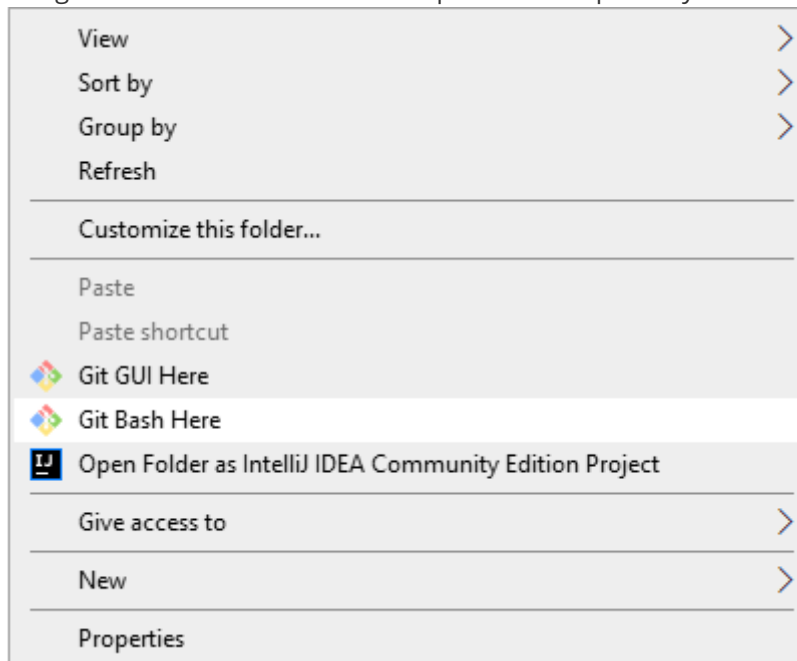
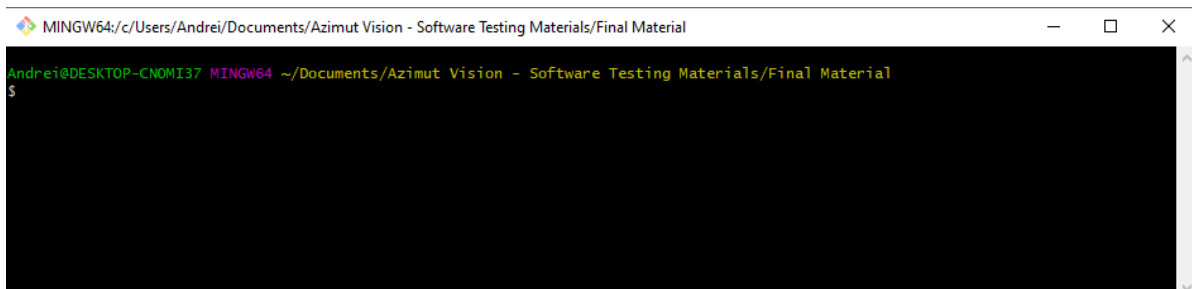


# Git & Github

Pentru cursul nostru vom folosi Git Bash pentru a face cea mai mare parte a muncii noastre. Mergem in folderul in care vrem sa pornim un repository local de Git si vom face click dreapta

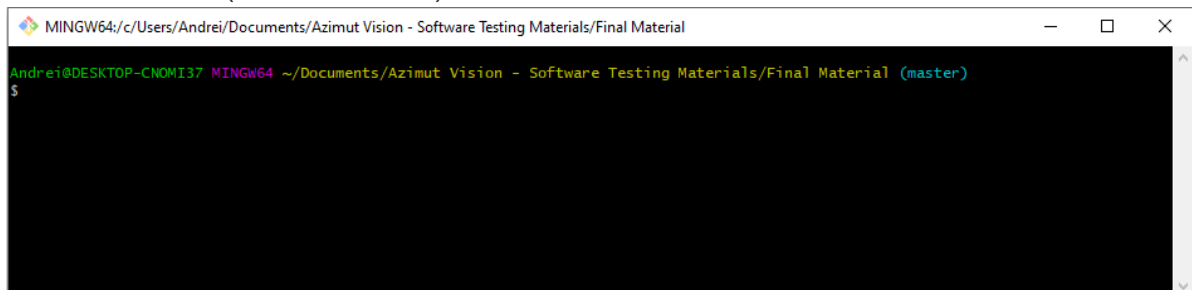


De aici vom selecta "Git Bash Here" si ni se deschide urmatoarea fereastra



⚠ Sa aveti grija ca sunteti in folderul dorit! Cititi cu atentie calea (Scrizul cu galben)

