Universidad Europea

Git

November 2024

Go Further

Index

00

Basic Things

Obvious Steps

Basic tools you'll need

Visual Studio Code

Git for Windows

- **Create a New Project in Anaconda**
- **Create a blank folder in your Desktop**
- **Connect Visual Studio Code to the blank folder**

01

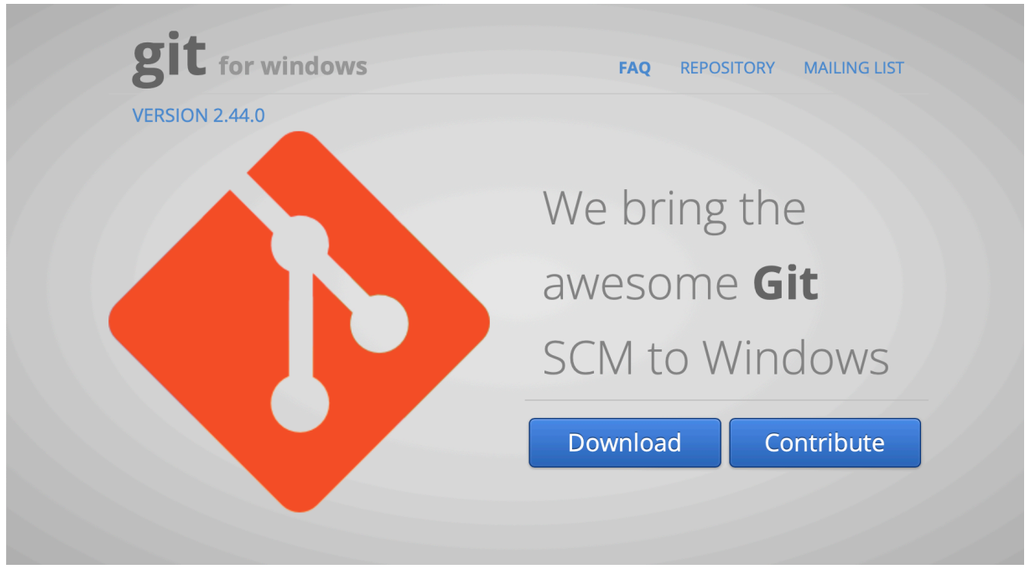
Gitting Started

Installation

In order to use git, we first must install it

Windows

If you're a Windows user, I'd recommend you use Git for Windows, as its easy to install, and features many useful things, like having a Bash Emulator that will make the rest of this course easier.



Git for Windows

- **Git BASH**
- **Git GUI**
- **Shell Integration**
- **Git Credential Manager**

Installation

If you're using an Apple laptop, the best idea is to use Homebrew to install anything.

Once you've installed homebrew simply open a terminal and run the line

brew install git

Configure

A correct configuration will make our lives easier

We usually use git to collaborate with others, and to do so we must first identify ourselves.
We open a **Terminal** on Visual Studio Code, and check how we are identified:

```
(Git_BI) alexander@Cientefico Business Intelligence % git config --list
credential.helper=osxkeychain
```

Most probably, as it is the first time we use git, we'll have to add a username and email to the configuration

```
git config --global user.email "alejandro.perdiguero@universidadeuropea.es"
```

```
git config --global user.name "perdiguero"
```

Once added, the result should look like this:

```
(Git_BI) alexander@Cientefico Business Intelligence % git config --list
credential.helper=osxkeychain
user.name=perdiguero
user.email=alejandro.perdiguero@universidadeuropea.es
```

02

Basic Commands

Index

- Basic Commands
  - init
  - status
  - add
  - commit

Initialising

The usual first step is to initialise the repo

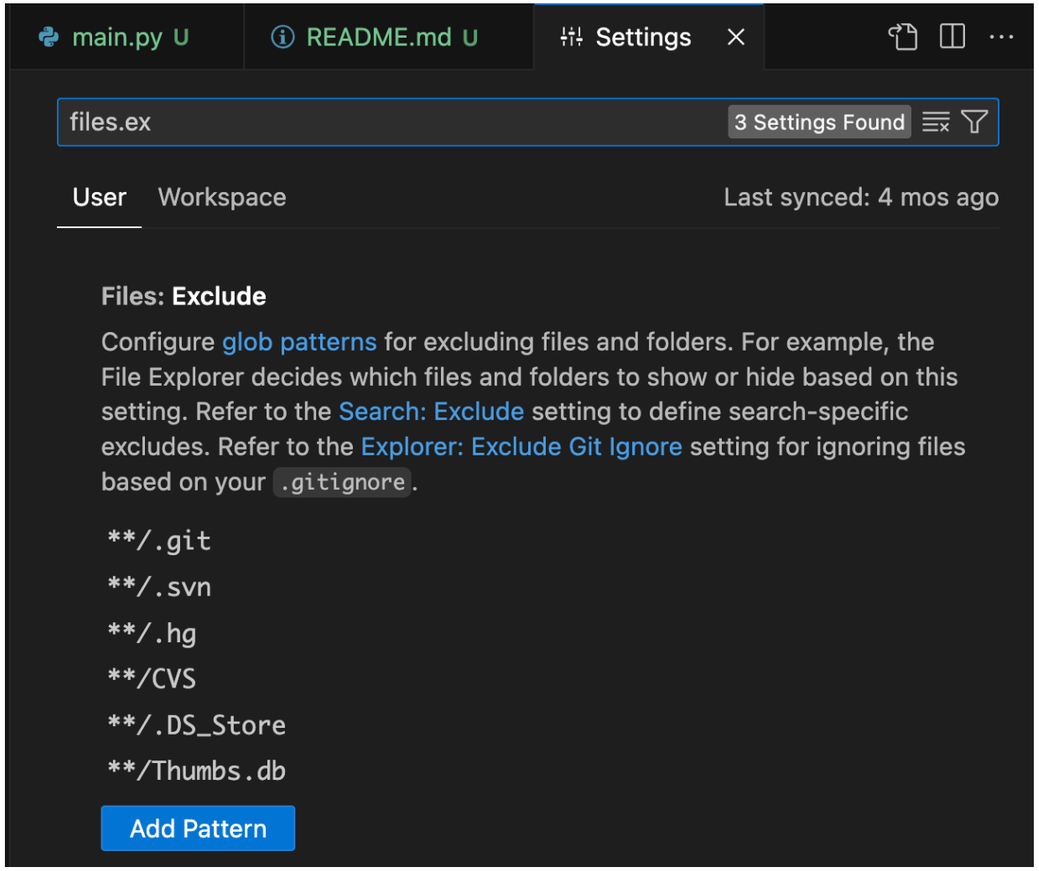
To do so, we'll use the simple command **git init** in the terminal

