***Aula 03 – Configurando o GIT***

***Git config –global user.name “NOME”: Configura o nome de usuário;***

***Git config –global user.email “EMAIL”: Configura o email;***

***Git config –global core.editor vscode: Dia ao sistema qual editor vai usar;***

***Git config use.name: Mostra qual nome esta registrado a maquina;***

***Git config use.email: Mostra qual email esta registrado a maquina;***

***Git config --list: Mostra todas as informações registradas na maquina;***

***Aula 04 – Iniciando um repositório***

***Git init: Inicia um repositório;***

***Aula 05 – Branch, README e Commit***

***Git status: Verifica se há alterações no projeto;***

***Git add NomeArquivo: Adiciona o arquivo ao git;***

***Git add -A: Adiciona todos os arquivos ao git;***

***Git commit -m “MENSAGEM”: Faz o commit na aplicação;***

***Git log: Mostra todos os commit realizados na aplicação;***

***Aula 06 – Revertendo Modificações***

***Git branch: Mostra todos branch no projeto;***

***Git commit -am “MENSAGEM”: Aqui commita e já adiciona os elementos criados;***

***Git reset --soft NumeroVersao: Volta para versão escolhida, com as outras alterações preparadas sem commitar;***

***Git reset --mixed NumeroVersao: Volta a versão escolhida, parecida, so que se precisa dar o add nos arquivos;***

***Git reset --hard NumeroVersao: Volta a versão escolhida, sem as modificações feitas após a versão selecionada;***

***Aula 07 – Trabalhando com diferentes Branches***

***Git branch NomeBranch: Cria uma nova branch. Copia todos os arquivos do branch anterior e inicia o seu projeto dali;***

***Git checkout NomeBranch: Entra na branch mencionada;***

***Aula 08 – Diferença entre arquivos***

***Git diff: Mostra as modificações feita nos arquivos;***

***Git diff –name-only: Mostra os nomes dos arquivos modificados;***

***Git diff NomeArquivo: Mostra apenas o que foi alterado no arquivo em si;***

***Git checkout HEAD(branch atual\*) -- : Volta o arquivo para versão antes das modificações;***

***Aula 10 – Conectando repositório Local com o Remoto***

***Ssh-keygen -t rsa -b 4096 -C “Email”: Gera uma chave de acesso segura do computador para a conta do git, permitindo o envio de dados;***

***Git remote add origin*** [***https://github.com/USUARIO /REPOSITORIO***](https://github.com/USUARIO%20/REPOSITORIO) ***: Faz a conexão entre o repositório local e remoto;***

***Git remote: Diz qual repositório esta conectado;***

***Git remote -v: Mostra mais informações;***

***Fetch: Capacidade de puxar alterações feitas por outros usuários;***

***Push: Capacidade de transferir do meu repositório local para o repositório remoto;***

***Git push -u DESTINO BranchDeEnvio;***

***Aula 11 – Fazendo alterações no repositório remoto***

***Git pull DESTINO BRANCH: Puxa todos arquivos;***

***Git push DESTINO BRANCH: Envia os arquivos para o repositório remoto;***

***Aula 13 – Revertendo sem perder o código (Revert)***

***Git revert –no-edit CodigoCommit: Reverte as alterações desse commit;***

***Aula 14 – Deletando branches locais e remotos***

***Git push origin :NomeBranch: Deleta a branch remoto;***

***Git branch -D NomeBranch: Dele a branch local (Precisa estar em outra branch);***

***Aula 15 – Puxando alterações de outras pessoas (pull)***

***Git pull DeQuemVaiPuxar NomeBranch: Pega os dados do repositório remoto e transfere para o repositório local;***

***É fundamental antes de usar o push usar o pull, pois assim ele busca todas as alterações feita no repositório e tras para o seu. Assim deixando o seu repositório atualizado e pronto pra usar o push.***

***Aula 16 – Clonando repositórios remotos***

***Git clone UrlDoRepositorio: Clona um repositório do git;***

***Aula 17 – Contribuindo com outros repositórios (Fork /pull request)***

***Fork: Seria como clonar o projeto;***

***Pull request: e solicitação de envio de contribuição para o proprietário do projeto;***