# Noted Web Technical Documentation

## 1. Introduction

### General Project Overview

The web application is the main platform to use Noted. You have access to all the features:

- Note Viewing
- Note Editing
- Recommendation Generation
- Group Management
- Quiz Generation (Coming soon)

### Purpose of the Technical Documentation

The purpose of this technical documentation is to provide detailed information about the project. It is intended for developers and anyone involved in the design, maintenance, or technical understanding of the project. Since the project is open-source, this documentation can serve as a solid knowledge base forfuture contributors.

### Technologies

The Noted website is using various technologies through the frontend and backend but this technical documentation will focus on fronted.

- Typescript: TypeScript is a free and open-source high-level programming language developed by Microsoft that adds static typing with optional type annotations to JavaScript. It is designed for the development of large applications and transpiles to JavaScript. Because TypeScript is a superset of JavaScript, all JavaScript programs are syntactically valid TypeScript, but they can fail to type-check for safety reasons. TypeScript may be used to develop JavaScript applications for both client-side and server-side execution (as with Node.js or Deno). Multiple options are available for transpilation. The default TypeScript Compiler can be used, or the Babel compiler can be invoked to convert TypeScript to JavaScript.

> Official Typescript link: https://www.typescriptlang.org

- React: React (also known as React.js or ReactJS) is a free and open-source front-end JavaScript library for building user interfaces based on components. It is maintained by Meta (formerly Facebook) and a community of individual developers and companies. React can be used to develop single-page, mobile, or server-rendered applications with frameworks like Next.js. Because React is only concerned with the user interface and rendering components to the DOM, React applications often rely on libraries for routing and other client-side functionality.

> Official Dart link: https://react.dev

For certain features such as data tracking, Google integration, and bug tracking, distribution, the application incorporates various Firebase technologies.

- **Firebase**: Firebase is a mobile and web application development platform offered by Google. It provides a set of tools and services to facilitate the development, management, and deployment of high-quality applications. Here's a quick overview of what Firebase is:
    - **Real-Time Database**: Firebase offers a real-time database (Firebase Realtime Database) that allows applications to synchronize and exchange data in real-time between clients and the server.
    - **Authentication**: Firebase provides secure authentication features for users. Developers can setup login systems using Google, Facebook, Twitter accounts, etc., or use email and password authentication.
    - **Hosting**: Firebase offers a web hosting service that enables the rapid deployment of static ordynamic web applications.
    - **Cloud Storage**: Firebase Storage offers secure and scalable storage space for files such as images, videos, and documents, with easy integration into applications.
    - **Cloud Notifications**: Firebase Cloud Messaging (FCM) allows sending push notifications tousers to keep them informed and engaged.
    - **Analytics**: Firebase Analytics provides insights into how users interact with your application,helping make informed decisions to improve user experience.
    - **Crash Reporting**: Firebase Crashlytics monitors application errors and crashes, making it easierto detect and resolve issues.
    - **Performance and Monitoring**: Firebase Performance Monitoring allows monitoring applicationperformance and identifying bottlenecks to ensure a smooth user experience.
    - **Machine Learning**: Firebase ML Kit offers ready-to-use machine learning features to add AIcapabilities to your applications, such as facial recognition, text detection, etc.
    - **Authentication and Authorization**: Firebase Security Rules allow defining security and accessrules for data stored in the database.

> Official Firebase link: https://firebase.google.com

In the Noted Mobile application, we use the following Firebase services:

- **Firebase App Distribution**: This service allows us to distribute the app during the Beta testing phase. It enables us to make the application available to users without the need to publish it on various app stores.

> Official link: https://firebase.google.com/docs/app-distribution

- **Firebase Analytics**: This service facilitates data collection about the app's usage, providing an overall view of how the app is utilized. It enables us to make informed development decisions thatalign with the needs of our users.

> Official link: https://firebase.google.com/docs/analytics

- **Firebase Crashlytics**: This service allows us to centralize the detection of bugs within the production application. It aids in identifying and subsequently resolving various issues.

> Official link: https://firebase.google.com/docs/crashlytics

- **Authentication**: This service enables us to seamlessly integrate third-party login support, such as Google, into the application. Users can create accounts on our web and mobile platforms using their Google accounts.

