**Rectangle

Description automatically generated with low confidence  
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BỘ MÔN CÔNG NGHỆ PHẦN MỀM**

**Logo

Description automatically generated**

**BÁO CÁO CUỐI KỲ**

**Đề tài:**

**XÂY DỰNG HỆ THỐNG CLOUD MÔ PHỎNG TRANG DẠY HỌC SỐ CÓ THỂ TỰ SCALE UP, SCALE DOWN**

**GIẢNG VIÊN HƯỚNG DẪN**

**TS. HUỲNH XUÂN PHỤNG**

**NHÓM SINH VIÊN THỰC HIỆN**

**LÊ ANH NHÂN – 20110689**

**PHẠM NGỌC THẮNG - 20110728**

**ĐỖ QUỐC VIỆT – 19110498**

**TP. HỒ CHÍ MINH, THÁNG 12 NĂM 2022**

|  |  |
| --- | --- |
| **ĐH SƯ PHẠM KỸ THUẬT TP. HCM**  **KHOA CNTT**  **\*\*\*\*\*\*\*** | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh phúc**  **\*\*\*\*\*\*\*** |

**PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

|  |  |
| --- | --- |
| Họ và tên Sinh viên 1: **Lê Anh Nhân** | MSSV 1: **20110689** |
| Họ và tên Sinh viên 2: **Phạm Ngọc Thắng** | MSSV 2: **20110728** |
| Họ và tên Sinh viên 3: **Đỗ Quốc Việt** | MSSV 3: **19110498** |

Ngành: **Công nghệ Thông tin**

Tên đề tài: **Xây dựng hệ thống Cloud mô phỏng trang dạy học số có thể tự scale up, scale down**

Họ và tên Giảng viên hướng dẫn: **Ts. Huỳnh Xuân Phụng**

**NHẬN XÉT**

1. Về nội dung đề tài và khối lượng thực hiện:

1. Ưu điểm:

1. Khuyết điểm

1. Điểm:

*Tp. Hồ Chí Minh, ngày 06 tháng 12 năm 2022*

Giảng viên hướng dẫn

*(Ký và ghi rõ họ tên)*

# LỜI CẢM ƠN

Nhóm chúng em xin phép gửi lời cảm ơn chân thành tới trường Đại học Sư phạm Kỹ thuật thành phố Hồ Chí Minh và khoa Công nghệ Thông tin đã tạo điều kiện cho chúng em được học tập, nghiên cứu, trau dồi kiến thức để có khả năng hoàn thành tốt đề tài.

Tiểu luận chuyên ngành Công nghệ Phần mềm với đề tài *“***Xây dựng hệ thống Cloud mô phỏng trang dạy học số có thể tự scale up, scale down***”* là kết quả của quá trình cố gắng không ngừng nghỉ của chúng em và được sự giúp đỡ tận tình của thầy cô. Qua đây, em xin gửi lời cảm ơn chân thành những người đã giúp đỡ chúng em trong quá trình học tập cũng như quá trình làm tiểu luận.

Chúng em xin trân trọng gửi đến cô Ts. Huỳnh Xuân Phụng, người đã trực tiếp hướng dẫn trong suốt quá trình thực hiện đề tài lời cảm ơn chân thành và trân trọng nhất.

Nhờ sự giúp đỡ của thầy cô, chúng em đã có nền tảng kiến thức vững chắc để thực hiện và hoàn thành đề tài một cách tốt nhất. Tuy nhiên, vì khả năng còn hạn hẹp nên không tránh khỏi xảy ra thiếu sót, chúng em mong sẽ nhận được sự góp ý của thầy cô để chúng em có thể rút kinh nghiệm và hoàn thiện hơn đề tài này.

Chúng em xin chân thành cảm ơn!

**Nhóm thực hiện**

Lê Anh Nhân – 20110689

Phạm Ngọc Thắng - 20110728

Đỗ Quốc Việt – 19110498

# MỤC LỤC

[LỜI CẢM ƠN 1](#_Toc122384482)

[MỤC LỤC 2](#_Toc122384483)

[PHẦN NỘI DUNG 3](#_Toc122384484)

[CHƯƠNG 1: MỞ ĐẦU 3](#_Toc122384485)

[1.1. Lý do chọn đề tài 3](#_Toc122384486)

[1.2. Mục tiêu 3](#_Toc122384487)

[CHƯƠNG 2: KHẢO SÁT HIỆN TRẠNG 4](#_Toc122384488)

[2.1. Khảo sát hiện trạng 4](#_Toc122384489)

[2.2. Kết luận 6](#_Toc122384490)

[CHƯƠNG 3: PHÂN TÍCH THIẾT KẾ HỆ THỐNG 7](#_Toc122384491)

[3.1. Xác định yêu cầu 7](#_Toc122384492)

[3.2. Mô hình hóa sơ đồ quan hệ 8](#_Toc122384493)

[CHƯƠNG 4: KIẾN TRÚC HỆ THỐNG VÀ CÁC CÔNG NGHỆ SỬ DỤNG 10](#_Toc122384494)

[4.1. Spring boot 10](#_Toc122384495)

[4.2. Spring data jpa 11](#_Toc122384496)

[4.3. Spring Security 12](#_Toc122384497)

[4.4. MariaDB 12](#_Toc122384498)

[4.5. Docker 13](#_Toc122384499)

[Các bước thực hiện 16](#_Toc122384500)

[Demo kết quả 18](#_Toc122384501)

# PHẦN NỘI DUNG

## CHƯƠNG 1: MỞ ĐẦU

### Lý do chọn đề tài

Hiện nay, dưới sự ảnh hưởng của nền công nghiệp 4.0 và sự phát triển mạnh mẽ của Internet, lượng truy cập và sử dụng website của mọi người ngày càng tăng cao. Kéo theo đó, nhu cầu xây dựng và phát triển website cũng tăng không kém.

Trong quá trình đó, việc đáp ứng nhu cầu lượng lớn người dùng đồng thời truy cập là rất cần thiết. Hệ thống nên tự scale để phù hợp với lượng người dùng.

Chính vì lý do đó, việc nghiên cứu và phát triển một hệ thống có thể tự scale là rất cần thiết. Nhóm chúng em đã nghiên cứu và thực hiện đề tài *“Xây dựng hệ thống Cloud mô phỏng trang dạy học số có thể tự scale up, scale down”* để nghiên cứu quá trình tự động scale của hệ thống.

### Mục tiêu

Trong quá trình làm đề tài, nhóm chúng em đề ra những mục tiêu sau:

+ Xây dựng một hệ thống trang đăng ký môn học mà có thể tự đáp ứng lượng nhu cầu người dùng nhiều hay ít;

+ Xây dựng một hệ thống trang đăng ký môn học mà sinh viên có thể đăng nhập, liệt kê và đăng ký môn học;

+ Giao diện thân thiện, dễ dàng sử dụng;

