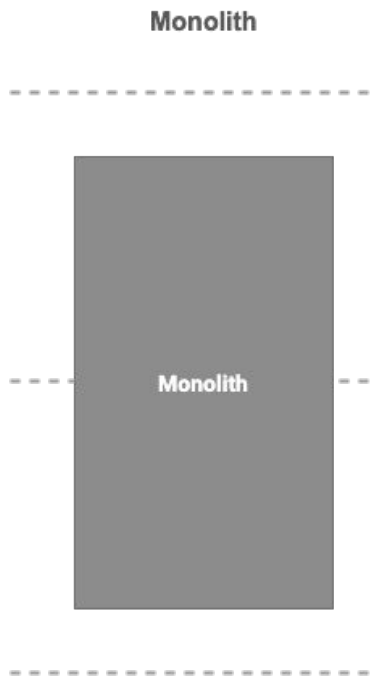




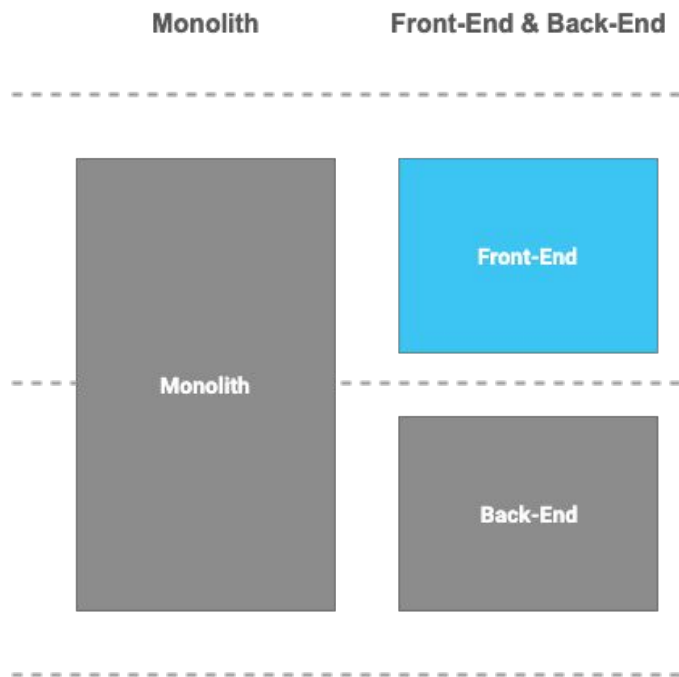
Weźże odśwież i zeskaluuj!

