

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN MÔN HỌC**

**ĐIỆN TOÁN ĐÁM MÂY**

**ĐỀ TÀI**

**XÂY DỰNG ỨNG DỤNG ĐƠN GIẢN VÀ**

**DEPLOY LÊN KUBERNETES**

**SVTH:** **MSSV:**

**Nguyễn Huỳnh Phúc 17110350**

**Phạm Văn Minh Tân** **17110364**

**GVHD: TS. Huỳnh Xuân Phụng**

**Tp. Hồ Chí Minh, tháng 1 năm 2021**

**MỤC LỤC**

[BẢNG PHÂN CÔNG NHIỆM VỤ, CHỨC NĂNG 4](#_Toc60792312)

[CHƯƠNG 1: GIỚI THIỆU CHUNG 5](#_Toc60792313)

[1.1. Lý do chọn đề tài 5](#_Toc60792314)

[1.2. Mục tiêu của đề tài 5](#_Toc60792315)

[1.3. Bố cục của báo cáo 5](#_Toc60792316)

[CHƯƠNG 2: KHÁI NIỆM 6](#_Toc60792317)

[2.1. Điện toán đám mây 6](#_Toc60792318)

[2.2. Docker và container trong docker 6](#_Toc60792319)

[2.2.1. Docker 6](#_Toc60792320)

[2.2.2. Container trong docker 6](#_Toc60792321)

[2.3. Kubernetes 7](#_Toc60792322)

[CHƯƠNG 3: XÂY DỰNG ỨNG DỤNG VÀ DEPLOY LÊN KUBERNETES 9](#_Toc60792323)

[3.1. Công nghệ và phần mềm sử dụng 9](#_Toc60792324)

[3.2. Xây dựng ứng dụng 9](#_Toc60792325)

[3.3. Deploy ứng dụng lên Kubernetes 13](#_Toc60792326)

[3.3.1. Xây dựng docker image cho back-end 13](#_Toc60792327)

[3.3.2. Xây dựng docker image cho front-end 14](#_Toc60792328)

[3.3.3. Deploy ứng dụng lên Kubernetes 16](#_Toc60792329)

[CHƯƠNG 4: KẾT LUẬN 20](#_Toc60792330)

[4.1. Kết quả đạt được 20](#_Toc60792331)

[4.2. Các hạn chế 20](#_Toc60792332)

[4.3. Hướng phát triển 20](#_Toc60792333)

[TÀI LIỆU THAM KHẢO 21](#_Toc60792334)

# BẢNG PHÂN CÔNG NHIỆM VỤ, CHỨC NĂNG

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Công việc | Người thực hiện | Đánh giá |
| 1 | Phân tích nghiệp vụ | Phạm Văn Minh Tân | Hoàn thành |
| 2 | Lập trình front-end | Nguyễn Huỳnh Phúc | Hoàn thành |
| 3 | Lập trình back-end | Phạm Văn Minh Tân | Hoàn thành |
| 4 | Deploy lên Kubernetes | Nguyễn Huỳnh Phúc | Hoàn thành |

# CHƯƠNG 1: GIỚI THIỆU CHUNG

## 1.1. Lý do chọn đề tài

Từ nhu cầu thực tiễn, các tổ chức, doanh nghiệp và cá nhân luôn muốn hạn chế chi phí cho các tài nguyên công nghệ thông tin, đặc biệt là các máy chủ. Điện toán đám mây là giải pháp để giải quyết vấn đề trên. Rất nhiều dịch vụ điện toán đám mây yêu cầu nền tảng dựa trên Kubernetes (Kubernetes-based platform).

Nhóm quyết định tìm hiểu về Kubernetes với đề tài: **“Xây dựng ứng dụng đơn giản và deploy lên Kubernetes”**.

