Project Plan

Version 3.0.0, last updated 4/29/19 by Zach

Cold-Caller

Nick Bonat, Jerry Xie, Vu Vo, Qi Han, Zach Domke

Previous Revisions:

Author	Date	Summary
Qi, Vu, Zach	4/10/19	Initial creation of Project Plan, SRS, and SDS.
Vu, Zach	4/16/19	Update documentation with up to date diagrams and descriptions. Include citations as needed.
Vu, Zach	4/23/19	Review documentation and minor changes before turning in.
All	4/28/19	Restructure Project Plan and add more information.
Zach	4/29/19	Project Plan Finalized

Purpose and Audience:

Intended Audience: The Cold-Caller will be used by teachers in classrooms (either Middle School, High School, or College). It is expected that the students will be able to see the application running on a projector at the front of the class.

Purpose: The Cold-Caller will provide the classroom setting with an even chance of being called. We all know that some students prefer to speak up while some will try to stay quiet. Cold-Caller will provide the students and teacher with a list of who is expected to speak up next. This allows the teacher to gain knowledge of who understands the material and who needs more help.

Motivation:

The system is partially motivated by the belief that many students have answers or discussions that they want to contribute but are anxious in a social environment. The article “Why More and More Students Won’t Speak Up in Class”, published by Minding The Campus (https://www.mindingthecampus.org/2019/01/08/why-more-students-wont-speak-up-in-class/) points out a few reasons why students have stopped speaking up in classes. The reasons pointed out in this article are that students are scared of being judged by their peers, which can be avoided by staying silent. Cold-Caller will help students who are anxious to speak up by letting them know when they will be called.

The system is also partially motivated by the belief that a teacher calling on his/her students will benefit both the teacher and student. An interesting read on Hack Learning (http://hacklearning.org/coldcalling/) immediately points out some benefits cold calling gives to students, including building confidence and accessing learning. These 2 aspects will develop a student into a hugely beneficial part of society. The fourth point listed at the top (improving conversations) benefits the student called on as well as any other students in the class. It is for these benefits that we have designed Cold Caller to make sure all students are called on evenly.

Team Roles:

Our team consists of 5 members: Nick Bonat, Jeremy Xie, Vu Vo, Qi Han, and Zach Domke. All tasks and deliverables were distributed evenly amongst the group members.

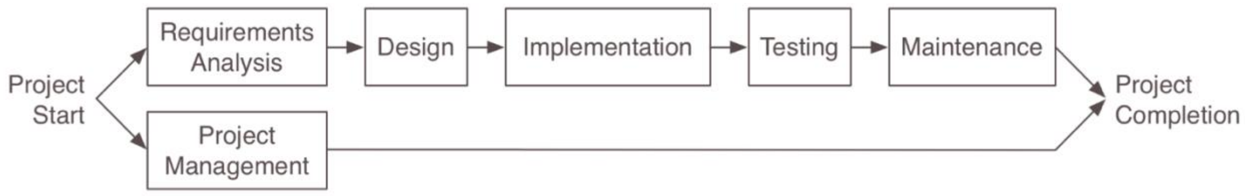
Role	**Responsible**	**Associated Deliverables**
Project Manager	Nick Bonat	Project Plan
Model Developer	Vu Vo, Qi Han	Model modules
Service Developer	Zach Domke	Service modules, SRS
View Developer	Jeremy Xie	View modules
Controller Developer	Nick Bonat	Controller modules
Architect	Jeremy Xie, Vu Vo	Diagrams, SDS

Risk Mitigation:

Risk	**Description**	**Mitigation**
Module Coupling	Modules have too tight of a coupling	Reduce module reliance
Exceeding due date	Our due date was approaching fast	Create a specific plan
Communication	Keeping in touch with each other	Better inform each other through group chat
Intergroup Requirements	We have problems communicating requirements to each other	Better communication about our expectations
User Interaction	Potential problems when importing a new roster.	Add try-excepts to verify usable file formats

Process:

The process we will use for this project is a modified version the waterfall method. This process was taught to us by Anthony Hornof. This consists of constructing a Requirements Analysis, Design, and then Implementation before we start Testing the Implementation and then providing Maintenance on our project. All of this is done while Project Management occurs. This will be our first time implementing this Process and we are excited to try it out.