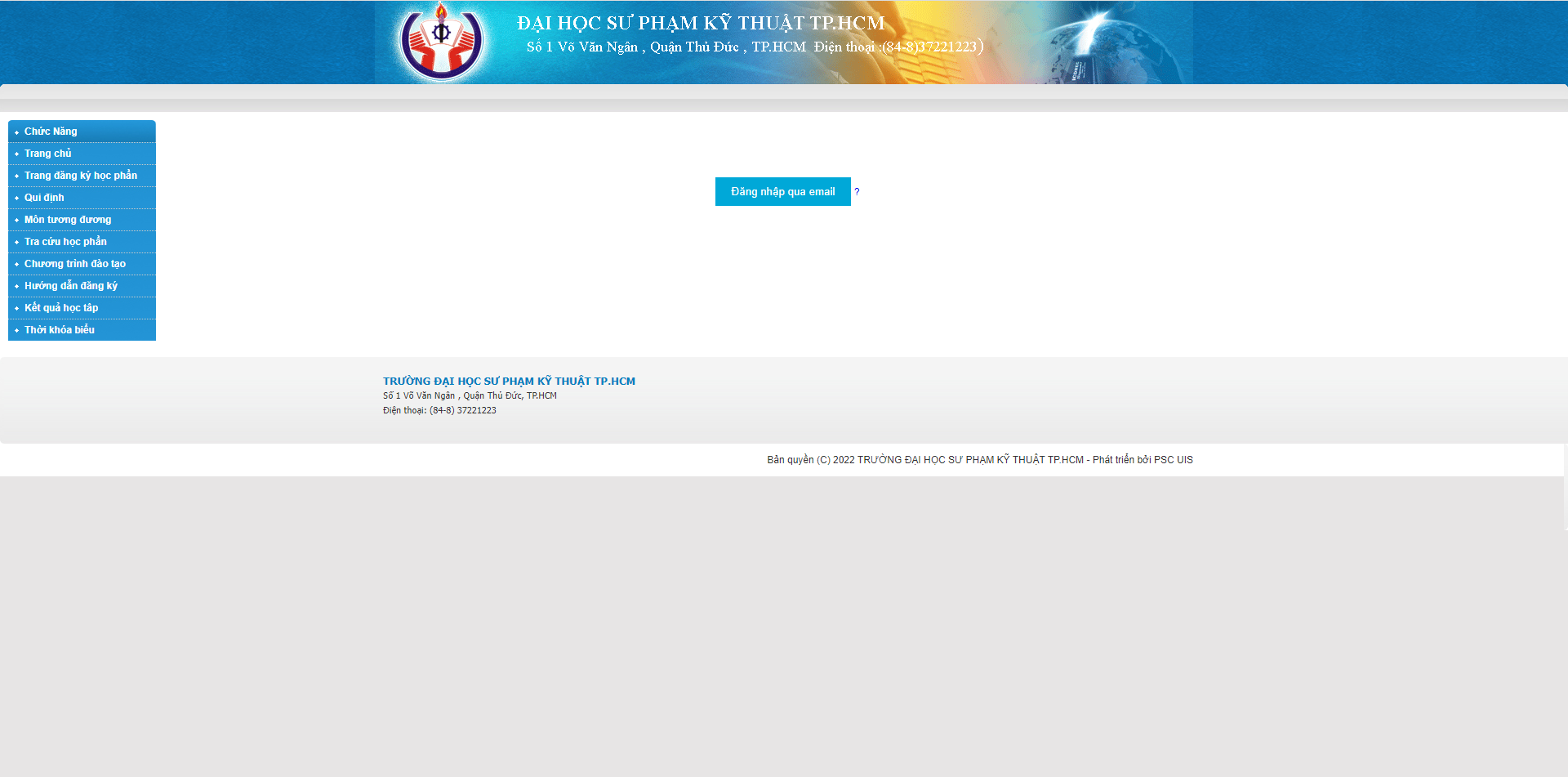
## CHƯƠNG 2: KHẢO SÁT HIỆN TRẠNG

### Khảo sát hiện trạng

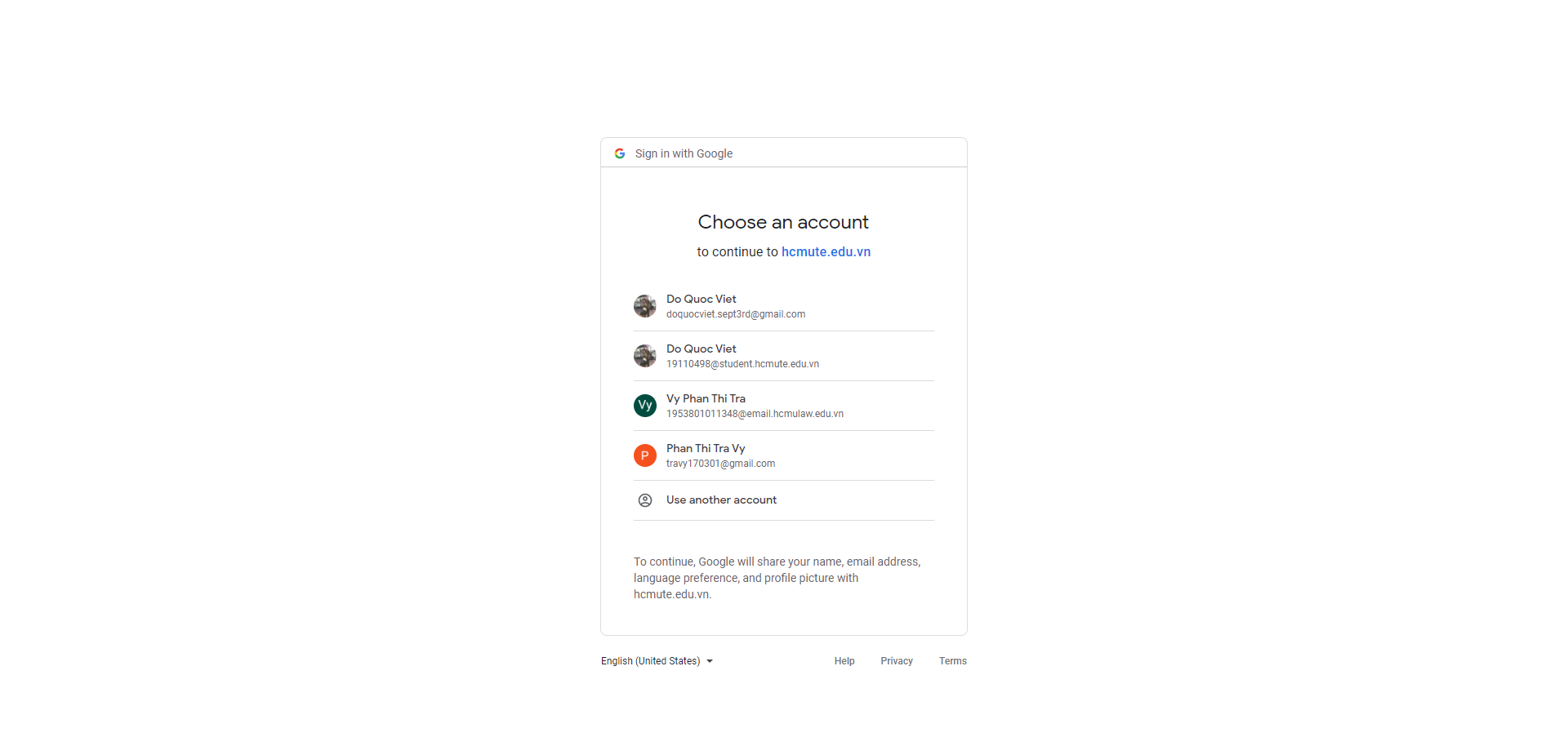
Nhóm chúng em đã thực hiện tiến hành khảo sát trên trang đăng ký môn học của trường Đại học Sư phạm Kỹ thuật.

+ dkmh (<https://dkmh.hcmute.edu.vn>);

Phần đăng nhập: giao diện đăng nhập chủ yếu sử dụng email mà sinh viên đã được cung cấp.



Sau đó, sinh viên sử dụng tài khoản miền @student.hcmute.edu.vn để đăng nhập vào hệ thống.



Sau khi đã đăng nhập vào hệ thống, ta thấy giao diện các môn học được liệt kê một cách chi tiết bao gồm các thông tin chính như STT, mã môn, tên môn, loại phần và số tín chỉ.