## 1.2. Mục tiêu của đề tài

- Xây dựng ứng dụng: trang web sinh ngẫu nhiên đơn giản

- Deploy ứng dụng đã xây dựng lên Kubernetes

## 1.3. Bố cục của báo cáo

Chương 1: Giới thiệu chung

Chương 2: Khái niệm

Chương 3: Xây dựng ứng dụng và deploy lên kubernetes

Chương 4: Kết luận

# CHƯƠNG 2: KHÁI NIỆM

## 2.1. Điện toán đám mây

Điện toán đám mây (tiếng Anh: cloud computing), còn gọi là điện toán máy chủ ảo, là mô hình điện toán sử dụng các công nghệ máy tính và phát triển dựa vào mạng Internet. Thuật ngữ "đám mây" ở đây là lối nói ẩn dụ chỉ mạng Internet (dựa vào cách được bố trí của nó trong sơ đồ mạng máy tính) và như sự liên tưởng về độ phức tạp của các cơ sở hạ tầng chứa trong nó. Ở mô hình điện toán này, mọi khả năng liên quan đến công nghệ thông tin đều được cung cấp dưới dạng các "dịch vụ", cho phép người sử dụng truy cập các dịch vụ công nghệ từ một nhà cung cấp nào đó "trong đám mây" mà không cần phải có các kiến thức, kinh nghiệm về công nghệ đó, cũng như không cần quan tâm đến các cơ sở hạ tầng phục vụ công nghệ đó. Theo tổ chức IEEE "Nó là hình mẫu trong đó thông tin được lưu trữ thường trực tại các máy chủ trên Internet và chỉ được được lưu trữ tạm thời ở các máy khách, bao gồm máy tính cá nhân, trung tâm giải trí, máy tính trong doanh nghiệp, các phương tiện máy tính cầm tay,...". Điện toán đám mây là khái niệm tổng thể bao gồm cả các khái niệm như phần mềm dịch vụ, Web 2.0 và các vấn đề khác xuất hiện gần đây, các xu hướng công nghệ nổi bật, trong đó đề tài chủ yếu của nó là vấn đề dựa vào Internet để đáp ứng những nhu cầu điện toán của người dùng.

## 2.2. Docker và container trong docker

### 2.2.1. Docker

Docker là một nền tảng để cung cấp cách để building, deploying và running ứng dụng dễ dàng hơn bằng cách sử dụng các containers (trên nền tảng ảo hóa). Ban đầu viết bằng Python, hiện tại đã chuyển sang Golang.

### 2.2.2. Container trong docker

Các containers cho phép lập trình viên đóng gói một ứng dụng với tất cả các phần cần thiết, chẳng hạn như thư viện và các phụ thuộc khác, và gói tất cả ra dưới dạng một package.

Bằng cách đó, nhờ vào container, ứng dụng sẽ chạy trên mọi máy Linux khác bất kể mọi cài đặt tùy chỉnh mà máy có thể có khác với máy được sử dụng để viết code.

Theo một cách nào đó, Docker khá giống virtual machine. Nhưng tại sao Docker lại phát triển, phổ biến nhanh chóng? Đây là những nguyên nhân:

- Tính dễ ứng dụng: Docker rất dễ cho mọi người sử dụng từ lập trình viên, sys admin… nó tận dụng lợi thế của container để build, test nhanh chóng. Có thể đóng gói ứng dụng trên laptop của họ và chạy trên public cloud, private cloud… Câu thần chú là “Build once, run anywhere”.

- Tốc độ: Docker container rất nhẹ và nhanh, bạn có thể tạo và chạy docker container trong vài giây.

- Môi trường chạy và khả năng mở rộng: Bạn có thể chia nhỏ những chức năng của ứng dụng thành các container riêng lẻ. Ví dụng Database chạy trên một container và Redis cache có thể chạy trên một container khác trong khi ứng dụng Node.js lại chạy trên một cái khác nữa. Với Docker, rất dễ để liên kết các container với nhau để tạo thành một ứng dụng, làm cho nó dễ dàng scale, update các thành phần độc lập với nhau.

Với xu hướng dịch chuyển sang microservices của các hệ thống lớn, Docker đang làm một thành phần cực kỳ quan trọng, làm cho nó trở thành một phần của nhiều công cụ DevOps. Hiện tại thế giới bắt đầu sử dụng thêm một công cụ quản lý container tiên tiến khác là Kubernetes.

## 2.3. Kubernetes

Kubernetes, hoặc k8s là một nền tảng mã nguồn mở tự động hoá việc quản lý, scaling và triển khai ứng dụng dưới dạng container hay còn gọi là Container orchestration engine. Nó loại bỏ rất nhiều các quy trình thủ công liên quan đến việc triển khai và mở rộng các containerized applications.

Kubernetes orchestration cho phép bạn xây dựng các dịch vụ ứng dụng mở rộng nhiều containers. Nó lên lịch các containers đó trên một cụm, mở rộng các containers và quản lý tình trạng của các containers theo thời gian.

