

Zadanie 1

Utwórz konto i pierwsze repozytorium (folderu). GitHub – serwis/platforma, od podstaw przeprowadza przez proces tworzenia konta i umieszczenia w nim pierwszego projektu.

1. Rejestracja w serwisie

<https://github.com/join>

Zarejestruj nowe konto na GitHub'ie przez: github.com/join. Podaj podstawowe dane i utwórz nazwę użytkownika - nazwę tę widać publicznie, konto na GitHub'ie w przyszłości podasz swojemu pracodawcy

Join GitHub · GitHub

GitHub, Inc. [US] | github.com/join

Why GitHub? Enterprise Explore Marketplace Pricing Search GitHub Sign in Sign up

Join GitHub

The best way to design, build, and ship software.

Step 1:
Set up your account

Step 2:
Choose your subscription

Step 3:
Tailor your experience

Create your personal account

Username *

KatarzynaGawrys

Username KatarzynaGawrys is not available.
KatarzynaGawrys-stack, KatarzynaGawrys-ux, or KatarzynaGawrys580 are available.

Email address *

We'll occasionally send updates about your account to this inbox. We'll never share your email address with anyone.

Password *

You'll love GitHub

- Unlimited public repositories
- Unlimited private repositories
- ✓ Limitless collaboration
- ✓ Frictionless development
- ✓ Open source community

Po założeniu konta na platformie GitHub, będziesz mógł udostępniać projekty w Internecie i umieszczać w terminie zadania projektowe.

Po rejestracji zainstaluj program/narzędzie Git – system kontroli wersji, dla swojego systemu operacyjnego ze strony:

<https://git-scm.com/>

2. Instalacja Git – Windows

Podczas instalacji zmień domyślne ustawienie korzystania z edytora Vim – wybierz edytor Nano – prosty i konsolowy edytor dobrze współpracujący z programem Git

Na pytanie dotyczące kontekstu w jakim chcemy korzystać z git: 1. opcja pozwala na używanie Git tylko w ramach Git Bash, 2. Instalator dodatkowo dopisze pliki wykonywalne git do zmiennej

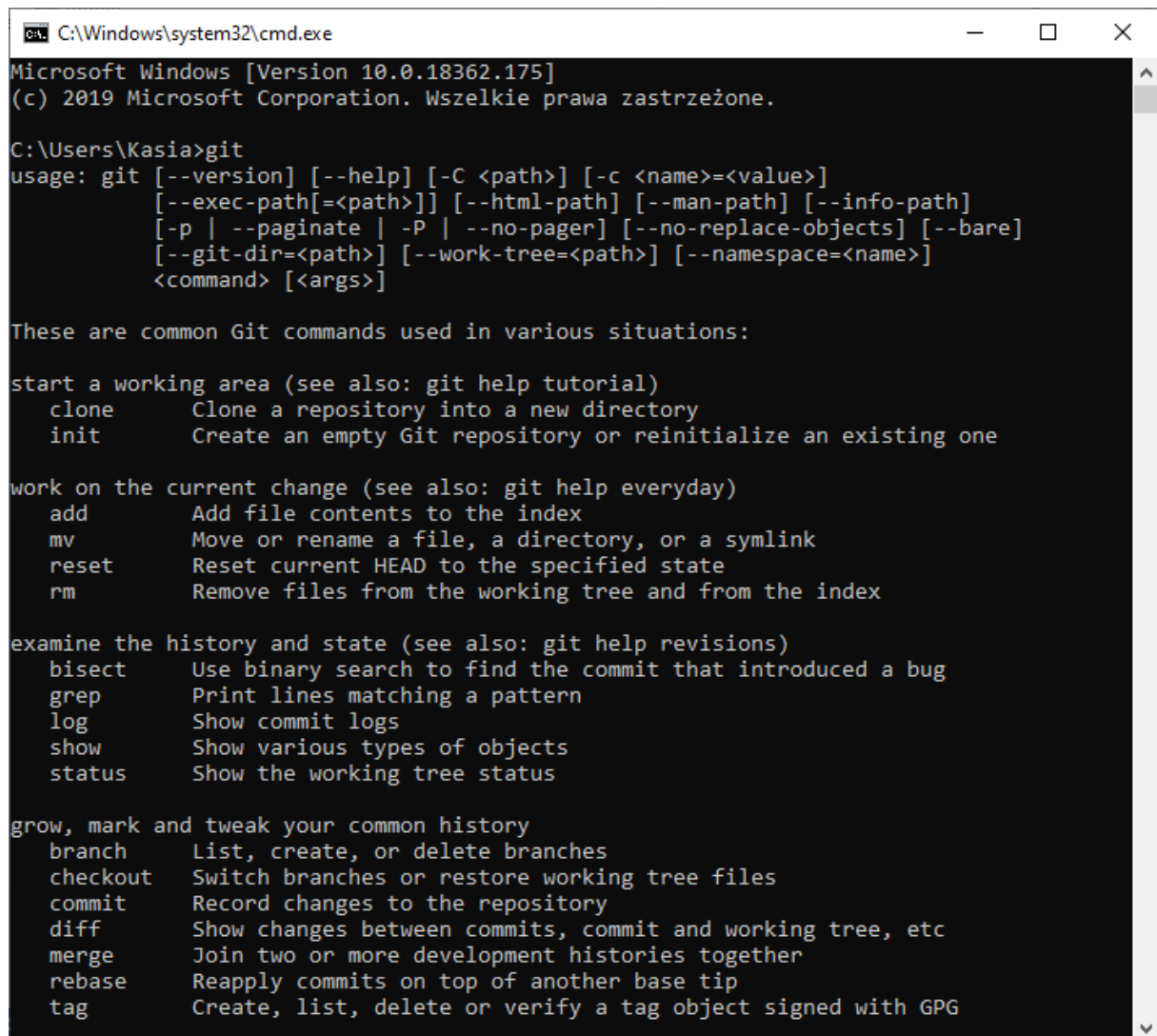
środowiskowej Path – dzięki czemu będzie można korzystać z git w standardowej linii poleceń. 3. opcja poza git, doda do zmiennej path szereg opcjonalnych narzędzi linuxowych, które są częścią git – przykryją one część natywnych poleceń Windows - Wybierz opcje 2.

Wybierz bibliotekę obsługującą szyfrowane połączenia – OpenSSL był przez długi czas jedyna opcja, od niedawna można wybrać natywne biblioteki Windows – wybierz Open SSL

Git pochodzi z linux, pomiędzy linux i Windows istnieje szereg różnic, w tym jak zakańczamy linie w plikach, w linux jest to znak końca linii, w windows mamy dodatkowo znak powrotu karetki – to może powodować różne problemy, zwłaszcza gdy na jednym repozytorium pracują osoby pracujące na różnych systemach operacyjnych. Opcja 1. najbardziej praktyczna dla użytkowników windows - git podczas pobierania plików z repozytorium zamienia linuxowe znaki końca linii na natywne dla windows, odwrotnie przy zapisie. - Wybierz opcję 1.

Kolejny ekran – dotyczy tylko Git Bash – zostaw domyślną opcje

Po instalacji sprawdź: (Windows+R – exe) run/cmd.exe wpisz: git



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.18362.175]
(c) 2019 Microsoft Corporation. Wszelkie prawa zastrzeżone.

C:\Users\Kasia>git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
        [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
        <command> [<args>]

These are common Git commands used in various situations:


start a working area (see also: git help tutorial)
    clone      Clone a repository into a new directory
    init       Create an empty Git repository or reinitialize an existing one

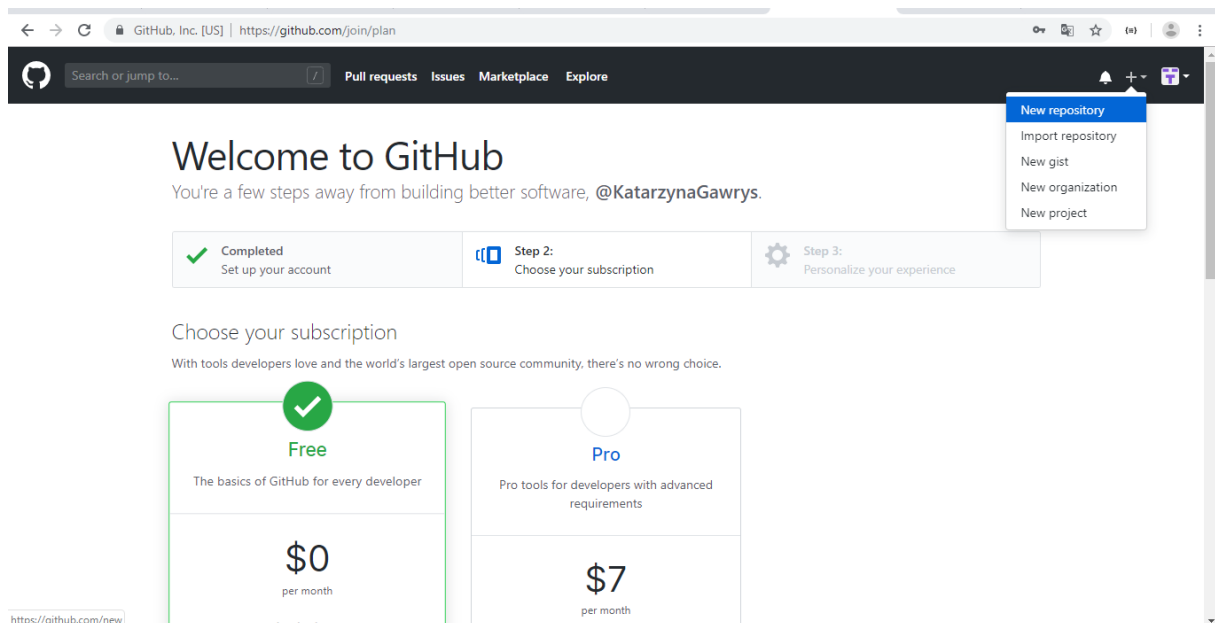

work on the current change (see also: git help everyday)
    add        Add file contents to the index
    mv         Move or rename a file, a directory, or a symlink
    reset      Reset current HEAD to the specified state
    rm         Remove files from the working tree and from the index


examine the history and state (see also: git help revisions)
    bisect     Use binary search to find the commit that introduced a bug
    grep       Print lines matching a pattern
    log        Show commit logs
    show       Show various types of objects
    status     Show the working tree status

