

# Angular 10 Firestore CRUD

## AngularFireStore

## Angular 10 Firestore CRUD: add/get/update/delete documents with AngularFireStore

Last modified: December 10, 2020 (<https://bezkoder.com/angular-10-firebase-crud-angularfire/>)   
bezkoder (<https://bezkoder.com/author/bezkoder/>)  Angular (<https://bezkoder.com/category/angular/>),  
Firebase (<https://bezkoder.com/category/firebase/>)

In this tutorial, I will show you how to build Angular 10 CRUD App with Firebase Cloud Firestore that uses `AngularFireStore` service to get/add/update/delete documents in a collection.

### Related Posts:

- Angular 10 Firebase CRUD Realtime DB | `AngularFireDatabase` (<https://bezkoder.com/angular-10-firebase-crud/>)
- Angular 10 Firebase Storage: File Upload/Display/Delete example (<https://bezkoder.com/firebase-storage-angular-10-file-upload/>)
- Angular 10 CRUD Application example with Web API (<https://bezkoder.com/angular-10-crud-app/>)

We use cookies to improve your experience with the site. To find out more, you can read the full Newer version: Angular 11 Firestore CRUD example (<https://bezkoder.com/angular-11-firebase-crud-angularfire/>) [Privacy & Policy](https://bezkoder.com/privacy-policy/) (<https://bezkoder.com/privacy-policy/>)

**Contents** [hide]

Angular 10 Firestore CRUD Overview  
AngularFireStore service  
    AngularFireStore for Document  
    AngularFireStore for Collection  
Technology  
    Setup the Firebase Project  
    Setup Angular 10 Project  
    Project Structure  
    Integrate Firebase into Angular 10 App  
    Define Model Class  
    Create Data Service  
    Component for creating Document  
    Component for List of Documents Display  
    Component for Document details  
    Define Routes for App Routing Module  
    Add Navbar and Router View to Angular Firebase App  
    Run & Check  
    Conclusion  
    Further Reading  
    Source Code

## Angular 10 Firestore CRUD Overview

We're gonna build an Angular 10 Firestore App using @angular/fire (<https://github.com/angular/angularfire>) library in which:

- Each Tutorial has id, title, description, published status.
- We can create, retrieve, update, delete Tutorials.

Here are the screenshots:

- Create a new Tutorial:

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

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

[bezKoder](#) [Tutorials](#) [Add](#)

# Angular 10 Firestore Crud

Title

Description

Submit

Cloud Firestore storage after the



Create Operations:

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

bezkoder-firebase ▾

# Cloud Firestore

[Data](#)   [Rules](#)   [Indexes](#)   [Usage](#)
[Home](#) > [tutorials](#) > JoFn0b8iJZev3...

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

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

bezKoder   [Tutorials](#)   [Add](#)

## Angular 10 Firestore Crud

### Tutorials List

- bezkoder Tut#1
- bezkoder Tut#2
- bezkoder Tut#3**
- bezkoder Tut#4
- bezkoder Tut#5

### Tutorial

Title:

Description:

Status: **Pending**

[Publish](#) [Delete](#) [Update](#)

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

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

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

[Accept](#)

## Tutorial

Title

Description

**Status:** Published

**UnPublish** **Delete** **Update**

The status was updated successfully!

```
JoFn0b8iJZev3ZtlUu2m
+ Start collection
+ Add field
description: "Tut#3 Description"
published: true
title: "bezkoder Tut#3"
```

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

## Tutorial

Title

Description

**Status:** Published

**UnPublish** **Delete** **Update**

The tutorial was updated successfully!

```
JoFn0b8iJZev3ZtlUu2m
+ Start collection
+ Add field
description: "Description for Tut#3"
published: true
title: "bezkoder Tut#3 (updated)"
```

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

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

# Angular 10 Firestore Crud

## Tutorials List

Please click on a Tutorial...	
bezkoder Tut#1	
bezkoder Tut#2	
bezkoder Tut#3 (updated)	
bezkoder Tut#5	
	tutorials
	+ Add document
	2ENi1bE9SMMCJWFQgZTi
	JoFn0b8iJZev3ZtlUu2m
	RAVrgiJA7x7EJswYt100
	ZGqCBfmayTiv0bETnvLo
	t2FIH13vG7EQF5TJZcuC >
	+ Start collection
	+ Add field
	description: "Tut#4 Description"
	published: false
	title: "bezkoder Tut#4"