Các ứng dụng production thực tế mở rộng nhiều containers. Các containers đó phải được triển khai trên nhiều server hosts. Kubernetes cung cấp khả năng phối hợp và quản lý cần thiết để triển khai các containers theo quy mô cho các workloads đó.

Kubernetes ban đầu được phát triển và thiết kế bởi các kỹ sư tại Google. Đây cũng là công nghệ đằng sau các dịch vụ đám mây của Google. Google đã và đang tạo ra hơn 2 tỷ container deployments mỗi tuần và tất cả đều được hỗ trợ bởi nền tảng nội bộ: Borg.

# CHƯƠNG 3: XÂY DỰNG ỨNG DỤNG VÀ DEPLOY LÊN KUBERNETES

## 3.1. Công nghệ và phần mềm sử dụng

- Visual studio 2019 v16.6.5

- Visual studio Code 1.52.1

- ASP.NET Core 3.1

- Angular 10

- Docker Desktop 2.4.0.0

- Kubernetes v1.18.8

## 3.2. Xây dựng ứng dụng

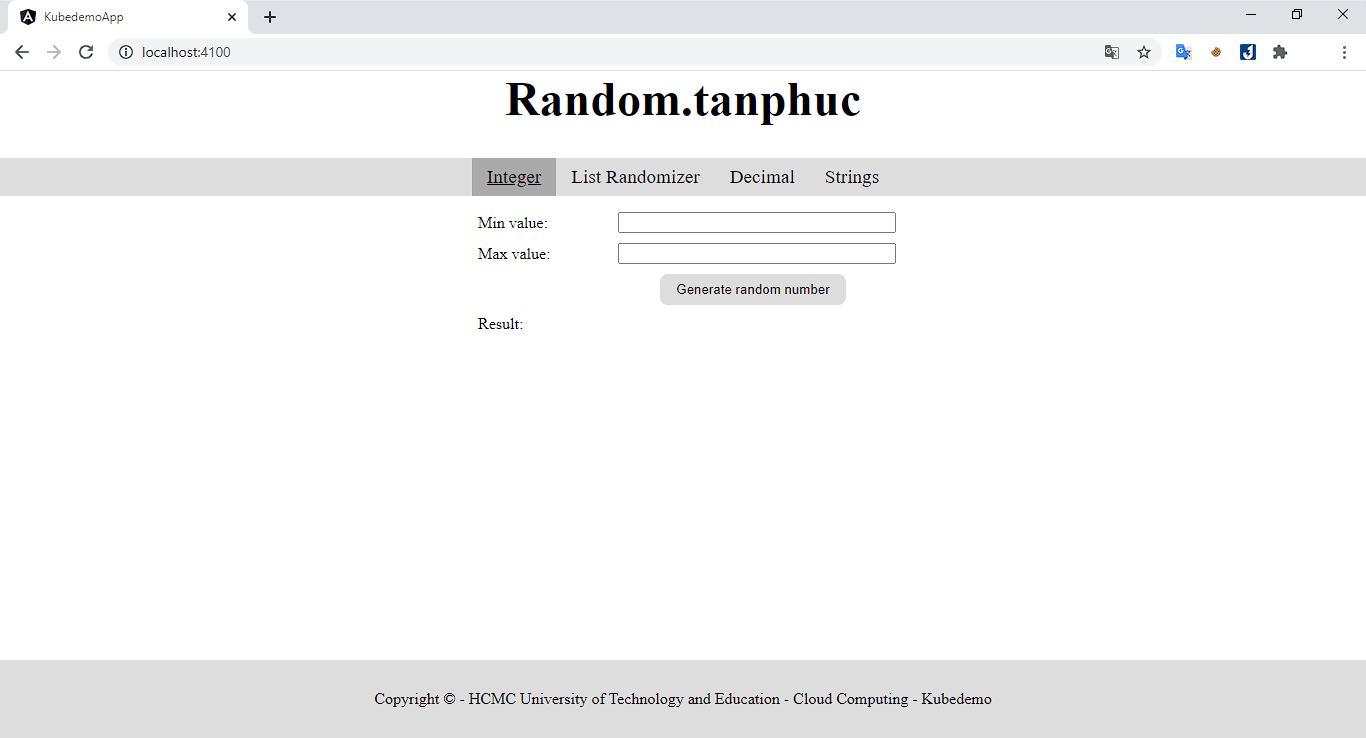
*Layout*

- Tiêu đề trang web

- Menu chức năng

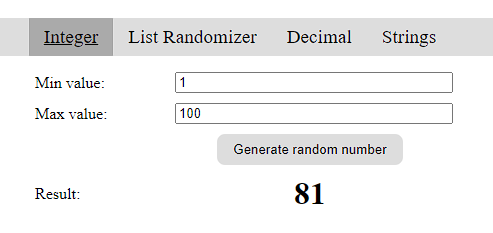
