

Vue 3 Firestore example: Build a CRUD Application

📅 Last modified: September 6, 2021 (<https://www.bezkoder.com/vue-3-firestore/>) 👤 bezkoder
 (<https://www.bezkoder.com/author/bezkoder/>) 📁 [Firestore](#)
 (<https://www.bezkoder.com/category/firebase/>), [Vue.js](#) (<https://www.bezkoder.com/category/vue/>)

In this tutorial, I will show you step by step to build a Vue 3 Firestore CRUD App example with Firebase Cloud Firestore Database.

Related Posts:

- [Vue 3 Firebase example: Build a CRUD App \(Realtime Database\)](https://www.bezkoder.com/vue-3-firebase/) (<https://www.bezkoder.com/vue-3-firebase/>)
- [Vue 3 CRUD example with Axios & Vue Router](https://www.bezkoder.com/vue-3-crud/) (<https://www.bezkoder.com/vue-3-crud/>)
- [Vue 3 Authentication with JWT, Vuex, Axios and Vue Router](https://www.bezkoder.com/vue-3-authentication-jwt/) (<https://www.bezkoder.com/vue-3-authentication-jwt/>)

Contents [hide]

[Vue 3 Firestore example Overview](#)
[CRUD Operations using Firestore CollectionReference](#)
[Technology](#)
[Setup the Firebase Project](#)
[Setup Vue 3 Firebase Firestore Project](#)
[Add Bootstrap to Vue 3 Firestore example](#)
[Add Vue Router to Vue 3 Firestore example](#)
[Add Navbar and Router View to Vue 3 Firestore example](#)
[Integrate Firebase into Vue 3 App](#)
[Create Data Service](#)
[Component for creating Documents](#)
[Component for List of Documents](#)
[Component for Document details](#)
[Run & Check](#)
[Conclusion](#)
[Further Reading](#)
[Source Code](#)

Vue 3 Firestore example Overview

We're gonna build an Vue 3 Firestore CRUD App using [firebase](https://www.npmjs.com/package/firebase) (<https://www.npmjs.com/package/firebase>)

library in which:

We use cookies to improve your experience with the site. To find out more, you can read the [full](#)

- Each Tutorial has key, title, description, published status.

Privacy & Policy (<https://bezkoder.com/privacy-policy/>) [Accept](#)

- We can create, retrieve, update, delete Tutorials (CRUD operations) from Firebase Firestore

Here are the screenshots:

– Create a new Tutorial:

bezKoder Tutorials Add

Vue 3 Firestore CRUD example

Title

bezkoder Tut#3

Description

Tut#3 Description

Submit

Cloud Firestore after the Create Operations:

bezkoder-firebase ▾

Cloud Firestore

Data Rules Indexes Usage

🏠 > tutorials > JoFn0b8iJZev3...

bezkoder-firebase	tutorials	JoFn0b8iJZev3ZtlUu2m
+ Start collection	+ Add document	+ Start collection
tutorials >	2ENi1bE9SMMCJWFQgZTi	+ Add field
	JoFn0b8iJZev3ZtlUu2m >	description: "Tut#3 Description"
	RAVrgiJA7x7EJswYtl00	published: false
	ZGqCBfmayTiv0bETnvLo	title: "bezkoder Tut#3"
	t2FIHi3vG7EQF5TJZcuC	

– Retrieve all Tutorials, the details will show when clicking on any Tutorial:



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

bezKoder Tutorials Add

Vue 3 Firestore CRUD example

Tutorials List

bezcoder Tut#1

bezcoder Tut#2

bezcoder Tut#3

bezcoder Tut#4

bezcoder Tut#5

Tutorial

Title

bezcoder Tut#3

Description

Tut#3 Description

Status:

Pending

Publish

Delete

Update

– Change status to **Published/Pending** using **Publish/UnPublish** button:

Tutorial

Title

bezcoder Tut#3

Description

Tut#3 Description

Status:

Published

UnPublish

Delete

Update

The status was updated successfully!

JoFn0b8iJZev3ZtlUu2m

+ Start collection

+ Add field

description: "Tut#3 Description"

published: true

title: "bezcoder Tut#3"

– Update the Tutorial details with **Update** button:



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezcoder.com/privacy-policy/\)](https://bezcoder.com/privacy-policy/)

Accept

Tutorial

Title

Description

Status:Published

The tutorial was updated successfully!

JoFn0b8iJZev3ZtlUu2m

+ Start collection

+ Add field

description: "Description for Tut#3"

published: true

title: "bezkode Tut#3 (updated)"

– Delete the Tutorial using **Delete** button:

Tutorials List

bezkode Tut#1

bezkode Tut#2

bezkode Tut#3 (updated)

bezkode Tut#5

Please click on a Tutorial...

tutorials

+ Add document

2ENi1bE9SMMCJWFQgZTi

JoFn0b8iJZev3ZtlUu2m

RAVrgiJA7x7EJswYtl00

ZGqCBfmayTiv0bETnvLo

t2FIHi3vG7EQF5TJZcuC >

t2FIHi3vG7EQF5TJZcuC

+ Start collection

+ Add field

description: "Tut#4 Description"

published: false

title: "bezkode Tut#4"

