**Kurs .NET Tomasz Bursztyński**

**GIT**

GIT to narzedzie stworzone do wygodnego przetrzymywania kodu źródłowego aplikacji. Umożliwia on nam tworzenie tzw. repozytorium w którym trzymamy cały kod. GIT umożliwia nam na zapisywanie etap po etapie wszystkich zmian w naszej aplikacji, dzięki czemu nie tylko mamy historię edycji, ale zmuszeni jesteśmy opisywać krótkimi wiadomościami co dana edycja wniosła nowego do projektu, kto oraz kiedy ją zrobił. GIT to tylko narzędzie, które instaluje się lokalnie na komputerze. Repozytorium zaś możemy trzymać tylko lokalnie, ale możemy też w chmurze.

GIT Hub to serwis online w którym po założenia konta możemy w chmurze przetrzymywać Nasze repozytoria. Wygodny jest do dzielenia się kodem z innymi współpracownikami / kursantami / rekruterami.

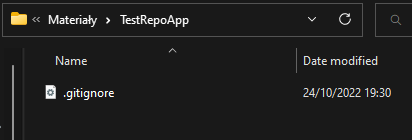
GIT posiada w sobie dwa kluczowe słowa: commit ora branch

Commit to jedna zmiana jaka robimy w aplikacji i opatrzyć trzeba ją krótkim komentarzem. Każdy commit wie co było wcześniej przed zmianami, co jest po zmianach, kto i kiedy go utworzył.

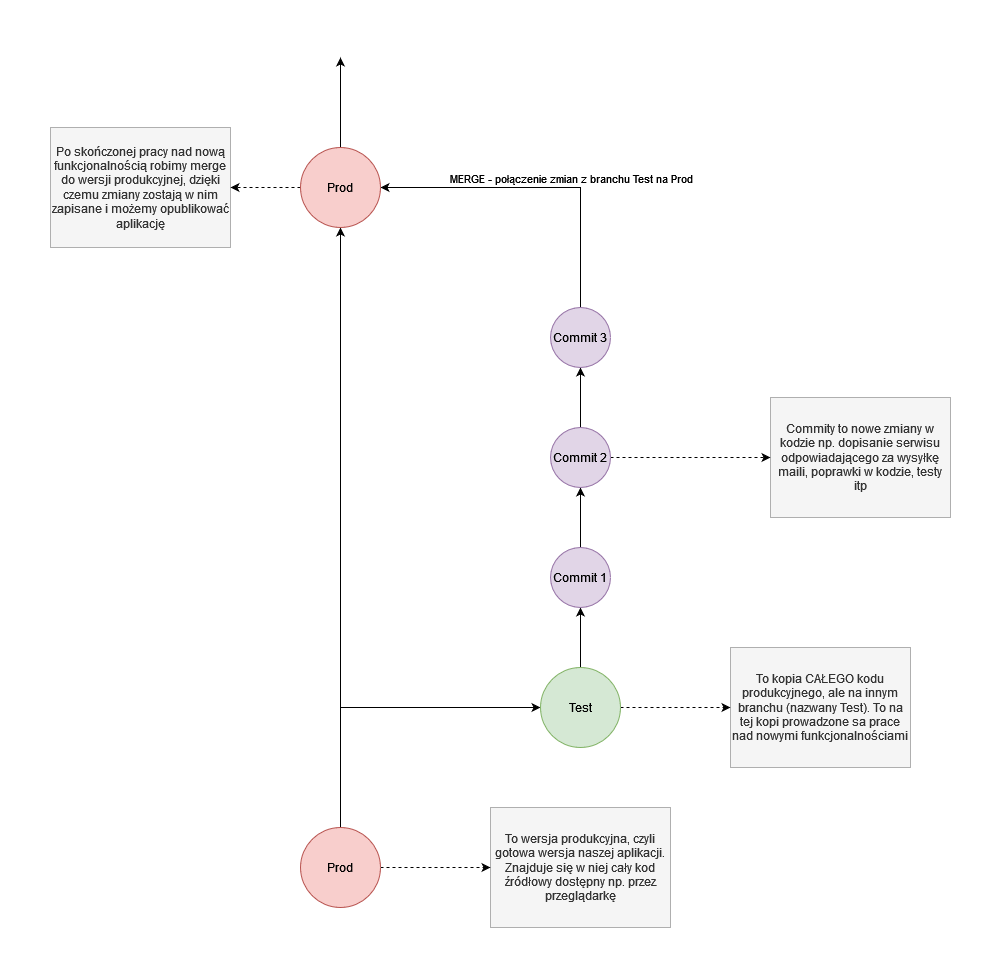
Branch – to wersja naszej aplikacji. Wyobraź sobie że tworzymy dla Naszej aplikacji dwa oddzielne branche – produkcyjny oraz testowy. Na testowym tworzymy nową funkcjonalność np. wysyłkę maili. Nie chcemy, by ten nowy feature pojawił się na wersji produkcyjnej póki nie zostanie on przetestowany. Dzięki temu nowe commity z nową funkcjonalnością puszczami na branch testowy, zaś kiedy dojdziemy do wniosku, że jest on już gotowy, możemy przeprowadzić ‘merge’.