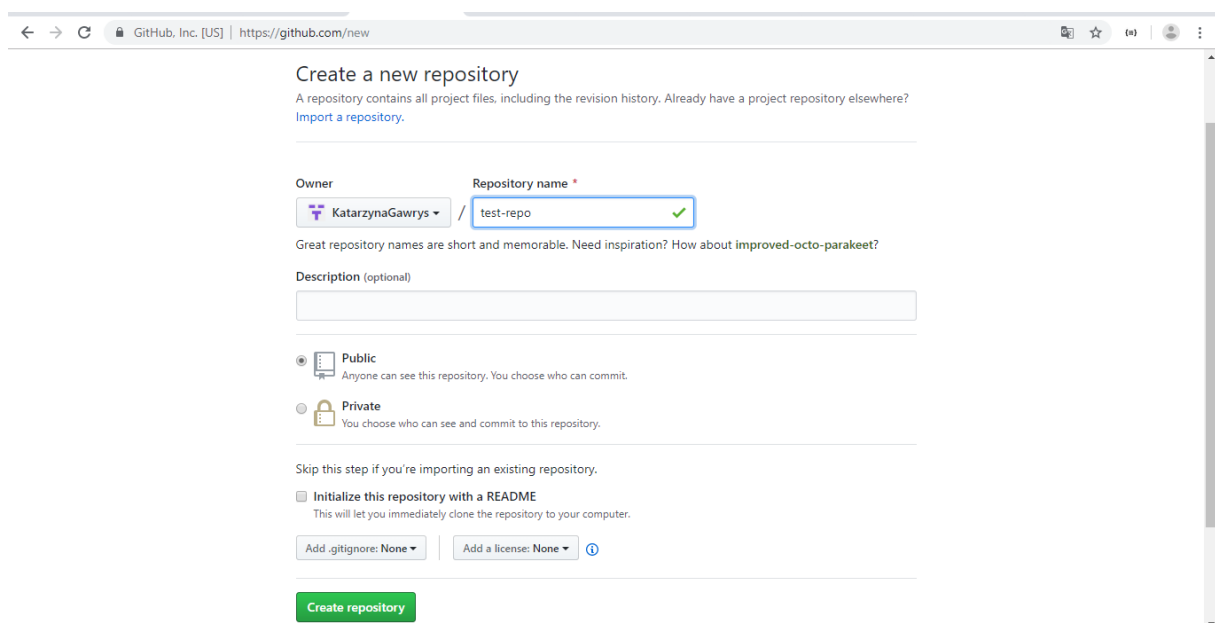

grow, mark and tweak your common history
    branch     List, create, or delete branches
    checkout   Switch branches or restore working tree files
    commit     Record changes to the repository
    diff       Show changes between commits, commit and working tree, etc
    merge      Join two or more development histories together
    rebase     Reapply commits on top of another base tip
    tag        Create, list, delete or verify a tag object signed with GPG
```

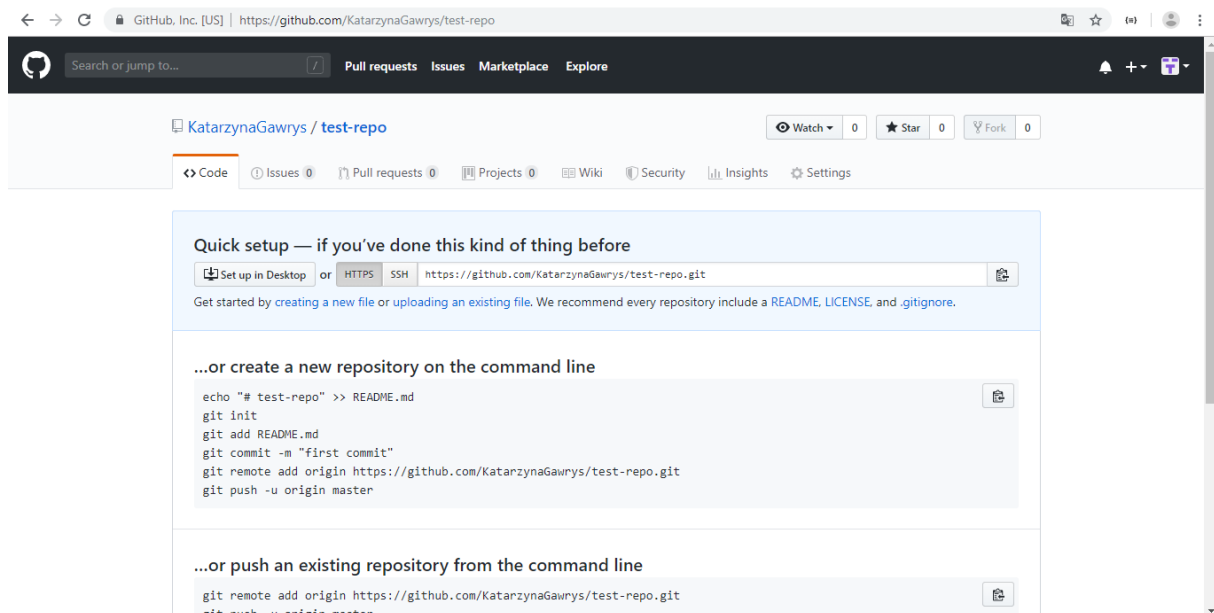
3. Utworzenie pierwszego repozytorium na GitHub'ie

Wróć do przeglądarki, na stronę główną GitHub.com. Dodaj nowe repozytorium:

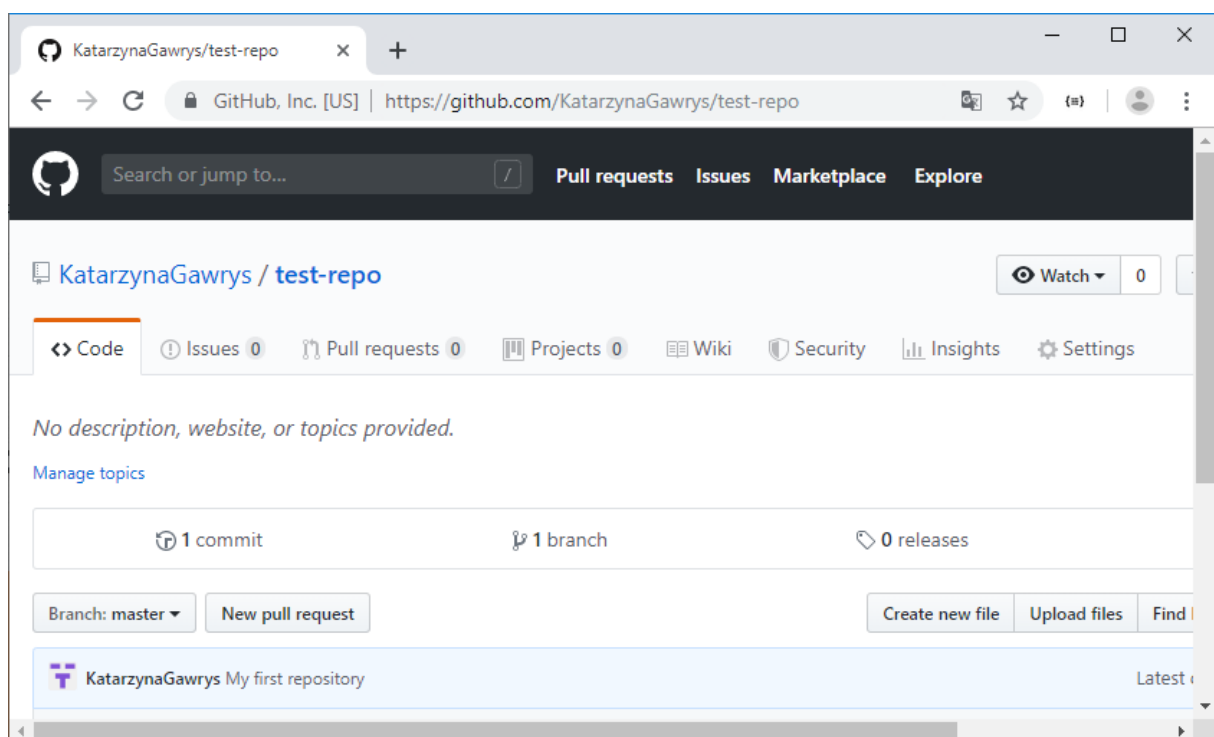


Podaj nazwę folderu (może być inna niż ta, którą masz u siebie na komputerze) oraz dodaj krótki opis projektu. Przykładowo - pierwsze repozytorium:



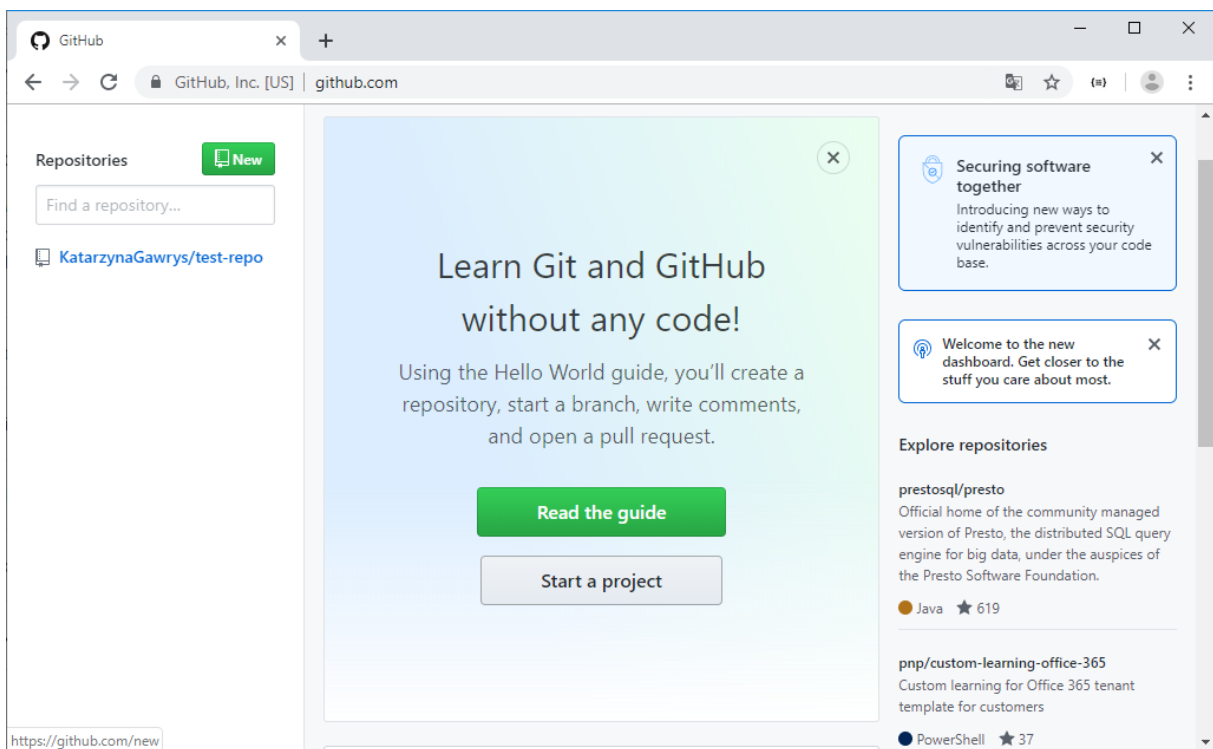


4. Dodanie plików z komputera do repozytorium

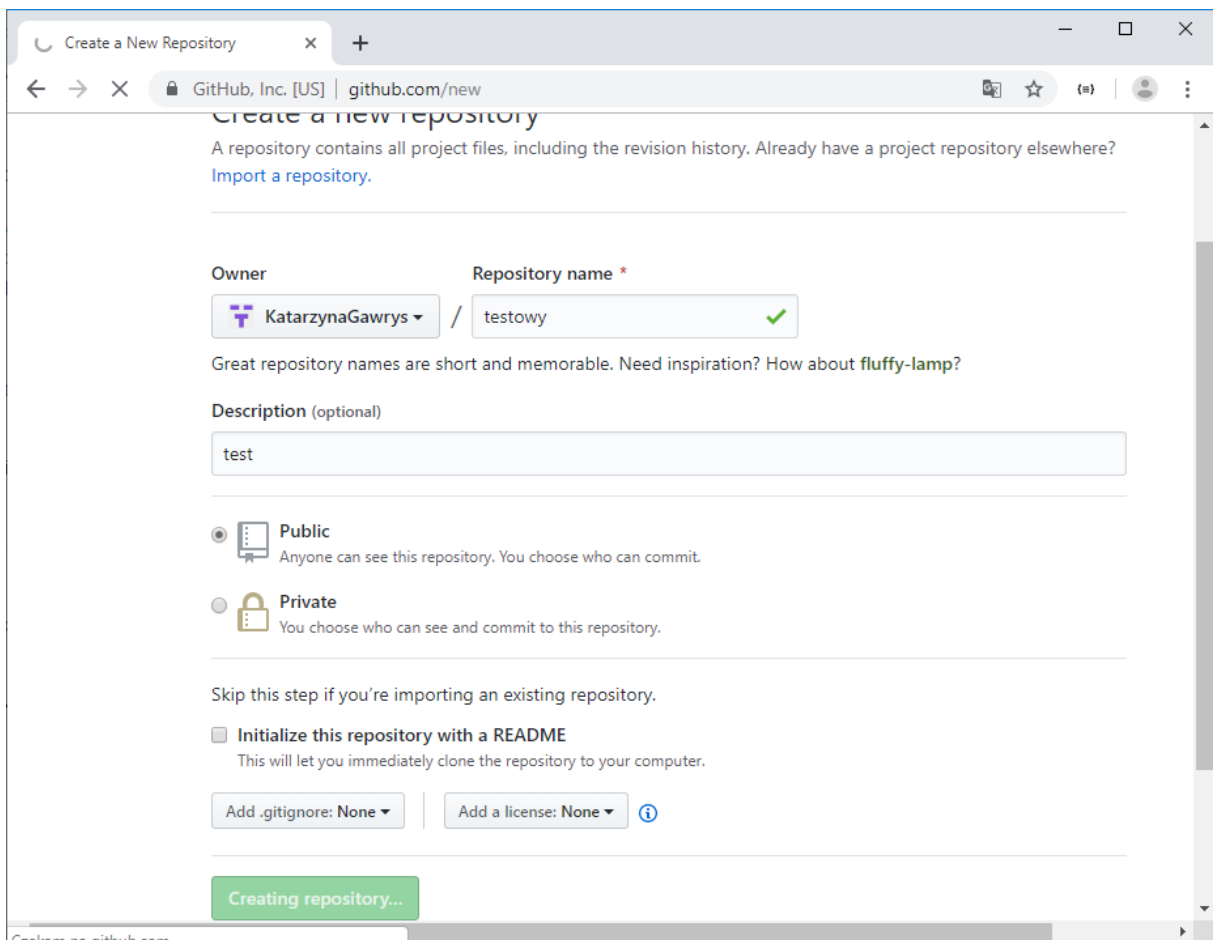


Praca w zespole - repozytoria zakładamy, aby współdzielić kod, nad którym wspólnie pracujemy oraz aby mieć kontrolę nad swoim projektem, albo dzielimy się pracą nad kodem. GitHub to duża strona społecznościowa, gdzie programiści zapisują swój kod, Git to narzędzie współpracujące ze stroną GitHub'a. Git może współpracować również z serwerem, który zakupimy i skonfigurujemy do umieszczania na nim plików.

Tworzymy nowy projekt na GitHub'ie. Kliknij przycisk New lub Start a project:

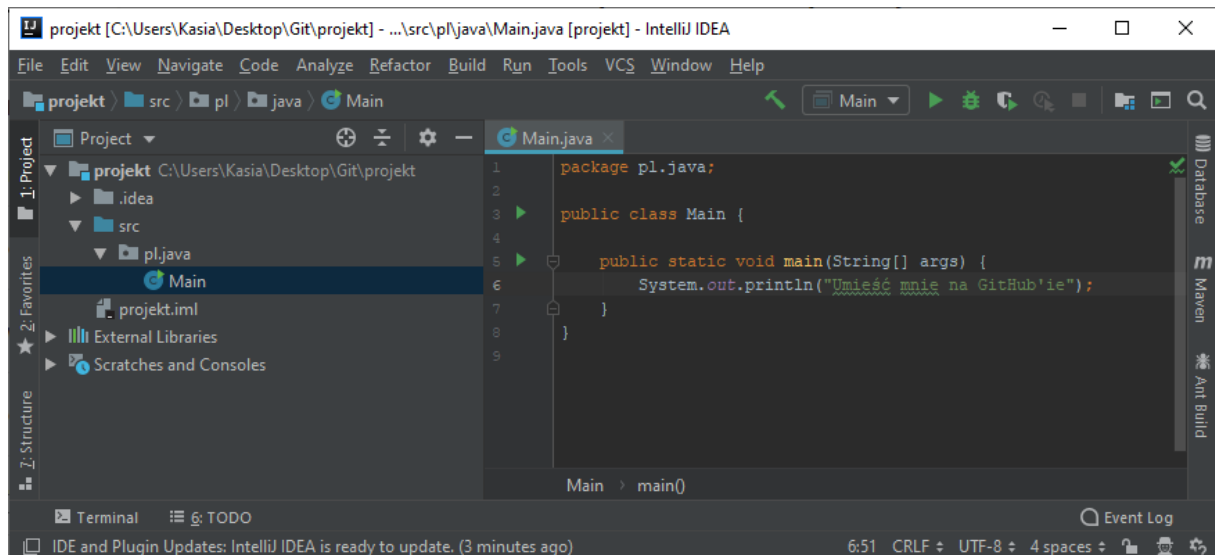


Wpisz jego nazwę i opis:



Następnie możesz wybrać opcja https lub ssh. Opcja ssh wymaga konfiguracji i nie wymaga późniejszego logowania, opcja http wymaga logowania ale nie konfiguracji. Wybierz opcję https. Więcej informacji w pliku README

Uruchom IntelliJ i utwórz nowy projekt typu Java i Commandline:



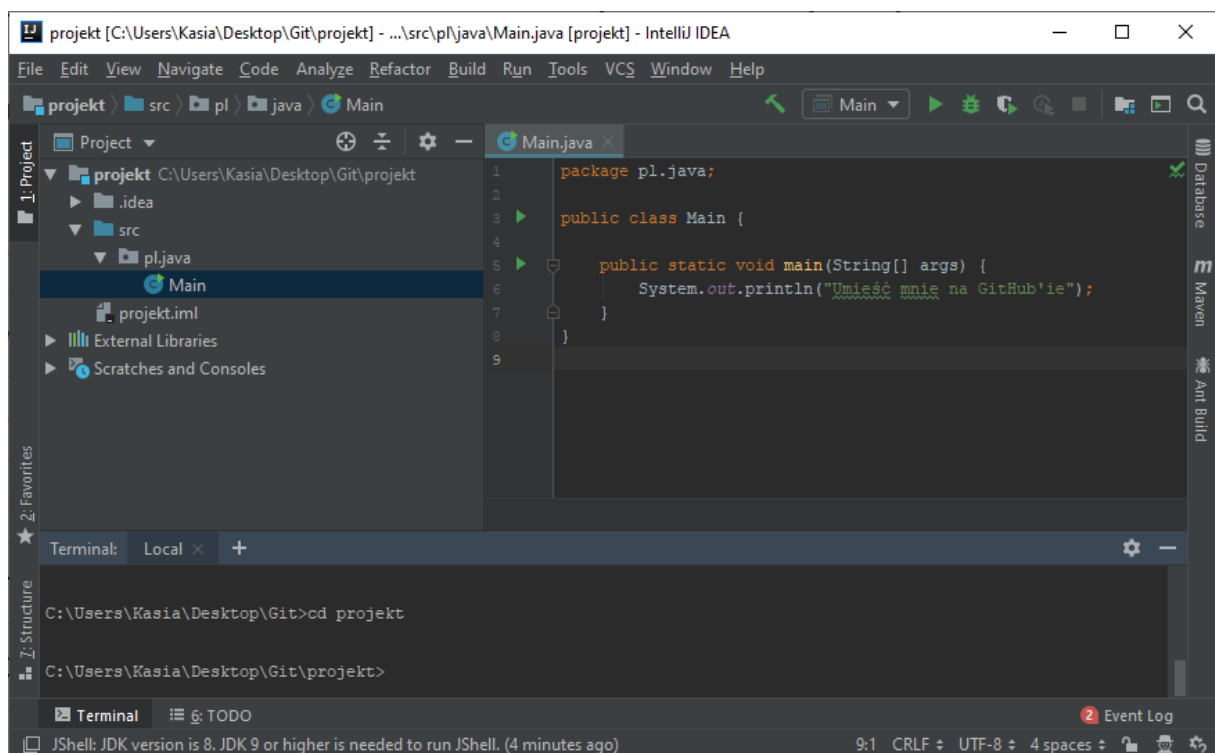
Na GitHub'ie jest opisane jak umieścić utworzony projekt w utworzonym repozytorium:

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

