

Produkcja



Dzisiaj

1. Continuous Integration & Deployment
2. Observability
3. Performance testing and monitoring

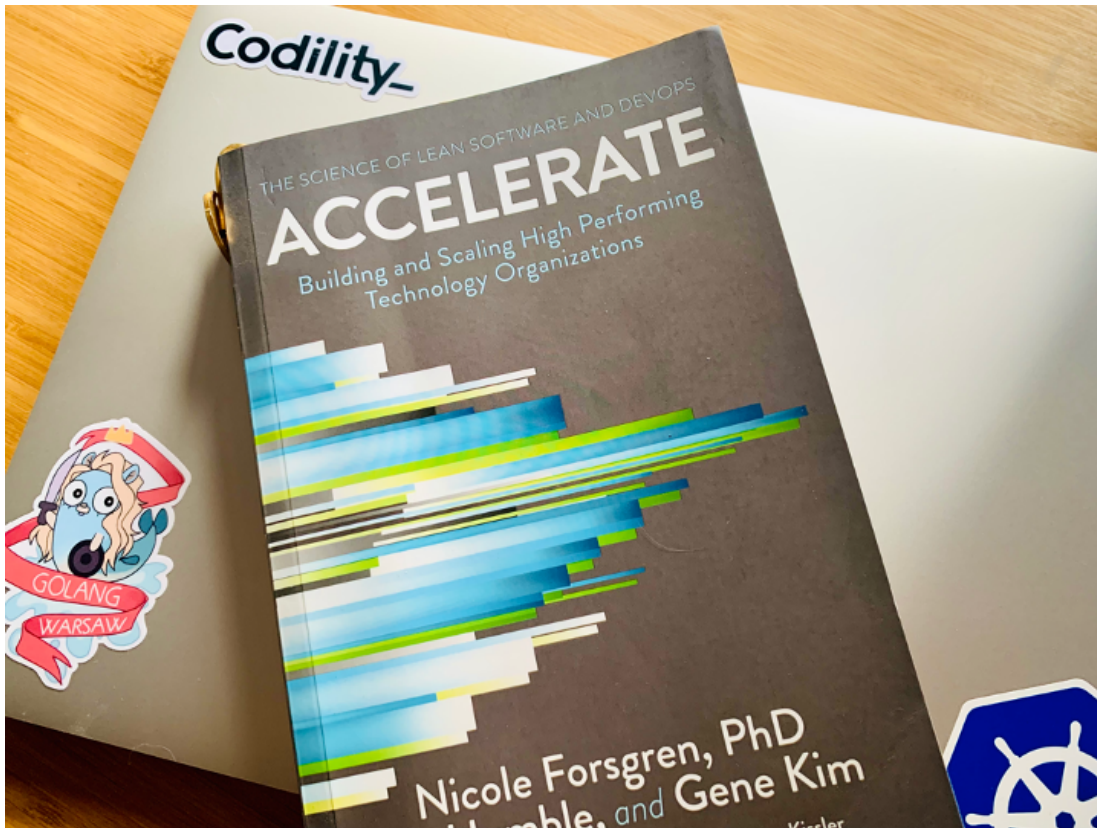
CI/CD

- Continuous Integration
- Continuous Deployment

Continuous Deployment

Dlaczego?

High Performance Teams



DORA metrics:

- deployment freq
- lead time
- failure rate
- MTTR

Continuous ____

- [Przykład 1](#)
- [Przykład 2](#)

Continuous Deployment

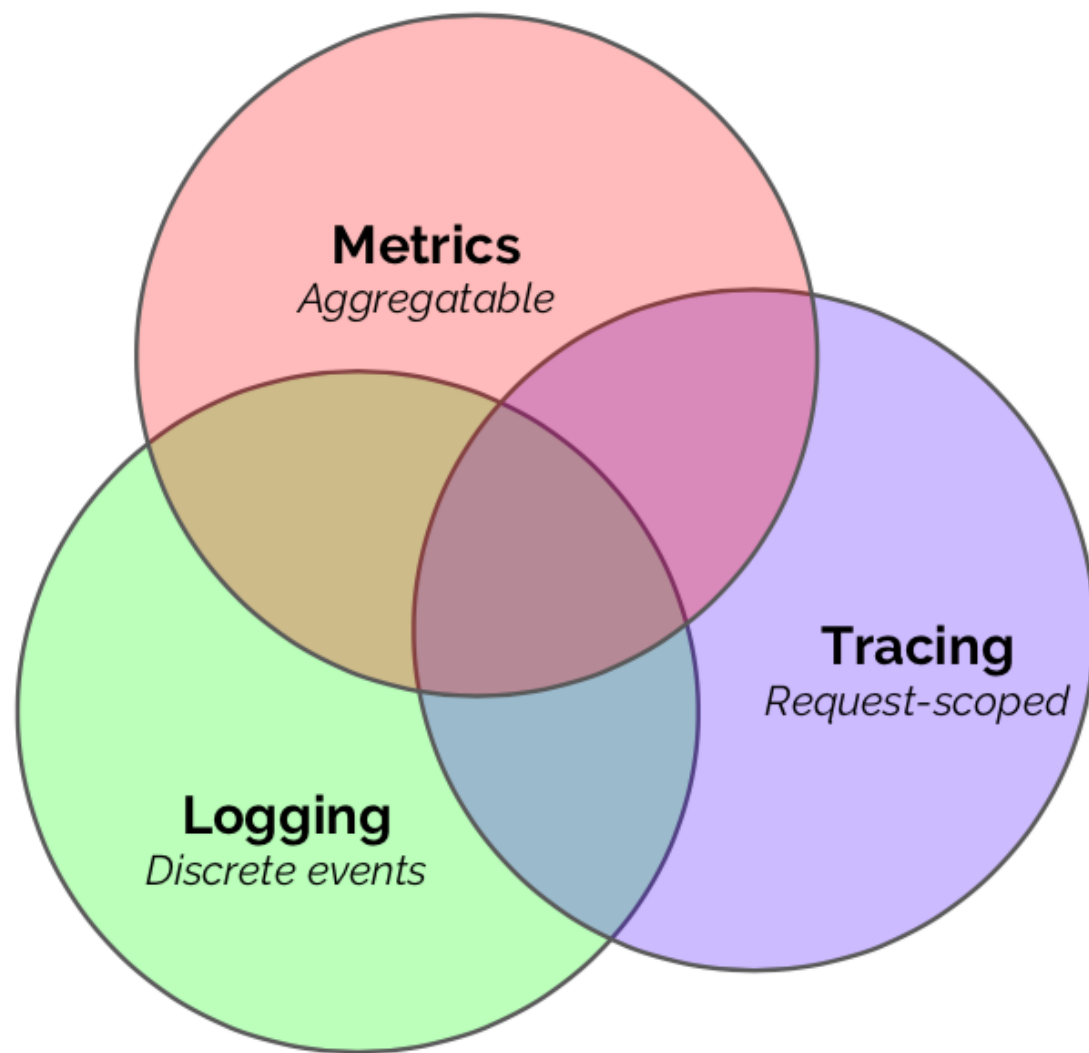
1. W każdym projekcie, niezawodny Continuous Deployment powinno być priorytetem!
2. Wszystko idzie przez repozytorium Gita.

