

Instituto Superior de Tecnologias Avançadas do Porto
CTESP DS - Curso Técnico Superior Profissional de
Desenvolvimento de Software

Ano letivo 2022/2023
DAS - Desenvolvimento Ágil de Software

Trabalho Prático Individual

Final

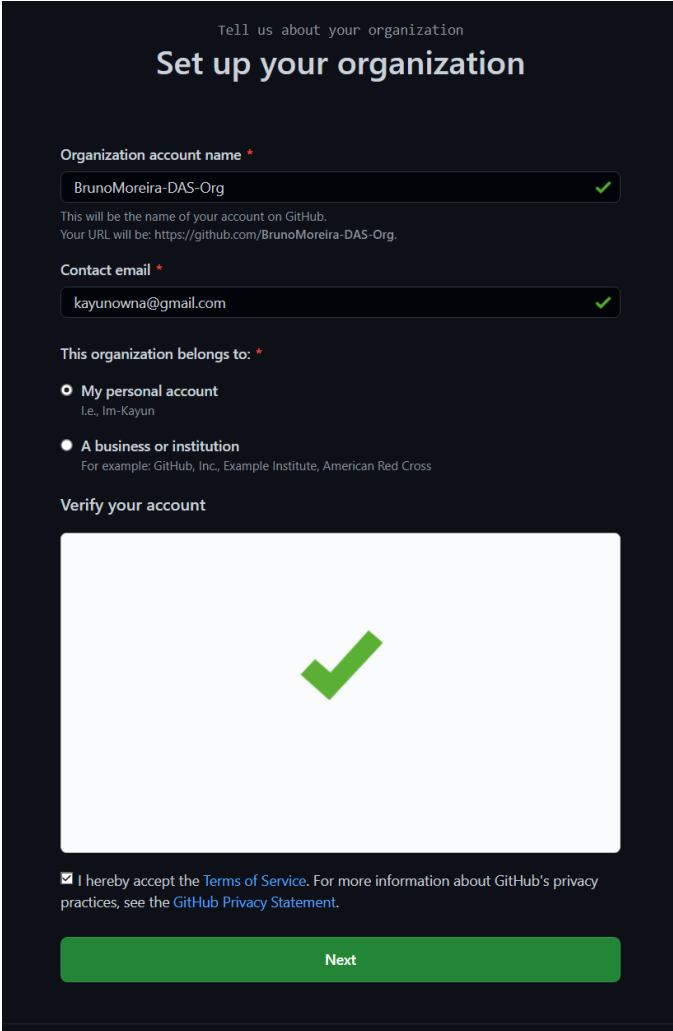


<https://github.com/BrunoMoreira-DAS-Org/ProjetoFinal>

Bruno Moreira

Criação de uma organização

Para criar uma organização, basta ir ao menu lateral e selecionar a opção “Your organizations”. De seguida clica-se no botão “New Organization”, onde irá aparecer um menu para criar uma nova organização.



Tell us about your organization

Set up your organization

Organization account name *

BrunoMoreira-DAS-Org ✓

This will be the name of your account on GitHub.
Your URL will be: <https://github.com/BrunoMoreira-DAS-Org>.


Contact email *

kayunowna@gmail.com ✓

This organization belongs to: *

- ☐ My personal account
I.e., I'm Kayun
- ☐ A business or institution
For example: GitHub, Inc., Example Institute, American Red Cross

Verify your account

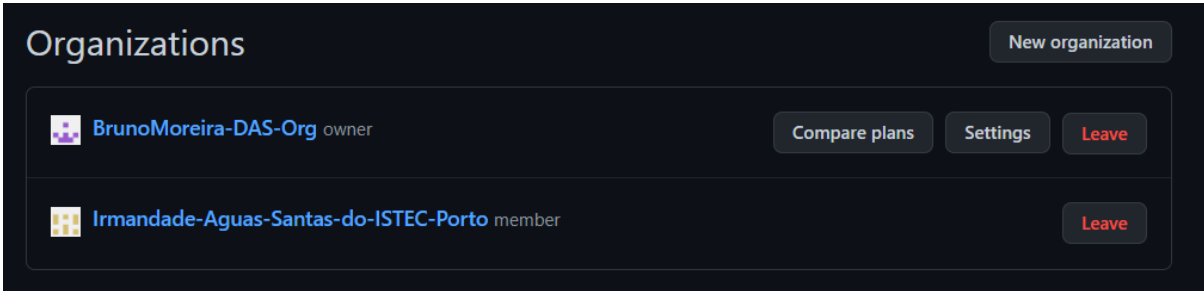


☒ I hereby accept the [Terms of Service](#). For more information about GitHub's privacy practices, see the [GitHub Privacy Statement](#).

