AI1 **LAB D**	Hubkiewicz Jakub Album 46525 Grupa 2	**Wersja 1**

REST API CLIENT

SPIS TREŚCI

CEL ZAJĘĆ

Celem głównym zajęć jest zdobycie następujących umiejętności:

- pobieranie danych z zewnętrznych zasobów za pomocą REST API
- zdobywanie wiedzy na temat zewnętrznych API za pomocą dokumentacji typu Swagger
- wysyłanie asynchronicznych żądań z wykorzystaniem XMLHttpRequest i Fetch API

W praktycznym wymiarze uczestnicy stworzą dynamiczną stronę HTML pozwalającą na wyświetlanie bieżącej informacji pogodowej oraz prognoz dla zadanej przez użytkownika miejscowości.

ROZPOCZĘCIE

Rozpoczęcie zajęć. Powtórzenie wykonywania połączeń synchronicznych i asynchronicznych z poziomu JS na stornie.

Wejściówka?

UWAGA

Ten dokument aktywnie wykorzystuje niestandardowe właściwości. Podobnie jak w LAB A wejdź do `Plik` -> `Informacje` -> `Właściwości` -> `Właściwości zaawansowane` -> `Niestandardowe` i zaktualizuj pola. Następnie uruchom ten dokument ponownie lub `Ctrl+A` -> `F9`.

Wymagania

W ramach LAB D przygotowane powinny zostać:

- pojedyncza strona HTML ze skryptem ładowanym z zewnętrznego pliku JS
- pole tekstowe (input typu „text”) do wprowadzania adresu
- przycisk „Pogoda”, po kliknięciu którego wykonywane jest zapytanie asynchroniczne:
    - do API Current Weather: https://openweathermap.org/current za pomocą XMLHttpRequest
    - do API 5 day forecast: https://openweathermap.org/forecast5 za pomocą Fetch API
- obsługa zwrotki z obu API – wypisanie pogody bieżącej oraz prognoz poniżej pola wyszukiwania.

Wygeneruj własny lub wykorzystaj gotowy klucz do API: 7ded80d91f2b280ec979100cc8bbba94

W przypadku blokady można posiłkować się filmem: https://www.youtube.com/watch?v=WoKp2qDFxKk jednakże spróbuj rozwiązać ten problem samodzielnie!

Prowadzący omówi powyższe wymagania. Upewnij się, czy wszystko rozumiesz.

Tu umieść swoje notatki:

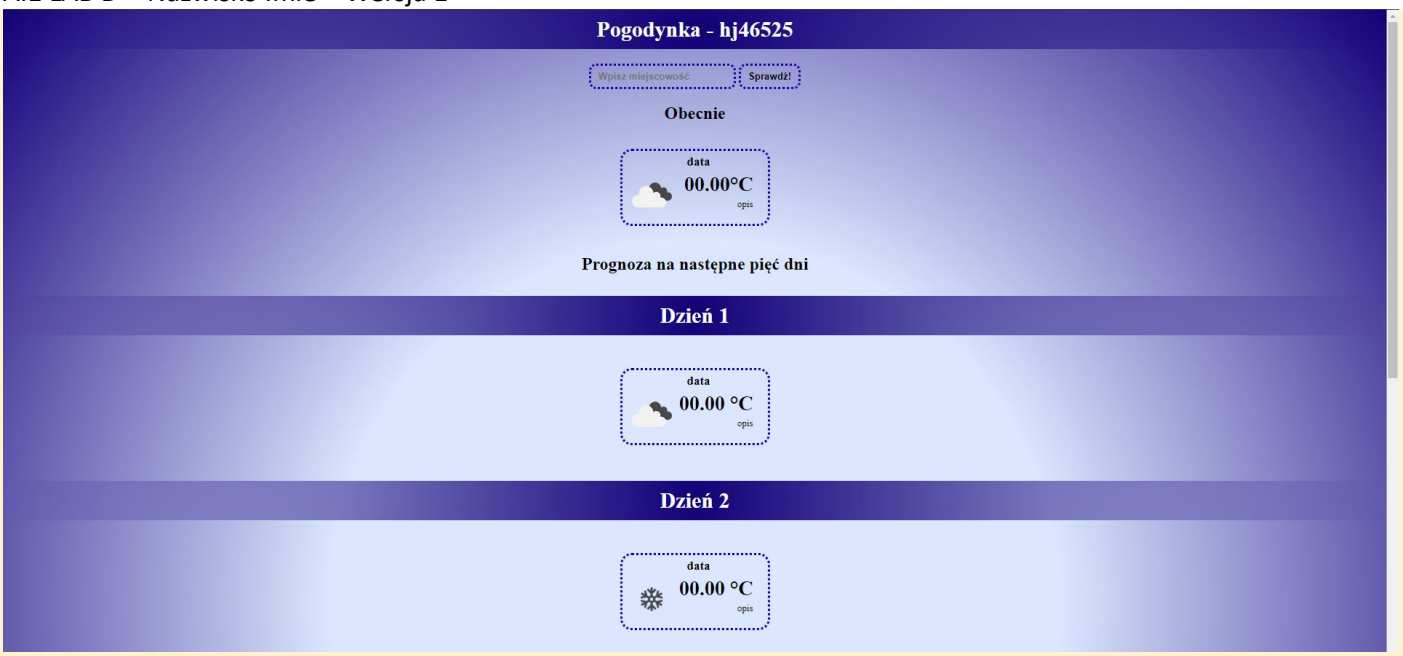
Badanie API

Poświęć kilka minut na wykonanie przykładowych zapytań do API z poziomu pasku adresu przeglądarki. Podaj wymagane parametry dla osiągnięcia różnych wyników. Zbadaj odpowiedzi API, aby uzyskać pełen obraz wymagań i możliwości API.

Implementacja

Tradycyjnie implementację należy zacząć od zbudowania w HTML + CSS wszystkich wymaganych elementów / placeholderów na te elementy. Następnie krok po kroku należy implementować poszczególne zachowania.

Wstaw zrzut ekranu zawierającego stronę ze wszystkimi elementami, tj. pole tekstowe, przycisk, miejsce do wyświetlenia pogody i prognozy:



Punkty:	0	1

Wstaw zrzut ekranu kodu odpowiedzialnego za wysyłanie żądania do current za pomocą XMLHttpRequest:

```
getCurrentWeather(cityQuery) {
  let url = this.getCurrentWeatherLink.replace("{cityQuery}", cityQuery);
  let req = new XMLHttpRequest();
  req.open("GET", url, true);
  req.addEventListener("load", () => {
    console.log(JSON.parse(req.responseText));
    this.currentWeather = JSON.parse(req.responseText);
    this.drawWeather(cityQuery);
  });
  req.send();
}
```

Wstaw zrzut ekranu pokazujący otrzymaną odpowiedź za pomocą `console.log()` w przeglądarce.

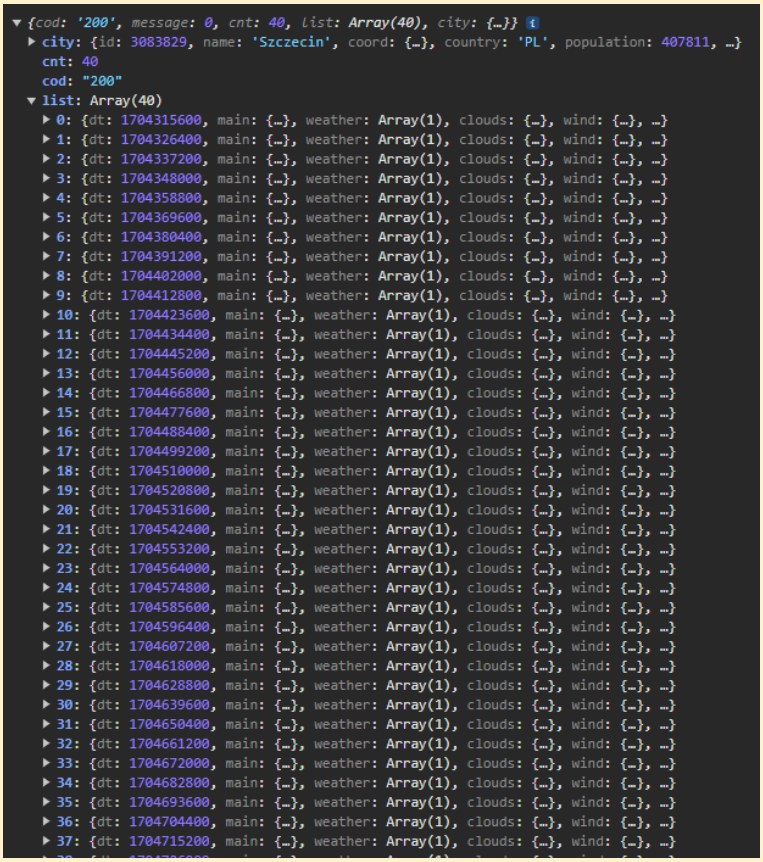
```
▼ Object i
    base: "stations"
  ▶ clouds: {all: 100}
    cod: 200
  ▶ coord: {lon: 14.553, lat: 53.4289}
    dt: 1704312393
    id: 3083829
  ▶ main: {temp: 8.41, feels_like: 6.26, temp_min: 5.61, temp_max: 9.12, pressure: 987, …}
    name: "Szczecin"
  ▶ sys: {type: 2, id: 19799, country: 'PL', sunrise: 1704266231, sunset: 1704293664}
    timezone: 3600
    visibility: 10000
  ▶ weather: [{…}]
  ▶ wind: {speed: 3.58, deg: 160}
  ▶ [[Prototype]]: Object
```

