**CIS 422 Project 1:**

**Classroom Cold-Call Assist Software**

**SDS**

Patrick Thomasma, David Han, Kassandra Morando, Brianna Vago, and Geli Zhang

January 10,2022 – v1.0

**Table of Contents**

**1. SDS Revision History**

**2. Project Description**

2.1 Overall Description

2.2 Roster Module

2.3 Instructor Control Module

2.4 Total System Diagram

**3. Acknowledgements**

**1. SDS Revision History**

**Date Author Description ————————————————————————————————————**

**1-6-2022 km Created the initial document.**

**1-9-2022 km Wrote the Project Plan rough draft.**

**1-9-2022 bv Began proofreading rough draft.**

**1-10-2019 bv Finished editing rough draft.**

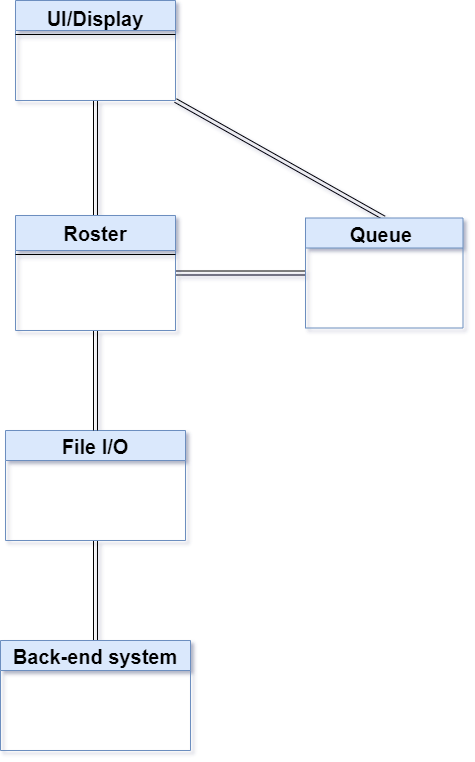
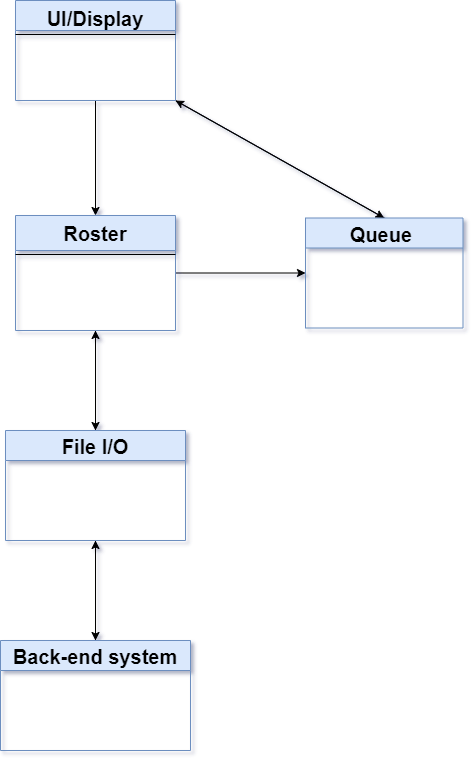
**2. Project Description**

**2.1 Overall Description**

This system will provide an instructor with the ability to “cold call” students in their classes. “Cold calling” is a method in which an instructor or professor will call on a random student in class and will notify the students who are next to be called on. The system will have a list shown on the instructor’s screen of who will be called on next and the following 3 people in line. The list will be shown horizontally at the top of the screen to prevent blocking any content from the students. The instructor will be able to check off the student’s name that has participated and the system will populate a new person’s name in the list. There will also be an ability to flag a student’s name if the professor wants to contact them at a future time. The instructor is able to close the system and reopen it to the same list that was presented before closing the system. The main goal of this system is to equally call on students across multiple days and to actively engage students with the lessons being taught using “cold calling”.

