**Git e GitHub**

**GitHub é uma plataforma de gerenciamento de projetos e versionamento de código**, criada para desenvolvedores (mas não precisa ser usado apenas por desenvolvedores). Na plataforma, você pode **hospedar** as versões de seus trabalhos, códigos e/ou projetos e compartilha-los, se quiser, com o resto do globo, além é claro de poder contribuir com projetos de terceiros.

O Git (Desenvolvido em Linux) é o sistema de controle de versão que nos ajuda e acompanhar as mudanças feitas no código, e também identificar quem efetuou mudanças nele. Além disso, através do Git podemos restaurar código removido ou modificado, e utiliza-lo em linha de comando (Git bash / CMD Git) para fazer tudo isso e mais, **criar repositórios**, fazer ***fork*** e **clonar** projetos, e enviar arquivos (**commits** e **pull-request**) para um repositório.

Digamos então que o **Git** é a parte operacional, responsável pelo famoso CRUD, que cria, altera, atualiza, remove e o **Hub** é a alma, a plataforma que possibilita essas ações.

Para utilizar o Git, é necessário baixa-lo (<https://git-scm.com/downloads>). Após a instalação, ele virá com o Git em CMD /Bash que é forma de entrar com os comandos em linha de comando (espécie de terminal). Ele também possui a opção Git GUI que é nada mais que uma interface que permite que você crie, clone ou abra repositórios,

Alguns comandos/dicas importantes:

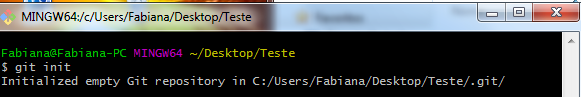
**Repositório** - ou repo, é o diretório onde serão **hospedadas** as versões de seus trabalhos, códigos e/ou projetos.

Para criar um repositório via linha de comando, você pode:

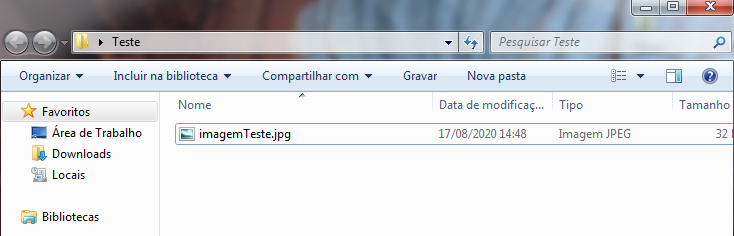
- Criar uma pasta (criei uma chamada ‘Teste’);

- Abrir o Git CMD/Bash dentro dela;

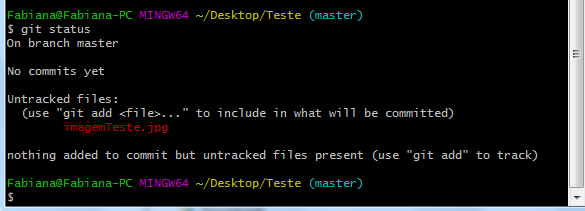
- Utilizar **git init** (a pasta será tratada como repositório);



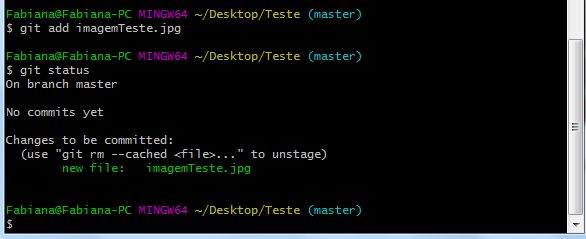
- Adicionar os arquivos que quer subir para o repositório (pode ser pastas, arquivos ou imagens);



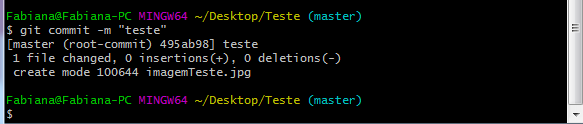
- Utilizar o **git status** para ver os arquivos modificados:



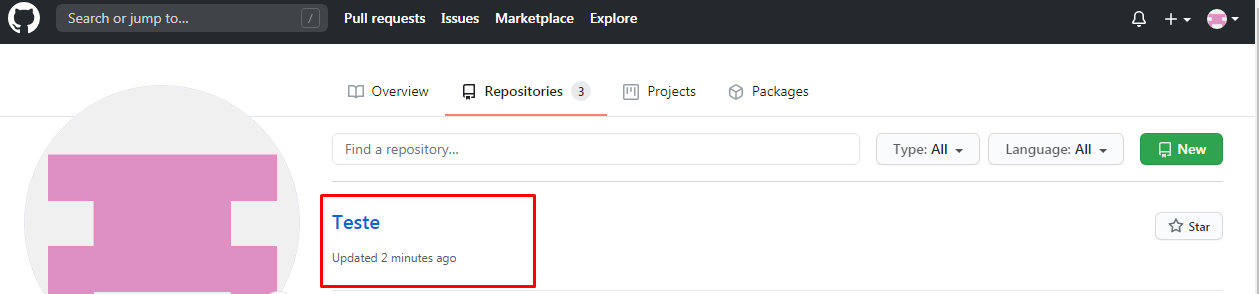
- Utilizar **o git add .** ou **git add nomeArquivo** para adicionar o (s) arquivo (s):



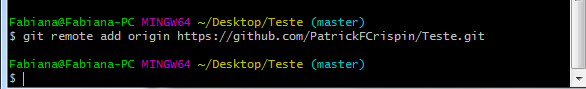
- Utilizar o **git commit -m “mensagem”** para persistir o arquivo no repositório:



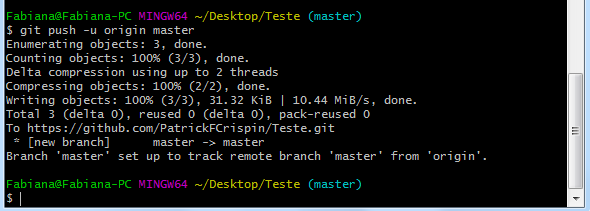
- No Github, criar um repositório com o mesmo nome da pasta criada no computador:



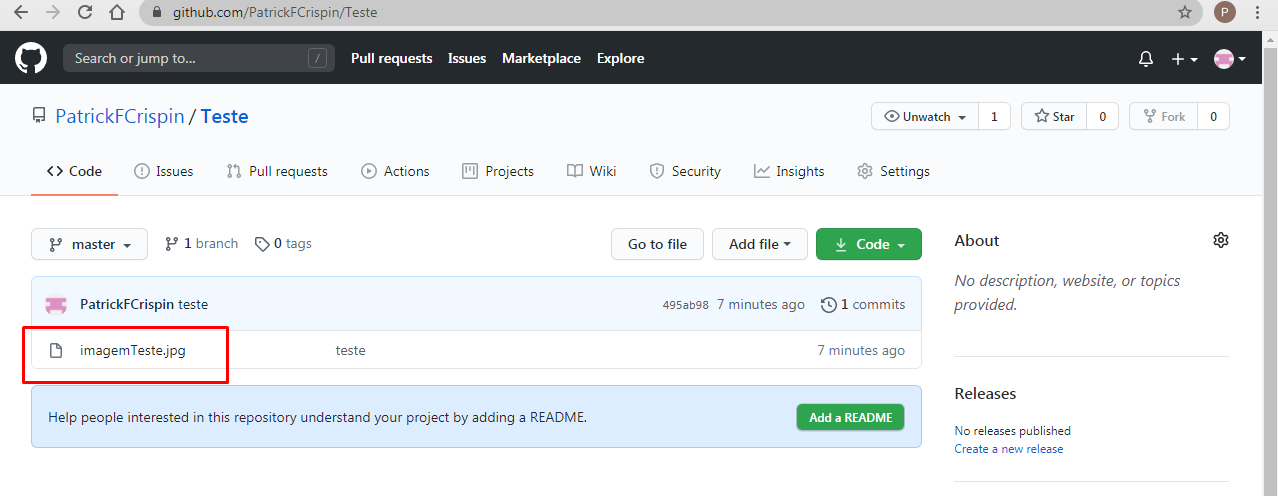
- Para finalizar, voltar ao Git e utilizar **git remote add origin ‘repositórioGit.git’**, onde o ‘repositórioGit’ é o link para o repositório criado no Github:



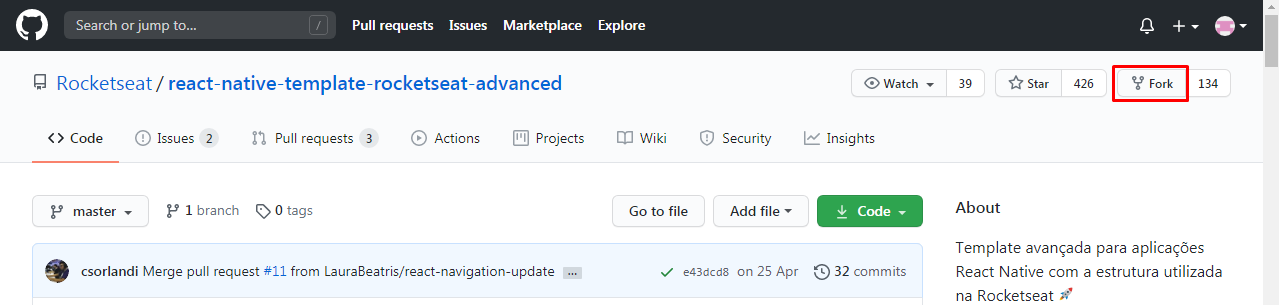
- Por fim, enviar o repositório, utilizar **git push -u origin master:**



No repositorio do Github, ao atualizar a página, temos o arquivo:



**Branch** - é uma cópia do diretório.

**Fork** - utilizado para criar uma cópia de um repositório para seu repositório. Isso é muito utilizado para utilizar pull-request. Basta clicar em ‘fork’ em algum repositório que você queira criar uma cópia. Exemplo: Rocketseat: 

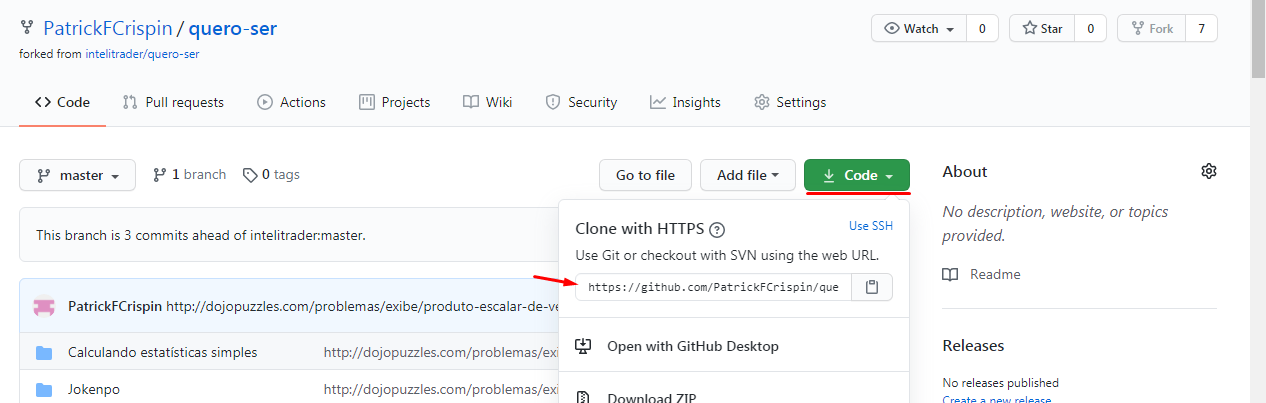
**Pull-request:**

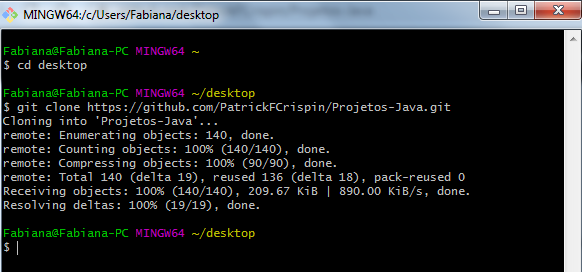
Utilizado basicamente para informar outros desenvolvedores / integrantes de um projeto que você irá implementar mudanças criadas no seu branch ao reposítório do projeto (geralmente master 🡪 master). Os colaboradores do repositório podem aceitar ou negar a Pull Request.

Através do Git, você pode:

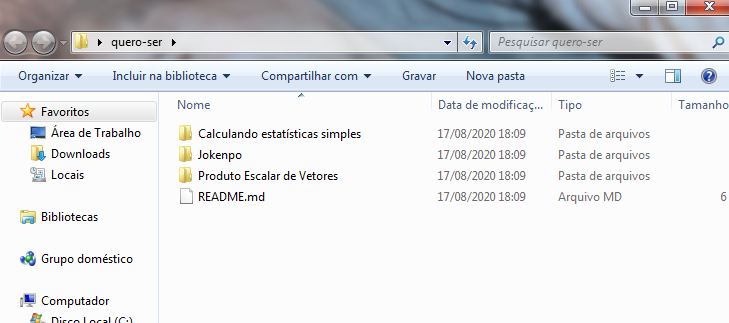
- Fazer um fork de um repositório;