Mateusz Ziarko, Engineering Manager @ VirtusLab

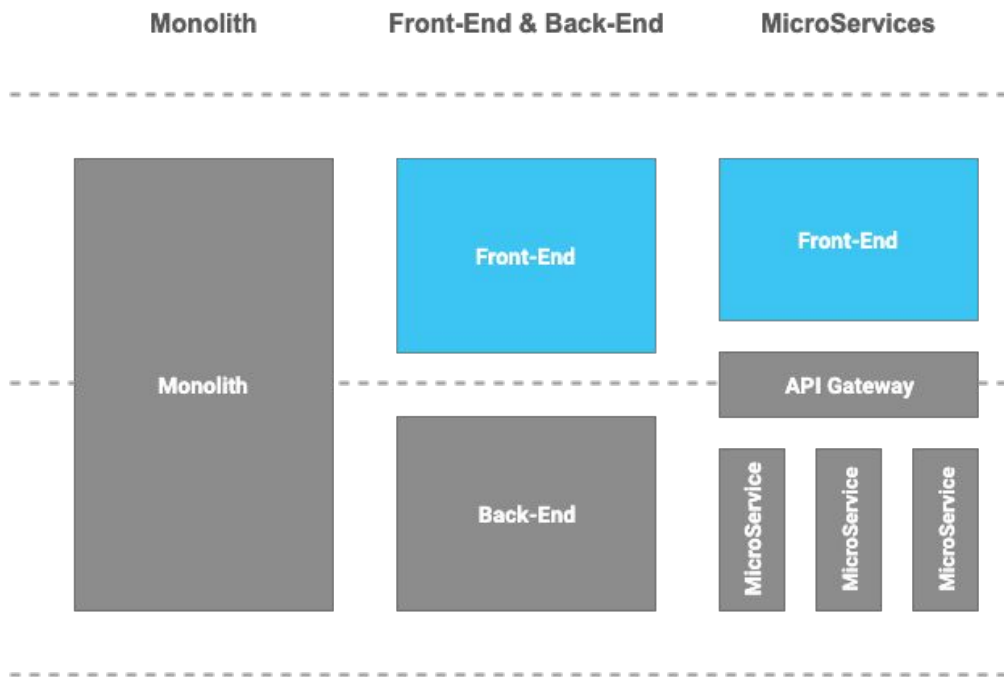
Ewolucja aplikacji



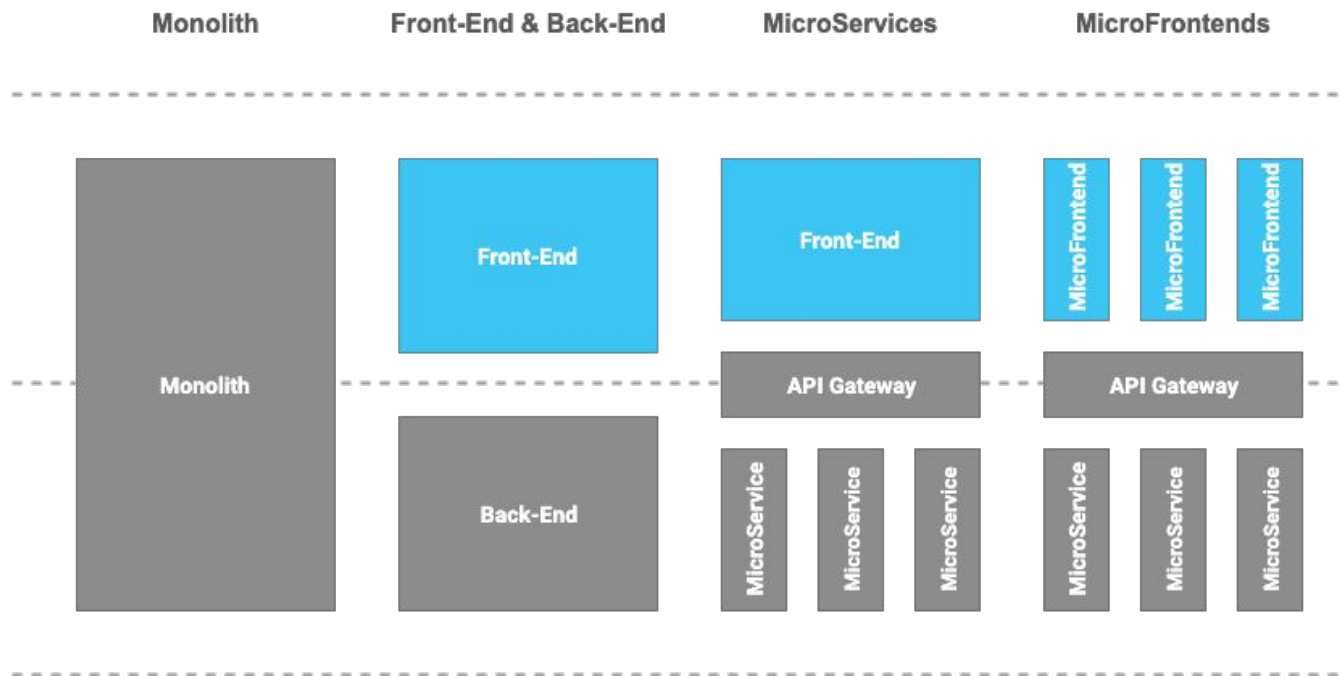
Ewolucja aplikacji



Ewolucja aplikacji

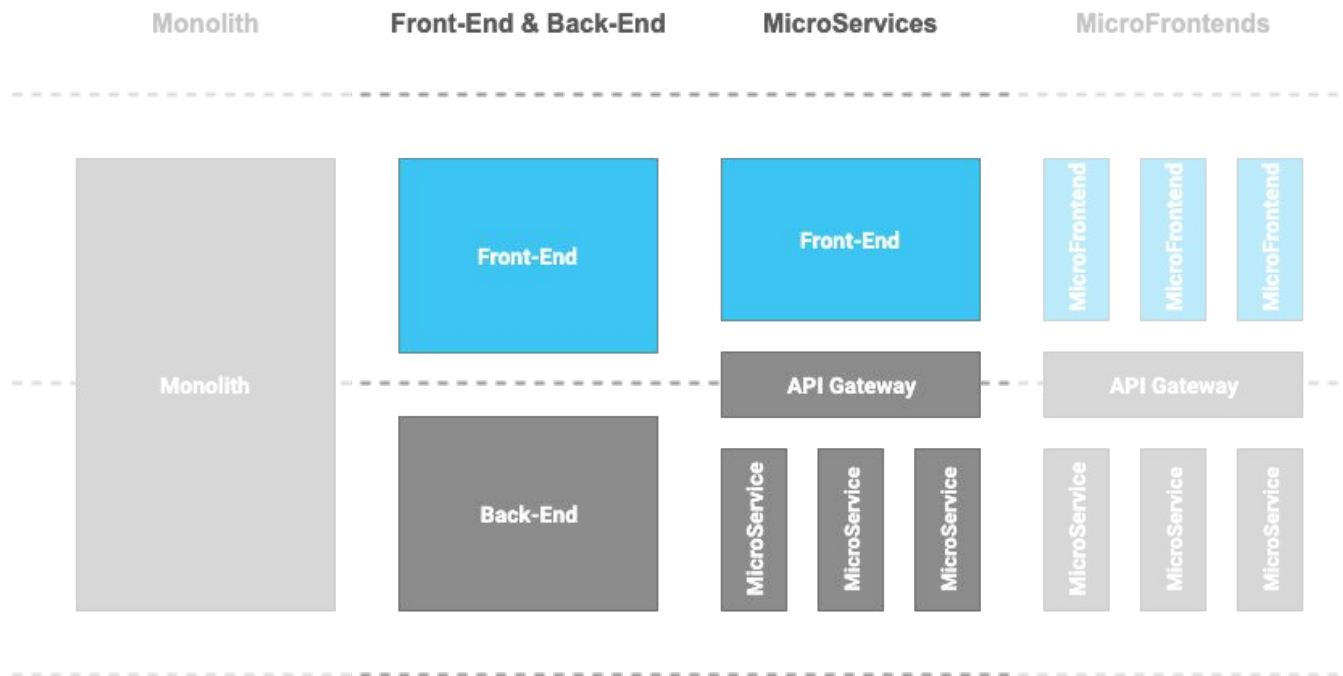


Ewolucja aplikacji



Ustalmy stan obecny

Stan obecny



Jakie są możliwości?