## AngularFireStore service

@angular/fire provides AngularFire service that allows us to work with the Firebase Firestore. It's an efficient, low-latency solution for apps that require synced states across clients in realtime.

```
import { AngularFirestore } from '@angular/fire/store';

export class TutorialService {
  constructor(private db: AngularFirestore) { }
}
```

## AngularFireStore for Document

The AngularFirestoreDocument is a service for manipulating and streaming document data which is created via AngularFire service.

- Create a document binding/ Retrieve:

```
tutorial: AngularFirestoreDocument<any>;
// db: AngularFire
this.tutorial = db.doc('tutorial');

// or
Observable<any> tutorial = db.doc('tutorial').valueChanges();
```

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

- Create/Update a document: [Privacy & Policy \(https://bezkoder.com/privacy-policy/\)](https://bezkoder.com/privacy-policy/) Accept

```
const tutRef = db.doc('tutorial');

// set() for destructive updates
tutRef.set({ title: 'zkoder Tutorial'});
```

- Update a document:

```
const tutRef= db.doc('tutorial');
tutRef.update({ url: 'bezkoder.com/zkoder-tutorial' });
```

- Delete a document:

```
const tutRef = db.doc('tutorial');
tutRef.delete();
```

## AngularFireStore for Collection

Through the AngularFireStore service, we can create AngularFireCollection service that helps to synchronize data as collection.

- Create a collection binding/ Retrieve:

- + Get an Observable of data as a synchronized array of JSON objects without snapshot metadata.

```
tutorials: Observable<any[]>;
// db: AngularFireStore
this.tutorials = db.collection('tutorials').valueChanges();
```

- + Get an Observable of data as a synchronized array of DocumentChangeAction[] with metadata (the underlying DocumentReference and snapshot id):

```
tutorials: Observable<any[]>;
this.tutorials = db.collection('tutorials').snapshotChanges();
```

- Create a collection and add a new document:

```
const tutorialsRef = db.collection('tutorials');
const tutorial = { title: 'zkoder Tutorial', url: 'bezkoder.com/zkoder-tutorial' };
tutorialsRef.add({ ...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

- Update a collection:

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

```
const tutorialsRef = db.collection('tutorials');
tutorialsRef.doc('id').set({ title: 'zkoder Tut#1', url: 'bezkoder.com/zkoder-tut-1' })
```

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

```
const tutorialsRef = db.collection('tutorials');
tutorialsRef.doc('id').update({ title: 'zkoder new Tut#1' });
```

- Delete a document in collection:

```
const tutorialsRef = db.collection('tutorials');
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>).

