

Git – Sistema de Versionamento de Códigos

Execução

Para verificar se o git está instalado, digite o comando a seguir:

```
git --version
```

```
C:\ProjetosJS\second-project-react-js>git --version
git version 2.34.1.windows.1
```

Deverá aparecer uma mensagem como acima.

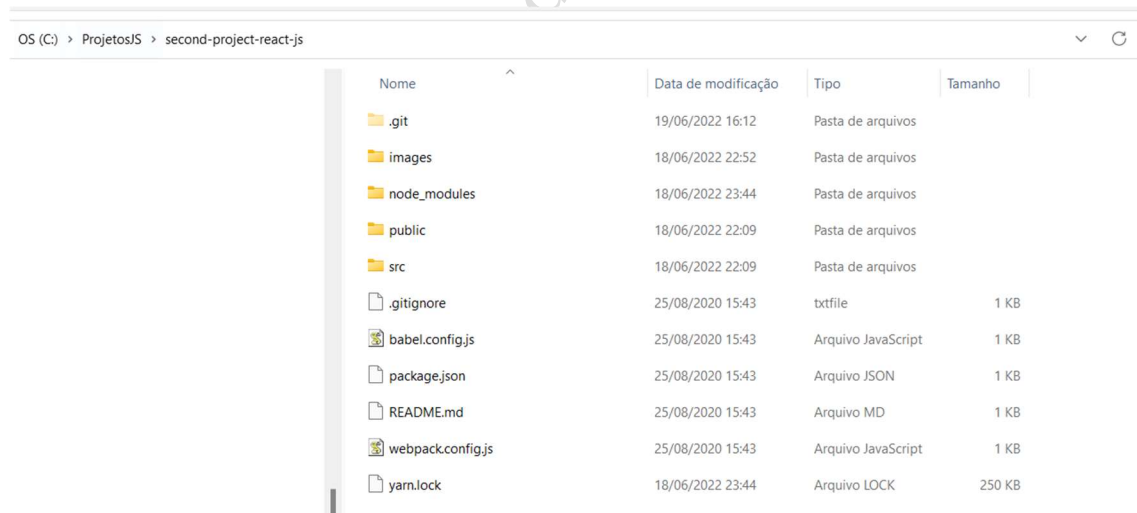
Entre no prompt de comando, na pasta do projeto, e digite o seguinte comando:

```
git init
```

```
C:\ProjetosJS\second-project-react-js>git init
Initialized empty Git repository in C:/ProjetosJS/second-project-react-js/.git/
```

Deverá aparecer uma mensagem como acima.

A pasta .git criada indica arquivos ocultos responsáveis pelo gerenciamento.



Nome	Data de modificação	Tipo	Tamanho
.git	19/06/2022 16:12	Pasta de arquivos	
images	18/06/2022 22:52	Pasta de arquivos	
node_modules	18/06/2022 23:44	Pasta de arquivos	
public	18/06/2022 22:09	Pasta de arquivos	
src	18/06/2022 22:09	Pasta de arquivos	
.gitignore	25/08/2020 15:43	txtfile	1 KB
babel.config.js	25/08/2020 15:43	Arquivo JavaScript	1 KB
package.json	25/08/2020 15:43	Arquivo JSON	1 KB
README.md	25/08/2020 15:43	Arquivo MD	1 KB
webpack.config.js	25/08/2020 15:43	Arquivo JavaScript	1 KB
yarn.lock	18/06/2022 23:44	Arquivo LOCK	250 KB

Na sequência, digite os seguintes comandos:

```
git add .gitignore
```

```
C:\ProjetosJS\second-project-react-js>git add .gitignore
warning: LF will be replaced by CRLF in .gitignore.
The file will have its original line endings in your working directory
```

Deverá aparecer uma mensagem como acima, com uma informação que pode ser ignorada.

```
git add .
```

```
C:\ProjetosJS\second-project-react-js>git add .
warning: LF will be replaced by CRLF in README.md.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in babel.config.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in package.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in public/bundle.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in public/index.html.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/App.css.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/App.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/components/Header.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/index.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in src/services/api.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in webpack.config.js.
The file will have its original line endings in your working directory
```

Deverá aparecer uma mensagem como acima, com uma informação que pode ser ignorada.

Comentários sobre os dois últimos comandos

O comando “git add .gitignore” confirma a seleção de arquivos desprezando os arquivos digitados no arquivo .gitignore, e o “git add .” adiciona no controle do git os arquivos alterados no projeto.

