infoShare ACADEMY

Firebase – Firestore i JS SDK

infoShare Academy

infoShareAcademy.com

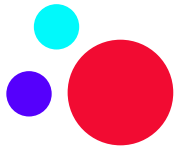
HELLO

Dawid Buliński

Front End Developer

infoShare ACADEMY

Agenda

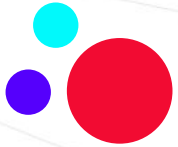


1. Konfiguracja ćwiczeń
2. Firebase - rejestracja, podstawowe informacje o konsoli
3. Firebase SDK, struktura, sposób użycia
4. Cloud Firestore - podstawowe informacje
5. Połączenie aplikacji z Firestore
6. Implementacja pobierania listy konwersacji
7. Ręczne tworzenie danych za pomocą konsoli
8. Implementacja pozostałych funkcjonalności (dodawanie konwersacji, wiadomości, usuwanie konwersacji, sortowanie)
9. Q & A.

Firebase

https://firebase.google.com/

Firebase

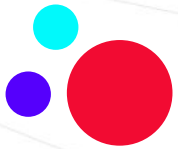


Firebase SDK

https://www.npmjs.com/package/firebase

https://firebase.google.com/docs/firestore/client/libraries?hl=en

Podczas szukania informacji o tym jak konfigurować i używać narzędzi od firebase, warto upewnić się, że odnoszą się do odpowiedniej wersji. W przykładach będziemy używać wersji **9.16.x**. Można spotkać się ze starszymi materiałami, które odnoszą się do wersji **8.x.x.** Między tymi dwoma wersjami występują bardzo duże różnice.



Firebase SDK

Biblioteka Firebase składa się z wielu niezależnych od siebie modułów. Podczas korzystania z niej, należy importować funkcje tylko z modułów których używamy. W przypadku dzisiejszych zajęć potrzebne moduły to:

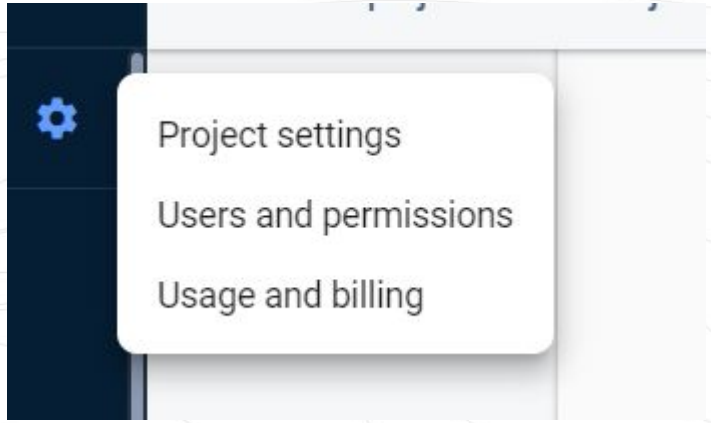
- **firebase/app**
- **firebase/firestore**

Cloud Firestore

1. Baza danych NoSQL
2. Umożliwia synchronizację realtime z aplikacjami klienckimi
3. Zapewnia wsparcie offline
4. Skalowalność poprzez serwisy Google Cloud
5. Wsparcie dla wielu języków

