

Uploading project to Github & submitting your Github URL to Moodle – CLI version

NOTE: You will need “git” installed at the command line for your operating system (it is installed on all the college computers already, like php)

Get Git from:

<https://git-scm.com/downloads>

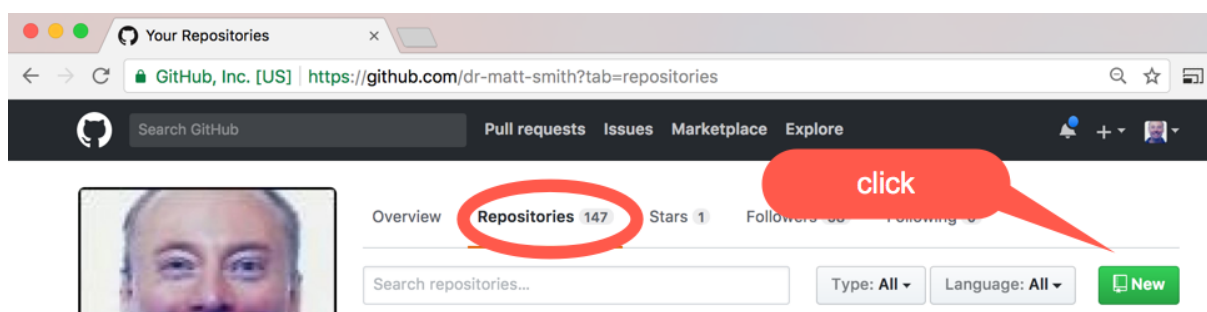
Summary for working at the CLI:

- Create new empty PRIVATE project on Github (with a README)
- Copy URL to clipboard
- At Terminal clone project to local computer
git clone <url>
- Cd into cloned folder
- Copy into cloned folder the files to be uploaded
- Track all new files
git add .
- Commit tracked files to repository “snapshot”
git commit -m “added files to project”
- Upload new commit contents to Github website
git push origin master
- Add **dr-matt-smith** as collaborator

That’s it !!!! You should then see on Github that your project files have been “committed” and uploaded

1. New repository create button

Go to the Repositories section of your Github web page and click “NEW”



2. New PRIVATE repository details

Enter a name for your project (e.g. php project) – note that any spaces in the name will become “-” (minus-signs) in the URL and ‘official’ repository name


Click PRIVATE – important, so the code is just for you (and any collaborators you add)

Add a README – this means you can easily work with the Git command line if you wish to

Create a new repository

A repository contains all the files for your project, including the

Owner

 dr-matt-smith ▾

Repository name

php project ✓

name (spaces
replaced by “-”

Great repository names are

Your new repository will be created as **php-project** it curly-octo-giggle.

Description (optional)

PHP project



Public

Anyone can see this repository and choose to clone it.



Private

You choose who can see and commit to this repository.

