

Trabalho Prático Individual Final

Git



Aluno: Afonso Alexandre Neves Almeida 2022006

Disciplina: Desenvolvimento Ágil de Software

Professor: André Carvalhas

O presente trabalho tem como objetivo criar um repositório no Github com as devidas restrições:

1. Deve ter criados todos os branches necessários para a utilização do modelo GitFlow no que toca à gestão de branches;
2. Ter pelo menos dois níveis de acesso ao GitHub, permitindo a developers a submissão de código, mas nunca a alteração de visibilidade do repositório (esta permissão deve ser dada apenas a proprietários do repositório);
3. Tornar obrigatória a revisão de código antes de um pull request ser aprovado;
4. Adicionar ficheiro .gitignore de forma a ignorar ficheiros, .docx e .doc;
5. Controlar as versões do relatório a entregar usando o repositório, sendo que é necessário que existam pelo menos:
 - a. 1 carregamento inicial do ficheiro no branch develop;
 - b. 5 alterações no próprio branch develop com os devidos comentários;
 - c. 1 release que contribua para o branch master;
 - d. 1 hotfix efetuado utilizando a gestão de branches GitFlow;
6. A versão do relatório final deve estar presente no repositório, no branch master.

Criação do repositório

Criação do repositório

Primeiramente devemos criar um repositório no github e configurar o nome e descrição.

The screenshot shows the 'Create a new repository' page on GitHub. A red arrow points to the 'Repository name' field, which contains 'Final DAS' and has a green checkmark. Another red arrow points to the 'Description' field, which contains 'Este repositório é o projeto final da disciplina de DAS'. A third red arrow points to the 'Create repository' button at the bottom. The form includes sections for 'Owner' (ATEK0), 'Repository name', 'Description (optional)', 'Public/Private' options (Public is selected), 'Initialize this repository with' (Add a README file is unchecked), 'Add .gitignore' (template: None), and 'Choose a license' (License: None). A note at the bottom states: 'You are creating a public repository in your personal account.'

Ao criarmos o repositório devemos criar uma pasta no nosso computador que vamos conectar ao repositório e depois trabalhar dentro dela.

Depois de criar a pasta devemos abrir o CMD/prompt e navegar para dentro da pasta.

Para configurar o repositório localmente:

```
echo # Final-DAS >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/ATEK0/Final-DAS.git
git push -u origin main
```

Agora já temos criado o nosso repositório. Temos o nosso primeiro commit, este envolve o envio do ficheiro README para o repositório.

O que é um ficheiro README.md

O ficheiro README é um ficheiro de marcação Markdown utilizado para dar uma descrição visível na primeira página e com formatação extra ao repositório.

1 - Criação do ambiente GitFlow

O que é GitFlow

GitFlow é um modelo de branching model para Git criado por Vincent Driessen.

É um conjunto de extensões Git que fornecem operações no repositório de alto nível. Este modelo é escalável e adequado para modelos colaborativos.

Como implementar o modelo GitFlow

Implementar o modelo GitFlow é extremamente simples, pois ao instalar o “Git for Windows” a extensão que permite implementar o modelo vem junto.

```
git flow init
```

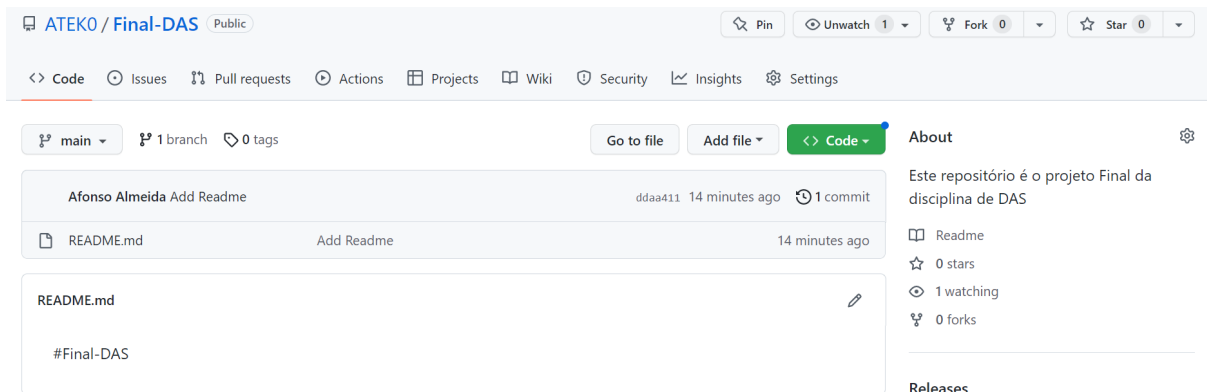
```
C:\istec\ DAS \Trabalho Final>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
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:/istec/DAS/Trabalho Final/.git/hooks]
```

recomendado usar os valores default
ENTER em todos os steps

Depois de executarmos todos os comandos acima enumerados o nosso repositório tem agora um ficheiro 'README.md' no branch *main*,



