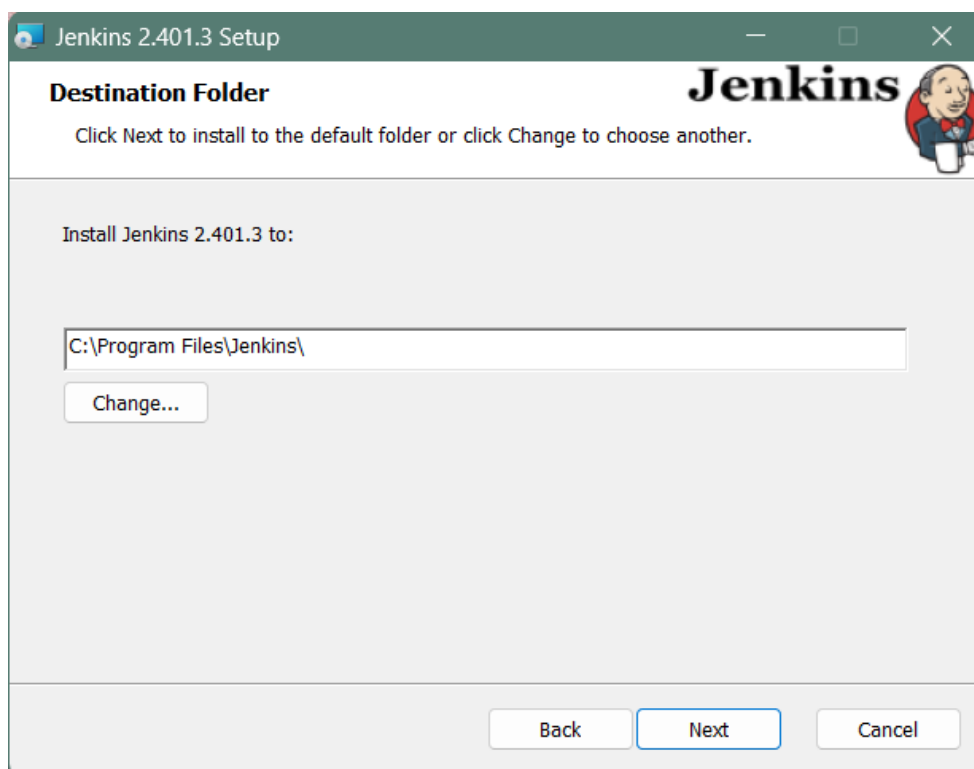
<https://www.jenkins.io/doc/book/installing/windows>

Tiến hành download và cài đặt theo hướng dẫn

B1: Setup Wizard



B2: Chọn đường dẫn cài đặt



B3: Setup Service logon credentials

Jenkins 2.401.3 Setup

## Service Logon Credentials

Enter service credentials for the service.

Jenkins 2.401.3 installs and runs as an independent Windows service. To operate in this manner, you must supply the user account credentials for Jenkins 2.401.3 to run successfully.


**Logon Type:**

☐ Run service as LocalSystem (not recommended)

☒ Run service as local or domain user:

Account:

Password:

 Credentials must be tested to continue

B4: Chọn Port cho Jenkins


Jenkins 2.401.3 Setup

## Port Selection

Choose a port for the service.

Please choose a port.

