informatyka+

Algorytmika i programowanie

Bazy danych

Multimedia, grafika i technologie internetowe

Sieci komputerowe

Tendencje w rozwoju informatyki i jej zastosowań

Człowiek – najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

informatyka+

Kuźnia Talentów Informatycznych: Multimedia, grafika i technologie internetowe

Tworzenie serwisów internetowych

Piotr Kopciał

Człowiek – najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

Tworzenie serwisów internetowych

Rodzaj zajęć: Kuźnia Talentów Informatycznych
Tytuł: Tworzenie serwisów internetowych
Autor: mgr inż. Piotr Kopciał

Redaktor merytoryczny: prof. dr hab. Maciej M Sysło

Zeszyt dydaktyczny opracowany w ramach projektu edukacyjnego **Informatyka+** – ponadregionalny program rozwijania kompetencji uczniów szkół ponadgimnazjalnych w zakresie technologii informacyjno-komunikacyjnych (ICT).
www.informatykaplus.edu.pl
kontakt@informatykaplus.edu.pl

Wydawca: Warszawska Wyższa Szkoła Informatyki
ul. Lewartowskiego 17, 00-169 Warszawa
www.wwsi.edu.pl
rektorat@wwsi.edu.pl

Skład: Recontra Studio Graficzne

Warszawa 2010

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

Tworzenie serwisów internetowych

Piotr Kopciał
Politechnika Warszawska
piotrkopcial@gmail.com

Streszczenie

Internet wkracza w coraz to nowe obszary naszego życia: e-nauczanie, elektroniczne biblioteki, wirtualne laboratoria, medycyna, usługi (bankowość, turystyka). Podstawowym elementem tych i podobnych serwisów są strony internetowe, które składają się na bardziej złożone witryny, portale i platformy internetowe. Zajęcia są poświęcone tworzeniu i funkcjonowaniu serwisów internetowych. W pierwszej części są opisane mechanizmy działania stron internetowych, w tym m.in. komunikacja w standardzie klient-serwer i strony dynamiczne. Omawiane są zalety i wady stron statycznych i dynamicznych oraz mechanizmy interakcji serwisów internetowych z użytkownikiem, stosowane we współczesnych stronach internetowych. Następnie przedstawione będą przykładowe interaktywne serwisy WWW w działaniu. Jednym z celów wykładu jest uwrażliwienie słuchaczy na dobre praktyki projektowania i tworzenia stron internetowych.

W części warsztatowej uczniowie poznają fundamentalny metajęzyk opisu struktury strony – HTML oraz ściśle z nim związany język CSS, odpowiedzialny za prezentację treści na stronie. Przy tym uczą się korzystania z programu (edytora) do tworzenia stron internetowych. Tematyka kursu obejmuje również metody i techniki tworzenia dynamicznych stron internetowych. Uczniowie poznają podstawy języka PHP oraz SQL, a także sposób ich współdziałania w praktycznych przykładach. Zajęcia obejmują również instalowanie i konfigurowanie serwera WWW Apache oraz relacyjnej bazy danych MySQL, a także obsługę środowiska programistycznego, służącego do tworzenia dynamicznych stron internetowych. Uzupełnieniem tych treści jest metodyka tworzenia złożonych projektów informatycznych. Zaprezentowano przykład w pełni funkcjonalnego serwisu.

Wiedzę zdobytą w trakcie wykładów i ćwiczeń, uczniowie weryfikują projektując własny dynamiczny serwis sieciowy w technologii klient-serwer. Serwis ten poza stroną wizualną, zawiera takie funkcje jak: obsługa interaktywnych formularzy użytkownika, przetwarzanie danych wprowadzonych przez użytkownika, obsługa logowania użytkownika, obsługa sesji użytkownika i inne. Celem kursu jest opanowanie umiejętności tworzenia atrakcyjnych, pełnowartościowych serwisów internetowych, a efektem końcowym – ma być serwis internetowy oraz dokumentacja projektu, utworzone samodzielnie przez każdego ucznia.

Spis treści

1. Wprowadzenie 5
1.1. Wstęp 5
1.2. Witryna w Internecie – zasady tworzenia i funkcjonowania 5
1.3. Przykładowe serwisy internetowe 16
1.4. Scenariusz – pierwszy etap dokumentacji projektu 17

2. Projektowanie serwisu WWW 18
2.1. Sposób podejścia do dużego projektu 18
2.2. Przykład serwisu interaktywnego: repozytorium materiałów 19
2.3. Założenia i wymagania – drugi etap dokumentacji projektu 26
2.4. Instalowanie i konfigurowanie środowiska programistycznego MySQL, Apache i PHP 27

3. Tworzenie własnego serwisu WWW 29
3.1. Tworzenie serwisu repozytorium 34
3.2. Baza danych, język SQL 36
3.3. Współpraca PHP z MySQL 37
3.4. Testowanie, poprawianie i prezentowanie własnego serwisu internetowego 47
3.5. Prezentacja działania, wyniki testów, wnioski – trzeci etap dokumentacji projektu 49

Literatura 49

Adresy w Internecie 49

1. WPROWADZENIE

1.1. WSTĘP

Pierwsze strony WWW, towarzyszące pojawieniu się Internetu na początku lat 90. XX wieku, były stosunkowo proste. Zawierały bloki tekstu, uzupełnione ilustracjami. Sieć Internet służyła wtedy do przeglądania raportów i prac naukowych. Dokumenty tekstowe były oznakowane za pomocą znaczników języka HTML, dzięki czemu przeglądarka internetowa wiedziała, w jaki sposób je wyświetlać. Hiperłącza umożliwiające nawigację pomiędzy stronami, skuteczność dystrybucji danych oraz coraz częściej pojawiające się multimedialne treści przyczyniły się do szybkiego rozpowszechnienia Internetu. Podstawowy język tworzenia stron internetowych, HTML, zaczęto wspierać dodatkowymi technologiami, które umożliwiały wzbogacanie stron WWW o nowe elementy.

Technologie internetowe w ostatnich latach znacznie ewoluowały. Dawniej społeczność internetowa dzieliła się na „posiadaczy" i „poszukujących". Ci pierwsi umieszczali informacje na stronach internetowych, ci drudzy natomiast mogli je tylko oglądać. W tamtym okresie istniały tylko statyczne strony WWW. Dziś istnieje wiele portali internetowych, umożliwiających użytkownikowi zalogowanie się do systemu i dostęp do bogatych zasobów, takich jak poczta elektroniczna, własna galeria zdjęć, wirtualny dziennik, interaktywny kalendarz, najnowsze informacje na wybrany przez użytkownika temat. Użytkownik może dostosowywać sposób prezentowania informacji według własnych potrzeb i upodobań.

Postrzeganie Internetu jest obecnie inne niż na początku jego istnienia, bardziej komercyjnie zorientowane. Nadzieje i oczekiwania pokładane w Internecie przyciągają inwestorów, angażujących spory kapitał. Globalna sieć stała się miejscem prowadzenia interesów. W początkowej fazie były to aukcje internetowe. Dziś funkcjonują wyspecjalizowane sklepy (np. e-apteki), , internetowe banki. Szacuje się, że e-biznes stanie się główną siłą ekonomii XXI wieku. Handel elektroniczny (tzw. e-handel) postrzegany jest obecnie jako jeden z najważniejszych aspektów sieci WWW.

Interaktywne serwisy WWW znajdują coraz szersze zastosowanie. Przykładem może być edukacja internetowa. Studenci mogą odbywać zajęcia o dowolnej porze dnia, w dowolnym dniu tygodnia. Platforma edukacyjna to połączenie strony internetowej, poczty elektronicznej, multimedialnych narzędzi i programów do nauczania oraz narzędzi wzajemnej komunikacji pomiędzy użytkownikami. Dane użytkowników (np. oceny z przedmiotów) zapisane w bazie danych składają się na system kontroli postępów w przyswajaniu wiedzy.

Internet to skarbnica wiedzy, której wykorzystanie nie byłoby możliwe bez nowoczesnych technik indeksowania, katalogowania, selekcjonowania i prezentowania dostępnych informacji. Technologie, takie jak PHP, umożliwiają tworzenie dynamicznych serwisów WWW, które coraz częściej są połączone z bazami danych, np. z MySQL.

Wraz z rozpowszechnieniem Internetu pojawiły się narzędzia lepiej integrujące użytkowników. Mogą oni umieszczać własne treści w Internecie, komentować swoje wypowiedzi, pisać dzienniki (tzw. blogi), wyrażać swoje opinie (np. na forach dyskusyjnych). Dzisiejsze strony WWW cechuje akcja i interaktywność, co jest zasługą nowoczesnych technologii tworzenia dynamicznych serwisów WWW.

1.2. WITRYNA W INTERNECIE – ZASADY TWORZENIA I FUNKCJONOWANIA

Instrukcja 1. Kurs rozpoczyna się wykładem wprowadzającym, dotyczącym zasad funkcjonowania i tworzenia statycznych i dynamicznych stron internetowych. Wykład jest wspomagany prezentacją „Witryna w Internecie – zasady tworzenia i funkcjonowania".

Jeszcze 50 lat temu trudno było uwierzyć, że komputery z całego świata mogą zostać ze sobą połączone. W tamtych czasach jedynie niewielka grupa zapaleńców marzyła o współpracy użytkowników komputerów z całego świata, o błyskawicznym wymienianiu się informacjami i różnego rodzaju danymi, takimi jak dokumenty, pliki itp.

INTERNET A INTRANET

Internet to sieć komputerowa, o ogólnoświatowym zasięgu, to sieć sieci. Internet to największa sieć, do której dostęp może mieć każdy użytkownik komputera. Uwaga! Nazwę Internet pisze się wielką literą, ponieważ jest to tzw. nazwa własna.

Czasem można spotkać pojęcie **intranet**. Nie jest to przejęzyczenie. Internet to gigantyczna sieć składająca się z komputerów rozsianych po całym świecie. Natomiast intranet to sieć o mniejszym zasięgu – np. obejmująca komputery w firmie, w szkole lub na uczelni. Intranet tworzy znacznie mniej komputerów, w porównaniu z Internetem.

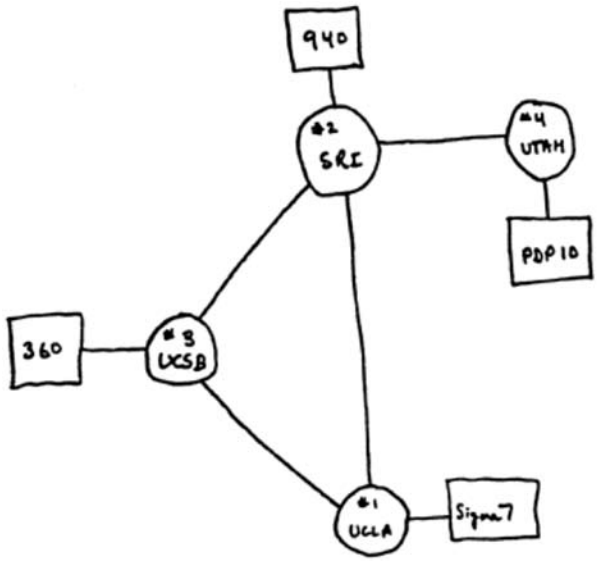
HISTORIA I PRZYSZŁOŚĆ

Historia Internetu sięga roku 1969 (czyli ma już 50 lat!), gdy została uruchomiona pierwsza sieć komputerowa ARPANet (sieć o przeznaczeniu militarnym). Prekursorem rozwiązań internetowych był Paul Baran, Polak z pochodzenia. Na rys.1 jest przedstawiony schemat sieci ARPANet. Sieć ta rozpoczęła działanie od 4 węzłów ulokowanych na amerykańskich uczelniach: University of California w Los Angeles, Stanford Univeristy, University of California w Santa Barbara, University of Utah.

W latach 1971/72 R. Tomlinson opracował protokół poczty elektronicznej, co znacznie przyspieszyło wymianę wiadomości. W roku 1983 powstał protokół TCP/IP – podstawowy protokół służący do wymiany danych pomiędzy komputerami w sieci. Jego twórcami byli Vinton Cerf i Robert Kahn.

W roku 1991 Tim Berners-Lee utworzył pierwszą stronę internetową oraz oprogramowanie do wyświetlania takich stron (przeglądarkę) – uważa się ten moment za początek **serwisu WWW** (ang. *World Wide Web*), czyli **globalnej pajęczyny**.

W Polsce po raz pierwszy połączono się z Internetem latem 1991 roku, chociaż wcześniej możliwy był dostęp do innych sieci o światowym zasięgu, takich jak BITNET.



Rysunek 1.
Schemat sieci ARPANet

Internet w dzisiejszej postaci określa się mianem sieci **Web 2.0**, odnoszącym się do serwisów internetowych, w których podstawową rolę odgrywa użytkownik wraz z generowanymi przez siebie treściami, zasobami, a także serwisami.

A co czeka nas w przyszłości? Zapewne Web 3.0 – dalsza ewolucjia Internetu w kierunku systemu przekazu wiedzy i modelu sieci semantycznej, czyli sieci „rozumiejącej" swoją zawartość oraz użytkowników sieci.

CO MOŻNA ZNALEŹĆ W INTERNECIE

W Internecie można znaleźć wiele informacji i danych, głównie dostępnych poprzez **strony internetowe** w serwisach WWW, począwszy od bardzo wartościowych materiałów i aktualnych treści, przez rozrywkę pod różnymi postaciami (np. gry sieciowe), a skończywszy na treściach zbędnych i szkodliwych. Przykładem wartościowych informacji, do których mamy dostęp w Internecie są:

- multimedialne encyklopedie i materiały edukacyjne, np.:
  http://portalwiedzy.onet.pl/encyklopedia.html, http://pl.wikipedia.org/wiki,
  http://mediawiki.ilab.pl/index.php, http://www.pwi.edu.pl/, http://wazniak.mimuw.edu.pl;

- wirtualne muzea: www.1944.wp.pl, http://www.zamek-lancut.pl;
- obserwacje z życia np. zwierząt – transmisja na żywo obrazu z kamery, np.: http://www.bociany.ec.pl, http://www.teleskopy.net;
- elektroniczne biblioteki: http://www.cm.umk.pl/~biblio/ambgb/b_eksiazki.htm, Google Book Search.

STRONA W INTERNECIE

Strona internetowa jest wynikiem interpretacji **dokumentu HTML**, czyli dokumentu napisanego w języku HTML. Taki dokument może być pobrany z lokalnego dysku komputera lub z serwera internetowego i jest interpretowany po stronie użytkownika przez **przeglądarkę**. Na stronie internetowej można umieszczać tekst, obrazy, tabele, wstawki dźwiękowe, animacje, sekwencje wideo.

Często słyszymy określenie witryna internetowa. **Witryna internetowa** jest określeniem rozbudowanej strony internetowej, która może składać się w wielu stron, do których dostęp uzyskujemy poprzez wybranie odpowiedniej opcji w menu witryny. W dalszej części będziemy na ogół pisać o stronie, bo witryna to zbiór stron.

HTML (ang. *Hypertext Markup Language*) jest językiem programowania, który służy do tworzenia opisów stron internetowych. Język HTML to zestaw znaczników, pomiędzy którymi umieszcza się tekst lub inne elementy mające pojawić się na stronie. Przykładowo dla fragmentu kodu HTML `raz<b> dwa</b> trzy`, wyraz dwa zostanie wyświetlony czcionka pogrubioną, ponieważ jest ograniczony znacznikiem `<b>`.

```
<HTML>
 <HEAD>
  <TITLE> Prosta strona WWW </TITLE>
 </HEAD>
 <BODY>
  <FONT FACE="Arial">
  <CENTER>
   <H1> Informatyka + </H1>
   Witajcie na zajęciach Wszechnicy informatycznej!
  </CENTER>
 </BODY>
</HTML>
```

Strona o powyższym opisie jest przedstawiona na rys. 2.

Rysunek 2.
Prosta strona internetowa

Serwer to komputer, na którym znajduje się plik zawierający opis strony internetowej utworzonej w języku HTML wraz z plikami zawierającymi elementy składowe strony (np. obrazy). Serwer udostępnia stronę innym komputerom za pośrednictwem sieci Internet. W sieci Internet istnieje wiele serwerów.

Przeglądarka to program służący do pobierania opisu stron internetowych z serwera i wyświetlania ich zawartości na ekranie monitora użytkownika. Przeglądarka tłumaczy kod HTML strony na postać oglądaną na ekranie.

Adres URL (ang. Uniform Access Locator) to adres, pod którym jest dostępna konkretna strona internetowa. Przykładowy adres URL to http://www.google.pl/. Adres URL jest adresem serwera, z którym przeglądarka kontaktuje się w celu pobrania opisu strony.

Znaczenie poszczególnych części adresu URL zestawiono w tab. 1.

Tabela 1.
Znaczenie poszczególnych części adresu URL

http:// (https://)	**nazwa_serwera.pl/**	**katalog/**	**plik.html**
nazwa protokołu sieciowego (sposobu przesyłania danych z serwera do przeglądarki)	nazwa domenowa serwera, z którego zostanie pobrany dokument HTML (wyświetlona jako strona)	nazwa folderu (katalogu) na serwerze	nazwa pobieranego pliku (dokumentu HTML) znajdującego się w tym folderze (katalogu)

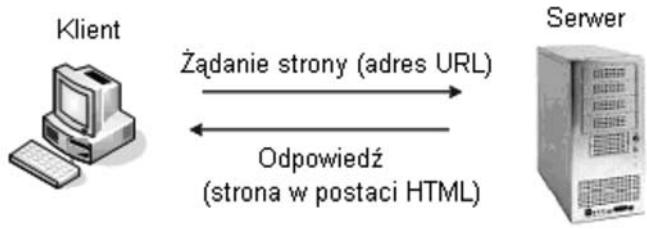
ZASADA DZIAŁANIA STRONY INTERNETOWEJ

Po utworzeniu, strona internetowa jest umieszczana na serwerze. W tym momencie staje się dostępna dla wszystkich użytkowników Internetu. Tak jak budynki na ulicy, każdy serwer ma swój adres (tzw. adres domenowy); a tak jak mieszkania w budynku – każda strona ma swój unikatowy adres.

Gdy użytkownik wpisze adres URL strony w przeglądarce, ta stara się odnaleźć w pierwszej kolejności serwer, a następnie daną stronę. Jeśli znajdzie, serwer odsyła do przeglądarki żądaną stronę w postaci pliku HTML, ewentualnie wraz z uzupełniającymi go plikami graficznymi. Przeglądarka wyświetla stronę na ekranie komputera użytkownika w postaci zdefiniowanej w pliku HTML.

Po to, aby komputer użytkownika (a dokładniej jego przeglądarka) mógł się porozumieć z serwerem, obydwa komputery komunikują się za pomocą protokołu HTTP (ang. *Hypertext Transfer Protocol*).

Taką komunikację nazywamy komunikacją **klient-serwer** (rys. 3). **Klientem** w tym określeniu jest komputer użytkownika, który przy użyciu przeglądarki żąda wyświetlenia wskazanej strony, której opis znajduje się na **serwerze**.



Rysunek 3.
Komunikacja klient-serwer

TWORZENIE STRONY INTERNETOWEJ

Jest wiele powodów, dla których warto umieć tworzyć strony internetowe:

- dla przyjemności – budowanie i prowadzenie własnej strony internetowej może przynieść wiele satysfakcji, możemy np. zaprezentować na niej swoją twórczość milionom internautów;
- w dzisiejszych czasach korzystanie z Internetu stało się tak powszechne, jak korzystanie z edytora tekstu Word do pisania;
- nie musimy płacić za zrobienie czegoś, co można zrobić samemu;
- tworzenie stron internetowych może sprawiać frajdę – być dobrą zabawą, niewymagającą szczególnych umiejętności; na początek wystarczy znajomość języka HTML, który jest łatwy do opanowania.

CO MOŻNA UMIEŚCIĆ NA STRONIE INTERNETOWEJ

1. Tekst. Niektóre strony zawierają wyłącznie tekst. Zaletą takich stron jest zwykle duża wartość informacyjna oraz szybkość wyświetlania w przeglądarce. Wadą jest brak elementów atrakcyjnych dla użytkownika.

2. Obrazy. Obrazy i elementy graficzne przyciągają uwagę użytkownika. Mogą to być np. własne zdjęcia, rysunki oferowanych przez firmę produktów lub mapka dojazdu na miejsce. Pobranie strony zawierającej elementy graficzne z serwera do przeglądarki trwa jednak znacznie dłużej niż pobranie strony tekstowej.
3. Formularze. Formularze stosuje się do zbierania informacji od użytkowników odwiedzających daną stronę (rejestracja, ankieta itp.) lub przekazywania danych przez użytkowników, chcących np. uzyskać informacje od właściciela strony. Formularze stanowią także formę zamówień w transakcjach internetowych.
4. Obramowania. Ramki stosuje się do podziału strony na kilka części, w których można grupować podobne informacje. Przykładowo na stronie księgarni internetowej informacje ogólne i pole wyszukiwania oddzielone są od części strony zawierającej opisy poszczególnych działów tematycznych.
5. Multimedia: sekwencje audio i wideo. Umieszczone na stronie multimedia są najbardziej atrakcyjnymi elementami dla osób odwiedzających Internet.

PROJEKTOWANIE WITRYNY

Nie można kopać dołu na fundamenty, nie mając gotowego projektu domu [6]. Słowa te oddają, jak ważne jest zrobienie dobrego planu (projektu), przed przystąpieniem do realizacji praktycznej. Dotyczy to również tworzenia stron internetowych.

Zaprojektowanie strony, którą chcemy utworzyć, to podstawa. Od tego, co i w jaki sposób chcemy umieścić na stronie, zależeć będą dalsze czynności. Należy odpowiedzieć sobie na pytania:

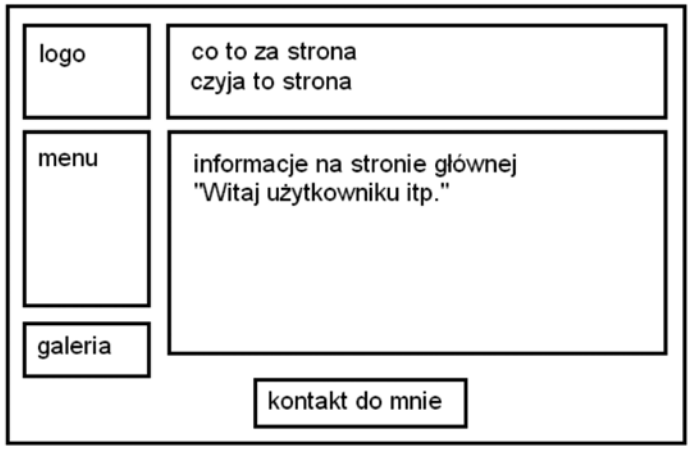
- co skłoniło mnie od tworzenia własnej strony?
- do kogo strona jest adresowana?
- co chcę umieścić na stronie?
- w jaki sposób chcę zaprezentować siebie (lub np. swoją firmę) innym?

