Rozwiązania Webowe

Aplikacje webowe

Dr inż. Arkadiusz Rzucidło, prof. PRz

Aplikacje webowe – fundament współczesnych systemów. Ewolucja aplikacji

- aplikacje desktopowe (instalowane lokalnie)
- model klient–serwer
- aplikacje webowe (przeglądarka)
- chmura (cloud computing)
- model SaaS

Aplikacje webowe – fundament współczesnych systemów. Ewolucja aplikacji

- Na początku aplikacje działały lokalnie - instalowane na jednym komputerze.
- Problem? Brak synchronizacji, trudne aktualizacje.
- Potem pojawił się model klient-serwer - część logiki przeniesiono na serwer.
- Dziś dominują aplikacje webowe:
  - nie instalujesz nic
  - wszystko działa w przeglądarce
- A dalej mamy cloud i SaaS - czyli aplikacje dostępne jako usługa (np. Google Docs).
  - To bezpośrednie rozwinięcie modelu klient–serwer, który poznaliście przy sieciach.

Aplikacje webowe – fundament współczesnych systemów. Ewolucja aplikacji

- **SaaS (Software as a Service)** to model dostarczania oprogramowania w chmurze, w którym aplikacje są **hostowane przez dostawcę** i udostępniane użytkownikom przez Internet w formie subskrypcji. Użytkownik korzysta z aplikacji przez przeglądarkę lub aplikację mobilną, bez konieczności instalacji lokalnej czy zarządzania infrastrukturą.
- **Kluczowe cechy działania** obejmują:
  - **Architekturę wielodostępową** – jedna instancja aplikacji obsługuje wielu klientów, z odseparowanymi danymi.
  - **Automatyczne aktualizacje** – dostawca wdraża poprawki i nowe funkcje bez ingerencji użytkownika.
  - **Model subskrypcyjny** – opłaty miesięczne lub roczne, przewidywalne koszty operacyjne.
  - **Integracje** – API umożliwiające łączenie z innymi systemami i automatyzację procesów.

Aplikacje webowe – fundament współczesnych systemów. Ewolucja aplikacji

- **Korzyści biznesowe**:
  - **Niższe koszty początkowe** – brak inwestycji w sprzęt i licencje lokalne.
  - **Skalowalność na żądanie** – szybkie dostosowanie mocy i funkcji do potrzeb.
  - **Szybkie wdrożenia** – uruchomienie w godziny zamiast miesięcy.
  - **Dostępność z dowolnego miejsca** – praca zdalna i mobilna bez ograniczeń.
  - **Stały dostęp do innowacji** – wbudowane technologie jak AI, uczenie maszynowe, IoT czy blockchain.
- **Potencjalne wyzwania**:
  - **Bezpieczeństwo danych** – konieczność weryfikacji standardów dostawcy.
  - **Zależność od dostawcy** – trudności w migracji do innego rozwiązania.
  - **Wydajność** – uzależnienie od jakości połączenia internetowego.
  - **Zgodność z regulacjami** – spełnienie wymogów prawnych i branżowych.

Aplikacje webowe – fundament współczesnych systemów. Ewolucja aplikacji

- **Przykłady zastosowań**:
  - CRM (np. Salesforce)
  - Pakiety biurowe online
  - Systemy księgowe i ERP
  - Platformy e-commerce
  - Narzędzia do zarządzania projektami i komunikacji
- **Przyszłość SaaS** to większa personalizacja, integracja AI, rozwój rozwiązań branżowych oraz środowisk hybrydowych i wielochmurowych, co pozwoli firmom szybciej reagować na zmiany rynkowe i zwiększać innowacyjność.

Aplikacje webowe – fundament współczesnych systemów. Ewolucja aplikacji

Dlaczego model SaaS jest bardziej efektywny niż klasyczne aplikacje desktopowe?