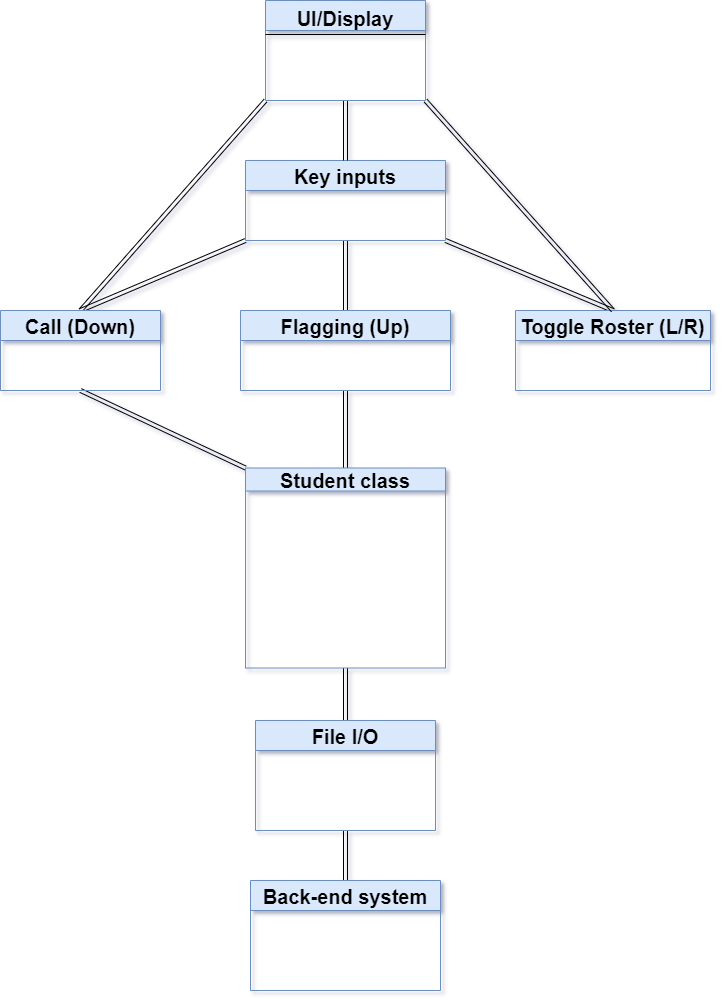
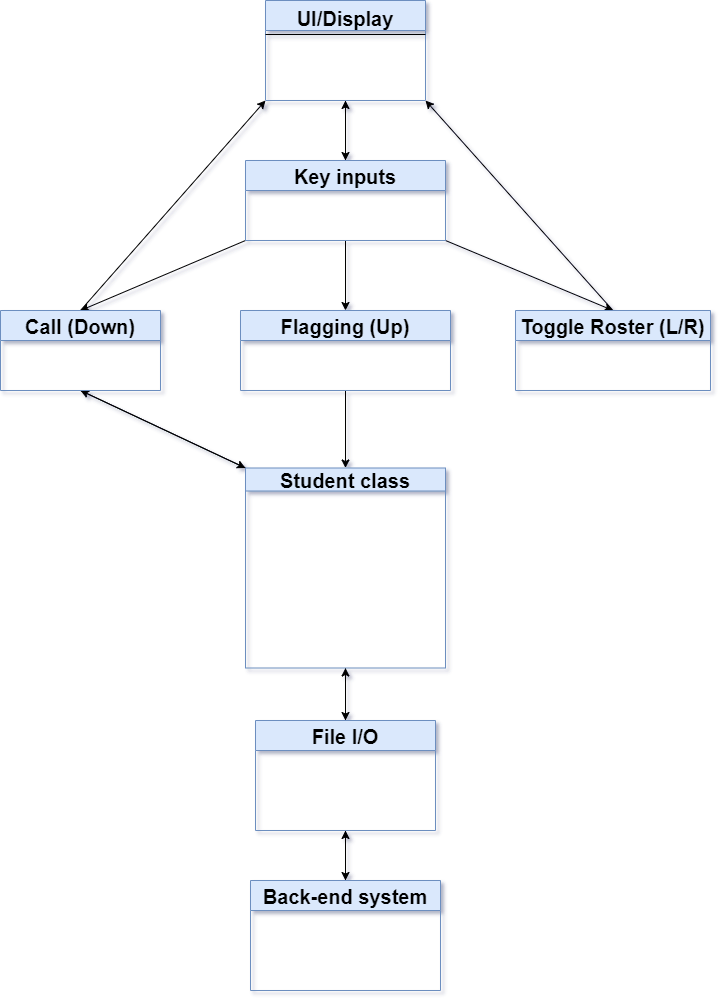
**2.2 Roster Module**