Daca executati comanda de "Git Bash Here" intr-un proiect deja initializat o sa va apara la final branchul de lucru (de obicei master)



## Proiect nou

In cazul in care doar ce ati facut un proiect, o sa fie nevoie sa il initializati. Pentru a executa toti pasi pana la urcarea lui pe Github urmati instructiunile de aici:

`git init` - inițializează un nou depozit Git și începe să urmărească un director existent. Adaugă un subfolder ascuns în directorul existent care găzduiește structura internă de date necesară pentru controlul versiunilor.

```
MINGW64/c:/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material

Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material
$ git init
Initialized empty Git repository in C:/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material/.git/

Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$
```

`git status` - arată starea modificărilor ca neurmărite, modificate sau în etape.

```
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  Curs 1 - Programare Java Intro.pdf
  Curs 2 - Tipuri de date neprimitive.pdf

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

`git add .` - Git urmărește modificările aduse bazei de cod, dar este necesar să puneți în staging pentru a le include în istoricul proiectului. Orice modificări care sunt puse în staging vor deveni parte din următorul commit și o parte din istoria proiectului.

```
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git add .

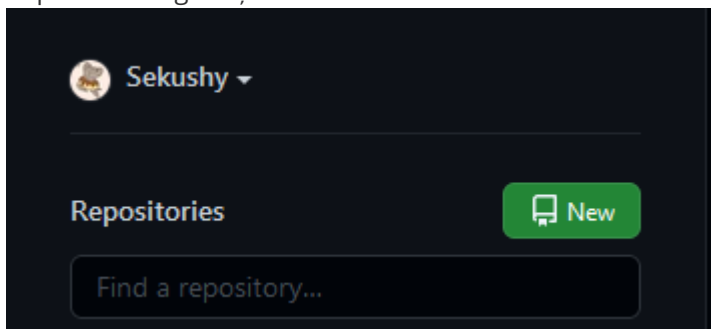
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ |
```

`git commit -m <MESAJUL_VOSTRU>` - salvează commit-ul în istoricul proiectului și finalizează procesul de urmărire a modificărilor. Pe scurt, un commit funcționează ca a face o fotografie. Orice lucru care a fost pus în staging cu `git add` va deveni parte a commitului cu `git commit`.

```
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git commit -m "Initial commit"
[master (root-commit) 6a7b419] Initial commit
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Curs 1 - Programare Java Intro.pdf
create mode 100644 Curs 2 - Tipuri de date neprimitive.pdf
```

Acum vom merge la Github și vom crea un nou repository pentru proiectul nostru.

În partea stângă veți vedea un buton verde cu textul "New"




Odată ce facem asta, va trebui să adăugăm un nume în depozitul nostru. Atentie! Nu ar trebui să aveți spații în numele repository-ului

## Create a new repository

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


---


**Owner \***  Sekushy ▾ /

Great repository names are short and memorable. Need inspiration? How about **super-duper-broccoli**?

**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.](#)


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

---


[Create repository](#)


Vom vedea acum această pagină, vom copia adresa URL din partea de sus a paginii și să revenim la terminalul Git Bash


**Quick setup — if you've done this kind of thing before**

 Set up in Desktop

 or 

 HTTPS

 SSH



Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

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



```
echo "# Github_Example" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Sekushy/Github_Example.git
git push -u origin main
```

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



