Instytut Teleinformatyki
Wydział Inżynierii Elektrycznej i Komputerowej
Politechnika Krakowska

Laboratorium Administrowania Systemami Komputerowymi

„*Apache - serwer WWW*”

ćwiczenie numer: 5

Spis treści

1. Wstępne informacje

1.1 TEMAT ĆWICZENIA

Tematem tego ćwiczenia jest serwer Apache 2.2. Jest to darmowy serwer WWW o bogatej funkcjonalności, stosowany na całym świecie. Serwer zainstalowany jest na systemie Linux Debian Squeeze w domyślnej instalacji z pakietu apache2.

1.2 UWAGI

Przed przystąpieniem do wykonania ćwiczenia należy zapoznać się z następującymi informacjami:

- ścieżka do przestrzeni WWW - /var/www,
- user Apacha – www-data,
- moduł pracy Apacha - prefork,
- w przypadku podglądania wyniku pracy w przeglądarce należy wyczyścić jej CACHE.

1.3 CEL ĆWICZENIA

Dzięki temu ćwiczeniu wykonujący pozna następujące funkcje administracyjne:

- podstawowe działanie serwera Apache,
- metody ograniczania dostępu do domeny,
- zasady tworzenia hostów wirtualnych,
- metody wzbogacania funkcjonalności serwera
- metody translacji adresów i przeadresowania
- obsługę błędów i odpowiedzi

2. Przebieg ćwiczenia

2.1 Przygotowanie ćwiczenia

Logowanie

W celu wykonania ćwiczenia konieczne jest zalogowanie się na konto administratora (login: root, hasło: lab).

Inicjalizacja ćwiczenia

Przed przystąpieniem do zajęć należy uruchomić skrypt konfiguracyjny, znajdujący się w katalogu domowym użytkownika apache, który skonfiguruje serwer Apache do rozpoczęcia pracy.

```
# cd /home/apache
```

```
# ./apache-init
```

2.2 Zadanie nr 1 – Uruchamianie/Wznawianie pracy serwera

Zadanie to polega na zapoznaniu się z pracą serwera WWW, poprzez wznawianie jego pracy oraz analizę podstawowych plików konfiguracyjnych. Pliki konfiguracyjne domyślnej instalacji serwera apache2.2 znajdują się w katalogu /etc/apache2.

Wylistuj ten katalog aby poznać jego zawartość:

```
# ls -l /etc/apache2
```


Podstawowym plikiem konfiguracyjnym jest apache2.conf. Znajdują się w nim bazowe ustawienia serwera, tj. maksymalna ilość klientów, serwerów macierzystych, serwerów potomnych, ścieżka zapisu logu błędów, etc. Poza podstawowymi ustawieniami apache2.conf zawiera ścieżki do innych plików konfiguracyjnych.

W pliku envvars, znajdują się wszystkie zmienne środowiskowe niezbędne do pracy serwera w systemie.

Poleceniem cat dokonaj podglądu poszczególnych plików.

```
# cat apache2.conf
# cat envvars
# cat ports.conf
# cat httpd.conf
```

Zwróć uwagę, że plik httpd.conf jest pusty. Jest to plik konfiguracyjny zarezerwowany dla użytkownika.

Serwer jest aktualnie wyłączony. Włącz przeglądarkę internetową i wpisz w pasku adresu:

```
http://localhost/
```

Przeglądarka nie jest w stanie połączyć się z serwerem.

Do operacji kontrolnych służy polecenie apachectl. Wpisz w konsoli:

```
# apachectl -h
```

by zobaczyć pełną funkcjonalność. Jest to jedynie alias dla polecenia apache2 (we wcześniejszych wersjach - httpd), aczkolwiek apachectl posiada także pewną funkcjonalność.

Wykorzystując apachectl, sprawdź wersję serwera Apache.

Celem wykonania ćwiczenia, jest uruchomienie serwera z domyślnymi ustawieniami:

```
# apachectl start
```


Otwórz przeglądarkę i zobacz czy domena localhost jest osiągalna.