Sinh viên có thể chọn đăng ký cho việc chọn môn học và đăng ký.



*Nhận xét:* Giao diện thân thiện, dễ dàng đăng ký môn học cho sinh viên.

### Kết luận

Sau khi đánh giá, khảo sát, nhóm chúng em đã rút ra được một số bài học cũng như có những kinh nghiệm để ứng dụng vào đề tài của của mình:

+ Giao diện phải bắt mắt, thân thiện với người dùng (UI).

+ Nâng cao trải nghiệm người dùng (UX).

+ Dễ dàng đăng nhập.

+ Tự động scale hệ thống để đáp ứng nhu cầu người dùng.

## CHƯƠNG 3: PHÂN TÍCH THIẾT KẾ HỆ THỐNG

### Xác định yêu cầu

#### Yêu cầu chức năng

Các yêu cầu chức năng của hệ thống bao gồm:

*+ Đăng nhập và phân quyền:*

* Đăng ký tạo tài khoản.
* Đăng nhập vào hệ thống.
* Phân quyền.

*+ Đăng ký môn:*

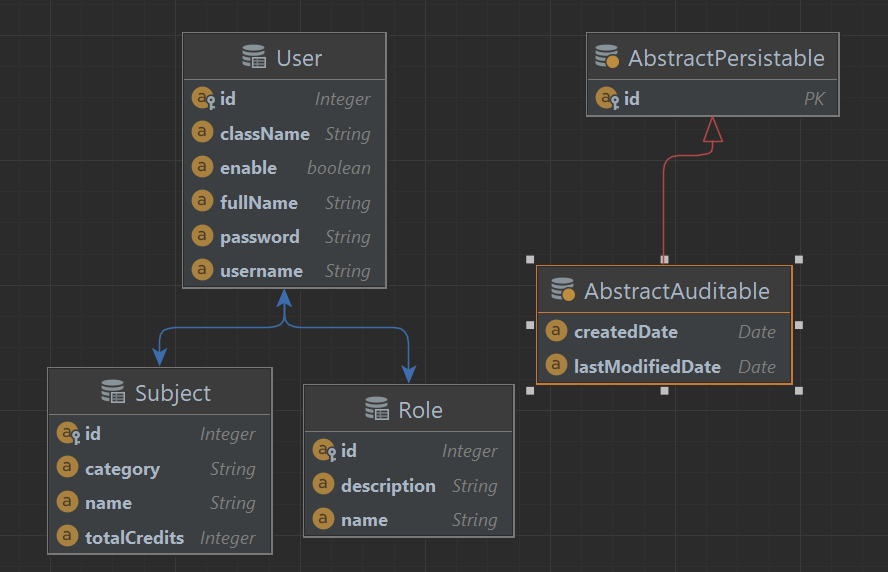
* Chọn môn học và tiến hành đăng ký.

1. Yêu cầu phi chức năng

Các yêu cầu phi chức năng bao gồm:

* Giao diện người dùng thân thiện, dễ sử dụng.
* Tốc độ xử lý các thao tác nhanh chóng và chính xác.
* Hệ thống có tính bảo mật, phân quyền rõ ràng.
* Việc bảo trì, nâng cấp và phát triển dễ dàng.

### Mô hình hóa sơ đồ quan hệ



Sơ đồ thực thể bao gồm:

+ User: Thông tin sinh viên (user);

+ Role: Chứa quyền;

+ Subject: Chứa thông tin môn học;

+ Register: Chứa thông tin đăng ký của sinh viên;

+ User\_Role: Quyền của sinh viên;

|  |  |  |
| --- | --- | --- |
| Table | Fields | PROPERTIES |
| User | id | INT NOT NULL PRIMARY KEY |
| className | VARCHAR (255) |
| enable | BIT |
| fullName | VARCHAR (255) |
| password | VARCHAR (255) |
| username | VARCHAR (255) UNIQUE |
| Subject | id | INT NOT NULL PRIMARY KEY |
| category | VARCHAR (255) |
| name | VARCHAR (255) |
| totalCredits | VARCHAR (255) |
| Role | id | INT NOT NULL PRIMARY KEY |
| description | VARCHAR (255) |
| name | VARCHAR (255) |
| Register | subjectId | INT NOT NULL PRIMARY KEY FOREIGN KEY REF SUBJECT |
| userId | INT NOT NULL PRIMARY KEY FOREIGN KEY REF USER |
| User\_Role | userId | INT NOT NULL PRIMARY KEY FOREIGN KEY REF USER |
| roleId | INT NOT NULL PRIMARY KEY FOREIGN KEY REF ROLE |

## CHƯƠNG 4: KIẾN TRÚC HỆ THỐNG VÀ CÁC CÔNG NGHỆ SỬ DỤNG

## Spring boot

Spring Boot là một extension của Spring Framework giúp các lập trình viên loại bỏ các bước cấu hình phức tạp mà Spring bắt buộc. Spring Boot là dự án phát triển bởi ngôn ngữ Java (JAVA) trong hệ sinh thái Spring framework.

*Đặc điểm nổi bật*

- Được phát triển tối ưu sao cho cấu hình XML trở nên đơn giản nhanh chóng và dễ dàng nhất trong Spring.

- Gia tăng được năng suất trong quá trình lập trình.

- Giảm ở mức tối thiểu thời gian lập trình.

- Giúp người dùng mặc dù không có nhiều kiến thức lập trình vẫn có thể xây dựng được một ứng dụng.

Một số tính năng nổi bật của Spring Boot đó là:

- Tạo ra các ứng dụng Spring mang tính độc lập.

- Nhúng trực tiếp Tomcat, Jetty hoặc Undertow mà không cần phải triển khai ra file WAR.

- Starter dependency giúp cho việc chạy cấu hình Maven trở nên đơn giản hơn.

- Tự động chạy cấu hình Spring khi cần thiết.

- Không sinh code cấu hình, đồng thời không yêu cầu phải cấu hình bằng XML.

Một số tính năng rất quan trọng của Spring Boot là:

- SpringApplication: Khi lập trình xong và bạn chỉ muốn chạy thử nghiệm thì nên làm thế nào? Bạn chỉ cần gọi run() là được, vì Spring Boot được thiết kế theo dạng “just run”. Giúp cho các lập trình viên chỉ cần cấu hình ít Spring nhất, phần còn lại Spring Boot sẽ lo liệu.

- Externalized Configuration: Bạn mong muốn tạo nên một ứng dụng có thể chạy trên nhiều loại môi trường khác nhau? Spring Boot sẽ giúp bạn config cấu hình từ ngoài và ứng dụng của bạn sẽ có thể chạy thoải mái.

- Profiles: Nếu có nhiều config khác nhau, bạn có thể sử dụng Profile để phân chia từng loại cho từng môi trường để dễ dàng quản lý hơn.

- Logging: Tính năng này được sử dụng cho toàn bộ chức năng log trong phạm vi nội bộ và nó được quản lý mặc định.

## Spring data jpa

Spring Data là một module của Spring Framework. Mục đích của Spring Data JPA là giảm thiểu việc thực hiện quá nhiều bước để có thể implement được JPASpring Data JPA, một phần của họ Spring Data lớn hơn, giúp dễ dàng triển khai các kho lưu trữ dựa trên JPA. Mô-đun này liên quan đến hỗ trợ nâng cao cho các lớp truy cập dữ liệu dựa trên JPA.