Next



Após preencher todos os campos, basta clicar no botão “Next” e concluir a criação da organização.

Para aceder às configurações da organização basta clicar no botão “Settings” que se encontra dentro do menu “Your Organizations”.



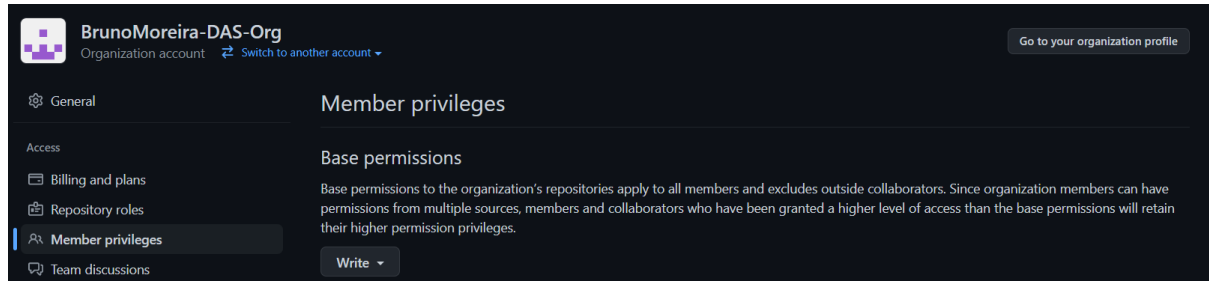
Organizations

New organization

 BrunoMoreira-DAS-Org owner	Compare plans	Settings	Leave
 Irmandade-Aguas-Santas-do-ISTEC-Porto member			Leave

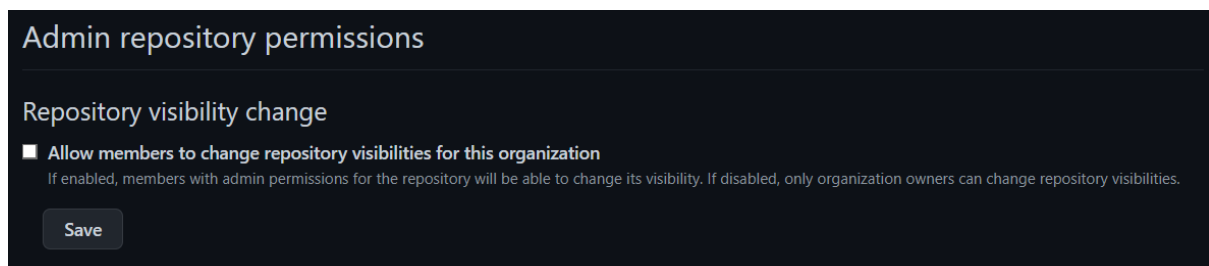
As organizações devem ter níveis de acesso para os desenvolvedores, com isto é possível limitar as permissões dentro da organização.

Para alterar os níveis de acesso, basta clicar no botão “Member Privileges” dentro da categoria “Access”.



