

# MOCKOWANIE API

Praktyczne ćwiczenia przed pierwszą randką z backendem.

Meet.js, Wrocław. 29.06.2023

Łukasz Nowak, HL Tech

# ŁUKASZ NOWAK

✉ [lukasz.nowak@hltech.com](mailto:lukasz.nowak@hltech.com)

🐙 [LukaszNowakPL](#)

in [Łukasz Nowak](#)

## HL TECH

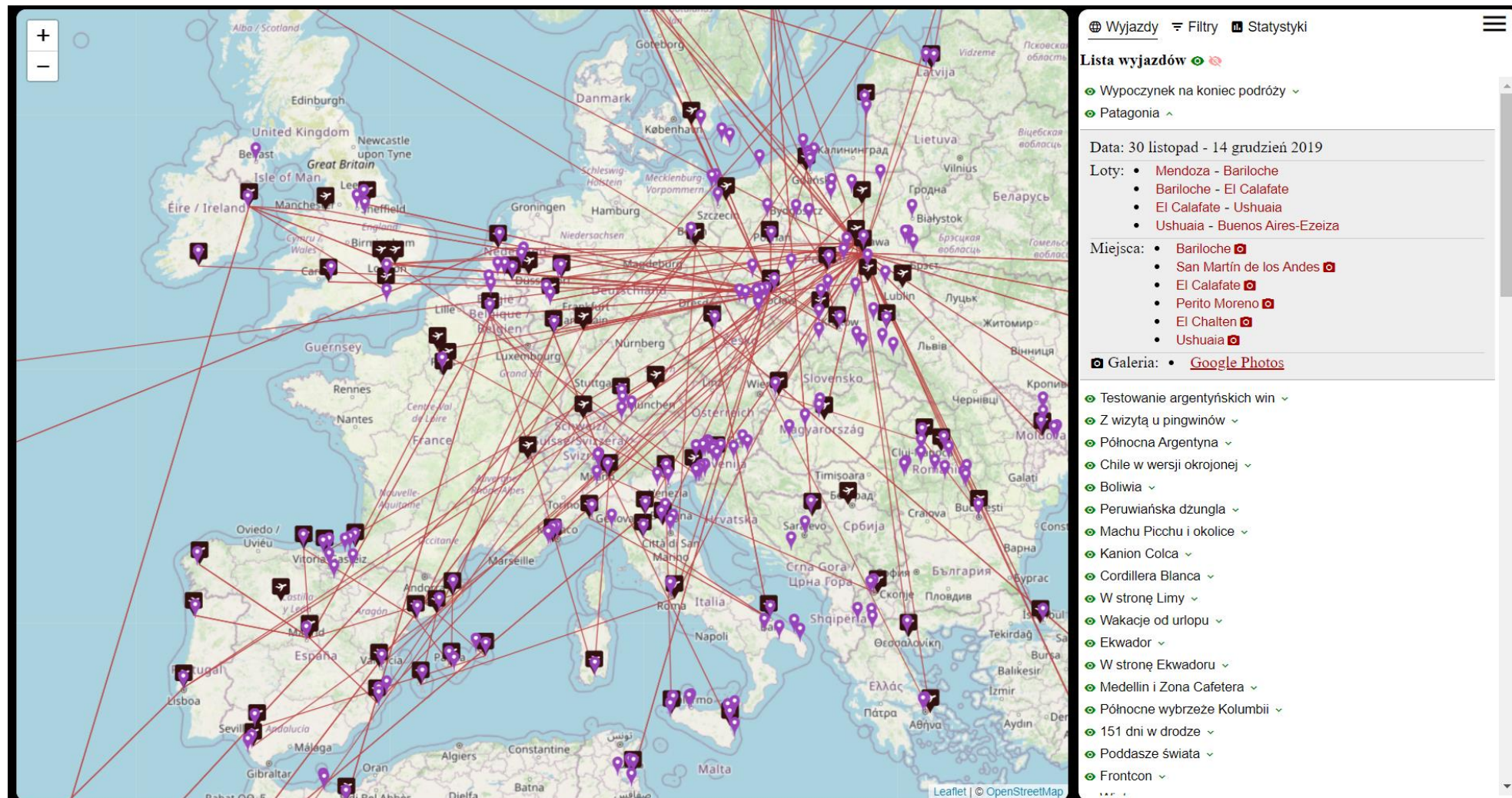
f [/HLTechCentre](#)