Observability

Observability

(Classic) types:

- logging
- monitoring
- tracing



[źródło](#)

Observability

	Metrics	Logging	Tracing
CapEx	Medium	Low	High
OpEx	Low	High	Medium
Reaction	High	Medium	Low
Investigation	Low	Medium	High

[źródło](#)

Monitoring: alerts

Alerts on metrics measuring user's experience (see [docs](#))



Observability

Więcej na:

- [Budowa i Administracja aplikacji w chmurze;](#)
- [Instrumentyzacja aplikacji i monitoring z Prometheusem.](#)

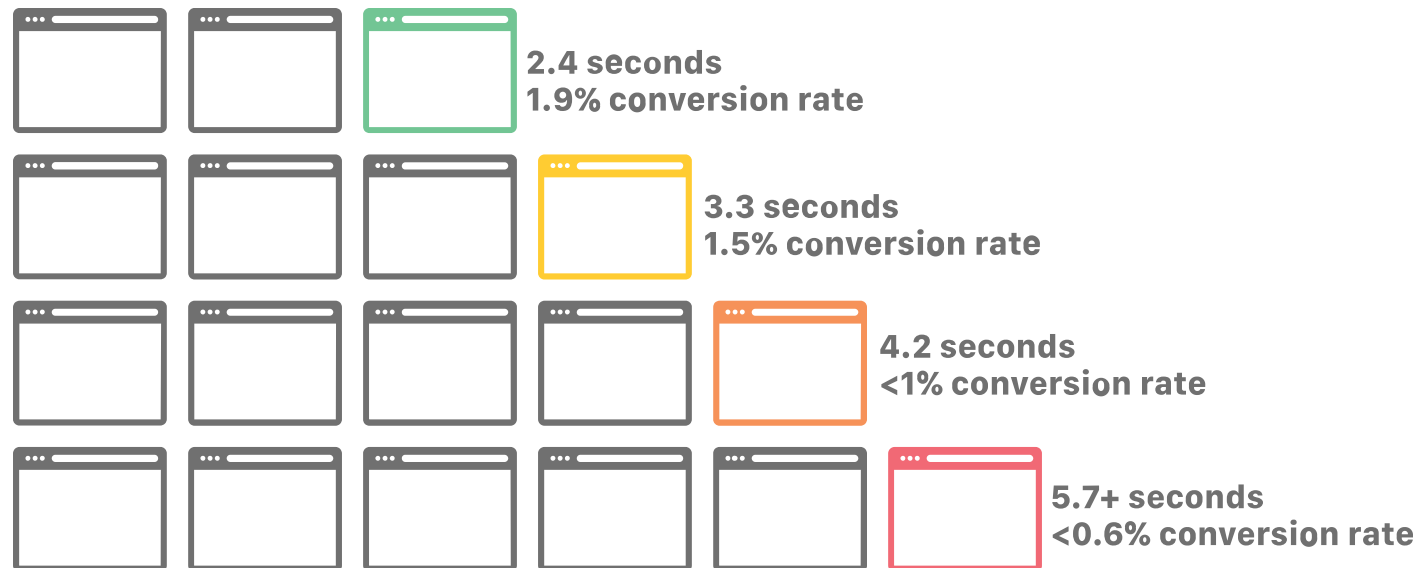
Rekomendacja

- Start with monitoring;
- Metrics? Start with Up/Down, than RED and USE;
- Drop-in solution Prometheus;
- Retrospektywy

Performance monitoring & testing

Dlaczego wydajność jest ważna

Ecommerce ([cloudflare](https://www.cloudflare.com/)):



Dlaczego wydajność jest ważna

Bounce rate: ([cloudflare](#)):

BBC discovered that they lost 10% of their total users for every additional second it took for their pages to load.

Dlaczego wydajność jest istotna

3 graniczne czasy dla reakcja strony ([Nilsen](#)):

- 0.1 sekundy - wrażenie, że strona natychmiast się załadowała
- 1.0 sekunda
- 10 sekund

Dlaczego wydajność jest istotna

3 graniczne czasy dla reakcja strony ([Nilsen](#)):

- 0.1 sekundy
- 1.0 sekunda - flow utrzymywane
- 10 sekund

Dlaczego wydajność jest istotna

3 graniczne czasy dla reakcja strony ([Nilsen](#)):

- 0.1 sekundy
- 1.0 sekunda
- 10 sekund - ile użytkownik maksymalnie może skupić się na interakcji ze stroną

Dlaczego wydajność jest istotna

- Mniej niż 200 ms (patrz [mental chronometry](#)),
- Czym bliżej 100 ms, tym lepiej.

Co to są testy wydajnościowe

- intuicja?
- ...
- ...
- ...

