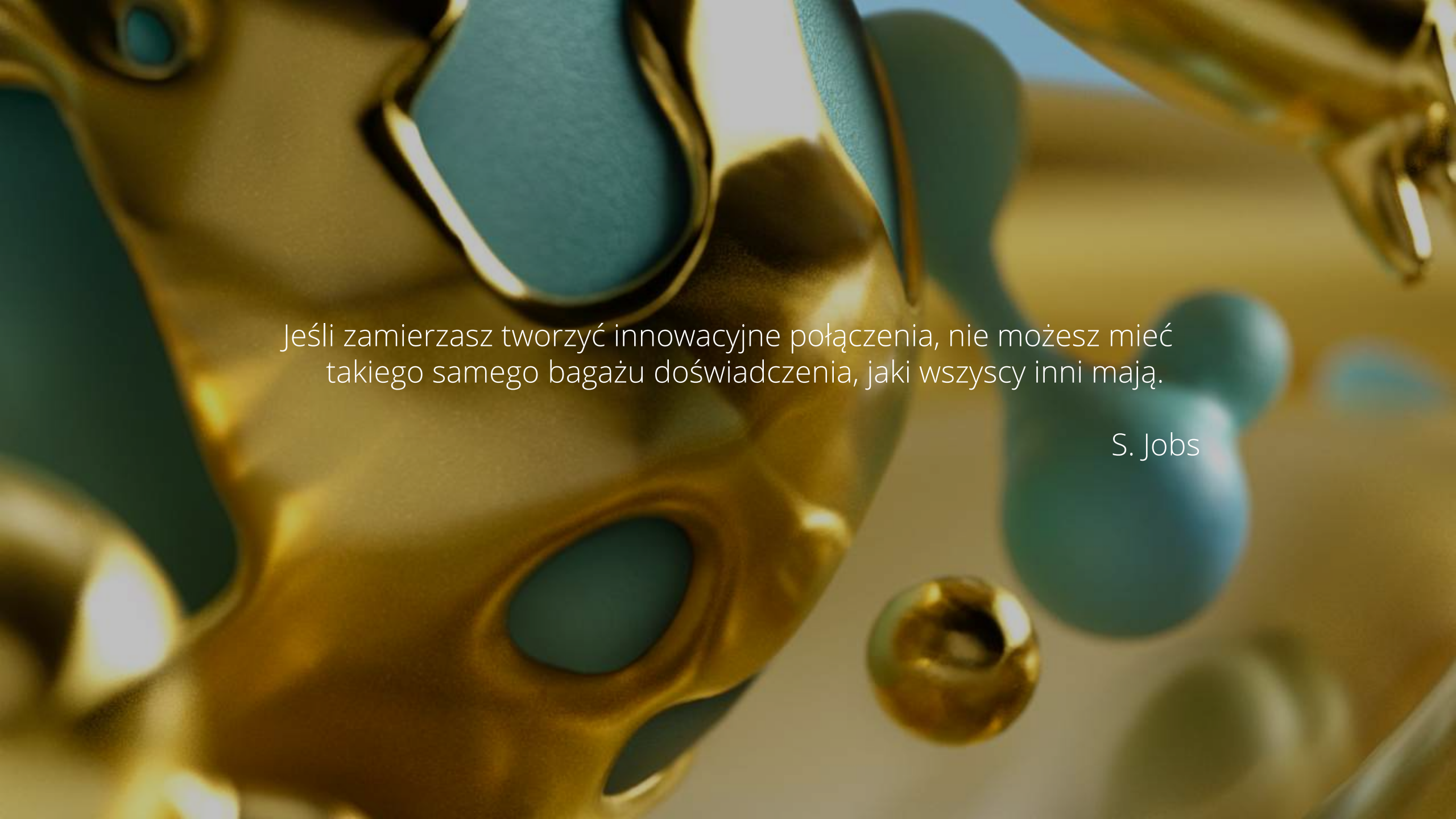


Marzec, 2022

# Technology – what's Next?

**Deloitte.**  
Digital





Jeśli zamierzasz tworzyć innowacyjne połączenia, nie możesz mieć  
takiego samego bagażu doświadczenia, jaki wszyscy inni mają.

S. Jobs



# Spis treści

- 01 Wprowadzenie
- 02 NextJs
- 03 GraphQL
- 04 GraphCMS
- 05 Vercel
- 06 Aplikacja
- 07 Koniec



The background features a blurred, abstract composition of overlapping geometric shapes. A large, curved, golden-yellow shape dominates the upper left, while various shades of blue and teal form the rest of the background, creating a sense of depth and movement.