```
git remote add origin https://github.com/Sekushy/Github_Example.git
git branch -M main
git push -u origin main
```

**...or import code from another repository**

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

[Import code](#)

 **ProTip!** Use the URL for this page when adding GitHub as a remote.

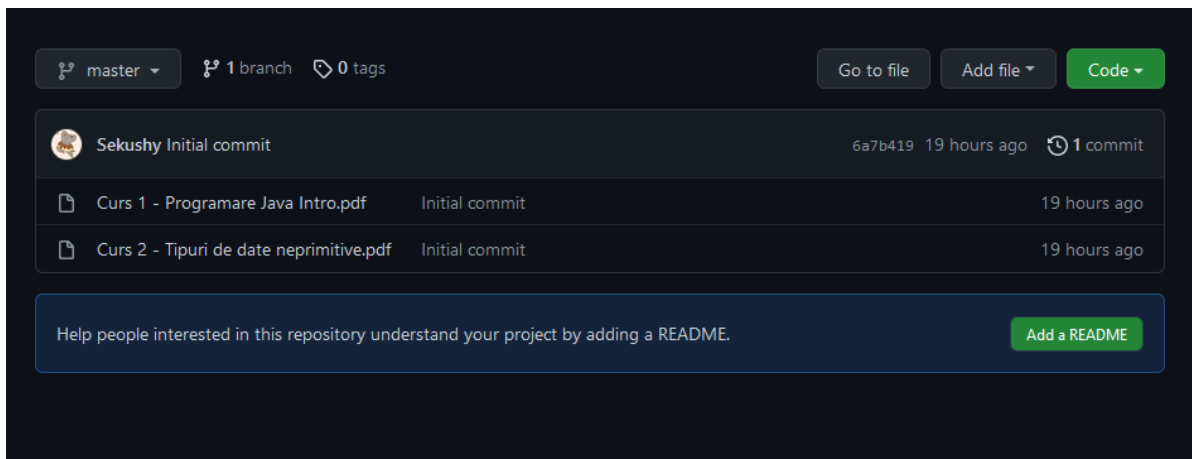
Acum vom seta un punct de origine la care putem încărca modificările noastre git pe Github, acest lucru se face prin comanda `git remote add origin <LINK>`

```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git remote add origin https://github.com/Sekushy/Github_Example
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$
```

Acum am adăugat un punct de origine din care putem descărca sau încărcă datele noastre. Următorul pas este să ne urcam schimbările prin comanda `git push origin master`

```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git remote add origin https://github.com/Sekushy/Github_Example
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 313.87 KiB | 31.39 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Sekushy/Github_Example
 * [new branch]      master -> master
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$
```

Dacă nu ați întâmpinat nicio problemă, ar trebui să vedeți acum modificările în Github după un refresh.



## Proiect existent (Adaugarea de schimbări)

În cazul în care faceți modificări în repository-ul vostru local și doriți să încărcăți acele modificări în repository-ul vostru de Github, vom avea următorul workflow.

Vom verifica modificările neurmărite cu `git status`

```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    New_Document.txt

nothing added to commit but untracked files present (use "git add" to track)
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$
```

Acum vom adăuga toate elementele noi în staging folosind `git add .`

```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git add .
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ |
```

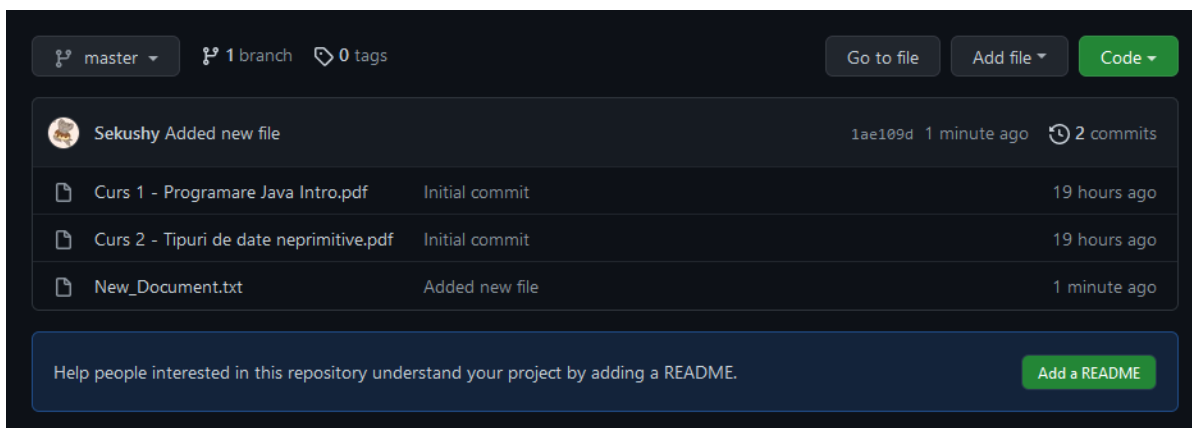
Acum putem verifica starea dacă vrem altfel, vom executa comanda `git commit -m "<MESAJUL_VOSTRU>"`

