z7. Prywatna chmura. System kontroli wersji GIT **v2018-11-18**

To ćwiczenie laboratoryjne może okazać się przydatne dla osób realizujących w ramach pracy inżynierskiej aplikację, której rozwój, testowanie i optymalizację trzeba poprawnie udokumentować oraz profesjonalnie przebadać tę aplikację pod różnymi kątami, aby móc ją następnie zoptymalizować, zgodnie ze współczesnymi technologiami i trendami.

Projektowanie, programowanie, testowanie oraz dokumentowanie aplikacji sieciowej to złożony proces, nawet jeśli jest wykonywane tylko przez jedną osobę, jak to ma zwykle miejsce w przypadku pracy inżynierskiej. Stopień trudności tego zadania rośnie wykładniczo z każdym kolejnym programistą, który dochodzi do teamu realizującego aplikację. Dlatego od wielu lat wykorzystywane są przy tworzeniu aplikacji sieciowych (i nie tylko) systemy kontroli wersji (version/revision control system), czyli programy ułatwiające śledzenie zmian wprowadzanych w kodzie źródłowym i łączenie zmian realizowanych przez wiele osób na różnych etapach realizacji danego projektu.

https://pl.wikipedia.org/wiki/System_kontroli_wersji

Spośród wielu systemów kontroli wersji wyróżnia się projekt Git, stworzony przez Linusa Torvaldsa (twórcy Linuxa) https://pl.wikipedia.org/wiki/Git_(oprogramowanie). Git jest rozproszonym systemem kontroli wersji. To rozproszenie polega na tym, że na każdym komputerze pracującym w projekcie przechowywany jest cały projekt i jego zmiany – tzw. repozytorium lokalne. System ten wyróżnia się większą niezawodnością, ponieważ nie ma głównego repozytorium przechowującego cały projekt, którego awaria mogłaby spowodować jego utratę. Git pełni rolę repozytorium zewnętrznego (remote), które pośredniczy między wszystkimi połączonymi maszynami i przechowuje cały projekt. Wyłączenie tego zewnętrznego repozytorium nie spowoduje utraty zapisanych zmian.

Popularność Gita wynika m.in. z jego relatywnie dużej szybkości i niezawodności w porównaniu z konkurencyjnymi projektami, a także prowadzenia z jego pomocą bardzo ważnych lub/i dużych projektów m.in. jądra Linuxa, ponadto Git jest systemem darmowym. Osoby zainteresowane zainstalowaniem Gita na swoim komputerze/serwerze mogą skorzystać z jednego z wielu portali podpowiadających jak go skonfigurować krok-po-kroku np.

https://git-scm.com/book/pl/v1/Pierwsze-kroki-Wprowadzenie-do-kontroli-wersji

Więcej informacji o projekcie Git znajduje się m.in. na stronie:

https://pl.wikibooks.org/wiki/Git

https://pl.wikibooks.org/wiki/Git/Podstawy

Aby wykorzystywać projekt Git nie trzeba instalować oprogramowania na własnym komputerze/serwerze, ponieważ istnieje wiele serwisów hostingowych umożliwiających bezpłatne lub płatne utrzymywanie repozytoriów Gita np. https://github.com/ i wiele innych, których cechy można przejrzeć i porównać korzystając m.in. z poniższych linków:

https://git.wiki.kernel.org/index.php/GitHosting

https://en.wikipedia.org/wiki/Comparison_of_source_code_hosting_facilities

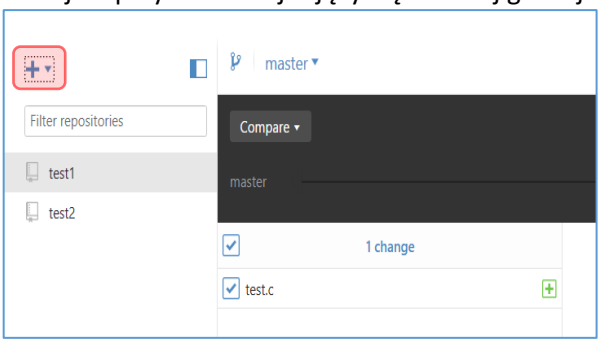
https://www.git-tower.com/blog/git-hosting-services-compared/