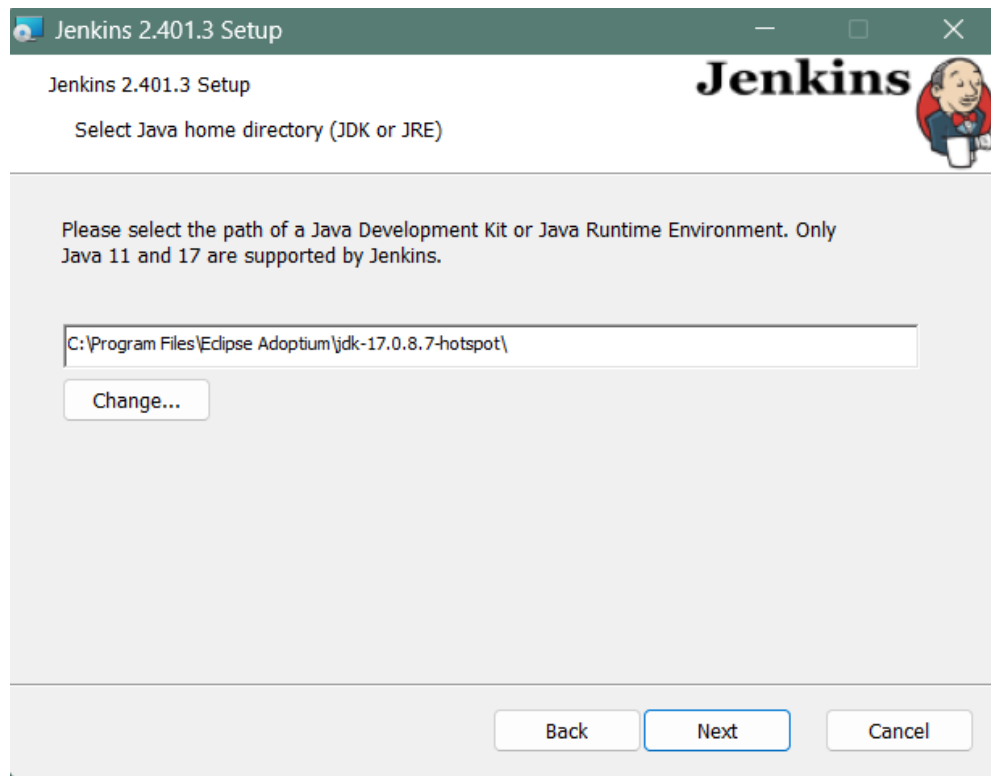
**Port Number (1-65535):**

 Click 'Test Port' button to proceed

It is recommended that you accept the selected default port.