```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git commit -m "Added new file"
```

Tot ce mai rămâne acum de făcut este să executați comanda `git push origin master`

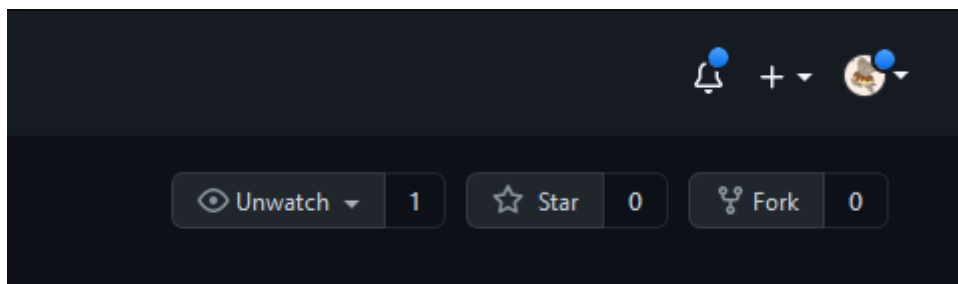
```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 288 bytes | 288.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Sekushy/Github_Example
6a7b419..1ae109d master -> master
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material (master)
$ |
```

Apoi puteți verifica modificările în Github



## Clonarea unui proiect nou de pe Github

Pentru a clona un repository creat de altcineva pe care ați dori să-i faceți modificări, mai întâi va trebui să îi faceți un fork. Din pagina Github, accesați repository-ul care vă interesează și faceți click pe butonul de 'Fork' pe care îl găsiți în colțul din dreapta, sus.



După ce ați făcut acest lucru, puteți utiliza următoarea comandă într-un folder din PC-ul vostru:

```
git clone <LINKUL_VOSTRU>
```

```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material
$ git clone https://github.com/Sekushy/Github_Example
Cloning into 'Github_Example'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 7 (delta 1), reused 7 (delta 1), pack-reused 0
Receiving objects: 100% (7/7), 314.11 KiB | 494.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material
$ |
```

Faceți un `cd` în folderul descărcat

```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material
$ cd Github_Example/
```

Creați un nou branch executând următoarea comandă `git checkout -b <NUMELE_BRANCHULUI>`

```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material/Github_Example
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material/Github_Example (master)
$ git checkout -b NewBranch
Switched to a new branch 'NewBranch'
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material/Github_Example (NewBranch)
$ |
```

De aici, pașii sunt aceiași cu cei din categoria **Proiect existent (Adaugarea de schimbări)**

## Adaugarea de schimbări în proiectul vostru local

Pentru a vă actualiza depozitul, va trebui doar să executați următoarea comandă `git pull origin <NUMELE_BRANCHULUI>`

```
MINGW64/c/Users/Andrei/Documents/Azimut Vision - Software Testing Materials/Final Material/Github_Example
Andrei@DESKTOP-CNOMI37 MINGW64 ~/Documents/Azimut Vision - Software Testing Materials/Final Material/Github_Example (NewBranch)
$ git pull origin master
From https://github.com/Sekushy/Github_Example
 * branch      master       -> FETCH_HEAD
Already up to date.
```