Studentų 50 – 213
Kompiuterių katedra

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

Informatikos inžinerijos studijų programa

Inžinerinis projektas

T120B172 Sistemų integracijos technologijos

Sistemų API technologijos

ATLIKO:

Dominykas Adomaitis _____ **IFB-8**
(Vardas Pavardė) (Parašas) (Grupė)

DĖSTYTOJAS:

Algirdas Dobrovolskis _____
(Vardas Pavardė) (Parašas)

DARBAS ATIDUOTAS:

08 d. 12 mėn. 2021

KAUNAS 2021

Sistemų API technologijos

Užduotis

Darbo tikslas yra sugalvoti ir sukurti taikymą panaudojant REST API technologijas. Sugalvoti ir sukurti tik klientinės dalies taikymą (maksimalus galimas gauti pažymys yra ribojamas iki 8). Būtina panaudoti bent 3 skirtingus esamus REST API.

Sistemų API technologijos sprendimo architektūra/aprašymas

Panaudoti API:

- Geolocation API https://developer.mozilla.org/en-US/docs/Web/API/Geolocation_API
- GeoDB Cities API https://rapidapi.com/wirefreethought/api/geodb-cities/details
- Current Weather Data API https://openweathermap.org/current
- Google Search API https://rapidapi.com/apigeek/api/google-search3/details

API tai aplikacijų programavimo sąsaja (angl. Application Programming Interface, API) – tai sąsaja, kurią suteikia kompiuterinė sistema, biblioteka ar programa tam, kad programuotojas per kitą programą galėtų pasiekti jos funkcionalumą ar apsikeistų su ja duomenimis.

Geolocation API: Naudoju Geolocation.getCurrentPosition() Užklausa gražina esama įrenginio vieta koordinatėmis.

CeoDB Cities naudoju užklausa:

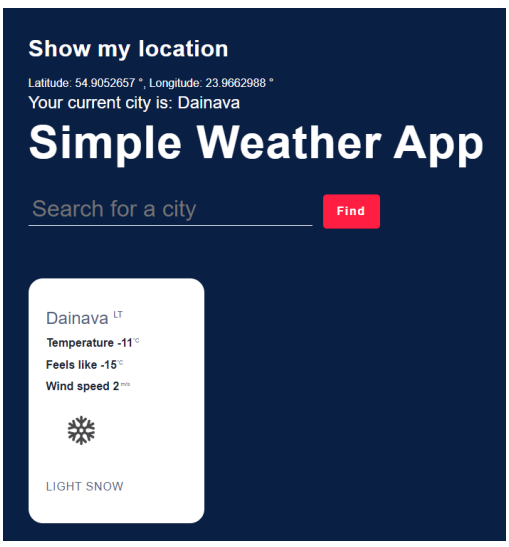
```
https://wft-geo-
db.p.rapidapi.com/v1/geo/locations/$%7Blatitude%7D+$%7Blongitude%7D/nea
rbyCities?radius=20&limit=2&minPopulation=2000&distanceUnit=KM
```

Ši užklausa pagal duotas koordinates 20 kilometrų spindulių randa miestą su daugiau nei 2000 gyventojų. Taip randamas miestas kuriame yra įrenginys.

Current Weather Data API naudoju užklausa:

```
https://api.openweathermap.org/data/2.5/weather?q=${inputVal}&appid=${a
piKey}&units=metric
```

Užklausoje nurodomos miestas. Gaunama dabartiniu momentu esantys orai tame mieste.