chose PRIVATE

add a README



Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license

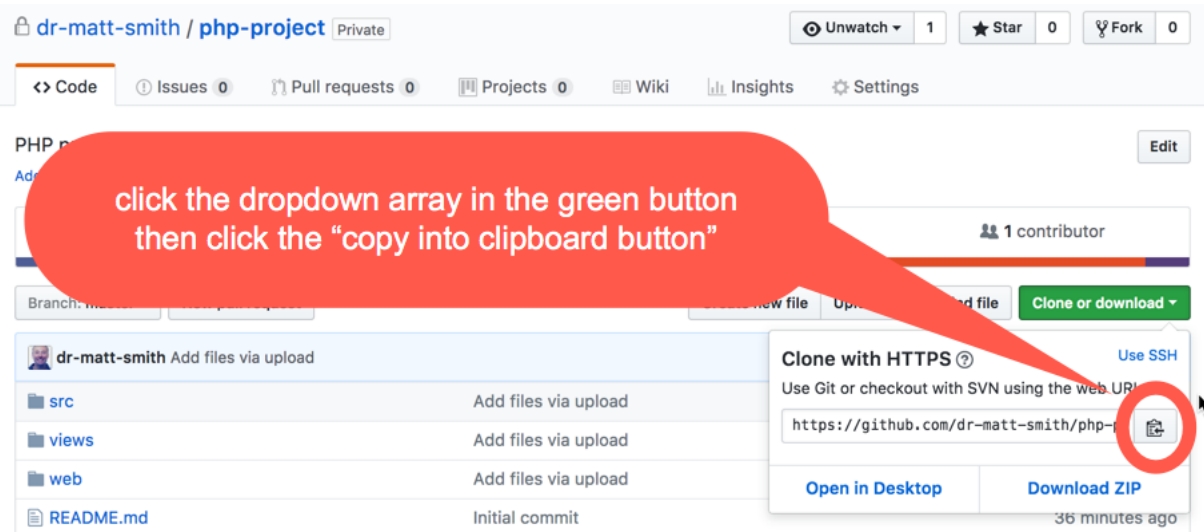
click to create

Create repository

3. Getting URL for “cloning” to local computer

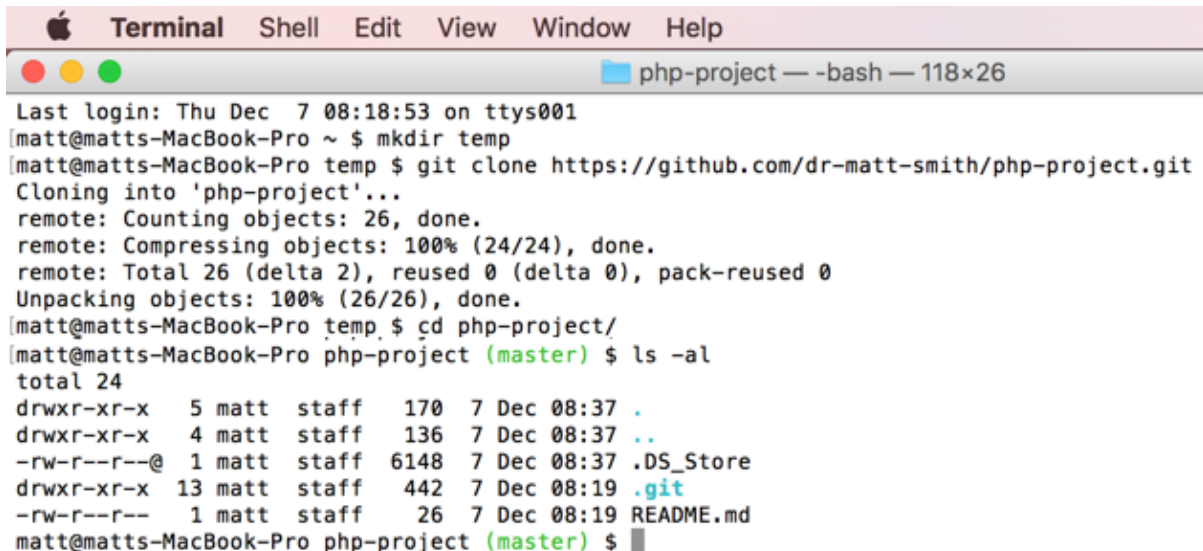
Once you have created your PRIVATE Github repository you can do all the other steps from the command line.

Get the special URL by clicking the dropdown array for the green button “Clone or Download”, then copy the URL into the clipboard



4. “clone” a copy of this repository to your local computer

- work in your Terminal / CLI Command console window
create a folder where to work (temporarily) on your Github project
(e.g. **temp** or **github**)
matt> mkdir temp
- cd into your temporary folder
matt> cd temp
- cd into your temporary folder
matt/temp> git clone <url>
- now you can cd into the newly download Github project folder (same name as your project)
matt/temp> cd php-project
- you can now see a copy “clone” of the Github files on your local computer, including the special **.git** hidden folder (which contains the entire history of changes made to your project!)
matt/temp/php-project>ls -al (or 'dir' for windows!)



```
Terminal  Shell  Edit  View  Window  Help
php-project — -bash — 118x26

Last login: Thu Dec  7 08:18:53 on ttys001
[matt@matts-MacBook-Pro ~ $ mkdir temp
[matt@matts-MacBook-Pro temp $ git clone https://github.com/dr-matt-smith/php-project.git
Cloning into 'php-project'...
remote: Counting objects: 26, done.
remote: Compressing objects: 100% (24/24), done.
remote: Total 26 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (26/26), done.
[matt@matts-MacBook-Pro temp $ cd php-project/
[matt@matts-MacBook-Pro php-project (master) $ ls -al
total 24
drwxr-xr-x  5 matt  staff   170  7 Dec 08:37 .
drwxr-xr-x  4 matt  staff   136  7 Dec 08:37 ..
-rw-r--r--@ 1 matt  staff  6148  7 Dec 08:37 .DS_Store
drwxr-xr-x 13 matt  staff   442  7 Dec 08:19 .git
-rw-r--r--  1 matt  staff    26  7 Dec 08:19 README.md
[matt@matts-MacBook-Pro php-project (master) $
```