CRUD Operations using Firestore CollectionReference

We're gonna use instance of `firebase.firestore.CollectionReference`

(<https://firebase.google.com/docs/reference/js/firebase.firestore.CollectionReference>) to read/write data from the Firestore.

```
var tutorialsRef = firebase.firestore().collection("/tutorials");
```

– Read collection once using `get()` :



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkode.com/privacy-policy/\)](https://bezkode.com/privacy-policy/)

Accept

```
tutorialsRef.get().then(function(snapshot) {
  vat tutorials = [];

  snapshot.forEach(function(childSnapshot) {
    var id = childSnapshot.id;
    var data = childSnapshot.val();
    // ...

    tutorials.push({ id: id, title: data.title, description: data.description});
  });
});
```

– Read collection with listening to the data changes using `onSnapshot()` :

```
tutorialsRef.onSnapshot(function(snapshot) {
  snapshot.docChanges().forEach(function(change) {
    if (change.type === "added") {
      console.log("New tutorial: ", change.doc.data());
    }
    if (change.type === "modified") {
      console.log("Modified tutorial: ", change.doc.data());
    }
    if (change.type === "removed") {
      console.log("Removed tutorial: ", change.doc.data());
    }
  });
});
```

– Listening for all value change events on a collection reference

```
tutorialsRef.onSnapshot(function(snapshot) {
  snapshot.forEach(function(childSnapshot) {
    var id = childSnapshot.id;
    var childData = childSnapshot.val();
    // ...
  });
});
```

– Detach the listener to stop using bandwidth to receive updates:



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

```

var unsubscribe = tutorialsRef.onSnapshot(function(snapshot) {
  // ...
});

// Stop listening to changes
unsubscribe();

```

– Create a new document in collection using `add()` :

```

tutorialsRef.add({
  title: "bezkoder Tut#1",
  description: "Helpful tutorial"
})
.then(function(docRef) {
  console.log("Tutorial created with ID: ", docRef.id);
})
.catch(function(error) {
  console.error("Error adding Tutorial: ", error);
});

```

– Update document by `id` in collection:

+ destructive update using `set()` : delete everything currently in place, then save the new value

```

tutorialsRef.doc(id).set({
  title: 'zkoder Tut#1',
  description: 'Tut#1 Description'
});

```

+ non-destructive update using `update()` : only updates the specified values

```

tutorialsRef.doc(id).update({
  title: 'zkoder new Tut#1'
});

```

– Delete a document by `id` in collection:

```

tutorialsRef.doc(id).delete();

```

– Delete entire collection: Deleting Firestore collections from a Web client is not recommended.

You can find the solution here (<https://firebase.google.com/docs/firestore/solutions/delete-collections>).



We use cookies to improve your experience with the site. To find out more, you can read the full [Privacy & Policy](#)

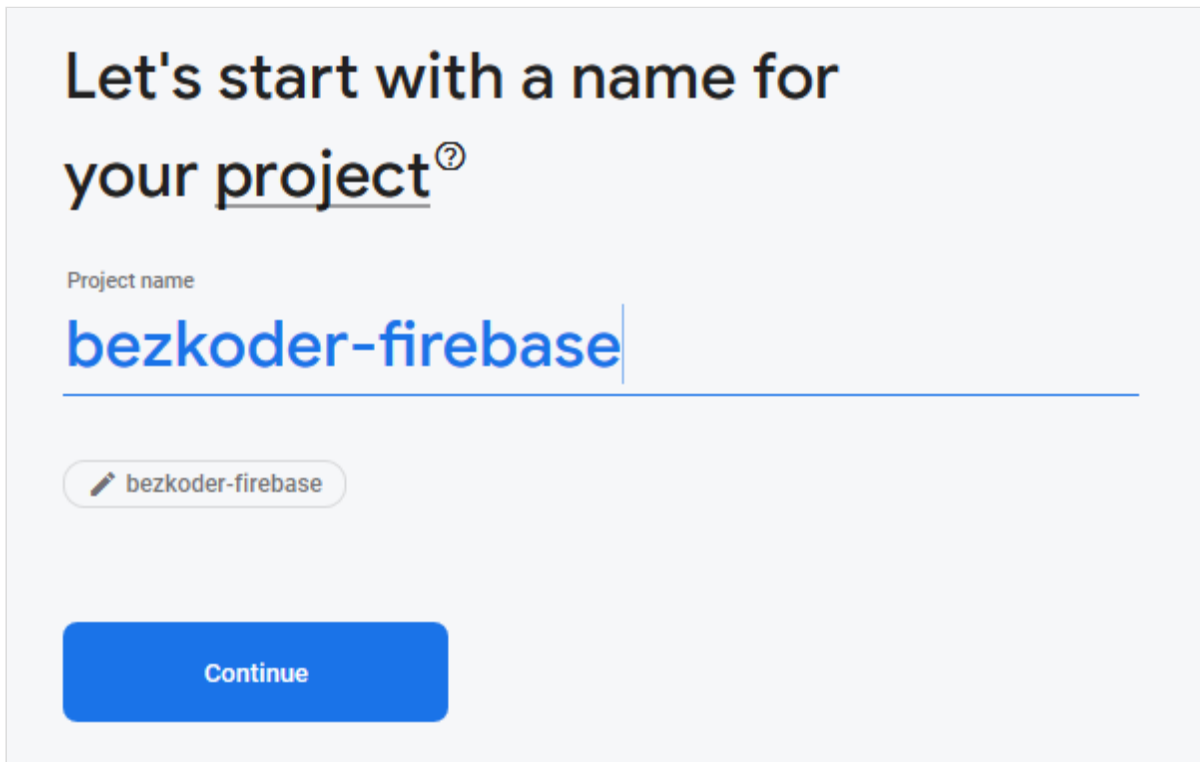
• Vue 3 [Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

- Vue Router 4
- Firebase 8
- Bootstrap 4

Setup the Firebase Project

Go to Firebase Console (<https://console.firebase.google.com/>), login with your Google Account, then click on Add Project.