Do zatrzymania pracy serwera służy polecenie apachectl stop, a do jego wznowienia - apachectl restart.

2.3 ZADANIE NR 2 – URUCHAMIANIE MODUŁÓW NA PRZYKŁADZIE PHP

Apache od wersji 2.0 jest serwerem modułowym, oznacza to że jego funkcjonalność osiągana jest poprzez dodawanie poszczególnych modułów. Istnieją dwa rodzaje modułów: wkompilowane, oraz dodawane "w locie". Moduły wkompilowane są niezbędne do pracy serwera i aby dodać nowy moduł należy przekompilować źródła. By zobaczyć listę modułów wkompilowanych należy użyć polecenia:

```
# apachectl -l
```

Moduły dodawane "w locie" to moduły wzbogacające funkcjonalność serwera.

Przejdź do katalogu /etc/apache2:

```
# cd /etc/apache2
```

Znajdują się w nim dwa foldery odnoszące się do modułów: mods-available, oraz mods-enabled. Pierwszy katalog zawiera wszystkie dostępne moduły, natomiast w drugim katalogu znajdują się moduły odblokowane do pracy z serwerem. W rzeczywistości, w folderze mods-enabled znajdują się linki symboliczne do plików zapisanych w folderze mods-available. Na tej podstawie, serwer, odnajdując dany link dowiaduje się, czy konkretny moduł jest odblokowany.

Do odblokowania zainstalowanych modułów służy polecenie a2enmod (polecenie to, dowiązuje link symboliczny do konkretnego pliku). Celem wykonania ćwiczenia należy odblokować moduł php:

```
# a2enmod php5
```

Do poprawnego działania konieczne jest zatrzymanie i ponowne uruchomienie serwera:

```
# apachectl stop
# apachectl start
```

W domenie localhost znajduje się przykładowy plik php, umożliwiający sprawdzenie działania modułu. Wpisz w pasku adresu przeglądarki:

```
http://localhost/phpinfo.php
```


2.4 ZADANIE NR 3 – STRONY UŻYTKOWNIKÓW

Głównym katalogiem wyświetlającym treść WWW jest folder /var/www. Wpisując w przeglądarce adres, serwer przeadresowuje go na taką ścieżkę. Jest to zdefiniowane w pliku konfiguracyjnym domyślnej strony internetowej, dyrektywą DocumentRoot. Zobacz treść pliku konfiguracyjnego:

```
# cat /etc/apache2/sites-available/default
```

Pliki takie składają się z dyrektyw, dzięki którym sterowana jest praca serwera. Możliwe jest na przykład dodanie aliasów, które wyświetlą zdefiniowany katalog. W pliku default dodany jest alias /doc/ wyświetlający zawartość katalogu /usr/share/doc/.

Wpisz do przeglądarki adres:

```
http://localhost/doc/
```

By zobaczyć działanie aliasu.

Możliwe jest jednak stworzenie osobnej przestrzeni WWW dla każdego użytkownika systemu. W tym celu należy posłużyć się modułem userdir:

```
# a2enmod userdir ; apachectl restart
```

Moduł posiada przykładowy plik konfiguracyjny:

```
# cat /etc/apache2/mods-available/userdir.conf
```

Dyrektywy UserDir służą określeniu nazwy katalogu przechowującej treść www, oraz zablokowaniu modułu dla użytkownika root.

W katalogu domowym użytkownika apache, utwórz katalog o nazwie public_html:

```
# mkdir /home/apache/public_html
```

Utwórz w nim przykładowy plik index.html dowolnym edytorem tekstu (np. nano):

```
<html>
   <head>
      <title>Strona testowa</title>
   </head>
   <body>
      <h1>
       Strona testowa uzytkownika apache
      </h1>
   </body>
</html>
```


Wpisz w przeglądarce adres:

```
http://localhost/~apache/
```

2.5 ZADANIE NR 4 – DYREKTYWY ALLOW/DENY, ORAZ PLIK .HTACCESS

Jedną z fundamentalnych zasad serwerów WWW jest ograniczanie dostępu nieuprawnionym użytkownikom. Dyrektywy allow/deny służą do ograniczania dostępu na podstawie adresu IP, puli adresów, nazwy hosta, lub zdefiniowanej zmiennej środowiskowej. W pliku konfiguracyjnym domyślnej strony WWW znajduje się dyrektywa ograniczająca dostęp do aliasu /doc/ tylko dla lokalnego komputera.

