SERVERSKE TEHNOLOGIJE

Vežba 6

Korisnički nalozi i sesije u PHP-u

04.12.2024.

U današnje vreme, većina web aplikacija zahteva rad sa korisnicima i njihovim podacima. Jedan od osnovnih zadataka web programiranja je omogućiti korisnicima da kreiraju svoje naloge, prijavljuju se na sajtovima (aplikacijama) i obavljaju različite operacije na osnovu njihovog statusa (ulogovan/samo gost).

PHP, u kombinaciji sa PDO (PHP Data Objects), omogućava sigurno i efikasno upravljanje korisničkim podacima u bazi. U ovoj vežbi, koristićemo PDO za povezivanje sa bazom podataka, rad sa korisničkim podacima i implementaciju sesija kako bismo obezbedili da samo ulogovani korisnici mogu pristupiti određenim funkcijama aplikacije.

Kreiranje korisničkih naloga

Kreiranje korisničkog naloga omogućava novim korisnicima da se registruju na aplikaciju (sajt). Ovo uključuje validaciju unosa (proveru formata email-a, dužine lozinke, i da li su polja popunjena), heširanje lozinke za bezbednost i čuvanje podataka u bazi podataka. Validacija je ključna kako bi se izbegli problemi kao što su SQL injekcije ili pogrešan unos.

- **Validacija email adrese**: Koristi se posebna PHP funkcija **filter_var()** sa opcijom FILTER_VALIDATE_EMAIL.
- **Hashovanje lozinke**: Funkcija **password_hash()** koristi sigurne algoritme za heširanje (kao što je bcrypt).
- **Provera jedinstvenosti podataka**: Proverava se da li korisnički email već postoji u bazi.

```
<?php
require 'config.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = trim($_POST['name']);
```

```
    $email = trim($_POST['email']);
    $password = $_POST['password'];

    // Validacija unosa
    if (empty($name) || empty($email) || empty($password)) {
        echo "Sva polja su obavezna!";
        exit;
    }

    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        echo "Email adresa nije validna!";
        exit;
    }

    // Provera da li email već postoji
    $stmt = $pdo->prepare("SELECT * FROM users WHERE email = ?");
    $stmt->execute([$email]);
    if ($stmt->rowCount() > 0) {
        echo "Korisnik sa ovim email-om već postoji!";
        exit;
    }

    // Hashovanje lozinke
    $hashedPassword = password_hash($password, PASSWORD_DEFAULT);

    // Unos korisnika u bazu
    $stmt = $pdo->prepare("INSERT INTO users (name, email, password) VALUES
                           (?, ?, ?)");
    if ($stmt->execute([$name, $email, $hashedPassword])) {
        echo "Registracija uspešna!";
    } else {
        echo "Došlo je do greške prilikom registracije.";
    }
}
?>
```

Validacija i logovanje

Logovanje omogućava korisnicima pristup personalizovanim delovima web aplikacije (sajta). Tokom logovanja:

- Korisnik unosi email i lozinku.
- Podaci se upoređuju sa podacima u bazi (email i heširana lozinka).
- Ako je unos ispravan, otvara se sesija koja čuva podatke o korisniku (npr. ime i ID).

Sesije omogućavaju da aplikacija "pamti" podatke o korisniku dok se kreće kroz različite stranice. Na taj način možemo vršiti više operacija nad trenutno ulogovanim korisnikom.

```
<?php
session_start();
require 'config.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $email = trim($_POST['email']);
    $password = $_POST['password'];

    // Validacija unosa
    if (empty($email) || empty($password)) {
        echo "Sva polja su obavezna!";
        exit;
    }
    // Preuzimanje korisnika iz baze
    $stmt = $pdo->prepare("SELECT * FROM users WHERE email = ?");
    $stmt->execute([$email]);
    $user = $stmt->fetch();

    if ($user && password_verify($password, $user['password'])) {
        // Uspešno logovanje - postavljanje sesije
        $_SESSION['user_id'] = $user['id'];
        $_SESSION['user_name'] = $user['name'];
        echo "Dobrodošli, " . $_SESSION['user_name'] . "!";
    } else {
        echo "Email ili lozinka nisu ispravni.";
    }
}
?>
```

Operacije za logomania korisnike

Ključni koraci za implementaciju operacija za ulogovane korisnike uključuju:

- **Proveru sesije:** Da bi se osiguralo da samo autentifikovani (ulogovani) korisnici mogu pristupiti određenim funkcijama, potrebno je proveriti da li sesija postoji i da li su podaci o korisniku ispravno setovani.
- **Zaštitu stranica i ruta:** Stranice i rute koje su namenjene isključivo ulogovanim korisnicima moraju biti zaštićene validacijom sesije.
- **Rad sa podacima korisnika:** Ove operacije uključuju rad sa podacima iz baze podataka koji su specifični za trenutno ulogovanog korisnika. Mogu uključivati pristup personalizovanim podacima, zatim ažuriranje korisničkog profila, pregled sadržaja i druge funkcionalnosti koje nisu dostupne anonimnim korisnicima.
- **Odjavu korisnika:** Omogućava sigurno zatvaranje sesije uklanjanjem sesijskih podataka i preusmeravanjem datog korisnika na početnu stranicu ili stranicu za logovanje, čime se sprečava neovlašćen pristup.

