**Dokumentacja Aplikacji "Recipe Search Application"**

**Spis Treści**

1. [Opis Projektu](#opis-projektu)
2. [Wymagania Systemowe](#wymagania-systemowe)
3. [Instalacja](#instalacja)
4. [Struktura Projektu](#struktura-projektu)
5. [Opis Komponentów](#opis-komponent%C3%B3w)
6. [Użycie API](#u%C5%BCycie-api)
7. [Stylowanie](#stylowanie)
8. [Uruchomienie i Publikacja](#uruchomienie-i-publikacja)

**Opis Projektu**

"Recipe Search Application" to aplikacja typu Single Page Application (SPA) napisana w React, która umożliwia użytkownikom wyszukiwanie przepisów kulinarnych przy użyciu zewnętrznego API (Edamam Recipe API). Aplikacja jest responsywna i umożliwia przeglądanie wyników wyszukiwania w zależności od rozdzielczości ekranu.

**Wymagania Systemowe**

* Node.js (v14 lub wyższa)
* npm (v6 lub wyższa)
* Konto na GitHub
* Konto na Vercel

**Instalacja**

1. **Klonowanie repozytorium:**

bash

Skopiuj kod

git clone <URL\_repozytorium>

cd recipe-search

1. **Instalacja zależności:**

bash

Skopiuj kod

npm install

1. **Konfiguracja kluczy API:**
   * Zarejestruj się na [Edamam Recipe API](https://developer.edamam.com/edamam-recipe-api) i uzyskaj app\_id oraz app\_key.
   * Utwórz plik .env w katalogu głównym projektu i dodaj do niego:

makefile

Skopiuj kod

REACT\_APP\_EDAMAM\_APP\_ID=your\_app\_id

REACT\_APP\_EDAMAM\_APP\_KEY=your\_app\_key

**Struktura Projektu**

lua

Skopiuj kod

src/

|-- components/

| |-- Header.js

| |-- RecipeList.js

| |-- SearchBar.js

|-- App.js

|-- index.js

|-- App.css

**Opis Komponentów**

**Header.js**

Komponent wyświetlający pasek górny z tytułem aplikacji.

javascript

Skopiuj kod

import React from 'react';

const Header = () => {

return (

<div className="header-bar">

Recipe Search Application

</div>

);

}

export default Header;

**SearchBar.js**

Komponent zawierający pasek wyszukiwania oraz przycisk do zatwierdzania wyszukiwania.

javascript

Skopiuj kod

import React, { useState } from 'react';

const SearchBar = ({ onSearch }) => {

const [query, setQuery] = useState('');

const handleSearch = () => {

onSearch(query);

};

return (

<div className="search-container">

<input

type="text"

value={query}

onChange={(e) => setQuery(e.target.value)}

placeholder="Search for a recipe..."

onKeyPress={(e) => e.key === 'Enter' && handleSearch()}

className="search-input"

/>

<button onClick={handleSearch} className="search-button">Search</button>

</div>

);

};

export default SearchBar;

**RecipeList.js**

Komponent wyświetlający listę przepisów.

javascript

Skopiuj kod

import React from 'react';

const RecipeList = ({ recipes }) => {

return (

<div className="recipe-container">

{recipes.map((recipe, index) => (

<div key={index} className="recipe-card">

<h3>{recipe.recipe.label}</h3>

<img src={recipe.recipe.image} alt={recipe.recipe.label} />

<p>Calories: {Math.round(recipe.recipe.calories)}</p>

<p>Source: {recipe.recipe.source}</p>

</div>

))}

</div>

);

};

export default RecipeList;

**App.js**

Główna logika aplikacji, która zarządza stanem i integruje komponenty.

javascript

Skopiuj kod

import React, { useState } from 'react';

import axios from 'axios';

import Header from './components/Header';

import SearchBar from './components/SearchBar';

import RecipeList from './components/RecipeList';

import './App.css';

const App = () => {

const [recipes, setRecipes] = useState([]);

const fetchRecipes = async (query) => {

const response = await axios.get(`https://api.edamam.com/search?q=${query}&app\_id=${process.env.REACT\_APP\_EDAMAM\_APP\_ID}&app\_key=${process.env.REACT\_APP\_EDAMAM\_APP\_KEY}`);

setRecipes(response.data.hits);

};

return (

<div className="container">

<Header />

<SearchBar onSearch={fetchRecipes} />

<RecipeList recipes={recipes} />

</div>

);

};

export default App;

**Użycie API**

* API: [Edamam Recipe API](https://developer.edamam.com/edamam-recipe-api)
* Do poprawnego działania aplikacji, wymagane jest uzyskanie app\_id oraz app\_key, które należy umieścić w pliku .env.

**Stylowanie**

Plik CSS (App.css) zawiera style dla komponentów aplikacji:

css

Skopiuj kod

body {

font-family: Arial, sans-serif;

background-color: #e8f0fe;

margin: 0;

padding: 0;

}

.container {

padding: 20px;

text-align: center;

}

.header-bar {

background-color: #00796b;

color: white;

padding: 10px;

font-size: 24px;

}

.search-container {

margin: 20px 0;

}

.search-input {

padding: 10px;

font-size: 16px;

width: 300px;

margin-right: 10px;

border: 1px solid #ccc;

border-radius: 4px;

}

.search-button {

padding: 10px;

font-size: 16px;

background-color: #00796b;

color: white;

border: none;

border-radius: 4px;

cursor: pointer;

}

.search-button:hover {

background-color: #004d40;

}

.recipe-container {

display: flex;

flex-wrap: wrap;

justify-content: center;

}

.recipe-card {

border: 1px solid #ccc;

border-radius: 4px;

margin: 10px;

padding: 10px;

width: 300px;

background-color: white;

box-shadow: 0 2px 4px rgba(0,0,0,0.1);

transition: transform 0.2s;

}

.recipe-card:hover {

transform: translateY(-5px);

}

.recipe-card h3 {

font-size: 18px;

margin: 0 0 10px;

}

.recipe-card img {

width: 100%;

border-radius: 4px;

}

.recipe-card p {

margin: 5px 0;

}

**Uruchomienie i Publikacja**

1. **Uruchomienie lokalne:**

bash

Skopiuj kod

npm start

1. **Publikacja na GitHub:**
   * Utwórz nowe repozytorium na GitHub.
   * Skopiuj adres URL repozytorium.
   * Wykonaj poniższe komendy w katalogu projektu:

bash

Skopiuj kod

git init

git remote add origin <URL\_repozytorium>

git add .

git commit -m "Initial commit"

git push -u origin master

1. **Publikacja na Vercel:**
   * Połącz swoje konto GitHub.
   * Wybierz repozytorium i kliknij "Import".
   * Vercel automatycznie zbuduje i opublikuje Twoją aplikację.