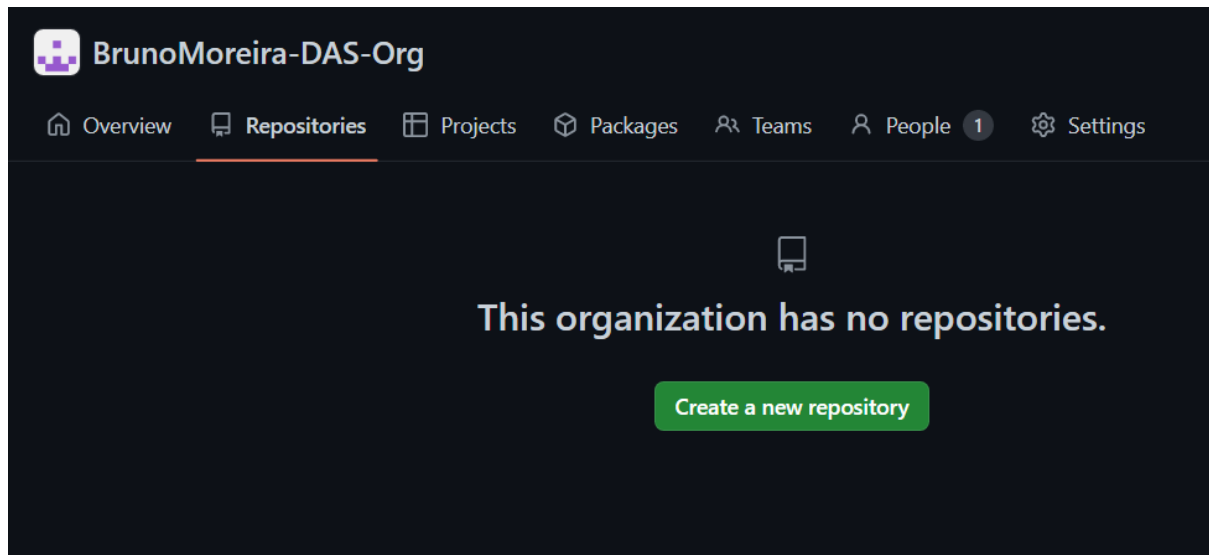
Seleciona-se a opção “Write” para que os desenvolvedores possam usar os comandos “clone”, “push” e “pull” para trabalhar nos repositórios pertencentes à organização.

Para não permitir que os desenvolvedores/membros da organização alterem a visibilidade, basta ir até ao fundo da página e na zona de “Admin repository permissions”, desativar a opção “Repository visibility change”.

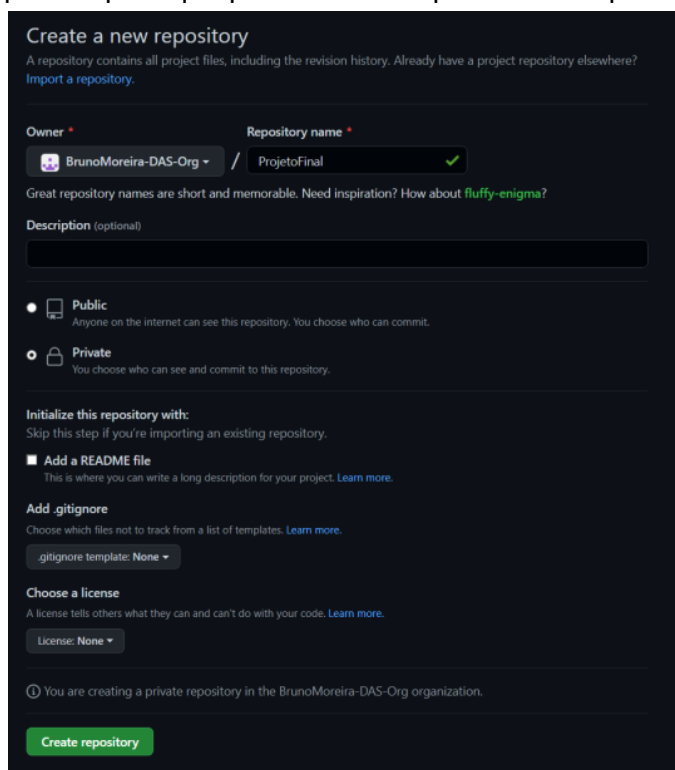


Criação de um repositório

Para criar um repositório dentro da organização, basta ir à página principal da organização e clicar no botão “Create a new repository” na opção “Repositories”.



Ao clicar no botão irá aparecer um menu para criar o novo repositório. O relatório deve ser público para que posteriormente possam ser aplicadas regras às branches.



Para criar o repositório, basta clicar no botão “Create repository”.

Após criar o repositório, é necessário criar uma pasta localmente no computador que irá estar conectada ao repositório remoto no *github*, isto para que seja possível trabalhar e enviar alterações para o repositório no *github*.