🐙 [/HLTech](#)

in [/company/hltechcentre](#)



# PIERWSZE KROKI W REACT



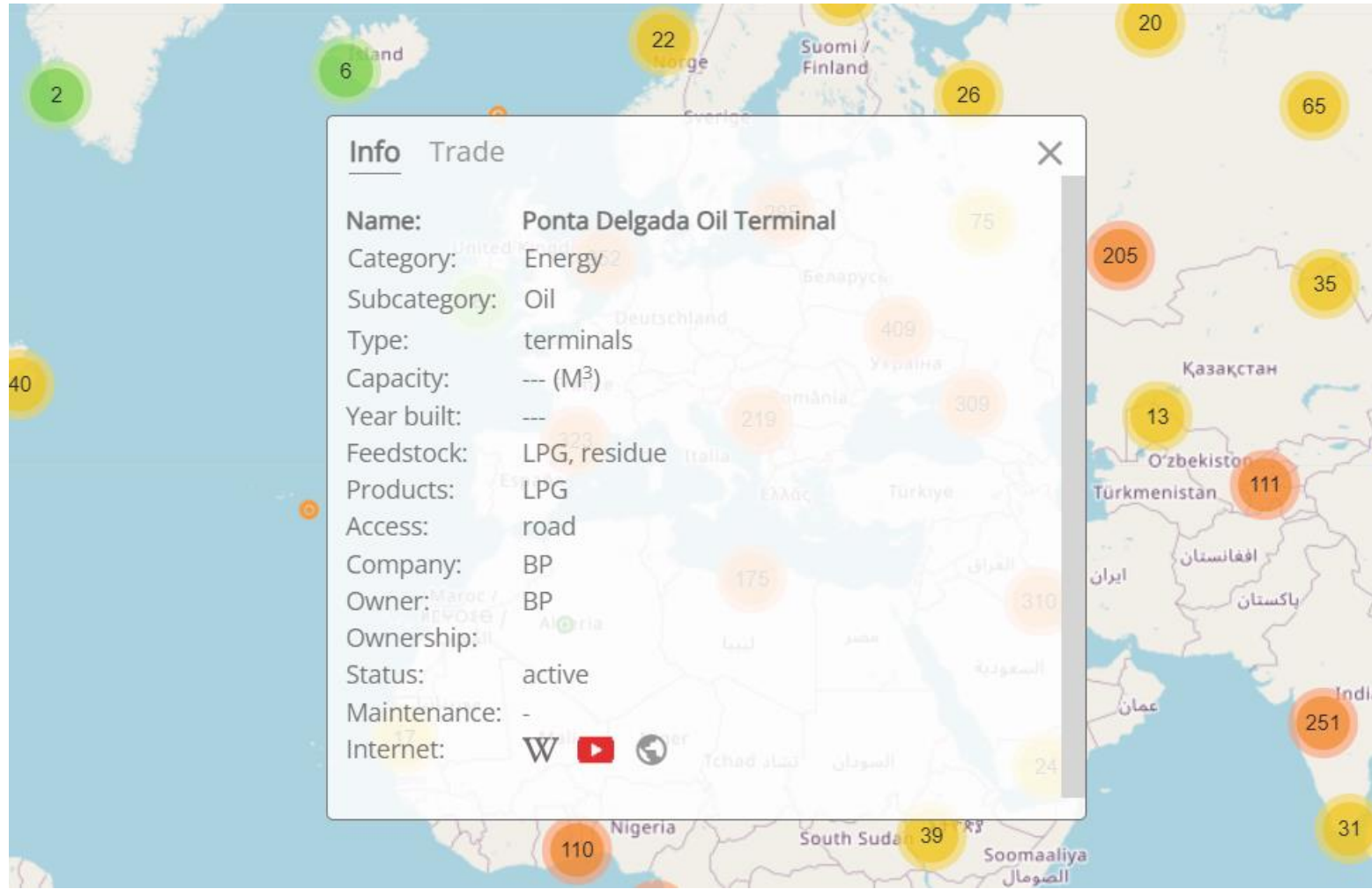
# PIERWSZE KROKI W REACT

The screenshot shows the Chrome DevTools Network tab. The top filter bar is set to 'Fetch/XHR'. The timeline at the top shows a request starting around 200ms and ending around 800ms. The list of requests on the left includes 'messaging', 'airlines.json', 'aircrafts.json', 'countries.json', 'airports.json', 'places.json' (which is selected), and 'trips.json'. The 'Response' tab for the selected request shows a JSON object with the following structure:

```
1771 },
1772 "pampeluna": {
1773   "lat": 42.812475,
1774   "lon": -1.645258,
1775   "name": "Pampeluna",
1776   "country": "hiszpania",
1777   "guides": [
1778     {
1779       "link": "http://tambylstwo.blogspot.com/2016/05/spostrzezenia-wasne-|
1780     }
1781   ],
1782   "galleries": [
1783     {
1784       "link": "https://goo.gl/photos/rt6hvM4SQcCZHaBd8",
1785       "tittle": "Pampeluna"
1786     }
1787   ]
1788 },
1789 "logrono": {
1790   "lat": 42.465456,
1791   "lon": -2.440997,
1792   "name": "Logroño",
1793   "country": "hiszpania",
1794   "guides": [
1795     {
1796       "link": "http://tambylstwo.blogspot.com/2016/05/spostrzezenia-wasne-|
1797     }
1798   ],
```



# PIERWSZY PRACA Z REACT



# MOCKOWANIE API

Repozytorium z  
przykładami

# REPOZYTORIUM Z PRZYKŁADAMI



 [LukaszNowakPL/talked-about-this](https://github.com/LukaszNowakPL/talked-about-this)

# TESTOWA APLIKACJA



# MOCKOWANIE API

## Ćw. 1: Użycie statycznych plików

# MOCKOWANIE API

## Ćw. 2: In-memory db

# ĆW. 2: IN-MEMORY DB



## Mirage JS

A client-side server to help you build, test and demo your JavaScript application.

