For my Rails project I decided to make a task manager web app. I wanted a user to be able to have its own tasks, and also be able to join a group of users too, where the group would have its own tasks. Because this made it so there were two distinct types of tasks, one that belongs to a group, and one that belongs to a user, I made two different task models. One I labelled Task and one I labelled Group\_Task. Unfortunately, this circumstance made my app at least twice as big as it would have been had I simply had a group with tasks, or a user with tasks.

I wanted all the tasks to be able to be associated with a goal. The idea being that a group can have a goal that they work towards, or a user can have different tasks associated with a goal or many goals. Each task also has the ability to be completed with a Boolean attribute for complete, and can be commented on, so that the user or other group members can see a comment on the task. One thing I would have liked to do is to have created a comment model for the tasks, but this ended up making the app that much bigger, and I figured a comment attribute was enough.

It became very apparent early on that my general understanding of associations, belong\_to, has\_many, etc. did not fully appreciate the implications that those associations might have as the app developed. This was apparent in my struggle to have one task model be for both users and groups. The first compromise that needed to be done, was to make the group\_task option belongs\_to group association, optional: true. This allows for users to create tasks in the group, and allows them to leave the group without taking all of their tasks with them. This was something that I kept when I created separate group\_task and task models. However, something I could not get around was viewing my tasks on the show page. Because the task model didn’t have a way of separating group from user tasks, and creating an attribute to separate them seemed to cause too much logic being necessary on the show page and also made the app very error prone, I opted for creating separate task models for user and group. Now I was able to write cleaner more concise code that separated the group and user concerns.

However, instead of creating different Goal models for users and groups, I opted to just have one Goal model and controller. This made more logic necessary in the controller in order to sort out whether the goal is meant for a user or a group.

My scope method is to see which user has the most tasks left to complete. The app also uses github’s omniauth, and regular sessions login and signout routes.