- Abrir o Git CMD/Bash no seu computador (em qualquer lugar) e clone o repositório - utilize ‘**git clone repositório’** onde o ‘repositorio’ é o HTTPS fornecido la no GitHub no repositório que você fez o fork:

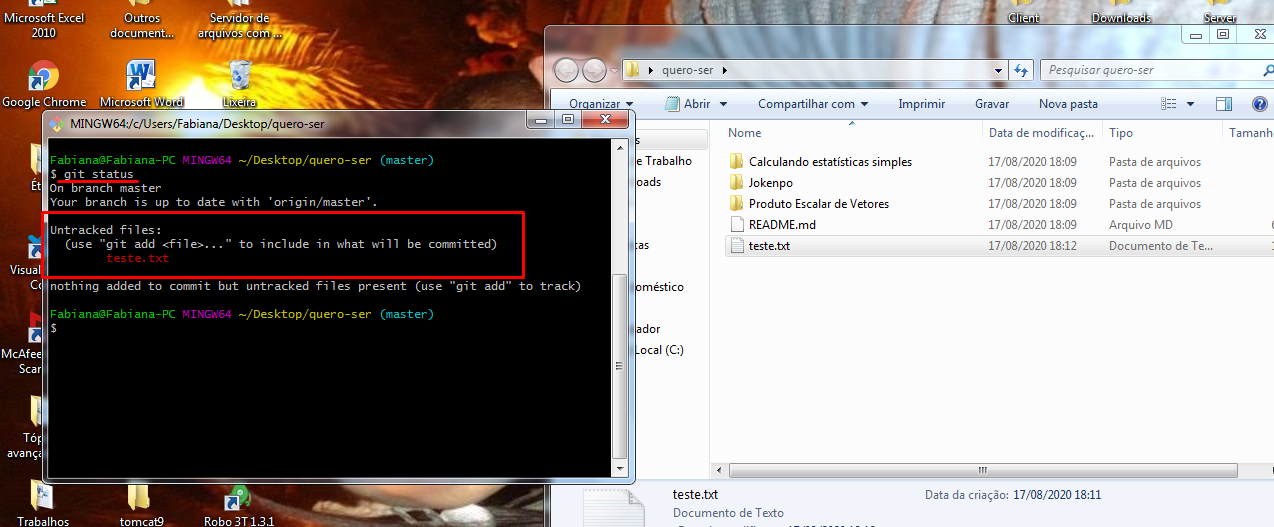




Neste momento foi criada uma pasta no seu computador, onde você abriu anteriormente o Git.



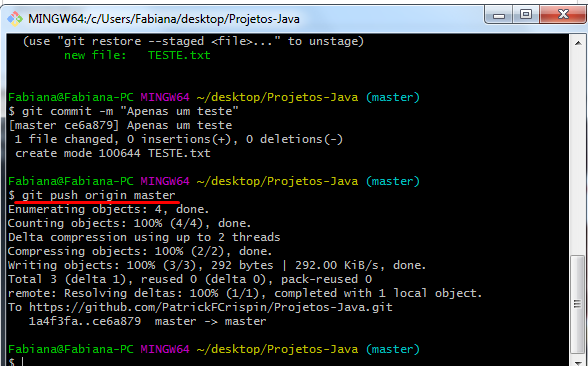
- Após, crie, adicione ou remova os arquivos da pasta. Criei um ‘teste.txt’ para exemplificar.

- Utilize o comando ‘**git status’** você pode verificar que em vermelhos são os itens modificados / alterados: 

- Utilize o comando **‘git add** .’ para adicionar todos os itens modificados. Se quiser adicionar apenas um determinado arquivo, utilize ‘**git add nomedoarquivo**’. Pronto, agora o Git adicionou o arquivo no repositório.

- Utilize o comando **‘git commit -m “mensagem que quiser”**’ para salvar a ação de adição anterior no repositório. Pronto, agora o arquivo foi persistido no repositório.

-Por fim, basta enviar as modificações ao repositório, e para isso digitamos o comando ‘**git push origin master’** ou somente **‘git push’**:



Obs.: se neste momento der algum erro, pode ser por causa da configuração de usuário. Então basta adicionar user.email e user.name de forma global e estes dados serão utilizados sempre:

Exemplo:

git config user.email "patrickfonseca208@gmail.com"

git config user.name "PatrickFCrispin" - adicionei conforme o nome do GitHub.

--

Para fazer o Pull-request basta clicar no botão ‘Pull-request’ > ‘New pull-request’ e então criar e enviar (send) o pull request.