Najczęściej na stronie umieszcza się:

- informacje o swoich zainteresowaniach (hobby) lub swojej firmie,
- zdjęcia prywatne lub zdjęcia oferowanych produktów wraz z ich opisem,
- formularze, dzięki którym osoby odwiedzające stronę mogą przekazywać i wymieniać informacje z właścicielem strony,
- elementy graficzne, które czynią stronę bardziej atrakcyjną wizualnie.

Zawartość strony zależy od przeznaczenia strony i jej odbiorców.

Kiedy już zdecydujemy, co ma znajdować się na stronie, należy następnie rozrysować układ strony na kartce papieru. Bardzo typowy układ strony internetowej przedstawiono na rys. 4.



Rysunek 4.
Typowy układ strony internetowej

NIE JESTEŚMY ODBIORCAMI SWOJEJ STRONY

Gdy tworzymy stronę na temat, który nas interesuje, możemy dojść do wniosku, że wszyscy odbiorcy strony są podobni do nas. Jest to błędne przekonanie. Należy bowiem zdawać sobie sprawę, że wiemy na temat naszej witryny znacznie więcej niż osoby, które widzą ją po raz pierwszy. Oznacza to jednocześnie,

że na temat użytkownika – odbiorcy naszej witryny wiemy mniej niż nam się wydaje. Jest to jedna z największych trudności piętrząca się przed twórcami stron internetowych.

Pamiętajmy! Nie projektujemy strony dla siebie. Projektujemy ją dla innych użytkowników Internetu, którzy będą odwiedzać naszą stronę.

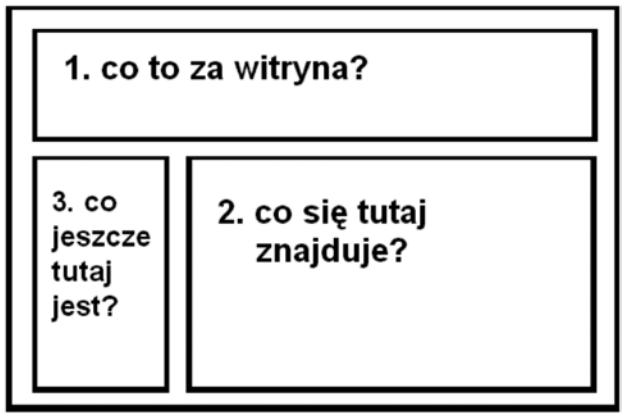
Najczęściej popełniane błędy:

- złe zaplanowanie struktury (układu) strony,
- brak przemyślanego grupowania informacji na wybrany temat,
- używanie żargonu i słów niezrozumiałych dla innych,
- zbytnie przeładowanie strony elementami, które rozpraszają, a nie przyciągają uwagę.

JAK LUDZIE WIDZĄ WITRYNY INTERNETOWE

Osoba trafiająca na naszą stronę powinna mieć możliwość szybkiego zorientowania się, co może na niej znaleźć. Należy wyraźnie zasygnalizować, jakie informacje użytkownik może uzyskać i w jaki sposób. Internauci są niecierpliwi. Jeżeli w ciągu 15 sekund użytkownik stwierdzi, że nie może znaleźć tego, czego szuka (informacje na dany temat, gry, łącza do innych stron), to jest bardzo prawdopodobne, że opuści stronę bezpowrotnie.

Badania zachowań internautów wykazują, że intuicyjnie przeglądają oni strony internetowe według pewnego powtarzającego się schematu. Zwykle na początku spoglądają na górę strony, aby zorientować się, co to za strona (rys. 5). Następnie kierują wzrok ku środkowi. Jeśli nie znajdą tam tego, czego szukają, zmierzają wzrokiem w kierunku lewej części strony, gdzie spodziewają się znaleźć elementy nawigacji (menu).



Rysunek 5.
Kolejność przeglądania strony internetowej przez internautę

JAK LUDZIE NAWIGUJĄ W INTERNECIE

Czasem warto zastanowić się, jak ludzie zachowują się, kiedy nawigują w Internecie. Z różnych mediów korzystamy w różny sposób:

- czasopisma – czytamy,
- radia – słuchamy,
- telewizję – oglądamy,
- w Internecie – nawigujemy tak, jakby to była przestrzeń.

W dzisiejszych czasach obserwujemy konwergencję mediów. Zarówno czasopisma, jak i radio oraz telewizja są dostępne przez Internet.

Globalna sieć jest zupełnie innym środkiem przekazu, niż druk, radiofonia czy telewizja. Ludzie przemieszczają się pomiędzy stronami w wirtualnej przestrzeni. Na każdej stronie poszukują sygnałów nawigacyjnych, skupiają się na dotarciu do miejsc docelowych i myślą „gdzie przejść, którą stronę odwiedzić".

Kursor myszy stanowi niejako przedłużenie ręki użytkownika. Z tego powodu nawigacja w Internecie jest podobna do nawigacji w przestrzeni fizycznej. Dlatego nawigacja w obrębie projektowanej witryny jest tak ważna.

5 SKUTECZNYCH SPOSOBÓW NA ODSTRASZENIE UŻYTKOWNIKÓW INTERNETU

Marzeniem każdego projektanta strony jest to, aby jego strona zyskała popularność i uznanie użytkowników. Może jednak zdarzyć się sytuacja, w której użytkownik odwiedzający naszą stronę po raz pierwszy, już nigdy nie zechce na nią wrócić. Aby do tego nie doszło, należy unikać następujących sytuacji:

1. Wyłączenie serwera, na którym umieszczona jest nasza strona (nikt nie będzie mógł się do niej dostać) – jeśli nie dysponujemy komputerem, który mógłby pełnić rolę serwera i pracować bez przerwy, skorzystajmy z usług firm świadczących usługi hostingowe.
2. Umieszczanie zbyt wielu elementów multimedialnych (grafika, dźwięk, film), spowalniających wyświetlanie strony (przeglądarka użytkownika będzie pobierać stronę bardzo długo),
3. Zmienianie rozmieszczenia elementów na stronie, co powoduje, że użytkownik powracający na stronę nie może poruszać się znajomymi drogami i znaleźć tego, czego szuka,
4. Umieszczanie odnośników do stron, których nie można wyświetlić (użytkownik spotka się z niezrozumiałym komunikatem serwera),
5. Brak aktualizowania treści (artykuły, zdjęcia, odnośniki do innych stron) witryny – jeśli co jakiś czas nie będą pojawiać się świeże informacje, to użytkownik nie będzie miał powodu do odwiedzenia naszej witryny.

5 SPOSOBÓW POPRAWY WITRYNY

1. Skoncentruj się przede wszystkim na tym, żeby witryna dobrze funkcjonowała. Wygląd stron ma znaczenie drugorzędne.
   Strony internetowe muszą się szybko ładować, jeśli ludzie mają ich używać. Być może konieczny będzie kompromis pomiędzy efektami, jakie chcemy uzyskać, a szybkością, która jest ograniczana przez te efekty. [6]
2. Myśl o użytkowniku.
   Projektant strony powinien wcielić się w użytkownika i wyobrazić sobie jak to jest, gdy korzysta się z powolnego łącza internetowego.
3. Projektuj stronę zgodnie z przyjętymi konwencjami.
   W ciągu ostatnich lat wypracowano sprawdzony schemat układu strony, do którego użytkownicy są przyzwyczajeni. W obrębie dobrze zaprojektowanej strony użytkownik porusza się intuicyjnie.
4. Zwróć uwagę na szczegóły.
   Potocznie błahe błędy, takie jak brak wyrównania, brak odpowiednich oznaczeń, mogą sprawić kłopot użytkownikowi.
5. Testuj.
   Najlepszym sposobem na sprawdzenie działania witryny jest jej przetestowanie przez użytkowników, a następnie poprawienie według poczynionych spostrzeżeń i sugestii.

JĘZYK HTML I STRUKTURA DOKUMENTU HTML

Opis stron internetowych jest tworzony w języku HTML. Nauka tego języka jest dość łatwa. HTML jest zestawem znaczników. Każdy znacznik umieszczony jest w nawiasach ostrych < >.

Przykładowo – znacznikiem rozpoczęcia opisu strony jest `<HTML>`. Większość znaczników występuje jako część otwierająca i zamykająca. Część zamykająca zawiera dodatkowy znak – ukośnik /. Znacznikiem zamykającym stronę jest zatem `</HTML>`.

Strukturę dokumentu HTML opisującego stronę określają 3 znaczniki: `<HTML>`, `<HEAD>` i `<BODY>`.

`<HTML>` – użycie tego znacznika jest obowiązkowe, gdyż wskazuje on na początek i koniec dokumentu. Znacznik `<HTML>` musi znaleźć się w pierwszym wierszu kodu strony.

`<HEAD>` – znaczniki definiujący nagłówek dokumentu. Można w nim określić takie elementy, jak nazwa i styl dokumentu, tytuł strony. Nagłówek umieszczamy na początku dokumentu, a kończymy go znacznikiem `</HEAD>`.

`<BODY>` – pomiędzy znacznikami `<BODY>` oraz `</BODY>` zawarta jest zasadnicza treść dokumentu. W tej części można definiować: rodzaj czcionki, kolor tekstu, tło strony itd.

Przykład prostego dokumentu HTML został podany wcześniej, przy okazji prezentowania na rys. 2 efektu jego interpretacji przez przeglądarkę.

HIPERŁĄCZA

Fragmenty na stronie internetowej, a także inne obiekty mogą odgrywać rolę **łącza** z innymi stronami i witrynami w Internecie – łącza takie nazywamy **hiperłączami**. Tekst na stronie internetowej określa się mianem **hipertekstu**, gdyż może zawierać hiperłącza (krócej łącza) i elementy multimedialne, nie będące tekstem. Hiperłączy można używać na dwa sposoby:

- jako odsyłaczy do innych stron naszej witryny,
- jako odsyłaczy do innych stron w Internecie.

Poniżej zilustrowano ten drugi przypadek. Umieszczenie hiperłącza na stronie wymaga użycia odpowiedniego znacznika HTML:

```
<HTML>
 <HEAD>
  <TITLE> Prosta strona WWW </TITLE>
 </HEAD>
 <BODY>
  <FONT FACE="Arial">
  <CENTER>
   <H1> Informatyka + </H1>
   Witajcie na zajęciach Wszechnicy informatycznej!</br>
   Więcej na temat programu Informatyka+ znajdziecie na
   <a href="http://http://informatykaplus.edu.pl/">
   stronie projektu</a>
  </CENTER>
 </BODY>
</HTML>
```

Strona o tym kodzie ma postać jak na rys. 6, a efektem kliknięcia w hiperłącze na tej stronie jest przejście do strony pokazanej na rys. 7.

Rysunek 6.
Prosta strona powitalna zawierająca hiperłącze

Rysunek 7.
Strona wyświetlona jako efekt kliknięcia w hiperłącze na stronie przedstawionej na rys. 6 [20.08.2009]

NARZĘDZIA DO TWORZENIA STRON

Kod HMTL można pisać w prostym edytorze tekstu, np. w Notatniku Windows. Wystarczy znać znaczniki HTML i zasady ich stosowania. Jednakże dużym ułatwieniem jest posłużenie się specjalnym programem do tworzenia stron internetowych, tzw. edytorem języka HTML.

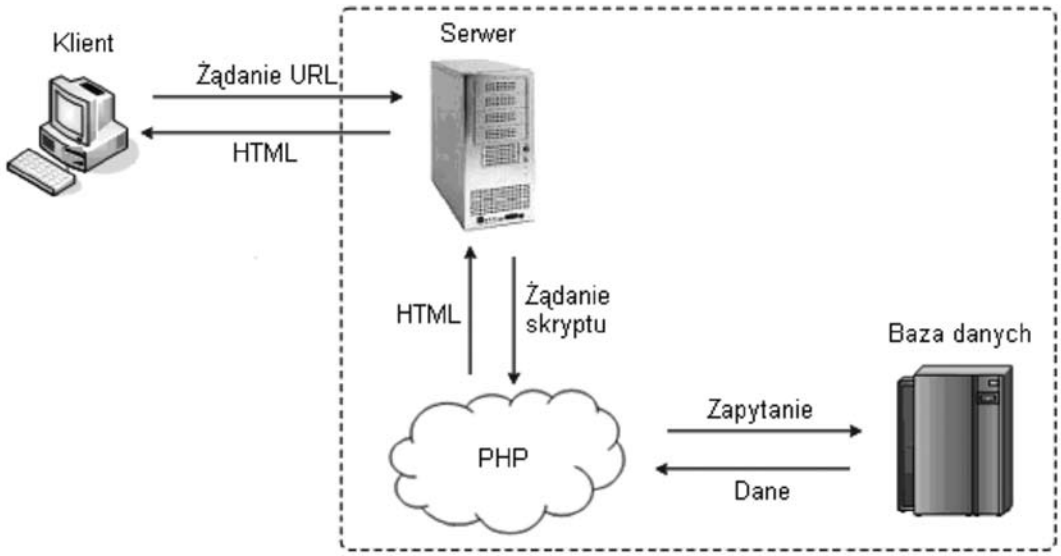
Edytory są pomocne przy tworzeniu bardziej złożonych elementów stron, takich jak np. tabele. Zamiast tworzyć w Notatniku każdą komórkę tabeli osobno, wystarczy skorzystać z narzędzia tworzenia tabel w takim edytorze.

Przykładem edytora HTML jest program FrontPage firmy Microsoft. Istnieje również wiele darmowych edytorów (lub dostępnych za darmo przez określoną liczbę dni, np. 60), które można pobrać z Internetu. Darmowe edytory HTML można pobrać z następujących stron: http://agerwebedytor.com/ (Ager Web Edytor), http://www.darmoweprogramy.org/programy/edytory_html.php (Alleycode HTML Editor, Web Design Toy, EasyHTML, HotHTML).

DYNAMICZNA STRONA INTERNETOWA I ZASADY JEJ DZIAŁANIA

Dzięki dynamicznym stronom internetowym można np. witać użytkownika odwiedzającego stronę ponownie w następujący sposób: „Witaj ponownie, Krzysiu!".

Dynamiczna strona internetowa jest tworzona przez serwer w momencie, kiedy użytkownik żąda jej wyświetlenia. Strony dynamiczne są generowane na bieżąco i mogą zawierać różne treści, w zależności od tego, kto je pobiera i w jakich okolicznościach. Na przykład tło strony może być jasne lub ciemne, w zależności od tego, czy akurat jest dzień czy noc.



Rysunek 8.
Działanie dynamicznej strony WWW

Na rys. 8 przedstawiono działanie dynamicznej strony WWW. Interakcja pomiędzy klientem a ser-werem zaczyna się w momencie wpisania w przeglądarce adresu strony lub kliknięcia łącza do strony dynamicznej. Za pomocą protokołu HTTP przeglądarka nawiązuje połączenie z serwerem. Serwer przesyła żądanie do interpretera języka skryptowego (np. PHP), który wykonuje kod skryptu – **skryptem** nazywamy kod napisany w języku przeznaczonym do tworzenia stron dynamicznych. Jeśli w skrypcie PHP są zapisane zapytania do bazy danych (np. w celu pobrania informacji o użytkowniku), interpreter języka skryptowego odpowiada za komunikację serwera z bazą danych. Po pobraniu zawartości strony, przeglądarka analizuje kod HTML, po czym wyświetla gotową stronę na ekranie monitora użytkownika.

Należy zwrócić uwagę, że dynamiczne fragmenty strony internetowej nie istnieją, dopóki ktoś nie zażąda wyświetlenia strony. Dopiero wtedy serwer buduje taką stronę, według instrukcji zawartych w kodzie HTML oraz w kodzie skryptu, a gdy użytkownik zamyka stronę dynamiczną w przeglądarce, to dynamiczne fragmenty strony przestają istnieć. W przypadku kolejnego wyświetlenia takiej strony, jej dynamiczne fragmenty są tworzone na nowo. Dzięki temu na stronach mogą ulegać zmianie: godzina, data, prognoza pogody, program telewizyjny itp.

W odróżnieniu od strony dynamicznej, treść strony statycznej nie zmienia się od momentu jej utworzenia do chwili zmiany opisu strony lub usunięcia go z serwera.

STRONA STATYCZNA A STRONA DYNAMICZNA

Statyczne strony WWW opisane w języku HTML, są przechowywane na serwerze i przesyłane są w takiej samej postaci do wszystkich użytkowników. Oznacza to, że każdy użytkownik widzi taką samą stronę pod względem treści i układu.

Natomiast strony dynamiczne są generowane przez serwer na bieżąco, w zależności od tego kim jest użytkownik (np. użytkownik zalogowany do serwisu ma dostęp do treści niedostępnych dla użytkowników niezalogowanych). Mechanizm ten wymaga od serwera większej pracy, aniżeli w przypadku stron statycznych, kiedy to rola serwera sprowadza się do przechowywania plików, oczekiwania na żądanie i przesłania strony wskazanej przed użytkownika do jego przeglądarki.

Ponadto, potrzebna jest baza danych zawierająca treści, które mają pojawić się na stronie. **Baza danych** jest elektronicznym magazynem informacji (danych) i narzędziem do zarządzania tymi informacjami.

Strony statyczne jak i strony dynamiczne mają swoje wady i zalety, co zilustrowano w tab. 2.

Tabela 2.
Wady i zalety stron statycznych i dynamicznych

	Wady	Zalety
Strony statyczne	– nie można szybko zmienić treści – interakcja z użytkownikiem bardzo ograniczona	– łatwo je utworzyć (kod HTML)
Strony dynamiczne	– trudniej je utworzyć (języki skryptowe są trudniejsze do opanowania niż HTML) – wymagają bazy danych na serwerze	– łatwo i szybko można zmienić treść – umożliwiają interakcję z użytkownikiem

Statyczne strony WWW, nawet te najbardziej atrakcyjne pod względem treści i grafiki, mają wadę, która polega na tym, że aktualizacja ich treści zajmuje sporo czasu, ponieważ wymaga modyfikowania każdej strony. Wady tej są pozbawione witryny z elementami dynamicznymi, których treść przechowywana jest w bazie danych i pobierana przy każdym otwarciu strony przez odwiedzającego. Ponadto zmiana treści dynamicznego fragmentu strony wymaga zmiany w jednym tylko miejscu – w bazie danych.

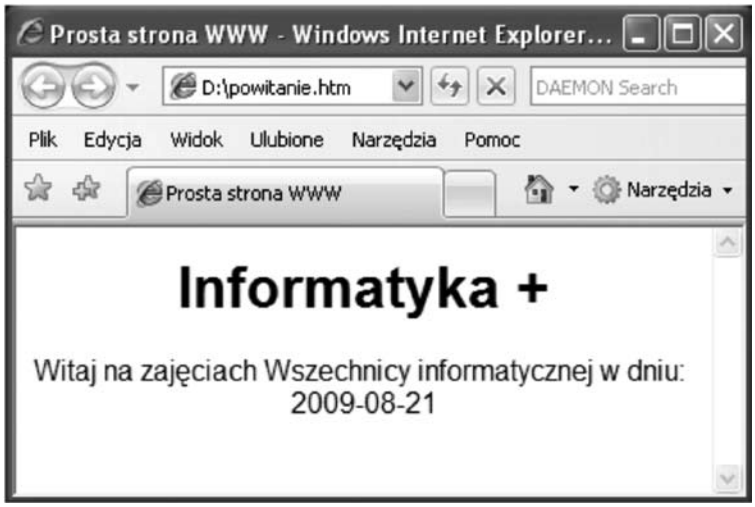
TWORZENIE STRON DYNAMICZNYCH – JĘZYK SKRYPTOWY

Dynamiczne strony internetowe tworzy się za pomocą tzw. **języków skryptowych**. Fragmenty kodu napisane w języku skryptowym są umieszczane pomiędzy znacznikami kodu HTML strony. W języku skryptowym można zdefiniować polecenia dla serwera, w jaki sposób ma budować (generować) stronę. Można np. wyświetlić aktualną datę i godzinę lub pobrać aktualne informacje (np. na temat pogody) z bazy danych. Najczęściej stosowanym i najprostszym do nauki językiem skryptowym jest PHP. Poniżej przedstawiono kod skryptu generującego aktualną datę.

```
<HTML>
 <HEAD>
  <TITLE> Prosta strona WWW </TITLE>
 </HEAD>
 <BODY>
  <FONT FACE="Arial">
  <CENTER>
   <H1> Informatyka + </H1>
   Witaj na zajęciach Wszechnicy informatycznej w dniu:
   <?php
```

```
    echo date(„Y-m-d”);
   ?>
  </CENTER>
 </BODY>
</HTML>
```

Efekt działania tego skryptu jest pokazany na rys. 9 – za każdym razem, gdy ta strona jest wyświetlana, pobierana jest aktualna data.



Rysunek 9.
Strona wyświetlająca aktualną datę

INTERAKCJA Z UŻYTKOWNIKIEM WIZYTÓWKĄ NOWOCZESNYCH STRON INTERNETOWYCH

Strony WWW uważa się za **interaktywne**, jeśli przechowują sesję użytkownika. **Sesja** jest sposobem przechowywania informacji o użytkowniku (np. towary, jakie gromadzi w koszyku w internetowym sklepie) pomiędzy następującymi po sobie żądaniem klienta i odpowiedzią serwera.

Interaktywność polega na tym, że treść strony może się dynamicznie zmieniać, w zależności od:

- Profilu użytkownika – osoby, które korzystały wcześniej z serwisu mogą automatycznie otrzymywać informacje na interesujące je tematy (np. wyniki sportowe, lokalna prognoza pogody). Inny przykład to różny zakres opcji menu dostępnych na stronie w zależności od tego, czy dana osoba jest administratorem, moderatorem, czy zwykłym użytkownikiem,
- Wprowadzonych danych – w zależności od danych wprowadzonych wcześniej przez użytkownika treść strony może być wyświetlana w różny sposób. Przykładem jest personalizacja serwisu, często stosowana w serwisach społecznościowych – użytkownicy mogą zmieniać wygląd strony (tło, format tekstu itp.) według własnych upodobań,
- Przeglądarki oraz systemu operacyjnego użytkownika – języki skryptowe (np. PHP) umożliwiają tworzenie aplikacji pobierających informacje o systemie operacyjnym i przeglądarce użytkownika po to, aby jak najlepiej dostosować do nich sposób wyświetlania strony,
- Czasu – np. wyświetlanie ciemniejszego tła serwisu w godzinach nocnych lub różnych motywów na stronie w zależności od pory roku,
- Położenia geograficznego.

