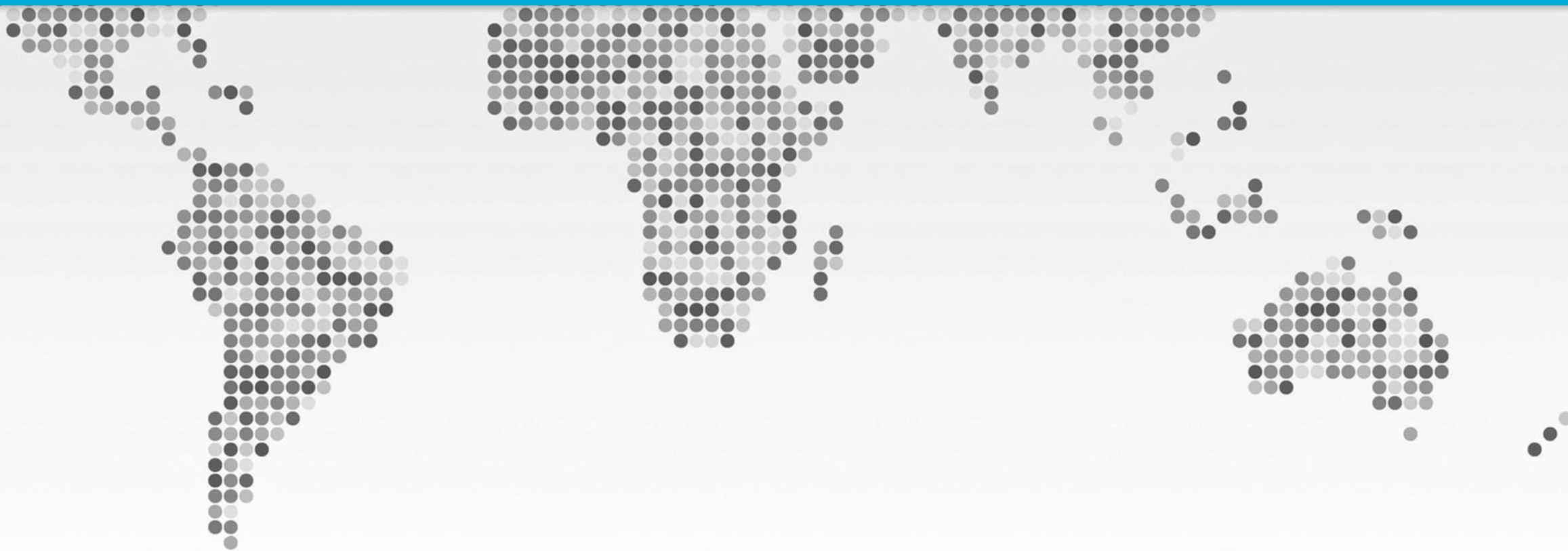


Esercitazione

Laboratorio di Ingegneria del Software



Esercitazione

L'esercitazione di oggi consiste nel fare il setup di un primo progetto.

Utilizzando il materiale di supporto che trovate sulla piattaforma e-learning, potete eseguire in autonomia i seguenti passi:

- 1) Scaricare l'IDE IntelliJ
- 2) Creare un nuovo progetto Maven in IntelliJ
- 3) Aggiungere la dipendenza JUnit al progetto Maven e compilarlo
- 4) Creare un account su GitHub
- 5) Creare una nuova repository su GitHub attraverso l'interfaccia grafica

Dopodiché, insieme caricheremo il progetto sulla piattaforma di hosting GitHub. Seguiremo “una versione modificata” di questi passi:

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

```
echo "# repository_test" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/broke31/repository_test.git
git push -u origin main
```

Esercitazione

Abbiamo il nostro progetto e vogliamo caricarlo su GitHub.

Posizioniamoci nella directory del nostro progetto e apriamo il terminale, oppure utilizziamo il terminale integrato in IntelliJ.

```
git init
```

Attraverso il comando init possiamo inizializzare la repository Git del nostro progetto.

