

*CEN 308 SOFTWARE ENGINEERING*



PROJECT DOCUMENTATION

# Online Digital Notepad

**Prepared by:**

**Hasan Tanich**

**Mahadi Babiker**

Date of submission: 21.06.2021

Proposed to:

**Nermina Durmić, Assist. Prof. Dr.**

**Aldin Kovačević, Teaching Assistant**

## [INTRODUCTION](#)

[About the Project](#)

[Project Functionalities and Screenshots](#)

[Project Structure](#)

[Technologies](#)

[Database Entities](#)

[Architectural Pattern](#)

[Design Patterns](#)

[Feature Service design pattern](#)

## [CONCLUSION](#)



# 1. INTRODUCTION

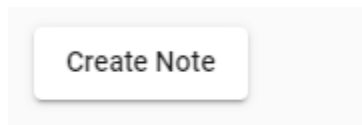
## 1.1. About the Project

A note-taking site is like a digital notebook where a user can create and store notes online.

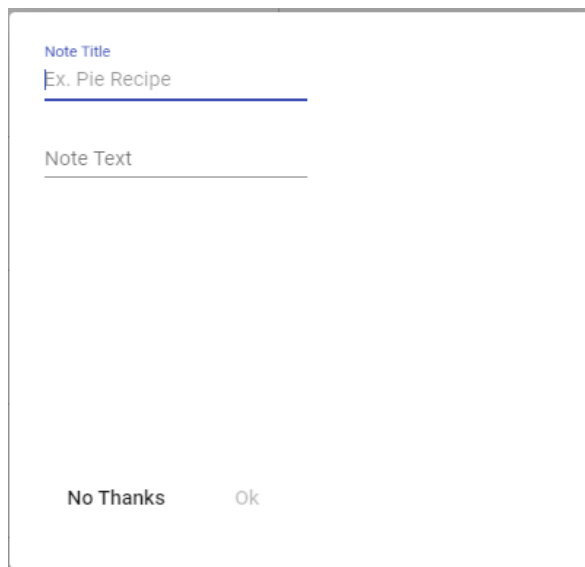
You can jot down text in the site and view it upon returning to the window, editing, or deleting the records if required. The notes can be organized depending on how the user prefers them to be, since the notes have the drag and drop functionality.

## 1.2. Project Functionalities and Screenshots

- Creating notepads



The create note button, will open the below form

A screenshot of a form for creating a note. The form is enclosed in a thin grey border. It has two input fields: "Note Title" with a blue label and a placeholder "Ex. Pie Recipe", and "Note Text" with a grey label. At the bottom, there are two buttons: "No Thanks" and "Ok".

- Being able to save those notepads online

- Being able to manage them (edit, delete)