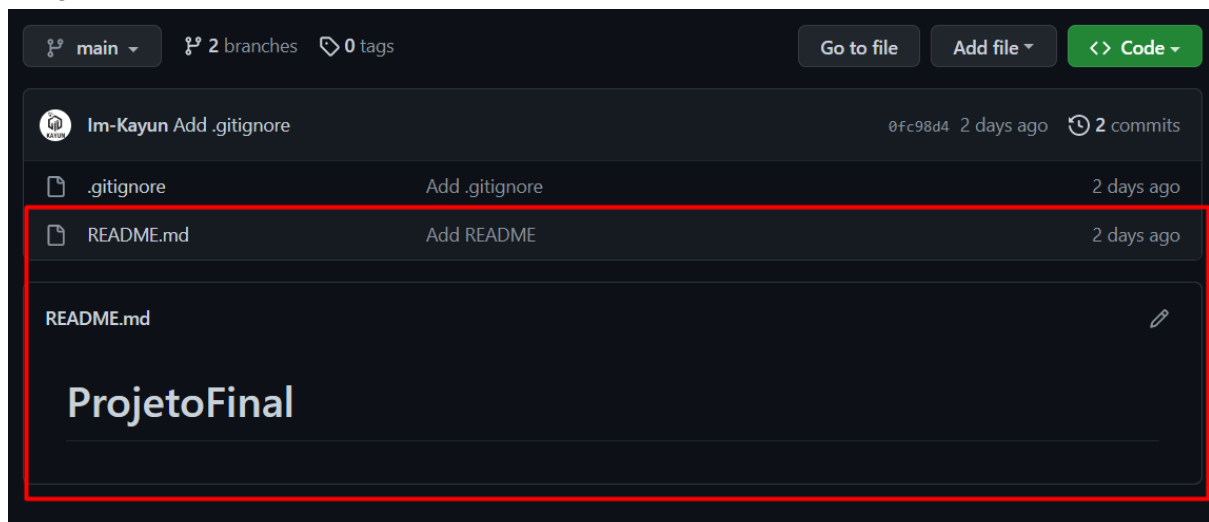
Neste caso, foi criada uma pasta chamada ProjetoDAS no Desktop. Para navegarmos até à pasta, é necessário abrir o git Bash e usar os seguintes comandos:

```
cd ./Desktop  
cd ProjetoDAS
```

Para configurar o repositório localmente deve-se usar os seguintes comandos:

```
echo "# ProjetoFinal" >> README.md  
git init  
git add README.md  
git commit -m "Add README"  
git branch -M main  
git remote add origin  
https://github.com/BrunoMoreira-DAS-Org/ProjetoFinal.git  
git push -u origin main
```

Após executar todos os comandos, o ficheiro README.md aparecerá no repositório remoto (no github) na branch main.