Git ci informerà sul nome del branch di default (master o main).

```
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git init
suggerimento: Using 'master' as the name for the initial branch. This default branch name
suggerimento: is subject to change. To configure the initial branch name to use in all
suggerimento: of your new repositories, which will suppress this warning, call:
suggerimento:
suggerimento:   git config --global init.defaultBranch <name>
suggerimento:
suggerimento: Names commonly chosen instead of 'master' are 'main', 'trunk' and
suggerimento: 'development'. The just-created branch can be renamed via this command:
suggerimento:
suggerimento:   git branch -m <name>
Inizializzato repository Git vuoto in /Users/giuliasellitto/IdeaProjects/progetto-is/.git/
```

git status

Il comando status ci dà informazioni sullo stato della repository, evidenziando i file che sono memorizzati in locale, quelli che sono nell'area di staging, e quelli che sono sincronizzati in remoto.

```
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git status
Sul branch master

Non ci sono ancora commit

File non tracciati:
  (usa "git add <file>..." per includere l'elemento fra quelli di cui verrà eseguito il commit)
    .idea/
    pom.xml
    src/
    target/
```

Possiamo vedere che la nostra repository è stata appena inizializzata, perché non ci sono ancora commit. Tutti i file del nostro progetto sono presenti solamente in locale (non tracciati). Vogliamo ora aggiungerli all'area di staging, per poi farne il push.

Attenzione! Non tutti i file del progetto devono essere sincronizzati in remoto!

```
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git status
Sul branch master

Non ci sono ancora commit

File non tracciati:
  (usa "git add <file>..." per includere l'elemento fra quelli di cui verrà eseguito il commit)
    .idea/
    pom.xml
    src/
    target/
```

Attenzione! Non tutti i file del progetto devono essere sincronizzati in remoto!

I file nelle cartelle `.idea` e `target` costituiscono la configurazione locale del progetto e gli output della compilazione, quindi è giusto che rimangano soltanto in locale. Ciò che vogliamo affidare al sistema di version control e collaborazione (Git) sono solamente i file che contengono il codice, e il `pom.xml`

```
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git status
Sul branch master

Non ci sono ancora commit

File non tracciati:
  (usa "git add <file>..." per includere l'elemento fra quelli di cui verrà eseguito il commit)
    .idea/
    pom.xml
    src/
    target/
```

Attenzione! Non tutti i file del progetto devono essere sincronizzati in remoto!

I file nelle cartelle `.idea` e `target` costituiscono la configurazione locale del progetto e gli output della compilazione, quindi è giusto che rimangano soltanto in locale. Ciò che vogliamo affidare al sistema di version control e collaborazione (Git) sono solamente i file che contengono il codice, e il `pom.xml`

```
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git status
Sul branch master

Non ci sono ancora commit

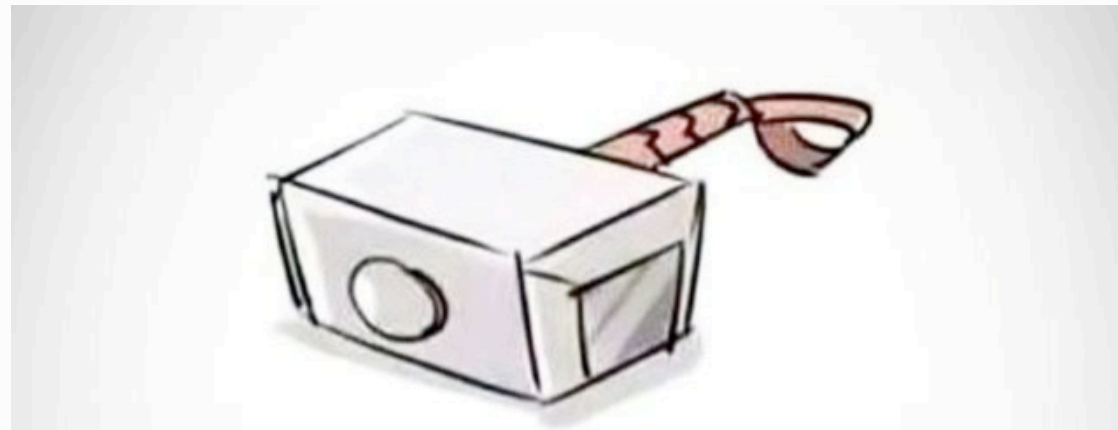
File non tracciati:
  (usa "git add <file>..." per includere l'elemento fra quelli di cui verrà eseguito il commit)
    .idea/
    pom.xml
    src/
    target/
```