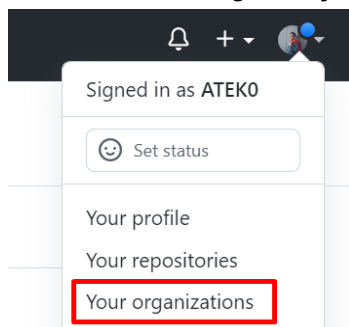
2 - Criar níveis de acesso ao GitHub

Devemos ter pelo menos dois níveis de acesso ao GitHub, permitindo a *developers* a submissão de código, mas nunca a alteração de visibilidade do repositório (esta permissão deve ser dada apenas ao/aos proprietário/os do repositório).

Para criar níveis de acesso ao repositório do github precisamos de criar uma organização e de transferir o repositório para a organização.

Criar uma organização

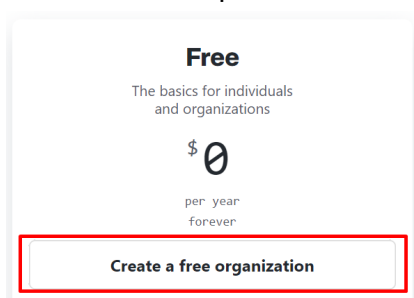
Para criar uma organização temos que clicar no nosso icon e ir para 'Your organizations'.



Agora temos que criar uma organização ao clicar no botão 'New Organization'.



Selecionamos o plano 'Free'.



Tell us about your organization

Set up your organization

Organization account name *

Projeto Final DAS ✓

This will be the name of your account on GitHub.
Your URL will be: <https://github.com/Projeto-Final-DAS>, which has been adjusted to comply with our naming rules.

Contact email *


afonso.almeida.2022006@my.istec.pt ✓

This organization belongs to: *

☒ My personal account
I.e., ATEKO (Afonso Almeida)

☐ A business or institution
For example: GitHub, Inc., Example Institute, American Red Cross

Verify your account



☒ I hereby accept the [Terms of Service](#). For more information about GitHub's privacy practices, see the [GitHub Privacy Statement](#).

Next

Precisamos de configurar as informações da nossa organização.

Precisamos de definir um nome, um email de contacto, a quem pertence a organização e precisamos de passar por uma espécie de *captcha* e aceitar os termos e condições.

Ao clicar no botão 'Next' podemos escolher se queremos adicionar membros à organização, caso não queiramos podemos passar este passo a frente.

O último passo é preencher um questionário para o github perceber para que fins vai ser utilizada a organização. Após preencher o questionário podemos clicar no botão 'Submit' no fim da página.

Transferir o repositório para a organização


Para transferir o repositório precisamos de aceder às definições do repositório. No final da página podemos ver a '**Danger Zone**' onde vamos clicar no botão 'Transfer'.

Danger Zone

| | |
|--|---|
| Change repository visibility This repository is currently public. | Change visibility |
| Transfer ownership Transfer this repository to another user or to an organization where you have the ability to create repositories. | Transfer |
| Archive this repository Mark this repository as archived and read-only. | Archive this repository |
| Delete this repository Once you delete a repository, there is no going back. Please be certain. | Delete this repository |

Finalmente, devemos seleccionar que queremos que o novo dono do repositório seja uma das minhas organizações, escrever a mensagem de confirmação na caixa de texto e clicar no botão para confirmar a transferência do repositório.


Transfer repository: ATEK0/Final-DAS

 To understand admin access, teams, issue assignments, and redirects after a repository is transferred, see [Transferring a repository](#) in GitHub Help.

Transferring may be delayed until the new owner approves the transfer.

New owner *


☒ Select one of my organizations

 Projeto-Final-DAS ▾

☐ Specify an organization or username

Repository name *

Final-DAS

 Final-DAS is available.

Type ATEK0/Final-DAS to confirm.

ATEK0/Final-DAS

[I understand, transfer this repository.](#)

Configurar os níveis de acesso ao repositório

Para configurar os níveis de acesso temos que aceder às definições da organização e na aba 'Member privileges' devemos mudar a 'Base permissions' para 'write' e desativar a *checkbox* da 'Repository visibility change'.

