

Firebase



React CRUD

React Firebase REST Post App https://race-rest.web.app

POSTS CREATE

 Morten Algy Bonderup
Senior Lecturer



Qui est esse

Est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui aperiam non debitis possimus qui neque nisi nulla

 Dan Okkels Brendstrup
Lecturer



Consequuntur deleniti eos quia temporibus ab aliquid at

Voluptatem cumque tenetur consequatur expedita ipsum nemo quia explicabo aut eum minima consequatur tempore cumque quae est et et in consequuntur voluptatem voluptates aut

 Kim Elkjær Marcher-Jepsen
Senior Lecturer



At nam consequatur ea labore ea harum

Cupiditate quo est a modi nesciunt soluta ipsa voluptas error itaque dicta in autem qui minus magnam et distinctio eum accusamus ratione error aut

 Birgitte Kirk Iversen
Senior Lecturer



Jes Arbov
Lecturer



 Maria Louise Bendixen
Senior Lecturer



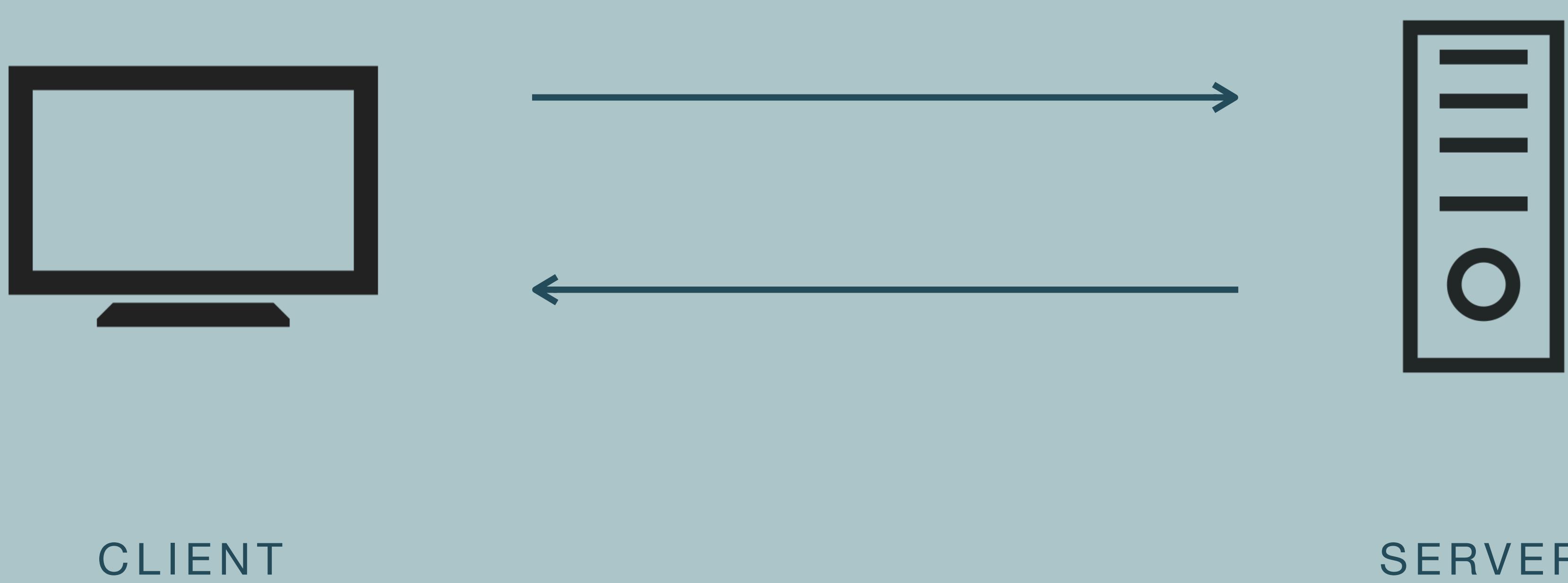
<https://race-rest.web.app/>

Content

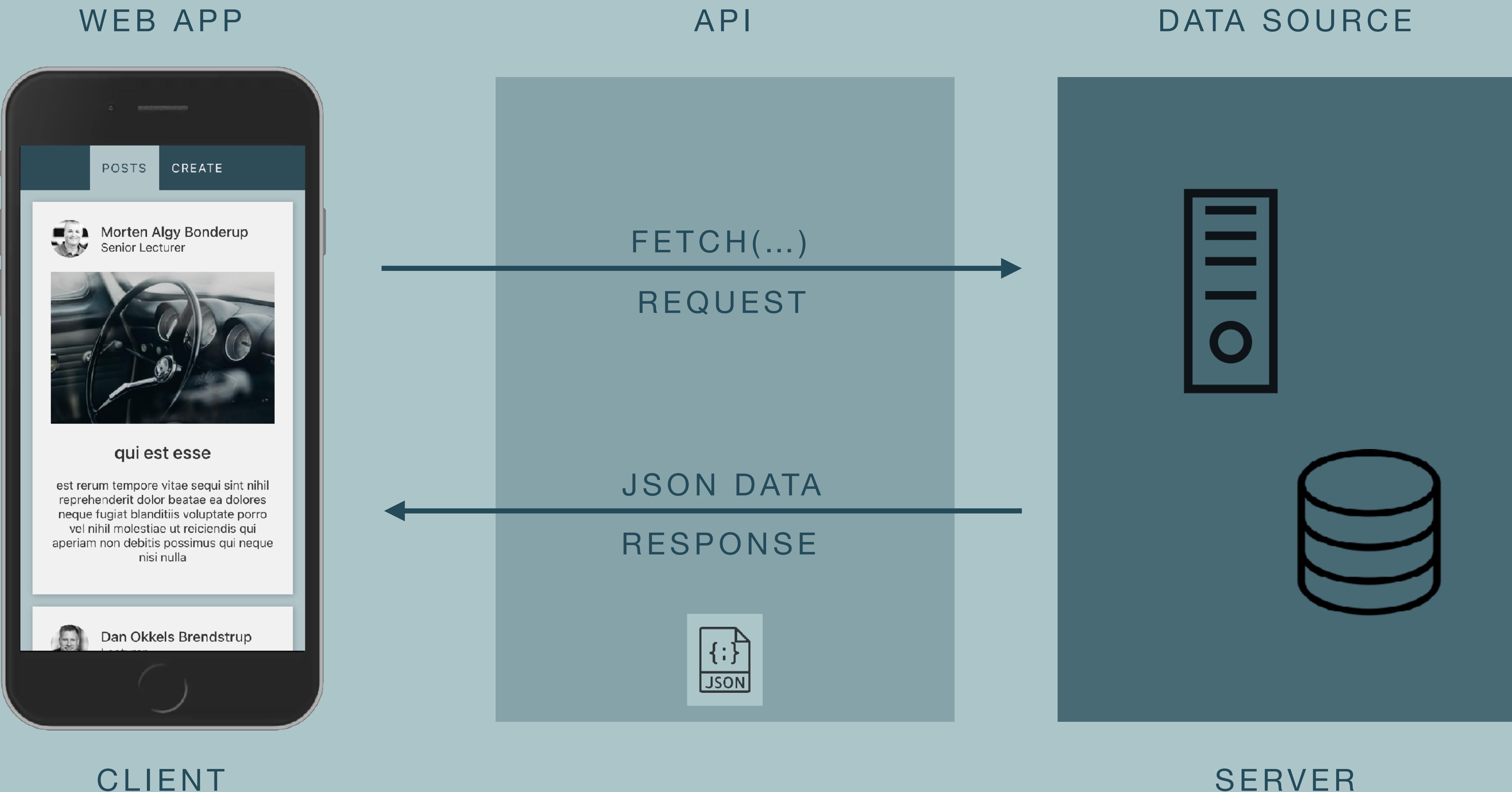
- Client-Server Model
- What's Firebase?
- Realtime Database
- Cloud Firestore
- Cloud Storage
- Firebase Authentication
- Security Rules

Client-Server Model

... how to communicate with backend/ data source (server)



Web Development



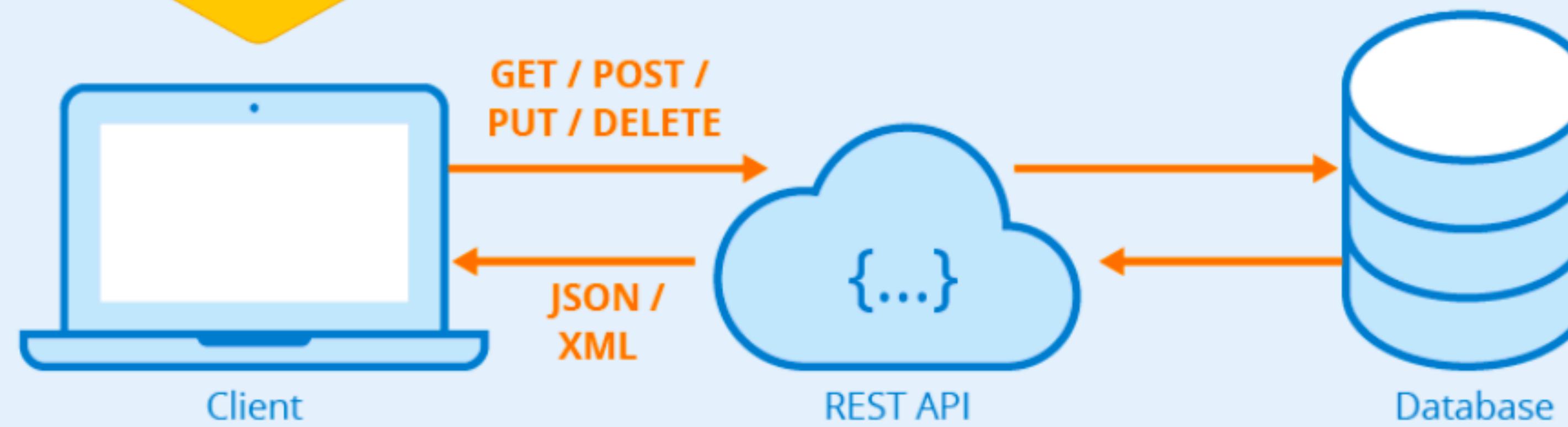
When someone asks you how to get data from a database in a js code



made with mematic

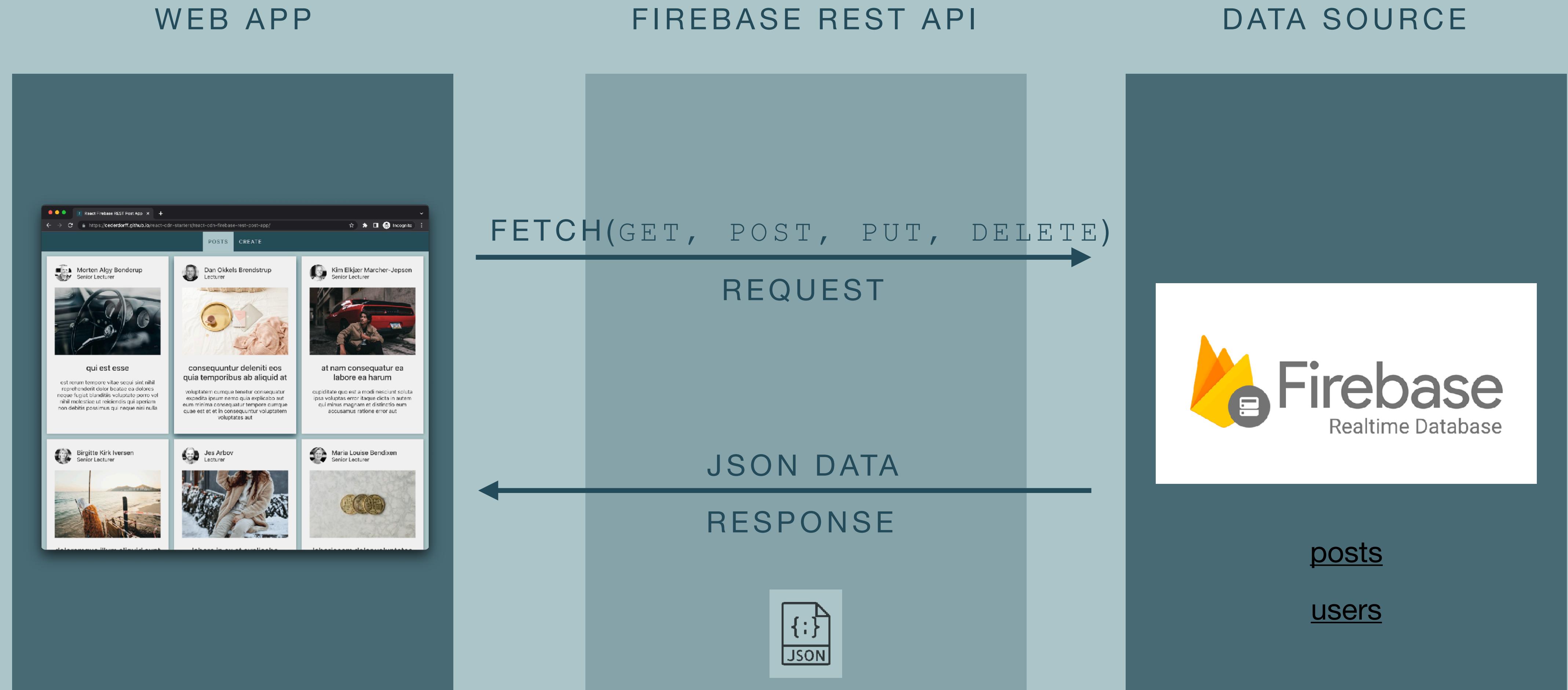


Firebase





Fetch, HTTP Request & Response





What is Firebase?