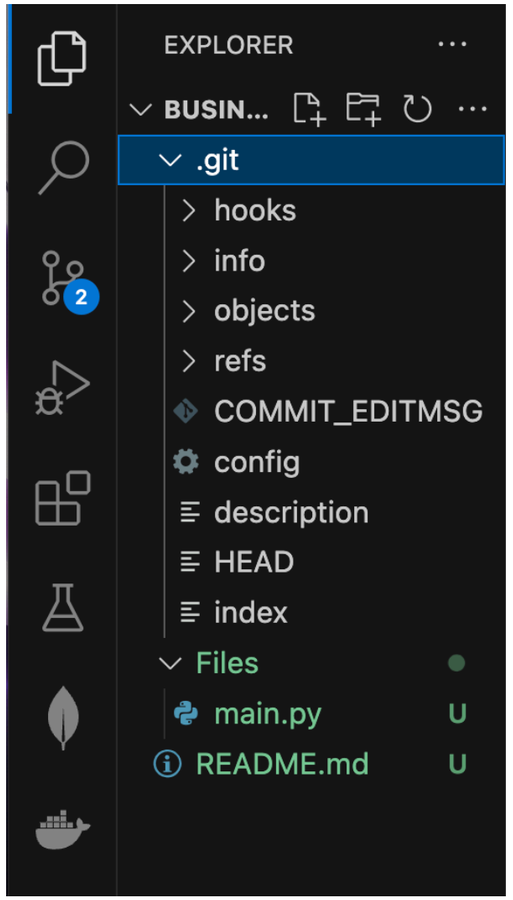
```
(Git_BI) alexander@Cientefico Business Intelligence % git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /Users/alexander/Desktop/Business Intelligence/.git/
```

Something else changed!

Initialising

Settings to change in order to see things





Status Check

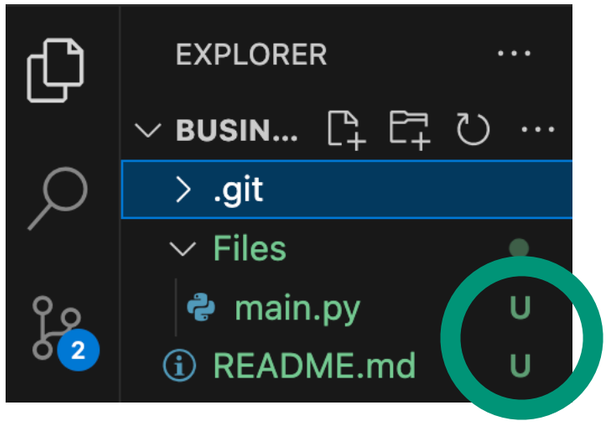
Need to check how things are going

```
(Git_BI) alexander@Cientefico Business Intelligence % git status
On branch master

No commits yet

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

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



In VS Code, a green file means its a new file, and a U means Untracked(meaning we won't be able to see the diff using git)

Universidad Europea

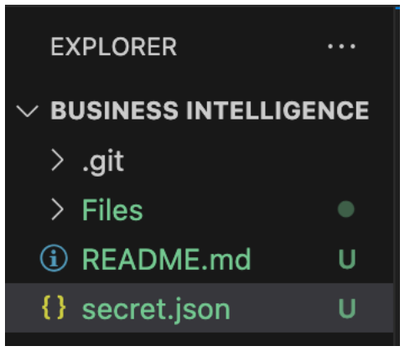
Status Check

Do we always want everything to be modifiable?

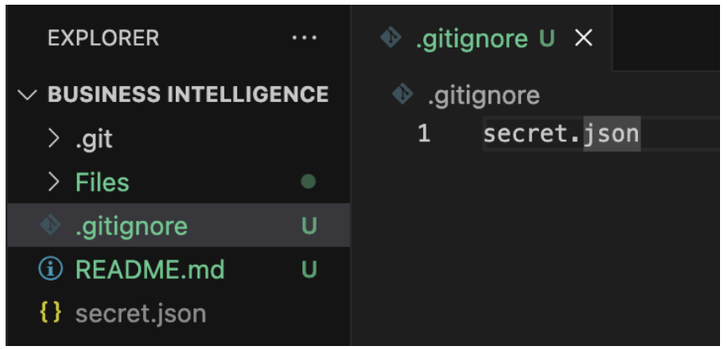
Obviously no.

What could be some examples?*

Most importantly, how do we ignore them?



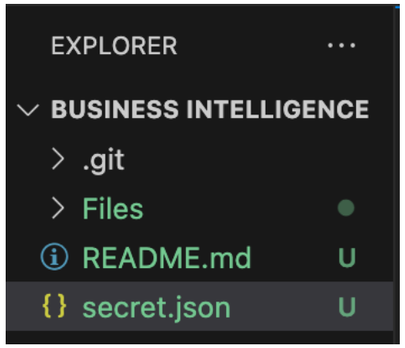
Create a new File



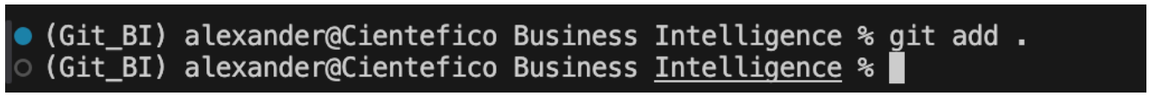
Create a -gitignore File with the file address

Adding Things - Staging Area

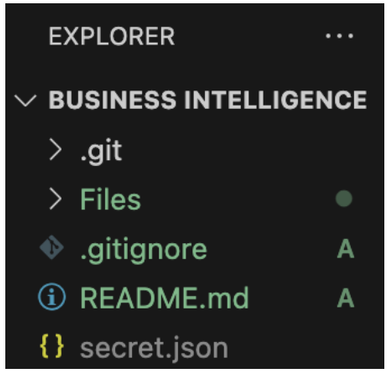
We need to add files to our repo



All files are Untracked



git add .

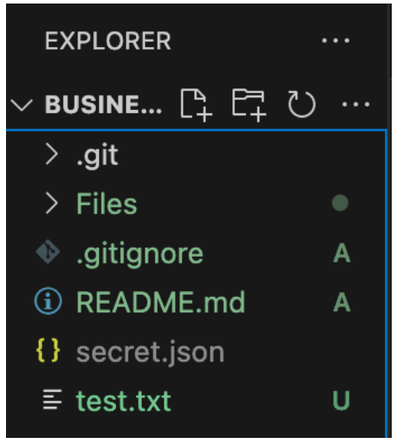


All files are Added

Basic way of doing it, add every single thing

Adding Things - Staging Area

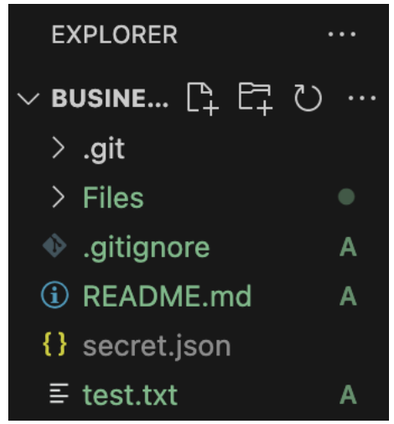
We need to add files to our repo



Our new file is Untracked