```
echo "# testowy" >> README.md
```

Wykonuj kolejno opisane kroki:

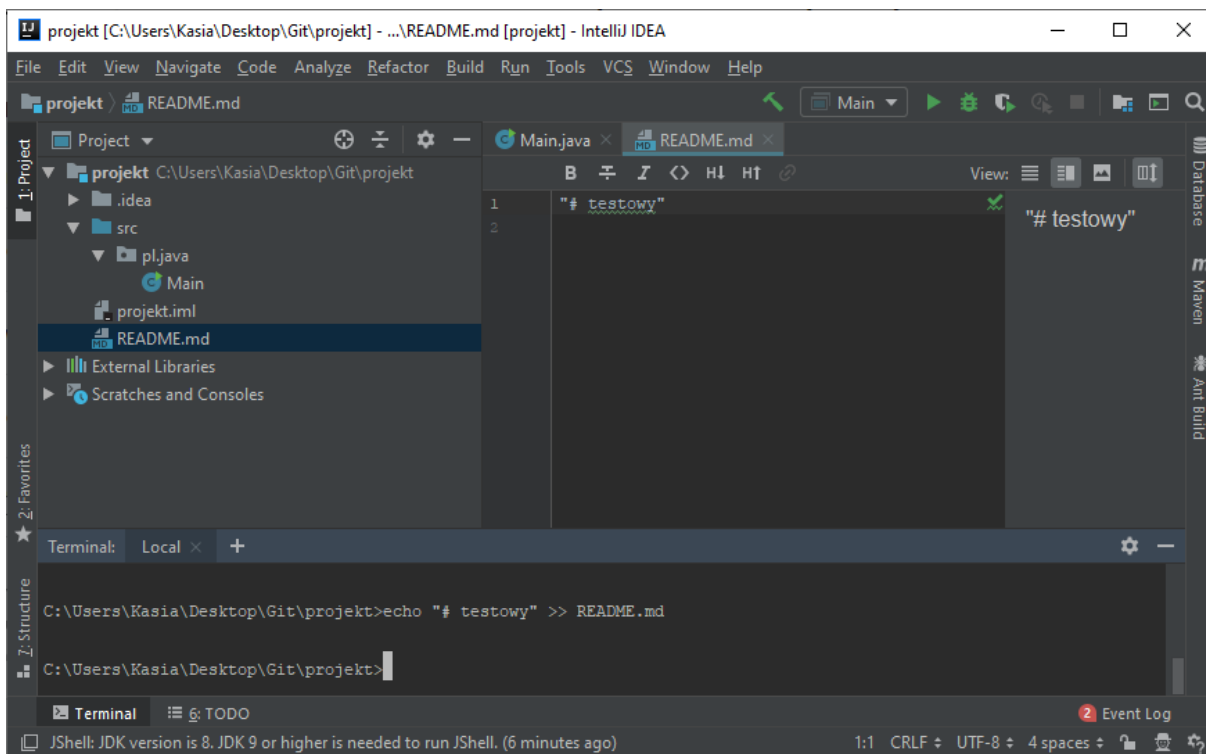
Uruchom w IntelliJ console – Alt+F12



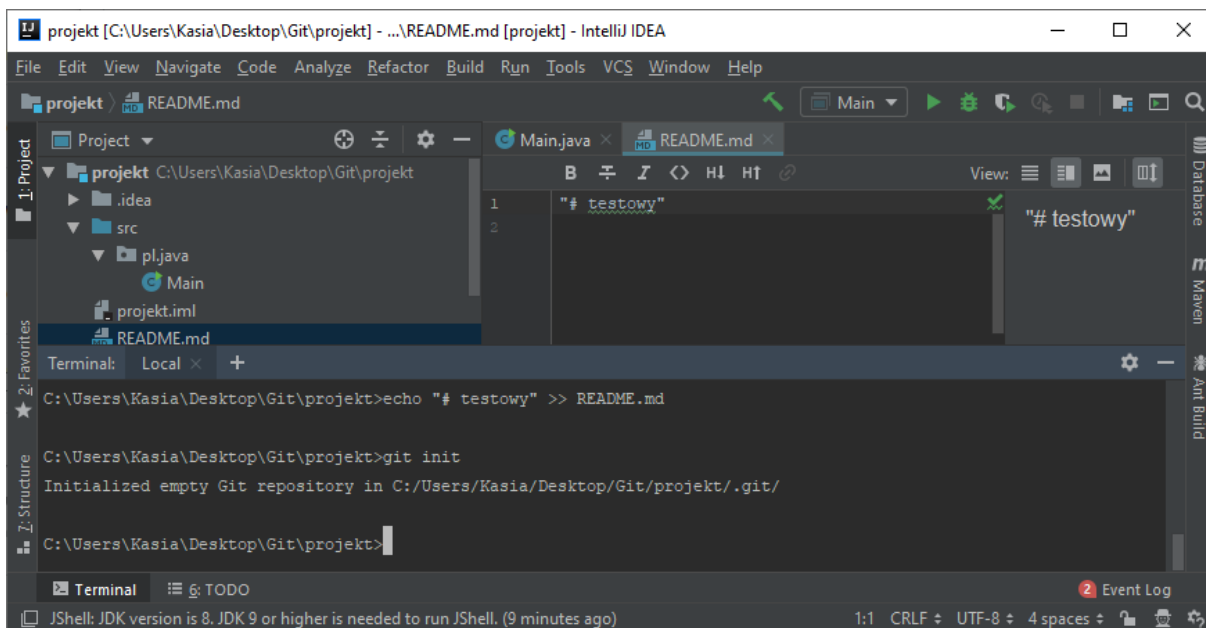
Wpisz w konsoli:

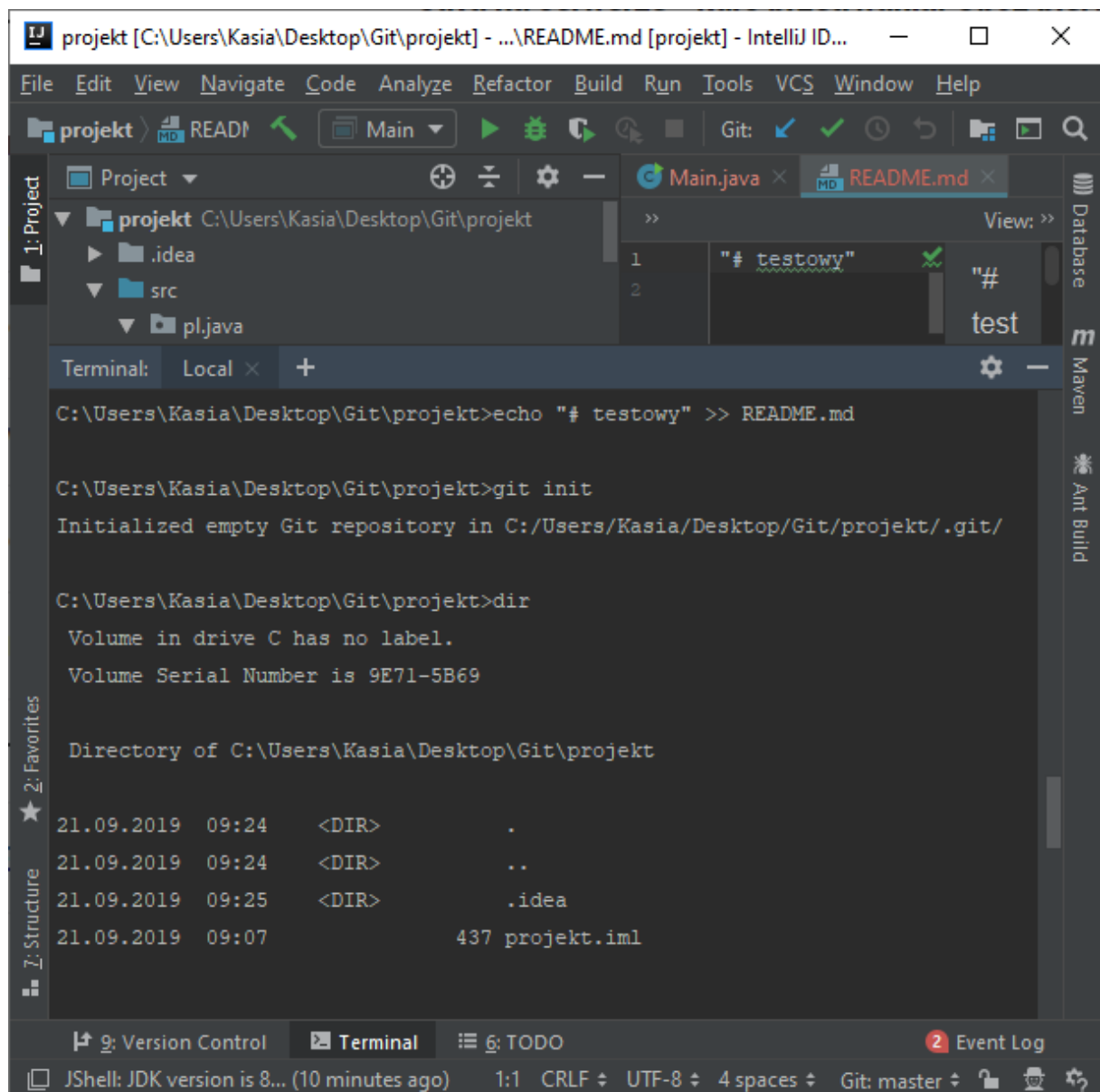
```
C:\Users\Kasia\Desktop\Git\projekt>echo "# testowy" >> README.md
```

Jest to komenda, która tworzy plik README, żeby w GitHub'ie był umieszczony opis projektu

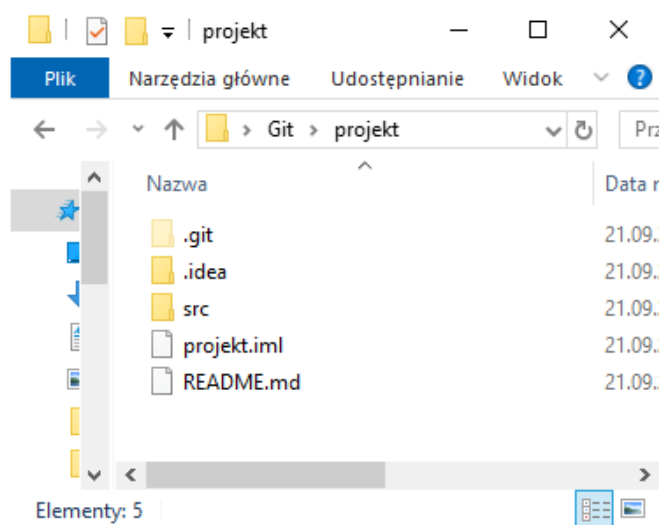


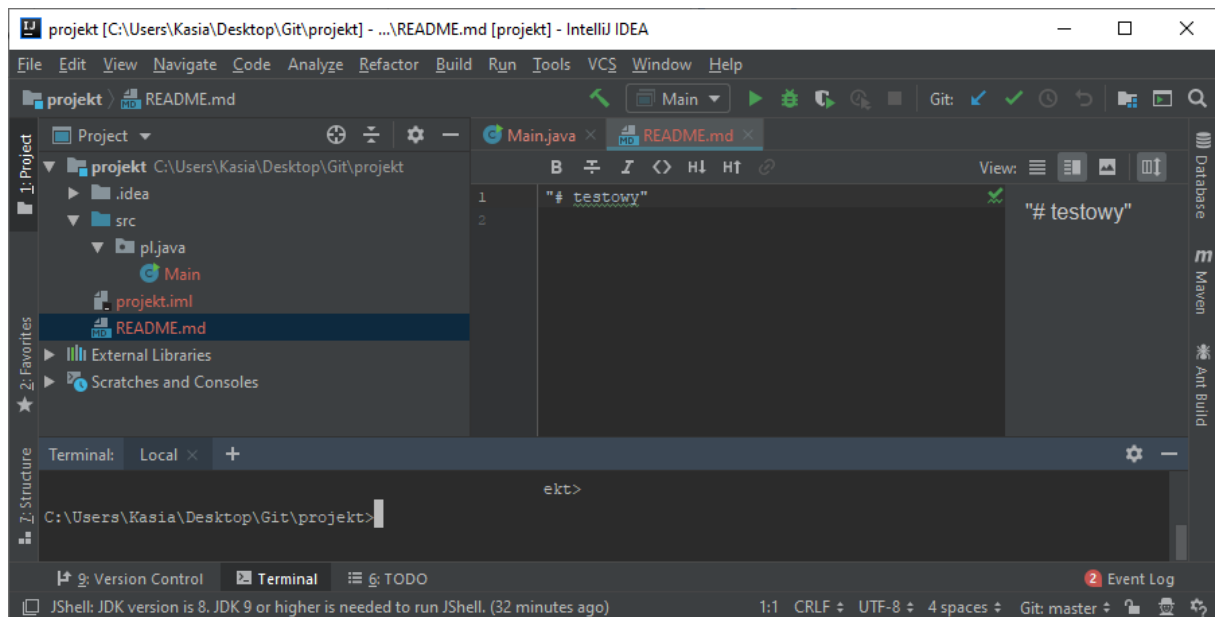
Teraz wpisz w konsole bardzo ważną komendę: `git init`, która zakłada repozytorium lokalne:





Otrzymujemy:

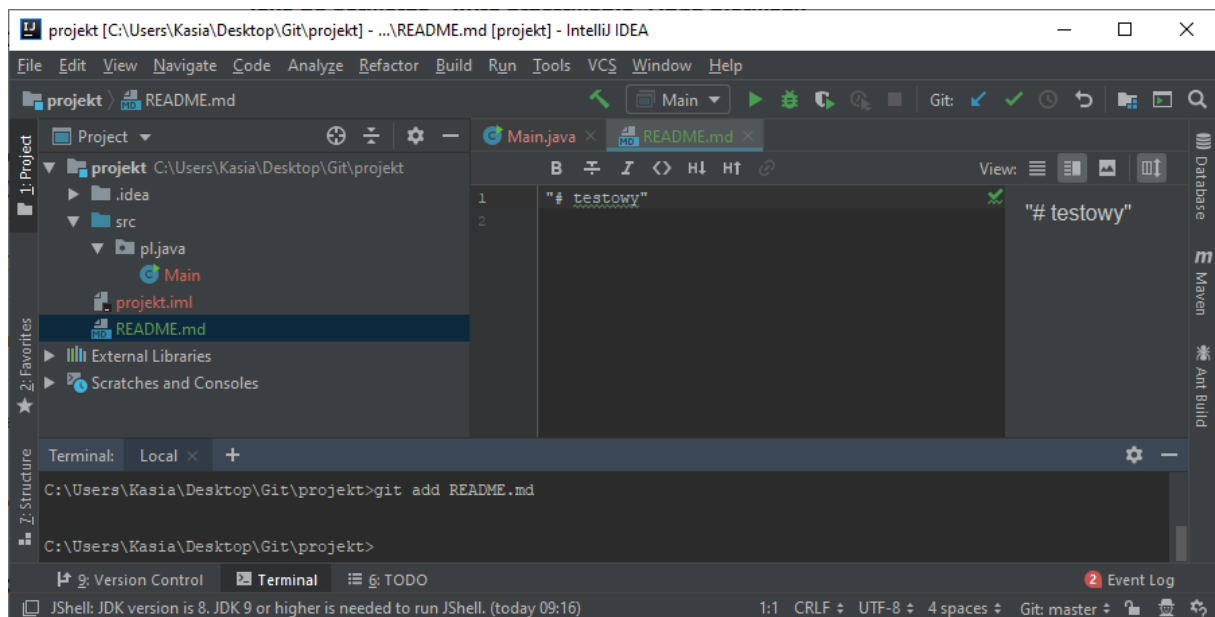




Utworzenie lokalnego repozytorium oznacza miejsce, gdzie lokalnie składujemy wszystkie wersje projektu.

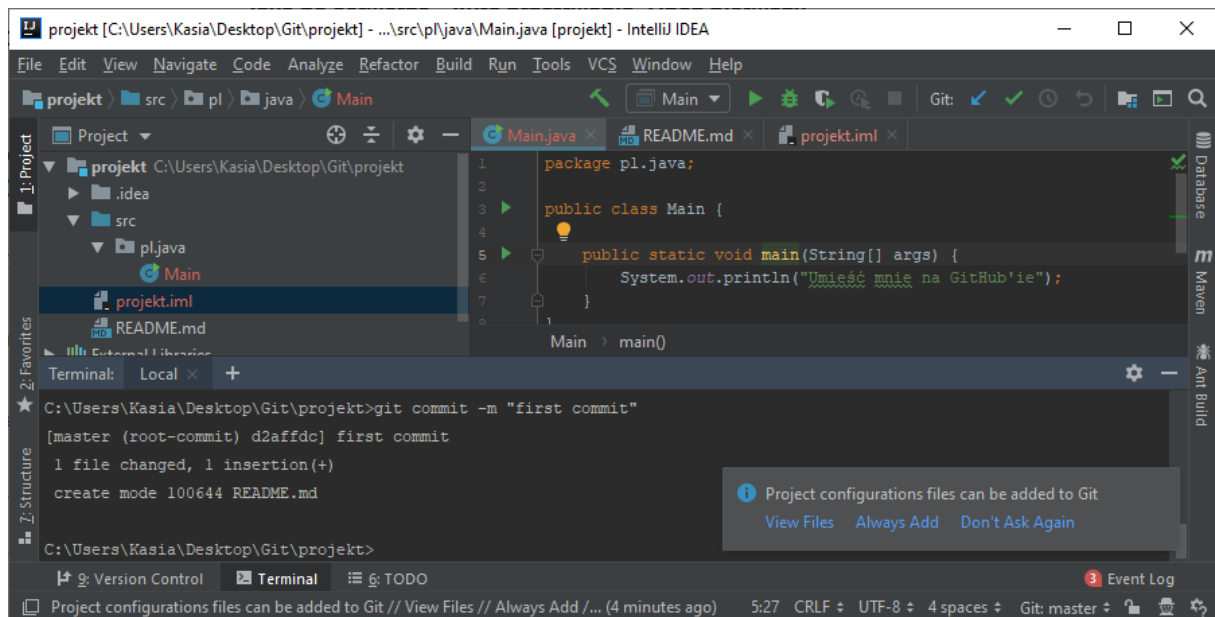
Wpisz kolejne polecenie: `git add README.md`

To znaczy że plik README zostanie dodany do obszaru roboczego – staging area, jego nazwa już nie będzie się podświetlała na czerwono lecz na zielono



