SYSTEMS FINAL PROJECT: GRAPE™

Nala Sharadjaya & Xinhui Xu, Period 10

Abstract

Are you tired of scrolling through (99+) messages on your Facebook group chat to figure out what the heck is going on with this CS project while you were cramming for that Calc test? Do you wish to get a quick, holistic view on your project's current status without going through the hassle of asking questions every time you log on? And, better yet, get it all done from the command line? Grape™ is a networked command line application aimed to help project managers and team workers to manage projects without clutter.

User interface

When first launched through terminal, the app will display a welcome message and the user will enter their username. There are the input/command options after login: *New project [0],* and *My projects [1].* The app will then prompt for user command. The number inside the bracket is the input to trigger the command; we chose to use a single digit to facilitate the speed of the process.

New project: user creates new project. The user will be prompted to enter the project name. The username of the creator is automatically saved.

My projects: project names the user participates in will be displayed in a numbered list. The user can again input a number to access the specified project. After input, the project displays its name, member names, and command options: *All tasks [0]* and *My tasks [1]*.

If user is the manager, there will be extra options: *Edit members [2]*, which displays full member info (name, ip), then allowing manager to *Add member [0]* (prompt name, ip) or *Remove member [1]* (prompt name to be removed; no number command here to minimize human errors). The second extra option, *New task [3]* which prompts the manager to enter task delegate, description, due date, and status.

All tasks [0] displays numbered list of tasks delegated to each member by the project manager. *My tasks [1]* will only display tasks assigned to current user. Members can then update the *status* of one of their tasks by entering its number. They can choose to mark it *Not yet started [0], In progress [1],* and *Complete [2]*.

Note that after each operation, the user will be prompted after the appropriate set of command options is displayed.

Technical Design

(Root dir is assumed to be project root dir for file paths.)

We will be using a server-client schema, with one [hosted] server running and many clients exchanging information with the server over network. All data will be stored in the server and accessed remotely from each of the clients.

Booleans will be used to determine which part of the app the user is currently on.

We will read from stdin and parse user command with the string library [or the readline library]. When a new project is created, the app creates a new remote directory projects/ on the server machine. Inside the project directory will be a file (projects/<new_proj>/members) for storing the list of users who have member access to this project. These users will be identified by usernames and later, possibly, passwords as well. There will be another csv file (.../tasks) with name, task description, due date, and status. We will parse the csv files into various type of arrays for server-side usage.¡

We will be using semaphores to ensure the completion of each operation.

Project breakdown and task delegation

Nala: Handling client input, semaphore implementation, file output

Xinhui: Server structure and functions, setting up network

Timeline (subject to change)

- CSV/string parsing functions: Saturday January 7
- Set up server scheme with basic functionality: Monday January 9
- Have project managers be able to create project and add members with access: Wednesday January 11
- Have project managers be able to create and assign tasks to users: Thursday January 12
- Have users be able to update tasks status (only ones they've been assigned to): Saturday January 14