## Technology

- Angular 10
- firebase 7
- @angular/fire 6
- rxjs 6

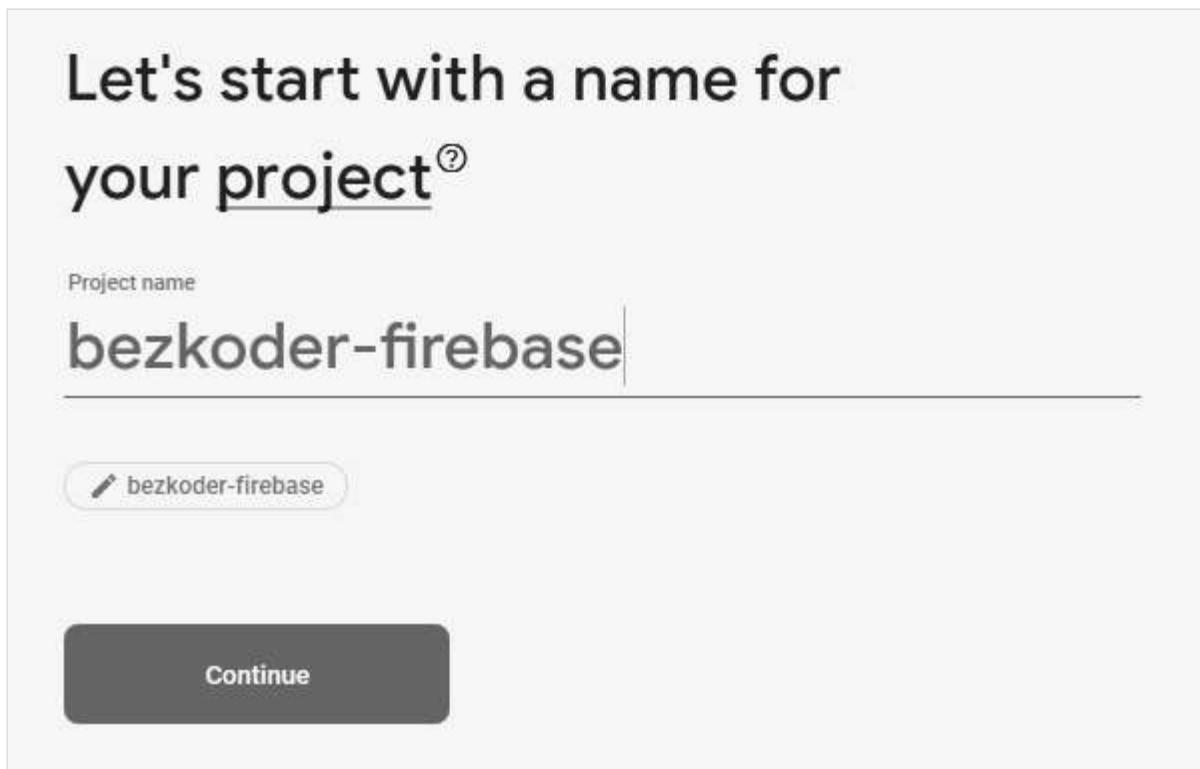
## Setup the Firebase Project

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

You will see the window like this:

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

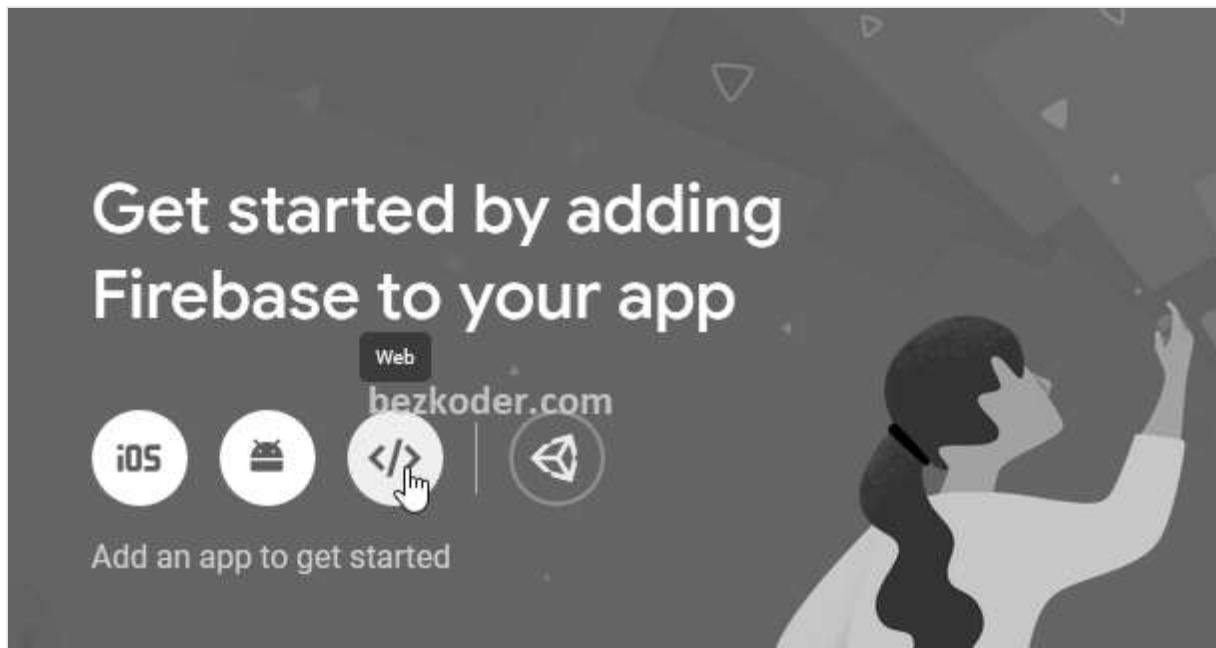
[Privacy & Policy \(<https://bezkoder.com/privacy-policy/>\)](#)      Accept



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:



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

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

## X Add Firebase to your web app

### 1 Register app

App nickname ?