Come facciamo a “dire” a Git che non vogliamo caricare sulla repository remota i file nelle cartelle `.idea` e `target`?

Attraverso il file `.gitignore`

Perché è importante il file .gitignore?

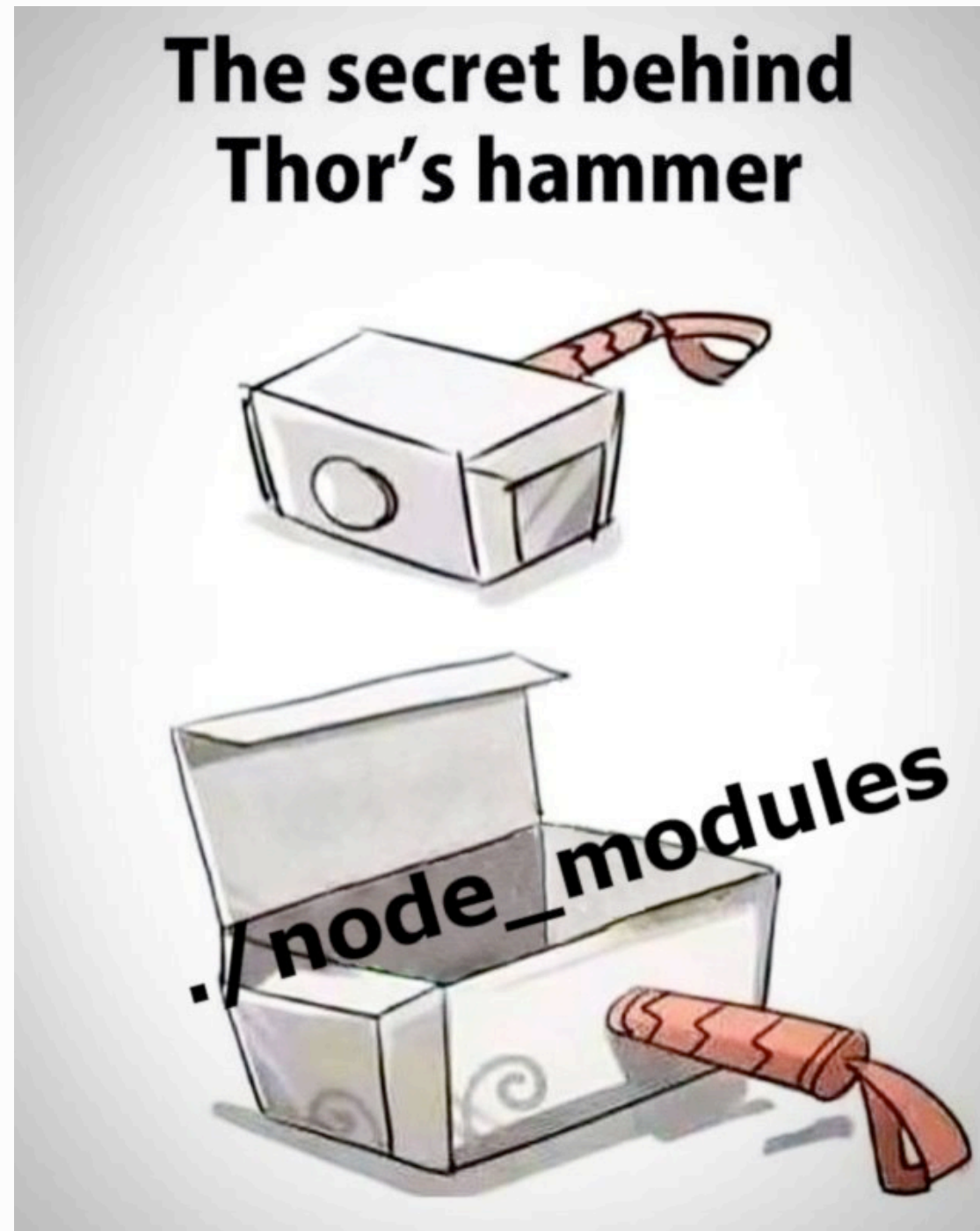
Perché è importante il file .gitignore?



