

PHP i PDO — CRUD produktów

Czas: 2 × 45 min

praca indywidualna lub w parach

Cel: Zbudowanie w PHP i PDO kompletnego panelu do zarządzania produktami sklepu: odczyt, dodawanie, edycja i usuwanie (CRUD) z zapytaniami przygotowanymi.

Po zajęciach potrafisz:

- łączy się z bazą przez PDO z trybem `ERRMODE_EXCEPTION` i kodowaniem `utf8mb4`
- wykonuje `SELECT`, `INSERT`, `UPDATE` i `DELETE` zapytaniami przygotowanymi (`prepare/execute`)
- stosuje wzorzec `Post/Redirect/Get` i usuwa dane tylko metodą `POST`
- waliduje dane formularza po stronie serwera i wypisuje je przez `htmlspecialchars`
- obsługuje błąd naruszenia klucza obcego (1451) przy usuwaniu

Sprzęt i materiały

- komputer z XAMPP (Apache, PHP 8, MariaDB 10.4)
- świeżo zaimportowana baza `sklep`

Oprogramowanie

- **XAMPP** — folder projektu
`C:\xampp\htdocs\panel`
[apachefriends.org/download.html](https://www.apachefriends.org/download.html)
- **VS Code** — edycja PHP
code.visualstudio.com
- **Dokumentacja PHP — PDO** —
`PDO::prepare`, `PDOStatement::execute`,
`fetch`, `fetchAll`, `rowCount`
php.net/manual/pl/book.pdo.php

Przygotowanie przed zajęciami

- Zaimportuj ponownie `sklep.sql` (zadanie zmienia tabelę `produkty`).
- Przypomnij sobie z prezentacji 2.12 i 2.13 budowę połączenia PDO i zapytań przygotowanych.

Bezpieczeństwo

- Każda wartość z `$_GET` i `$_POST` trafia do SQL wyłącznie jako parametr `execute()`. Nie sklejaj zapytań z danymi.
- Usuwanie wykonuj tylko metodą `POST` (formularz z przyciskiem) — link `GET` mógłby zostać „kliknięty” przez przeglądarkę lub robota.
- Panel nie ma logowania, więc działa tylko lokalnie. Nie publikuj go w internecie w tej postaci.
- Przed testami usuwania zrób kopię tabeli: `CREATE TABLE produkty_kopia AS SELECT * FROM produkty;`

Przebieg zadania

1 Połączenie i lista (Read)

Struktura projektu: `połaczenie.php`, `index.php`, `formularz.php`, `dodaj.php`, `edytuj.php`, `usun.php`.

```
<?php
// polaczenie.php
$pdo = new PDO('mysql:host=localhost;dbname=sklep;charset=utf8mb4', 'root', '', [
    PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    PDO::ATTR_EMULATE_PREPARES   => false,
]);
function h($s) { return htmlspecialchars((string)$s, ENT_QUOTES, 'UTF-8'); }
```

1. Z1. Utwórz `połaczenie.php` jak wyżej i dołączaj go w pozostałych plikach przez `require`.
2. Z2. W `index.php` wyświetl tabelę produktów: id, nazwa, nazwa kategorii (LEFT JOIN, brak — „—”), cena, stan i kolumna akcji: link „edytuj” (`edytuj.php?id=...`) i przycisk „usuń” w formularzu POST do `usun.php` z ukrytym polem `id` i potwierdzeniem `confirm()`.
3. Z3. Nad tabelą wyświetl komunikat zależny od parametru `?msg=` (dodano, zapisano, usunięto, w_zamowieniach). Tekst komunikatu wybieraj z tablicy w PHP — nie wypisuj wartości z adresu.

