

# Git & GitHub

---

Git is a free and open source distributed version control system.

## Instalation git

---

For Instalation [Click here to download](#)

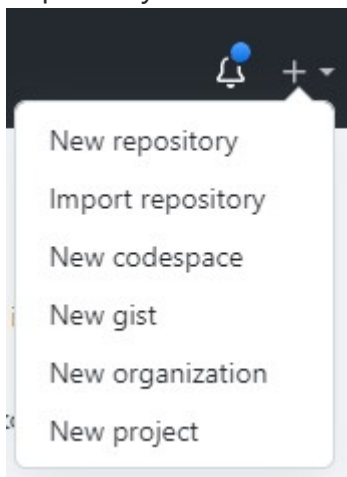
## Create Profile on GitHub

---

Link to [GitHub](#)

### 1. Create repository in GitHub

From top main navigation click on plus icon and from dropdown menu chooese New Repository



### 2. Type repository name sf-homework

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \*

 darevski1 ▾

Repository name \*

/ sf-homework ✓

Great repository names are available. Need inspiration? How about **probable-octo-robot**?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: None ▾

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: None ▾

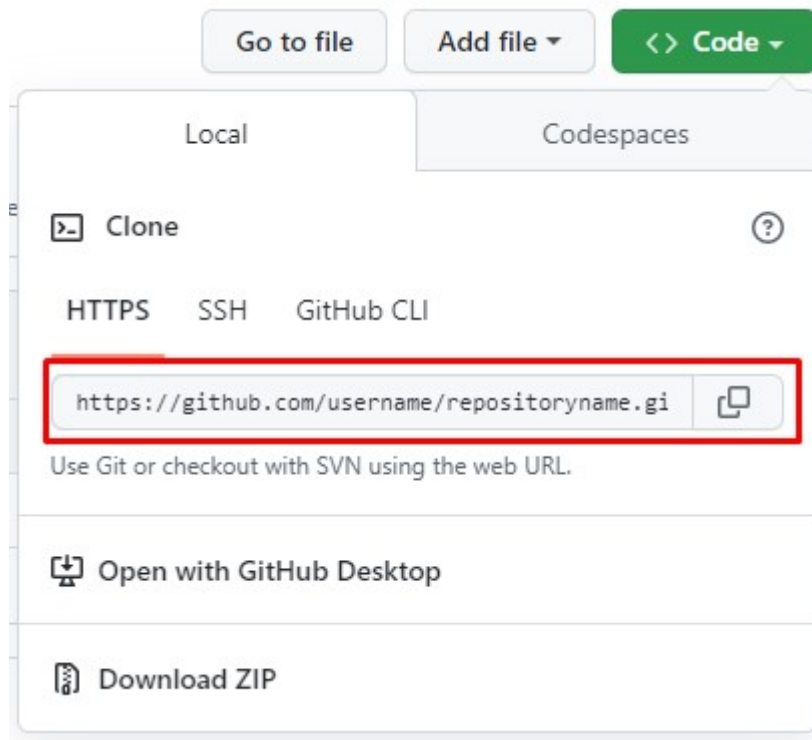
 You are creating a public repository in your personal account.

Create repository

3. Choose a repository visibility - Private or Public
4. Add README.md file \* optional or leave it as it is
5. Add .gitignore \* optional or leave it as it is
6. Choose a license \* optional or leave it as it is
7. Click CREATE REPOSITORY

# Cloning

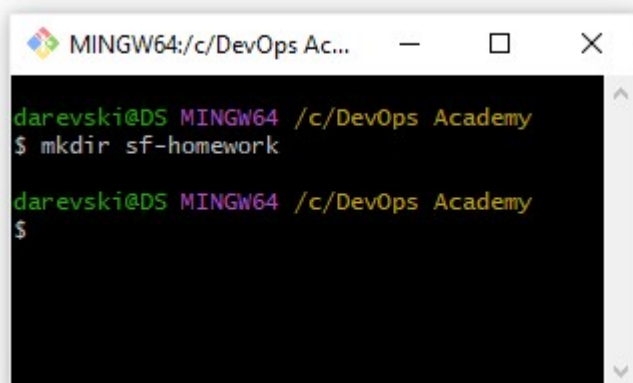
You can clone your repository to create a local copy on your computer and sync between the two locations. Open your repository.



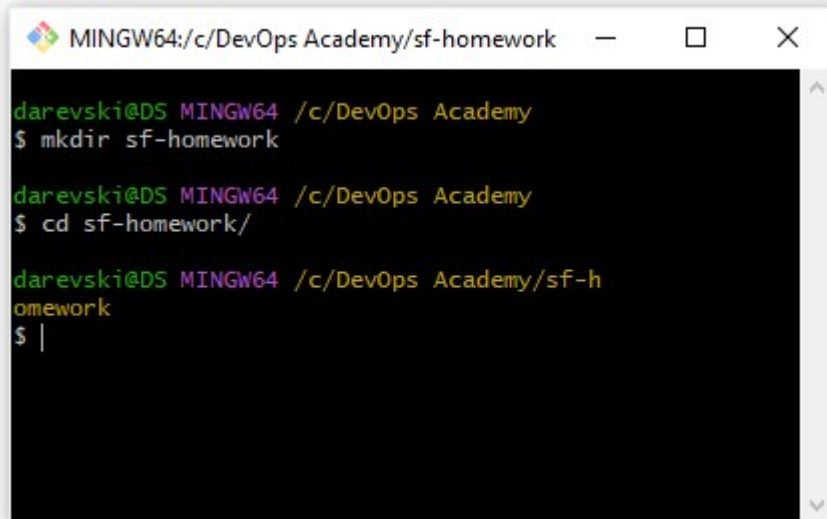
## Create a new repo from scratch and connect to github.