Perché è importante il file .gitignore?



Perché è importante il file .gitignore?



Il file .gitignore

Il file .gitignore specifica le cartelle e i file del progetto che NON vogliamo caricare sulla repository remota. Consiste in un semplice file di testo, in cui ogni riga rappresenta un file o una cartella che vogliamo escludere dalla sincronizzazione.

Il file .gitignore

Il file .gitignore specifica le cartelle e i file del progetto che NON vogliamo caricare sulla repository remota. Consiste in un semplice file di testo, in cui ogni riga rappresenta un file o una cartella che vogliamo escludere dalla sincronizzazione.

Per creare il nostro file .gitignore, utilizziamo IntelliJ e selezioniamo la directory parent del progetto. Con il tasto destro, scegliamo di creare un nuovo file, e lo chiamiamo .gitignore

Attenzione: è molto importante che il file si trovi nella directory parent del progetto, e che il suo nome sia esattamente .gitignore

Il file .gitignore

Il file .gitignore specifica le cartelle e i file del progetto che NON vogliamo caricare sulla repository remota. Consiste in un semplice file di testo, in cui ogni riga rappresenta un file o una cartella che vogliamo escludere dalla sincronizzazione.

Per creare il nostro file .gitignore, utilizziamo IntelliJ e selezioniamo la directory parent del progetto. Con il tasto destro, scegliamo di creare un nuovo file, e lo chiamiamo .gitignore

Attenzione: è molto importante che il file si trovi nella directory parent del progetto, e che il suo nome sia esattamente .gitignore

Ora cosa scriviamo nel file?

Il file .gitignore

Ora cosa scriviamo nel file?

Esistono dei template di file .gitignore relativi a specifici linguaggi di programmazione o framework utilizzati per lo sviluppo.

Il nostro è un progetto Java che utilizza Maven come sistema di build, quindi possiamo copiare e incollare nel nostro file .gitignore i template messi a disposizione da GitHub:

<https://github.com/github/gitignore/blob/main/Java.gitignore>

<https://github.com/github/gitignore/blob/main/Maven.gitignore>

Inoltre, dobbiamo aggiungere una riga in fondo al file per ignorare un'ulteriore cartella:

.idea/

Esercitazione

Avendo aggiunto il file .gitignore al nostro progetto, abbiamo indicato i file e le cartelle che non vogliamo porre sotto version control.

Vediamo cosa ci dice Git se invochiamo di nuovo il comando:

```
git status
```

```
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git status
Sul branch master

Non ci sono ancora commit

Modifiche di cui verrà eseguito il commit:
  (usa "git rm --cached <file>..." per rimuovere gli elementi dall'area di staging)
    nuovo file:          .gitignore

Modifiche non nell'area di staging per il commit:
  (usa "git add <file>..." per aggiornare gli elementi di cui sarà eseguito il commit)
  (usa "git restore <file>..." per scartare le modifiche nella directory di lavoro)
    modificato:          .gitignore

File non tracciati:
  (usa "git add <file>..." per includere l'elemento fra quelli di cui verrà eseguito il commit)
    pom.xml
    src/
```

Possiamo vedere che le cartelle target e .idea non sono più considerate da Git, come se non esistessero nella directory del progetto.

