

Projekt końcowy cz. 2 — backend

Czas: 2 × 45 min

praca w zespołach 2–3-osobowych (każda osoba na własnej gałęzi funkcji)

Cel: Zbudowanie zaplecza aplikacji zespołowej: logowanie i role, CRUD głównego zasobu, reguła biznesowa, walidacja po stronie serwera i zabezpieczenia wg listy kontrolnej, potwierdzone planem testów ręcznych.

Po zajęciach potrafisz:

- implementuje rejestrację, logowanie i wylogowanie z haszowaniem haseł i odświeżaniem sesji
- ogranicza dostęp do stron i rekordów według roli i właściciela
- programuje CRUD z zapytaniami przygotowanymi, walidacją i komunikatami flash (PRG)
- zabezpiecza formularze tokenem CSRF i escapuje dane wyjściowe
- planuje i wykonuje testy ręczne oraz dokumentuje wyniki

Sprzęt i materiały

- komputer z XAMPP, VS Code i Git;
- repozytorium zespołu z zadania 31
- folder `r2_31_projekt` z Pliki_do_zadan (lista kontrolna bezpieczeństwa, plan testów)

Oprogramowanie

- **XAMPP** — PHP 8.2+, MariaDB, phpMyAdmin
apachefriends.org/download.html
- **Visual Studio Code** — rozszerzenie PHP Intelephense (opcjonalnie)
code.visualstudio.com
- **Dokumentacja PHP — PDO i hasła** —
password_hash:
https://www.php.net/password_hash
php.net/manual/pl/book.pdo.php

Przygotowanie przed zajęciami

- Zaktualizujcie specyfikację o uwagi nauczyciela z zadania 31; każda osoba wybiera swoje Issues z kamienia „cz. 2 backend”.
- Przypomnij sobie zadania 9 (PDO CRUD), 12 (logowanie) i 14 (lista kontrolna bezpieczeństwa).

Bezpieczeństwo

- Testuj zabezpieczenia wyłącznie na własnej, lokalnej instalacji — sprawdzasz, czy aplikacja poprawnie odrzuca nieprawidłowe żądania, a nie próbujesz łamać cudzych systemów.
- Hasła kont testowych zapisuj tylko w README jako dane demonstracyjne; nie używaj swoich prawdziwych haseł.

Przebieg zadania

1 Wspólna podstawa (lider)

Jeden plik dołączany na początku każdej strony: konfiguracja z .env (jak w zadaniu 29), połączenie PDO, sesja, nagłówki bezpieczeństwa i funkcje pomocnicze.

```
// src/bootstrap.php – wymagane funkcje
e($s) // htmlspecialchars
przekieruj($adres) // header Location + exit
flash($tekst, $typ) // komunikat na jedno wyświetlenie
wymagaj_logowania(), wymagaj_admina()
csrf_token(), csrf_pole(), sprawdz_csrf()
```

1. Napisz `src/bootstrap.php` z funkcjami z ramki. Ciasteczko sesji ustaw przez `session_set_cookie_params` (httponly, samesite Lax, secure przy HTTPS) przed `session_start()`.
2. Wyślij nagłówki `X-Content-Type-Options: nosniff`, `Referrer-Policy` i `Content-Security-Policy` dopuszczający tylko własne pliki i używany CDN. Sprawdź je w F12 → Network.
3. Przygotuj wspólny nagłówek i stopkę strony (`src/widoki/`) z menu zależnym od roli i miejscem na komunikat flash. Scal przez pull request — pozostali zaczynają od tej wersji.

2 Konta i role

1. Rejestracja: walidacja loginu (wzorzec), długości pól, hasła (min. 10 znaków) i powtórzenia hasła; komunikaty przy polach, zachowanie wpisanych wartości (bez hasła), sprawdzenie unikalności loginu.
2. Logowanie z `password_verify` i `session_regenerate_id(true)`; ten sam komunikat przy złym loginie i złym hasle. Wylogowanie formularzem POST z tokenem CSRF.
3. Zabezpiecz strony: użytkownik niezalogowany → przekierowanie do logowania, zwykły użytkownik na stronie administratora → kod 403. Sprawdzenie ma być po stronie serwera na każdej stronie panelu.