> Official link: https://firebase.google.com/docs/auth

# 2. Installation and Configuration

## Development Environment Setup

First, you need to clone the "web" repository from the Noted GitHub.

Repository Link: **https://github.com/noted-eip/web-desktop**

Run the following command to clone the repository from a terminal:

```
git clone git@github.com:noted-eip/web-desktop.git
```

Once the repository is cloned, run the following command at the root of the repository:

```
make update-submodules
```

The `update-submodules` command in the `Makefile` is used to update submodules in a Git repository. Submodules are separate Git repositories embedded within another Git repository, acting as separate directories within the main repository. This is the case with the submodule `protorepo` found within the `mobile` repository.

This specific command performs the following actions:

1. **`git submodule update --init`**: This part of the command initializes the submodules, meaning it fetches the submodules specified in the repository's configuration file and initializes them in their current state.
2. **`remote`** : This option tells Git to update the submodules from their remote reference (origin) rather than from the version recorded in the main repository. This ensures that submodules are synchronized with their remote repositories.

In summary, the `update-submodules` command ensures that submodules embedded in your main repository are up to date with the latest versions available in their remote repositories. This can be particularly usefulwhen working with projects composed of multiple submodules and you want to ensure all parts of the project are synchronized and up to date.

Then run the following command at the root of the repository:

```
npm install
```

This command will install all the dependencies that are needed to run the app.

Then you can start the frontend with this command:

```
npm start
```

# 3. Application Architecture

## Overall Application Architecture

React is fundamentally different from several other user interface libraries and web frameworks; it does not have a fixed architectural pattern. Instead, it is fully customizable based on the purpose of the UI, and as a component-oriented approach, it is suitable for the gradual improvement of React applications.

The React architecture primarily handles a web application's "view" layer. Consequently, that gives you a lot of flexibility in organizing your codebase and structuring React components. React components are the basic unit of structure within the React architecture. For example, a button or text label is an example of simple React components, while a user registration form is an example of a more complex one.

# 4. Project Structure

## Project Structure

Once the repository is cloned, you will have access to the project's file structure.The

first level of the structure is common to all Flutter projects:

At the root of the repository, you will find different folders:

- **/public:** This folder will contain the main html file.
- **/src:** This folder will contain all of the Noted frontend codebase
    - **/assets:** This folder contain all the images, icon and fonts
    - **/components:** This folder contain all the TSX components that we use on the app
    - **/contexts:** This folder contain all the React contexts that are used through the app
    - **/hooks:** This folder contain all the functions and types that are used to call the API
    - **/panels:** This folder contain the TSX components for the different panels
    - **/styles:** This folder contain all the CSS files
    - **/views:** This folder contain all the TSX components that create the complete view of the page

You will also find different files:

**.gitignore:** This file lists the files that will be ignored during commits and pushes to the repository.

**makefile:** This Makefile contains several commands for different actions on the repository.

# 5. Main Features

In this section, we will delve into the core functionalities of the application, explaining each one comprehensively. We'll also provide code examples to illustrate how these features are implemented.

## Adding a New Page to the Application

To create a new view, you will simply need to create a new tsx file. This file represents the skeleton of a web page, this is where you are going to call all the components that you made.

For instance, let's create a new page named “Test”

```
// src/views/Test.tsx

import React from 'react'

import 'package:flutter/material.dart';

const <ViewTitle>: React.FC = () => {
  return <TSX Element>
}
```

## Call the API

Call the API to get data from the backend is also a fundamental thing that is needed is nearly all the pages.

Here is an example to get the content of a note:

```
// src/views/note/NoteView.tsx

…
import { useGetNoteInCurrentGroup } from '../../hooks/api/notes'
import { useNoteIdFromUrl } from '../../hooks/url'

const NoteView: React.FC = () => {
       const noteId = useNoteIdFromUrl()
       const noteQuery = useGetNoteInCurrentGroup({ noteId })

       return <div>
                {
                        noteQuery.data ? <NoteViewEditor note={noteQuery.data.note} /> : null
                }
                </div>
}
```

As we can see in this example, we call the “use” function that are hooks who call the backend and fetch the data of a note and with a condition we can chose to display the data to the user or display nothing if there is no data that was fetched.

