# Git & GitHub Crash Course For Beginners

<https://www.youtube.com/watch?v=SWYqp7iY_Tc&t=992s>

21/06/2021

--------------

$ git init //Initialize Local Git Repository

$ git add <file> //Add File(s) to Index

$ git status //Check status of working tree

$ git commit //Commit changes in index

$ git push //Push to remote Repository

$ git pull //Pull latest from remote repository

$ git clone //Clone repository into a new directory

$ touch index.html //Create new file

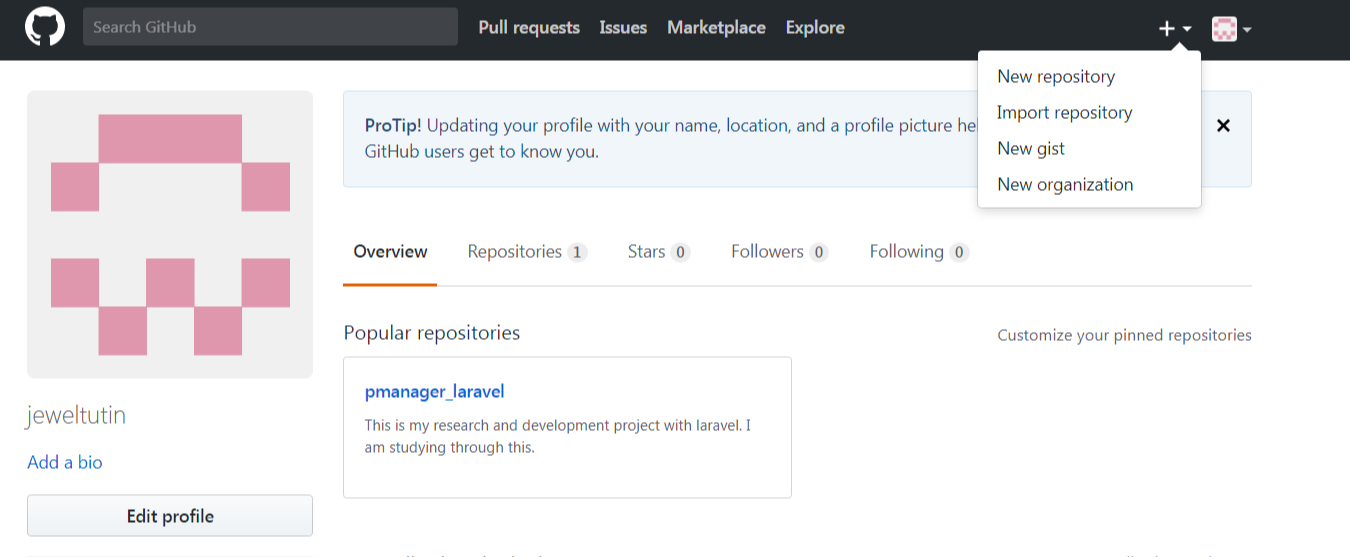
$git remote –v

// **Shows URLs of remote repositories when listing your current remote connections**. By default, listing remote repositories only shows you their shortnames (e.g. "origin"). Using the "-v" option, you will also see the remote's URLs in listings.

First we need to create an account in :

<https://github.com>

Go to your profile and from there u can create repositories and see all your repositories here:



Then download git from

<https://git-scm.com/download/win>

run gitbash

You can use gitbash anywhere in any folder by right clicking and select use gitbash here

$ git --version

You can increase or decrease gitbashes font size ctrl + mouse wheel

$ git init //for initialization

Then set your name and email

$ git config --global user.name 'jeweltutin'

$ git config --global user.email 'jeweltutin@gmail.com'

To show the recent status

$ git status

Add/track a particular file

$ git add index.php

Add all files which have same extension

$git add \*.php

Remove/Untrack a particular file

$git rm –cached index.php

Add/track all file

$git add \* // or $git add .

