

PHP — transakcje, błędy i wzorzec repozytorium

Czas: 2 × 45 min

praca indywidualna lub w parach

Cel: Zaprogramowanie w PHP składania zamówienia jako jednej transakcji, która przy każdym błędzie wycofuje wszystkie zmiany, oraz uporządkowanie kodu dostępu do bazy w klasach.

Po zajęciach potrafisz:

- wykonuje kilka poleceń SQL w transakcji PDO (`beginTransaction`, `commit`, `rollBack`)
- obsługuje wyjątki `PDOException` i własne wyjątki, przekazując błąd dalej po wycofaniu transakcji
- przechowuje dane logowania do bazy w pliku konfiguracyjnym poza katalogiem publicznym
- wydziela zapytania SQL do klasy repozytorium i używa jej w kodzie aplikacji
- blokuje odczytany wiersz (`SELECT ... FOR UPDATE`), aby uniknąć sprzedaży nieistniejącego towaru

Sprzęt i materiały

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

Oprogramowanie

- **XAMPP** — PHP w konsoli:
`C:\xampp\php\php.exe` (przycisk „Shell”, polecenie `php`)
apachefriends.org/download.html
- **VS Code** — projekt z podkatalogiem `src`
code.visualstudio.com
- **Dokumentacja PHP — transakcje PDO** — `beginTransaction`, `commit`, `rollBack`, `inTransaction`
php.net/manual/pl/pdo.transactions.php

Przygotowanie przed zajęciami

- Zaimportuj ponownie `sklep.sql` — testy zmieniają stany magazynowe i dodają zamówienia.
- Przypomnij sobie procedurę `zloz_zamowienie` z prezentacji 2.07 oraz klasy i konstruktory w PHP.

Bezpieczeństwo

- Plik `config.php` z hasłem trzymaj poza `htdocs` (np. `C:\xampp\sklep_config`) — pliki w `htdocs` może pobrać każdy, kto zna adres. W projekcie z Git dopisz go do `.gitignore`.
- Komunikaty dla użytkownika nie mogą zawierać treści błędów SQL — szczegóły zapisuj funkcją `error_log()`.
- Przed testami zrób kopię: `CREATE TABLE produkty_kopia AS SELECT * FROM produkty;` albo zaimportuj `sklep.sql` po zajęciach.

Przebieg zadania

1 Konfiguracja i połączenie

Struktura plików:

```
C:\xampp\sklep_config\config.php      <- dane logowania (poza htdocs)
C:\xampp\htdocs\sklep22\zamow.php     <- skrypt testowy
C:\xampp\htdocs\sklep22\src\Baza.php
C:\xampp\htdocs\sklep22\src\ProduktRepozytorium.php
C:\xampp\htdocs\sklep22\src\ZamowienieSerwis.php
```

1. Z1. Utwórz `config.php`, który zwraca (`return [...]`) tablicę z kluczami `dsn`, `uzytkownik`, `haslo`.
2. Z2. Napisz klasę `Baza` z metodą statyczną `polacz(): PDO`, która wczytuje konfigurację (`require __DIR__ . '/.././.././sklep_config/config.php'`) i tworzy PDO z `ERRMODE_EXCEPTION`, `FETCH_ASSOC` i bez emulacji zapytań przygotowanych.
3. Z3. Sprawdź w przeglądarce, że adres `http://localhost/sklep_config/config.php` nie istnieje (404), a `http://localhost/sklep22/zamow.php` działa (na razie może wypisać tylko „OK”).

2 Repozytorium produktów

Repozytorium to klasa, która zbiera w jednym miejscu wszystkie zapytania dotyczące jednej tabeli. Reszta aplikacji nie pisze SQL — wywołuje metody.

1. Z4. Napisz klasę `ProduktRepozytorium` z konstruktorem przyjmującym `PDO` i metodami: `znajdz(int $id): ?array`, `wKategorii(int $idKategorii): array`.
2. Z5. Dodaj metodę `znajdzDoSprzedazy(int $id): ?array` z `SELECT ... FOR UPDATE` (do użycia w transakcji) oraz `zmniejszStan(int $id, int $ile): void` z warunkiem `stan >= ?` — gdy `rowCount()` nie jest równe 1, rzuć `RuntimeException`.
3. Z6. Przetestuj w `zamow.php`: wypisz nazwę i cenę produktu 5 oraz liczbę produktów w kategorii 4.

