Aluno: Diego Aquino Souza

Atividade: CI/CD e Git

1. O que é CI/CD e qual é o seu objetivo? CI/CD é a sigla para Integração Contínua e Entrega Contínua. O objetivo é automatizar testes e a entrega de programas, para que as atualizações sejam mais rápidas e seguras.

2. Quais são os benefícios do CI/CD?
- Diminui erros.
- Dá um retorno rápido se algo der errado.
- Permite entregas mais frequentes e seguras.
- Garante mais confiança no processo.

3. Quais são as diferenças entre Integração Contínua (CI) e Entrega Contínua (CD)?
- **CI:** Automatiza o teste do código sempre que ele é alterado.
- **CD:** Automatiza a entrega dessas alterações para o local onde o programa é usado.

4. Como o CI/CD pode melhorar a qualidade do software?
- Encontra problemas (bugs) mais cedo com testes automáticos.
- Evita que os problemas se acumulem no código.
- Mantém o código consistente em diferentes ambientes.

5. Como o CI/CD pode melhorar a colaboração entre os desenvolvedores?
- Permite que os desenvolvedores façam pequenas mudanças e enviem-nas com mais frequência.
- Ajuda a reduzir conflitos de código.
- Facilita a revisão do código por outras pessoas.

6. Como o controle de versão está relacionado ao CI/CD? Ferramentas como o Git disparam o processo de CI/CD toda vez que um desenvolvedor salva ou envia novas mudanças, mantendo tudo organizado e fácil de rastrear.

7. Quais ferramentas e tecnologias são comumente usadas em uma implementação de CI/CD?
- **Servidores CI/CD:** Jenkins, GitHub Actions, GitLab CI.
- **Controle de versão:** Git.
- **Containers:** Docker.

8. O que é o Git e por que é usado em desenvolvimento de software? Git é um sistema que rastreia as mudanças no código de um projeto. Ele é usado para que as equipes possam trabalhar juntas e manter um histórico completo do que foi alterado.

9. Como o Git difere de outros sistemas de controle de versão? O Git é distribuído, o que significa que funciona mesmo sem internet. Ele também é muito rápido e bom para gerenciar diferentes versões de um mesmo projeto.

10. O que é um repositório do Git? É o local onde o projeto e seu histórico ficam guardados. Pode ser no seu computador ou em um servidor na internet, como o GitHub.

11. Quais são os principais comandos do Git?

- git init: Começa um novo projeto Git.
- git clone: Copia um projeto que já existe.
- git add: Prepara os arquivos para serem salvos.
- git commit: Salva as mudanças no histórico.
- git push: Envia as mudanças para a internet.
- git pull: Pega as atualizações da internet.

12. O que é um commit no Git? É um "salvamento" de mudanças que você fez no código, com uma mensagem explicando o que foi mudado.

13. O que é um branch no Git e como ele é usado? Um *branch* é uma versão separada do seu código. Ele serve para que você possa trabalhar em uma nova função sem bagunçar o código principal.

14. O que é um merge no Git e como é realizado? É quando você junta as mudanças de um *branch* com o código principal.

15. O que é um conflito de merge no Git e como ele é resolvido? Acontece quando duas pessoas mudam a mesma linha de código em diferentes *branches*. Você precisa resolver o conflito manualmente, decidindo qual das mudanças manter.

16. O que é o GitHub e como ele se relaciona com o Git? GitHub é um site que usa o Git para hospedar projetos. Ele facilita a colaboração entre desenvolvedores.

17. Quais são as melhores práticas para trabalhar com Git em equipes?

- Fazer *commits* pequenos e frequentes, com mensagens claras.
- Usar *branches* para novas funções.
- Revisar o código dos colegas.