Ministerul Educației și Cercetării al Republicii Moldova
IP Colegiul “Iulia Hasdeu” din Cahul

Asistență pentru managementul proiectelor software

Tema 3: Inițializarea unui depozit

adăugarea și comiterea fișierelor

Apareci Aurica
Babaianu Savelie
Grupa: AAW 2042

Inițializarea unui depozit

> **Depozit Git (Repository):** Un depozit Git este un spațiu în care Git stochează toate informațiile despre un proiect. Acesta include toate fișierele, directoarele și istoricul schimbărilor legate de proiect. De obicei, există două tipuri principale de depozite: local și remote.

Inițializarea unui depozit Git este primul pas în gestionarea versiunilor pentru proiectele tale. Prin inițializarea unui depozit git, se crează un mediu de urmărire a modificărilor proiectului. Pentru a crea un nous depozit Git, utilizezi comanda **`git init`.** Aceasta comandă creează un director ascuns numit ".git" în directorul curent, care conţine toate informaţiile necesare pentru gestionarea versiunilor.

1 Inițializarea depozitului

```
mkdir new_project
cd new_project
git init
```

Duplicarea unui depozit Git

> Comanda **`git clone`** este folosită pentru a crea o copie locală a unui depozit Git existent. Comanda este esenţială atunci când doreşti să începi să lucrezi la un proiect existent sau să colaborezi cu alţi dezvoltatori, deoarece îţi oferă o copie locală a întregului depozit şi a istoricului său, permiţându-ţi să lucrezi la proiect în mod eficient şi să gestionezi versiunile folosind Git.

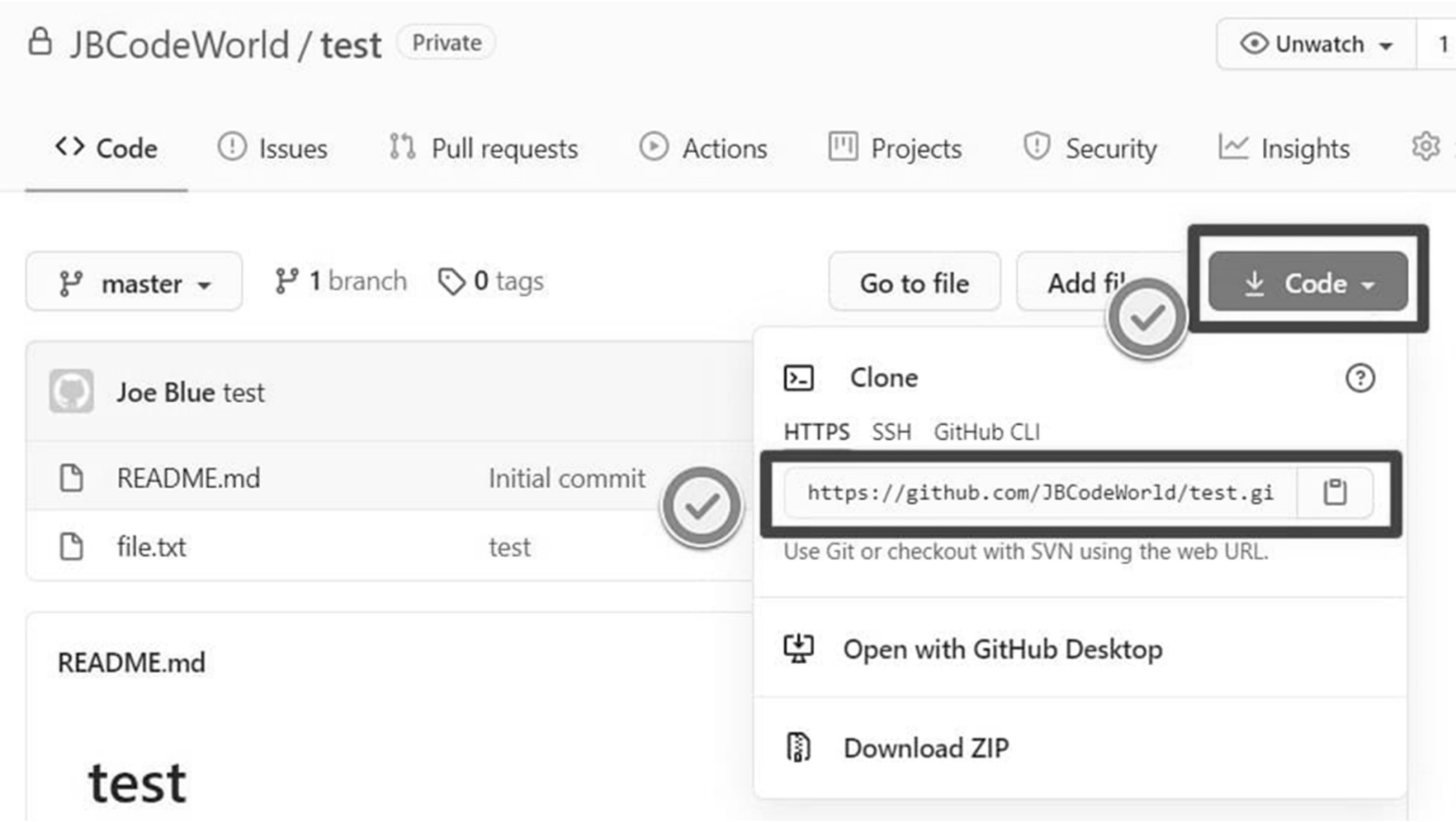
Această comandă face următoarele:

1. **Descărcare**
   Clonează un depozit Git de pe un server remote (cum ar fi GitHub) sau din alt repositoriu local şi aduce toate datele şi istoricul asociat cu acel depozit pe computerul tău.

2. **Crearea unei copii de lucru**
   Creează o copie a conţinutului depozitului într-un director local, astfel încât poţi lucra cu fişierele şi conţinutul din depozit pe computerul tău.

3. **Setarea unui remote**
   Adaugă o referinţă la depozitul remote din care ai clonat, astfel încât să poţi să trimiti şi să primeşti modificări între depozitul local şi cel remote folosind comenzile git push şi git pull.

4. **Iniţializează ramura principală**
   Creează o ramură locală numită de obicei "master" sau "main" care este o copie a ramurii implicite din depozitul remote.


JBCodeWorld / test
Private
Unwatch
1
Code
Issues
Pull requests
Actions
Projects
Security
Insights
master
1 branch
0 tags
Go to file
Add fi
Code
Joe Blue test
README.md
Initial commit
file.txt
test
Clone
HTTPS
SSH
GitHub CLI
https://github.com/JBCodeWorld/test.gi
Use Git or checkout with SVN using the web URL.
Open with GitHub Desktop
Download ZIP
README.md
test

2 Duplicarea depozitului

git clone <URL_dep>

v

<URL_dep> reprezintă adresa URL a depozitului Git pe care dorești să-l clonezi. Acesta poate fi un URL HTTP/HTTPS sau un URL SSH, și poate fi de pe GitHub, GitLab, sau oricare alt server Git.

Adăugarea fișierelor

Comanda **`git add`** este utilizată pentru a adăuga sau pregăti fișierele sau modificările pentru a fi incluse în următorul commit. Aceasta transferă modificările din directorul de lucru (working directory) în staging area (zona de pregătire), ceea ce înseamnă că aceste modificări sunt gata să fie comise în istoricul depozitului Git. Poate fi folosită pentru adăugarea fișierelor noi sau a modificărilor la fișierele existente.

Comiterea fișierelor

> Comanda **`git commit`** este folosită pentru a crea un nou commit care înregistrează starea curentă a staging area în istoricul depozitului Git. La fiecare commit, se adaugă un mesaj de commit care descrie modificările efectuate în cadrul acelui commit. Commit-urile sunt puncte de referință în istoricul depozitului și sunt utilizate pentru a urmări și a reveni la stări anterioare ale proiectului.

3 **Adăugarea fișierelor**

git add .

git add new_folder

4 **Comiterea fișierelor**

git commit –m “Added new folder”

Comenzile **git log, git reset** şi **git revert** sunt instrumente esenţiale în gestionarea istoricului şi a modificărilor într-un depozit Git.

1. **git log**
   este folosit pentru a afişa istoricul commit-urilor dintr-un depozit GIT, în ordine cronologică inversă, cele mai recente fiind afişate primele. Aceasta arată informaţii precum autorul, data şi mesajul fiecărui commit.

2. **git revert**
   este folosită pentru a crea un nou commit care anulează efectele unui commit anterior. În loc să şteargă istoricul, git revert adaugă un nou commit care face "reversul" modificărilor introduse de commit-ul specificat.

3. **git reset**
   este folosit pentru a modifica starea staging şi istoricul commit-urilor.

bash

```
git reset --soft name_commit
```

bash

```
git reset name_commit
```

bash

```
git reset --hard name_commit
```

bash

```
Aura@LAPTOP-AQMH74EU MINGW64 ~/Desktop/Learning-ASP.NET-MVC (main)
$ git log
commit 0497c4daa05885cf7946726125112782517 2d56f (HEAD -> main)
Author: 
Date:   Sat Oct 28 01:47:33 2023 +0300

    Updated for collection of materials for the guide

commit 363ae153650b503781de83ad0629e2a2c4c20ccd
Author: 
Date:   Sat Oct 28 01:36:28 2023 +0300

    Added unnecessary files to project

    Changes made during the collection of materials for the guide
```

bash

```
git revert <SHA_commit>
```

`SHA_commit` reprezintă codul unic al commit-ului pe care dorești să-l revoci. Acesta va crea un nou commit care elimină modificările introduse de commit-ul specificat, menținând totuși istoricul neschimbat.