Merge to po prostu połączenie jednego brancha z drugim np. wersji produkcyjnej z testowym.

Gitignore – to plik konfiguracyjny nazwany typu .gitgnore i nie ma swojej nazwy, w folderze z aplikacją dosłownie leży jako „.gitignore”. Narzedzie dotnet umożliwia nam na wygenerowanie domyślnego pliku gitignore który zawiera skróty informujące jakich folderów i plików nasze repozytorium ma unikać. Po co? Otóż w repozytorium chcemy trzymać TYLKO kod źródłowy Naszej aplikacji. NIGDY nie trzymamy w repozytorium skompilowanej aplikacji czyli zawartości folderu „bin” lub „obj”.

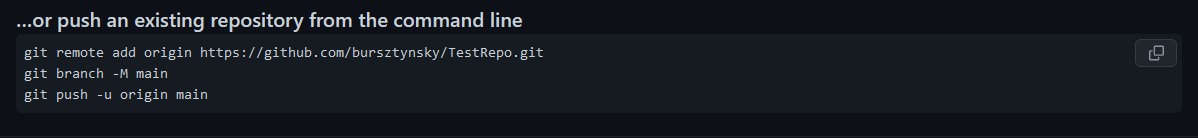


Rysunek Plik konfiguracyjny .gitignore



Rysunek Opis jak wygląda tworzenie branchu Test na podstawie Prod oraz na koniec merge

**ZADANIE DOMOWE:**

1. Zainstaluj GIT - <https://git-scm.com/downloads>. Podczas instalacji możesz wszystko przeklikać jako „next”, nie przejmuj się dodatkową konfiguracją, nam wystarcza domyślna.
2. Załóż konto na GitHub <https://github.com/> i się zaloguj
3. Kliknij w prawym górnym rogu na ikonkę swojego konta i kliknij „Your repositories”
4. Kliknij „New”
5. Nazwij repozytorium „TestRepoApp” w pozycji „Repository name”
6. Zaznacz jako „Public”
7. Kliknij „Create Repository”
8. Zrób na pulpicie folder o nazwie „TestRepoApp”
9. W środku folderu otwórz konsole aplikacji (cmd.exe) i za pomocą komendy „dotnet new console” stwórz aplikacje konsolową
10. Korzystając z komendy „dotnew new gitignore” stwórz plik .gitignore
11. Za pomocą komendy „git init” stwórz repozytorium w folderze z aplikacją
12. Za pomocą komendy „git status” wyświetl status naszego repo. Pokaże on nam pliki źródłowe Naszej aplikacji oznaczone na czerwono – to znaczy, że są to nowe pliki i można je dodać do repo. Możemy je dodawać po kolei albo korzystając z komendy poniżej dodać od razu wszystko
13. Za pomocą komendy „git add .” dodaj wszystkie pliki jakie znajdują się w naszej aplikacji
14. Za pomocą komendy „git commit -m” stwórz pierwszy commit. UWAGA – po słowie „-m” musisz dać spację oraz w cudzysłów wpisać opis co dodajemy. Możesz napisać to tak: git commit -m „First commit to create a repo”
15. Wróć do swojego repozytorium na Github i skopiuj tą część do notatnika
    1. Komenda remote add łączy Nasze lokalne repo z tym w github
    2. Komenda branch -M main tworzy główny branch o nazwie main w którym trzymamy „produkcyjną” wersję aplikacji. Nazwa main może być zmieniona na master, osobiście polecam z takiej korzystać 😊
    3. Komenda git push wysyła wszystko do Naszego zdalnego repo
16. Wpisz do konsoli po kolei wszystkie trzy komendy i odpal je jedna po drugiej
17. Po komendzie git push przejdź do github, odśwież stronę i sprawdź czy pojawiła się aplikacja