Co to są testy wydajnościowe

- Developer Tools w przeglądarce

Scenariusze

- Czarny piątek
- Nowa architektura
- Wolno działająca aplikacje

Scenariusze

- Zgłaszane bugi od klientów
- Alarmy z monitoringu dotyczące wydajności lub błędów w przypadku większego natężenia ruchu
- Część procesu budowy oprogramowania

Najlepsze praktyki

1. Mierzemy jak wygląda doświadczenie użytkownika naszego systemu;
2. Dane wydajnościowe z produkcji są krytyczne;
3. Czasami trzeba się uciec do [napkin math](#).

Najlepsze praktyki

Nie tylko live traffic/[RUM](#), warto dodać syntetyki, oraz end2end API testy na produkcji.

Najlepsze praktyki

Monitorowanie (oraz [alerty](#)) naszej aplikacji w produkcji oraz błędy również z punktu widzenia użytkownika.

Najlepsze praktyki

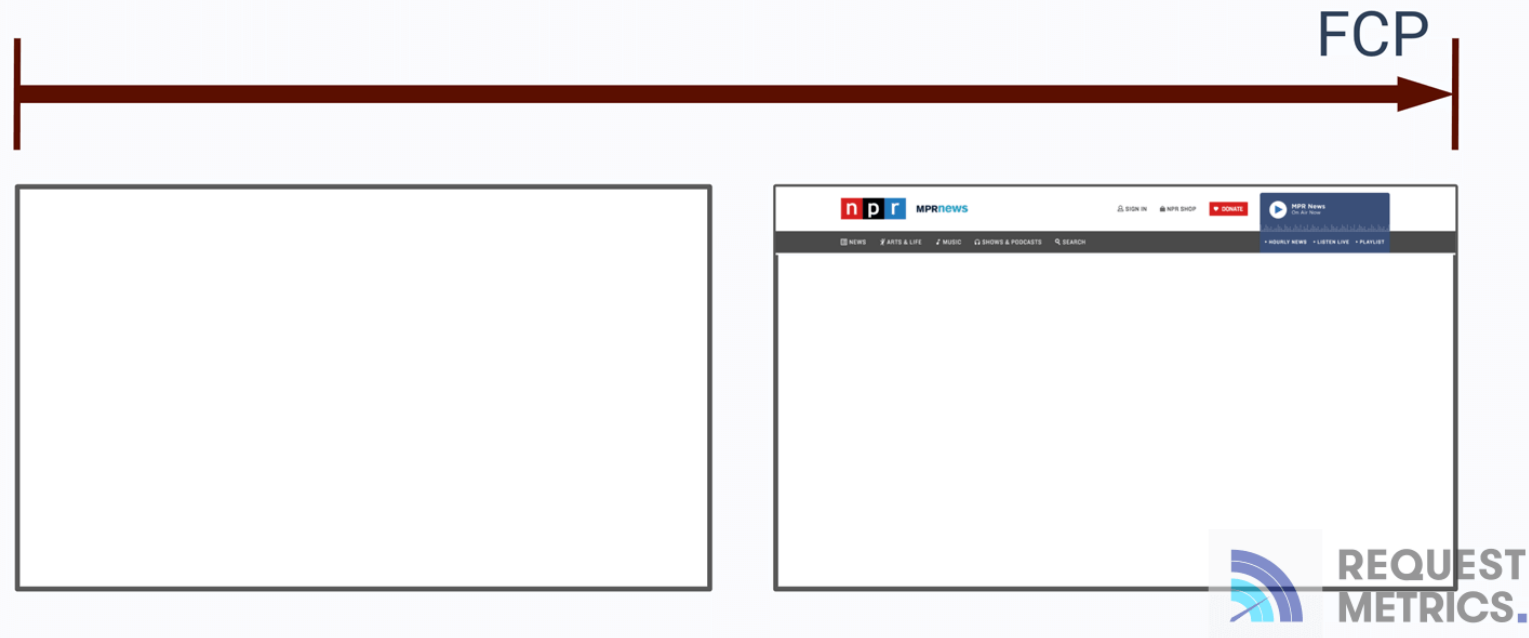
- Performance jest odpowiedzialnością całego zespołu,
- Regular reviews of the production perf data by product/dev team!

Performance

- web performance
- mobile app performance
- backend

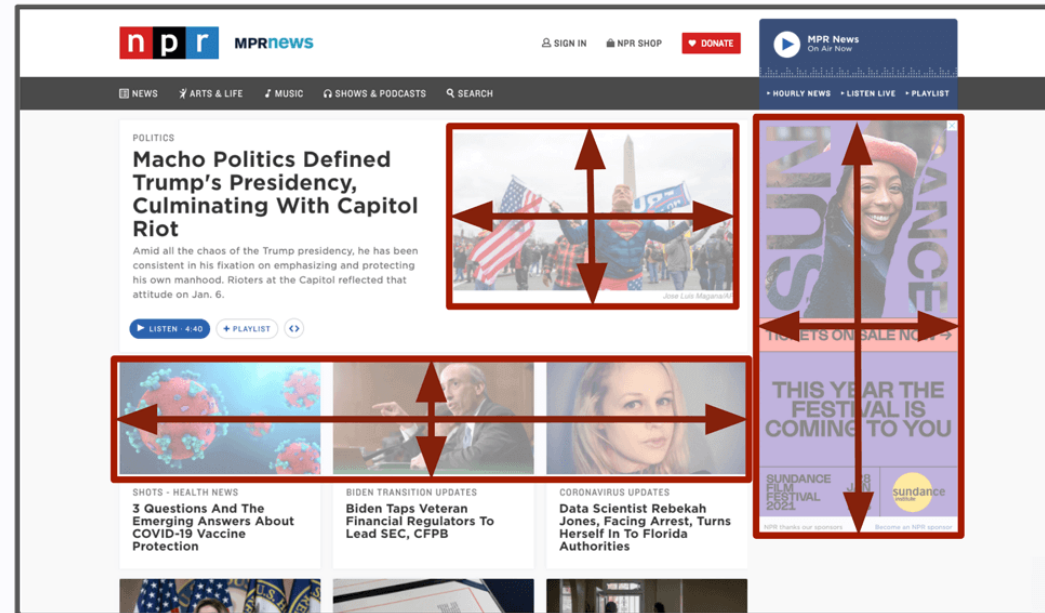
Web vitals

CORE WEB VITALS FIRST CONTENTFUL PAINT (FCP)