Punkty:	0	1

Wstaw zrzut ekranu kodu odpowiedzialnego za wysyłanie żądania do forecast za pomocą Fetch:

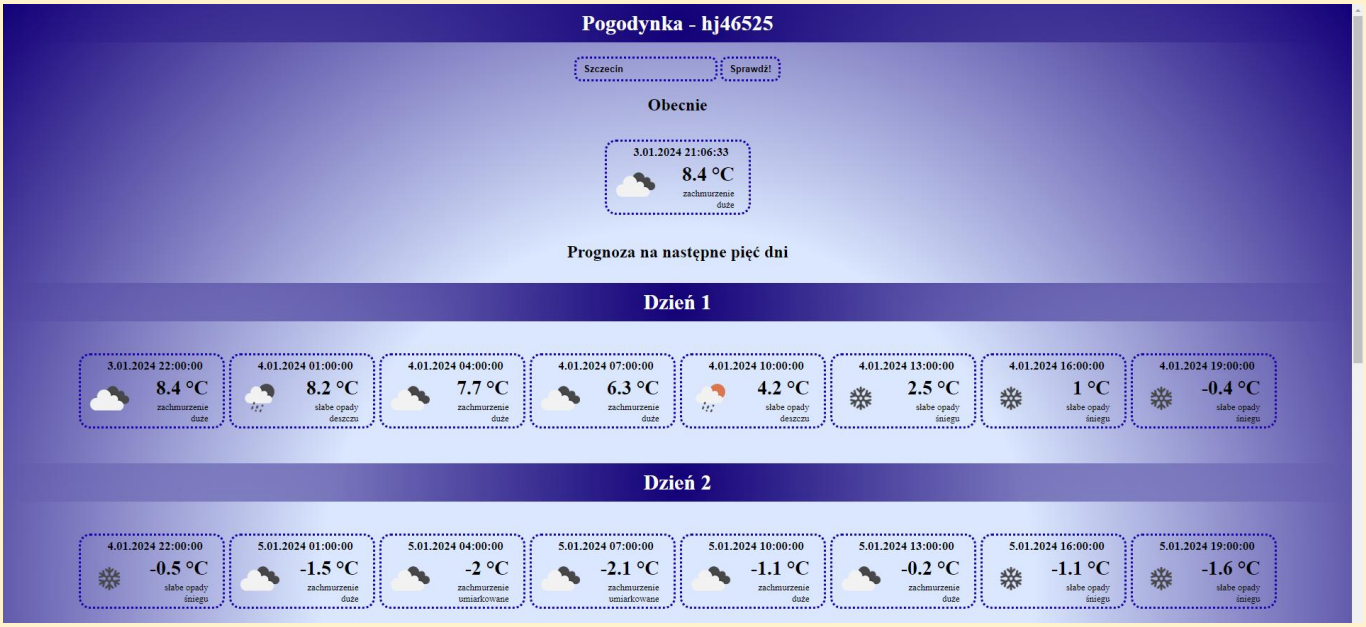
```
getForecast(cityQuery) {
  let url = this.forecastLink.replace("{cityQuery}", cityQuery);
  fetch(url)
    .then((response) => {
      return response.json();
    })
    .then((data) => {
      console.log(data);
      this.forecast = data.list;
      this.drawWeather(cityQuery);
    });
}
```

Wstaw zrzut ekranu pokazujący otrzymaną odpowiedź za pomocą `console.log()` w przeglądarce.



Punkty:	0	1

Wstaw zrzut ekranu przedstawiającego wizualizację prognoz pogody:



Upewnij się, że widoczne są pasek wyszukiwania ze wskazaną miejscowością, a także zarówno pogoda bieżąca jak i prognozy pogody.

Punkty:	0	1

COMMIT PROJEKTU DO GIT

Zacommituj i pushnij swoje rozwiązanie do repozytorium GIT.

Upewnij się, czy wszystko dobrze się wysłało. Jeśli tak, to z poziomu przeglądarki utwórz branch o nazwie `lab-c` na podstawie głównej gałęzi kodu.

Podaj link do brancha `lab-d` w swoim repozytorium:
https://github.com/huuuuubi/AI1-LA-grN2-Hubkiewicz-Jakub/tree/lab-d/LD

PODSUMOWANIE

W kilku zdaniach podsumuj zdobyte podczas tego laboratorium umiejętności.
Nauczyłem się podstawowej prezentacji danych uzyskiwanych z zewnętrznego systemu za pomocą API.

Zweryfikuj kompletność sprawozdania. Utwórz PDF i wyślij w terminie.