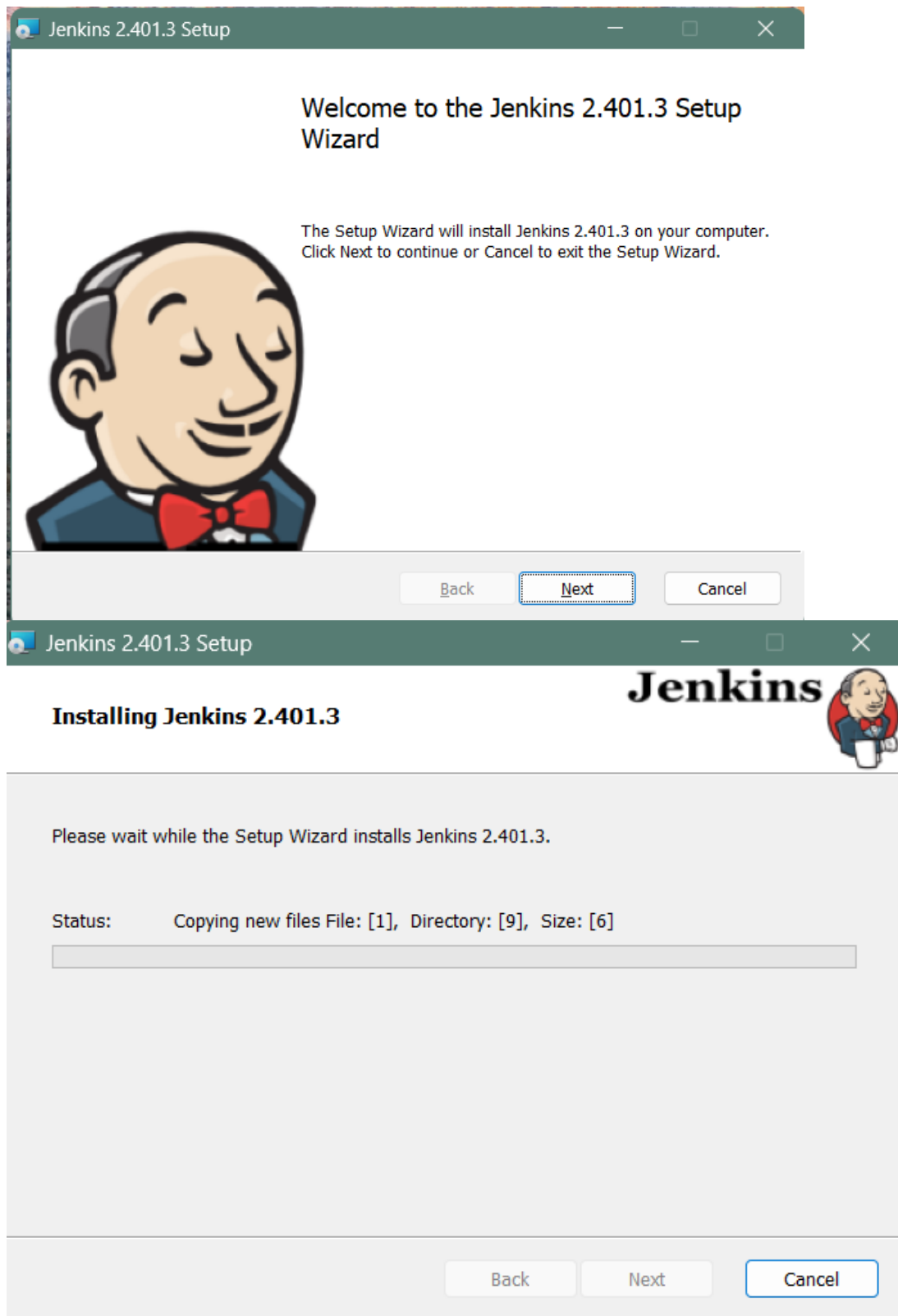
Ở đây mình sử dụng port là 8082

B6: Chọn đường dẫn Java home



B6: Install Jenkins





## Khởi chạy Jenkins

Truy cập <http://localhost:8082> để chạy Jenkins sau khi cài đặt xong

Ở lần chạy đầu tiên ta cần tiến hành unlock Jenkins

Getting Started


# Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (**not sure where to find it?**) and this file on the server:

```
C:\ProgramData\Jenkins\.jenkins\secrets\initialAdminPassword
```

Please copy the password from either location and paste it below.

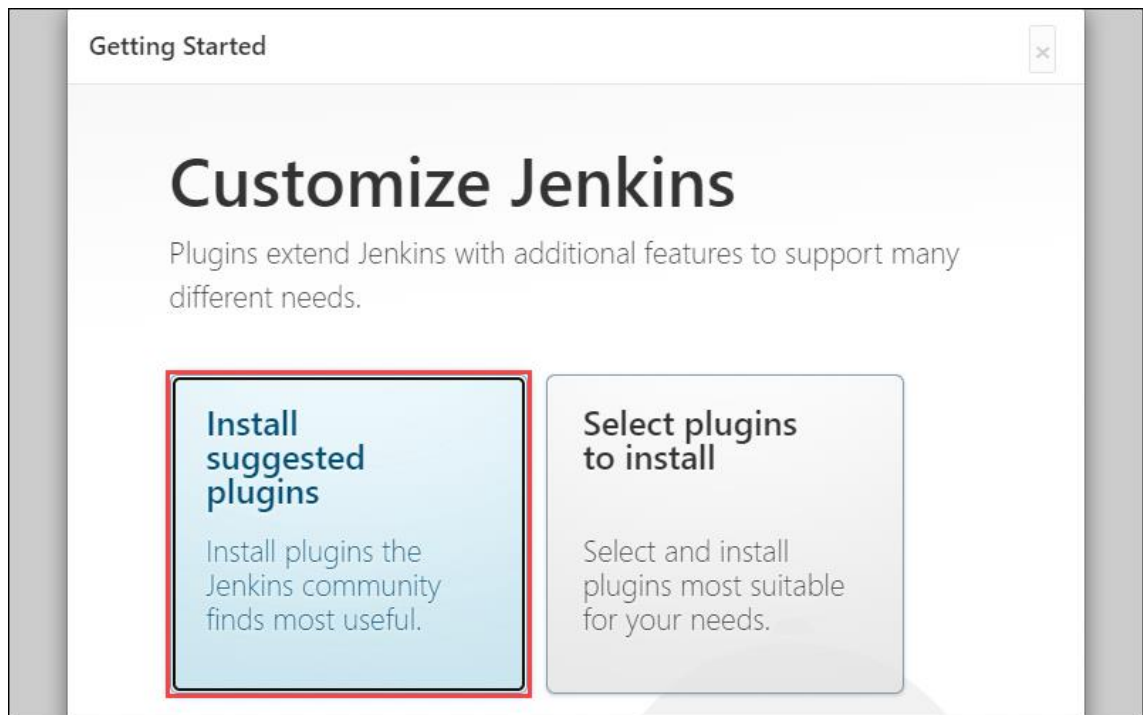
**Administrator password**



Continue

Vào đường dẫn như đường dẫn trong máy của bạn như trong ảnh, để tiến hành lấy password, sau đó dán vào ô Administrator password và nhấn Continue để thực hiện bước kế tiếp.