5. Copy your project files from your local computer into the newly cloned temporary folder

NOTE – do NOT copy folder: **/vendor**

Vendor can always be re-created with command: **composer install**

It can be a large folder, and it's all third party stuff and autoloader – nothing in there needs to be archived. If you do include vendor it will just slow things down and make bigger files – but it won't break your project – avoiding vendor for Github just speeds things up ...

Learn about .gitignore files – a way to have ignored folders that can be in your project folder but will NOT be added to your commits or uploaded to Github

You have your working PHP project files somewhere on your local computer, let's assume in:

matt/itb/php/sports-project

We have created a temporary clone of our Github project in:

matt/temp/php-project

We need to simply COPY all the project files into our php-project folder, then we can upload them to Github

Copy the contents of your working PHP project into our temporary Github project folder

Either (A) drag-and-drop with your systems GUI

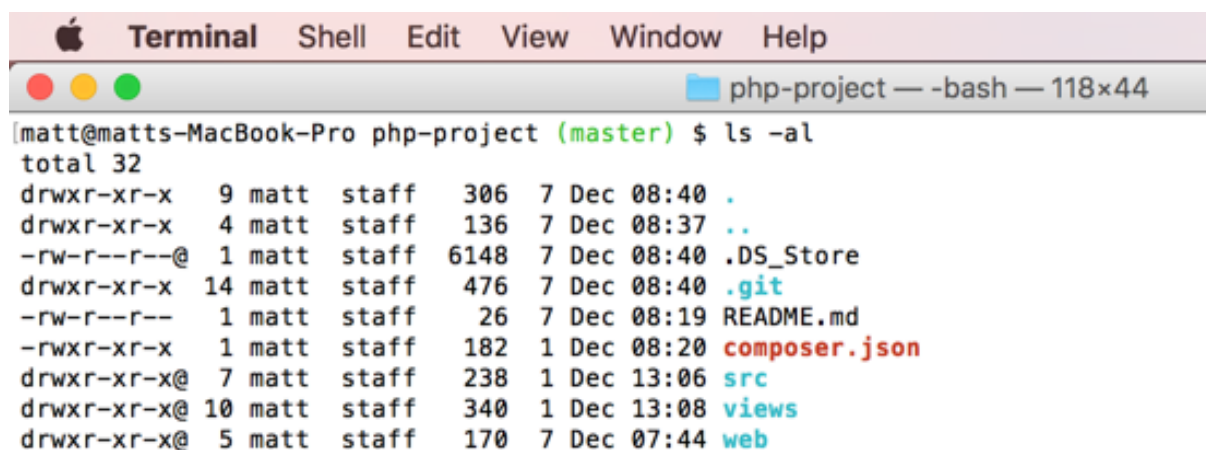
Or (B) copy the files at the command line, e.g.

- Ensure you are in your Github project folder
matt/temp/php-project>
- Use your CLI copy command (cp for linux, copy for windows) to copy FROM your working project folder into the current folder “.”), e.g.

cp matt/itb/php/sports-project/*.* .

Now list your project files – they should have been added to your Github projects files:

- **matt/temp/php-project>ls -al (or 'dir' for windows!)**



```
Terminal  Shell  Edit  View  Window  Help
php-project — -bash — 118x44
[matt@matts-MacBook-Pro php-project (master)] $ ls -al
total 32
drwxr-xr-x  9 matt  staff   306  7 Dec 08:40 .
drwxr-xr-x  4 matt  staff   136  7 Dec 08:37 ..
-rw-r--r--@ 1 matt  staff  6148  7 Dec 08:40 .DS_Store
drwxr-xr-x 14 matt  staff   476  7 Dec 08:40 .git
-rw-r--r--  1 matt  staff    26  7 Dec 08:19 README.md
-rwxr-xr-x  1 matt  staff   182  1 Dec 08:20 composer.json
drwxr-xr-x@  7 matt  staff   238  1 Dec 13:06 src
drwxr-xr-x@ 10 matt  staff   340  1 Dec 13:08 views
drwxr-xr-x@  5 matt  staff   170  7 Dec 07:44 web
```