3 Główny zasób i reguła biznesowa

Przykład z projektu wzorcowego: rezerwacja sprzętu odrzucana, gdy koliduje z inną rezerwacją (oczekuje lub zatwierdzona), zaczyna się w przeszłości albo trwa ponad 14 dni.

1. Administrator: lista, dodawanie, edycja i usuwanie głównego zasobu (jedna strona z formularzem lub osobne). Walidacja wszystkich pól na serwerze; błąd unikalności (kod 23000) zamień na komunikat przy polu.
2. Usuwanie i zmiany stanu wyłącznie metodą POST z tokenem CSRF i potwierdzeniem; po zapisie przekierowanie z komunikatem flash (PRG).
3. Użytkownik: tworzenie własnych rekordów (np. rezerwacji) z regułą biznesową sprawdzaną w osobnej funkcji, lista „moje ...” i anulowanie tylko własnych rekordów — warunek `uzytkownik_id = ?` z id z sesji, nie z formularza.
4. Administrator: zmiana statusu rekordów (np. zatwierdź / odrzuć / zwrot) z dozwolonymi przejściami (UPDATE ... WHERE status = poprzedni).

4 Lista kontrolna i testy

1. Przejdź listę `lista_kontrolna_bezpieczenstwa.txt` dla całego kodu zespołu: przy każdym punkcie wpisz plik i linię, która go spełnia, albo utwórz Issue z poprawką.
2. Uzupełnij plan `testy_reczne.txt` (min. 12 przypadków, w tym T9–T11: reguła biznesowa, formularz bez tokenu, zmiana cudzego rekordu) i wykonaj testy. Błędy zgłoś jako Issues i popraw.
3. Każda osoba scala swoje funkcje przez pull request z przeglądem drugiej osoby. Na koniec oznacz wersję tagiem `v0.2-backend`.

W sprawozdaniu umieść

- ☐ lista funkcji backendu z podziałem na osoby i odnośnikami do pull requestów
- ☐ wypełniona lista kontrolna bezpieczeństwa (plik i linia przy każdym punkcie)
- ☐ plan testów ręcznych z wynikami i zrzutami T2, T4, T9, T10
- ☐ fragment kodu reguły biznesowej i sprawdzania właściciela rekordu
- ☐ odpowiedzi na pytania (indywidualnie)

Pytania kontrolne

1. Dlaczego ukrycie przycisku „Panel” w menu nie wystarcza do ochrony strony administratora?
2. Po co `session_regenerate_id(true)` zaraz po zalogowaniu?
3. Dlaczego usuwanie rekordu nie powinno działać przez zwykły odnośnik GET?
4. Dlaczego id użytkownika przy zapisie bierze się z sesji, a nie z ukrytego pola formularza?

Kryteria oceny

Kryterium	Punkty
bootstrap.php: PDO z .env, sesja z bezpiecznymi parametrami, nagłówki, funkcje pomocnicze; wspólny układ stron	3
Rejestracja z walidacją, logowanie z <code>password_verify</code> i regeneracją sesji, wylogowanie POST	3
Kontrola dostępu: przekierowanie niezalogowanych, 403 dla roli, ochrona cudzych rekordów	3
CRUD głównego zasobu z walidacją, PRG, flash, obsługą błędów unikalności	4
Reguła biznesowa i zmiana statusów z dozwolonymi przejściami	2
Lista kontrolna bezpieczeństwa i min. 12 testów ręcznych z wynikami; PR każdej osoby, tag	3
Odpowiedzi na pytania kontrolne (indywidualnie)	2
Razem	20

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