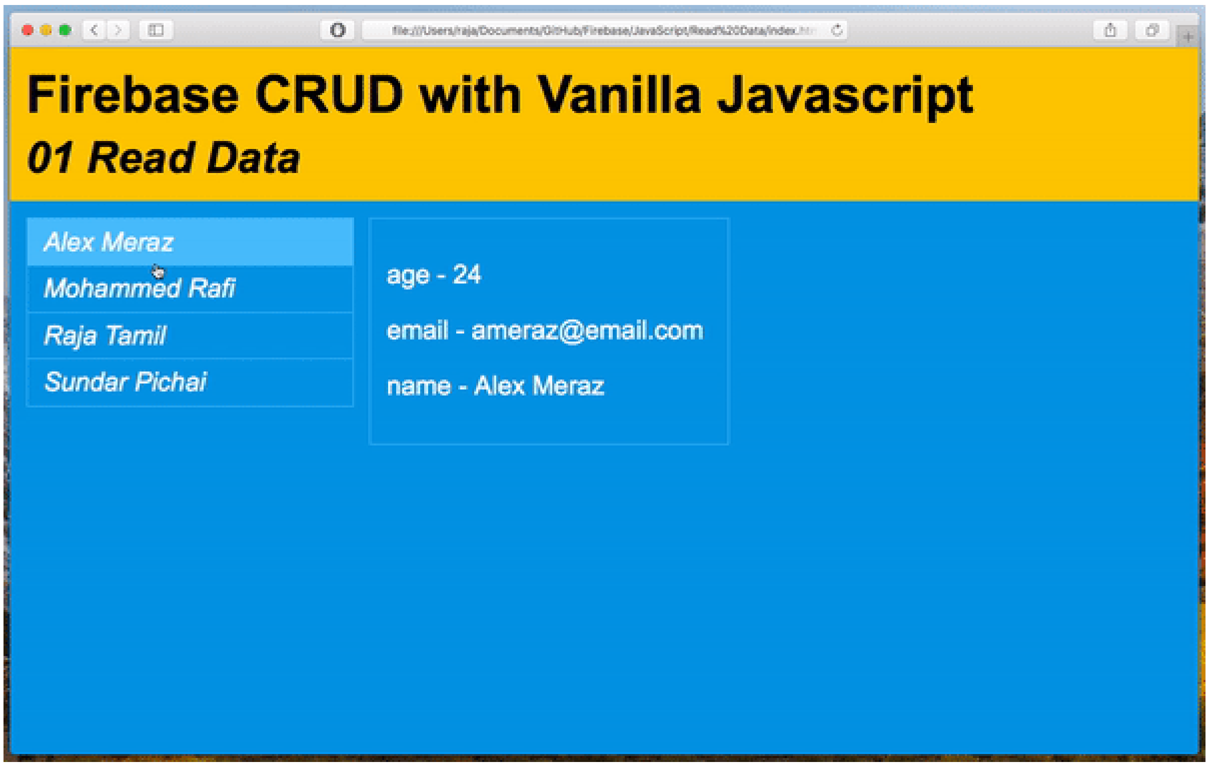
https://medium.com/@rajatamil/learn-to-build-firebase-crud-app-with-javascript-part-1-reading-data-b3f8f8e0d924 Sign up and get an extra one for free.

Build A Firebase CRUD App with Vanilla JavaScript NOW — Part 1

Raja Tamil
Feb 15, 2018 · 8 min read ★



Firebase CRUD with Vanilla Javascript
01 Read Data

Alex Meraz
Mohammed Rafi
Raja Tamil
Sundar Pichai

age - 24
email - ameraz@email.com
name - Alex Meraz

In this Firebase tutorial, I will be guiding you how to **Read/Retrieve Data** from Firebase Real-time database.

This is the first part of the Firebase tutorial series *Learn to Build a simple Firebase CRUD App with vanilla JavaScript*.

Part #1: Firebase JavaScript App → Read Data

If you're already familiar with **Firebase** and **JavaScript,** then you may want to go to next section what are we building in this tutorial with Firebase?