By understanding and employing these main features, newcomers to the project can gain insight into the app's architecture and its capabilities. This approach enhances collaboration and encourages the development of new functionalities while maintaining code quality

# 6. Dependencies

Regarding the various dependencies of the project, the following are the main dependencies:

- tailwindCSS; an open-source CSS framework. The main feature of this library is that, unlike other CSS frameworks like Bootstrap, it does not provide a series of predefined classes for elements such as buttons or tables. Instead, it creates a list of "utility" CSS classes that can be used to style each element by mixing and matching.

- heroicons: an Icon library

- axios: a promise-based HTTP Client for node.js and the browser. It is isomorphic (= it can run in the browser and nodejs with the same codebase).

- firebase: All the firebase dependencies are explained in the first part

# 7. Best Practices

## React Development

- Using Functional Components and Hooks Instead of Classes, you can use class or functional components with hooks. You should, however, use functional components and hooks more often as they result in more concise and readable code compared to classes.
- Avoid Using State, React state keeps track of the data which when changed triggers the React component to re-render. When building React applications, avoid using state as much as possible since the more state you use, the more data you have to keep track of across your app.
- Organize Files Related to the Same Component in One Folder, when deciding on a folder structure for your React app, go for a component-centric one. This means storing all the files concerning one component in one folder. If you were creating a Navbar component, for example, create a folder named Navbar containing the component file, the style sheet, and other JavaSript and asset files used in the component. A single folder containing all of a component's files makes it easy to reuse, share, and debug. If you need to see how a component works you only need to open a single folder.

## Typescript Development

- Use enums to define a set of named constants and define standards that can be reused in your code base. We recommend that you export your enums one time at the global level, and then let other classes import and use the enums. Assume that you want to create a set of possible actions to capture the events in your code base. TypeScript provides both numeric and string-based enums.
- Use interfaces, An interface is a contract for the class. If you create a contract, then your users must comply with the contract.

# 8. Contribution

Dear prospective contributor,

We are delighted that you are considering contributing to our project. To ensure a smooth and productive contribution experience, please follow these guidelines:

1. **Comprehensive Documentation:** We encourage you to provide comprehensive and clear documentation about the project's overall architecture, key components, and workflows. This will helpnew contributors, like yourself, quickly grasp the project's big picture.
2. **Issue and Task Tracking:** Utilize our issue tracking system (such as GitHub Issues) to identify ongoing tasks and issues in the project. Please make sure to label these tasks with appropriate tags like "enhancement," "bug," "feature," etc. This will assist you in selecting tasks to work on.
3. **Use of Separate Branches:** We strongly encourage you to work on separate branches for each newfeature, enhancement, or fix. This approach simplifies managing your contributions and allows other team members to review your changes before merging them into the main branch.
4. **Submitting Pull Requests (PRs):** When you are ready to submit your contributions, please do so through Pull Requests. This method provides an opportunity for other developers to discuss and review your changes before merging them. Ensure that you provide a clear description of the purposeof your PR, the changes made, and the tests you have conducted.
5. **Code Review Importance:** Your contribution will undergo a code review by other team members.This step is critical to maintaining high quality and consistency in our codebase.
6. **Comprehensive Testing:** We strongly encourage you to include unit tests and/or integration tests to validate your changes. This will ensure that your new features or fixes do not adversely affect other parts of the code.
7. **Adherence to Coding Conventions:** Please ensure that the code you submit adheres to our definedcoding conventions. This includes indentation style, code formatting, variable and function naming conventions, and more.
8. **Explanatory Comments:** Feel free to add explanatory comments, especially for complex portions ofthe code you modify. This will aid other team members in understanding your contribution and in future maintenance.
9. **Respectful Communication:** Finally, we encourage respectful and constructive communication among contributors. Please focus your comments on ideas and solutions rather than personal criticisms.

We look forward to seeing your contributions and collaborating with you to enhance our project. Your commitment will contribute to advancing our work while maintaining the quality of our code and community.

Thank you for your interest and your future contribution!

Best regards,
The Development Tea