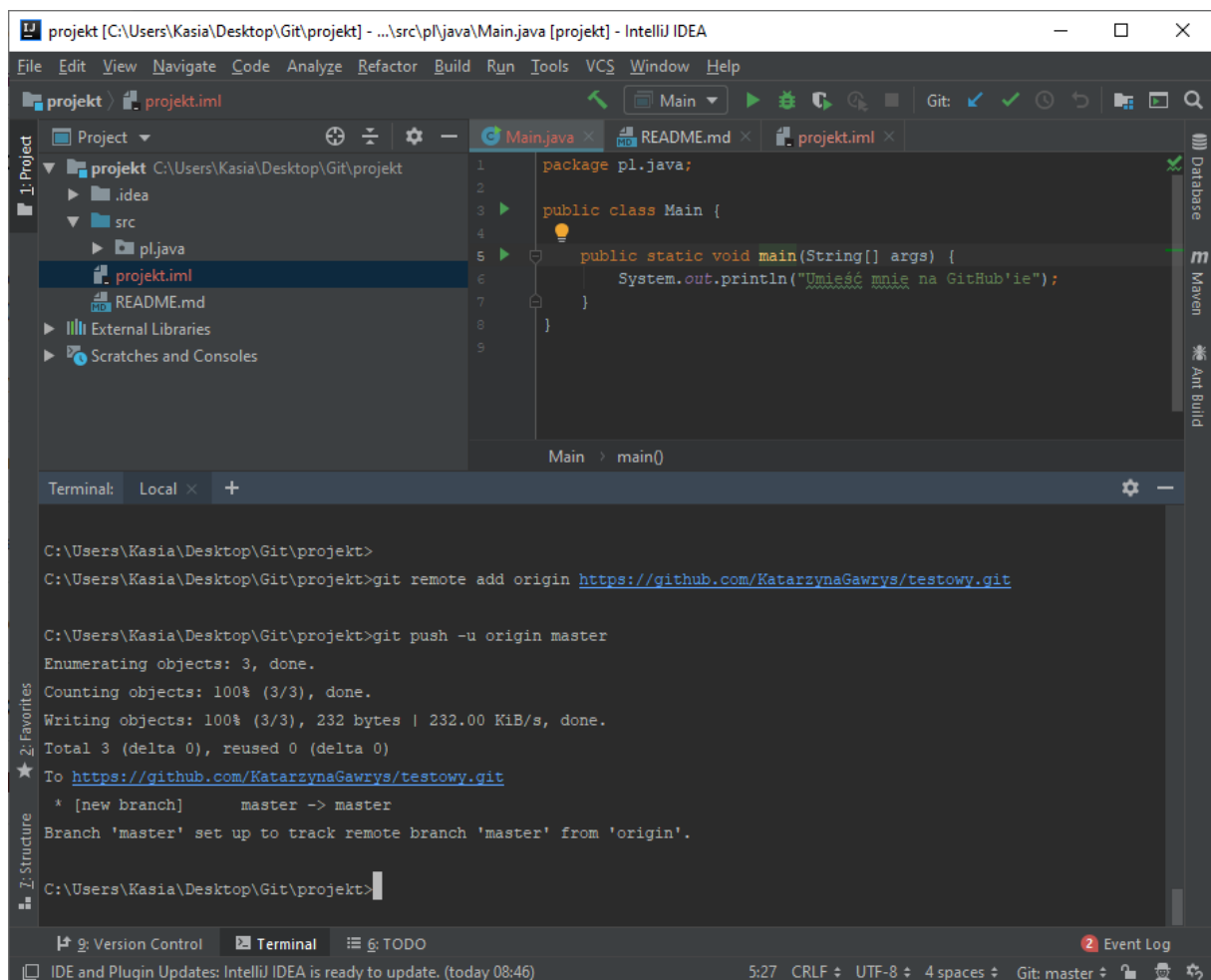
Utwórz teraz pierwszy commit, co oznacza, że tworzymy wersję i ma ona swoją nazwę:

`git commit -m "first commit"`



Utworzyliśmy pierwszą wersję systemu. W kolejnej opcji podajemy miejsce, gdzie znajduje się nasze zdalne repozytorium, u nas jest to GitHub ale mógłby to być również nasz prywatny serwer:

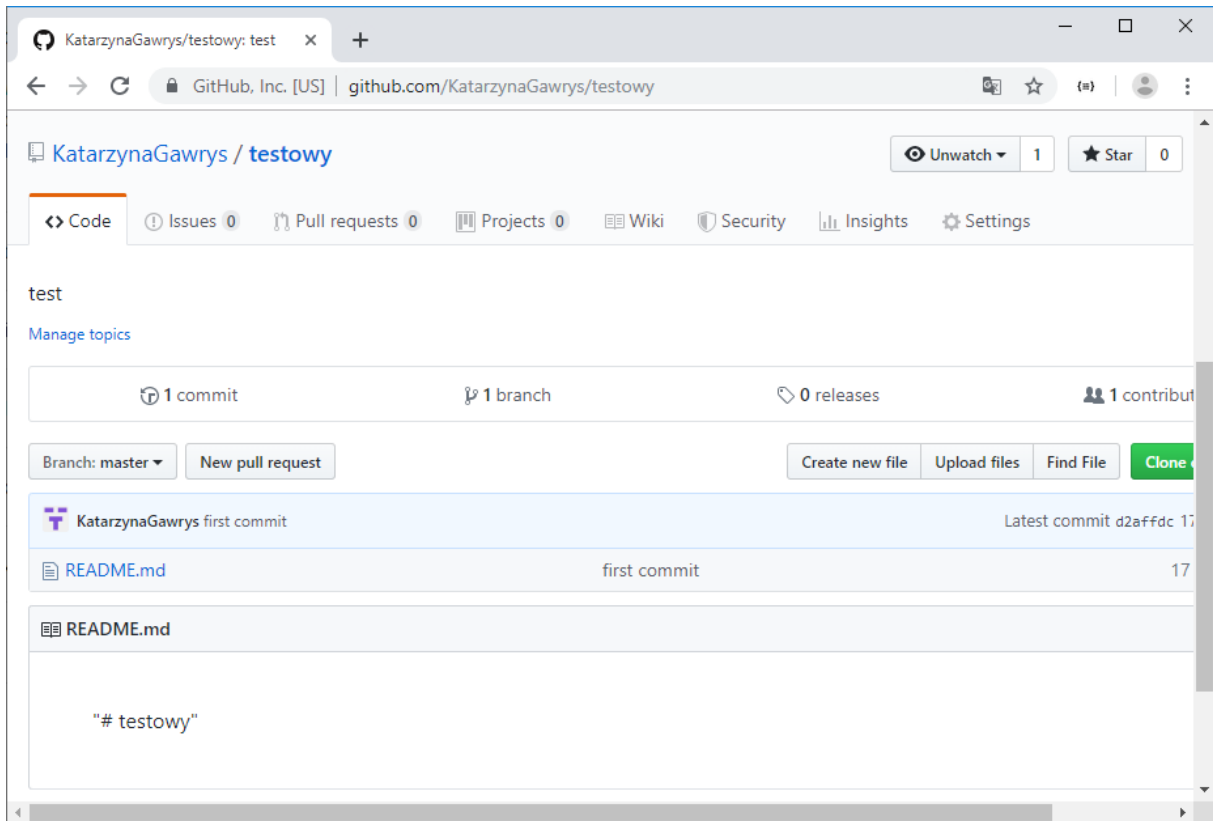
```
git remote add origin https://github.com/KatarzynaGawrys/testowy.git
```



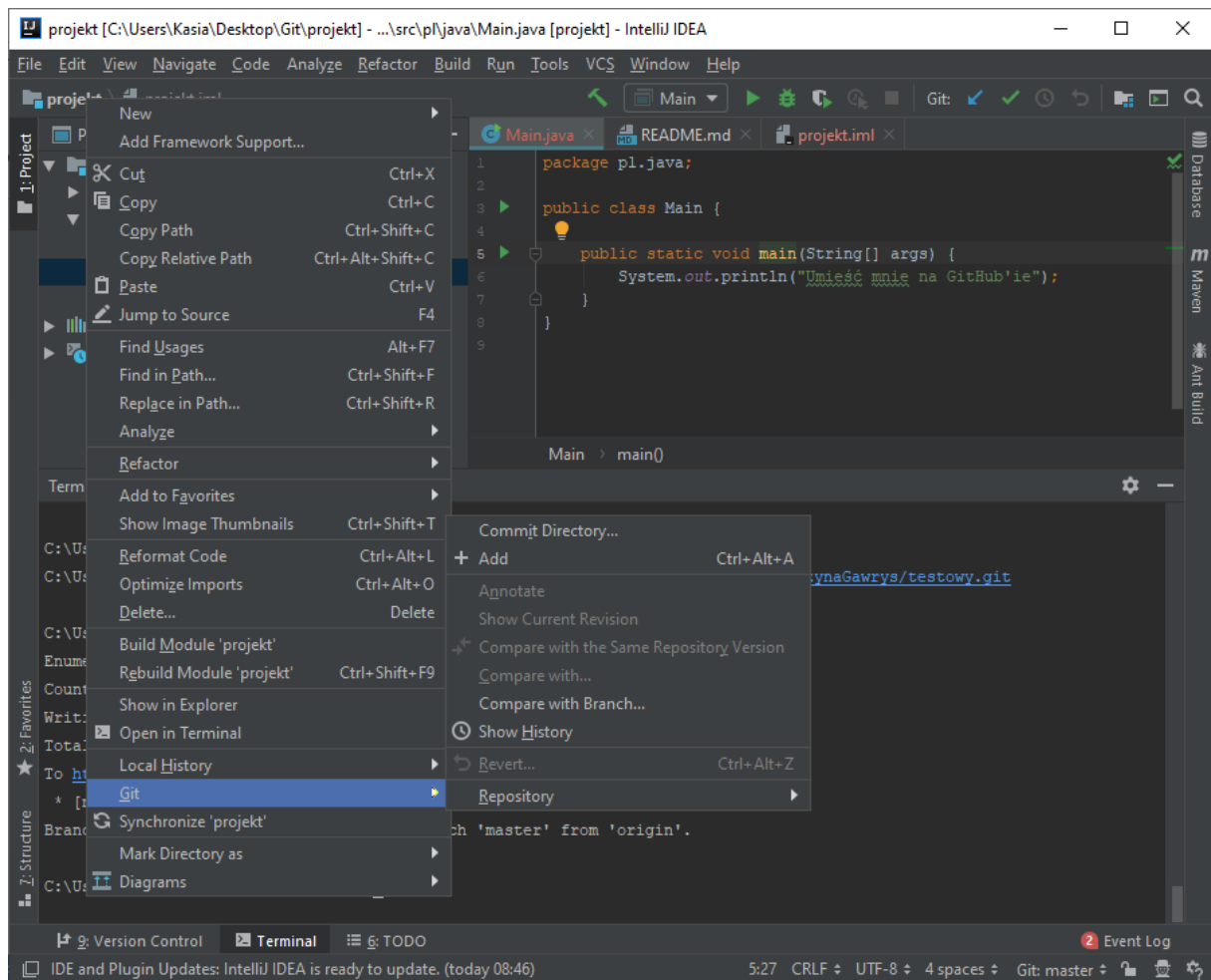
Ostania skopiowana komenda umieszcza - „wypycha” lokalną wersję na serwer:
`git push -u origin master`

Może się okazać, że musisz wpisać login i hasło (w zależności od tego jak skonfigurowano serwer)

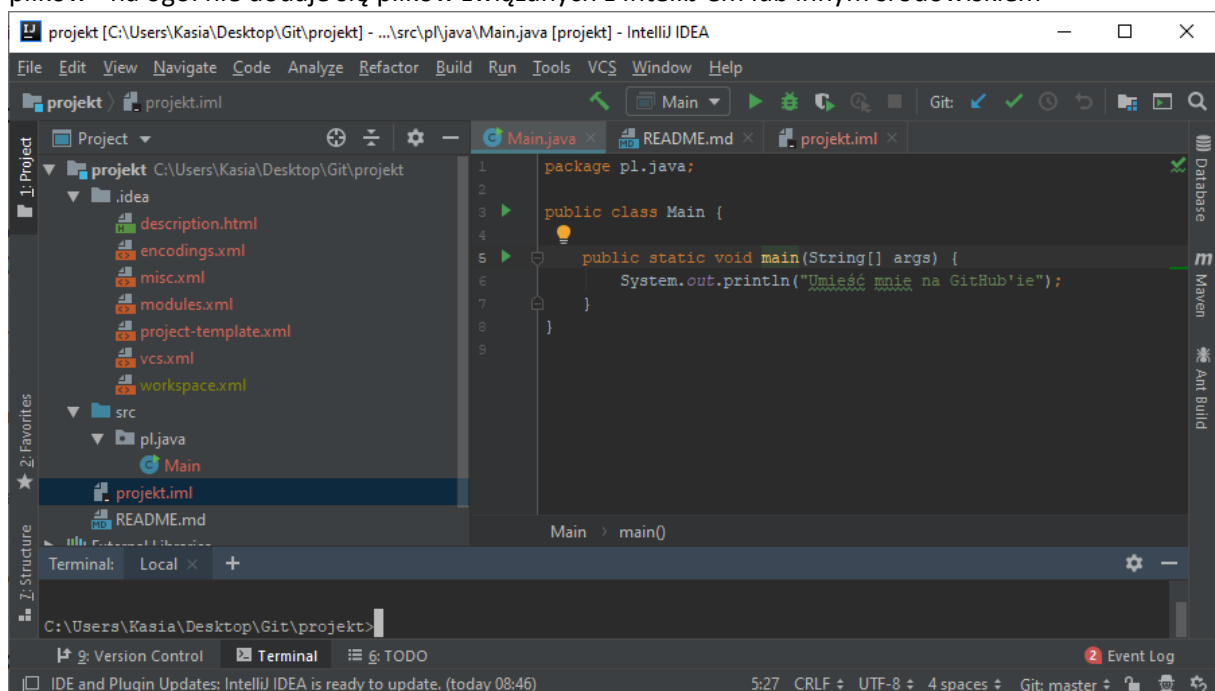
Przejrzyj projekt na GitHub’ie. Po odświeżeniu repozytorium testowy, widać że umieszczony w nim jest tylko plik README:



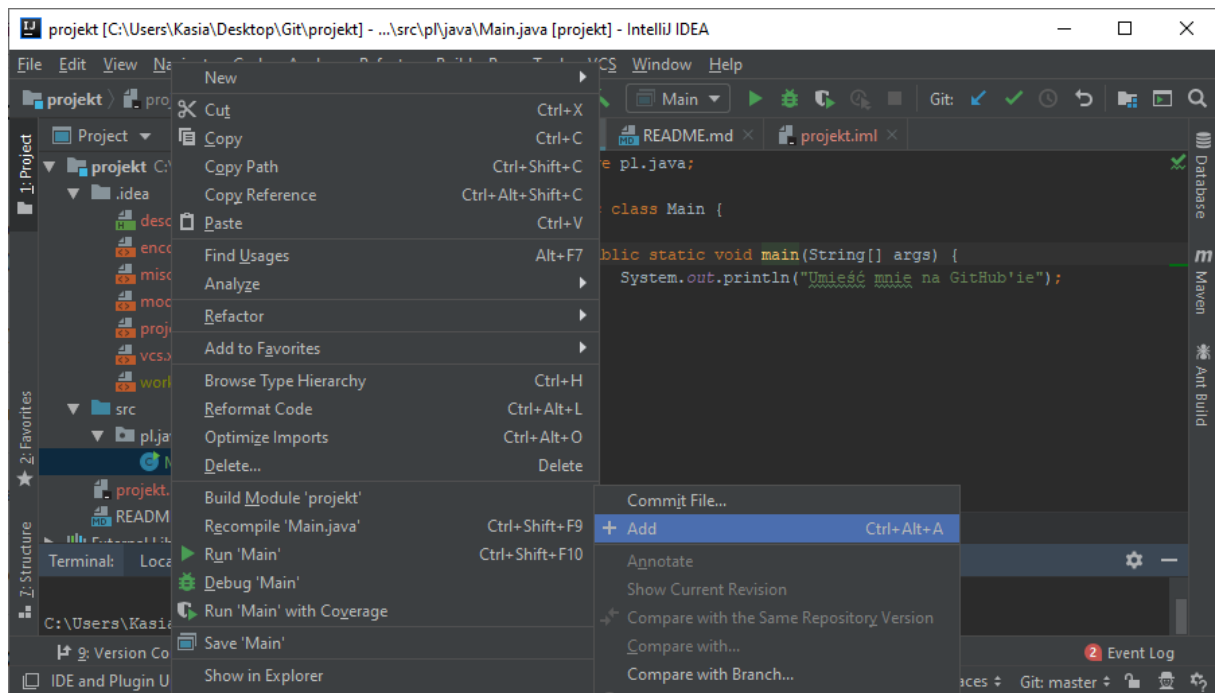
Dodaj pozostałe pliki. Przejdź do IntelliJ. Kliknij prawym przycisk myszy na nazwie projektu i z podmenu wybierz Git



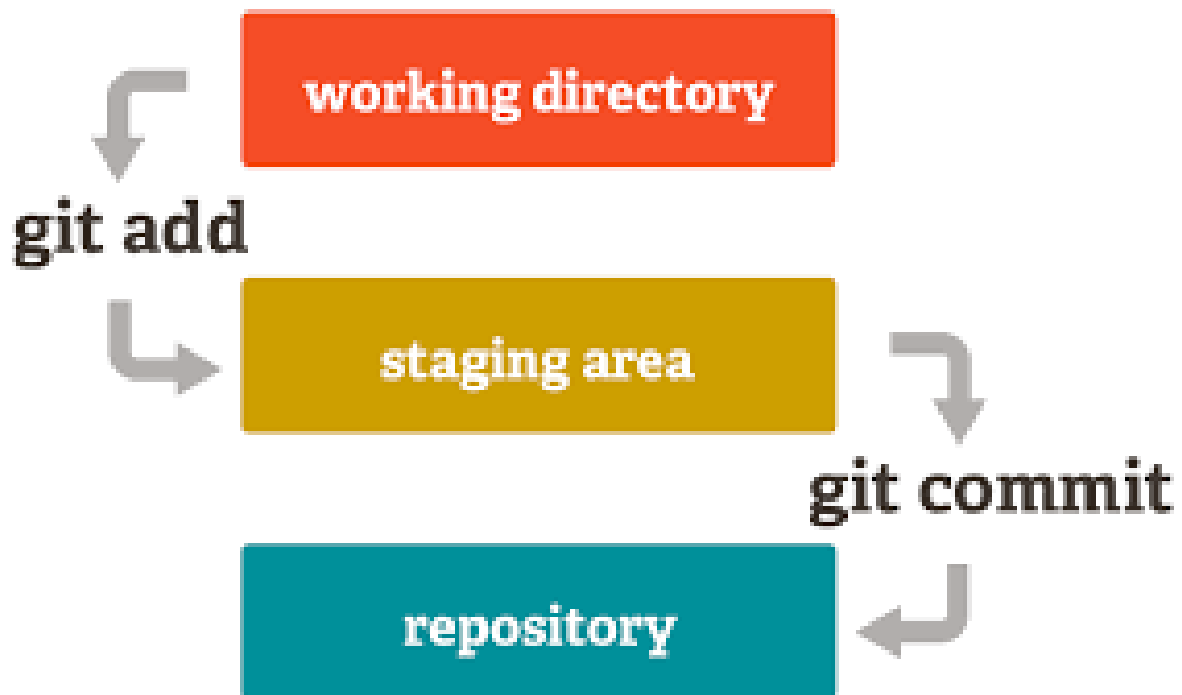
Pliki które są wyświetlone na czerwono trzeba dodać do repozytorium, nie dodajemy wszystkich plików - na ogół nie dodaje się plików związanych z IntelliJ'em lub innym środowiskiem



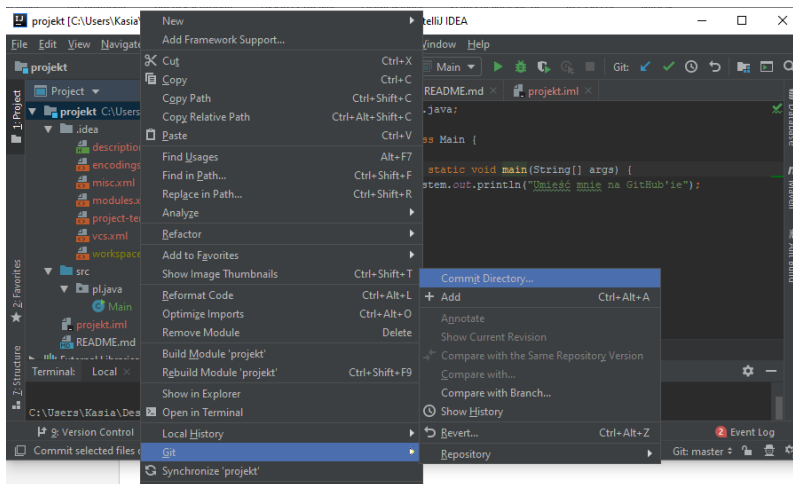
Zaznacz plik Main i go dodaj: prawy przycisk myszy i wybierz z podmenu add – Ctrl+Alt+A



Plik Main został dodany do obszaru staging area (przechowalnia), to obszar roboczych zmian ale nie zatwierdzonych i nie wypchniętych do repozytorium – jest jeszcze na zielono

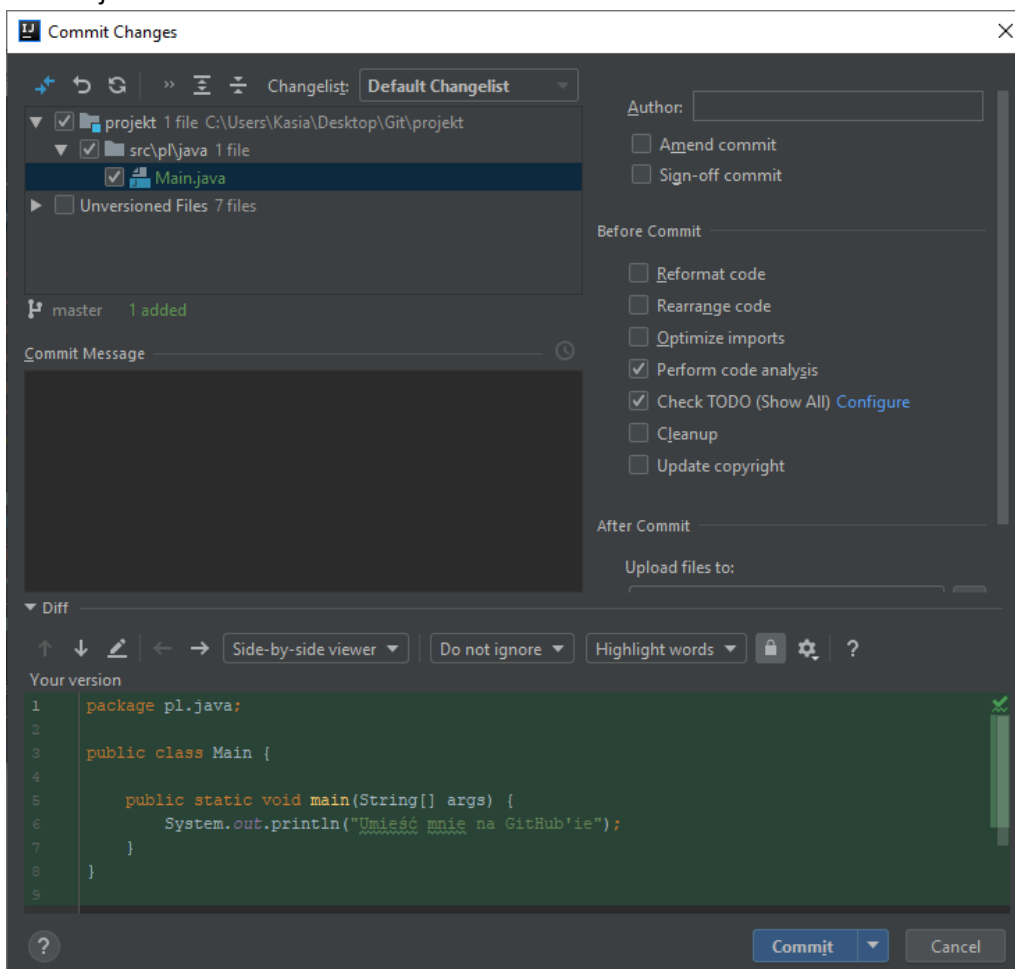


Kliknij prawym przyciskiem na nazwie projektu (w oknie z drzewem struktury projektu) i z podmenu wybierz Git i Commit Directory..

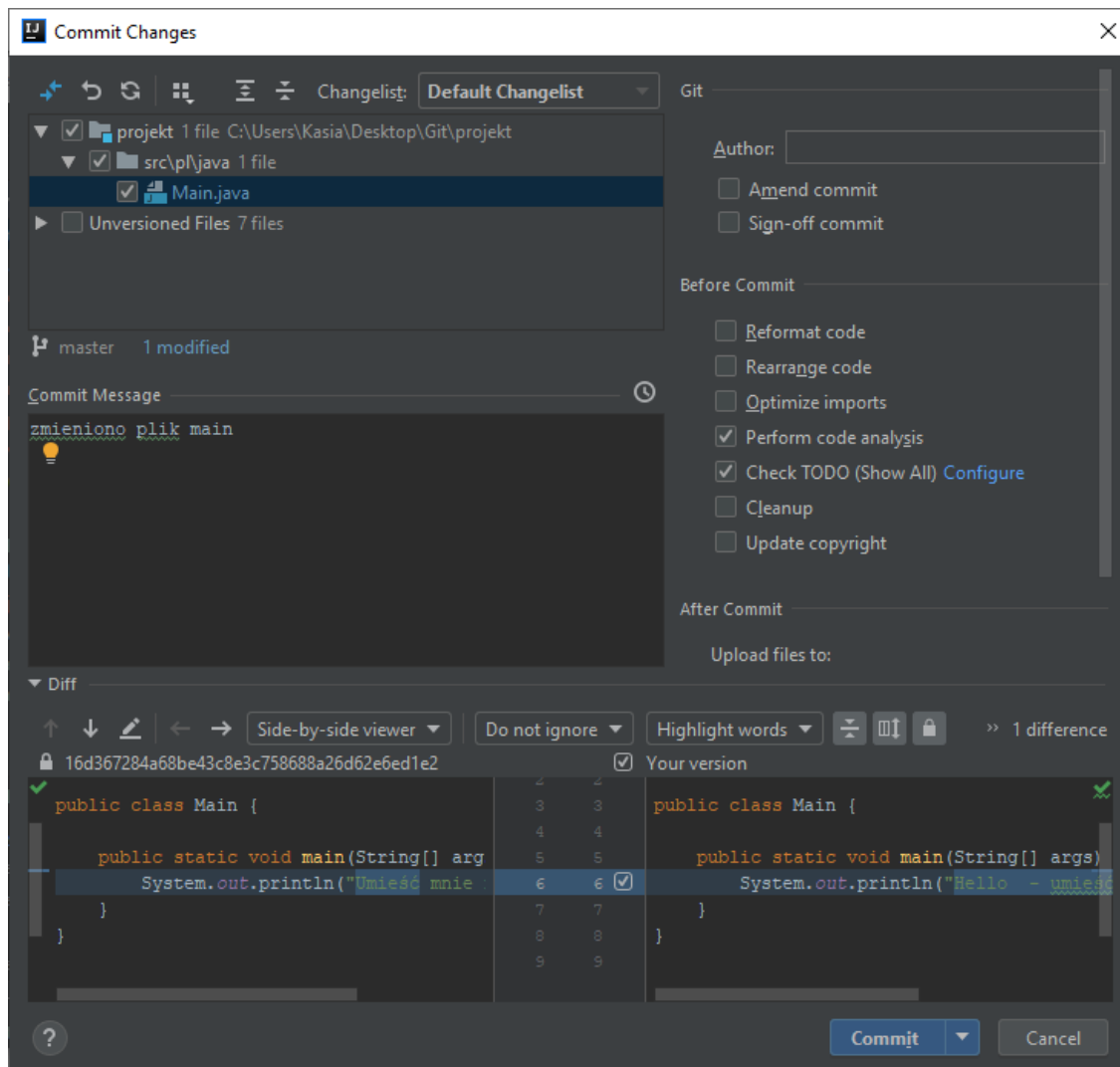


