

Relatório Projeto Final DAS

António Pedro

ISTEC



Primeiramente comecei por criar o repositório com o nome do meu Projeto “ProjetoFinalDAS” e publiquei-o no Git.

1) No primeiro passo criei as branches necessárias e inicializei o repositório no Git Bash, criando também o clone do meu repositório no Git Desktop.

```

Pedro@DESKTOP-2K46RFU MINGW64 ~
$ cd ProjetoFinalDAS

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS
$ git flow init
Initialized empty Git repository in C:/Users/Pedro/ProjetoFinalDAS/.git/
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/Pedro/ProjetoFinalDAS/.git/hooks]

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (develop)
$

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (develop)
$ git branch master
fatal: a branch named 'master' already exists

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (develop)
$ git checkout master

```

2) No segundo tópico atribuí os privilégios aos membros na qual os developer conseguem fazer submissão de código, sem podendo fazer alterações de visibilidade.

Protect matching branches

☒ **Require a pull request before merging**
When enabled, all commits must be made to a non-protected branch and submitted via a pull request before they can be merged into a branch that matches this rule.

☒ **Require approvals**
When enabled, pull requests targeting a matching branch require a number of approvals and no changes requested before they can be merged.
Required number of approvals before merging: 1

☐ **Dismiss stale pull request approvals when new commits are pushed**
New reviewable commits pushed to a matching branch will dismiss pull request review approvals.

☒ **Require review from Code Owners**
Require an approved review in pull requests including files with a designated code owner.

☐ **Restrict who can dismiss pull request reviews**
Specify people, teams, or apps allowed to dismiss pull request reviews.

☐ **Allow specified actors to bypass required pull requests**
Specify people, teams, or apps who are allowed to bypass required pull requests

☐ **Require approval of the most recent reviewable push**
Whether the most recent reviewable push must be approved by someone other than the person who pushed it.

Member privileges

Base permissions

Base permissions to the organization's repositories apply to all members and excludes outside collaborators. Since organization members can have permissions from multiple sources, members and collaborators who have been granted a higher level of access than the base permissions will retain their higher permission privileges.

Write ▾

Repository creation

Members will be able to create only selected repository types. Outside collaborators can never create repositories.

☒ **Public**
Members will be able to create public repositories, visible to anyone.

☐ **Private**
Members will be able to create private repositories, visible to organization members with permission.

Save

Repository forking

☐ **Allow forking of private repositories**
If enabled, forking is allowed on private and public repositories. If disabled, forking is only allowed on public repositories. This setting is also configurable per-repository.

Save

Pages creation

Members will be able to publish sites with only the selected access controls.

☒ **Public**
Members will be able to create public sites, visible to anyone.

☐ **Private**
Members will be able to create private sites, visible to anyone with permission. [Why is this option disabled?](#)

Save

3) No terceiro passo, tornei obrigatória a revisão de código antes dos pull requests serem aprovados.