You will see the window like this:



Enter Project name, set Project Id and click on **Continue**.

Turn off *Enable Google Analytics for this project*, then click **Create Project**.

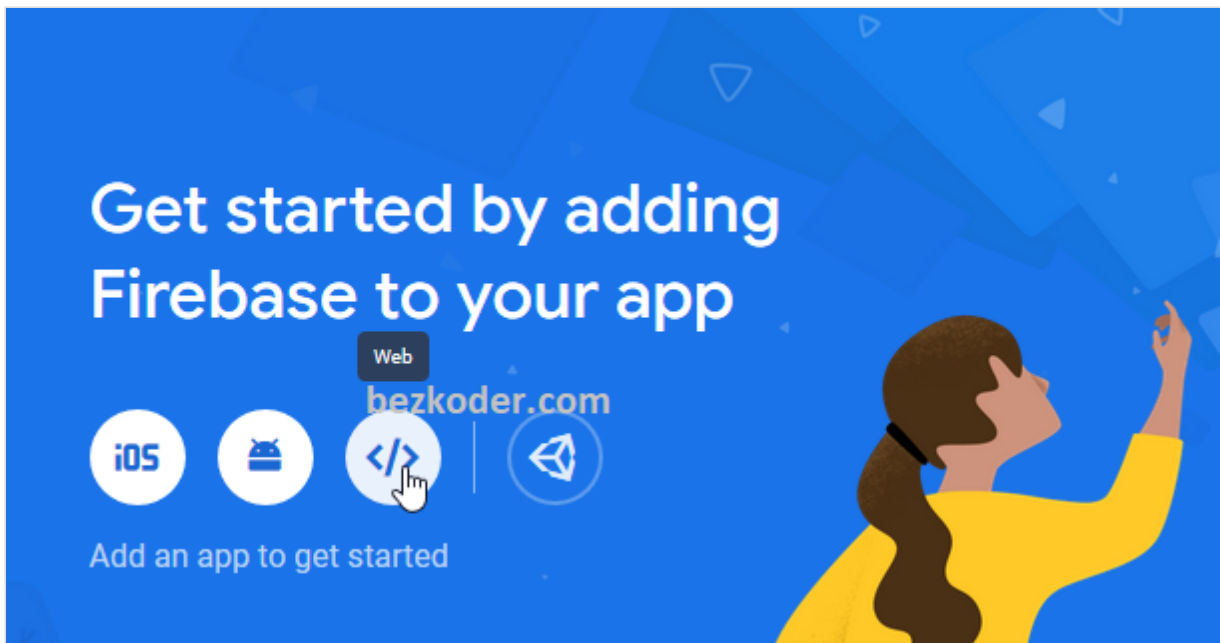
Now, browser turns into following view:



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept



If you don't see it, just choose Project **Overview**.

Click on **Web App**, a window will be shown:

× **Add Firebase to your web app**

1

Register app

App nickname ?

bezcoder-firebase

☐ Also set up **Firebase Hosting** for this app. [Learn more](#) ↗

Hosting can also be set up later. It's free to get started anytime.

Register app

2

Add Firebase SDK

Set the nickname and choose **Register App** for next step.



We use cookies to improve your experience with the site. To find out more, you can read the full

Privacy & Policy (<https://bezcoder.com/privacy-policy/>)

Accept



Register app

2

Add Firebase SDK

Copy and paste these scripts into the bottom of your <body> tag, but before you use any Firebase services:

```
<!-- The core Firebase JS SDK is always required and must be listed first -->
<script src="https://www.gstatic.com/firebasejs/7.17.1/firebase-app.js"></script>

<!-- TODO: Add SDKs for Firebase products that you want to use
      https://firebase.google.com/docs/web/setup#available-libraries -->

<script>
  // Your web app's Firebase configuration
  var firebaseConfig = {
    apiKey: "AIza[REDACTED]Hw4",
    authDomain: "[REDACTED].firebaseapp.com",
    databaseURL: "https://[REDACTED].firebaseio.com",
    projectId: "[REDACTED]db",
    storageBucket: "[REDACTED].appspot.com",
    messagingSenderId: "312[REDACTED]1",
    appId: "1:[REDACTED]:web:52[REDACTED]fd0"
  };
  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
</script>
```

Learn more about Firebase for web: [Get Started](#), [Web SDK API Reference](#), [Samples](#)

[Continue to console](#)

Copy the script for later use.