- Nội dung chức năng

- Footer trang web

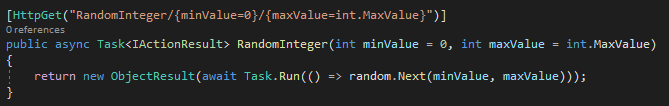


*Chức năng sinh ngẫu nhiên số nguyên trong khoảng nhập vào*

- Giao diện: nhãn, trường nhập giá trị khoảng, nút sinh ngẫu nhiên, nhãn kết quả và nhãn hiển thị kết quả



- Cài đặt xử lí

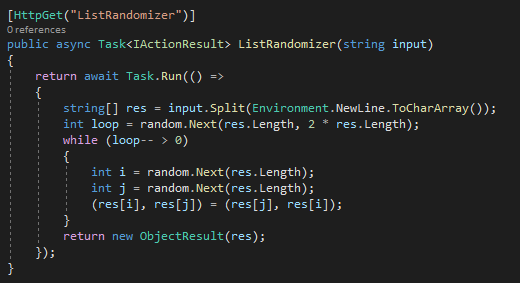


*Chức năng sinh danh sách mới với thứ tự ngẫu nhiên từ danh sách nhập vào*

- Giao diện: nhãn, trường nhập danh sách, nút sinh ngẫu nhiên, nhãn kết quả và nhãn hiển thị kết quả

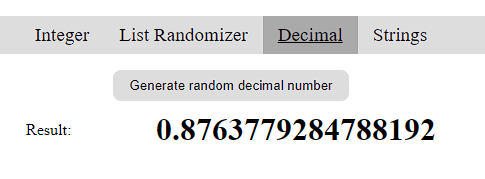


- Cài đặt xử lí

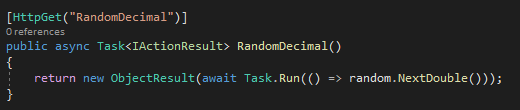


*Chức năng sinh số thực ngẫu nhiên*

- Giao diện: nút sinh ngẫu nhiên, nhãn kết quả và nhãn hiển thị kết quả

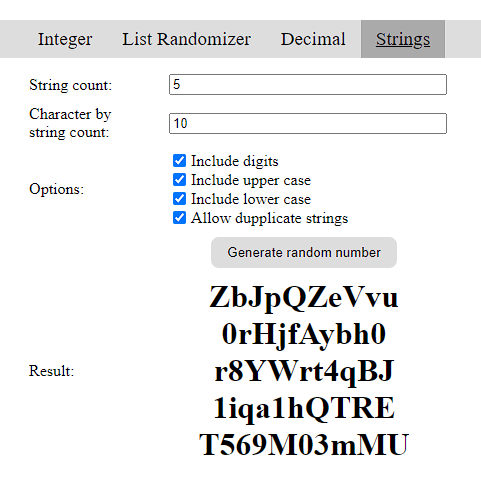


- Cài đặt xử lí

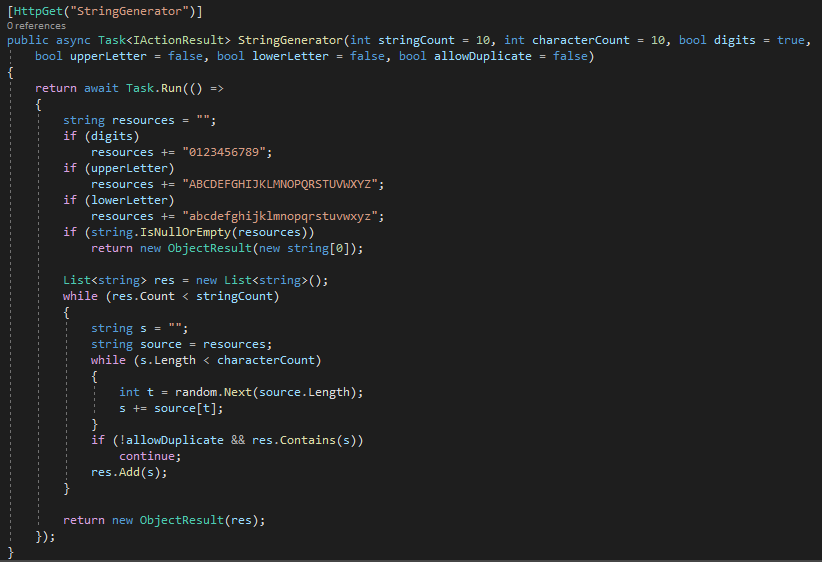


*Chức năng danh sách ngẫu nhiên (mỗi phần tử của danh sách là một chuỗi các ký tự)*