Big Bang

**Legacy
as a Service**

Jakie są możliwości?

Big Bang

Legacy

as a Service

Legacy as a Service

Front-End

Back-End

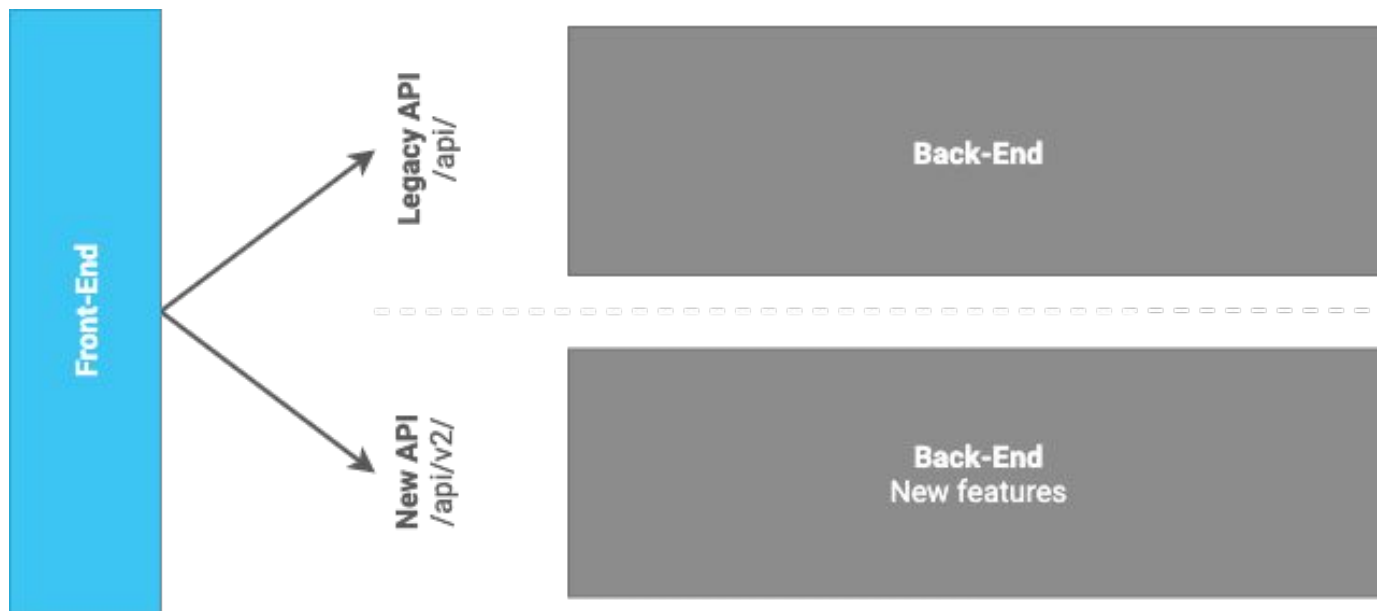
Legacy as a Service

Front-End

Back-End



Odcięcie i powolna migracja

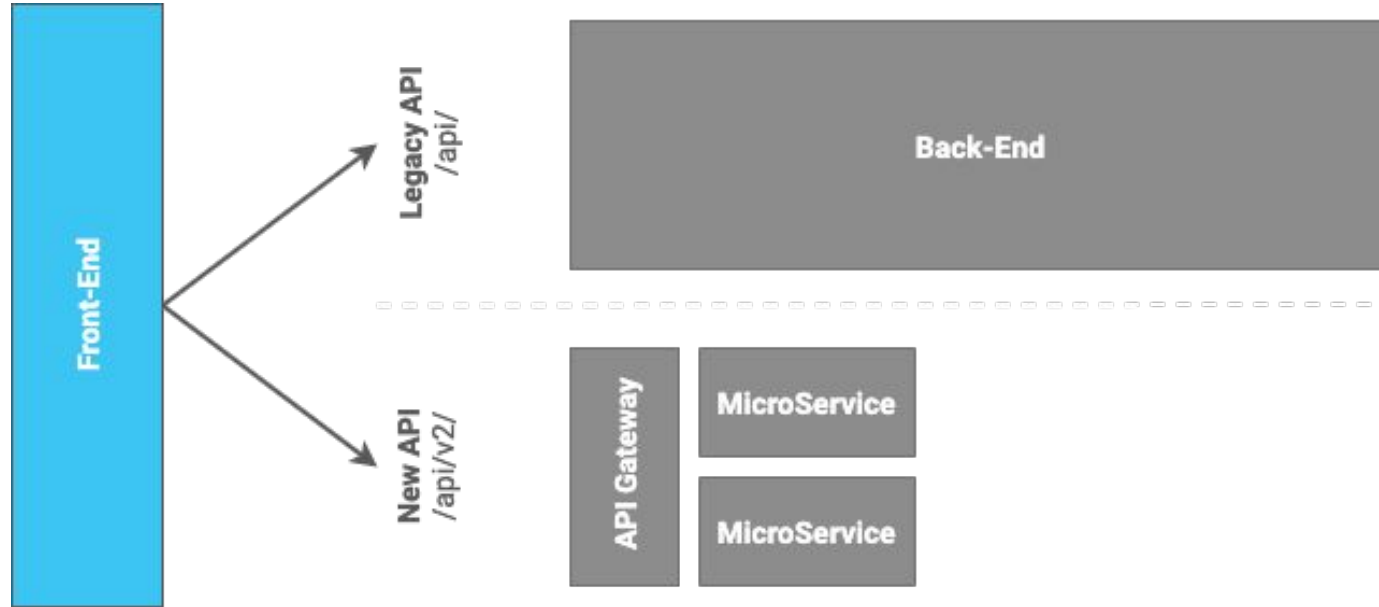


MicroServices