Nó giúp việc xây dựng các ứng dụng dựa trên Spring sử dụng các công nghệ truy cập dữ liệu trở nên dễ dàng hơn.Việc triển khai lớp truy cập dữ liệu của một ứng dụng đã khá cồng kềnh trong một thời gian dài. Quá nhiều mã soạn sẵn phải được viết để thực hiện các truy vấn đơn giản cũng như thực hiện phân trang và kiểm tra. Spring Data JPA nhằm mục đích cải thiện đáng kể việc triển khai các lớp truy cập dữ liệu bằng cách giảm nỗ lực xuống mức thực sự cần thiết.

Với tư cách là nhà phát triển, bạn viết các giao diện kho lưu trữ của mình, bao gồm các phương thức tìm kiếm tùy chỉnh và Spring sẽ tự động cung cấp triển khai.Đặc trưnHỗ trợ tinh vi để xây dựng kho lưu trữ dựa trên Spring và JPA.

Hỗ trợ các vị từ Querydsl và do đó các truy vấn JPA an toàn về kiểu.

Kiểm toán minh bạch của lớp miền.

Hỗ trợ phân trang, thực thi truy vấn động, khả năng tích hợp mã truy cập dữ liệu tùy chỉnh.

Xác thực các @Query truy vấn được chú thích tại thời điểm khởi động.

Hỗ trợ ánh xạ thực thể dựa trên XML.

Cấu hình kho lưu trữ dựa trên JavaConfig bằng cách giới thiệu @EnableJpaRepositories.

## Spring Security

Spring Security là một khung xác thực và kiểm soát truy cập mạnh mẽ và có khả năng tùy biến cao. Đây là tiêu chuẩn thực tế để bảo mật các ứng dụng dựa trên Spring.

Spring Security là một framework tập trung vào việc cung cấp cả xác thực và ủy quyền cho các ứng dụng Java. Giống như tất cả các dự án Spring, sức mạnh thực sự của Spring Security nằm ở việc nó có thể được mở rộng dễ dàng như thế nào để đáp ứng các yêu cầu tùy chỉnh.

Đặc trưng:

Hỗ trợ toàn diện và có thể mở rộng cho cả Xác thực và Ủy quyền.

Bảo vệ chống lại các cuộc tấn công như cố định phiên, clickjacking, giả mạo yêu cầu trang chéo, v.v.

Tích hợp API Servlet.

Tích hợp tùy chọn với Spring Web MVC.

## MariaDB

MariaDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở (DBMS), thay thế cho MySQL . Nó được tạo ra như một nhánh của MySQL bởi các nhà phát triển. Những người này đóng vai trò quan trọng trong công việc xây dựng gốc cơ sở dữ liệu. Họ đã nghĩ ra MariaDB vào năm 2009 để đáp lại việc mua lại MySQL của Oracle Corp

MariaDB dựa trên SQL và hỗ trợ xử lý dữ liệu ACID-style. Ngoài ra, cơ sở dữ liệu cũng hỗ trợ các API JSON, sao chép dữ liệu song song và nhiều công cụ lưu trữ, bao gồm InnoDB, MyRocks, Spider, Aria, TokuDB, Cassandra và MariaDB ColumnStore.

Ưu điểm của MariaDB là gì?

- Hoàn toàn miễn phí

- Khắc phục những hạn chế của MySQL

- Bổ sung thêm nhiều Engine hơn

- Kết hợp cả SQL và NoSQL

- Hỗ trợ tiếng Việt

## Docker

*+ Docker là gì?*

Docker là một nền tảng để cung cấp cách để building, deploying và running ứng sử dụng các containers (trên nền tảng ảo hóa).

*+ Container trong Docker là gì?*

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.

*+ Các khái niệm liên quan*

Docker Engine : là thành phần chính của Docker, như một công cụ để đóng gói ứng dụng

Docker Hub : là một “github for docker images”. Trên DockerHub có hàng ngàn public images được tạo bởi cộng đồng cho phép bạn dễ dàng tìm thấy những image mà bạn cần. Và chỉ cần pull về và sử dụng với một số config mà bạn mong muốn.

Images: là một khuôn mẫu để tạo một container. Thường thì image sẽ dựa trên 1 image có sẵn với những tùy chỉnh thêm. Ví dụ bạn build 1 image dựa trên image Centos mẫu có sẵn để chạy Nginx và những tùy chỉnh, cấu hình để ứng dụng web của bạn có thể chạy được. Bạn có thể tự build một image riêng cho mình hoặc sử dụng những image được chia sẽ từ cộng đồng Docker Hub. Một image sẽ được build dựa trên những chỉ dẫn của Dockerfile.

Container: là một instance của một image. Bạn có thể create, start, stop, move or delete container dựa trên Docker API hoặc Docker CLI.

Dockerfile: là một tập tin bao gồm các chỉ dẫn để build một image .

Volumes: là phần dữ liệu được tạo ra khi container được khởi tạo.

Docker Compose: Thực hiện chạy ứng dụng thông qua các định nghĩa cấu hình các Docker Container, thực hiện thông qua file cấu hình.

Docker machine: Là hệ thống tạo ra các Host Docker trên máy chủ. Các máy này sao khi được tạo ra đã được cài đặt docker. Và sau này mình có thể sử dụng docker swarm để kết nối các host docker.

*+ Docker swarm là gì?*

Docker Swarm là một nhóm các máy chạy Docker và tập hợp lại với nhau thành một cluster. Không như docker engine, sau khi các máy này tập hợp vào Swarm, mọi câu lệnh Docker sẽ được thực thi trên Swarm manager.

Các máy tham gia vào swarm được gọi là worker node. Các node này chỉ có khả năng cung cấp khả năng hoạt động chứ không có quyền quản lý các node khác.

Docker Swarm có khả năng khởi chạy các container trên nhiều máy (cluster - máy ảo hoặc máy vật lý) hoặc trên một máy duy nhất (standalone)

Những tính năng nổi bật của Docker Swarm

Quản lý cụm (cluster) với Docker Engine: Sử dụng Docker Engine CLI để tạo ra một cụm Docker engine - nơi bạn có thể triển khai các dịch vụ ứng dụng mà không cần một phần mềm điều phối bổ sung để quản lý.

Thiết kế phân cấp: Với Docker engine, bạn có thể phân cấp các nodes với vai trò manager hay worker. Điều này giúp bạn tạo một mô hình phân cấp chỉ từ một image duy nhất.

Scaling: Đối với mỗi service, bạn có thể chạy số lượng task mà mình mong muốn. Khi tăng hoặc giảm tỷ lệ, swarm sẽ tự động điều chỉnh bằng cách thêm hoặc xóa các task để đạt trạng thái mong muốn.

Load balaning: Bạn có thể xuất các cổng trong node cho bộ cân bằng tải bên ngoài. Còn nội bộ, swarm cho phép bạn chỉ định cách phân phối request đến các containers trong các node.

Và còn nhiều tính năng khác.

*+ Các khái niệm trên Docker Swarm*

Services là một khái niệm bên trong docker, như ở trên các bạn có define các service trong file docker-compose đó. Để deploy một image app khi sử dụng docker swarm thì bạn cần phải tạo một service. Thông thường thì một service sẽ là image cho một microservice trong một ứng dụng lớn. Với docker swarm nó có hỗ trợ một hoặc nhiều replica tasks, nên mỗi service sẽ chứa một hoặc nhiều replica container tùy theo sự monitor của ta.