👤 13 followers



<https://miragejs.com>



@miragejs

# ĆW. 2: IN-MEMORY DB

# ĆW. 2: IN-MEMORY DB

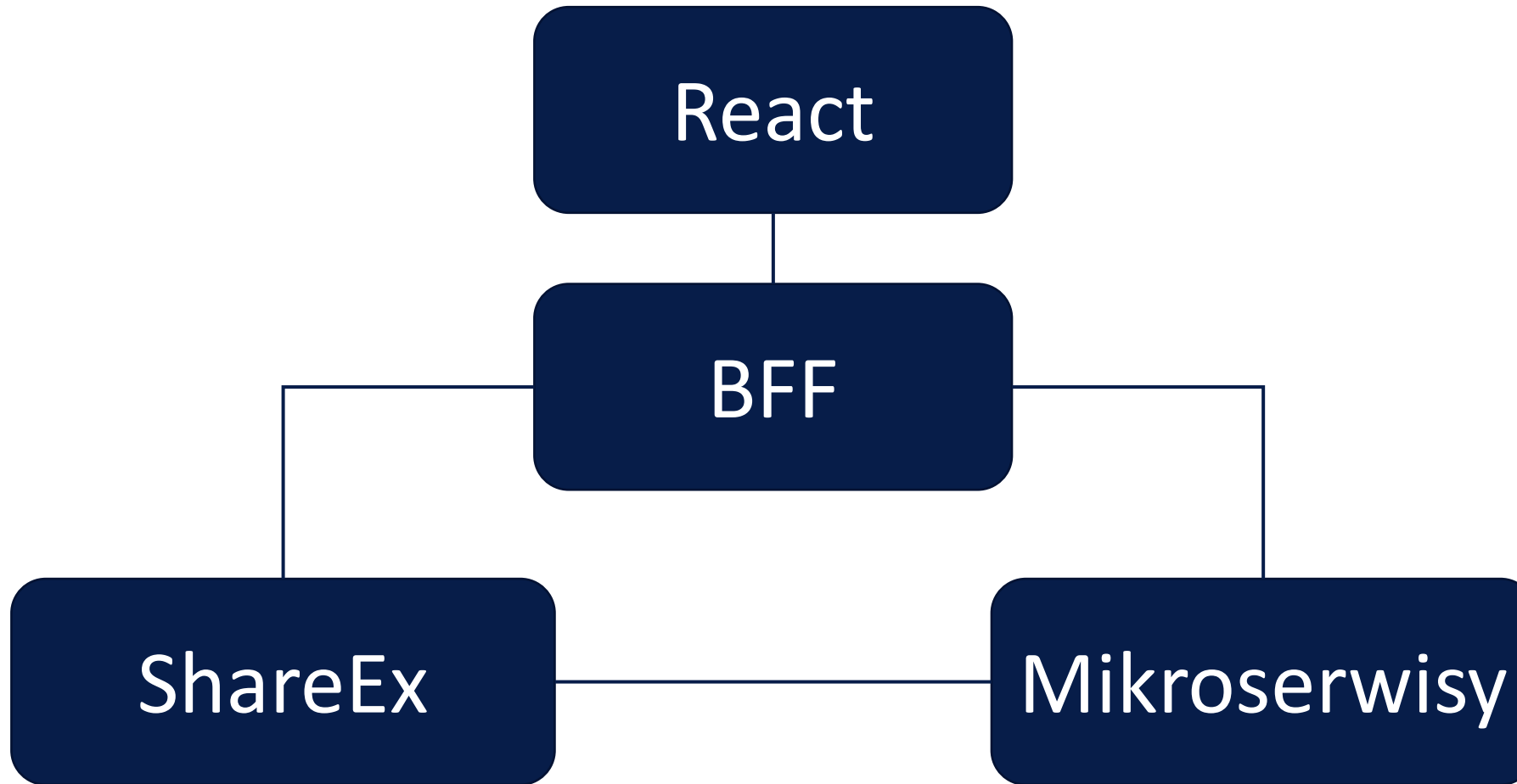
## Uwagi do rozwiązania:

- Możliwe RESTful api
- Modyfikacja danych w cyklu życia aplikacji
- Mirage JS i dane testowe dodane do bundle
- Brak integracji z backendem
- Brak komunikacji z api

# MOCKOWANIE API

## Ćw. 3: Integracja serwisów

# MOCKOWANIE API





# ĆW. 3: INTEGRACJA SERWISÓW

## Założenia projektu:

- npm run all-components
- Frontend: <http://localhost:3000/>
- Backend: <http://localhost:4000/>

# ĆW. 3: INTEGRACJA SERWISÓW

## JSON Server

---

Get a full fake REST API with **zero coding** in **less than 30 seconds** (seriously)

Created with <3 for front-end developers who need a quick back-end for prototyping and mocking.