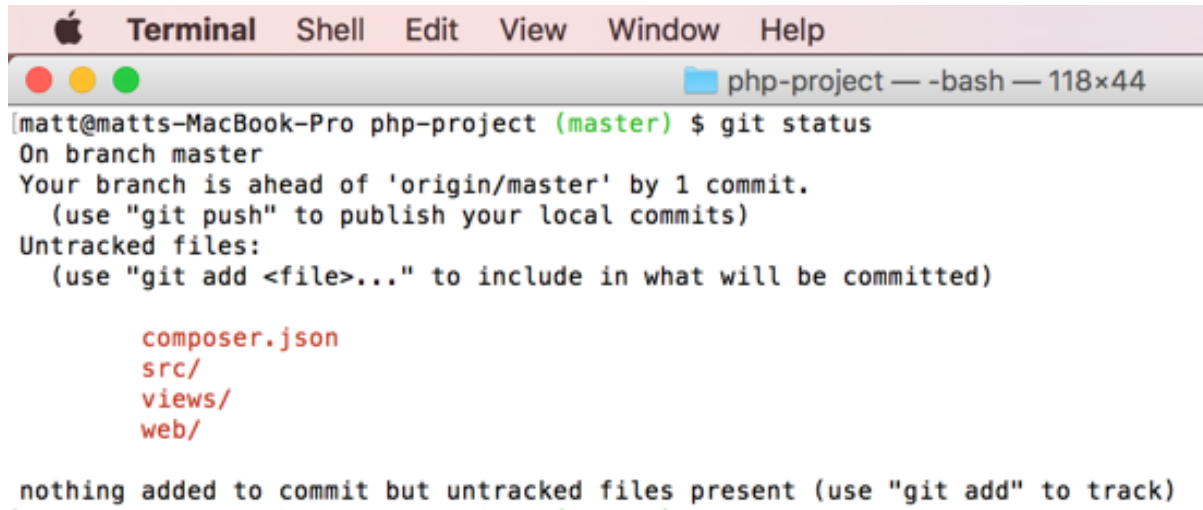
6. Check status of files in your repo folder: git status

Type

git status

to see status of files/folders at any time:

Files in **red** (if your CLI has colour settings on) need to be added (or ignored)



```
Terminal  Shell  Edit  View  Window  Help
php-project — -bash — 118x44
[matt@matts-MacBook-Pro php-project (master) $ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        composer.json
        src/
        views/
        web/

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

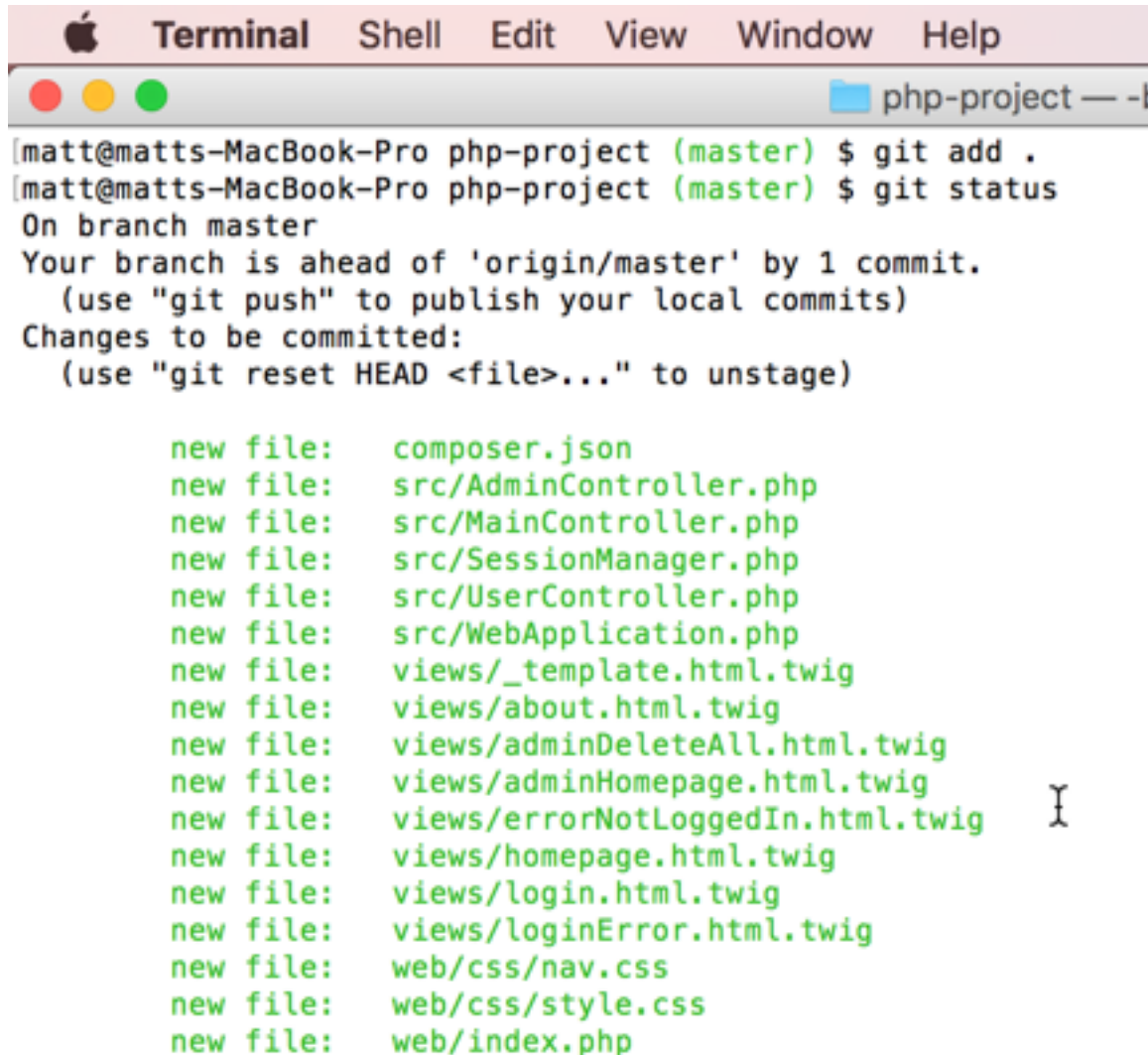
7. Add files to be 'tracked': git add

Type

git add .

to add all untracked changed files to the list to be tracked

The check status again – all those files/folders should be **green** now!

A screenshot of a macOS Terminal window. The title bar shows 'Terminal' with menu options 'Shell', 'Edit', 'View', 'Window', and 'Help'. The window title is 'php-project'. The terminal shows the following commands and output:

```
[matt@matts-MacBook-Pro php-project (master)] $ git add .
[matt@matts-MacBook-Pro php-project (master)] $ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