Docker stack: là một command được nhúng vào Docker CL. Nó có thể chạy các dịch vụ và quản lý các dịch vụ trên Swarm. Các dịch vụ này đã được cấu hình trong file docker-compose (.yml)

*+ Phần hệ thống*

Trong bài này, thì hệ thống sử dụng các ứng dụng để dám sát hệ thống:

cAdvisor (Container Advisor) cung cấp cho người dùng container sự hiểu biết về việc sử dụng tài nguyên và đặc điểm hiệu suất của các container đang chạy của họ. Đây là một daemon đang chạy thu thập, tổng hợp, quy trình và xuất khẩu thông tin về các container đang chạy. Cụ thể, đối với mỗi container, nó giữ các thông số cách ly tài nguyên, sử dụng tài nguyên lịch sử, biểu đồ sử dụng tài nguyên lịch sử hoàn chỉnh và thống kê mạng. Dữ liệu này được xuất khẩu bằng container và trên toàn máy.

Prometheus là một hệ thống giám sát dịch vụ và hệ thống. Nó thu thập các chỉ số từ các mục tiêu đã định cấu hình trong các khoảng thời gian nhất định, đánh giá các biểu thức quy tắc, hiển thị kết quả và có thể kích hoạt cảnh báo nếu một số điều kiện được quan sát là đúng. Ở đây, Prometheus sẽ giám sát các thông số được cập nhật từ các container thông cAdvisor, từ số liệu đó sẽ kiểm tra hiệu năng của cpu, sau đó thiết lập các luật để để có thể scale up và scale down.

*+ Công cụ Apache HTTP server benchmarking tool (ab).*

Công cụ Apache HTTP server benchmarking tool (ab): được phát triển bởi Apache Software Foundation, là công cụ tạo ra tải mô phỏng cho Web Server bằng cách gửi các request đồng thời tới máy chủ. Công cụ này được tích hợp mặc định sau khi cài dịch vụ HTTP Apache Web Server phục vụ việc đánh giá Web server sau khi cài đặt.

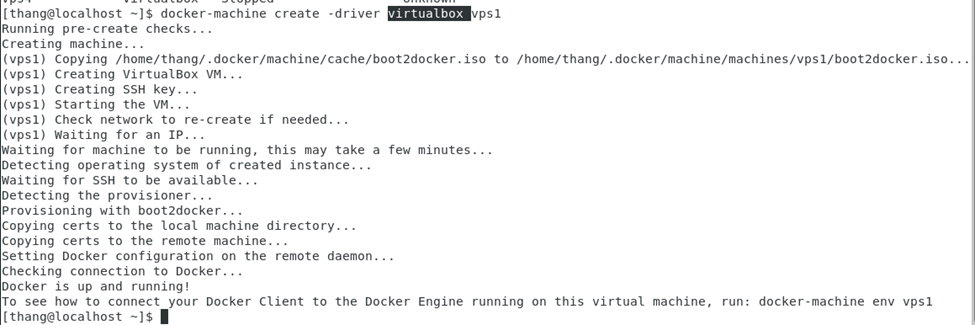
# Các bước thực hiện

CÀI ĐẶT MÔI TRƯỜNG

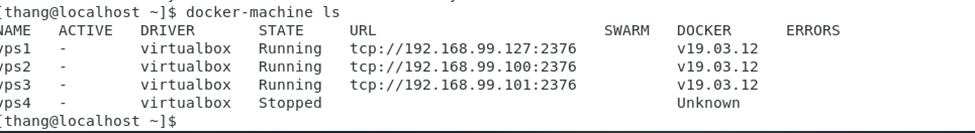
* CentOS 7
* Docker
* Virtualbox
* Apache Bench

Bước 1: Khởi tạo các Host Docker

Sử dụng lệnh docker-machine create để tạo ra 3 Host Docker. Các máy này sau khi tạo đã được cài đặt docker



Sau khi tạo ra 3 Host Docker ta sử dụng lênh docker-machine ls để kiểm tra



Bước 2: Khởi tạo docker swarm

Để khởi tạo docker swarm ta truy cập vào một Host Docker bất kỳ sao đó khởi tạo nó ở chế độ Swarm mode bằng lệnh docker swarm init. Sau khi khởi tạo thì máy này sẽ trở thành một Node manager



Tiếp theo join các Host docker còn lại vào swarm bằng lệnh docker swarm join… Đã gợi ý phía trên





Sau khi thực hiện xong ta kiểm tra bằng lệnh docker node ls



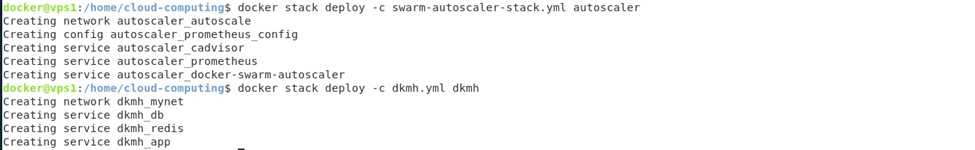
Bước 3: Copy folder project vào Host Docker

Sử dụng lệnh docker-machine scp



Bước 4: Chạy Docker Stack

Nội dung của ứng dụng đã được đóng gói thành các docker image để trên docker hub, Được để sẵn trong file docker compose, chỉ cần chạy deploy stack là có thể hoạt động được



Bước 5: Chạy visualizer



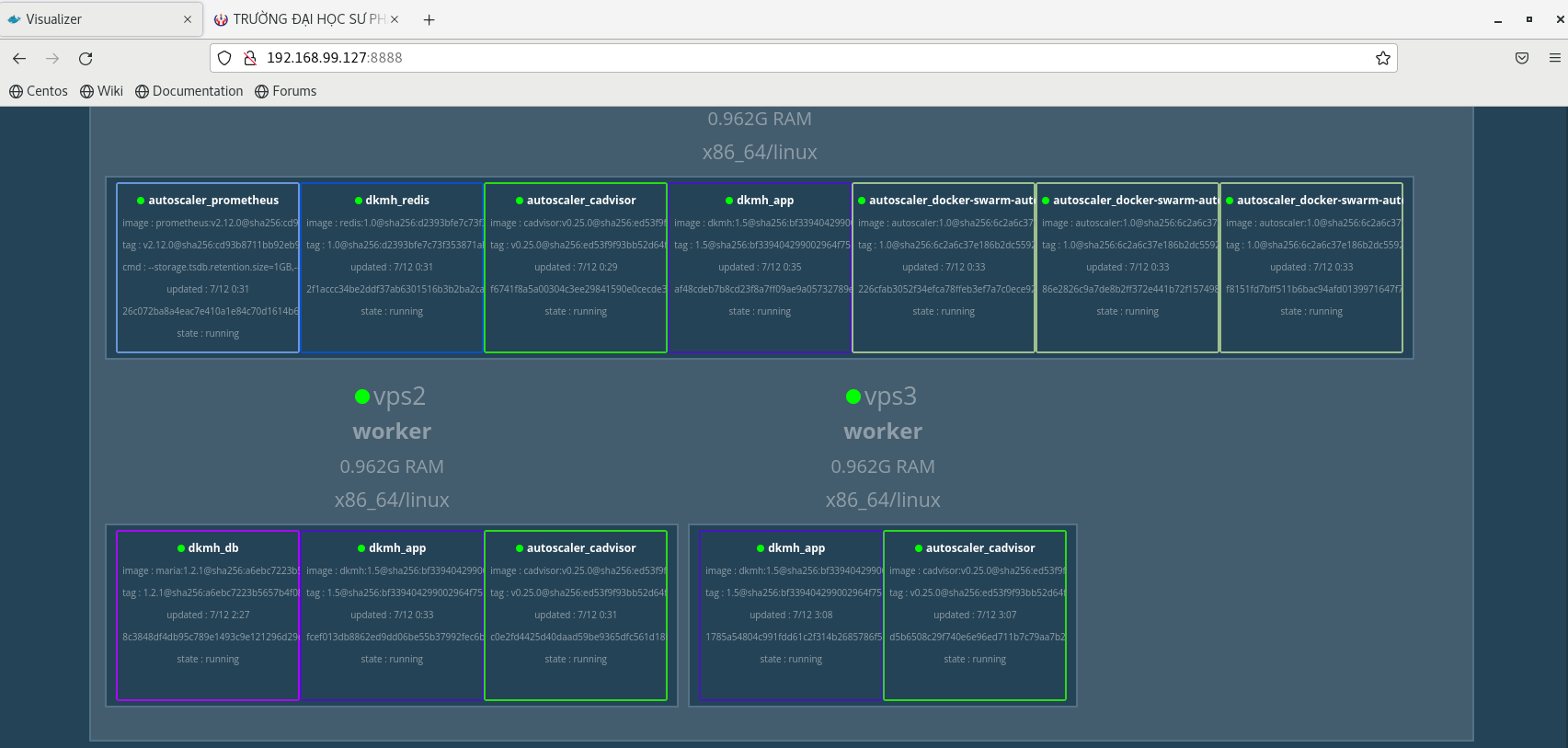
Bước 6: Kiểm tra container đang chạy trong hệ thống

