Deadline: 19.04.2020
Punkty: 1

Przygotował: Mateusz Ratajczak

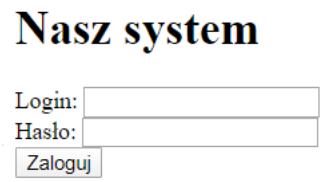
PHP

Celem ćwiczenia jest przygotowanie prostego systemu wykonanego w języku PHP wykorzystując podstawowe mechanizmy oraz formularze. Poniższe zadania pozwolą na przećwiczenie elementów, które zawarłem w wykładzie do tych laboratoriów. Do wykonania ćwiczenia potrzebny jest dowolny **serwer WWW** np. XAMPP oraz dowolny edytor plików tekstowych i przeglądarka.

1. Link do filmu pokazującego instalację serwera XAMPP: https://www.youtube.com/watch?v=WSeKPbVZBoo
2. Stwórz trzy pliki w tym samych katalogu serwera www np. dla XAMPP w folderze htdocs: index.php, user.php, cookie.php.
3. W każdym z nich umieść podstawowy kod HTML:

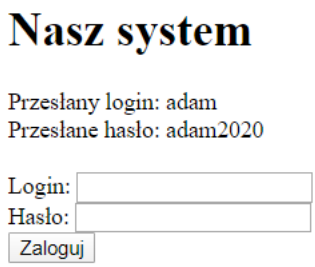
```
<!DOCTYPE html>
<html>
   <head>
        <title>PHP</title>
        <meta charset='UTF-8' />
   </head>
   <body>

   </body>
</html>
```

4. Wyświetl na ekranie komunikat z poziomu kodu php „Nasz system” w sekcji body pliku index.php. Zastosuj również nagłówek pierwszego poziomu.
5. Utwórz formularz zawierający trzy pola: login(text), hasło (password) oraz przycisk zaloguj (submit)
    a. Ustaw wartość action na stronę index.php, typ przesyłania danych na metodę POST. Nazwy pól formularza powinny być następujące: login, haslo, zaloguj.
    b. Strona powinna wyglądać na tą chwilę mniej więcej tak:



6. Teraz zajmiemy się odbiorem danych przesłanych przez formularz. Tak jak ustawiliśmy formularz ma zostać przesłany na stronę, na której się znajduje. Używając instrukcji warunkowej if oraz funkcji isSet() sprawdź czy została przesłana zmienna zaloguj. Jeśli tak, odbierz dane z pól formularza i wyświetl je na ekranie.

a.



7. Jednym z zagrożeń jest wstrzykiwanie w formularz fragmentów kodu, próbujących zakłócić działanie naszego systemu. Aby zminimalizować taką możliwość zastosuj funkcję test_input, znajdziesz ją na jednym z ostatnich slajdów wykładu do tych laboratoriów.
    a. Z tej funkcji będziemy korzystać w każdym z trzech plików .php, więc dla wygody utwórz nowy plik funkcje.php i umieść w nim tylko funkcję test_input.
    b. Aby skorzystać z tej funkcji na pozostałych stronach dołącz plik funkcje.php przy użyciu funkcji require() do każdego z nich. Ta instrukcja musi znaleźć się powyżej pierwszego użycia funkcji.
8. Na tą chwilę możemy odebrać i wyświetlić dane z formularza. Chcielibyśmy jednak umożliwić zalogowanie użytkownikowi oraz wyświetlić jego imię i nazwisko. Nie dopinamy bazy danych, dlatego utworzymy sobie dwóch użytkowników. Dla uproszczenia umieść następujący kod w pliku funkcje.php

```
class Osoba {
   public $login;
   public $haslo;
   public $imieNazwisko;
}

$osoba1 = new Osoba;
$osoba1->login = "adam";
$osoba1->haslo = "adam2020";
$osoba1->imieNazwisko = "Adam Kowalski";

$osoba2 = new Osoba;
$osoba2->login = "agata";
$osoba2->haslo = "2020agata";
$osoba2->imieNazwisko = "Agata Nowak";
```

    a. Aby wyświetlić wartość zmiennej: `echo $osoba1->login;`
9. Po odebraniu danych z formularza logowania sprawdź, czy akurat zalogowała się jedna z tych dwóch osób. Jeżeli tak, to utwórz zmienną sesji o nazwie „zalogowanyImie” i umieść w niej imię i nazwisko danego użytkownika.
   Utwórz drugą zmienną sesji o nazwie „zalogowany” i ustaw jej wartość na 1. Na tym etapie zakomentuj wyświetlanie przesłanego loginu i hasła. (Komentarz //).
    a. Aby działały zmienne sesji uruchom sesję w każdym pliku (oprócz pliku funkcje.php). Umieść następujący blok php **w pierwszej linii** powyżej DOCTYPE
       `<?php session_start(); ?>`
    b. Dodatkowo po udanym logowaniu przekieruj użytkownika na podstronę *user.php* używając `header("Location: user.php");`
10. Teraz wykonaj eksperyment. Wypełnij i prześlij formularz **podając błędne dane**. Następnie odśwież stronę. Powinien pojawić się komunikat „Potwierdź ponowne przesłanie formularza”. Czy pojawił się? Jest to jeden z głównych problemów języka PHP. Przebuduj program, aby ten problem nie występował po nieudanej autoryzacji:

a. Stwórz dodatkowy plik logowanie.php, który obsłuży dane przesłane formularzem. Nie zapomnij zmienić parametru action.
b. W pliku logowanie.php po udanej autoryzacji przenieś użytkownika na podstronę user.php. W przypadku błędnego logowania, przekieruj użytkownika na stronę index.php. Posłuż się kodem, który już stworzyłeś do autoryzacji w pliku index.php.
c. Sprawdź, czy występuje problem ponownego przesłania formularza.