We decided to use this Process because we have a set amount of time to work on this project, which will enable us to create a better schedule. We also chose this method as we have a great set of example requirements from the customer. This will give us a great guideline for both the Requirements Analysis and the Design.

We have been able to stick to this Waterfall process for the most part, but we did some basic prototyping in the Design phase in order to gain a better understanding of what our design should look like. This worked well for us.

Mechanism and Techniques:

For this project, we are utilizing an incremental development approach for our software management. First, we perform a requirement analysis on the given SRS so that we can identify what components our system is composed of. Then, we utilize the data flow diagram, as well as other UML diagrams, to describe our modular software design. For software implementation, we used object-oriented programming technique. Each object will make up each component that we identify in the Requirements Analysis phase. Next, we test. We will perform unit testing on each component to test each implementation independently before integrating the components. After we test each individual module, we will integrate the modules together and test our program as a whole. Lastly, for maintenance, we will distribute our program to friends who will be able to run our app. We will ask them for input and make changes to our program as needed.

Milestones: (Schedule included as PDF)

- Create a Plan, SRS, and SDS
- Models (Student, Queue) Design
- Views (Main View, Config View) Design
- Services (Importing, Exporting, File Management) Design
- Controllers (Views Controllers, Some helper services) Design
- Present Preliminary Project to Client (Prof. Hornof)
- Models Implementation
- Views Implementation
- Services Implementation
- Controllers Implementation
- Combine/Connect Model, View, Services, and Controller Implementations
- Finalize Documentation
- Present Final Project to Client (Prof. Hornof)
- Meetings At Least Twice a Week (each meeting is a milestone)

Meeting Notes:

1. Date: 4/7
   Attendees: All
   Agenda: Documentation and project prototype

Task and Deliverable	Responsible	Start Date	End Date	Estimated Time
Documentation	NB, ZD, QH, VV	4/7	4/11	2hr
Project prototype	JX	4/7	4/16	2hr

2. Date: 4/9
   Attendees: All
   Agenda: Documentation and project prototype

Task and Deliverable	Responsible	Start Date	End Date	Estimated Time
Documentation	NB, ZD, QH, VV	4/7	4/11	2.5hr
Project prototype	JX	4/7	4/13	2hr

3. Date: 4/14
Attendees: All
Agenda: Documentation, project prototype, IO Service, Log Service

Task and Deliverable	Responsible	Start Date	End Date	Estimated Time
Documentation v2.0	QH, VV	4/10	4/16	2.5hr
Project prototype v2	JX	4/10	4/13	2hr
IO Service Prototype	ZD	4/10	4/14	2hr
Log Service Prototype	NB	4/10	4/15	2hr

4. Date: 4/16
Attendees: All
Agenda: Documentation, project prototype, IO Service, Log Service

Task and Deliverable	Responsible	Start Date	End Date	Estimated Time
Documentation v2.2	QH, VV	4/15	4/16	2.5hr
View Protorype	JX	4/15	4/16	3hr
IO Service Prototype	ZD	4/10	4/14	1hr
Log Service, Student Class	NB	4/10	4/15	2.5hr

5. Date: 4/21
Attendees: All
Agenda: ColdCallerService, IO Service implementation, Log Service implementation, Student Queue, documentation

Task and Deliverable	Responsible	Start Date	End Date	Estimated Time
Documentation v2.3	VV	4/16	4/16	2.5hr
Controller Prototype	JX	4/14	4/21	2hr
IO Service Implementation	ZD	4/16	4/21	3hr
Log Service Implementation	NB	4/16	4/21	3hr
Model Prototype	QH	4/16	4/21	1.5hr

