Git e Github

- **O que é Git?**

É um sistema distribuído opensource de controle de versão, Seu objetivo é registrar todas as mudanças feitas no código-fonte de um projeto, evitando que algo importante se perca no meio do caminho.

(opensource: software de livre utilização/ controle de versão: armazena conteúdo e mantém um histórico das alterações feitas nele/ Sistema distribuído: cada desenvolvedor tem uma cópia completa do repositório com todo o histórico de mudanças, permitindo trabalho independente de uma rede ou servidor central.)

- **Git Status**

Exibe as condições do diretório de trabalho e da área de staging, ou seja, permite inspecionar quais alterações foram despreparadas, quais não foram e quais arquivos não estão sendo monitorados pelo Git. Ao executar esse comando, conseguimos saber o status completo do repositório.

- **Commits**

É o conjunto de alterações feitas no projeto

1. git add [nome do arquivo] ou git add . (todos arquivos)
2. git commit -m “[mensagem]”

- **Comandos mais utilizados**
  1. git initil (cria um repositório vazio ou transforma uma pasta que você já tem e que não está com controle de versão em um repositório)
  2. git clone (comando para baixar o código-fonte existente de um repositório remoto)
  3. git branch <nome-da-branch> (criar uma nova branch local)
  4. git push -u <remote> <nome-da-branch> (upar a nova branch para o repositório remoto)
  5. git branch || git branch --list (ver as ramificações)
  6. git branch -d <nome-da-branch> (deleta a branch)
  7. git branch -d <nome-da-branch> (mudar de uma branch para outra ou então verificar arquivos e commits)
  8. git status (fornecer algumas informações importantes sobre a branch em que você estiver no momento)
  9. git diff (comparação nas fontes de dados Git e mostra quais linhas foram adicionadas e removidas.)
  10. git add <arquivo> (adicionar apenas um arquivo)
  11. git add -A (adicionar todas os arquivos modificados de uma só vez)

12. git commit -m "mensagem explicando a mudança no código" (ponto de verificação no processo de desenvolvimento)
13. git push <remote> <nome-do-branch>(enviar as alterações para o servidor remoto)
14. git push -u origin <nome-do-branch> (caso a branch for criada recentemente, também precisará fazer upload do branch)
15. git pull <remote> (usado para obter atualizações do repositório remoto.)
16. git revert 'número do hash' (maneira segura de desfazer os commits/ obs para conseguir o número do hash: git log--oneline)
17. git merge <nome-da-branch> (vai mesclar, no seu repositório local, todas as alterações feitas.)
18. git stash (salvar as alterações sem commit)
19. git stash list (ver todas as stashes que você guardou)
20.