Base permissions

Base permissions to the org permissions from multiple s retain their higher permissic

Write ▾

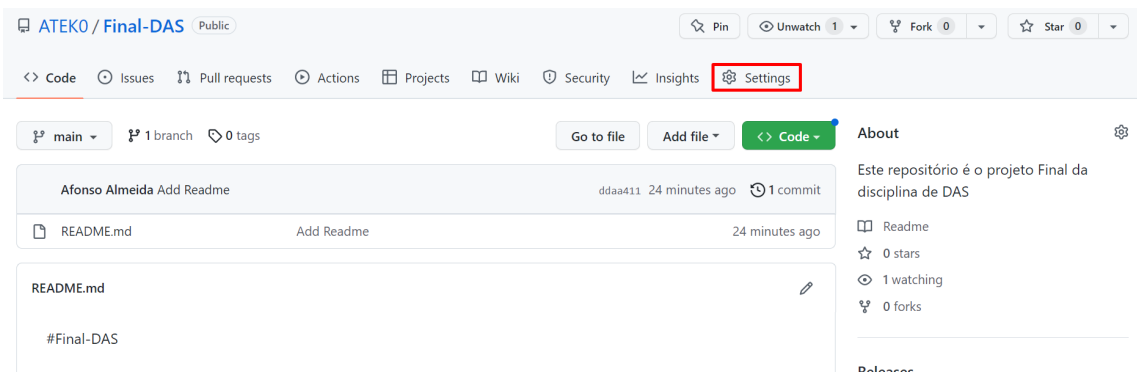
Repository visibility change

☐ **Allow members to change repository visib**
If enabled, members with admin permissions for th

Save

3 - Tornar obrigatória a revisão do código antes de um pull request ser aprovado

Para definir este tipo de permissões precisamos de aceder às definições do repositório. Para isso abrimos a página do repositório e clicamos na aba 'Settings'.



ATEK0 / Final-DAS Public

Pin Unwatch 1 Fork 0 Star 0

<> Code Issues Pull requests Actions Projects Wiki Security Insights **Settings**

main ▾ 1 branch 0 tags

Go to file Add file ▾ <> Code ▾ About

Afonso Almeida Add Readme ddaa411 24 minutes ago 1 commit

README.md Add Readme 24 minutes ago

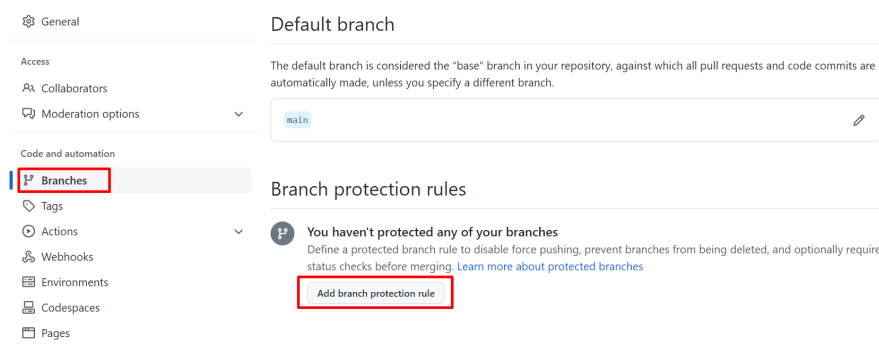
README.md #Final-DAS

Este repositório é o projeto Final da disciplina de DAS

Readme 0 stars 1 watching 0 forks

Releases No releases published [Create a new release](#)

Nas definições devemos ir para as definições de ‘branches’ e criar uma regra nova para a *branch* ‘main’



e marcar as *checkboxes* ‘Require a pull requests before merging’ e ‘Require approvals’. Na *selection* da segunda *checkbox* podemos escolher quantos utilizadores precisam de aprovar o código de um *pull request*. Escolhi apenas uma aprovação. Também devemos definir para que *branch* queremos criar a definição, criei no ‘main’ (o nome deve ser escrito na caixa de texto).

No final da página encontramos um botão verde para criar a regra. Ao clicar nele a regra fica salva e está ativa.



Agora, sempre que alguém fizer *commit* é preciso criar um *pull request*. Por sua vez, o *pull request* precisa de ser aprovado por uma pessoa.

