Realtime Databse: Instrukcje + Zadania

DOKUMENTACJA

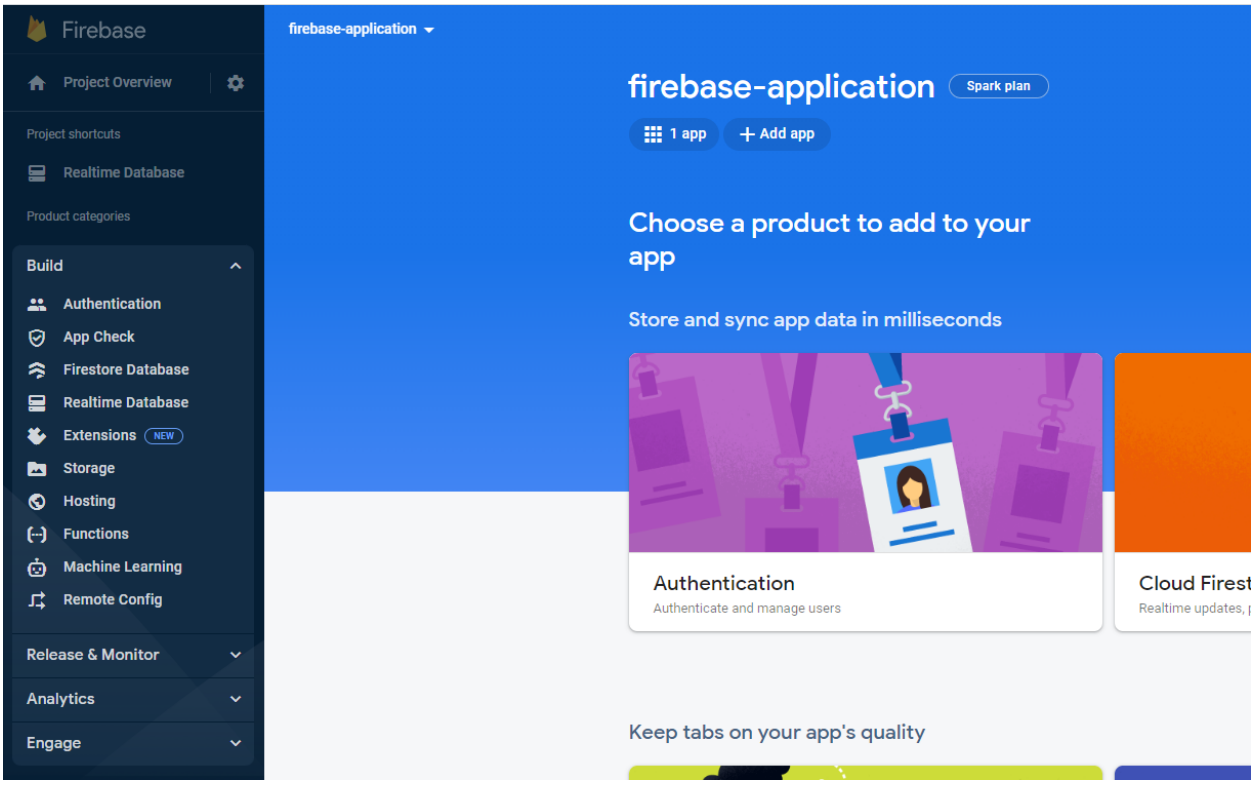
Firebase Realtime Database

Store and sync data with our NoSQL cloud database. Data is synced across all clients in realtime, and remains available when your app goes offline.