Provera sesije

Da bismo omogućili pristup funkcijama samo ulogovanim korisnicima, prva stvar koju radimo jeste provera sesije na početku svake zaštićene stranice ili skripte.

Primer koda za proveru sesije:

```
<?php
session_start();

if (!isset($_SESSION['user_id'])) {
    // Ako sesija nije postavljena, korisnik nije ulogovan
    echo "Pristup nije dozvoljen. Molimo, prijavite se.";
    header("Location: login.php"); // Redirekcija na stranicu za logovanje
    exit;
}
?>
```

U ovom primeru, proverava se da li je podešena sesija sa ključem user_id. Ako nije, korisnik se preusmerava na stranicu za logovanje. Ako je sesija validna, korisnik će moći da vidi sadržaj svoje personalizovane stranice, uz poruku dobrodošlice, kao na primer:

```
echo "Dobrodošli, " . htmlspecialchars($_SESSION['user_name']) . "! Ovo je zaštićena stranica.";
```

Ažuriranje korisničkog profila

Jedna od osnovnih operacija za ulogovane korisnike jeste ažuriranje podataka profila. To uključuje promenu imena, email adrese ili lozinke. Ažuriranje lozinke zahteva posebnu pažnju jer uključuje dodatne provere, kao što je validacija trenutne lozinke pre promene.

Primer koda za ažuriranje lozinke:

```
<?php
session_start();
require 'config.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $currentPassword = $_POST['current_password'];
    $newPassword = $_POST['new_password'];

    // Preuzimanje trenutnog korisnika iz baze
    $stmt = $pdo->prepare("SELECT * FROM users WHERE id = ?");
    $stmt->execute([$_SESSION['user_id']]);
    $user = $stmt->fetch();

    // Provera trenutne lozinke
    if ($user && password_verify($currentPassword, $user['password'])) {
        // Ažuriranje lozinke
        $hashedPassword = password_hash($newPassword, PASSWORD_DEFAULT);
        $stmt = $pdo->prepare("UPDATE users SET password = ? WHERE id = ?");
        if ($stmt->execute([$hashedPassword, $_SESSION['user_id']])) {
            echo "Lozinka uspešno promenjena.";
        } else {
            echo "Došlo je do greške.";
        }
    } else {
        echo "Trenutna lozinka nije ispravna.";
    }
}
?>
```

Primer koda za ažuriranje ostalih podataka:

```
<?php
session_start();
require 'config.php';

if (!isset($_SESSION['user_id'])) {
    echo "Pristup nije dozvoljen.";
    header("Location: login.php");
    exit;
}

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = trim($_POST['name']);
    $email = trim($_POST['email']);
    $userId = $_SESSION['user_id'];

    // Validacija unosa
    if (empty($name) || empty($email)) {
        echo "Sva polja su obavezna!";
        exit;
    }

    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        echo "Email adresa nije validna!";
        exit;
    }

    // Ažuriranje korisničkih podataka u bazi
    $stmt = $pdo->prepare("UPDATE users SET name = ?, email = ? WHERE id = ?");
    if ($stmt->execute([$name, $email, $userId])) {
        echo "Profil je uspešno ažuriran.";
    } else {
        echo "Došlo je do greške prilikom ažuriranja profila.";
    }
}
?>
```

Ovaj kod omogućava korisnicima da ažuriraju svoje ime i email. Proverava se validnost unosa, a podaci se ažuriraju samo za trenutno ulogovanog korisnika.

Prikaz personalizovanih podataka

Još jedna važna funkcionalnost je prikaz podataka specifičnih za korisnika, kao što su lične poruke, istorija narudžbina ili bilo koji drugi podaci vezani za nalog.

Primer koda za prikaz podataka:

```
<?php
session_start();
require 'config.php';

if (!isset($_SESSION['user_id'])) {
    echo "Pristup nije dozvoljen.";
    header("Location: login.php");
    exit;
}

$userId = $_SESSION['user_id'];

// Preuzimanje korisničkih podataka iz baze
$stmt = $pdo->prepare("SELECT * FROM orders WHERE user_id = ?");
$stmt->execute([$userId]);
$orders = $stmt->fetchAll();

if ($orders) {
    echo "<h3>Vaše narudžbine:</h3>";
    foreach ($orders as $order) {
        echo "Narudžbina #{$order['id']}: {$order['product_name']}
                                        - {$order['status']}<br>";
    }
} else {
    echo "Nemate nijednu narudžbinu.";
}
?>
```

Ovaj primer prikazuje listu svih narudžbina za trenutno ulogovanog korisnika, koje se preuzimaju iz baze na osnovu njihovog user ID-ja.