# Headless CMS

Czym jest i czemu go potrzebujemy.

# Architektura Headless

Typowe CMS'y posiadają część frontendową **HEAD** odpowiedzialną za pobieranie danych z bazy i wyświetlania ich w szablonach wraz z stylami css i innymi dodatkami.

Gdy pozbawimy CMS'a tej części i zostawimy jedynie API Restowe otrzymujemy **Headless CMS**.

**Zaletami** takiego rozwiązania jest fakt że możemy te dane wykorzystać bez problemów w **IoT**, **aplikacjach mobilnych** czy po prostu w **aplikacjach przeglądarkowych**.

Możemy wybrać dowolną technologię frontendową i nie jesteśmy zmuszeni uczenia się technologii narzucanej przez CMS.

Nasz frontend jest niezależny od backendu i możemy rozdzielić je na osobne serwery czy repozytoria a aktualizacja frontendu nie ingeruje w systemy zarządzania treścią.

Headless CMS często są udostępniane w modelu **SaaS**.



The background features a blurred, abstract composition of blue and gold geometric shapes, possibly representing architectural elements or modern design. The blue shapes are more prominent in the foreground, while the gold shapes are visible in the upper and right portions of the frame.

# SaaS

Czym są i czemu są wygodne dla biznesu.

# Oprogramowanie jako usługa

Codziennie każdy z nas korzysta z usług takich jak E-mail zwykle hostowanych przez Google czy Microsoft. To jeden z wielu przykładów usług SaaS. Spotykamy się z tym na każdym kroku również jako programiści, nasze aplikacje mogą korzystać z zabezpieczeń czy skalowalności jakich używają korporacje bez dużych kosztów czy nakładów pracy. Wiele takich usług ma darmowe pakiety community które wystarczają na dosyć długo.



The background features a blurred, abstract composition of blue and gold geometric shapes, possibly representing server racks or data centers, creating a sense of depth and technology.

# Serverless

Czemu coraz częściej o nich słyszymy?



# Technologia i Usługa

Dostawca takiego rozwiązania/usługi zapewnia środowisko oraz odpowiada za jego wydajność, skalowalność i bezpieczeństwo.

Kwestie wymagające największej **odpowiedzialności** i **zasobów** są zmartwieniem dostawcy w skrócie by **tworzyć, rozwijać** i **utrzymywać** aplikację webową nie potrzebujemy własnych serwerów i infrastruktury.

Przykładowo bazami danych, bezpieczeństwem, skalowalnością i innymi czynnikami zarządza odpowiada dostawca, który

świadczy tę usługę wielu odbiorcom, dzięki czemu możliwe jest obniżenie ceny jednostkowej.

Korzystając z takich rozwiązań również płacimy za realnie wykorzystane zasoby a współdzielenie sprzętu czy infrastruktury nie wpływa w żadnym stopniu na ich wydajność.



The background features a blurred, abstract composition of blue and gold geometric shapes, possibly representing architectural elements or modern design. The blue shapes are more prominent in the foreground, while the gold shapes are visible in the upper and background areas.

# Dlaczego Next<sub>js</sub>?

Duża garść fajnych rzeczy.