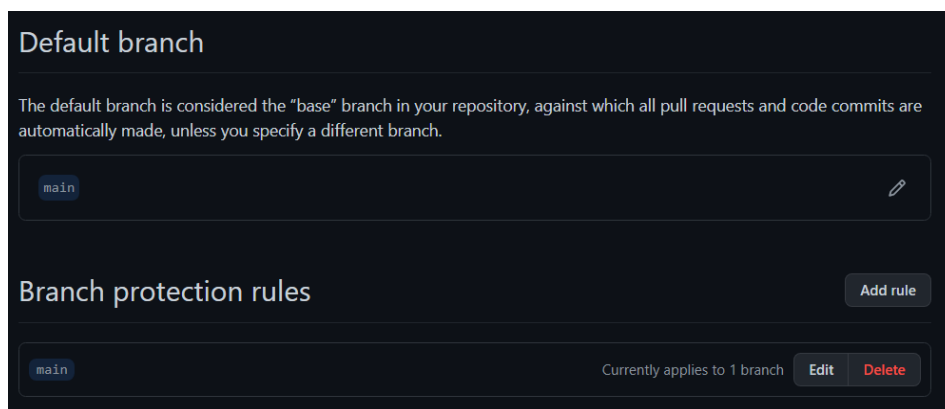
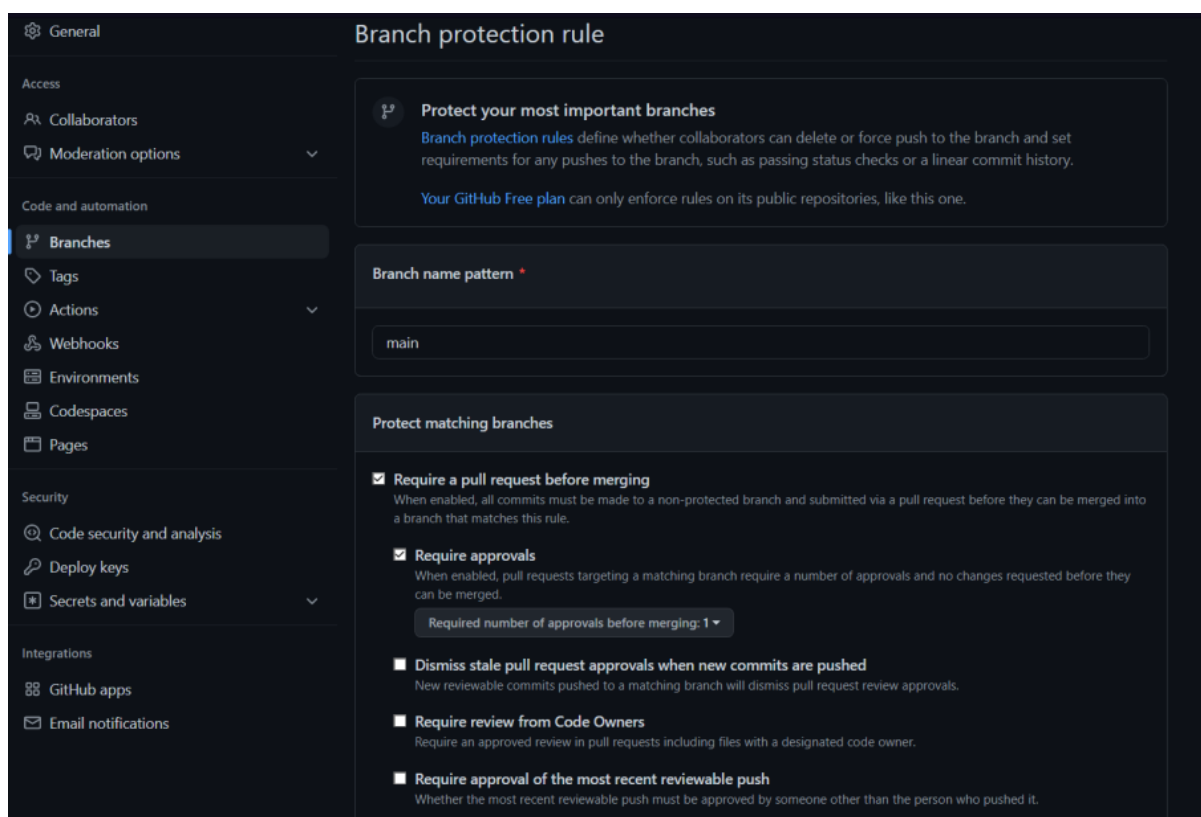


Regras de Proteção nas Branches

Definir regras de proteção nas branches é essencial, pois protege muitas vezes de código indesejado. Para adicionar regras de proteção é necessário ir à aba Settings do repositório. Quando se clica no botão “New Rule”, irá aparecer um menu com várias opções, sobre as regras que podem ser aplicadas. Neste caso será aplicada uma regra que torna obrigatória a revisão de código antes de um pull request ser aprovado na branch main. Para isso deve-se marcar com checkbox as opções “Require a pull request before merging” e “Require approvals”.

No campo Branch name pattern, deve-se colocar main, pois assim as regras serão aplicadas a qualquer branch que possua main no nome.

EX: maintest, mainexample



O que é o Ficheiro README.md?

O ficheiro README é normalmente o primeiro ficheiro que os utilizadores lêem. É um ficheiro de texto que contém informação para o utilizador sobre o software, projeto, código, ou pode conter instruções, ajuda, ou detalhes sobre as correções ou atualizações. Criação do Ambiente Git Flow

O que é o Git Flow?

Gitflow dá-nos um conjunto de regras de como trabalhar com branches Git individuais, atribuindo-lhes funções específicas e definindo como interagem entre si.