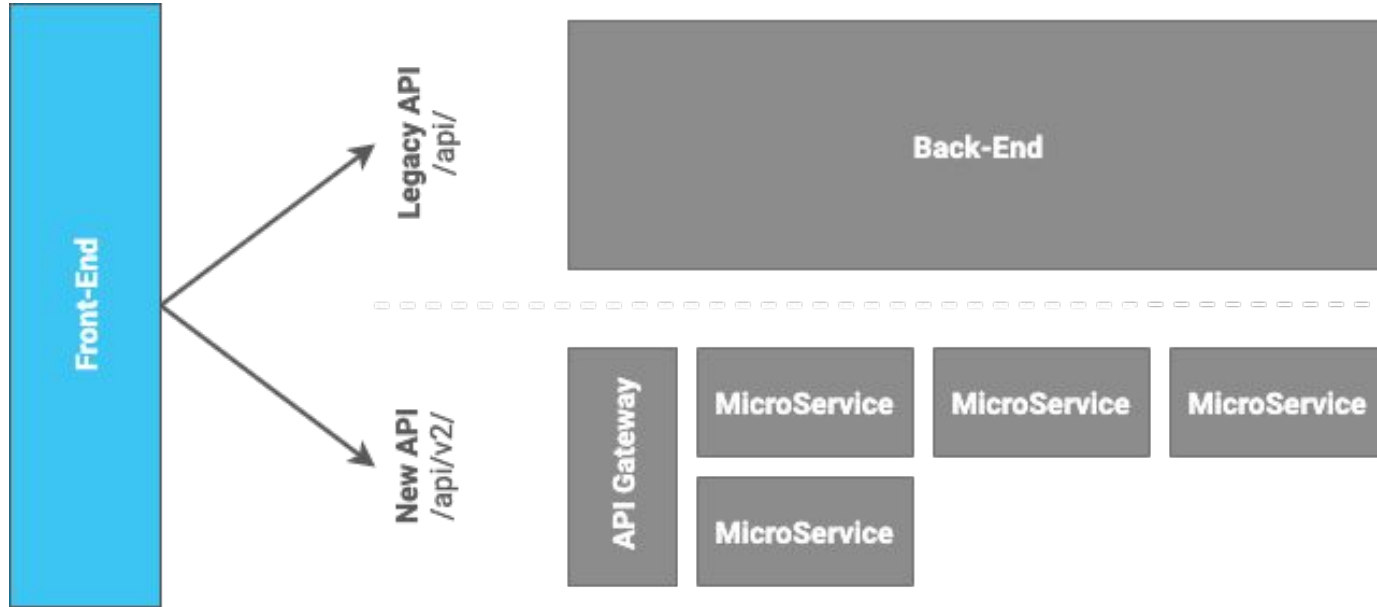
Serwisy w architekturze MicroService'owej:

- są niezależnie develop'owane i deploy'owalne
- są zdecentralizowane i możliwie jak najbardziej zautomatyzowane
- mogą być napisane z wykorzystaniem różnych języków, być połączone z różnymi bazami danych, sprzętem i środowiskiem deweloperski.