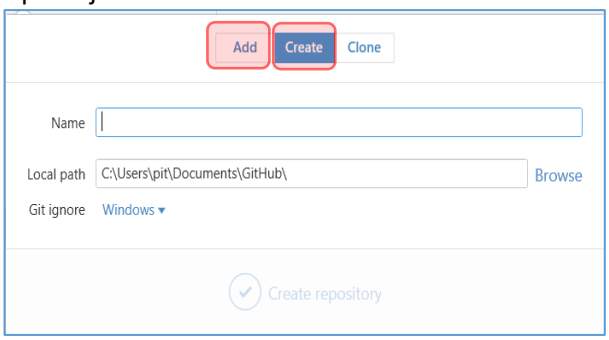
Korzystając z wymienionych aplikacji można utrzymywać kod rozwijany prywatnie lub udostępniany publicznie. Prywatne repozytoria kodu służą firmom rozwijającym aplikacje np. wspólnie ze współpracującymi programistami z zewnątrz, natomiast dzięki publicznym repozytoriom wielu pracodawców poszukujących programistów do realizacji konkretnych projektów znajduje odpowiednich kandydatów lub weryfikuje możliwości potencjalnych pracowników, prosząc ich o podanie konta na Githubie, aby się przyjrzeć próbkom ich kunsztu programistycznego. Portal github.com jest zatem wykorzystywany zarówno przez wielkich developerów, jak i m.in. tzw. wolnych strzelców.

1. Wejdź na portal https://github.com/
2. Utwórz swój prywatny projekt https://github.com/personal (jeśli posiadasz swoje konto na Githubie pomiń ten punkt i przejdź od razu do realizacji aplikacji, przy czym wykorzystuj Githuba przy jej tworzeniu).
3. Utwórz dla siebie bezpłatne konto, umożliwiające Ci dzielenie się swoim kodem ze światem lub przynajmniej potencjalnymi pracodawcami https://github.com/join
    a. Utwórz nazwę swojego konta (Create your personal account). Zastanów się chwilę nad sensowną nazwą.
    b. Podaj swój adres ***email*** (podaj istniejący email, trzeba go potem wykorzystać do potwierdzenia rejestracji).
    c. Stwórz ***hasło*** dostępu do portalu (istnieją wymagania co do złożoności hasła, zapisz gdzieś to hasło).
    d. Kliknij przycisk ***Create an account***
    e. Wybierz opcję ***Unlimited public repositories for free***.
    f. Naciśnij klawisz ***Continue***
    g. Wprowadź informacje o sobie: stopień zaawansowania w programowaniu, plany wykorzystania GitHuba, itd. lub pomiń ten krok.
    h. Kliknij klawisz ***Start a project***
    i. Wejdź na swój serwer pocztowy, odbierz mail z Githuba i kliknij zawarty w nim link, aby potwierdzić utworzenie konta.
    j. Utwórz nazwę repozytorium np. *z7*
    k. Utwórz opis repozytorium np. *Zadanie laboratoryjne PAS z7*

l. Wybierz opcję ***Public*** dla swojego repozytorium, ponieważ na Githubie prywatne repozytoria wymagają wnoszenia opłat.
m. Naciśnij klawisz ***Create repository***.

