

## PHP + bazy danych

Celem ćwiczenia jest przygotowanie prostej aplikacji internetowej wykorzystującej technologię PHP. Ćwiczenie prezentuje podstawowe aspekty poprawnego programowania aplikacji internetowej wykorzystującej bazę danych (wiązanie zmiennych, ochrona przed włamaniem typu „SQL injection”, rozróżnienie między metodami GET i POST, ochrona przed wielokrotnym zgłoszeniem tego samego formularza).

Ćwiczenie można wykonać na dowolnym komputerze, na którym zainstalowano serwer HTTP (np. Apache) z obsługą PHP oraz bazę danych MySQL. Rozwiązania ćwiczeń omawianych w poniższym zestawie zostały przygotowane z wykorzystaniem pakietu XAMPP (dostępnego pod adresem: <http://www.apachefriends.org/en/xampp.html>), który jest zestawem programów pozwalających na szybkie rozpoczęcie pracy ze skryptami PHP. W skład pakietu wchodzi, między innymi, następujące, wstępnie skonfigurowane narzędzia: Apache, PHP, MySQL, FileZilla, Perl, OpenSSL, MercuryMail, Webalizer i inne.

1. Uruchom panel kontrolny pakietu XAMPP i uruchom serwery Apache i MySQL. Otwórz w przeglądarce adres <http://localhost/phpmyadmin/> i utwórz nową bazę danych o nazwie **instytut**.
2. Utwórz tabelę **etaty** o strukturze:

KOLUMNA	TYP	WŁAŚCIWOŚCI
NAZWA	VARCHAR(10)	PRIMARY KEY
PLACA_OD	DECIMAL(6,2)	NULL
PLACA_DO	DECIMAL(6,2)	NULL

3. Po utworzeniu tabeli kliknij na zakładkę **Import** w pasku narzędziowym. Utwórz plik o nazwie *etaty.sql* i wypełnij go poniższą zawartością. Następnie, wykorzystaj formularz do importowania danych aby załadować zawartość pliku do tabeli **etaty**. Upewnij się, że jako format importowanego pliku zaznaczono **SQL**.

```
INSERT INTO `etaty` (`NAZWA`, `PLACA_OD`, `PLACA_DO`) VALUES
  ('PROFESOR', 3000.00, 4000.00),
  ('ADIUNKT', 2510.00, 3000.00),
  ('ASYSTENT', 1500.00, 2100.00),
  ('DOKTORANT', 800.00, 1000.00),
  ('SEKRETARKA', 1470.00, 1650.00),
  ('DYREKTOR', 4280.00, 5100.00);
```

4. Po załadowaniu danych kliknij na zakładkę **Browse** i sprawdź, czy dane zostały poprawnie załadowane do tabeli **etaty**.
5. Analogicznie, utwórz tabelę **zespolo** o strukturze jak poniżej.

KOLUMNA	TYP	WŁAŚCIWOŚCI
ID_ZESP	SMALLINT(6)	PRIMARY KEY
NAZWA	VARCHAR(20)	NULL
ADRES	VARCHAR(50)	NULL

6. Utwórz plik `zespoly.sql`, wypełnij go poniższą zawartością i załaduj zawartość pliku do tabeli `zespoly`.

```
INSERT INTO `zespoly` (`ID_ZESP`, `NAZWA`, `ADRES`) VALUES
(10, 'ADMINISTRACJA', 'PIOTROWO 2'),
(20, 'SYSTEMY ROZPROSZONE', 'PIOTROWO 3A'),
(30, 'SYSTEMY EKSPERCKIE', 'STRZELECKA 14'),
(40, 'ALGORYTMY', 'WIENIAWSKIEGO 16'),
(50, 'BADANIA OPERACYJNE', 'MIELZYNSKIEGO 30');
```