Platform, a suite of tools & Backend-as-a-Service
for Web & App Development

100 *SECONDS OF*

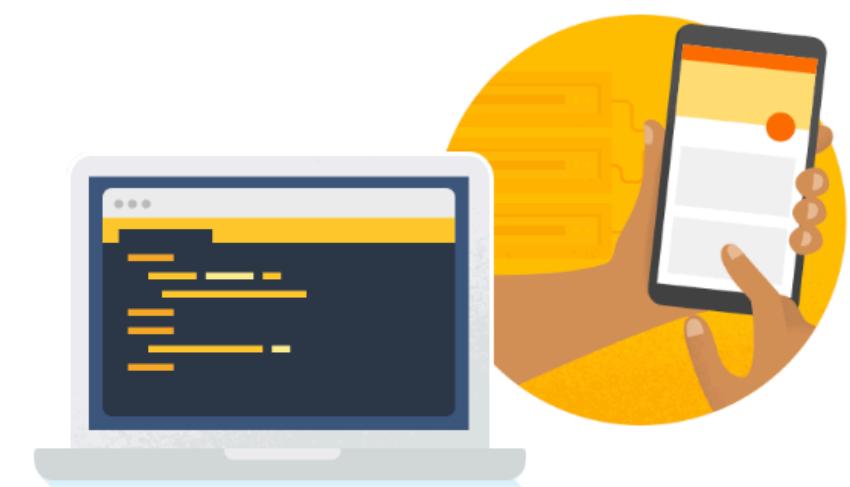


<https://www.youtube.com/watch?v=vAoB4VbhRzM>



Introducing Firebase

<https://www.youtube.com/watch?v=iosNuldQoy8>



Build better apps



Cloud Firestore

Store and sync app data at global scale



ML Kit BETA

Machine learning for mobile developers



Cloud Functions

Run mobile backend code without managing servers



Authentication

Authenticate users simply and securely



Hosting

Deliver web app assets with speed and security



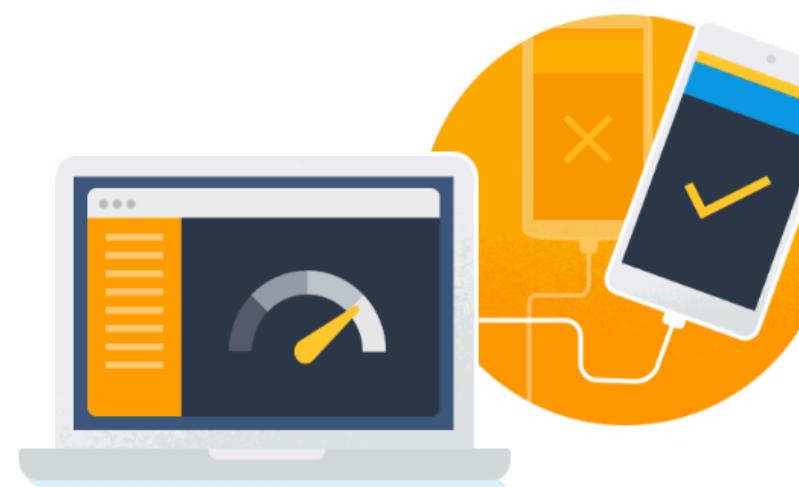
Cloud Storage

Store and serve files at Google scale



Realtime Database

Store and sync app data in milliseconds



Improve app quality



Crashlytics

Prioritize and fix issues with powerful, realtime crash reporting



Performance Monitoring

Gain insight into your app's performance



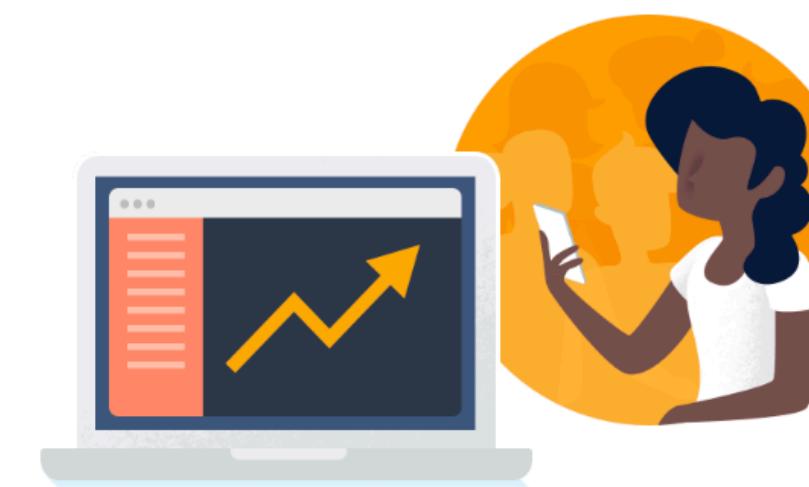
Test Lab

Test your app on devices hosted by Google



App Distribution BETA

Distribute pre-release versions of your app to your trusted testers



Grow your business



In-App Messaging BETA

Engage active app users with contextual messages



Google Analytics

Get free and unlimited app analytics



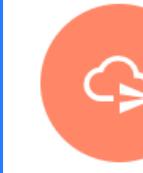
Predictions

Smart user segmentation based on predicted behavior



A/B Testing BETA

Optimize your app experience through experimentation



Cloud Messaging

Send targeted messages and notifications



Remote Config

Modify your app without deploying a new version



Dynamic Links

Drive growth by using deep links with attribution

MANY DEVICES
ONE PLATFORM





**my firebase doesn't
work. Can you help me?**

**read the
documentation**

But...

**read the
documentation**



Realtime Database & REST API

Store and sync data in real time
REST API or SDK

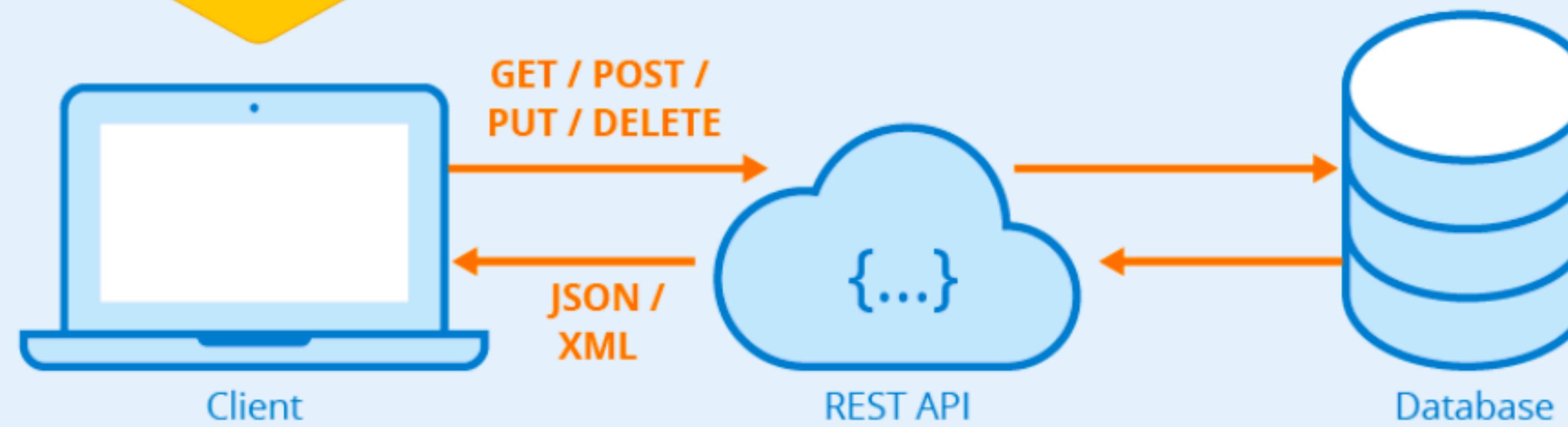
<https://firebase.google.com/products/realtime-database>



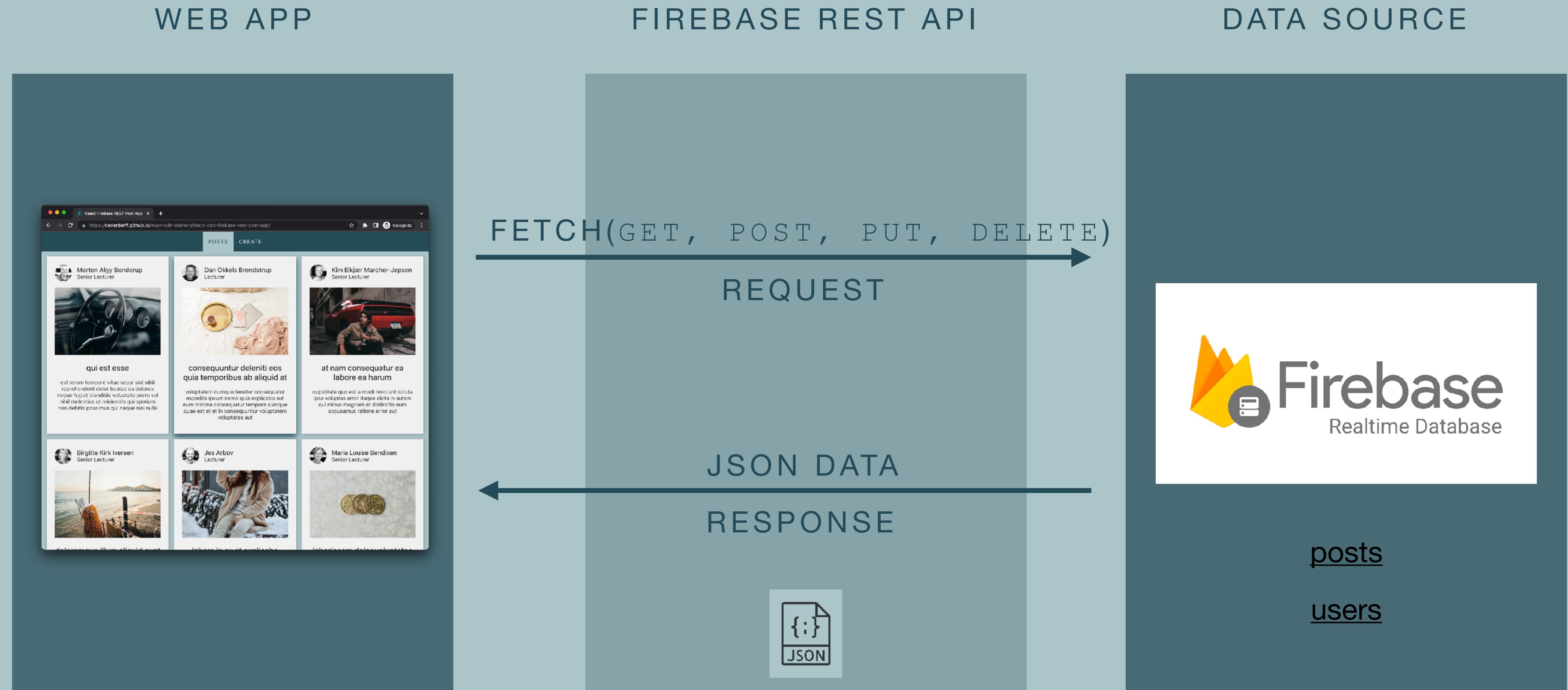
Database



Firebase



Fetch, HTTP Request & Response



Realtime Database REST API

[https://firebase.google.com/docs/
database/rest/start](https://firebase.google.com/docs/database/rest/start)

```
export default function PostsPage() {
  const [posts, setPosts] = useState([]);
  const [showLoader, dismissLoader] = useIonLoading();

  async function getPosts() {
    const response = await fetch("https://race-rest-default-rtbd.firebaseio.com/posts.json");
    const data = await response.json();
    // map object into an array with objects
    const postsArray = Object.keys(data).map(key => ({ id: key, ...data[key] }));
    return postsArray;
  }

  async function getUsers() {
    const response = await fetch("https://race-rest-default-rtbd.firebaseio.com/users.json");
    const data = await response.json();
    // map object into an array with objects
    const users = Object.keys(data).map(key => ({ id: key, ...data[key] }));
    return users;
  }
}
```



