Course Name: Software Engineering

Course Number and Section: 14:332:152

VirtualLogicLabs Report #1

Github: https://github.com/SagarPhanda/VirtualLogicLabs

Date Submitted: February 18th, 2018

Group #10:

Sagar Phanda, Khalid Akash, Dhruvik Patel, Vikas Khan, Joe Cella, Yiwen Tao

Individual Contributions Breakdown

All team members contributed equally

Table of Contents

Customer Statement of Requirements **3**

1. Customer Statement of Requirements

1.1. Problem Statement

1.1.1. Problem:

In a world where technology is so prevalent and easily accessible, many students are using it to supplement learning from lectures. Many students would greatly benefit from the ability to observe the workings and functions of a laboratory settings on their own time—however, it is difficult to provide labs for large numbers of students due to the sheer number of students and logistical or financial restrictions of the universities. Both students and instructors would greatly benefit from a resource that would allow for a laboratory experience that would enhance what they learned in class, but would also allow for instructors to track the progress of large numbers of student easily. Specifically, the focus should be on Digital Logic Design Labs. These labs are very technical and require a lot of different parts (power source, protoboard, gates, wires, LEDs, etc.). There are, however, labs that one can use to gain a good comprehension of the material, but do not necessarily require physical interactions with the pieces to gain a full understanding of the working of the circuits. Such labs would be perfect labs to be completed online.

Even in scenarios where students do get experience in an actual lab, professors must manually grade and track each student's results and lab reports, which can be extremely tedious and time consuming. In addition to keeping data for the students, the school would need to hire lab instructors to manually help and serve hundreds of different students which take time, and resources. With hundreds of students in lab classes, returning grades to students in a timely manner is becoming harder and harder. The process for grading is extremely time consuming, with a procedure that includes collecting hard copies of reports, observing students work on their labs, physically grading each report, and lastly uploading the grades. This procedure is also an inconvenience for the students as well because it is hard for them to know how they are doing in the class at critical times, like the last day to drop a class. It is very difficult to grade all the hard copies the students submit, especially when students fail to follow the specified format or make small mistakes that cause discrepancies in the entire lab.

1.1.2. Proposed Solution:

The best way to solve this would be to have virtual sessions where students can simulate labs on their own computers. These sessions can accurately depict the procedures involved in the labs and allow the students to become more familiar with the material. The main goal of this treatment is to facilitate student learning rather than helping the instructors find ways to grade the students. The latter is of lower priority. If students are able to learn at their own pace and time, it will assist with the overall improvement of the course and student learning. Some of the things a software system should include are listed below:

- Labs accessible 24/7:
    - The student will be tasked to do three labs that are typically given in a real digital logic design laboratory setting. The labs will allow them to apply their digital logic knowledge in various situations, such converting binary code to gray code through digital logic devices. The student will be able to select any devices needed for the lab that is available through the laboratory library and mix and match them anyway they like just like they would be able to in the lab. The laboratory will be the implementation of the first three labs in this manual: http://www.ece.rutgers.edu/~marsic/Teaching/DLD/lab-man_2012.pdf

- Receiving Feedback:
    - The most important aspect of any form of learning is feedback of whether the student was successful or unsuccessful in any given task. The virtual lab should check for any wrongdoings upon user request for completion and give the user instant feedback to not only see what they did wrong, but also guide them to online references so that they can learn the specific areas that they need to improve on. In addition to providing references, visual indicators will be given throughout the lab to really show the interaction between each circuit devices.

- Reinforced Learning:
    - Before and after each lab, the student should be forced to take a quiz, similar to what happens currently with reports in class. The pre-quizzes purpose is to insure the student will be able to perform the lab adequately and that he or she has the adequate prerequisite knowledge to perform. Since the purpose of the pre-quiz is to aid preparation, it will not be graded but will be checked for completeness and correctness. After the lab is performed, a post quiz will be given to ask general questions about the lab to attempt to see if the student really learned anything valuable from the lab. This one will be included in the final grade report. The quizzes in general can involve things such as truth tables, karnaugh maps, and logic equations in addition to general questions.

- Instant Grading:
    - After configuring a circuit, based on the lab the student is performing, the operation of the circuit will be checked for correctness. The student will lose points for everytime they attempt to finish a lab with the incorrect configuration. The post quiz is also graded and added to the final grade report (with no opportunity to correct incorrect answers).