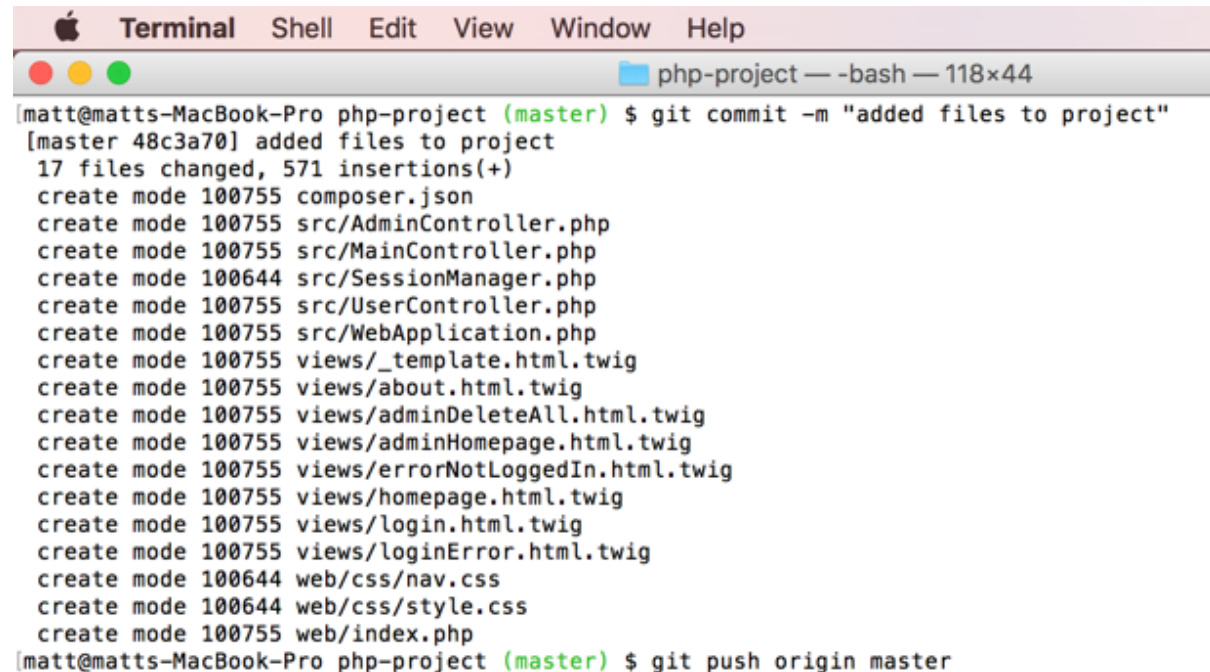
        new file:   composer.json
        new file:   src/AdminController.php
        new file:   src/MainController.php
        new file:   src/SessionManager.php
        new file:   src/UserController.php
        new file:   src/WebApplication.php
        new file:   views/_template.html.twig
        new file:   views/about.html.twig
        new file:   views/adminDeleteAll.html.twig
        new file:   views/adminHomepage.html.twig
        new file:   views/errorNotLoggedIn.html.twig
        new file:   views/homepage.html.twig
        new file:   views/login.html.twig
        new file:   views/loginError.html.twig
        new file:   web/css/nav.css
        new file:   web/css/style.css
        new file:   web/index.php
```

8. Commit with message

We now need to commit to these changes for a new version of our repository

Type:

git commit -m "added files to project"



```
Terminal Shell Edit View Window Help
php-project — -bash — 118x44
[matt@matts-MacBook-Pro php-project (master)] $ git commit -m "added files to project"
[master 48c3a70] added files to project
17 files changed, 571 insertions(+)
create mode 100755 composer.json
create mode 100755 src/AdminController.php
create mode 100755 src/MainController.php
create mode 100644 src/SessionManager.php
create mode 100755 src/UserController.php
create mode 100755 src/WebApplication.php
create mode 100755 views/_template.html.twig
create mode 100755 views/about.html.twig
create mode 100755 views/adminDeleteAll.html.twig
create mode 100755 views/adminHomepage.html.twig
create mode 100755 views/errorNotLoggedIn.html.twig
create mode 100755 views/homepage.html.twig
create mode 100755 views/login.html.twig
create mode 100755 views/loginError.html.twig
create mode 100644 web/css/nav.css
create mode 100644 web/css/style.css
create mode 100755 web/index.php
[matt@matts-MacBook-Pro php-project (master)] $ git push origin master
```

9. “push” up to the Github website repository

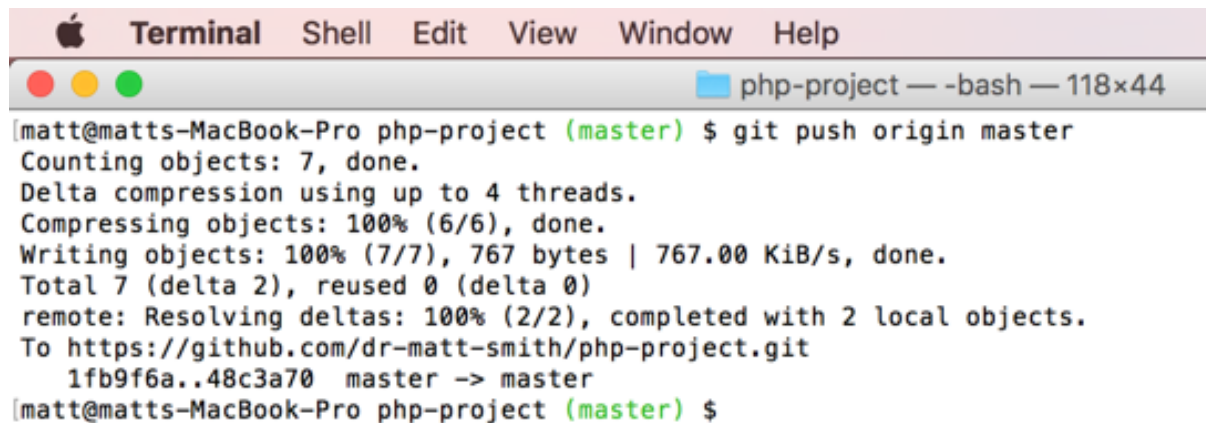
We now need to “push” the changed project up to the Github website with **git push origin master**

because we started this process by “cloning” down onto our computer from the Github repository, then Git knows where the online repository is (its “remote”). So when we give the “push” command it knows to push changes back to the same Github URL.

In the screenshot you can see it was pushed To:
<https://github.com/dr-matt-smith/php-project.git>

which you can break down as: **https://github.com/<git user>/<repo name>.git**

If you want to see the remote settings just type:
git remote -v

A screenshot of a macOS Terminal window titled "Terminal" with a menu bar (Shell, Edit, View, Window, Help). The window shows the execution of the 'git push origin master' command. The output indicates that 7 objects were counted, compressed, and written to the remote repository. The commit was pushed to the 'master' branch of the repository at 'https://github.com/dr-matt-smith/php-project.git'. The commit hash '1fb9f6a..48c3a70' is shown, along with the message 'master -> master'. The prompt returns to the user's shell.

