Sarah Edris
6.170

3.2 Sticky Notes

Overview

Purpose and Goals

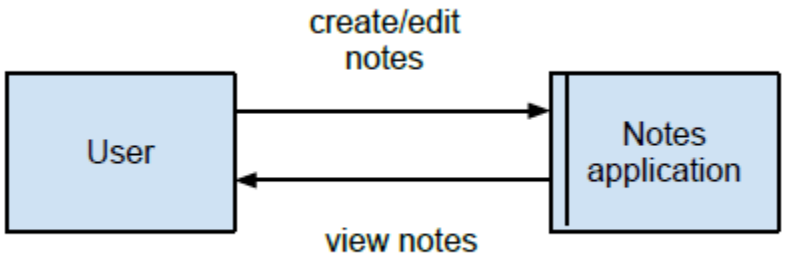
Notes is a lightweight sticky note web application that allows for users to easily create online notes that can be looked at and edited later. The app can be used immediately upon coming to the site, as all actions are done on one webpage using AJAX requests. Existing apps such as Google Keep and listhings already provide a note feature, but these tools have many extra features that are useful, but sometimes not needed, like font editing or adding of pictures. A key purpose of this application is to create something easy to use, and to learn more about JavaScript and AJAX.

Purpose and Goals - Extension

Notes also has the ability to create to-do list notes, in addition to regular notes. This allows for a wider range of operations and makes the app more useful for the average user, as creating to do lists and marking off items is something that people often do.

Context Diagram for both MVP and extension

Users have the ability to create, edit, and delete their notes, and can use the application to view their notes.



Concepts

Key Concepts

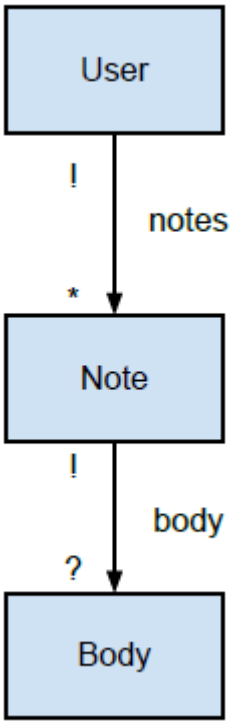
The key concept in this application is a **note**. Each user has a collection of notes, which each have a body that holds the formatted text for that note, x and y position so that it can be displayed in the same location each time the application is loaded, and a width and height so it has the same dimensions.

Key Concepts - Extension

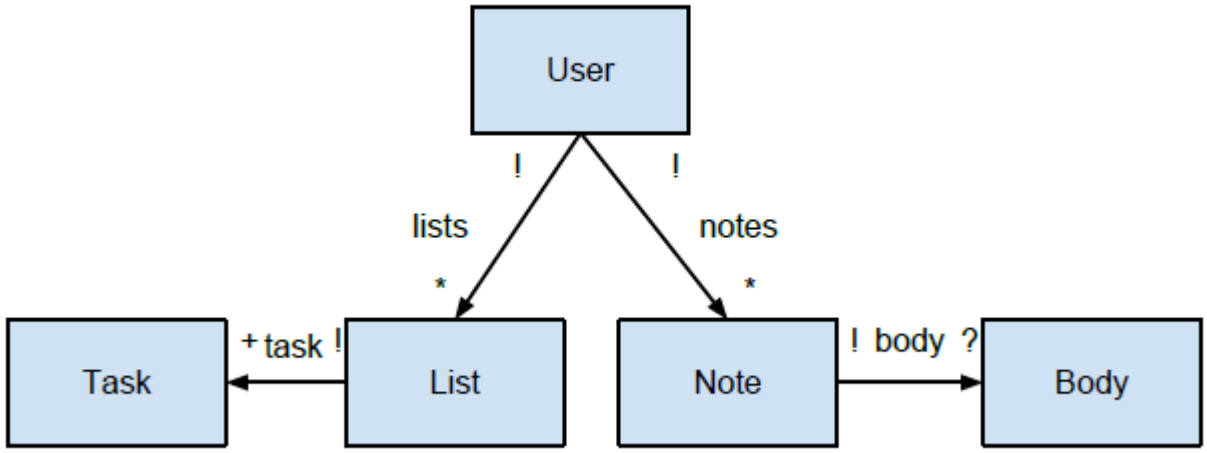
The key concepts in the extension are **notes** and **lists**. Each user has a collection of notes and lists, each note has a body for text, and each list has a collection of **tasks**, which are either completed (marked as done) or not. Each list also has an x and y position, width, and height.