- In-Application Communication:
    - The application should host a chat service, akin to that of GroupMe, Messenger, and Piazza. The chat service should be able to connect all the users registered in the application to each other. The purpose of this is to facilitate group learning. It is found

that most of the learning in students is done through social communication for both students that have questions, and students that have answers. In addition to being able to communicate with fellow students, instructors should also be able to participate in this chat. We believe this open communication can allow all the students to learn as much as possible both in and out of the classroom.

- Cheating Verification:
    - Unfortunately, an unintended consequence of having a virtualized lab session through software is the increased likelihood of some students taking advantage of the opportunity to cheat by allowing others to do these labs for them. A solution to this issue is to implement facial recognition software. Each student should be forced to take a picture that will be analyzed by a facial recognition software. The camera of the student's laptop/computer will be forced to stay on during the application which will detect whether the original student is the one doing the lab experiment.

1.2. Glossary of Terms

Boolean	A binary variable, having two possible values, “true” and “false.”.
AND gate	A boolean operator that gives the value one if and only if all the operands are one, and otherwise has a value of zero
Combinational system	A logical system with no memory for which the output depends only on the current input values
Full Adder	The full-adder circuit adds three one-bit binary numbers (C A B) and outputs two one-bit binary numbers, a sum (S) and a carry (C1). The full-adder is usually a component in a cascade of adders, which add 8, 16, 32, etc. binary numbers.
Grey Code	Grey code is an ordering of the binary numeral system such that two successive values differ in only one bit (binary digit).
Karnaugh Map	A diagram consisting of a rectangular array of squares, each representing a different combination of the variables of a Boolean function
LED (light-emitting diode)	A semiconductor device that emits visible light when an electric current passes through it
Logic Diagram	Logic diagrams are diagrams in the field of logic, used for representation and to carry out certain types of reasoning.
Logical sum	a logical term that is the Boolean OR of two or more variables
Logical product	a logical term that is the Boolean AND of two or more variables
Minterm	A Boolean expression resulting in 1 for the output of a single cell, and 0s for all other cells in a Karnaugh map, or truth table. If a minterm has a single 1 and the remaining cells as 0s, it would appear to cover a minimum area of 1s.
NAND gate	A Boolean operator that gives the value zero if and only if all the operands have a value of one, and otherwise has a value of one (equivalent to NOT AND).
OR gate	A Boolean operator that gives the value one if at least one operand (or input) has a value of one, and otherwise has a value of zero.
Protoboard	A board for making an experimental model of an electric circuit
Switch	A device for making and breaking the connection in an electric circuit

Truth table	A tabular list of all possible input combinations for a combinational system and the corresponding outputs
XOR gate	A digital logic gate that gives a logic one output when the number of true inputs is odd, and otherwise has a value of zero
Administrator	An “Actor” of this project that takes the role of viewing grades, create and delete users. Also referred to as “Teacher”, “Professor”, and “Instructor”.
Student	An “Actor” of this project that takes the role of making use of the laboratory simulations that is graded.

2. System Requirements

2.1. Enumerated Functional Requirements

2.1.1. General Requirements

Number	PW	Requirement
REQ1	5	The user shall be able to place any circuit material given onto a protoboard that accurately portrays the correct logic.
REQ2	2	The user shall be able to use a logic probe around any logical circuit to test for high or low potentials.
REQ3	3	The user shall be able to use an oscilloscope that provides a constant 5 voltage source, and a ground that the user can connect with wires.
REQ4	5	The user should be able to place logic switches into the circuit board that can be turned on to let current through, and off to cut off current.
REQ5	4	The system should be able to auto-grade the performance of the student for both the viewing of the student and the professor
REQ6	2	The system should let the student use LEDs that turn on and off based on the input given to it.
REQ7	2	The system should give the student a limited amount of time before it decreases the grade if the time limit has passed.
REQ8	5	The system should give a list of all the materials available, and decrease the student's grade if the wrong material for a given lab is selected.
REQ9	5	The system should allow flexible wiring to be added to the protoboard and other logic devices.
REQ10	2	The system shall allow the student to use a "magnifying glass" that allows the student to see the inner workings of any logic chips.
REQ11	1	The system should be able to display to the student their achievements / "stars" for each lab they complete.
REQ12	2	The system shall take a picture of the student that is in a well lit condition that serves as the "base picture", and then a picture should be taken every minute thereafter and match the pictures by facial recognition algorithms.
REQ13	3	The system shall assign students grades based on their performance in the labs which is decided by: materials chosen, materials placed in the right place, time limit, and final results.
REQ14	1	The system shall connect the professors to all the students to answer questions in a chat like service inside the application.