```
[matt@matts-MacBook-Pro php-project (master)] $ git push origin master
Counting objects: 7, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 767 bytes | 767.00 KiB/s, done.
Total 7 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/dr-matt-smith/php-project.git
  1fb9f6a..48c3a70  master -> master
[matt@matts-MacBook-Pro php-project (master)] $
```

That’s it – you have now committed and uploaded changes to your project.

NOTE:

If you keep this folder on your laptop as your working project folder, each time you make a change you’d like to keep (and upload) just type:

```
git add .
git commit -m "<summary of change>"
git push origin master
```

And you’ll have added a new “version” of your project on Github

10. Upload complete –files and folders in repository on Github

The screenshot shows the GitHub interface for a repository named 'php-project' by user 'dr-matt-smith'. The repository is marked as 'Private'. The 'Code' tab is selected, showing a file list. A red callout bubble points to the 'README.md' file, stating 'files + README uploaded'.

dr-matt-smith / php-project Private

<> Code Issues 0 Pull requests 0 Projects

PHP project
Add topics

2 commits 1 branch

Branch: master New pull request

dr-matt-smith Add files via upload

src	Add files via upload
views	Add files via upload
web	Add files via upload
README.md	Initial commit
composer.json	Add files via upload

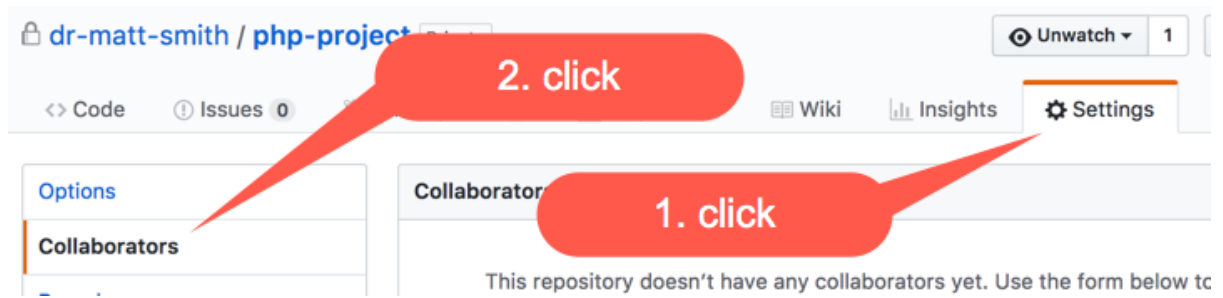
README.md

php-project

PHP project

files +
README
uploaded

11. Choose settings - collaborators

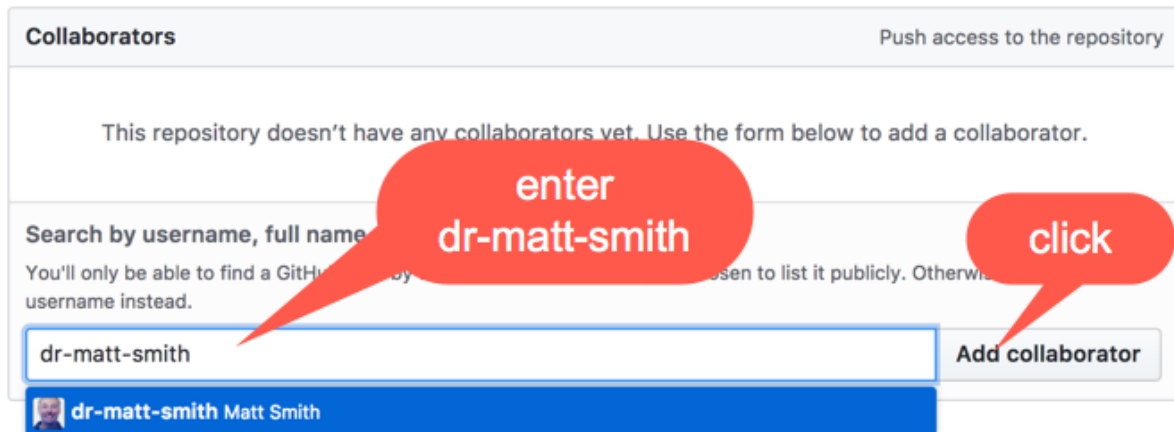


12. Enter dr-matt-smith as collaborator to your project

You **must** add dr-matt-smith as a collaborator to your project

Since your repository is PRIVATE Matt can only download your files if you do this step

