**GIT Zadania**

**Zad. 1**

Wymień przykłady serwisów/aplikacji, które mogą pełnić rolę GIT-owych repozytoriów.

**Zad. 2**

Poniższe zadanie składać się będzie z wielu podpunktów i kroków do realizacji. Wszystko przeprowadzać będziemy na jednym repozytorium.

1. Stwórz dowolne repozytorium na GH. Dodaj do niego plik README o dowolnej zawartości oraz .gitignore, w którym zignorujesz wybrane pliku z projektu.
2. Do swojego repozytorium dodaj 3 dowolne commity.
3. Wyświetl historię zmian.
4. Sklonuj tak udostępnione repozytorium w inne miejsce na swoim komputerze (do innego katalogu). Oznaczymy go jako **katalog nr 2**.
5. Wejdź do takiego katalogu, zmień jego zawartość w dowolny sposób i wypuszuj do tego samego repozytorium.
6. Następnie wróć do **katalogu nr 1** i spróbuj wypuszować dowolną zmianę. Czy upload plików będzie możliwy? Jaki błąd wówczas nastąpi? Jak go rozwiązać?
7. Po wypuszowaniu zmian, sprawdź historię commitów jak i wszystkich zmian przeprowadzonych w repozytorium.

**Zaawansowane:**

1. Zmodyfikuj dowolny plik, ale nie commituj zmian.
2. Utwórz nowy branch w projekcie, nazwij go dev-02 i przenieś swoją pracę właśnie na tak utworzoną gałąź (wraz z wcześniej przeprowadzoną zmianą)
3. Zacommituj zmiany i wypushuj je do swojego repozytorium na branch dev-02
4. Wystaw pull request i poproś mentora o Code Review (niech nie zatwierdza zmian)
5. Zmień nazwę poprzedniego commita (nadpisz commit) na zapis: FEAT-1234 | Example commit i wypushuj zmiany. Po takiej operacji mentor może już przeprowadzić merge projektu.

**Zad. 3**

Do tej pory mieliśmy do czynienia z wystawieniem merge requestów między branchem feature-owym a masterem (mainem). Scalania możemy jednak dokonywać nie tylko z gałęzią główną projektu, ale również na poziomie dwóch wybranych branchów. Umożliwiają to dwa polecenie: git merge oraz git rebase.

Zapoznaj się z każdym z nich i odpowiedz na pytanie:

* Czym różnię się rebasowanie od mergowania?
* Kiedy używać git rebase a kiedy git merge?

Dodatkowo:

* Utwórz dwa dowolne branche z różniącą się historią commitów oraz spróbuj je ze sobą scalić. Rozwiąż ewentualne merge conflicty.

**Zad. 4**

Czym się różni GIT Flow od GitHub Flow? Na czym polega każde z podejść?

**Zad. 5**

Dobrą i niezwykle częstą praktyką podczas pracy z repozytorium jest podpinanie pod projekt, tzw. systemu CI (Continuous Integration). Zapoznaj się wstępnie, z czym związane jest to pojęciem i jakie korzyści daje ono w pracy zespołowej.

**Zad. 6**

Utwórz na GH prywatne repozytorium, które służyć nam będzie do przechowywania rozwiązań do dalszych zadań (już bardziej programistycznych) w ramach mentoringu Devs-Mentoring.pl. Następnie udostępnij tak utworzone repo swojemu mentorowi (musisz dodać go do zakładki Contributors w obrębie ustawień projektu).