A. Bo działa szybciej lokalnie i wymaga dostępu do Sieci tylko sporadycznie
B. Bo nie wymaga sieci w ogóle, w tym jego największa zaleta
C. Bo aktualizacje i dostęp są centralnie zarządzane dla wszystkich użytkowników
D. Bo nie korzysta z baz danych a jeśli już to sporadycznie

Ograniczenia aplikacji desktopowych

- instalacja na każdym urządzeniu
- brak centralnej aktualizacji
- zależność od systemu operacyjnego
- trudna współpraca wielu użytkowników
- brak mobilności

Ograniczenia aplikacji desktopowych

- Wyobraź sobie firmę z 100 komputerami:
  - aktualizacja = 100 instalacji
- Problemy:
  - wersje się różnią
  - dane są lokalne
  - brak współpracy
- Dlatego desktop przegrał z „webem” w większości zastosowań.

Ograniczenia aplikacji desktopowych

Który problem aplikacji desktopowych najbardziej utrudnia zarządzanie w dużej organizacji?

A. Wysokie zużycie pamięci
B. Brak centralnego zarządzania i aktualizacji
C. Brak interfejsu graficznego
D. Brak połączenia z Internetem

Dlaczego web wygrał

- dostęp z dowolnego miejsca
- brak instalacji
- jedna wersja aplikacji
- łatwe aktualizacje
- współpraca w czasie rzeczywistym

Dlaczego web wygrał

- Największa zmiana:
  - aplikacja przestała być „na komputerze”
  - jest „w sieci”

Przykład: Google Docs

  - kilka osób edytuje jednocześnie
  - wszystko zapisuje się w bazie danych
  - komunikacja odbywa się przez sieć

- Tu łączą się: sieci + bazy danych + aplikacja

Dlaczego web wygrał

Która cecha aplikacji webowych najbardziej umożliwia pracę zespołową?

A. Wspólna, centralna baza danych dostępna przez sieć
B. Instalacja lokalna w przeglądarce
C. Szybkość procesora lokalnego komputera i system operacyjny
D. Offline mode

Dostępność (availability)

- dostęp 24/7
- dostęp z różnych urządzeń
- brak zależności od lokalizacji
- przeglądarka jako interfejs
- Internet jako medium

Dostępność (availability)

- Aplikacja webowa działa wszędzie:
  - laptop
  - Telefon
  - tablet
- Warunek: Internet + przeglądarka
- To bezpośrednie wykorzystanie sieci, które poznaliście na wykładzie z Sieci:
  - klient (przeglądarka) → serwer

Dostępność (availability)

Który element jest kluczowy dla dostępności aplikacji webowej?

A. System operacyjny użytkownika
B. Lokalna instalacja programu
C. Typ procesora
D. Dostęp do sieci i serwera aplikacji

Skalowalność

- obsługa wielu użytkowników
- dynamiczne zwiększanie zasobów
- chmura (cloud)
- load balancing
- mikroserwisy (intro)

Skalowalność

- Skalowalność = aplikacja rośnie razem z użytkownikami

Przykład: sklep internetowy

  - 10 użytkowników → działa
  - 10 000 użytkowników → nadal działa

- Jak?
  - więcej serwerów
  - rozłożenie ruchu

- To bez sieci i infrastruktury byłoby niemożliwe.

Skalowalność

Co umożliwia obsługę dużej liczby użytkowników w aplikacji webowej?

A. Lokalna instalacja dobrej przeglądarki
B. Skalowanie serwerów i rozkład ruchu (load balancing)
C. Szybszy komputer użytkownika
D. Szybszy DNS i infrastruktura sieciowa

TECHNOLOGIE INTERNETOWE
Niezależność sprzętowa i systemowa

- działa w przeglądarce
- brak zależności od systemu operacyjnego
- ten sam kod dla wszystkich użytkowników
- urządzenia: PC, tablet, telefon
- brak instalacji lokalnej

TECHNOLOGIE INTERNETOWE
Niezależność sprzętowa i systemowa

- Aplikacja webowa działa w przeglądarce
  - przeglądarka jest wszędzie.
- To oznacza:
  - nie musisz pisać osobnej wersji dla Windows, Mac, Linux
  - nie interesuje Cię sprzęt użytkownika

Przykład:

- Ten sam system bankowy działa na telefonie i laptopie.
- To ogromna przewaga nad aplikacjami desktopowymi.