Otwórz plik dowolnym edytorem tekstu, np:

```
# nano /etc/apache2/sites-available/default
```

I odnajdź fragment:

```
Alias /doc/ "/usr/share/doc/"
<Directory "/usr/share/doc/">
  Options Indexes MultiViews FollowSymLinks
  AllowOverride None
  Order deny,allow
  Deny from all
  Allow from 127.0.0.0/255.0.0.0 ::1/128
</Directory>
```

Zakomentuj dyrektywę Allow używając znaku # i zapisz plik:

```
#Allow from 127.0.0.0/255.0.0.0 ::1/128
```

Zrestartuj serwer i wpisz w przeglądarce adres:

```
http://localhost/doc/
```

by zobaczyć rezultat.

Dyrektywy allow/deny można stosować zarówno do folderów jak i do plików. Kolejne ćwiczenie będzie polegało na zablokowaniu dostępu do plików jpg na stronie domowej użytkownika apache. W tym celu wykorzystany zostanie także plik .htaccess.

Pliki .htaccess umożliwiają konfigurację konkretnej pod-przestrzeni WWW. Całą konfigurację danej podprzestrzeni można także zawrzeć w głównym pliku konfiguracyjnym httpd.conf stosując dyrektywę <Directory>, lecz w przypadku gdy

użytkownik nie ma do niego dostępu, może posłużyć się plikiem .htaccess, umieszczając go w pożądanym katalogu.

W celu wykonania ćwiczenia skopiuj z katalogu domowego użytkownika przykładowy plik jpg:

```
# cp /home/apache/img.jpg  /home/apache/public_html
```

Następnie wpisz adres obrazka w przeglądarce w celu wyświetlenia go.

W katalogu domowym użytkownika, w folderze public_html, utwórz plik o nazwie .htaccess:

```
# nano /home/apache/public_html/.htaccess
```

Wprowadź do niego następującą treść:

```
<Files ~ "\.jpg$">
  Order allow,deny
  Deny from all
</Files>
```

Zapisz plik i spróbuj ponownie zobaczyć obrazek w przeglądarce.

2.6 ZADANIE NR 5 – AUTORYZACJA UŻYTKOWNIKÓW

Allow/Deny ograniczają dostęp z określonego adresu, istnieje jednak możliwość ograniczenia go konkretnemu użytkownikowi. W tym celu wykorzystywane są moduły autoryzacji i autentykacji. Istnieje szereg modułów w zależności od sposobu przechowywania nazw i haseł użytkowników. W tym ćwiczeniu przedstawiony zostanie moduł pobierający dane użytkowników z pliku tekstowego, oraz moduł wykorzystujący do autoryzacji użytkowników systemu Linux.

- Autentykacja z pliku tekstowego

Do poprawnego działania, niezbędne jest odblokowanie wymaganych modułów:

```
# a2enmod auth_basic authn_file authz_user
# apachectl restart
```

Każda autoryzacja (authz_user) wymaga określenia sposobu przesyłania danych: basic, lub digest (oba można łączyć z ssl) oraz sposobu autentykacji danej autoryzacji

(weryfikowania poprawności wprowadzonych danych) - w tym przypadku z pliku (authn_file).

