Poradnik dotyczący instalacji i użytkowania gita oraz githuba

### Różnice miedzy Gitem a GitHubem:

Git to **system kontroli wersji** (Version Control System), który działa lokalnie na Twoim komputerze. Jego główne cechy to:

1. **Zarządzanie wersjami kodu:**
   1. Git pozwala śledzić zmiany w plikach, wracać do wcześniejszych wersji oraz pracować nad różnymi gałęziami (branchami) projektu.
2. **Praca lokalna:**
   1. Git działa na Twoim komputerze. Nie wymaga internetu ani zdalnego repozytorium, aby zarządzać wersjami plików.
3. **Praca zespołowa:**
   1. Git umożliwia pracę wielu osób nad jednym projektem, łącząc zmiany i rozwiązując konflikty.
4. **Narzędzie wiersza poleceń:**
   1. Git działa głównie przez terminal (CMD, Bash, PowerShell). Nie posiada graficznego interfejsu użytkownika.
5. **Lokalne repozytoria:**
   1. Wszystkie zmiany w projekcie są przechowywane w lokalnym repozytorium, które działa niezależnie od chmury czy serwera.

GitHub to **platforma hostująca zdalne repozytoria Git**, która działa w chmurze. Jego główne cechy to:

1. **Zdalne repozytoria:**
   1. GitHub przechowuje kod w chmurze, co pozwala na łatwą współpracę i dostęp do projektu z dowolnego miejsca.
2. **Praca zespołowa:**
   1. GitHub oferuje narzędzia do współpracy, takie jak pull request
3. **Interfejs graficzny:**
   1. GitHub posiada intuicyjny interfejs graficzny dostępny w przeglądarce, który umożliwia zarządzanie projektami bez znajomości poleceń Git.
4. **Integracje i automatyzacja:**
   1. GitHub pozwala na integrację z narzędziami do ciągłej integracji (CI/CD), testowania, monitorowania i zarządzania projektami.
5. **Dostępność publiczna i prywatna:**
   1. GitHub umożliwia tworzenie zarówno publicznych repozytoriów (widocznych dla każdego), jak i prywatnych (dostępnych tylko dla wybranych osób).
6. **Historia zmian online:**
   1. GitHub przechowuje pełną historię zmian w projekcie i umożliwia przeglądanie commitów, branchów i tagów w przeglądarce.

### **Krok 1: Instalacja Gita**

1. Przejdź na stronę: <https://git-scm.com/downloads>.
2. Pobierz instalator odpowiedni dla Twojego systemu operacyjnego (Windows, macOS, Linux).
3. Uruchom pobrany plik instalacyjny i postępuj zgodnie z instrukcjami. Zalecane ustawienia domyślne będą wystarczające do większości użytkowników.

### **Krok 2: Utworzenie konta na GitHubie**

1. Przejdź na stronę: [https://github.com](https://github.com/).
2. Zarejestruj się na platformie, tworząc nowe konto.
3. Zaloguj się na swoje konto po zakończeniu rejestracji.

### **Krok 3: Konfiguracja Git na Twoim komputerze**

1. Po zainstalowaniu Gita, musisz skonfigurować swoje dane użytkownika, aby Git wiedział, kto wysyła zmiany do repozytoriów.
2. Otwórz **CMD** lub **Terminal**.
3. Wpisz poniższe polecenia, aby skonfigurować swoje dane użytkownika:

git config --global user.name "Twoje Imię"

git config --global user.email [twojemail@example.com](mailto:twojemail@example.com)

### **Krok 4: Tworzenie nowego repozytorium na GitHubie**

1. Zaloguj się na swoje konto GitHub.
2. Na stronie głównej kliknij na przycisk **New repository** (Nowe repozytorium).
3. Podaj nazwę repozytorium i wybierz, czy ma być publiczne, czy prywatne.
4. Kliknij **Create repository**.