TECHNOLOGIE INTERNETOWE
Niezależność sprzętowa i systemowa

- Dlaczego aplikacje webowe są niezależne od systemu operacyjnego?

A. Bo działają szybciej
B. Bo używają tylko zatwierdzonych węzłów do komunikacji
C. Bo działają w przeglądarce jako wspólnym środowisku
D. Bo nie korzystają z sieci w ogóle

Architektura aplikacji webowej

- frontend (interfejs użytkownika)
- backend (logika biznesowa)
- baza danych (przechowywanie danych)
- serwer aplikacji
- komunikacja przez sieć

Architektura aplikacji webowej

- Każda aplikacja webowa ma trzy główne części:
  - frontend – to co widzi użytkownik (HTML, UI)
  - backend – logika (co się dzieje po kliknięciu)
  - baza danych – gdzie są dane
- To dokładnie łączy:
  - sieci (komunikacja)
  - bazy danych (przechowywanie)

Architektura aplikacji webowej

Który element odpowiada za przechowywanie danych w aplikacji webowej?

A. Frontend
B. Backend
C. Baza danych
D. Przeglądarka

Frontend

- interfejs użytkownika
- HTML, CSS, JavaScript (pojęciowo)
- działa w przeglądarce
- reaguje na działania użytkownika
- komunikuje się z backendem

Frontend

- Frontend to wszystko, co widzi użytkownik.
  - klikasz, określasz parametry we frontendzie
  - frontend wysyła zapytanie
  - backend odpowiada
  - frontend wyświetla wynik.
- To jak „pilot”, który steruje systemem.
- Interakcja wymaga różnych „trigger’ów”

Frontend

Co jest główną rolą frontendu?

A. Przechowywanie danych użytkownika na komputerze lokalnym
B. Obsługa interakcji użytkownika i komunikacja z backendem
C. Zarządzanie siecią w kwestii wysyłania danych
D. Routing pakietów do serwera w dbałości o poprawność i ich kolejność

Backend

- logika aplikacji
- przetwarzanie danych
- komunikacja z bazą danych
- obsługa zapytań
- kontrola dostępu

Backend

- Backend to „mózg” aplikacji webowej.

Przykład:

  - logowanie → wypełnienie formularza i wysłanie danych
  - backend sprawdza i waliduje dane
  - hasło bazy danych jest weryfikowane
  - backend zwraca wynik i autoryzuje/nie autoryzuje użytkownika
- Tu dzieje się cała logika biznesowa.

Backend

Który element odpowiada za sprawdzenie poprawności loginu i hasła?

A. Frontend
B. DNS z bazą haseł
C. Backend aplikacji
D. Router

Baza danych w aplikacji webowej

- przechowuje dane użytkowników
- przechowuje dane aplikacji
- zapewnia trwałość danych
- umożliwia zapytania (np. SQL)
- współpracuje z backendem

Baza danych w aplikacji webowej

- To dokładnie to, co już znacie z baz danych.

Przykład:

- użytkownik się loguje
- backend wysyła zapytanie SQL
- baza danych odpowiada

- Bez bazy danych większość aplikacji webowych nie istnieje.

Baza danych w aplikacji webowej

Dlaczego baza danych jest kluczowa w aplikacjach webowych?