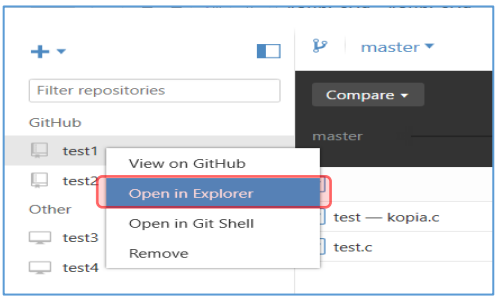
4. Ze strony https://desktop.github.com/ pobierz aplikację, za pomocą której będziesz synchronizować swoje lokalne wersje projektów z wersjami przechowywanymi na Githubie.
   a. W zależności od posiadanego systemu operacyjnego - kliknij odpowiedni link ***Download GitHub Desktop***
   b. W przypadku systemu Windows zapisz na pulpicie aplikację ***GitHubSetup.exe***, za pomocą której można ściągnąć właściwą aplikację GitHub Desktop.
   c. Aplikacja zainstaluje się w katalogu ***Napęd***:\Users***Nazwa_Użytkownika***\AppData\Local\GitHub
   d. Uruchom tę aplikację i zaloguj się do swojego konta na Githubie podając: ***mail*** oraz ***hasło***.
   e. Od tej chwili możesz używać synchronizacji katalogów, w których trzymasz źródła aplikacji na swoim komputerze oraz katalogi na Githubie.
   f. Kliknij na przycisk + znajdujący się w lewej górnej części aplikacji

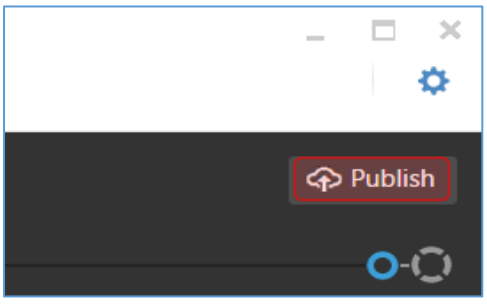




   g. Utwórz nowe repozytoria o nazwach test1, a potem test2 i test3, możesz to zrobić przez wydanie polecenia dodaj (Create) lub wskaż (Add) nowy katalog git. Jeśli wskazany katalog nie zawiera systemowych plików Gita (.gitignore, .gitattributes, .git – podkatalog z całą zawartością), a tak się będzie działo, gdy wskażemy nowoutworzony katalog np. na pulpicie – wówczas aplikacja poprosi o zezwolenie na ich utworzenie.

   h. Kliknij na jednym z utworzonych repozytoriów test1 i wybierz opcję ***Open in Explorer***. Explorer plików otworzy katalog skojarzony z danym repozytorium. Utwórz w tym katalogu przykładowy plik test.c, zwróć uwagę, że aplikacja wyświetliła informację o zmianie, która nastąpiła we wskazanym repozytorium, jeśli skopiujesz plik test.c, wówczas kolejna zmiana w repozytorium zostanie wyświetlona.