What you need to know before further reading:

1. **JSON**: You should be familiar with JSON because Firebase uses NoSQL database that means all the data stored in JSON tree structure as key-value pairs.
2. **JavaScript**: You should be familiar with basics of JavaScript. If not, you may want to check out my JavaScript video series.

Why Firebase?

Have you ever spent most of the time building back-end architecture such as structuring database **schema** up front, **scalability**, user **authentication** rather than spending time on building your app **Unique** and **Awesome**?

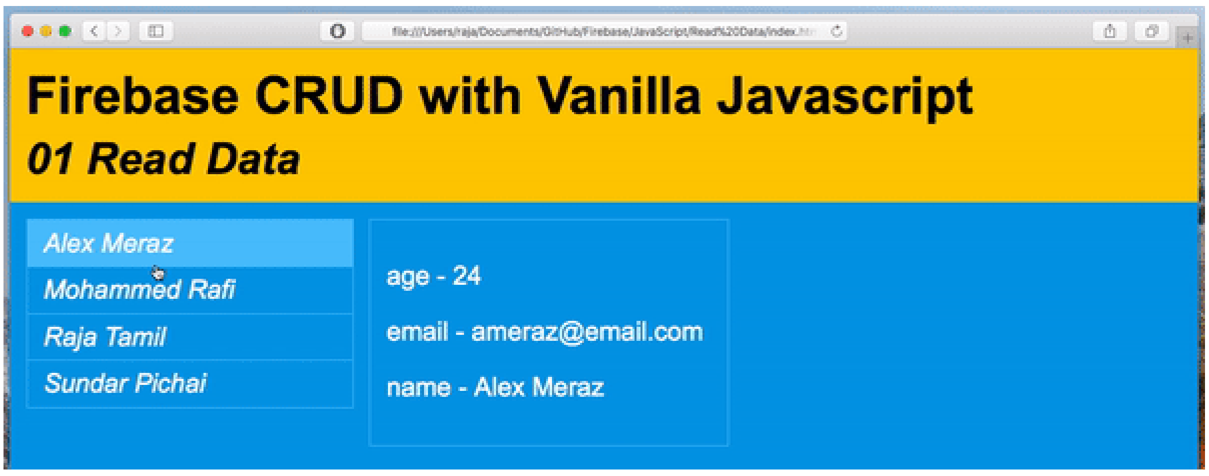
If **yes,**

Then you should try **Firebase.** ☺

- Firebase is a **back-end service** that your app can interact with. It has a lot of features such as **Real-time database, User Authentication**, **File Storage** and much more.
- With **Firebase,** we do not have to create database schema up front becauseFirebase is very flexible to make changes to the schema as we progress with our application.
- As our application evolves over the period of time, it's recommended to build an app with Firebase and change a schema **simultaneously** based on the requirements.

I find It's pretty cool! 😎

What are we building?



Firebase CRUD with Vanilla Javascript

01 Read Data

Alex Meraz
Mohammed Rafi
Raja Tamil
Sundar Pichai

age - 24

email - ameraz@email.com

name - Alex Meraz

1. **Get Users List**: This simple web app fetches Users data from Firebase Real-time Database using Firebase SDK and populates data on the browser.
2. **Get Selected User**: When selecting a user on the left, more information about the user appears on the right.

Pretty simple and straightforward.

Now, we know what we are going to accomplish the end of this tutorial.

We just need these 6 steps below in order to achieve the final outcome:

STEP #1: Setting up a Firebase App on the Firebase Console

STEP #2: Setting up our simple User List JavaScript project

STEP #3: Initialize Firebase into the app by adding the code snippet

STEP #4: Enable Read and Write Permission to the Firebase Database

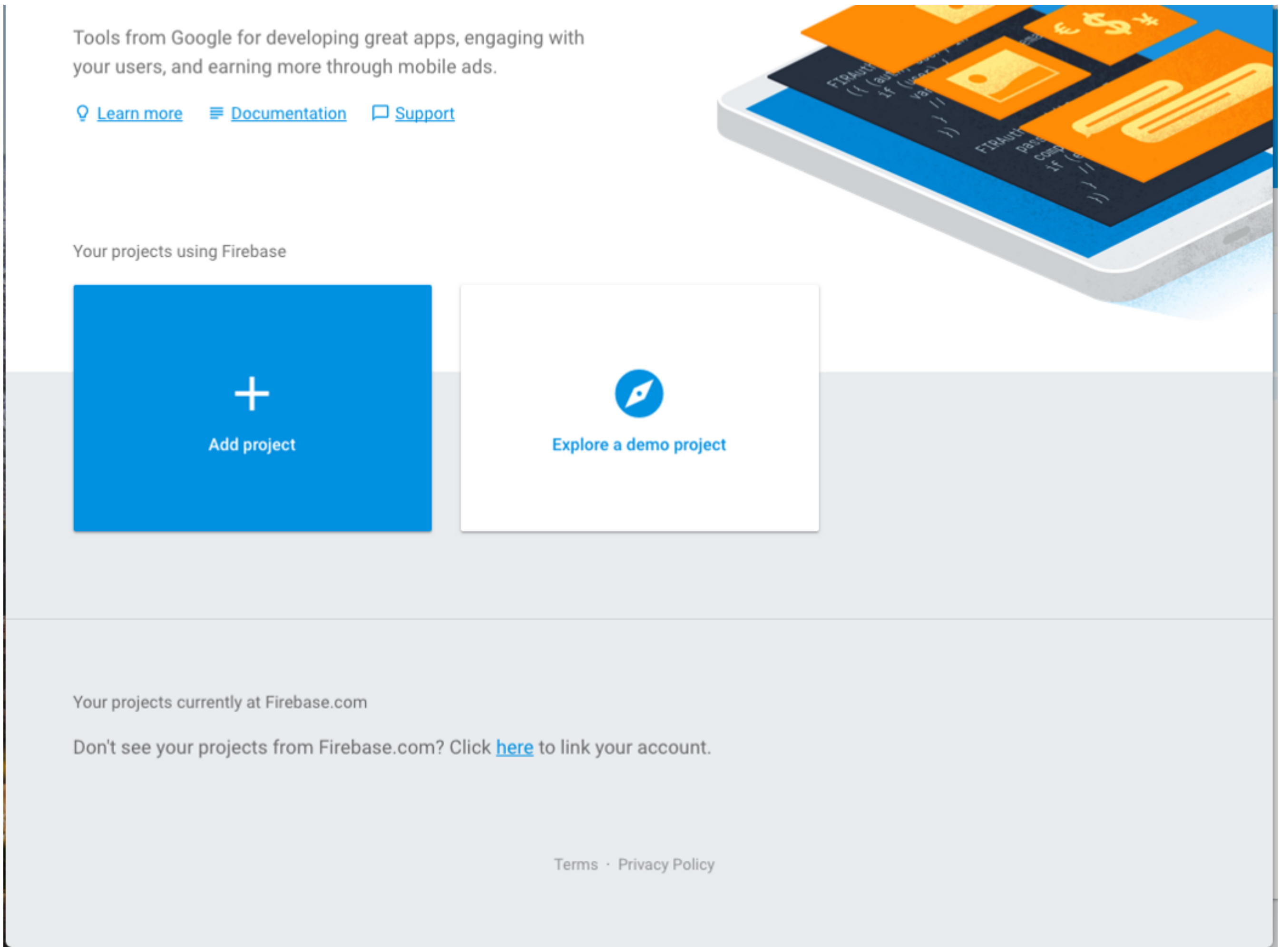
STEP #5: Import Users Schema JSON file into the Database

STEP #6: Read Users Data from the Firebase using child_added() method.

Let's get started…

1. Setting up a Firebase App on the Firebase Console:

If you already have a Gmail account, go to Firebase Console and log in. The reason you use Gmail because Firebase has been acquired by Google and became a part of it.


Tools from Google for developing great apps, engaging with your users, and earning more through mobile ads.

Learn more
Documentation
Support
Your projects using Firebase
Add project
Explore a demo project
Your projects currently at Firebase.com

Don't see your projects from Firebase.com? Click here to link your account.

Once you logged in, you will see the project explorer window like the image above and it may look different depends on when you read this article.

Go ahead click “**Add Project**” that will create a modal window in which you can put your project name and select your country then click **Create Project**.

