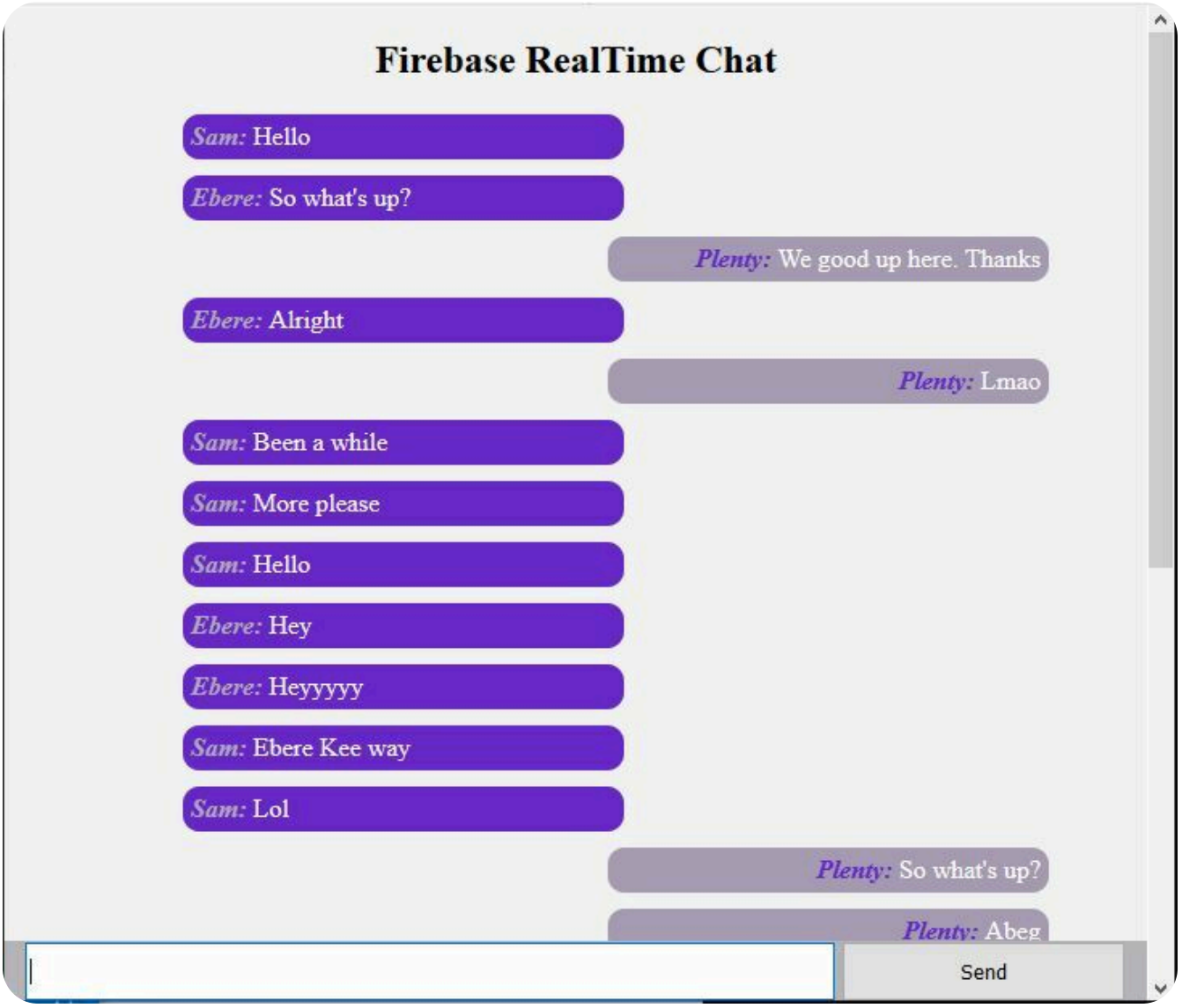
How to Build a Chat App with Firebase

This tutorial will show you how to build your own chat application using Firebase's Realtime Database.

Njoku Samson Ebere • Mar 22, 2021

Have you thought about how difficult it is to build a chat application, showing you messages as soon as they are sent? In this tutorial we will do just that. I will guide you through building a web based chat application using Firebase as a realtim database and some simple HTML, CSS and Javascript for the interface.You ca view the source code for the application on Github.

Note: If you're looking for a chat application but don't want to build your own, you should consider CometChat. Our robust suite of clou Login Schedule a demo ice & video solutions, ranging from simple drag-and-drop plugins to UI Kits, APIs and fully customizable SDKs, plus a host of unique ready-to-use extensions, will seamlessly integrate into your website & apps, quickly and securely; saving you countless hours & resources, and dramatically growing engagement on your website. Give us a try for free today!