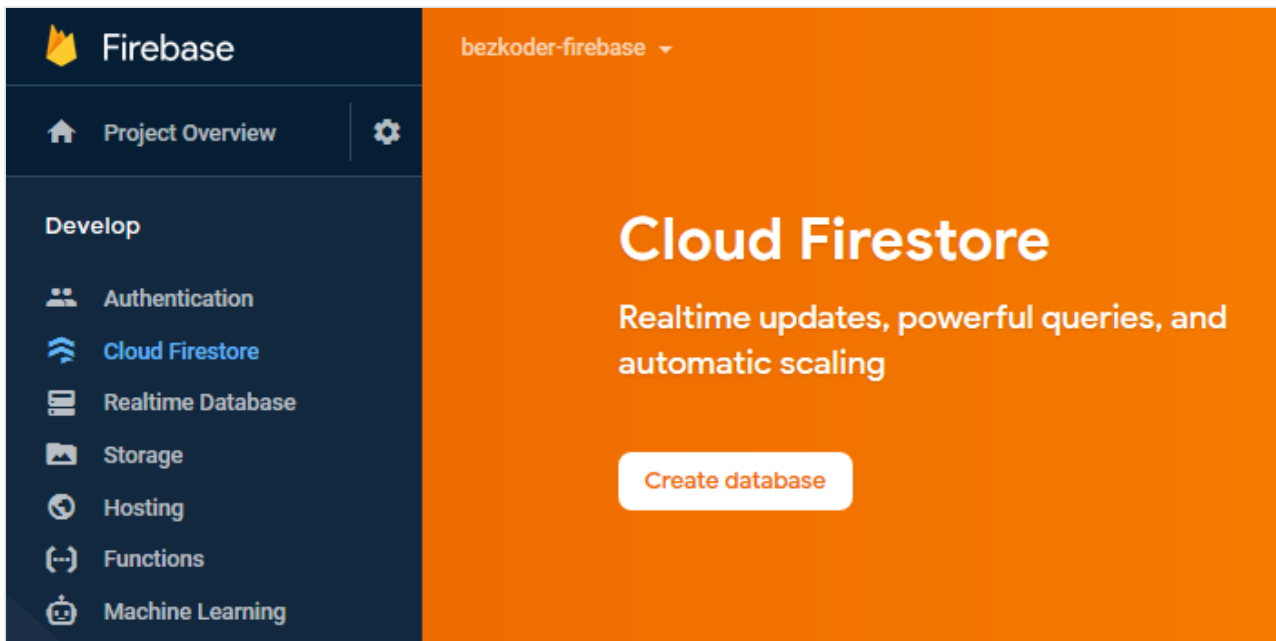
Choose **Cloud Firestore** on the left (list of Firebase features) -> **Create Database**.



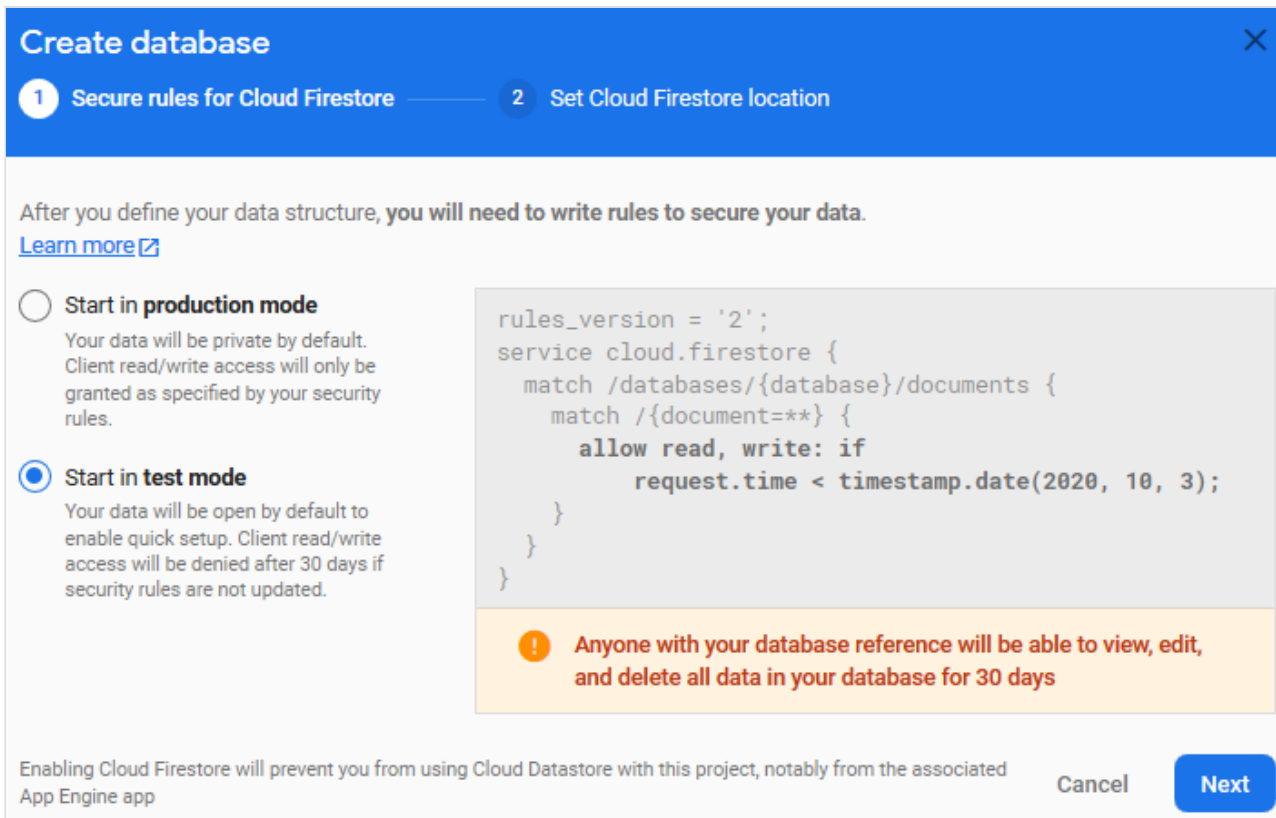
We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept



In this tutorial, we don't implement Authentication, so let's choose **test mode**:



Or if you come from another situation, just open Tab **Rules**, then change `allow read, write` value to `if true`.