A. Bo przechowuje i udostępnia dane aplikacji
B. Bo przyspiesza pracę aplikacji webowej
C. Bo obsługuje bezpośrednio zgłoszenia użytkownika
D. Bo zarządza scenariuszem aplikacyjnym

Serwer aplikacji

- uruchamia backend
- obsługuje zapytania
- komunikuje się z klientem
- przetwarza logikę
- zarządza sesją

Serwer aplikacji

- Serwer aplikacji to miejsce, gdzie działa backend.
- To tutaj trafia każde zapytanie z przeglądarki.
- Scenariusz:
  - przeglądarka → serwer → baza → odpowiedź
- Serwer odpowiedzialny jest za przetwarzanie scenariusza
- Miejsce generowania odpowiedzi i przygotowania gotowego kodu dla przeglądarki do wyświetlenia

Serwer aplikacji

Gdzie wykonywana jest logika aplikacji webowej?

A. W przeglądarce
B. W systemie operacyjnym użytkownika
C. Na serwerze aplikacji (backend)
D. W bazie danych aplikacji webowej

Komunikacja klient–serwer

- To dokładnie model, który już znacie z sieci.
  - klient wysyła zapytanie
  - serwer odpowiada
- Każde kliknięcie w aplikacji to komunikacja sieciowa.
- Komunikacja realizowana jest ściśle w oparciu o scenariusz aplikacji
  - przetwarzanie danych
  - Bezpieczeństwo
  - gromadzenie danych
  - Przygotowanie odpowiedzi

Komunikacja klient–serwer

Który element inicjuje komunikację w aplikacji webowej?

A. Serwer
B. Baza danych
C. Karta sieciowa (komputer klienta)
D. Klient (przeglądarka)

HTTP (intuicyjnie)

- protokół komunikacji webowej
- request → response
- bezstanowość (stateless)
- działa na portach (80/443)
- fundament webu

HTTP (intuicyjnie)

- HTTP to „język”, którym rozmawiają klient i serwer.

Przykład:

  - GET, POST /strona
  - serwer odpowiada HTML
- Bez HTTP nie ma webu.
  - Bez HTML nie ma serwisu (struktury)
  - Bez HTML nie ma odpowiedzi
  - Bez HTML nie ma zapytania

HTTP (intuicyjnie)

Dlaczego HTTP jest kluczowy dla aplikacji webowych?

A. Bo przechowuje dane klienta w systemie
B. Bo umożliwia komunikację między klientem a serwerem
C. Bo zarządza identyfikacją klienta w aplikacji
D. Bo zastępuje pozostałe technologie internetowe

API (Application Programming Interface)

- komunikacja między systemami
- backend ↔ frontend
- backend ↔ backend
- przesył danych (JSON – intuicyjnie)
- integracja systemów

API (Application Programming Interface)

- API to sposób komunikacji między systemami.

Przykład:

- aplikacja mobilna korzysta z tego samego backendu co strona
- API to fundament nowoczesnych systemów.
  - jest zbiorem reguł i protokołów, które definiują sposób komunikacji między różnymi aplikacjami lub systemami
  - nie jest zmienny w swoim podstawowym działaniu,
  - dla tabletów i komputerów powinny działać na tych samych zasadach i protokołach.
  - jest kluczowym elementem integracji systemów i aplikacji

API (Application Programming Interface)

Do czego służy API w aplikacji webowej?

A. Do wyświetlania strony internetowej i aplikacji webowej
B. Do przechowywania danych klienckich i systemowych
C. Do zarządzania systemami sieciowymi
D. Do komunikacji między różnymi komponentami systemu

Przepływ danych (end-to-end)

1. użytkownik wysyła żądanie
2. frontend przekazuje do backendu
3. backend komunikuje się z bazą
4. baza zwraca dane
5. odpowiedź wraca do użytkownika

Przepływ danych (end-to-end)

- To najważniejszy przepływ:
  - klik → request → backend → DB → response → ekran
- To połączenie:
  - sieci
  - baz danych
  - aplikacji
- Scenariusz jest uniwersalny w aplikacji webowej

Przepływ danych (end-to-end)

Który etap odpowiada za pobranie danych z bazy?