2 Dodawanie (Create)

1. Z4. W `formularz.php` napisz funkcję `waliduj(array $d): array` zwracającą listę błędów: nazwa niepusta i do 100 znaków, cena liczbą > 0 , stan liczbą całkowitą ≥ 0 (`filter_var` z `FILTER_VALIDATE_INT`).
2. Z5. Napisz funkcję `formularz()` wypisującą pola: nazwa, lista kategorii z bazy (pierwsza opcja „— brak —” z pustą wartością), cena, stan. Pola mają zachować wpisane wartości po błędzie walidacji (`value` przez `h()`).
3. Z6. W `dodaj.php` po poprawnej walidacji wykonaj INSERT z parametrami nazwanymi (pusta kategoria $\rightarrow$ NULL, data dodania `CURDATE()`) i przekieruj `header('Location: index.php?msg=dodano');` `exit;`.
4. Z7. Dodaj produkt `Kamera internetowa 4K "Pro"` (kategoria Akcesoria, 349.99 zł, stan 8). Sprawdź, że cudzysłowy w nazwie wyświetlają się poprawnie, i odśwież stronę po dodaniu (F5) — czy produkt dodał się drugi raz?

3 Edycja (Update) i usuwanie (Delete)

1. Z8. W `edytuj.php` dla GET pobierz produkt po id zapytaniem przygotowanym; gdy go nie ma — kod 404 i komunikat „Nie ma takiego produktu.”. Dla POST zwaliduj dane, wykonaj UPDATE i przekieruj z `msg=zapisano`.
2. Z9. Zmień kamerze nazwę na `Kamera internetowa 4K`, cenę na 299 zł i stan na 10. Sprawdź zmiany w phpMyAdmin.
3. Z10. W `usun.php` przyjmij tylko POST (dla innej metody kod 405). Wykonaj DELETE w bloku try/catch; przy błędzie 1451 (`$e->errorInfo[1]`) przekieruj z `msg=w_zamowieniach`.
4. Z11. Usuń kamerę, a potem spróbuj usunąć „Laptop Vertex 14” (id 1). Otwórz w przeglądarce `usun.php?id=15` — co się stało?
5. Z12. Sprawdź walidację: pusta nazwa, cena -5, stan 1.5. Wszystkie trzy błędy mają się wyświetlić naraz, a wpisane wartości pozostać w polach.

4 Sprawdzenie i oddanie

1. Z13. Wyświetl w phpMyAdmin wynik `SELECT COUNT(*), MAX(id_produktu) FROM produkty;` i porównaj ze stanem początkowym. Wyjaśnij, dlaczego nowy produkt dostałby id 17, a nie 16.
2. Spakuj folder `panel` do archiwum `nazwisko_r2_21.zip` i dołącz zrzuty ekranu listy, formularza z błędami i komunikatu o produkcie w zamówieniach.

W sprawozdaniu umieść

- ☐ pliki projektu (6 plików PHP) w archiwum zip
- ☐ zrzuty ekranu: lista produktów, formularz z błędami walidacji, komunikat po próbie usunięcia produktu z zamówień
- ☐ tabela testów Z7–Z13 z wynikami
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

1. Na czym polega wzorzec Post/Redirect/Get i jaki problem rozwiązuje (Z7)?
2. Dlaczego usuwanie nie powinno być wykonywane linkiem GET?
3. Co oznacza błąd 1451 i jak inaczej można by rozwiązać usuwanie produktów, które były już zamawiane?
4. Dlaczego walidacja w JavaScript lub atrybut `required` w HTML nie wystarczy i trzeba sprawdzać dane w PHP?

Kryteria oceny

Kryterium	Punkty
Połączenie PDO i lista z komunikatami (Z1–Z3)	3
Dodawanie z walidacją i PRG (Z4–Z7)	5
Edycja z obsługą braku produktu (Z8–Z9)	4
Usuwanie POST z obsługą błędu 1451 (Z10–Z12)	4
Testy, zrzuty i odpowiedzi na pytania	4
Razem	20

Sprawozdanie przygotuj na wzorze dołączonym do zadania (plik .docx o tym samym numerze).