Finally, we need to set Cloud Firestore Location:



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

Create database

Secure rules for Cloud Firestore

2 Set Cloud Firestore location

Your location setting is where your Cloud Firestore data will be stored.

After you set this location, you cannot change it later. Also, this location setting will be the location for your default Cloud Storage bucket.
 [Learn more](#)

Cloud Firestore location

nam5 (us-central)

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

Cancel
Enable

Setup Vue 3 Firebase Firestore Project

Open cmd at the folder you want to save Project folder, run command:

```
vue create vue-3-firestore
```

You will see some options, choose **Default ([Vue 3] babel, eslint)**.

After the process is done. We create new folders and files like the following tree:

```

├── public
│   └── index.html
├── src
│   ├── components
│   │   ├── AddTutorial.vue
│   │   ├── Tutorial.vue
│   │   └── TutorialsList.vue
│   ├── services
│   │   └── TutorialDataService.js
│   ├── App.vue
│   ├── firebase.js
│   └── main.js

```

We use cookies to improve your experience with the site. To find out more, you can read the full

package. [Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

Let me explain it briefly.

- **package.json** contains 3 main modules: `vue` , `vue-router` , `firebase` .
- `firebase.js` configures information to connect with Firebase Project and export Firebase Firestore (<https://firebase.google.com/docs/reference/js/firebase.firestore.Firestore>) service.
- `TutorialDataService` exports `TutorialDataService` that uses `firebase` 's Firestore `CollectionReference` to interact with Firestore collection.
- There are 3 components that uses `TutorialDataService` :
 - `AddTutorial` for creating new item
 - `TutorialsList` contains list of items, parent of `Tutorial`
 - `Tutorial` shows item details
- **router.js** defines routes for components.
- `App.vue` contains Router View and navigation bar.

Add Bootstrap to Vue 3 Firestore example

Run command: `npm install bootstrap@4.6.0 jquery popper.js` .

Open `src/main.js` and import Bootstrap as following-

```
import { createApp } from 'vue'
import App from './App.vue'
import 'bootstrap'
import 'bootstrap/dist/css/bootstrap.min.css'
...
```

Add Vue Router to Vue 3 Firestore example

- Run the command: `npm install vue-router@4` .
- In `src` folder, open `router.js` and define `Router` as following code:



We use cookies to improve your experience with the site. To find out more, you can read the [full](#)

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

```
import { createWebHistory, createRouter } from "vue-router";

const routes = [
  {
    path: "/",
    alias: "/tutorials",
    name: "tutorials",
    component: () => import("./components/TutorialsList")
  },
  {
    path: "/add",
    name: "add",
    component: () => import("./components/AddTutorial")
  }
];

const router = createRouter({
  history: createWebHistory(),
  routes,
});

export default router;
```

We create the `routes` as an array, each route has:

- `path` : the URL path where this route can be found.
- `name` : optional name to use when we link to this route.
- `component` : component to load when this route is called.

We also use `createWebHistory` to switch from using hash to `history` mode inside the browser, using the HTML5 history API.

– Open `src/main.js` and import the router in our application:

```
...
import router from './router'

createApp(App).use(router).mount('#app')
```

Add Navbar and Router View to Vue 3 Firestore example

Let's open `src/App.vue`, this `App` component is the root container for our application, it will contain a navbar .



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

```

<template>
  <div id="app">
    <nav class="navbar navbar-expand navbar-dark bg-dark">
      <router-link to="/" class="navbar-brand">bezKoder</router-link>
      <div class="navbar-nav mr-auto">
        <li class="nav-item">
          <router-link to="/tutorials" class="nav-link">Tutorials</router-link>
        </li>
        <li class="nav-item">
          <router-link to="/add" class="nav-link">Add</router-link>
        </li>
      </div>
    </nav>

    <div class="container mt-3">
      <h2>Vue 3 Firebase CRUD example</h2>
      <router-view />
    </div>
  </div>
</template>

<script>
export default {
  name: "app"
};
</script>

<style scoped>
.container h2 {
  text-align: center;
  margin: 25px auto;
}
</style>

```

Integrate Firebase into Vue 3 App

First run the command: `npm install firebase@8.10.0`.