Po odblokowaniu modułów, w pierwszej kolejności należy utworzyć plik z hasłami poleceniem htpasswd. Ważne by umieścić go w katalogu dostępnym dla demona apacha:.

```
# htpasswd -c /usr/data/.passwords username
New password: mypassword
Re-type new password: mypassword
Adding password for user username
```

Następnie utwórz plik .htaccess w katalogu który wymaga zabezpieczenia:

```
# nano /var/www/auth_file/.htaccess
```

I zapisz w nim odpowiednie dyrektywy:

```
AuthType Basic
AuthName "Restricted (authn_file)"
AuthUserFile /usr/data/.passwords
Require valid-user
```

Wpisz w przeglądarce adres:

```
http://localhost/auth_file/
```

- Autoryzacja z wykorzystaniem użytkowników systemu Linux

Autoryzacja może być wykonana przy użyciu modułu PAM, lecz wówczas wymagane jest by demon Apacha pracował na uprawnieniach roota, co nie jest zalecane. Drugim sposobem jest użycie modułu authnz_external, wykorzystującego autoryzację i autentykację zewnętrznym programem, np. pwauth. pwauth przeszukuje pliki shadows do weryfikacji i zwraca 0 lub 1 w zależności od powodzenia. Moduł wymaga także weryfikacji grupy użytkownika, więc musi współpracować z modułem authz_unixgroup.

Odblokuj wymagane moduły:

```
# a2enmod authnz_external authz_unixgroup

# apachectl restart
```


Pierwszym krokiem jest ustawienie programu współpracującego z modułem odpowiednią dyrektywą w głównym pliku konfiguracyjnym:

```
# nano /etc/apache2/httpd.conf
```

W otwartym pliku dodaj dyrektywy:

```
AddExternalAuth  pwauth /usr/sbin/pwauth
SetExternalAuthMethod pwauth pipe
```

Zmiany w pliku httpd.conf wymagają restartu serwera:

```
# apachectl restart
```

Następnie, analogicznie do autentykacji z pliku, utwórz plik .htaccess w katalogu który należy zabezpieczyć:

```
# nano /var/www/auth_unix/.htaccess
```

Dodaj następujące dyrektywy:

```
AuthType Basic
AuthBasicProvider external
AuthExternal pwauth
AuthzUnixGroup on
AuthName "Restricted (pwauth)"
Require user valid-user
Require group www-access
```

Sprawdź działanie pod adresem:

```
http://localhost/auth_unix/
```

UWAGA: Hasło użytkownika apache : apache

2.7 ZADANIE NR 6 – HOSTY WIRTUALNE

Serwer Apache umożliwia tworzenie hostów wirtualnych. W dużym uproszczeniu, polega to na utworzeniu kilku nazw domenowych dla jednego adresu IP, lub przypisaniu jednej nazwy domenowej kilku adresom IP. Z przyczyn technicznych, ćwiczenie sprowadza się do realizacji pierwszego odwzorowania - do adresu 127.0.0.1 (localhost) przypisanych będzie kilka hostów wirtualnych.

W efekcie wykononania ćwiczenia utworzone zostaną dwa hosty: host1.com dla katalogu /var/www, oraz host2.com dla katalogu użytkownika /home/apache/public_html.

W pierwszej kolejności należy przypisać nazwy host1.com oraz host2.com do adresu 127.0.0.1, by przeglądarka nie próbowała tłumaczyć ich poprzez serwer DNS.

Otwórz plik /etc/hosts:

```
# nano /etc/hosts
```

Znajdź linię: '127.0.0.1 localhost', i dodaj nazwy host1.com, oraz host2.com:

```
127.0.0.1 localhost host1.com host2.com
```

Wirtualne hosty dołączane są analogicznie jak moduły. W katalogu /etc/apache2/sites-available znajduje się zbiór wszystkich dostępnych hostów wirtualnych w postaci ich plików konfiguracyjnych, natomiast katalog /etc/apache2/sites-enabled zawiera linki symboliczne do odblokowanych hostów.