- Giao diện: nhãn, trường nhập số lượng phần tử và ký tự mỗi phần tử, nhãn và checkbox các lựa chọn cho phép chứa chữ số, ký tự in thường, ký tự in hoa, cho phép các phần tử giống nhau, nút sinh ngẫu nhiên, nhãn kết quả và nhãn hiển thị kết quả



- Cài đặt xử lí



## 3.3. Deploy ứng dụng lên Kubernetes

### 3.3.1. Xây dựng docker image cho back-end

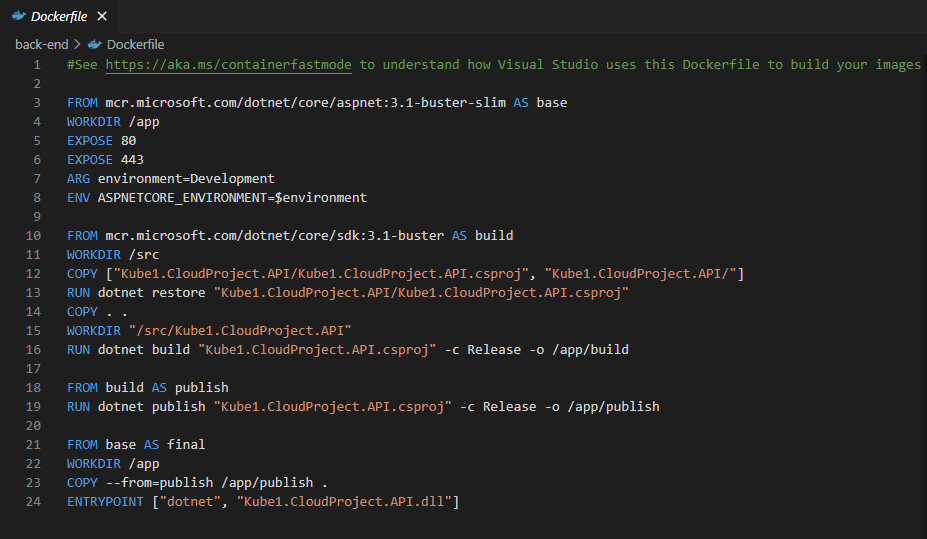
- Xây dựng Dockerfile:

+ “Kube1.CloudProject.API” là tên project

+ “Kube1.CloudProject.API/Kube1.CloudProject.API.csproj” là đường dẫn tới file project ASP.NET Core từ vị trí Dockerfile

+ “Kube1.CloudProject.API/” là thư mục được tạo

+ “Kube1.CloudProject.API.dll” là điểm khởi chạy ứng dụng

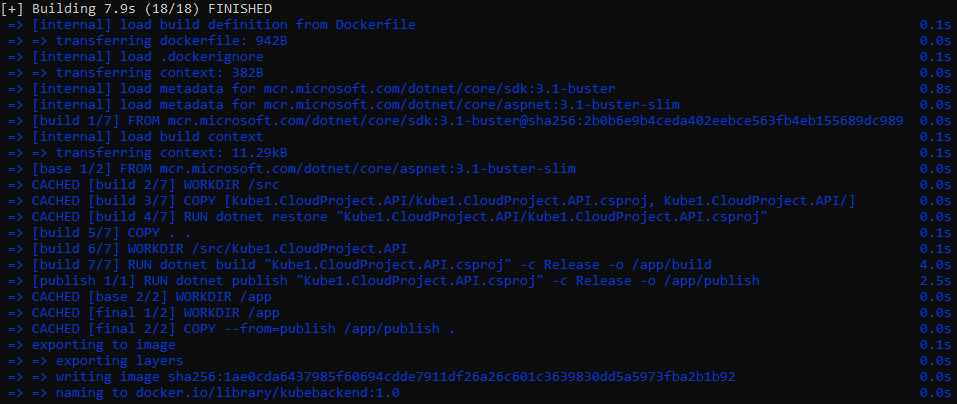


- Xây dựng docker image

+ Từ vị trí Dockerfile chạy lệnh:

docker build -t kubebackend:1.0 . (có dấu chấm cuối)