A. Frontend
B. Formularz w interfejsie webowym
C. Backend aplikacji
D. Parser

Typy aplikacji webowych

- aplikacje statyczne
- aplikacje dynamiczne
- SPA (Single Page Application)
- MPA (Multi Page Application)
- aplikacje mobilne (webowe)

Typy aplikacji webowych

- Nie każda strona to „pełna aplikacja”.
  - statyczna → tylko treść (np. wizytówka)
  - dynamiczna → dane z bazy
- SPA (np. Gmail):
  - działa jak aplikacja, bez przeładowania strony
- To ważne, bo wpływa na sposób działania i wydajność.

Typy aplikacji webowych

- Nie każda strona to „pełna aplikacja”.
  - statyczna → tylko treść (np. wizytówka)
  - dynamiczna → dane z bazy
- SPA (np. Gmail):
  - działa jak aplikacja, bez przeładowania strony (AJAX)
- To ważne, bo wpływa na sposób działania i wydajność.

Typy aplikacji webowych

Która cecha najlepiej opisuje aplikację typu SPA?

A. **Interfejs działa dynamicznie bez pełnego przeładowania strony**
B. Każda akcja przeładowuje stronę ale nie łączy się z serwerem
C. Nie korzysta z bazy danych zatrzymując dane dla siebie w systemie operacyjnym klienta
D. Działa tylko offline

Globalna dostępność

- dostęp z każdego miejsca
- Internet jako platforma
- brak granic geograficznych
- globalni użytkownicy
- skalowalność systemów

Globalna dostępność

- To jest efekt „globalnej wioski”.
  - aplikacja działa w Polsce
  - korzysta ktoś w USA
- To zmienia:
  - biznes
  - komunikację
  - edukację

Globalna dostępność

Co umożliwia globalne działanie aplikacji webowych?

A. Sieciowy system operacyjny
B. Lokalna instalacja aplikacji webowej
C. Internet jako wspólna infrastruktura komunikacyjna
D. DNS w lokalnym środowisku wpływające na efektywność połączeń

Integracja systemów

- API między systemami
- wymiana danych
- mikroserwisy (intro)
- systemy rozproszone
- współpraca aplikacji

Integracja systemów

- Dziś systemy nie działają samodzielnie.
- sklep → system płatności → bank → system dostawy
- To wszystko łączy się przez API.
- Idea (pomysł) łączy w scenariuszu wiele aktywności (aplikacji)
  - Efektywność
  - Bezpieczeństwo
  - Wspólny schemat

Integracja systemów

Dlaczego API jest kluczowe w nowoczesnych aplikacjach?

A. Przyspiesza odbiór Internetu i wymianę danych
B. Umożliwia komunikację między różnymi systemami
C. Zastępuje bazę danych w kluczowych etapach pracy web-aplikacji
D. Zarządza połączeniami z Siecią globalną

E-commerce (sklepy internetowe)

- sprzedaż online
- koszyk zakupowy
- płatności
- baza produktów
- obsługa użytkowników

E-commerce (sklepy internetowe)

- To klasyczny przykład aplikacji webowej.
  - użytkownik → przegląda produkty
  - backend → pobiera dane z bazy
  - system → obsługuje płatność
- To integracja wszystkiego:
  - sieci
  - bazy danych
  - aplikacja

E-commerce (sklepy internetowe)

Który element jest kluczowy dla działania sklepu internetowego?

A. System operacyjny klienta na urządzeniu klienckim
B. Lokalny dysk użytkownika i umożliwienie zapisu przeglądarce
C. Baza danych przechowująca produkty i zamówienia
D. Trasa pakietów na drodze od klienta do serwera.

E-banking

Który element jest kluczowy dla działania sklepu internetowego?