4 - Adicionar .gitignore para ignorar ficheiros .docx e .doc

O que é um ficheiro .gitignore

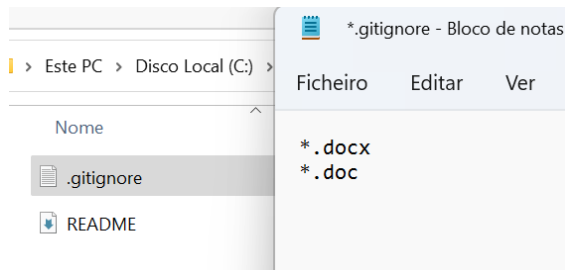
O ficheiro .gitignore serve para especificar ficheiros, tipos de ficheiros, padrões, etc. Os ficheiros aqui especificados ficaram *untracked* dentro do repositório. Estes ficheiros vão ser ignorados ao dar stage (por exemplo).

Como funciona o ficheiro?

O ficheiro .gitignore é simplesmente um ficheiro de texto. Dentro dele colocamos o nome do ficheiro, ou alguma condição de ficheiro que queremos ignorar.

Criar o ficheiro .gitignore

Queremos ignorar todos os ficheiros .docx e .doc, então definimos o * para identificar que pode ter qualquer coisa para o lado esquerdo do nome do ficheiro que queremos ignorar, desde que terminem com .docx ou .doc



Para adicionarmos o ficheiro abrimos a pasta do repositório e criamos um ficheiro de texto com o nome “.gitignore” e dentro dele escrevemos as condições desejadas. No caso em questão são as condições mostradas na imagem acima.

É ainda preciso enviar o ficheiro para o *branch* ‘main’. Para isso usamos os comandos:

```
git checkout main
git add .
git commit -m "Adding .gitignore file"
git push origin main
```

5 - Controlar as versões do relatório a entregar usando o repositório, sendo que é necessário que existam pelo menos:

1- Carregamento inicial do ficheiro no branch develop

Para carregarmos o ficheiro inicial do relatório vamos precisar de colocar o relatório na pasta do nosso repositório local.

Após colocar o relatório dentro da pasta, vamos abrir o CMD e navegar para a pasta do repositório local. Como definimos o ficheiro *ignore* para excluir ficheiros .doc e .docx temos de colocar o relatório em formato .pdf.

Quando abrirmos o CMD devemos verificar em que branch estamos a trabalhar utilizando o comando:

```
git branch
```

```
C:\istec\DAS\Trabalho Final>git branch
* develop
main
```

A *branch* que aparece a verde é a *branch* em que estamos a trabalhar.

Agora precisamos de dar *stash* ao ficheiro do relatório:


```
git add .
```


Agora vamos fazer *commit* do ficheiro para a seguir o podemos transferir para o repositório online.


```
git commit -m "Adicionar primeira vers do relatório"
```

Para enviar o relatório para a *branch* ‘develop utilizamos:

```
git push origin develop
```


 develop ▾

 2 branches


 0 tags

Go to file

Add file ▾


<> Code ▾

This branch is 1 commit ahead of main.

 #1


Afonso Almeida Adicionar primeira vers do relatorio

5ec6bb5 4 minutes ago 3 commits

 .gitignore.txt


Adding .gitignore file

22 minutes ago

 README.md

Add Readme

1 hour ago

 Trabalho Prático Individual Final - Git...

Adicionar primeira vers do relatorio

4 minutes ago

2 - 5 alterações no branch develop com os devidos comentários

1 alteração - Acrescento do ponto 5.1

```
git flow feature start alteracao1
#depois de fazer as alterações
git add .
git commit -m "Acrescento do ponto 5.1"
git flow feature finish alteracao1
git push origin develop
```

2 alteração - Relatório apenas com o ponto 1(criar repo)

```
git flow feature start alteracao2
#depois de fazer as alterações
git add .
git commit -m "Relatório apenas com o ponto 1(criar repo)"
git flow feature finish alteracao2
git push origin develop
```

3 alteração - Relatório criar níveis de acesso (organização)

```
git flow feature start alteracao3
#depois de fazer as alterações
git add .
git commit -m "Relatório criar níveis de acesso (organização)"
git flow feature finish alteracao3
git push origin develop
```

4 alteração - Relatório criar níveis de acesso (transferir)

```
git flow feature start alteracao4
#depois de fazer as alterações
git add .
git commit -m "Relatório criar níveis de acesso (transferir)"
git flow feature finish alteracao4
git push origin develop
```

5 alteração - Relatório até 5c

```
git flow feature start alteracao5
#depois de fazer as alterações
git add .
```

```
git commit -m "Relatório até 5c"  
git flow feature finish alteracao5  
git push origin develop
```