Ora possiamo aggiungere tutti i file all'area di staging:

```
git add .
```

Attenzione allo spazio prima del punto!

```
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git add .
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git status
Sul branch master

Non ci sono ancora commit

Modifiche di cui verrà eseguito il commit:
  (usa "git rm --cached <file>..." per rimuovere gli elementi dall'area di staging)
    nuovo file:          .gitignore
    nuovo file:          pom.xml
    nuovo file:          src/main/java/it/unisa/gisellitto/Main.java
```

I file aggiunti all'area di staging sono esclusivamente quelli che vogliamo considerare, ovvero: i file sorgenti (.java), il pom.xml, e il file .gitignore

Ora possiamo fare il commit, facendo attenzione a scrivere un buon messaggio:

```
git commit -m "Upload project"
```

Esercitazione

Ora possiamo fare il commit, facendo attenzione a scrivere un buon messaggio:

```
git commit -m "Upload project"
```

Per sincronizzare le modifiche nella repository remota, dobbiamo aggiungere il riferimento remoto alla nostra repository locale.

Recuperiamo il link della nostra repository remota su GitHub (lo stesso link che serve per fare il clone della repository).

```
git remote add origin <repo-remote-link>
```

Esercitazione

Ora possiamo fare il commit, facendo attenzione a scrivere un buon messaggio:

```
git commit -m "Upload project"
```

Per sincronizzare le modifiche nella repository remota, dobbiamo aggiungere il riferimento remoto alla nostra repository locale.

Recuperiamo il link della nostra repository remota su GitHub (lo stesso link che serve per fare il clone della repository).

```
git remote add origin <repo-remote-link>
```

```
git remote -v
```

Ci permette di verificare il collegamento tra la repository remota e quella locale

Esercitazione

Ora possiamo fare il commit, facendo attenzione a scrivere un buon messaggio:

```
git commit -m "Upload project"
```

Per sincronizzare le modifiche nella repository remota, dobbiamo aggiungere il riferimento remoto alla nostra repository locale.

Recuperiamo il link della nostra repository remota su GitHub (lo stesso link che serve per fare il clone della repository).

```
git remote add origin <repo-remote-link>
```

```
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git remote add origin https://github.com/giuliasellitto7/Laboratorio-IS.git
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git remote -v
origin https://github.com/giuliasellitto7/Laboratorio-IS.git (fetch)
origin https://github.com/giuliasellitto7/Laboratorio-IS.git (push)
```

```
git remote -v
```

Ci permette di verificare il collegamento tra la repository remota e quella locale