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 \(<https://bezkoder.com/privacy-policy/>\)](#)

Accept



Register app



## 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.firebaseio.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: "AIz[REDACTED]Hw4",
    authDomain: "[REDACTED].firebaseapp.com",
    databaseURL: "https://[REDACTED].firebaseio.com",
    projectId: "[REDACTED]bezkoder.com tutorial",
    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 \(<https://bezkoder.com/privacy-policy/>\)](#)

Accept

The screenshot shows the Firebase Project Overview page. On the left sidebar under 'Develop', 'Cloud Firestore' is listed. The main content area features a large 'Cloud Firestore' heading with the subtext 'Realtime updates, powerful queries, and automatic scaling'. A 'Create database' button is visible.

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

The screenshot shows the 'Create database' wizard. Step 1: Secure rules for Cloud Firestore. It asks to define data structure and write rules to secure data. It provides two options: 'Start in production mode' (radio button not selected) and 'Start in test mode' (radio button selected). The test mode rule is shown in the code editor:

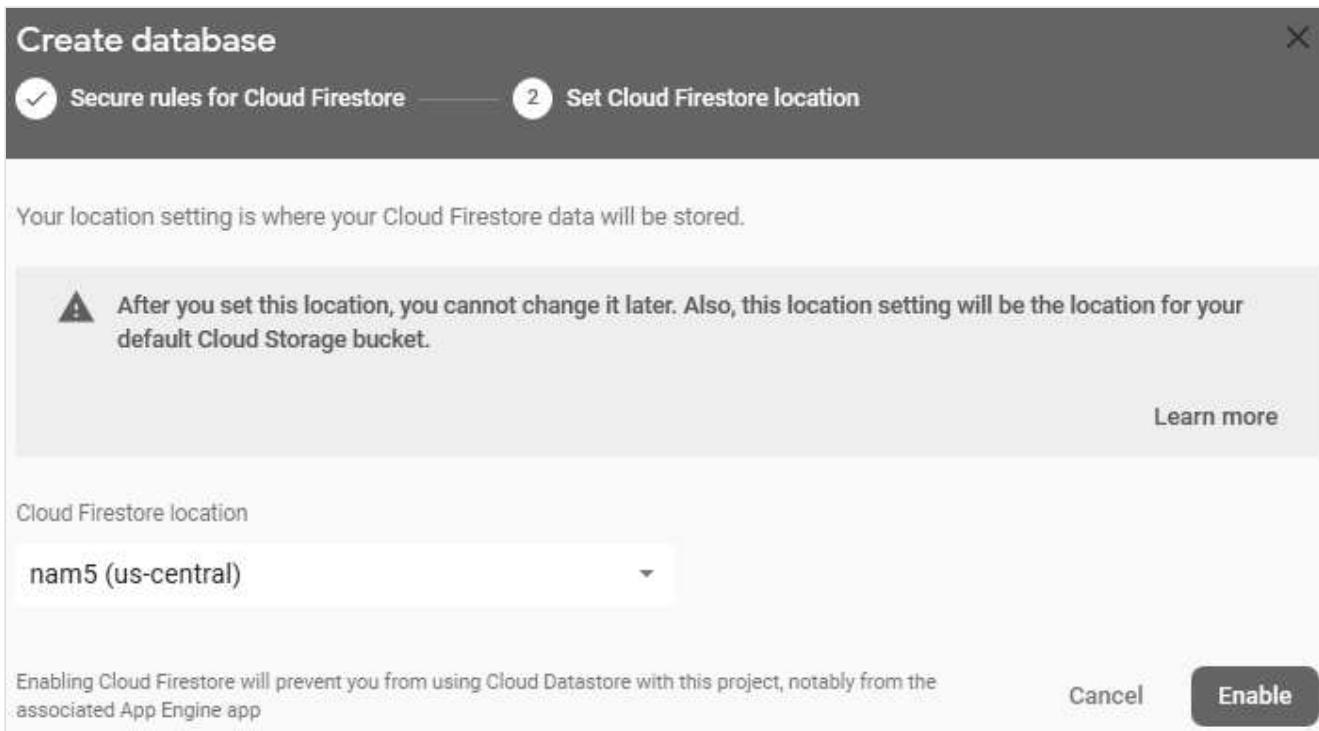
```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if
        request.time < timestamp.date(2020, 10, 3);
    }
  }
}
```