Veja a seguir que o comando `git status` mostra os 4 arquivos alterados e que ainda não foram commitados, digamos assim.

```
C:\ProjetosJS\second-project-react-js>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   .gitignore
    new file:   README.md
    new file:   babel.config.js
    new file:   images/browser_projeto01.gif
    new file:   package.json
    new file:   public/bundle.js
    new file:   public/index.html
    new file:   src/App.css
    new file:   src/App.js
    new file:   src/assets/background.png
    new file:   src/components/Header.js
    new file:   src/index.js
    new file:   src/services/api.js
    new file:   webpack.config.js
    new file:   yarn.lock
```

Neste momento iremos criar um ponto no histórico do projeto, para isto iremos digitar o comando “`git commit`”. No entanto, iremos indicar, por meio da opção `-m`, uma mensagem para caracterizar o commit, algumas opções de mensagem (`-m`), com alguns padrões de prefixos, são as seguintes:

`git commit -m "chore"` (commit inicial ou expansão de uma funcionalidade da aplicação)

`git commit -m "fix"` (indica a correção de um bug)

`git commit -m "feature"` (uma funcionalidade para um módulo que está sendo desenvolvido)

Então, para iniciar os commits faça o seguinte comando:

```
git commit -m "chore: Commit inicial"
```

Obs.: não copie e cole no prompt, digite o comando para não ser apresentado erro.

```
C:\ProjetosJS\second-project-react-js>git commit -m "chore: Commit inicial"
[master (root-commit) feb353c] chore: Commit inicial
15 files changed, 5631 insertions(+)
create mode 100644 .gitignore
create mode 100644 README.md
create mode 100644 babel.config.js
create mode 100644 images/browser_projeto01.gif
create mode 100644 package.json
create mode 100644 public/bundle.js
create mode 100644 public/index.html
create mode 100644 src/App.css
create mode 100644 src/App.js
create mode 100644 src/assets/background.png
create mode 100644 src/components/Header.js
create mode 100644 src/index.js
create mode 100644 src/services/api.js
create mode 100644 webpack.config.js
create mode 100644 yarn.lock
```

O resultado acima informa que os arquivos foram inseridos no histórico do git.

Faça o seguinte comando:

```
C:\ProjetosJS\second-project-react-js>git status
On branch master
nothing to commit, working tree clean
```

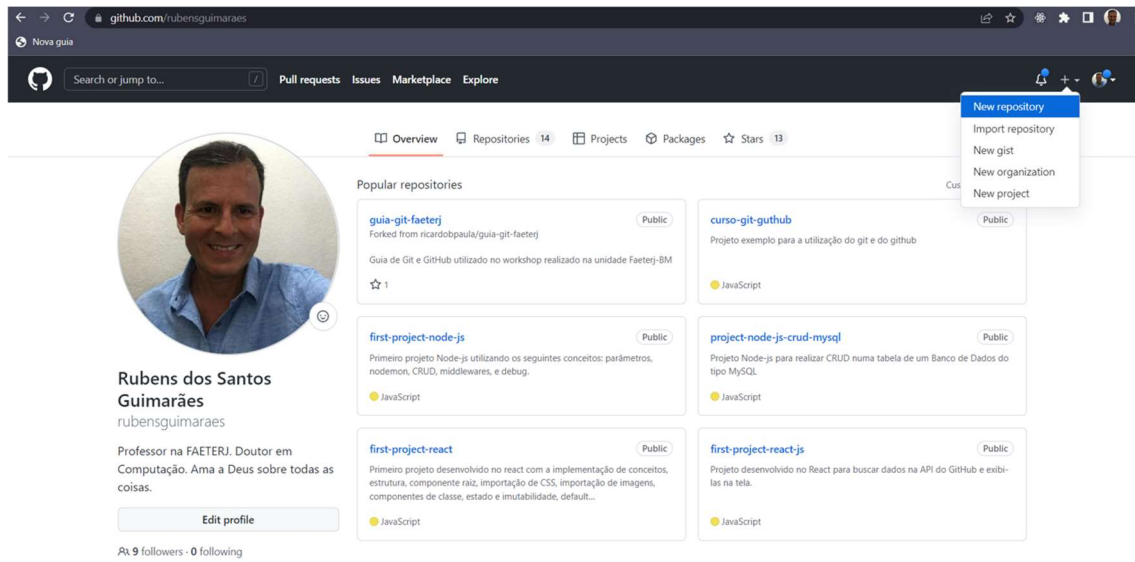
Veja que o git status nos indicou que está tudo OK, que nosso commit funcionou.

Github – Atualizando o Projeto no Github

Vamos agora atualizar o nosso projeto no Github.

