6.170 Software Studio
Fall 2014

TeamMotivate: Problem Analysis

Motivation

TeamMotivate is an online app with the intent of providing a user friendly, and simple interface for members of a large-scale project to monitor and track progress of the project as a whole. It allows for members to create new projects and add or assign them to other members. It also lets the user keep track of and prioritize their individual tasks

Purpose

- **Provide a means for different departments of a company to keep track of progress of a project**. In the case of large scale projects, it is essential to be able communicate between departments to efficiently gauge and monitor the progress across disciplines.
- **Help workers to keep track of pending tasks and to prioritize these tasks**. By having an interface that allows users to keep track of what tasks they have to complete within different projects, TeamMotivate seeks to clarify each user's responsibilities and help users prioritize their work.
- **Allow members to see how their work fits in to the rest of the project**. The app helps users understand what part of the project they are contributing to and how the completion of their part allows other parts of the project to progress.

Context Diagram

The scope of this project would be a large company setting, with multiple departments that need to communicate effectively to accomplish a task.

Concepts

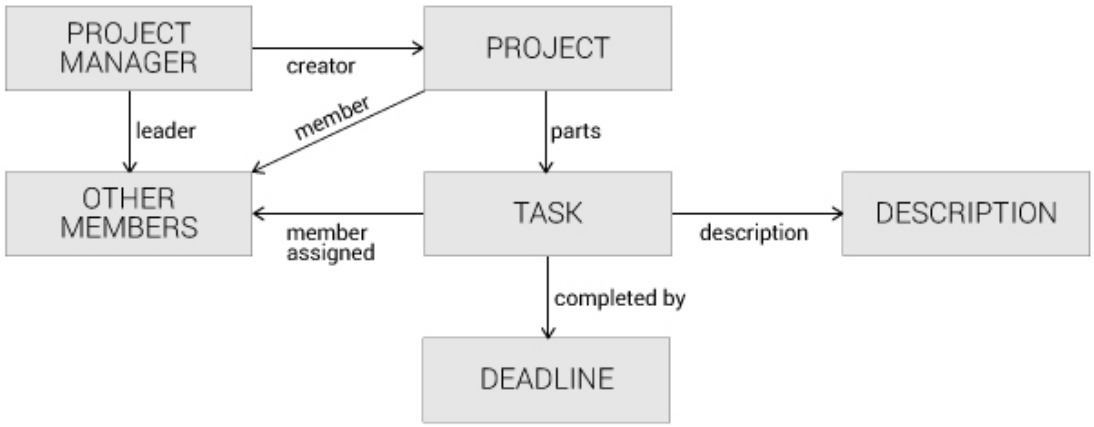
Project: This is the key concept our application is based on. Our application runs on the existence and completion of large scale projects. A project keeps the tasks organized and allows users to see how their work fits in with everything else.

Tasks: Each project can be split up in to multiple tasks that may be assigned and kept track of individually. Tasks help members stay organized.

Deadline: A deadline may simply be thought as an attribute of a project or task. However, in the case of large scale projects, deadline is a pivotal concept that determines the success of a project. The idea of a strict deadline is utilized: if a project is not completed by this deadline, repercussion will be enforced. Deadlines help users keep organized by knowing which projects need to be done first.

Progress: The progress is a value that ranges from 0 to 100%, showing the progress on an action item. This can be set by team members, and when taken together with an action item's estimated time to completion, can be used to calculate the total progress on the entire project. This fulfills the purpose of tracking project progress.

Data Model



Design Challenges

Creation and priority assignment of tasks

How are projects and tasks created? Similarly, how is priority assigned? The project manager should keep track of all projects created under his main project. However, it is useful for members to create projects that suit their organizational needs in implementation.

Potential solutions:

1. *Only project manager can create projects or tasks and assign priority*: power is solely in the hands of the project manager.
2. *All users can create projects or tasks and assign priority*: project manager oversees the whole project, so it is important for them to have the final say on the choice of projects and tasks to be completed.
3. *Other members can propose new projects or tasks and priority assignments for the project manager to approve. Project manager can create projects and assign priority*: encourages other members to actively participate in the overall process of completing the project, while giving the ultimate control to the project manager.

Chosen solution: 1. Only project managers can create projects and tasks so that the project manager is aware of all upcoming todos. This is also the simplest approach.

Dropping out of a project

If a member drops out of a project, he is removed from the database, and steps are taken to ensure that the tasks do not disappear with him. The tasks must be reassigned to another member without breaking the application.

Potential Solutions:

1. One way to prevent this is when a member decides to drop out, the project manager reassigns the tasks to people with proper qualifications and are least busy before actually removing the member from the database.
2. Another less feasible way is to prevent a member from dropping out before completing all his assigned tasks. Doing this will conveniently result in not needing to change anything within our application, but there are probably legal reasons why this cannot be implemented.