Do ciekawych efektów stosowanych na stronach internetowych należą m.in.:

- podświetlanie przycisków po najechaniu na nie kursorem myszy,
- zmiana kształtu kursora myszy,
- pojawianie się okien dialogowych,
- mechanizm przeciągnij-i-upuść,
- manipulowanie grafiką (np. przełączanie obrazków),
- uruchamianie wyskakujących okienek (np. pojawianie się okienka informacyjnego, gdy użytkownik umieści wskaźnik myszy na obrazku),

- przesuwanie mapy,
- zwijanie i rozwijanie menu.

Efekty te tworzy się przy użyciu techniki AJAX (ang. Asynchronous Javascript And XML), wykorzy-stującej m.in. język skryptowy JavaScript.

1.3. PRZYKŁADOWE SERWISY INTERNETOWE

Instrukcja 2. W tej części zajęć uczniowie zapoznają się z przykładami serwisów internetowych. Przy użyciu komputera z łączem internetowym prowadzący prezentuje kolejno każdy z serwisów. Należy zwrócić uwagę na charakter i przeznaczenie prezentowanych serwisów. Pomysł na przeznaczenie nowo powstającego serwisu internetowego najczęściej podyktowany jest względami praktycznymi. Serwis może służyć np. do wymiany dokumentów, plików, informacji, odnośników do witryn o określonej tematyce itp. Wskazane jest, aby w trakcie omawiania prezentowanych serwisów, uczniowie zastanowili się nad przeznaczeniem i funkcjami własnego serwisu, który mają zamiar utworzyć na tych zajęciach.

STUDENT.LEX.PL

Student.lex.pl jest portalem internetowym przeznaczonym dla studentów prawa. Celem założycieli było stworzenie wyczerpującej bazy wiedzy potrzebnej do studiowania na tym kierunku. Portal Student.lex.pl zawiera materiały dydaktyczne, zamieszczane przez samych studentów, w tym m.in.:

- notatki,
- opracowania wykładów,
- pytania egzaminacyjne,
- cytaty,
- wybór aktualnych aktów prawnych, a także wiele innych narzędzi pomocniczych:
- katalog opracowań – uczelnie,
- katalog opracowań – przedmioty,
- ogłoszenia drobne,
- lista dyskusyjna,
- księgarnia internetowa,
- słownik ang.-pol. i pol.-ang.

GRATYZCHATY.PL

GratyZChaty.pl jest polskim portalem wymiany niepotrzebnych elementów wyposażenia domu. Użytkownicy mogą bezpłatnie dodawać ogłoszenia typu „oddam" lub „przyjmę". Można przeglądać ogłoszenia w kategoriach takich, jak:

- kuchnia,
- łazienka,
- ogród,
- meble,
- łóżka,
- artykuły gospodarstwa domowego AGD,
- odzież,
- buty,
- kosmetyki,
- biżuteria,
- i inne.

DIGG.COM

Digg.com jest jednym z najpopularniejszych serwisów służących do wymiany odnośników do stron o dowolnej tematyce. Każdy użytkownik może poinformować innych o ciekawych artykułach, filmach i wszelkiego rodzaju plikach dostępnych w sieci. Punktowa ocena użytkownika przekłada się na pozycję odnośnika w rankingu – najlepsze są prezentowane na stronie głównej.

W Polsce serwisem działającym na takiej samej zasadzie jak Digg.com jest Wykop.pl. Wzrost popularności tego typu serwisów związany jest z zalewem sieci przez informacje bezużyteczne (spam), co rodzi potrzebę eksponowania informacji najbardziej wartościowych.

ALLEGRO.PL

Allegro.pl jest najpopularniejszym polskim serwisem aukcyjnym. W sieci istnieje wiele serwisów aukcyjnych. Można na nich kupić niemal wszystko: od drobiazgów i używanych rzeczy, które nie są już potrzebne ich właścicielom, poprzez nowe, firmowe towary wystawiane przez sklepy internetowe, skończywszy na samochodach. Takie zróżnicowanie oferty wpływa na popularność serwisu. W roku 2008 na największym polskim serwisie aukcyjnym sprzedano 97 mln przedmiotów o wartości ponad 5,2 mld PLN, a liczba użytkowników przekroczyła 8 mln.

Serwis aukcyjny jest wyposażony w narzędzia ułatwiające pracę. Początkującego użytkownika prowadzi przyjazny system pomocy. Aby sprzedać przedmiot, należy się zarejestrować. Dla zapewnienia bezpieczeństwa transakcji i płatności, serwis oferuje bezpieczną rejestrację. Każdy użytkownik posiada swój profil oraz możliwość monitorowania prowadzonych przez siebie aukcji i zarządzania nimi. Za pośrednictwem forum można nawiązać kontakt z innymi użytkownikami.

Procedura wystawiania przedmiotu na aukcji jest intuicyjnie prosta. Użytkownik prowadzony jest krok po kroku: wybiera typ aukcji, kategorię przedmiotu, wprowadza informacje o sprzedawanym przedmiocie, dołącza pliki graficzne (np. zdjęcia przedmiotu), podaje cenę i czas trwania aukcji. Po kilku minutach przedmiot jest dostępny dla milionów internautów. W momencie zakupienia przedmiotu, wystawca dostaje stosowną informację w wiadomości e-mail, a kupujący kontaktuje się z nim w celu sfinalizowania transakcji. Kupujący i sprzedający mogą wystawiać sobie nawzajem komentarze, do których inni użytkownicy mają również dostęp. Pozytywne komentarze przyczyniają się do tworzenia reputacji wiarygodnego sprzedawcy, jak również uczciwego kupującego.

PROPOZYCJE TEMATÓW

- Wyszukiwarka materiałów dydaktycznych, np. dla przedmiotów szkolnych.
- Repozytorium własnych materiałów, np. materiały dotyczące mojego hobby.
- Sklep używanych książek, zbliżony działaniem do Allegro.
- Serwis wymiany plików, zawierający wykaz plików w różnych kategoriach, wraz z ich opisem.
- Forum dyskusyjne, blog, księga gości (jako element własnej witryny).
- Propozycja własna…

Instrukcja 3. Uwzględniając powyższe propozycje tematów projektów oraz przedstawione serwisy internetowe, każdy z uczniów wybiera/proponuje temat własnego serwisu.

1.4. SCENARIUSZ – PIERWSZY ETAP DOKUMENTACJI PROJEKTU

Celem niniejszego kursu jest utworzenie przez każdego uczestnika własnego serwisu internetowego. Aby skutecznie spełnić to zadanie, należy przygotować dokumentację projektu. Dokumentacja służy dokładnemu sformułowaniu celu pracy, określeniu założeń i wymagań. Treść dokumentacji powinna oddawać cały proces powstawania serwisu – opis poszczególnych etapów realizacji projektu, a po jego ukończeniu – opis efektu końcowego, wyniki testów oraz wnioski.

Najczęściej spotykany schemat dokumentacji ma następującą postać:

1. Wstęp – cel pracy
2. Scenariusz – opis działania systemu (serwisu)
3. Przegląd istniejących rozwiązań
4. Założenia i wymagania
5. Realizacja projektu
6. Uruchamianie i testowanie
7. Podsumowanie i wnioski
8. Literatura i załączniki

Na tym etapie zajmiemy się punktami 1-3.

Instrukcja 4. Efektem końcowym kursu, oprócz utworzonego serwisu będzie jego dokumentacja. Dokumentacja będzie rozwijana przez cały czas trwania kursu, aby przyjąć ostateczną postać po zrealizowaniu projektu. W tej części zajęć uczniowie tworzą pierwszy etap dokumentacji projektu – scenariusz. Należy utworzyć nowy dokument w edytorze tekstowym (np. Word), zatytułować go „Serwis internetowy elektroniczna_biblioteka (lub inny temat zaproponowany przez ucznia)", a w treści utworzyć listę numerowaną, odpowiadającą poszczególnym etapom realizacji projektu (1. Wstęp … 8. Literatura). Następnie każdy z uczniów rozwija punkty 1.-3 jak poniżej.

1. Wstęp – cel pracy
   Wstęp ma na celu wprowadzenie czytelnika w tematykę pracy. Pamiętajmy, że dokumentację tworzymy nie tylko dla siebie – piszmy więc tak, aby osoba z zewnątrz na podstawie dokumentacji mogła łatwo zorientować się, co było celem naszych działań. Napiszmy, jaki wybraliśmy temat pracy i dlaczego właśnie taki.
2. Scenariusz – opis działania systemu (serwisu)
   W tej części należy opisać, jak ma działać projektowany system, jakie funkcje, z punktu widzenia użytkownika, ma realizować.
3. Przegląd istniejących rozwiązań
   Fragment ten ma udokumentować, że przed przystąpieniem do części praktycznej, przeanalizowano istniejące w Internecie serwisy, realizujące podobne funkcje jak projektowany przez nas serwis. Dobrym pomysłem jest utworzenie tabeli, ułatwiającej porównanie serwisów pod kątem wyglądu, sposobu działania, liczby odbiorców itp.

2. PROJEKTOWANIE SERWISU WWW

Instrukcja 5. W tej części zajęć, w trakcie jednogodzinnego wykładu wprowadza się uczniów w tematykę projektowania rozbudowanego serwisu internetowego. Główny nacisk kładzie się na przemyślane, metodyczne podejście, uwzględniające takie elementy, jak: zastosowanie inżynierii oprogramowania, planowanie i prowadzenie projektu, tworzenie kodu łatwego w utrzymaniu, standardy kodowania, wybór środowiska programistycznego, oraz testowanie.

2.1. SPOSÓB PODEJŚCIA DO DUŻEGO PROJEKTU

Dotychczas przedstawiliśmy projekty składające się co najwyżej z kilku skryptów. Jednak przy tworzeniu w pełni funkcjonalnych serwisów rzadko kiedy występują tak proste zadania. Dzisiejsze serwisy interaktywne to aplikacje z prawdziwego zdarzenia, zawierające tysiące wierszy kodu. Tej wielkości projekty wymagają zaplanowania i zarządzania.

ZASTOSOWANIE INŻYNIERII OPROGRAMOWANIA

Inżynieria oprogramowania to systematyczne, etapowe podejście to tworzenia oprogramowania. Wiele projektów informatycznych poniosło porażkę z powodu braku podejścia zgodnego z inżynierią oprogramowania. Inżynieria oprogramowania opiera się na zaplanowanym działaniu, systematycznym tworzeniu projektu zgodnie z przyjętym planem. Jednym z powodów, dla których w wielu projektach informatycznych nie stosuje się takiego podejścia jest pośpiech związany z nadchodzącym terminem ukończenia projektu. Jeśli projekt nie zostanie odpowiednio zaplanowany, efektem będą błędy w działaniu aplikacji oraz nieczytelny kod.

PLANOWANIE I PROWADZENIE PROJEKTU

Nie ma jedynej właściwej metody najlepszego prowadzenie projektu informatycznego. Istnieje jednak wiele kwestii, które należy wziąć pod uwagę:

- Przed rozpoczęciem pracy należy zdefiniować cel – co ma być efektem końcowym projektu. Cel ten należy mieć cały czas na uwadze. Należy zastanowić się, jakie są oczekiwania użytkowników.
- Projekt należy podzielić na części składowe. Pozwoli to nie tylko rozdzielić pracę pomiędzy programistów, ale także uchroni przed popełnieniem wielu błędów i czasochłonnymi poprawkami.
- Po podzieleniu aplikacji na mniejsze części należy sprawdzić, czy niektóre komponenty nie zostały zrobione już wcześniej. Można spróbować zastosować je w projekcie. Zwłaszcza w przypadku środowisk open source wiele kodów aplikacji jest dostępnych za darmo.
- W trakcie tworzenia projektu należy stosować się do przyjętych zasad i konwencji: standardów kodowania, struktury kodu źródłowego, wersji środowiska programistycznego itp.
- Należy testować projekt na każdym etapie jego tworzenia – im wcześniej wykryjemy błędy, tym mniej czasu zajmie ich usuwanie.
- Po utworzeniu aplikacji prototypowej, należy pokazać ją odbiorcom do przetestowania, a następnie poprawić według ich zaleceń.

TWORZENIE KODU ŁATWEGO W UTRZYMANIU

W przypadku złożonego projektu informatycznego zazwyczaj pracę dzieli się pomiędzy kilku/kilkunastu programistów. Kod napisany przez każdego z nich powinien być zgodny z kodem pozostałych osób. Taki kod jest również łatwiejszy do wykorzystania przez innych. Z tego powodu tak ważne jest trzymanie się pewnych standardów.

STANDARDY KODOWANIA

Większość dużych firm z branży IT ma wypracowane własne standardy kodowania:

- konwencje nazewnicze – sposoby nazywania plików i zmiennych, odróżnienie zmiennych od stałych itp.,
- komentowanie kodu – należy opisywać znaczenie funkcji, klas, plików, skomplikowanych fragmentów kodu,
- wcinanie – sposób oddzielenia bądź zagnieżdżania bloków kodu, sposób stosowania nawisów,
- dzielenie kodu – w celu uczynienia kodu bardziej czytelnym dla innych,
- stosowanie standardowej struktury katalogów – pliki wchodzące w skład aplikacji finalnej powinny być uporządkowane w postaci struktury katalogów. Służy to oddzielaniu części odpowiedzialnej za bazę danych od części odpowiedzialnej np. za wygląd witryny.

WYBÓR ŚRODOWISKA PROGRAMISTYCZNEGO

Przy tworzeniu projektów programistycznych korzysta się z programów ułatwiających tworzenie kodu, sprawdzanie poprawności działania aplikacji, poprawianie błędów. To, którego programu użyjemy, zależy od rozmiaru projektu, a także naszych umiejętności. Zazwyczaj początkujący programiści korzystają z prostych edytorów języków skryptowych, natomiast profesjonaliści zatrudnieni w firmach – z bardziej zaawansowanego oprogramowania.

TESTOWANIE

Testowanie jest podstawowym punktem inżynierii oprogramowania. Proces testowania powinien być nieodłącznym elementem każdego etapu tworzenia aplikacji. Należy oddzielnie testować aplikacje cząstkowe, a po połączeniu ich ze sobą przetestować aplikację finalną. Im więcej testów przeprowadzimy, tym większe będą szanse, że wykryjemy więcej błędów. Podczas testowania może okazać się, że aplikacja działa w sposób, którego nikt nie planował. Niekiedy okazuje się, że osoby testujące zgłaszają potrzebę dopracowania projektu (np. dodanie jeszcze jednej funkcji).

2.2. PRZYKŁAD SERWISU INTERAKTYWNEGO: REPOZYTORIUM MATERIAŁÓW

Instrukcja 6. W trakcie jednogodzinnego wykładu prowadzący omawia (ewentualnie uzupełniając wykład prezentacją Power Point) serwis internetowy – repozytorium materiałów. W trakcie wykładu uczniowie określają, jakie komponenty („cegiełki") są potrzebne do realizacji ich własnych projektów.

Przedstawimy przykład rozbudowanego serwisu internetowego repozytorium materiałów. Poszczególne etapy tworzenia tego serwisu zostaną omówione w dalszej części kursu. Działanie aplikacji opiera się na zasadzie wzajemnego dzielenia się wiedzą przez użytkowników, którzy mogą zamieszczać, edytować i usuwać odnośniki do materiałów np. dydaktycznych. Aplikacja została zrealizowana w oparciu o darmowe oprogramowanie (*open source*): platformę serwerową Apache HTTP Server w wersji 2.2.8, system baz danych MySQL 5.0.22 oraz język skryptowy PHP w wersji 5.2.5.

ZAŁOŻENIA

Podstawowym zadaniem stawianym przed projektowaną aplikacją było udostępnienie użytkownikom narzędzia umożliwiającego wyszukiwanie wartościowych materiałów, np. odnośników do kursów edukacyjnych w Internecie. Narzędzie powinno udostępniać odnośniki do materiałów, np. zasobów internetowych pomocnych do nauki w szkole.

W trakcie projektowania przyjmujemy następujące założenia:

- Układ strony głównej powinien być przejrzysty. Uporządkowanie informacji jest często czynnikiem decydującym o pozostaniu użytkownika na stronie lub jej opuszczeniu. Przeciętny internauta poświęca kilka sekund na zorientowanie się w układzie witryny, po czym zaczyna z niej korzystać, albo opuszcza ją bezpowrotnie.
  Witryna WWW ma tylko dwie lub trzy sekundy na przyciągnięcie uwagi użytkownika. Dlatego pierwsza strona – i pierwsze informacje, które pojawią się na tej stronie – musi wywoływać u czytelnika dobre wrażenie (zalecenia America Online). [3]
- Informacje powinny być zorganizowane w możliwie klarowny sposób, aby ich znalezienie było intuicyjnie łatwe.
- Projekt powinien być otwarty na koncepcje udoskonalenia algorytmu wyszukiwania oraz dodawanie nowych funkcji. Użyte środowisko programistyczne powinno gwarantować możliwości wprowadzenia zmian (tzw. architektura otwarta).

WYMAGANIA FUNKCJONALNE

- Interfejs użytkownika
  Z punktu widzenia użytkownika projektowana wyszukiwarka powinna być interaktywną stroną WWW z dostępem do bazy danych, w której będą gromadzone informacje wprowadzane przez użytkowników.
- Prezentacja danych
  Odpowiedzią na zapytanie użytkownika powinna być lista odnośników do plików lub linków do stron internetowych. Interfejs użytkownika powinien gwarantować szybkie i intuicyjne odnalezienie się na stronie. Odpowiedź powinna być na tyle precyzyjna, aby użytkownik dotarł do wskazanej pozycji np. klikając w łącze na stronie lub podając dane odnośnika w bibliotece. Owocem wyszukiwania powinien być odnośnik do konkretnego dokumentu HTML (strona WWW), dokumentu tekstowego (.doc, .pdf), nagrania video (.wmv) lub audio (.wav), linku do tematu na forum dyskusyjnym (np. z informacjami od opiekunów dydaktycznych o udostępnianych materiałach) czy też odnośnik do konkretnego e-kursu.
- Typy użytkowników i ich uprawnienia
  Z zaprojektowanego narzędzia korzystać mają użytkownicy różnych typów, np. gość (użytkownik niezarejestrowany), użytkownik zarejestrowany oraz administrator. Po zalogowaniu, użytkownik powinien mieć możliwość przeglądania materiałów zgromadzonych w repozytorium, a także dodawania nowych materiałów, edytowania ich i usuwania. Największe uprawnienia powinien posiadać administrator.

WYMAGANIA TECHNICZNE

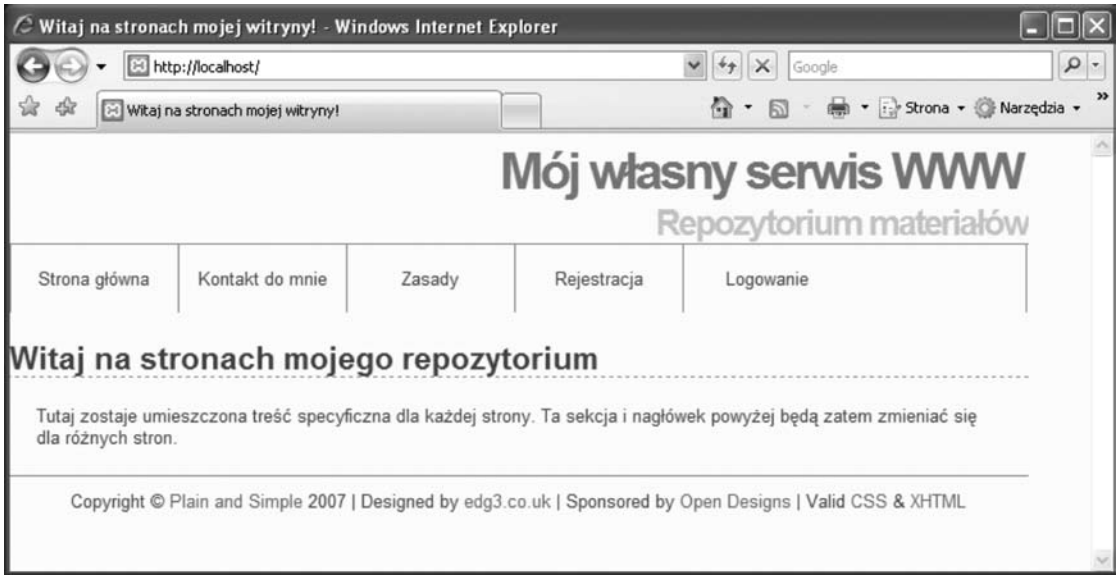
- Serwer aplikacji
  Aplikacja oraz baza danych powinny być kompatybilne z komputerem szkolnym lub domowym. Preferowane jest użycie sprawdzonego środowiska programistycznego. Wskazany jest wybór środowiska darmowego. Rozwiązanie takie cieszy się zazwyczaj popularnością, jest dobrze udokumentowane i chętnie rozwijane przez szerokie grono użytkowników.
- Oprogramowanie użytkownika
  Przy projektowaniu interfejsu graficznego należy wziąć pod uwagę rozdzielczość ekranu użytkownika. Ponadto należy uwzględnić użycie popularnych przeglądarek internetowych, z których najczęściej korzystają uczniowie i studenci. Nie bez znaczenia pozostaje również kwestia używanego systemu operacyjnego.

Należy tak zaprojektować naszą aplikację, aby mogło z niej korzystać jak najwięcej potencjalnych użytkowników. Obsługa polskich znaków to kolejne istotne wymaganie, z którym należy się liczyć. Standardem kodowania polskich znaków jest ISO-8859-2.