Remove/Untrack all file

$git rm –cached \*

Remove/Untrack all files which have same extention

$git rm --cached \*.php

$ rm -r f4returnvalue.php delete a file from repository

$ rm -r \* delete all file from repository

Then we need to add the files and then commit that and push that to server

Clear ------ for clearing the screen

Make a gitignore file

$Touch .gitignore

For ignoring a file

$Touch log.txt

To ignore a folder (Save like below to gitignore file)

/newFolder

Mytest.php

\*.txt

Making a branch

$Git branch projectnew

$git commit –m ‘my new changes’

$git checkout branchname

For merging branch Go to the master branch

$git merge projectnew

Type I for insert

Type a note

And wq ---for save quiet editor mode

$git clone address

git remote add origin https://github.com/jeweltutin/pmanager\_laravel.git

git push -u origin master

**…or create a new repository on the command line //New 2021**

echo "# blog-laravel-21" >> README.md

git init

git add README.md

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/jeweltutin/blog-laravel-21.git

git push -u origin main

**…or push an existing repository from the command line**

git remote add origin https://github.com/jeweltutin/blog-laravel-21.git

git branch -M main

git push -u origin main

**…or create a new repository on the command line**

echo "# pmanager\_laravel" >> README.md

git init

git add README.md

git commit -m "first commit"

git remote add origin https://github.com/jeweltutin/pmanager\_laravel.git

git push -u origin master

### …or push an existing repository from the command line

git remote add origin https://github.com/jeweltutin/pmanager\_laravel.git

git push -u origin master

### …or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

I'm looking for help on dealing with the whole setup process of a Laravel project. Currently this is my walkthrough list:

* Install Virtual Box and Vagrant.
* Run vagrant box add laravel/homestead
* Run git clone https://github.com/laravel/homestead.git Homestead on your favorite folder, using your favorite Bash (in my case I'm using Git BASH because I'm on Windows 10)
* Setup your SSH Keys with ssh-keygen -t rsa -C "email@domain.blah"
* Setup SSH connector file[¹].
* Setup your Homestead.yaml as you wish.
* Run init.sh / init.bat.
* SSH into the Vagrant.
* run composer global require "laravel/installer"
* run laravel new project
* Go back to the host machine, into the project folder and run git init, git add ., git commit -m "clean project"
* Push the project with git remote add origin https://bitbucket.org/you/yourproject and git push -u origin --all

Now I have a brand new Laravel project hook up to Git for versioning. My problem is that Laravel ignores /vendor by default. Considering this fact, I want to clone my project in another computer because I have 2 computers to work and/or a co-worker wants to clone the same project so we can work on it together.

What would be the proper walkthrough to clone the project and have Laravel work on another machine? Do I have to add /vendor to the repository and push? Should I add the homestead box into the repository? If so, how?

Thanks in advance.

Answer

Ensure that your Git repository includes composer.json (package settings) and composer.lock (optional, but recommended by Composer to ensure 100% version match across all servers) files, but doesn't include 'vendor' folder or .env file (at the moment it looks like you are adding everything with 'git add .' – which is not safe). A standard practice to deploy ('clone') your code would be:

1. git clone <https://bitbucket.org/you/yourproject> (on remote machine)
2. cd yourproject
3. composer install (this will create 'vendor' folder and download all packages)
4. Create and edit .env file

Pretty simple really!

Moreover, you can find a number of services in Internet that can deploy your application automatically whenever BitBucket repository has updates.

##Windows users:

- Download wamp: http://www.wampserver.com/en/

- Download and extract cmder mini: https://github.com/cmderdev/cmder/releases/download/v1.1.4.1/cmder\_mini.zip

- Update windows environment variable path to point to your php install folder (inside wamp installation dir) (here is how you can do this http://stackoverflow.com/questions/17727436/how-to-properly-set-php-environment-variable-to-run-commands-in-git-bash)

cmder will be refered as console

##Mac Os, Ubuntu and windows users continue here:

- Create a database locally named `homestead` utf8\_general\_ci

- Download composer https://getcomposer.org/download/

- Pull Laravel/php project from git provider.

- Rename `.env.example` file to `.env`inside your project root and fill the database information.

