GitLab - gestiunea temelor/proiectelor

Pentru colaborarea în cadrul unei echipe care dezvoltă proiecte este necesară folosirea unui sistem de versionare a fișierelor. Acesta permite colaborarea la distanță și rezolvarea problemelor generate de partajarea acelorași fișiere. Pentru proiectele dezvoltate individual, versionarea este indicată pentru că:
- ajută la identificarea mai ușoară modificării care a generat un bug
- permite salvarea surselor în diferite stagii ale dezvoltării proiectului
- permite revenirea la o versiune specificată a surselor.

GitLab este o aplicație software pentru gestionarea ciclului de viață de dezvoltare software. Principalele sale caracteristici includ managementul de proiect, managementul codului sursă, integrare continuă/livrare continuă (CI/CD), monitorizare și securitate.

Terminologie

git add - Adăugă conținutul fișierului/fișierelor nou/noi la indexul repository-ului local

git branch - Afișează, crează sau șterge un branch

git clone - Copiază un repository intr-un folder local

git fetch - Descarcă fișiere din alt repository

git init - Crează un repository gol

git commit - Înregistrează modificările din cadrul repository-ului

git log - Afișează log-urile

git merge - Integrează două sau mai multe branch-uri de dezvoltare

git pull - Descarcă și integrează fișiere din alt repository sau branch local

git push - Actualizează fișierele din repository-ul remote

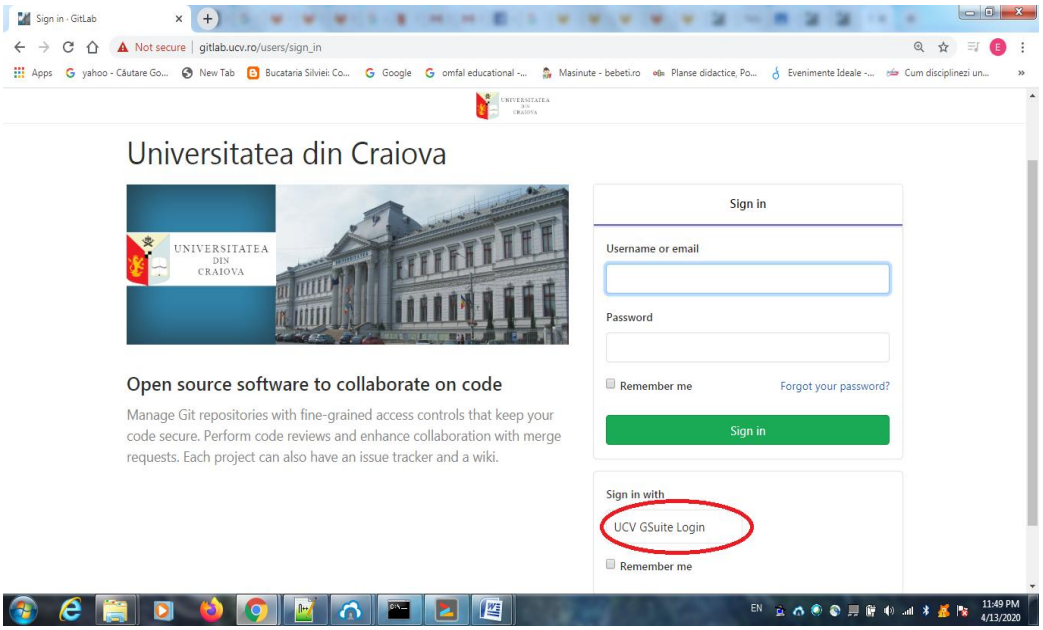
git tag - Crează, afișează sau șterge un tag