Object Model

MVP



Object Model - Extension



Behavior

Feature Descriptions

- **Signing up/Logging in:** The currently logged in user is saved using a session. All users must log in to access the site.
- **Note creation/deletion:** A user can create a note by clicking 'New Note', and delete a note by clicking on the 'X' in the top right-hand corner.
- **Note dragging:** A note can be dragged around the page, and its position saved once the user has stopped dragging it.
- **Note resizing:** A note can be resized by dragging the lower right-hand side triangle, and the new dimensions are saved once done resizing.
- **Note editing:** A note's content can be edited by clicking on it, typing, and clicking somewhere off the note once done.

Feature Descriptions - Extension

- **To do list creation/deletion:** A user can create/delete a to do list and edit it in the same manner as a note.
- **Adding/Deleting/Completing Tasks:** Each time a user presses the enter button when typing in the list, a new task is created, with a small checkbox on the left side to signify this. The list starts with one empty task. A user can add tasks to be completed to a to do list, delete tasks, mark tasks as complete (which draws a line through the text and checks off the box), and mark completed tasks as not complete (which removes the line and unchecks the box).

Security Concerns

Notes requires that each board and its notes are only visible and editable by the user that created it. The application may be vulnerable to the following threats:

- An attacker may attempt a cross-site scripting attack by entering malicious text in a note, or tricking a user to do so. The text is then displayed on a board and can lead to the execution of a malicious script. This problem is mitigated by the sanitization of the text for a note.
- An attacker may attempt directly going to the URL for a user's boards in order to edit them. This attack is mitigated by requiring users to log in before seeing their own notes, and only allowing a board to be reachable by the user that created it.