Cloud Firestore

<https://www.youtube.com/watch?v=QcsAb2RR52c>

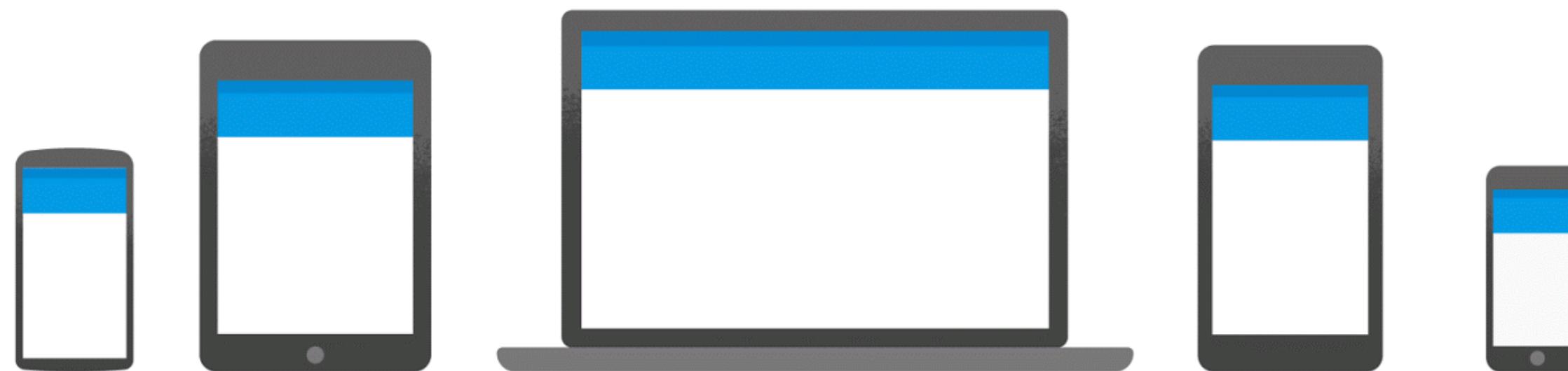


CLOUD FIRESTORE

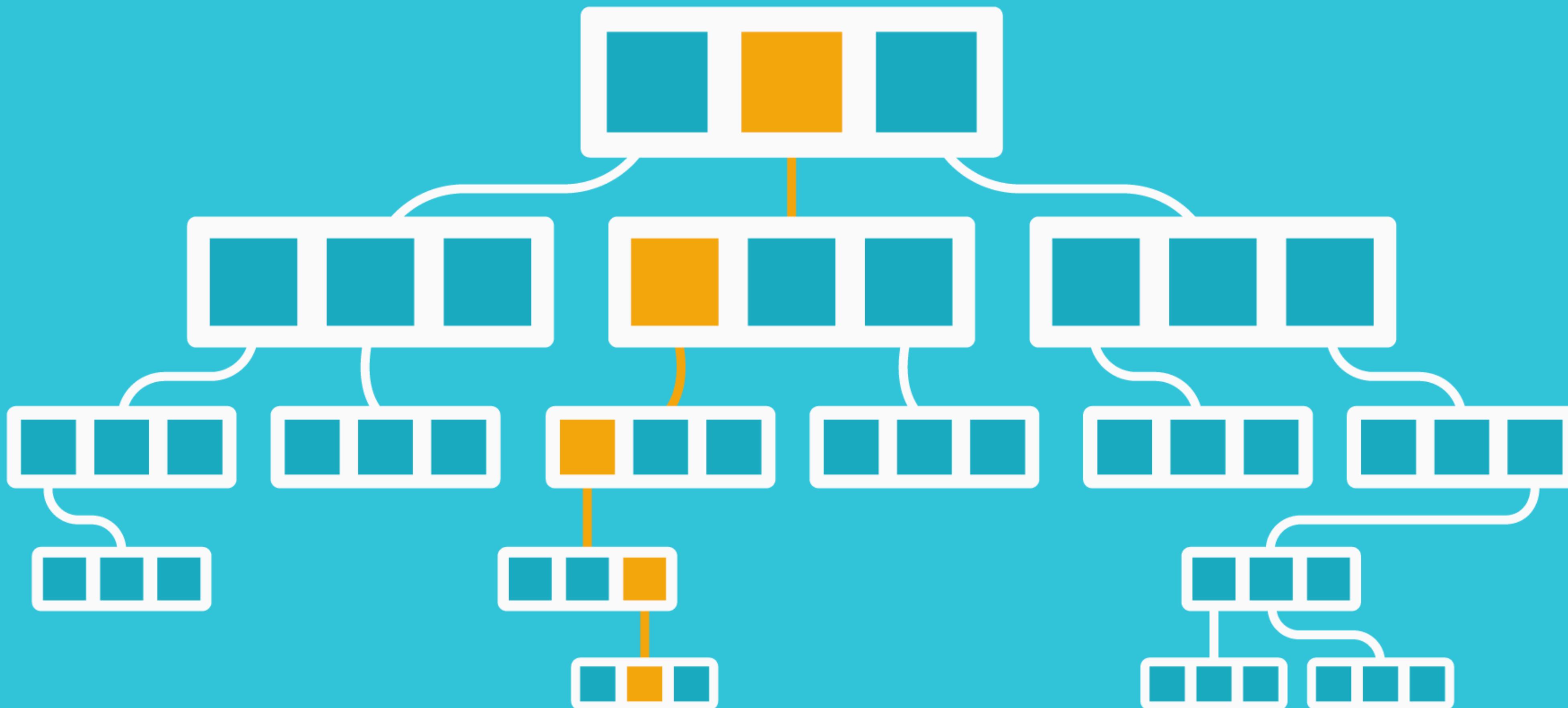
STORE AND SYNC DATA ACROSS PLATFORMS

<https://firebase.google.com/products/firestore/>

REALTIME UPDATES

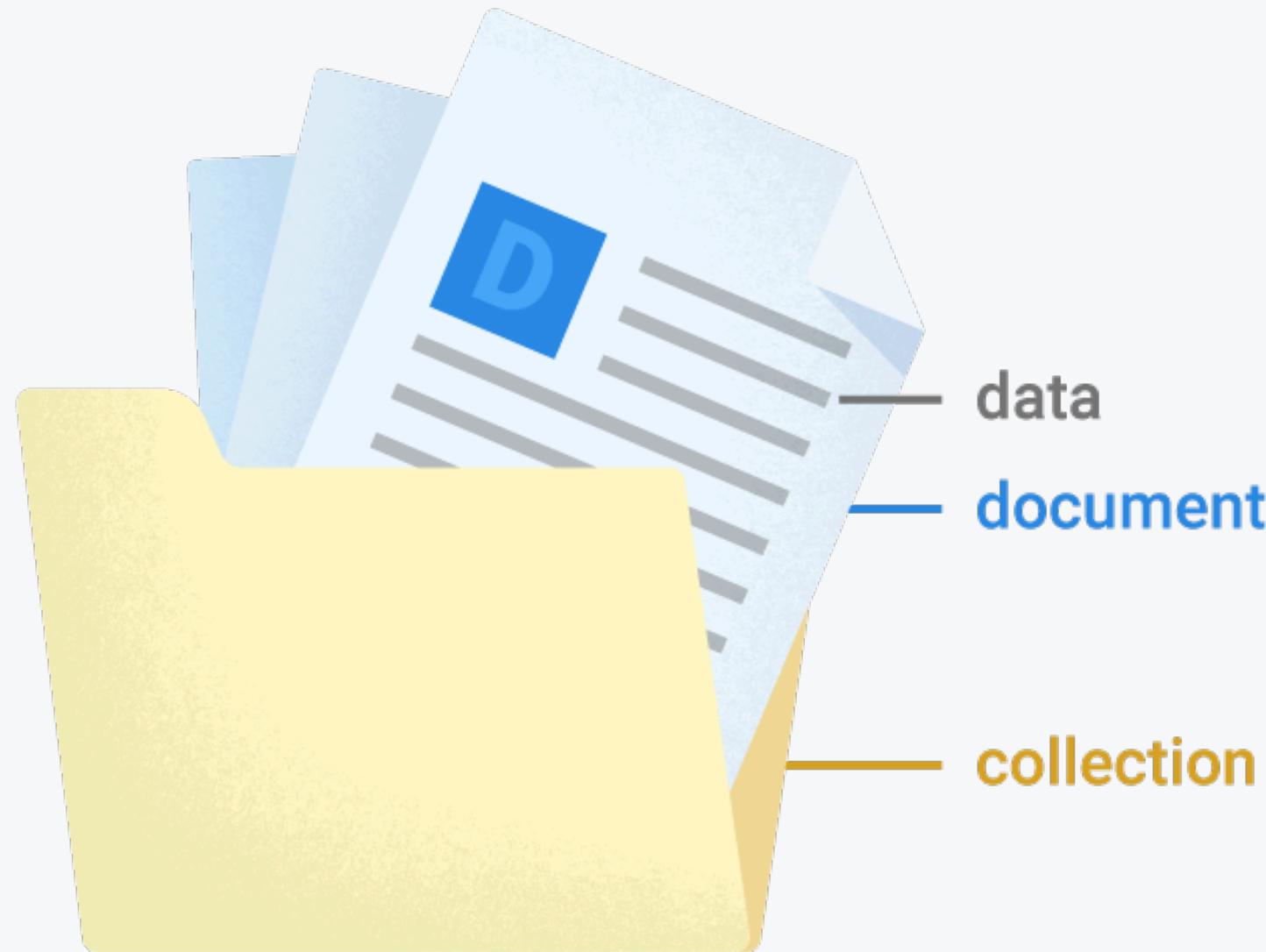


COLLECTIONS & DOCUMENTS



 > users > seme0qoWIzQh...

 mdu-e18front	 users	 seme0qoWIzQhZDfI1m0L
+ Start collection	+ Add document	+ Start collection
users	IS35ivFQ0BBYr0bwZX7k cNzvoH8XSRib6XXiUYuX seme0qoWIzQhZDfI1m0L	+ Add field mail: "bki@eaaa.dk" name: "Birgitte"

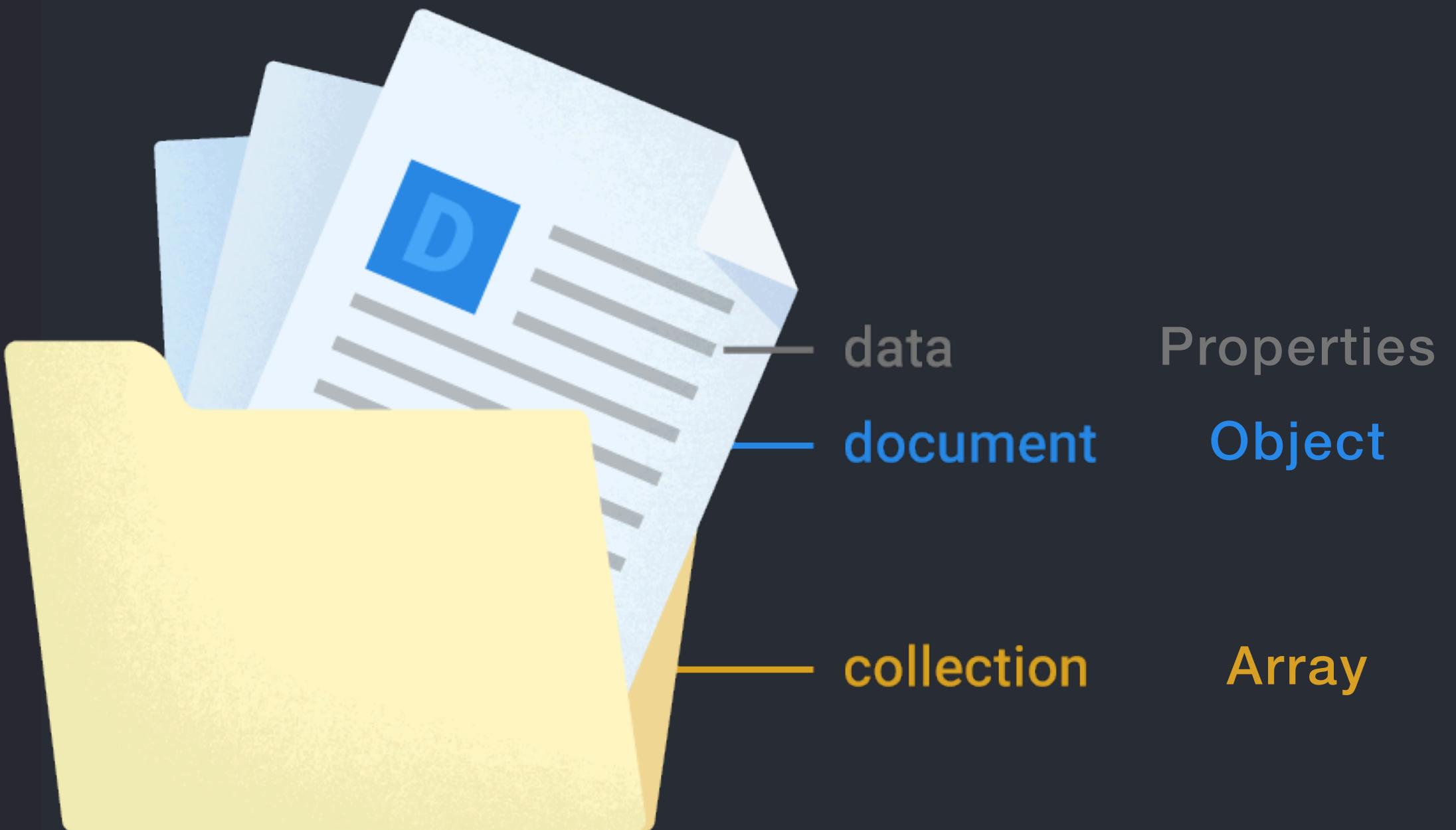


Home > users > seme0qoWIzQh...

mdu-e18front	users	seme0qoWIzQhZDfI1m0L
+ Start collection	+ Add document	+ Start collection
users >	IS35ivFQ0BBYr0bwZX7k cNzvoH8XSRib6XXiUYuX seme0qoWIzQhZDfI1m0L >	+ Add field mail: "bki@eaaa.dk" name: "Birgitte"

COLLECTION, DOCUMENT & DATA

```
let teachers = [  
    {  
        name: "Birgitte Kirk Iversen",  
        initials: "bki",  
        mail: "bki@baaa.dk",  
        phone: "72286316",  
        address: "Sønderhøj 30, 8260 Viby J",  
        position: "Senior Lecturer",  
        department: "Multimedia Design",  
        img: "https://www.baaa.dk/CropUp/headshot/media/1524902/birgitte-kirk-iversen.jpg"  
    }, {  
        name: "Gertie Margrethe Kolding Jensen",  
        initials: "gkj",  
        mail: "gkj@baaa.dk",  
        phone: "72286349",  
        address: "Sønderhøj 30, 8260 Viby J",  
        position: "Senior Lecturer",  
        department: "Multimedia Design",  
        img: "https://www.eaaa.dk/CropUp/headshot/media/2046228/Gertie-Kolding.jpg"  
    }, {  
        name: "Kim Elkjær Marcher-Jepsen",  
        initials: "kije",  
        mail: "kije@baaa.dk",  
        phone: "7228 6325",  
        address: "Sønderhøj 30, 8260 Viby J",  
        position: "Lecturer",  
        department: "Multimedia Design",  
        img: "https://www.baaa.dk/CropUp/headshot/media/3124373/Kim-Elkjaer-Marcher-Jepsen.jpg"  
    }, {  
        name: "Rasmus Cederdorff",  
        initials: "rc",  
        mail: "rc@baaa.dk",  
        phone: "7228 6326",  
        address: "Sønderhøj 30, 8260 Viby J",  
        position: "Lecturer",  
        department: "Multimedia Design",  
        img: "https://www.baaa.dk/CropUp/headshot/media/3124374/Rasmus-Cederdorff.jpg"  
    }]
```





12.54 ⓘ

THE BIG FRIDGE

Pluk selv Egen råvare Overskud

Julie Pedersen 21/2/2018 kl. 22:22

Rød peber 2 stk 10 kr

Overskud 28/6 God Aarhus V

Rasmus Cederdorff 8/5/2018 kl. 17:52

KØB

FØLGER SÆLG BESKEDER MIN PROFIL

15.05 ⓘ

SØG

Nyeste brugere

Han... Birthe Ande... Jesper Chri... Lars Jensen Ka...

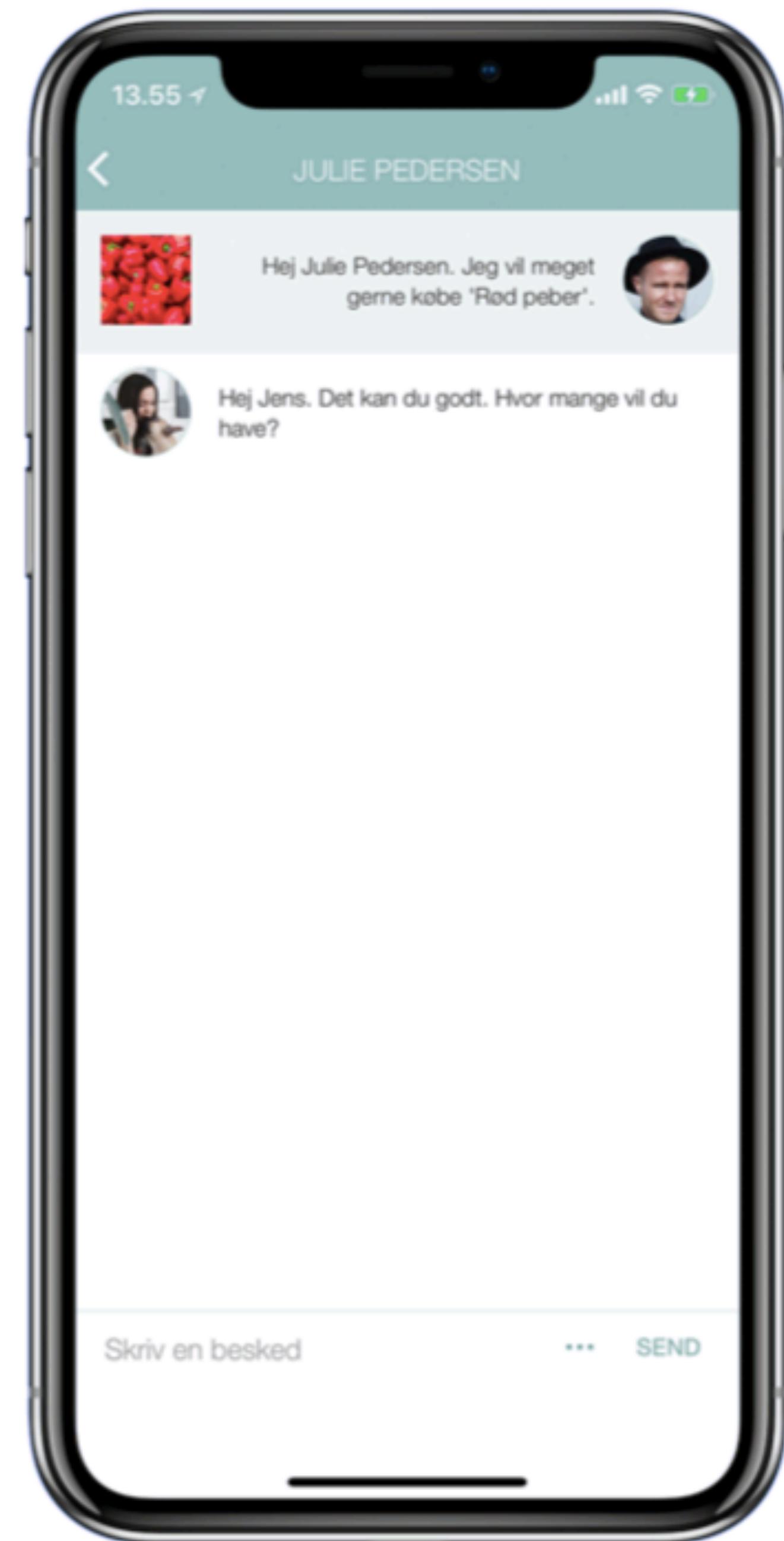
Nyeste salgsindlæg

Rasmus Cederdorff 8/5/2018 kl. 17:52

Peber 10 stk 15 kr

KØB

FØLGER SÆLG BESKEDER MIN PROFIL



BEGIVENHEDER



Juni 2018



Man Tir Ons Tor Fre Lør Søn

18

19

20

21

22

23

24

22/06/2018

Hele dagen

Alle dage er en fredag 🙌

09.06

10.06

Det er jo fredag! 😊

09.45

10.00

Fredagsmøde med management

FRAVÆRENDE

I dag

**Rasmus Cederdorff (RC)**

Arbejder hjemme hele dagen. Fang mig på mobil.

AH

**Rikke Gravesen Horsevad (RGH)**

21 + 22 fri

FRI

**Gitte Egholm Hansen (GH)**

Fri fredag d. 22/6

FRI

**Bettina Beermann (BBE)**

Går ved 12-tiden fredag 22/6

GT

Kommende

**Rasmus Cederdorff (RC)**

Ferie fra 30/06 til 07/07.

FERIE

**Rasmus Cederdorff (RC)**

ANDET

Michael Andersen (MA)

Er i Belgien. Fang mig på mobilen.

FR

Anett Andersen (AAN)

Ferie fra 22. juli til 14. august.