+ Kết quả

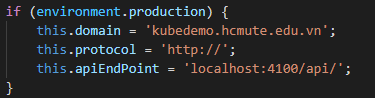


+ Docker image được tạo trên Docker Desktop



### 3.3.2. Xây dựng docker image cho front-end

- Cài đặt api endpoint cho môi trường production: cổng kết nối 4100 là cổng kết nối công khai ra bên ngoài của cụm được deploy, sẽ được trình bày khi xây dựng file deploy.yaml



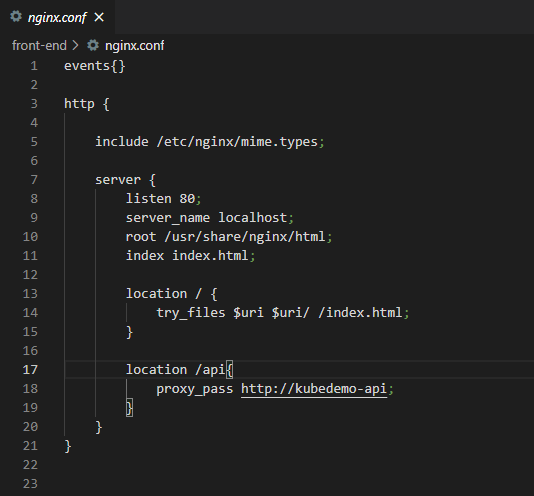
- Xây dựng nginx.conf file

+ Cài đặt thư mục gốc

+ Cài đặt file index

+ Điều hướng truy cập dựa vào đường dẫn

+ “kubedemo-api” là tên của service trong cụm được deploy bằng kubernetes, sẽ được trình bày khi xây dựng file deploy.yaml

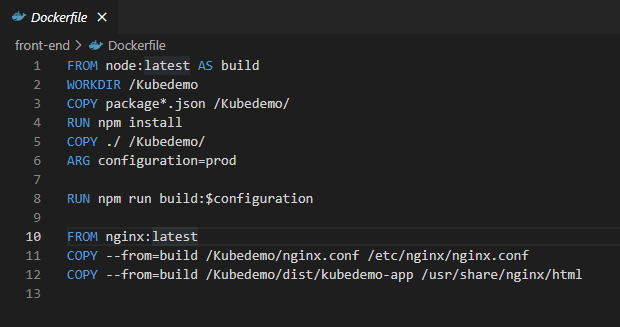


- Xây dựng Dockerfile:

+ Được đặt ở thư mục frone-end

+ Bao gồm cài đặt của nginx từ file nginx.conf

+ Bỏ tất cả nội dung front-end vào thư mục gốc của nginx

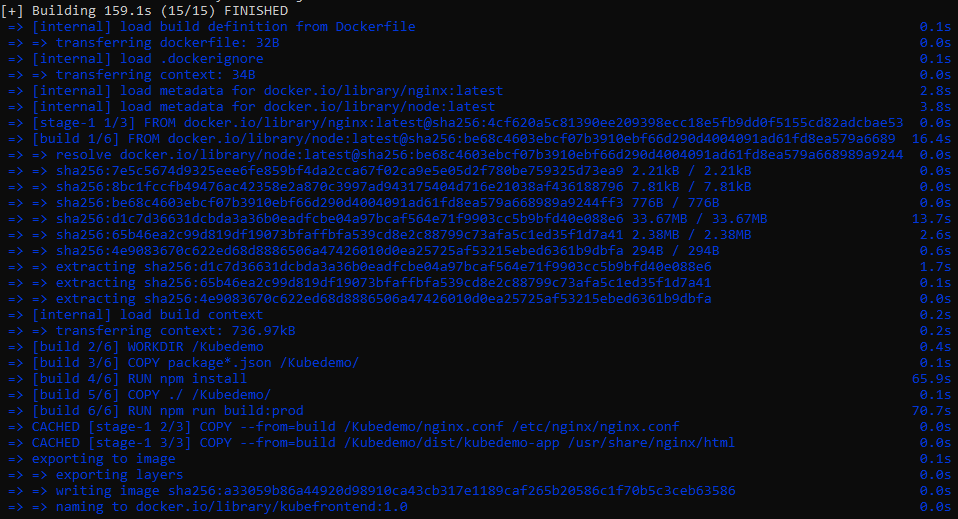


- Xây dựng docker image

+ Từ vị trí Dockerfile chạy lệnh:

docker build -t kubefrontend:1.0 . (có dấu chấm cuối)