Chọn Install Suggested plugins hoặc select plugins to install



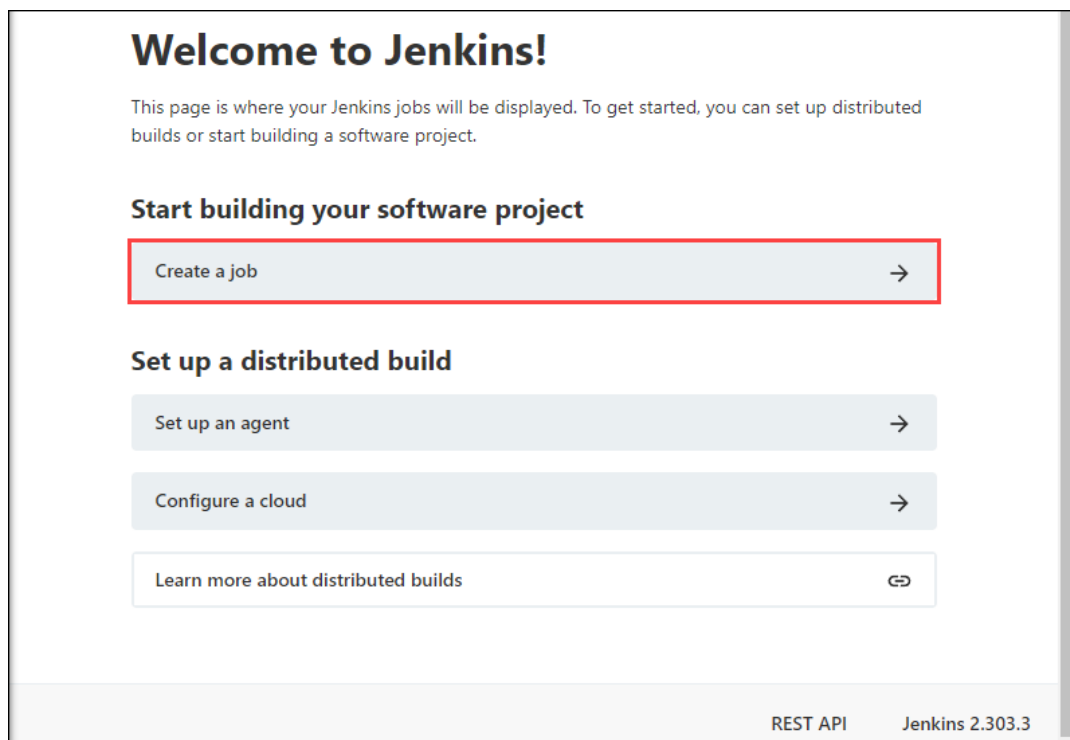
Khởi tạo username và password

The screenshot shows the 'Getting Started' window of Jenkins. The main heading is 'Create First Admin User'. Below it, there is a form with the following fields:

- Username:
- Password:
- Confirm password:
- Full name:
- E-mail address:

The form fields are highlighted with a red border. At the bottom of the window, there is a footer with the text 'Jenkins 2.303.3' on the left, 'Skip and continue as admin' in the center, and a blue button labeled 'Save and Continue' on the right, which is also highlighted with a red border.

Khởi tạo thành công



## Tiến hành cấu hình github webhook

Đưa localhost:8082 ra bên ngoài internet bằng ngrok

Cài đặt ngrok, sau đó thực hiện authentication, sau đó thực hiện lệnh ngrok http 8082