A note below the code states: 'Anyone with your database reference will be able to view, edit, and delete all data in your database for 30 days'. At the bottom, it says enabling Cloud Firestore prevents using Cloud Datastore, with 'Cancel' and 'Next' buttons.

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



## Setup Angular 10 Project

Let's open cmd and use Angular CLI to create a new Angular Project as following command:

```
ng new Angular10FirestoreCrud
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? CSS
```

We also need to generate some Components and Services:

```
ng g s services/tutorial

ng g c components/add-tutorial
ng g c components/tutorial-details
ng g c components/tutorials-list
```

Now you can see that our project directory structure looks like this.

## Project Structure

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

```

    < src
      < app
        < components
          > add-tutorial
          > tutorial-details
          > tutorials-list
        < models
          TS tutorial.ts
        < services
          TS tutorial.service.ts
          TS tutorial.service.spec.ts
        TS app-routing.module.ts
        # app.component.css
        <> app.component.html
        TS app.component.ts
        TS app.component.spec.ts
        TS app.module.ts
      > assets
      < environments
        TS environment.ts
        TS environment.prod.ts
      <> index.html
      TS main.ts
    {} angular.json
    K karma.conf.js
    {} package.json
    {} package-lock.json
  
```

Let me explain it briefly.

- `environment.ts` configures information to connect with Firebase Project.
  - `models/tutorial.ts` defines data model class.
  - `services/tutorial.service.ts` exports `TutorialService` that uses `@angular/fire`'s `AngularFireStore` to interact with Firebase FireStore.
  - There are 3 components that uses `TutorialService` :
    - `add-tutorial` for creating new item
    - `tutorials-list` contains list of items, parent of `tutorial-details`
    - `tutorial-details` shows item details
  - `app-routing.module.ts` defines routes for each component.
  - `app.component` contains router view and navigation bar.
- We use cookies to improve your experience with the site. To find out more, you can read the full  
 - `app.module.ts` declares Angular components and imports necessary environment &  
 modules. [Privacy & Policy \(<https://bezkoder.com/privacy-policy/>\)](https://bezkoder.com/privacy-policy/)      Accept

# Integrate Firebase into Angular 10 App

First run the command: `npm install firebase @angular/fire .`

Open **src/environments/environment.ts**, add Firebase configuration that we have saved when Popup window was shown:

```
export const environment = {
  production: false,
  firebase: {
    apiKey: 'xxx',
    authDomain: 'bezkoder-firebase.firebaseio.com',
    databaseURL: 'https://bezkoder-firebase.firebaseio.com',
    projectId: 'bezkoder-firebase',
    storageBucket: 'bezkoder-firebase.appspot.com',
    messagingSenderId: 'xxx',
    appId: 'xxx'
  }
};
```

Open `app.module.ts`, import `AngularFireModule` , `AngularFirestoreModule` and `environment` :

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 { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';

import { AngularFireModule } from '@angular/fire';
import { AngularFirestoreModule } from '@angular/fire/firestore';
import { environment } from '../environments/environment';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { AddTutorialComponent } from './components/add-tutorial/add-tutorial.compone
import { TutorialDetailsComponent } from './components/tutorial-details/tutorial-det
import { TutorialsListComponent } from './components/tutorials-list/tutorials-list.c

