Firebase Firestore — Notes App

A polished, student-ready guide to run a local Firestore CRUD demo with Security Rules and Cloud Functions

Project:	DBMS project
Deliverable:	Local emulator demo — Firestore + Functions
Audience:	Teacher / Course submission

Overview
This document guides you step-by-step to run the Notes app locally using the Firebase Emulator Suite. It covers configuration, initialization, emulator startup, and verification steps to demonstrate CRUD, Security Rules, and Cloud Function triggers.

Prerequisites

• Node.js (v16+) and npm
• Firebase CLI (will be installed)
• A Firebase project (you created: DBMS project)
• Project files downloaded (index.html, app.js, firestore.rules, functions/index.js, README.md)

Add Firebase Config to app.js

1. In Firebase Console > Project settings > Your apps → select your web app.
2. Copy the SDK configuration object (apiKey, authDomain, projectId, etc.).
3. Paste into **app.js** replacing the placeholder firebaseConfig object and save.

```
const firebaseConfig = {
  apiKey: "AIzaSy...YOUR_KEY...",
  authDomain: "dbms-project-7b75f.firebaseapp.com",
  projectId: "dbms-project-7b75f",
  storageBucket: "dbms-project-7b75f.appspot.com",
  messagingSenderId: "764501181894",
  appId: "1:764501181894:web:4c520fc9075b2988fcdec7"
};
```

Initialize Firebase in your project

Open a terminal in the project folder and run these commands:

```
npm install -g firebase-tools
firebase login
firebase init # select Firestore, Functions, Emulators
```

When firebase init asks about overwriting files, choose NO for firestore.rules and functions/index.js to keep the provided files.

Install Cloud Functions dependencies

Run in the project root:

```
cd functions
npm init -y # if package.json missing
npm install firebase-admin firebase-functions
```

Start the Emulators

Start the Emulator Suite (Firestore + Functions). The emulator UI helps you inspect data and logs.

Command:

```
firebase emulators:start --only firestore,functions
```

Emulator UI: http://localhost:4000
Serve frontend: npx http-server . -p 5000 (open http://localhost:5000)

Testing & Verification (what to demonstrate)

• Use the frontend to create a note (title + content) and click Create.
• Confirm the document appears in the Firestore Emulator UI under collection **notes**.
• Watch the functions emulator logs — the Cloud Function should create an entry in **auditLogs**.
• Attempt invalid writes (title length > 100 or missing ownerId) to show security rules rejecting the write.

Key Code Snippets

```
await addDoc(notesCol, {
  title,
  content,
  ownerId: fakeUid,
  createdAt: serverTimestamp(),
  updatedAt: serverTimestamp()
});
```

```
function validNote(newData) {
  return newData.keys().hasAll(['title','content','ownerId','createdAt','updatedAt']
) &&
         newData.title is string &&
         newData.title.size() > 0 &&
         newData.title.size() <= 100;
}
```

Demo Checklist (follow this script)

• Start the emulator: firebase emulators:start
• Serve frontend: npx http-server . -p 5000
• Open frontend at http://localhost:5000
• Create a valid note and show it in Emulator UI
• Show auditLogs entry created by Cloud Function
• Try invalid write to show security rules enforcement

Troubleshooting & Tips

• permission-denied errors: check firestore.rules and whether you are using a temporary demo rule (ownerId == 'user_123').
• Function not firing: confirm functions emulator is running and review terminal logs.
• Frontend shows no data: ensure firebaseConfig.projectId matches and emulator ports are correct (Firestore 8080).