Web vitals

CORE WEB VITALS LARGEST CONTENTFUL PAINT (LCP)

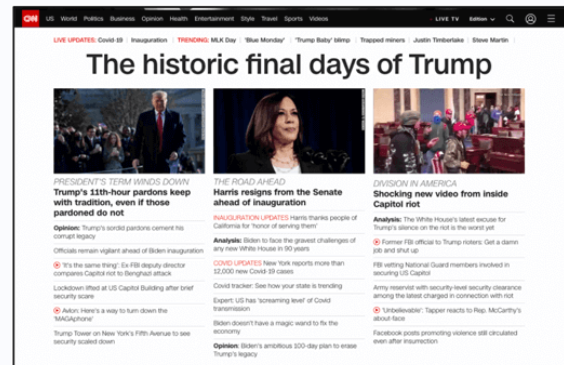


Web vitals

Cumulative Layout Shift: [przykład 1](#) i [przykład 2](#).

Web vitals

CORE WEB VITALS FIRST INPUT DELAY (FID)



BROWSER
BACKGROUND AND
ASYNC WORK

LOOKS READY



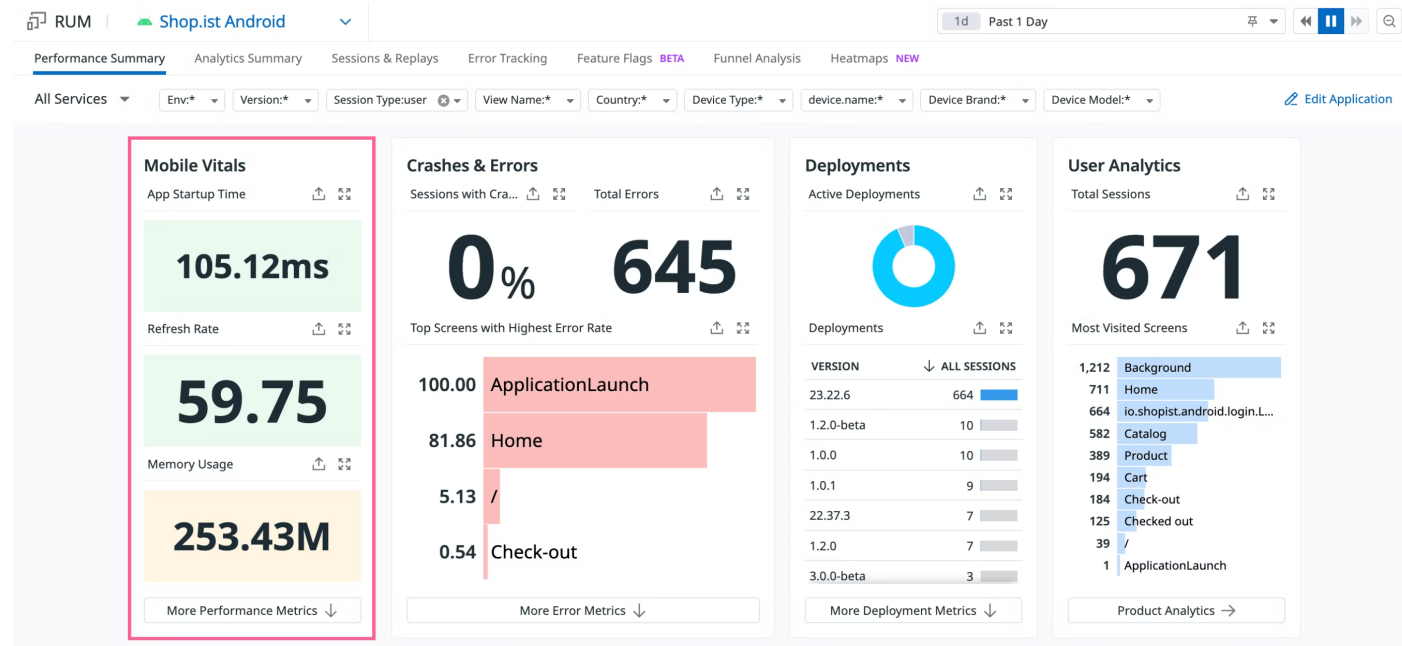
EXECUTE

FID



REQUEST
METRICS.

Mobile App Vitals



[Datadog documentation](#)

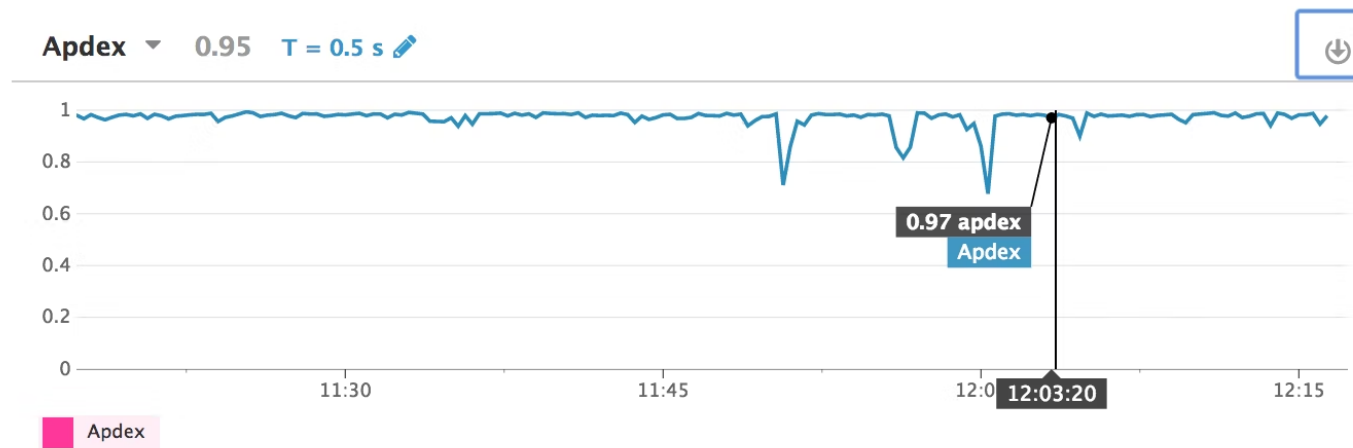
Backend / API

RED / [4 Golden signals](#):