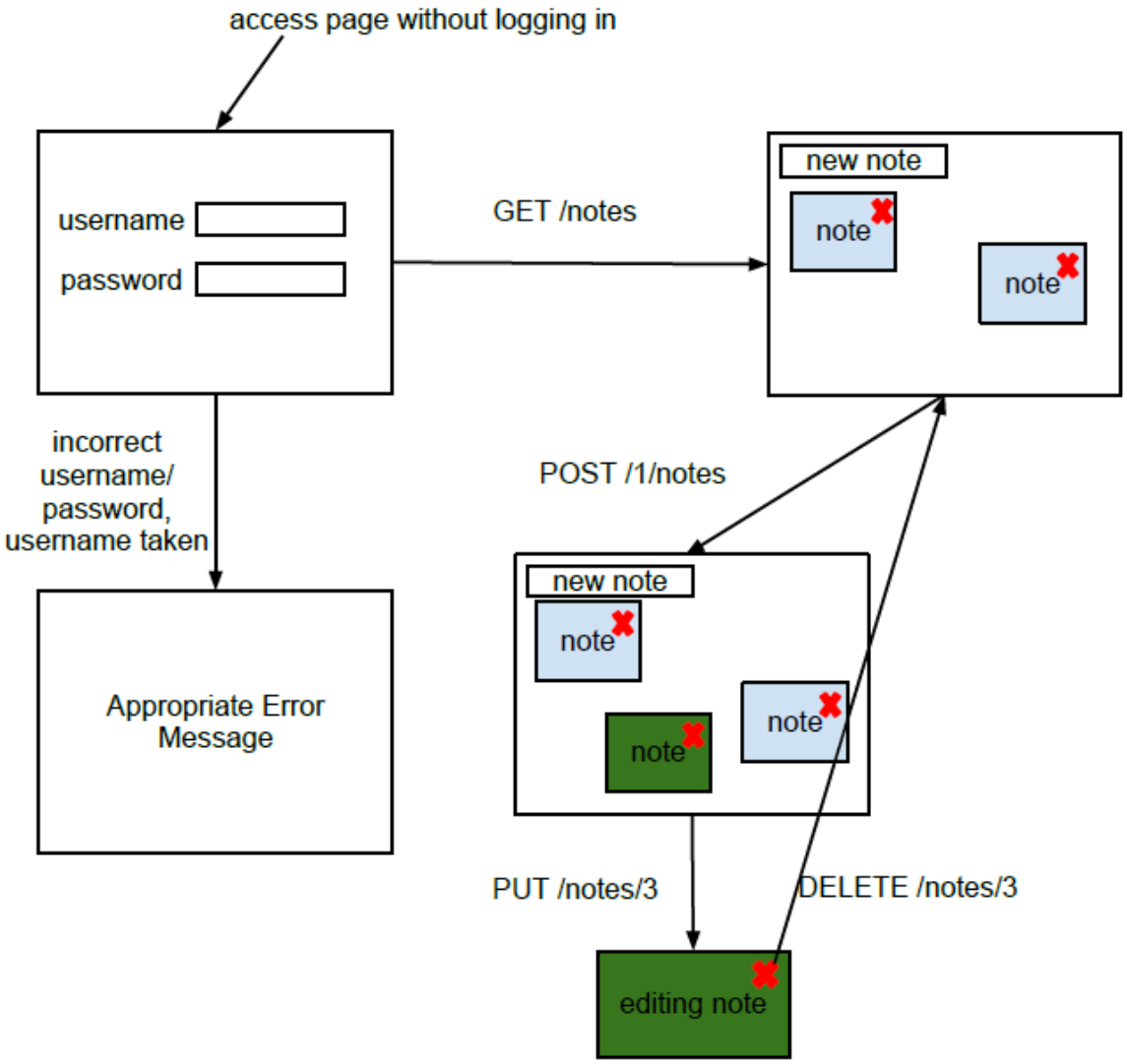
Security Concerns - Extension

The security concerns for the extension portion of this application are the same.

