Use cases:

- I am basing this TODO list based on my own experience
- I find that I constantly maintain separate plaintext TODO list for each of the projects that I do. I also find that I keep making new TODO lists as each project list gets unwieldly, which potentially leads to redundancies in the list, and missed relationships between entries.
  - This leads to tagging being a possible solution, which will allow for per-entry search
- I also frequently nest my list into sub lists, to cover related tasks, and single entries might touch on multiple topics
  - This implies that tags should have a tiered structure.

Current plans for the project:

- Desired features:
  - Basic features of list and tagging
    - Tagging can of course be unlimited, both in total numbers in tag cloud and per item
  - For now, a flat list is planned, but multi-level lists are in the works, but potentially difficult due to complex database schema's. This is desired due to both trying to satisfy the expected use case of huge
    - Nested list will require a list-entry object that can accept a list id xor a item id. Research on how to set up this database schema is still ongoing
      - Alternatively, I might make a list identical to a item, with the only change being that a entry can either be done, not-done, or a list, but this might make the database much more complicated and/or more suited to a non-relational schema.
    - Alternatively, I might instead break the list up into smaller sub-list using the tag system, and design the front end around it
      - <u>This is in line with my issues with multiple/unwieldly lists for each project I do.</u>
      - <u>In writing this document, the idea of putting the tiered structure into the tags rather than the entries themselves suggest itself. More experimentation is required.</u>
  - Storage is planned to be either explicit saving or save as you go. In both cases, the default option to access the list again is an autogenerated human readable ID, like "21TerribleBananasRunFast", or potentially a user selected username.
    - Alternatively, taking inspiration from another website, I could try to store the data as part of the URL, for privacy conscious folks. See: https://github.com/topaz/paste

Current problems:

- Initial hiccups with comprehending the JSX syntax and trying to understand the workflow asked of me. Particularly, the logic behind the rails generators and file structure of the project was opaque to me before research, and my mental model of how the rails server actually work is still lacking in many areas. More research is needed before I can reason about the workings of the rails server to my satisfaction.
- I somewhat underestimated the time needed to perform the project, in light of my other obligations laid bare to me. Crunch time is likely required.
- Some issues related to project setup. I think I haven't set up the test/production project configuration properly, due to the rails project creation frontloading too many concepts at once. I will have to take another look at it later, once I start trying to host this online.

Anticipated problems:

- There are two software development task that I have little to no practical experience with: Databases, and Online Deployment.
  - Database configuration might be a problem, but I think Active Records will probably make it more streamlined than usual. There might still be some problems with trying to reconcile my mental models of using the database command line and setting up Active Records.
  - Online Deployment is potentially more problematic, since I have zero experience with it, largely due to cost, and personal focus on projects with lower barriers to entry. Although cost is not a problem with Heroku, this is still a task I have absolutely no prior experience with, so I have no idea what to expect at all, unlike Databases, which I have some experience with the theory at least.