At this point, you have successfully created a Firebase project. Go ahead click the **project** to get into the **console (Dashboard)**.

2. Setting up our simple User List JavaScript project.

File structure for our simple application is pretty straightforward.

```
users-app
| - index.html
| - app.js
| - style.css
```

HTML

```
<!DOCTYPE html>
<html>
 <head>
 <meta charset="utf-8">
 <title>Sample CRUD Firebase Javascript - 01 Read Data</title>
 <link rel="stylesheet" href="style.css">
 </head>
 <body>
 <h1>Firebase CRUD with Vanilla Javascript<br/><small><em>01 Read
Data</em></small></h1>

 <ul class="userList"></ul>

<div class="userDetail">
 <p >Name :<strong class="detailName"></strong></p>
 <p >Age: <strong class="detailAge"></strong></p>
 <p >Email: <strong class="detailEmail"></strong></p>
 </div>

 <!-- firebase sdk link goes here -->
 <script type="text/javascript" src="app.js"></script>
 </body>

</html>
```

CSS

```
body, h1, h2 {
 margin:0;
 padding:0;
}
body {
 background:#039be5;
 font-family: Arial, sans-serif;
}
h1 {
 padding:10px;
 background:#ffcc00;
}

/*---------------------------*/
/* Read Users */
/*---------------------------*/
.userList {
 margin:0;
 padding:0;
 width:200px;
 float:left;
 margin:10px;
 border:1px solid #4fc3fc;
}
```

```
.userList h2 {
 padding:10px;
 margin:0;
 color:white;
}

.userList li {
 padding:5px 10px;
 border-top:1px solid #4fc3fc;
 cursor: pointer;
 color:white;
 font-style: italic;
}

.userList li:hover {
 background: #4fc3fc;
}

.userDetail {
 float:left;
 width:200px;
 margin:10px;
 margin-left:0;
 padding:10px;
 border:1px solid #4fc3fc;
 color:white;
}
```

3. Initialize Firebase SDK into the app by adding the code snippet

1. Go to **Firebase Console.**

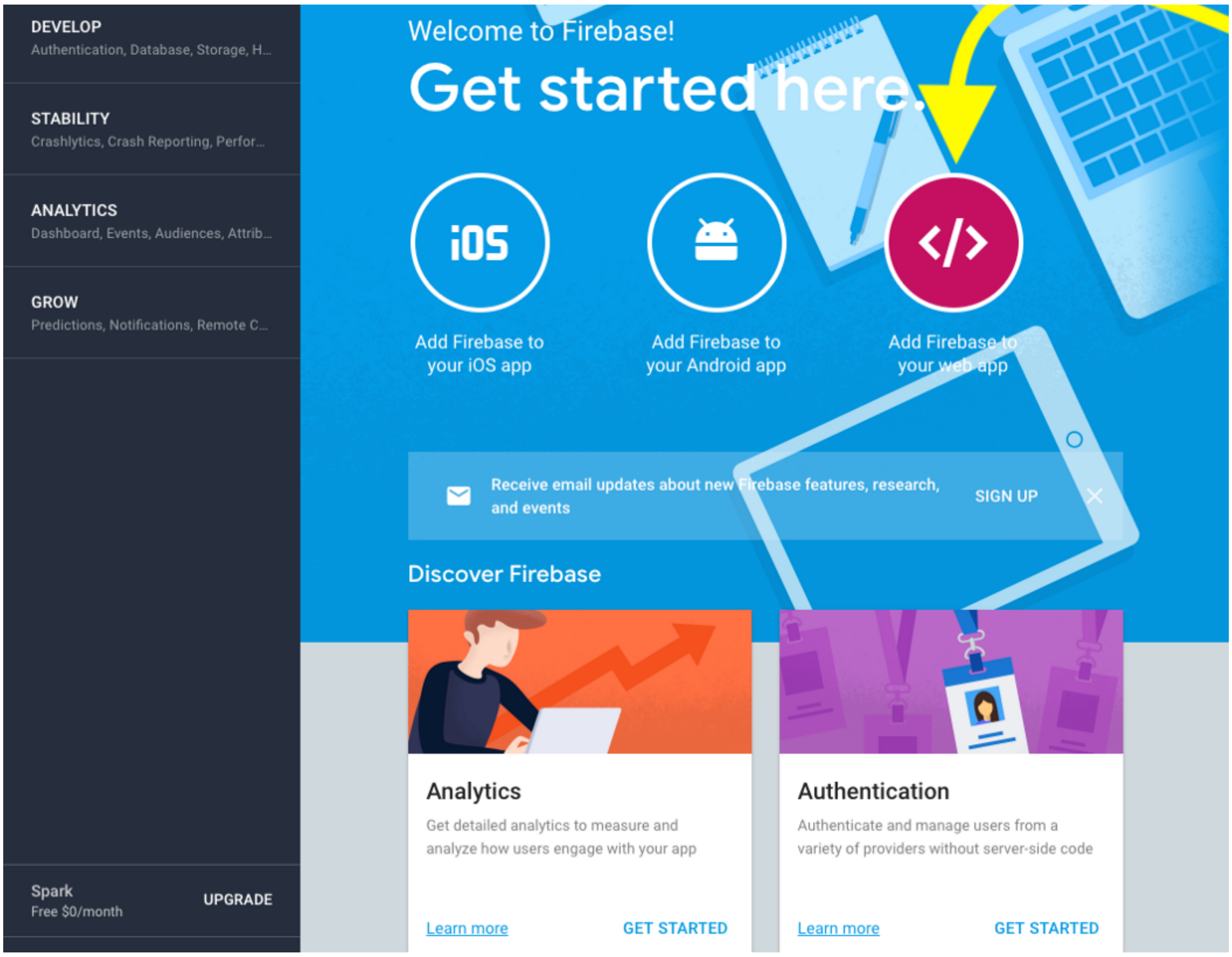
2. Click the **Project** that you have created earlier.