```
(Git_BI) alexander@Cientefico Business Intelligence % git add test.txt
(Git_BI) alexander@Cientefico Business Intelligence %
```

git add test.txt

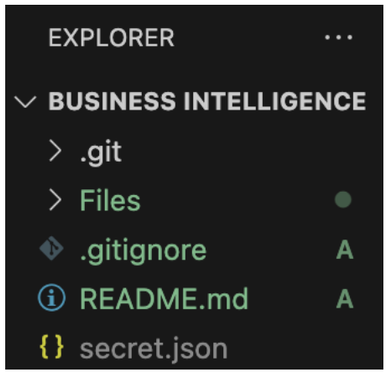


All files are Added

We can add a single file in an easy way

!Adding Things - Staging Area

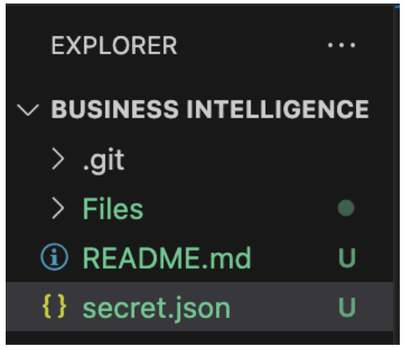
We might need to delete files from the repo



All files are Added

```
(Git_BI) alexander@Cientefico Business Intelligence % git reset .
(Git_BI) alexander@Cientefico Business Intelligence %
```

git reset .

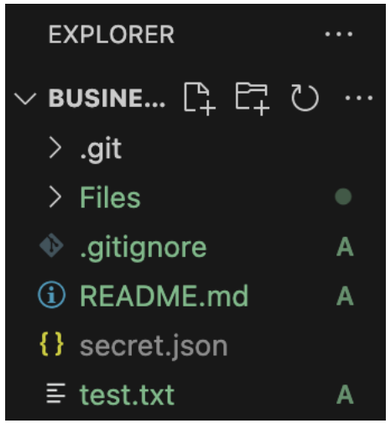


All files are Untracked

Basic way of doing it, resetting every single thing

!Adding Things - Staging Area

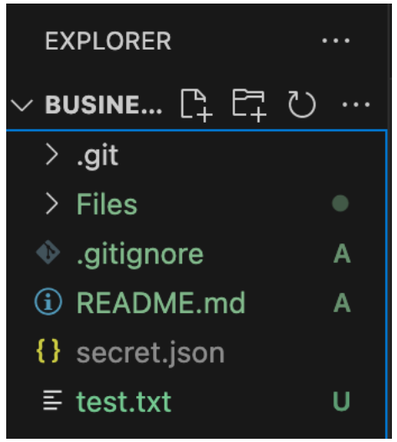
We might need to delete files from the repo



All Files are Added

```
(Git_BI) alexander@Cientefico Business Intelligence % git reset .
(Git_BI) alexander@Cientefico Business Intelligence %
```

git reset test.txt

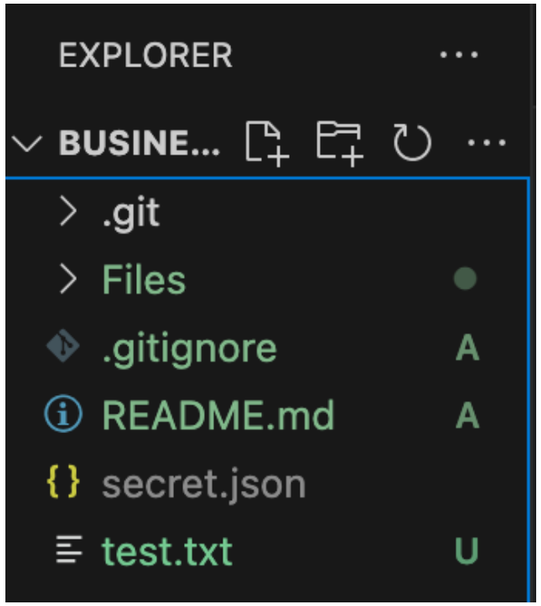


Our selected file is Untracked

Also applied to single files

Status Check

Need to check how things are going



Our selected file is Untracked

```
(Git_BI) alexander@Cientefico Business Intelligence % git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitignore
        new file:   Files/main.py
        new file:   README.md