Abra o github no seu navegador: github.com

Caso não possua, crie um perfil e em seguida faça o acesso para prosseguirmos. Após criar seu perfil vamos criar este novo repositório:




Abra a sua conta e clique em New repositior conforme indica a figura acima.

Informe o nome do repositório conforma mostrado a seguir e clique em Create repository.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *

 rubensguimaraes ▾

Repository name *

/ second-project-react-js ✓

Great repository names are short and memorable. Need inspiration? How about [expert-octo-meme?](#)

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▾

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

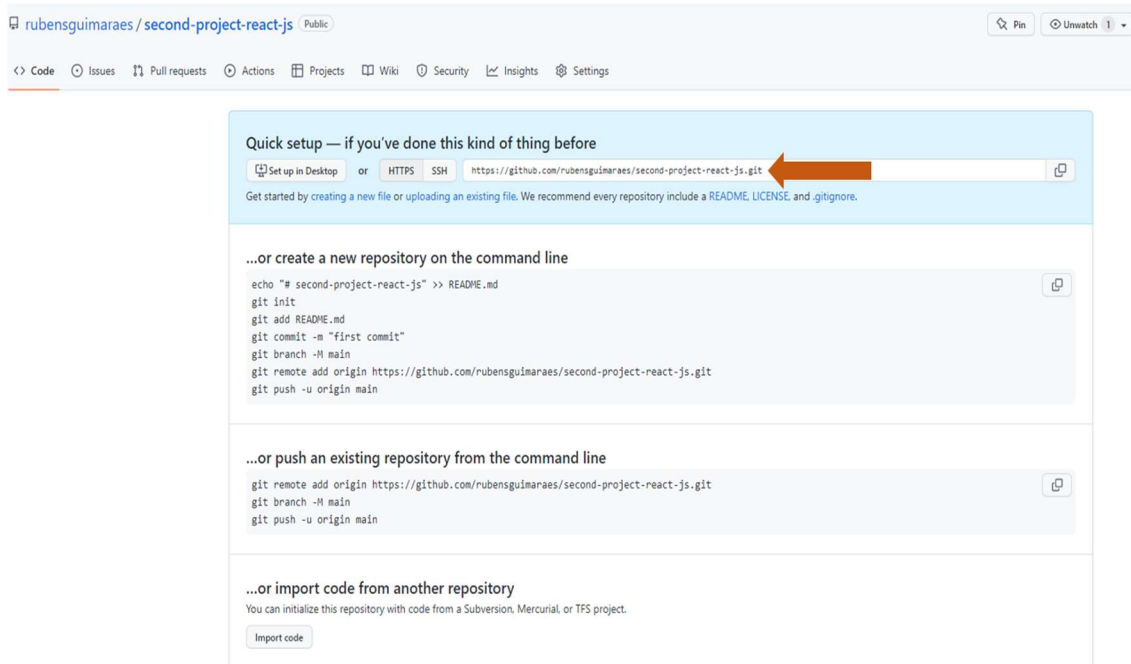
License: None ▾

 You are creating a public repository in your personal account.

Create repository



Prontinho! Repositório criado. Observe que ele nos fornece algumas instruções após a criação do repositório no github. Neste momento, obviamente, o github não possui informação sobre o nosso repositório local criado. Para isto, selecione a url informada por ele.



Prontinho! Repositório criado. Observe que ele nos fornece algumas instruções após a criação do repositório no github. Neste momento, obviamente, o github não possui informação sobre o nosso repositório local criado. Para isto, selecione a url informada por ele.

Vá no prompt do git e digite o seguinte:

```
git remote add origin https://github.com/rubensguimaraes/second-project-react-js.git
```

Após o comando acima ele irá processar internamente. Observe que a opção origin é um padrão utilizado, poderia ser qualquer coisa, mas iremos adotar o padrão seguindo as boas práticas.