Sử dụng lệnh docker ps



# Demo kết quả

Truy cập vào địa chỉ node manager trong hệ thống Swarm theo port 8888 để vào Viualizer. Công cụ này giúp ta kiểm soat được các container đang chạy trong hệ thống Swarm



Sử dụng docker stats để kiểm tra các container trong hệ thống và có thể xem mục CPU và Memory đang sử dụng

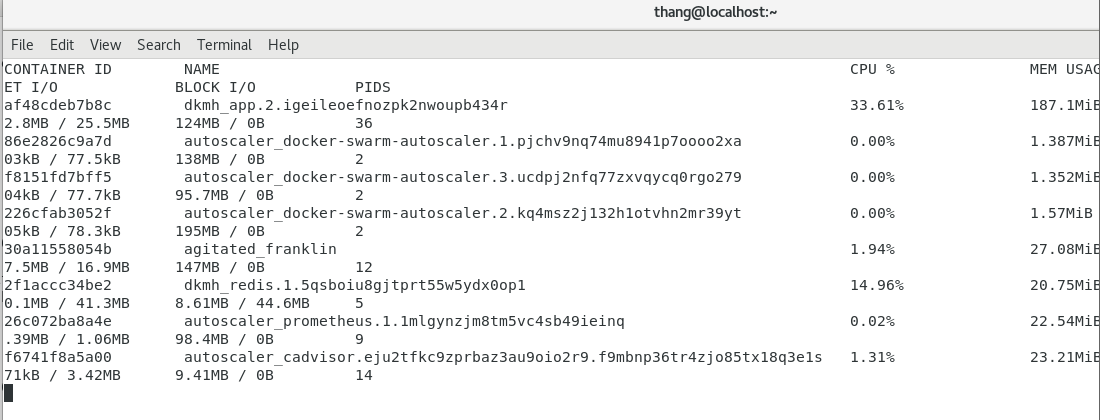


Container chúng ta quan tâm là dkmh\_app nó sẽ chạy app đăng ký môn học. Lúc này chưa có request nào truy cập vào nên CPU đang ở mức rất thấp

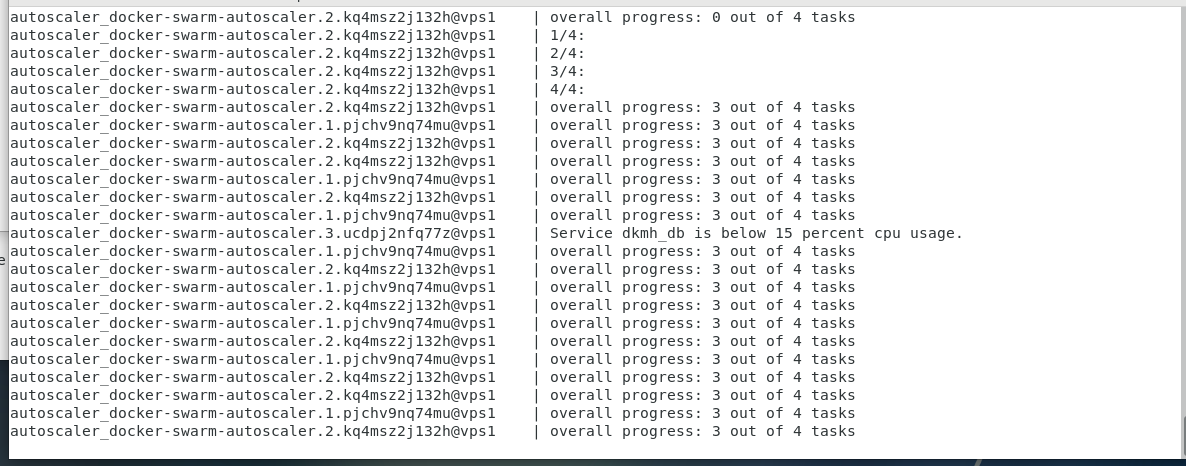
Ta sử dụng công cụ Apache bench để tang số lượng request trên trang dkmh.



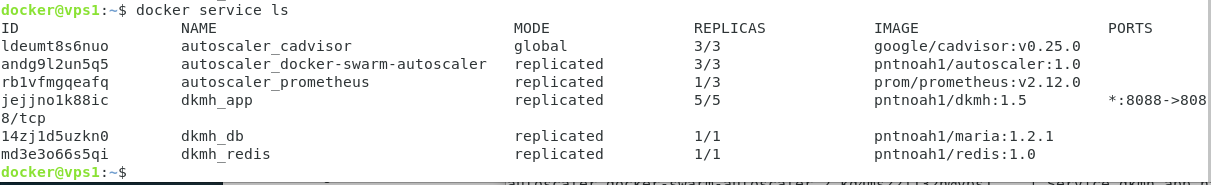
Lúc này CPU sẽ tăng lên mức cao (> 30%) lúc này đợi một thời gian thì hệ thống sẽ scale up.



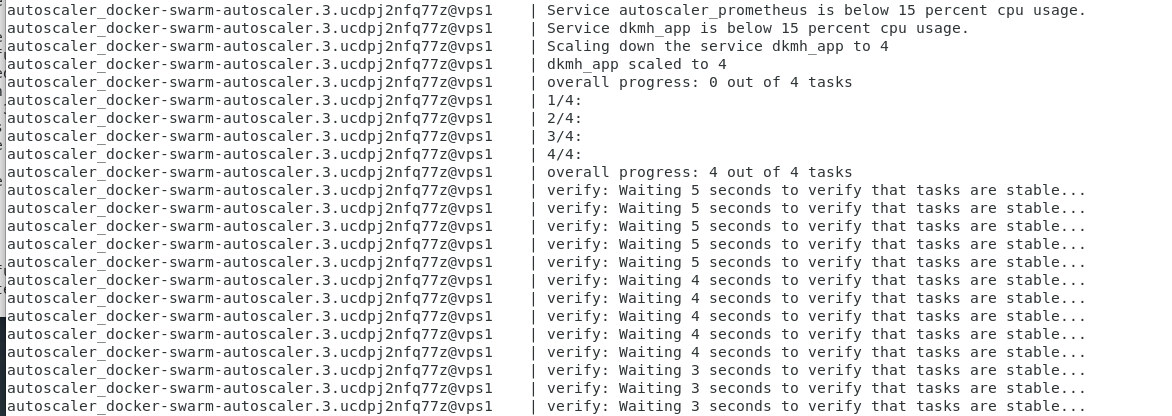
Sau một thời gian thì hệ thống đã scale up lên 4 task, 4 bản sao container