REQ15	1	The system shall produce grades in a common markup language for the professor to use and upload to his/her grading system.
REQ16	4	The system shall give pre and post quiz that is specific to each laboratory to ensure the student is properly prepared for the lab.
REQ38	3	The system shall have a login functionality for both instructors and students.
REQ39	3	The system will allow admins to create new users and put them into the application.
REQ40	1	The system will allow users to recover passwords.
REQ41	5	The system shall allow the users to select what labs to perform.
REQ42	2	The system shall save the state of each lab so that the user is able to return if disconnected.
REQ25	1	The last screen of the lab module will display a congratulations message and prompt the user to return to the main menu.

2.1.2. Lab 1 Requirements

Number	**PW**	**Requirement**
REQ17	4	The system will let the user implement a simple logic function to become familiar with future labs.
REQ18	5	The system will ask the user to drag three ICs that can be used to implement the given logic function.
REQ19	3	The system will have a click and drag interface for the logic probe.
REQ21	5	The system will present the user with a logic function and a truth table that is partially filled out. Then the system will ask the user fill the rest of it out.
REQ22	3	The system will have a “Check” button that allows the users to check their answers. If the answer is wrong, the button changes to “Try Again” and the user will have two more attempts.
REQ23	5	The system will display a breadboard, power supply, ICs, and toggle switches in order to implement the boolean function. There will be a logic probe for debugging.
REQ24	4	The system will prompt the user to answer whether the logic function can be implemented with solely NAND gates. For this lab, the answer is “Yes” but if the user selects “No”, an explanation will be provided as to why.

2.1.3. Lab 2 Requirements

Number	PW	Requirement
REQ26	3	The student should be able to mark Karnaugh maps with the logic function as a minterm given.
REQ27	2	The student should be able to get the minimal sum-of-products expression and draw the logic diagram with a 2 input NAND, 3 input NAND gate, 2 input OR gate and a hex inverter.
REQ28	3	The student should be able to get the outcome waveform with the logic diagram drawn.
REQ29	2	The student should be able to modify the sum-of-products expression to eliminate timing hazard and make a new logic diagram.
REQ30	4	The system should allow the student to be able to construct a given logic circuit, and use it to construct a logic table of the output given each set of inputs
REQ31	5	The system should allow the student to use the given components to construct a full adder and verify that it works correctly for all input combinations.

2.1.4. Lab 3 Requirements

Number	PW	Requirement
REQ32	5	The system should be able to let the student select from a list of available materials, however, for this lab, three multiplexers and an exclusive OR gate will be used.
REQ33	5	The system should check if the student has put the lab materials in the right configuration in the protoboard for the decoder configuration, and encoder configuration for Binary to Gray Code.
REQ34	5	The user will be asked to enter the correct response to the equation needed to solve the Binary to Gray code conversion.