Brisanje podataka iz baze

Brisanje podataka iz baze podataka je popularna operacija, bilo da se radi o uklanjanju zastarelih podataka, neželjenih unosa ili korisničkih podataka. Prilikom implementacije ove operacije, važno je obezbediti validaciju unosa kako bi se sprečile potencijalne greške ili zlonamerne radnje (npr. SQL injekcije).

Primer: Brisanje korisnika na osnovu njihovog ID-ja.

```
<?php
require 'config.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // Proverava da je ID broj
    $userId = intval($_POST['user_id']);

    // Validacija da je ID prosleđen
    if (empty($userId)) {
        echo "ID korisnika nije validan!";
        exit;
    }

    // Brisanje korisnika iz baze
    $stmt = $pdo->prepare("DELETE FROM users WHERE id = ?");
    if ($stmt->execute([$userId])) {
        echo "Korisnik sa ID-jem $userId uspešno obrisan.";
    } else {
        echo "Došlo je do greške prilikom brisanja korisnika.";
    }
}
?>
```

Računanje prosečne/ukupne/najmanje/najveće vrednosti

Računanje nekih vrednosti na osnovu podataka iz baze podataka je takođe popularno, posebno u analizama podataka. Na primer, možemo izračunati prosečnu ocenu korisnika za određeni proizvod ili prosečno trajanje korisničkih sesija.

```php
<?php
require 'config.php';

try {
    // SQL upit za računanje prosečne ocene
    $stmt = $pdo->query("SELECT AVG(rating) AS average_rating FROM users");

    // Dobijanje rezultata
    $result = $stmt->fetch();
    if ($result['average_rating'] !== null) {
        echo "Prosečna ocena korisnika je: " . round($result['average_rating'], 2);
    } else {
        echo "Nema dostupnih ocena za računanje proseka.";
    }
} catch (PDOException $e) {
    echo "Greška prilikom računanja prosečne vrednosti: " . $e->getMessage();
}
?>
```

Odjava korisnika (logout)

Odjava korisnika je ključna za zaštitu privatnosti i sigurnost podataka. Kada se korisnik odjavi, sesija se zatvara, čime se uklanjaju svi podaci povezani sa trenutnom sesijom.

Primer koda za odjavu korisnika:

```php
<?php
session_start();

// Brisanje svih podataka iz sesije
session_unset();

// Uništavanje sesije
session_destroy();

// Redirekcija korisnika na početnu stranicu
header("Location: index.php");
exit;
?>
```

Zadatak za samostalni rad

Kreirajte sistem za biblioteku koja omogućava korisnicima da se prijave, pregledaju dostupne knjige, rezervišu ih, prate iznajmljene knjige, i vraćaju iste nakon korišćenja. Cilj je da implementirate osnovne funkcionalnosti pomoću PHP-a i PDO-a, uz razmišljanje o pravilnoj logici i relacijama između korisnika i knjiga. Stilizovanje odradite po želji.

1. Registracija korisnika

- Kreirajte funkcionalnost za dodavanje novih članova biblioteke.
- Validirajte ime, email i lozinku, i čuvajte podatke u bazi sa heširanjem lozinke.

2. Pregled dostupnih knjiga

- Napravite stranicu na kojoj korisnik može videti listu dostupnih knjiga.
- Podaci o knjigama se povlače iz baze: *naslov, autor, godina izdanja, broj dostupnih primeraka*.
- Korisnik može rezervisati knjigu klikom na dugme.

3. Rezervacija knjige

- Kada korisnik rezerviše knjigu, potrebno je smanjiti broj dostupnih primeraka u bazi i zabeležiti informaciju o korisniku koji je izvršio rezervaciju.
- Proveriti da li korisnik već ima rezervisanu knjigu ili da li su svi primerci zauzeti.

4. Praćenje iznajmljenih knjiga

- Napravite stranicu na kojoj korisnik može videti listu svih knjiga koje je trenutno iznajmio, uključujući datume iznajmljivanja.
- Dodajte logiku za prikazivanje obaveštenja ukoliko se približava rok za vraćanje knjige (npr. 7 dana pre isteka).

5. Vraćanje knjige

- Implementirajte funkcionalnost vraćanja knjige. Kada korisnik vrati knjigu, broj dostupnih primeraka se povećava, a zapis o rezervaciji se uklanja iz baze.