Chosen solution: 1. We allowed the leader the authority to remove a member from a projects. When a user leaves, it makes sense to allow the leader to redistribute tasks and remove that member individually.

Completion of tasks and projects

After tasks and projects are completed, should they be deleted right away or archived?

Potential solutions:

1. *Delete completed tasks and projects*: keeps the pending tasks and projects free of clutter, but at the same time it becomes hard for users to determine which tasks are done and which have yet to be added.
2. *Remove tasks and projects from the hierarchy and store them in an archived section*: keeps the pending tasks and projects free of clutter, but users would have to check the archived section to verify the completion of a project.
3. *Update their status to be completed but keep them in the hierarchy*: users can continue to keep track of what has been done for a project, but pending tasks and projects list will be harder to navigate through.

Chosen solution: 1. Deleted tasks and projects are removed from the database and not archived. This is the simplest approach and also keeps the database clean.

Owner of Tasks in Data Model

One design challenge was to determine what should own tasks in the database.

Potential solutions:

1. *Have tasks owned directly by projects, and have the tasks store which users are assigned to them.* This approach places the project at the highest level of importance.
2. *Let tasks be its own top-level collection, and to have everything that uses tasks store a list of task id's.* This approach places tasks at the highest level of importance.

Chosen solution: 2, because many different things depend on tasks, so giving tasks its own collection provides the best protection against performance bottlenecks and explicitly states its importance.

Progress Tracking

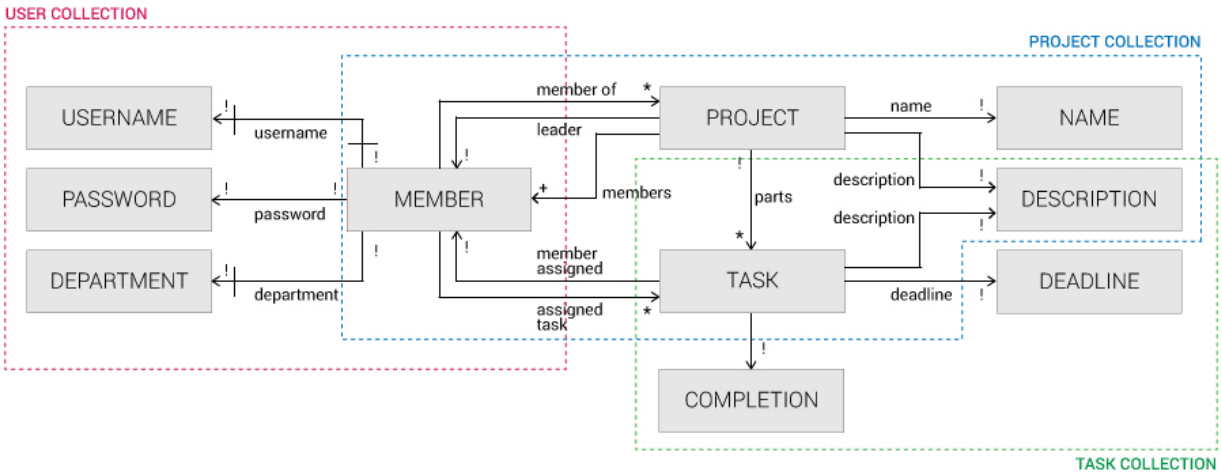
How should the tracking of progress be done? Many different ways were considered for project progress to be tracked.

Potential solutions:

1. *Qualitative progress tracking by allowing team members to write short progress reports.* This is the approach that allows for the most detail, but is also the most complex feature.
2. *Allow team members to directly edit the estimated time to completion*. One disadvantage of this approach is that it causes the original size of the task to be lost.
3. *Allow team members to exercise their own judgment and assign a percentage completion rate to tasks.* This is a small additional feature that relies on a user's intuition in order to be useful.

Chosen solution: 3. This quantitative approach allows the overall completion to be calculated by aggregating the progress for individual tasks. At the same time, this choice allows for a vivid depiction of task progress.

Data Design



One subtle design element is that a department attribute was included in a member. This is a key focus for the problem that is addressed, because the problem that large scale projects often have is the lack of communication between departments. The department attribute is set here as a reminder of our purpose.

A project should have a deadline, but since project is comprised of tasks, and each task has a deadline; thus, it is redundant to attribute a deadline to the project too. A task's deadline should never be later than a project's deadline.

A task has a completion attribute that keeps track of the percentage completion of the task. This is updated by the assignee's judgement to how complete the task is.