If you don't, then you haven't succeeded in making files available to Matt for grading !



The screenshot shows the 'Collaborators' tab of a GitHub repository. At the top right is a link 'Push access to the repository'. Below the header, a message states: 'This repository doesn't have any collaborators yet. Use the form below to add a collaborator.' The main section is titled 'Search by username, full name'. Below this, a note explains: 'You'll only be able to find a GitHub user by username if they have chosen to list it publicly. Otherwise, you'll need to use their full name.' A search input field contains the text 'dr-matt-smith'. A red speech bubble with the text 'enter dr-matt-smith' points to this input field. Below the input field, a dropdown menu is open, showing a search result: a profile picture, the username 'dr-matt-smith', and the full name 'Matt Smith'. To the right of the search input is a button labeled 'Add collaborator'. A red speech bubble with the text 'click' points to this button.

13. Copy project URL to clipboard and submit to Moodle

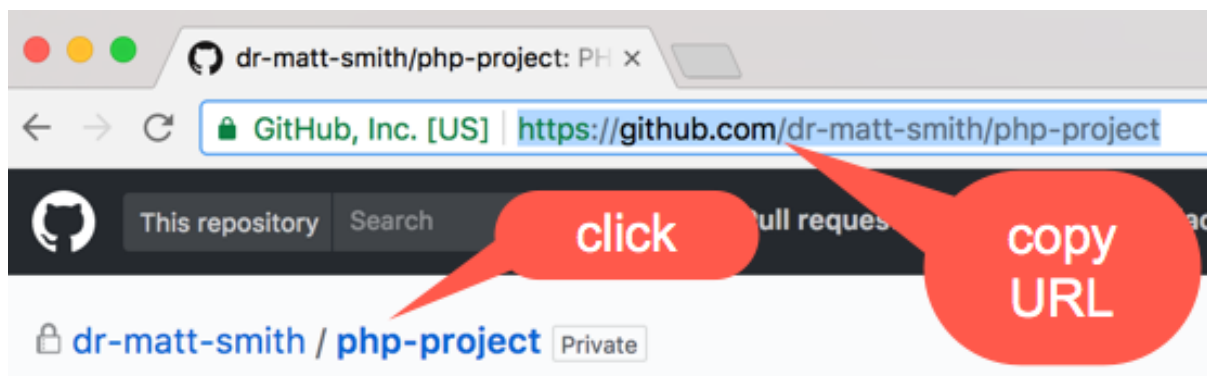
The final step is getting the project's URL – to submit to Moodle

Just click the blue project name (in my example “php-project”) to return to the project home page

Then copy the URL web address in the web browser address bar

Then go to Moodle and submit that URL

That's it – you have created a private Github project and made it available for Matt to download.



Think of Github as a free cloud storage for all your college work

- It works best:
 - With text-based projects (like computer programs!)
 - But can be used with Word documents, PDFs, Excel files
 - Even multimedia projects such semester 2 Unity games ...
 - When you get the hang of the command line approach, you'll find that even faster to use ...