Utwórz plik konfiguracyjny dla host1.com:

```
# nano /etc/apache2/sites-available/host1
```

Zapisz w nim następującą strukturę:

```
<VirtualHost *:80>
      ServerAdmin webmaster@host1.com

      ServerName host1.com
      DocumentRoot /var/www
      <Directory />
            Options FollowSymLinks
            AllowOverride None
      </Directory>
      <Directory /var/www/>
            Options Indexes FollowSymLinks MultiViews
            AllowOverride All
            Order allow,deny
            allow from all
      </Directory>

</VirtualHost>
```


Analogicznie dla host2.com:

```
# nano /etc/apache2/sites-available/host2

<VirtualHost *:80>
      ServerAdmin webmaster@host2.com

      ServerName host2.com
      DocumentRoot /var/www
      <Directory />
            Options FollowSymLinks
            AllowOverride None
      </Directory>
      <Directory /var/www/>
            Options Indexes FollowSymLinks MultiViews
            AllowOverride All
            Order allow,deny
            allow from all
      </Directory>

</VirtualHost>
```

Zrestartuj serwer:

```
# apachectl restart
```

W przeglądarce wpisz następujące adresy:

```
http://host1.com/
http://host1.com/doc/
http://host2.com/
```

Nazwy domenowe tłumaczone są na adres 127.0.0.1, lecz wynikiem jest ten sam host co w przypadku domeny localhost.

Odblokuj hosty używając polecenia:

```
# a2ensite host1 host2
```

Zrestartuj serwer i zobacz efekt wpisując ponownie powyższe adresy.

2.8 ZADANIE NR 7 – TRANSLACJA ADRESÓW

Do translacji adresów służy moduł rewrite. Obciąża on znacznie działanie serwera, lecz pomimo tego jest bardzo często stosowany ze względu na bogatą funkcjonalność. Translacja adresów to tłumaczenie adresu podanego przez user-agenta, który spełnia zadane kryteria, na rzeczywisty adres odzwierciedlający dany zasób.

Przykład:
Adres

http://site.com/sub/a,b,c.html,

tłumaczony jest na adres rzeczywisty, pod którym wykonywany jest określony skrypt, zwracający dany zasób:

http://site.com/scripts/getresource.php?where=sub&val1=a&val2=b&val3=c&client=mozilla&ie=utf-8&etc=itp

Translacja realizowana jest przy użyciu specjalnej dyrektywy:

```
RewriteRule Pattern Substitution [Flags]
```

`Pattern` - fragment adresu który ma być zamieniony
`Substition` - ciąg zamieniający Pattern
`[Flags]` - opcjonalne flagi

Istnieje możliwość łączenia kilku reguł, oraz stosowania wyrażeń warunków przy użyciu dyrektywy RewriteCond.

Do poprawnego wykonania ćwiczeń należy odblokować moduł rewrite:

```
# a2enmod rewrite ; apachectl restart
```

W domenie http://localhost/rewrite_1/ znajdują się dwie strony: A i B. Otwórz je za pomocą przeglądarki:

```
http://localhost/rewrite_1/A.html

http://localhost/rewrite_1/B.html
```

Utwórz w katalogu /var/www/rewrite_1 plik .htaccess o następującej treści:

```
RewriteEngine on
RewriteRule ^A.html$ B.html
```

Fragment tekstu A.html jest teraz tłumaczony na B.html. Zobacz w przeglądarce stronę A

W domenie http://localhost/rewrite_2/ znajduje się skrypt index.php do którego można przekazać zmienną page, przez adres. Zapoznaj się z działaniem skryptu wpisując w przeglądarce:

```
http://localhost/rewrite_2/index.php?page=dowolny_ciag_liter
```

Utwórz w katalogu /var/www/rewrite_2 plik .htaccess o następującej treści:

```
RewriteEngine on
RewriteRule ^page/([^/\.]+)/?$ index.php?page=$1 [L]
```

a następnie wpisz w przeglądarce adres:

```
http://localhost/rewrite_2/page/dowolny_ciag_liter
```

