Specification — Notes managing and sharing Web App

March 22, 2025

The goal of this Web application is to provide a platform where users can write and organize their notes on some particular topic, lecture, article or book and subsequently share these notes in between each other.

Functional requirements

After registering an account and logging in, the user will be able to create, edit and manage their notes in a tree structure. All notes will have an access value associated with them which the user will be able to change within their own notes. The four values will be: public (available to all), friends-only (available to all the user's friends), group access (available to the users in the groups specified in the notes file) and private (available to the file owner alone).

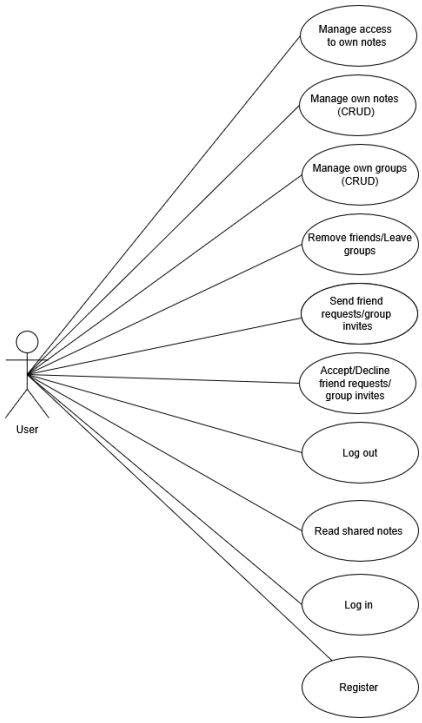
Both logged in and logged out users will be able to search through existing accounts and read notes which are accessible for them. A logged in user may view the files available to a group he's a member of in a separate view. A user will not be able to manage notes which are not owned by them.

A logged in user will be able to send friends requests to another user which can be accepted or declined. A friendship relation will allow the users to share friends-only notes between each other. A user can at any time end his friendship with another user.

Use Case diagram

Constraints:

- Read shared notes, register and log in don't require the user to be logged into an account. The latter two use cases in particular require the user to be logged out.
- A successful log in requires a valid username and password pair associated with an already registered user account.
- All other use cases require the user to be logged in.



Data model

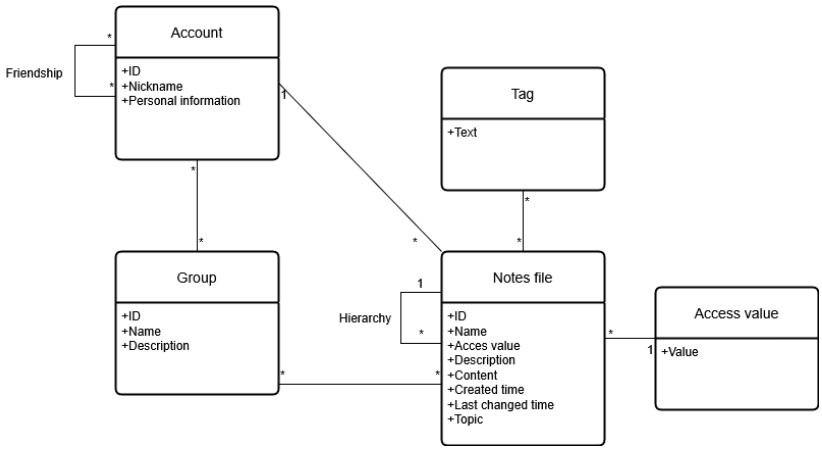
Entities and relations

Account

- A user Account is created after registration and is identified by an unique username. An Account is protected by a password.
- It may hold personal information that the user wishes to relate, such as a bio, full name, email, country of residence and so on.
- It holds the user's personal Notes files.
- It also holds a list of Groups which the Account is a member of.
- Accounts can become friends. Each account holds a list of all of their friends.

Friendship relation

- The Friendship relation allows the users to share files which hold the friends-only access value.
- A Friendship is initiated by a friend request, which can be either accepted or declined.



Notes file

- Notes files are files which hold user managed content.
- Besides the content itself, they also hold values such as the name of the file, created/last changed time, description, the topic of the notes, the list of groups associated with the Notes file, the access value and tags, which can include the language of the notes etc.

Hierarchy relation

- Hierarchy relation allows Notes files to be organized in a tree structure. All Notes files can thus serve the function of folders or organizing units.

Group

- The Group is a mechanism with which the user can share his notes with a group of other users.
- Accounts can be a member of multiple groups and a Notes file can be made to be associated with multiple groups.
- A user may create Groups and invite other users to join them.

Architecture

The app will use the server-client architecture and the SPA (Single Page Application) approach.

Technological requirements

- Client side: HTML5, JavaScript, CSS, React 18
- Server side: Node.js 23, JavaScript
- Database: PostgreSQL 17
- Interface: REST API
- Hosting: render.com
- Supported browsers: Google Chrome, Firefox

Time schedule

Week 4

- Learning the required technologies, begin frontend.

Week 5

- Begin backend, develop user authentication and file editing.

Week 6

- Develop group management and friendship functionality.

Week 7

- Develop user access control.

Week 8

- Finish up frontend, test.

Week 9

- Beta version.

Week 10

- Receive and implement the feedback from beta testers.

Week 11

- Finishing touches.