Como iniciar o Git Flow?

Para iniciar o Git Flow no repositório que foi criado localmente, basta usar o seguinte comando:

```
git flow init
```

Deve-se manter todos os nomes das branches que vêm por *default*.

```
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? []
```

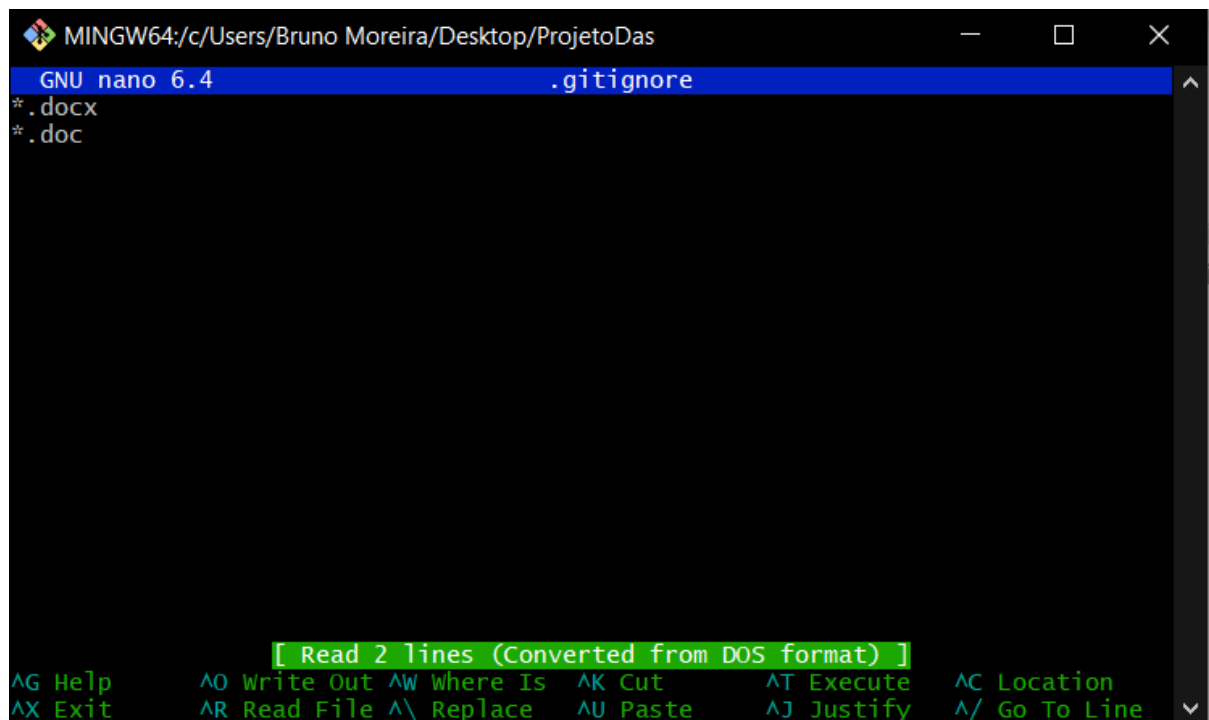
Ficheiro .gitignore

Quando é necessário ignorar um ou vários ficheiros utiliza-se um ficheiro chamado **.gitignore**. Neste caso deseja-se ignorar todos os ficheiros com a extensão **.docx** e **.docs**. Para criar o ficheiro .gitignore usa-se os seguintes comandos:

```
git checkout main
nano .gitignore
cat .gitignore
git add .
git commit -m "Add .gitignore"
git push -u origin main
git checkout develop
git pull origin main
```

Utiliza-se o * para indicar que os ficheiros ignorados podem ter qualquer nome desde que usem uma das extensões mencionadas.

A combinação de teclas Ctrl + O e de seguida Enter guarda o ficheiro, o Ctrl X sai do modo de edição.