7. Kliknij na zakładkę **SQL** w panelu nawigacyjnym i umieść w oknie następujące polecenie:

```
DROP TABLE IF EXISTS `pracownicy`;
CREATE TABLE `pracownicy` (
  `ID_PRAC` SMALLINT(6),
  `NAZWISKO` VARCHAR(30) NOT NULL,
  `IMIE` VARCHAR(15) NULL,
  `ETAT` VARCHAR(10) NULL,
  `ID_SZEFA` SMALLINT(6),
  `ZATRUDNIONY` DATE DEFAULT NULL,
  `PLACA_POD` DECIMAL(6,2) DEFAULT NULL,
  `PLACA_DOD` DECIMAL(6,2) DEFAULT NULL,
  `ID_ZESP` SMALLINT(6) DEFAULT NULL,
  PRIMARY KEY (`ID_PRAC`),
  FOREIGN KEY (`ID_SZEFA`) REFERENCES `pracownicy`(`ID_PRAC`),
  FOREIGN KEY (`ID_ZESP`) REFERENCES `zespoly`(`ID_ZESP`)
) ENGINE=MyISAM;
```

a następnie wypełnij tabelę `pracownicy` poniższymi danymi:

```
INSERT INTO `pracownicy` VALUES
(100, 'Marecki', 'Jan', 'DYREKTOR', NULL, '1968-01-01', 4730.00, 980.50, 10),
(110, 'Janicki', 'Karol', 'PROFESOR', 100, '1973-05-01', 3350.00, 610.00, 40),
(120, 'Nowicki', 'Pawel', 'PROFESOR', 100, '1977-09-01', 3070.00, NULL, 30),
(130, 'Nowak', 'Piotr', 'PROFESOR', 100, '1968-07-01', 3960.00, NULL, 20),
(140, 'Kowalski', 'Adam', 'PROFESOR', 130, '1975-09-15', 3230.00, 805.00, 20),
(150, 'Grzybowska', 'Maria', 'ADIUNKT', 130, '1977-09-01', 2845.50, NULL, 20),
(160, 'Krakowska', 'Anna', 'SEKRETARKA', 130, '1985-03-01', 1590.00, NULL, 20),
(170, 'Opolski', 'Roman', 'ASYSTENT', 130, '1992-10-01', 1839.70, 480.50, 20),
(190, 'Kotarski', 'Konrad', 'ASYSTENT', 140, '1993-09-01', 1971.00, NULL, 20),
(180, 'Makowski', 'Marek', 'ADIUNKT', 100, '1985-02-20', 2610.20, NULL, 10),
(200, 'Przywarek', 'Leon', 'DOKTORANT', 140, '1994-07-15', 900.00, NULL, 30),
(210, 'Kotlarczyk', 'Jan', 'DOKTORANT', 130, '1993-10-15', 900.00, 570.60, 30),
(220, 'Siekierski', 'Jacek', 'ASYSTENT', 110, '1993-10-01', 1889.00, NULL, 20),
(230, 'Dolny', 'Tomasz', 'ASYSTENT', 120, '1992-09-01', 1850.00, 390.00, NULL);
```

8. Powróć do głównego ekranu programu phpMyAdmin. Kliknij na odnośnik **Privileges**. Kliknij na odnośnik **Add a new user** i utwórz lokalnego użytkownika `scott` z hasłem `tiger`. Zwróć uwagę, aby użytkownik był lokalny. Nie generuj nowej bazy danych dla użytkownika i nie nadawaj użytkownikowi żadnych przywilejów globalnych.
9. Kliknij na ikonę edycji przywilejów przy użytkowniku `scott`. Znajdź grupę przywilejów specyficznych dla danej bazy danych i z listy rozwijanej wybierz bazę danych `instytut`.
10. Kliknij odnośnik **Check All** aby nadać użytkownikowi `scott` wszystkie przywileje do bazy danych `instytut` i kliknij przycisk **Go**.