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

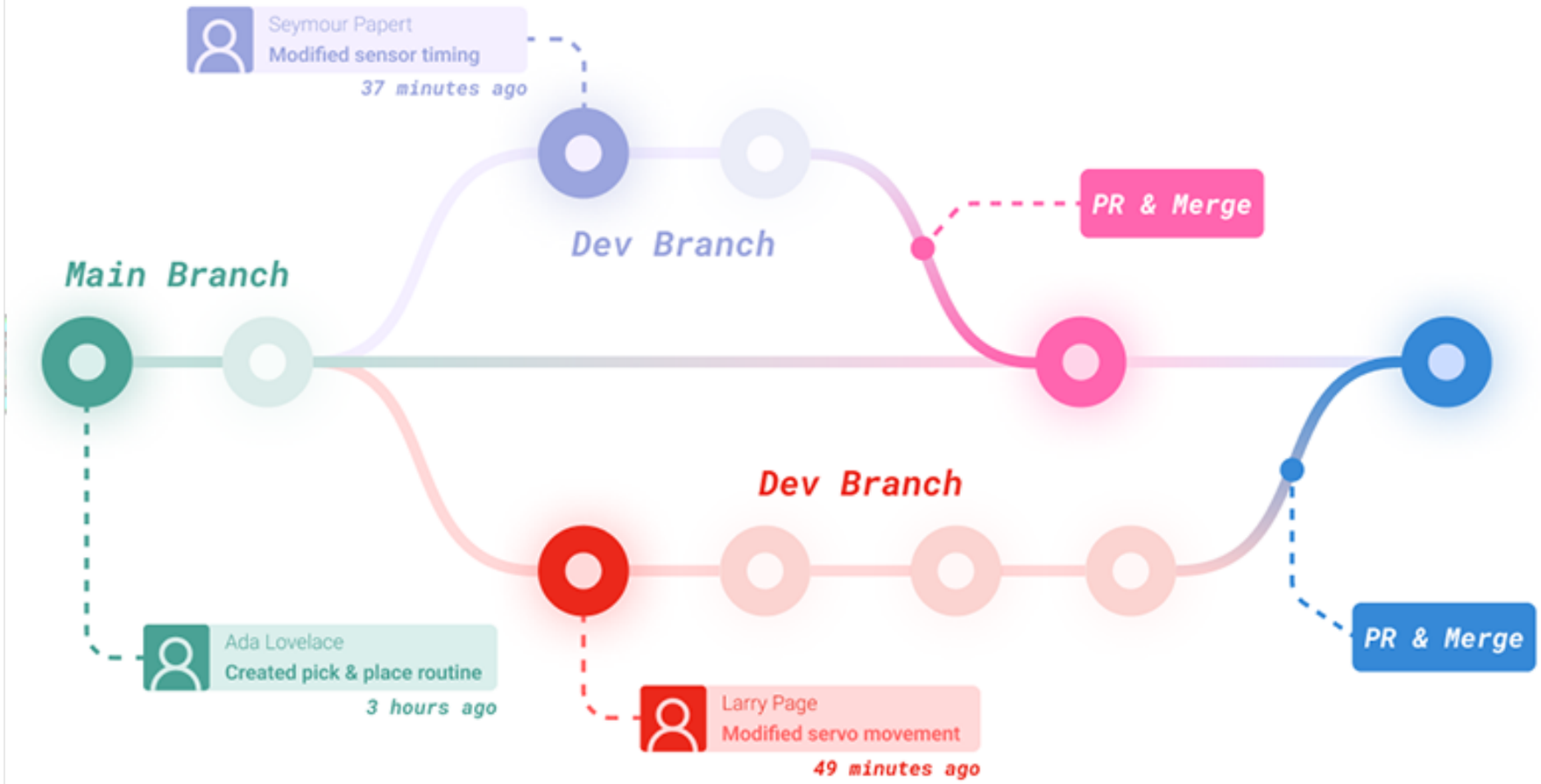
Files are ready to be committed

Committing?

What is git?

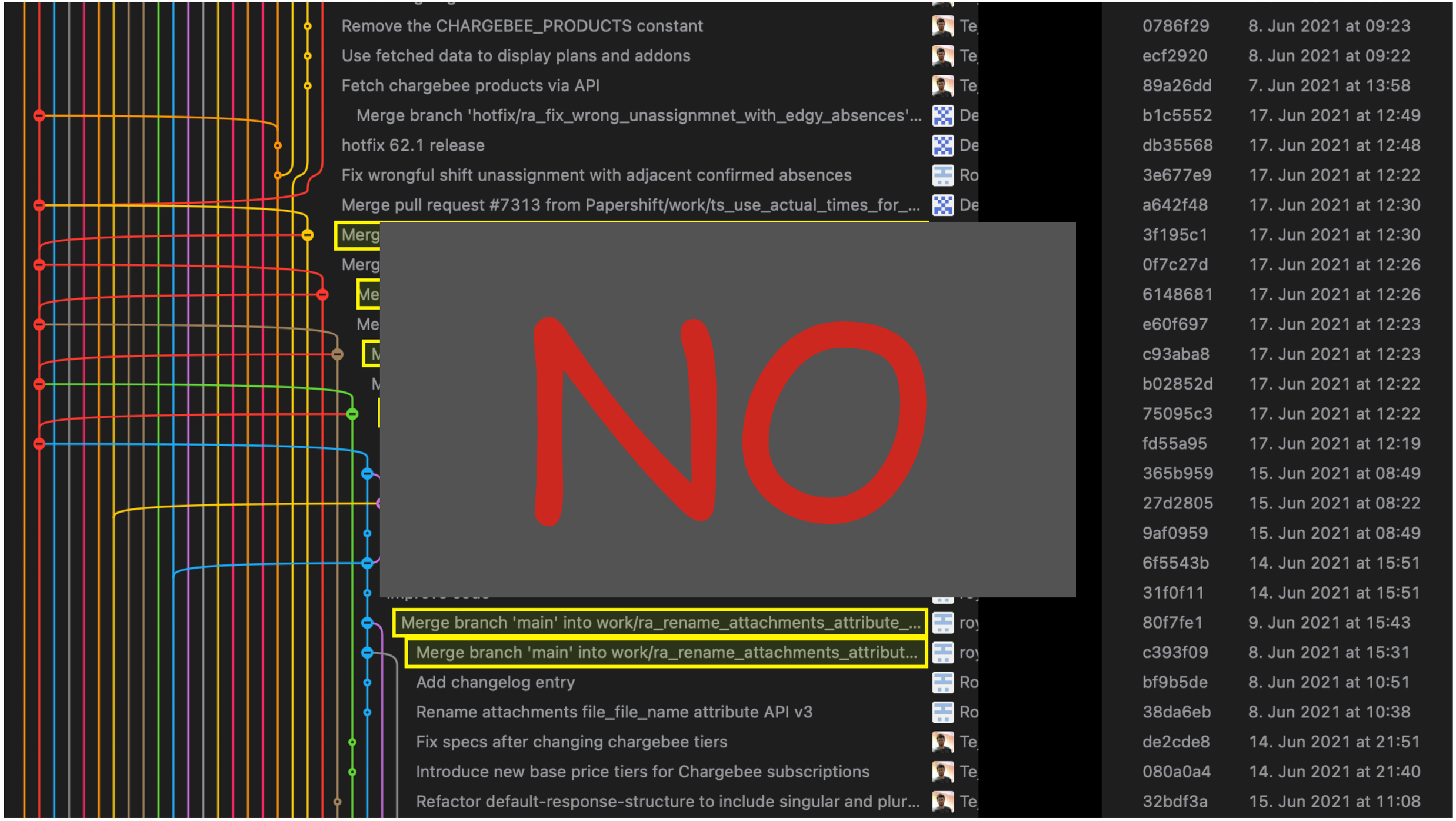
Why is git?

Because



Is it perfect?