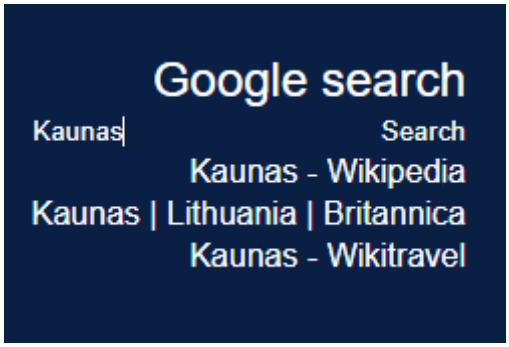


pav. 1 Gauta orų prognozė

Google Search API naudoju užklausa:

```
https://google-search3.p.rapidapi.com/api/v1/search/q=${keyWord}
```

Įvedus raktinį žodį gaunami pirmi trys paieškos rezultatai su nuorodomis juos.



pav. 2 Gauti paieškos rezultatai

Sistemų API technologijos sprendimo programinis kodas

Index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <link rel="stylesheet" href="style.css" />
    <title>Wheather</title>
  </head>
  <section class ="page-footer">
    <div class="container">
      <h1>Google search</h1>
        <input class="searchWord" id ="searchWord" placeholder="Google search" autofocus />
        <button id="googleSearch" >Search</button>
        <br>
        <a id="titlelink1" target="_blank"></a>
        <br>
        <a id="titlelink2" target="_blank"></a>
        <br>
        <a id="titlelink3" target="_blank"></a>
        <br>
    </div>
  </section>

  <section class="top-banner">
    <div class="container">
      <button class="locationHeading" id="find-me" >
        Show my location
      </button>
      <p class="locationHeading" id="status"></p>
      <a id="map-link" target="_blank"></a>
      <h1 class="yourlocation" id="gps"></h1>
    </div>
  </section>

  <section class="top-banner">
    <div class="container">
      <h1 class="heading">Simple Weather App</h1>
      <form>
```

```
        <input type="text" placeholder="Search for a city" autofocus />
        <button type="submit" id="weatcherSearch">Find</button>
        <span class="msg"></span>
      </form>
    </div>
  </section>

  <section class="ajax-section">
    <div class="container">
      <ul class="cities"></ul>
    </div>
  </section>

</div>
  <script src="script.js"></script>
</html>
```

Style.css

```
/* RESET STYLES
---------------------------------------------------- */
:root {
  --bg_main: #0a1f44;
  --text_light: #fff;
  --text_med: #53627c;
  --text_dark: #1e2432;
  --red: #ff1e42;
  --darkred: #c3112d;
  --orange: #ff8c00;
}

* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-weight: normal;
}

a {
  color: inherit;
  text-decoration: none;
}

button {
  cursor: pointer;
}

input {
  -webkit-appearance: none;
}

button,
input {
  border: none;
  background: none;
  outline: none;
  color: inherit;
}

img {
  display: block;
  max-width: 100%;
  height: auto;
}

ul {
  list-style: none;
}
```

```
body {
  font: 1rem/1.3 "Roboto", sans-serif;
  background: var(--bg_main);
  color: var(--text_dark);
  padding: 70px;
}

button:hover,
a:hover {
  color: rgb(109, 108, 108);
}

.container {
  width: 100%;
  max-width: 1200px;
  margin: 0 auto;
  padding: 0 15px;
}

/* SECTION #1
------------------------------------------------- */
.top-banner {
  color: var(--text_light);
}

.heading {
  font-weight: bold;
  font-size: 4rem;
  letter-spacing: 0.02em;
  padding: 0 0 30px 0;
}

.locationHeading {
  font-weight: bold;
  font-size: 2rem;
  letter-spacing: 0.02em;
  padding: 0 0 10px 0;
  color: rgb(255, 255, 255);
}
.top-banner form {
  position: relative;
  display: flex;
  align-items: center;
}

.top-banner form input {
  font-size: 2rem;
  height: 40px;
  padding: 5px 5px 10px;
  border-bottom: 1px solid;
}

.top-banner form button {
  font-size: 1rem;
  font-weight: bold;
  letter-spacing: 0.1em;
  padding: 15px 20px;
  margin-left: 15px;
  border-radius: 5px;
  background: var(--red);
  transition: background 0.3s ease-in-out;
}

