This system is a sticky note application. I wanted to create something slightly different than the normal note taking application that everyone else would make, So this uses much smaller notes, that are in-line editable and that each have titles. Theres a save button on each note, which needs to be pressed in order for the database to be updated. There is also a delete button which removes both the physical note and the note as represented in the database. Each note also has a + button which creates a new note. This note is created on top of the existing note slightly below and to the side. Each note can also be dragged, and it will be brought to the top of all other notes. I am also using an Ajax call that will save the x and y coordinate of where the note was last dropped.

Each note has a :name, :title, :content, :lastX, :lastY entry in the database. The title and content are obvious, and right now every note has one. LastX, LastY, and LastZ represent the x, y, and z coord of the top most corner of the note. Their purpose is that when a user drags and drops a note, the position of the note will be saved so on return they will be placed their again. Also would be used when a new note is created it will be placed on top of and slightly to the left and down of the note that was clicked on.

Design Challenges:

Challenge 1:

How do I represent the relationship between Users and notes?

Overview:

Any user can create any note. Any user can share the note that they created with any other user. Any user that is shared a note can read/edit that note. However they cannot delete it. They can delete their relationship with that note, but only the originally creator may delete that note. Any Collaborator may also add another collaborator to their note.

Options:

1. Create different user types: Owner, and Collaborator.
2. Create a seperate Object Considered a Collaboration. Each Note would hold one collaboration, and each Collaboration would have and belong to many users.
3. Create a Has many and Belongs to Many relationship between Posts and Users
4. Create a Has many through relationship between Posts, and Users through collaborations, as well as creating a create user

Evaluation:

Option 1: I strayed from this option because I felt that it created an unnecessary different user type. Although Owners and Collaborators interact with the same note Differently, I didn't think that this project required enough different functionality to implement two user types. It seemed like unnecessary work in the scope of this project

Option 2: This was my initial plan. I wanted to create a separate object Collaboration. Each post would have a has_one relationship with a collaboration, and each collaboration would belong to a post. Then each collaboration would have many and belong to many Users. I chose not to follow this path because after planning it out and thinking the relationships, a collaboration had no Unique attributes, or responsibilities aside from acting as the connection between posts and Users. And thus seemed pointless

Option 3: While I wanted to do this, and use this relationship, I felt that in the scheme of my relationships with users and notes, it would be hard to distinguish the difference between a

creator and a collaborator. I felt it would be easier to create a relationship between users and posts which lead me to the final, and implemented option.

Option 4: I decided to create a relationship table which was called a Collaboration. This way I would have a seperate table that related all posts with users that are collaborating on that post. And This allowed Users to access all the posts they are collaborating on through their collaborations. Also, I had already implemented a has many and belongs to relationship between the Users and notes when notes were created so this allowed me to keep this, which would then keep track of the creator of the notes.

Challenge 2:

How Will I handle the UI of notes when represented as collaborations?

Overview:

the UI of the notes was created before the concrete details were planned out, and thus I had a loose idea of the end goal framework of the application while creating them. It required a lot of work to set up the notes as they were, and I had to decide whether I wanted to show a user Collaborative notes In the same fashion as other notes.

Options:

1. Show notes and collaborative notes on the same page
    1. Have A separate class/interaction with the collaborative notes

2. Have Collaborating notes look the same (buttons and all) as notes on the posts page.
    1. Create a separate page to display collaborations

Evaluations:

Option 1: I quickly realized this would be a dumb way to represent collaborations because it would not allow the user to realize whether a note was one they created or collaborating on. So then I had to decide whether I wanted to create a separate UI for collaborations that way I could display all notes on the same page, and have a user tell the difference. I chose against this because it seemed like extra unnecessary work to create a different looking note, when its functionality would be almost identical.

Option 2: I decided that in order to avoid creating a new UI for the collaborations I would need to display them on a separate page. So I created a separate link to collaborations and allowed users to see these notes here. However I needed to keep in mind how the user would interact with these notes. I decided that the save and add buttons would act the same since there was no need to change it, but I had to edit the posts destroy method so that if I was creating a request to destroy a post that I did not create the collaboration would be deleted instead. This was decided to avoid needing to create different ajax requests when pressing buttons.