2.9 ZADANIE NR 8 – PRZEKIEROWANIA

Moduł rewrite może zostać użyty do realizacji przekierowań na inny adres, lub protokół. Taki efekt można uzyskać poprzez zastosowanie flagi [R] oznaczającej jawne przekierowanie (redirect). W tym ćwiczeniu przedstawione zostanie przekierowanie z http do https. Szyfrowane połączenie zostanie nawiązane z domyślną stroną, której plik konfiguracyjny znajduje się w katalogu /etc/apache2/sites-available. Przejrzyj plik:

```
# cat /etc/apache2/sites-available/default-ssl
```

Z pliku wynika, że strona wymaga aktywnego modułu SSL. Ponadto można wydobyć z niego takie informacje jak port na którym działa host (443), oraz lokalizację klucza i certyfikatu bezpieczeństwa (dyrektywy SSLCertificateFile, SSLCertificateKeyFile). Można utworzyć własny certyfikat, np. przy użyciu pakietu ssl-cert. W ćwiczeniu wykorzystany zostanie utworzony wcześniej certyfikat.

Aktywuj moduł SSL, oraz host default-SSL:

```
# a2enmod ssl ; a2ensite default-ssl ; apachectl restart
```

Utwórz plik .htaccess w katalogu /var/www o następującej treści:

```
RewriteEngine On
RewriteCond %{SERVER_PORT} 80
RewriteRule ^(.*)$ https://localhost/$1 [R,L]
```


RewriteCond to wyrażenie warunkowe, sprawdzające na którym porcie serwer komunikuje się z klientem.

Sprawdź działanie dyrektyw wpisując w przeglądarce adres:

```
http://localhost/
```

2.10 ZADANIE NR 9 – OBSŁUGA BŁĘDÓW I ODPOWIEDZI

Z punktu widzenia administracji serwerem, istotnym elementem do analizy błędów są logi. Ścieżka do logów przechowywana jest w zmiennej APACHE_LOG_DIR w pliku envvars (domyślnie /var/log/apache2). W pliku konfiguracyjnym danego hosta podana jest lokalizacja pliku logów poprzez dyrektywę ErrorLog. Dyrektywą LogLevel definiuje się poziom istotności informacji zapisywanych w logach. Do wyboru są następujące opcje: debug, info, notice, warn, error, crit, alert, emerg.
Każdy host może mieć utworzonych kilka własnych plików z logami.

Przejrzyj domyślny plik z logami błędów:

```
# cat /var/log/apache2/error.log
```

Tworzenie własnych logów jest możliwe przy użyciu przy użyciu dyrektywy LogFormat.

Otwórz plik konfiguracyjny apache2.conf i odszukaj przykładowe formaty logów:

```
# cat /etc/apache2/apache2.conf
```

Z punktu widzenia klienta, istotne są odpowiedzi z serwera jakie otrzyma w przypadku błędu (np. brak określonego zasobu, brak dostępu do zasobu, itp.). Apache umożliwia tworzenie własnych odpowiedzi w przypadku wygenerowania kodu błędu HTTP. Służy do tego dyrektywa ErrorDocument.

W pliku localized-error-pages znajduje się przykładowa realizacja obsługi odpowiedzi.

Otwórz go edytorem tekstu:

```
# nano /etc/apache2/conf.d/localized-error-pages
```