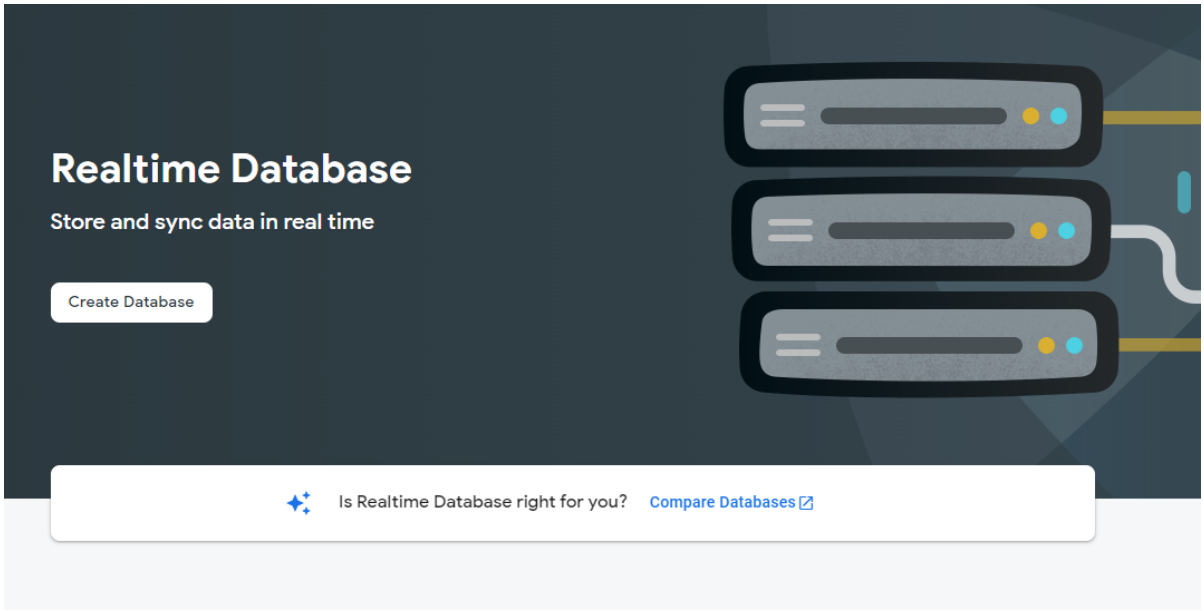
https://firebase.google.com/docs/database?hl=en

KONFIGURACJA MODUŁU

Zakładamy, że znajdujesz się w konsoli widoku projektu Firebase.



1. Uruchom moduł Realtime Database, znajdujący się w zakładce Build.
2. Utwórz instancję bazy danych dla Twojego projektu poprzez przycisk Create Database.



1. W pierwszym kroku wybierz region, który znajduje się najbliżej Ciebie (w docelowych aplikacjach serwer powinien znajdować się najbliżej Twoich potencjalnych użytkowników).
2. W następnym kroku wybierz opcję: “Start in **test mode**”. Pozwoli Ci to na wykonywanie dowolnych operacji na bazie danych przez najbliższy miesiąc. Jeżeli nie zaznaczysz tej opcji, wykonanie jakichkolwiek operacji nie będzie możliwe ze względu na zasady bezpieczeństwa.

3. Gotowe!

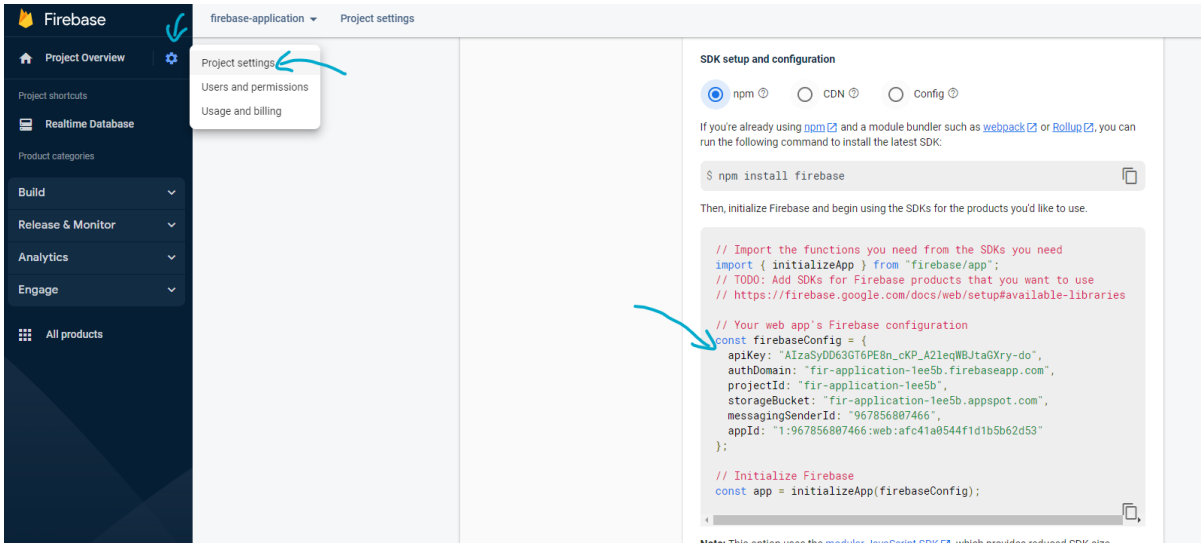
KODOWANIE - CRUD

1. W pliku `public/index.html` podłącz plik `app.js`
    a. Znajdzie się tam kod dot. obsługi operacji CRUD w Realtime Database.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Realtime Databse - CRUD</title>
  </head>