FERIE

**Anne Skvum (AS)**

Hjem



Kalender



Fraværende



Kontakter



Min side



CLOUD FIRESTORE

CREATE, READ, UPDATE & DELETE DATA

<https://firebase.google.com/products/firestore/>



FIREBASE TEMPLATES

firebase - *

spa-firebase - *

react-cdn-firebase - *

React-cdn-firebase - *



FIREBASE DOCS

Getting started video: https://youtu.be/iosNuldQoy8?list=PLI-K7zZEsYLmOF_07IayrTntevxtbUxDL

Cloud Firestore Introduction: <https://firebase.google.com/docs/firestore>

Get started with Cloud Firestore: <https://firebase.google.com/docs/firestore/quickstart>

Add and manage data: <https://firebase.google.com/docs/firestore/manage-data/add-data>

Delete data: <https://firebase.google.com/docs/firestore/manage-data/delete-data>

Read data: <https://firebase.google.com/docs/firestore/query-data/get-data>

Read data - realtime updates: <https://firebase.google.com/docs/firestore/query-data/listen>

Order and limit data: <https://firebase.google.com/docs/firestore/query-data/order-limit-data>



Documentation

<https://firebase.google.com/docs/firestore/>[Overview](#) [Guides](#) [Reference](#) [Samples](#) [Libraries](#)[Send feedback](#)

Guides

[Get started with Firebase](#)[Manage your Firebase projects](#)[Prototype and test with Emulator Suite](#)

Analytics

DEVELOP

Authentication

Realtime Database

Cloud Firestore

Introduction

- [Get started](#)
- [Understand Cloud Firestore](#)
- [Add and manage data](#)
- [Read data](#)
- [Secure and validate data](#)
- [Solutions](#)
- [Usage, limits, and pricing](#)
- [Cloud Firestore integrations](#)
- [API reference](#)
- [Samples](#)

Storage

Firebase > Docs > Guides



Cloud Firestore

 [iOS](#) [Android](#) [Node.js](#) [Java](#) [Python](#) [Go](#)

Use our flexible, scalable NoSQL cloud database to store and sync data for client- and server-side development.

Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform. Like Firebase Realtime Database, it keeps your data in sync across client apps through realtime listeners and offers offline support for mobile and web so you can build responsive apps that work regardless of network latency or Internet connectivity. Cloud Firestore also offers seamless integration with other Firebase and Google Cloud Platform products, including Cloud Functions.

[Get started](#)

Key capabilities

Flexibility

The Cloud Firestore data model supports flexible, hierarchical data structures. Store your data in documents, organized into collections. Documents can contain complex nested objects in addition to subcollections.

Contents

[Key capabilities](#)[How does it work?](#)[Implementation path](#)[Next steps](#)

CREATE

```
addDoc(_usersRef, newUser);
```

READ

```
onSnapshot(_usersRef, snapshot => { ...  
});
```

UPDATE

```
const userRef = doc(_usersRef, _selectedUserId);  
updateDoc(userRef, userToUpdate);
```

DELETE

```
const userRef = doc(_usersRef, id);  
deleteDoc(userRef);
```



GETTING STARTED

Getting Started with Firebase Cloud Firestore & React



R E A D



READ DOCS

```
import { useState, useEffect } from "react";
import { onSnapshot } from "firebase/firestore";
import { postsRef } from "../firebase-config";           Imports

export default function HomePage() {
  const [posts, setPosts] = useState([]);                 State

  useEffect(() => {
    onSnapshot(postsRef, data => {
      const postsData = data.docs.map(doc => {
        return { ...doc.data(), id: doc.id };
      });
      setPosts(postsData);
    });
  }, []);                                                 useEffect

  return (
    <section className="page">
      <section className="grid-container">
        {posts.map(post => (
          <article key={post.id}>
            <img src={post.image} alt={post.title} />
        ))
      )
    )
  );
}                                         Transform data &
                                            save in state

Use map to return
HTML & display
every post
```



SET UP YOUR OWN
FIREBASE PROJECT



Your Firebase projects

[Add project](#)**mdu-e18front**

mdu-e18front

</>

mdu-f18front

mdu-f18front

</>

LoginApp

loginapp-a7a17

eaaa-front-fall-18

eaaa-front-fall-18

[CONSOLE.FIREBASE.GOOGLE.COM](https://console.firebaseio.google.com)

Firebase User CRUD – Overview

console.firebaseio.google.com/u/1/project/user-crud-race/overview

Firebase User CRUD

Go to docs

Project Overview

Build

- Authentication
- Firestore Database
- Realtime Database
- Storage
- Hosting
- Functions
- Machine Learning

Release and monitor

Analytics

Engage

Extensions

Spark

Free \$0/month

Upgrade

Firebase User CRUD

Spark plan

Get started by adding Firebase to your app

Add an app to get started

Store and sync app data in milliseconds

The screenshot shows the Firebase Project Overview page for a project named "user-crud-race". The main header is "Firebase User CRUD". On the left sidebar, under the "Build" section, the "Web" icon is highlighted with a blue border. Below the sidebar, there's a call-to-action button labeled "Add an app to get started". The main content area features a large blue background with a white illustration of two people interacting with a digital interface. At the bottom, there are two cards: one showing a stack of databases and another showing a magnifying glass focusing on a database icon.

Firebase User CRUD – Cloud Fi X

console.firebaseio.google.com/u/1/project/user-crud-race.firebaseio

Firebase User CRUD

Go to docs

Project Overview

Build

- Authentication
- Firestore Database
- Realtime Database
- Storage
- Hosting
- Functions
- Machine Learning

Create database

Cloud Firestore

Real-time updates, powerful queries and automatic scaling

Is Cloud Firestore right for you? Compare Databases

Release and monitor

Analytics

Engage

Extensions

Learn more

How do I get started?

View the docs

Introducing Cloud Firestore

Watch later

Share

The screenshot shows the Firebase Cloud Firestore dashboard. The main title is "Cloud Firestore" with the subtitle "Real-time updates, powerful queries and automatic scaling". A prominent orange button labeled "Create database" is visible. To the right, there's a magnifying glass icon over a document containing the Firestore logo, symbolizing search and querying. A callout box asks "Is Cloud Firestore right for you?" with a "Compare Databases" link. The left sidebar includes links for Project Overview, Authentication, Firestore Database (which is currently selected), Realtime Database, Storage, Hosting, Functions, Machine Learning, Release and monitor, Analytics, Engage, and Extensions. At the bottom, there's a "Learn more" section with a "How do I get started?" link and a video thumbnail for "Introducing Cloud Firestore" with options to "Watch later" or "Share".

CREATE DATABASE

Firebase User CRUD – Cloud Fi X

console.firebaseio.google.com/u/1/project/user-crud-race/firestore

Firebase User CRUD

Go to docs

Project Overview

Build

- Authentication
- Firestore Database
- Realtime Database
- Storage
- Hosting
- Functions
- Machine Learning

Release and monitor

Analytics

Engage

Extensions

Cloud Firestore

Create database

1 Secure rules for Cloud Firestore 2 Set Cloud Firestore location

After you've defined your data structure, **you will need to write rules to secure your data.**

[Learn more](#)

Start in **production mode**
Your data is private by default. Client read/write access will only be granted as specified by your security rules.

Start in **test mode**
Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if
        request.time < timestamp.date(2021, 12, 17);
    }
  }
}
```

! The default security rules for test mode allow anyone with your database reference to view, edit and delete all data in your database for the next 30 days

Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project, notably from the associated App Engine app

Cancel Next

Watch later Share

The screenshot shows the 'Create database' dialog for Cloud Firestore. The 'test mode' option is selected. A code snippet for the default security rules is displayed:

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if
        request.time < timestamp.date(2021, 12, 17);
    }
  }
}
```

A warning message states: "The default security rules for test mode allow anyone with your database reference to view, edit and delete all data in your database for the next 30 days".

SECURITY RULES: test mode

Firebase User CRUD – Cloud Fl X

console.firebaseio.google.com/u/1/project/user-crud-race/firestore/data/~2F

Firebase User CRUD

Project settings

store

Users and permissions

Usage and billing

Build

- Authentication
- Firestore Database
- Realtime Database
- Storage
- Hosting
- Functions
- Machine Learning

Release and monitor

Analytics

Engage

Extensions

Project settings

Get started

Prototype and test end-to-end with the Local Emulator Suite, now with Firebase Authentication

user-crud-race

+ Start collection

Your database is ready to go. Just add data.

The screenshot shows the Firebase Firestore console interface. On the left, there's a sidebar with various services like Authentication, Firestore Database, and Machine Learning. The main area shows a collection named 'user-crud-race'. A context menu is open over the collection name, with 'Project settings' highlighted. Other options in the menu include 'Users and permissions', 'Usage and billing', and 'Get started' which links to the Local Emulator Suite documentation. At the bottom right of the main area, there's a message: 'Your database is ready to go. Just add data.' with a small icon of a database.

GO TO: Project settings

The screenshot shows the Firebase Project Settings page for a project named "React Firebase". The left sidebar contains navigation links for Project Overview, Authentication, Firestore Database, Realtime Database, Storage, Hosting, Functions, Machine Learning, Crashlytics, Performance, Test Lab, App Distribution, Extensions, Spark (No cost \$0/month), and Upgrade. The main content area is titled "SDK setup and configuration" and includes three radio button options: "npm" (selected), "CDN", and "Config". It provides instructions for installing the npm SDK and shows sample code for initializing Firebase:

```
$ npm install firebase
```

Then, initialise Firebase and begin using the SDKs for the products that you'd like to use.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyBA0u3NqaUza7vzDQli9dYRaUHwLZldiWw",
  authDomain: "race-react-firebase.firebaseio.com",
  projectId: "race-react-firebase",
  storageBucket: "race-react-firebase.appspot.com",
  messagingSenderId: "2118672388",
  appId: "1:2118672388:web:06a1d58d898e9702581551"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
```

Note: This option uses the [modular JavaScript SDK](#), which provides reduced SDK size.

COPY firebaseConfig and paste in firebase-config.js

The screenshot shows the Firebase Cloud Firestore interface in the Firebase console. The left sidebar contains navigation links for Authentication, Firestore Database (selected), Realtime Database, Storage, Hosting, Functions, and Machine Learning. The main area is titled "Cloud Firestore" and has tabs for Data, Rules, Indexes, and Usage. Under the Data tab, the path "posts > 1mNVXMNbZWjKsiaTFsDw" is selected. The interface includes a header with "race-react-firebase", "posts", and a document ID "1mNVXMNbZWjKsiaTFsDw". Below this are buttons for "Start collection", "Add document", and "Add field". The document details section shows fields: "body" (containing Latin placeholder text), "createdAt" (6 March 2022 at 17:13:55 UTC+1), "image" (a URL to a placeholder image from unsplash.com), "title" ("qui est esse"), and "uid" ("GFCogUyrdizw6dD221Qp").

Path	Value
race-react-firebase	posts
posts	1mNVXMNbZWjKsiaTFsDw
body	"est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui aperiam non debitis possimus qui neque nisi nulla"
createdAt	6 March 2022 at 17:13:55 UTC+1
image	"https://images.unsplash.com/photo-1642049888276-9c9f0a1a81.2.1&q=80&w=400"
title	"qui est esse"
uid	"GFCogUyrdizw6dD221Qp"

Add posts collection and a few docs



ADD DOC

```
await addDoc(postsRef, newPost);
```

[https://firebase.google.com/docs/firestore/manage-data/add-data#add a document](https://firebase.google.com/docs/firestore/manage-data/add-data#add_a_document)

Help? [react-cdn-firebase-post-app](#)



UPDATE DOC

```
await updateDoc(docRef, postToUpdate);
```

<https://firebase.google.com/docs/firestore/manage-data/add-data#update-data>

Help? [react-cdn-firebase-post-app](#)



DELETE DOC

```
// ===== DELETE =====  
function deleteUser(id) {  
  const docRef = doc(_usersRef, id);  
  deleteDoc(docRef);  
}
```

<https://firebase.google.com/docs/firestore/manage-data/delete-data>

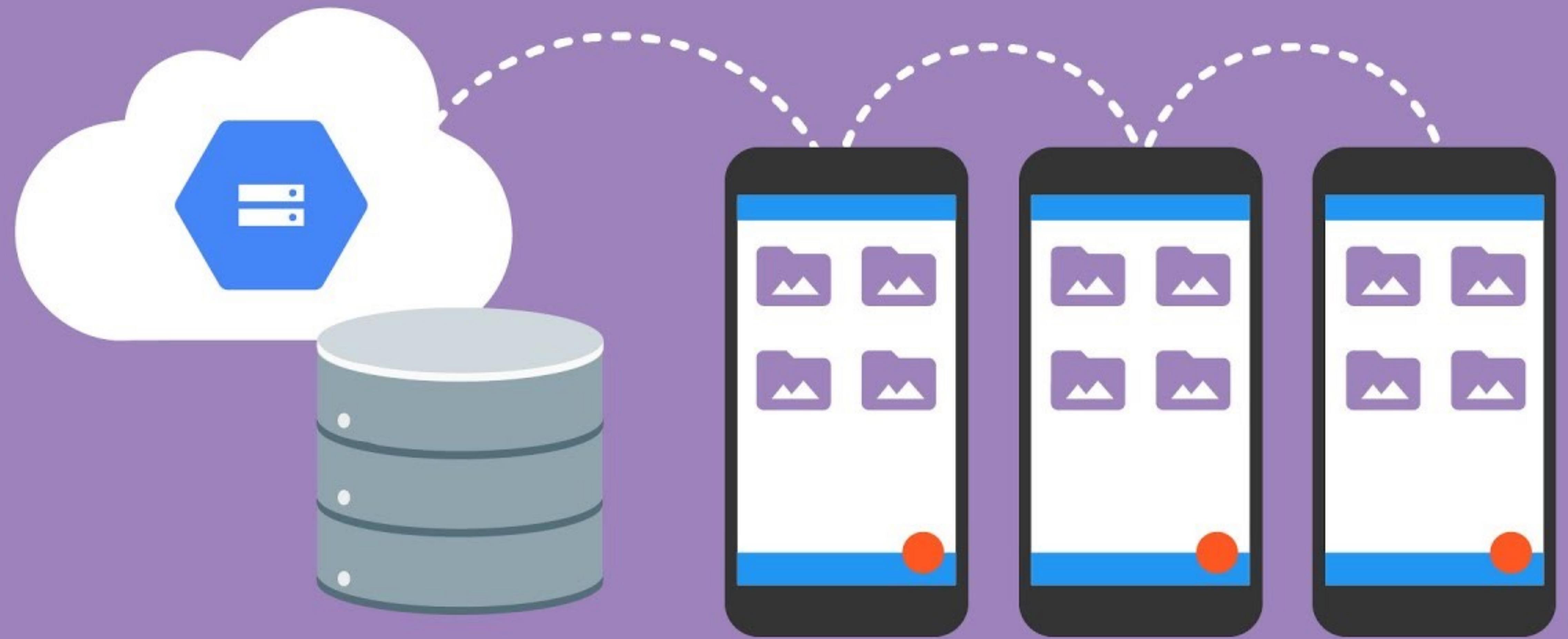
Help? [react-cdn-firebase-post-app](#)



Cloud Storage

Store and serve content, images, files, etc.

<https://firebase.google.com/products/storage>



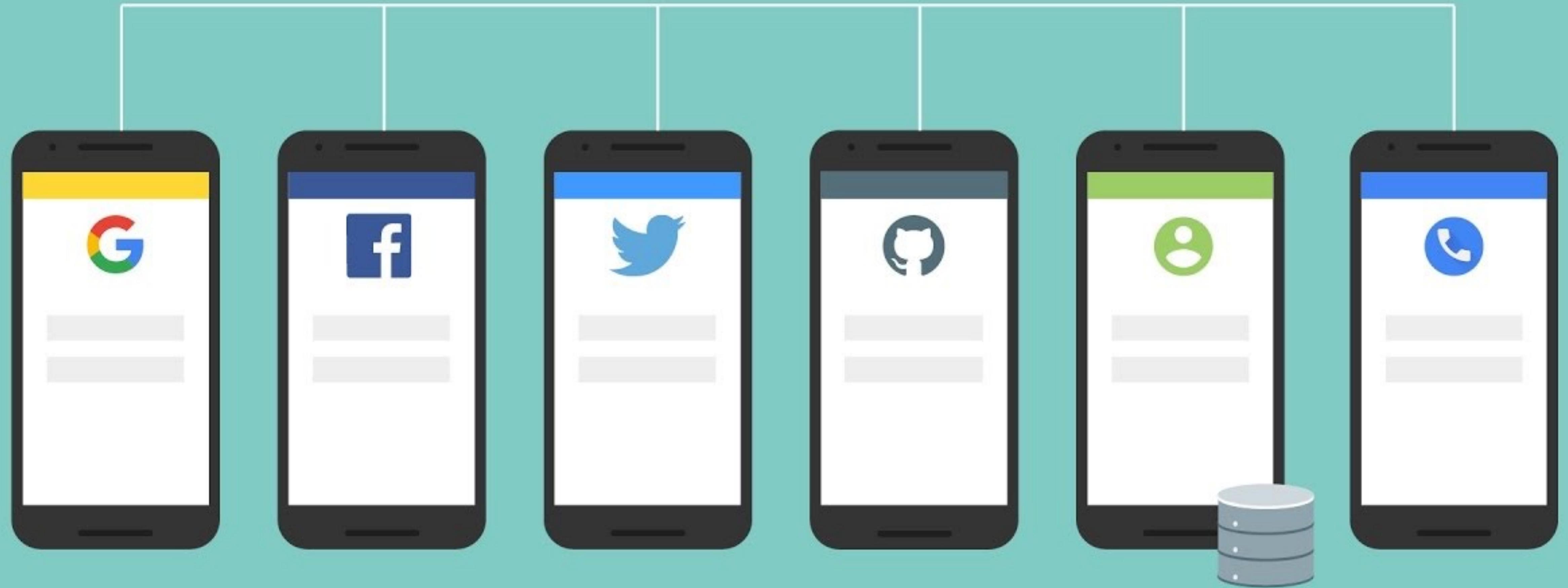
Storage



Firebase Authentication

Sign Up, Sign In and authenticate users

<https://firebase.google.com/products/auth>



Authentication



Project Overview



Develop

Authentication

Database

Storage

Hosting

Functions

ML Kit

Quality

Crashlytics, Performance, Test Lab

Analytics

Dashboard, Events, Conversions, Au...

Grow

Predictions, A/B Testing, Cloud Mes...

Spark
Free \$0/month

Upgrade

Authentication

Users Sign-in method Templates Usage

Sign-in providers

Provider	Status
Email/Password	<input checked="" type="checkbox"/> Enable
Email link (passwordless sign-in)	<input type="checkbox"/> Enable

Allow users to sign up using their email address and password. Our SDKs also provide email address verification, password recovery and email address change primitives. [Learn more](#)

Cancel Save

Phone	Disabled
Google	Disabled
Play Games	Disabled
Game Center <small>Beta</small>	Disabled
Facebook	Disabled
Twitter	Disabled
Github	Disabled



Authentication

Users Sign-in method Templates Usage

Develop



 Database



Hosting



Quality

Crashlytics, Performance, Test Lab

Analytics

Dashboard. Events. Conversions. Au...

Grow

Predictions, A/B Testing, Cloud Mes

Spark

Free \$0/month

Upgrade

Search results for users				
Identifier	Providers	Created	Signed In	User UID ↑
rasmus@cederdorff.com		9 Sep 2019	9 Sep 2019	mdAhTk3kLwhl3enYjMspg0e6g492

when you ask Rasmus
for help and he says
"Read documentation"



SET UP LOGIN

1. Explore [react-cdn-firebase-post-app-auth](#)
2. Use your own [Firebase Account \(and project\)](#) to set up login and authentication.
3. Add a [SignUpPage](#) and [SignInPage](#) component and implement:
 1. Sign up form using [createUserWithEmailAndPassword\(...\)](#) on submit.
 2. Sign In form using [signInWithEmailAndPassword\(...\)](#) on submit.
4. Implement an [Authentication state observer](#) in `App.js`.
5. Customise your [routes with public and private routes](#).
6. Make sure that you are able to sign up, log in and log out.
7. Style and implement login in your web app project (if needed).

