TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

----- 🙡 🕮 🙣 -----



**BÁO CÁO PROJECT III**

***Triển khai cài đặt ELK lên Docker và đọc log từ API***

Giảng viên: **TS. Nguyễn Đức Toàn**

|  |  |
| --- | --- |
| Sinh viên thực hiện: | **Lê Hà Hưng** |
| MSSV: | **20183757** |

**Hà Nội, tháng 11 năm 2021**

Mục lục

[1. Tiến hành cài đặt ELK Stack lên docker 3](#_Toc87099227)

[**1.1.** **Cơ sở lý thuyết** 3](#_Toc87099228)

[**1.2.** **Cài đặt ELK Stack lên Docker** 4](#_Toc87099229)

[2. Đọc log của API 7](#_Toc87099230)

[**2.1.** **Đọc log trên console** 7](#_Toc87099231)

[**2.2.** **Ghi ra file log với Logback** 8](#_Toc87099232)

[3. Đọc log của hệ thống với ELK Stack 11](#_Toc87099233)

1. **Tiến hành cài đặt ELK Stack lên docker**
   1. **Cơ sở lý thuyết**

* **Khái niệm:**
  + Docker là một nền tảng cho developers và sysadmin để develop, deploy và run application với container. Nó cho phép tạo các môi trường độc lập và tách biệt để khởi chạy và phát triển ứng dụng và môi trường này được gọi là container. Khi cần deploy lên bất kỳ server nào chỉ cần run container của Docker thì application sẽ được khởi chạy ngay lập tức.
  + **Container:** Docker container là một run-time environment mà ở đó người dùng có thể chạy một ứng dụng độc lập.
  + **Docker image**: là một file bất biến - không thay đổi, chứa các source code, libraries, dependencies, tools và các files khác cần thiết cho một ứng dụng để chạy.
* **Các lệnh cơ bản trong docker:**
  + **Docker images:** xem các images đã cài đặt trên docker
  + **Docker run <images>:** chạy một container dựa trên một image mà ta có sẵn.
  + **Docker build:**
* **Tiến hành cài đặt docker và chạy chương trình đầu tiên:**
  + Môi trường thực thi: Windows
  + Tiến hành tải xuống và cài đặt Docker desktop từ trang chủ:
  + Chạy lệnh: docker run hello-world trên cmd để chạy chương trình hello-world: Text

    Description automatically generated
  + Ở Docker Desktop sẽ hiển thị những images đã và đang được cài đặt và sử dụng: A screenshot of a computer

    Description automatically generated with medium confidence
  1. **Cài đặt ELK Stack lên Docker**
* Sử dụng IDE IntellJ IDEA để chỉnh sửa.
* Khởi tạo 1 folder ELK
* Thêm file docker-compose.yaml.
  + Trong file, config như sau:
* version: "3"  
    
  networks:  
   backend:  
   driver: bridge  
  services:  
   elasticsearch:  
   image: docker.elastic.co/elasticsearch/elasticsearch:7.8.0  
   ports:  
   - 9200:9200  
   - 9300:9300  
   restart: unless-stopped  
   environment:  
   discovery.type: single-node  
   networks:  
   - backend  
   logstash:  
   image: docker.elastic.co/logstash/logstash:7.8.0  
   ports:  
   - 5000:5000  
   networks:  
   - backend  
   volumes:  
   - ./pipeline:/usr/share/logstash/pipeline  
   kibana:  
   image: docker.elastic.co/kibana/kibana:7.8.0  
   environment:  
   ELASTICSEARCH\_HOSTS: http://elasticsearch:9200  
   networks:  
   - backend  
   ports:  
   - 5601:5601
* Chạy lệnh docker compose up để tiến hành chạy Text

  Description automatically generated
* Khi đấy, folder pipeline sẽ được tạo thành. Text

  Description automatically generated with medium confidence
* Trong pipeline, tạo file logstash.conf để config cho logstash:

A screenshot of a computer

Description automatically generated with medium confidence

* Trong file logstash.conf này gồm có input, filter và output tương ứng với 3 bước xử lý chính. Để cho đơn giản hơn, em bỏ đi thành phần bộ lọc (filter), chỉ để lại input và output.
  + Input: tiếp nhận/thu thập dữ liệu sự kiện log ở dạng json
  + Output: tiếp nhận lưu trữ và hiển thị log.
* Sau khi config, ta bấm nút chạy trên IDE để khởi động dịch vụ. A screenshot of a computer

  Description automatically generated with medium confidence
* Kiểm tra ở Docker desktop, ta thấy ELK Stack đã được cài đặt thành công trên docker: A screenshot of a computer

  Description automatically generated with medium confidence

1. **Đọc log của API**

Em đã tự xây dựng 1 API đầy đủ 4 phương thức GET, POST, PUT, DELETE bằng java spring boot.