  <body>
    <!-- Dołączamy plik, w którym znajdzie się nasz kod
      dotyczący obsługi operacji CRUD w Realtime Database -->
    <script src="app.js" type="module"></script>
  </body>
</html>
```

2. Znalezienie wartości konfiguracyjnych do Twojego projektu.



3. W pliku `public/app.js` umieścimy logikę obsługi operacji CRUD w Realtime Database.
4. Inicjalizacja aplikacji wykorzystującej Firebase oraz zdefiniowanie konfiguracji.

```
// Import zależności pozwalającej na incjalizację aplikacji Firebasowej
import { initializeApp } from "https://www.gstatic.com/firebasejs/9.4.0/firebase-app.js";

// Wartości konfiguracyjne. Ważne, żeby znalazły się tu wartości dot. Twojego projektu!
// Gotowy obiekt firebaseConfig znajdziesz w widoku projektu w przeglądarce.
const firebaseConfig = {
  apiKey: "AIzaSyDD63GT6PE8n_cKP_A2leqWBJtaGXry-do",
  authDomain: "fir-application-1ee5b.firebaseapp.com",
  projectId: "fir-application-1ee5b",
  storageBucket: "fir-application-1ee5b.appspot.com",
  messagingSenderId: "967856807466",
  appId: "1:967856807466:web:afc41a0544f1d1b5b62d53"
};

// Inicjalizowanie aplikacji Firebasowej
const app = initializeApp(firebaseConfig);
```

5. Dodanie zależności dot. Realtime Databse.

```
import { initializeApp } from "https://www.gstatic.com/firebasejs/9.4.0/firebase-app.js";

// Import zależności dot. modułu Realtime Databse
import {
    getDatabase
} from "https://www.gstatic.com/firebasejs/9.8.1/firebase-database.js";

const firebaseConfig = {
```

```
  ...
};

const app = initializeApp(firebaseConfig);

// Uruchomienie modułu Realtime Database wewnątrz naszej aplikacji
const database = getDatabase();
```

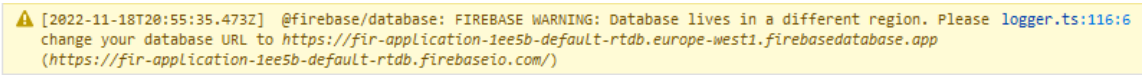
6. Zapisanie danych w bazie danych.

```
// Import zależności dot. modułu Realtime Databse
import {
    getDatabase,
    ref,
    set
} from "https://www.gstatic.com/firebasejs/9.8.1/firebase-database.js";

// W jaki sposób zapisać dane w bazie danych?
//  1. Musimy wskazać, gdzie chcemy zapisać dane. Taki wskaźnik składa się z dwóch informacji:
//      1.1. W jakiej bazie chcemy zapisać dane? Zmienna database
//      1.2. Musimy wskazać, gdzie chcemy zapisać nasze dane, np. users/uzytkownik1.
//           Trochę na zasadzie wskazanie folderu, w którym chcielibyśmy zapisać dane.

set(ref(database, 'users/uzytkownik1'), {
    username: 'pierwszy użytkownik',
    email: 'uzytkownik@mail.com',
    profile_picture: 'https://gravatar.com/avatar/754aff9a816fd9e0de1750c274e10e20?s=400&d=robohash&r=x'
});
```

7. Pojawi się błąd, który wynika z tego, że nie dodaliśmy do konfiguracji URL bazy danych.
    a. URL bazy danych możemy pobrać z komunikatu o błędzie.



    b. Uzupełnienie wartości konfiguracyjnych.