Open `src/firebase.js`, import `firebase` library and add configuration that we have saved when Popup window was shown:



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

```
import firebase from "firebase/app";
import "firebase/firestore";

let config = {
  apiKey: "xxx",
  authDomain: "bezkode-firebase.firebaseio.com",
  databaseURL: "https://bezkode-firebase.firebaseio.com",
  projectId: "bezkode-firebase",
  storageBucket: "bezkode-firebase.appspot.com",
  messagingSenderId: "xxx",
  appId: "xxx",
};

firebase.initializeApp(config);

export default firebase.firestore();
```

Don't forget to export `firebase.firestore.Firestore` (<https://firebase.google.com/docs/reference/js/firebase.firestore.Firestore>) service with `firebase.firestore()`.

Create Data Service

This service will use `Firestore` service exported above to interact with Firebase Cloud Firestore. It contains necessary functions for CRUD operations.

services/*TutorialDataService.js*



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkode.com/privacy-policy/\)](https://bezkode.com/privacy-policy/)

Accept

```
import firebase from "../firebase";

const db = firebase.collection("/tutorials");

class TutorialDataService {
  getAll() {
    return db;
  }

  create(tutorial) {
    return db.add(tutorial);
  }

  update(id, value) {
    return db.doc(id).update(value);
  }

  delete(id) {
    return db.doc(id).delete();
  }
}

export default new TutorialDataService();
```

Component for creating Documents

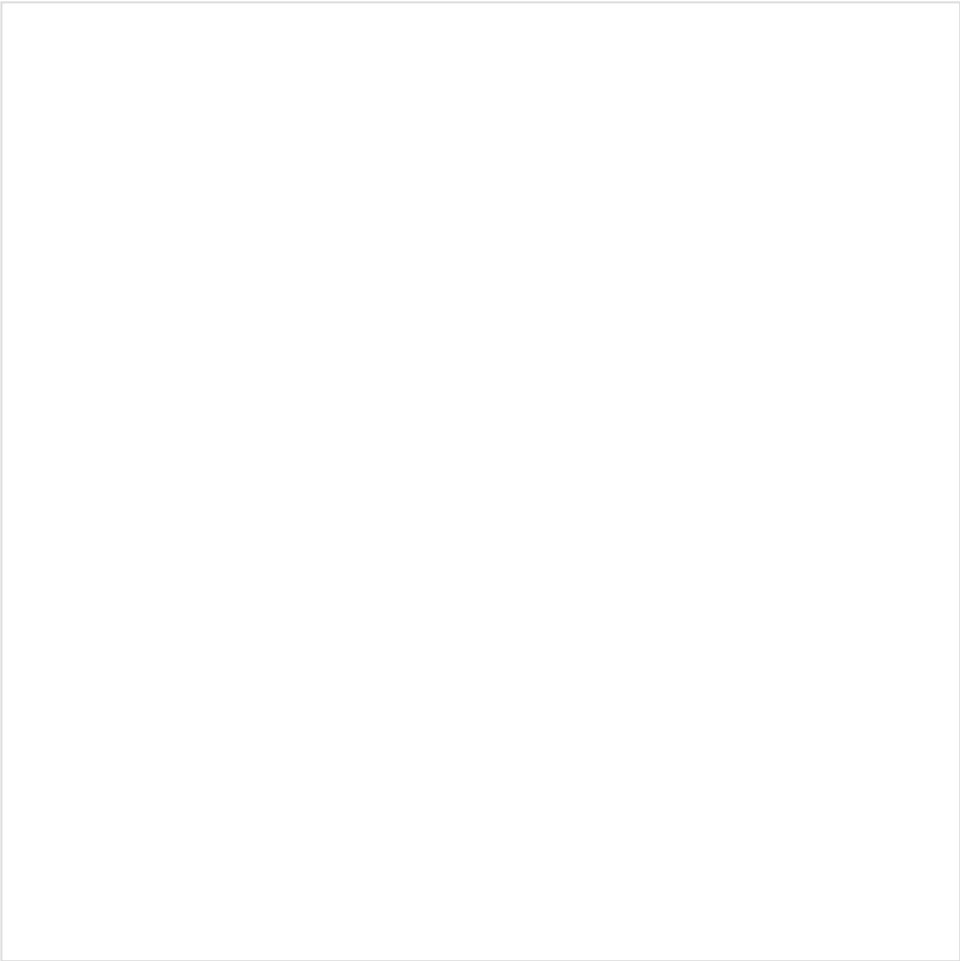
This component has a Form to submit new Tutorial with 3 fields: `title`, `description` & `published` (*false* by default). It calls `TutorialDataService.create()` method.



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept



components/*AddTutorial.vue*



We use cookies to improve your experience with the site. To find out more, you can read the full

Privacy & Policy (<https://bezkoder.com/privacy-policy/>) Accept

```

<template>
  ...
</template>

<script>
import TutorialDataService from "../services/TutorialDataService";

export default {
  name: "add-tutorial",
  data() {
    return {
      tutorial: {
        title: "",
        description: "",
        published: false
      },
      submitted: false
    };
  },
  methods: {
    saveTutorial() {
      var data = {
        title: this.tutorial.title,
        description: this.tutorial.description,
        published: false
      };

      TutorialDataService.create(data)
        .then(() => {
          console.log("Created new item successfully!");
          this.submitted = true;
        })
        .catch(e => {
          console.log(e);
        });
    },

    newTutorial() {
      this.submitted = false;
      this.tutorial = {
        title: "",
        description: "",
        published: false
      };
    }
  }
};

```



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

There are 2 main variables return from `data()` :

- `tutorial` for tutorial data
- `submitted` status

We also have a function to get value of the form (state) and call `TutorialDataService.create()` method.