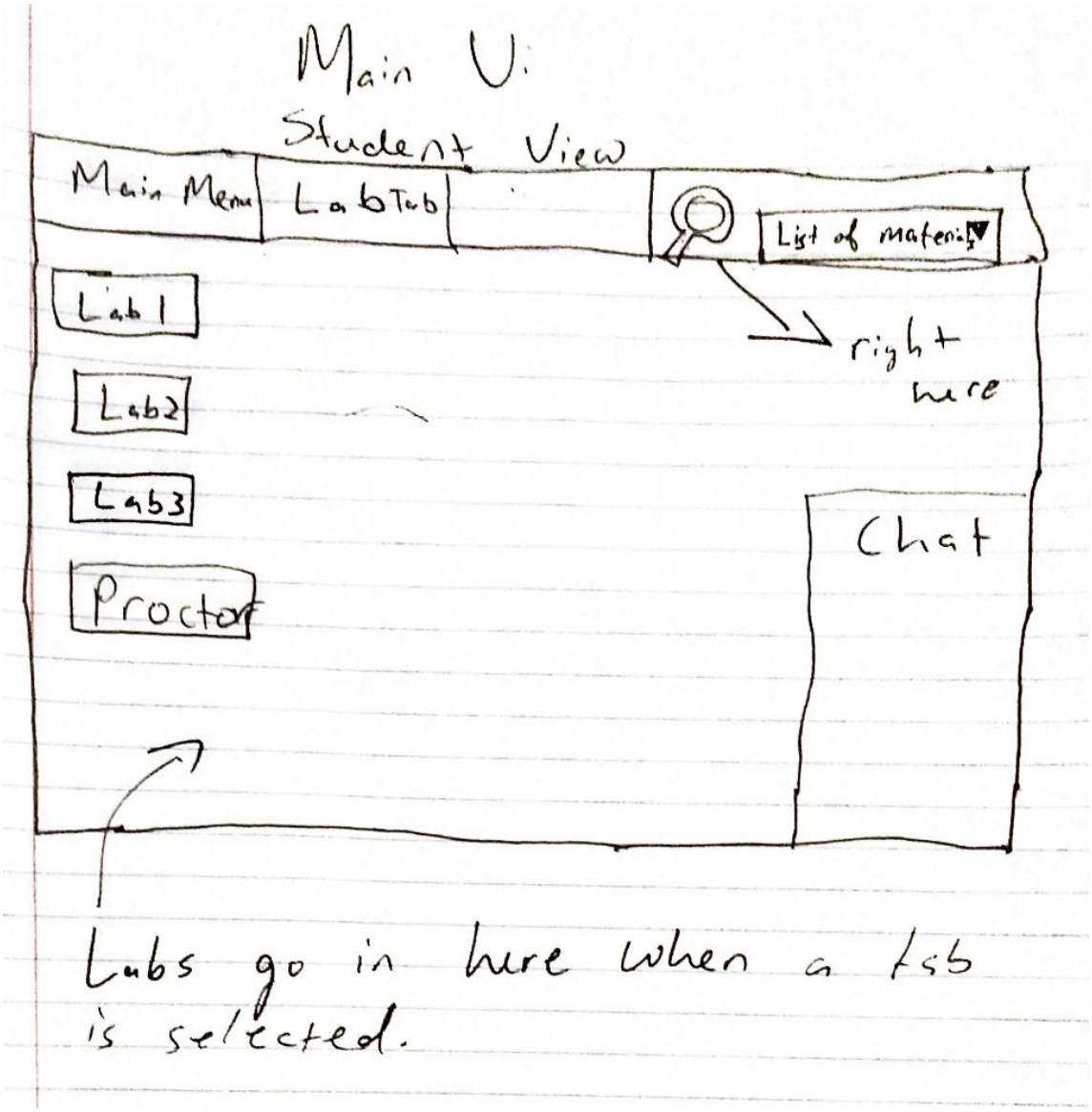
2.2. Enumerated Nonfunctional Requirements

Number	PW	Requirement
REQ35	2	The system should give feedback to every right or wrong action.