A. System operacyjny klienta na urządzeniu klienckim
B. Lokalny dysk użytkownika i umożliwienie zapisu przeglądarce
C. Baza danych przechowująca produkty i zamówienia
D. Trasa pakietów na drodze od klienta do serwera.

E-banking

- Bankowość online to jeden z najbardziej wymagających systemów.
  - każda operacja musi być:
    - bezpieczna
    - poprawna
    - szybka
- Temat rozwiniemy w kolejnym wykładzie: bezpieczeństwo + płatności

E-banking

Dlaczego e-banking wymaga szczególnie wysokiego poziomu bezpieczeństwa?

A. Bo działa w środowisku wielodostępnym
B. Bo korzysta z sieci Internet
C. Bo działa na różnych interfejsach klienckich
D. Bo operuje na wrażliwych danych i środkach finansowych

E-government

- usługi publiczne online
- dostęp do dokumentów
- składanie wniosków
- cyfryzacja administracji
- dostępność dla obywateli

E-government

- Administracja publiczna też przeszła do webu.
  - Podatki
  - Dokumenty
  - Usługi
- To zwiększa:
  - Dostępność
  - Efektywność
  - Zasięg
  - Wygodę

E-government

Jaka jest główna zaleta e-government?

A. **Ułatwienie dostępu do usług publicznych dla obywateli**
B. Zmniejszenie liczby komputerów w instytucjach
C. Przejście z formy tradycyjnej (okienka urzędowego) na formę elektroniczną
D. Zmiana systemu operacyjnego

E-learning

- nauka online
- platformy edukacyjne
- dostęp do materiałów
- testy i interakcja
- globalny dostęp

E-learning

- Dokładnie to, co robimy teraz ☺
  - wiedza dostępna online
  - interakcja
  - testy
- To też aplikacja webowa oparta o:
  - bazę danych
  - backend
  - frontend

E-learning

Co umożliwia platformom e-learningowym skalowanie na dużą liczbę studentów?

A. Lokalna instalacja platformy e-learningowej
B. Dostępność w ramach programu studiów
C. Architektura webowa i centralne zarządzanie systemem
D. Konsekwentny update zainstalowanego środowiska

Wydajność aplikacji

- czas odpowiedzi
- liczba użytkowników
- optymalizacja zapytań
- caching
- infrastruktura

Wydajność aplikacji

- Jakość usługi to w większości efektywność narzędzia
- Wydajność to klucz:
  - wolna aplikacja = użytkownik odchodzi
- Problemy mogą wynikać z:
  - sieci
  - backendu
  - bazy danych

Wydajność aplikacji

Który element najczęściej wpływa na spowolnienie aplikacji webowej?

A. Poziom UX środowiska platformy
B. Lokalne zasoby komputera klienckiego
C. Szybkość bazy danych
D. Niewydajne zapytania do bazy danych

Skalowanie aplikacji

- więcej serwerów
- load balancing
- rozdzielenie usług
- chmura
- automatyzacja

Skalowanie aplikacji

- Gdy rośnie liczba użytkowników:
  - dokładamy serwery
  - rozkładamy ruch
- To dokładnie to, co omawialiśmy przy sieciach.
- Load balancing – równomierne obciążenie – kluczowy proces w zarządzaniu strukturą informatyczną
- Load balabcing ma szczególne znaczenie w środowiskach wielodostępnych gdzie wielu użytkowników korzysta z tych samych zasobów
- Skalowalność techniczna jest pochodną idei web-aplikacji
- Projekt musi zakładać skalowalność obciążenia

Skalowanie aplikacji

Co umożliwia równomierne rozłożenie ruchu w aplikacji?

A. Load balancing – zarządzając zapytaniami do konkretnych zasobów
B. DNS przez dostarczanie informacji o dostępnych zasobach
C. DHCP – umiejętnie gospodarując dostępem do Internetu
D. Baza danych – kolejkując i prioretyzując zapytania

Problemy w aplikacjach webowych

- brak Internetu
- problem DNS
- problem backendu
- problem bazy danych
- problem użytkownika

Problemy w aplikacjach webowych

- Diagnostyka = dokładnie to, co znacie z sieci.
  - gdzie jest problem?

