**Bài tập thực hành**

**HỌC PHẦN : HỆ PHÂN TÁN**

**CHƯƠNG 1: TỔNG QUAN VÀ KIẾN TRÚC HPT**

**Họ và tên :** Nguyễn Hoàng Long

**Mssv :** 20183791

1. **Web server apache2**

**Câu hỏi 1 :** Đường dẫn đến file html chứa nội dung mặc định của trang web các bạn vừa xem là gì?

**Trả lời :** /var/www/html/index.html

**Câu hỏi 2 :** Cổng mặc định của dịch vụ www là gì ?

**Trả lời :** port 80

**Câu hỏi 3 :** Hãy giải thích quyền mang số 755 là gì ?

**Trả lời** :

* các quyền được biểu thị dưới dạng 3 số :
* Read =4
* Write=2
* Execute=1
* 755 có nghĩa là
* Owner = 7 =4+2+1 : có thể read,write,excute
* Group = 4+1 : có thể write,excute
* Other = 4+1 : có thể write,excute

**Câu hỏi 4 :** Bạn quan sát thấy nội dung gì sau khi gõ 2 địa chỉ trên giải thích ?

**Trả lời :** sau khi truy cập vào example.com và test.com em thấy màn hình website hiển thị “Success! The example.com virtual host is working!” và “Success! The test.com virtual host is working!”

**Giải thích :** vì khi tạo thành công 2 domain của 2 website tương ứng

DocumentRoot /var/www/html/example.com/public\_html/

Khi có domain example.com trỏ tới thì sẽ gọi source code trong trong folder /var/www/html/example.com/public\_html/

Tương tự cho test.com

**Câu hỏi 5 :** Thử truy cập từ các máy tính khác trong cùng mạng LAN vào 2 trang web đó

Trả lời : nội dung tương tự như trên máy gốc sẽ được hiển thị ở browser máy tính khác cùng mạng LAN

1. **Interface trong Java**

**Câu hỏi 6 :** Hãy tự viết một đoạn code để thực hiện 1 vòng lặp while sao cho nó sẽ nhận các số mà người dùng gõ và gửi về server, cho đến khi nào người dùng gõ ký tự rỗng rồi ấn enter. Gợi ý: hãy dùng lệnh sau để nhận xâu ký tự người dùng gõ vào:  
String message = scanner.nextLine();

String message = scanner.nextLine();  
boolean value=message.equals(" ");  
while (value==false){  
 out.println(message);  
 message=scanner.nextLine();  
}

**Câu hỏi 7 :** Vai trò của phương thức *run* là gì? Khi nào thì nó được gọi?

Bài làm

* Khi có 1 client kết nối đến đến server thì phương thức run() sẽ được gọi
* Vai trò của phương thức run() :
* Tạo luồng vào ra cho server
* Gửi lời chào đến client
* Nhận message từ client
* Sắp xếp chuỗi số được gửi từ client
* Trả kết quả cho client

1. **Kiến trúc Microservices**

**Câu hỏi 1 :** Hãy thực hiện gõ những lệnh tương tự như trên với 3 dịch vụ còn lại.

**Đối với dịch vụ catalog :**

>docker build --tag=microservice-kubernetes-demo-catalog microservice-kubernetes-demo-catalog

>docker tag microservice-kubernetes-demo-catalog  
longnh09/microservice-kubernetes-demo-catalog:latest

>docker push longnh09/microservice-kubernetes-demo-catalog

Text

Description automatically generated

**Đối với dịch vụ customer**

>docker build --tag=microservice-kubernetes-demo-customer microservice-kubernetes-demo-customer

>docker tag microservice-kubernetes-demo-customer  
longnh09/microservice-kubernetes-demo-customer:latest

>docker push longnh09/microservice-kubernetes-demo-customer

Text

Description automatically generated

**Đối với dịch vụ order**

>docker build --tag=microservice-kubernetes-demo-order microservice-kubernetes-demo-order

>docker tag microservice-kubernetes-demo-order  
longnh09/microservice-kubernetes-demo-order:latest

>docker push longnh09/microservice-kubernetes-demo-order

Text

Description automatically generated

**Câu hỏi 2 :** Vào trang web DockerHub và đăng nhập vào tài khoản của bạn. Bạn thấy những gì mới xuất hiện trên docker hub repository của bạn?

Đây là ảnh chụp lại docker hub repository của tài khoản của em

Graphical user interface, text, application, email

Description automatically generated

**Câu hỏi 3 :** Trạng thái (status) của các pods vừa mới tạo được là gì? Bây giờ, hãy chờ vài phút và gõ lại lệnh đó, trạng thái mới của các pods giờ đã chuyển thành gì?