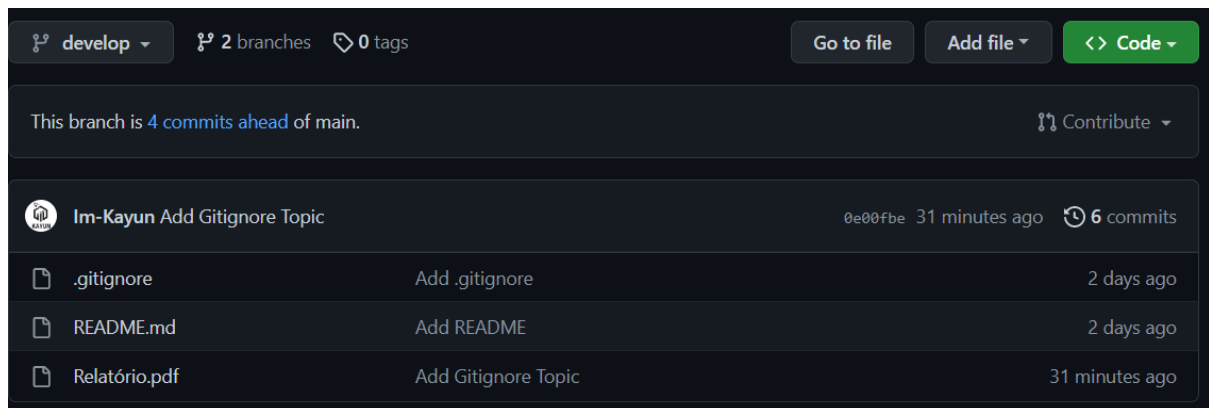
```
MINGW64:/c/Users/Bruno Moreira/Desktop/ProjetoDas
GNU nano 6.4 .gitignore
*.docx
*.doc

[ Read 2 lines (Converted from DOS format) ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```


Relatório no Github

Para enviar o relatório para o repositório remoto no Github é bastante simples. Para isso deve-se usar os seguintes comandos:

```
git checkout develop
[Adicionar o ficheiro à pasta]
git add .
git commit -m "Relatório Inicial"
git push origin develop
```



Criação de Features

Para criar e terminar uma feature usa-se os seguintes comandos:

```
git flow feature start nome_feature
git flow feature finish nome_feature
```

Seguindo esse exemplo serão criadas 5 features:

Feature 1:

```
git flow feature start RelatorioOrganizacao
[COLOCAR AS ALTERAÇÕES NA PASTA]
git add .
git commit -m "Add Organização Topic"
git flow feature finish RelatorioOrganizacao
git push origin develop
```

Feature 2:

```
git flow feature start RelatorioRepositorio
[COLOCAR AS ALTERAÇÕES NA PASTA]
git add .
git commit -m "Add Repositorio Topic"
git flow feature finish RelatorioRepositorio
git push origin develop
```

Feature 3:

```
git flow feature start RelatorioGitignore  
[COLOCAR AS ALTERAÇÕES NA PASTA]  
git add .  
git commit -m "Add Gitignore Topic"  
git flow feature finish RelatorioGitignore  
git push origin develop
```

Feature 4:

```
git flow feature start RelatorioAdd  
[COLOCAR AS ALTERAÇÕES NA PASTA]  
git add .  
git commit -m "Add Relatorio Topic"  
git flow feature finish RelatorioAdd  
git push origin develop
```

Feature 5:

```
git flow feature start RelatorioFeatures  
[COLOCAR AS ALTERAÇÕES NA PASTA]  
git add .  
git commit -m "Add Features Topic"  
git flow feature finish RelatorioFeatures  
git push origin develop
```

Feature 6:

```
git flow feature start RelatorioRelHot  
[COLOCAR AS ALTERAÇÕES NA PASTA]  
git add .  
git commit -m "Add Release and Hotfix Topics"  
git flow feature finish RelatorioRelHot  
git push origin develop
```

Criação de uma Release

Para criar uma release usa-se os seguintes comandos:

```
git flow release start REL_1.0 develop  
[COLOCAR AS ALTERAÇÕES NA PASTA]  
git add .  
git commit -m "Release"  
git flow release publish REL_1.0  
git flow release finish REL_1.0 -m "Release 1.0"  
git push origin main
```

Criação de um Hotfix

Para criar um hotfix usa-se os seguintes comandos:

[Adicionar o ficheiro à pasta]

```
git add .  
git commit -m "Hotfix"  
git push origin develop  
git checkout main  
git push origin develop  
git checkout main  
git pull origin develop  
git push origin main
```