Sprawdzony przez  
największych

**hulu**

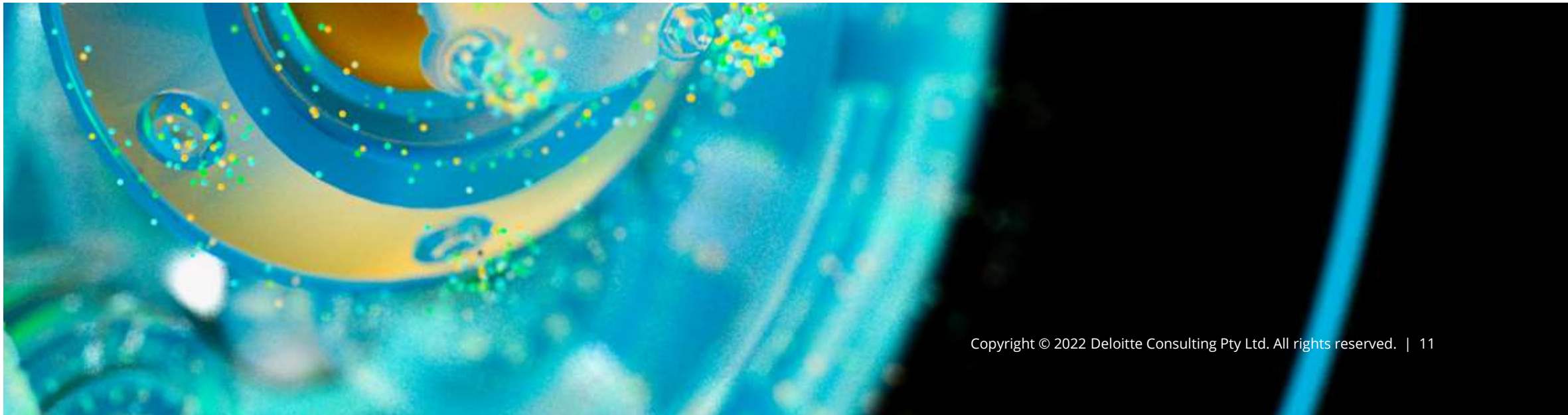


**NETFLIX**

**▲ Vercel**



 **GitHub Copilot**



# Najciekawsze funkcje Nextjs

## 01

### SSR

Jeśli strona korzysta z funkcji renderowania po stronie serwera, kod HTML strony jest generowany przy każdym żądaniu.

Kod generowany przez komponenty Reactowe jest widoczny jak przy zwykłych stronach HTML.

## 02

### SSG

Generowanie statycznych stron HTML oraz plików JSON z danymi.

Jeśli zajdzie taka potrzeba możemy wywołać akcję która pobierze aktualną zawartość z serwera i przykładowo zastąpi dane zapisane w JSON.

## 03

### ISR

Incremental Static Regeneration (umożliwia korzystanie z generowania statycznego dla każdej strony, bez konieczności przebudowywania całej witryny.

Dzięki ISR można zachować korzyści płynące ze statyki przy skalowaniu do milionów stron.



# Dobrze wiedzieć o

## 01

### Code Splitting

W skrócie użytkownik domyślnie dostaje od serwera **tylko ten kod JS który jest potrzebny** na danej podstronie.

W zwykłych aplikacjach reactowych domyślnie użytkownik otrzyma od razu cały kod css/js aplikacji.

## 02

### File-system Routing

W **React Router** przy dłuższym projekcie plik z routingiem zaczyna się strasznie rozrastać bądź zawiera rzeczy które już nie używamy.

**Next** ma własne wbudowane rozwiązanie, routing oparty na strukturach folderów i plików.

## 03

### File system API

Opcjonalnie możemy stworzyć nawet API, tworząc pod folder /api/ a w nim pliki o nazwie endpointów.

Na **Vercelu** możemy dodawać nawet pliki PHP/Go/Python/Ruby do tego folderu.

# Dobrze wiedzieć o

## 04

### No Config

Czyli kompilacja, bundlowanie i optymalizacja pod produkcję już na starcie.

## 05

### CSS in JS

Next wspiera większość sposobów stylowania: **styled components**, **emotion** etc.

## 06

### Łatwy Deploy

Jeśli połączymy go z **Vercel**em i **Github**em to całość zajmie nam kilka minut i sprowadza się do założenia konta, podpięcia repo/domeny i ewentualnie ustawienia kluczy do używanych przez nas API.

# Dobrze wiedzieć o

## 07

### Pełna kontrola

*(Babel/Webpack/Server)*  
Jeśli chcemy to mamy dostęp do konfiguracji narzędzi w przeciwieństwie do innych frameworków czy **CRA React** (jak zrobimy **eject** w React to już tego nie cofniemy a tu po prostu tworzymy plik z konfiguracją)

## 08

### <Image>

Komponent ten zawiera wiele wbudowanych optymalizacji wydajności, które pomagają w osiągnięciu dobrych wyników **Core Web Vitals**. Te wyniki są ważnym miernikiem komfortu użytkownika witryny i są uwzględniane w rankingach wyszukiwania Google.

