

Creating a simple search app in 30 minutes using Algolia and Firebase

Learn how to search in minutes!

Search is a critical part of the modern web. There are countless ways to use it, but there's no need to reinvent the wheel when it comes to building this functionality into your website or web app. There are some awesome APIs (in other words, services) out there that will help you focus on writing your code instead of worrying about search! Today I'm going to show you how to use [Algolia](#) and [Firebase](#) to create a simple search application.



Here's the Algolia home page! As we can see, it's used by some pretty big companies. For the sake of this tutorial, we're going to be focusing on creating a simple website that will use the Algolia API for search.

Setup

0. Sign up for an Algolia account

- Follow [this link](#) to create your Algolia account!

Algolia offers a completely free account in their Community tier. This is great because you're able to get full API functionality for development purposes without incurring any cost! There is no credit card required to get started.

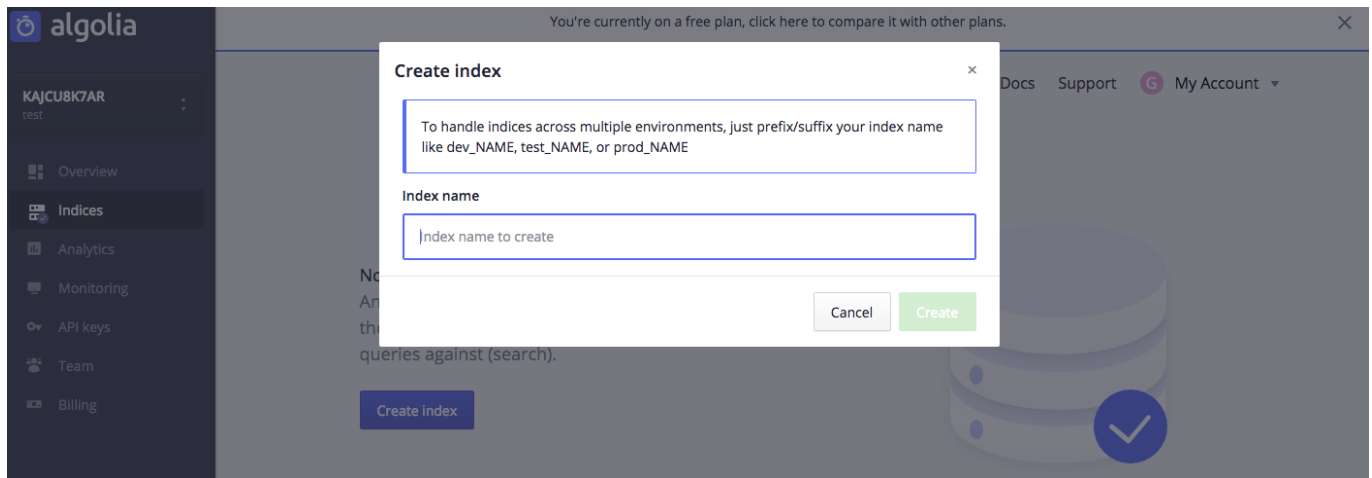
1. Find your API credentials

After successfully creating your account, click on **API Keys** in the left-hand column of your dashboard to find your API credentials. You'll need them in a few minutes! These credentials are sent along with your search requests in order to tell the Algolia servers which developer account the activity is associated with.

Note: Your Admin API Key is secret! Don't share it publicly or post it online.

2. Create an index

Click on **Indices** in the left-hand column of your dashboard and then click the **Create index** button. Enter a name for your index and then click the **Create** button.



Getting Started

3. Create an index.html file

- Go to [this link](#) to download the example **index.html** file needed to create our search page or copy and paste it from here:

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/bootstrap.min.css"
">
    <style>
      .algolia-autocomplete {
        width: 100%;
      }
      .algolia-autocomplete .aa-input, .algolia-autocomplete .aa-hint {
        width: 100%;
      }
      .algolia-autocomplete .aa-hint {
        color: #999;
      }
      .algolia-autocomplete .aa-dropdown-menu {
        width: 100%;
        background-color: #fff;
        border: 1px solid #999;
        border-top: none;
      }
      .algolia-autocomplete .aa-dropdown-menu .aa-suggestion {
        cursor: pointer;
        padding: 5px 4px;
      }
      .algolia-autocomplete .aa-dropdown-menu .aa-suggestion.aa-cursor {
```

```
        background-color: #B2D7FF;
    }
    .algolia-autocomplete .aa-dropdown-menu .aa-suggestion em {
        font-weight: bold;
        font-style: normal;
    }
</style>
</head>
<body>
    <div class="container">
        <div class="row">
            <div class="col-sm-6 col-sm-offset-3">
                <form action="#" class="form">
                    <h3>Algolia API Tutorial</h3>
                    <input class="form-control" id="search-input" name="contacts"
type="text" placeholder='Search for events...' />
                </form>
            </div>
        </div>
    </div>