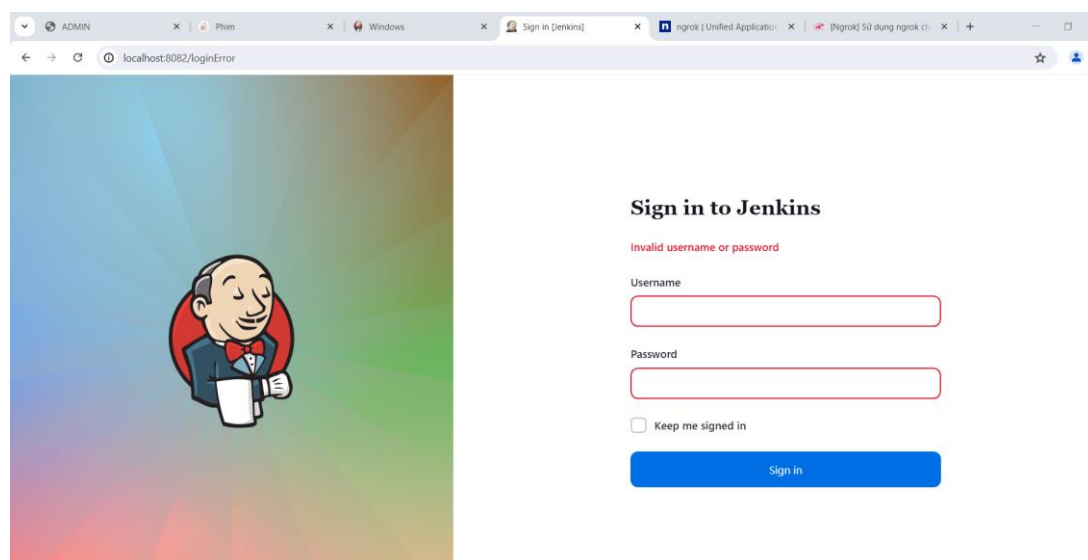
```
Command Prompt - ngrok http 8082
ngrok
New guides https://ngrok.com/docs/guides/site-to-site-apis/
Session Status      online
Account             vantohuu1234567@gmail.com (Plan: Free)
Version             3.11.0
Region              Asia Pacific (ap)
Latency             40ms
Web Interface       http://127.0.0.1:4040
Forwarding           https://5843-1-53-97-88.ngrok-free.app -> http://localhost:8082
Connections
  ttl    opn    rt1    rt5    p50    p90
    0     0     0.00  0.00  0.00  0.00
```

Vào webhook github tạo một webhook

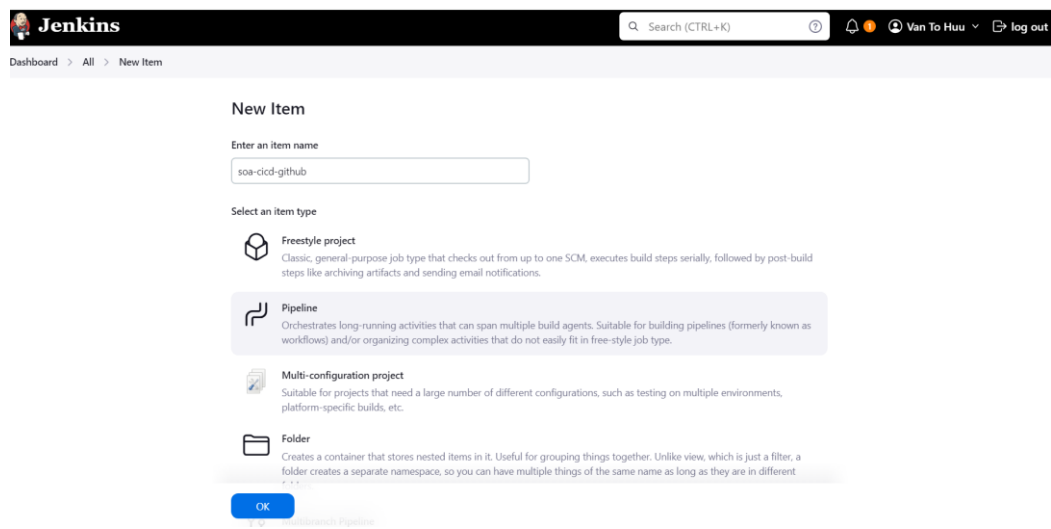


```
.gitignore pod.yaml Jenkinsfile application.yml mvnw.cmd
1 pipeline {
2   environment {
3     dockerimageName = "hvu2412002/soa-backend:v01"
4     dockerImage = ""
5   }
6   agent any
7   stages {
8     stage('Checkout Source') {
9       steps {
10        cleanWs()
11        git 'https://github.com/vantohuu/soa-backend.git'
12      }
13    }
14    stage('Build image') {
15      steps {
16        script {
17          dockerImage = docker.build dockerimageName
18        }
19      }
20    }
21  }
22 }
23 }
24 }
25 }
```