2.3. On-Screen Appearance Requirements

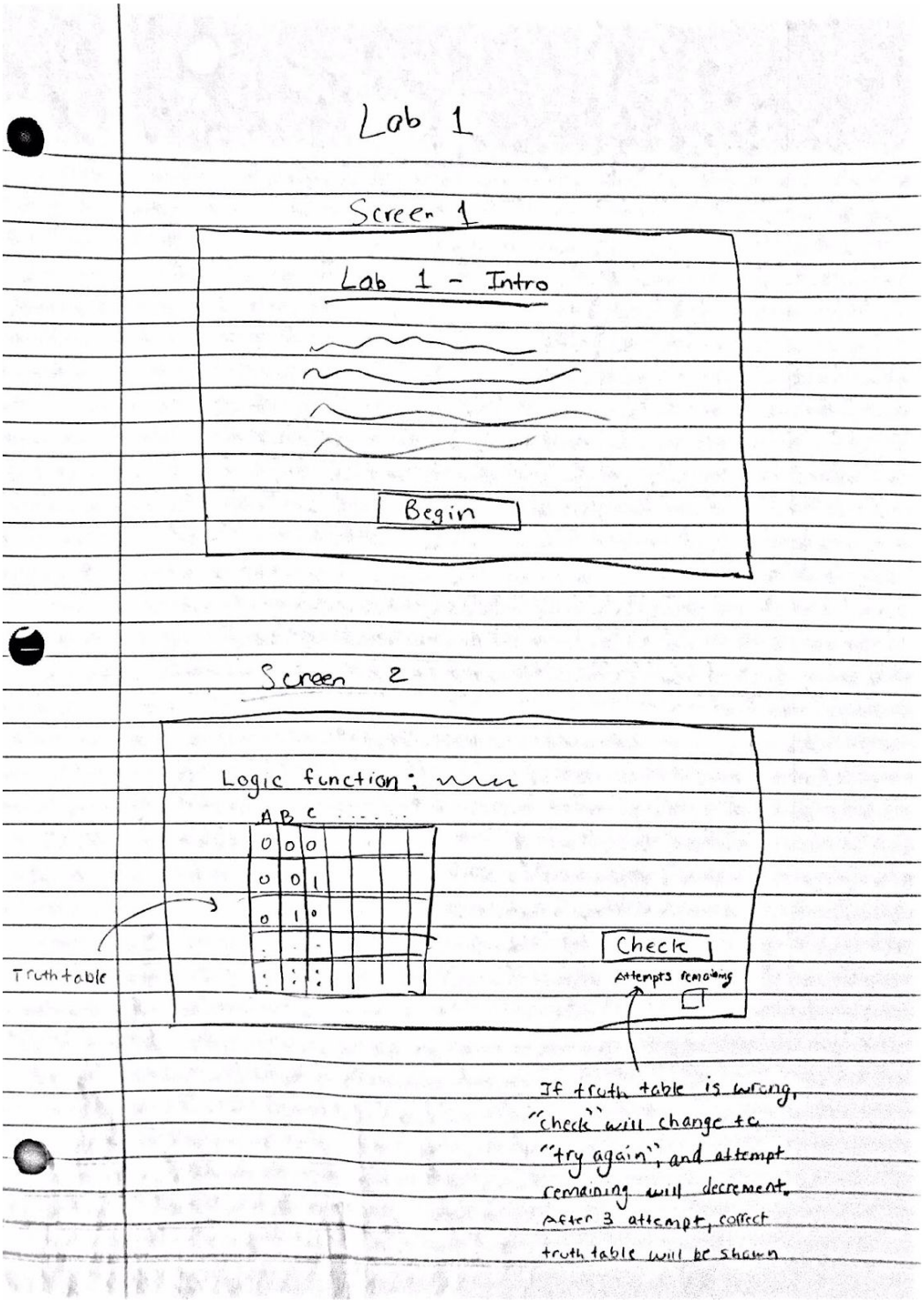
Each subsection below contains hand-drawn diagrams of each specific interface that we plan to include in this project. Each has a small description below it to distinguish what it is and when it will be presented to the user.