What You Will Need

You will need a basic understanding of HTML, CSS and JavaScript as well as a google account to use with Firebase.

What is Firebase?

Firebase is a back-end platform provided by Google for building full-stack applications. It provides programmers with authentication options, storage, databases, hosting, A/B testing and other services.

Firebase helps you to focus on developing the front-end of your applications while it does the hidden jobs for you.

Setting Up Firebase

Firebase offers two types of databases (Cloud Firestore and Realtime Database). For the his tutorial, you will be using a Realtime Database.

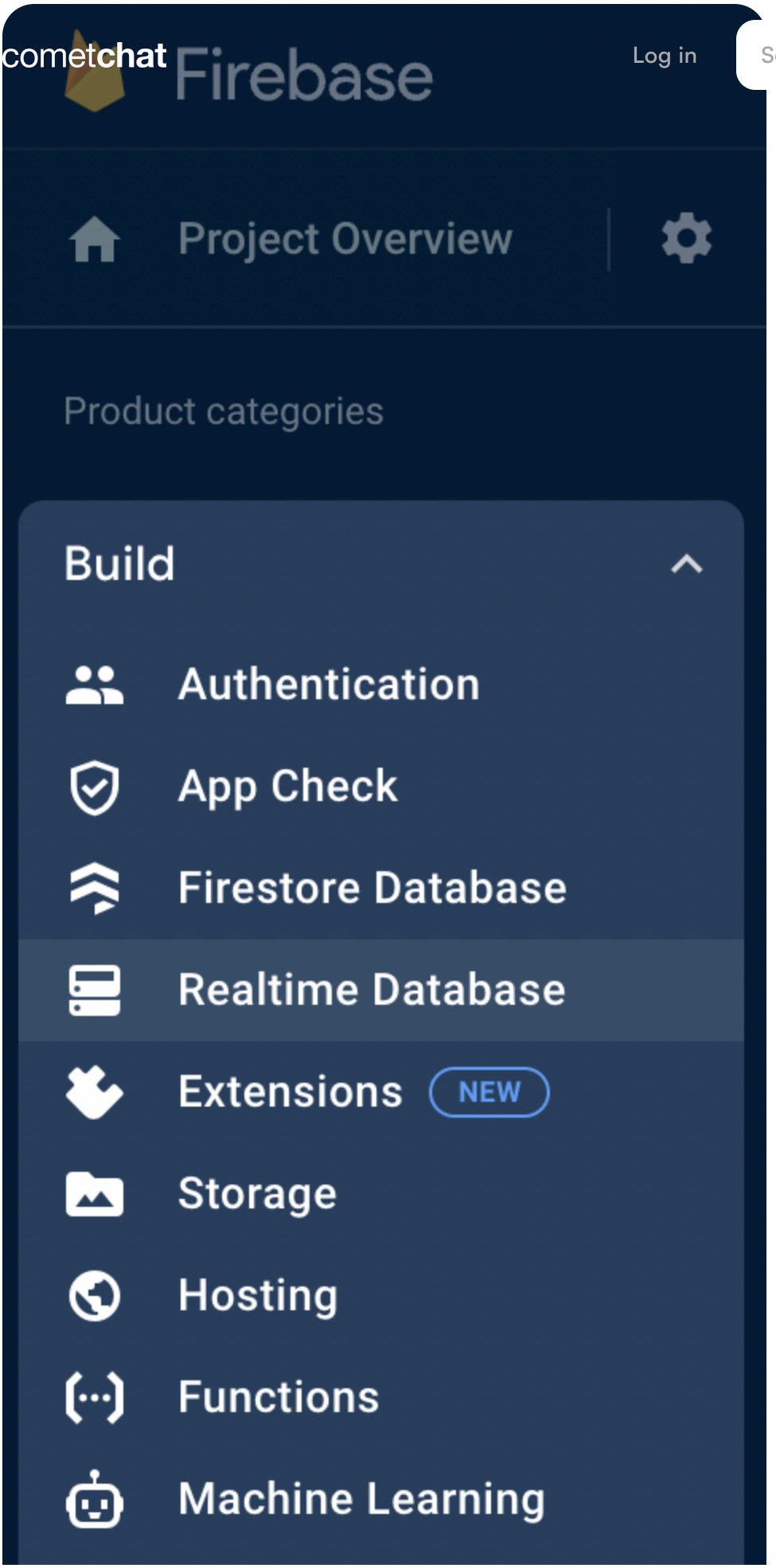
You will need to create a new Firebase project:

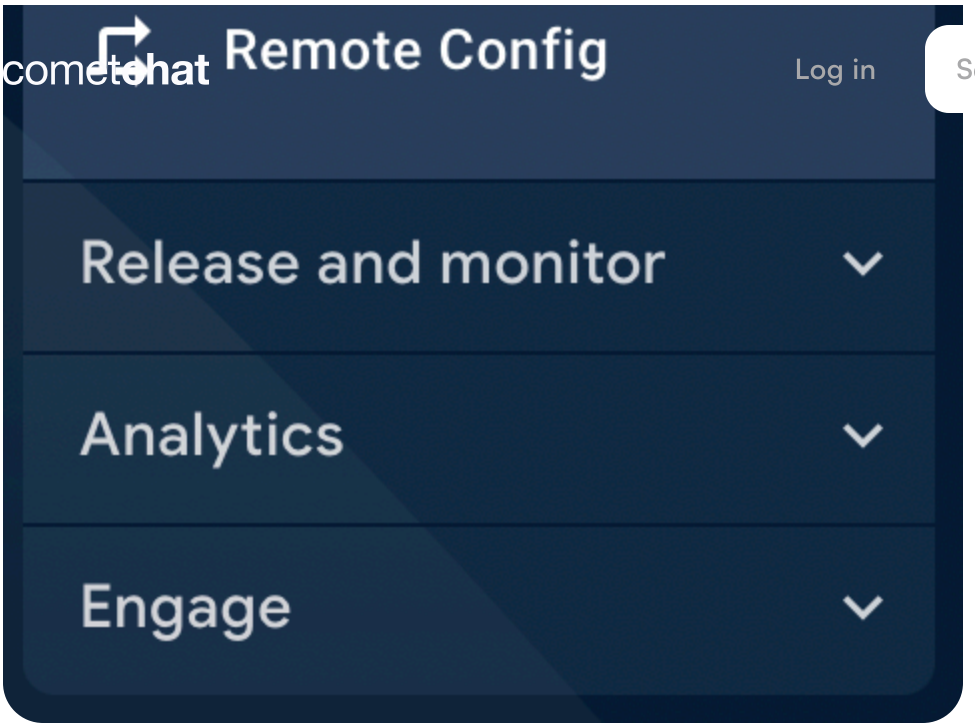
- Go to the Firebase Console
- Create a new project

Create a Realtime Database

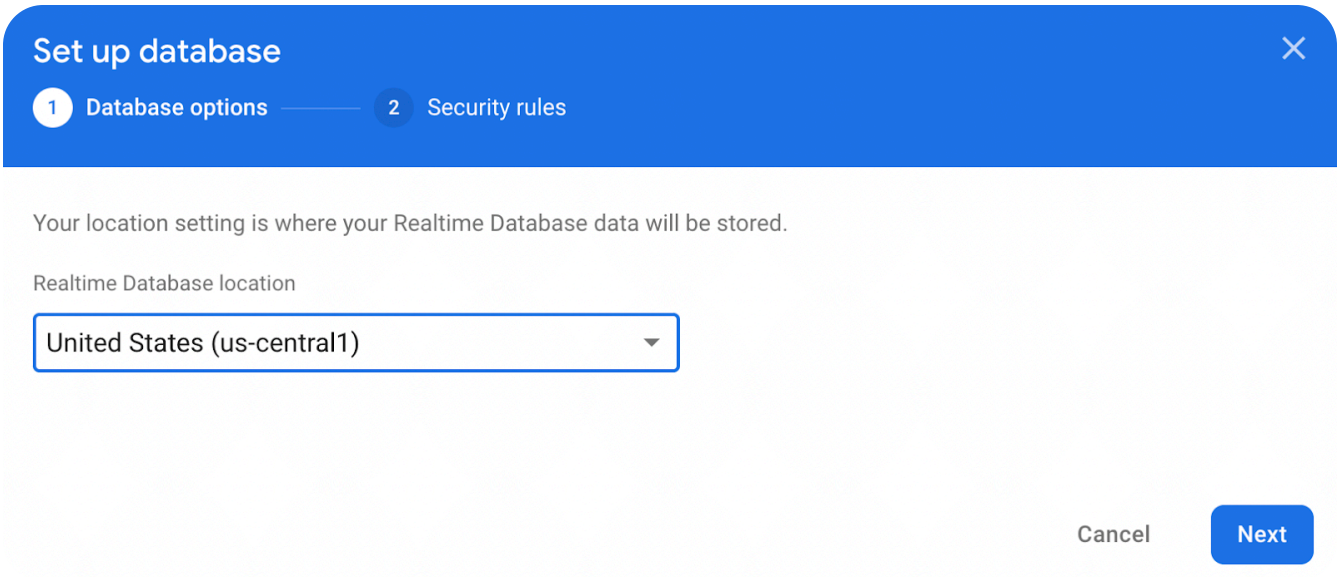
In the sidebar on the left

- Open the '**Build**' menu
- Click on '**Realtime Database**' and then '**Create Database**'