Pie Recipe	Edit	Delete
Good Movies	Edit	Delete
TV Shows to watch	Edit	Delete

## - Register

login
Register

**online-notepad-app**  
Register your account

Email

Password

Register

## - Login/Logout

login
Register

**online-notepad-app**  
Log in to your account

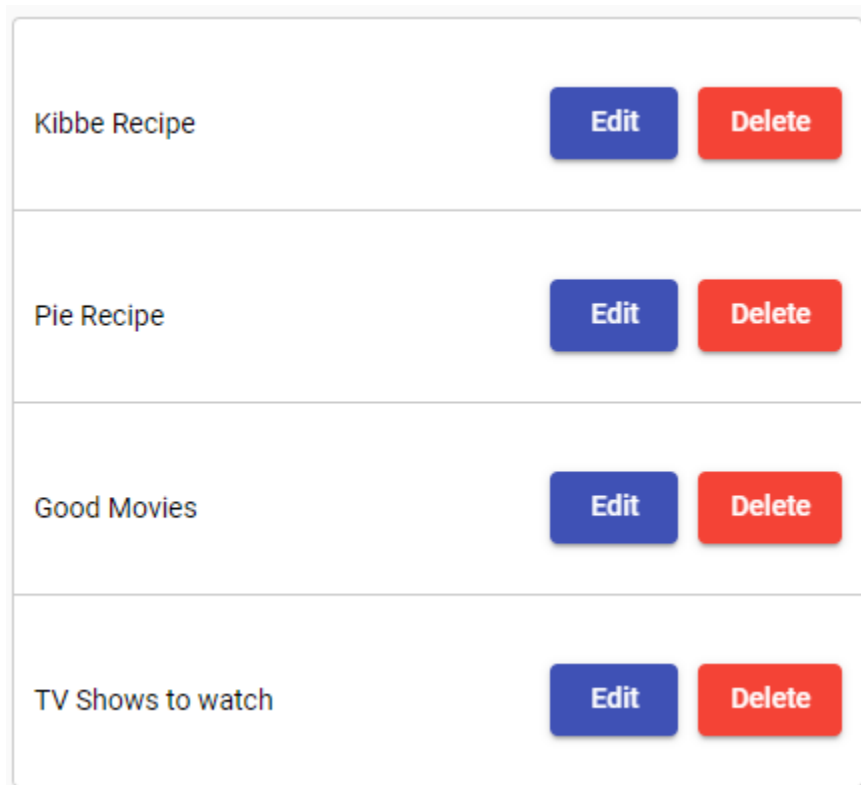
Email

Password

☐ Remember my email address

Login
Reset Password

- Prioritize notepads, using drag and drop feature



- view notepads by selecting one, and selecting their background color.



## 2. Project Structure

### 2.1. Technologies


Programming Languages: Javascript/Typescript, Java, HTML, CSS.

Frameworks: NodeJs, Angular CLI, Angular Material, firebase, Selenium.

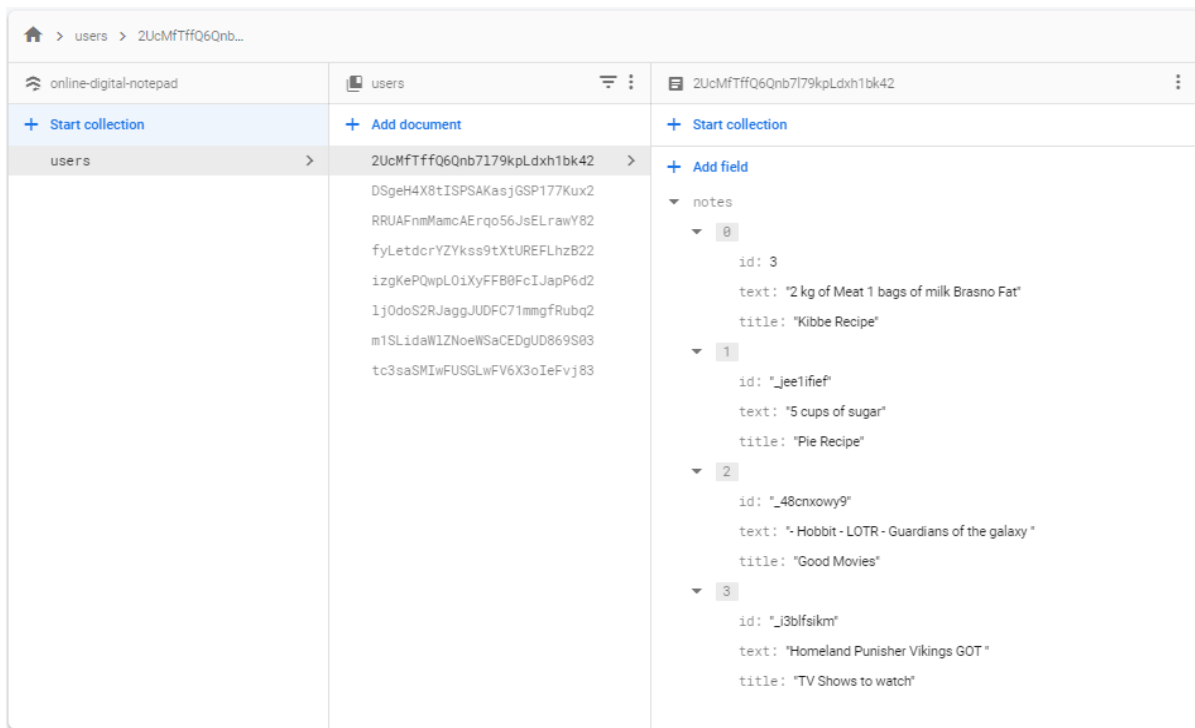
The coding standard we use is: **Angular coding style guide**

### 2.2. Database Entities

Authentication/Users, where we have our users.

