**Passo básicos de GIT**

**Acesse o GITBash**

1. **git config** –global user.name (Cadastro o nome do usuário)
2. **git config** –global user.email (Cadastra o e-mail padrão do usuário)
   1. Nota de observação: modificando o ponto de acesso inicial.
      1. Abrir as propriedades do GITBash
      2. Modificar o endereço da página destino para a desejada, prestar a atenção se não existe o comando no final –cd-to-home, se estiver deletar essa extensão que irá abrir na página destino selecionada.
3. **git init** (repositório padrão) - master
   1. **git init –bare** (repositório local para várias maquinas trabalharem)
4. **git add** (Comando para adicionar modificações)
   1. **git add** ‘arquivo especifico’ (index.php) – Adiciona somente o arquivo selecionado
   2. **git add** ‘extensão determinada’ (.php; .htm; .html; .txt) – Adiciona somente os arquivos com essa determinada extensão
   3. **git add .** – Adiciona todos os arquivos da pasta onde está trabalhando
5. **git status** – Mostra todos os arquivos que foram adicionados e removidos no git add.
6. **git commit -m** ‘Mensagem do commit – o que você fez nessa alteração’ – Adiciona os arquivos da área intermediaria, os arquivos que você preparou para teste.
   1. **git commit –amend -m** ‘Mensagem da modificação’ – Função de editar o mesmo commit, sem criar um novo, onde adicionamos ou eliminamos um arquivo que esquecemos.
      1. Fazer todas as etapas normais de um commit, para adicionar a alteração do commit desejada.
7. **.gitignore** – arquivo utilizado para esconder determinada pastas do repositório do commit.
   1. Nota de observação:
      1. Criação do .gitignore
         1. Abrir o bloco de notas, e salvar a extensão com o final .gitignore, e selecionar a opção todos os arquivos.
         2. Para que o commit ignore determinados arquivos, você deve adicionar dentro do .gitignore o nome do arquivo que deve ser ignorado.
8. **git diff** – Mostra as modificações do repositório
9. **git log** – Apresenta as alterações realizadas
   1. **git log -p** – Detalha todas as alterações realizadas pelos os usuários, mostrando desde arquivos adicionados, até o conteúdo existente dentro deles.
      1. **Git log -p -1** ‘Determine qual a última alteração que você quer visualizar’ – Mostra somente o ultimo commit realizado, de maneira detalhada, podendo mudar de acordo com a variação numérica que determinar. (Ex: se escolher -2 visualiza as duas últimas modificações).
   2. **git log –pretty=oneline** – Mostra os últimos dados do commit realizado
10. **gitk** – Abre o painel gráfico do GITBash, onde permitir visualizar de maneira mais didática as alterações (commits) realizados.
11. **git reser HEAD** ‘nome do arquivo’ – retira o arquivo que não deseja ser adicionado ao commit
12. **git checkout --** ‘nome do arquivo’ – Volta as funções anteriores do arquivo
    1. **git checkout** ‘escolher a tag desejada’
13. **git tag** – Determinar qual é a versão que está trabalhando
    1. **git tag -a** ‘nomeia a versão’ - m ‘Mensagem sobre a versão trabalhada’
       1. Nota de observação:
          1. **git log –pretty-oneline**: para ter acessos a todas as versões e a chave de acesso de identificação
          2. **git tag -a** (determina a versão) (chave de acesso do log) - m “Mensagem exemplificando o que foi modificado”
    2. **git show** – Mostra a tag que está trabalhando
    3. **git checkout ‘**local que pretender ir**’** – Comando dado para voltar ao branch original, o comando master
    4. **git tag -d** ‘Versão da tag que pretende deletar’
14. **git remote add origin** ‘endereço de hospedagem’ – Local remoto onde vai hospedar as versões trabalhadas
    1. **git push origin master** – encaminha o commit para local remoto já determinado
15. **git branch** ‘o que pretende fazer’ – Abre uma versão de teste, quando você pretende fazer uma modificação teste sem prejudicar a versão original que está em funcionamento.
16. **git merge** – faz com que a versão teste (branch) onde está trabalhando passe para a versão final (master)