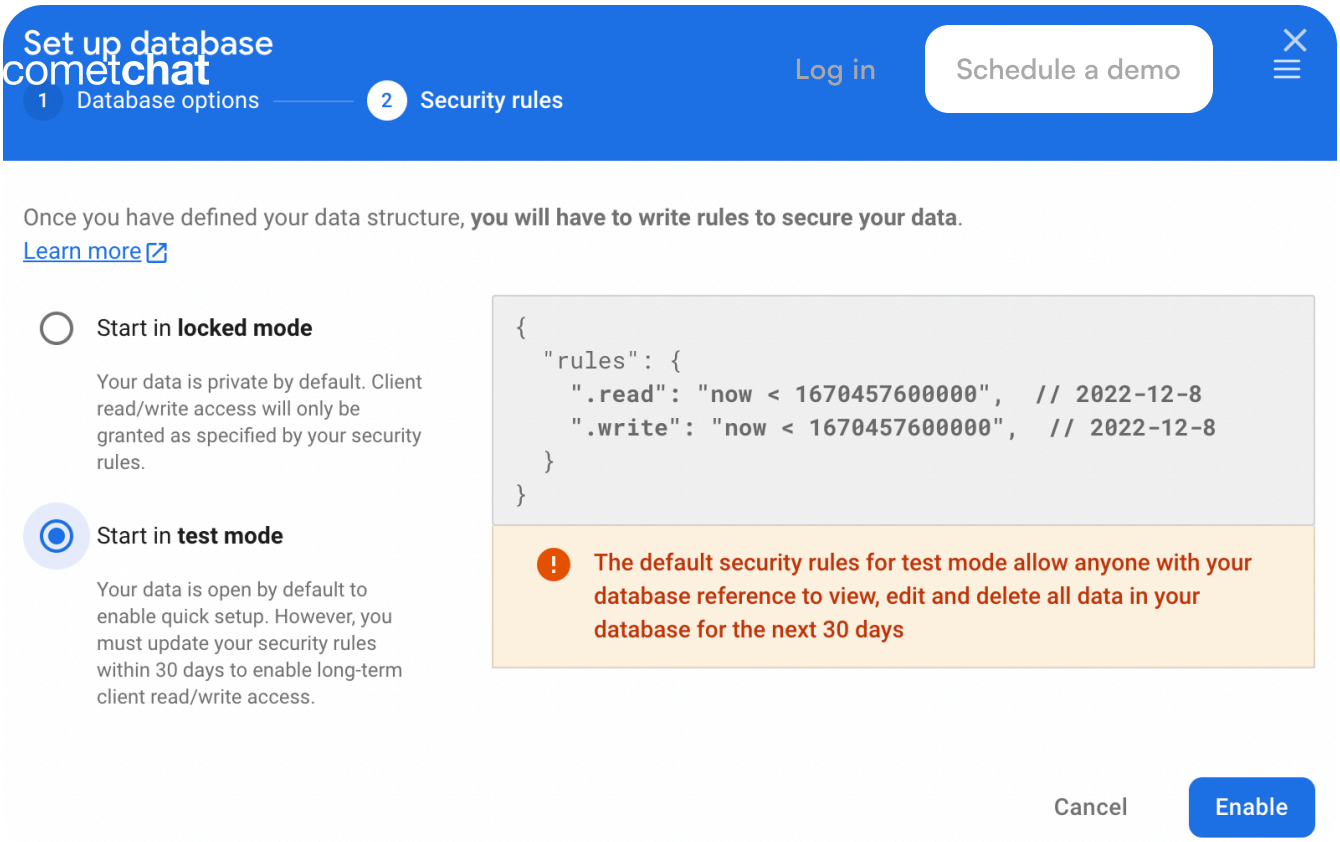

cometchat
Firebase
Log in
Schedule a demo
Project Overview
Product categories
Build
Authentication
App Check
Firestore Database
Realtime Database
Extensions
NEW
Storage
Hosting
Functions
Machine Learning



After that, select the region you want to use



Finally, start the database in '**test mode**'