Remove the CHARGEBEE_PRODUCTS constant 0786f29 8. Jun 2021 at 09:23
Use fetched data to display plans and addons ecf2920 8. Jun 2021 at 09:22
Fetch chargebee products via API 89a26dd 7. Jun 2021 at 13:58
Merge branch 'hotfix/ra_fix_wrong_unassignmnet_with_edgy_absences'... b1c5552 17. Jun 2021 at 12:49
hotfix 62.1 release db35568 17. Jun 2021 at 12:48
Fix wrongful shift unassignment with adjacent confirmed absences 3e677e9 17. Jun 2021 at 12:22
Merge pull request #7313 from Papershift/work/ts_use_actual_times_for_... a642f48 17. Jun 2021 at 12:30
Merge branch 'main' into work/ts_use_actual_times_for_mobile_working_... 3f195c1 17. Jun 2021 at 12:30
Merge pull request #7342 from Papershift/work/tb_pricing_v4 0f7c27d 17. Jun 2021 at 12:26
Merge branch 'main' into work/tb_pricing_v4 6148681 17. Jun 2021 at 12:26
Merge pull request #7276 from Papershift/work/tb_working_area_resour... e60f697 17. Jun 2021 at 12:23
Merge branch 'main' into work/tb_working_area_resource_naming c93aba8 17. Jun 2021 at 12:23
Merge pull request #7356 from Papershift/work/tb_chargebee_new_ti... b02852d 17. Jun 2021 at 12:22
Merge branch 'main' into work/tb_chargebee_new_tiers 75095c3 17. Jun 2021 at 12:22
Merge pull request #7316 from Papershift/work/ra_rename_attachm... fd55a95 17. Jun 2021 at 12:19
Merge remote-tracking branch 'origin/work/ra_rename_attachmen... 365b959 15. Jun 2021 at 08:49
Merge branch 'main' into work/ra_rename_attachments_attribut... 27d2805 15. Jun 2021 at 08:22
Add tests 9af0959 15. Jun 2021 at 08:49
Merge branch 'main' into work/ra_rename_attachments_attribut... 6f5543b 14. Jun 2021 at 15:51
Improve code 31f0f11 14. Jun 2021 at 15:51
Merge branch 'main' into work/ra_rename_attachments_attribute_... 80f7fe1 9. Jun 2021 at 15:43
Merge branch 'main' into work/ra_rename_attachments_attribut... c393f09 8. Jun 2021 at 15:31
Add changelog entry bf9b5de 8. Jun 2021 at 10:51
Rename attachments file_file_name attribute API v3 38da6eb 8. Jun 2021 at 10:38
Fix specs after changing chargebee tiers de2cde8 14. Jun 2021 at 21:51
Introduce new base price tiers for Chargebee subscriptions 080a0a4 14. Jun 2021 at 21:40
Refactor default-response-structure to include singular and plur... 32bdf3a 15. Jun 2021 at 11:08

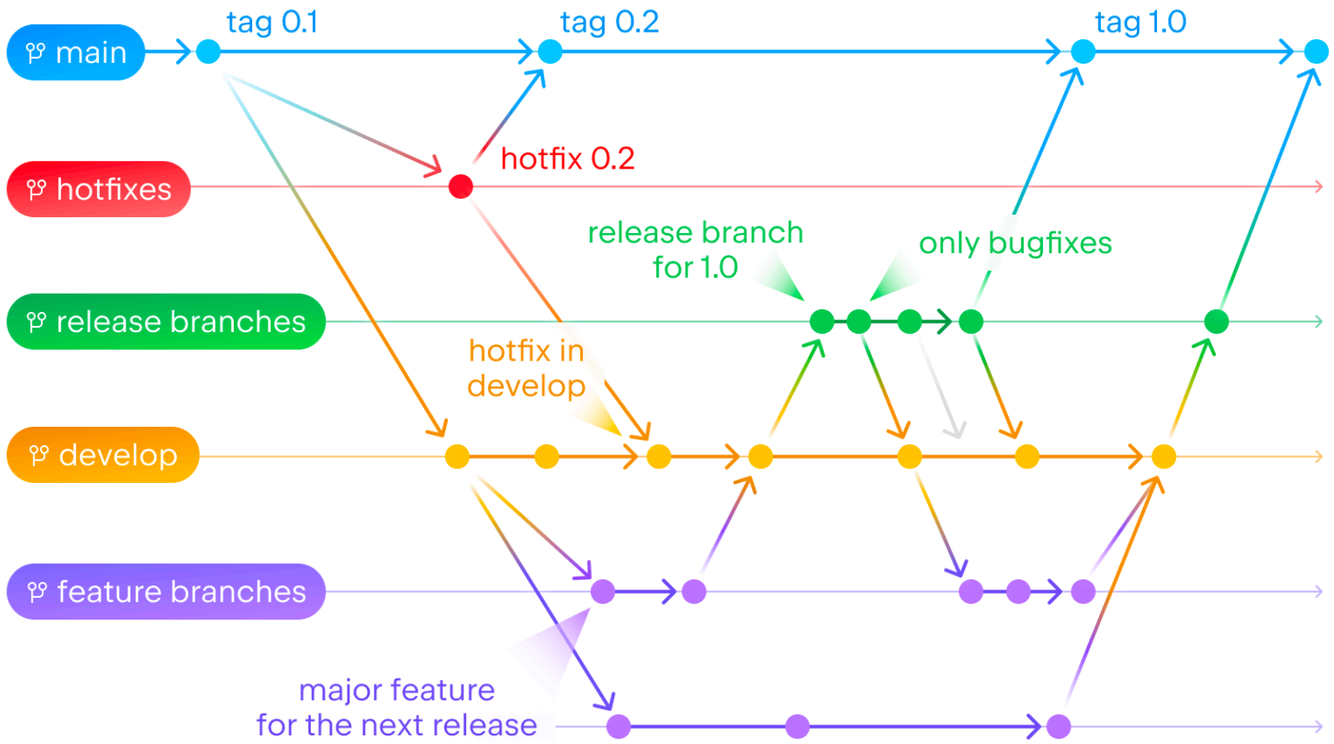

Remove the CHARGEBEE_PRODUCTS constant
Use fetched data to display plans and addons
Fetch chargebee products via API
Merge branch 'hotfix/ra_fix_wrong_unassignmnet_with_edgy_absences'...
hotfix 62.1 release
Fix wrongful shift unassignment with adjacent confirmed absences
Merge pull request #7313 from Papershift/work/ts_use_actual_times_for_...
NO
Merge branch 'main' into work/ra_rename_attachments_attribute_...
Merge branch 'main' into work/ra_rename_attachments_attribut...
Add changelog entry
Rename attachments file_file_name attribute API v3
Fix specs after changing chargebee tiers
Introduce new base price tiers for Chargebee subscriptions
Refactor default-response-structure to include singular and plur...
0786f29 8. Jun 2021 at 09:23
ecf2920 8. Jun 2021 at 09:22
89a26dd 7. Jun 2021 at 13:58
b1c5552 17. Jun 2021 at 12:49
db35568 17. Jun 2021 at 12:48
3e677e9 17. Jun 2021 at 12:22
a642f48 17. Jun 2021 at 12:30
3f195c1 17. Jun 2021 at 12:30
0f7c27d 17. Jun 2021 at 12:26
6148681 17. Jun 2021 at 12:26
e60f697 17. Jun 2021 at 12:23
c93aba8 17. Jun 2021 at 12:23
b02852d 17. Jun 2021 at 12:22
75095c3 17. Jun 2021 at 12:22
fd55a95 17. Jun 2021 at 12:19
365b959 15. Jun 2021 at 08:49
27d2805 15. Jun 2021 at 08:22
9af0959 15. Jun 2021 at 08:49
6f5543b 14. Jun 2021 at 15:51
31f0f11 14. Jun 2021 at 15:51
80f7fe1 9. Jun 2021 at 15:43
c393f09 8. Jun 2021 at 15:31
bf9b5de 8. Jun 2021 at 10:51
38da6eb 8. Jun 2021 at 10:38
de2cde8 14. Jun 2021 at 21:51
080a0a4 14. Jun 2021 at 21:40
32bdf3a 15. Jun 2021 at 11:08

Universidad Europea

Committing

What is going on with the branches?