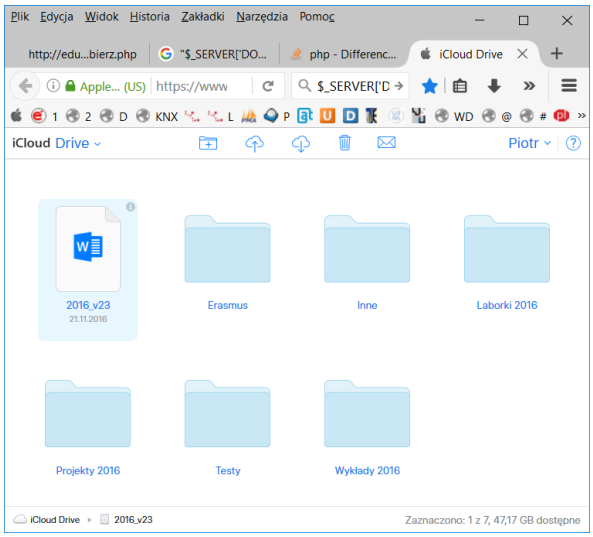




   i. Dopóty dopóki dane repozytorium nie zostanie opublikowane, będzie ono przechowywane jedynie na lokalnym komputerze. Aby to zmienić należy kliknąć na ikonę ***Publish***, znajdującą się w prawym górnym rogu aplikacji. Dopiero wówczas odblokowuje się menu kontekstowe danego repozytorium pozwalając otworzyć je na Githubie ***View on GitHub***. Wykonaj szereg eksperymentów z tworzeniem repozytoriów, ich publikowaniem, synchronizacją oraz usuwaniem. Zwróć uwagę na informacje wyświetlane lokalnie przez aplikację ***GitHub Desktop*** oraz na informacje wyświetlane przez portal github.com.
   j. Przećwiczenie tych opcji jest bardzo ważne, aby zrozumieć wszystkie mechanizmy działania Githuba i potem swobodnie z niego korzystać, bez obawy, że zgubimy gdzieś przez przypadek jaką ważną wersję tworzonej aplikacji, do której będzie trzeba wrócić.

5. Wykorzystując program kontroli wersji Github stwórz aplikację do przechowywania prywatnych plików na serwerze hostingowym, wzorowaną na aplikacjach chmurowych firm Google, Microsoft oraz Apple. Portal powinien umożliwiać realizację następujących funkcjonalności:
   - Rejestrowanie się i logowanie się użytkowników, których hasłach mają być trzymane w tabeli ***users.***
   - Informacja o stanie zalogowania ma być przechowana za pomocą technologii ciasteczek.

- Data i godzina pomyślnego oraz próby błędnego logowania na dane konto mają być zapisywane w tabeli ***logi***.
- Należy wprowadzić limit ilości błędnych logowań – np. po 3 błędnych logowaniach dane konto jest blokowane na 1 minutę. Ma to służyć do uniemożliwienia łamania hasła metodą Brute-force.
- Gdy po błędnym logowaniu następuje poprawne logowanie – użytkownikowi wyświetla się **na czerwono** komunikat ostrzegawczy o dacie i godzinie ostatniego błędnego logowania. Takie rozwiązanie stosowane jest w wielu portalach internetowych umożliwiających dostęp do kont i transakcji bankowych.
- Każdemu nowemu użytkownikowi przyporządkowywany jest na serwerze jego katalog macierzysty o nazwie równej nazwie użytkownika.
- Po pomyślnym zalogowaniu użytkownicy będą mogli:
    - uploadować (umieszczać) swoje pliki na serwerze w swoim katalogu macierzystym,
    - downloadować (ściągać) pliki ze swojego katalogu macierzystego,
    - tworzyć katalogi bezpośrednio w swoim katalogu macierzystym,
    - umieszczać i ściągać pliki ze swoich podkatalogów,
    - opcjonalnie można dodać funkcje usuwania plików i katalogów oraz tworzenie kilkukrotnie zagłębionych struktur podkatalogów.



6. Do wysyłania plików na serwer możesz wykorzystać poniższy formularz wyślij.html oraz plik odbierz.php.
7. W PHP pola typu "file" w formularzu (plik wyslij.html) i odpowiednie zmienne w skrypcie odbierającym dane (plik odbierz.php) umożliwiają proste przesyłanie plików z komputera łączącego się z portalem WWW na serwer, na którym posadowiony jest portal. Proszę jednak zwrócić uwagę, że w przypadku tego zadania użytkownicy mają mieć swoje pliki we własnych katalogach macierzystych o nazwach takich samych jak ich loginy. Zatem poniższy skrypt odbierz.php należy zmienić dodając nazwę użytkownika do ścieżki umożliwiającej zapis pliku.