@NgModule({
  declarations: [
    AppComponent,
    AddTutorialComponent,
    TutorialDetailsComponent,
    TutorialsListComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    AngularFireModule.initializeApp(environment.firebaseio),
    AngularFirestoreModule, // for firestore
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }

```

## Define Model Class

Let's create `Tutorial` class with 4 fields: `id` , `title` , `description` , `published` .

**models/tutorial.ts**

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

```
export default class Tutorial {  
  id: string;  
  title: string;  
  description: string;  
  published = false;  
}
```

## Create Data Service

This service will use `AngularFirestore` and `AngularFirestoreCollection` to interact with Firebase Firestore. It contains necessary functions for CRUD operations.

**services/tutorial.service.ts**

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 { Injectable } from '@angular/core';
import { AngularFirestore, AngularFirestoreCollection } from '@angular/fire/firestore';
import Tutorial from '../models/tutorial';

@Injectable({
  providedIn: 'root'
})
export class TutorialService {

  private dbPath = '/tutorials';

  tutorialsRef: AngularFirestoreCollection<Tutorial> = null;

  constructor(private db: AngularFirestore) {
    this.tutorialsRef = db.collection(this.dbPath);
  }

  getAll(): AngularFirestoreCollection<Tutorial> {
    return this.tutorialsRef;
  }

  create(tutorial: Tutorial): any {
    return this.tutorialsRef.add({ ...tutorial });
  }

  update(id: string, data: any): Promise<void> {
    return this.tutorialsRef.doc(id).update(data);
  }

  delete(id: string): Promise<void> {
    return this.tutorialsRef.doc(id).delete();
  }
}

```

## Component for creating Document

This component has a Form to submit new Tutorial with 2 fields: title & description . It calls TutorialService.create() method.

**components/add-tutorial/add-tutorial.component.ts**

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 { Component, OnInit } from '@angular/core';
import { TutorialService } from 'src/app/services/tutorial.service';
import Tutorial from 'src/app/models/tutorial';

@Component({
  selector: 'app-add-tutorial',
  templateUrl: './add-tutorial.component.html',
  styleUrls: ['./add-tutorial.component.css']
})
export class AddTutorialComponent implements OnInit {

  tutorial: Tutorial = new Tutorial();
  submitted = false;

  constructor(private tutorialService: TutorialService) { }

  ngOnInit(): void {
  }

  saveTutorial(): void {
    this.tutorialService.create(this.tutorial).then(() => {
      console.log('Created new item successfully!');
      this.submitted = true;
    });
  }

  newTutorial(): void {
    this.submitted = false;
    this.tutorial = new Tutorial();
  }
}

```

### **components/add-tutorial/add-tutorial.component.html**

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

```

<div class="submit-form">
  <div *ngIf="!submitted">
    <div class="form-group">
      <label for="title">Title</label>
      <input
        type="text"
        class="form-control"
        id="title"
        required
        [(ngModel)]="tutorial.title"
        name="title"
      />
    </div>

    <div class="form-group">
      <label for="description">Description</label>
      <input
        class="form-control"
        id="description"
        required
        [(ngModel)]="tutorial.description"
        name="description"
      />
    </div>

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

  <div *ngIf="submitted">
    <h4>You submitted successfully!</h4>
    <button class="btn btn-success" (click)="newTutorial()">Add</button>
  </div>
</div>

```

## Component for List of Documents Display

This component calls TutorialService methods:

- `getAll()`
- `deleteAll()`

**components/tutorials-list/tutorials-list.component.ts**

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 { Component, OnInit } from '@angular/core';
import { TutorialService } from 'src/app/services/tutorial.service';
import { map } from 'rxjs/operators';

@Component({
  selector: 'app-tutorials-list',
  templateUrl: './tutorials-list.component.html',
  styleUrls: ['./tutorials-list.component.css']
})
export class TutorialsListComponent implements OnInit {

  tutorials: any;
  currentTutorial = null;
  currentIndex = -1;
  title = '';

  constructor(private tutorialService: TutorialService) { }

  ngOnInit(): void {
    this.retrieveTutorials();
  }

  refreshList(): void {
    this.currentTutorial = null;
    this.currentIndex = -1;
    this.retrieveTutorials();
  }

  retrieveTutorials(): void {
    this.tutorialService.getAll().snapshotChanges().pipe(
      map(changes =>
        changes.map(c =>
          { id: c.payload.doc.id, ...c.payload.doc.data() })
        )
      )
    ).subscribe(data => {
      this.tutorials = data;
    });
  }

  setActiveTutorial(tutorial, index): void {
    this.currentTutorial = tutorial;
    this.currentIndex = index;
  }
}

```

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

In the code above, to get the `id` of each item, we use `snapshotChanges()` with RxJS `map()` operator. This `id` is unique and important for update operation.

