Firebase Realtime Database API Documentation

Introduction:

This application programming interface (API) documentation describes how to incorporate the Firebase Realtime Database into a JavaScript-based web app. It includes settings for data storage, database reference, and initialization.

Prerequisites:

In order to use this API, you must first have Firebase set up and the required Firebase modules installed.

```
import { initializeApp } from
"https://www.gstatic.com/firebasejs/9.19.1/firebase-app.js";
import { getDatabase, ref, set } from
"https://www.gstatic.com/firebasejs/9.19.1/firebase-database.js";
```

Initialize Firebase
initializeApp(config: object): App

The Firebase configuration object must be supplied in order to initiate Firebase. This method returns a new instance of a Firebase application.

config (object): Configuration object containing your Firebase project credentials.

```
const app = initializeApp(firebaseConfig);
```

Get a Reference to the Database:

getDatabase(app: App): Database

Get a reference to the Firebase Realtime Database service associated with the initialized Firebase app.

app (App): The Firebase app instance.

```
const db = getDatabase(app);
```

Store User Data:
Event Listener:

To save user information when a submit button is pressed in an HTML form, add a click event listener to the button.

```
document.getElementById("submit").addEventListener('click', function(e) {
   e.preventDefault();
   // Code for storing user data goes here.
});
```

set(ref: Reference, data: object): Promise<void>

Save information about users in the Firebase Realtime Database using a given identifier.

- **ref** (Reference): A database reference where the data will be stored.
- **data** (object): The data to be stored in the database.

```
set(ref(db, 'user/' + document.getElementById("username").value), {
   username: document.getElementById("username").value,
   email: document.getElementById("email").value,
```

```
  PhoneNumber: document.getElementById("phone").value
});
```

Example Usage:

```
<!DOCTYPE html>
<html>
<head>
  <!-- Include Firebase scripts and initialize Firebase -->
</head>
<body>
  <form>
    <input type="text" id="username" placeholder="Username">
    <input type="text" id="email" placeholder="Email">
    <input type="text" id="phone" placeholder="Phone Number">
    <button id="submit">Submit</button>
  </form>

  <script type="module">
    // JavaScript code as provided in your question
  </script>
</body>
</html>
```

Conclusion:

This application programming interface (API) paper describes how to use Firebase Realtime Database in a standard web project. It's flexible enough to be customized to fit your needs. Check out the Firebase docs for all the details and advanced options.

Rapid API Documentation:

Access points to which your app is linked. A log including all the request data will also be available to you. If you simply want to see metrics for a certain API in the app, you can do that too.

- **API Calls:** how many requests are being made.
- **Error rates:** how many requests are error some.
- **Latency:** how long (on average) requests take to execute

Headers sent as response:

server: The current version of the API proxy used by RapidAPI.

x-ratelimit-requests-limit: The number of requests the plan you are currently subscribed to allows you to make, before incurring overages.

x-ratelimit-requests-remaining: The number of requests remaining before you reach the limit of requests your application is allowed to make, before experiencing overage charges.

X-RapidAPI-Proxy-Response: This header is set to true when the RapidAPI proxy generates the response, (i.e. the response is not generated from the our servers).

```
import http.client

conn = http.client.HTTPSConnection("covid-193.p.rapidapi.com")

headers = {
   'x-rapidapi-host': "covid-193.p.rapidapi.com",
   'x-rapidapi-key': "XxXxXxXxXxXxXxXxXxXxXxXx"
   }
```

```
conn.request("GET", "/countries", headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Sample responses:


```
{
 "get": "countries",
 "parameters": [],
 "errors": [],
 "results": 193,
 "response": [
  "Afghanistan",
  "Albania",
  "Algeria",
  "Andorra",
  "Angola",
  "Antigua-and-Barbuda",
  "Argentina",
  "Armenia",
  "Aruba",
  "Australia",
  "Austria",
  "Azerbaijan",
  "Bahamas",
  "Bahrain",
  "Bangladesh"
 ]
}
```

Statistics:

```
import http.client

conn = http.client.HTTPSConnection("covid-193.p.rapidapi.com")

headers = {
    'x-rapidapi-host': "covid-193.p.rapidapi.com",
    'x-rapidapi-key': "XxXxXxXxXxXxXxXxXxXxXxXx"
    }

conn.request("GET", "/statistics", headers=headers)

res = conn.getresponse()
data = res.read()

print(data.decode("utf-8"))
```

Samples responses:


```
{
  "get": "statistics",
  "parameters": {
    "country": "usa"
  },
  "errors": [],
  "results": 1,
  "response": [
    {
      "continent": "North-America",
      "country": "USA",
      "population": 330848770,
      "cases": {
        "new": "+15408",
        "active": 1145446,
        "critical": 16939,
        "recovered": 621439,
        "1M_pop": "5666",
        "total": 1874731
      },
      "deaths": {
        "new": "+921",
        "1M_pop": "326",
```

```
      "total": 107846
    },
    "tests": {
     "1M_pop": "55817",
     "total": 18466841
    },
    "day": "2020-06-02",
    "time": "2020-06-02T21:00:06+00:00"
   }
 ]
}
```


CONCLUSION:

The access points to which your app is linked. A log including all the request data will also be available to you. If you simply want to see metrics for a certain API in the app, you can do that too.