Text

Description automatically generated

**Ảnh trang Oder Processing :**

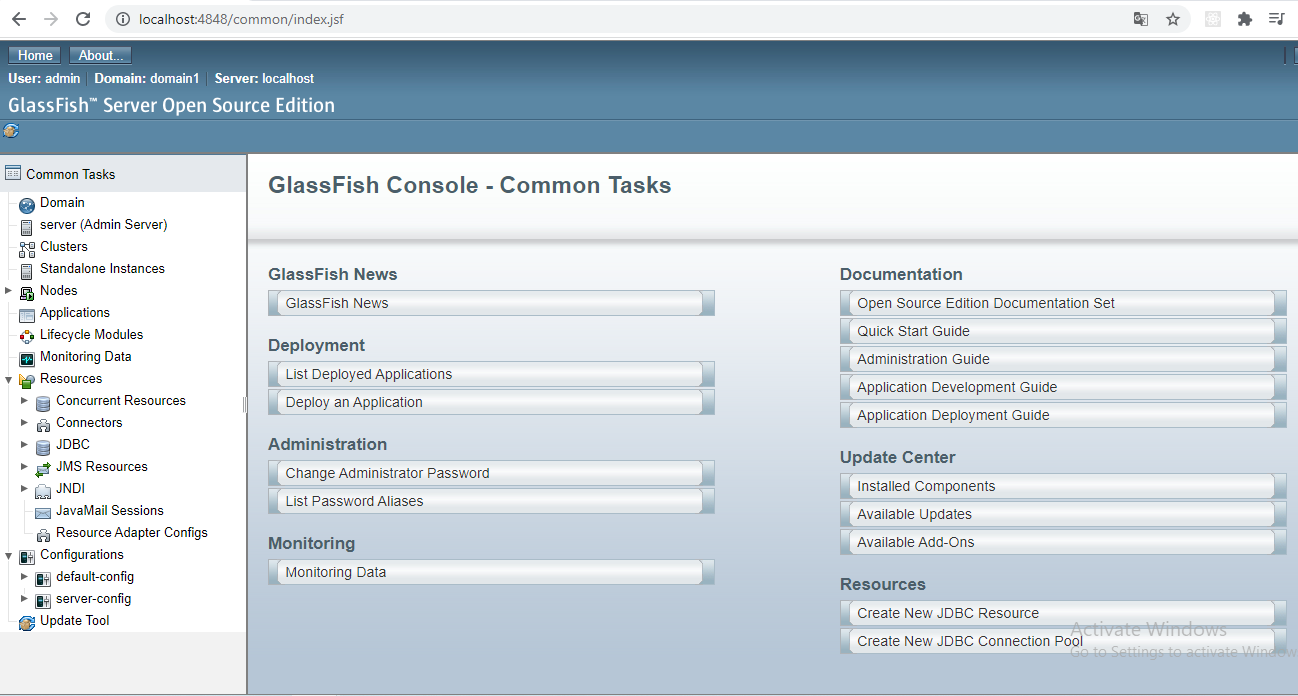
**Graphical user interface, text, application

Description automatically generated**

1. **Kiến trúc JMS và DDS**

**Câu hỏi 1 :** Giải thích vai trò của application server glassfish ?

* Glassfish là một applications server ( máy chủ ứng dụng)
* Glassfish cung cấp cơ sở để tạo nên ứng dụng phần mềm và một môi trường server để chạy những ứng dụng này.
* Glassfish còn đóng vai trò là một tập các thành phần mà admin có thể truy cập thông qua một API đã được định nghĩa trước.



Ảnh đã truy cập vào glashfish

**Câu hỏi 2 :** Tại sao lại phải tạo 2 JNDI như trên?

Khi xây dựng một ứng dụng JMS, ta cần phải tạo tài nguyên cấu hình cho nó. Cụ thể là Connection Factory và Destination Resource. Connection Factory là đối tượng được sử dụng để client tạo kết nối với một nhà cung cấp JMS. Connection Factory bao gói một tập các tham số cấu hình kết nối mà đã được định nghĩa bởi người quản lý. Trong một ứng dụng mô hình P/S, client sẽ gửi(publish) các message đến một Destination, mà ở đây gọi là Topic. Topic này hoạt động giống như một bảng tin. Các client subscribe với message của bảng tin này. Hệ thống sẽ đảm nhiệm phân tán các message đến từ các publisher của một Topic đến các subscriber của topic đó. Java Naming and Directory Interface (JNDI) là một giao diện cho phép những ứng dụng phân tán tìm kiếm dịch vụ một cách trừu tượng, không phụ thuộc vào tài nguyên. Việc tạo ra 2 JNDI như trên cho phép các JMS client có thể tìm thấy 2 đối tượng tài nguyên Connection Factory và Topic một cách dễ dàng qua cách sử dụng không gian tên JNDI.

**Câu hỏi 3** : Sau khi chạy thử chương trình Sender và Receiver, vận dụng lý thuyết kiến trúc hướng sự kiện đã học trên lớp để giải thích cơ chế chuyền và nhận thông điệp của Sender và Receiver?

Text

Description automatically generated

**Câu hỏi 4: So sánh JMS và DDS**