```
function App() {
  const auth = getAuth();
  const [isAuth, setIsAuth] = React.useState(localStorage.getItem("isAuth"));

  onAuthStateChanged(auth, user => {
    if (user) {
      // User is signed in
      setIsAuth(true);
      localStorage.setItem("isAuth", true);
    } else {
      // User is signed out
      setIsAuth(false);
      localStorage.removeItem("isAuth");
    }
  });
}
```

```
return (
  <main>
    {isAuth ? (
      <>
        <Nav />
        <Routes>
          <Route path="/" element={<PostsPage />} />
          <Route path="/create" element={<CreatePage />} />
          <Route path="/profile" element={<ProfilePage />} />
          <Route path="/posts/:postId" element={<UpdatePage />} />
          <Route path="*" element={<Navigate to="/" />} />
        </Routes>
      </>
    ) : (
      <Routes>
        <Route path="/signin" element={<SignInPage />} />
        <Route path="/signup" element={<SignUpPage />} />
        <Route path="*" element={<Navigate replace to="/signin" />} />
      </Routes>
    )}
  </main>
);
}
```

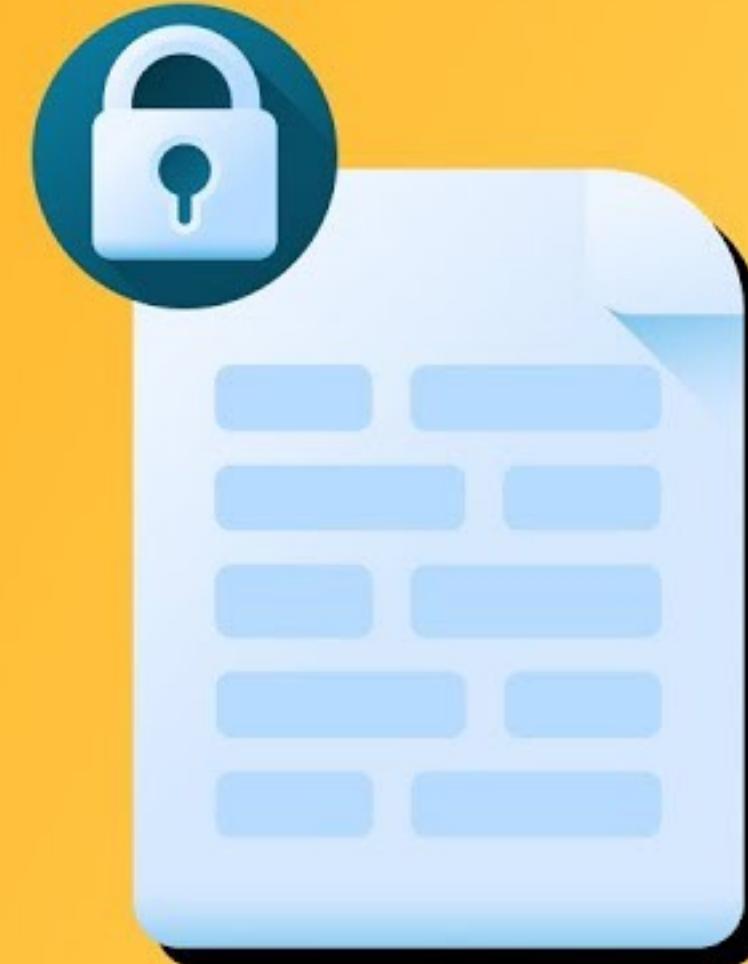
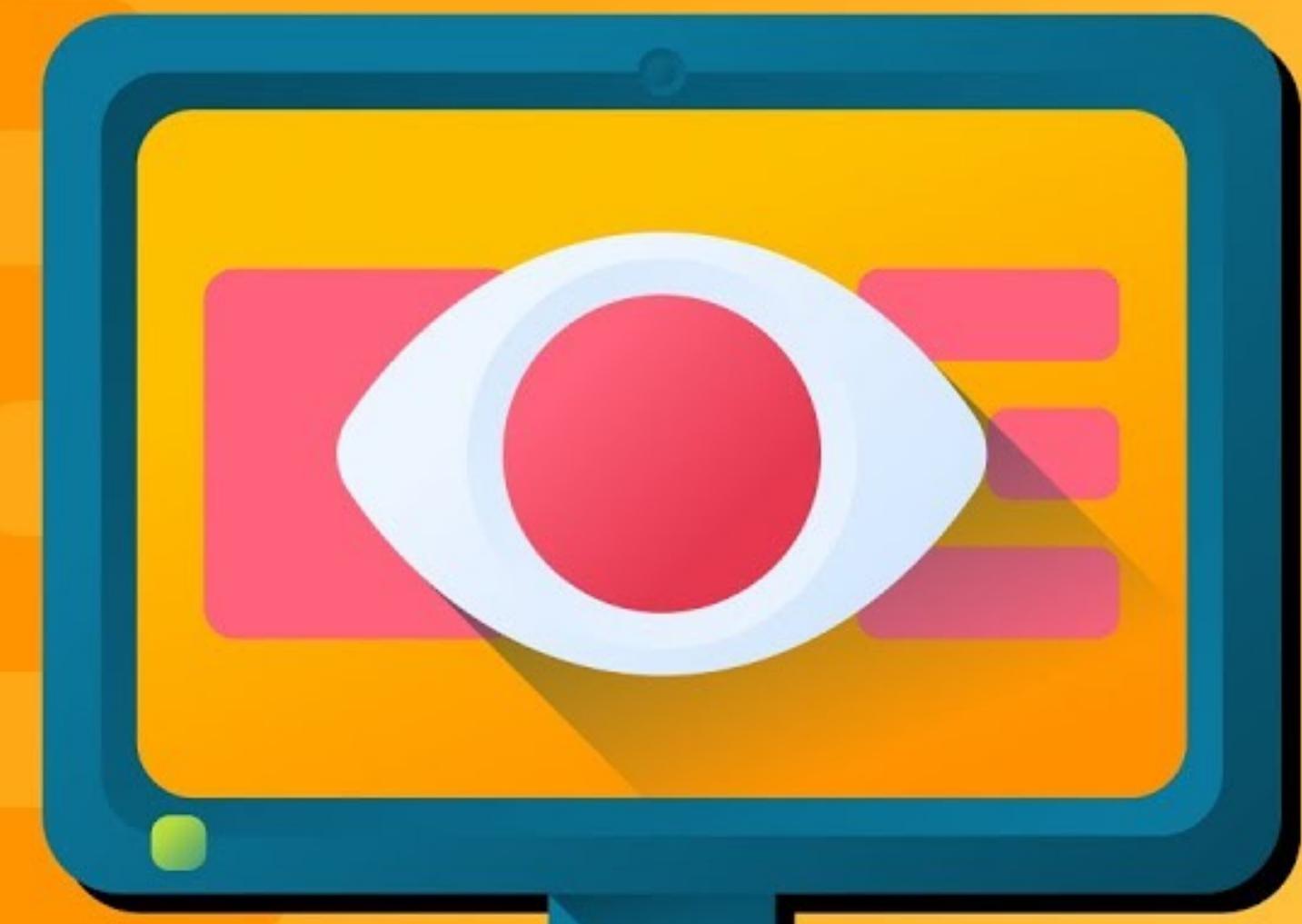


VALIDATION & ERROR HANDLING



Firebase Security Rules

100 *SECONDS OF*





Firebase Security Rules



Doug Stevenson presents:

Firebase security rules

Introduction to
Firebase security rules

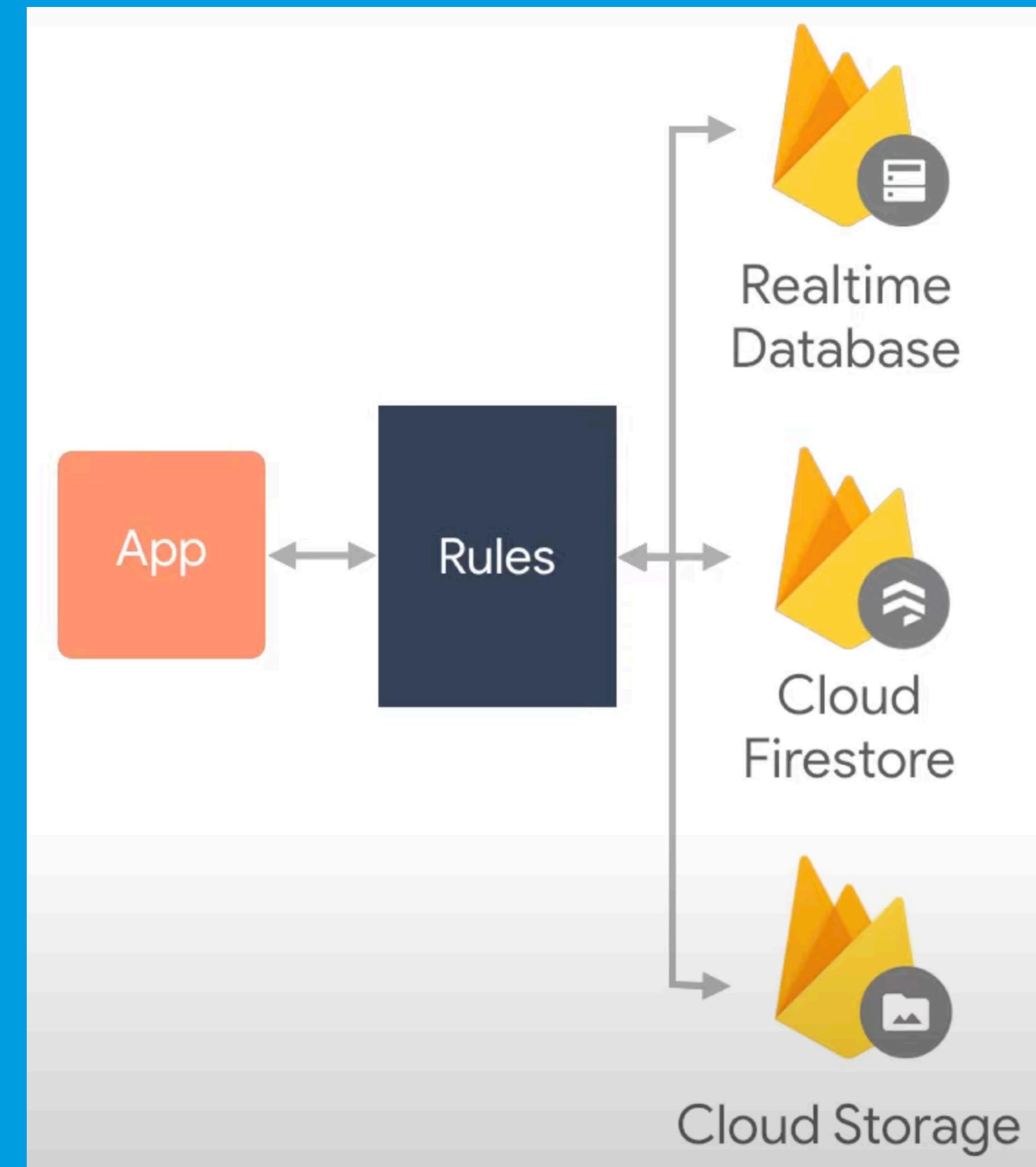


Firecasts

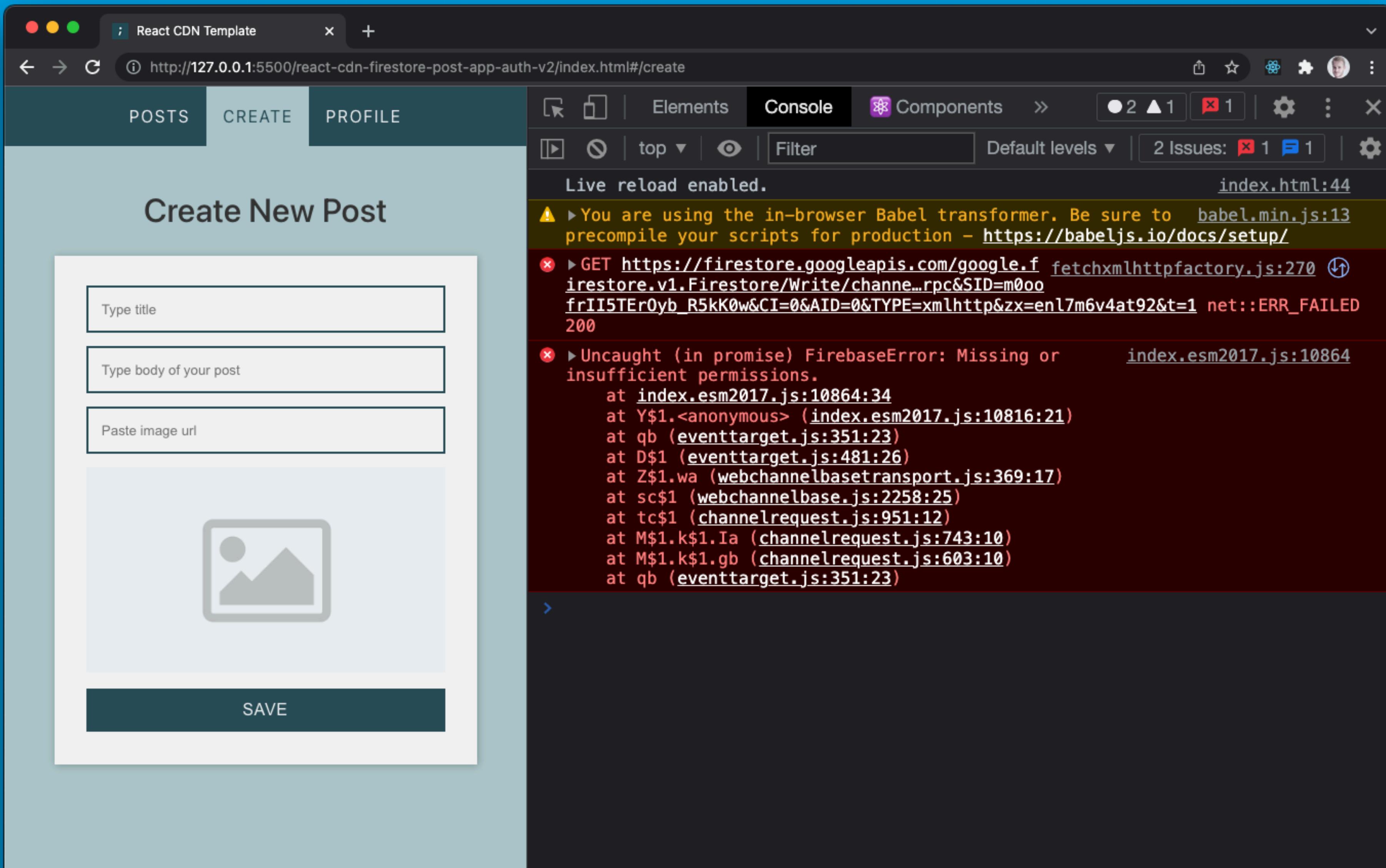
For Firebase Developers



Firebase Security Rules



SECURITY RULES



The screenshot shows a web browser window titled "React CDN Template" displaying a "Create New Post" form. The form has three input fields: "Type title", "Type body of your post", and "Paste image url". Below these is a placeholder image icon. At the bottom is a "SAVE" button. The browser's address bar shows the URL `http://127.0.0.1:5500/react-cdn-firebase-post-app-auth-v2/index.html#/create`. To the right of the browser is the developer tools interface, specifically the "Console" tab. The console output includes:

- A message: "Live reload enabled. index.html:44"
- A warning: "⚠ You are using the in-browser Babel transformer. Be sure to [babel.min.js:13](https://babeljs.io/docs/setup/) precompile your scripts for production – <https://babeljs.io/docs/setup/>"
- An error: "✖ GET https://firebase.googleapis.com/google.firebaseio/v1/Firestore/Write/channel/rpc&SID=m0oofrII5TEr0yb_R5kK0w&CI=0&AID=0&TYPE=xmlhttp&zx=enl7m6v4at92&t=1 net::ERR_FAILED 200"
- An error: "✖ Uncaught (in promise) FirebaseError: Missing or insufficient permissions. at [index.esm2017.js:10864:34](#) at [Y\\$1.<anonymous>](#) ([index.esm2017.js:10816:21](#)) at [qb](#) ([eventtarget.js:351:23](#)) at [D\\$1](#) ([eventtarget.js:481:26](#)) at [Z\\$1.wa](#) ([webchannelbasetransport.js:369:17](#)) at [sc\\$1](#) ([webchannelbase.js:2258:25](#)) at [tc\\$1](#) ([channelrequest.js:951:12](#)) at [M\\$1.k\\$1.Ia](#) ([channelrequest.js:743:10](#)) at [M\\$1.k\\$1.gb](#) ([channelrequest.js:603:10](#)) at [qb](#) ([eventtarget.js:351:23](#))

SECURITY RULES

The screenshot shows the Firebase Cloud Firestore Rules editor interface. On the left, the Firebase navigation sidebar is visible with options like Project Overview, Authentication, Firestore Database, Realtime Database, Storage, Hosting, Functions, Machine Learning, Crashlytics, Performance, Test Lab, App Distribution, Analytics, and Extensions. The main area is titled "Cloud Firestore" and has tabs for Data, Rules, Indexes, and Usage. The Rules tab is selected, showing a timeline of rule versions. The current version, edited on Mar 10, 2022 at 6:17 pm, is displayed with its code:

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read;
    }
    match /posts/{post} {
      allow delete;
      allow write: if request.resource.data.title.size() > 0
        && request.resource.data.body.size() > 0
        && request.resource.data.image.size() > 0
        && request.resource.data.uid.size() > 0
    }
    match /users/{user} {
      allow write;
    }
  }
}
```