**Static: Dynamic:**



The Roster module will be responsible for dealing with the input and output of the program. In this module we will read and parse the input text files for the student classes, we will then randomly pick from our roster of students and put them into the dock which will be available for the instructor to see and it will contain a list of the students and their ID numbers. Roster module is the main view that the instructor will see from our implementation of the cold call system and will fulfill the motivation of making a system such as this.

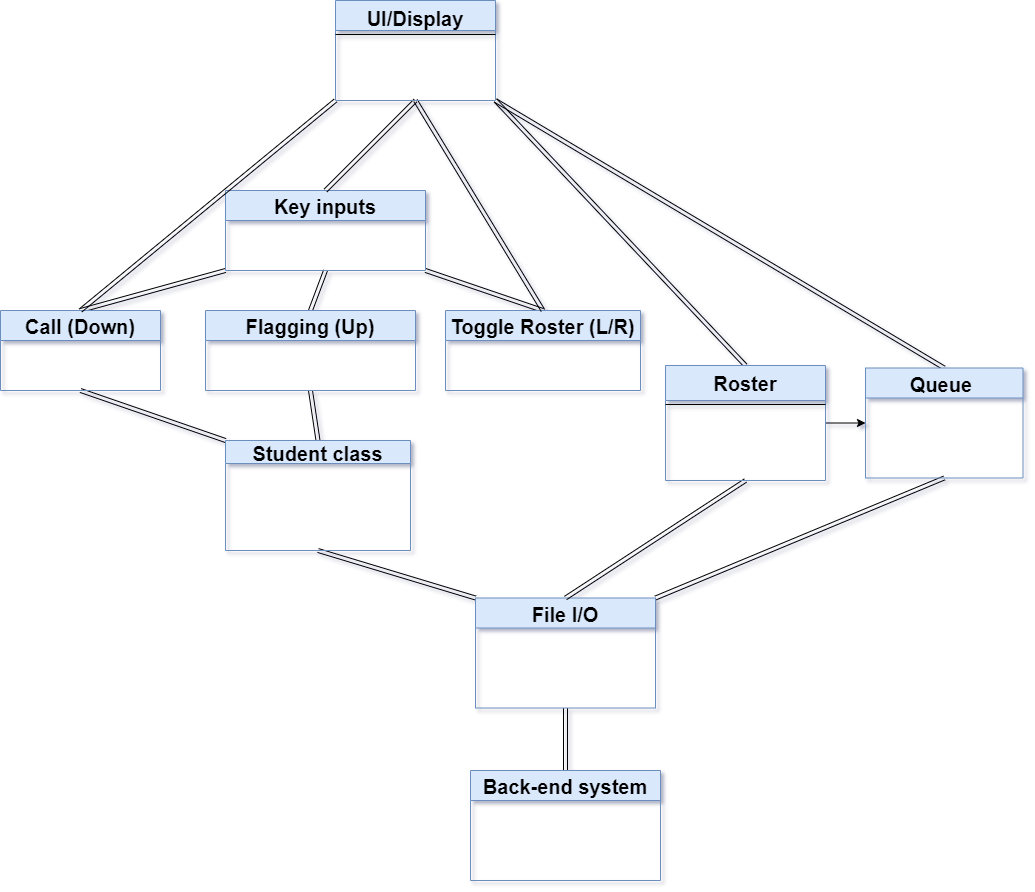
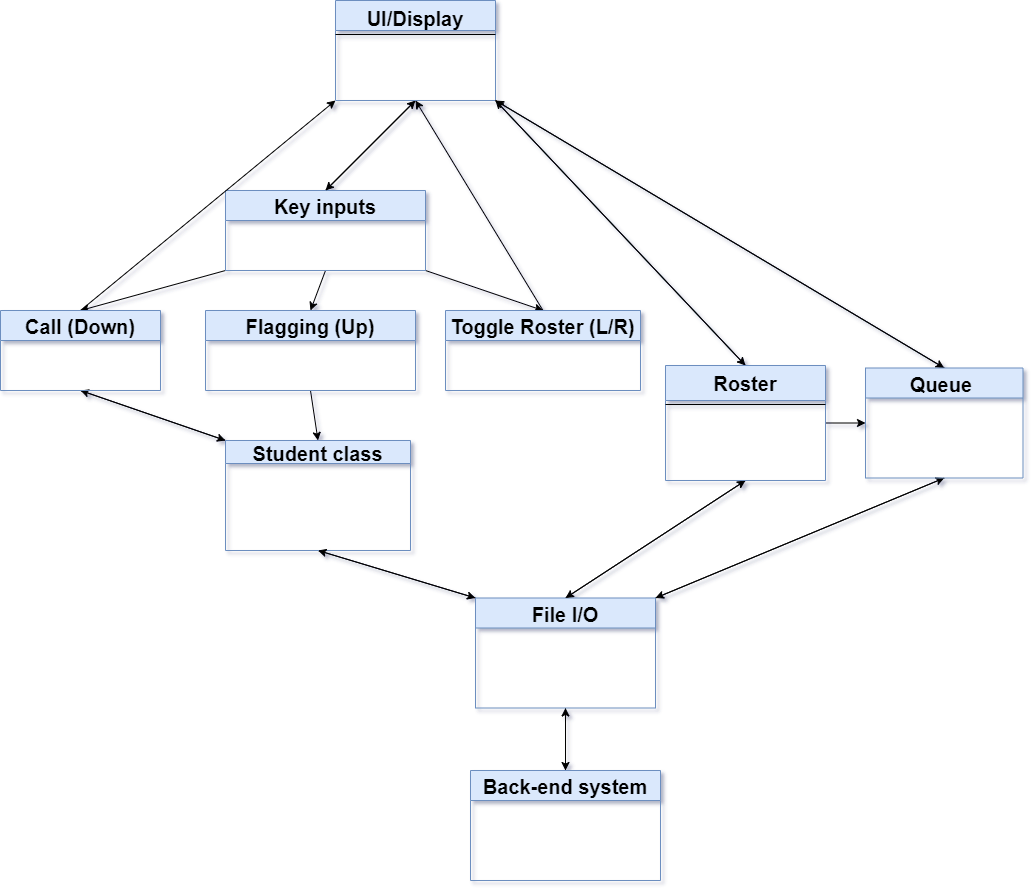
**2.3 Instructor Control Module**

**Static: Dynamic:**

Instructor control Module will be in charge of handling the user controls for the program. This module is responsible for allowing the user to navigate the system, to manage the queue, flag students, and email the flagged students. This module will send signals between the other modules and give the user control of the program. The user will be able to control the Cold call system interface using these components and will be able to do what they want with the program using this interface.

**2.4 Total system diagram**

**Static: Dynamic:**



In total, the system we will want to build will have the Roster module and Instructor Control module interacting with each other in order to build what we want for the cold calling system. We have the back end and UI as the two top components that all the other components will either get their information from or will feed into, the diagrams we provided our ordered in levels representing that the back-end of our system will get the initial program running and initialize the queue and class of students that will feed into the cold call queue. From our UI system in the diagram we show what controls that the user has access to and how that will feed into our Student class so we can update values as we go and increase the randomness of our cold calling system.

**2.4 Design Rationale**

We designed these modules and broke them apart in parts like this to show how each of the systems work separately and how they will also eventually interact with each other when the program is done and all the build components work. The idea with these designs is to show how each component feeds into to each other and where they will get their information from in order to process and compile then feed into the next component, we included arrows in order to illustrate the direction that each component will feed into the next and which exact component is giving information to one and another.

**3. Acknowledgements**

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