.top-banner form button:hover {
  background: var(--darkred);
}

.top-banner form .msg {
  position: absolute;
  bottom: -40px;
  left: 0;
  max-width: 450px;
  min-height: 40px;
```

```
}

.top-banner form .yourlocation {
  position: absolute;
  bottom: -40px;
  left: 0;
  max-width: 450px;
  min-height: 40px;
}

/* SECTION #2
------------------------------------------------- */
.ajax-section {
  margin: 70px 0 20px;
}

.ajax-section .cities {
  display: grid;
  grid-gap: 35px 60px;
  grid-template-columns: repeat(4, 1fr);
}

.ajax-section .city {
  position: relative;
  padding: 40px 10%;
  border-radius: 20px;
  background: var(--text_light);
  color: var(--text_med);
}

.ajax-section figcaption {
  margin-top: 20px;
  text-transform: uppercase;
  letter-spacing: 0.05em;
}

.ajax-section .city-temp {
  font-size: 1rem;
  font-weight: bold;
  margin-top: 10px;
  color: var(--text_dark);
}

.ajax-section .city sup {
  font-size: 0.5em;
}

.page-footer {
  text-align: right;
  font-size: 1rem;
  color: var(--text_light);
  margin-top: 40px;
}

.page-footer span {
  color: var(--red);
}
```

Script.js

```
//document.querySelector(".googleSearch").addEventListener("click", googleSearch);
document.querySelector("#googleSearch").addEventListener("click", googleSearch);
function googleSearch() {
  var keyWord = document.getElementById("searchWord").value;
  const titlelink1 = document.querySelector("#titlelink1");
  const titlelink2 = document.querySelector("#titlelink2");
  const titlelink3 = document.querySelector("#titlelink3");
  //var keyWord = document.querySelector(".searchWord").value;
  //var keyWord = "asus";
  console.log(keyWord);
  //Google search API
```

```
  const googleurl = `https://google-search3.p.rapidapi.com/api/v1/search/q=${keyWord}`;
  fetch(googleurl, {
    method: "GET",
    headers: {
      "x-user-agent": "desktop",
      "x-proxy-location": "US",
      "x-rapidapi-host": "google-search3.p.rapidapi.com",
      "x-rapidapi-key": "3b18eaeafamsh446d7d16459bedcp1216f5jsn8f88c4df3a35",
    },
  })
    .then((response) => response.json())
    .then((data) => {
      console.log(data);
      //console.log(data.results[0].description);
      //var description = document.getElementById("description").textContent;
      //document.getElementById("description").innerHTML = data.results[0].description;
      //document.getElementById("title").innerHTML = data.results[0].title;
      //document.getElementById("link").innerHTML = data.results[0].link;
      titlelink1.href = data.results[0].link;
      titlelink1.textContent = data.results[0].title;
      titlelink2.href = data.results[1].link;
      titlelink2.textContent = data.results[1].title;
      titlelink3.href = data.results[2].link;
      titlelink3.textContent = data.results[2].title;
    })
    .catch((err) => {
      console.error(err);
    });
}

//Coordinates finder API
var city = "";
document.querySelector("#find-me").addEventListener("click", geoFindMe);
function geoFindMe() {
  const status = document.querySelector("#status");
  const mapLink = document.querySelector("#map-link");
  const yourlocation = document.querySelector("#gps");
  mapLink.href = "";
  mapLink.textContent = "";
  yourlocation.textContent = "";

  function success(position) {
    const latitude = position.coords.latitude;
    const longitude = position.coords.longitude;
    //latitude = 54.77897039100206;
    //longitude = 24.63542203285122;

    status.textContent = "";
    mapLink.href = `https://www.openstreetmap.org/#map=18/${latitude}/${longitude}`;
    mapLink.textContent = `Latitude: ${latitude} °, Longitude: ${longitude} °`;
    var request = `https://wft-geo-
db.p.rapidapi.com/v1/geo/locations/${latitude}+${longitude}/nearbyCities?radius=20&limit=2&min
Population=2000&distanceUnit=KM`;
    //console.log(request);
    cityFinder(request);
  }

  function error() {
    status.textContent = "Unable to retrieve your location";
  }

  if (!navigator.geolocation) {
    status.textContent = "Geolocation is not supported by your browser";
  } else {
    status.textContent = "Locating…";
    navigator.geolocation.getCurrentPosition(success, error);
  }