The code uses standard JSON-like syntax for defining security rules, including `rules_version`, `service`, `match` clauses for documents and posts, and `allow` statements with specific conditions for `read` and `write` operations.

SECURITY RULES

A screenshot of a web browser window displaying the Firebase documentation for security rules conditions. The title bar shows the URL <https://firebase.google.com/docs/firestore/security/rules-conditions>. The page content includes sections on Authentication and other common patterns.

Authentication

One of the most common security rule patterns is controlling access based on the user's authentication state. For example, your app may want to allow only signed-in users to write data:

```
service cloud.firestore {  
  match /databases/{database}/documents {  
    // Allow the user to access documents in the "cities" collection  
    // only if they are authenticated.  
    match /cities/{city} {  
      allow read, write: if request.auth != null;  
    }  
  }  
}
```

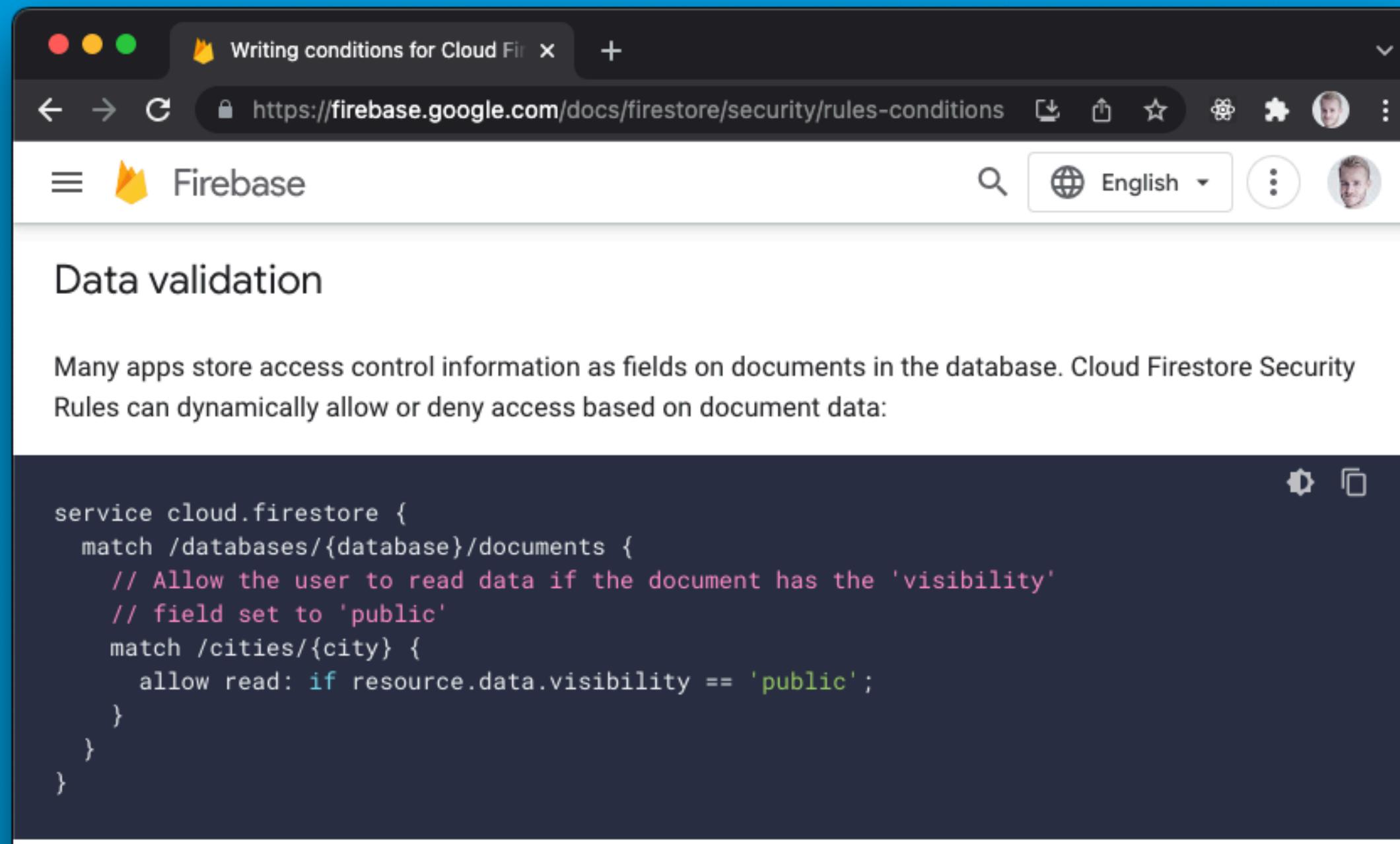
Another common pattern is to make sure users can only read and write their own data:

```
service cloud.firestore {  
  match /databases/{database}/documents {  
    // Make sure the uid of the requesting user matches name of the user  
    // document. The wildcard expression {userId} makes the userId variable  
    // available in rules.  
    match /users/{userId} {  
      allow read, update, delete: if request.auth != null && request.auth.uid == userId;  
      allow create: if request.auth != null;  
    }  
  }  
}
```

A screenshot of a code editor showing a snippet of Firebase security rules. The tab bar at the top has three options: 'Auth required' (which is selected), 'Deny all', and 'Allow all'. The code below the tabs is:

```
// Allow read/write access on all documents to any user signed in to the application  
service cloud.firestore {  
  match /databases/{database}/documents {  
    match /{document=**} {  
      allow read, write: if request.auth != null;  
    }  
  }  
}
```

SECURITY RULES



The screenshot shows a web browser window with the URL <https://firebase.google.com/docs/firestore/security/rules-conditions>. The page title is "Writing conditions for Cloud Fire". The content includes a section titled "Data validation" which explains how Cloud Firestore Security Rules can dynamically allow or deny access based on document data. It shows a code snippet for a ruleset:

```
service cloud.firestore {
  match /databases/{database}/documents {
    // Allow the user to read data if the document has the 'visibility' field set to 'public'
    match /cities/{city} {
      allow read: if resource.data.visibility == 'public';
    }
  }
}
```

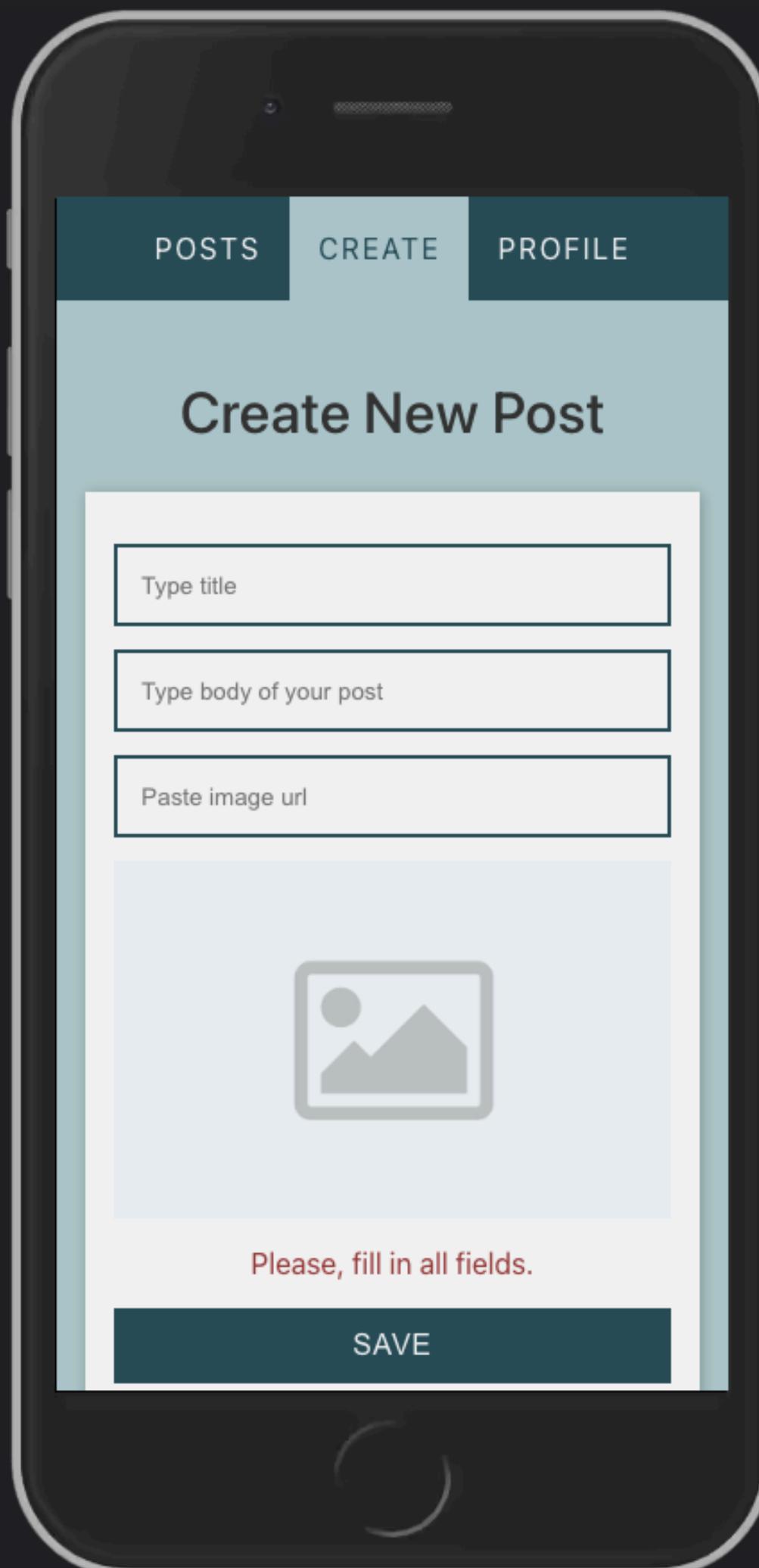
The `resource` variable refers to the requested document, and `resource.data` is a map of all of the fields and values stored in the document. For more information on the `resource` variable, see [the reference documentation](#).