- Rate
- Error
- Duration

Backend / API

Dobrze również znać [Apdex](#) ([datadoc docs](#)):



Backend / queues

USE ([src](#)):

- Utilization
- Saturation
- Errors

Przeprowadzanie testów

Przygotowanie

- Document opisujący test
- Dane z produkcji
- Najbliższe warunkom produkcyjnym
- Narzędzie
- Monitoring

Przygotowanie

Wyznaczony cel:

- system ma wymaganą wydajność,
- jedna z implementacji jest lepsza,
- jaka jest maksymalna wydajność,

Dokument / Design doc

Na przykład:

- ile użytkowników i co będą robić*
- ile sesji równoległe
- ile razy powtórzymy testy
- jakie środowisko, zakres
- jakie metryki będziemy mierzyć*
- zaplanuj i zaprojektuj testy

Dokument

- współdziel dokument z całą zespołu
- przygotuj skrypty dla twojego narzędzia

Dane z produkcji

Jeśli masz dostęp do danych z produkcji, warto wykorzystać je w zaprojektowaniu testów.

Jak szacować

- Estymuj, np., z napkin math
- Zweryfikuj
- Popraw estymate oraz założenia lub elementu twojego testu

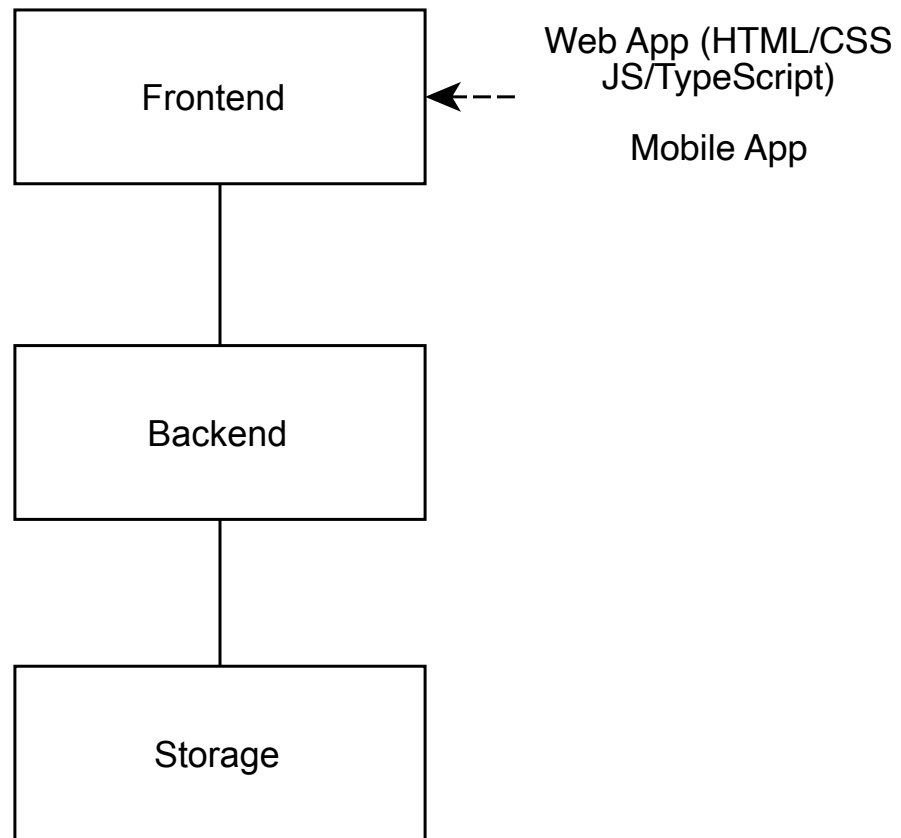
Narzędzia

- [Locust](#)
- [JMeter](#) - najbardziej złożony
- [k6s](#)
- siege - old timer
- iperf3 - sieć komputerowa