For the template, we check the `submitted` state, if it is true, we show **Add** button for creating new Tutorial again. Otherwise, a Form will display.



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

```

<template>
  <div class="submit-form">
    <div v-if="!submitted">
      <div class="form-group">
        <label for="title">Title</label>
        <input
          type="text"
          class="form-control"
          id="title"
          required
          v-model="tutorial.title"
          name="title"
        />
      </div>

      <div class="form-group">
        <label for="description">Description</label>
        <input
          class="form-control"
          id="description"
          required
          v-model="tutorial.description"
          name="description"
        />
      </div>

      <button @click="saveTutorial" class="btn btn-success">Submit</button>
    </div>

    <div v-else>
      <h4>You submitted successfully!</h4>
      <button class="btn btn-success" @click="newTutorial">Add</button>
    </div>
  </div>
</template>

<script>
import TutorialDataService from "../services/TutorialDataService";

export default {
  name: "add-tutorial",
  ...
};
</script>

```

```

<style>
  .submit-form {

```

```

    max-width: 300px;

```

```

    margin: auto;

```



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy Policy \(https://bezcoder.com/privacy-policy/\)](https://bezcoder.com/privacy-policy/)

Accept

```
}  
</style>
```

Component for List of Documents

This component has:

- a *tutorials* array displayed as a list on the left.
- a selected Tutorial which is shown on the right.



So we will have following state:

- `tutorials`
- `currentTutorial` and `currentIndex`

We also need to use `TutorialDataService` 's `getAll()` method with `.orderBy("title", "asc")` for sort the list by `title` field in ascending order. X

Privacy & Policy (<https://bezkoder.com/privacy-policy/>)

Accept

And add `Tutorial` into this component as child component.

components/TutorialsList.vue



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezcoder.com/privacy-policy/\)](https://bezcoder.com/privacy-policy/)

Accept

```

<template>
  ...
</template>

<script>
import TutorialDataService from "../services/TutorialDataService";
import TutorialDetails from "./Tutorial";

export default {
  name: "tutorials-list",
  components: { TutorialDetails },
  data() {
    return {
      tutorials: [],
      currentTutorial: null,
      currentIndex: -1,
      unsubscribe: null
    };
  },
  methods: {
    onDataChange(items) {
      let _tutorials = [];

      items.forEach((item) => {
        let id = item.id;
        let data = item.data();
        _tutorials.push({
          id: id,
          title: data.title,
          description: data.description,
          published: data.published,
        });
      });

      this.tutorials = _tutorials;
    },

    refreshList() {
      this.currentTutorial = null;
      this.currentIndex = -1;
    },

    setActiveTutorial(tutorial, index) {
      this.currentTutorial = tutorial;
      this.currentIndex = index;
    },
  },
  mounted() {
    this.unsubscribe = TutorialDataService.getAll().orderBy("title", "asc").onSnapshot(th

```



We use cookies to improve your experience with the site. To find out more, you can read the full

Privacy & Policy (<https://bezcoder.com/privacy-policy/>)

this.unsubscribe = TutorialDataService.getAll().orderBy("title", "asc").onSnapshot(th

```
    },  
    beforeDestroy() {  
      this.unsubscribe();  
    }  
  };  
</script>
```

In the code above, we add a listener for data value changes in `mounted()` and detach the listener in `beforeDestroy()`.

Inside listener function, we get the `id` and other fields of each item. This `id` is unique and important for update operation.

We also have `refreshList()` function for every time delete operation is done.

Let's continue to implement the template:



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept


```

<template>
  <div class="list row">
    <div class="col-md-6">
      <h4>Tutorials List</h4>
      <ul class="list-group">
        <li
          class="list-group-item"
          :class="{ active: index == currentIndex }"
          v-for="(tutorial, index) in tutorials"
          :key="index"
          @click="setActiveTutorial(tutorial, index)"
        >
          {{ tutorial.title }}
        </li>
      </ul>
    </div>
    <div class="col-md-6">
      <div v-if="currentTutorial">
        <tutorial-details
          :tutorial="currentTutorial"
          @refreshList="refreshList"
        />
      </div>
      <div v-else>
        <br />
        <p>Please click on a Tutorial...</p>
      </div>
    </div>
  </div>
</template>

<script>export default {
  name: "tutorials-list",
  components: { TutorialDetails },
  ...
};
</script>

<style>
.list {
  text-align: left;
  max-width: 750px;
  margin: auto;
}
</style>

```



We use cookies to improve your experience with the site. To find out more, you can read the full

You can see that when we click on any item, `setActiveTutorial()` function will be invoked to change current active Tutorial, which data is passed to `Tutorial` component.

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

Component for Document details

This component is the child of `TutorialsList` component. It bind `tutorial` data and invoke `refreshList` of the parent.