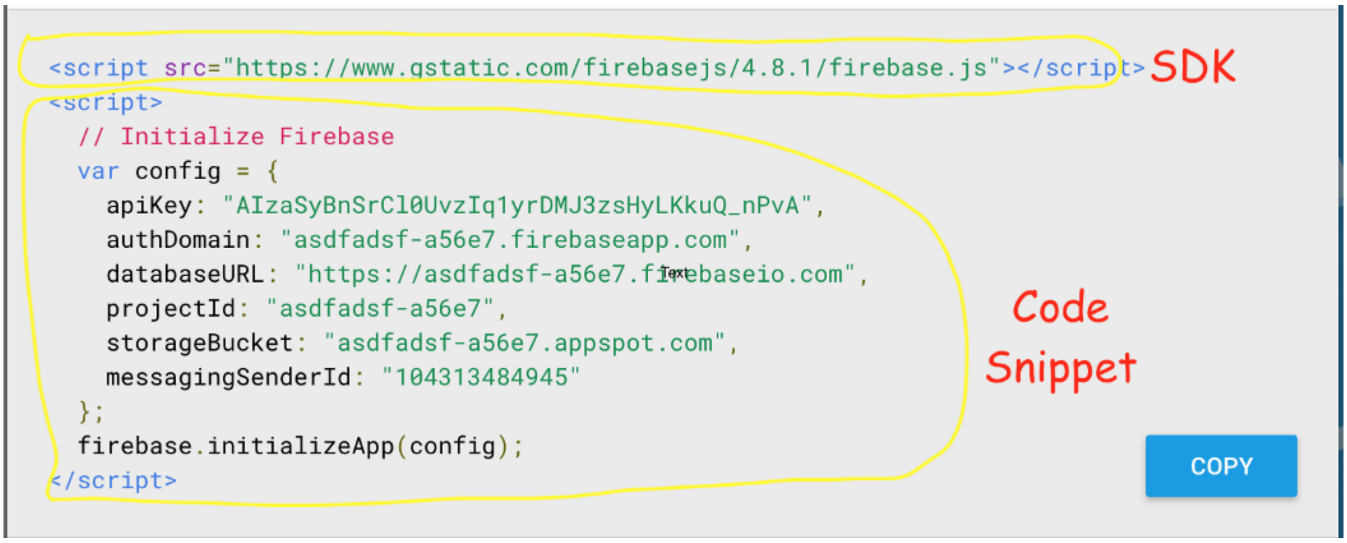
3. Then, click the **Add Firebase to your web app** icon as shown in the image below.

