O que é um Repositório?

* É onde o código será armazenado;
* Na maioria das vezes cada projeto tem um repositório;
* Quando criamos um repositório estamos iniciando um projeto;
* O repositório pode ir para servidores que são especializados em gerenciar repos, como: Github e Bitbucket;
* Cada um dos desenvolvedores do time pode baixar o repositório e criar versões diferentes em sua máquina;

Criando Repositórios

* Para criar um repositório utilizamos o comando: git init
* Desta maneira o git vai criar os arquivos necessários para inicializá-lo;
* que estão na pasta oculta .git; ls -la
* Após este comando o diretório atual será reconhecido pelo git como um projeto e responderá aos seus demais comandos;
* git init
* git config –global user.name””
* git config –global user.email””
* git config –global defaultBranch main
* git status
* git add .
* git commit -m “”
* git remote add origin
* git push -u origin main

Verificando mudanças de projeto

* As mudanças de projeto podem ser verificadas por: git status
* Este comando é utilizado muito frequentemente
* Aqui serão mapeadas todas as alterações do projeto
* como: arquivos não monitorados e arquivos modificados
* podemos também dizer que é a diferença do que já está enviado ao servidor ou salvo no projeto

Adicionando arquivos ao projeto

* Para adicionar arquivos novos a um projeto utilizamos: git add
* Podemos adicionar um arquivo específico como também diversos de uma vez só
* Somente adicionando arquivos eles serão monitorados pelo git
* Ou seja, se não adicionar ele não estará no controle de versão
* É interessante usar esse comandos de tempos em tempos para não perder algo por descuido

Salvando alterações do projeto

* As alterações salvas do projeto são realizadas por: git commit
* Podemos commitar arquivos específicos ou vários de uma vez com a flag -a
* É uma boa prática enviar uma mensagem a cada commit, com as alterações que foram feitas
* A mensagem pode ser adicionada com a flag -m

Enviando código ao repo remoto

* Quando finalizamos uma funcionalidade nova, enviamos o código ao repositório remoto, que é o código-fonte.
* Esta ação é feito pelo git push
* Após esta ação o código do servidor será atualizado baseando-se nesse código local enviado

Recebendo as mudanças

* é comum também ter que sincronizar o local com as mudanças do remoto
* Esta ação é feita pelo git pull
* Após o comando serão buscadas atualizações, se encontradas elas serão unidas ao código atual existente na nossa máquina

Clonando repositórios

* O ato de baixar um repositório de um servidor remoto é chamado de clonar repositório;
* Para esta ação utilizamos git clone
* Passando a referência do repositório remoto
* Este comando é utilizado quando entramos em um novo projeto, por exemplo
* git clone <link> <novo nome para o repo>

Removendo arquivos do repo

* Os arquivos podem ser deletados da monitoração do git
* O comando para deletar é git rm
* Após deletar um arquivo do git ele não terá mais suas atualizações consideradas pelo git
* Apenas quando for adicionado novamente pelo git add

Histórico de alterações

* Podemos acessar um log de modificações feitas no projeto
* O comando para este recurso é git log
* Você receberá uma informação dos commits realizados no projeto até então

Renomeando arquivos

* com o comando git mv podemos renomear um arquivo
* O mesmo também pode ser movido para outra pasta
* E isso fará com que este novo arquivo seja monitorado pelo git
* O arquivo anterior é excluído
* exemplo mover arquivo: git mv rodape.css css/rodape.css
* exemplo renomear arquivo: git mv css/banner.css css/banner2.css

Desfazendo alterações

* O arquivo modificado pode ser retornado ao estado original
* o comando utilizado é o git checkout
* Após a utilização do mesmo o arquivo sai do staging
* Caso não seja feita uma próxima alteração, ele entra em staging novamente.
* git checkout css/syle.css (esse comando cancela as alterações feitas)
* substituirá o conteúdo do arquivo com a versão mais recente commitada no repositório.

Ignorando arquivos no projeto

* Uma técnica muito utilizada é ignorar arquivos do projeto
* Devemos inserir um arquivo chamado .gitignore na raiz do projeto
* Nele podemos inserir todos os arquivos que não devem entrar no versionamento
* isso é útil para arquivos gerados automaticamente ou arquivos que contém informações sensíveis
* primeiro adicionar o arquivo ou pasta no .gitignore e depois criar a mesma
* node\_modules/\* = todos arquivos dentro dessa pasta serão ignorados

Resetando uma branch / Desfazendo todas as alterações

* com o comando git reset podemos resetar as mudanças feitas
* Geralmente é utilizado com a flag –hard
* Todas as alterações commitadas e também as pendentes serão excluídas
* git reset –hard origin/main