For getting update, delete the Tutorial, we're gonna use two `TutorialDataService` methods:

- `update()`
- `delete()`

components/*Tutorial.vue*



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

```

<template>
  ...
</template>

<script>
import TutorialDataService from "../services/TutorialDataService";

export default {
  name: "tutorial",
  props: ["tutorial"],
  data() {
    return {
      currentTutorial: null,
      message: "",
    };
  },
  watch: {
    tutorial: function(tutorial) {
      this.currentTutorial = { ...tutorial };
      this.message = "";
    },
  },
  methods: {
    updatePublished(status) {
      TutorialDataService.update(this.currentTutorial.id, {
        published: status,
      })
        .then(() => {
          this.currentTutorial.published = status;
          this.message = "The status was updated successfully!";
        })
        .catch((e) => {
          console.log(e);
        });
    },

    updateTutorial() {
      const data = {
        title: this.currentTutorial.title,
        description: this.currentTutorial.description,
      };

      TutorialDataService.update(this.currentTutorial.id, data)
        .then(() => {
          this.message = "The tutorial was updated successfully!";
        })
        .catch((e) => {
          console.log(e);
        });
    },
  },
};

```

We use cookies to improve your experience with the site. To find out more, you can read the full [Privacy & Policy](#)

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

```
    },

    deleteTutorial() {
      TutorialDataService.delete(this.currentTutorial.id)
        .then(() => {
          this.$emit("refreshList");
        })
        .catch((e) => {
          console.log(e);
        });
    },
  },
  mounted() {
    this.message = "";
    this.currentTutorial = { ...this.tutorial }
  },
};
</script>
```

And this is the code for template:



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

```

<template>
  <div v-if="currentTutorial" class="edit-form">
    <h4>Tutorial</h4>
    <form>
      <div class="form-group">
        <label for="title">Title</label>
        <input
          type="text"
          class="form-control"
          id="title"
          v-model="currentTutorial.title"
        />
      </div>

      <div class="form-group">
        <label for="description">Description</label>
        <input
          type="text"
          class="form-control"
          id="description"
          v-model="currentTutorial.description"
        />
      </div>

      <div class="form-group">
        <label><strong>Status:</strong></label>
        {{ currentTutorial.published ? "Published" : "Pending" }}
      </div>
    </form>

    <button
      class="badge badge-primary mr-2"
      v-if="currentTutorial.published"
      @click="updatePublished(false)"
    >
      UnPublish
    </button>
    <button
      v-else
      class="badge badge-primary mr-2"
      @click="updatePublished(true)"
    >
      Publish
    </button>
  </div>

```

```

<button class="badge badge-danger mr-2" @click="deleteTutorial">
  Delete
</button>

```

We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

```

    <button type="submit" class="badge badge-success" @click="updateTutorial">
      Update
    </button>
    <p>{{ message }}</p>
  </div>

  <div v-else>
    <br />
    <p>Please click on a Tutorial...</p>
  </div>
</template>

<script>export default {
  name: "tutorial",
  props: ["tutorial"],
  ...
};
</script>

<style>
.edit-form {
  max-width: 300px;
  margin: auto;
}
</style>

```

Run & Check

You can run this App with command: `npm run serve`.

DONE Compiled successfully!

App running at:

- Local: `http://localhost:8080/`
- Network: `http://192.168.1.7:8080/`

Note that the development build is not optimized.
To create a production build, run `npm run build`.

Open browser with url: `http://localhost:8080/` and check the result.

Conclusion

Today we've built Vue Firestore example with a CRUD Application successfully using `firebase` library.



Now we can display, modify, delete documents and collection at ease.
We use cookies to improve your experience with the site. To find out more, you can read the full

If you want to use Realtime Database instead:

Privacy & Policy (<https://www.bezkoder.com/privacy-policy/>)

Accept

Vue 3 Firebase example: Build a CRUD App (Realtime Database) (<https://www.bezkoder.com/vue-3-firebase/>)

Vue 3 CRUD example with Axios & Vue Router (<https://www.bezkoder.com/vue-3-crud/>)

Further Reading

- ## Fullstack CRUD App:

- ## Source Code

vue router (<https://www.bezkoder.com/tag/vue-router/>)

Your email address will not be published. Required fields are marked *

We use cookies to improve your experience with the site. To find out more, you can read the full

Privacy & Policy (<https://bezkoder.com/privacy-policy/>).

Accept

Name *

Email *

Website

☐

Save my name, email, and website in this browser for the next time I comment.

POST COMMENT

◀ Spring Boot + SQL Server: CRUD Operations example (<https://www.bezkoder.com/spring-boot-sql-server/>)

Search...

Q

FOLLOW US



(htt

ps://

ww

w.yo

utub

e.co

m/c

han



nel/



(htt UCp (htt

ps:// 0mx ps://

face 9RH gith

boo 0Jxa ub.c

k.co Fsm om/

m/b MvK bezk

ezko XA8 oder




we use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept

TOOLS

Json Formatter (<https://www.bezkoder.com/json-formatter/>)

 (<https://www.dmca.com/Protection/Status.aspx?ID=3f543dd5-c6d8-4208-9a6b-0e92057fd597&refurl=https://www.bezkoder.com/vue-3-firestore/>)

[Home \(https://bezkoder.com/\)](https://bezkoder.com/)

[Privacy Policy \(https://www.bezkoder.com/privacy-policy/\)](https://www.bezkoder.com/privacy-policy/)

[Contact Us \(https://www.bezkoder.com/contact-us/\)](https://www.bezkoder.com/contact-us/)

[About Us \(https://www.bezkoder.com/about/\)](https://www.bezkoder.com/about/)

BezKoder 2019



We use cookies to improve your experience with the site. To find out more, you can read the full

[Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/)

Accept