Região Académica I

◊ Luanda ◊ Bengo ◊

ESCOLA SUPERIOR PEDAGÓGICA DO BENGO

DEPARTAMENTO DE ENSINO, INVESTIGAÇÃO E EXTENSÃO DE CIÊNCIAS EXATAS

TRABALHO DE DESENVOLVIMENTO WEB

Estudante: **Elias Pedro António**

Curso: Ensino de Informática

Período: Tarde

Ano: 3º

Sala: 4

Docente

_____

Dr.: Euclides Catumbela

Caxito/2023

Responde as questões

1-O que é o Git?

R: O Git é uma ferramenta de controle de versão que permite que os desenvolvedores acompanhem e controlem as alterações em seu código-fonte. Ele facilita o trabalho colaborativo, permitindo que várias pessoas contribuam para o mesmo projeto sem conflitos

O Git foi criado em 2005 por Linus Torvalds, o famoso criador do kernel do sistema operacional Linux. Ele é um projeto de código aberto com manutenção ativa e é usado por muitos projetos de software, tanto comerciais quanto de código-fonte aberto.

2- O que é o github?

R: O GitHub é uma plataforma de hospedagem de código-fonte e arquivos com controle de versão usando o Git. Ele permite que programadores, utilitários ou qualquer usuário cadastrado na plataforma contribuam em projetos privados e/ou Open Source de qualquer lugar do mundo

O GitHub é um serviço baseado em nuvem que hospeda um sistema de controle de versão (VCS) chamado Git. Ele facilita o trabalho colaborativo, permitindo que várias pessoas contribuam para o mesmo projeto sem conflitos.
O GitHub também funciona como uma rede social, onde você pode explorar e procurar projetos que te interessam, seguir desenvolvedores, participar de discussões, revisar códigos, fazer pull requests e muito mais.

3- Diferença entre git e github

R: A diferença entre o Git e o GitHub é que o Git é um sistema de controle de versão de código aberto que permite aos desenvolvedores acompanhar e gerenciar as mudanças em seu código-fonte, enquanto o GitHub é uma plataforma de hospedagem baseada na web que usa o Git para armazenar, compartilhar e colaborar em projetos de software.

4- O que é o docker?

R: O Docker é uma tecnologia de conteinerização de código aberto que permite aos desenvolvedores criar e executar aplicações dentro de containers virtuais. Os containers são ambientes isolados que contêm tudo o que uma aplicação precisa para funcionar, como código, bibliotecas, dependências e configurações. Os containers são mais leves, rápidos e portáteis do que as máquinas virtuais, pois compartilham o mesmo sistema operacional do host e não precisam de um sistema operacional convidado.

O Docker facilita o desenvolvimento, a implantação e a distribuição de aplicações, pois permite empacotar o código e as dependências em uma imagem que pode ser executada em qualquer ambiente que suporte o Docker.

5- Principais componentes do Docker

R: Os principais componentes do Docker são:

• Docker Engine: é o software que cria e executa os containers, usando o sistema de controle de versão Git. Ele é composto por três elementos.

• Docker Daemon: é o processo que gerencia as imagens, os containers, as redes e os volumes do Docker. Ele também escuta e processa as solicitações da API do Docker.

• Docker Engine REST API: é a interface que permite aos clientes se comunicarem com o daemon, usando comandos HTTP.

• Docker Client: é a ferramenta que permite aos usuários interagirem com o daemon, usando uma linha de comando ou uma interface gráfica.

• Docker Hub: é um serviço de hospedagem e compartilhamento de imagens do Docker, que permite aos usuários armazenar e distribuir seus containers. Ele também oferece recursos como integração contínua, automação, colaboração e segurança.

• Docker Compose: é uma ferramenta que permite definir e executar aplicações multi-container, usando um arquivo YAML. Ele facilita a configuração, o gerenciamento e a escalabilidade dos containers.

• Docker Swarm: é uma ferramenta que permite agrupar e gerenciar vários hosts Docker como um único cluster. Ele facilita a orquestração, o balanceamento de carga, a tolerância a falhas e a descoberta de serviços dos containers.

6- Qual é a utilidade prática do docker?

R: A utilidade prática do Docker é que ele permite criar e executar aplicações dentro de containers virtuais, que são ambientes isolados que contêm tudo o que uma aplicação precisa para funcionar, como código, bibliotecas, dependências e configurações. Isso traz várias vantagens, como:

• Consistência: as aplicações funcionam da mesma forma em qualquer ambiente que suporte o Docker, sem problemas de compatibilidade ou dependências.

• Portabilidade: as aplicações podem ser facilmente transferidas de um sistema para outro, sem a necessidade de instalar ou configurar nada.

• Desempenho: os containers são mais leves, rápidos e eficientes do que as máquinas virtuais, pois compartilham o mesmo sistema operacional do host e não precisam de um sistema operacional convidado.

• Escalabilidade: as aplicações podem ser facilmente dimensionadas, ativando mais containers conforme a demanda, usando ferramentas de orquestração como o Kubernetes ou o Docker.