(windows wont let you do it, so you have to open your console cd your project root directory and run `mv .env.example .env` )

- Open the console and cd your project root directory

- Run `composer install` or ```php composer.phar install```

- Run `php artisan key:generate`

- Run `php artisan migrate`

- Run `php artisan db:seed` to run seeders, if any.

- Run `php artisan serve`

#####You can now access your project at localhost:8000 :)

## If for some reason your project stop working do these:

- `composer install`

- `php artisan migrate`

* Clone your project
* Go to the folder application using cd
* Run composer install on your cmd or terminal
* Copy .env.example file to .env on root folder. You can type copy .env.example .env if using command prompt Windows or cp .env.example .env if using terminal Ubuntu
* Open your .env file and change the database name (DB\_DATABASE) to whatever you have, username (DB\_USERNAME) and password (DB\_PASSWORD) field correspond to your configuration.   
  By default, username is root and you can leave password field empty. **(This is for Xampp)**   
  By default, username is root and password is also root. **(This is for Lamp)**
* Run php artisan key:generate
* Run php artisan serve
* Go to [localhost:8000](http://localhost:8000/)

# Run Laravel project after downloading from github/gitla

<https://www.youtube.com/watch?v=D5MZaCmpxvM>

Step:

1. download or clone project
2. Go to the folder application using cd
3. Run composer install on your cmd or terminal
4. Copy .env.example file to .env on root folder. You can type copy .env.example .env if using command prompt Windows or cp .env.example .env if using terminal Ubuntu
5. Open your .env file and change the database name (DB\_DATABASE)
6. Run php artisan key:generate
7. Run php artisan migrate
8. Run php artisan serve

# [How can I resolve “Your requirements could not be resolved to an installable set of packages” error?](https://stackoverflow.com/questions/29318709/how-can-i-resolve-your-requirements-could-not-be-resolved-to-an-installable-set)

Run this command:

composer install --ignore-platform-reqs

or

composer update --ignore-platform-reqs

## How to show your Git username

There are at least three ways to show your Git username:

1. The git config command
2. The git config --list command
3. Looking in your Git configuration file

### *1) The `git config` command*

Here’s the git config command:

git config user.name

which in my case returns:

Alvin Alexander

### *2) The `git config --list` command*

Another way to show your Git username is with this git config command:

git config --list

which returns this output:

user.name=Alvin Alexander

user.email=[omitted]

merge.tool=vimdiff

## How to change your Git username

You can change your Git username like this:

git config --global user.name "Alvin J. Alexander"

Another way to change it is to edit the Git config file in your HOME directory and change it there:

vi ~/.gitconfig

I just did that on my test system, and it seems to work fine.

Again, it’s important to note that this is your “global” username. You can also have a different username on a per-project basis ... I just looked it up, and you should be able to change your Git username on a per-project basis like this, without the --global option:

git config user.name "Alvin J. Alexander"

## How to change your Git email address

While I’m in the Git username neighborhood, I’ll also add that you can view your Git email address with this command:

git config user.email

And you can change your Git email address like this:

git config --global user.email [your email address here]

Finally, you can also see your password by viewing the Git config file in your HOME directory:

more ~/.gitconfig

Source:: <https://alvinalexander.com/git/git-show-change-username-email-address/>

Check the difference between files

$ git diff HEAD app/Http/Controllers/FrontendController.php

$ git diff HEAD check the full changes

\*\* After adding a file in git with git command git add . if you need to remove a specific file

$git reset resource/view/website/tag.blade.php

\*\*Clear Laravel cache via artisan command

$php artisan cache:clear

$php artisan view:cache

$php artisan view:clear

For more info: <https://beyondco.de/blog/laravel-caches-and-all-ways-to-clear-them>

# [Git asks for username every time I push](https://stackoverflow.com/questions/11403407/git-asks-for-username-every-time-i-push)

You can store your credentials using the following command

$ git config credential.helper store

$ git config credential.helper store

$ git push <http://example.com/repo.git> //$git push –u origin main 🡪 If you were in a repository

Username: <type your username>

Password: <type your password>

//Remove credential

git config --unset credential.helper

$ git config --unset credential.helper