When writing data, you may want to compare incoming data to existing data. In this case, if your ruleset allows the pending write, the `request.resource` variable contains the future state of the document. For `update` operations that only modify a subset of the document fields, the `request.resource` variable will contain the pending document state after the operation. You can check the field values in `request.resource` to prevent unwanted or inconsistent data updates:

```
service cloud.firestore {
  match /databases/{database}/documents {
    // Make sure all cities have a positive population and
    // the name is not changed
    match /cities/{city} {
      allow update: if request.resource.data.population > 0
        && request.resource.data.name == resource.data.name;
    }
  }
}
```

```
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read;
    }
    match /posts/{post} {
      allow delete;
      allow write: if request.resource.data.title.size() > 0
        && request.resource.data.body.size() > 0
        && request.resource.data.image.size() > 0
        && request.resource.data.uid.size() > 0
    }
    match /users/{user} {
      allow write;
    }
  }
}
```

VALIDATE FORM

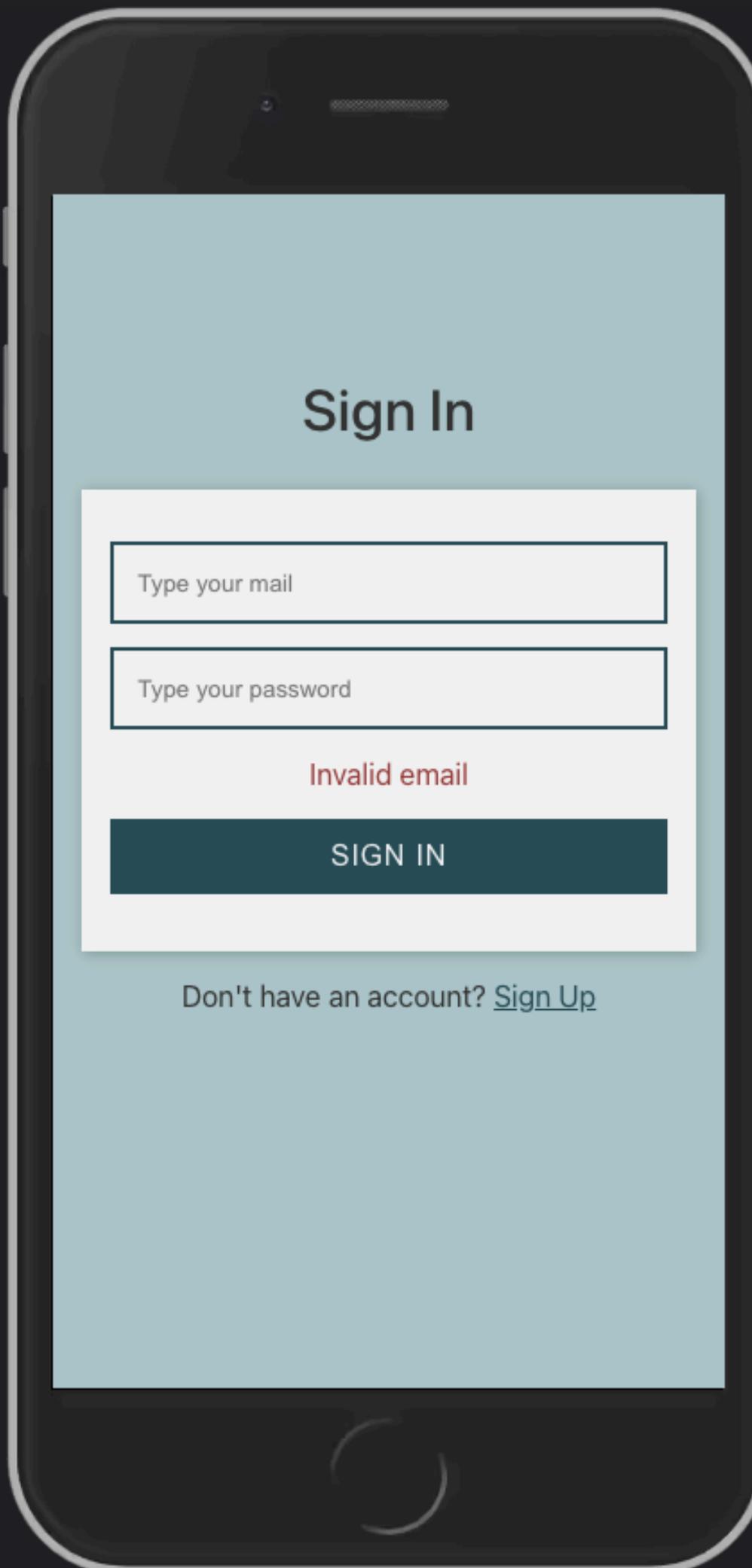


```
const [errorMessage, setErrorMessage] = React.useState("");

function submitEvent(event) {
  event.preventDefault();
  const validForm = formData.title && formData.body && formData.image;
  if (validForm) {
    handleSubmit(formData);
  } else {
    setErrorMessage("Please, fill in all fields.");
  }
}

return (
  <form onSubmit={submitEvent}>
    <input type="text" value={formData.title} onChange={handleChange}>
    <input value={formData.body} onChange={handleChange} name="body">
    <input type="url" value={formData.image} accept="image/*" onChange={handleChange}>
    <img className="image-preview" src={formData.image} alt="Choose image" />
    <p className="text-error">{errorMessage}</p>
    <button>Save</button>
  </form>
);
```

VALIDATE FORM



```
const [errorMessage, setErrorMessage] = React.useState("");

function signIn(event) {
  //...
  signInWithEmailAndPassword(auth, mail, password)
    .then(userCredential => {
      // Signed in
      const user = userCredential.user;
    })
    .catch(error => {
      let code = error.code;
      code = code.replaceAll("-", " ");
      code = code.replaceAll("auth/", "");
      setErrorMessage(code);
    });
}

return (
  <section className="page">
    <h1>Sign In</h1>
    <form onSubmit={signIn}>
      <input type="email" name="mail" placeholder="Type your mail" />
      <input type="password" name="password" placeholder="Type your password" />
      <p className="text-error">{errorMessage}</p>
      <button>Sign in</button>
    </form>
    <p className="text-center">
      Don't have an account? <a href="/signup">Sign Up</a>
    </p>
  </section>
)
```



USER MANAGEMENT IN FIREBASE



Firebase

Project Overview



mdu-e18front ▾

Go to docs



Authentication

[Users](#) [Sign-in method](#) [Templates](#) [Usage](#)

Develop

[Authentication](#)[Database](#)[Storage](#)[Hosting](#)[Functions](#)[ML Kit](#)

Quality

Crashlytics, Performance, Test Lab, ...

Analytics

Dashboard, Events, Conversions, Au...

Grow

Predictions, A/B Testing, Cloud Mes...

Extensions

Spark
Free \$0/month

Upgrade

Identifier	Providers	Created	Signed In ↓	User UID
race@eaaa.dk	✉	10 Sep 2019	12 Feb 2020	j7WsepsaogO7mvb2S35LEfdQLm...
rasmus@cederdorff.com	✉	9 Sep 2019	12 Feb 2020	mdAhTk3kLwhl3enYjMspg0e6g492
jannick_sp@live.dk	✉	27 Nov 2019	12 Dec 2019	phO2NqNruoN0k8SpeOLdpArkhB42
merylu@vp.pl	✉	12 Dec 2019	12 Dec 2019	kAHUPTEc6pWrdKtztjU3Ljf1elF3
zdfds@asdas.com	✉	12 Dec 2019	12 Dec 2019	RRZJqvB2AwS0mNnuoMyuDxbm...
try@try.com	✉	6 Dec 2019	6 Dec 2019	wCDzB33mUWdATyAprPb0dvrU6S...
eaamhoe@students.eaaa.dk	✉	27 Nov 2019	27 Nov 2019	sVvWja0amYXQdJY5DcPzgZ8BwR...
scheelfeldt.lykkegaard@gma...	✉	27 Nov 2019	27 Nov 2019	B31rv7V3ZXevbnKkmfz7sZVNZzw2
eaaklve@students.eaaa.dk	✉	27 Nov 2019	27 Nov 2019	TcKV5Hxm8FhRVlvOpUaptmVlz1
chrillbill@hotmail.com	✉	27 Nov 2019	27 Nov 2019	JPhI5uLEAlcEnclukqQY15D42TC2
eaamhoe@eaaa.dk	✉	27 Nov 2019	27 Nov 2019	2QSzDbrNJVR4bfKn1Gxlwczbpj2
christina.brandstrup@gmail....	✉	27 Nov 2019	27 Nov 2019	k1j5DJu4q6dXhb7eVwdEcWgaRq1
lyslys@hotmail.com	✉	27 Nov 2019	27 Nov 2019	hnSRfS7tZaUTZlgNmWHH6UPSx0...

[Add user](#)

M A N A G E U S E R

1. Sometimes you need to store more information (properties) to the “auth user”.
2. You have to combine the auth user with a user object in the database.
3. Explore [react-cdn-firebase-post-app-auth](#) to see how it’s done.

COMBINE POST & USER DATA

The image shows two screenshots of the Firebase console side-by-side.

Left Screenshot (posts Collection):

- Path: race-react-firebase > posts > 1mNVXMNbZWjKsiaTFsDw
- Fields:
 - body: "est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui aperiam non debitis possimus qui neque nisi nulla"
 - createdAt: 6 March 2022 at 17:13:55 UTC+1
 - image: "https://images.unsplash.com/photo-164204981.2.1&q=80&w=400"
 - title: "qui est esse"
 - uid: "GFCogUyrdizw6dD221Qp" (string)

Right Screenshot (users Collection):

- Path: race-react-firebase > users > GFCogUyrdizw6dD221Qp
- Fields:
 - image: "https://www.baaa.dk/media/hi4lv5hw/morten-anchor=center&mode=crop&width=800&height=800"
 - mail: "moab@eaaa.dk"
 - name: "Morten Algy Bonderup"
 - phone: "72286339"
 - title: "Senior Lecturer"

A thick blue arrow points from the "uid: "GFCogUyrdizw6dD221Qp" field in the left screenshot to the "uid" field in the right screenshot, indicating a reference between the two documents.

COMBINE AUTH USER & USER DATA

Identifier	Providers	Created ↓	Signed in	User UID
123@test.com	✉️	15 Mar 2022	15 Mar 2022	KKG6xEbgD1SoYru3J7gzZRFiLB83
parajuli.ajay@hotmail.com	✉️	15 Mar 2022	15 Mar 2022	OovZE13tApPJRouucPzEuAY70wq2
mail@mail.dk	✉️	15 Mar 2022	15 Mar 2022	RKYiAOI1TSasellbYzOK984naaV2
eaafgra@students.eaaa.dk	✉️	15 Mar 2022	15 Mar 2022	d9Mj03KwEihTtK5wYUnu96VGws2
medgangnu@gmail.com	✉️	15 Mar 2022	15 Mar 2022	T3eO83MdKIOssQentPIYpR9jNKS2
r@cederdorff.com	✉️	6 Mar 2022	8 Mar 2022	X3X3c44sR2Me4ZgkrlWUy9nWA...
race@eaaa.dk	✉️	6 Mar 2022	14 Mar 2022	Xq4yLFji0jQqq22sxNw6Mo9IK952