11. Pierwszy skrypt nawiąże połączenie z bazą danych i wyświetli podstawowe informacje o serwerze bazy danych. Utwórz plik *file01.php* i umieść w nim poniższy kod.

```
<?php
$link = mysqli_connect("localhost", "scott", "tiger", "instytut");

if (!$link) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
}

printf("<h1>Host information</h1>");
printf("<ul>");
printf("<li>server information: %s</li>", mysqli_get_server_info($link));
printf("<li>character version: %s</li>", mysqli_character_set_name($link));
printf("<li>host information: %s</li>", mysqli_get_host_info($link));
printf("</ul>");

mysqli_close($link);
?>
```

12. Drugi skrypt obrazuje sposób wydania najprostszego zapytania do bazy danych i wyświetlenia wyniku. Utwórz plik *file02.php* i umieść w nim poniższy kod.

```
<?php
$link = mysqli_connect("localhost", "scott", "tiger", "instytut");

if (!$link) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
}

$result = mysqli_query($link, "SELECT * FROM zespol");

if (!$result) {
    printf("Query failed: %s\n", mysqli_error($link));
}

printf("<h1>ZESPOLY</h1>");
printf("<table border='1'>");

$fields = mysqli_fetch_fields($result);

printf("<tr><th>%s</th><th>%s</th><th>%s</th></tr>",
    $fields[0]->name, $fields[1]->name, $fields[2]->name);

while ($row = mysqli_fetch_array($result))
    printf("<tr><td>%s</td><td>%s</td><td>%s</td></tr>", $row[0], $row[1], $row[2]);

printf("</table>");
printf("<i>query returned %d rows </i>", mysqli_num_rows($result));

mysqli_free_result($result);

mysqli_close($link);
?>
```

13. Kolejny skrypt pokazuje, w jaki sposób odczytać wynik zapytania w postaci tablicy asocjacyjnej. Utwórz plik *file03.php* i umieść w nim poniższy kod. Zwróć uwagę na różnice między kodem z pliku *file03.php* i pliku *file02.php* dotyczące sposobu pobrania wiersza z wyniku.

```
<?php
$link = mysqli_connect("localhost", "scott", "tiger", "instytut");

if (!$link) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
}

$result = mysqli_query($link, "SELECT * FROM etaty ORDER BY placa_od");

if (!$result) {
    printf("Query failed: %s\n", mysqli_error($link));
}

printf("<h1>ETATY</h1>");

printf("<table border='1'>");

printf("<tr><th>NAZWA</th><th>PLACA_OD</th><th>PLACA_DO</th></tr>");

while ($row = mysqli_fetch_assoc($result))
    printf("<tr><td>%s</td><td>%6.2f</td><td>%6.2f</td></tr>",
        $row["NAZWA"], $row["PLACA_OD"], $row["PLACA_DO"]);

printf("</table>");
printf("<i>query returned %d rows </i>", mysqli_num_rows($result));

mysqli_free_result($result);

mysqli_close($link);
?>
```

14. Interfejs MySQLi dostarcza także trzeciego sposobu odczytywania wierszy wyniku. Utwórz plik *file04.php*, umieść w nim poniższy kod i porównaj ten kod z poprzednimi dwoma rozwiązaniami.

```
<?php
$link = mysqli_connect("localhost", "scott", "tiger", "instytut");

if (!$link) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
}

$result = mysqli_query($link, "SELECT * FROM pracownicy");

if (!$result) {
    printf("Query failed: %s\n", mysqli_error($link));
}

printf("<h1>PRACOWNICY</h1>");

printf("<table border='1'>");

printf("<tr><th>ID_PRAC</th><th>NAZWISKO</th><th>ETAT</th><th>ZATR.</th></tr>");

while ($obj = mysqli_fetch_object($result))
    printf("<tr><td>%d</td><td>%s</td><td>%s</td><td>%s</td></tr>",
        $obj->ID_PRAC, $obj->NAZWISKO, $obj->ETAT, $obj->ZATRUDNIENIE);

printf("</table>");
printf("<i>query returned %d rows </i>", mysqli_num_rows($result));

mysqli_free_result($result);

mysqli_close($link);
?>
```