4. It will open the modal window with code like the image below

5. **Firebase JavaScript SDK:** Add the script inside the **index.html** file before **app.js** script link at the bottom.
   Note: if you add the SDK script link after app.js, the app WON'T work.

6. **Firebase Initialize Code Snippet:** This code will go to **app.js** at the top and starting and ending **<script>** need to be removed as we are adding the code snippet inside a javascript file.


DEVELOP
Authentication, Database, Storage, H...
STABILITY
Crashlytics, Crash Reporting, Perfor...
ANALYTICS
Dashboard, Events, Audiences, Attrib...
GROW
Predictions, Notifications, Remote C...
Spark
Free $0/month
UPGRADE
Welcome to Firebase!
Get started here.
Add Firebase to your iOS app
Add Firebase to your Android app
Add Firebase to your web app
Receive email updates about new Firebase features, research, and events
SIGN UP
Discover Firebase
Analytics
Get detailed analytics to measure and analyze how users engage with your app
Learn more
GET STARTED
Authentication
Authenticate and manage users from a variety of providers without server-side code
Learn more
GET STARTED



```
<script src="https://www.gstatic.com/firebasejs/4.8.1/firebase.js"></script>
<script>
  // Initialize Firebase
  var config = {
    apiKey: "AIzaSyBnSrCl0UvzIq1yrDMJ3zsHyLKkuQ_nPvA",
    authDomain: "asdfadsf-a56e7.firebaseapp.com",
    databaseURL: "https://asdfadsf-a56e7.firebaseio.com",
    projectId: "asdfadsf-a56e7",
    storageBucket: "asdfadsf-a56e7.appspot.com",
    messagingSenderId: "104313484945"
  };
  firebase.initializeApp(config);
</script>
```

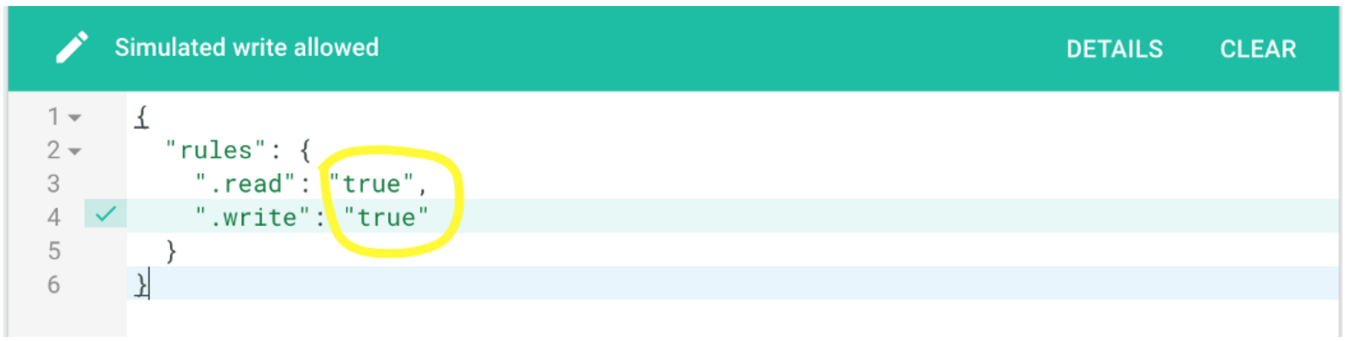
STEP #4. Enable Read and Write Permission to the Firebase Database

- Go to Firebase **Console.**
- Go to **DEVELOP** → **Database** → **Get Started** → **RULES** Tab.
- Change **read** and **write** properties to **true.**

- Hit **PUBLISH.**

That's it.

Note: As you know, we are **NOT** using **Firebase Authentication** in this article. If you want to get started, you can check it out in here



```
{
  "rules": {
    ".read": "true",
    ".write": "true"
  }
}
```

STEP #5: Create and Import JSON Schema into Firebase Database

- As Firebase uses **JSON** tree structure to store data, you can create a **JSON** file and structure your data.
- JSON structure has to be **FLAT** which means try to avoid nesting more than **two levels deep**.