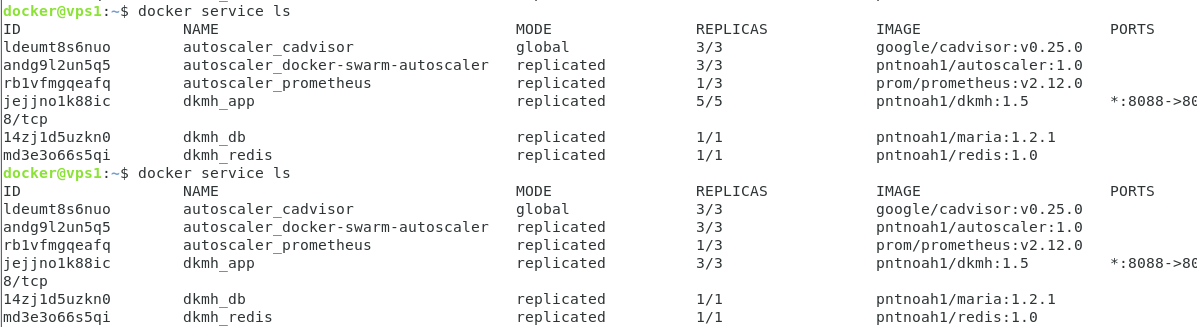
Sử dụng lệnh docker service ls để xem kết quả



Đợi một thời gian sau khi apache bench ko còn gửi request nữa thì hệ thống sẽ giảm số lượng task xuống

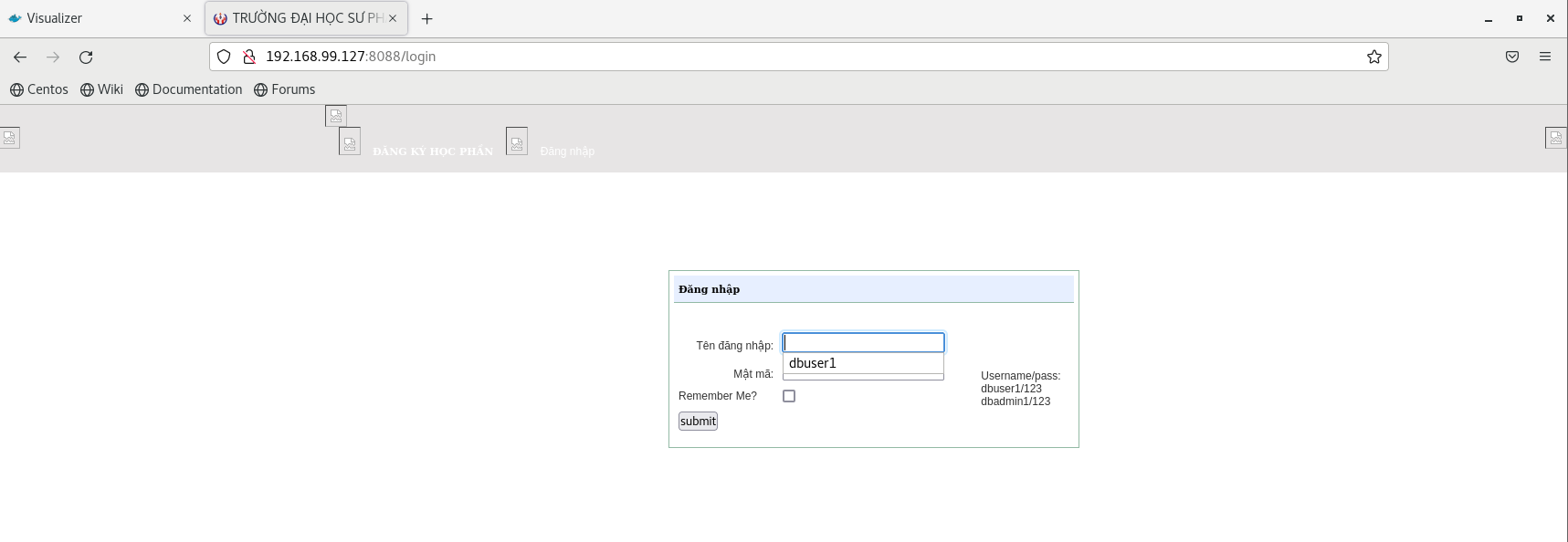


Sử dụng lệnh docker service ls để xem kết quả. Kết quả có được là đã giảm từ 5 task xuống còn 3 task.