Pojawia się okno. Naciśnij Commit bez push. Plik Main jest na „zielono” bo jeszcze nie jest wypchnięty, ale jest w obszarze roboczych zmian. Pierwsza komenda, którą trzeba wykonać to commit, czyli zatwierdzenie zmiany. Dodaj opis (na ogół pisze się opisy commit'ów po angielsku):

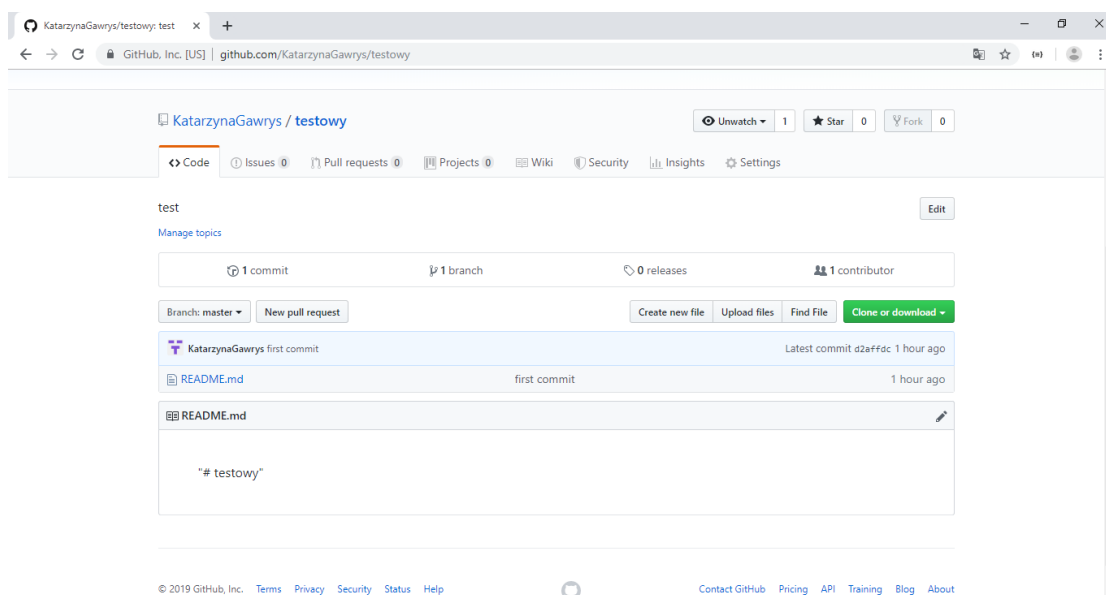
Naciśnij Commit:



Teraz zmodyfikuj plik Main, dodaj i opis commit: zmieniono plik main



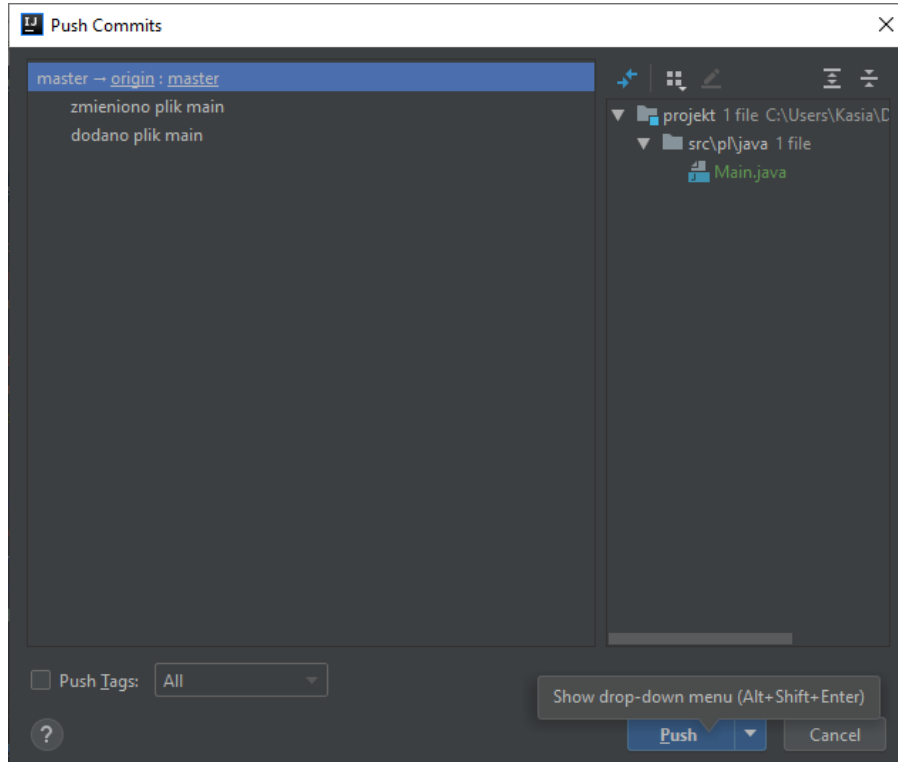
Sprawdź czy na GitHubie w repozytorium jest umieszczony plik Main:



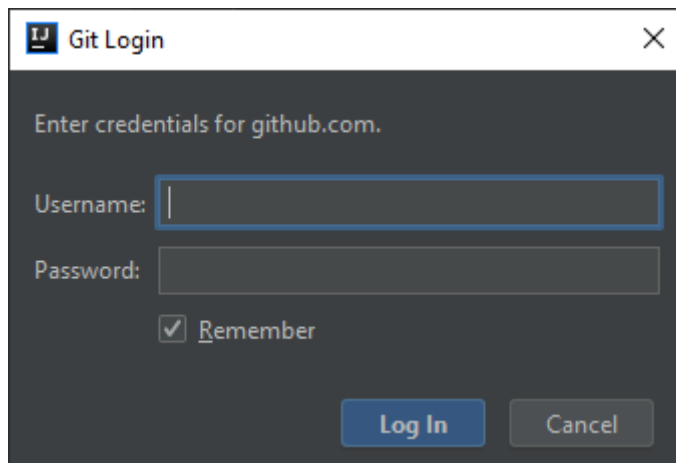
Pliku Main jeszcze nie ma w repozytorium.

Commit'y wykonujemy często na lokalnym repozytorium, dopiero gdy chcemy podzielić się kodem, to wypchamy zmiany czyli commit'y na serwer – Git/Repository/Push

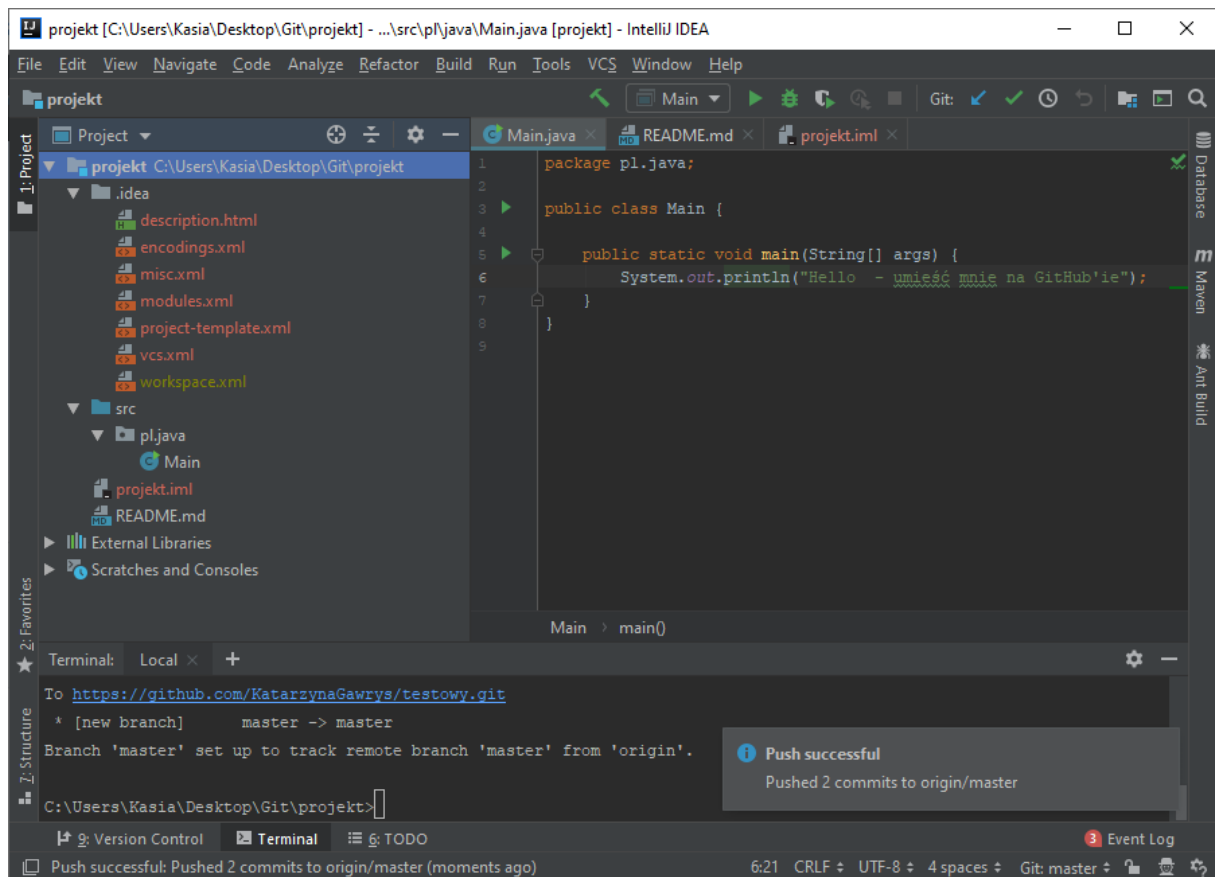
Zaznacz projekt prawym przyciskiem myszy i z podmenu wybierz push: git-repository-push:



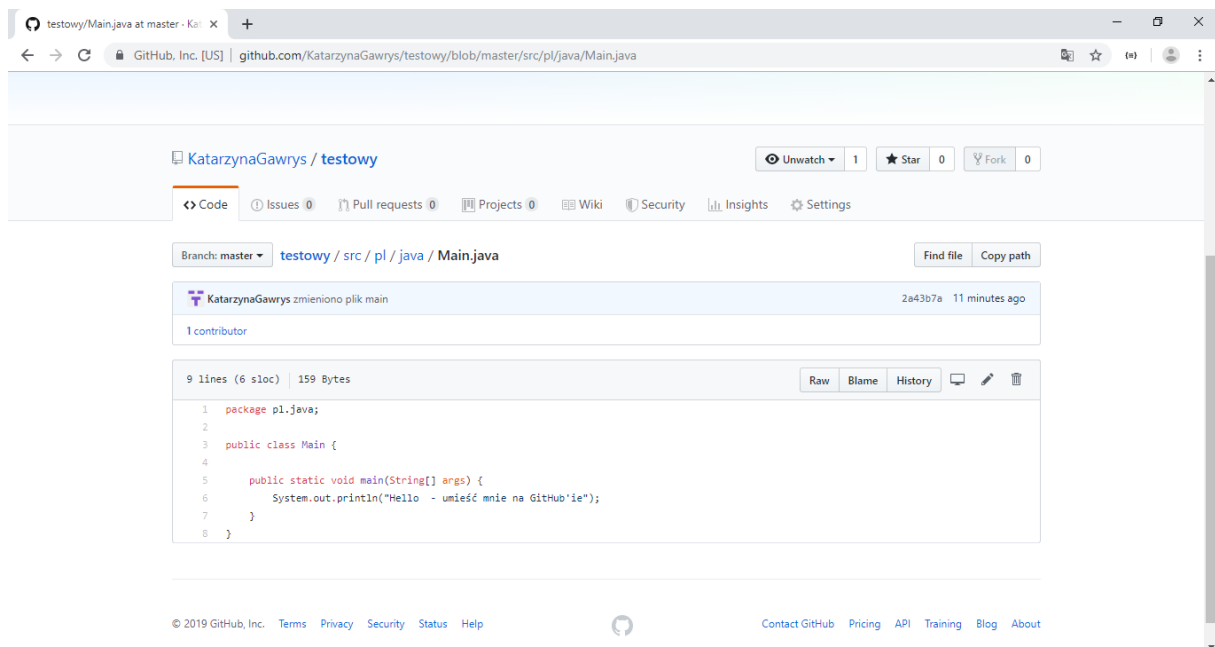
Pojawi się okno do logowania. Wpisz dane:



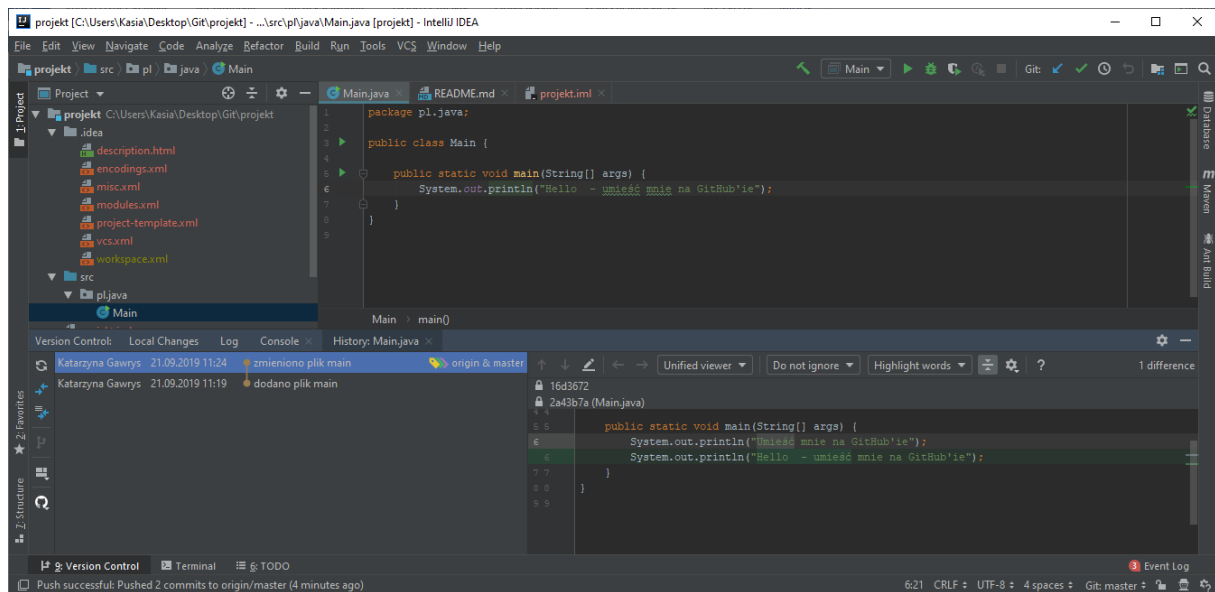
Pojawi się informacja ze wypchnięto dwa commit'y:



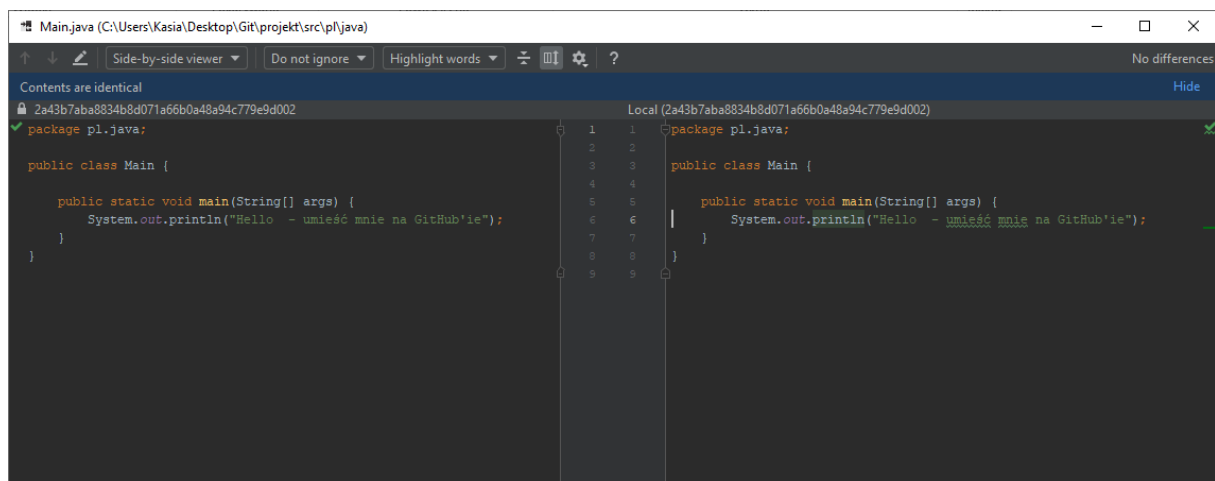
Na serwerze, pojawił się plik Main:



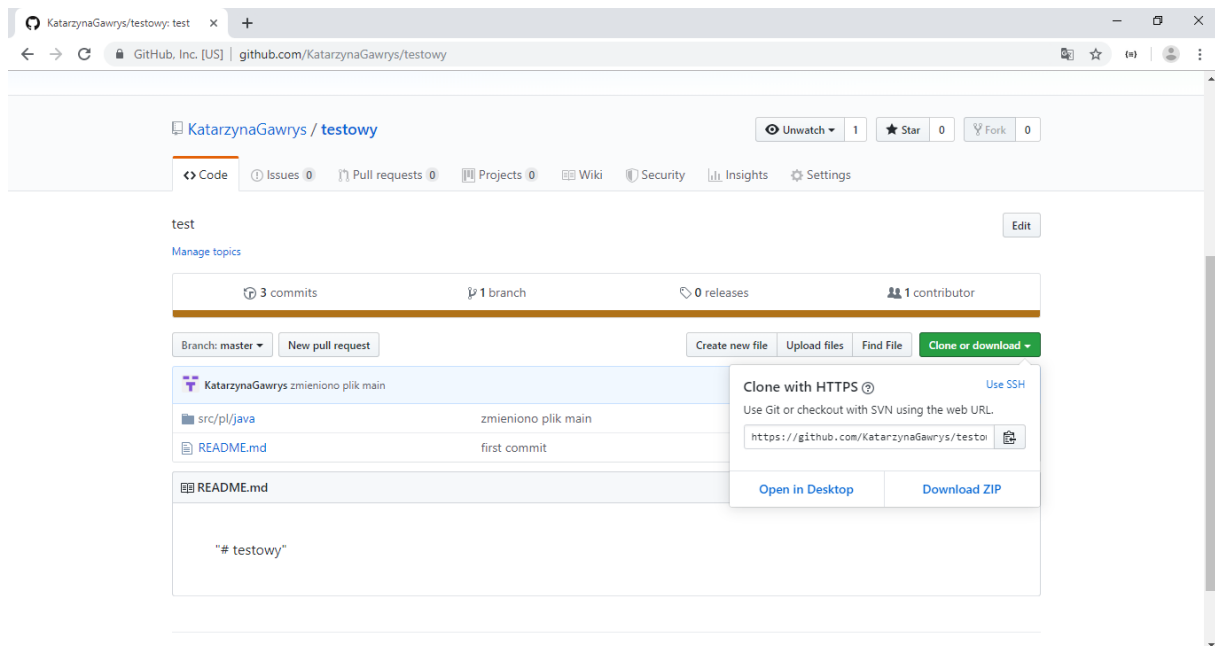
Zobacz historie pliku: w IntelliJ wybierz: Git- Show history:



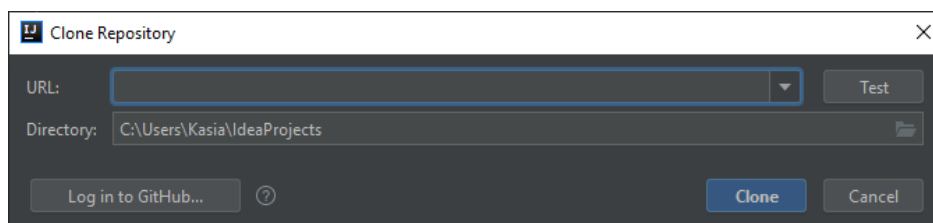
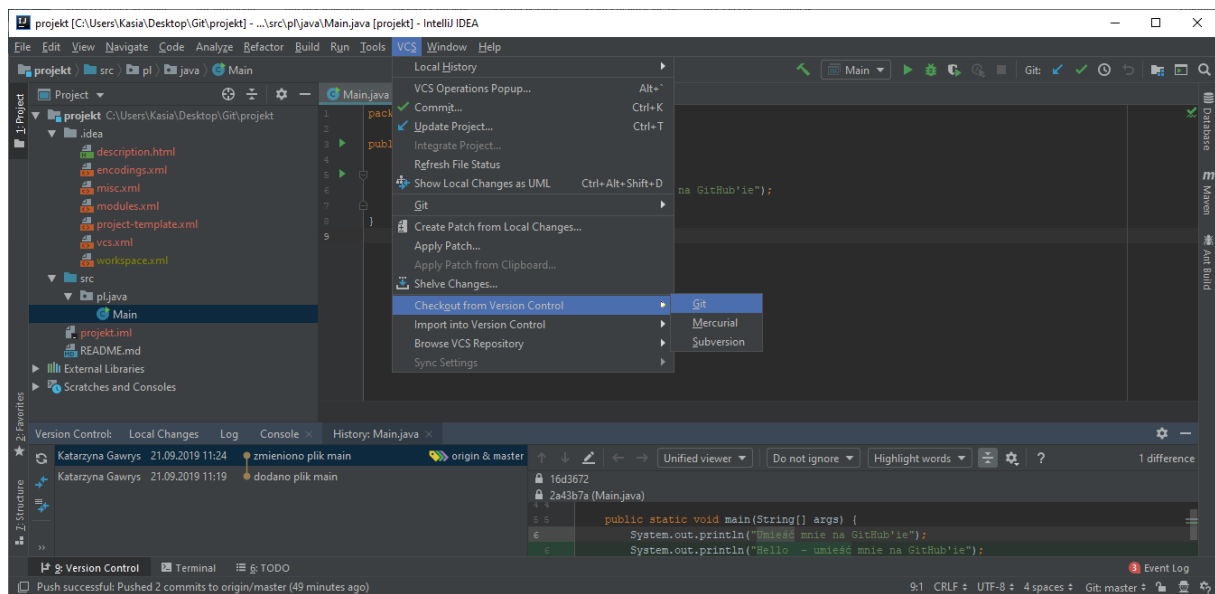
Lub Git- Compare



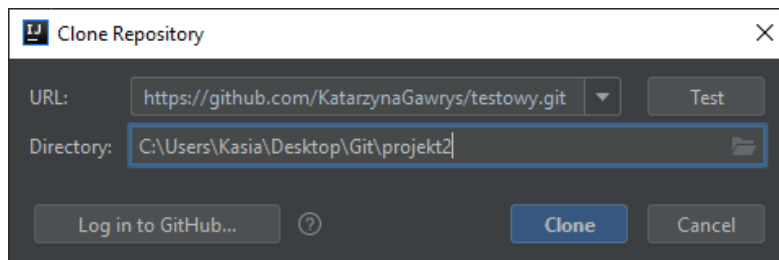
Korzystając z Git'a i commit'ów można wrócić do każdej zmiany. Commity wykonujemy na lokalnym repozytorium. W tym momencie kod jest umieszczony w zdalnym repozytorium i inni członkowie zespołu mogą z niego korzystać.



Wybierz Clone or download i skopiuj link. W IntelliJ wybierz z menu **VCS i Checkout from version control/ Git**



Wpisz adres repozytorium oraz podaj gdzie zapisać sklonowane repozytorium