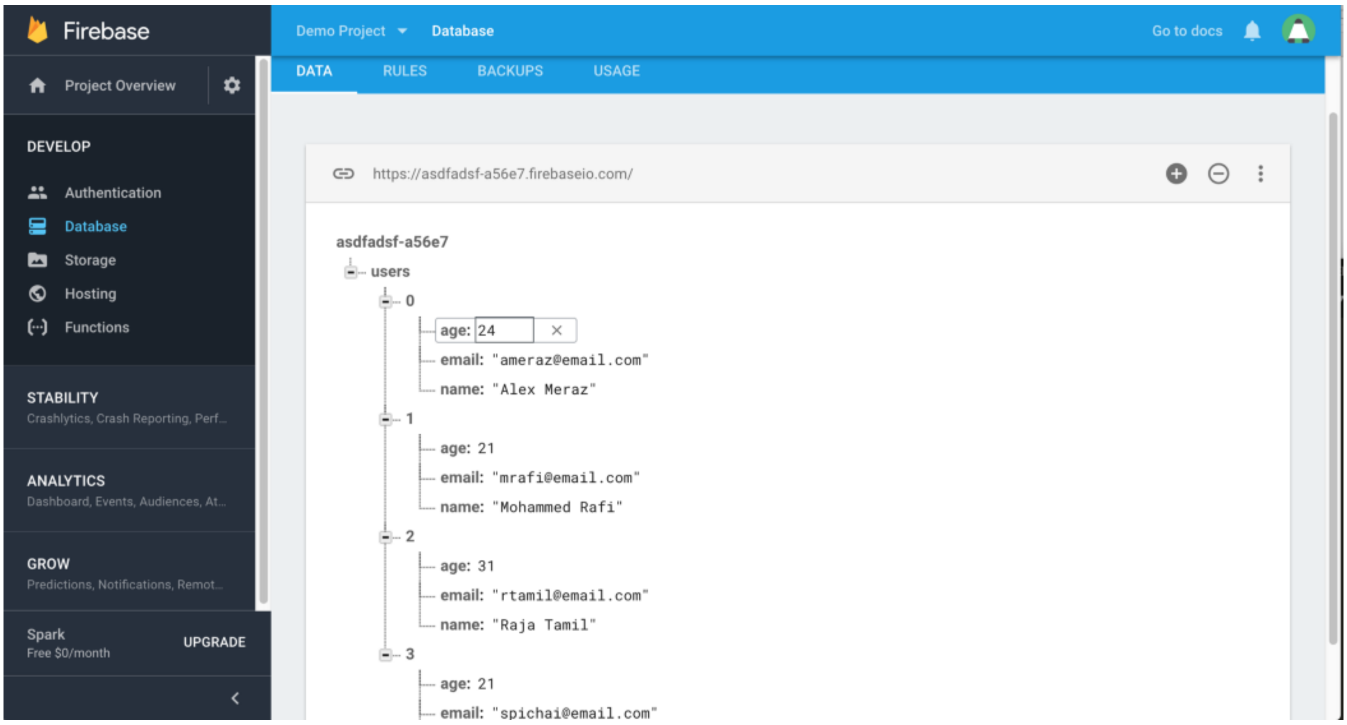
Here is the sample user JSON file below.

```
{
  "users" : [
     {"name": "Alex Meraz",
       "age" : 24,
       "email" : "ameraz@email.com"
     },
     {"name":"Mohammed Rafi",
       "age" : 21,
       "email" : "mrafi@email.com"
     },
     {"name":"Raja Tamil",
       "age" : 31,
       "email" : "rtamil@email.com"
     },
     { "name":"Sundar Pichai",
       "age" : 21,
       "email" : "spichai@email.com"
     }]
}
```

As you can see in the above JSON code, there is a **Users** key and it has a few key-value pairs and pretty straightforward.

Once you save this code as a **JSON** file, you can **import** it into Firebase Database.