Branch protection rule

Branch name pattern *

master

Applies to 1 branch

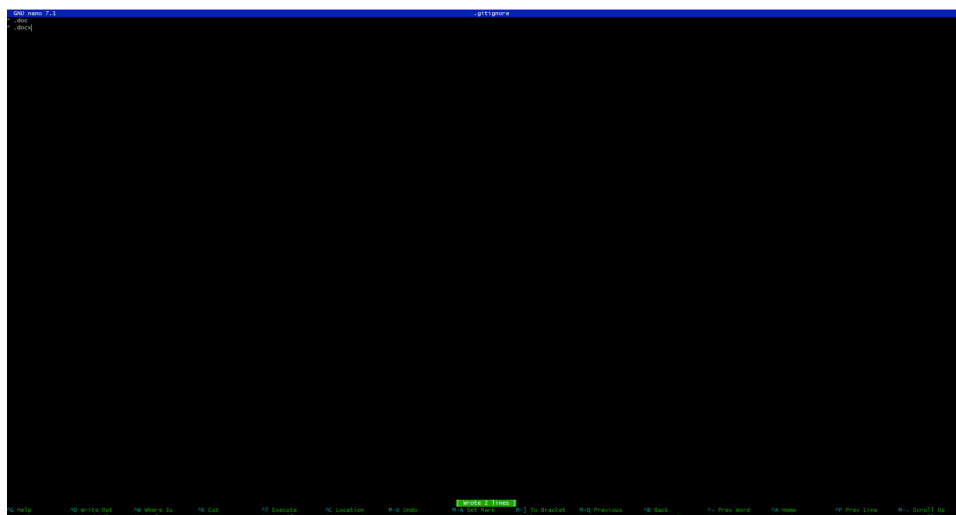
master

Protect matching branches

- ☒ **Require a pull request before merging**
When enabled, all commits must be made to a non-protected branch and submitted via a pull request before they can be merged into a branch that matches this rule.
- ☒ **Require approvals**
When enabled, pull requests targeting a matching branch require a number of approvals and no changes requested before they can be merged.
Required number of approvals before merging: 1 ▼
- ☐ **Dismiss stale pull request approvals when new commits are pushed**
New reviewable commits pushed to a matching branch will dismiss pull request review approvals.
- ☒ **Require review from Code Owners**
Require an approved review in pull requests including files with a designated code owner.
- ☐ **Restrict who can dismiss pull request reviews**
Specify people, teams, or apps allowed to dismiss pull request reviews.
- ☐ **Allow specified actors to bypass required pull requests**
Specify people, teams, or apps who are allowed to bypass required pull requests.
- ☐ **Require approval of the most recent reviewable push**
Whether the most recent reviewable push must be approved by someone other than the person who pushed it.

4) Neste passo, adicionei o ficheiro .gitignore.

```
Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (master)
$ nano .gitignore
```



5)

carregamento inicial do ficheiro no branch develop

```
Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (develop)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (master)
$ git commit -m "gitignore"
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  .gitignore

nothing added to commit but untracked files present (use "git add" to track)

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (master)
$ git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (master)
$ git commit -m "gitignore"
[master 8dabc5] gitignore
1 file changed, 2 insertions(+)
create mode 100644 .gitignore
```

```
Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (master)
$ git add .

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (master)
$ git commit -m "Carregamento Inicial"
On branch master
Your branch is up to date with 'origin/master'.

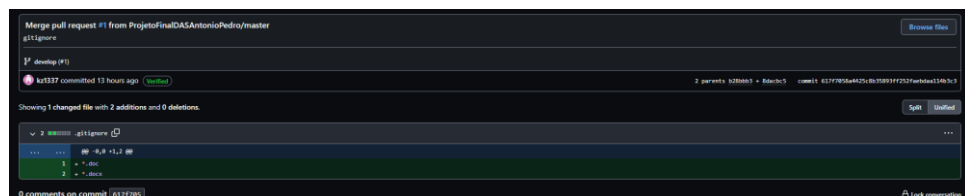
nothing to commit, working tree clean

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (master)
$ git checkout develop
Switched to branch 'develop'
Your branch is up to date with 'origin/develop'.

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (develop)
$ git commit -m "Carregamento Inicial"
On branch develop
Your branch is behind 'origin/develop' by 2 commits, and can be fast-forwarded.
  (use "git pull" to update your local branch)

nothing to commit, working tree clean

Pedro@DESKTOP-2K46RFU MINGW64 ~/ProjetoFinalDAS (develop)
$ git push origin develop
remote: Permission to ProjetoFinalDASAntonioPedro/ProjetoFinalDAS.git denied to kanza1337.
fatal: unable to access 'https://github.com/ProjetoFinalDASAntonioPedro/ProjetoFinalDAS.git/': The requested URL returned error: 403
```



alterações no próprio branch develop com os devidos comentários