## 09

### Web Vitals Reporting

Możemy w prosty sposób analizować statystyki ładowania naszych podstron

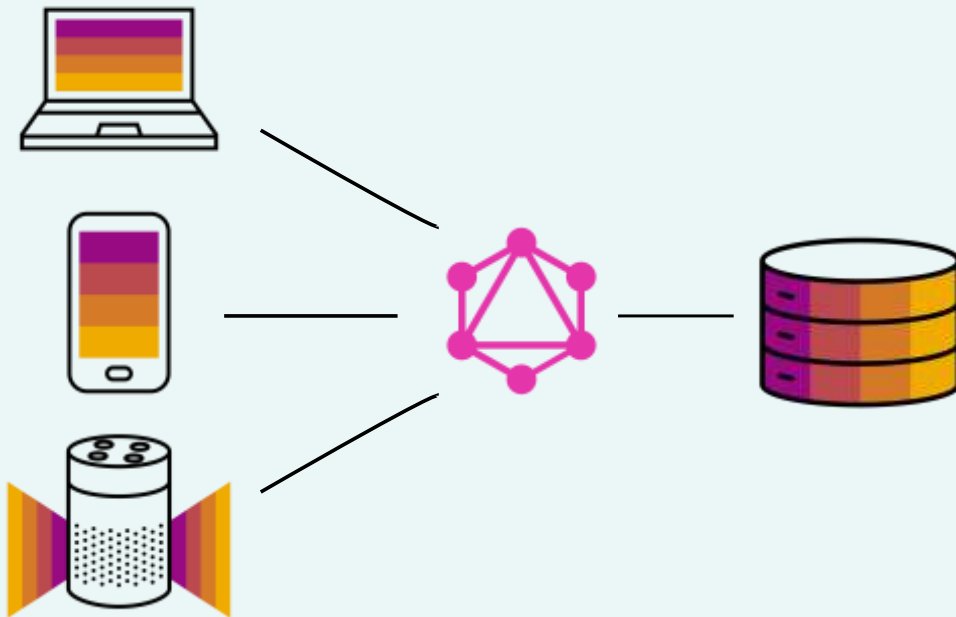
# Kiedy używać?

- Wtedy gdy aplikacja nie będzie wewnętrzna a dostępna dla ogółu.
- Gdy chcemy mieć sprawdzony zestaw narzędzi routing, optymalizacja, struktura, mniej kodu do napisania :)
- Bezpieczeństwo, dzięki temu że używają go duże firmy oraz wiele teamów ewentualne dziury będą szybciej wyłapane.
- Wsparcie społeczności



# Kiedy to zły wybór

- Mała prosta strona
- Nie potrzebujemy Reacta
- Projekt bez większych planów na rozwój



# REST in peace



## Pora na **GraphQL**

W skrócie to *Query Language* wykorzystywany do obsługi **API**. Oferuje większą precyzję i elastyczność w porównaniu do klasycznego **REST** API.

Dzięki niemu możesz *zapytać* tylko o te dane które są ci potrzebne bez nadmiarowych danych które zwykle się pojawiają w REST API.

Przykładowo dla użytkowników mobilnych możemy *zapytać API* tylko o tytuł i adres do obrazka a w desktopowej aplikacji dodatkowo o autora wraz z jego avatarą i treść.

Pozwala to oszczędzać transfer i czas wykonywania zapytania. To jest szczególnie ważne gdyż urządzenia mobilne to około 2/3 ruchu na stronach www.

Został stworzony przez **Facebook'a** w **2018** by rozwiązać problemy związane z różnicami potrzebnych danych urządzenia/mobile/desktop oraz dużej ilości requestów do endpointów by je uzyskać.

# Jak działa GraphQL

## 01

### Queries

Komunikacja opiera się o zapytania które definiujemy z pomocą słowa **Query** oraz nazwy poszczególnych zasobów. W odpowiedzi otrzymujemy obiekt **JSON** o takiej samej strukturze, lecz wypełnionymi danymi.

## 02

### Resolvers

Do serwera GraphQL musimy dostarczyć dane, z pomocą przechodzą nam Resolvery które wskazują skąd GraphQL ma pobrać dane.

## 03

### Schema

Opisana struktura danych wraz z ich typami które wymieniamy między klientem a serwerem.

Programista na frontendzie oraz backendzie wie dzięki temu dokładnie z czego może złożyć zapytanie oraz z czego będzie składała się informacja zwrotna od serwera.

# Rest API

Obok jest przykład zapytania wykonanego w klasyczny sposób.