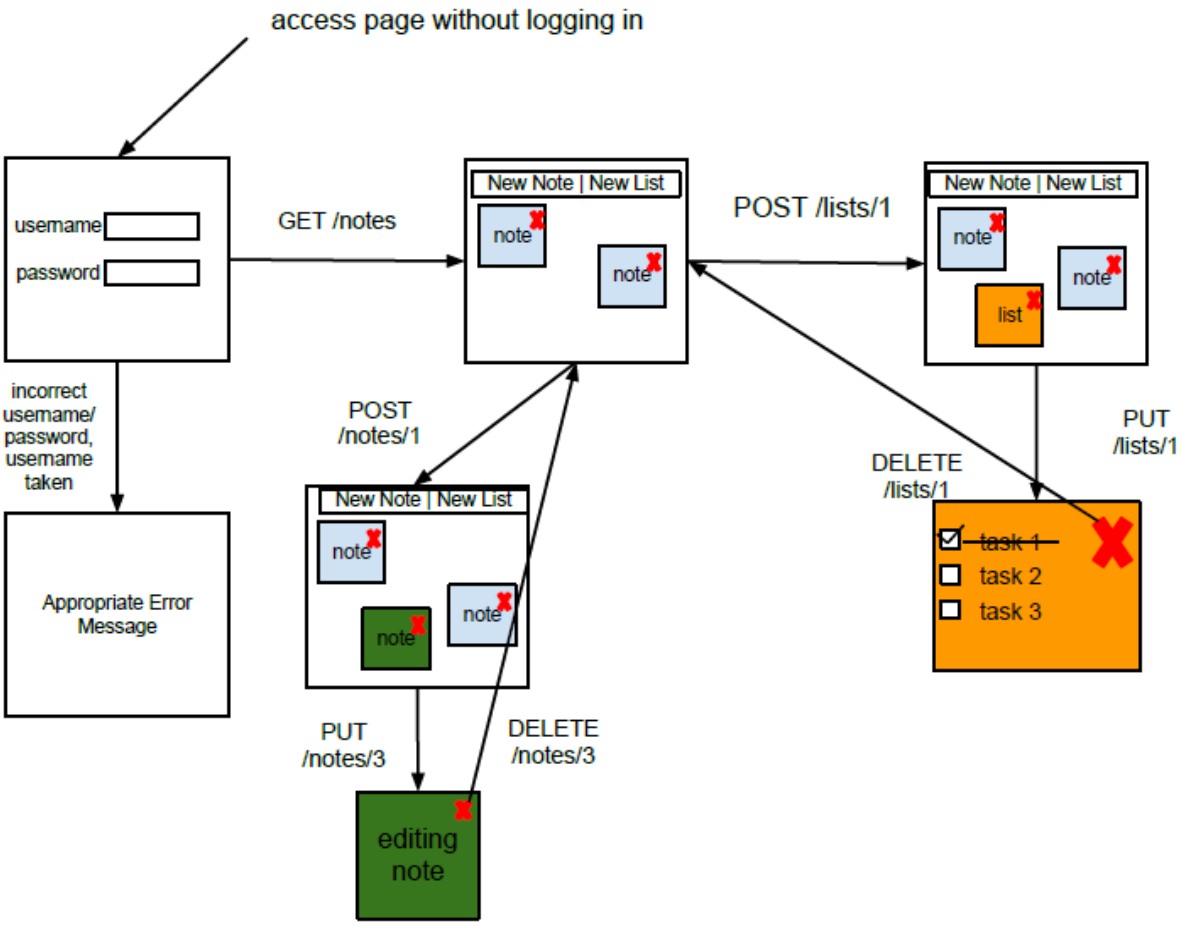
User Interface

Note: all actions are done using AJAX requests on the same webpage

MVP:



User Interface – Extension



Challenges

Design Challenges

Privacy of boards and access control mechanism

To enforce that only a collection of notes' creator can view and edit it, I considered the following options:

1. Require sign up and log in by users
2. Use a secret access key, which is emailed to the user

Notes are meant to be easy to access immediately, without need of finding a secret access key in one's email. Using a key would be better if notes were meant to be something only accessed a few times, then never again; but

instead, users should be able to easily access them over long periods of time, during which it's easier to just remember a login, which is why I chose option 1.

Positioning of notes and lists

I want to make notes and lists can be dragged around the page to help recreate the feeling of a real push pin corkboard, so they must save position in some way.

I considered the following for the positioning of notes and lists:

1. Saving order of creation, and displaying by order
2. Display by no order
3. Saving position and making notes draggable

The first two options don't involve making notes draggable, and the second option is the easiest to implement since notes are arbitrarily displayed. This isn't very user intuitive or friendly though. Displaying by order creation is more useful for the user, but still doesn't allow for any modification. I ruled out the first two options because they don't allow for changes by the user, and I want to recreate the feeling of a real push pin corkboard of notes, where notes can be moved around. The third option allows for this and allows for notes to be displayed in relatively the same location on different screen types.

Design Challenges - Extension

Saving of List Items

Lists contain tasks that can be marked as completed or not with a strikethrough effect, and have a checkbox next to them. I considered the following options for how to do this:

1. Save lists as notes, and simply change how they are displayed on the front end with javascript
2. Save lists as a different type of object, containing tasks that are marked as completed or not

The first option doesn't require much or any changes in the Rails backend, as the javascript simply marks lines with a strikethough or not depending on if the user selects it as done. The second option requires changes on the backend, but saves more information about the state of the list and what items are done. This information can be gathered from the first option, but the body must be parsed through and HTML and CSS examined to determine which tasks are marked with a strikethrough effect or not. The second option also allows for the application to be further extended in an easier manner, like if I want to have all completed tasks later displayed somewhere else on the page. This is why I decided on the second option.