Git Setup

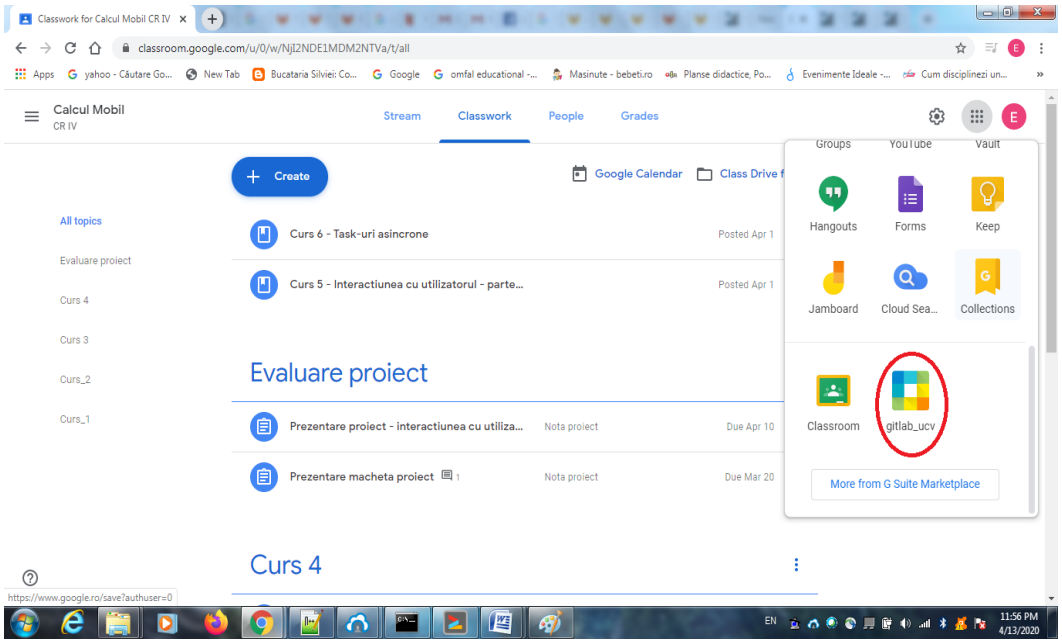
Git poate fi instalat ca o componentă opțională a Visual Studio și este unul dintre instrumentele Xcode. De asemenea se poate instala manual:
- Windows: download
- macOS: download
- GNU/Linux: instructions

Autentificare GitLab:

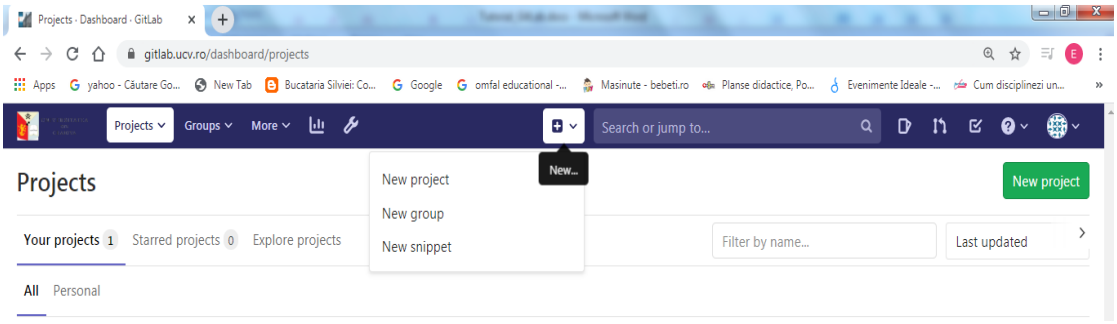
- Access la instanța GitLab: https://gitlab.ucv.ro
- Autentificarea se va face folosind contul de student: <id_student>@student.ucv.ro



- Autentificarea este disponibila și din Google Apps:



- După autentificare este afisat dashboard-ul user-ului autentificat



Setare profil și cheie SSH

1. Access la Profile settings
2. Se pot vizualiza diferite categorii pentru setarea profilului:
   - informații despre nume și e-mail se găsesc în tab-ul Profile
   - informații despre cont
   - setarea modului în care va fi vizualizat codul submis (tab-ul Design)

În cadrul paginii Profile settings foarte important este adăugarea unei chei SSH. Generarea chei SSH se poate face folosind programul ssh-keygen, iar acesta poate fi rulate din Git Bash:
$ ssh-keygen

Când se execută comanda, se va lăsa numele fișierului și parola necompletate, in felul acesta se vor utiliza valorile implicite.

În tab-ul Profile settings, pagina SSH keys de pe GitLab se poate adăuga această cheie: click pe Add key => se specifică un nume pentru această cheie => se copiază conținutul cheii publice generate în căsuța Key.