    <script
src="https://cdn.jsdelivr.net/algoliasearch/3/algoliasearch.min.js">
</script>
    <script
src="https://cdn.jsdelivr.net/autocomplete.js/0/autocomplete.min.js">
</script>
    <script>
        var client = algoliasearch('YourApplicationID', 'YourAPIKey');
        var index = client.initIndex('your_index_name');
        autocomplete('#search-input', { hint: false }, [
            {
                source: autocomplete.sources.hits(index, { hitsPerPage: 5 }),
                displayKey: 'name',
                templates: {
                    suggestion: function(suggestion) {
                        return suggestion._highlightResult.name.value;
                    }
                }
            }
        ]).on('autocomplete:selected', function(event, suggestion, dataset) {
            console.log(suggestion, dataset);
            alert('dataset: ' + dataset + ', ' + suggestion.name);
        });
    </script>
</body>
</html>
```

4. Insert your API credentials and index name

- Grab the API credentials from the **API Keys** section of your Algolia dashboard and insert them into the appropriate spots in the code (**YourApplicationID** and **YourAPIKey**). Then, insert the index name

you created earlier into the appropriate spot in the code (`your_index_name`).

```
46 <script src="https://cdn.jsdelivr.net/algoliasearch/3/algoliasearch.min.js"></script>
47 <script src="https://cdn.jsdelivr.net/autocomplete.js/0/autocomplete.min.js"></script>
48 <script>
49   var client = algoliasearch('YourApplicationID', 'YourAPIKey');
50   var index = client.initIndex('your_index_name');