Đăng nhập



Chọn pipeline và gõ tên



**Jenkins** Search (CTRL+K) Van To Huu log out

Dashboard > All > New Item

### New Item

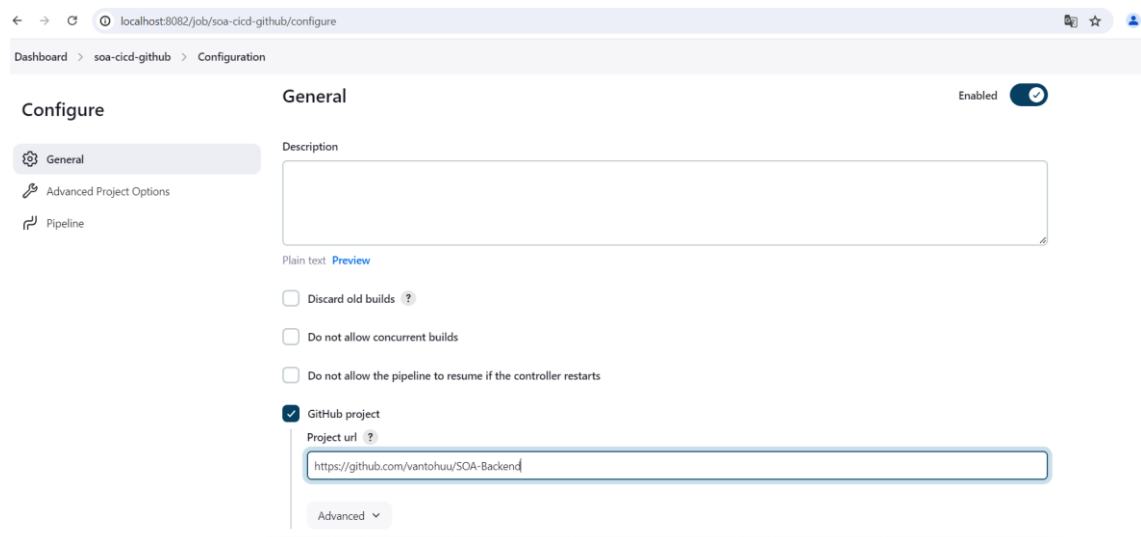
Enter an item name  
soa-cicd-github

Select an item type

- Freestyle project**  
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**  
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different namespaces.

OK Cancel

(tùy chọn) Chọn mô tả là github project và dán đường dẫn kink git hub vào



localhost:8082/job/soa-cicd-github/configure

Dashboard > soa-cicd-github > Configuration

### Configure

- General
- Advanced Project Options
- Pipeline

### General

Enabled

Description

Plain text [Preview](#)

☐ Discard old builds ?

☐ Do not allow concurrent builds

☐ Do not allow the pipeline to resume if the controller restarts

☒ GitHub project

Project url ?  
https://github.com/vantohuu/SOA-Backend

Advanced

Tick chọn Git hook trigger for GIT Scm polling để nhận thông tin từ github webhook

☐ Pipeline speed/durability override ?  
☐ Preserve stashes from completed builds ?  
☐ This project is parameterized ?  
☐ Throttle builds ?

**Build Triggers**

☐ Build after other projects are built ?  
☐ Build periodically ?  
☒ GitHub hook trigger for GITScm polling ?  
☐ Poll SCM ?  
☐ Quiet period ?  
☐ Trigger builds remotely (e.g., from scripts) ?

---

**Advanced Project Options**

Advanced ▾

Tùy chọn pipeline trong SCM để chạy code jenkinsfile từ github và cấu hình các thông tin cho chúng

Pipeline script from SCM ▾

SCM ?

Git ▾ ?