Identifier	Providers	Created	Signed In	User UID 
------------	-----------	---------	-----------	--

In firestore databases, we have users collection, where the document id is the userid from Authentication/users table, and each document has notes array, where the user notes are saved.



The screenshot shows the Firebase console interface. On the left, the 'users' collection is selected. In the center, a specific user document is open, showing a list of user IDs. On the right, the 'notes' array is expanded, showing four entries with their respective IDs, text, and titles.

Notes Array Data
<ul style="list-style-type: none"><li>id: 3</li><li>text: "2 kg of Meat 1 bags of milk Brasno Fat"</li><li>title: "Kibbe Recipe"</li></ul>
<ul style="list-style-type: none"><li>id: "Jee1ifie"</li><li>text: "5 cups of sugar"</li><li>title: "Pie Recipe"</li></ul>
<ul style="list-style-type: none"><li>id: "_48cnxowy9"</li><li>text: "-Hobbit - LOTR - Guardians of the galaxy"</li><li>title: "Good Movies"</li></ul>
<ul style="list-style-type: none"><li>id: "j3blfsikm"</li><li>text: "Homeland Punisher Vikings GOT"</li><li>title: "TV Shows to watch"</li></ul>

## 2.3. Architectural Pattern

Since our application is made with Angular framework, we are using an Angular architecture pattern that is defined in three main layers: Core layer, Abstraction layer, and Presentation layer. This is also called High-Level Architecture.

**Presentation Layer** — This layer is mainly responsible for the design and user interface. Here we are defining all our Angular components. It is displaying all templates and handling user events. Presentation layer is also responsible for Unidirectional data flow.

**Abstraction layer** — The abstraction layer decouples the presentation layer from the core layer. This layer will behave as a mediator and the facade for the presentation layer. It means that it will expose an API and coordinate the communication between presentation layer components and the core of the application. For example, here we call services to have data communication between template and core layer.

**Core layer** — This layer is the innermost layer of an application, though all outside communication happens here. In this layer, one can place all state management, core modules, and services. The core layer is responsible for encapsulating the state and behavior of an application. In angular, we have services to write all API calls which contain `@Injectable` to inject APIs to communicate with outside. All services lie in the core layer. In this layer, we could also place any validators, mappers, or more advanced use-cases that require manipulating many slices of our UI state.

## 2.4. Design Patterns

### 2.4.1. Feature Service design pattern

We used the feature service design pattern in our services, the Feature Service design pattern is a way to pull out all of this feature logic from our Feature Component into a single Feature Service. The Feature Service is a Singleton Service that is Injected at the Feature Component level in the component provider. This ensures that the Feature Service is only instantiated once for the Feature Component.

With Angular applications it is common that our data will be exposed via RxJS Observables from either API endpoints or NgRx state management. We pull this logic into our Feature Service, so our Feature Component is unaware of where to get the data or



how to save it. We have thoroughly encapsulated all of the logic of our feature into a single class.

The feature service design pattern is applied all around in our project, since we have a lot of services, and we use them in most components.

**path to one use case:** `src\app\auth\login\login.component.ts`

**path to one use case:** `src\app\my-notes\note-list\note-list.component.ts`

### 3. CONCLUSION

It was a wonderful learning experience for us while working on this project.

This project was a challenging project for both of us because we stepped out of our comfort zones and decided to build a challenging project.

The challenging part that we faced was Firebase (CRUD) operations, we couldn't figure out how to create notes separately for each user but with hard work and long nights without sleep we managed to make it work.

In the future we would definitely change our UI design and add more features like export notes, create notes with drawing, notes with images and much more !