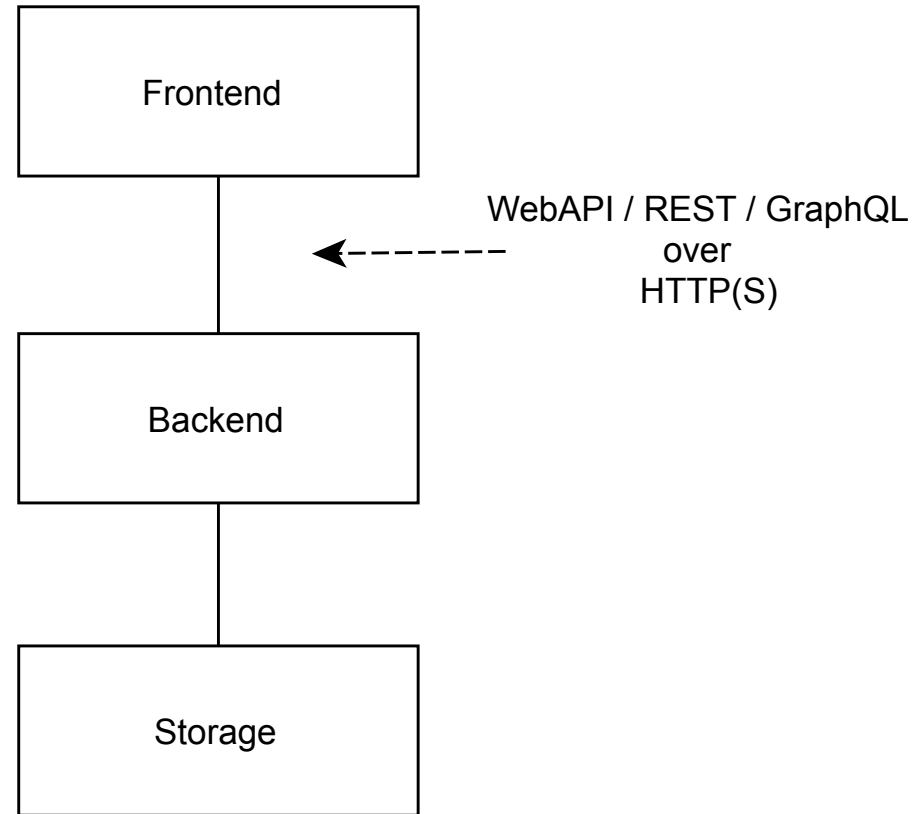
Uruchamianie testów

Let's go deeper

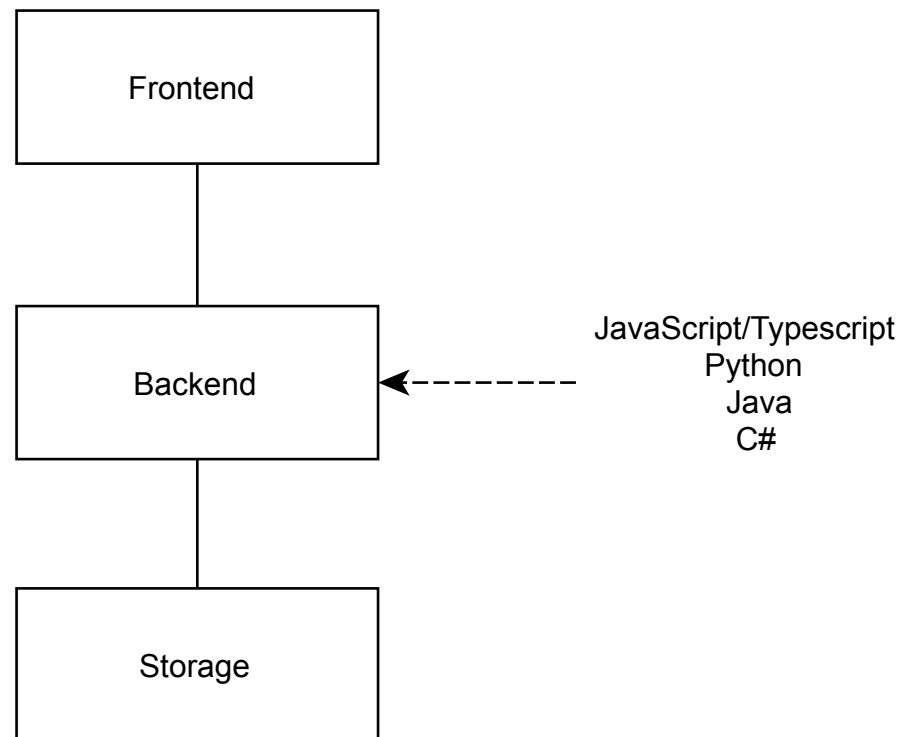
Architektura aplikacji 1 (uproszczona)