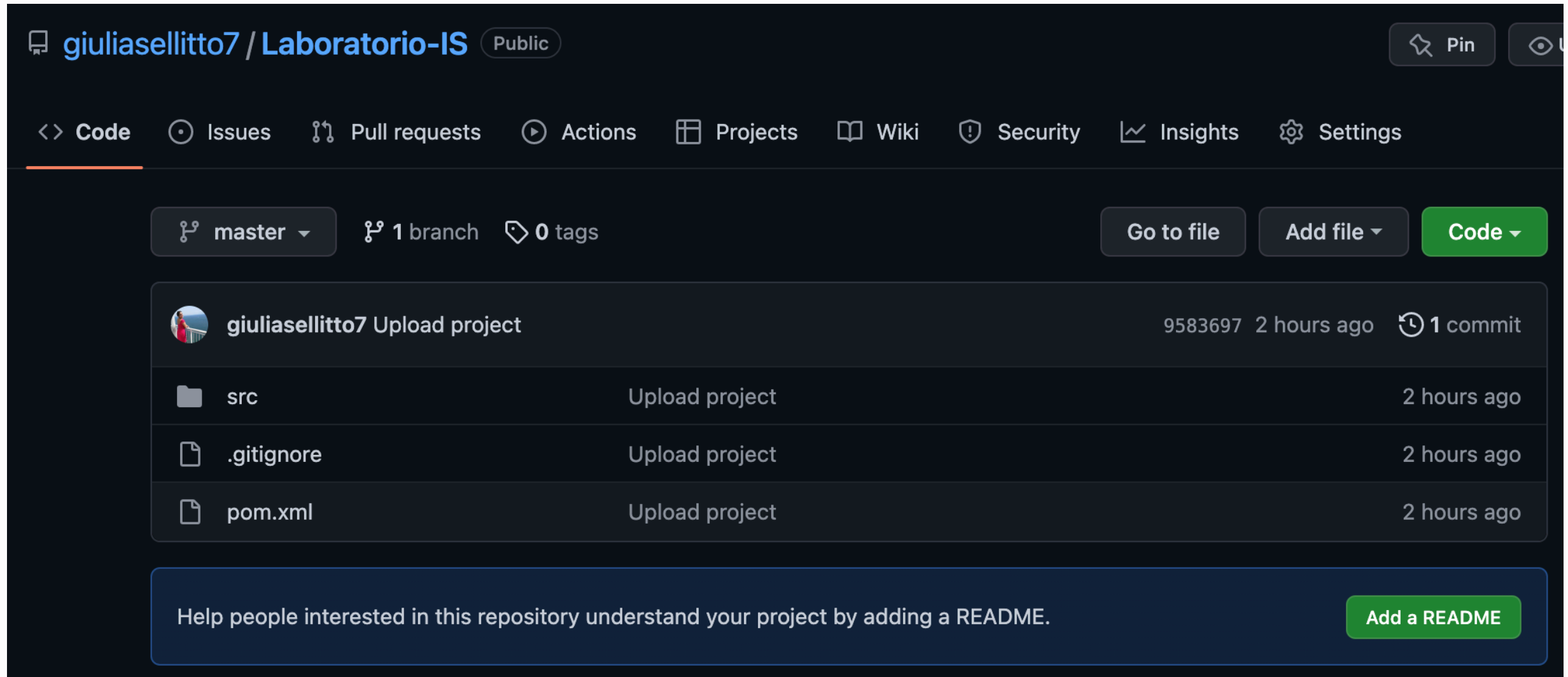
Ora possiamo fare il push del commit alla repository remota, stando attenti ad indicare il branch su cui vogliamo sincronizzare le modifiche:

```
git push -u origin <branch-name>
```

```
MacBook-Pro-di-Giulia:progetto-is giuliasellitto$ git push -u origin master
Enumerazione degli oggetti in corso: 17, fatto.
Conteggio degli oggetti in corso: 100% (17/17), fatto.
Compressione delta in corso, uso fino a 8 thread
Compressione oggetti in corso: 100% (7/7), fatto.
Scrittura degli oggetti in corso: 100% (17/17), 1.85 KiB | 945.00 KiB/s, fatto.
17 oggetti totali (0 delta), 0 riutilizzati (0 delta), 0 riutilizzati nel file pack
To https://github.com/giuliasellitto7/Laboratorio-IS.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

Esercitazione

Andando su GitHub, possiamo vedere che il nostro progetto è stato correttamente caricato sulla piattaforma di hosting.



The screenshot shows the GitHub interface for a repository named 'giuliasellitto7 / Laboratorio-IS'. The repository is public. The 'Code' tab is selected, showing the file structure. The repository has 1 branch (master) and 0 tags. The commit history shows a single commit by giuliasellitto7, titled 'Upload project', with a commit hash of 9583697, made 2 hours ago. The commit includes three files: 'src', '.gitignore', and 'pom.xml', all uploaded 2 hours ago. A prompt at the bottom encourages adding a README file.

giuliasellitto7 / Laboratorio-IS Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags Go to file Add file Code

giuliasellitto7 Upload project 9583697 2 hours ago 1 commit

src	Upload project	2 hours ago
.gitignore	Upload project	2 hours ago
pom.xml	Upload project	2 hours ago

Help people interested in this repository understand your project by adding a README. Add a README

Esercitazione

Laboratorio di Ingegneria del Software