Open your terminal in desire location and create folder

```
mkdir sf-homework
```



```
cd sf-homework
```

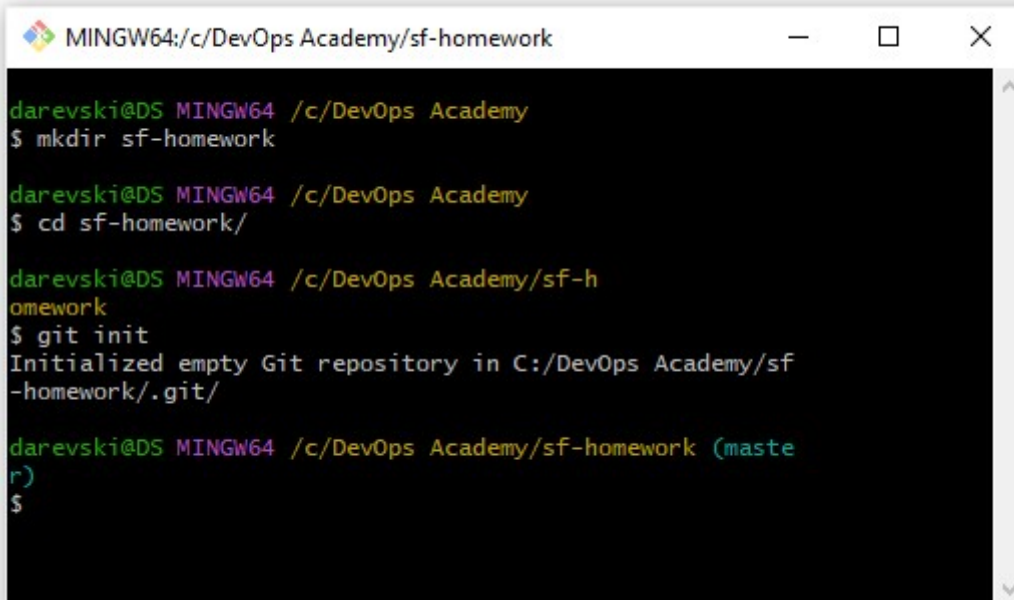


```
MINGW64:/c/DevOps Academy/sf-homework
darevski@DS MINGW64 /c/DevOps Academy
$ mkdir sf-homework

darevski@DS MINGW64 /c/DevOps Academy
$ cd sf-homework/

darevski@DS MINGW64 /c/DevOps Academy/sf-h
omework
$ |
```

```
git init
```



```
MINGW64:/c/DevOps Academy/sf-homework
darevski@DS MINGW64 /c/DevOps Academy
$ mkdir sf-homework

darevski@DS MINGW64 /c/DevOps Academy
$ cd sf-homework/

darevski@DS MINGW64 /c/DevOps Academy/sf-h
omework
$ git init
Initialized empty Git repository in C:/DevOps Academy/sf
-homework/.git/

darevski@DS MINGW64 /c/DevOps Academy/sf-homework (maste
r)
$
```

```
create README.md file
git add . too add all files in working directory or use git add file1 file2
git commit -m "inital push" // write commit message
git branch -M main
git remote add origin https://github.com/username/sf-homework.git
git push -u origin main
```

## Basic git commands

```
$ git config --global user.name youname
```

Setup an username

---

```
$ git config --global user.email example@example.com
```

 Setup an email address that will be associated with each history marker

---

```
$ git status
```

 Gives information on the current content status of a git repository and its contents

---

```
$ git init
```

 Create new repository, Before we can do anything git-related, we must initialize a repo first.

Example - create new folder **myfirstrepo** open the folder via cmd and type **git init**

```
mkdir myfirstrepo // create folder
cd myfirstrepo // access the folder
git init // initialize a repo
```

Output: Initialized empty Git repository in **C:/path/to/the/repository/myfirstrepo/.git/**

---

```
$ git log
```

 Shows a default output for quickly reviewing the commit history

---

Example

Author: Darko [example@example.com](mailto:example@example.com) Date: Sun Feb 19 01:19:41 2023 +0100

```
readme file added in git
```

## Git add & commit

```
$ git add --all
```

 // will add all files changes in staging Example:

```
$ git add --all
```

```
$ git add file1 file2 file3
```

 // will add only file we specify to staging phase

```
$ git add file.txt file2.txt file3.txt
```

\$ **git commit -m "you commit message"** // after we add files in staging phase we have to add commit message **Example**

```
: $ git commit -m "login page functionality created"
```