Politechnika Wrocławska

Wyniki etapu III: Ocena architektury systemu:
System e-learningowy

Sprawozdanie

Prowadzący:
dr inż. Bogumiła Hnatkowska

Wykonali:
Jakub Staniszewski 266876
Kamil Wojcieszak 264487
Kasjan Kardaś 263505

Wrocław, 27 Styczeń 2026r.

Spis treści

1) Przegląd podejść architektonicznych 3
2) Drzewo użyteczności 3
3) Analiza wybranych scenariuszy 4
4) Punkty wrażliwości i kompromisy 5
5) Ryzyka i nie-ryzyka 5
6) Wnioski 6

1) Przegląd podejść architektonicznych

Architektura ocenianego systemu e-learningowego została zaprojektowana z myślą o spełnieniu kluczowych wymagań jakościowych, takich jak wydajność, skalowalność, bezpieczeństwo, wysoka dostępność oraz zgodność z RODO. System wykorzystuje architekturę warstwową, wdrożoną w środowisku chmurowym.

Podstawowy podział architektury obejmuje:

- **Warstwę prezentacji (Frontend)** - aplikację typu SPA zrealizowaną w technologii React, serwowaną przez Nginx, odpowiedzialną za interakcję z użytkownikiem końcowym.
- **Warstwę logiki biznesowej (Backend)** - aplikację backendową (Python), udostępniającą API obsługujące użytkowników, kursy, testy, postępy oraz mechanizmy realizujące wymagania RODO.
- **Warstwę danych** - relacyjną bazę danych PostgreSQL w konfiguracji klastrowej lub replikacyjnej oraz magazyn obiektowy Amazon S3, przeznaczony do przechowywania plików multimedialnych.

Komunikacja pomiędzy warstwami realizowana jest wyłącznie poprzez zdefiniowane interfejsy API, co zapewnia luźne powiązania pomiędzy komponentami oraz umożliwia ich niezależny rozwój i skalowanie. System zostanie wdrożony w architekturze wieloinstancyjnej, z wykorzystaniem mechanizmów autoskalowania i load balancingu, eliminujących pojedyncze punkty awarii.

W zakresie bezpieczeństwa zastosowano szyfrowanie całej komunikacji z użyciem protokołu TLS oraz bezpieczne przechowywanie haseł użytkowników w postaci skrótów bcrypt z losową solą. Wymagania RODO realizowane są poprzez wydzielony moduł logiczny backendu, odpowiedzialny za kontrolowane przetwarzanie danych osobowych, ich eksport oraz trwałe usuwanie.

2) Drzewo użyteczności

Atrybut jakości	Udoskonalenie atrybutu	Scenariusze
Wydajność i skalowalność	Czas odpowiedzi na zapytanie	Utrzymanie czasu odpowiedzi API na poziomie średnio 300ms (95 percentyl)
	Liczba użytkowników	Wsparcie dla co najmniej 1 000 równoczesnych użytkowników bez spadku wydajności
Bezpieczeństwo	Ochrona danych	Wartości haseł wszystkich użytkowników nie mogą być możliwe do przeczytania
	Ochrona komunikacji	Brak możliwości przechwycenia przez użytkownika postronnego komunikacji między klientem a serwerem
Dostępność	Czas dostępności	Zapewnienie dostępności do aplikacji przez co najmniej 99% czasu
	Skalowalność	System musi obsłużyć wzmożony ruch użytkowników w okresach szczytowych
RODO	Pozyskiwanie danych	Mechanizm umożliwiający pobranie danych osobowych (JSON/CSV)
	Pozbywanie się danych	Możliwość całkowitego usunięcia konta wraz z danymi w ciągu maksymalnie 30 dni

Tabela 1: Drzewo użyteczności - atrybuty jakości i scenariusze

3) Analiza wybranych scenariuszy

Scenariusz: A1	Wsparcie dla co najmniej 1 000 równoczesnych użytkowników bez spadku wydajności.			
Atrybuty	Wydajność, Skalowalność			
Środowisko	Normalny tryb pracy			
Bodziec	Jednoczesne korzystanie z systemu przez ≥1000 użytkowników			
Odpowiedź	Frontend i backend skalują się automatycznie, aby zapewnić obsługę równoczesnych połączeń bez spadku jakości usług.			
Decyzje architektoniczne	Wrażliwość	Kompromis	Ryzyko	Brak ryzyka
Autoskalowanie frontend	S1	T1	R1	
Load balancer	S2	T2		N1
Analiza	Autoskalowanie frontendu i load balancer eliminują przeciążenia przy dużym ruchu. Ryzyko R1 występuje przy opóźnionej synchronizacji skalowania.			