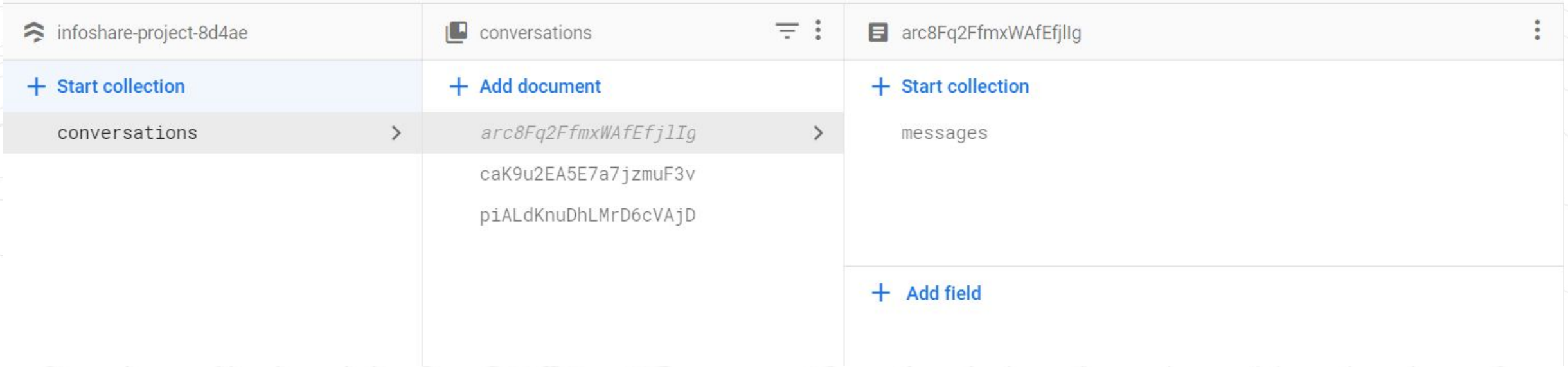
Ćwiczenie 1. Połączenie z Firebase

1. Utwórz nowy plik o nazwie *firebase.js* w folderze *src.*
2. W przeglądarce przejdź do ustawień projektu, a następnie skopiuj dane konfiguracyjne (znajdują się na dole strony).
3. Wklej dane konfiguracyjne w nowo-utworzonym pliku.
4. Zaimportuj stworzony przez siebie plik w **App.jsx**
5. Uruchom aplikację i sprawdź w konsoli czy nie pojawiły się żadne błędy. Jeśli nie - zadanie zostało wykonane poprawnie.



Konsola Firestore

Za pomocą graficznego eksploratora udostępnianego przez konsole Firestore, mamy możliwość manipulacji danymi spoza naszej aplikacji.

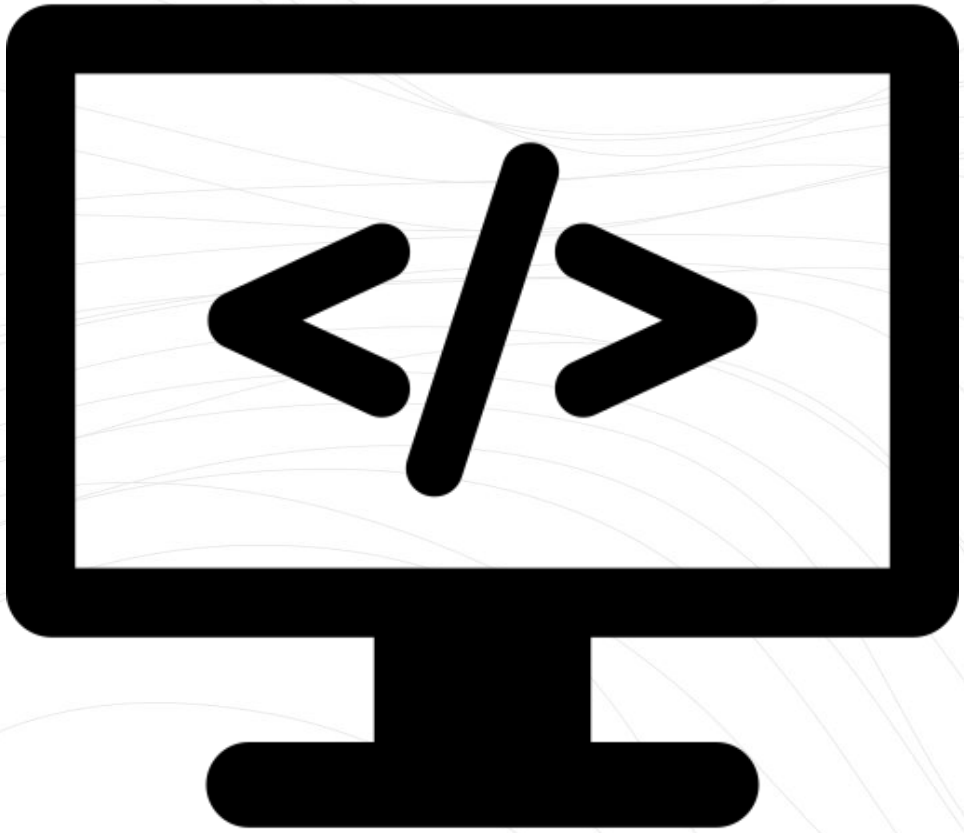


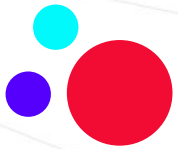
Ćwiczenie 2. Dodaj manualnie dane

1. Za pomocą konsoli Firestore, stwórz nową kolekcję o nazwie *conversations*
2. Dodaj nowy dokument, nadaj mu automatyczne ID, a następnie dodaj pole *name* typu *string* i dodaj dowolną wartość np (‘Pierwsze konwersacja’).
3. Wewnątrz stworzonego dokumentu dodaj nową kolekcję *messages*
4. Dodaj nowy dokument wewnątrz kolekcji *messages* i dodaj do niego pola:
   author: ‘Swoje imie’
   message: ‘To jest pierwsza wiadomosc z Firestore’
   createdAt: 1675111053136