11. Przejdź do konfiguracji pliku user.php. Na tą chwilę powinien on wyglądać mniej więcej tak:

```
<?php session_start(); ?>
<!DOCTYPE html>
<html>
   <head>
        <title>PHP</title>
        <meta charset='UTF-8' />
   </head>
   <body>
        <?php
             require_once("funkcje.php");
             echo "Zalogowano";
        ?>
   </body>
</html>
```

12. Pierwszą rzeczą będzie zabezpieczenie dostępu do pliku user.php, aby dostęp do niego miały jedynie osoby zalogowane. W tym celu na początku sprawdź, czy zmienna sesji o nazwie „zalogowany" jest ustawiona i posiada wartość 1. Jeśli nie, cofnij użytkownika do strony index.php.
    a. Dodaj hiperłącza na stronie user.php do strony index.php oraz w drugą stronę.
    b. Sprawdź, czy bez poprawnego zalogowania można przejść na podstronę user.php. Ale uwaga, prawdopodobnie już jesteś zalogowany, a nie zrobiłeś jeszcze przycisku wyloguj, stąd zamknij przeglądarkę i otwórz ponownie – uruchomi się nowa sesja.
13. W pliku user.php wyświetl imię i nazwisko zalogowanej osoby korzystając z utworzonej wcześniej zmiennej sesji.
14. Dodaj przycisk wylogowania. Umieść na stronie user.php formularz, który będzie miał tylko przycisk typu submit o nazwie „wyloguj" i value=„wyloguj". Przycisk ma przenieść użytkownika na stronę index.php. Następnie dodaj w pliku index.php obsługę przesłania tego formularza zmieniając wartość zmiennej sesji „zalogowany" na wartość 0.
15. Na stronie user.php dodaj formularz, który umożliwi upload pliku na serwer. Przykład kodu znajduje się na wykładzie do tego laboratorium. Dodaj plik graficzny w formacie jpg lub png, a następnie wyświetl go na tej podstronie. Załóż, że nazwa pliku jest znana i może być na sztywno wpisana w kod HTML. Jeśli zdjęcie nie znajduje się jeszcze na serwerze, wyświetl tekst alternatywny dla zdjęcia. Przesłanie tego formularza powinno zostać obsłużone na podstronie user.php
    a. Formularz dodatkowo musi zawierać atrybut enctype.

```
<form action='index.php' method='POST' enctype='multipart/form-data'>
```

16. Kolejną częścią zadania będzie stworzenie i testowanie zmiennych cookie. W tym celu dodaj kolejny formularz na stronie index.php posiadający jedno pole typu number i przycisk submit. Nazwij je kolejno „czas" oraz „utworzCookie". Formularz ma przenosić użytkownika na podstronę cookie.php, która powinna w tym momencie wyglądać tak jak plik wejściowy

user.php, z którym zaczynaliśmy działać w zadaniu 10. Formularz do ma realizować przesyłanie zmiennych metodą GET. Czas życia cookie będziemy podawać w sekundach!

17. W pliku cookie.php odbierz przesłany czas życia ciasteczka w sekundach oraz utwórz cookie za pomocą poniższego polecenia. Nadaj dowolną nazwę i wartość. Wyświetl komunikat o powodzeniu dodania ciasteczka.

```
setcookie("nazwa", "wartość", time() + (86400 * 30), "/");
```

    a. Zwróć uwagę na adres URL po przesłaniu formularza – zawiera on doklejone do adresu nazwy oraz wartości zmiennych jakie przesyłamy formularzem.
18. Dodaj hiperłącze „wstecz” na stronie cookie.php przenoszące na stronę index.php
19. Na stronie index.php dodaj sekcję wyświetlającą wartość cookie, jeśli ono istnieje. Przetestuj czas życia zmiennej. Ustaw kilka czasów zaczynając od krótkiego. Odświeżaj stronę główną i patrz kiedy cookie straci ważność i wygaśnie. Dodatkowo sprawdź, czy zamykając przeglądarkę i otwierając ją ponownie, zmienna cookie będzie przechowywać ustawioną wartość
20. Zadbaj o przejrzystość strony – umieść informacje obok formularzy, jaką pełnią rolę. Możesz podłużyć się znacznikiem HTML – fieldset oraz legend: https://www.w3schools.com/tags/tag_fieldset.asp