Git flow

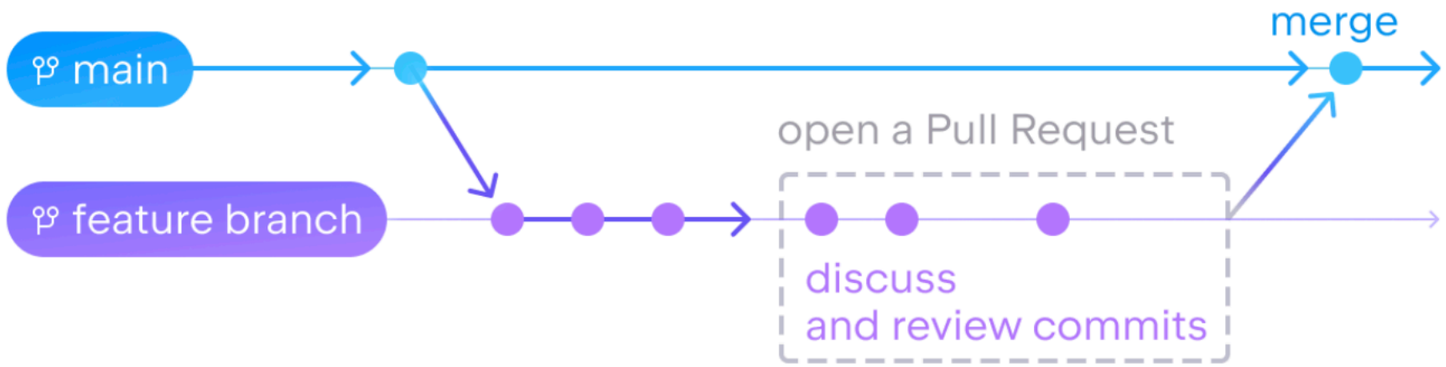


- The `main` branch is for production code only.
- The `develop` branch is for development code.
- `feature` branches are created from the `develop` branch.
- `hotfix` branches are created from the `main` branch.
- `release` branches are created from the `develop` branch.

Committing

What is going on with the branches?

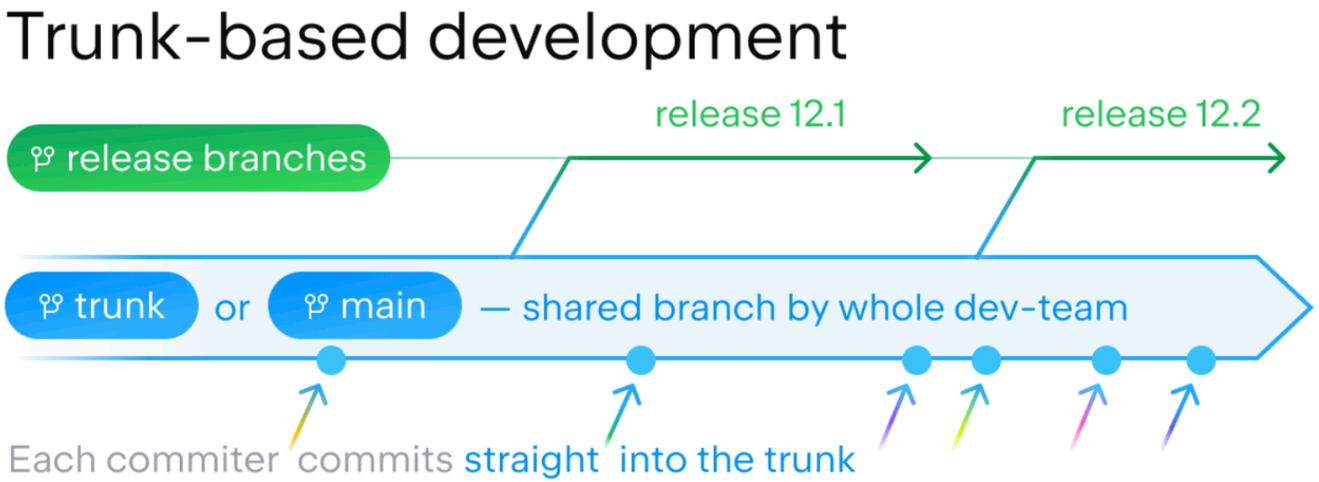
GitHub flow



- The `main` branch is for production code only.
- Development is done in separate `feature` branches which are then merged into the `main` branch when ready.
- Pull requests are used for code reviews and must satisfy branch protection rules before being merged.
- Releases are cut from the `main` branch and tagged for easy versioning.

Committing

What is going on with the branches?



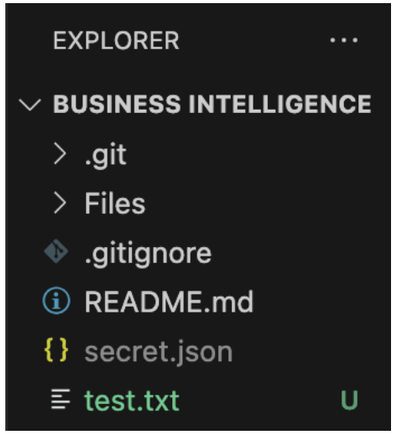
- All code is committed to a single branch, usually called `trunk`.
- The `trunk` branch is always production-ready.
- Development work is done directly on the `trunk` branch.
- There aren't any long-lived `feature` branches.
- Before pushing changes, developers perform a full pre-integration build (compile, unit tests, etc.) on their local machines.
- After the changes are already in `trunk`, there may be a post-integration code review.

Committing

Okay, now the fun part

```
(Git_BI) alexander@Cientefico Business Intelligence % git commit -m 'Mi Primerita Vez'
[master (root-commit) 319f219] Mi Primerita Vez
 3 files changed, 5 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 Files/main.py
 create mode 100644 README.md
(Git_BI) alexander@Cientefico Business Intelligence %
```

We run a simple command: git commit -m with a message



All files that were added have changed colour

```
(Git_BI) alexander@Cientefico Business Intelligence % git status
On branch master
nothing to commit, working tree clean
(Git_BI) alexander@Cientefico Business Intelligence %
```

Committing

Who's done what?

```
(Git_BI) alexander@Cientefico Business Intelligence % git log
commit 319f219a4449f49d143b52f5d0c0393f0180a1e1 (HEAD -> master)
Author: perdiguero <alejandro.perdiguero@universidadeuropea.es>
Date:   Thu Feb 29

    Mi Primerita Vez
(Git_BI) alexander@Cientefico Business Intelligence %
```

Is this the best way of logging things?