51   autocomplete('#search-input', { hint: false }, [
52     {
```

- Save the ***index.html*** file.

5. Testing it out

You can create a new [GitHub](#) repository, upload your ***index.html*** file to it and enable GitHub Pages to create a website from it. An even easier way to test is to use [CodePen](#) by creating a new pen and pasting in the contents of your ***index.html*** file.

If all went well, you should see a page similar to the following!

Algolia API Tutorial

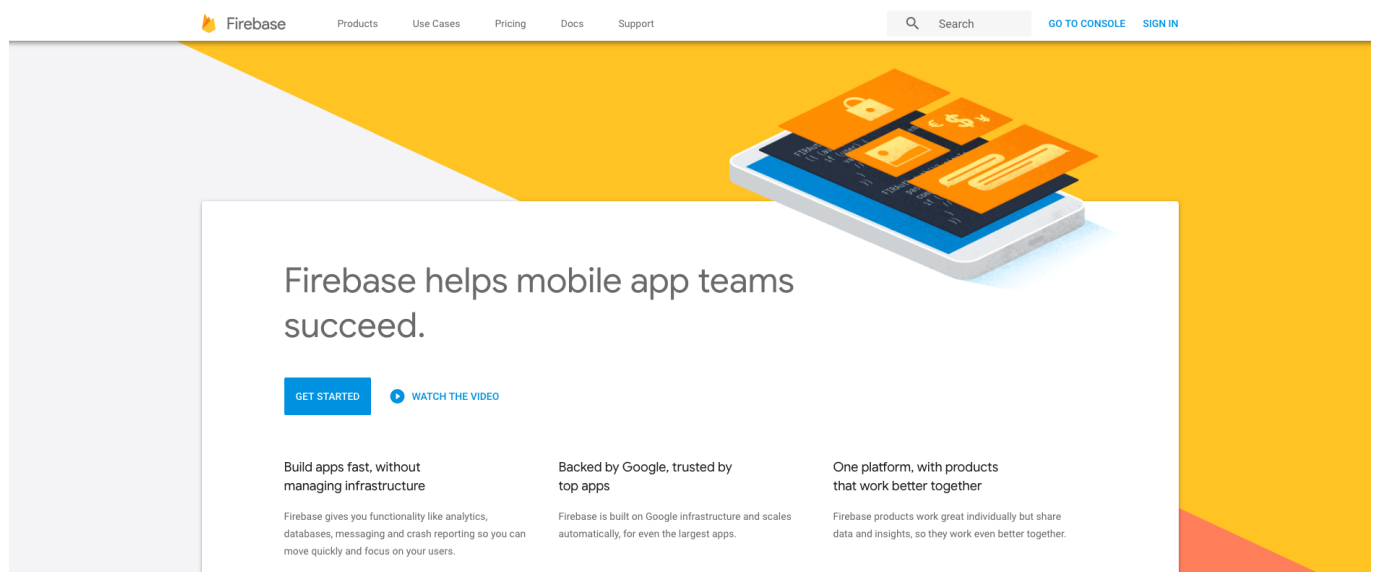
Connecting Firebase

6. Sign up for a Firebase account

- Follow [this link](#) to create your Firebase account!

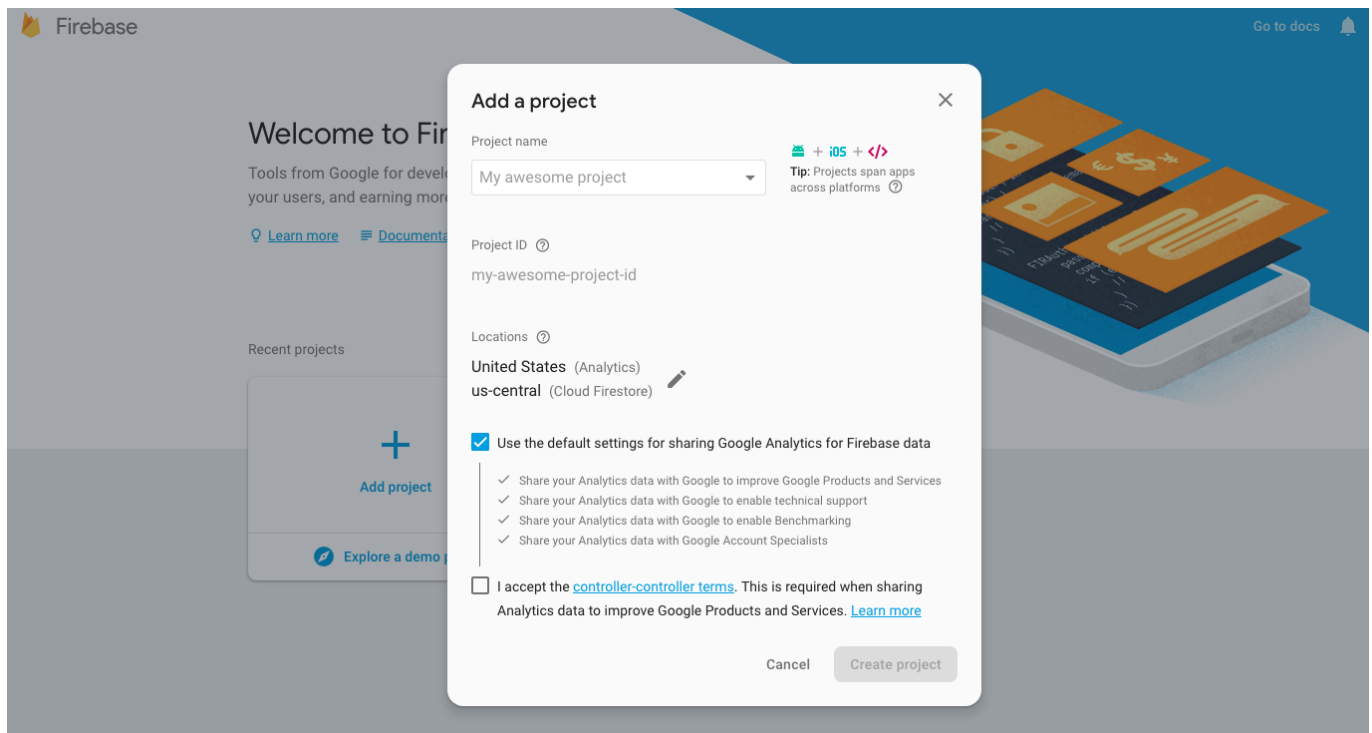
Note: Your Firebase account is tied to your Google account, so if you already have a Google account you can sign in with it.

Firebase offers a completely free account for development and testing purposes. That's what we'll use for this tutorial. Again, no credit card is required to get started.



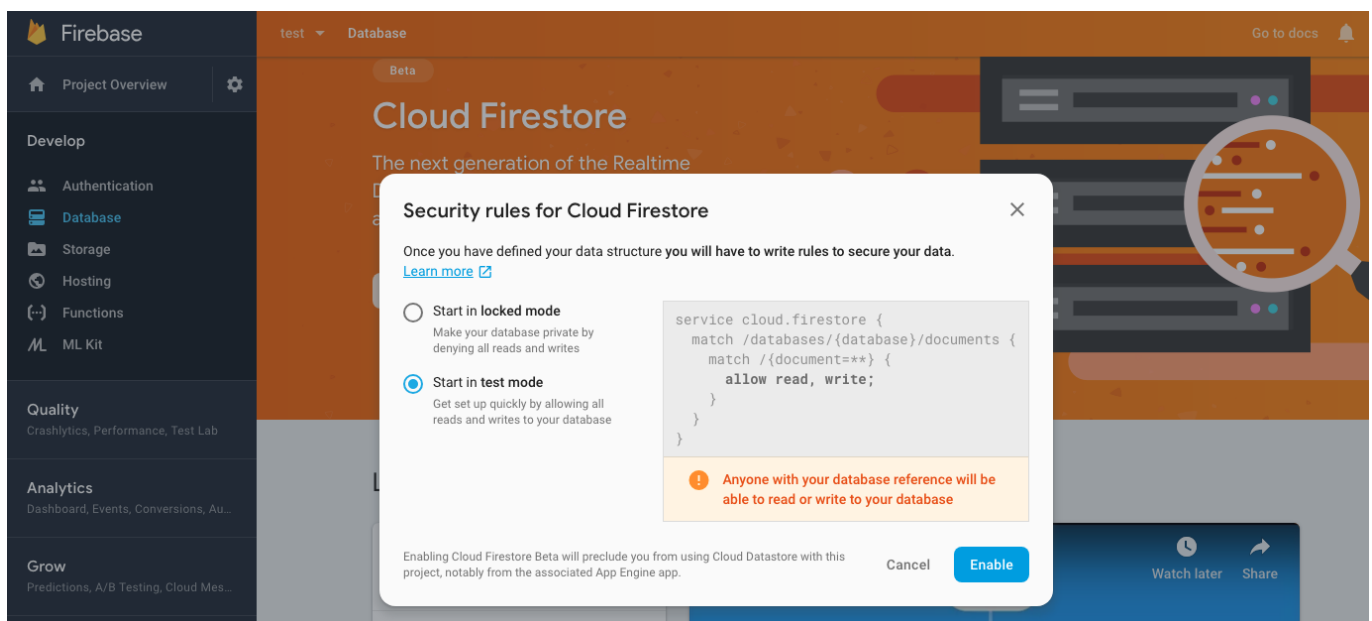
7. Create a new Firebase Application

- Click the **Add project** button in your Firebase dashboard.
- Enter a name for it.
- Click the **Create project** button.



8. Create a new database

- Click on **Database** in the left-hand column of your project dashboard.
- Click on the **Create database** button.
- Choose the radio button to **Start in test mode**.
- Click the **Enable** button.

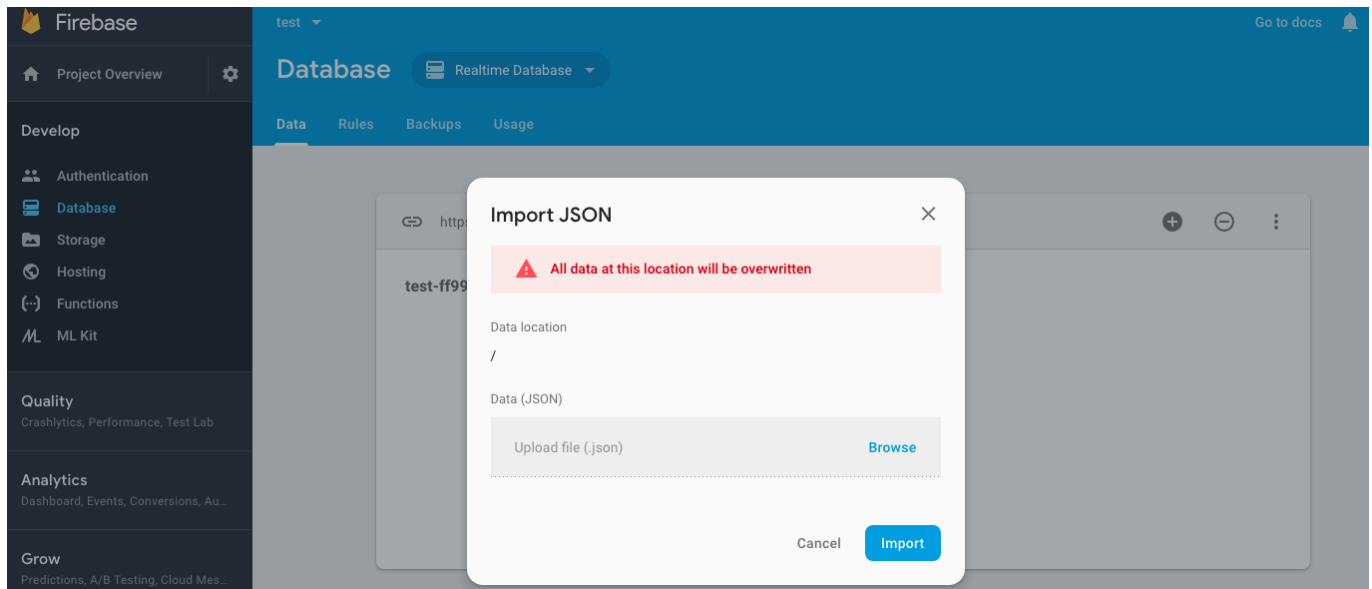


9. Add your data

- Go to [this link](#) to download the example **mlh_member_events.json** file needed to create our database or copy and paste it from here:

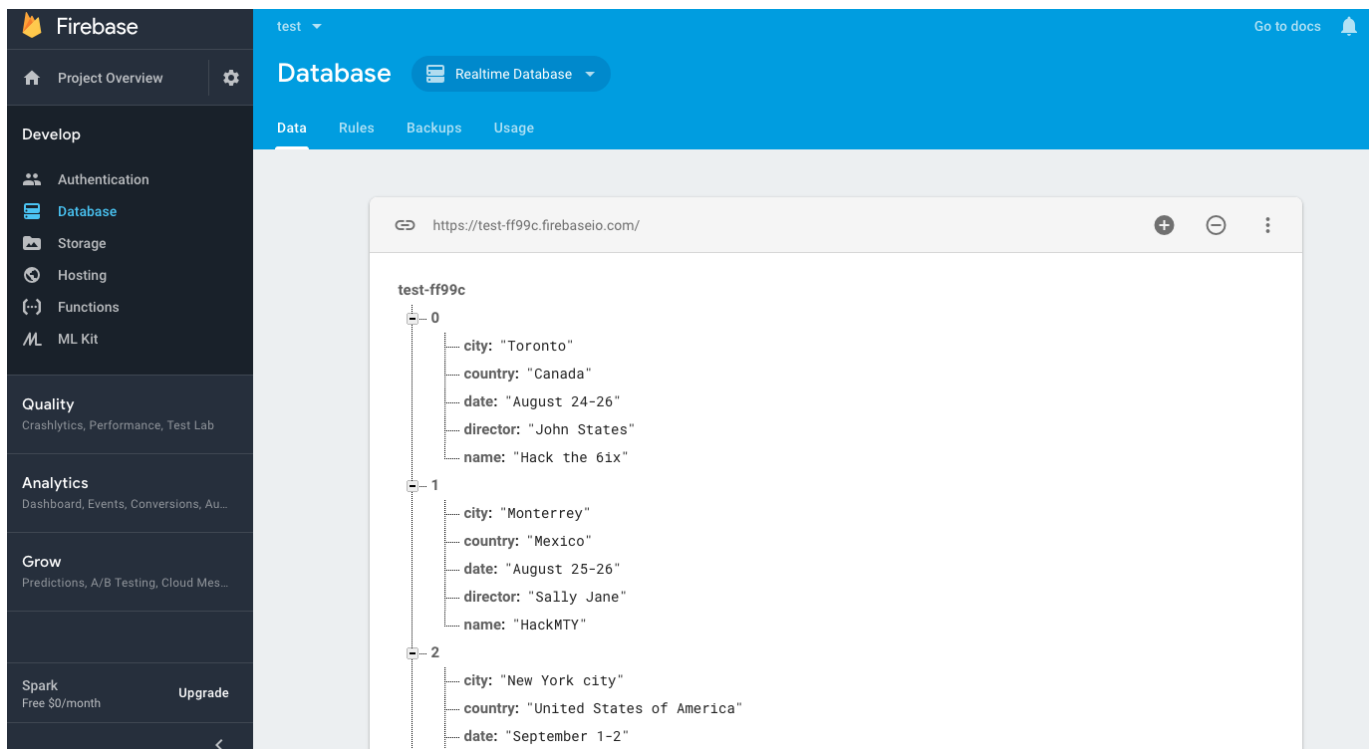