15. Wiązanie zmiennych przy zapytaniach do bazy danych ma kolosalne znaczenie: pozwala uniknąć ataków typu „SQL Injection” oraz umożliwia optymalizatorowi zapytań efektywne wykorzystanie skompilowanych planów wykonania zapytania. Utwórz plik *file05.php* i umieść w nim poniższy kod, prześledź bardzo uważnie procedurę przygotowywania i wykonywania zapytania wykorzystującego zmienne wiązane.

```
<?php
$link = mysqli_connect("localhost", "scott", "tiger", "instytut");

if (!$link) {
    printf("Connect failed: %s\n", mysql_connect_error());
    exit();
}

printf("<h1>Pracownicy</h1>");

$sql = "SELECT nazwisko, etat FROM pracownicy WHERE id_prac = ?";
$stmt = mysqli_stmt_init($link);

mysqli_stmt_prepare($stmt, $sql);

for ($i=100; $i<230; $i+=10)
{
    mysqli_stmt_bind_param($stmt, "i", $i);
    mysqli_stmt_execute($stmt);
    mysqli_stmt_bind_result($stmt, $nazwisko, $etat);
    mysqli_stmt_fetch($stmt);

    printf("$nazwisko pracuje jako $etat <br/>");
}

mysqli_stmt_close($stmt);

mysqli_close($link);
?>
```

16. Ostatni przykład obrazuje fundamentalny wzorzec postępowania z aplikacjami internetowymi, POST→REDIRECT→GET. Utwórz skrypt *file06.php* i umieść w nim poniższy kod. Przetestuj aplikację. Jakie dostrzegasz błędy i wady tego rozwiązania?

```
<?php
$link = mysqli_connect("localhost", "scott", "tiger", "instytut");
if (!$link) {
    printf("Connect failed: %s\n", mysqli_connect_error());
    exit();
}
print<<<KONIEC
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
</head>
<body>
    <form action="file06.php" method="POST">
        id_prac <input type="text" name="id_prac">
        nazwisko <input type="text" name="nazwisko">
        <input type="submit" value="Wstaw">
        <input type="reset" value="Wyczyszc">
    </form>
KONIEC;
if (isset($_POST['id_prac']) &&
    is_numeric($_POST['id_prac']) &&
    is_string($_POST['nazwisko']))
{
    $sql = "INSERT INTO pracownicy(id_prac,nazwisko) VALUES(?,?)";
    $stmt = $link->prepare($sql);
    $stmt->bind_param("is", $_POST['id_prac'], $_POST['nazwisko']);
    $result = $stmt->execute();
    if (!$result) {
        printf("Query failed: %s\n", mysqli_error($link));
    }
    $stmt->close();
}
$sql = "SELECT * FROM pracownicy";
$result = $link->query($sql);
foreach ($result as $v) {
    echo $v["ID_PRAC"]." " . $v["NAZWISKO"]."<br/>";
}
$result->free();
$link->close();
?>
```

Spróbuj wprowadzić niepoprawne dane (np. zduplikuj wartość klucza podstawowego). Nawiguj między kolejnymi stronami za pomocą przycisków przeglądarki, odśwież zawartość okna w przeglądarce. Jak widzisz powyższe rozwiązanie nie jest idealne, ponieważ nie zastosowano w nim wzorca POST → REDIRECT → GET. Popraw powyższe rozwiązanie dzieląc je na trzy osobne pliki:

- *form06\_post.php* – w tym pliku umieść formularz do wstawiania nowego pracownika, który wysyła dane żądaniem POST na adres *form06\_redirect.php*; dodaj również link pozwalający przejść na stronę *form06\_get.php* zawierającą listę wszystkich pracowników;
- *form06\_redirect.php* – w tym pliku umieść obsługę żądania POST z danymi nowego pracownika; po przetwarzaniu danych zakończonym sukcesem wykonaj przekierowanie na stronę *form06\_get.php* z odpowiednim komunikatem o sukcesie, natomiast po przetwarzaniu zakończonym błędem wykonaj przekierowanie z powrotem na stronę *form06\_post.php* z odpowiednim komunikatem o błędzie (wiadomości powinny zostać przekazane w sesji w taki sposób, aby wyświetliły się tylko raz, tj. po odświeżeniu strony z komunikatem o sukcesie/błędzie, powinien on zniknąć);
- *form06\_get.php* – tutaj wyświetl listę wszystkich pracowników; dodaj również link prowadzący do strony umożliwiającej dodanie nowego pracownika *form06\_post.php*.