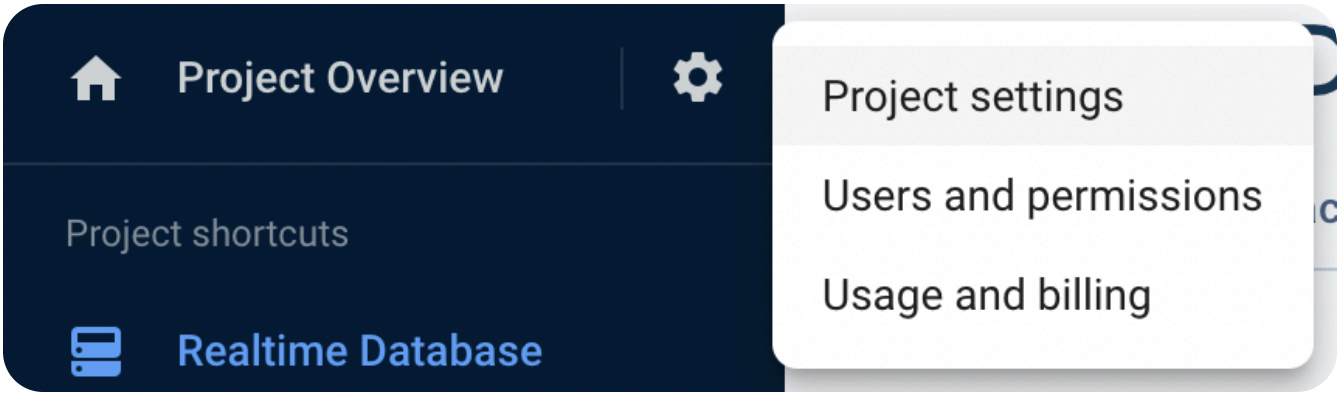


Get Your Configuration Details

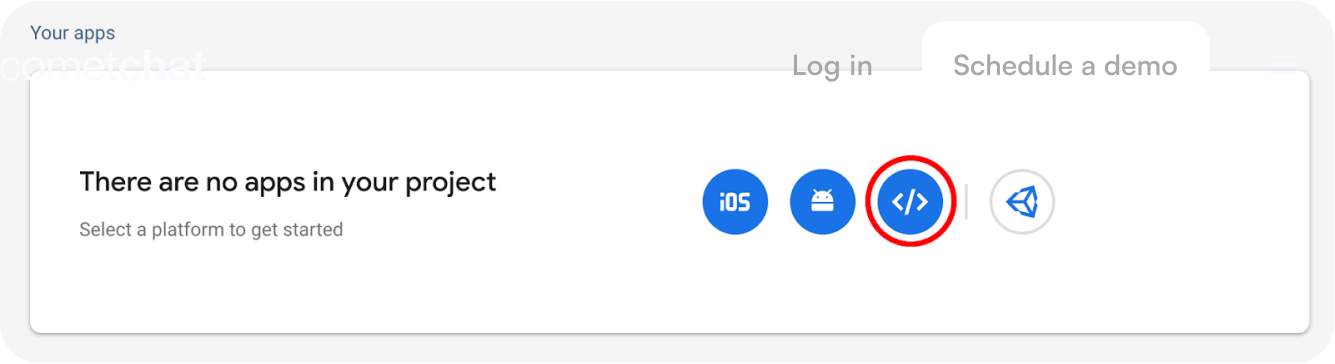
Once your database is setup, the only thing left to do is to get your configuration details. You will use these in order to authenticate your application with the Firebase service.

You can find your configuration details in the Firebase console:

- Click the Gear icon next to 'Project Overview'
- Choose '**Project Settings**'



Under ‘Your apps’, select ‘Web app’:

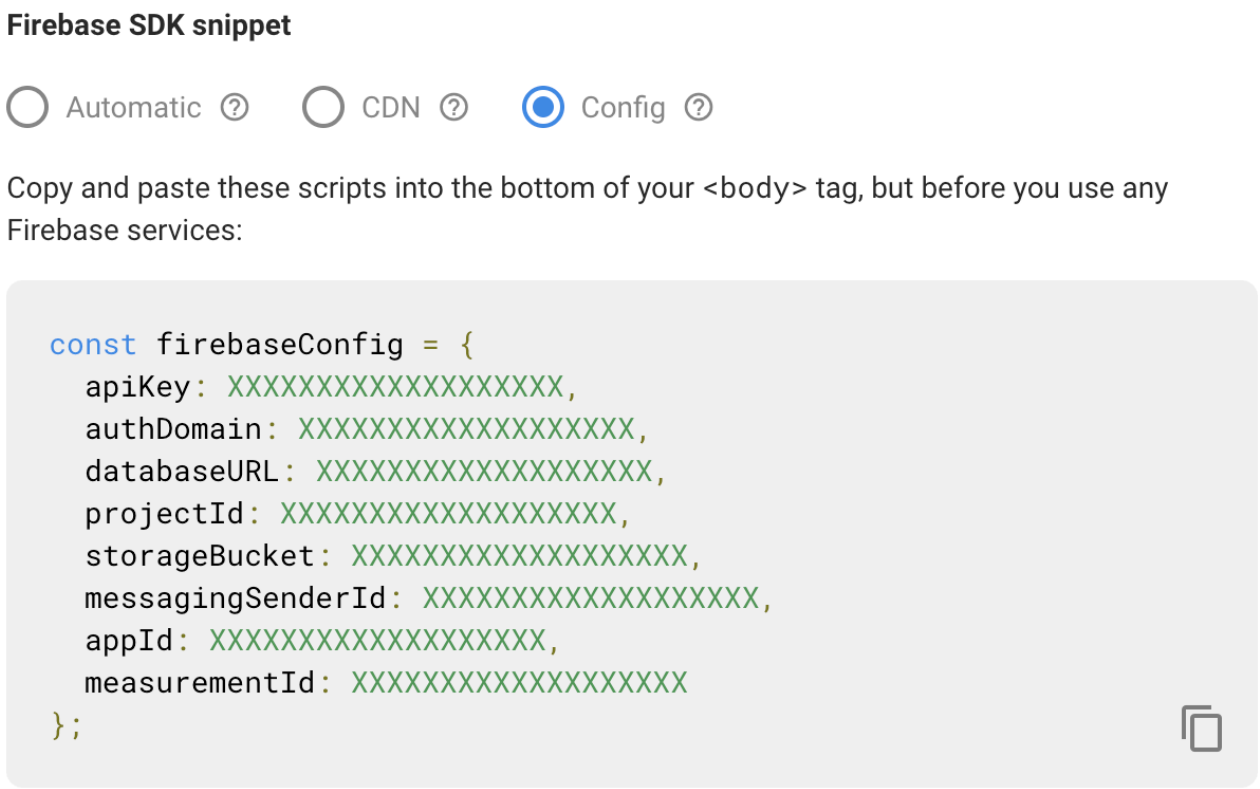


Next you need to register your application:

- Enter a name for your application
- Click on '**Register App**'

You can ignore the next screen and click on '**Continue to Console**'

Your application's config values will then be visible in the 'Your apps' section



Project Setup

You project will consist of 3 files, create a folder with the 3 files named:

- index.html
- index.css
- index.js

Laying The Foundation

The index.html file will contain the HTML for your application as well as links to the index.css, and index.js files, as well as the Firebase SDK.

Add the following code to your **index.html** file:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <title>Firebase RealTime Chat</title>
    <link rel="stylesheet" href="./index.css">
  </head>
  <body>

    <!-- scripts -->
    <script src="https://www.gstatic.com/firebasejs/8.2.1/firebase-app.js"></script>
    <script src="https://www.gstatic.com/firebasejs/8.2.1/firebase-database.js"></script>
    <script src="index.js"></script>
  </body>
</html>
```

Notice that the script tags for the Firebase SDKs and your index.js file are included in the *<body>* element. These load the Firebase SDK and make it available to your application. The index.css file is also included in the *<head>* element and will be used to style the application.

Next, Add the following content to the *<body>* element above the *<script>* tags:

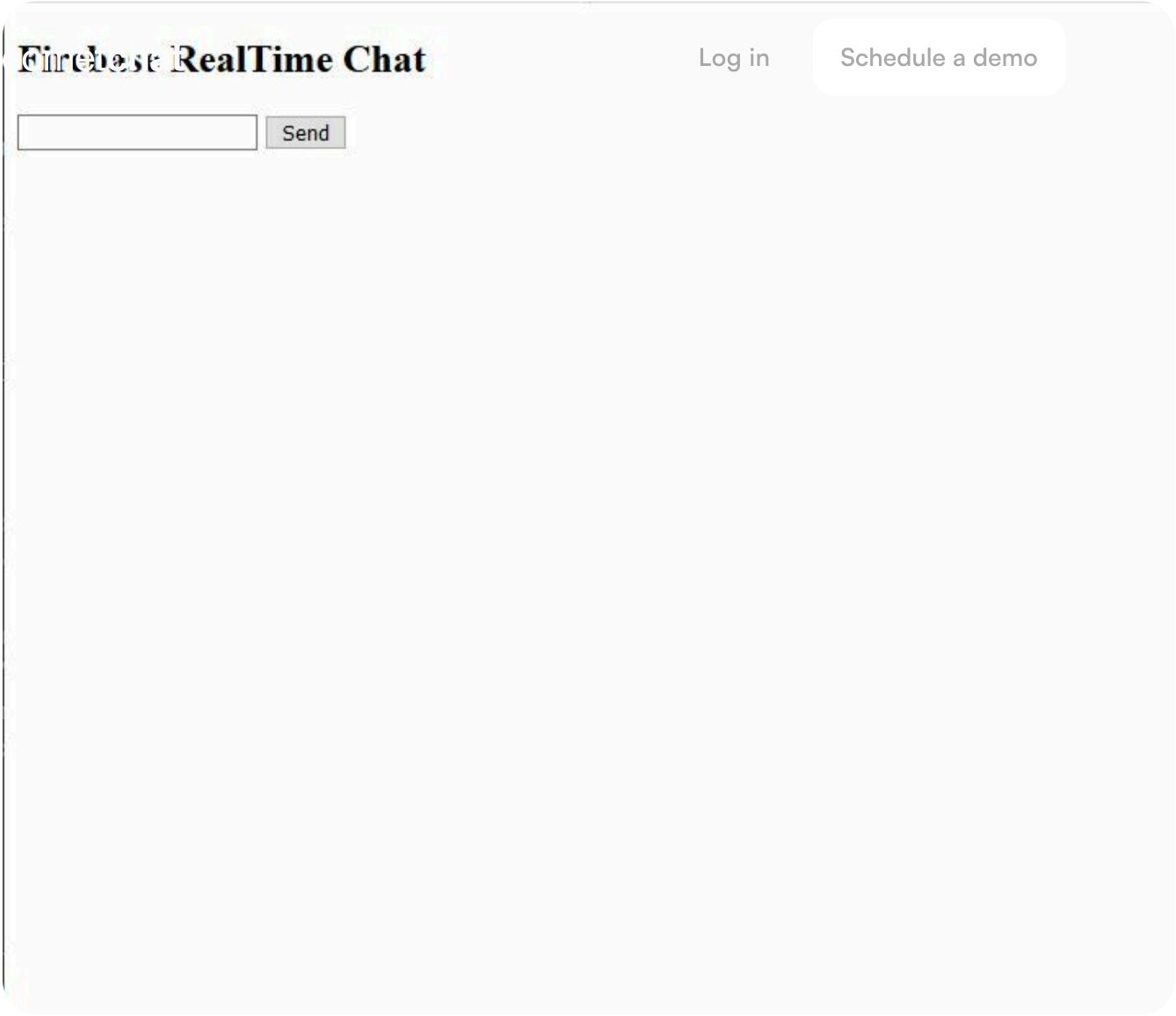
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width,
initial-scale=1" />
    <title>Firebase RealTime Chat</title>
    <link rel="stylesheet" href="./index.css">
  </head>
  <body>
    <header>
      <h2>Firebase RealTime Chat</h2>
    </header>

    <div id="chat">
      <!-- messages will display here -->
      <ul id="messages"></ul>

      <!-- form to send message -->
      <form id="message-form">
        <input id="message-input" type="text" />
        <button id="message-btn" type="submit">Send</button>
      </form>
    </div>
    <!-- scripts -->
    <script
src="https://www.gstatic.com/firebasejs/8.2.1/firebase-
app.js"></script>
    <script
src="https://www.gstatic.com/firebasejs/8.2.1/firebase-
database.js"></script>
    <script src="index.js"></script>
  </body>
</html>
```