- sieć
- frontend
- backend
- baza

Problemy w aplikacjach webowych

Jeśli aplikacja nie działa dla wszystkich użytkowników, problem najczęściej leży:

A. W komputerze użytkownika
B. W DNS lokalnym
C. W backendzie lub infrastrukturze serwera
D. W frontendzie

Rola sieci w środowisku webowym

- przesył danych
- komunikacja klient–serwer
- protokoły (HTTP)
- routing
- dostępność

Rola sieci w środowisku webowym

- Bez sieci nie ma środowiska webowego.
  - wszystko to komunikacja
  - każde kliknięcie = pakiety
- Jedyna wada tego rozwiązania to konieczny dostęp do Sieci

Rola sieci w środowisku webowym

Dlaczego sieci komputerowe są fundamentem aplikacji webowych?

A. Bo przechowują dane i administrują nimi
B. Bo zarządzają bazą i realizacją żądań klientów w web-aplikacji
C. Bo w sytuacjach krytyczny zastępują backend
D. Bo umożliwiają komunikację między klientem a serwerem

Rola baz danych

- przechowywanie danych
- zapytania
- trwałość
- spójność
- integracja z backendem

Rola baz danych

- To dokładnie to, co znacie z poprzednich wykładów.
  - bez bazy → brak aplikacji
  - Brak funkcjonalności
  - Rozwiązania webowe są silnie uzależnione od zasobów baz danych
  - Scenariusz działania dotyczy w głównej mierze realizacji zapytań w oparciu o parametry pozyskane z frontendu

Rola baz danych

Dlaczego baza danych jest niezbędna w większości aplikacji webowych?

A. Przyspiesza komunikację sieciową frontendu z backendem
B. Zarządza zasobami DNS przez co wpływa na efektywność połączeń
C. Zapewnia przechowywanie i dostęp do danych w web-aplikacji
D. Pełni rolę zapasowego systemu dla frontendu

Synergia wszystkich omawianych składników

- sieci + bazy + aplikacje
- klient–serwer
- przepływ danych
- integracja systemów
- architektura webowa

Synergia wszystkich omawianych składników

Który element NIE jest częścią architektury aplikacji webowej?

A. Backend
B. Baza danych
C. Frontend
D. Przeglądarka użytkownika

Dlaczego web dominuje?

- dostępność
- skalowalność
- globalność
- łatwe utrzymanie
- integracja

Dlaczego web dominuje?

- Rozwiązanie web wygrało, bo jest:
  - prostsze dla użytkownika
  - łatwiejsze dla organizacji
  - skalowalne globalnie
  - ….

Dlaczego web dominuje?

Który czynnik najbardziej przyczynił się do dominacji aplikacji webowych?

A. **Dostępność i łatwość skalowania globalnego**
B. Brak konieczności posiadania stałego łącza do Internetu
C. Korzystanie z sieciowego systemu operacyjnego przez klientów
D. Lokalna instalacja oprogramowania i jej funkcjonalność

Podsumowanie

- czym są aplikacje webowe
- jak działają
- dlaczego są ważne
- jak łączą technologie
- przygotowanie do bezpieczeństwa

Podsumowanie

- Po tym wykładzie powinno się:
  - rozumieć architekturę
  - widzieć powiązania
  - rozumieć przepływ danych

- następny krok: **bezpieczeństwo** !

Podsumowanie

Która umiejętność najlepiej świadczy o zrozumieniu aplikacji webowych?

A. Zapamiętanie definicji poszczególnych składników
B. Zrozumienie przepływu danych i współpracy komponentów
C. Znajomość nazw technologii
D. Znajomość portów działania składników systemu