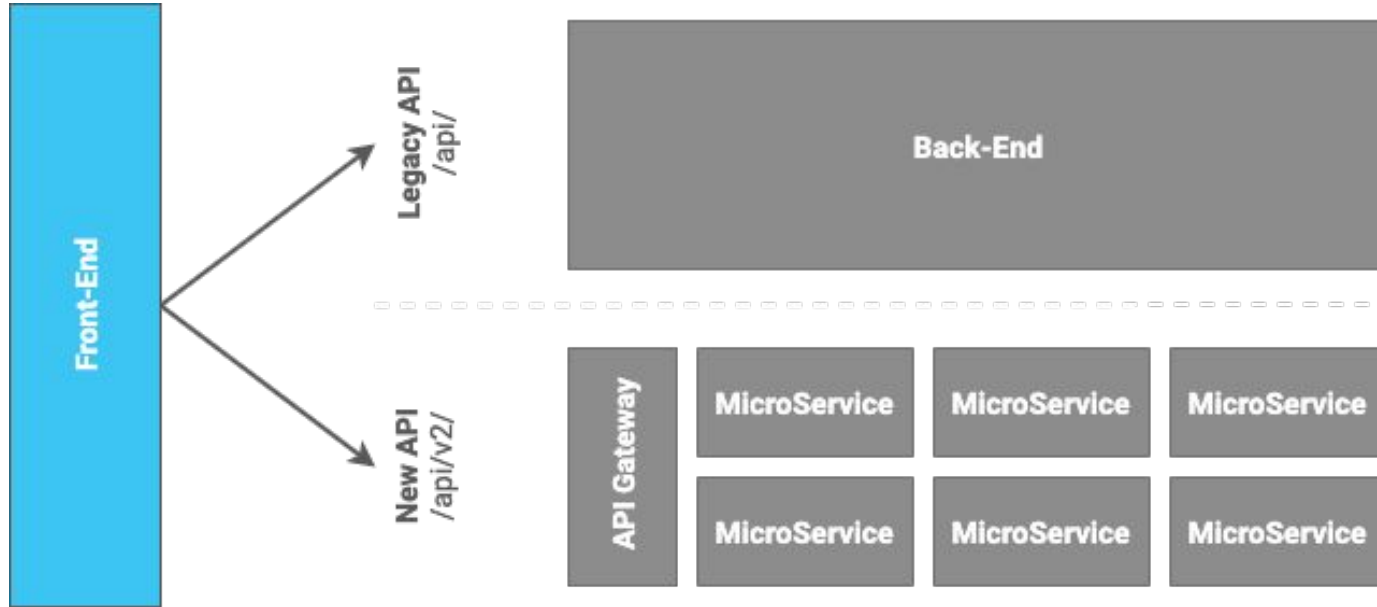
MicroServices



MicroServices



MicroServices



MicroServices



MicroServices

```
import { NestFactory } from '@nestjs/core';
import { Transport } from '@nestjs/microservices';
import { ApplicationModule } from './app.module';

async function bootstrap() {
  const app = await NestFactory.createMicroservice(ApplicationModule, {
    transport: Transport.TCP,
  });
  app.listen(() => console.log('Microservice is listening'));
}

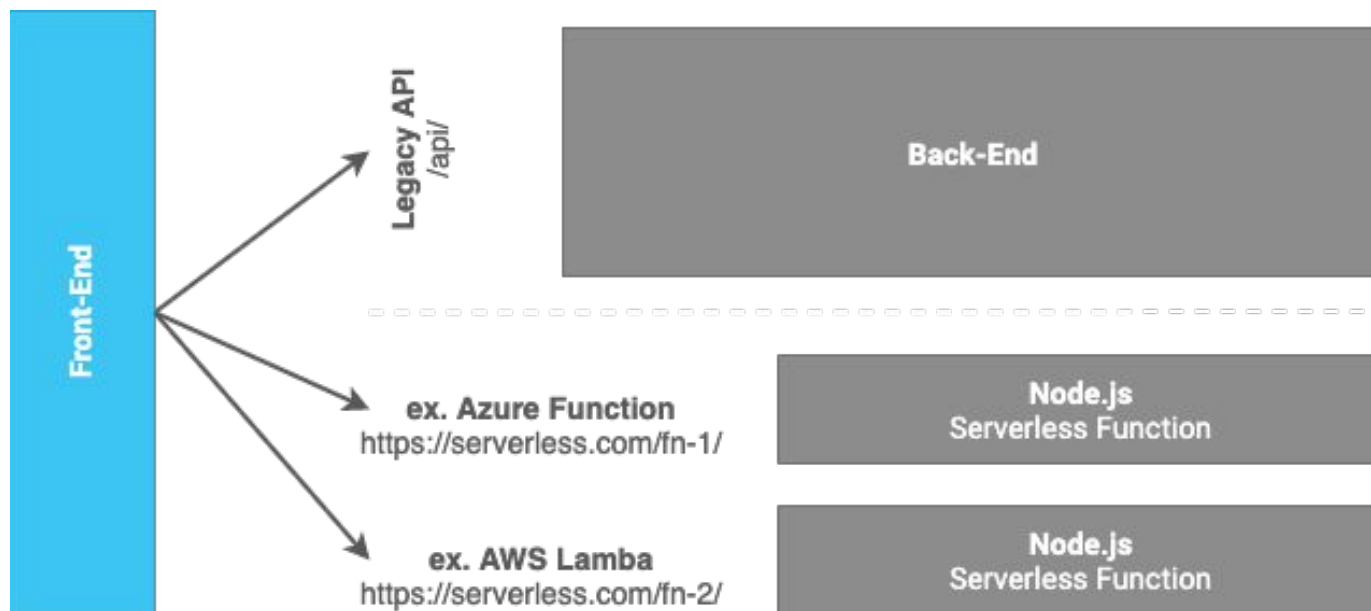
bootstrap();
```

Serverless

Lambdas, Functions itd.:

- bardzo podobne w głównych założeniach do MicroServices
- mniej rozbudowana logika i zdecydowanie bardziej restrykcyjne ramy działania
- mniej zasobożerne w porównaniu z pełnoprawnymi np. AWS EC2
- każda platforma wymaga definicji funkcji i hook'ów w specyficzny dla siebie sposób