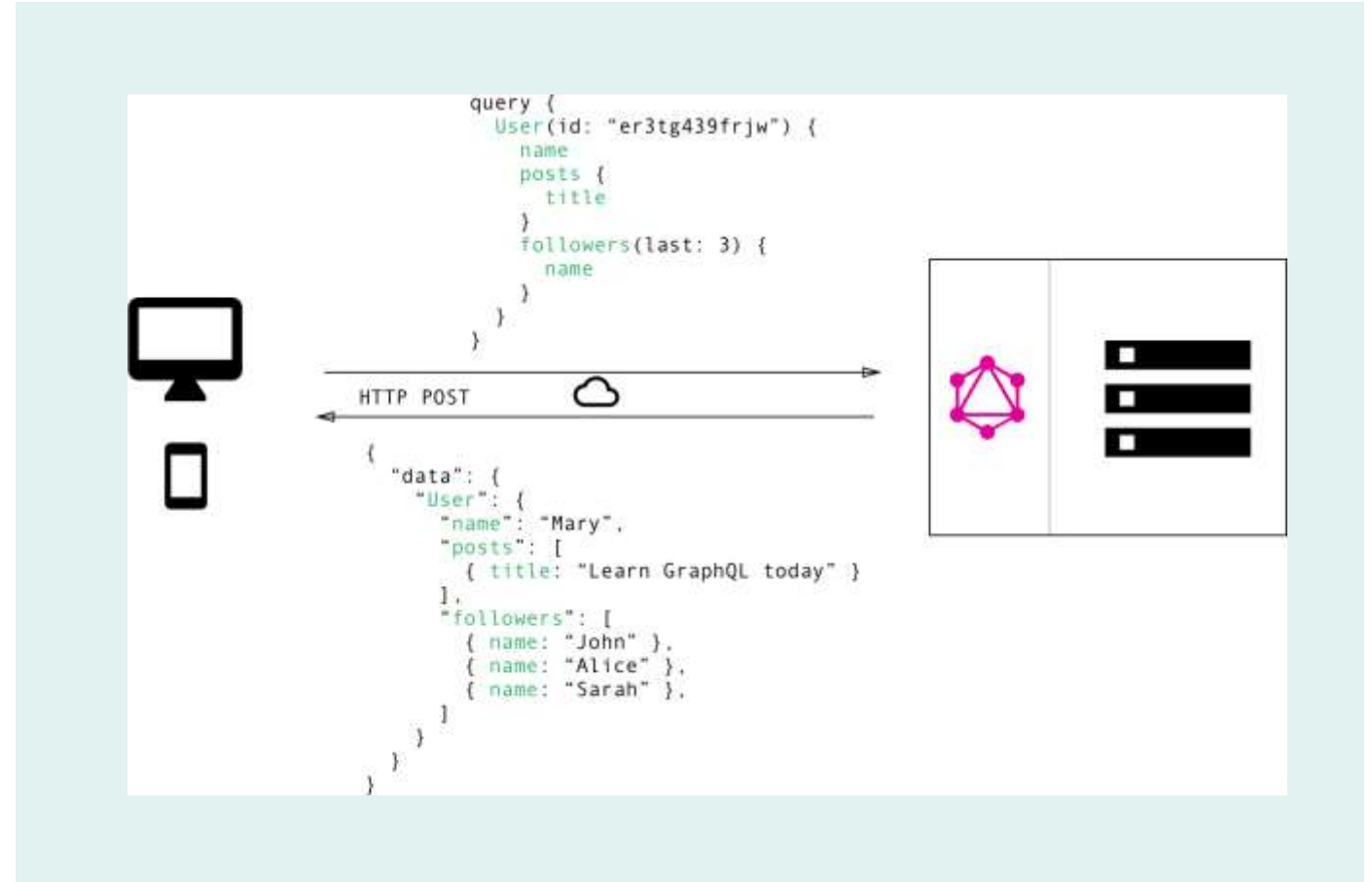




# GraphQL

To samo zapytanie tylko przy użyciu GraphQL.

Budowa jest dużo prostsza i naturalna w zrozumieniu.





