CIS 422 Project 1: Classroom Cold-Call Assist Software Project Plan

Bethany Van Meter (bvm), Mikayla Campbell (mc), Joseph Goh (jg), Olivia Pannell (op), and Ben Verney (bv)

January 16, 2020 – v1.0

Table of Contents

1. Project Plan Revision History

Date	**Author**	**Description**
1-14-2020	jg	Created the initial document and wrote previously discussed and recorded elements of the project plan (such as the management, schedule, and monitoring sections).
1-16-2020	jg	Added Monitoring and Reporting section as well as Work Breakdown Schedule section.
1-17-2020	jg	Added Build Plan < ***first working draft***

2. Management Plan

2.1. Organization and Roles

Each group member is assigned the following role which includes the outlined responsibilities:

- Writing lead: Bethany Van Meter
    - The writing lead will verify the quality and completion of the writing in project documents such as the SRS and SDS.
    - The writing lead will assign writing related work to other members while monitoring progress.
- Record keeping: Joseph Goh
    - The record keeper will ensure that task assignment and completion is being recorded in a complete and timely fashion.
    - The record keeper will summarize any decisions made or issues brought up during group meetings.
    - The record keeper will make sure that any revisions to records or documents are properly noted and archived.
- Design lead: Mikayla Campbell
    - The design lead will log design issues and changes that need to be made and present them to other members for discussion.
    - The design lead will be referred to when any minor design decisions that are yet to be discussed need to be made.
- Code lead: Ben Verney
    - The code lead will monitor progress and verify completion of assigned implementation tasks.
    - The code lead will, upon completion of a task, assign members to begin a new task or assist in ongoing implementation tasks.
    - The code lead will routinely check the quality of the code such as performance, style, maintainability etc.
- Test lead: Olivia Pannell
    - The test lead will check throughout the development process that the currently implemented components are being robustly tested by group members.
    - The test lead will make sure that any issues or bugs that have been raised are corrected in a timely fashion and assign debugging tasks.

While each member has an assigned role, the group could be described as having a flat organizational structure. Most decisions will subject to prior discussion and every member is expected to assist in all the above responsibilities if need be. The type and amount of responsibilities for each member will be fluid to accommodate any issues or imbalances in workload. Every member will participate in routine meetings and communicate with each other outside of meetings to make decisions, be assigned tasks, and track the progress as the project goes on.

2.2. Meetings and Communication

Group members will attend regular meetings at the following times:

- Mondays @ 11:30 am
- Thursdays @ 4:00 pm
- *Additional meetings will be scheduled as needed*

Meetings will usually take place in the Price Science Commons (the science library) and last around an hour or longer as time permits for individual members.

Group members will have discussions and report their progress outside of meetings via the following communication methods:

- Slack
- Group MMS messaging

Members are expected to read such messages to stay up to date on any updates or issues that might arise during development.

3. Work Breakdown Schedule

- Week 1 (second week of the term)
    - Create project plan
    - Create working drafts of the SRS and SDS
    - (Initial project documents due on Friday)
- Week 2 (third week of the term)
    - Meet with client (Prof. Hornof) for discussion of and adjustment of software architecture and requirements/design
    - Begin implementation and assign implementation tasks to all members
    - Create working implementations of non-GUI components
    - Create implementations of the visual and user input parts of GUI components
    - Bring code base (including GUI) to buildable state by end of week
- Week 3 (fourth week of the term)
    - Finish implementing 'must have' requirements for all components
    - Begin testing and debugging phase
    - If on or ahead of schedule, consider implementation of 'nice to have' requirements (such as name learning system)
    - Have a stable release candidate by end of week
- Week 4 (fifth week of the term)
    - (Project due on Monday)

The responsibility for monitoring progress of each milestone will be in accordance with the roles outlined under the **Management Plan** section. Members will likely have roughly equal distribution of implementation tasks.

4. Monitoring and Reporting

Small milestones and tasks whose completion can be objectively verified will be assigned and recorded on a shared spreadsheet. The spreadsheet will be frequently updated, and users will self-report completion of each task. Each task marked as completed must be verified by another group member. This should be done by the relevant lead member, the record keeper, or whomever is available at the time. The spreadsheet will be regularly archived (though no more frequently than once a day).

5. Build Plan

5.1. Plan Details

As further outlined in the SDS, the program will consist of eight files grouped into six modules. Those modules are: the main view, roster, instructor controls, email view, queue interface, and flashcard view.

The first modules to be implemented are the non-GUI ones. Specifically, the roster and queue interface's functionality for reading, manipulating, displaying, and outputting student data according to the system requirements will be implemented.

In tandem, the visual and user input control elements of the GUI components will be implemented. Implementation will begin with the main view and trickle down to the other view components as well as the instructor controls module.

It should be noted that though the implementation is categorized into two separate groups, both groups should have close communication throughout and make sure that functionality and requirements that cross different modules are planned for and not compromised.

Once each component can execute its individual functionalities and have been tested for appropriate requirements, the modules will be connected to reach a buildable state. Afterward, remaining 'must-have' requirements will be implemented and the viability of implementing remaining 'should-have' or 'nice-to-have' requirements will be evaluated.

Unless optional requirements and functionality are being added, all resources should be focusing on testing and debugging to reach a stable release candidate by the project deadline.

For a temporal breakdown of these phases, refer to the work breakdown schedule.

5.2. Rationale

By breaking down the system into these parts, the functionality of the program can be consolidated by categories such as file I/O, student data manipulation, GUI, etc. Furthermore, the modularity allows integration of later-developed functionality such as a flashcard view mode to be relatively simple.

Development of the non-GUI and GUI components have been separated to allow the team to work to their expertise and be able to focus on similar work through the initial implementation phase.

We do expect team members to have more difficulty in creating and integrating the GUI elements. We will have to assess the difficulty and efficiency of implementing each module early on and reallocate team members accordingly. In addition, even members who are not part of the initial development of GUI components should be required to make themselves familiar with the relevant libraries.

6. Acknowledgements

The formatting of this document was based on a Software Requirement Specification template provided by Professor Anthony Hornof.