Link project: <https://github.com/hahungle2khust/learn-spring.git>

Khởi chạy project và test với swagger ui:

Graphical user interface

Description automatically generated

Graphical user interface, text, application, email

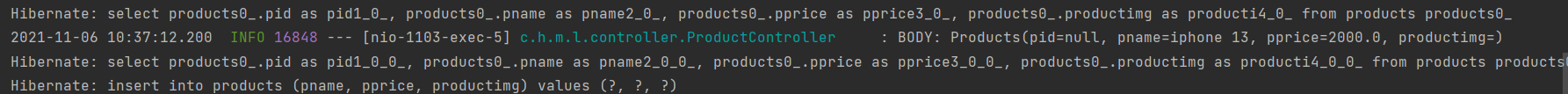
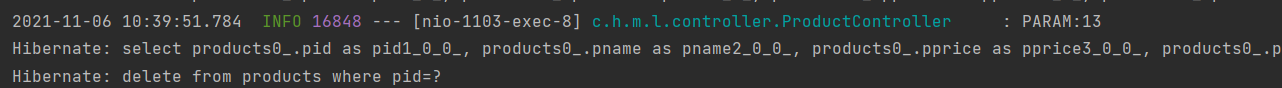
Description automatically generated

* 1. **Đọc log trên console**
* Sau khi tạo thành công API và test nó với Swagger UI hoặc Postman, em sử dụng apache Log4j để thêm 1 logger đánh dấu vào các phương thức Put, Get by ID và Delete để khi tiến hành gọi các request thì log sẽ được ghi ra console.
* Thêm vào phương thức put, get và delete:

//Đánh dấu log PUT  
*logger*.info("BODY: "+ products.toString());

Graphical user interface, text

Description automatically generated

* Khi tiến hành put trên swager ui, ở console sẽ trả về như sau :
* Ở hình trên, console đã ghi ra log thể hiện các giá trị của Products đã bị thay đổi và thời gian thực hiện thay đổi trên.
* Tương tự, khi get và delete thì console cũng ghi ta log để cho người xem biết được những thay đổi 
  1. **Ghi ra file log với Logback**
* **Cơ sở lý thuyết:**
  + Logback là thư viện dùng để đọc log trong java.
  + Sử dụng logback để có thể quản lý log dễ dàng hơn.
  + Logback giúp ta quan sát log của hệ thống được trên cả console và ghi ra file log.
  + Thành phần của logback gồm có :
    - Appender: là thẻ chịu trách nhiệm ghi log tới các vị trí được cấu hình.
    - Layout: Định ra cho log dễ dàng đọc hơn.
* **Cài đặt:**
  + Ta tiến hành thêm 1 file có tên Logback.xml ở thư mục resources của project:

A picture containing graphical user interface

Description automatically generated

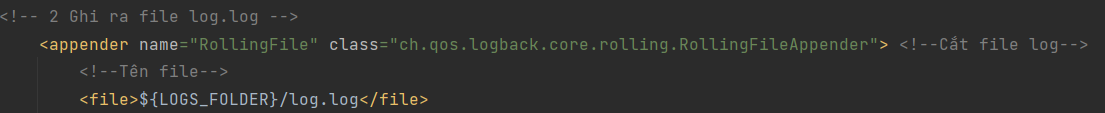
* + Trong file Logback.xml này, ta tiến hành config cho nó :
    - Để ghi ra file log: khi chạy project, 1 thư mục logs để chứa các log file sẽ được tạo ra. Graphical user interface, text

      Description automatically generated
* Ghi ra console :
  + Để ghi log ra console, để name của thẻ appender là « Console » :

Text

Description automatically generated

* + Kết quả khi ghi ra console : Text

    Description automatically generated with medium confidence
* Ghi ra file log :
  + Ghi ra file log, ở thẻ appender là "RollingFile":
  + Ở thẻ file, để đặt tên cho file log: 
  + Dùng thẻ layout để định dạng cho log được ghi ra.
  + Thẻ rollingPolicy:
    - Thẻ fileNamePattern: đặt tên đánh dấu log file được tạo ra ở thời gian nào
    - maxHistory: Số file log tối đa lưu trữ
    - totalSizeCap: dung lượng tối đa cho 1 log file.
  + Text

    Description automatically generated
  + A screenshot of a computer

    Description automatically generated with medium confidenceVà đây là kết quả khi chạy API: Có 1 thư mục tên logs được tạo ra ở project
  + Những file log trên sẽ ghi ra tất cả các log các ngày khi API được khởi chạy. A computer screen capture

    Description automatically generated with medium confidence

1. **Đọc log của hệ thống với ELK Stack**

* **Cơ sở lý thuyết:**
  + - So với việc đọc log qua console hay đọc qua file, nếu ở hệ thống nhỏ sẽ dễ dàng quản lý. Tuy nhiên nếu đưa vào microservice với nhiều API để quản lý, việc đọc log truyền thống là rất khó khăn và phức tạp với người vận hành cũng như bảo trì. Vì vậy ELK Stack sẽ giúp quản lý việc đọc và sử dụng log dễ dàng hơn, từ đó giúp nhanh chóng bảo trì và sửa chữa hệ thống.
* **Tiến hành cài đặt**
  + - Ở file Logback.xml trong hệ thống, ở thẻ appender, ta dùng name là "logstash\_TCP"
    - Ở mục tiếp theo, ta để đường dẫn là port đã cấu hình khi cài đặt ELK ở file logstash.conf.
    - **Text

      Description automatically generated**
    - Tiến hành khởi chạy API, và chạy ELK Stack trên Docker:

A screenshot of a computer

Description automatically generated with medium confidence

* + - * Trên trình duyệt: Graphical user interface, text, application, email

        Description automatically generated
      * Vào Stack Management: Graphical user interface, application

        Description automatically generated
      * Chọn index patterns là report\_service\_logs\* Graphical user interface, application

        Description automatically generated
      * Khi đấy log của API sẽ được ghi lên đây.
      * Ta có thể dễ dàng quan sát được log, cũng như quản lý thời gian của chúng. Graphical user interface, text, application

        Description automatically generated