Scenariusz: A2	Zapewnienie dostępu do aplikacji przez co najmniej 99% czasu			
Atrybuty	Dostępność			
Środowisko	Normalny tryb pracy			
Bodziec	Awaria pojedynczej instancji backendu lub frontendu			
Odpowiedź	Ruch użytkowników zostaje automatycznie przekierowany do pozostałych instancji aplikacji, zapewniając ciągłość działania systemu.			
Decyzje architektoniczne	Wrażliwość	Kompromis	Ryzyko	Brak ryzyka
Load Balancer	S3	T3	R2	N2
Wieloinstancyjne wdrożenie backendu i frontendu	S4	T4	R3	N3
Analiza	Wieloinstancyjne wdrożenie wraz z load balancerem eliminuje pojedynczy punkt awarii, zwiększając dostępność systemu. Ryzyka R2 i R3 dotyczą błędnej konfiguracji load balancera lub niewłaściwego wdrożenia instancji, natomiast N2 i N3 zapewniają dostępność na poziomie ≥99%.			

Scenariusz: B1				
Atrybuty				
Środowisko	Normalny tryb pracy			
Bodziec				
Odpowiedź				
Decyzje architektoniczne	Wrażliwość	Kompromis	Ryzyko	Brak ryzyka
Analiza				

4) Punkty wrażliwości i kompromisy

S1: Autoskalowanie frontendowe jest wrażliwe na opóźnioną reakcję przy gwałtownym wzroście ruchu, co może skutkować chwilowym przeciążeniem serwerów i spadkiem wydajności dla użytkowników końcowych.

S2: Load balancer wymaga prawidłowej konfiguracji rozdzielania ruchu; niewłaściwe ustawienia mogą prowadzić do nierównomiernego obciążenia instancji i częściowych przerw w dostępności usług.

S3: Load balancer jest wrażliwy na awarie lub nieprawidłowe health checki, co może skutkować kierowaniem ruchu do niedostępnych instancji lub chwilową utratą możliwości obsługi żądań użytkowników.

S4: Wieloinstancyjne wdrożenie backendu i frontendu wymaga spójnego zarządzania stanem aplikacji; brak synchronizacji lub błędy w konfiguracji mogą prowadzić do niespójności danych lub przerw w działaniu funkcji systemu.

T1: Zapewnia obsługę dużej liczby jednoczesnych połączeń kosztem większej złożoności infrastruktury, wymagając monitorowania zasobów i dynamicznej konfiguracji autoskalowania w odpowiedzi na zmieniające się obciążenie systemu.

T2: Poprawia dostępność systemu, ale wprowadza dodatkową warstwę pośrednią w architekturze, która wymaga utrzymania i monitoringu, a także może wprowadzać niewielki narzut czasowy w przetwarzaniu żądań.

T3: Umożliwia ciągłość działania systemu przy awarii pojedynczej instancji kosztem monitoringu i utrzymania, ponieważ wszystkie instancje muszą być stale nadzorowane pod kątem zdrowia i wydajności.

T4: Poprawia dostępność i skalowalność systemu kosztem złożoności wdrożenia i synchronizacji instancji, wymagając dodatkowych mechanizmów koordynacji stanu aplikacji oraz procedur zapewniających spójność danych między instancjami.

5) Ryzyka i nie-ryzyka

R1: Opóźnione skalowanie frontendowych instancji może prowadzić do chwilowego spadku wydajności przy gwałtownym wzroście ruchu użytkowników, skutkując wydłużonym czasem odpowiedzi lub krótkotrwałym przeciążeniem serwerów.

R2: Nieprawidłowa konfiguracja load balancera może powodować nierównomierne rozdzielenie ruchu między instancje, prowadząc do częściowych przerw w dostępności usług lub nieefektywnego wykorzystania zasobów systemu.

R3: Błędne wdrożenie wielu instancji backendu lub frontendu może skutkować utratą ciągłości działania, np. przez niespójność danych między instancjami lub czasową niedostępność wybranych funkcjonalności aplikacji.

N1: Prawidłowo skonfigurowane autoskalowanie frontendu umożliwia obsługę co najmniej 1 000 równoczesnych użytkowników bez spadku wydajności, zapewniając płynną pracę systemu nawet przy dużym obciążeniu.

N2: Load balancer poprawnie rozdzielający ruch zapewnia wysoką dostępność systemu, umożliwiając działanie aplikacji nawet w przypadku awarii pojedynczej instancji, minimalizując przestoje dla użytkowników.

N3: Wieloinstancyjne wdrożenie backendu i frontendu, w połączeniu z odpowiednią synchronizacją stanu, pozwala utrzymać dostępność systemu na poziomie ≥99% czasu, zapewniając ciągłość kluczowych usług platformy.

6) Wnioski