Podstawowe operacje w Firestore

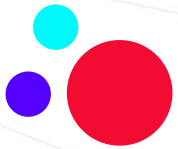
Live coding





Ćwiczenie 3. Pobierz dane w aplikacji

1. Podłącz aplikację do Firestore. W pliku *firebase.js* dodaj import
   ```
   import { getFirestore } from "firebase/firestore";
   ```
   A następnie stwórz zmienną db za pomocą poniższej linijki kodu:
   ```
   export const db = getFirestore(firebaseApp);
   ```
2. W pliku **conversations-list/index.jsx** stwórz nowy efekt, uruchamiany tylko raz, po zamontowaniu komponentu
3. Za pomocą funkcji *collection* i *onSnapshot* pobierz dane z kolekcji *conversations*.
4. W stanie *conversations* ustaw tablicę składającą się z pól *id* i *name* (id powinno być identyfikatorem dokumenty, a name pochodzić z pola w dokumencie o tej samej nazwie)
5. Funkcja *onSnapshot* zwraca funkcję która po wywołaniu odsubkrybuje nas od nasłuchiwania na zmiany w kolekcji *conversations*. Wywołaj ją w momencie, gdy komponent będzie odmontowywany.
6. Lista konwersacji po prawej stronie aplikacji powinna teraz wyświetlać jedną konwersację.
7. Nie zamykając aplikacji przejdź do konsoli Firebase, a następnie zmień pole *name* w konwersacji.
8. Zweryfikuj, czy nazwa konwersacji została zmieniona automatycznie.



Ćwiczenie 4. Pobieranie wiadomości

1. W pliku **conversation/index.jsx** stwórz nowy efekt, uruchamiany zawsze, gdy zmieni się wartość stringa *id*.
2. Pobierz kolekcję **conversations/:id/messages**
3. Każdorazowo, gdy dane zostaną zmienione ustawiaj jest w stanie *messages*. Powinna to być tablica zawierająca wszystkie dane z dokumentu, dodatkowo wzbogacona o pole *id*, jako identyfikator dokumentu
4. Zweryfikuj czy dane wyświetlane są poprawnie.

Ćwiczenie 5. Dodawanie wiadomości

1. W pliku **conversation/new-message.jsx** zaimplementuj funkcję *handleSubmit*
2. Użyj funkcji *addDoc* aby do kolekcji *conversations/:id/messages* dodać nowy dokument.
3. Pola dokumentu to: *author, message, createdAt*. Dwie pierwsze wartości pochodzić powinny z formularza, a ostatnia to *Date.now()*.
4. Zweryfikuj, czy po przesłaniu formularza wiadomość pojawiła się na liście.

Ćwiczenie 6. Dodawanie konwersacji

1. W pliku **conversations-list/new-conversation.jsx** zaimplementuj funkcję *onConversationCreate.*
2. Użyj funkcji *addDoc* aby do kolekcji *conversations* dodać nowy dokument.
3. Pola dokumentu to: *name*, którego wartość to wartość inputa ‘title’.
4. Zweryfikuj, czy po przesłaniu formularza konwersacja pojawiła się na liście po prawej stronie.
5. Po wykonaniu operacji przekieruj użytkownika do nowej konwersacji, a następnie zamknij modal.

Ćwiczenie 7. Usuwanie konwersacji

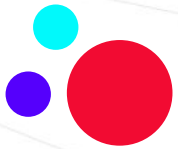
1. W pliku **conversations-list/index.jsx** zaimplementuj funkcję *deleteConversation*.
2. Użyj funkcji *deleteDoc* aby z kolekcji *conversations/:id/messages* usunąć dokument o odpowiednim id.
3. Po wykonaniu operacji przekieruj użytkownika na ścieżkę '/'.
4. Zweryfikuj, czy po przesłaniu requestu konwersacja zniknęła z listy po prawej stronie, a użytkownik przekierowany został do głównej strony.

Ćwiczenie 7. Usuwanie konwersacji

1. W pliku **conversations-list/index.jsx** zaimplementuj funkcję *deleteConversation*.
2. Użyj funkcji *deleteDoc* aby z kolekcji *conversations/:id/messages* usunąć dokument o odpowiednim id.
3. Po wykonaniu operacji przekieruj użytkownika na ścieżkę '/'.
4. Zweryfikuj, czy po przesłaniu requestu konwersacja zniknęła z listy po prawej stronie, a użytkownik przekierowany został do głównej strony.

Ćwiczenie 8. Sortowanie wyników

1. Posortuj listę wiadomości z użyciem standardowej funkcji *sort*, tak aby wiadomości wysłane najpóźniej znajdowały się na dole listy.
2. Następnie usuń funkcję sort i postaraj się uzyskać taki sam efekt za pomocą funkcji *orderBy*.
3. Zastanów się które rozwiązanie jest lepsze i dlaczego?



Q&A

infoShare ACADEMY

infoShare ACADEMY

Koniec

infoShare Academy

infoShareAcademy.com