- Wydajność
  Należy zwrócić uwagę, iż zaprojektowana aplikacja docelowo ma działać na serwerze szkolnym, który utrzymuje już inne aplikacje oraz strony internetowe, lub komputerze domowym. Należy tak dobrać parametry projektowanej aplikacji, aby nie obciążała ona serwera – nie zakłócała jego działania, a z drugiej strony, aby posiadała wystarczający zasób przestrzeni operacyjnej do poprawnego działania. Należało również oszacować maksymalne obciążenie generowane przez użytkowników i uwzględnić je przy projektowaniu aplikacji.
- Monitoring
  Powinna istnieć możliwość monitorowania działania aplikacji przez administratora. Kontrola poprawności działania może odbywać się na podstawie analizy określonych parametrów (np. średni czas ładowania strony, czas odpowiedzi na zapytanie). Monitorowane dane mogą również posłużyć do analizy korzystania z narzędzia przez użytkowników i w konsekwencji usprawnienia działania aplikacji.
- Bezpieczeństwo
  Z uwagi na fakt, iż aplikacja jest tworzona, aby docelowo działać na serwerze uczelnianym, należy zadbać o stabilność działania. Należy się przygotować na niebezpieczeństwo dostępu nieuprawnionych bądź złośliwych użytkowników, ingerowania w bazę danych, oraz dołożyć starań w celu zwiększenia bezpieczeństwa witryny.

DZIAŁANIE SERWISU

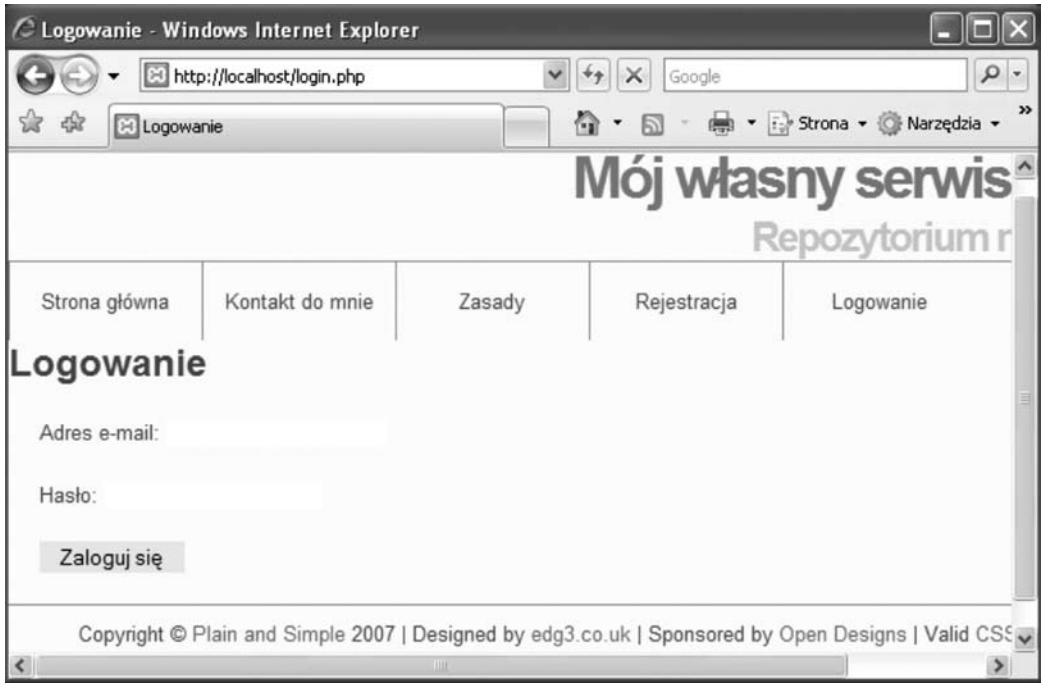
Przy projektowaniu układu strony należy wziąć pod uwagę, iż większość użytkowników Internetu jest przyzwyczajona do schematu budowy strony internetowej (do najczęściej spotykanego układu na stronie). W związku z tym należy zadbać o intuicyjne rozmieszczenie elementów strony: elementów graficznych, opcji menu, przycisków. Przemyślana budowa strony (rys. 10) sprawia, że korzystanie z niej jest intuicyjnie proste i skuteczne.



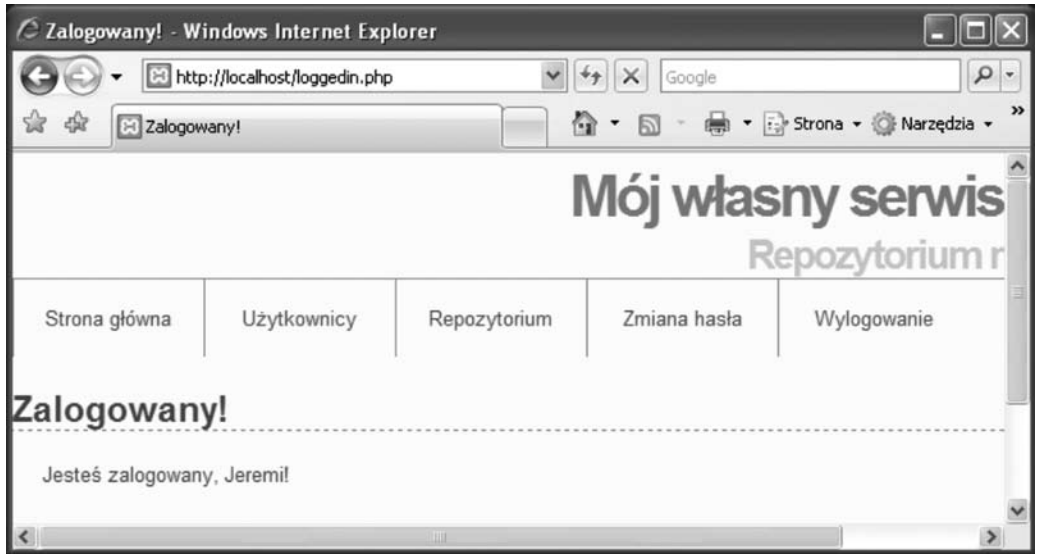
Rysunek 10.
Strona główna serwisu repozytorium

Aby ułatwić korzystanie z serwisu nowym użytkownikom, na stronie głównej widnieje do zasad użytkowania serwisu oraz łącze do rejestracji. Algorytm korzystania z serwisu jest następujący:

1. Po wejściu na stronę główną użytkownik wybiera opcję w menu [Logowanie].
2. Zostaje wyświetlone okno logowania, w którym użytkownik wpisuje adres email i hasło, a następnie naciska [Zaloguj się] (rys. 11).
3. Po zalogowaniu się do systemu, w menu są dostępne już inne opcje (rys. 12).

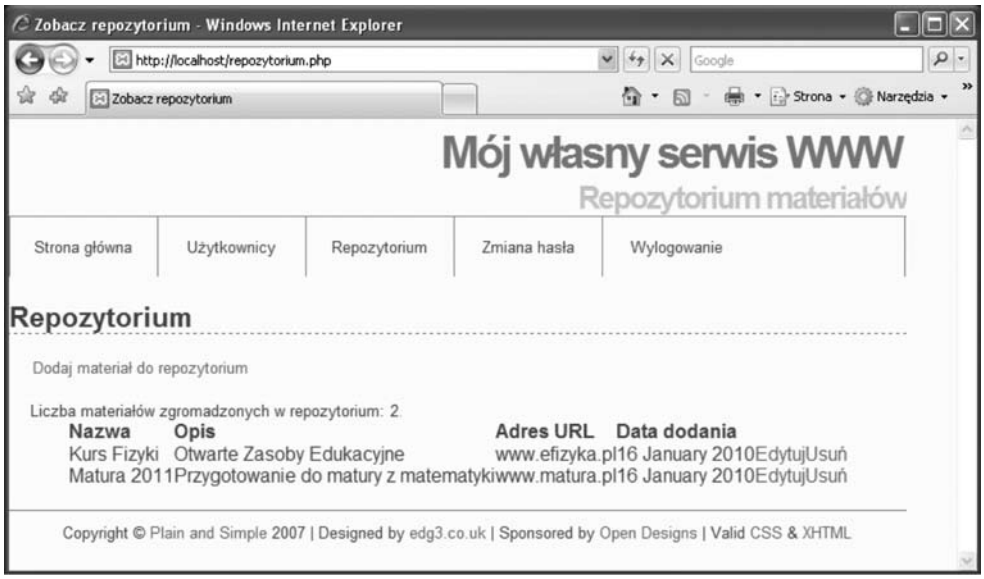


Rysunek 11.
Logowanie



Rysunek 12.
Widok serwisu po zalogowaniu

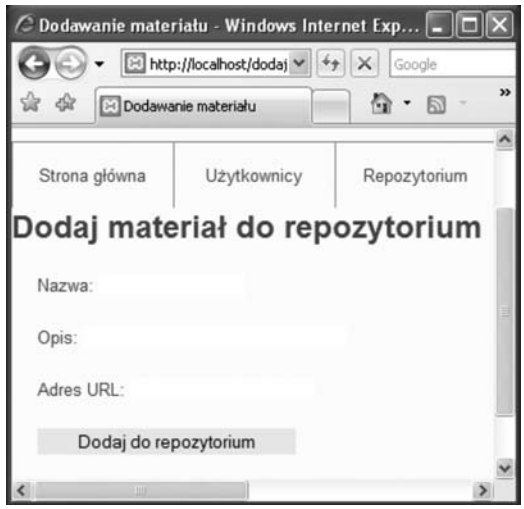
4. Użytkownik wybiera [Repozytorium] i zostaje wyświetlone właściwe okno zawierające zbiór materiałów (rys. 13). Należy również uwzględnić, jakie informacje o danym materiale strona powinna wyświetlać. Są to:
   - nazwa,
   - opis,
   - adres URL lub „miejsce przebywania",
   - data dodania do bazy danych.
5. Aby się wylogować, użytkownik wybiera w menu [Wylogowanie].



Rysunek 13.
Przeglądanie repozytorium

Użytkownik może także dodawać materiały do repozytorium. Dodawanie nowego materiału przebiega następująco:

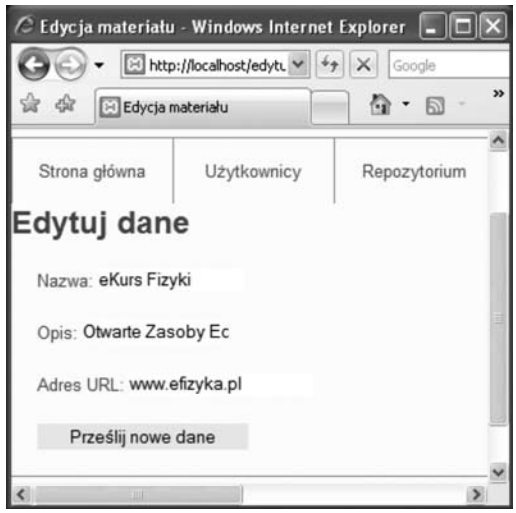
1. Po zalogowaniu się i wybraniu łącza [Repozytorium] użytkownik wybiera łącze [Dodaj materiał do repozytorium],
2. Wyświetla się formularz do wypełnienia (rys. 14),



Rysunek 14.
Formularz dodawania materiału do repozytorium

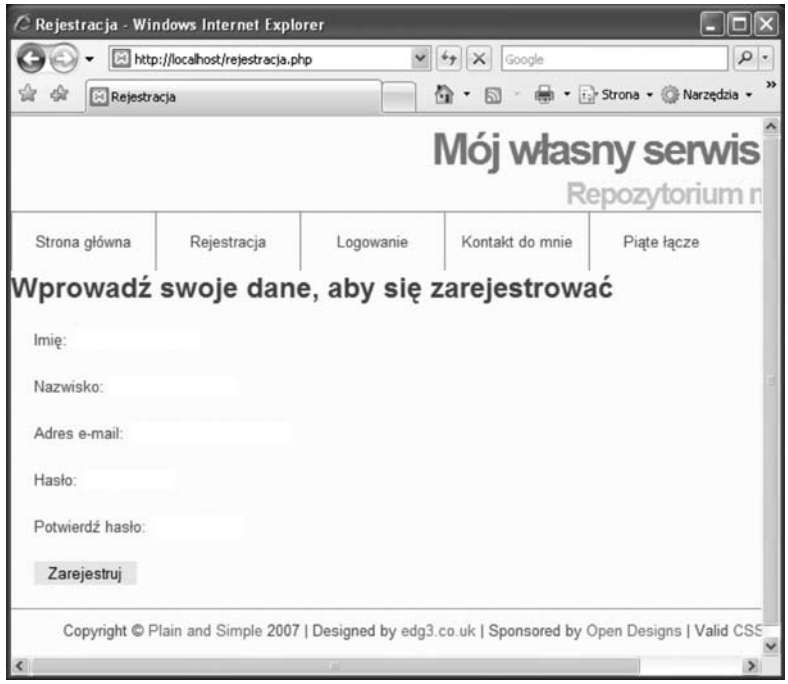
3. Użytkownik wypełnia pola formularza (podanie adresu URL jest opcjonalne),
4. Użytkownik naciska [Dodaj do repozytorium], aby zatwierdzić wpisywanie informacji – dane są zapisywane w bazie.

Poza tym użytkownik może edytować materiały już istniejące (rys. 15). Taka sytuacja może mieć miejsce wtedy, gdy zmieni się adres url, miejsce przebywania materiału lub jego zastosowanie (co można zasygnalizować edytując opis). Edycja może również okazać się niezbędna, by poprawić błędy popełnione przez innych użytkowników (np. przypisanie niedziałającego linku do strony). Natomiast usunięcie odnośnika może być potrzebne np. wtedy, gdy pewne informacje staną się nieaktualne.



Rysunek 15.
Strona edycji materiału

Rejestracja użytkownika (rys. 16) odbywa się przy użyciu adresu email, po zapoznaniu się z zasa-dami korzystania z serwisu. Po zarejestrowaniu się, użytkownik może się logować do serwisu.



Rysunek 16.
Rejestracja

Profil użytkownika w bazie danych zawiera:
- imię,
- nazwisko,
- adres email,
- hasło,
- datę rejestracji.

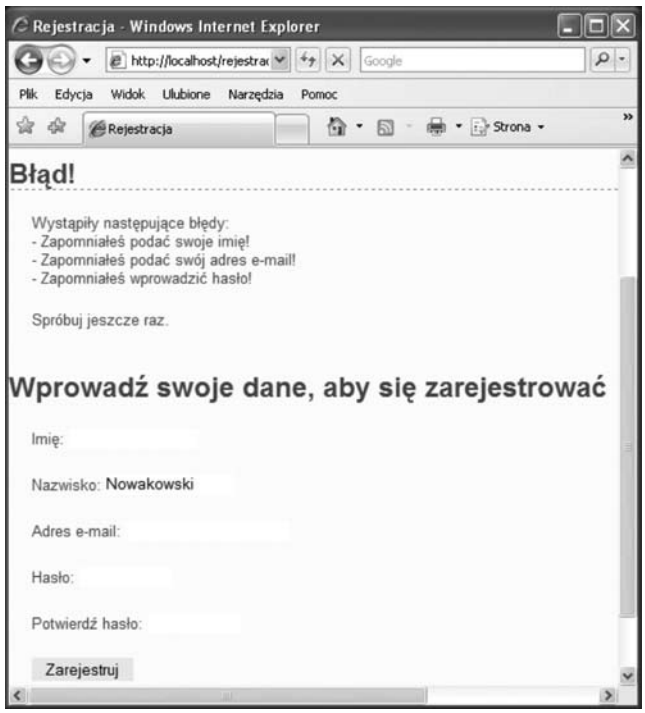
IMPLEMENTACJA

Zaczynając prace programistyczne, należy odpowiednio się przygotować. Projekt wymaga etapowego, systematycznego podejścia do pisania skryptów programu. Właściwe zaplanowanie projektu może przy-

czynić się do zminimalizowania ewentualnych usterek w działaniu aplikacji końcowej. Ważnym etapem jest zdekomponowanie projektu na mniejsze składniki. Tworzenie i częste testowanie podprogramów niezależnie od siebie pozwoli uniknąć błędów już na samym początku pisania kodu. Tworzenie od początku kodu łatwego w utrzymaniu, przyjęcie powszechnych standardów kodowania, struktury katalogów, nazewnictwa, systematyczne zapisywanie kopii roboczych w znacznym stopniu wpłynie na skrócenie czasu tworzenia aplikacji.

BEZPIECZEŃSTWO I STABILNOŚĆ

Z punktu widzenia bezpieczeństwa oraz stabilności systemu, w trakcie projektowania aplikacji należy zabezpieczyć ją przed wprowadzaniem (celowym bądź pomyłkowym) nieprawidłowych danych przez użytkowników. Po każdorazowym wprowadzeniu danych przez użytkownika, instrukcje języka PHP powinny sprawdzać ich poprawność, a w przypadku wprowadzenia błędnych danych – wyświetlać odpowiedni komunikat. Na rys. 17 przedstawiono przypadek, w którym użytkownik w trakcie rejestracji nie wypełnił formularza w sposób kompletny.



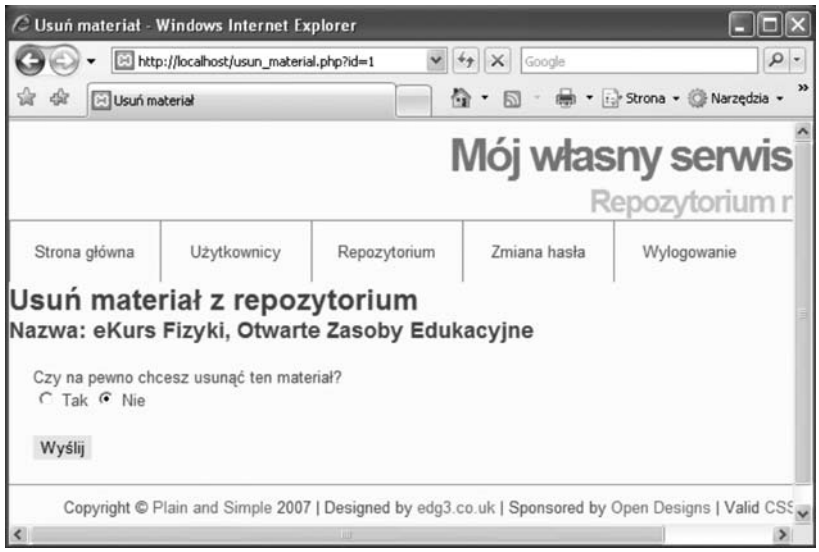
Rysunek 17.
Komunikat o błędach w trakcie rejestracji użytkownika

Witryna WWW nigdy nie jest całkowicie zabezpieczona albo niezabezpieczona; może tylko być bardziej lub mniej bezpieczna. Bezpieczeństwo to nie przełącznik, lecz skala, po której można się poruszać w górę lub w dół. [5]

Od początku tworzenia projektu należy mieć na uwadze bezpieczeństwo, np. w trakcie tworzenia bazy danych:

- użytkownicy powinni mieć wyłącznie te uprawnienia, które są dla nich absolutnie niezbędne,
- podczas wprowadzania poufnych informacji do tabel (np. haseł) powinny być one zabezpieczane przy użyciu funkcji szyfrującej,
- przed wpisaniem danych przez użytkownika, powinna być sprawdzana ich poprawność,
- skrypt nawiązujący połączenie z bazą powinien być przechowywany poza głównym katalogiem witryny.

Sprawne korzystanie z narzędzia powinno iść w parze z interakcją z użytkownikiem. Dlatego np. na stronie usuwania materiałów, aplikacja powinna wymagać od użytkownika potwierdzenia operacji usunięcia, poprzez wybranie odpowiedniego przycisku (rys. 18). Taki sposób postępowania zapobiega przypadkowej modyfikacji informacji w bazie.



Rysunek 18.
Usuwanie odnośnika z bazy danych

PERSPEKTYWY ROZBUDOWY SERWISU

W związku z nieustannym rozwojem technologii sieciowych, przy projektowaniu aplikacji należy brać pod uwagę możliwości jej dalszej rozbudowy w przyszłości. PHP jako język rozszerzalny, daje niemal nieograniczone możliwości. Jest nieustannie rozwijany przez programistów. Powstają coraz nowsze wersje, jeszcze bardziej wydajne i stabilne. PHP może współpracować niemal z każdym systemem baz danych, co daje dużą elastyczność. Kontynuacja może polegać na zastąpieniu użytych technik sieciowych nowymi – bardziej wydajnymi (np. nowsza wersja PHP, MySQL) ale także rozszerzeniu zakresu tematycznego, np. repozytorium uzupełnione o inne narzędzie Web 2.0., np. blog.

Innym rozszerzeniem może być umożliwienie użytkownikom przesyłania plików na serwer bazy danych. W przypadku, gdy do jakiegoś odnośnika nie istnieje adres url (np. jest to notatka z wykładu, czy też rozwiązanie zadania na papierze), użytkownik mógłby zeskanować dokument i wgrać go na serwer, zamieszczając łącze do tego pliku. Jednakże takie rozwiązanie dodatkowo obciążałoby serwer bazy danych, osłabiając wydajność jego działania. Trudno byłoby również oszacować potrzebne zasoby pamięci dysku twardego serwera.

2.3. ZAŁOŻENIA I WYMAGANIA – DRUGI ETAP DOKUMENTACJI PROJEKTU

Instrukcja 7. Na tym etapie uczeń powinien sprecyzować swoje plany i scenariusz. Uczniowie uzupełniają dokumentację o założenia i wymagania dotyczące własnego projektu, według poniższych instrukcji:

Drugi etap tworzenia dokumentacji projektu polega na sformułowaniu założeń i wymagań. **Założenia** to przede wszystkim zakres pracy: liczba i rodzaj odbiorców, koszty (np. wykorzystanie wyłącznie darmowego oprogramowania – *open source*), czas realizacji (liczba dni, godzin) projektu itp. **Wymagania** są uwarunkowane przede wszystkim parametrami technicznymi, np. właściwe działanie i wygląd serwisu niezależnie od systemu operacyjnego oraz przeglądarki, z której odbiorca korzysta. Niniejszy rozdział dokumentacji projektu warto opracować starannie, gdyż zawarte w nim informacje będą stanowić punkt odniesienia, który posłuży do określenia, czy cel został osiągnięty.

REALIZACJA PROJEKTU

Rozdział dokumentacji poświęcony realizacji projektu (punkt 5. schematu dokumentacji) nie może zostać pominięty. Powinien stanowić odzwierciedlenie całego procesu powstawania rozwiązania – pokazywać, w jaki sposób twórca pracował. Należy zatem notować wszelkie swoje działania (praca nad treścią serwisu, nad wyglądem, nad tworzeniem elementów składowych) i umieszczać je w tej części dokumentacji.