**Repositories ?**

Repository URL ?  
 https://github.com/vantohuu/SOA-Backend

Credentials ?  
 vantohuu/\*\*\*\*\* (github credentials) ▾

+ Add ▾

Advanced ▾

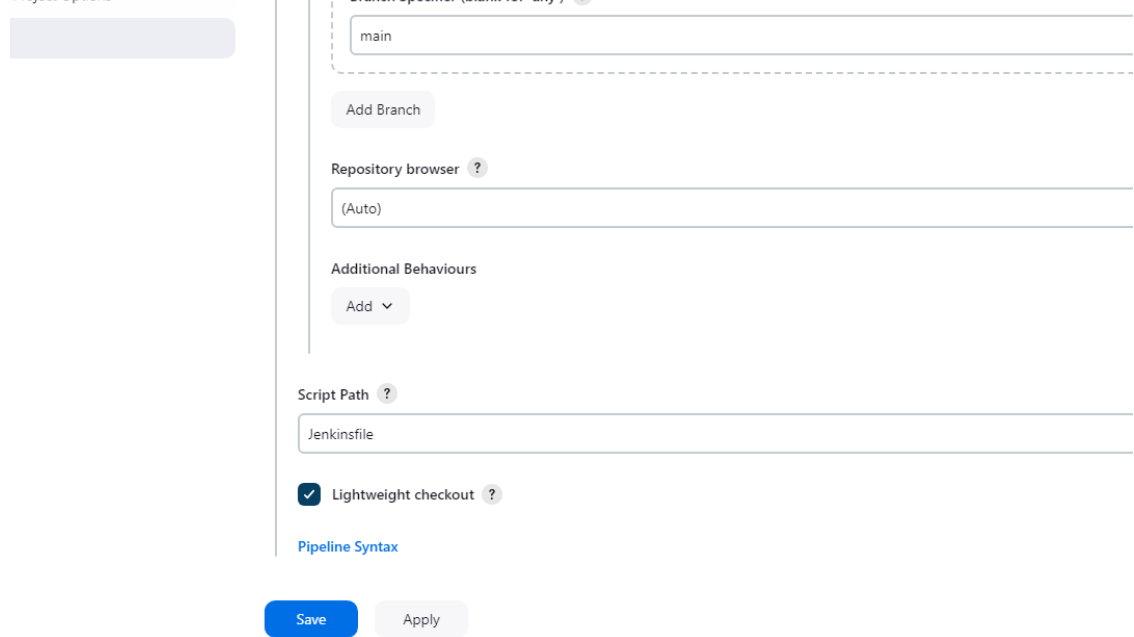
Add Repository

**Branches to build ?**

Branch Specifier (blank for 'any') ?  
 /main



Nhấn save để lưu



The image shows the Jenkins Pipeline Configuration page. On the left, there is a sidebar with a 'Project Options' section containing a 'Project name' field. The main configuration area on the right includes a 'Branch specifier (blank for any)' field with 'main' entered, an 'Add Branch' button, a 'Repository browser' dropdown set to '(Auto)', an 'Additional Behaviours' section with an 'Add' button, a 'Script Path' field set to 'Jenkinsfile', and a checked 'Lightweight checkout' checkbox. At the bottom, there are 'Save' and 'Apply' buttons. A 'Pipeline Syntax' link is also visible.

## Thực hiện cicd

Tiến hành push code lên git



The image is a screenshot of a terminal window. The title bar shows 'Terminal' and 'Local'. The terminal content shows the following commands and output:


```
View build details: docker-desktop://dashboard/build/default/default/8rlcx82a3xyal5oe32eldn/

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS D:\STUDY\HK8\DichVu\cicd\SOA-Backend> git add .
PS D:\STUDY\HK8\DichVu\cicd\SOA-Backend> git commit -m "test"
[master 796cb39] test
 1 file changed, 2 insertions(+), 2 deletions(-)
PS D:\STUDY\HK8\DichVu\cicd\SOA-Backend> git push
```

At the bottom of the terminal, there is a prompt 'SOA-Backend >' followed by a blue icon and the text 'pom.xml'.

Jenkins nhận thay đổi và tự động build

→ ↺ ⓘ localhost:8082/job/soa-cicd-github/3/pipeline-graph/

 **Jenkins**

Dashboard > soa-cicd-github > #3 > Pipeline Overview

Pipeline

```
graph LR; Start((Start)) --> CheckoutSCM[Checkout SCM]; CheckoutSCM --> CheckoutSource[Checkout Source]; CheckoutSource --> BuildImage[Build image]; BuildImage --> End((End));
```