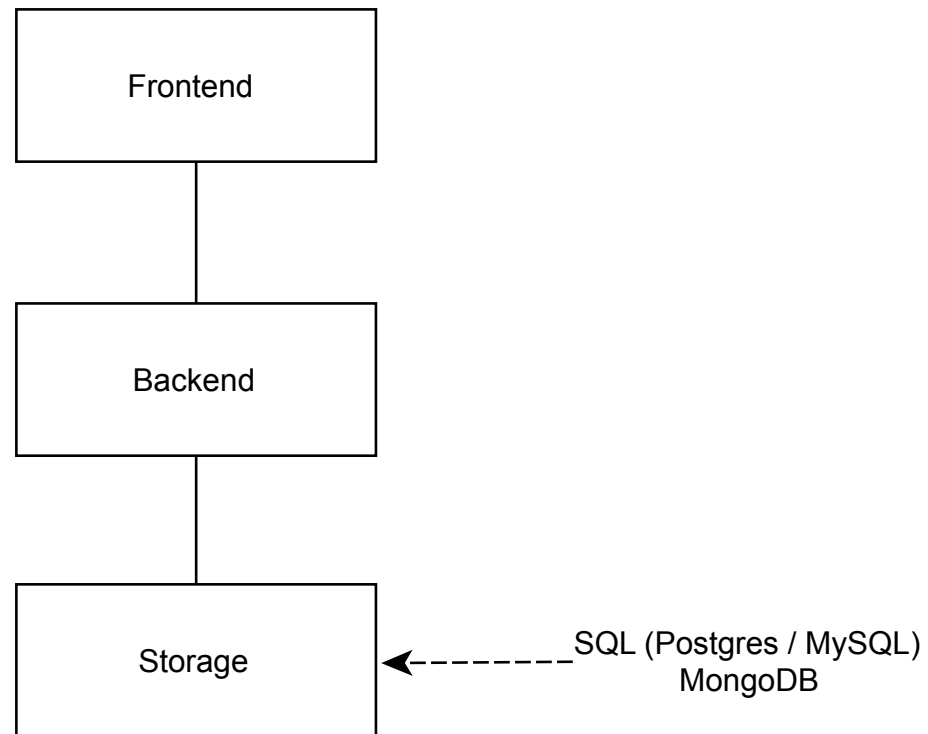
Architektura aplikacji 2



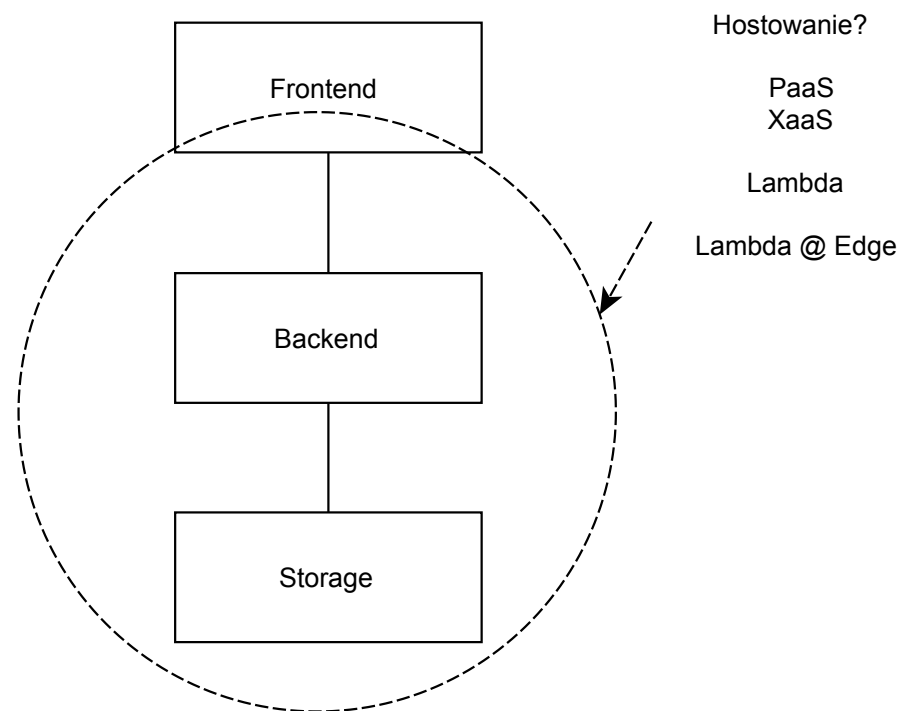
Architektura aplikacji 3



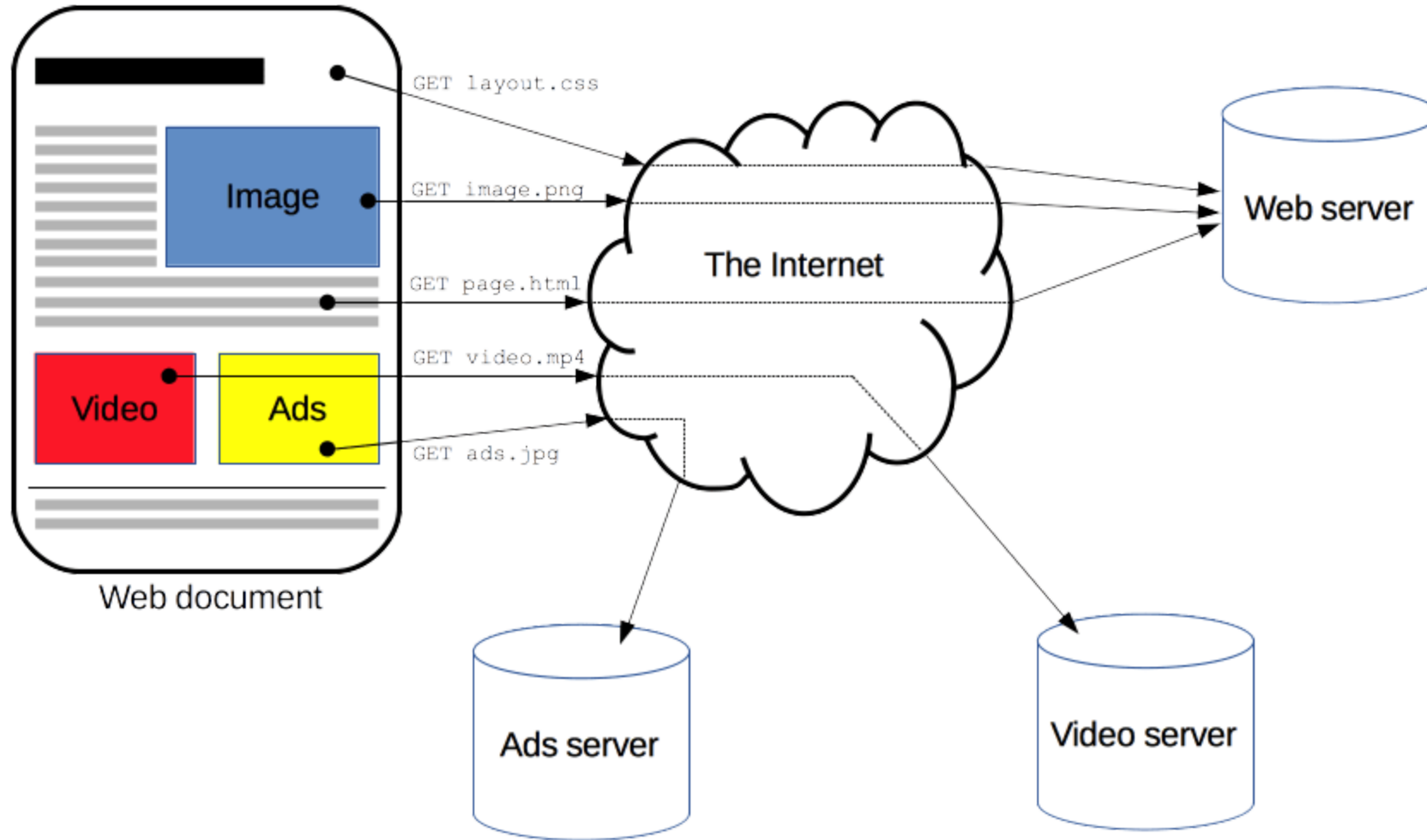
Architektura aplikacji 4



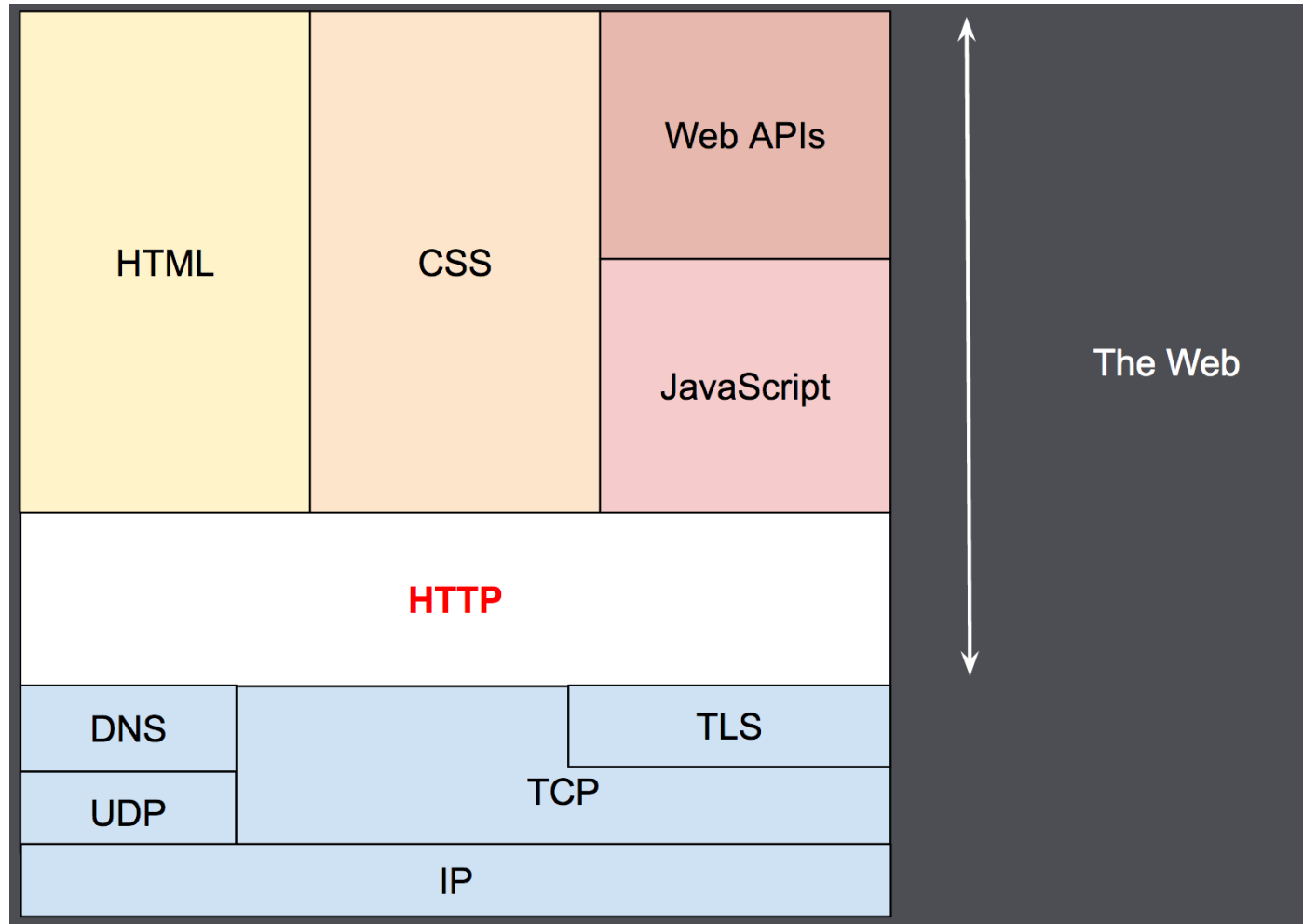
Architektura aplikacji 5



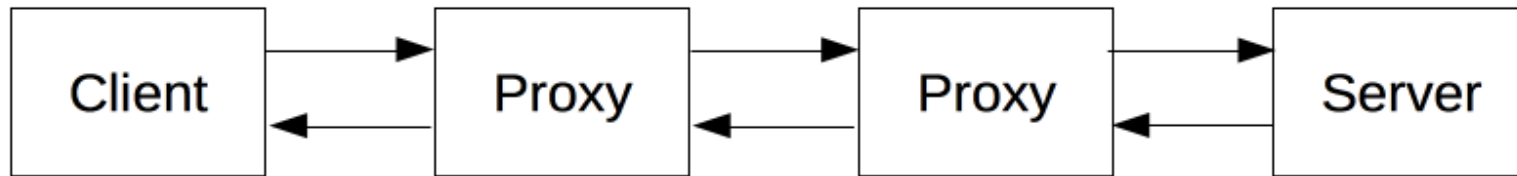
HTTP



HTTP



HTTP



[mozilla docs](#)

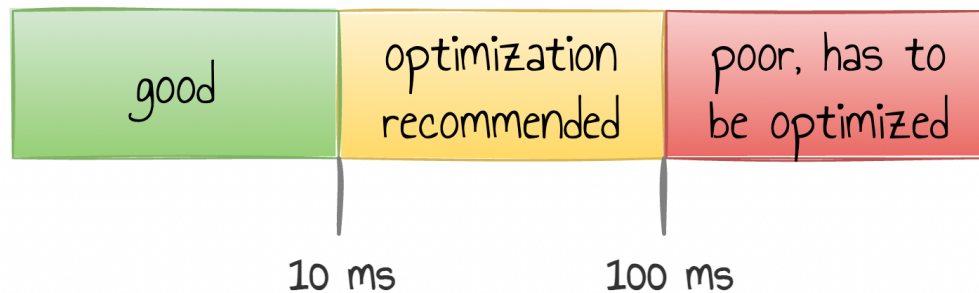
Baza danych

Co to znaczy wolne zapytanie ([src](#)):

SQL

execution time

(for web & mobile workloads)



Dziękuję

Backup slides

Materiały dodatkowe

- [Engineering You, Martin Thompson](#)
- [Designing for Performance, Martin Thompson](#)
- [Modeling is everything](#)
- [Why Mechanical sympathy](#)

Materiały dodatkowe

Metodologie – warto zacząć od Brendana Gregga:

- <http://www.brendangregg.com/usemethod.html>
- <http://www.brendangregg.com/methodology.html>
- [Hałas, a wydajność](#)

Materiały dodatkowe

- wiki.c2.com/?PrematureOptimization