Înainte de a folosi Git-ul, se recomandă setarea proprietăților specifice identității utilizatorului:
git config --global user.name "<Nume student>"
git config --global user.email <id_student>@student.ucv.ro

Creare proiect nou

După autentificare pe GitLab, din Dashboard în partea stângă apare următorul meniu:
- Projects - este pagina pentru gestionarea proiectelor; se pot vizualiza toate proiectele în care utilizatorul este implicat: ca owner, ca viewer sau ca developer
- Activity - arată activitatea curentă: de exemplu crearea unui proiect nou
- Groups - gestionarea grupurilor și vizualizarea grupurilor existente
- Milestones - gestionarea pașilor importanți corespunzători proiectelor
- Issues - probleme de rezolvat într-unul dintre proiecte

- Merge Requests - oferă posibilitatea combinării branch-ului default (master-ului) cu un altul (în cazul rezolvării unui bug sau adăugării unui feature)

- Snippets - fragmente de text care pot fi făcute private sau publice

- Help - documentația GitLab.

Pentru crearea unui nou proiect:

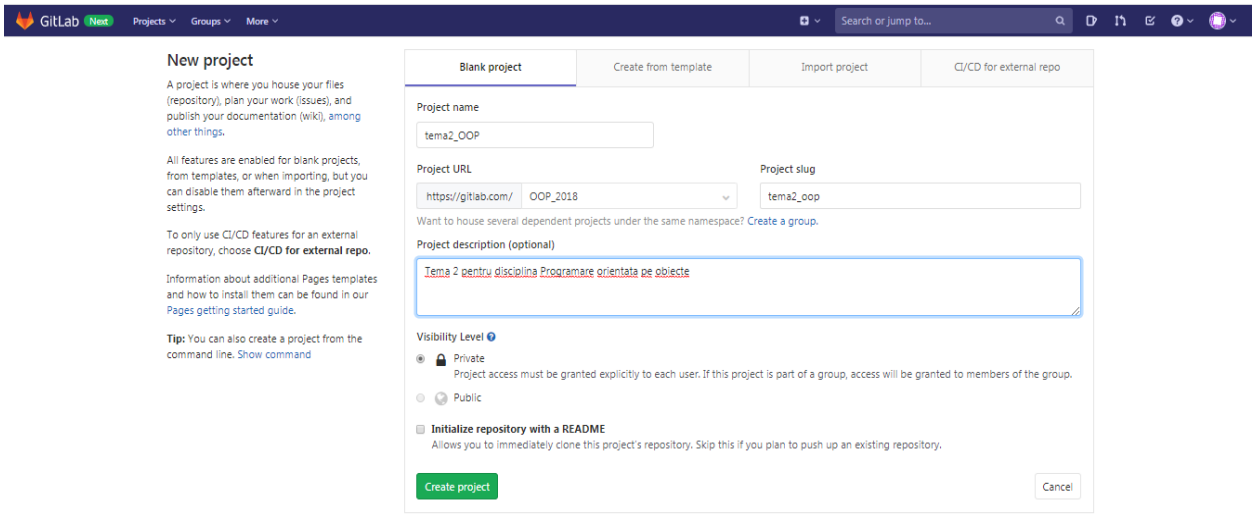
- Click pe New project

- Se specifică numele proiectului: de exemplu *tema2_OOP*.

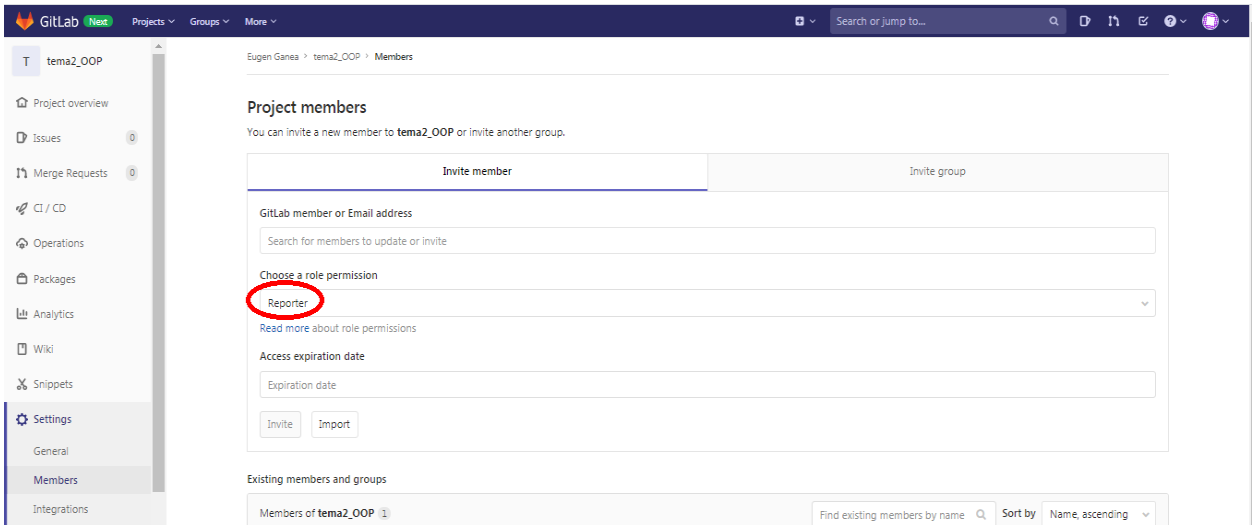
- Se specifică o descriere corespunzătoare pentru proiect

