Shirk

6.170 Project 3 Design Draft

Tricia Divita

Overview:

Description:

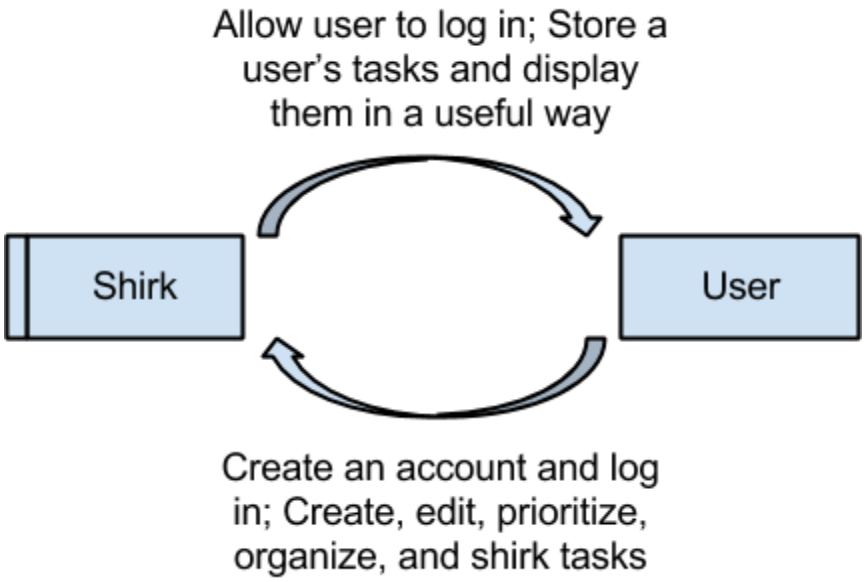
Shirk is a todo-list web application which allows users not only to create, manage, and prioritize lists of tasks to be done, but also allows them to easily shirk their responsibilities by pushing them off until later (while still reminding them that the tasks need to be done eventually). It will seek to handle task management better than many other options that exist by allowing for postponement of tasks while also allowing longer-term planning and maintaining organization.

Purposes:

1. **Help users remember what they need to do.** Allow a user to keep track of all the tasks that they need to do, and which ones they have already completed.
2. **Help users remember when they need to do things.** Make it easy for the user to keep track of what has to be done when by having a completion date associated with each task. This will not make sense for all tasks, so make the date part optional on a per-task basis.
3. **Let users procrastinate.** Allow users to easily shirk tasks by putting them off to be done at a later time. At the same time, make sure that they don't simply forget to do them entirely by continuing to remind them of the tasks, rather than simply having them go unnoticed if they are not done on time.
4. **Help users get organized.** Facilitate creation of multiple lists of related tasks (tasks to be done this week, errands to run, chores to do around the house, etc.). This may help users to not be overwhelmed by all of their tasks at once, and also to keep them straight in their head better, since they can think about them in a more organized fashion.
5. **Focus on important tasks.** Let users prioritize their tasks and easily view the most important ones.

Context:

- Shirk should not really require interaction with any other system. There could be a possibility of adding text or email reminders for tasks, which would require use of some external service, but that is not part of the current plan for the project.
- The basic flow of interaction with the application will be as follows:

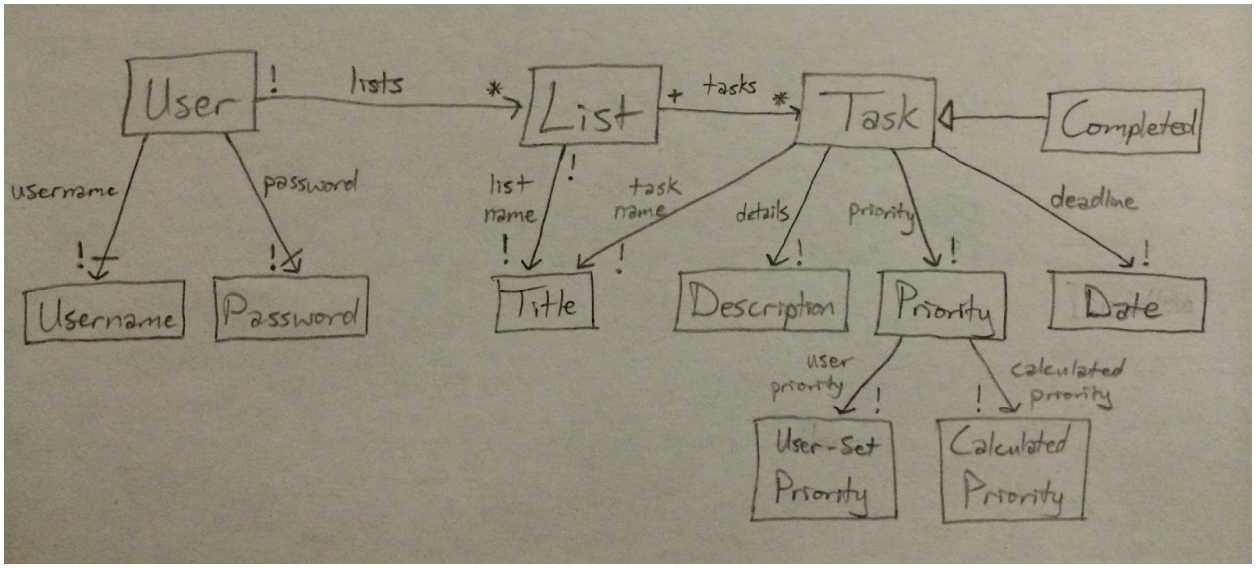


Design Model:

Concepts:

- **Task** - A task that the user needs to do at some point. Each task may have a set deadline and/or a description, but is only required to have a title.
- **User** - A user of the system. Each user has their own tasks associated with them, which they control completely.
- **List** - A collection of related tasks, defined by the user. Used for organizing the way tasks are viewed.
- **Deadline** - The day by the end of which a given task needs to be completed. Not necessary to have for every task.
- **Priority** - The importance level of a task, either set by the user or determined by the system. Used for organizing the way tasks are viewed and emphasizing the most important ones.
- **Shirking** - The act of postponing a task's deadline until later, to allow the user to procrastinate.
- **Completion** - Whether or not a given task has been completed yet. Tasks are considered uncompleted upon creation, and then the user may mark them as completed once they have done them. This will be used to determine how/where tasks are displayed.

Data Model:



Design Challenges:

Design Challenges the Project will Face:

- **How far can shirking go?** Users will have the ability to postpone tasks with due dates. However, we will have to decide whether this is something that can be done infinitely, or whether it is limited. There is something to be said for letting the user do whatever they want, but there is also a risk of the site becoming ineffective or annoying if all the user does is shirk everything repeatedly. In this case, it seems more reasonable to let the user manage their tasks themselves, and just make sure that the site reminds them of their shirked tasks while not being overly obnoxious.
- **How should tasks be prioritized?** The user should definitely be able to set task priorities to mark which ones are and are not important to them. However, it is also good for the system to be intelligent, and know for example that tasks with deadlines of today or tomorrow are important as well, so that the user does not have to waste effort figuring out what they need to do soon and giving that a high priority, for example. The user should always be able to cut the system out, though, and see just what they think is important; to not allow this could cause substantial user frustration. Therefore both systems of prioritization should exist, but it should be possible to separate them.
- **How are tasks without deadlines handled?** It certainly makes sense to have the concept of a task without a definite deadline, because some real tasks do not have set timelines for completion. However, these tasks do not fit in well with ideas like automatically generating a list of tasks for today, for example, because it is unclear whether any of the tasks without deadlines should appear in the list. One option is to show these tasks in a completely separate list of their own, which would help prevent their being forgotten.