Skrypt, z którego wysyłany jest plik (wyslij.html)	Skrypt obsługujący odbiór wysłanego pliku (odbierz.php)

<html>
<body>
<form action="odbierz.php" method="POST"
ENCTYPE="multipart/form-data">
<input type="file" name="plik"/>
<input type="submit" value="Wyślij plik"/>
</form>
</body>
</html>
	<?php
	if (is_uploaded_file($_FILES['plik']['tmp_name']))
	{
	echo 'Odebrano plik: '.$_FILES['plik']['name'].'
	';
	move_uploaded_file($_FILES['plik']['tmp_name'],
	$_SERVER['DOCUMENT_ROOT'].$_FILES['plik']['name']);
	}
	else {echo 'Błąd przy przesyłaniu danych!';}
	?>

8. Jeśli zachodzi potrzeba weryfikowania typu pliku i jego rozmiaru można dokonać zmian w pliku obsługującym odbiór wysłanego pliku (odbierz.php), dodając do niego np. sprawdzanie typu pliku i przekroczenie maksymalnego dopuszczalnego rozmiaru pliku. Tego typu weryfikacja rozmiaru i rodzaju plików przydaje się w sytuacji, w której zamierzamy rozbudować kod programu np. o elementy graficzne przesyłane przez użytkowników na serwer. Tymi elementami graficznymi są zwykle zdjęcia przedmiotów przeznaczonych na sprzedaż, fotografie albumu zdjęć lub graficzne ilustracje umożliwiające lepsze zilustrowanie wybranych zagadnień portali e-learningowych itd.

```
<?php
 $max_rozmiar = 1000;
 if (is_uploaded_file($_FILES['plik']['tmp_name']))
   {
    if ($_FILES['plik']['size'] > $max_rozmiar) {echo "Przekroczenie rozmiaru $max_rozmiar"; }
    else
    {
         echo 'Odebrano plik: '.$_FILES['plik']['name'].'<br/>';
         if (isset($_FILES['plik']['type'])) {echo 'Typ: '.$_FILES['plik']['type'].'<br/>'; }
         move_uploaded_file($_FILES['plik']['tmp_name'],$_SERVER['DOCUMENT_ROOT'].$_FILES['plik']['name']);
   }
   }
  else {echo 'Błąd przy przesyłaniu danych!';}
?>
```

9. Prawidłowo zrealizowane aplikacje sieciowe powinny być dobrze udokumentowane oraz zgodne z wytycznymi dotyczącymi technologii realizacyjnych, dlatego po zrealizowaniu aplikacji należy wykonać poniższe badania:
    a. Audyt zgodności z technologią HTML 5 oraz CSS3 (należy do tego wykorzystać specjalistyczne portale internetowe lub narzędzia webmasterskie).
    b. Sprawdzanie, czy nie ma błędów kodu w portalu (j.w.).
    c. Sprawdzanie, czy portal nie wykazuje podatności na włamania typu SQL injection i in (j.w.).
    d. Wykonanie przykładowego automatycznego testu interfejsu użytkownika z wykorzystaniem specjalistycznego narzędzia np. http://www.seleniumhq.org/.
    e. Sprawdzenie szybkości ładowania strony, ewentualna optymalizacja zastosowanej grafiki i ponowne sprawdzenie szybkości ładowania strony (optymalizacja portalu pod względem szybkości ładowania). Jeśli te działania nie przyniosą oczekiwanego efektu można próbować także wykorzystać pamięć podręczną przeglądarki i ponownie zbadać szybkość ładowania strony.
    f. Generowanie mapy portalu (należy do tego wykorzystać specjalistyczne portale internetowe).
    g. Próba ściągnięcia całego portalu za pomocą narzędzi do pobierania statycznych wersji serwisów internetowych (np. Teleport Pro). Po ściągnięciu zawartości testowanego portalu na dysk należy sprawdzić, czy aplikacji udało się zaciągnąć jako wartość offline jakieś elementy, których nie życzylibyśmy sobie udostępniać osobom niezalogowanym, bo np. osłabiałoby to bezpieczeństwo portalu.
    h. Dokumentacja BD MySQL – zastosowanych w aplikacji tabel i relacji z wykorzystaniem specjalistycznych narzędzi np. http://dbdesc.com, Workbench, http://www.dbschema.com/

10. Zgłoś zakończenie realizacji zadania.