```
// Wartości konfiguracyjne. Ważne, żeby znalazły się tu wartości dot. Twojego projektu!
// Gotowy obiekt firebaseConfig znajdziesz w widoku projektu w przeglądarce.
const firebaseConfig = {
    apiKey: "AIzaSyDD63GT6PE8n_cKP_A2leqWBJtaGXry-do",
    authDomain: "fir-application-1ee5b.firebaseapp.com",
    projectId: "fir-application-1ee5b",
    storageBucket: "fir-application-1ee5b.appspot.com",
    messagingSenderId: "967856807466",
    appId: "1:967856807466:web:afc41a0544f1d1b5b62d53",
```

```
    databaseURL: "https://fir-application-1ee5b-default-rtdb.europe-west1.firebasedatabase.app"
};
```

8. Odczyt danych z bazy danych.

```
// Import zależności dot. modułu Realtime Databse
import {
    getDatabase,
    ref,
    set,
    get,
    onValue
} from "https://www.gstatic.com/firebasejs/9.8.1/firebase-database.js";

// W jaki sposób odczytać dane z bazy danych?

// Do odczytu danych możemy podejść na dwa sposoby:
//  1. Jednorazowe odczytanie danych z bazy.
//  2. Odczytanie danych z bazy oraz nasłuchiwanie na zmiany.

// Sposób 1: Jednorazowe odczytanie danych z bazy (get).
//  1. Musimy wskazać jakie dane chcemy odczytać. Taki wskaźnik składa się z dwóch informacji:
//      1.1. Z jakiej bazy chcemy odczytać dane? Zmienna database
//      1.2. Z jakiej ścieżki chcemy odczytać dane? Np. users/uzytkownik1
const userProperties = await get(ref(database, 'users/uzytkownik1'));
if (userProperties.exists()) {
    // Chcąc dostać się do właściwości wpisu z bazy danych musimy skorzystać z metody val()
    console.log(userProperties.val());
} else {
    console.log("User not found!");
}
// Przy ręcznej edycji takiego wpisu nie zauważymy zmian w konsoli w przeglądarce.

// Sposób 2: Odczytanie danych z bazy oraz nasłuchiwanie na zmiany (onValue).
//  1. Musimy wskazać jakie dane chcemy odczytać. Taki wskaźnik składa się z dwóch informacji:
//      1.1. Z jakiej bazy chcemy odczytać dane? Zmienna database
//      1.2. Z jakiej ścieżki chcemy odczytać dane? Np. users/uzytkownik1
onValue(ref(database, 'users/uzytkownik1'), (snapshot) => {
  const userProperties = snapshot.val();
  console.log(userProperties);
});
// Przy ręcznej edycji takiego wpisu POJAWIĄ SIĘ zmiany w konsoli w przeglądarce.
```

9. Aktualizowanie danych w bazie danych.

```
// Import zależności dot. modułu Realtime Databse
import {
    getDatabase,
    ref,
    set,
    get,
```

```
    onValue,
    update
} from "https://www.gstatic.com/firebasejs/9.8.1/firebase-database.js";

// W jaki sposób aktualizować dane w bazie danych?
//  1. Musimy wskazać, który wpis chcemy zmodfyikować. Taki wskaźnik składa się z dwóch informacji:
//      1.1. W jakiej bazie chcemy zmodyfikować dane? Zmienna database
//      1.2. Musimy wskazać, gdzie chcemy zmodyfikować nasze dane, np. users/uzytkownik1.

// Modyfikacji ulegną wyłącznie te pola, które zostaną wskazane w obiekcie.
// Pozostałe wartości nie zostaną zmienione.

// Jeżeli jakiegoś pola wcześniej nie było to zostanie ono utworzone.
update(ref(database, "users/uzytkownik1"), {
    username: "Zmieniona nazwa użytkownika!",
    dateOfBirth: "01-01-2000"
})
```

10. Usuwanie danych z bazy danych.

```
// Import zależności dot. modułu Realtime Databse
import {
    getDatabase,
    ref,
    set,
    get,
    onValue,
    update,
    remove
} from "https://www.gstatic.com/firebasejs/9.8.1/firebase-database.js";

// W jaki sposób usuwać dane w bazie danych?
//  1. Musimy wskazać, który wpis chcemy usunąć. Taki wskaźnik składa się z dwóch informacji:
//      1.1. W jakiej bazie chcemy usunąć dane? Zmienna database
//      1.2. Musimy wskazać jaki węzeł chcemy usunąć, np. users/uzytkownik1.

// Możemy wykonać to na dwa sposoby. Poprzez funkcję remove lub ustawienie wartości null
// dla danego wpisu z bazy danych.