```
[
  {
    "name": "Hack the 6ix",
    "date": "August 24-26",
    "city": "Toronto",
    "country": "Canada",
    "director": "John States"
  },
  {
    "name": "HackMTY",
    "date": "August 25-26",
    "city": "Monterrey",
    "country": "Mexico",
    "director": "Sally Jane"
  },
  {
    "name": "ByteHacks",
    "date": "September 1-2",
    "city": "New York city",
    "country": "United States of America",
    "director": "Jack Doe"
  },
  {
    "name": "MedHacks",
    "date": "September 7-9",
    "city": "Baltimore",
    "country": "United States of America",
    "director": "Rachel Diaz"
  }
]
```

- Click on the dropdown next to **Database** and choose the **Realtime Database** option.
- Click on the **more** button (the vertical ellipsis).
- Click on **Import JSON**.
- Browse to the **mlh_member_events.json** file you downloaded and click the **Import** button.



- Click on the **Expand data** button (the plus symbol).

If all went well, you should see the example data imported into your database similar to the following!



10. Create a Node.js application

Note: If you haven't used Node.js before, go to [this link](#) to install it on your system.

- Create a new folder.
- Run `npm init` from the command line inside of the new folder.
- Accept the defaults in the wizard (just keep pressing **Enter**) to create a **package.json** file.
- Run `npm install dotenv algoliasearch firebase --save` from the command line inside of the new folder.

11. Configure your environment

- Go to [this link](#) to download the example `.env` file needed to set up our environment or copy and paste it from here:

```
ALGOLIA_APP_ID=<algolia-app-id>
ALGOLIA_API_KEY=<algolia-api-key>
ALGOLIA_INDEX_NAME='algolia-index-name'
FIREBASE_DATABASE_URL=https://<my-firebase-database>.firebaseio.com
```

- Insert your Algolia API credentials, Algolia index name and Firebase database URL into this file.
- Save this file into the same folder that you ran the `npm init` command in.

12. Create a main index.js file

- Go to [this link](#) to download the example `index.js` file needed for our Node.js application or copy and paste it from here:

```
const algoliasearch = require('algoliasearch');
const dotenv = require('dotenv');
const firebase = require('firebase');

// load values from the .env file in this directory into process.env
dotenv.load();

// configure firebase
firebase.initializeApp({
  databaseURL: process.env.FIREBASE_DATABASE_URL,
});
const database = firebase.database();

// configure algolia
const algolia = algoliasearch(
  process.env.ALGOLIA_APP_ID,
  process.env.ALGOLIA_API_KEY
);
const index = algolia.initIndex(process.env.ALGOLIA_INDEX_NAME);

// Get all contacts from Firebase
database.ref('/my-firebase-database').once('value', my-firebase-database => {
  // Build an array of all records to push to Algolia
  const records = [];
  contacts.forEach(contact => {
    // get the key and data from the snapshot
    const childKey = contact.key;
    const childData = contact.val();
    // We set the Algolia objectID as the Firebase .key
    childData.objectID = childKey;
    // Add object for indexing
    records.push(childData);
  });
});
```



