PROVA Regular I

Disciplina: Tópicos Avançados em Desenvolvimento Web Data: 26/03/2022

Professor: Bruno Alves Reis Nascimento Turma: INFO084

Aluna: Bárbara Eduarda Souza Menezes Matrícula: 20104019

1º) 1- Git Clone = é usado para copiar um repositório Git existente em um novo diretório local.

2- Git Checkout = é usado para alternar entre as versões de branches e restaurar arquivos.

3- Git Add = é usado para adicionar alterações no repositório.

4- Git Commit= é usado para comentar uma alteração adicionada.

5- Git Push = é usado para enviar o conteúdo do repositório local para um repositório remoto.

2º) Um sistema de controle de versão tem como finalidade de gerenciar diferentes versões de um documento, oferecendo uma maneira inteligente e eficaz de organizar um projeto, pois é possível acompanhar o histórico de desenvolvimento, desenvolver paralelamente, customizar uma versão, incluir outros requisitos, finalidades especificas, sem mexer no projeto principal ou resgatar o sistema em um ponto que estava estável.

Os arquivos do projeto ficam armazenados em um repositório e o histórico de suas versões é salvo nele. Os desenvolvedores podem acessar e resgatar a última versão disponível e fazer uma cópia local, na qual poderão trabalhar em cima dela e continuar o processo de desenvolvimento. A cada alteração feita, é possível enviar novamente ao servidor e atualizar a sua versão a partir outras feitas pelos demais desenvolvedores, além do mais é possível identificar quem fez cada alteração no projeto.

3º) Exemplo da Problemática:

Dois desenvolvedores estão trabalhando na mesma branch de um repositório remoto.

O desenvolvedor 1 faz uma alteração na linha 15 e salva a ação realizada dando um novo commit local. Tendo em mente que em seguida esse desenvolvedor irá usar o git pull para receber o repositório remoto, mesclar o código na sua máquina para então poder atualizar o repositório com as suas próprias alterações. Ao executar esse comando aparece uma mensagem de Conflito pois o desenvolvedor 2 executou um commit seguido de um push antes do desenvolvedor 1, ambos alteraram a mesma linha (15) do código de forma diferente.

Resolução:

O primeiro passo é localizar os arquivos conflituosos. Usando o git status é possível visualizar os arquivos que foram modificados pelo merge ou fazer uma busca geral no projeto.

O Git gera uma marcação com as duas versões do mesmo código: A versão existente, e a versão que o desenvolvedor está tentando mesclar. Para resolver é preciso alterar o arquivo para que contenha apenas uma versão, apagando o código incorreto e removendo as marcações de conflito. Ou seja, escolher qual versão do código será deletada.

4º) O Node.js se caracteriza como um ambiente de execução JavaScript. Com ele, o usuário pode criar aplicações sem depender do *browser* para isso. Com alta capacidade de escalabilidade, boa flexibilidade, arquitetura e baixo custo.

Em um servidor tradicional, como os recursos são limitados, as novas requisições só seriam tratadas depois que os recursos estiverem liberados, o que pode atrasar o usuário.

O Node.js utiliza apenas um thread que executa o código da aplicação. Dessa maneira, menos recursos computacionais são exigidos, pois não é necessário criar uma nova thread para cada requisição recebida, chamada de *Event Loop*, que cria novos eventos a cada requisição recebida. Ações de entrada e saída, por exemplo, não precisam aguardar a finalização de uma operação para que outra seja iniciada.

5º) HTTP é sigla de HyperText Transfer Protocol que em português significa "Protocolo de Transferência de Hipertexto". É um protocolo de comunicação entre sistemas de informação que permite a transferência de dados entre redes de computadores, principalmente na Internet.

Como Funciona:

O HTTP usa a arquitetura cliente-servidor. Clientes e servidores se comunicam trocando mensagens individuais (ao contrário de um fluxo de dados). As mensagens enviadas pelo cliente, geralmente um navegador da Web, são chamadas de solicitações *(requests)*, ou também requisições, e as mensagens enviadas pelo servidor como resposta são chamadas de respostas *(responses)*.

1- O usuário clica em um link do browser.
2- O browser formata a solicitação e faz o envio ao servidor.
3- O servidor encontra a página solicitada.
4- O servidor formata a resposta e envia para o browser.
5- O browser resgata o HTML e compila em formato visual para o usuário.

POST: criar dados no servidor;

GET: leitura de dados no host;

DELETE: excluir as informações;

PUT: atualizações de registro;

PATCH: modifica parcialmente.