# GraphCMS

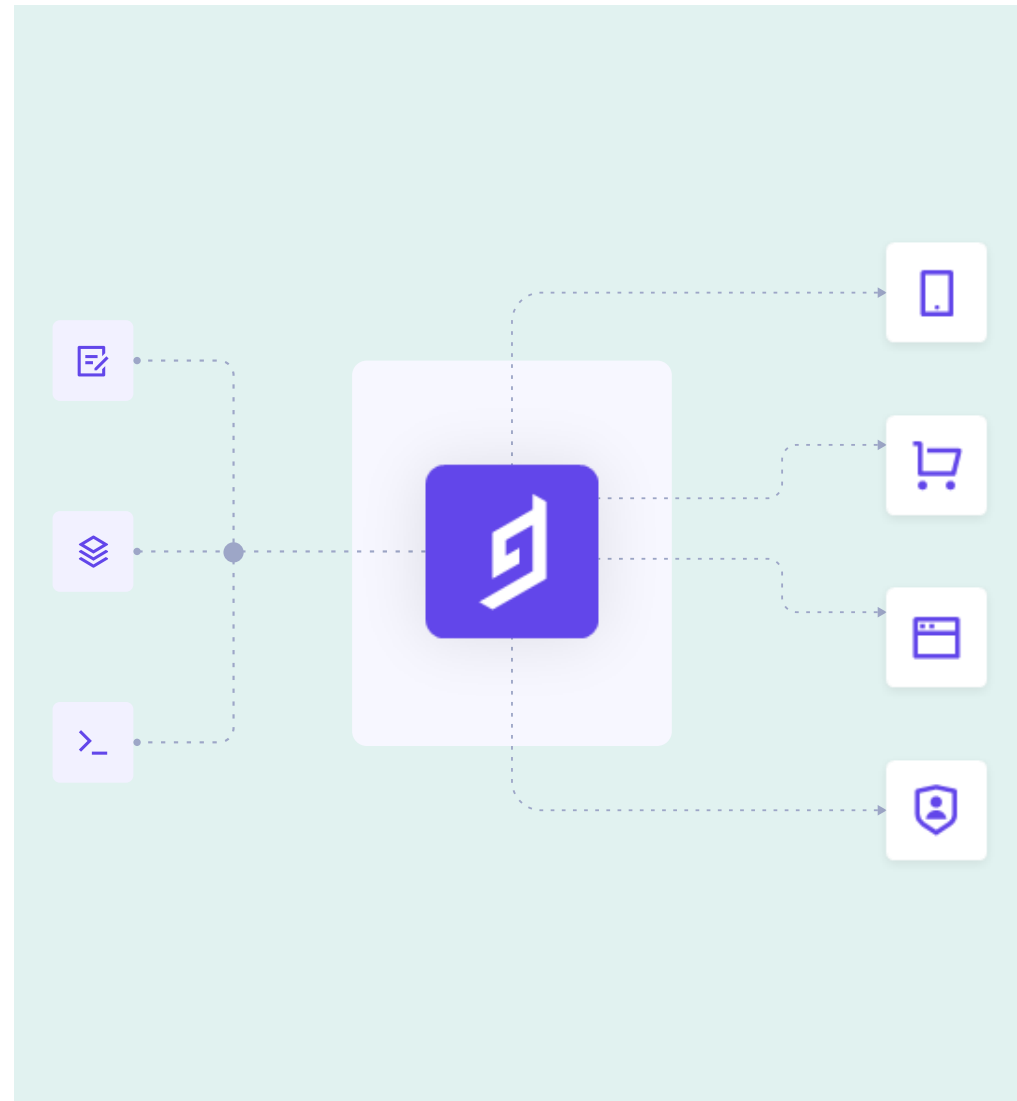
CMS jakiego zawsze potrzebowałeś.

# Idealny do pracy

**PWA, eCommerce, aplikacje desktopowe i wiele innych** możliwości dzięki **Next.js**

**GraphCMS** dysponuje elastycznym i wydajnym API do zarządzania treścią i **schema**, a także błyskawicznie działającym API do zarządzania treścią.

**Wiele gotowych repozytoriów** często uruchomienie projektu to wybranie projektu w **NextJs**, **Angularze** czy **Vue** a następnie zatwierdzenie szablonu oraz sklonowanie repozytorium



**Develop.**

**Preview.**



**Ship.**



# Develop

Vercel został stworzony *przez developerów dla developerów*.

Starają się by zapewnić nam to czego brakuje nam na co dzień w projektach.

W połączeniu z **Nextem** (na którego niedawno otrzymali **108 milionów dolarów**) możemy mieć nawet edycję, podgląd a nawet dyskusję całego teamu **na żywo!**

Dzięki tej funkcji mamy dostęp bezpośrednio w przeglądarce dzięki czemu jest to szybsze od środowiska lokalnego.



# Preview

Rozwój frontendu nie powinien być działaniem wykonywanym w pojedynkę.

Platforma Vercel sprawia, że staje się to doświadczeniem opartym na współpracy, z podglądem każdej zmiany kodu, dzięki bezproblemowej integracji z GitHub, GitLab i Bitbucket.

## **Push to Deploy**

Wystarczy zwykły **Push** na wybraną gałąź by uruchomić **Deploy**.

## **Get your Preview URL**

Każdy build na gałęzi z podglądem aplikacji dostaje w komentarzu linka do podglądu.