```
// Add or update new objects
index
  .saveObjects(records)
  .then(() => {
    console.log('Contacts imported into Algolia');
  })
  .catch(error => {
    console.error('Error when importing contact into Algolia', error);
    process.exit(1);
  });
});
```

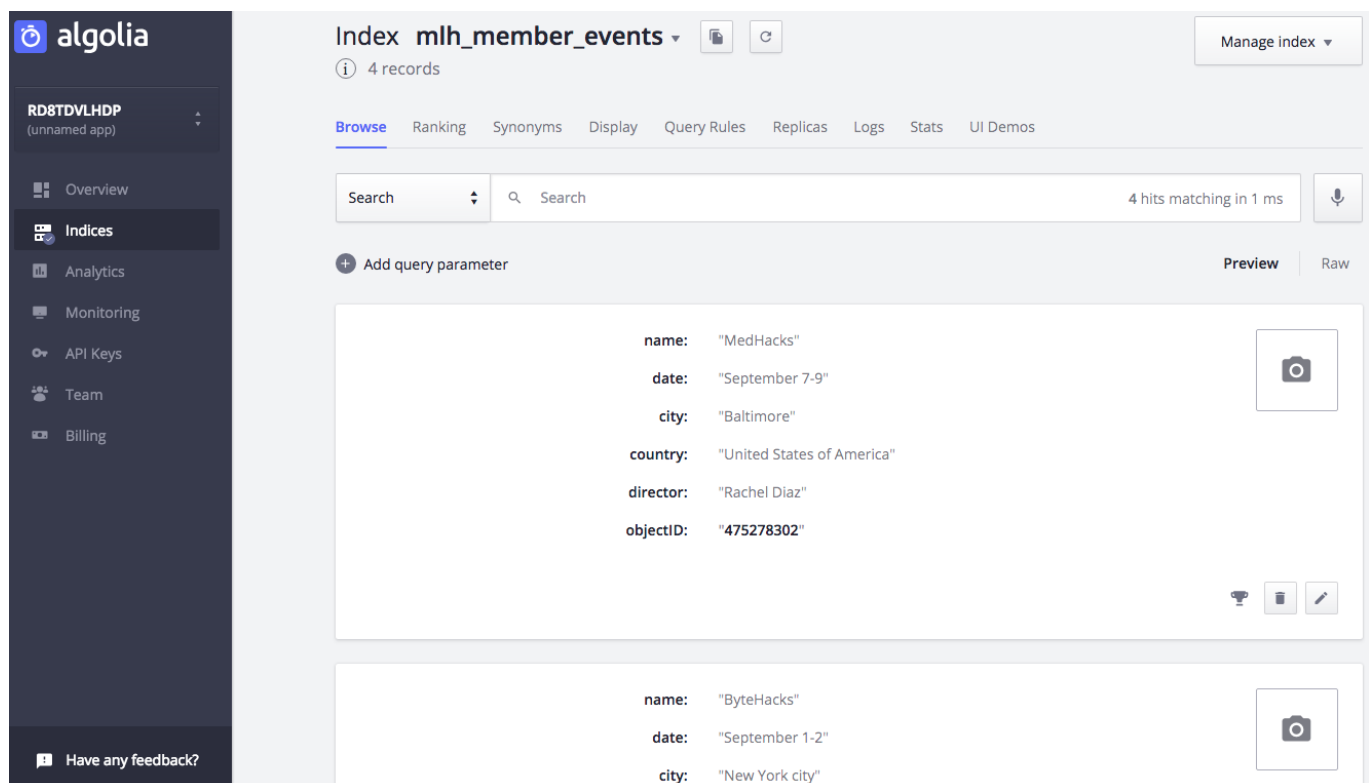
- Replace the two instances of `my-firebase-database` with the name of your Firebase database.
- Save this file into the same folder that you ran the `npm init` command in.
- Run `node index.js` from the command line inside of the same folder.

Verifying and Testing

13. Verify the synchronization

- Click on **Indices** in the left-hand column on your Algolia dashboard.

If all went well, you should see the example data from Firebase synchronized similar to the following!



The screenshot shows the Algolia dashboard interface. On the left is a dark sidebar with the Algolia logo and a navigation menu including Overview, Indices, Analytics, Monitoring, API Keys, Team, and Billing. The main content area is titled 'Index mlh_member_events' and shows '4 records'. Below this is a search bar with the text 'Search' and a magnifying glass icon. To the right of the search bar, it says '4 hits matching in 1 ms'. Below the search bar, there are tabs for 'Browse', 'Ranking', 'Synonyms', 'Display', 'Query Rules', 'Replicas', 'Logs', 'Stats', and 'UI Demos'. The 'Browse' tab is selected. Below the tabs, there is a section for 'Add query parameter' and a 'Preview' button. The 'Preview' button is active, showing a list of records. The first record has the following fields: name: "MedHacks", date: "September 7-9", city: "Baltimore", country: "United States of America", director: "Rachel Diaz", and objectID: "475278302". The second record has the following fields: name: "ByteHacks", date: "September 1-2", and city: "New York city".