6. Administracija knjiga

- Dodajte administratorski panel koji omogućava osoblju biblioteke da dodaju nove knjige, ažuriraju postojeće informacije (npr. broj primeraka), i brišu knjige iz sistema. Takođe administratorima možete omogućiti da pogledaju sve korisnike, i, na primer, promene im lozinku ili email.

Kada pravite tabelu sa korisnicima, možete dodati indikatore da se zna ko je admin a ko samo član biblioteke. Na glavnoj stranici treba i članovima i adminima da bude dostupan login, a signup je samo za članove biblioteke. Pretpostavićemo da su admini inicijalno dodati. Bazu možete popunjavati proizvoljnim podacima (ChatGPT, podaci “iz glave”, itd).

Primeri koda

Pregled dostupnih knjiga

```
<?php
require 'config.php';

$stmt = $pdo->query("SELECT title, author, year, available_copies FROM books WHERE
available_copies > 0");
$books = $stmt->fetchAll();
?>

<h2>Dostupne knjige</h2>
<table>
    <tr>
        <th>Naslov</th>
        <th>Autor</th>
        <th>Godina</th>
        <th>Dostupni primerci</th>
        <th>Rezervacija</th>
    </tr>
    <?php foreach ($books as $book): ?>
    <tr>
        <td><?= htmlspecialchars($book['title']) ?></td>
        <td><?= htmlspecialchars($book['author']) ?></td>
        <td><?= htmlspecialchars($book['year']) ?></td>
        <td><?= htmlspecialchars($book['available_copies']) ?></td>
        <td>
            <form method="POST" action="reserve.php">
                <input type="hidden"
                 name="book_title" value="<?= htmlspecialchars($book['title'])?>">
                <button type="submit">Rezerviši</button>
            </form>
        </td>
    </tr>
    <?php endforeach; ?>
</table>
```

Rezervacija knjige od strane ulogovanog korisnika

```
<?php
session_start();
require 'config.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_SESSION['user_id'])) {
    $userId = $_SESSION['user_id'];
    $bookTitle = $_POST['book_title'];

    // Provera dostupnosti knjige
    $stmt = $pdo->prepare("SELECT available_copies FROM books WHERE title = ?");
    $stmt->execute([$bookTitle]);
    $book = $stmt->fetch();

    if ($book && $book['available_copies'] > 0) {
        // Smanjenje dostupnih primeraka
        $pdo->prepare("UPDATE books SET available_copies =
             available_copies - 1 WHERE title = ?") ->execute([$bookTitle]);

        // Dodavanje rezervacije
        $pdo->prepare("INSERT INTO reservations (user_id, book_title, reserved_at)
                       VALUES (?, ?, NOW())")
            ->execute([$userId, $bookTitle]);

        echo "Knjiga je uspešno rezervisana!";
    } else {
        echo "Nažalost, svi primerci ove knjige su trenutno zauzeti.";
    }
} else {
    echo "Morate biti ulogovani da biste rezervisali knjigu.";
}
?>
```

Funkcionalnost za vraćanje knjige

```
<?php
session_start();
require 'config.php';

if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_SESSION['user_id'])) {
    $userId = $_SESSION['user_id'];
    $bookTitle = $_POST['book_title'];

    // Provera da li korisnik ima rezervaciju za ovu knjigu
    $stmt = $pdo->prepare("SELECT * FROM reservations WHERE user_id = ?
                           AND book_title = ?");
    $stmt->execute([$userId, $bookTitle]);
    $reservation = $stmt->fetch();

    if ($reservation) {
        // Povećanje dostupnih primeraka
        $pdo->prepare("UPDATE books SET available_copies =
                       available_copies + 1 WHERE title = ?")
            ->execute([$bookTitle]);

        // Brisanje rezervacije
        $pdo->prepare("DELETE FROM reservations WHERE user_id = ? AND
                       book_title = ?")->execute([$userId, $bookTitle]);

        echo "Knjiga je uspešno vraćena!";
    } else {
        echo "Nemate rezervaciju za ovu knjigu.";
    }
} else {
    echo "Morate biti ulogovani da biste vratili knjigu.";
}
?>
```

Korisni resursi

- Najbolji vodič po mom mišljenju:
  https://phpgurukul.com/php-pdo-tutorials/

- Ostali vodiči i priručnici sa primerima

1. https://medium.com/@vihangamallawaarachchi.dev/mastering-php-sessions-a-comprehensive-guide-82ea171734b1
2. https://www.w3schools.com/php/php_sessions.asp
3. https://reintech.io/blog/managing-sessions-cookies-php

- Koristan video tutorijal:
  https://youtube.com/playlist?list=PL3Y-E4YSE4waHmKVoZ5CsviKygYUA8B92&si=q28paDNbchlehfyf