Project TABL

Devin Johnson – Jacob Stilwell – Vincent Waters – Mason Wray
CS 309 – Fall 2017 – Team AMC2

Project Description

Project TABL is a project management and workforce optimization tool. It is intended to manage and track the completion of project of any size, across multiple platforms.

Key Features:
- Multi-platform support, as well as a web client
- Integrated team messaging functionality
- Intuitive tree view for task management

Users:
- Administrators
- Managers
- Users

Module Interfaces

data.php – handles all interactions with the SQL database from the PHP server. If a different database is used, or the schema is changed, only this file needs to be modified to return perform the corrected queries. All persistent data is extracted from this class with function calls.

frame.php – handles all functions necessary for sending the main application window to the web client. Significant changes to CSS, or HTML layouts can be made to the template used by this class without changing the functionality of the application, since this class determines the correct location for the content within the page, regardless of style or layout.

update.js – this is a javascript function that gets content from the server, and places it into an HTML element on the web client DOM. Similar functions exist pre-packaged in JQuery and other frameworks, but to increase decoupling, we encapsulated those inside our own wrapper function.

Teamwork

What Went Right

Initially, figuring out the correct style for building an xAMP application was a struggle, as none of us had any considerable experience with it. Later, AJAX became an issue, as we had not been able to configure it correctly in the beginning, and implementing it later proved to be much more of an undertaking than we had initially expected. Overall, the biggest challenge proved to be unforeseen requirements that we were not initially aware of, such as the requirements of modularity and decoupling that were revealed at late stages in the semester.

What Went Wrong

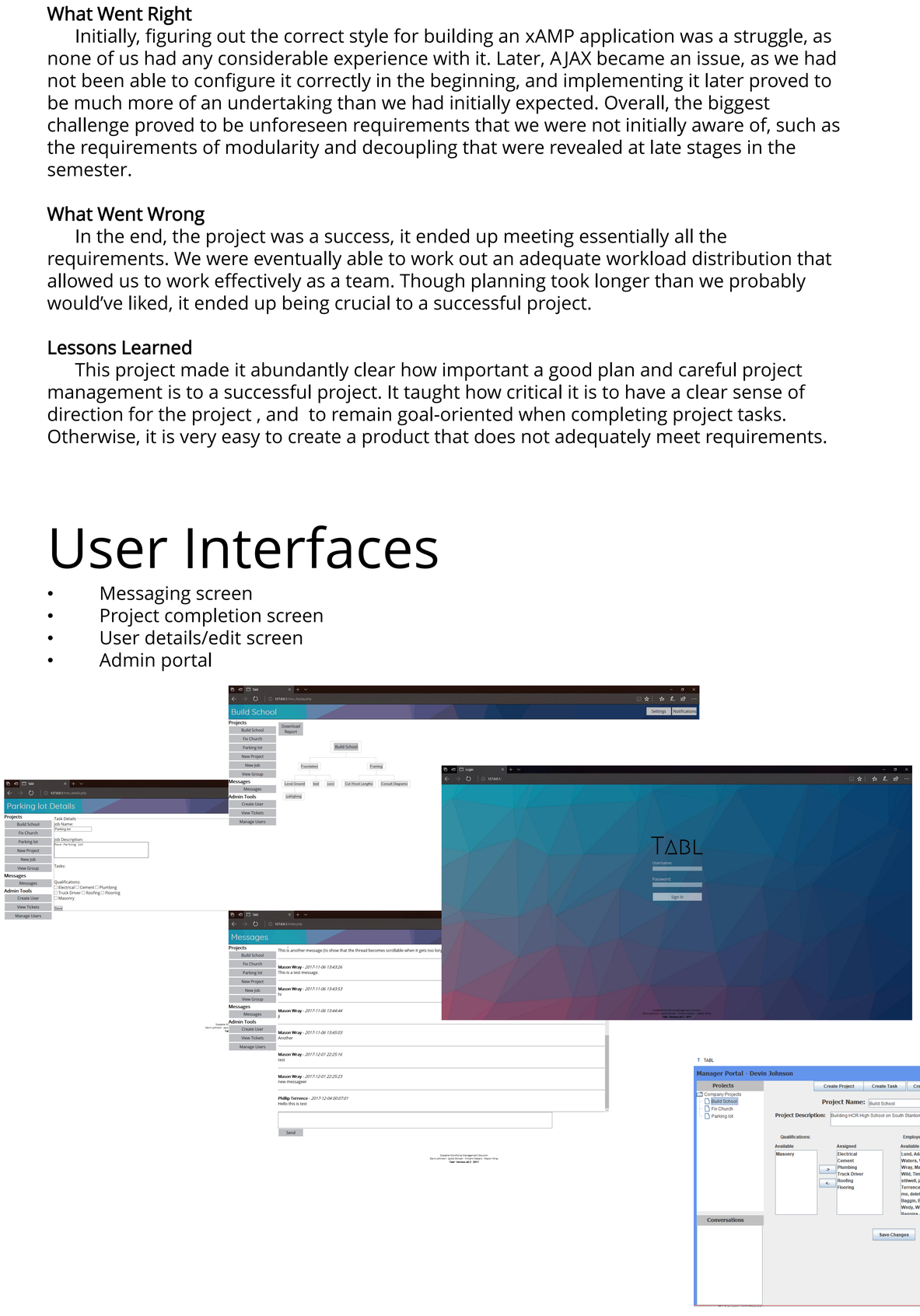
In the end, the project was a success, it ended up meeting essentially all the requirements. We were eventually able to work out an adequate workload distribution that allowed us to work effectively as a team. Though planning took longer than we probably would've liked, it ended up being crucial to a successful project.