// Sposób 1: Wykorzystnie funkcji remove
remove(ref(database, "users/uzytkownik1"));

// Sposób 2: Ustawienie dla wpisu wartości null
set(ref(database, "users/uzytkownik1"), null);
```

11. Pobieranie wszystkich danych z danego węzła.

```
// W jaki sposób wyświetlić wszystkie dane z danego poziomu?
//  Możemy to sobie wyobrazić jako wyświetlenie całej zawartości folderu.

// Pobieramy całą zawartość "folderu" users
```

```
onValue(ref(database, "users"), (snapshot) => {
    const allUsersSnapshot = snapshot.val();

    // Jeśli chcemy wyświetlić tylko klucze:
    const userKeys = Object.keys(allUsersSnapshot);
    console.log(userKeys);

    // Jeśli chcemy wyświetlić wartości:
    Object.values(allUsersSnapshot).forEach(userProperties => {
        console.log(userProperties);
    })
})
```

12. Możliwość nasłuchiwania na wydarzenia na danym węzłu.

```
// Import zależności dot. modułu Realtime Databse
import {
    getDatabase,
    ref,
    set,
    get,
    onValue,
    update,
    remove,
    onChildAdded,
    onChildChanged
} from "https://www.gstatic.com/firebasejs/9.8.1/firebase-database.js";

// Realtime Database pozwala nam także na reagowanie na konkretne akcje w konkretnej ścieżce.
// Np. onChildAdded -> Zostanie wywołany, gdy pojawi się nowy wpis na tym poziomie.
// Docs: https://firebase.google.com/docs/database/web/lists-of-data?hl=en

const commentsRef = ref(database, 'users');
onChildAdded(commentsRef, (data) => {
  console.log(data.key, data.val());
});

onChildChanged(commentsRef, (data) => {
    console.log(data.key, data.val());
});
```

ZADANIA - CRUD

1. Uruchom moduł Realtime Database w konsoli Firebase.
2. Napisz fragment kodu, który doda do bazy danych poniższy obiekt pod ścieżką:
   `products/firebase-course`

```
{
    title: "Firebase Course",
```

```
    duration: 32,
    students: [
        "John",
        "Kate",
        "Steven"
    ]
}
```

3. Napisz fragment kodu, który doda do bazy danych poniższy obiekt pod ścieżką: `products/react-course`

```
{
    title: "React Course",
    duration: 48,
    students: [
        "John",
        "Susan"
    ]
}
```

4. Napisz fragment kodu, który doda do obu kursów nowe pole `discount: 0`.
5. Napisz fragment kodu, który będzie nasłuchiwał na zmiany oraz wyświetlał w konsoli w przeglądarce wartości pól kursu `react-course`.
6. Napisz fragment kodu, który dla kursu `react-course` zmodyfikuje pole `discount: 30`. Informacja o zmianie wartości pola powinna pojawić się w konsoli w przeglądarce ze względu na kod dodany w ramach zadania 6.
7. Napisz fragment kodu, który wyświetli w konsoli w przeglądarce nazwy wszystkich kursów.
8. Napisz fragment kodu, który wyświetli w konsoli w przeglądarce liczbę studentów przypisanych do każdego z kursów w postaci: `Firebase Course: 3`.
9. Do UI dodaj przycisk ‘Pokaż liczbę kursów’.
    a. Po naciśnięciu takiego przycisku - na stronie powinna pojawić się informacja o tym ile jest dostępnych kursów w bazie danych.
    b. Sprawdź, czy funkcja odpowiedzialna za wyświetlanie liczby dostępnych kursów działa prawidłowo - w tym momencie powinna być wyświetlana wartość 2.
10. Napisz fragment kodu, który usunie z bazy danych kurs `react-course`.
11. Sprawdź, czy funkcja odpowiedzialna za wyświetlanie liczby dostępnych kursów działa prawidłowo - w tym momencie powinna być wyświetlana wartość 1.
12. ** Do UI dodaj wyświetlanie listy wszystkich kursów wraz z informacjami na ich temat.

a. Napisz tą funkcjonalność w taki sposób, że zmiana wartości w bazie spowoduje odświeżenie listy kursów i ich własności po stronie UI.

b. Poniżej przykład w jaki może zostać opisany każdy z kursów:

```
Firebase Course
- Duration: 48
- Students: 3
- Duration: 0
```