6. Date: 4/23
Attendees: All
Agenda: Documentation and project prototype

Task and Deliverable	**Responsible**	**Start Date**	**End Date**	**Estimated Time**
Documentation v2.3	QH	4/16	4/23	1hr
View Implementation	JX	4/16	4/23	1.5hr
Testing	NB, ZD,	4/21	4/23	4hr
Model Implementation	VV, QH	4/16	4/23	2hr

7. Date: 4/27
Attendees: All
Agenda: Documentation, project testing

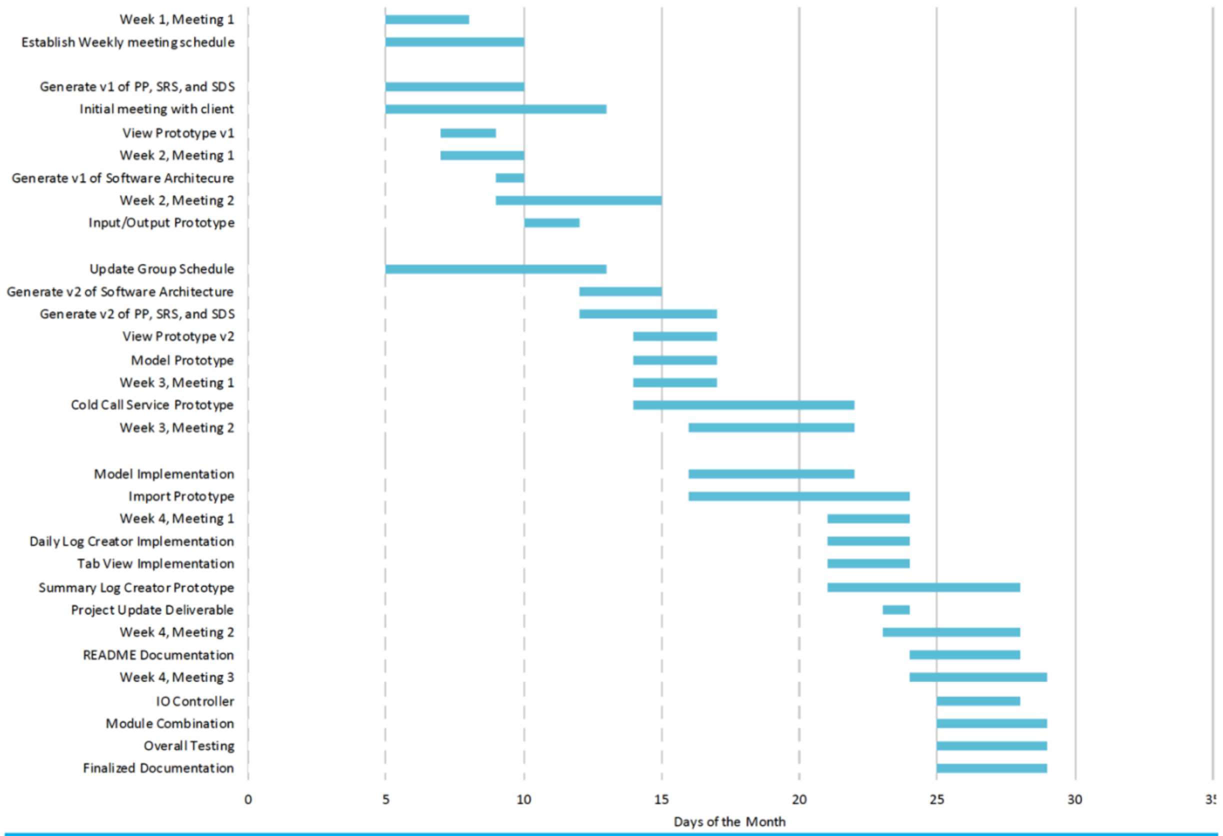
Task and Deliverable	**Responsible**	**Start Date**	**End Date**	**Estimated Time**
Documentation v3	QH, VV	4/23	4/27	3hr
Module Integration	NB, ZD, JX	4/23	4/27	4hr
Testing	NB, ZD, JX	4/21	4/27	2hr