Lessons Learned

This project made it abundantly clear how important a good plan and careful project management is to a successful project. It taught how critical it is to have a clear sense of direction for the project , and to remain goal-oriented when completing project tasks. Otherwise, it is very easy to create a product that does not adequately meet requirements.

User Interfaces

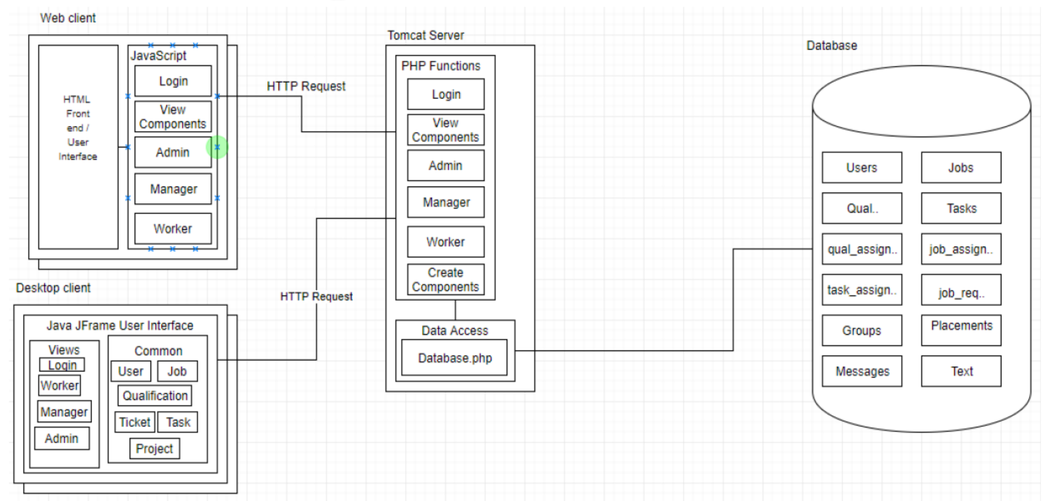
- Messaging screen
- Project completion screen
- User details/edit screen
- Admin portal



Design Decisions

- Decided to store tree structure with database records, rather than storing a custom data structure in a database object
- Decided to use Java Swing for desktop GUI rather than WPF to increase portability
- Single-threaded
- Database is schema is designed to support an arbitrarily large number of separate organizations using TABL independently (Creating a separate database for each deployment is not necessary)

Block Diagram



Actors

Administrator:

Create manager/worker accounts: This will allow the administrator to create accounts for each of their employees with their information.

Assign permissions: Assigning permissions controls which managers can create new jobs, assign employees to which labor pools. Which managers can assign which managers. Also assign tags to workers to control which jobs/tasks they can pick up.

Manager:

Create project: Will create a new project that will consist of jobs and tasks.

Create job: These jobs will be smaller parts of the project. For example if the project is to construct a house, a job could be running electrical, or plumbing. The manager would then assign people who have electrical experience to the electrical job.

Create Tasks: These would be the smallest parts of the project (leaf nodes in the project structure). Continuing the example from above, a task of the electrical job might be 'install light switches'.

Assign workers: Workers will get assigned to proper jobs that they can perform.

Mark a job complete (close a job): Managers can close a job to signify its completion.

Approve or Deny Tasks: The manager can approve tasks that the workers have proposed for the job.

Verify task/job completion: After a lower tier manager or worker has completed a task, a manager must approve the completion.

Worker:

Take Task: Will see a listing of tasks that have been posted and are unassigned to a specific worker, but that the worker meets the tag requirements for.

Propose Tasks: Worker proposes tasks for a job they are assigned, there will be some that might already exist, but workers can propose additional ones so that what they are doing can be tracked. Worker proposed tasks must be first approved by a manager. After which the task is actually created

Finish task: Workers can finish tasks, but not jobs. Once they have completed a task they can mark it as done. This will dim and cross it out, as well as mark it to be processed by the manager.