In the HTML above, you created a *<header>* with the name of your app. The *<ul>* element will be used to to display messages loaded from the server while the *<form>* element will enable users to submit their messages.

If you open your application in a browser now, it should look something like this:



Adding Functionality

You now need to open the index.js file. This is where we will build out the functionality of our application step by step.

Initializing the Application

You need to tell your application how to connect to the Realtime Database we created and initialize the Firebase application.

Copy the *firebaseConfig* for your application from the Firebase Project Console and paste it into your index.js file. Then add the following lines of code:

```
var firebaseConfig = {
  apiKey: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  authDomain: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  databaseURL: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
```

```
  projectId: "xxx-xxx-xxx-xxx-xxx-xxx-xxx"
  storageBucket: "xxx-xxx-xxx-xxx-xxx-xxx
  messagingSenderId: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  appId: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
};

firebase.initializeApp(firebaseConfig);

const db = firebase.database();

const username = prompt("Please Tell Us Your Name");
```


Sending Messages

In order to send messages, users will type into the input form provided and click on the ‘Send’ button. Your app will subscribe to the *submit* event when the ‘Send’ button is clicked and then call the *sendMessage()* function.

The *sendMessage()* function will:

- Prevent the default form behaviour
- Capture and store the message sent by the user
- Clear the input box
- Scroll to the bottom of the page where the new message will appear
- Add the message to the database

Edit your index.js file to contain the following code:

```
var firebaseConfig = {
  apiKey: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  authDomain: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  databaseURL: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  projectId: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  storageBucket: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  messagingSenderId: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  appId: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
};
```

```
firebase.initializeApp(firebaseConfig);

const db = firebase.database();

const username = prompt("Please Tell Us Your Name");

function sendMessage(e) {
  e.preventDefault();

  // get values to be submitted
  const timestamp = Date.now();
  const messageInput = document.getElementById("message-input");
  const message = messageInput.value;

  // clear the input box
  messageInput.value = "";

  //auto scroll to bottom
  document
    .getElementById("messages")
    .scrollIntoView({ behavior: "smooth", block: "end", inline: "nearest" });

  // create db collection and send in the data
  db.ref("messages/" + timestamp).set({
    username,
    message,
  });
}
```

Receiving Text Messages

When a message is sent by one user it needs to be received by the other users of the application.