8. Date: 4/28
Attendees: All
Agenda: Documentation, testing and presentation

Task and Deliverable	**Responsible**	**Start Date**	**End Date**	**Estimated Time**
Documentation v3.1	QH, VV	4/27	4/29	3hr
Testing	NB, ZD, JX	4/21	4/29	3hr

9. Date: 4/29
Attendees: All
Agenda: Documentation, testing and presentation

Task and Deliverable	**Responsible**	**Start Date**	**End Date**	**Estimated Time**
Documentation v3.1	NB, ZD, QH, VV, JX	4/27	4/29	3hr
Testing	JX	4/21	4/29	1hr
Presentation	NB	4/28	4/29	2hr



TASK NAME	START DATE	DAY OF MONTH	END DATE	DURATION (WORK DAYS)	DAYS COMPLETE	DAYS REMAINING	TEAM MEMBER	PERCENT COMPLETE
Week 1 Assignments								
Week 1, Meeting 1	4/5	5	4/7	3	3	0	All	100%
Establish Weekly meeting schedule	4/5	5	4/9	5	5	0	All	100%
Week 2 Assignments								
Generate v1 of PP, SRS, and SDS	4/5	5	4/9	5	5	0	All	100%
Initial meeting with client	4/5	5	4/12	8	8	0	All	100%
View Prototype v1	4/7	7	4/8	2	2	0	Jerry	100%
Week 2, Meeting 1	4/7	7	4/9	3	3	0	All	100%
Generate v1 of Software Architecure	4/9	9	4/9	1	1	0	Jerry	100%
Week 2, Meeting 2	4/9	9	4/14	6	6	0	All	100%
Input/Output Prototype	4/10	10	4/11	2	2	0	Zach	100%
Week 3 Assignments								
Update Group Schedule	4/5	5	4/12	8	8	0	Zach, Vu, Jerry	100%
Generate v2 of Software Architecture	4/12	12	4/14	3	3	0	Jerry	100%
Generate v2 of PP, SRS, and SDS	4/12	12	4/16	5	5	0	Zach	100%
View Prototype v2	4/14	14	4/16	3	3	0	Jerry	100%
Model Prototype	4/14	14	4/16	3	3	0	Qi, Vu	100%
Week 3, Meeting 1	4/14	14	4/16	3	3	0	All	100%
Cold Call Service Prototype	4/14	14	4/21	8	8	0	Jerry	100%
Week 3, Meeting 2	4/16	16	4/21	6	6	0	Vu, Jerry, Qi	100%
Week 4 Assignments								
Model Implementation	4/16	16	4/21	6	6	0	Qi, Vu	100%
Import Prototype	4/16	16	4/23	8	8	0	Zach	100%
Week 4, Meeting 1	4/21	21	4/23	3	3	0	All	100%
Daily Log Creator Implementation	4/21	21	4/23	3	3	0	Nick	100%
Tab View Implementation	4/21	21	4/23	3	3	0	Jerry	100%
Summary Log Creator Prototype	4/21	21	4/27	7	7	0	Nick	100%
Project Update Deliverable	4/23	23	4/23	1	1	0	Zach, Vu	100%
Week 4, Meeting 2	4/23	23	4/27	5	5	0	All	100%
README Documentation	4/24	24	4/27	4	4	0	Vu, Jerry, Qi	100%
Week 4, Meeting 3	4/24	24	4/28	5	5	0	All	100%
IO Controller	4/25	25	4/27	3	3	0	Zach	100%
Module Combination	4/25	25	4/28	4	4	0	All	100%
Overall Testing	4/25	25	4/28	4	4	0	Jerry	100%
Finalized Documentation	4/25	25	4/28	4	4	0	All	100%