q Final Report - #teamname

Team 15: Ryan Cooper, Michael Gohde, Kyle Helmick, Cooper Kernan, Nicholas Williams
Website: 10.203.54.50 (only accessible on campus wifi)
Github and Zenhub

Project Reflection:

Successful Tools and Methodologies:

Agile: Using an agile methodology seemed to be more effective for our project than what we might achieve through a waterfall approach. Initially, we had many different ideas about what features and designs should be included in our app. Through our agile meetings, we were able to adapt our focus to the most important ideas based on the current state of the project, with everyone working individually to make the collaborative work done faster.

Bootstrap: We bootstrapped our to-do list and an index.html template. This helped us accomplish the our objective faster and have to work less on aesthetics and the overall visual appeal.

Apache/MySql/Raspberry Pi/Language Choice: We chose a fairly standard Linux, Apache, and MySQL setup because of its familiarity, durability, and wealth of documentation. This proved to be a good choice, as it allowed us to deploy our framework on our Raspberry Pi webserver and numerous development machines with a minimum of effort. For backend development, everything was originally developed in python to keep the pace of implementation as high as possible. After most of the API was written, C++ was used to improve the performance of various critical functionality.

Slack: Slack proved to be an extremely effective tool for general communication within the team. It was helpful in coordinating meetings, planning what parts of the project we were going to work on, as well as answering low level technical questions. Slack's integration with Github and Google Docs proved to be a helpful feature, as it was easy to find the current milestone documents we were working on and to stay up to date with other members' git commits.

Google Calendar: Google's calendar feature was useful to our project in assisting with the scheduling of meeting times that worked for all of our hectic schedules. By all members posting their weekly availability for the semester, we could look at the calendar to see potential meeting times. A potential weakness of this calendar was that we used a static schedule for each week instead of updating it dynamically. Consistent updating of the calendar would make even less communication necessary when scheduling meetings at the cost of more time spent updating the calendar.

Zenhub: Zenhub proved to be a generally positive agile management tool. It served all our needed purposes of adding tasks, assigning them, and marking them completed after finishing. It was convenient to have both our agile management tool, and our version control tool, Github, on the same website.

Unsuccessful Tools and Methodologies:

Angular: Because of a lack of familiarity with web development, Angular was not the right choice of framework for our project. We began with this initially as it was suggested by past students of the class, but there weren't any members on the team who had a deep understanding of the framework, who could assist other members and help drive the project forward.

Moodle: Moodle was not an overly effective tool for coordinating our project milestones. Some of the milestones were not posted as assignments, but rather only contained in the milestones document, making it less obvious what dates and information was needed for each assignment. Further, there was inconsistency in the fact that some milestones needed to be submitted to Moodle, while others just needed to be pushed to Github.

In-person-meeting: We experienced some difficulty in scheduling our agile and general work meetings. There were a couple meetings where members cancelled at the last minute, resulting in the team not meeting as planned. There were also a couple meetings where members didn't show up. While these difficulties did provide short term setbacks in product development, they did not prove to be frequent enough problems that they had large effects on our final product.

Project Report:

Accomplishments:

We made a web app! As a team this was honestly one of our biggest challenges as we were fresh faced basically kids that had barely touched anything to do with web servers, HTML, CSS, Javascript, MySQL and on and on. However, because this entire project was in itself a miracle that just happened, we here at q consider every aspect of our web app as an accomplishment, from the ability to make user accounts and have running session ids / user information stored in self deleting cookies, to just making the thing look good (thanks bootstrap). Some of the more noteworthy though are dynamic web pages depending on login status, the cookies that handle log in credentials and a flowy easy to use interface.

Current Issues:

Cosmetic Issues:

- Listing of todo items on a user's Queue page can sometimes collide with the div that contains the "Your q" title and "New…" button. This is dependant on the user's resolution and amount of todo items.
- The iPhone on the frontpage couldn't be resized to not overlap the text underneath it without manually changing the picture in photo editing software.
- The form fields for adding a todo item are not using placeholder texts, and "placeholder" text must be removed manually.
- Navigation buttons on drop down bar are not centered vertically.

Front End Issues:

- Not all backend fields are used in each message.
- More sorting and customization features are necessary to make full use of the application.
- Should be encrypted.
- Should be modified along with the backend to better support user account security.

Back End Issues:

- The C++ interface needs to be more thoroughly bugfixed. Currently, there is a large memory leak that may have something to do with the MySQL client library. This should not affect program operation (the operating system cleans up program memory on exit), but could potentially provide an attack vector for security.

- API functions should be added to combine commonly repeated actions into more efficient units.
- Should make use of data collection already being done for security purposes.

Future of q:

We have given thought to expansions (outside of what was not able to fit within the time frame of the project) on the project however as sort of a “future of q.” We had been tossing around the idea of making it modular in a way that if you were a teacher, you could download a package of sorts that helped to assign assignments and work, with special attributes for the content. Things like mass assigning of todos to other account or accounts in user groups, changing what the contents are of the todo (switching over to mongodb from mysql so our database isn’t static when containing people’s todos), while on the opposite spectrum a user might be able to add friends or import friends from popular social networking sites to organize an event (moving the app from just things to do, to things you can do). Or if you were say in a class about the development, methods, and tools of software you could mimic the functionality of trello or zenhub, but include better linking to git and slack (through more packages) and have a more robust todo list for a project. Modularity is what is today’s market and we hope to capitalize on specifically that.