  //City finder API
  function cityFinder(request) {
    fetch(request, {
      method: "GET",
      headers: {
        "x-rapidapi-host": "wft-geo-db.p.rapidapi.com",
```

```
        "x-rapidapi-key": "3b18eaeafamsh446d7d16459bedcp1216f5jsn8f88c4df3a35",
      },
    })
      .then((response) => response.json())
      .then((data) => {
        console.log(data);
        console.log(data.data[0].city);
        console.log(data.data[0].latitude);
        console.log(data.data[0].longitude);
        city = data.data[0].city;
        yourlocation.textContent = `Your current city is: ${city}`;
        //console.log(data.data[0].population);
      })
      .catch((error) => {
        error();
      });
  }
}

const form = document.querySelector(".top-banner form");
const input = document.querySelector(".top-banner input");
const msg = document.querySelector(".top-banner .msg");
const list = document.querySelector(".ajax-section .cities");

const apiKey = "593f569c313b704098e1f53ce0d4b627";
let i = 0;
form.addEventListener("submit", (e) => {
  e.preventDefault();

  let inputVal;
  if (i == 0) {
    inputVal = city;
    i++;
  } else inputVal = input.value;
  //check if there's already a city
  const listItems = list.querySelectorAll(".ajax-section .city");
  const listItemsArray = Array.from(listItems);

  if (listItemsArray.length > 0) {
    const filteredArray = listItemsArray.filter((el) => {
      let content = "";
      content = el.querySelector(".city-name span").textContent.toLowerCase();
      return content == inputVal.toLowerCase();
    });

    if (filteredArray.length > 0) {
      msg.textContent = `You already know the weather for ${
        filteredArray[0].querySelector(".city-name span").textContent
      }`;
      form.reset();
      input.focus();
      return;
    }
  }

  //Wheather API
  const url =
`https://api.openweathermap.org/data/2.5/weather?q=${inputVal}&appid=${apiKey}&units=metric`;

  fetch(url)
    .then((response) => response.json())
    .then((data) => {
      const { main, name, sys, weather, wind } = data;
      const icon = `https://openweathermap.org/img/wn/${weather[0]["icon"]}@2x.png`;

      const li = document.createElement("li");
      li.classList.add("city");
      const markup = `
        <h2 class="city-name" data-name="${name},${sys.country}">
          <span>${name}</span>
          <sup>${sys.country}</sup>
        </h2>
        <div class="city-temp">Temperature ${Math.round(
          main.temp
```

```
        )}<sup>°C</sup></div>
        <div class="city-temp">Feels like ${Math.round(
          main.feels_like
        )}<sup>°C</sup></div>
        <div class="city-temp">Wind speed ${Math.round(
          wind.speed
        )}<sup> m/s</sup></div>
        <figure>
          <img class="city-icon" src="${icon}" alt="${
        weather[0]["description"]
      }">
          <figcaption>${weather[0]["description"]}</figcaption>
        </figure>
      `;
      li.innerHTML = markup;
      list.appendChild(li);
    })
    .catch(() => {
      msg.textContent = "Please search for a valid city";
    });

  msg.textContent = "";
  form.reset();
  input.focus();
});
```

Rezultatų apibendrinimas

Inžineriniame projekte pavyko pritaikyti REST API technologijos. Panaudoti 4 skirtingi API.