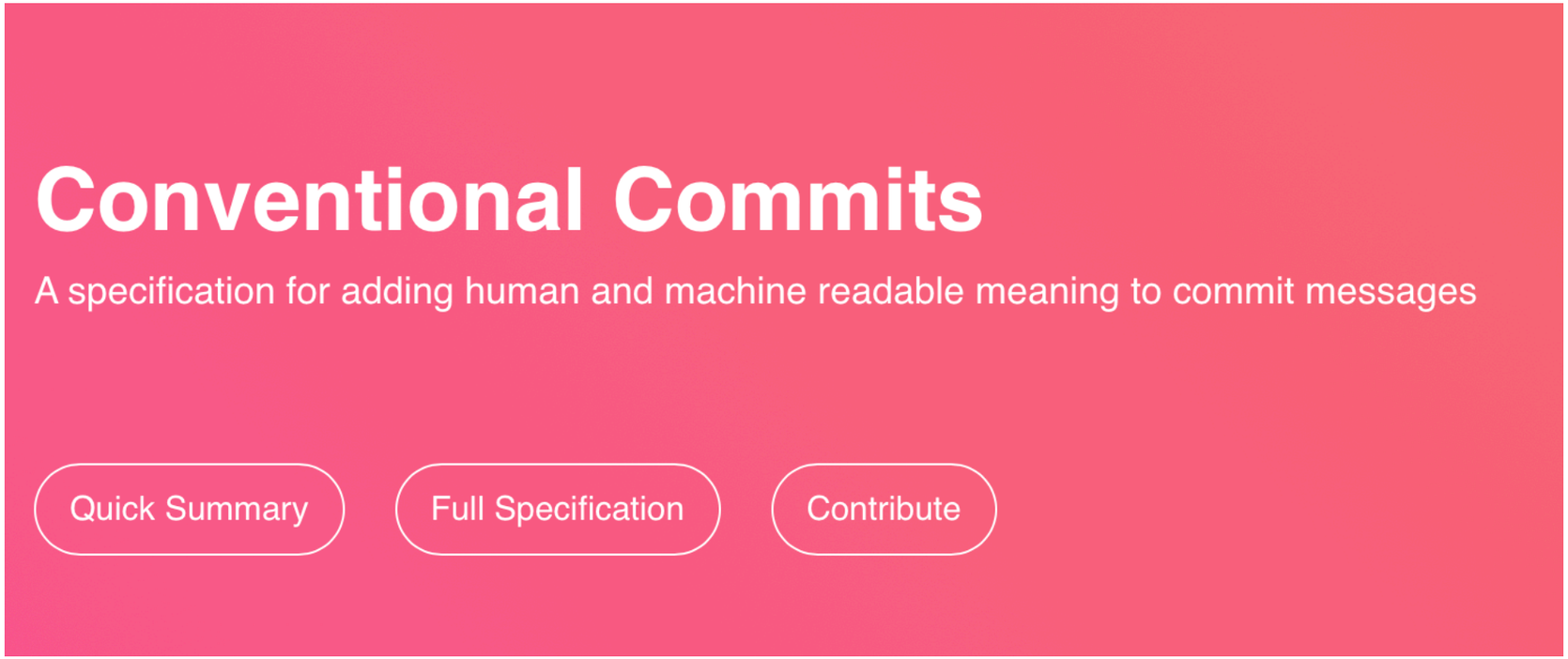
Committing

Who's done what?

Is this the best way of logging things?

Committing

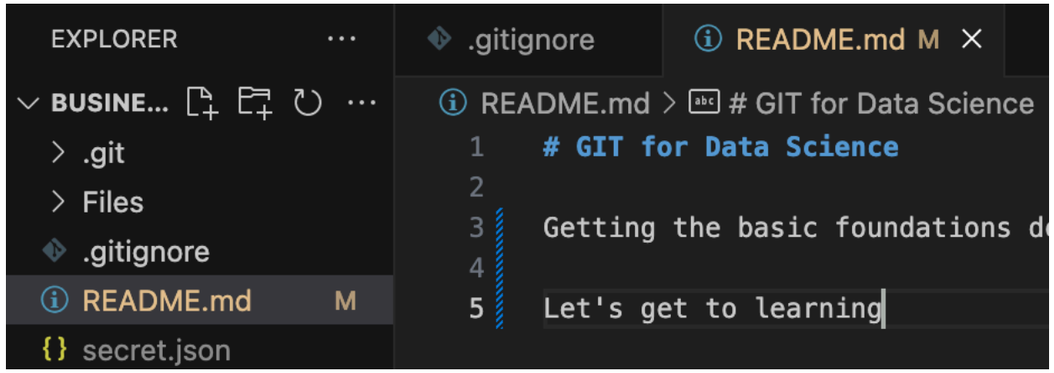
Proper Message Structure



YES

Committing

What about modifications to files?