Przykładowo możemy przejrzeć 5 ostatnich **releasów** ot tak klikając link w komentarzu

## **Share and Collaborate**

Możemy przekazać czy to testerom czy klientowi linka do wybranego przez nas builda w celu sprawdzenia a następnie po prostu zaktualizować wersję produkcyjną.



# Ship

Zmiany są wprowadzane natychmiastowo w globalnej sieci brzegowej (eng. Edge network).

Wszystko jest załatwiane za ciebie od SSL przez kompresję zasobów po walidację cache.

Szybkość ma kluczowe znaczenie dla **klientów** oraz **SEO**. **Vercel** wykracza poza zwykłe buforowanie kodu jakie oferują nam zwykłe hostingi, potrafi skalować się do milionów stron dzięki dynamicznemu wykonywaniu kodu.

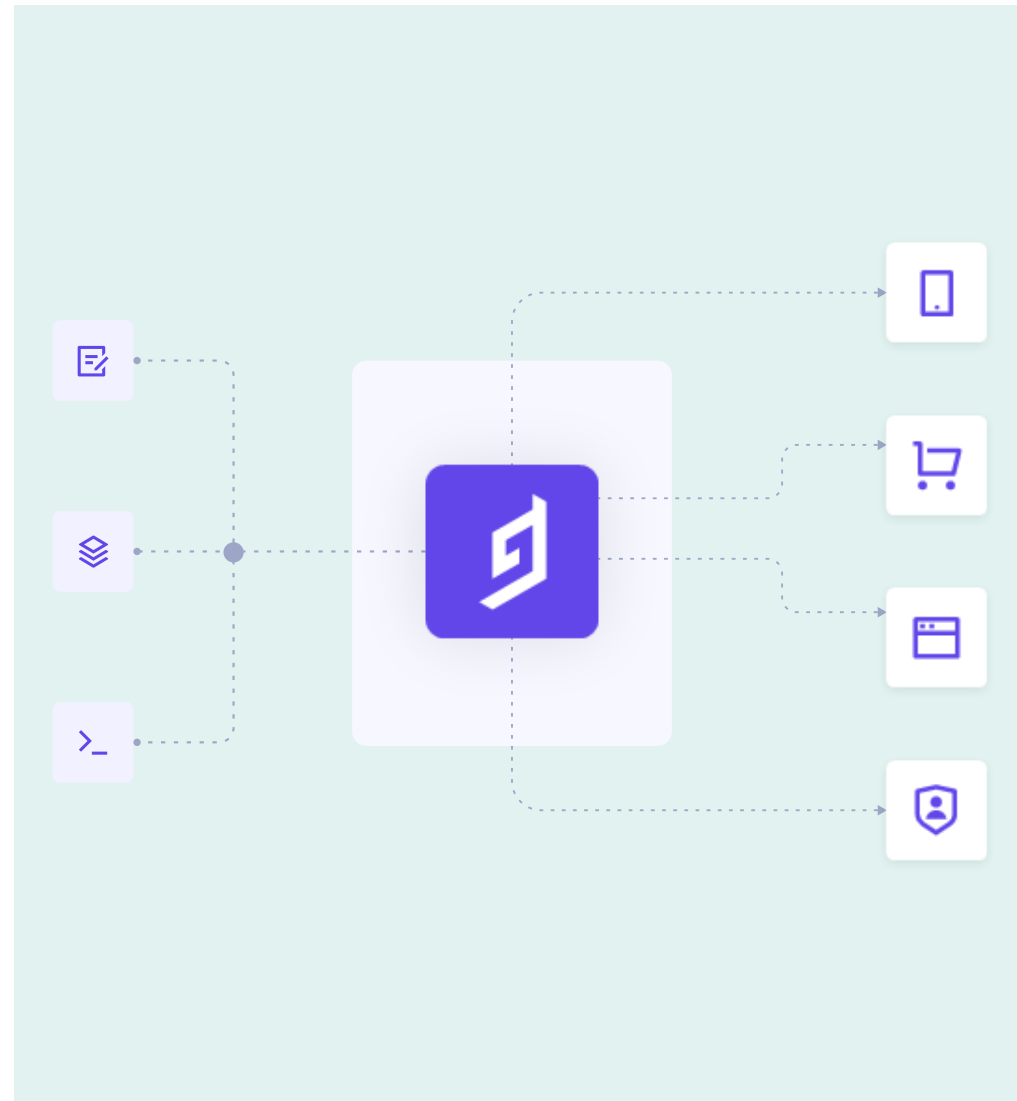


# Zbawienie dla developerów

Gdy zajmujemy się frontendem czy backendem w js to ostatnie o czym myślimy to hosting/środowisko pod naszą aplikację.