- Nivelul de vizibilitate trebuie să fie setat ca fiind privat

- Click Create project



După ce proiectul a fost creat, se va selecta tab-ul Members din cadrul acestui proiect și va/vor fi adăugat/adăugați profesorul/profesorii titular/i pentru disciplina pentru corespunzătoare proiectului, cu drept de access Reporter.



Adăugare de fișiere

Pentru toate temele/proiectele, primul pas este inițializarea unui repository local în directorul în care vom păstra toate fișierele corespunzătoare temei/proiectului.
$ git init
Initialized empty Git repository in D:/School_2019_2020/OOP/.git/

Asocierea repository-ul local cu cel de pe server se realizeaza cu următoarea comandă:
$ git remote add origin https://gitlab.com/Eugen79/tema2_oop
Pe git se vor salva fișierele sursă; fișierele obiect și/sau executabile se pot genera de fiecare dată când este nevoie, pentru o configurație specificată dacă există un fișier CMake cu reguli de generare/compilare corespunzătoare.

Un fișier nou creat poate fi adăugat la repository folosind comanda *add*. Este recomandat ca toate schimbările din cadrul fișierului/fișierelor care sunt salvate pe server să aibă asociat un mesaj pentru a specifica exact ce schimbare a fost produsă ăn cadrul modificării curente.
Pentru a publica schimbările locale și pe server, se folosește comanda push:
$ git push origin master

Rezolvarea fiecărei teme/proiect presupune împărțirea temei/proiectulu în etape/faze, ca de exemplu:
- definirea ierarhiei de clase necesare rezolvării temei
- citirea și prelucrarea datelor de intrare
- adăugarea de teste
- modificări care se referă la formatarea codului.

După fiecare etapă enumerată se va salva varianta de cod pentru că în felul acesta se poate reveni oricând la varianta precedentă a codului, asigurând astfel o mai bună organizare, iar în cazul lucrului în echipă se poate identifica foarte ușor cine a făcut o anumită modificare a codului.
Așadar, după fiecare fază trebuie să publicate modificările, iar comenzi folosite sunt următoarele:
 # adaugam 3 fisiere
$ git add fisier1 fisier_2 fisier_3
 # adaugam ierarhia de clase
$ git commit -m "ierarhia de clase"
$ git push origin master

Dacă se dorește revenirea la o versiunea precedentă, se folosește mai întâi comanda *git log*:
$ git log
commit ec77f44342887202
Author: <Nume_student> <id_student>@student.ucv.ro
Date: Tue Apr 07 18:51:09 2020
* ierarhia de clase*

commit ab66f43332884582
Author: <Nume_student> <id_student>@student.ucv.ro
Date: Mon Apr 06 16:45:01 2020
* added .Readme file*

Dacă commit-ul căutat este ec77f44342887202 atunci pentru revenirea la această variantă se va folosi comanda:
$ git revert ec77f44342887202

Git tags

Pentru proiectele realizate în cadrul unei echipe, este recomandat ca fiecare membru să lucreze pe un branch, iar ulterior codul să fie unificat prin operația de ***merge***.
La nivelul unui întreg repository pentru a versiona codul se folosesc tag-uri pot fi create și gestionate pe parcursul dezvoltării proiectului:
$ git tag
v1.0
v2.0

Resurse

1. GitLab University
2. Git Immersion
3. Git Tagging