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

### **components/tutorials-list/tutorials-list.component.html**

```
<div class="list row">
  <div class="col-md-6">
    <h4>Tutorials List</h4>
    <ul class="list-group">
      <li
        class="list-group-item"
        *ngFor="let tutorial of tutorials; let i = index"
        [class.active]="i == currentIndex"
        (click)="setActiveTutorial(tutorial, i)"
      >
        {{ tutorial.title }}
      </li>
    </ul>
  </div>
  <div class="col-md-6">
    <div *ngIf="currentTutorial">
      <app-tutorial-details
        (refreshList)="refreshList()"
        [tutorial]="currentTutorial"
      ></app-tutorial-details>
    </div>

    <div *ngIf="!currentTutorial">
      <br />
      <p>Please click on a Tutorial...</p>
    </div>
  </div>
</div>
```

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-details` component.

## **Component for Document details**

This component is the child of `tutorial-list`. It bind `tutorial` data and emit `refreshList` event to the parent. 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

- update()
- delete()

**components/tutorial-details/tutorial-details.component.ts**

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 { Component, OnInit, Input, OnChanges, Output, EventEmitter } from '@angular/
import { TutorialService } from 'src/app/services/tutorial.service';
import Tutorial from 'src/app/models/tutorial';

@Component({
  selector: 'app-tutorial-details',
  templateUrl: './tutorial-details.component.html',
  styleUrls: ['./tutorial-details.component.css']
})
export class TutorialDetailsComponent implements OnInit, OnChanges {

  @Input() tutorial: Tutorial;
  @Output() refreshList: EventEmitter<any> = new EventEmitter();
  currentTutorial: Tutorial = null;
  message = '';

  constructor(private tutorialService: TutorialService) { }

  ngOnInit(): void {
    this.message = '';
  }

  ngOnChanges(): void {
    this.message = '';
    this.currentTutorial = { ...this.tutorial };
  }

  updatePublished(status): void {
    this.tutorialService.update(this.currentTutorial.id, { published: status })
      .then(() => {
        this.currentTutorial.published = status;
        this.message = 'The status was updated successfully!';
      })
      .catch(err => console.log(err));
  }

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

```

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

```
deleteTutorial(): void {
  this.tutorialService.delete(this.currentTutorial.id)
    .then(() => {
      this.refreshList.emit();
      this.message = 'The tutorial was updated successfully!';
    })
    .catch(err => console.log(err));
}
}
```

**components/tutorial-details/tutorial-details.component.html**

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

```

<div *ngIf="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"
        [(ngModel)]="currentTutorial.title"
        name="title"
      />
    </div>
    <div class="form-group">
      <label for="description">Description</label>
      <input
        type="text"
        class="form-control"
        id="description"
        [(ngModel)]="currentTutorial.description"
        name="description"
      />
    </div>

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

```

```

<button
  class="badge badge-primary mr-2"
  *ngIf="currentTutorial.published"
  (click)="updatePublished(false)"
>
  UnPublish
</button>
<button
  *ngIf="!currentTutorial.published"
  class="badge badge-primary mr-2"
  (click)="updatePublished(true)"
>
  Publish

```

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  
 <button class="badge badge-danger mr-2" (click)="deleteTutorial()">

```

    Delete
</button>

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

<div *ngIf="!currentTutorial">
    <br />
    <p>Cannot access this Tutorial...</p>
</div>

```

## Define Routes for App Routing Module

There are 2 main routes:

- /add for add-tutorial component
- /tutorials for tutorials-list component

*app-routing.module.ts*