3 Zamówienie w transakcji

1. Z7. Napisz klasę `ZamowienieSerwis` z metodą `zloz(int $idKlienta, array $pozycje): int` (`$pozycje = [id_produkta => ilość]`). Pusta tablica — `InvalidArgumentException` jeszcze przed transakcją.
2. Z8. W transakcji: INSERT do `zamowienia` (status 'nowe', `NOW()`), `lastInsertId()`, a dla każdej pozycji: odczyt produktu z blokadą, sprawdzenie, czy istnieje i czy stan wystarcza (inaczej `RuntimeException` z czytelnym komunikatem), INSERT pozycji z aktualną ceną i zmniejszenie stanu. Na końcu `commit()` i zwrot numeru zamówienia.
3. Z9. Cały kod transakcji umieść w `try`; w `catch (Throwable $e)` wywołaj `rollback()` i rzuć wyjątek dalej (`throw $e;`).
4. Z10. W `zamow.php` wykonaj pięć testów, każdy w osobnym try/catch (PDOException: zapis do dziennika i numer błędu, Exception: komunikat): T1 klient 3: produkt 7 × 2 i 11 × 1; T2 klient 3: produkt 8 × 1 i 12 × 10; T3 klient 3: produkt 7 × 1 i 999 × 1; T4 klient 99: produkt 7 × 1; T5 klient 3 bez pozycji. Na koniec wypisz stany produktów 7, 8, 11, 12 i liczbę zamówień.

4 Sprawdzenie w bazie

1. Z11. W phpMyAdmin wyświetl zamówienia o numerze większym niż 12 i ich pozycje. Czy po nieudanych testach zostały „puste” zamówienia albo zmienione stany?
2. Z12. Dodaj jedno zamówienie ręcznie (`INSERT INTO zamowienia (id_klienta, data_zamowienia) VALUES (1, NOW());`) i sprawdź `LAST_INSERT_ID()`. Wyjaśnij lukę w numeracji.
3. Z13. Sprawdź, co by się stało bez transakcji: zakomentuj `beginTransaction`, `commit` i `rollback`, zaimportuj bazę ponownie, uruchom testy i porównaj wynik Z11. Przywróć transakcję.

W sprawozdaniu umieść

- ☐ kod czterech plików PHP i `config.php` (z zamaskowanym hasłem)
- ☐ wynik działania `zamow.php` (zrzut lub tekst z konsoli)
- ☐ wyniki zapytań Z11 i Z12 oraz wyjaśnienie luki w numeracji
- ☐ porównanie działania z transakcją i bez niej (Z13)
- ☐ odpowiedzi na pytania kontrolne

Pytania kontrolne

- Dlaczego po `rollback()` wyjątek jest rzucany dalej, zamiast od razu wypisać komunikat w klasie serwisu?
- Po co `SELECT ... FOR UPDATE` w `znajdzDoSprzedazy` i dlaczego w `zmniejszStan` jest jeszcze warunek `stan >= ??`?
- Jakie korzyści daje repozytorium w porównaniu z zapytaniami SQL rozrzuconymi po plikach strony? Jak ten pomysł rozwijają biblioteki ORM (np. Doctrine, Eloquent)?
- Dlaczego plik z hasłem do bazy nie powinien leżeć w katalogu `htdocs`?

Kryteria oceny

Kryterium	Punkty
Konfiguracja poza <code>htdocs</code> i klasa Baza (Z1–Z3)	3
ProduktRepozytorium z metodami i testem (Z4–Z6)	4
Transakcja w ZamowienieSerwis z obsługą błędów (Z7–Z9)	6
Testy T1–T5 i sprawdzenie w bazie (Z10–Z13)	4
Odpowiedzi na pytania kontrolne	3
Razem	20

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