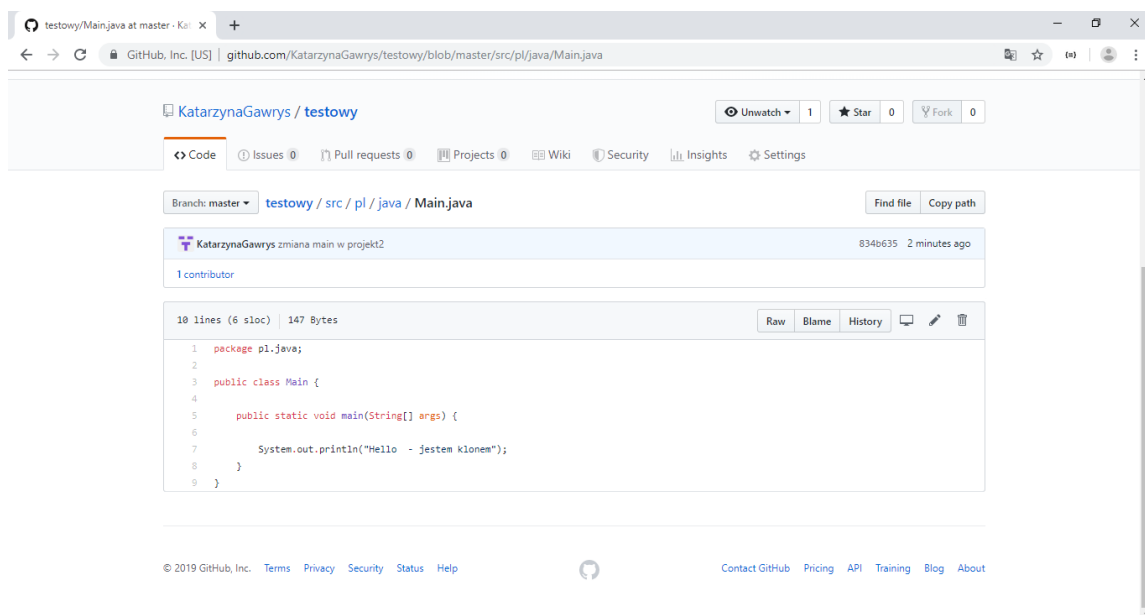
Po sklonowaniu otwórz projekt w nowym oknie w IntelliJ.

Zmodyfikuj w sklonowanym projekcie projekt2 plik Main.

Następnie wykonaj git commit – zmiana w projekcie 2.

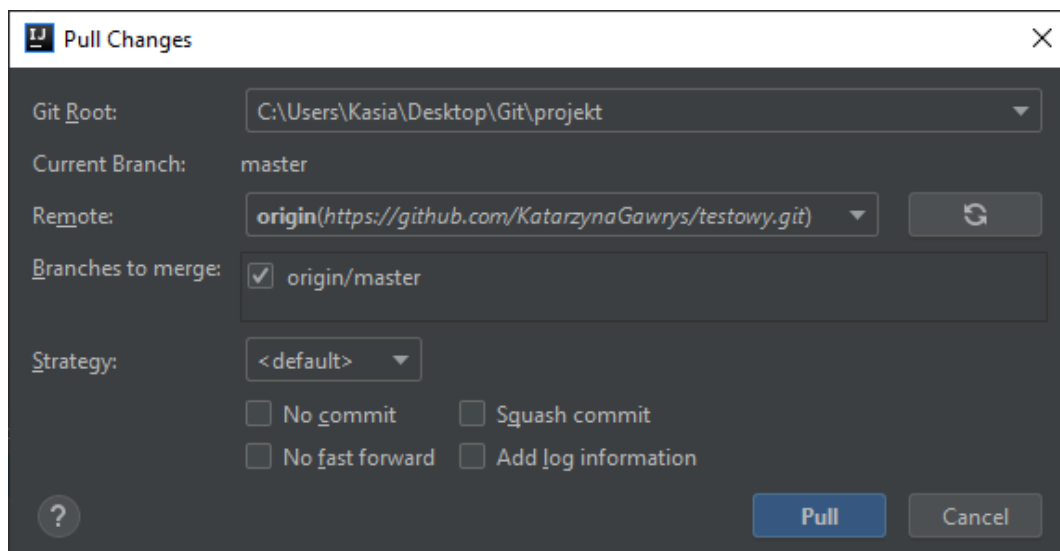
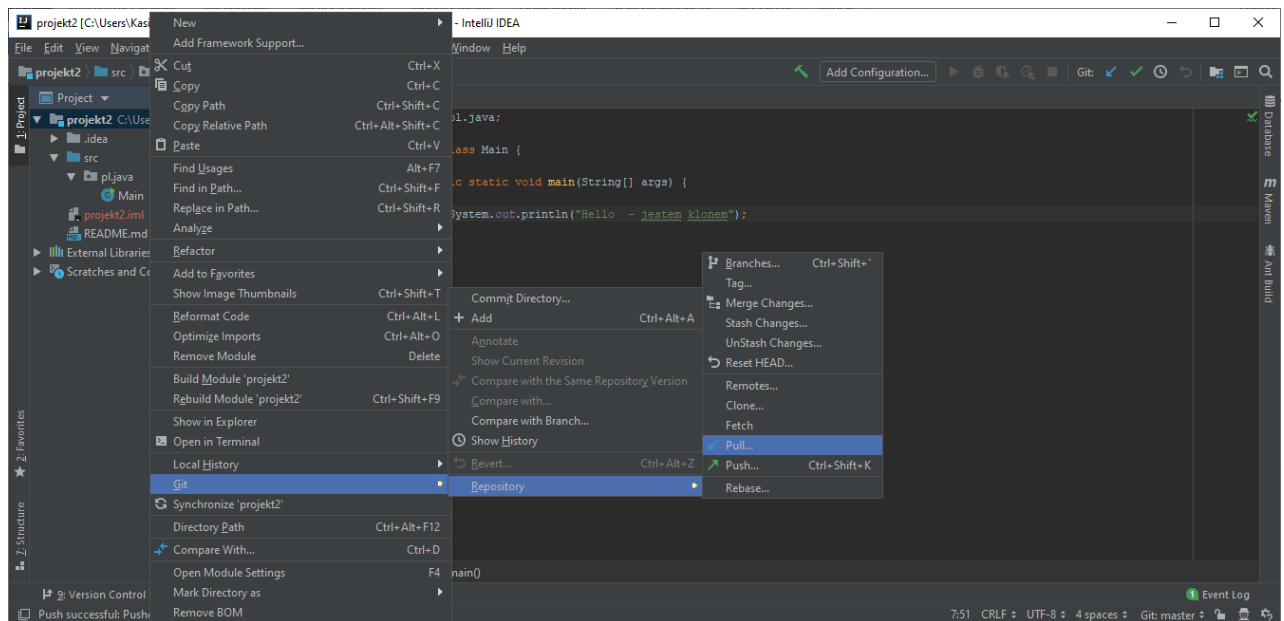
Commit and push – zatwierdź i wypchnij zmianę.

Na serwerze zdalnym pojawiła się zmiana wykonana w sklonowanym projekcie:



Przejdź do pierwszego okna w którym jest oryginalny projekt1 – ściągnij zmianę zrobioną w pliku Main w projekcie2

Wybierz Git/repository/pull



Sprawdź w oryginalnym projekcie czy zmiana została ściągnięta.

W ten sposób synchronizujemy zmiany z wszystkimi w zespole. I to jest podstawa korzystania z Git'a i z GitHub'a. Twój kod będzie widoczny w Internecie. Git i GitHub - do pracy w zespole projektowym, ale gdy pracujesz nad projektem sam, również korzystnie jest używać Git i GitHub, po to żeby zapamiętać swoje zmiany.