+ Kết quả



+ Docker image được tạo trên Docker Desktop



### 3.3.3. Deploy ứng dụng lên Kubernetes

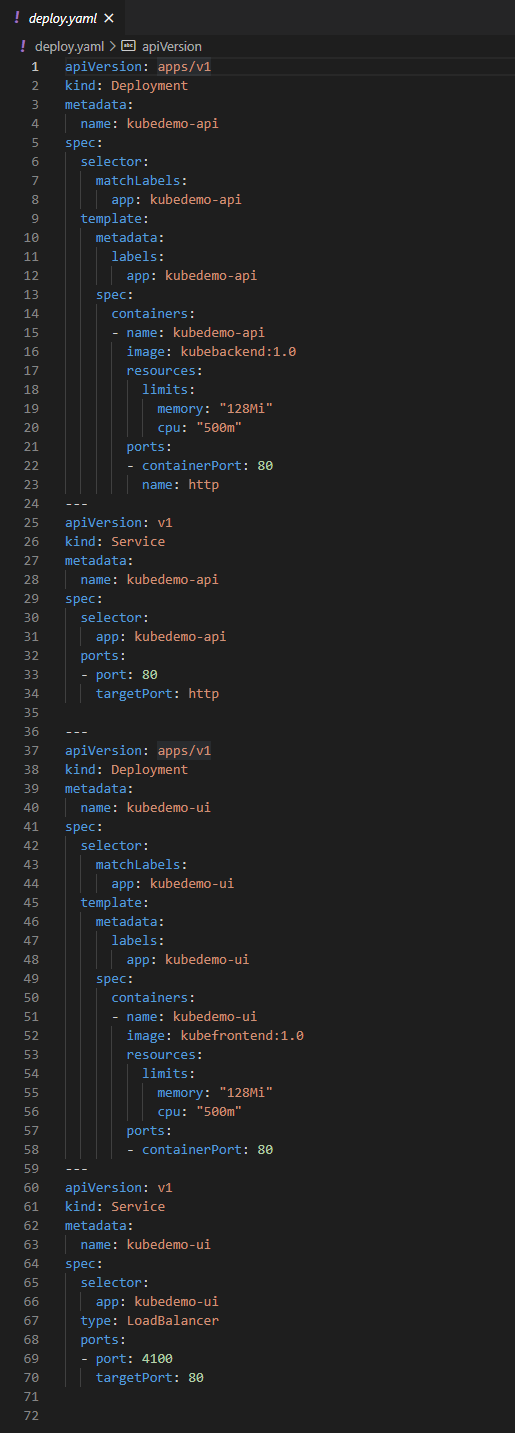
- Xây dựng deploy.yaml file

+ Deployment backend: tên kubedemo-api, sử dụng image kubebackend:1.0, sử dụng cổng kết nối 80

+ Service backend: tên kubedemo-api, trỏ tới deployment backend với cổng kết nối 80

+ Deployment frontend: tên kubedemo-ui, sử dụng image kubefrontend:1.0, sử dụng cổng kết nối 80

+ Service frontend: tên kubedemo-ui, trỏ tới deployment frontend với cổng kết nối 80, loại là **LoadBalancer** với cổng kết nối công khai là 4100

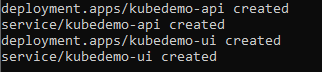


- Thực thi deploy ứng dụng:

+ Từ vị trí deploy.yaml file chạy lệnh:

kubectl apply -f deploy.yaml

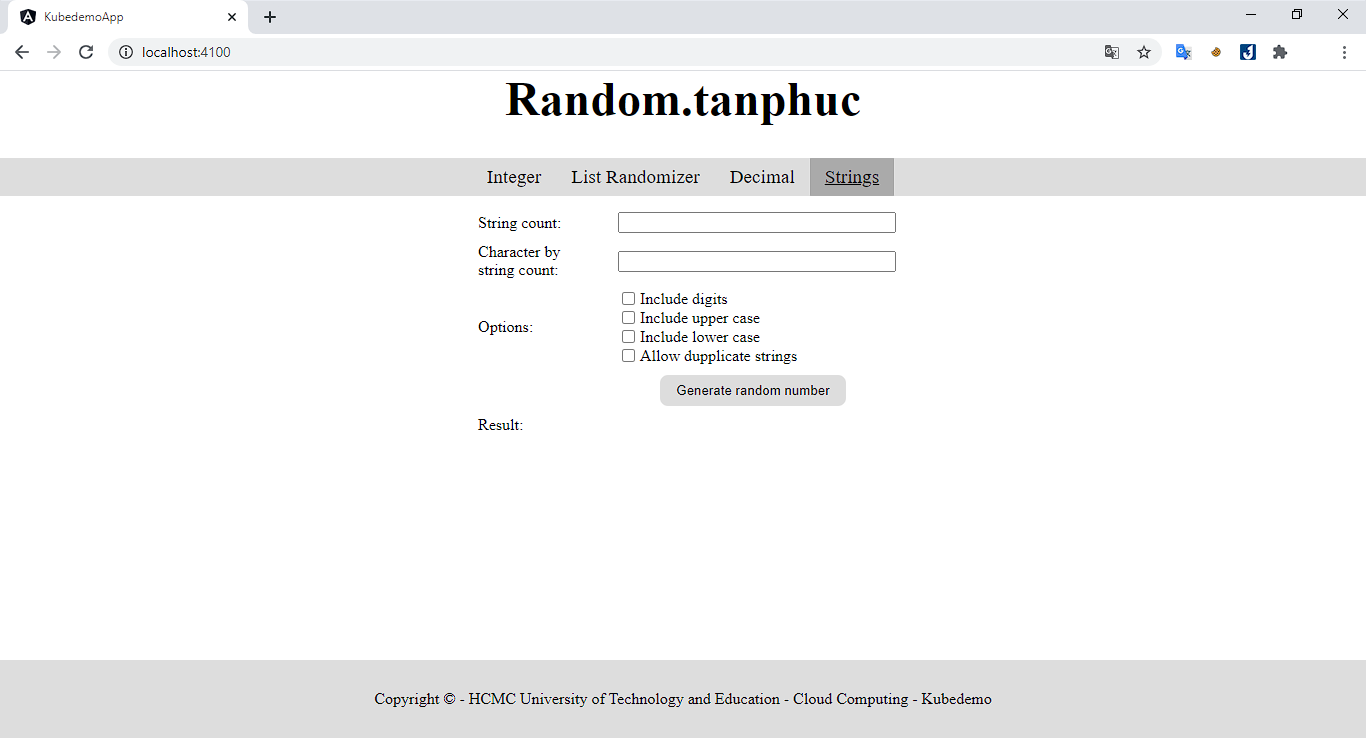
+ Kết quả



+ Các container đã chạy trên Docker Desktop



+ Trang web có thể truy cập thông qua cổng kết nối 4100



# CHƯƠNG 4: KẾT LUẬN

## 4.1. Kết quả đạt được

- Xây dựng được ứng dụng: trang web sinh ngẫu nhiên đơn giản

- Deploy được ứng dụng đã xây dựng lên Kubernetes

## 4.2. Các hạn chế

Nhóm thực hiện đề nhằm tìm hiểu, nâng cao kiến thức không tránh khỏi những sai sót, hạn chế, bất cập. Một số hạn chế như:

* Ứng dụng còn tương đối đơn giản
* Đề tài chỉ thực hiện trên localhost

## 4.3. Hướng phát triển

Áp dụng để thực hiện các ứng dụng thực tế và deploy lên các dịch vụ điện toán đám mây.

# TÀI LIỆU THAM KHẢO

1. [Điện toán đám mây, Wikipedia](https://vi.wikipedia.org/wiki/%C4%90i%E1%BB%87n_to%C3%A1n_%C4%91%C3%A1m_m%C3%A2y#:~:text=%C4%90i%E1%BB%87n%20to%C3%A1n%20%C4%91%C3%A1m%20m%C3%A2y%20(ti%E1%BA%BFng,tri%E1%BB%83n%20d%E1%BB%B1a%20v%C3%A0o%20m%E1%BA%A1ng%20Internet.)
2. [Docker là gì? Tìm hiểu về Docker, topdev.vn](https://topdev.vn/blog/docker-la-gi/)
3. [Kubernetes là gì? Cùng tìm hiểu cách hoạt động, topdev.vn](https://topdev.vn/blog/kubernetes-la-gi/)