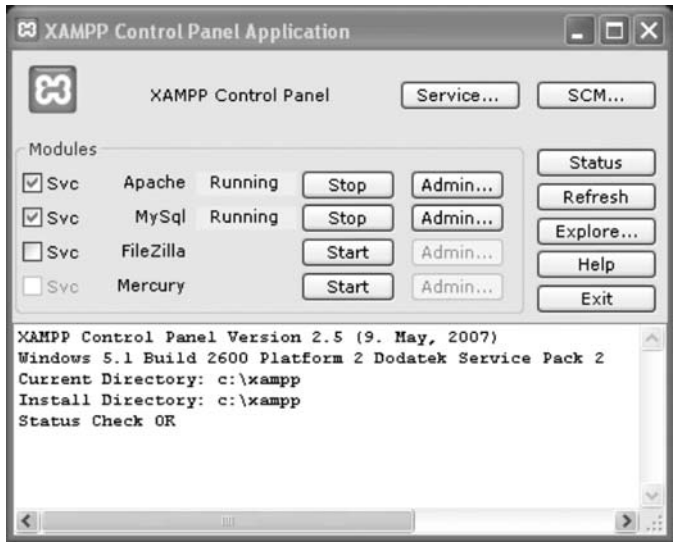
Instrukcja 8. W trakcie realizacji projektu prowadzący powinien systematycznie przypominać uczniom o uzupełnianiu dokumentacji, np. po zakończeniu etapu tworzenia wyglądu serwisu, a przed rozpoczęciem tworzenia bazy danych, należy nakazać uczniom uzupełnienie dokumentacji.

2.4. INSTALOWANIE I KONFIGUROWANIE ŚRODOWISKA PROGRAMISTYCZNEGO MYSQL, APACHE I PHP

Instrukcja 9. W tym momencie każdy z uczniów pobiera, instaluje i konfiguruje niezbędne narzędzia – oprogramowanie to tworzenia własnego serwisu, według następujących instrukcji.

Do tworzenia i testowania działania dynamicznych stron internetowych niezbędne jest zainstalo-wanie i skonfigurowanie środowiska programistycznego: serwera WWW oraz środowiska obsługi jezyka skryptowego (w tym przypadku PHP), uzupełnionego bazą danych (w tym przypadku My-SQL). Każdy z powyższych komponentów można instalować i konfigurować oddzielnie, jednakże wygodniejszym rozwiązaniem jest użycie aplikacji, która zainstaluje wszystkie te komponenty we właściwej kolejności. Jednym z programów, które instalują i konfigurują serwer WWW, PHP oraz MySQL jest XAMPP. Program ten instaluje dodatkowo serwer poczty elektronicznej oraz aplikację phpMyAdmin, służącą do obsługi bazy danych.

Program XAMPP odnajdziemy za pośrednictwem witryny http://www.apachefriends.com lub bezpośrednio po kliknięciu http://www.komputerswiat.pl/download/30001,2711/init,program.aspx. Po dwukrotnym kliknięciu na pobranym pliku rozpoczyna się instalacja. Uwaga: pakiet XAMPP instalujemy w katalogu C:/xampp, a serwery Apache oraz MySQL instalujemy jako usługi (ang. *as service*). Po zainstalowaniu możemy uruchomić panel administracyjny – rys. 19.



Rysunek 19.
Panel administracyjny XAMPP

W dalszej kolejności, ze względów bezpieczeństwa, należy skonfigurować hasło użytkownika root serwera MySQL. W tym celu w systemie Windows wybieramy z menu Start polecenie Uruchom, wpisujemy cmd i naciskamy OK. Następnie przechodzimy do katalogu mysql/bin, używając polecenia `cd C:\xampp\mysql\bin`. W celu ustawienia hasła wykonujemy polecenie:

```
mysql –u root –p hasło
```

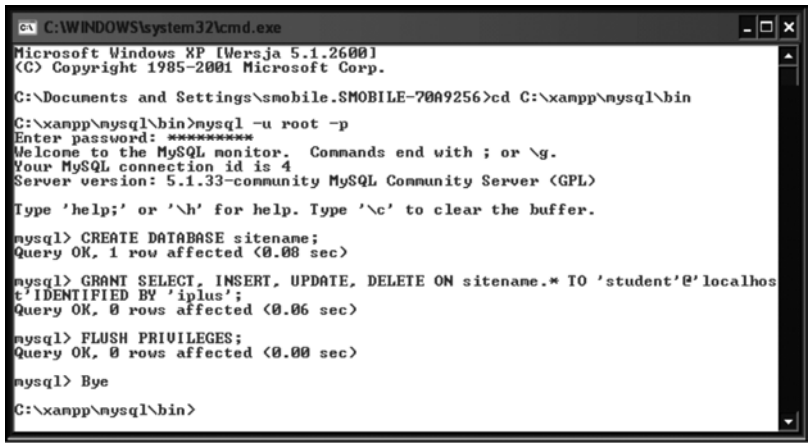
gdzie w miejsce słowa hasło wpisujemy nasze hasło). Po skonfigurowaniu hasła dla użytkownika root można utworzyć konta innych użytkowników, z dostępem do testowej bazy danych. Do utworzenia bazy danych o nazwie sitename posługujemy się poleceniem:

```
CREATE DATABASE sitename;
```

Następnie tworzymy konto użytkownika, z podstawowymi uprawnieniami dotyczącymi bazy danych, używając polecenia:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON sitename.* TO 'student'@'localhost'
IDENTIFIED BY 'haslo';.
```

Aby zatwierdzić wprowadzone zmiany używamy polecenia `FLUSH PRIVILEGES` – rys. 20.



Rysunek 20.
Tworzenie konta użytkownika w bazie danych MySQL

W celu przetestowania instalacji należy utworzyć dwa skrypty PHP. Pierwszy z nich o nazwie phpinfo.php sprawdza, czy oprogramowanie PHP jest aktywne i wyświetla dane o instalacji. Treść pliku phpinfo.php jest następująca:

```
<?php
phpinfo();
?>
```

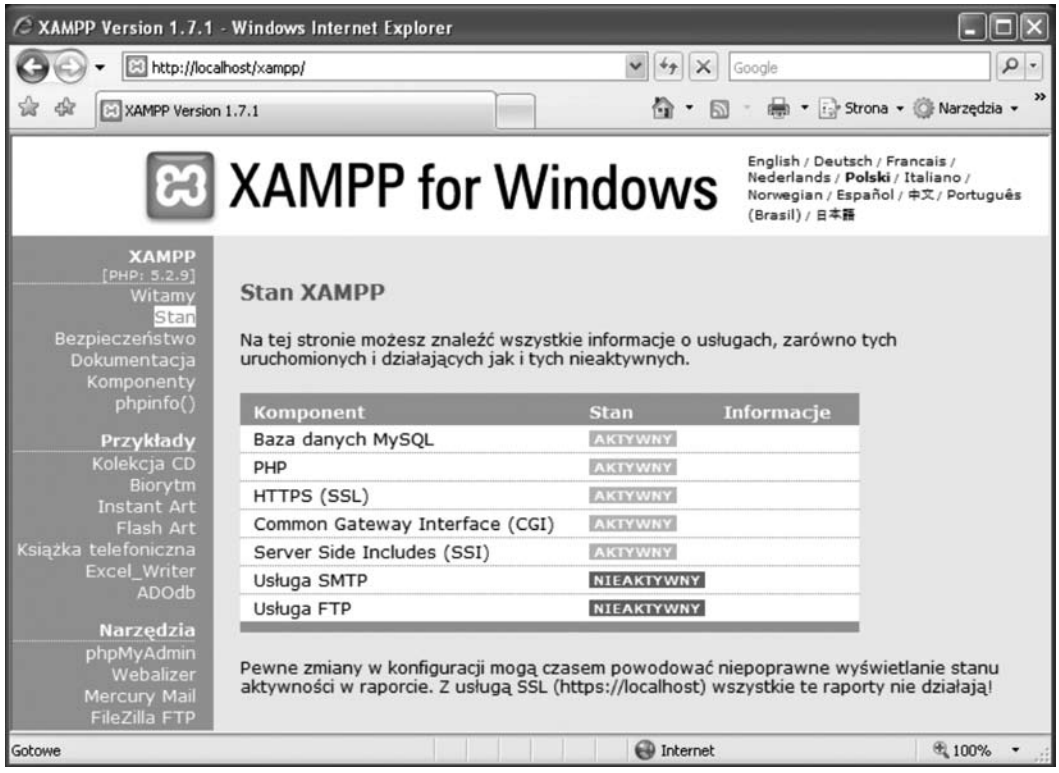
Plik możemy utworzyć w prosty edytorze tekstowym, np. w np. Notatniku. Efekt jego uruchomienia w przeglądarce będzie taki, jak na rys. 21.

Rysunek 21.
Informacja o zainstalowanej wersji oprogramowania PHP

Drugi plik testowy o nazwie mysli_test.php sprawdza połączenie z bazą danych MySQL. Treść pliku jest następująca:

```
<?php
mysqli _ connect ('localhost','student','haslo','sitename');
?>
```

Po uruchomieniu pliku w przeglądarce skrypt próbuje połączyć się z serwerem MySQL przy użyciu nazwy konta użytkownika i hasła. Jeśli proces przebiegnie poprawnie, zobaczymy pustą stronę. Jeśli wystąpią błędy, strona będzie zawierać stosowne komunikaty. Warto przyjrzeć się witrynie XAMPP, dostępnej po wpisaniu http://localhost/xampp/ w przeglądarce. Przy użyciu tej strony możemy np. sprawdzić stan działania środowiska programistycznego – rys. 22, jak również zapoznać się z dokumentacją, przykładowymi aplikacjami.



Rysunek 22.
Witryna XAMPP na serwerze lokalnym

3. TWORZENIE WŁASNEGO SERWISU WWW

Instrukcja 10. Mój własny serwis WWW – etap I. Każdy uczeń przystępuje do realizacji własnego projektu (tworzy poszczególne podstrony serwisu, w tym treść (język HTML), wygląd (język CSS) oraz sposób działania (język PHP).

W tej części każdy z uczniów pracuje nad własnym serwisem internetowym. W pierwszej fazie projektujemy serwis, definiujemy założenia i wymagania. Następnie tworzymy poszczególne fragmenty (podstrony) serwisu, po czy łączymy je w całość. Ostatnim etapem jest testowanie serwisu i poprawianie jego działania. Prowadzący koordynuje pracę uczniów i służy pomocą.

Od czego zacząć tworzenie złożonego projektu? Od rzeczy najłatwiejszych. W przypadku serwisu internetowego będzie to z pewnością jego treść, a w dalszej kolejności wygląd. Przy użyciu znaczników języka HTML, który jest stosunkowo łatwy do opanowania, możemy utworzyć poszczególne podstrony serwisu (strona główna, kontakt, pomoc, regulamin, strona rejestracji, strona logowania itp.). Wygląd tych strona możemy definiować przy użyciu kaskadowych arkuszy stylów – CSS. Natomiast najwięcej pracy pochłania zaprogramowanie działania poszczególnych podstron i powiązanie ich ze sobą.

Przy tworzeniu pojedynczych stron internetowych, ich struktury i treści, warto skorzystać z otwartych zasobów edukacyjnych. Wykłady i ćwiczenia są udostępnione pod adresem http://wazniak.mimuw.edu.pl/index.php?title=Aplikacje_WWW.

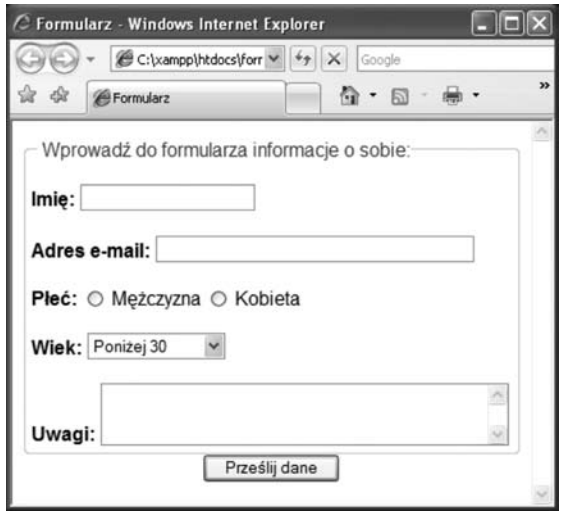
TREŚĆ – JĘZYK HTML

Informacje na temat wykorzystania znaczników HTML są udostępnione pod adresem http://wazniak.mimuw.edu.pl/index.php?title=AWWW-1st3.6-w01.tresc-1.0-toc, ćwiczenia utrwalające wiedzę – http://wazniak.mimuw.edu.pl/index.php?title=AWWW-1st3.6-l01.tresc-1.0, natomiast praktyczny kurs języka HTML można znaleźć na stronie http://www.kurshtml.boo.pl/html/zielony.html.

Ćwiczenie 1. W tym ćwiczeniu wykorzystamy język HTML do utworzenia formularza, w którym użytkownik będzie mógł umieścić informacje o sobie. W jednym z następnych ćwiczeń utworzymy skrypt w języku PHP do obsługi formularza. Aby utworzyć formularz, należy w edytorze tekstu utworzyć plik o nazwie formularz.html i o poniższej treści:

```
<html>
<head>
    <title>Formularz</title>
</head>
<body>
   <font face="Arial">
   <form action="obsluga _ formularza.php" method="post">
   <fieldset><legend>Wprowadź do formularza informacje o sobie:</legend>
  <p><b>Imię:</b> <input type="text" name="name" size="20" max-length="40"
/></p>
  <p><b>Adres e-mail:</b> <input type="text" name="email" size="40" max-
length="60" /></p>
  <p><b>Płeć:</b> <input type="radio" name="gender" value="M" /> Mężczy-
zna <input type="radio" name="gender" value="K" /> Kobieta</p>
   <p><b>Wiek:</b>
   <select name="age">
      <option value="ponizej30">Poniżej 30</option>
      <option value="miedzy30a60">Między 30 a 60</option>
      <option value="powyzej60">Powyżej 60</option>
   </select></p>
    <p><b>Uwagi:</b> <textarea name="comments" rows="3" cols="40"></te-
xtarea></p>
   </fieldset>
    <div align="center"><input type="submit" name="submit" value="Prze-
ślij dane" /></div>
   </form>
</body>
</html>
```

Efekt wyświetlenia formularza w przeglądarce powinien być jak na rys. 23.



Rysunek 23.
Formularz HTML

WYGLĄD – JĘZYK CSS

Kaskadowe arkusze stylów (CSS) stanowią uzupełnienie języka HTML. Podczas gdy HTML definiuje strukturę i treść strony, CSS odpowiada za formatowanie jego wyglądu. Informacje na temat wykorzystania kaskadowych arkuszy stylów, uzupełnione o ćwiczenia można znaleźć pod adresami przytoczonymi w poprzednim punkcie. Kaskadowe arkusze stylów zostaną wykorzystane do formatowania wyglądu witryny serwisu repozytorium.

SPOSÓB DZIAŁANIA – JĘZYK PHP

Zaprogramowanie działania serwisu internetowego jest niewątpliwie największym wyzwaniem dla początkującego twórcy stron internetowych. Jednakże kiedy poświęcimy czas na zrozumienie działania języka skryptowego (w naszym przypadku jest to PHP), opanujemy tę umiejętność. Aby zapoznać się z podstawowymi pojęciami i logiką języka PHP, warto posłużyć się kursem dostępnym w Internecie: http://phpkurs.pl/. Ćwiczenia można znaleźć pod adresem http://wazniak.mimuw.edu.pl/index.php?title=AWWW-1st3.6-l06.tresc-1.0.

Ćwiczenie 2. Przed przystąpieniem do tworzenia dynamicznych stron internetowych, należy nauczyć się przesyłać dane do przeglądarki. Jedną z najpopularniejszych funkcji służących do tego celu jest funkcja echo(). Aby przesłać dane do przeglądarki należy wykonać kroki, które następują.

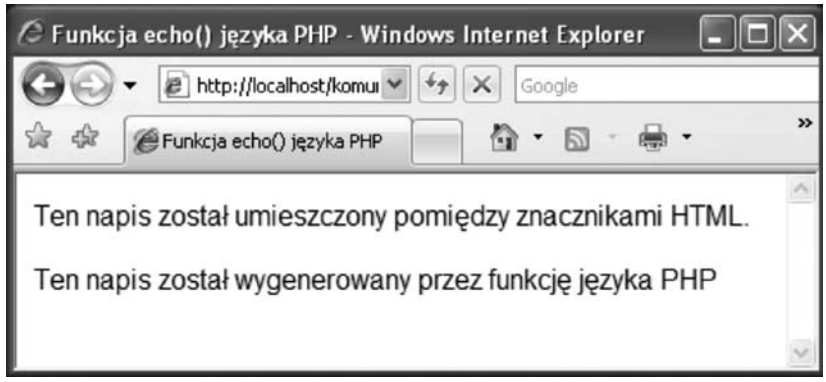
1. W edytorze tekstu utworzyć plik o nazwie komunikat.php,
2. W języku HTML utworzyć treść pliku jak poniżej:

```
<html>
<head>
   <title> Funkcja echo() języka PHP</title>
</head>
<body>
   <font face="Arial">
   <p> Ten napis został umieszczony pomiędzy znacznikami HTML.</p>
</body>
</html>
```

3. W treści pliku umieścić instrukcję echo() języka PHP o następującej treści:

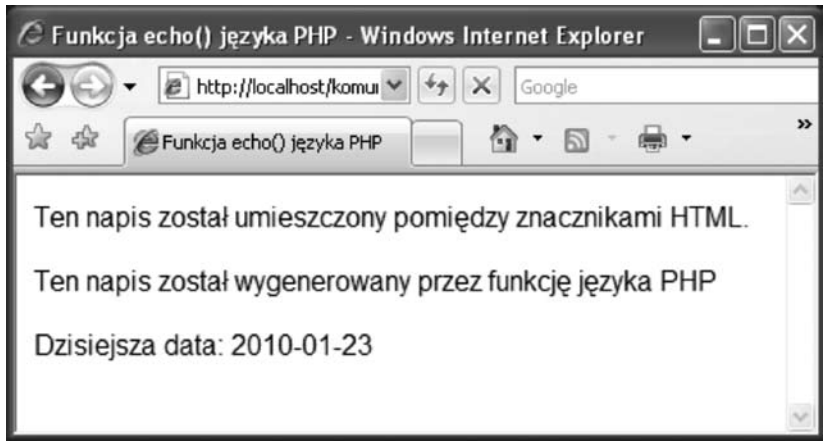
```
<?php
   echo 'Ten napis został wygenerowany przez funkcję języka PHP';?>
```

4. Po zapisaniu zmian w pliku, należy uruchomić go w przeglądarce internetowej. Efekt powinien być podobny jak na rys. 24



Rysunek 24.
Działanie funkcji echo języka PHP

Ćwiczenie 3. W tym ćwiczeniu utworzymy pierwszy dynamiczny element na stronie – datę jej wyświetlenia (rys. 25). Zmodyfikuj kod pliku komunikat.php, dodając w odpowiednim miejscu wiersz:
`echo date(„Y-m-d”);`



Rysunek 25.
Strona wyświetla aktualną datę

Ćwiczenie 4. W tym ćwiczeniu utworzymy skrypt w języku PHP do obsługi formularza z ćwiczenia 1. W tym celu należy w edytorze tekstu utworzyć plik o nazwie obsluga_formularza.php i następującej treści.

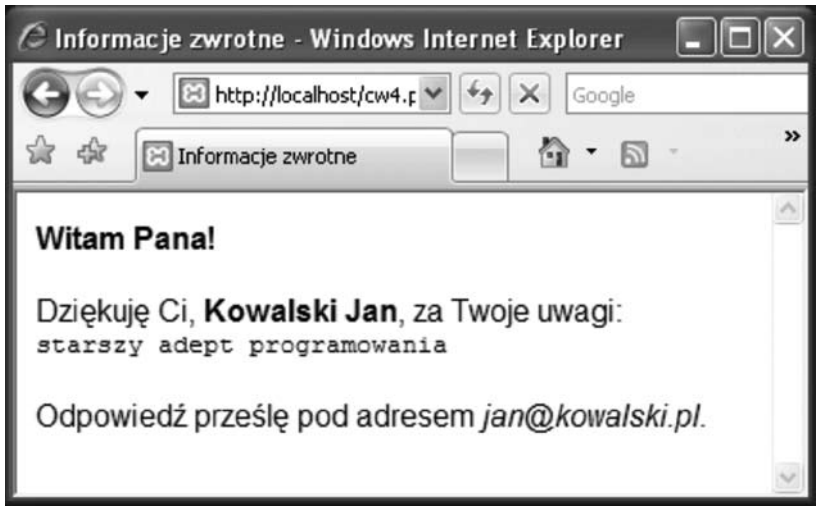
```
<html>
<head>
   <title>Informacje zwrotne</title>
</head>
<body>
<?php

// Utwórz skróty zmiennych formularza.
$name = $_REQUEST['name'];
$email = $_REQUEST['email'];
$comments = $_REQUEST['comments'];

// Wyświetl otrzymane informacje.
echo „<p>Dziękuję Ci, <b>$name</b>, za Twoje uwagi:<br />
<tt>$comments</tt></p>
<p>Odpowiedź prześlę pod adresem <i>$email</i>.</p>\n”;
```