### **Krok 5: Klonowanie repozytorium z GitHub na swój komputer**

1. Aby pracować z repozytorium na GitHubie, musisz je najpierw pobrać na swój komputer (tzw. klonowanie repozytorium).
2. Przejdź do swojego repozytorium na GitHubie.
3. Kliknij na zielony przycisk **Code** i skopiuj adres URL repozytorium.
4. Otwórz **CMD** lub **Terminal**.
5. Wybierz lokalizację na swoim komputerze, gdzie chcesz przechowywać repozytorium, np. cd C:\projekty\repo.
6. Wykonaj polecenie:

git clone https://github.com/TwojeImie/nazwa- repozytorium.git

### **Krok 6: Instalacja wtyczki Git oraz GitHub w Visual Studio Code**

1. Otwórz **Visual Studio Code**.
2. Przejdź do zakładki **Extensions** (Wtyczki) klikając na ikonę rozszerzeń po lewej stronie (lub użyj skrótu **Ctrl+Shift+X**).
3. Wyszukaj **Git** oraz **GitHubPullRequest** w polu wyszukiwania.
4. Zainstaluj wtyczki (często wtyczka Git jest zainstalowana domyślnie).
5. Po zainstalowaniu wtyczki, będziesz mógł używać Gita bezpośrednio w VSCode, a także zintegrowanego terminala, aby wykonywać operacje Git (np. commit, push, pull).

### **Jak działa wtyczka GitHub w praktyce?**

1. **Logowanie:**
   1. Po zainstalowaniu wtyczki GitHub (np. **GitHub Pull Requests and Issues**), VS Code poprosi o zalogowanie się do twojego konta GitHub.
   2. Logowanie odbywa się przez przeglądarkę, gdzie musisz potwierdzić dostęp dla Visual Studio Code.
2. **Praca z repozytorium:**
   1. Otwierając **folder repozytorium** w VS Code, wtyczka automatycznie rozpoznaje, że jest to projekt z Gitem.
   2. W zakładce **Source Control** znajdziesz listę zmian, które możesz dodawać, zatwierdzać (commitować) i wysyłać (pushować) na GitHuba.
3. **Tworzenie pull requestów:**
   1. Po wprowadzeniu zmian możesz utworzyć pull request bezpośrednio w VS Code.(pamietaj o dodaniu komentarza w polu message)
   2. Wtyczka pozwala przeglądać komentarze innych użytkowników i wprowadzać poprawki.
4. **Podgląd kodu i historii:**
   1. W zakładce **Source Control** możesz zobaczyć różnice między wersjami plików.
   2. Wtyczka pokazuje także historię commitów dla danego pliku.

**Wyjasnienia pojęć:**

#### **1. Commit (zatwierdzenie zmian):**

* **Czym jest?**
  + Commit to operacja, która zapisuje zmiany w repozytorium lokalnym (na twoim komputerze).
  + Każdy commit jest jak „moment w czasie”, który dokumentuje, jakie zmiany zostały wprowadzone w plikach.
* **Kiedy używać?**
  + Po wykonaniu zmian w kodzie lub plikach projektu, które chcesz zapisać.
* **Przykład użycia:**

git add nazwa\_pliku  
git commit -m "Opis zmian, np. Dodano funkcję logowania"

Lub w **VSCode**, możesz użyć opcji **Stage Changes** i **Commit** w zakładce **Source Control**.

#### **2. Push (wysyłanie zmian):**

* **Czym jest?**
  + Push przesyła zatwierdzone zmiany (commity) z twojego lokalnego repozytorium na zdalne repozytorium, np. na GitHub.
  + Dzięki temu inni użytkownicy mogą zobaczyć twoje zmiany w repozytorium online.
* **Kiedy używać?**
  + Po wykonaniu commitów, gdy chcesz zsynchronizować swoje zmiany z zdalnym repozytorium.
* **Przykład użycia:**

git push origin main

Gdzie main to nazwa głównej gałęzi (branch) twojego repozytorium.

#### **3. Pull (pobieranie zmian):**

* **Czym jest?**
  + Pull pobiera najnowsze zmiany ze zdalnego repozytorium na twój komputer.
  + Działa jak „aktualizacja”, synchronizując twoją lokalną kopię z najnowszą wersją na serwerze.
* **Kiedy używać?**
  + Zanim zaczniesz pracę nad nowymi zmianami, aby upewnić się, że masz najnowszą wersję kodu.
  + Jeśli ktoś inny pracował nad projektem i wprowadził zmiany, które są już w repozytorium zdalnym.
* **Przykład użycia:**

git pull origin main

### **Zadanie:** -wykonaj dowolny kod, który prześlesz do swojego repozytorium na githubie.