Notes Web App

1. Summary

- I am an avid Google Keep user. I love the convenience of being able to open the app and quickly jot down a note. However, due to Google's recent involvement with AI and rumors that they are using our emails and documents to train Gemini, I am becoming increasingly less confident in the security of my notes. Due to this, I've decided to create a note web app of my own, where I can be confident that my data is private and secure.

2. Project Description

- The primary goal of this project is to create a note-taking web app that can work as an alternative to Google Keep, having all of its features and extra features that I desire. The solution to this problem is to create a web app, as stated previously. However, this being a solo developer project, the challenges lie in the limited knowledge and experience I have. To overcome this, I'll take advantage of the vast amount of resources available on the internet to solve trickier problems I will come across when I'm past my knowledge level. Despite these limitations, this project is a necessity for me, since my desire for complete data privacy (and confidence in that fact) far outweighs my doubts or anxiety about taking on this project.

3. Project Scope

- In-Scope
    - creation of the web app UI/UX
    - creation of the web app api
    - creation of the web app database
    - hosting of the web app (database, api, and UI/UX)
- Out-of-Scope
    - Creating a mobile version of the web app

4. Business Drivers

- Desire for data privacy
- Desire for better note-taking features

5. Current Process

- The current process requires the use of Google Keep as the main note-taking app.

6. Proposed Process

- With this proposed note-taking web app, note-taking would primarily take place in the web app.

Highlight Key

Entity

Attribute

7. Functional Requirements

- The user must be able to create an account.
- user must be able to create as many notes as they desire w/n their account
- Users should be able to edit and delete these notes, as well as create them.
- said notes should have a title and a place to write content
- notes should be able to be labeled
- Users should be able to create as many labels as they desire
- The user should be able to filter through notes using a search bar. If a user types a word and/or sentence that is found in a note, it should show up in the text result.
- Users should be able to delete their accounts, taking all their notes with them.
- The user should be able to share notes with other account holders; however, they will be view-only unless the person gives them editing privileges.

- When a label gets deleted, only remove the label from the notes that have said label, not the notes themselves

8. Non-Functional Requirements

- notes should be customizable through a variety of means, like different note colors, different font types (like a heading and a section and whatnot), different backgrounds, many different text customization features like making text bold, italics, underline, cross strike, changing the font size, color and type, etc... notes should be able to contain sections and checkboxes that can be added in a section
- Users should be able to customize the way their ui looks, whether it be by having a primary and accent color, a background pic, or a custom web app font.
- The user should be given a set of pre-made ui looks.
- Users should have the ability to hide/show notes; these 'hidden' notes would go in an archive.
- The user should have a profile settings menu, where they can implement the previously stated customization features.
- user should be able to filter the displayed notes by their tags, by their creation date, owner
- Users should be able to pin certain notes to the home screen.
- The user should have different ways to display the notes: list, compact, grid, details, and views.
- The user should have the ability to delete their account on the plain site, not hidden in many settings.
- Users should have the ability to attach images in the middle of the text. In Google Docs, the images immediately get grouped at the top.
- Deleted notes should have a 30-day delay and have the option to be deleted permanently b4 that time period.
- Labels should be editable and deletable.

Entities

- User
  - Username
  - Password
  - UserID
- Note
  - NoteID
  - Title
  - TextContent
  - pinned
  - Hidden
  - Cosmetics
  - ViewOnly
  - CreatedAt
  - UpdatedAt
  - UserID
  - LabelName
  - Deleted
  - timeLeftBeforeDeletion
- Label
  - LabelName
  - UserID
- NoteColor
  - userID
  - colorHex
- UITemplate
  - templateID
  - templateName
  - TemplateDetails
  - userID

Commented [KL1]: I decided to use auto generated IDs instead of using the user's username as the pk since id like the users to have the ability to change their usernames. Due to this ability, if a user changes their name it would have a cascading effect; ideally pks should be permanent

Commented [KL2]: I decided to have this attribute in order to be able to separate the notes that are deleted from the ones that arent, also to allow for the timeLeftBeforeDeletion attribute to have a trigger to start the 30 day countdown as well as a way to reset it

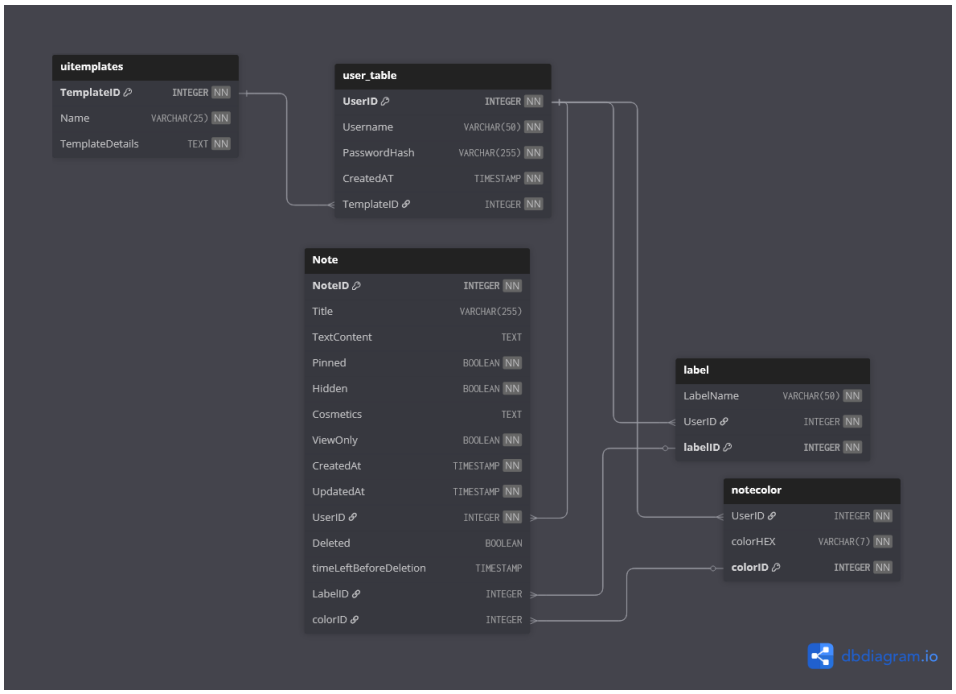
Commented [KL3]: I decided to turn this UI/UX feature into an entity since I'd like the users to be able to save colors they'd like to reference later

Use case: User wants all notes to have the same background color

Commented [KL4]: I made this into an entity since the templates will be referenced across all users and I'd like the users to be able to create their own down the line

Commented [KL5]: For now I will leave this entity like this, since I have yet to figure out how im going to create and store said templates. However I know im going to need to know what template the user ends up using, hence why I have a userID attribute

Physical Diagram



Commented [KL6]: Unfortunately, the free diagramming tools I found online only support physical diagrams, not logical ones, so I skipped directly to a physical diagram. I used FluxStack.io to make the ERD and transferred the uploaded its generated sql into dbdiagram.io to get a png of the diagram