Orientações para utilizar o Git/GitHub

Conceitos:

O **Git** é um sistema de controle de versão distribuído que permite rastrear mudanças no código, facilitando o trabalho em equipe e o gerenciamento de projetos. Ele possibilita criar diferentes versões do código, chamadas de **branches** (ramificações), permitindo que desenvolvedores trabalhem em funcionalidades separadas sem interferir no código principal.

Cada alteração no código pode ser salva por meio de um **commit**, que registra uma versão do projeto com uma mensagem descritiva sobre as mudanças feitas. Para compartilhar essas mudanças com o repositório remoto, usamos o comando **push**.

Se várias pessoas estiverem trabalhando no mesmo projeto, pode ser necessário atualizar seu código local antes de enviar mudanças. Isso é feito com o comando **pull**, que sincroniza seu repositório local com as últimas alterações do repositório remoto. Caso haja diferenças entre o código local e o remoto, o Git pode precisar realizar um **merge**, que combina as alterações de diferentes fontes.

O **GitHub** é uma plataforma online que hospeda repositórios Git e facilita a colaboração remota entre desenvolvedores. Ele oferece recursos como pull requests, que permitem sugerir e revisar mudanças antes de adicioná-las ao código principal, issues, para rastrear bugs e tarefas, e integração com ferramentas de CI/CD para automação de testes e deploys.

Com esses recursos, o Git e o GitHub tornam o desenvolvimento mais organizado, seguro e eficiente, permitindo um fluxo de trabalho colaborativo e versionado.

Orientações para o projeto:

O projeto SAMA terá apenas uma **branch**, ou seja, a ramificação principal (**main**). Por padrão, no git a branch principal vem com o nome de master, portanto é necessário alterar seu nome.

O arquivo **.gitignore** está presente no repositório para evitar arquivos que são criados automaticamente ou que não têm necessidade de estar ali.

O repositório deverá obrigatoriamente ter um **readme.md**, contando com 5% da nota da documentação.

Todos os **commits** deverão seguir um padrão de mensagem, para que os avaliadores possam observar a pessoa que executou o comando, quando e o quê.

Será obrigatório conter um diretório apenas para a documentação, ou seja, toda a documentação deverá aparecer no GitHub além do readme.md.

Passo a passo:

Primeiramente, antes de começar a utilizar o repositório no GitHub da equipe, a pessoa deve analisá-lo. Link:

https://github.com/Equipe-DotTec/SAMA-API

Para começar a utilizar o GitHub, tenha certeza que na sua máquina está instalado o Git, e prefira utilizar os comandos sempre no **Git Bash**, porém saiba que é possível utilizar também pelo cmd, powershell, etc.

Provavelmente nas primeiras vezes será requisitado um login ou algo do tipo. Inicialmente execute dois comandos: "**git config --global user.name '<seu_nome>'**" e "**git config --global user.email '<seuemail@example.com>'**"

Dentro do diretório escolhido onde ficará o projeto na sua máquina digite o comando "**git clone** https://github.com/Equipe-DotTec/SAMA-API.git". Com isso ele clonará o repositório do GitHub, puxando todas as branchs e os commits, ou seja, todo o histórico do projeto.

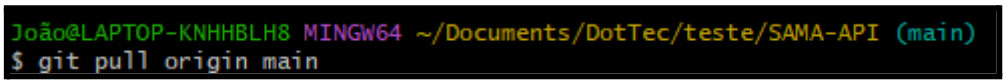
```
João@LAPTOP-KNHHBLH8 MINGW64 ~/Documents/DotTec/teste
$ git clone https://github.com/Equipe-DotTec/SAMA-API.git
```

Se tudo ocorrer bem, ele escreverá:

```
João@LAPTOP-KNHHBLH8 MINGW64 ~/Documents/DotTec/teste
$ git clone https://github.com/Equipe-DotTec/SAMA-API.git
Cloning into 'SAMA-API'...
remote: Enumerating objects: 52, done.
remote: Counting objects: 100% (52/52), done.
remote: Compressing objects: 100% (31/31), done.
remote: Total 52 (delta 12), reused 48 (delta 8), pack-reused 0 (from 0)
Receiving objects: 100% (52/52), 268.61 KiB | 4.80 MiB/s, done.
Resolving deltas: 100% (12/12), done.
```

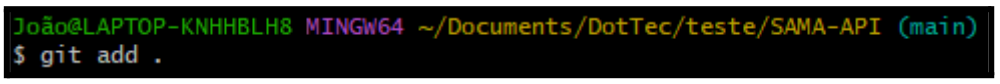
Dessa forma, o projeto na versão atual estará na sua máquina. Para adicionar uma alteração no repositório, você seguirá alguns passos:

1. Antes de executar o commit, lembre-se de sempre checar se alguém realizou alguma alteração enquanto modifica o código, pois estaremos trabalhando com apenas uma branch. Se existir, pegue as novas modificações do GitHub utilizando o comando "**git pull origin main**". Para a segurança do projeto, sempre antes de fazer um commit, faça um pull.



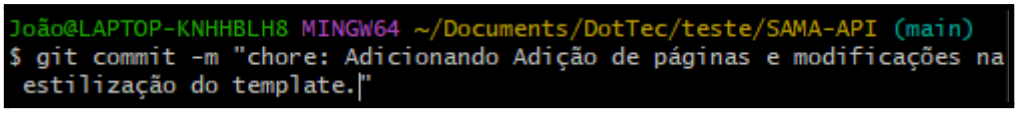
Se houver alguma alteração ele fará um merge com o seu atual repositório, então fique atento nas modificações.

2. Dê o comando "**git add .**". Dessa maneira, ele adicionará todas as suas modificações em um local de espera até que seja dado o commit.



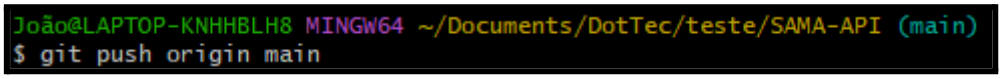
Dica: Sempre dê um "**git status**" para saber a situação de modificações e adições dos arquivos!

3. Execute o commit com o comando "**git commit -m '<mensagem_do_commit>**'". Lembre-se que teremos um padrão de mensagens para os commits.



Dica: Fique atento nas mensagens que o git retorna!

4. Com o commit feito sem nenhum erro, faça o upload para o GitHub com o comando "**git push origin main**".



Assim ele fará um upload que poderá ser visto o progresso pelo próprio bash.

5. Por fim, avise o restante do grupo sobre as modificações feitas. Comunicação é fundamental.