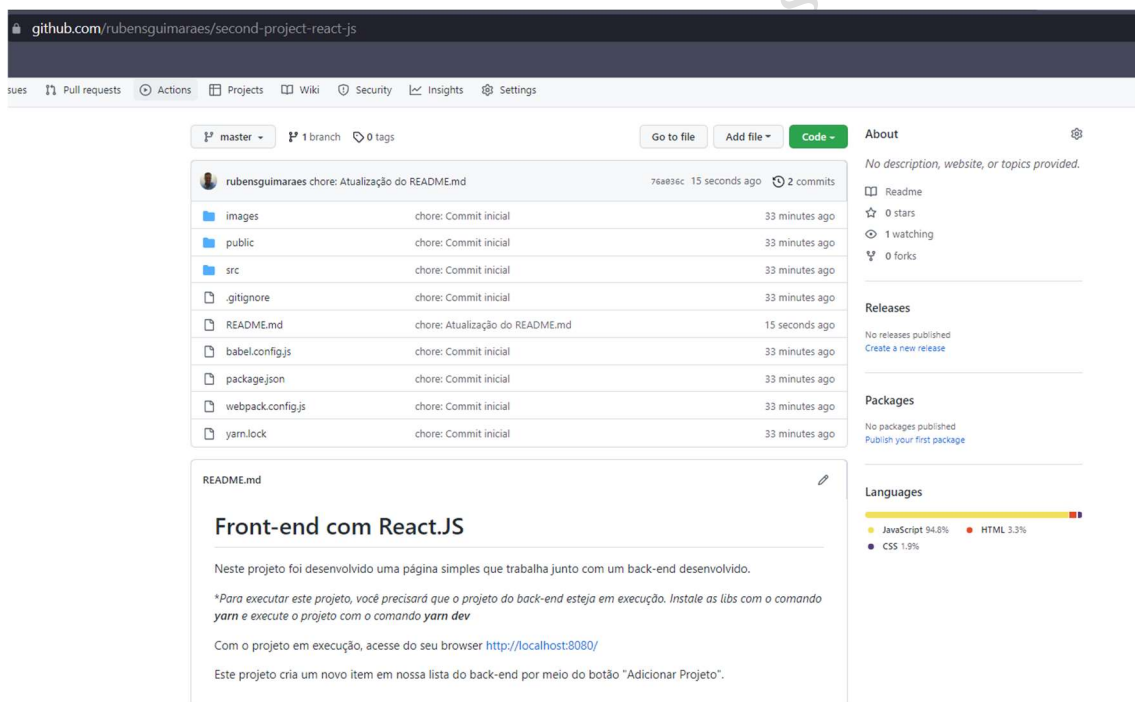
Em seguida, vamos executar o comando para enviar os nossos arquivos do projeto até o ponto do commit, veja o comando a seguir.

git push -u origin master

```
C:\ProjetosJS\second-project-react-js>git push -u origin master
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 8 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (23/23), 813.30 KiB | 19.84 MiB/s, done.
Total 23 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/rubensguimaraes/second-project-react-js.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

As mensagens acima indicam o sucesso do processamento, e que agora meu github reflete o meu git local.

Retorne ao navegador e atualize a página do github, veja que ele irá exibir os arquivos do projeto já no github.



Observe as mensagens “Commit inicial” e “Atualização do README.md” no histórico ao lado do nome dos arquivos, referentes a commits que executamos anteriormente.

Após novas atualizações do projeto, execute a seguinte sequência de comandos:

git add .

git commit -m "chore: Atualização do README.md"

git push -u origin master

Principais Comandos

git init // Inicia o projeto a partir da pasta atual (**Atenção:** caso o repositório não exista)

git status // Verifica o status dos arquivos no repositório

git add .gitignore // Define ao git para ignorar os arquivos da pasta . gitignore

git add . // Adiciona todos os arquivos ao repositório

git status // Verifica o status dos arquivos no repositório

git commit -m "Commit inicial" // Adiciona um ponto na história do repositório

git status // Verifica o status dos arquivos no repositório

git remote add origin <https://github.com/rubensguimaraes/first-project-react-js.git>

// Adicionar repositório remoto ao local

git push -u origin master // Criando master no repositório remoto (enviando os arquivos)

Para forçar um git push:

=====

git push -u origin master --force

Para remover um arquivo do repositório:

=====

git rm -r .vscode // Utilizei a opção -r por se tratar de uma pasta

após executar o comando acima fazer o seguinte:

git commit -m ".vscode removido"

e

git push -u origin master

=====

git pull --verbose // Traz do repositório remoto para o repositório local

OBS. IMPORTANTE referente ao git pull --verbose:

- Observa-se que em alguns casos ele não atualiza adequadamente, ou não atualiza, o repositório local. Nesta caso a alternativa é a seguinte:

git fetch --all

e em seguida:

git reset --hard origin/master

Explicação:

`git fetch` baixa a versão mais recente do controle remoto sem tentar mesclar ou refazer nada.

Em seguida, `git reset` redefine o ramo mestre para o que você acabou de buscar. A `--hard` opção altera todos os arquivos em sua árvore de trabalho para corresponder aos arquivos em `origin/master`

Professor Rubens dos Santos Guimarães