Serverless



Serverless

```
import { NowRequest, NowResponse } from '@now/node'

export default (req: NowRequest, res: NowResponse) => {
  res.json({ name: 'John', email: 'john@example.com' })
}
```

```
module.exports = function(context, res) {
  context.res = {
    body: { name: 'John', email: 'john@example.com' }
  };
  context.done();
}
```

Serverless

```
{  
  "bindings": [{  
    "authLevel": "anonymous",  
    "type": "httpTrigger",  
    "direction": "in",  
    "name": "req"  
  }, {  
    "type": "http",  
    "direction": "out",  
    "name": "res"  
  }]  
}
```

Serverless

Zeit.co

GCloud

Azure

AWS

Serverless

Zeit.co

GCloud

Azure

AWS

Legacy as a Service

Front-End

Back-End

Legacy as a Service

Front-End

Back-End

MicroFrontends

Mówiąc o MicroFront'endach trzeba zacząć myśleć o aplikacji jako pewnej kompozycji niezależnych funkcjonalności, która jest budowana przez kilka zespołów:

- o zróżnicowanym focus'ie biznesowym
- multi-funkcjonalnych i działających end-to-end

MicroFrontends

Mówiąc o MicroFront'endach trzeba zacząć myśleć o aplikacji jako pewnej kompozycji niezależnych funkcjonalności, która jest budowanym przez kilka zespołów:

- o zróżnicowanym focus'ie biznesowym
- multi-funkcjonalnych i działających end-to-end

Pełny Ownership

MicroFrontends

Na MicroFront'end składają się **4 główne** założenia:

- pełny ownership zespołu
- Tech agnostic
- Browser Native API > Custom API
- Zawsze dostępne

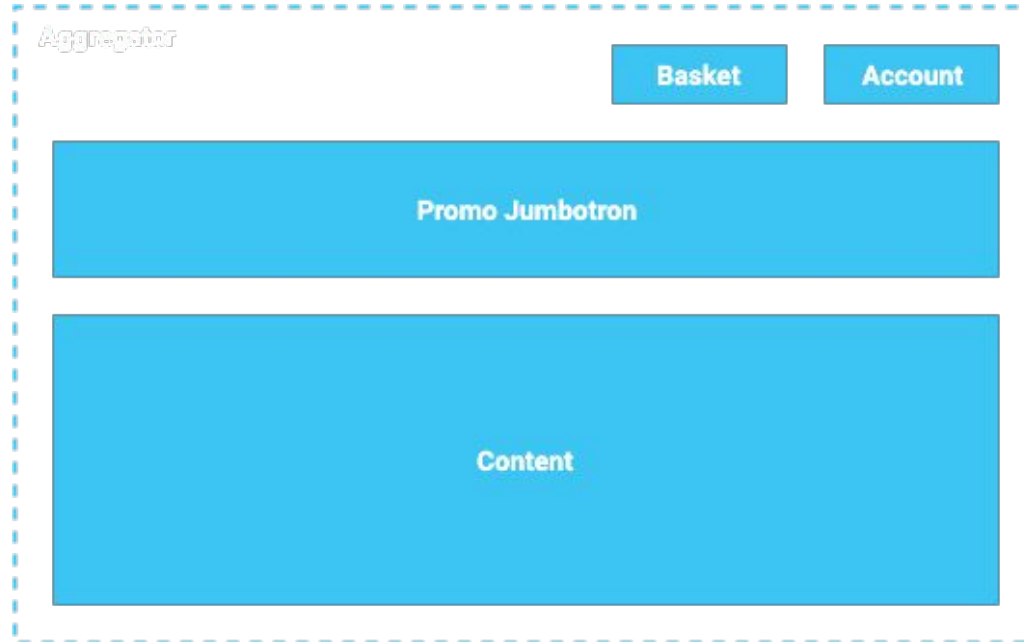
MicroFrontends

Na MicroFront'end składają się **4 główne** założenia:

- pełny ownership zespołu
- Tech agnostic
- Browser Native API > Custom API
- Zawsze dostępne

SSR / Universal Rendering

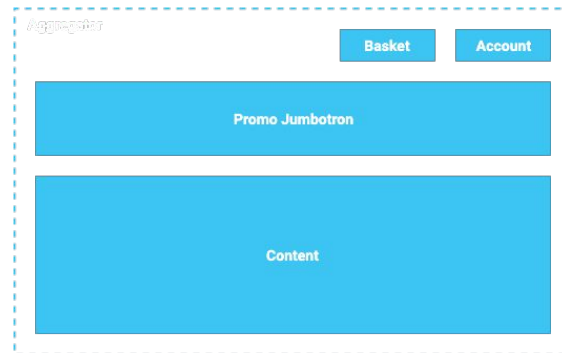
MicroFrontends



MicroFrontends

Można podzielić pracę na **4 zespoły**:

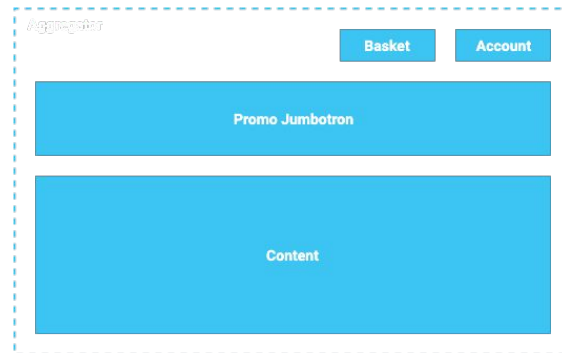
- Identity Team
- Checkout Team
- Marketing Team
- Content Team



MicroFrontends

Można podzielić pracę na **4 zespoły**:

- Identity Team - **ReactJS & TypeScript**
- Checkout Team - **VueJS & ESNext**
- Marketing Team - **Czyste ES6 lub TypeScript**
- Content Team - **Angular & TypeScript**



MicroFrontends

hypernova

<https://github.com/airbnb/hypernova>

★ 5.3k

single-spa

<https://github.com/CanopyTax/single-spa>

★ 4.2k

MicroFrontends

```
const hypernova = require('hypernova/server');  
hypernova({  
  devMode: true,  
  getComponent(name) {  
    if (name === 'MyComponent.js') {  
      return require('./app/assets/javascripts/MyComponent.js');  
    }  
    return null;  
  },  
  port: 3030,  
});
```

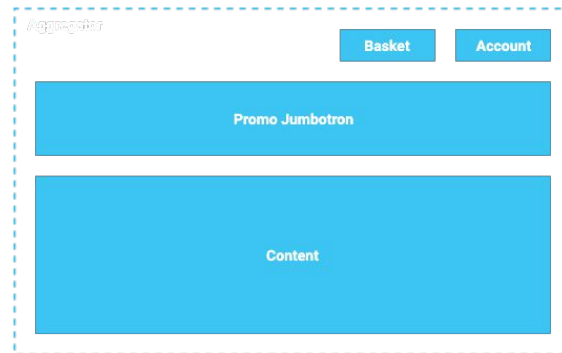
MicroFrontends

```
const React = require('react');
const renderReact = require('hypernova-react').renderReact;
function MyComponent(props) {
  return <div>Hello, {props.name}!</div>;
}
module.exports = renderReact('MyComponent.js', MyComponent);
```

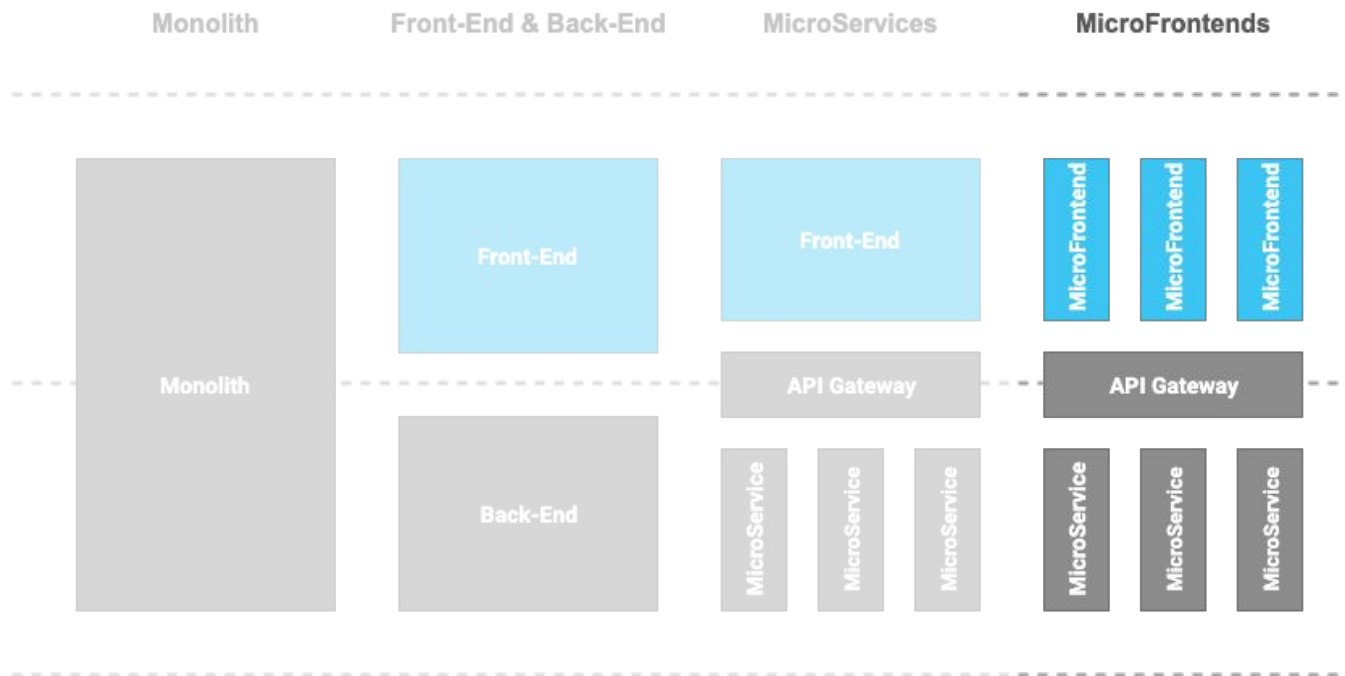
MicroFrontends

Jakie widoczne plusy dają nam MicroFront'endy?