```
?>
</body>
</html>
```

Po wprowadzeniu danych przez użytkownika, zostanie wyświetlona strona jak na rys. 26.

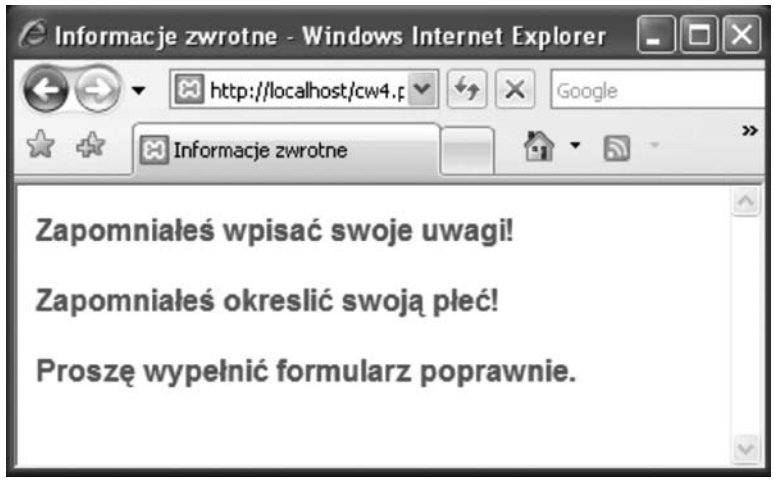
Rysunek 26.
Informacja zwrotna do użytkownika

Ćwiczenie 5. Ćwiczenie polega na zmodyfikowaniu skryptu obsluga_formularza.php tak, aby sprawdzał, jakiego wyboru dokonał użytkownik – i w zależności od tego – wyświetlał odpowiednie dane. W ćwiczeniu należy wykorzystać instrukcje warunkowe i operatory języka PHP. Przykład możliwej modyfikacji skryptu obsluga_formularza.php – kod źródłowy w wersji elektronicznej znajduje się w pliku o nazwie cw4_obsluga_formularza.php. Efekt działania powinien być podobny, jak na rys. 27



Rysunek 27.
Działanie instrukcji warunkowych języka PHP

Ćwiczenie 6. Kolejną modyfikacją skryptu obsługi formularza będzie weryfikacja danych wprowadzonych przez użytkownika w formularzu. Mechanizm weryfikacji danych jest nieodłącznym elementem dynamicznych serwisów internetowych. Zmodyfikowany skrypt do obsługi może mieć treść jak w pliku cw5_obsluga_formularza.php. W momencie przesłania niepełnego formularza powinien zostać wyświetlony komunikat jak na rys. 28.



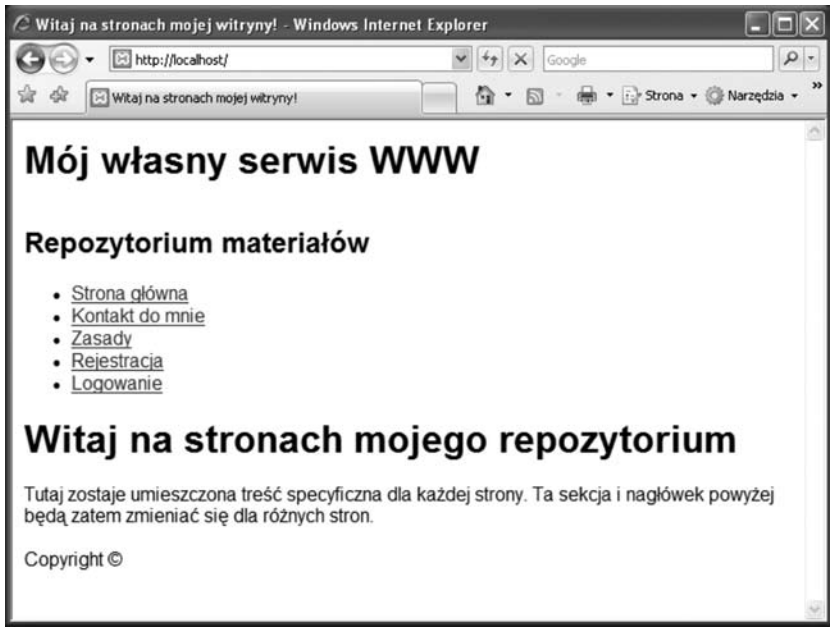
Rysunek 28.
Komunikaty do użytkownika

3.1. TWORZENIE SERWISU REPOZYTORIUM

Po tej 'rozgrzewce' przystąpimy do tworzenia własnego serwisu WWW. Serwis będzie składać się z wielu stron, dostępnych po kliknięciu w menu. Każda ze stron wchodzących w skład serwisu jest niezależnym plikiem HTML lub PHP. Dlatego w tym miejscu zajmiemy się zagadnieniem dołączania plików.

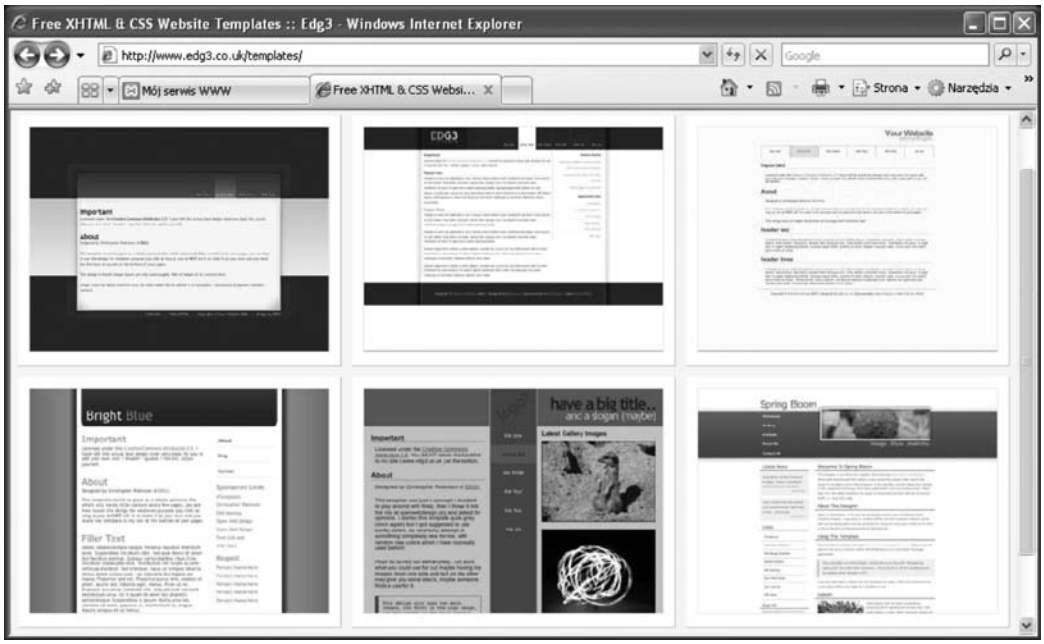
W dalszej kolejności przejdziemy do ważnego zagadnienia jakim jest korzystanie z plików zewnętrznych. Utworzymy serwis internetowy składający się z kilku podstron. Będzie to szablon serwisu, który można dowolnie modyfikować. Dołączanie plików zewnętrznych działa w ten sposób, że cała zawartość pliku o podanej nazwie zostaje włączana w miejscu odwołania do tego pliku.

Ćwiczenie 7. W tym ćwiczeniu, wykorzystując mechanizm dołączania plików, oddzielimy kod HTML odpowiedzialny za formatowanie tekstu od kodu PHP. W pierwszym kroku, przy użyciu języka HTML, należy zaprojektować wygląd aplikacji, tzw. stronę główną. Przykładowy układ w kodzie HTML zawarto w pliku cw6_index.html. Strona o powyższym kodzie, nazwana index.html, wygląda jak na rys. 29.



Rysunek 29.
Strona główna tworzonego serwisu

Aby nadać utworzonej witrynie bardziej atrakcyjny wygląd, posłużymy się arkuszami stylów CSS. Do sformatowania wyglądu strony użyto arkuszy stylów CSS, udostępnionych na stronie **www.edg3.co.uk.** (http://www.edg3.co.uk/templates/) – rys. 30. Więcej propozycji arkuszy stylów znajdziemy na stronie http://www.opendesigns.org/.

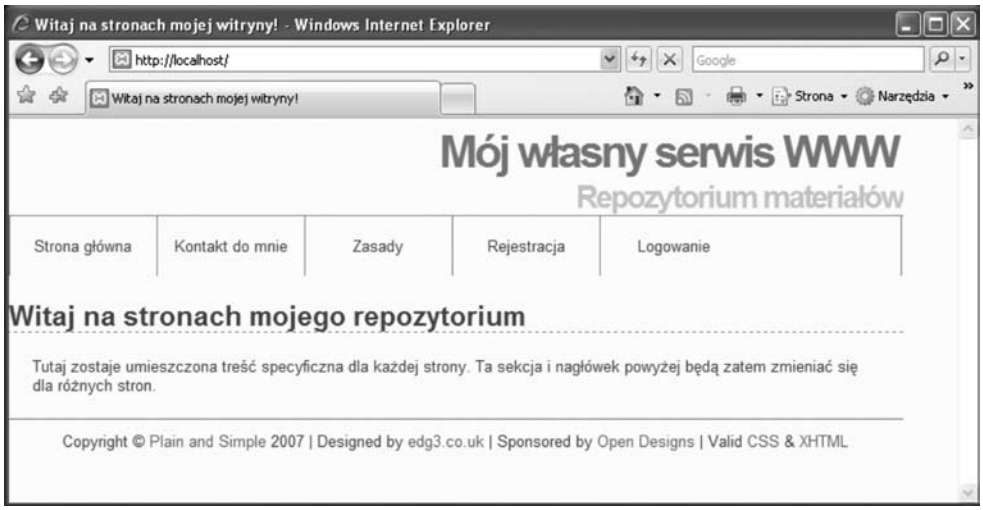


Rysunek 30.
Wzory arkuszy stylów dostępne w Internecie

Należy pobrać plik o rozszerzeniu .css i umieścić go w katalogu includes utworzonym w katalogu witryn WWW na serwerze. Następnie w nagłówku strony index.html należy dodać wiersz:

```
<link rel="stylesheet" href="includes/style.css" type="text/css" media="screen" />
```

będący odwołaniem to arkusza stylów. Po dołączeniu pliku style.css (kod źródłowy w pliku o nazwie cw6_style.css). Po zastosowaniu stylów CSS strona wygląda jak na rys. 31 (na dole strony umieściliśmy informację o autorze szablonu witryny).



Rysunek 31.
Strona główna sformatowana przy użyciu CSS

Ćwiczenie 8. Dobrym pomysłem jest wyodrębnienie nagłówka i stopki jako zewnętrznych plików. Dzięki temu nie będzie trzeba powielać tego samego kodu na wszystkich podstronach serwisu. W katalogu includes tworzymy skrypt o nazwie naglowek.html i treści jak w pliku cw7_naglowek.html. Tworzymy skrypt o nazwie stopka.html i treści jak w pliku cw7_stopka.html. Na końcu należy utworzyć stronę główną, która będzie odwoływać się do plików składowych. Strona ta, o nazwie index.php będzie mieć treść jak w pliku cw7_index.php.

Menu utworzonej witryny zawiera kilka łączy. Łącze Strona domowa prowadzi do strony głównej (wskazuje plik index.php). Kolejne łącza posłużą do rejestrowania użytkowników oraz logowania. Mogą także wskazywać na inne podstrony, np. informacje o nas, galeria zdjęć, kontakt itp. Każdą z tych podstron tworzymy jako oddzielny plik, analogicznie jak stronę index.php

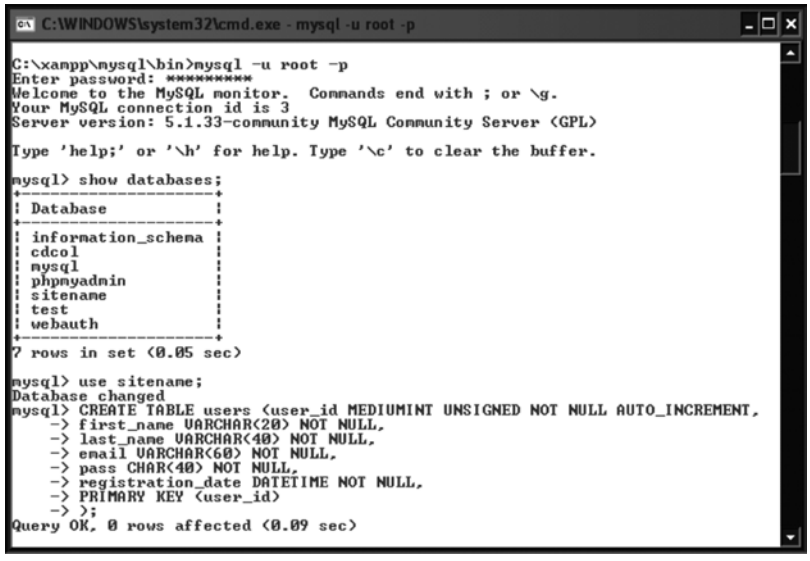
3.2. BAZA DANYCH, JĘZYK SQL

Instrukcja 11. Mój własny serwis WWW – etap II. Projektowanie i tworzenie bazy danych (baza danych MySQL, język SQL). W tej części uczniowie zapoznają się z metodyką tworzenia i funkcjonowania relacyjnej bazy danych. W tym celu posłużą się materiałami dostępnymi w ramach Otwartych Zasobów Edukacyjnych na portalu http://wazniak.mimuw.edu.pl.

Z problematyką projektowania bazy danych zapoznamy się korzystając z otwartych zasobów edukacyjnych, udostępnionych pod adresem http://wazniak.mimuw.edu.pl/index.php?title=Bazy_danych. Każdy z uczniów zaprojektuje tabele w bazie danych i relacje pomiędzy nimi na potrzeby swojego serwisu.

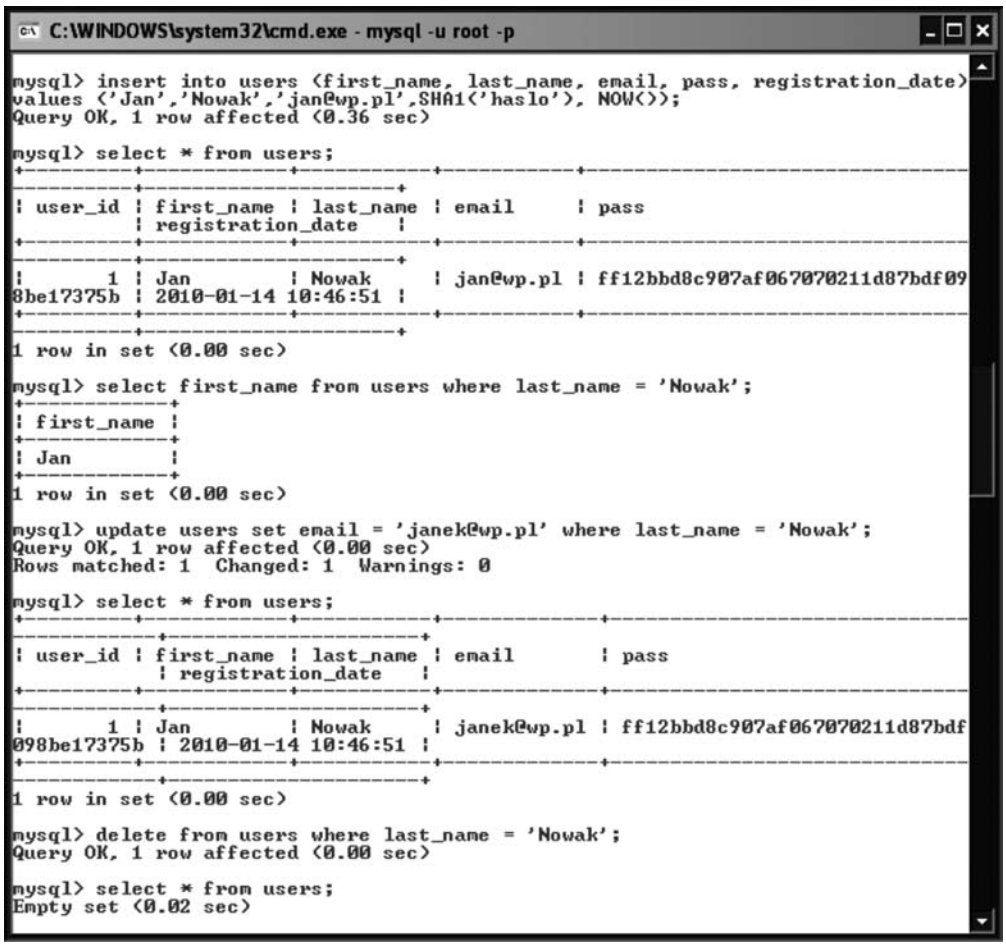
W tej części uczniowie, posługując się poleceniami języka SQL, utworzą własną bazę danych i tabele w tej bazie. Kolejnym krokiem będzie wprowadzanie rekordów do tabel oraz ich modyfikowanie i usuwanie. Najważniejsze z punktu widzenia tworzonego serwisu będzie formułowanie zapytań, umożliwiających odczytywanie informacji z tabel w bazie danych.

Ćwiczenie 9. Poniższe ćwiczenia ilustrują podstawowe operacje na bazie danych. Z menu Start systemu Windows wybieramy Uruchom, wpisujemy cmd i klikamy OK. Przechodzimy do odpowiedniego katalogu (cd C:\xampp\mysql\bin), logujemy się do bazy danych, a następnie tworzymy tabelę (polecenie `CREATE TABLE`) – rys. 32.



Rysunek 32.
Tworzenie tabeli w bazie danych

Następnie przeprowadzamy operacje wprowadzania informacji do bazy danych, odczytywania wprowadzonych informacji, modyfikowania ich oraz usuwania – rys. 33.



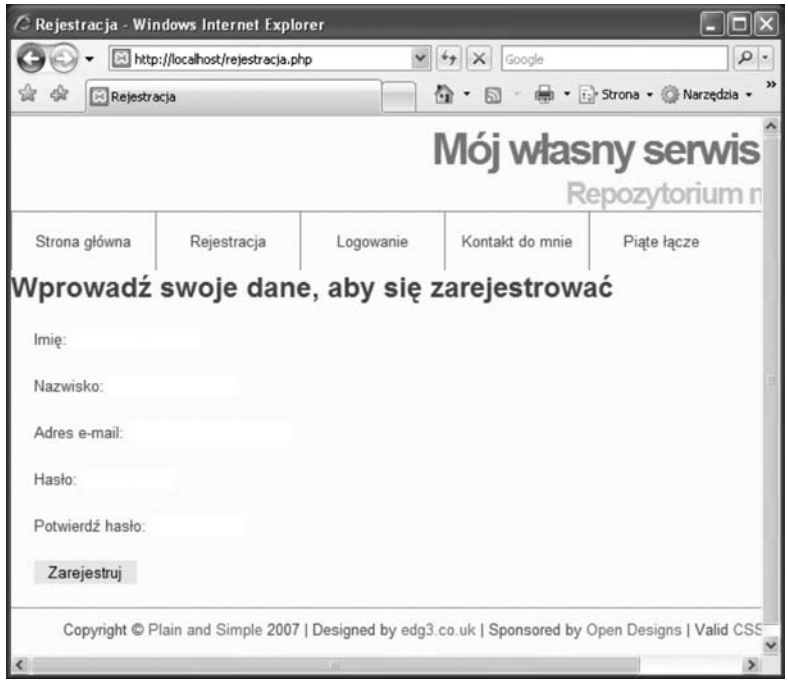
Rysunek 33.
Operacje na bazie danych

3.3. WSPÓŁPRACA PHP Z MYSQL

Instrukcja 12. W tej części uczniowie nauczą się operowania na bazie danych MySQL z poziomu skryptów PHP, czyli odczytywanie oraz modyfikowanie informacji zgromadzonych w bazie danych, bezpośrednio ze strony internetowej.

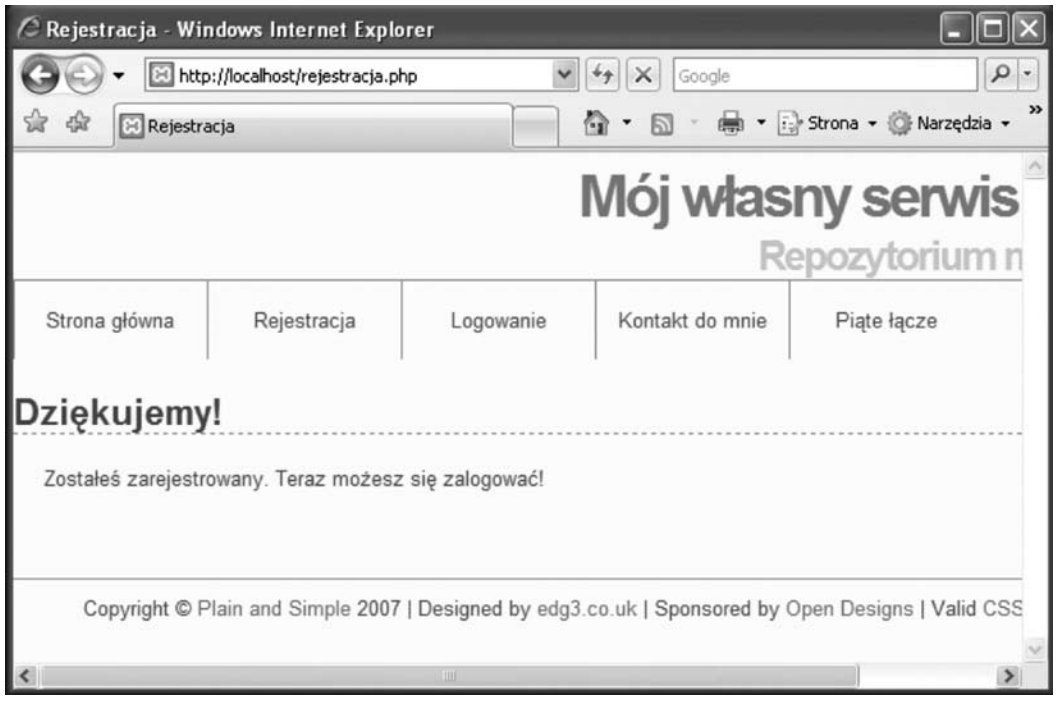
Współpraca PHP z MySQL jest potrzebna m.in. do obsługi logowania użytkownika oraz do wpro-wadzania treści przez użytkowników. Zanim jednak użytkownik będzie mógł się zalogować, musi się zarejestrować – jego login i hasło muszą znaleźć się w bazie danych.

Ćwiczenie 10. Utworzymy kolejną część naszego serwisu – aplikację umożliwiającą rejestrowanie użytkowników. Za komunikację z bazą danych odpowiada skrypt musql_connect.php. Plik ten umieszczamy w katalogu nadrzędnym w stosunku do katalogu witryny. Ma on postać jak w pliku cw9_mysql_connect.php. Skrypt rejestracja.php, odpowiedzialny za rejestrowanie użytkowników, ma postać jak w pliku cw9_rejestracja.php. Strona rejestracji w przeglądarce ma postać jak na rys. 34.

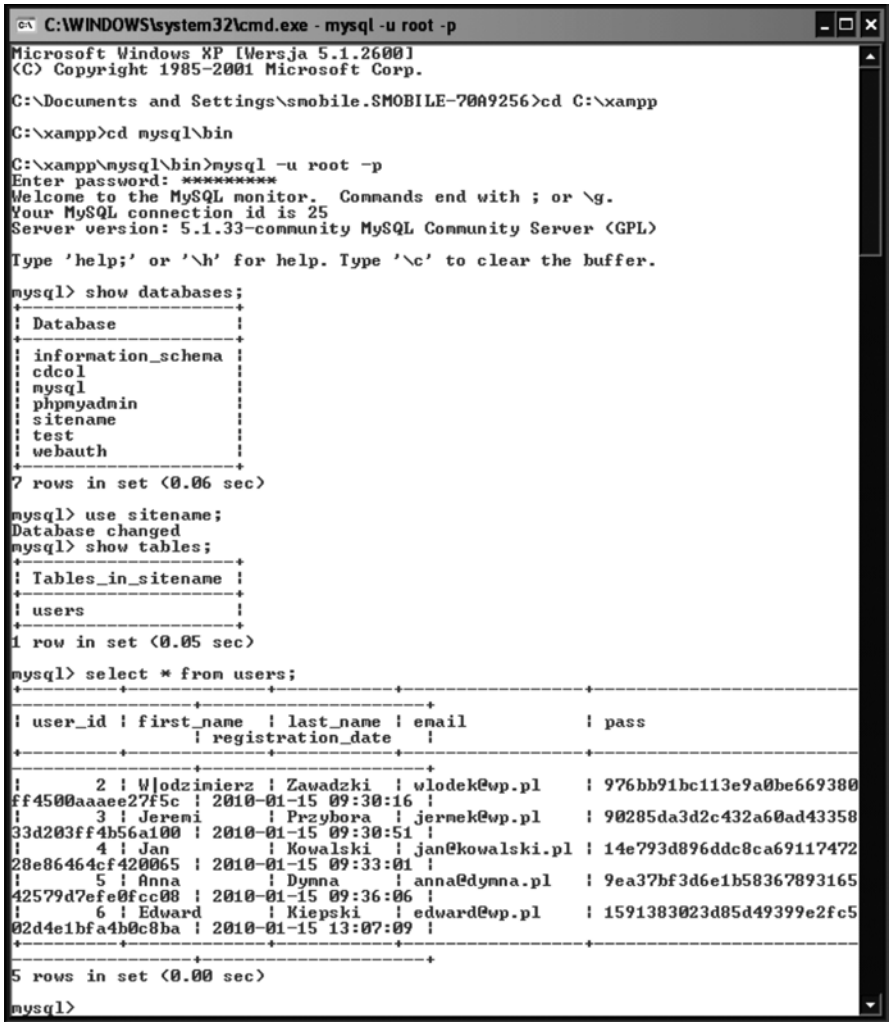


Rysunek 34.
Rejestracja

Po poprawnym wypełnieniu formularza rejestracyjnego, wyświetlane jest podziękowanie (rys.35). W bazie danych sprawdzamy czy użytkownik rzeczywiście został dodany – rys. 36:

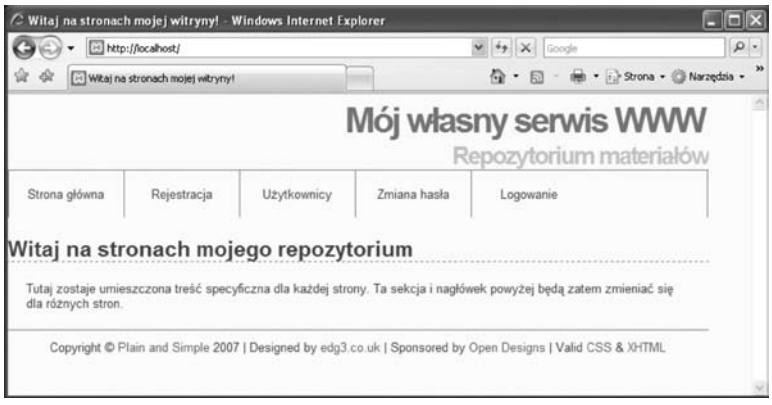


Rysunek 35.
Komunikat wyświetlany po pomyślnej rejestracji



Rysunek 36.
Podgląd bazy danych – tabela users, czyli użytkownicy

Ćwiczenie 11. W tym ćwiczeniu utworzymy skrypt do pobierania listy użytkowników z bazy danych. W menu witryny utworzymy opcję Użytkownicy. Należy zmodyfikować plik nagłówka (naglowek.php) tak, aby trzecim łączem było Użytkownicy, czwartym – Zmiana hasła, a piątym – Logowanie – rys. 37.

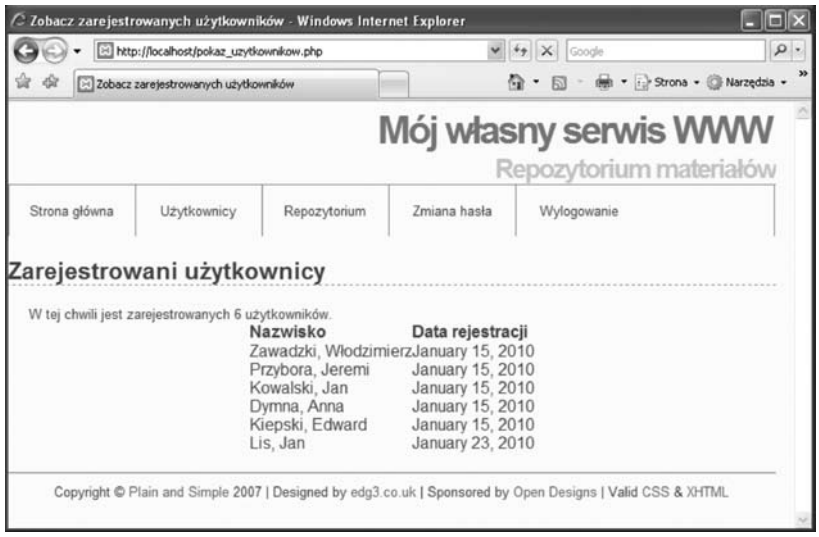


Rysunek 37.
Strona główna serwisu

W odpowiednim miejscu pliku nagłówka umieszczamy fragment kodu:

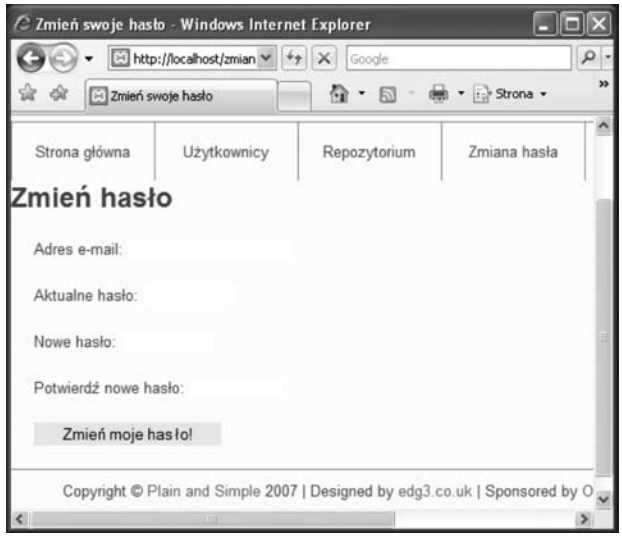
```
<li><a href="index.php">Strona główna</a></li>
      <li><a href="rejestracja.php">Rejestracja</a></li>
      <li><a href="pokaz _ uzytkownikow.php">Użytkownicy</a></li>
      <li><a href="zmiana _ hasla.php">Zmiana hasła</a></li>
      <li><a href="login.php">Logowanie</a></li>
      <!-- <li><a href="kontakt.html">Kontakt do mnie</a></li>
        <li><a href="#">Piąte łącze</a></li>
       -->
```

Skrypt pokaz_uzytkownikow.php odpowiedzialny za wyświetlanie listy zarejestrowanych użytkowników ma postać jak w pliku o nazwie cw10_pokaz_uzytkownikow.php. Działanie skryptu przedstawiono na rys. 38.

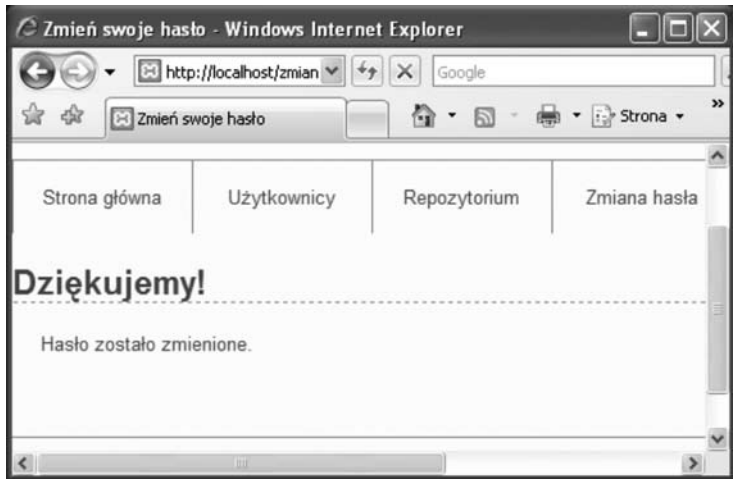


Rysunek 38.
Lista zarejestrowanych użytkowników

Ćwiczenie 12. Należy utworzyć plik umożliwiający użytkownikowi zmianę hasła – zmiana_hasla.php. Skrypt taki może mieć postać jak w pliku cw11_zmiana_hasla.php. Efekty działania skryptu zmiany hasła przedstawiono na rys. 39 oraz 40.



Rysunek 39.
Zmiana hasła

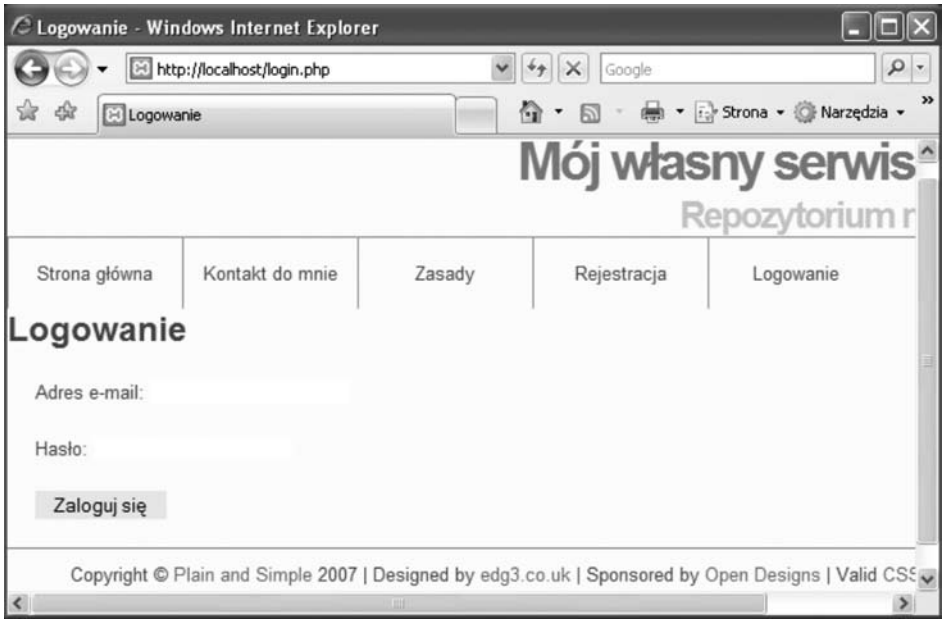


Rysunek 40.
Zmiana hasła – cd.

Ćwiczenie 13. Kolejny etap tworzenia serwisu polega na obsłudze logowania – wykorzystaniu sesji. Mechanizm sesji służy do przypisania użytkownikowi identyfikatora, z którym można powiązać pewne informacje, np.:

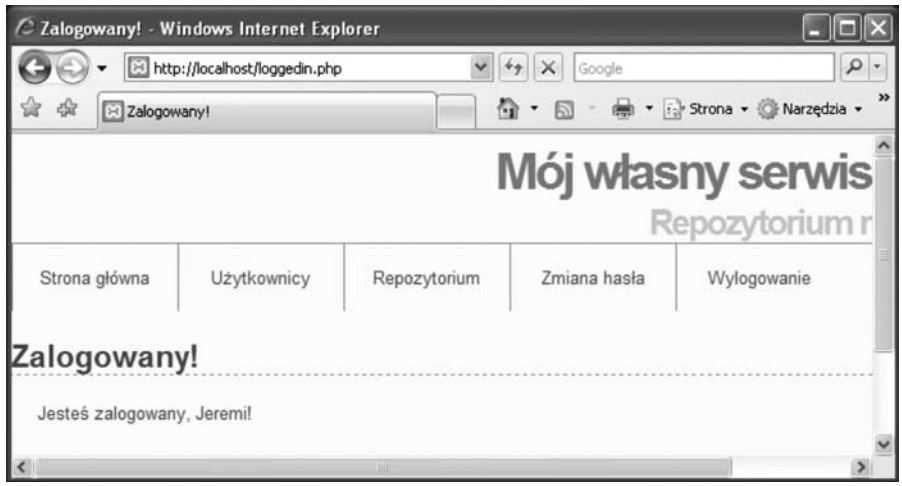
- strony odwiedzane przez użytkownika,
- artykuły dodane do koszyka w sklepie internetowym,
- status użytkownika – zalogowany bądź nie.

Kiedy użytkownik odwiedza stronę obsługującą sesje, otrzymuje swój unikatowy identyfikator sesji. Stan sesji jest przechowywany w tablicy w pliku tymczasowym. Skrypt login.php może mieć treść jak w pliku cw12_login.php. Działanie tego pliku w przeglądarce przedstawiono na rys. 41.



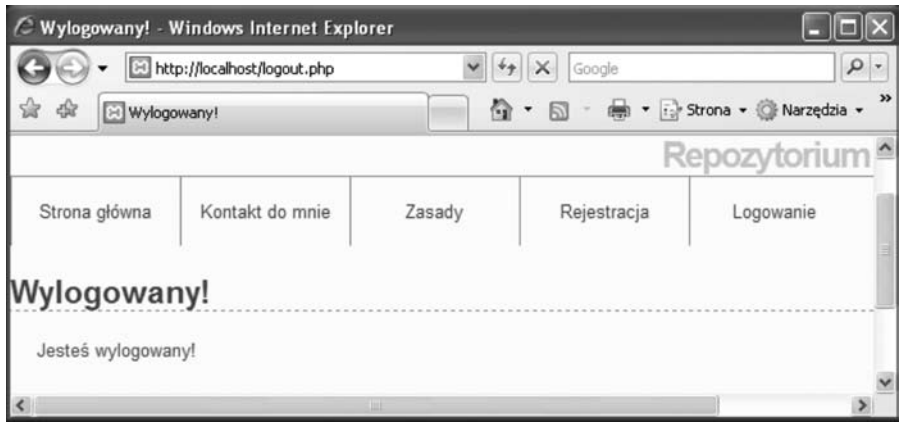
Rysunek 41.
Logowanie

Skrypt wyświetlający komunikat po zalogowaniu (rys. 42) – loggedin.php ma treść jak w pliku cw12_loggedin.php.



Rysunek 42.
Po zalogowaniu

Należy zmodyfikować plik naglowek.php tak, aby mogło zostać wyświetlone łącze umożliwiające wylogowanie. Zmodyfikowany skrypt ma postać jak w pliku cw12_naglowek.php. Po zalogowaniu w menu są dostępne inne opcje – rys 42. Skrypt logout.php, umożliwiający wylogowanie ma treść jak w pliku cw12_logout.php. Działanie skryptu logout.php przedstawiono na rys. 43.



Rysunek 43.
Po wylogowaniu

Skrypt mysql_conect.php uzupełniamy o funkcję escape_data, przetwarzające dane wprowadzane do bazy danych. Skrypt ma postać jak w pliku cw12_mysql_connect.php.

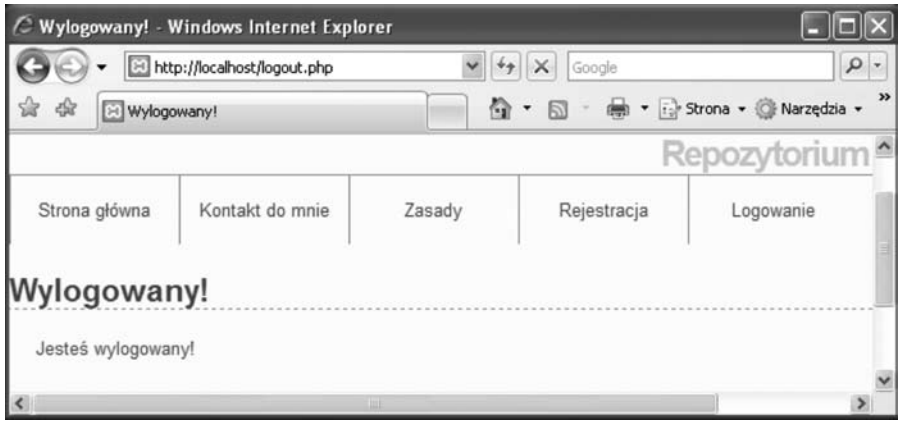
Ćwiczenie 14. W tym momencie oddzielimy strony publiczne od tych dostępnych po zalogowaniu.

Zanim użytkownik się zaloguje, będzie widział łącza:
- Strona główna,
- Kontakt do mnie,
- Zasady,
- Rejestracja,
- Logowanie.

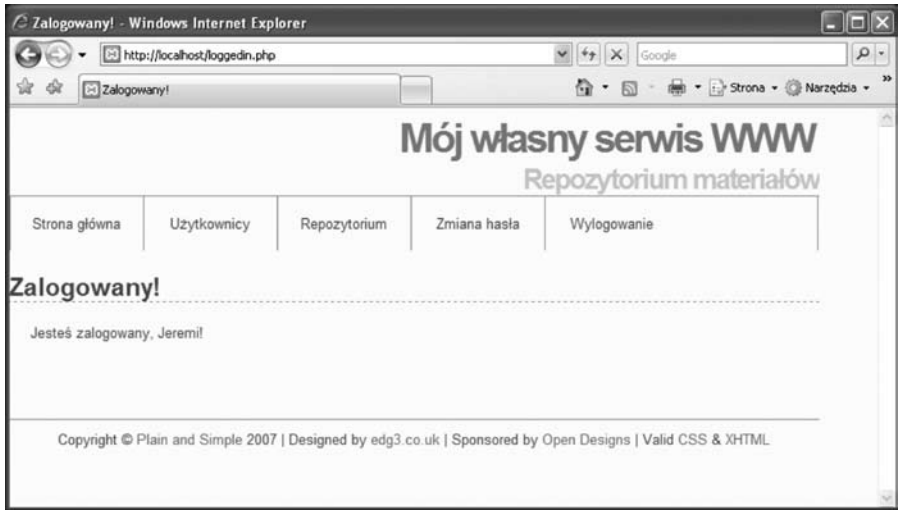
Natomiast po zalogowaniu menu będzie wyglądać następująco:
- Strona główna,
- Użytkownicy,
- Repozytorium,

- Zmiana hasła,
- Wylogowanie.

Pierwszym krokiem będzie zmodyfikowanie pliku naglowek.html – treść znajduje się w pliku cw13_naglowek.html. Efekt działania skryptu przedstawiono na rys. 44, a po zalogowaniu – na rys. 45.



Rysunek 44.
Menu przed zalogowaniem



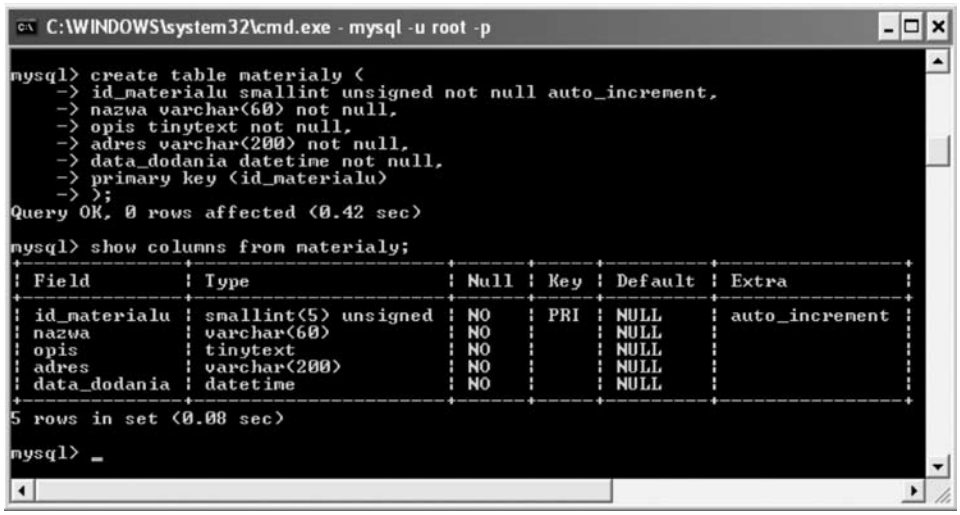
Rysunek 45.
Menu po zalogowaniu

Aby po zalogowaniu menu pozostało nie zmienione, należy w każdym pliku (index.php, pokaz_uzytkownikow.php, repozytorium.php, zmiana_hasla.php) w pierwszym wierszu umieścić funkcję session_start(); oznaczającą rozpoczęcie sesji.

Ćwiczenie 15. Repozytorium będzie zbiorem informacji o zasobach (w wersji elektronicznej i nie tylko). Każdy materiał będzie opisany przez:

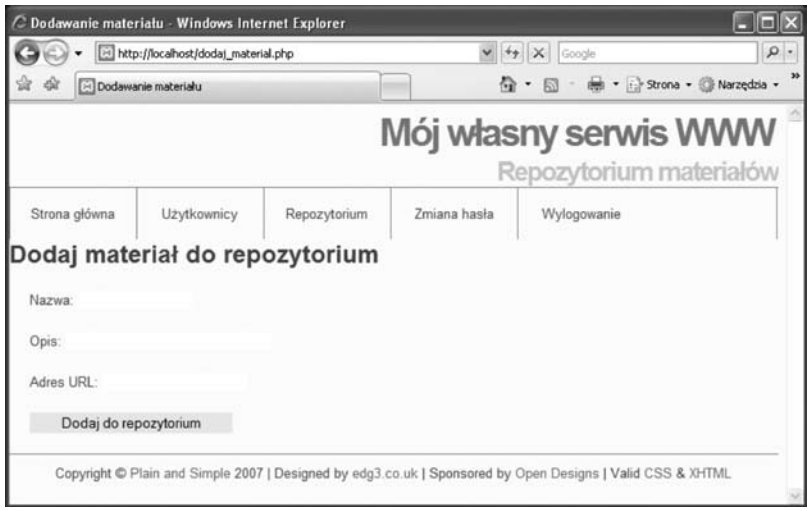
- Nazwę (tytuł),
- Opis (dokładniejsze informacje),
- Adres_url (jeśli materiał jest dostępny przez Internet),
- Datę dodania do bazy danych (rys. 46).

Pierwszym krokiem będzie utworzenie odpowiedniej tabeli o nazwie materialy w bazie danych. Następnie tworzymy skrypt repozytorium.php (plik cw14_repozytorium.php).

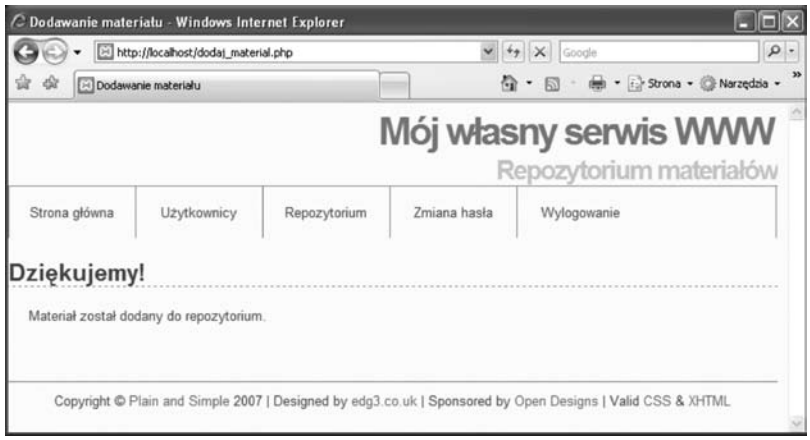


Rysunek 46.
Tabela materialy w bazie danych

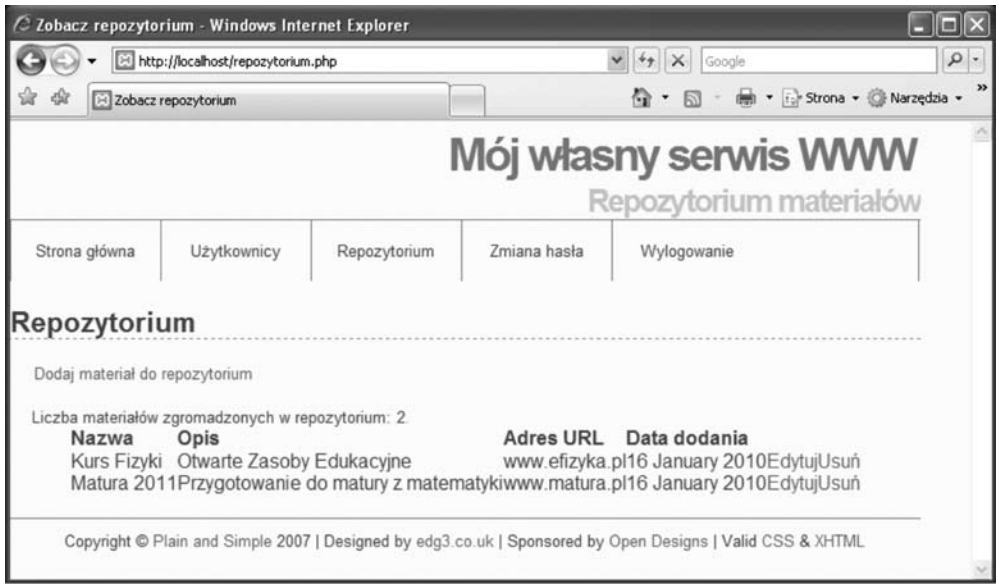
Ćwiczenie 16. Tworzymy skrypt umożliwiający dodawanie materiałów do repozytorium (rys. 47) – dodaj_material.php. Treść jak w pliku cw15_dodaj_material.php.



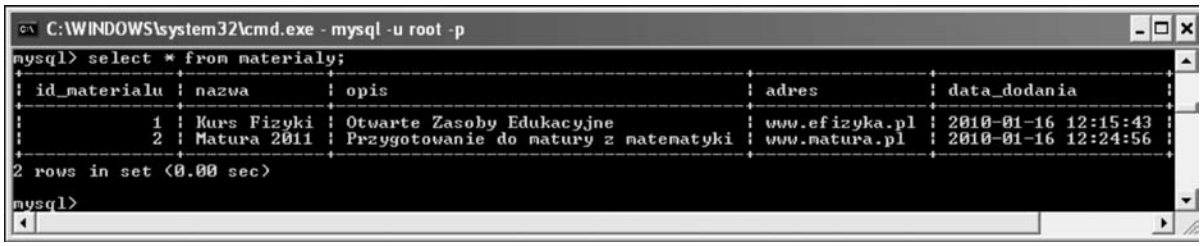
Rysunek 47.
Formularz dodawania materiału do repozytorium



Rysunek 48.
Po dodaniu materiału do repozytorium

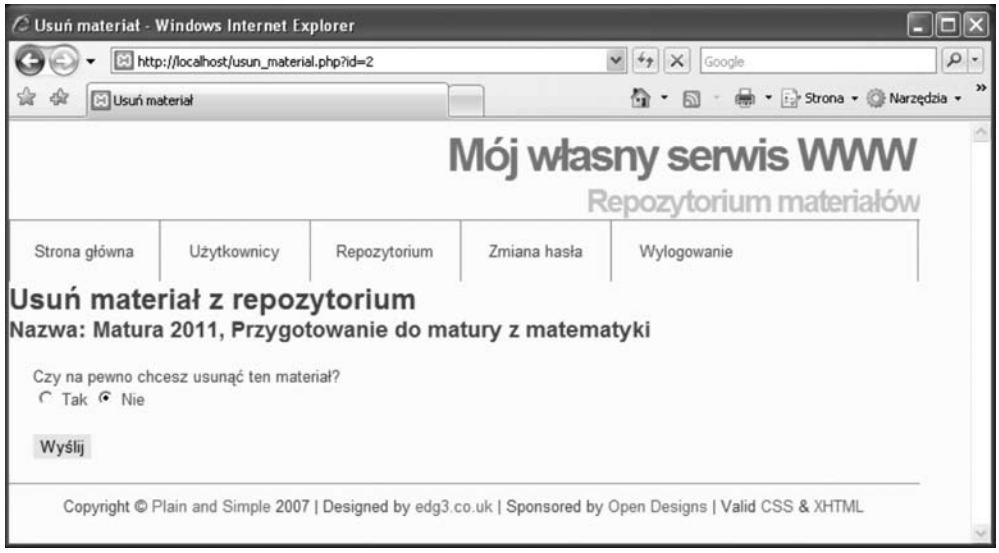


Rysunek 49.
Stan repozytorium po dodaniu nowego materiału

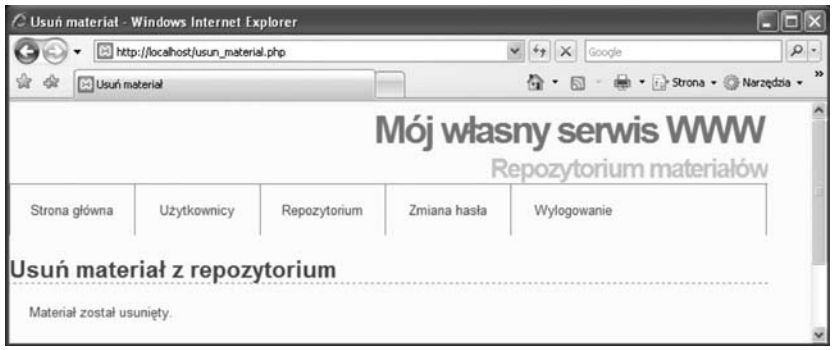


Rysunek 50.
Sprawdzenie stanu bazy danych

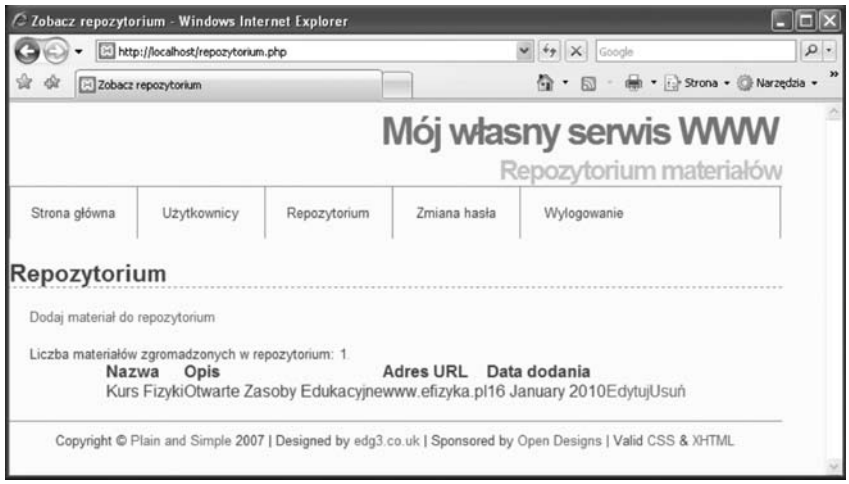
Ćwiczenie 17. Pozostaje już tylko utworzyć strony do edytowania (rys. 54) i usuwania (rys. 51) materiałów. Treść skryptu usun_material.php znajduje się w pliku cw16_usun_material.php.



Rysunek 51.
Usuwanie materiałów



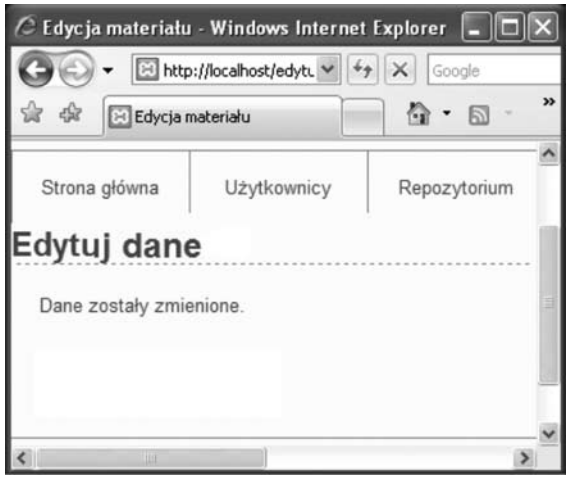
Rysunek 52.
Usuwanie materiałów – cd.



Rysunek 53.
Łącza edycji i usuwania materiałów w repozytorium

Ćwiczenie 18. Ostatnim elementem, niezbędnym do sprawnego działania serwisu, jest utworzenie skryptu umożliwiającego edycję materiałów w bazie danych – edytuj_material.php. Treść jak w pliku cw17_edytuj_material.php.

Rysunek 54.
Edycja materiału



Rysunek 55.
Po edycji materiału

3.4. TESTOWANIE, POPRAWIANIE I PREZENTOWANIE WŁASNEGO SERWISU INTERNETOWEGO

Instrukcja 13. Mój własny serwis WWW – etap III. W tej części każdy z uczniów przeprowadza testowanie według instrukcji zamieszczonych poniżej. W momencie wystąpienia błędu, należy go natychmiast poprawić.

Częstym błędem początkujących twórców stron internetowych jest zbagatelizowanie problemu testowania swojego dzieła. Testowanie działania stron i ich poprawianie zajmuje co najmniej 25% czasu i wysiłku poświęconego na realizację projektu. Wynika to z faktu, że niezwykle rzadko udaje się stworzyć projekt idealny – niewymagający żadnych poprawek.

Testowanie zaprojektowanej aplikacji i weryfikacja poprawności jej działania powinny się odbywać równolegle do pisania kodu. Tworzenie projektu może się odbywać się w kilku etapach, a pojedyncze programy, realizujące np. obsługę logowania użytkowników, wpisywanie informacji do bazy, wyświetlanie danych, powinny być pisane i testowane na bieżąco i niezależnie od siebie. Po ukończeniu tworzenia serwisu należy przetestować go raz jeszcze – jako całość.

Należy oszacować, ile osób w skrajnym przypadku jednocześnie będzie chciało się zalogować. Tylu użytkowników powinno mieć zapewniony dostęp do narzędzia. Rzeczywista wydajność musi być większa i być gwarancją poprawnego działania na wypadek innych, nieprzewidzianych okoliczności.

Koniecznym wymogiem stawianym przed wszelkiego typu aplikacjami sieciowymi jest zapewnienie monitorowania działania, wykrywania błędów i ich szybkiej naprawy. Raz zaimplementowana na serwerze baza danych wraz z aplikacją działa samodzielnie, bez konieczności nieustannej kontroli. Wskazana jest natomiast kontrola okresowa. Użytkownicy dawno zarejestrowani (data rejestracji użytkownika jest przechowywana w bazie danych) mogą być po pewnym czasie (np. po kilku latach) administracyjnie usunięci z bazy, a przy próbie zalogowania zobaczą komunikat: Twoje konto wygasło. Zarejestruj się ponownie.

Testowanie wydajności należy przeprowadzić w celu sprawdzenia, czy zaprojektowana aplikacja jest na tyle wydajna, aby sprawnie działać. W dalszej części opisano przykładowe wyniki testowania.

Serwer, na którym dokonywano pomiarów, charakteryzował się następującymi parametrami technicznymi:

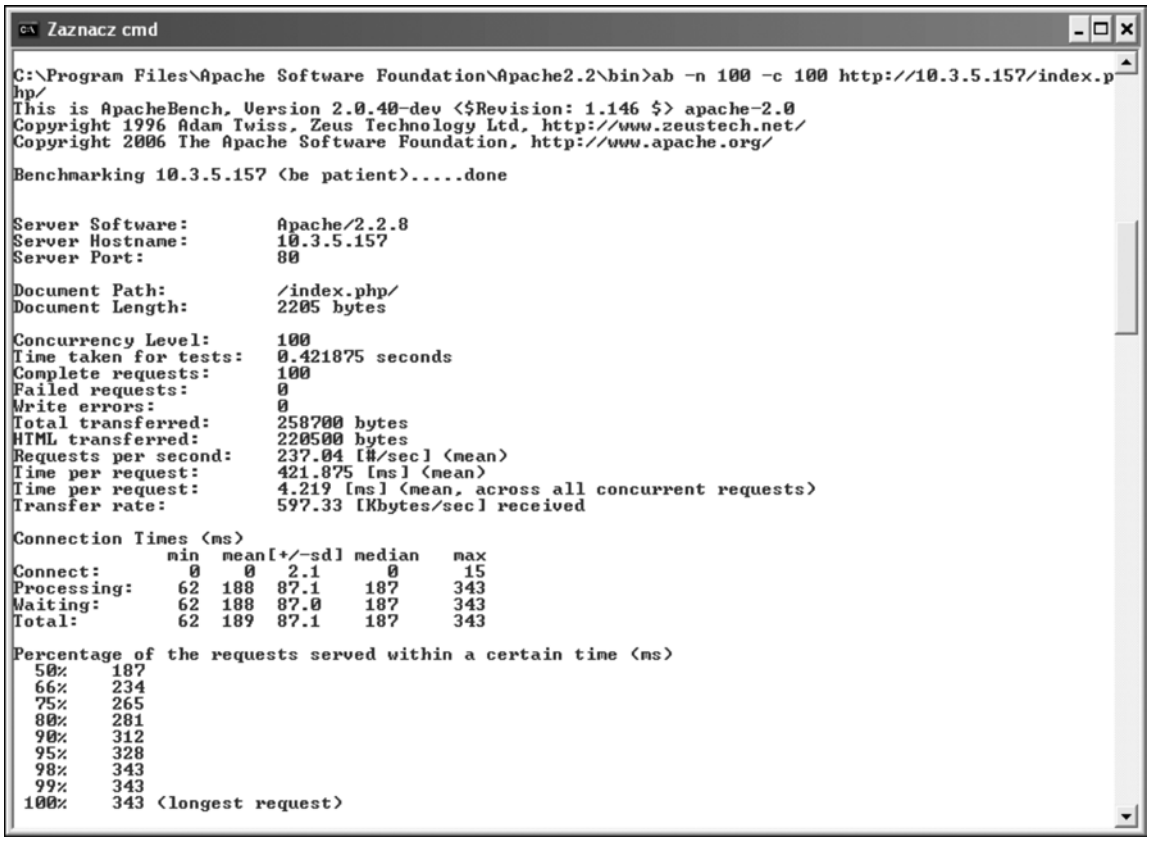
- procesor Intel Celeron M 1.7 GHz
- 512 MB pamięci RAM
- system operacyjny MS Windows XP
- Apache HTTP Server 2.2.8
- PHP 5.2.5
- MySQL 5.0.22

Powyższe parametry techniczne są charakterystyczne dla typowego komputera domowego. Aplikacja nie należy do wymagających dużych mocy obliczeniowych, a zatem należało po prostu upewnić się, że projekt spełnia założenia teoretyczne.

Przy wyborze programu do testowania starano się trzymać przyjętej zasady open source. Mimo że istnieje wiele komercyjnych narzędzi, które symulują ruch, generując żądania do serwera, zdecydowano o wybraniu narzędzia ApacheBench. Jest to program dołączony do dystrybucji Apache, służący do testowania serwera pod wybranym obciążeniem oraz generujący dokładne i uporządkowane raporty. Przy użyciu programu symulacyjnego ApacheBench przetestowano aplikację według najgorszego scenariusza. Wykonano polecenie:

```
# /ab –n 100 –c 100 http://10.3.5.157/index.php/
```

które pobiera stronę główną witryny przy użyciu 100 klientów jednocześnie. Żądania wygenerowane przez program zostały wysłane przez Internet do serwera wyszukiwarki. Raport został przedstawiony na rysunku 56.



Rysunek 56.
Testowanie – raport programu ApacheBench

Warto zwrócić uwagę na wynik informujący o liczbie żądań na sekundę (ang. Requests per second) – było ich 237. Średni czas realizacji żądania (ang. *Time per request*) wynosił 4.219 ms. Z raportu można również odczytać, że wszystkie żądania zostały wykonane w czasie krótszym, niż pól sekundy, a 80% z nich w czasie krótszym niż 0.3 sekundy. W trakcie wykonywania żądania wyświetlenia strony, przepływność (ang. Transfer rate) wynosiła 264 KB/s.

Trudno oszacować dostępną szybkość łącza potencjalnego użytkownika, jednakże biorąc pod uwagę fakt, że dopuszczalny czas ładowania strony, zadowalający użytkownika, powinien wynosić mniej niż 3 sekundy, na podstawie przeprowadzonych testów można stwierdzić, że zaprojektowana aplikacja spełnia to kryterium

Ćwiczenie 19. Przetestuj utworzony przez siebie serwis i zbierz wyniki. Testowanie należy przeprowadzić według przynajmniej dwóch scenariuszy: optymistycznego i pesymistycznego. Scenariusz optymistyczny polega na sprawdzeniu, czy serwis funkcjonuje prawidłowo gdy jest obsługiwany przez inteligentnego użytkownika o pozytywnym nastawieniu. Scenariusz pesymistyczny natomiast polega na sprawdzeniu jak zachowuje się aplikacja przy wprowadzaniu przez użytkownika błędnych danych lub niewprowadzaniu ich wcale i kilkukrotnym naciskaniu tych samych przycisków – niekiedy takie testowanie nazywa się idiotoodpornym. Fragment dotyczący testowania, jego wyników oraz naniesionych poprawek, powinien znaleźć się również w dokumentacji (punkt 6. schematu dokumentacji: Testowanie).

3.5. PREZENTACJA DZIAŁANIA, WYNIKI TESTÓW, WNIOSKI – TRZECI ETAP DOKUMENTACJI PROJEKTU

Instrukcja 14. Uczniowie tworzą ostatnią część dokumentacji projektu: prezentują działanie serwisów, opisują sposób testowania, wyniki testów oraz wnioski końcowe.
W czasie tworzenia dokumentacji projektu, wybrani uczniowie mogą prezentować swoje strony innym uczniom.

Ostatnim etapem tworzenia dokumentacji projektu jest zaprezentowanie działania utworzonego serwisu. Może to być plan korzystania z serwisu przez użytkownika, przedstawiony w postaci kolejnych kroków lub zrzuty ekranu uzupełnione komentarzami. W części podsumowującej powinny się znaleźć również wyniki testów użytkowania oraz sformułowane na ich podstawie wnioski, pozwalające odpowiedzieć na pytanie, czy cel został osiągnięty, czy projekt działa zgodnie z założeniami, czy zakończył się sukcesem. Jeśli okaże się, że pewne założenia nie zostały spełnione, należy wskazać przyczyny niepowodzenia wraz z komentarzem. Można tu zamieścić również własne spostrzeżenia i uwagi:

- podkreślić swoje oryginalne pomysły,
- wskazać trudności, napotkane w trakcie realizacji projektu,
- zaproponować pomysł dalszej rozbudowy serwisu.

Zakończeniem dokumentacji jest wykaz literatury. Pamiętajmy o wyszczególnieniu książek zawierających przykładowe aplikacje, z których korzystaliśmy. Przy wskazywaniu źródeł internetowych, należy oprócz pełnego adresu URL konkretnej witryny umieścić datę wykorzystania.

LITERATURA

1. Cohen J., *Serwisy WWW. Projektowanie, tworzenie, zarządzanie*, Helion, Gliwice 2004
2. Meloni J.C., *PHP, MySQL i Apache dla każdego*, Helion, Gliwice 2007
3. Price J., Price L., *Profesjonalny serwis WWW*, Helion, Gliwice 2002
4. Sokół R., *Internet. Ilustrowany przewodnik*, Helion, Gliwice 2007
5. Ullman L., *PHP i MySQL. Dynamiczne strony WWW*, Helion, Gliwice 2004
6. Welling L., Thomson L., *PHP i MySQL. Vademecum profesjonalisty*, Helion, Gliwice 2005

ADRESY W INTERNECIE

http://wazniak.mimuw.edu.pl/index.php?title=Aplikacje_WWW
http://wazniak.mimuw.edu.pl/index.php?title=Bazy_danych

W projekcie **Informatyka +**, poza wykładami i warsztatami, przewidziano następujące działania:

- 24-godzinne kursy dla uczniów w ramach modułów tematycznych
- 24-godzinne kursy metodyczne dla nauczycieli, przygotowujące do pracy z uczniem zdolnym
- nagrania 60 wykładów informatycznych, prowadzonych przez wybitnych specjalistów i nauczycieli akademickich
- konkursy dla uczniów, trzy w ciągu roku
- udział uczniów w pracach kół naukowych
- udział uczniów w konferencjach naukowych
- obozy wypoczynkowo-naukowe.

Szczegółowe informacje znajdują się na stronie projektu **www.informatykaplus.edu.pl**