```

import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

import { TutorialsListComponent } from './components/tutorials-list/tutorials-list.c
import { AddTutorialComponent } from './components/add-tutorial/add-tutorial.compone

const routes: Routes = [
    { path: '', redirectTo: 'tutorials', pathMatch: 'full' },
    { path: 'tutorials', component: TutorialsListComponent },
    { path: 'add', component: AddTutorialComponent }
];

@NgModule({
    imports: [RouterModule.forRoot(routes)],
    exports: [RouterModule]
})
export class AppRoutingModule { }

```

Add Navbar and Router View to Angular Firebase App

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

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

Accept

Let's open **src/app.component.html**, this App component is the root container for our application, it will contain a nav element.

```
<div>
  <nav class="navbar navbar-expand navbar-dark bg-dark">
    <a href="#" class="navbar-brand">bezKoder</a>
    <div class="navbar-nav mr-auto">
      <li class="nav-item">
        <a routerLink="tutorials" class="nav-link">Tutorials</a>
      </li>
      <li class="nav-item">
        <a routerLink="add" class="nav-link">Add</a>
      </li>
    </div>
  </nav>

  <div class="container mt-3">
    <h2>{{ title }}</h2>
    <router-outlet></router-outlet>
  </div>
</div>
```

## Run & Check

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

Open browser with url: `http://localhost:4200/` and check the result.

## Conclusion

Today we've built an Angular 10 CRUD Application successfully working with Firebase Firestore using AngularFireStore from `@angular/fire` library. Now we can display, modify, delete documents and collection at ease.

If you want to work with Firebase Realtime Database instead, please visit:

[Angular 10 Firebase CRUD Realtime DB | AngularFireDatabase](https://bezkoder.com/angular-10-firebase-crud/)

(<https://bezkoder.com/angular-10-firebase-crud/>)

Or File upload:

[Firebase Storage + Angular 10: File Upload/Display/Delete example](https://bezkoder.com/firebase-storage-angular-10-file-upload/)

(<https://bezkoder.com/firebase-storage-angular-10-file-upload/>)

You can also find how to create Angular HTTP Client for working with Restful API in:

[Angular 10 CRUD Application example with Web API \(<https://bezkoder.com/angular-10-crud/>\)](https://bezkoder.com/angular-10-crud/)  
We use cookies to improve your experience with the site. To find out more, you can read the full app/)

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

Accept

Happy learning, see you again!

## Further Reading

- Angular Template Syntax (<https://angular.io/guide/template-syntax>)
- Angular Router Guide (<https://angular.io/guide/router>)
- <https://www.npmjs.com/package/@angular/fire>  
(<https://www.npmjs.com/package/@angular/fire>)
- Firebase Web Get Started (<https://firebase.google.com/docs/database/web/start>)
- [@angular/fire Firestore docs](https://github.com/angular/angularfire/tree/master/docs/firestore)  
(<https://github.com/angular/angularfire/tree/master/docs/firestore>)

Fullstack CRUD Application:

- Angular 10 + Node.js Express + MySQL example (<https://bezkoder.com/angular-10-node-js-express-mysql/>)
- Angular 10 + Node.js Express + MongoDB example (<https://bezkoder.com/angular-10-mongodb-node-express/>)
- Angular 10 + Node.js Express + PostgreSQL example (<https://bezkoder.com/angular-10-node-express-postgresql/>)
- Angular 10 + Spring Boot + MySQL example (<https://bezkoder.com/angular-10-spring-boot-crud/>)
- Angular 10 + Spring Boot + PostgreSQL example (<https://bezkoder.com/angular-10-spring-boot-postgresql/>)
- Angular 10 + Spring Boot + MongoDB example (<https://bezkoder.com/angular-10-spring-boot-mongodb/>)
- Angular 10 + Django example (<https://bezkoder.com/django-angular-10-crud-rest-framework/>)

Newer version: Angular 11 Firestore CRUD with AngularFireStore  
(<https://bezkoder.com/angular-11-firebase-crud-angularfirestore/>)

## Source Code

You can find the complete source code for this tutorial on Github  
(<https://github.com/bezkoder/angular-10-firebase-crud>).



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