14. Test the search

- Load your GitHub Page or your CodePen Pen that you created earlier.
- Try typing in the search box.

If all went well, you should see results from the example data similar to the following!

Algolia API Tutorial

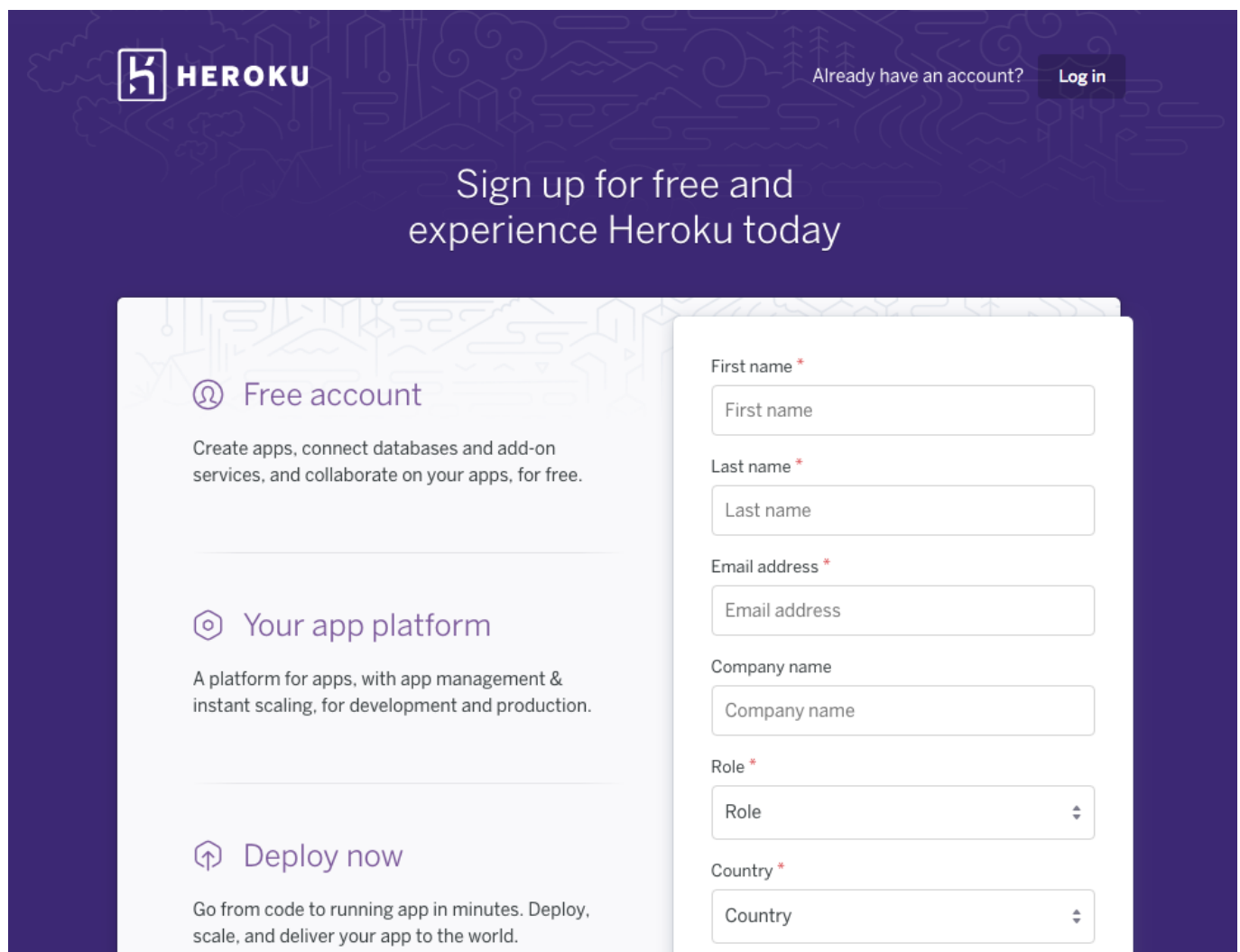
h
HackMTY
Hack the 6ix

Optional Steps

15. Sign up for a Heroku account

- Follow [this link](#) to create your Heroku account!

Heroku offers a completely free account for use with development and testing. We can use this to host our Node.js application for synchronizing our Firebase database with our Algolia index. Just like with Firebase and Algolia, no credit card is required to get started.

The image shows the Heroku sign-up page. At the top, there's a Heroku logo and a 'Log in' button for existing users. The main heading says 'Sign up for free and experience Heroku today'. Below this, there are three sections: 'Free account' (Create apps, connect databases and add-on services, and collaborate on your apps, for free.), 'Your app platform' (A platform for apps, with app management & instant scaling, for development and production.), and 'Deploy now' (Go from code to running app in minutes. Deploy, scale, and deliver your app to the world.). On the right side, there's a sign-up form with fields for First name, Last name, Email address, Company name, Role (dropdown), and Country (dropdown). All fields are marked with a red asterisk indicating they are required.

16. Modify index.js file

- Remove everything from the bottom of the **index.js** file starting from the line beginning with `// Get all contacts from Firebase` and replace it with the following code:

```
const contactsRef = database.ref('/my-firebase-database');
contactsRef.on('child_added', addOrUpdateIndexRecord);
contactsRef.on('child_changed', addOrUpdateIndexRecord);
```

```
contactsRef.on('child_removed', deleteIndexRecord);

function addOrUpdateIndexRecord(contact) {
  // Get Firebase object
  const record = contact.val();
  // Specify Algolia's objectID using the Firebase object key
  record.objectID = contact.key;
  // Add or update object
  index
    .saveObject(record)
    .then(() => {
      console.log('Firebase object indexed in Algolia', record.objectID);
    })
    .catch(error => {
      console.error('Error when indexing contact into Algolia', error);
      process.exit(1);
    });
}

function deleteIndexRecord(contact) {
  // Get Algolia's objectID from the Firebase object key
  const objectID = contact.key;
  // Remove the object from Algolia
  index
    .deleteObject(objectID)
    .then(() => {
      console.log('Firebase object deleted from Algolia', objectID);
    })
    .catch(error => {
      console.error('Error when deleting contact from Algolia', error);
      process.exit(1);
    });
}
```

- Replace the instance of `my-firebase-database` with the name of your Firebase database.
- Save the file.

17. Deploy to Heroku

- Go to [this link](#) and follow the tutorial there to deploy your modified Node.js synchronization app to Heroku.

Getting Started on Heroku with Node.js

Introduction

Set up

Prepare the app

Deploy the app

View logs

Define a Procfile

Scale the app

Declare app dependencies

Introduction

This tutorial will have you deploying a Node.js app to Heroku in minutes.

Hang on for a few more minutes to learn how to get the most out of the Heroku platform.

The tutorial assumes that you have a free [Heroku account](#), and that you have [Node.js](#) and [npm](#) installed locally.

I'm ready to start

([Log in](#) to save and track your progress)

Congratulations! You've built a simple search page using the Algolia API with Firebase as your database.

I hope you enjoyed this tutorial on using [Algolia's](#) API to power search in your website.

The full source code is [available on GitHub here!](#)