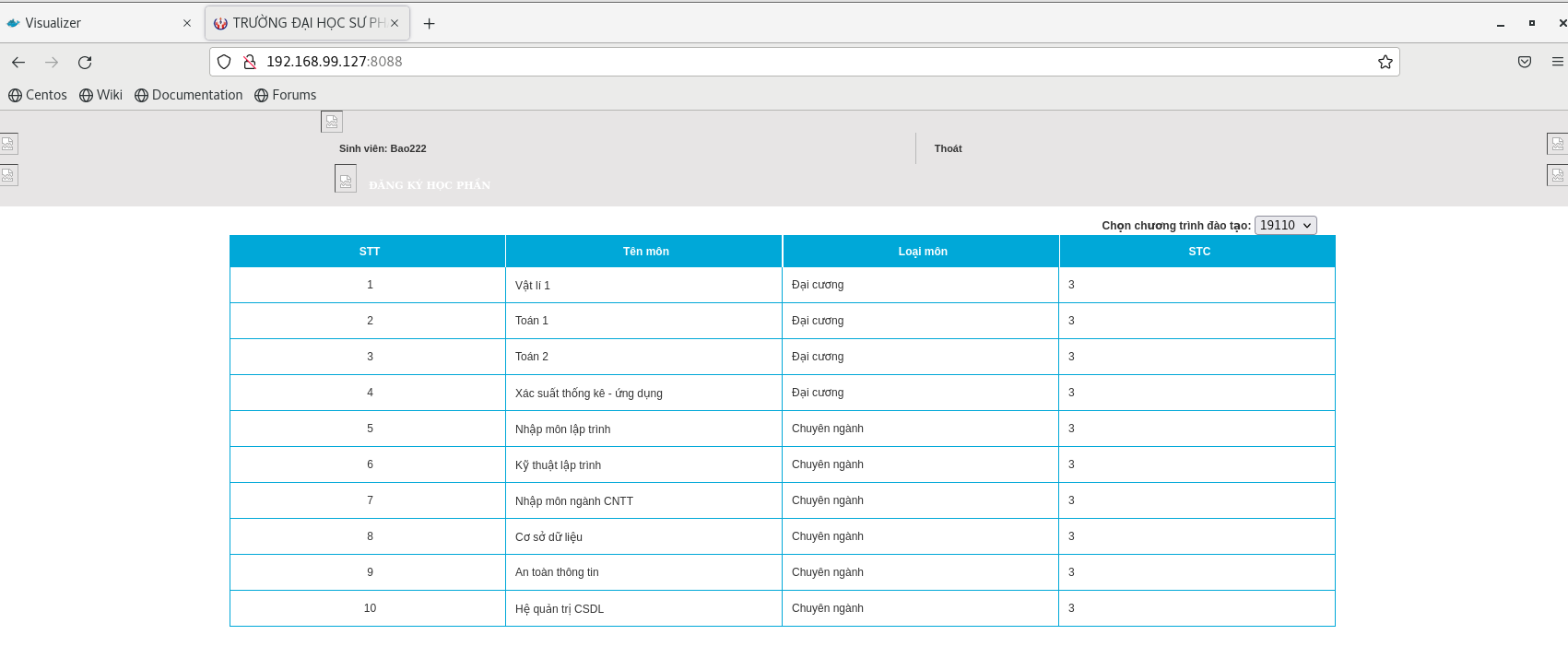


DEMO Trang web đăng ký môn học

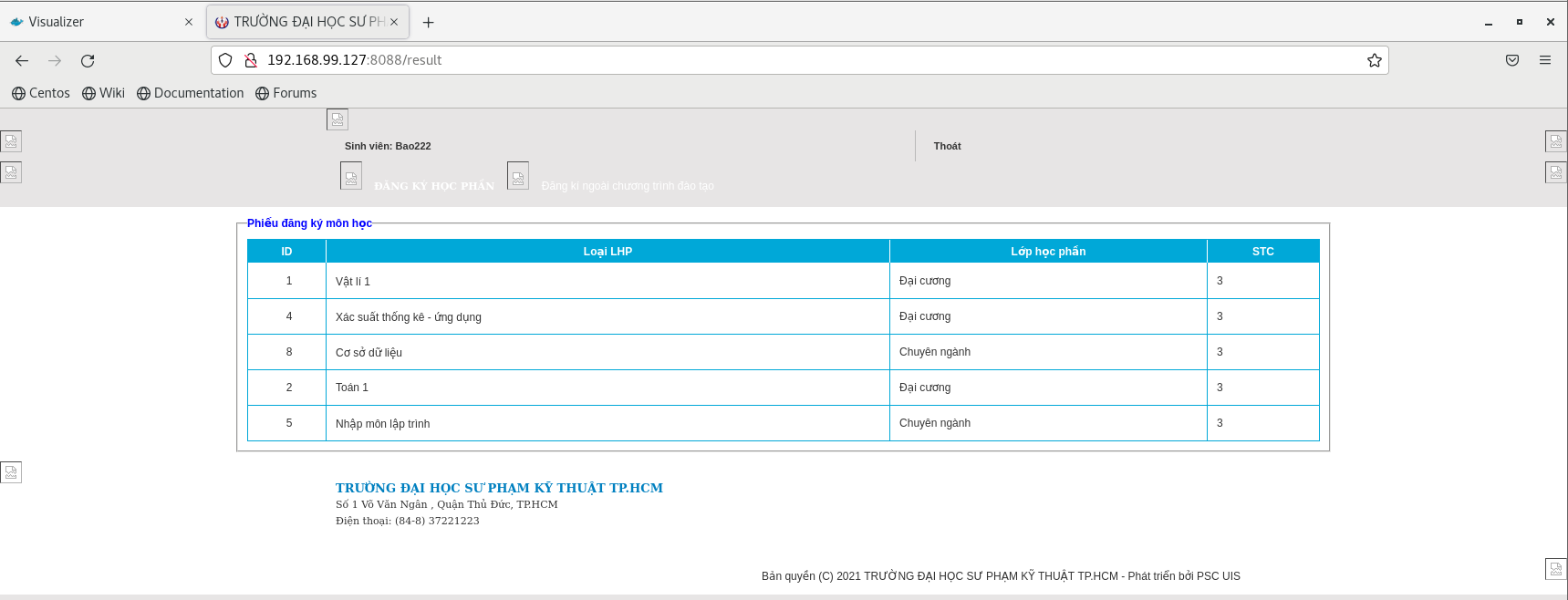
Trang đăng nhập



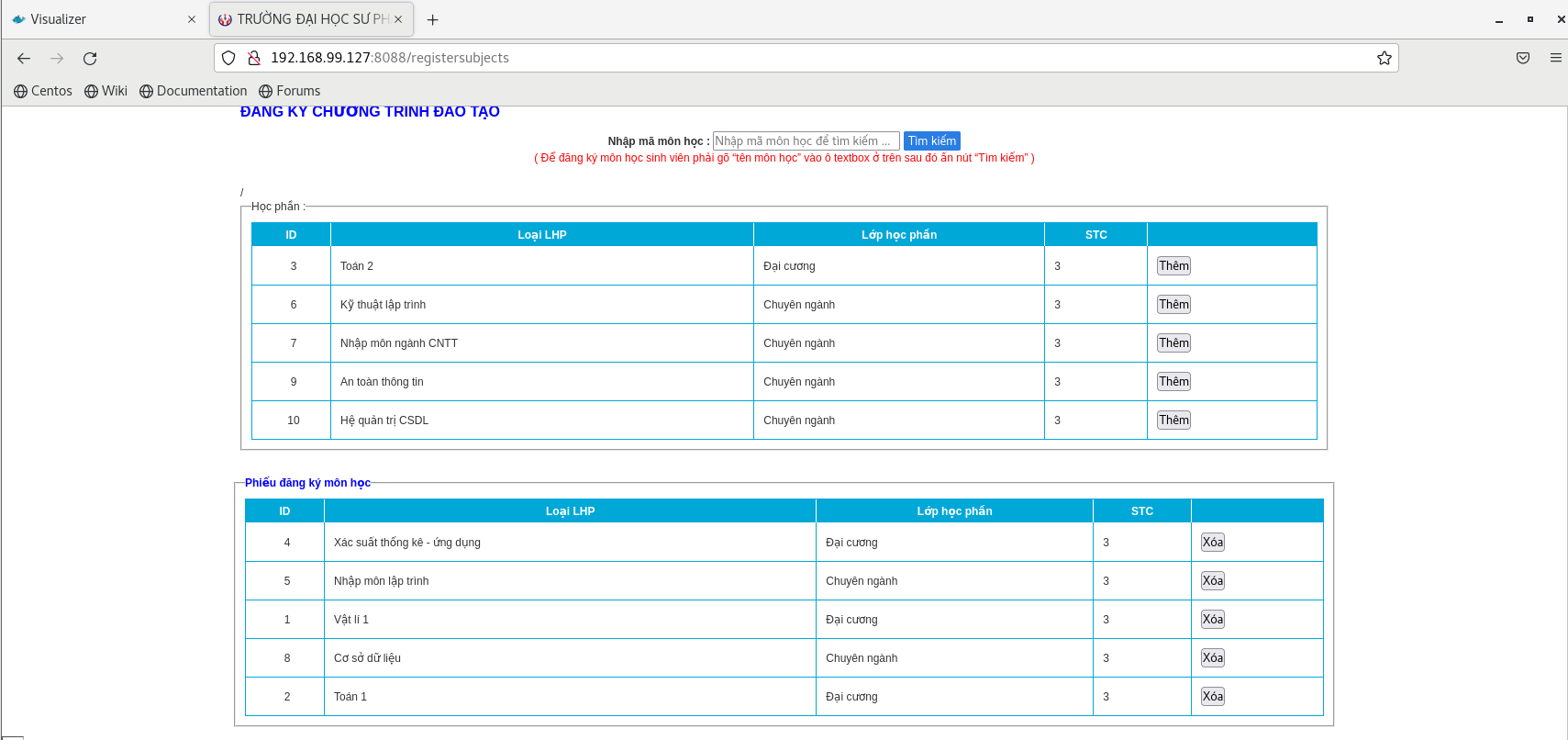
Xem các môn học có trong hệ thống



Xuất kết quả đăng ký của user này



Thêm/xóa môn vào danh sách đăng ký, Tìm kiếm môn học tại thanh tìm kiếm



TÀI LIỆU THAM KHẢO

*https://spring.io/guides/gs/spring-boot-docker/*

*https://hub.docker.com/r/toughiq/mariadb-cluster*

*https://hub.docker.com/r/prom/prometheus*

*https://hub.docker.com/r/google/cadvisor/*

*https://hub.docker.com/\_/redis*

*https://github.com/dockersamples/docker-swarm-visualizer*