We modified one of the files on the repo

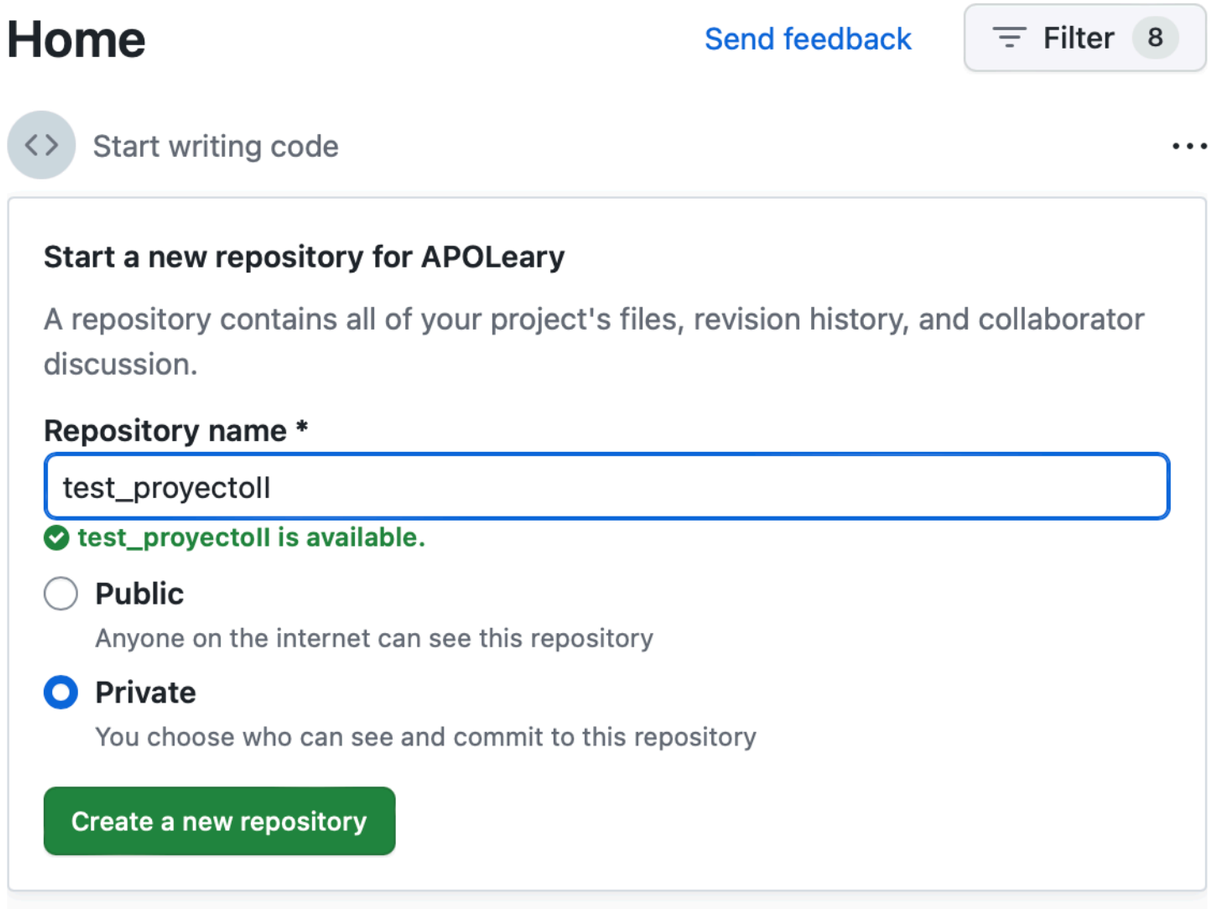
```
(Git_BI) alexander@Cientefico Business Intelligence % git commit -a -m 'escrito másmejor'
[master 4a57fc9] escrito másmejor
 1 file changed, 3 insertions(+), 1 deletion(-)
(Git_BI) alexander@Cientefico Business Intelligence %
```

03

Remote Working

Remote

We already created a repo on GitHub using Visual Studio, can we do it the opposite way?



The next step will be to sync this new repo

....

But, didn't we do this already?

Remote

Yes!!!

We've done this already, we can check this with the **git remote** function:

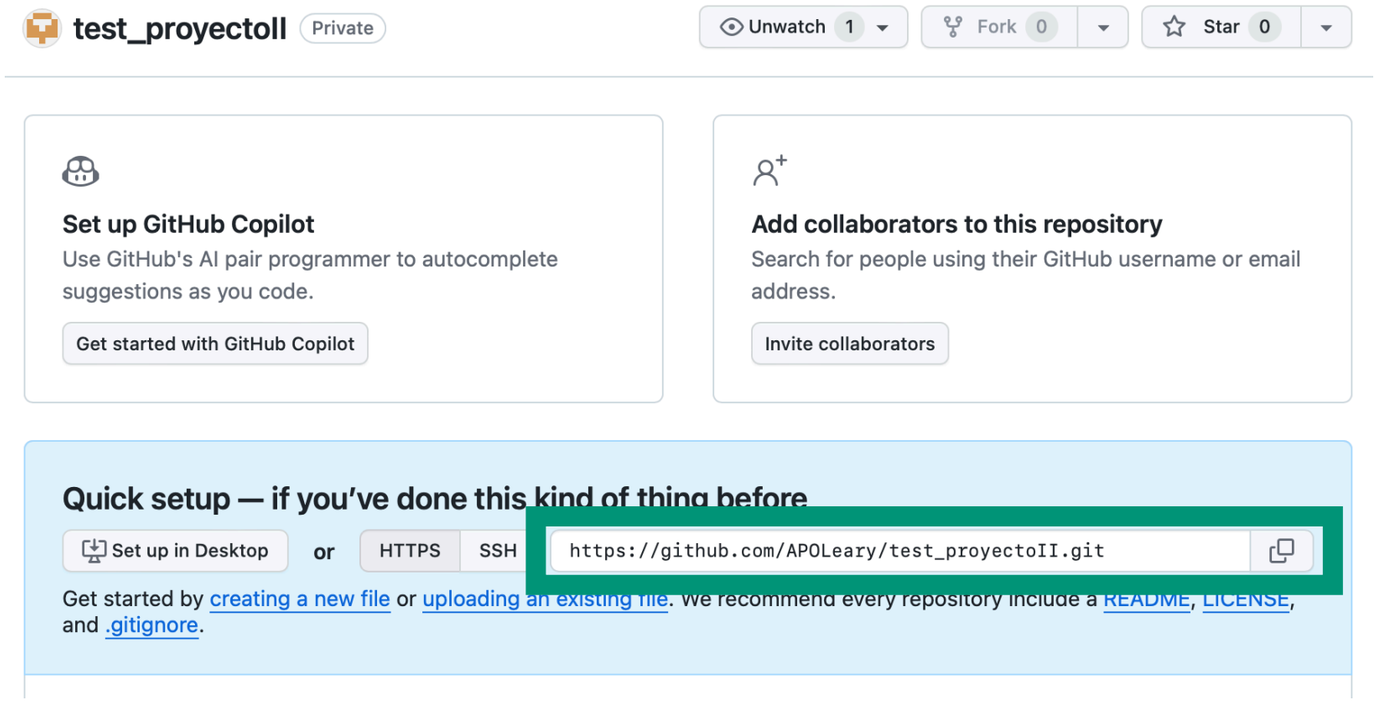
```
(base) alexander@Cientefico Business Intelligence % git remote
origin
(base) alexander@Cientefico Business Intelligence %
```

Obviously this is not enough info, so we can get more info using **git remote show origin**

```
(base) alexander@Cientefico Business Intelligence % git remote show origin
* remote origin
  Fetch URL: https://github.com/APOLeary/Business-Intelligence.git
  Push  URL: https://github.com/APOLeary/Business-Intelligence.git
  HEAD branch: master
  Remote branch:
    master tracked
  Local branch configured for 'git pull':
    master merges with remote master
  Local ref configured for 'git push':
    master pushes to master (up to date)
(base) alexander@Cientefico Business Intelligence %
```

Remote

We'll copy the url from the Quick Setup section



Universidad Europea

Tips

A correct configuration will make our lives easier

Trying to edit the configuration list because you messed up?

- **git config --global --edit**
-