- [Egghead.io free video tutorial - Creating demo APIs with json-server](#)
- [JSONPlaceholder - Live running version](#)
- [My JSON Server](#) - no installation required, use your own data

# ĆW. 3: INTEGRACJA SERWISÓW

# ĆW. 3: INTEGRACJA SERWISÓW

## Uwagi do rozwiązania:

- Ultraproste i realistyczne mockowanie backendu
- Przydatne w architekturze rozproszonej z różnymi poziomami dostępności

# MOCKOWANIE API

## Ćw. 4: Testy integracyjne z mockaniem warstwy api

# PATTERN COMPONENT > HOOK > API

`<NewAirportForm />`

Component

`useAirportAddition()`

Hook

`postAirport()`

Api

# ĆW. 4: TESTY INTEGRACYJNE Z MOCKOWANIEM API

## Założenia testów:

- Testy w Jest / React Testing Library
- Integracja z użytkownikiem i backendem
- Testowanie całych widoków
- Wrappowanie testowanych widoków
- Mockowanie wartości api



# ĆW. 4: TESTY INTEGRACYJNE Z MOCKOWANIEM API

## Uwagi do rozwiązania:

- Bardzo szeroki zakres integracji w teście
- Można to zrobić lepiej

# MOCKOWANIE API

## Ćw. 5: Testy integracyjne z użyciem MSW

# ĆW. 5: TESTY INTEGRACYJNE Z UŻYCIEM MSW



## Mock Service Worker

---

Mock Service Worker (MSW) is a seamless REST/GraphQL API mocking library for browser and Node.js.

LATEST

**V1.2.2**

DOWNLOADS

**10M/MONTH**

CHAT

**ONLINE**

# ĆW. 5: TESTY INTEGRACYJNE Z UŻYCIEM MSW

## Założenia projektu:

- Komunikacja z api handlowana przez MSW
- Handlowanie tylko spodziewanych calli

# ĆW. 5: TESTY INTEGRACYJNE Z UŻYCIEM MSW

## Uwagi do rozwiązania:

- Hiper bezpieczne testowanie sposobu komunikacji
- Niezależność od szczegółów implementacji
- Jest nie wyrabia na testach dużych komponentów

# PLAYWRIGHT COMPONENT TESTING

🏠 > Guides > Experimental: components

## Experimental: components

Playwright Test can now test your components.



# MOCKOWANIE API

## Ćw. 6: Testy funkcjonalne z użyciem Playwright



# ĆW. 6: TESTY FUNKCJONALNE Z UŻYCIEM PLAYWRIGHT



npm v1.35.1 chromium 115.0.5790.40 firefox 114.0.2 webkit 16.4

[Documentation](#) | [API reference](#)

Playwright is a framework for Web Testing and Automation. It allows testing [Chromium](#), [Firefox](#) and [WebKit](#) with a single API. Playwright is built to enable cross-browser web automation that is **ever-green**, **capable**, **reliable** and **fast**.

	Linux	macOS	Windows
Chromium 115.0.5790.40	✓	✓	✓
WebKit 16.4	✓	✓	✓
Firefox 114.0.2	✓	✓	✓

# ĆW. 6: TESTY FUNKCJONALNE Z UŻYCIEM PLAYWRIGHT

## Założenia projektu:

- Testowanie produkcyjnych bundli
- Testowanie user journey
- Mockowanie api przez playwright

# ĆW. 6: TESTY FUNKCJONALNE Z UŻYCIEM PLAYWRIGHT



Request mocking for Puppeteer and Playwright

npm v1.3.1 Node.js CI passing

Mockiavelli is HTTP request mocking library for [Puppeteer](#) and [Playwright](#). It was created to enable effective testing of Single Page Apps in isolation and independently from API services.

# MOCKOWANIE API

## Podsumowanie

# MOCKOWANIE API

## Podsumowanie:

- Sukces internetu dzięki komunikacji
- Mockowanie podczas developmentu
- Dokładne testy sposobu komunikacji z backendem

# MOCKOWANIE API

Q & A

# MOCKOWANIE API

Praktyczne ćwiczenia przed pierwszą randką z backendem.

Meet.js, Wrocław. 29.06.2023

Łukasz Nowak, HL Tech