Rows per page: 50 | 1 - 7 of 7 | < >

race-react-firebase

users

+ Start collection

+ Add document

posts

users

A10r0n80qPLrqA0vJl48

GFCogUyrdizw6dD221Qp

H1vRHr58C05gu0L164k5

LWPSKpRvx7Htyjl8Jqv

SaW39AG0fU8DMUUG0FhJ

WKjy6B8IfDJxIsBbyxEe

X3X3c44sR2Me4ZgkrlWUy9nWAF3

Xq4yLFji0jQqq22sxNw6Mo9IK952

ZfPTVLMQKf9vhNiUh0bj

Si84KRoYw5pRZEWCq2Z

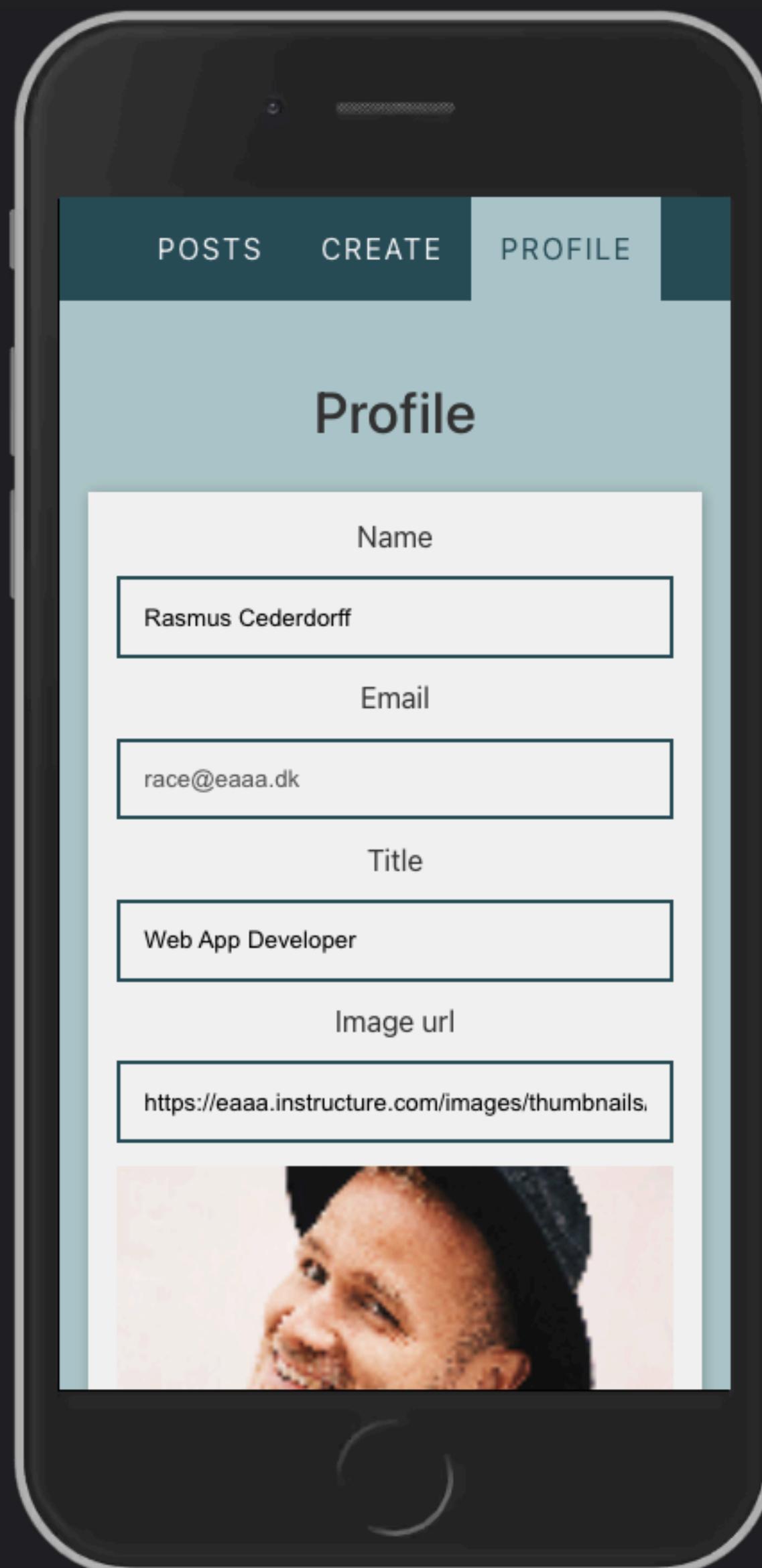
fjpRTTjZHwrq3tTLhri

pqzGY1MnHYm3I4Ca79Xn

qzeiwWfvceyfY3ntBPGv

vnH85C4TbGnSTny1Yf8I

COMBINE AUTH USER & USER DATA



```
function ProfilePage({ showLoader }) {
  const [user, setUser] = React.useState({});
  const auth = getAuth();

  React.useEffect(async () => {
    showLoader(true);

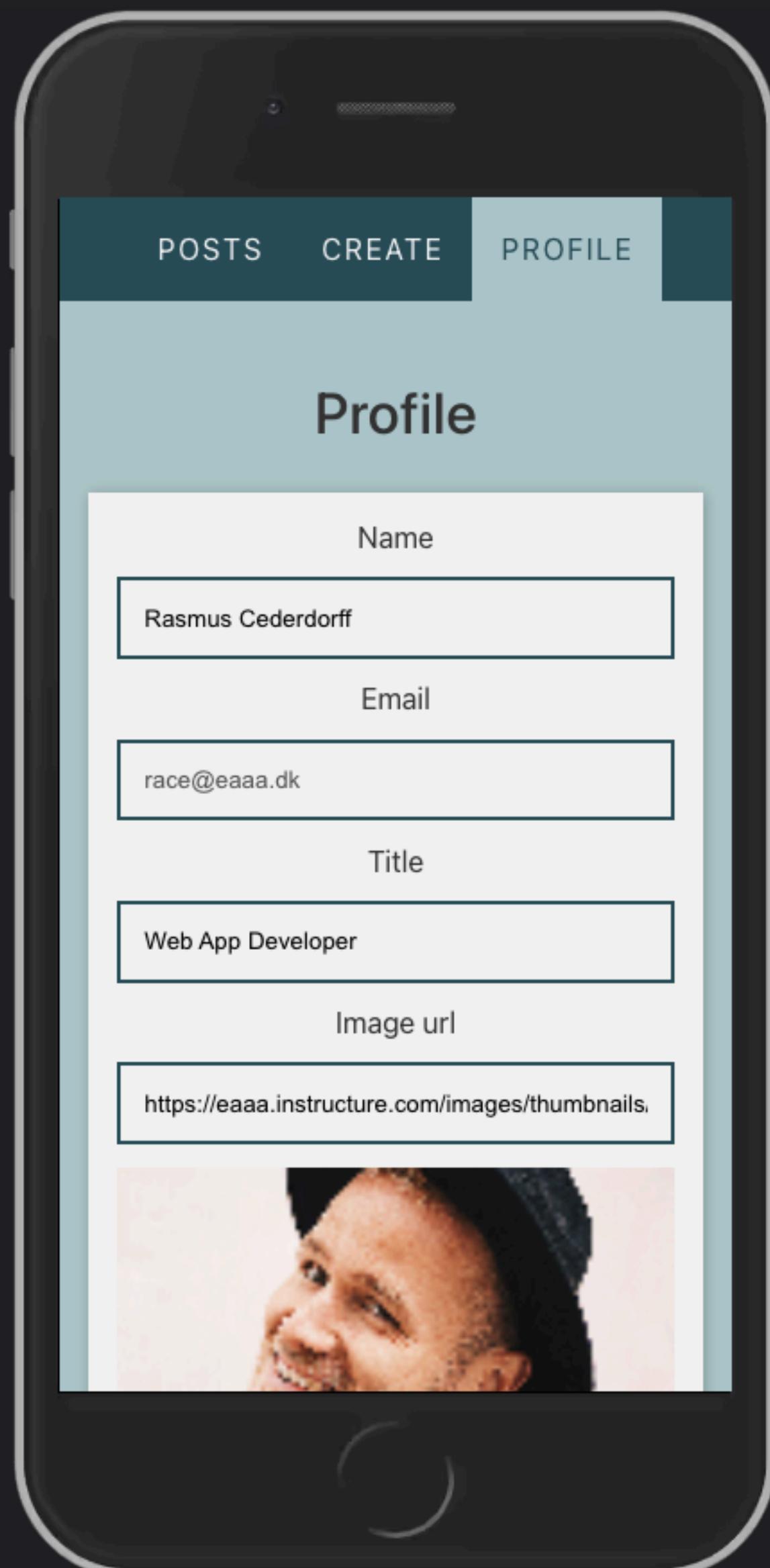
    if (auth.currentUser) {
      setUser(auth.currentUser);
    }

    const docRef = doc(usersRef, auth.currentUser.uid);
    const docSnap = await getDoc(docRef);
    if (docSnap.data()) {
      setUser(prevUser => ({ ...prevUser, ...docSnap.data() }));
    }
  }

  showLoader(false);
}, [auth.currentUser]);
```

Concatenating auth user with user data from users collection

COMBINE AUTH USER & USER DATA



```
function ProfilePage({ showLoader }) {
  const [user, setUser] = React.useState({});
  const auth = getAuth();

  ...

  async function submitEvent(event) {
    event.preventDefault();
    showLoader(true);

    const userToUpdate = { name: user.name, title: user.title, image:
      console.log(userToUpdate);
      const docRef = doc(usersRef, user.uid);

      await setDoc(docRef, userToUpdate);
      showLoader(false);
    }
  }
}
```

Setting user data in
users collection
using auth user id



ORDER & SORT

PERFORM SIMPLE AND COMPOUND QUERIES IN CLOUD FIRESTORE
<https://firebase.google.com/docs/firestore/query-data/queries>

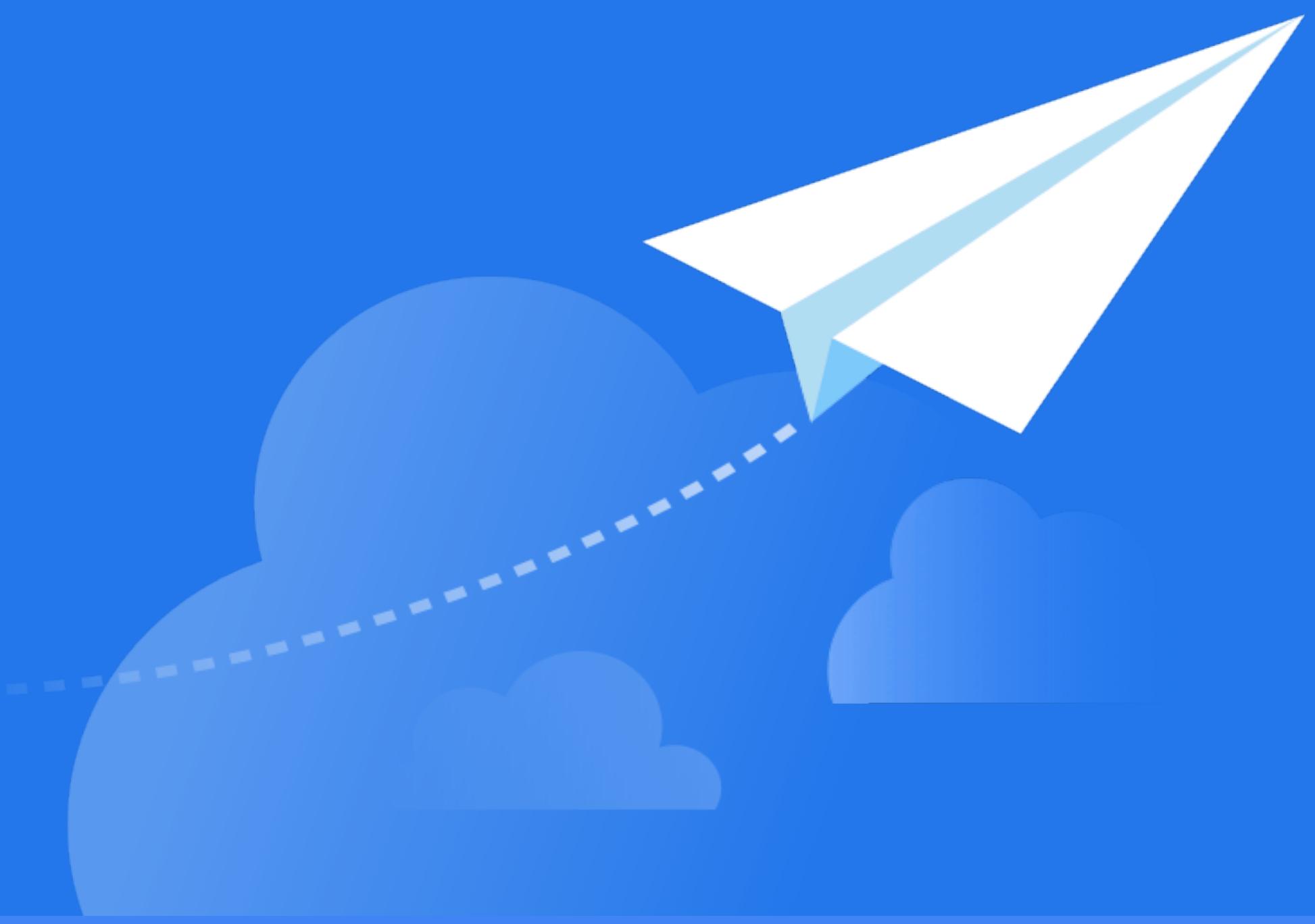
ORDER AND LIMIT DATA WITH CLOUD FIRESTORE
<https://firebase.google.com/docs/firestore/query-data/order-limit-data>

PAGINATE DATA WITH QUERY CURSORS
<https://firebase.google.com/docs/firestore/query-data/query-cursors>

```
const q = query(postsRef, orderBy("createdAt", "desc"));
onSnapshot(q, data => {
});
```

```
const q = query(postsRef, where("uid", "==", "Xq4yLFji0jQqq22sxNw6Mo9IK952"));
onSnapshot(q, data => {
});
```

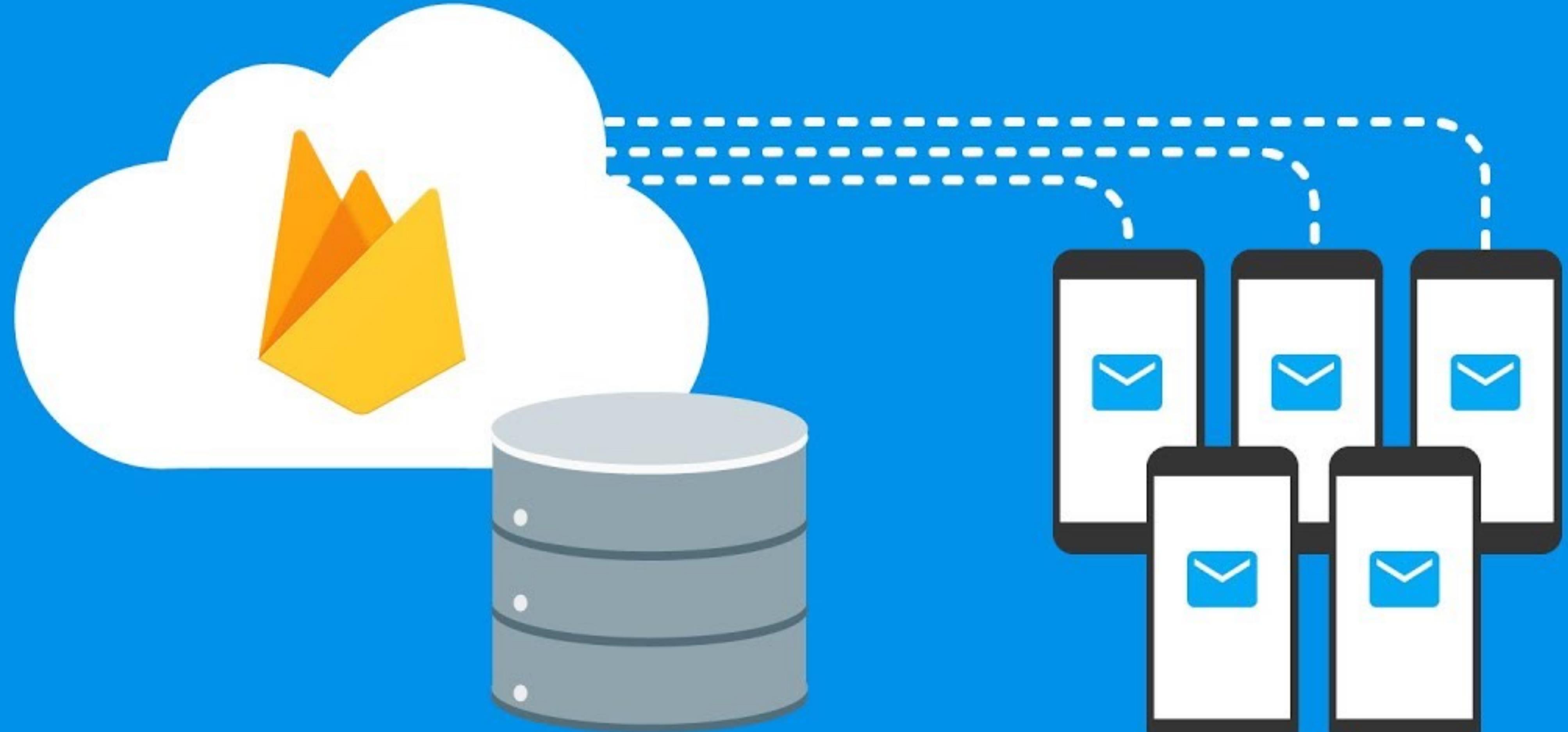
https://firebase.google.com/docs/firestore/query-data/queries#simple_queries



Cloud Messaging

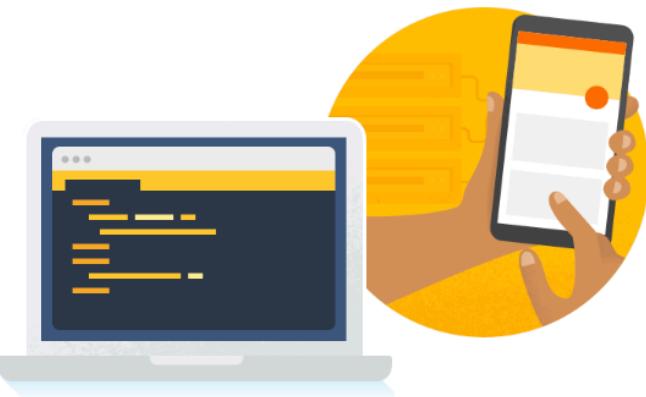
Send notifications across platforms

<https://firebase.google.com/products/cloud-messaging>



Cloud Messaging

WHAT ELSE?



Build better apps



Cloud Firestore

Store and sync app data at global scale



ML Kit BETA

Machine learning for mobile developers



Cloud Functions

Run mobile backend code without managing servers



Authentication

Authenticate users simply and securely



Hosting

Deliver web app assets with speed and security



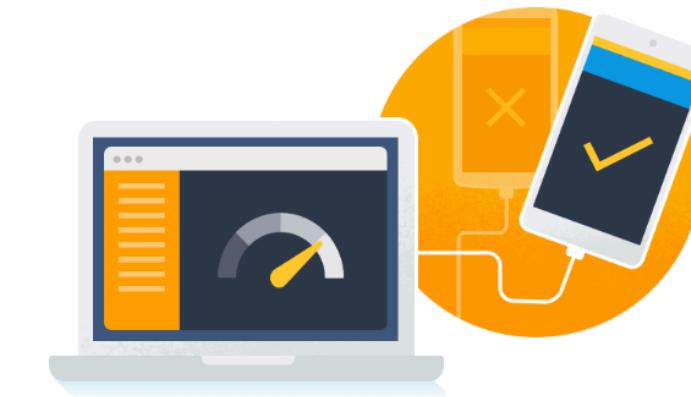
Cloud Storage

Store and serve files at Google scale



Realtime Database

Store and sync app data in milliseconds



Improve app quality



Crashlytics

Prioritize and fix issues with powerful, realtime crash reporting



Performance Monitoring

Gain insight into your app's performance



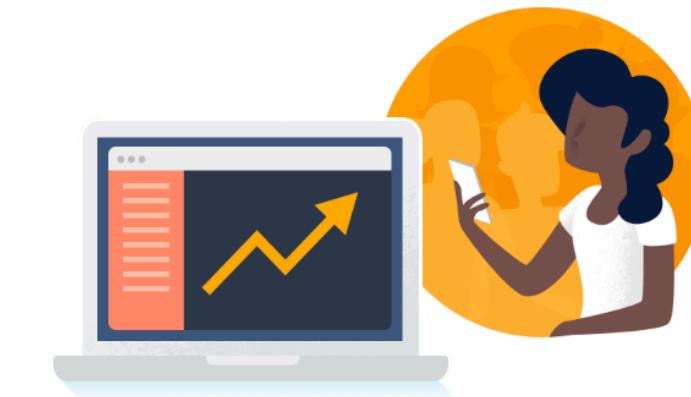
Test Lab

Test your app on devices hosted by Google



App Distribution BETA

Distribute pre-release versions of your app to your trusted testers



Grow your business



In-App Messaging BETA

Engage active app users with contextual messages



Google Analytics

Get free and unlimited app analytics



Predictions

Smart user segmentation based on predicted behavior



A/B Testing BETA

Optimize your app experience through experimentation



Cloud Messaging

Send targeted messages and notifications



Remote Config

Modify your app without deploying a new version



Dynamic Links

Drive growth by using deep links with attribution



Code
Every
Day