Edit your index.js file to contain the following code:

```
var firebaseConfig = {
```

```
    apiKey: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xx"
    authDomain: "xxx-xxx-xxx-xxx-xxx-xxx-xx
    databaseURL: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
    projectId: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
    storageBucket: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
    messagingSenderId: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
    appId: "xxx-xxx-xxx-xxx-xxx-xxx-xxx-xxx",
  };

  firebase.initializeApp(firebaseConfig);

  const db = firebase.database();

  const username = prompt("Please Tell Us Your Name");

  function sendMessage(e) {
    e.preventDefault();

    // get values to be submitted
    const timestamp = Date.now();
    const messageInput = document.getElementById("message-
  input");
    const message = messageInput.value;

    // clear the input box
    messageInput.value = "";

    //auto scroll to bottom
    document
      .getElementById("messages")
      .scrollIntoView({ behavior: "smooth", block: "end",
  inline: "nearest" });

    // create db collection and send in the data
    db.ref("messages/" + timestamp).set({
      username,
      message,
    });
  }

```

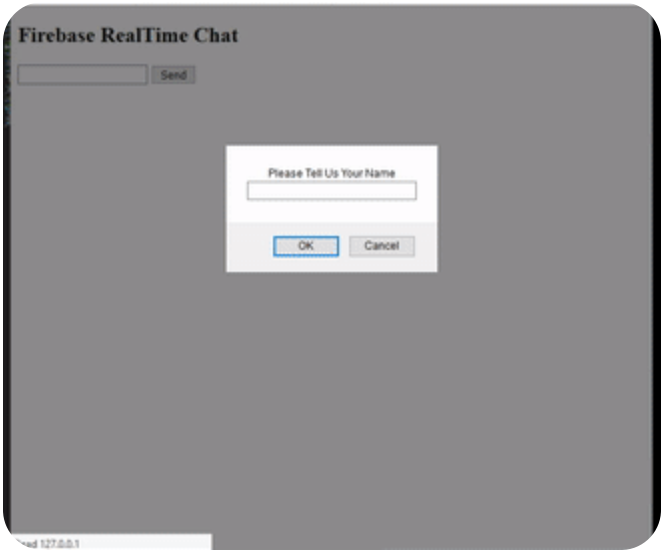
```
const fetchChat = db.ref("messages/");

fetchChat.on("child_added", function (snapshot) {
  const messages = snapshot.val();
  const message = `<li class=${
    username === messages.username ? "sent" : "receive"
  }><span>${messages.username}: </span>${messages.message}
</li>`;
  // append the message on the page
  document.getElementById("messages").innerHTML += message;
});
```

In the code above, you subscribe to the database's *child_added* event in order to be notified when new messages are added to the database. When a new message is added, it is appended to the <ul> element you created earlier as a new list item.

At this point, if you open your application in a browser it should be possible to:

- Specify a username
- Send and receive messages



Chat App Demo Without CSS

Styling The App

To style your application you need to add some CSS to the index.css file. Feel free to copy and paste the styles below to get you started:

```
body {
    background-color: #f3f2f3;
    margin: 0;
    padding: 0;
}

header {
    text-align: center;
}

#messages{
    padding-bottom: 30%;
}

li {
    list-style-type: none;
    margin-bottom: 10px;
    background-color: #6929ca;
    padding: 5px;
    border-radius: 10px;
    color: white;
    width: 50%;
}

li span {
    font-style: italic;
    font-weight: bolder;
    color: #b5b0b9;
}

#chat {
    width: 80%;
    margin: auto;
}

#message-form {
    text-align: center;
    position: fixed;
    left: 0;
    bottom: 0;
```

```
41        width: 100%;
42        background-color: #b7b5b9;
43    }
44
45    input {
46        width: 70%;
47        height: 30px;
48    }
49
50    button {
51        width: 25%;
52        height: 38px;
53    }
54
55    .sent {
56        text-align: right;
57        background-color: #a79cb3;
58        margin-left: 50%;
59    }
60
61    .sent span {
62        margin-left: 5px;
63        color: #6929ca;
64    }
```

Conclusion

You have now built a simple realtime chat application using the Firebase SDK and Realtime Database as a datastore.

If you're looking for a robust application that's built to scale, consider CometChat. You can create an account for free, and start building right away. Develop for as long as you need, and don't pay a thing until you're ready to scale.

Njoku Samson Ebere
CometChat

He is a software engineer who is interested in building solutions to real world problems and teaching others about the things he knows. Something he

really enjoys doing aside writing codes and technical articles is body building.

Share it with everyone!

- Facebook
- Twitter
- LinkedIn

Developers

Resources

Company