Znajdź fragment:

```
#<IfModule mod_negotiation.c>
# <IfModule mod_include.c>
#  <IfModule mod_alias.c>
#
#    Alias /error/ "/usr/share/apache2/error/"
#
#    <Directory "/usr/share/apache2/error">
#      AllowOverride None
#      Options IncludesNoExec
#      AddOutputFilter Includes html
#      AddHandler type-map var
#      Order allow,deny
#      Allow from all
#      LanguagePriority en cs de es fr it nl sv pt-br ro
#      ForceLanguagePriority Prefer Fallback
#    </Directory>
#
#    ErrorDocument 400 /error/HTTP_BAD_REQUEST.html.var
#    ErrorDocument 401 /error/HTTP_UNAUTHORIZED.html.var
#    ErrorDocument 403 /error/HTTP_FORBIDDEN.html.var
#    ErrorDocument 404 /error/HTTP_NOT_FOUND.html.var
#    ErrorDocument 405 /error/HTTP_METHOD_NOT_ALLOWED.html.var
#    ErrorDocument 408 /error/HTTP_REQUEST_TIME_OUT.html.var
#    ErrorDocument 410 /error/HTTP_GONE.html.var
#    ErrorDocument 411 /error/HTTP_LENGTH_REQUIRED.html.var
#    ErrorDocument 412 /error/HTTP_PRECONDITION_FAILED.html.var
#    ErrorDocument 413 /error/HTTP_REQUEST_ENTITY_TOO_LARGE.html.var
#    ErrorDocument 414 /error/HTTP_REQUEST_URI_TOO_LARGE.html.var
#    ErrorDocument 415 /error/HTTP_UNSUPPORTED_MEDIA_TYPE.html.var
#    ErrorDocument 500 /error/HTTP_INTERNAL_SERVER_ERROR.html.var
#    ErrorDocument 501 /error/HTTP_NOT_IMPLEMENTED.html.var
#    ErrorDocument 502 /error/HTTP_BAD_GATEWAY.html.var
#    ErrorDocument 503 /error/HTTP_SERVICE_UNAVAILABLE.html.var
#    ErrorDocument 506 /error/HTTP_VARIANT_ALSO_VARIES.html.var
#  </IfModule>
# </IfModule>
#</IfModule>
```

I odkomentuj go, usuwając znak # w każdej linii.

Zapisane strony odpowiedzi znajdują się w katalogu /usr/share/apache2/error, można jednak tworzyć własne strony błędów, lub przekierowywać na zewnętrzny serwer.

Do działania odpowiedzi konieczna jest aktywacja modułów alias, negotiation i include:

```
# a2enmod alias negotiation include ; apachectl restart
```

Wpisz w przeglądarce błędny adres, oraz adres do domeny error, by zobaczyć działanie:

```
http://localhost/error/
http://localhost/zlyadres/
```

Prawie każdy serwer www posiada własne strony błędów. Jeśli masz połączenie z Internetem, wpisz w przeglądarce adres:

```
http://google.pl/zlyadres
```


2.11 OPRACOWANIE ĆWICZENIA I SPRAWOZDANIE

Wykonanie ćwiczenia polega na praktycznej realizacji wszystkich zadań **Rozdziału 2** niniejszej instrukcji zatytułowanego „**Przebieg Ćwiczenia**". Należy sporządzić sprawozdanie z wykonania ćwiczenia (w formie dokumentu elektronicznego) i w ciągu najdalej dwóch tygodni od dnia wykonania ćwiczenia oddać je prowadzącemu zajęcia.

Kompletne opracowanie ćwiczenia powinno zawierać:

- ✓ Część opisową odnoszącą się do teorii przerabianego ćwiczenia. Ta część sprawozdania powinna wykazać dobrą ogólną znajomość zagadnień leżących u podstaw przerabianego tematu, znajomość odnośnej literatury, samodzielność myślenia i umiejętność pisania opracowań o charakterze technicznym.

- ✓ Wnioski praktyczne wynikające z wykonania ćwiczenia, a w tym:

  - uwagi odnoszące się do przebiegu ćwiczenia (np. czy dane ćwiczenie może być wykonane z pełnym rozumieniem zawartych w nim czynności i problemów, czy ćwiczenie jest możliwe do wykonania w czasie przeznaczonym na zajęcia, czy ćwiczenie jest zbyt trudne/ zbyt łatwe, itp.,
  - uwagi odnoszące się do sposobu przygotowania i jakości (waloru dydaktycznego) instrukcji do ćwiczenia,
  - uwagi odnoszące się do ewentualnych utrudnień technicznych lub organizacyjnych pojawiających się w trakcie wykonywania ćwiczenia,
  - postulaty merytoryczne i techniczne dotyczące usprawnienia/ulepszenia jakości wykonywanego ćwiczenia,
  - inne

  Wnioski z drugiej części sprawozdania posłużą do usprawnienia i poprawy zajęć laboratoryjnych w latach następnych.