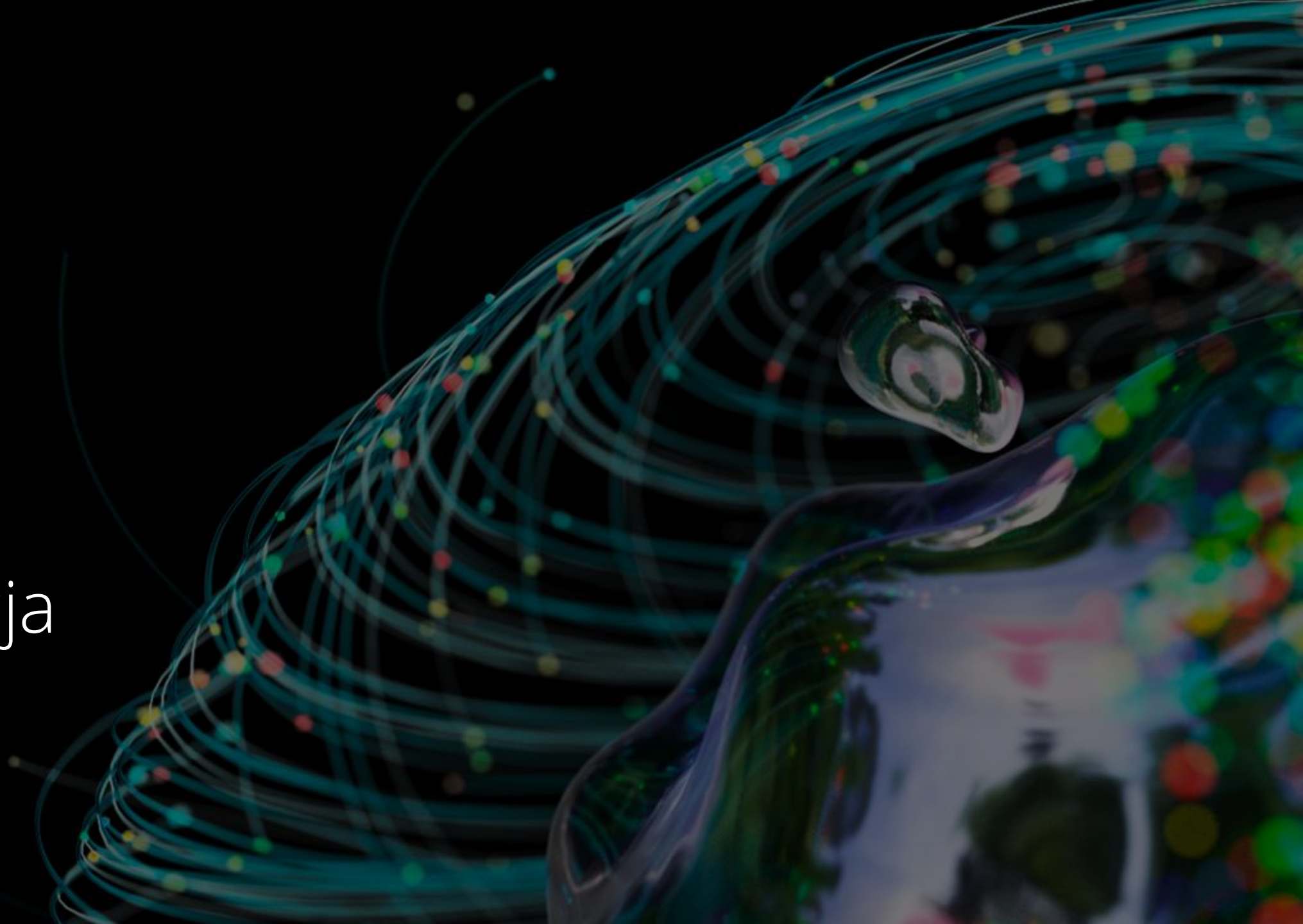
**Vercel** przychodzi nam z pomocą. Oferuje nam w 100% darmowy bez potrzeby podawania karty pakiet **community** a w nim:

- Zautomatyzowany Deploy
- Integracje z repozytorium Git
- 100GB pojemności
- 6K minut miesięcznie na buildy
- 10s na Serverless Function



# Aplikacja

Część Praktyczna





**Deloitte.**  
Digital