- Go to Firebase **Console.**
- Go to **DEVELOP** → **Database → Get Started** → **DATA** Tab.
- Click the **⋮** vertical ellipsis icon at the right, then choose **import JSON** option from the drop-down.
- Go ahead and choose the **SAVED** user **JSON** file from your computer.
- Finally, Click **Import** from the modal dialog



```
https://asdfadsf-a56e7.firebaseio.com/

asdfadsf-a56e7
  users
    0
      age: 24
      email: "ameraz@email.com"
      name: "Alex Meraz"
    1
      age: 21
      email: "mrafi@email.com"
      name: "Mohammed Rafi"
    2
      age: 31
      email: "rtamil@email.com"
      name: "Raja Tamil"
    3
      age: 21
      email: "spichai@email.com"
```

STEP #6: Read Data from Firebase Database

The **app.js** file will look like this so far.

```
// Initialize Firebase
var config = {
 apiKey: "cAIzaSyBnSrCl0UvzIq1yrDMJ3zsdummy",
 authDomain: "asdfadsf-a56e7.firebaseapp.com",
 databaseURL: "https://asdfadsf-a56e7.firebaseio.com",
 projectId: "asdfadsf-a56e7",
 storageBucket: "asdfadsf-a56e7.appspot.com",
  messagingSenderId: "104313484945"
};

firebase.initializeApp(config);
```

Note: You would need to use your version of the code snippet. DO NOT use the code above because it's dummy one and won't work

- **Database Reference:** The First line of the code below has a reference to the main root of the Firebase database. The second line has a reference to the **users** key root of the database. If we want to get all the values of the **users**, we simply use **users** root.

```
const dbRef = firebase.database().ref();
const usersRef = dbRef.child('users');
```

- **Get User List using Child_Added()** method:

```
const userListUI = document.getElementById("userList");

usersRef.on("child_added", snap => {
   let user = snap.val();
   let $li = document.createElement("li");
   $li.innerHTML = user.name;
   $li.setAttribute("child-key", snap.key);
   $li.addEventListener("click", userClicked)
   userListUI.append($li);
});
```

- **userListUI:** Get a DOM element reference for userList
- **child_added:** Attach a child_added event to the userRef database reference object. It is a Firebase event similar to click event in JavaScript and it typically retrieves a list of items from the Firebase database.
- **callback**: This event takes TWO arguments, a string "child_added" and the callback which will run on each interaction.
- **snap**: In each interaction **snap** object, which is a parameter of the callback, will hold information about a single user item that we can have access to.

- **snap.key**: This will hold an index number of each user item as we store them in an array in our Firebase Database JSON tree structure.
- **snap.val():** val() function will return user object so that we can access any item in that object using . notation.
- **snap**: Assign each user object to a variable **user,** at this point I would just need only one value out of the user object which is **.name.**
- **li.innerHTML:** Create **li** element and set the name of the user using **user.name** value.
- **child-key**: Set an attribute called **child-key** to li which will have the key of each li.
- **child-key**: Set an attribute called **child-key** to li which will have the key of each li.
- **userClicked**: Attach click event to **li** so that if any user clicks on the left we can show more information on the right.
- **append:** This will add li to ul on every iteration.
- **Show User Detail on li click**:

```
function userClicked(e) {

  var userID = e.target.getAttribute("child-key");

  const userRef = dbRef.child('users/' + userID);

  const userDetailUI = document.getElementById("userDetail");
  userDetailUI.innerHTML = ""

  userRef.on("child_added", snap => {
    var $p = document.createElement("p");
    $p.innerHTML = snap.key + " - " + snap.val()
    userDetailUI.append($p);
  });

}
```

- **userID:** get the **child-key** attribute on clicking the username (li)
- **userRef:** This time the root is “users/” + userID. which will give us a specific user object when we use child_added event.

- **child_added**: get the snap object on each iteration which will have all the key-value pairs of a user object.
- **Finally,** add each key and value into p element then append them into userDetailUI DOM element.

At this state, you should have an application that can talk to Firebase database and retrieve data to the browser using vanilla JavaScript.

If you have any question, feel free to comment below. I will be able to answer as soon as I can.

You can find full source code on **GitHub**

Firebase JavaScript Web Web Development Development