- Prostszy maintenance
- Niezależny deployment funkcjonalności
- Szybkość, szybkość czyli duch Isomorphic Web App

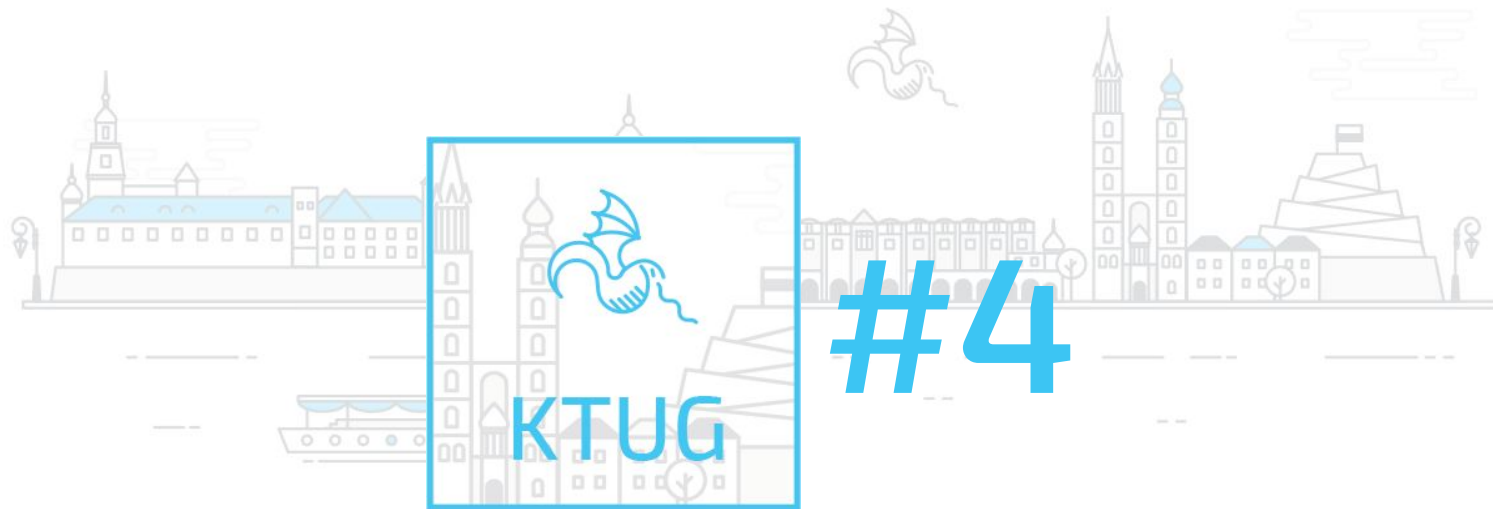


Cel osiągnięty!





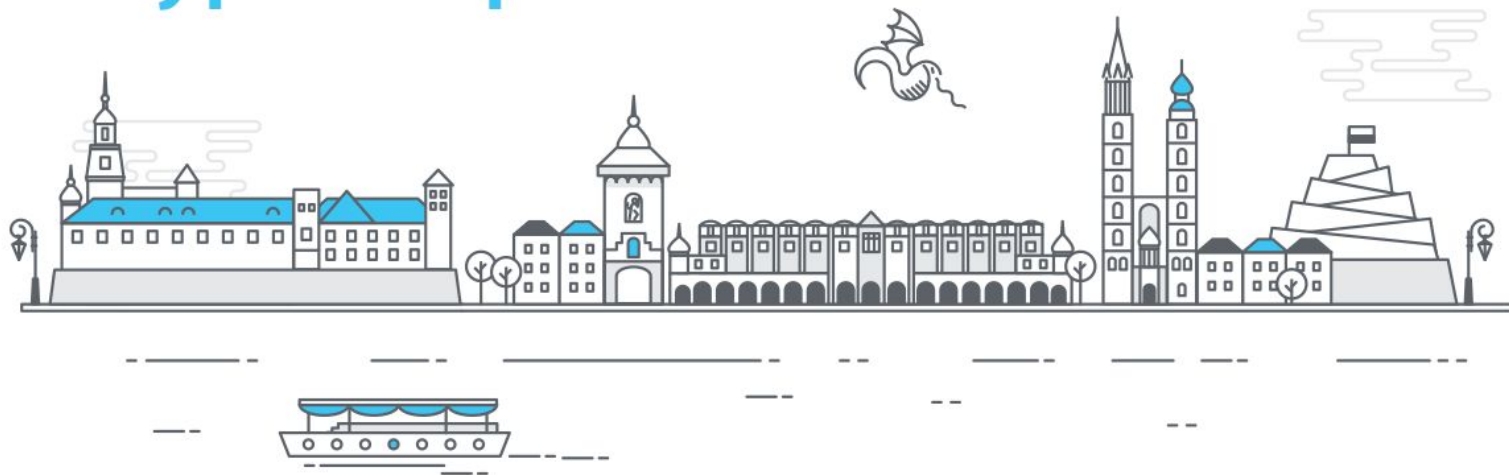
Pytania?



Dziękuję za uwagę!

Kraków TypeScript

A Type-Safe Kingdom



Sponsored by