2.3.1. Main UI



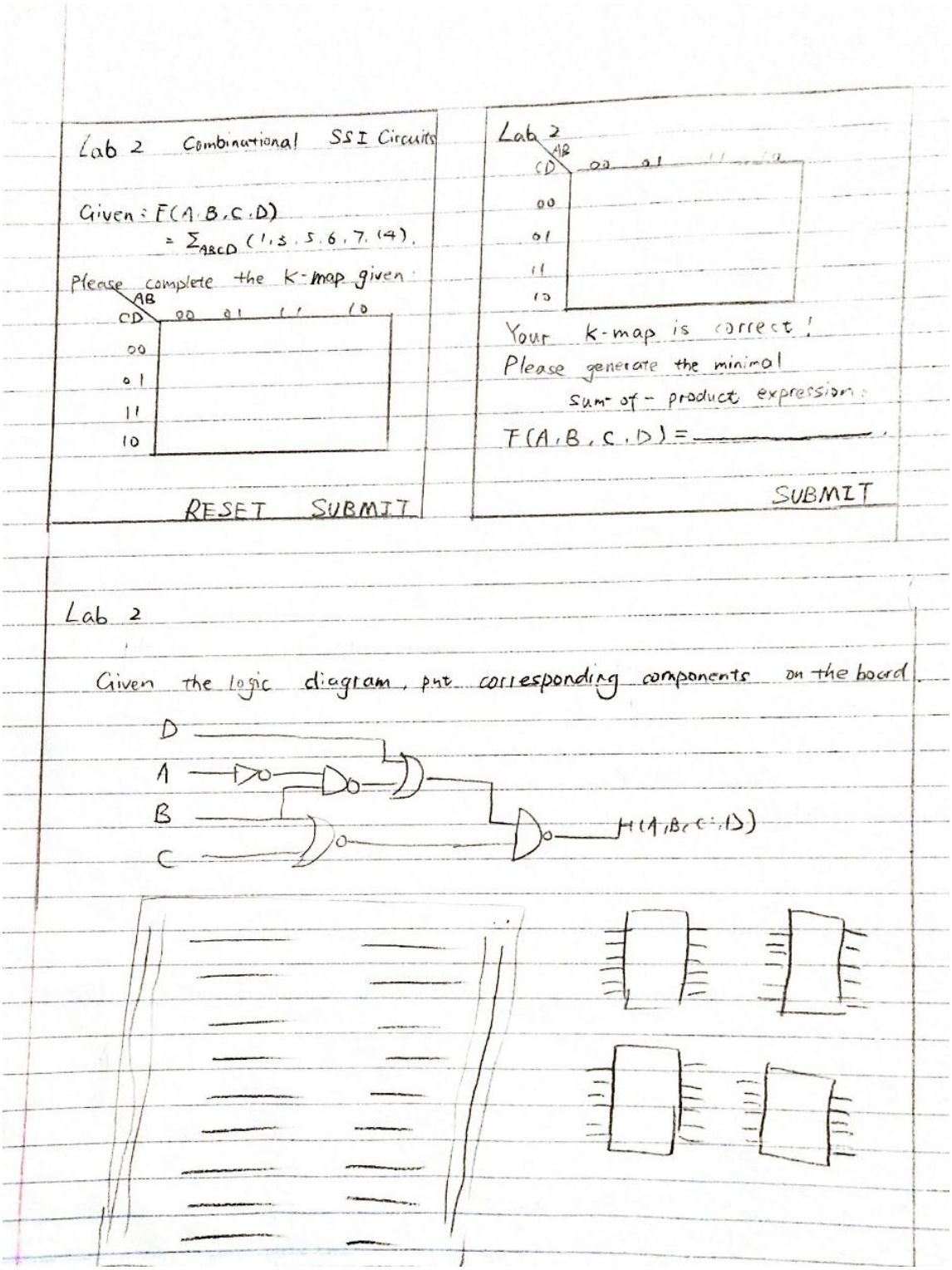
Above is the diagram of our main menu. The bar at the top contains a main menu button (top left), as well as a drop-down tab that will allow students to select the lab that they wish to complete. At the top right is a drop-down tab that will allow users to select chips and wires to use in circuits, and a magnifying glass to inspect the elements. The bottom right has a small chat window where students can interact with each other and the instructors.

2.3.2. Lab 1 UI



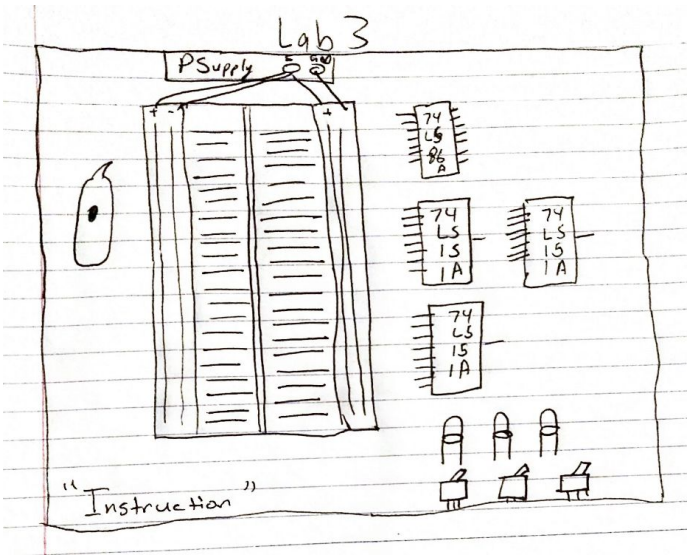
This image shows the interface for the first lab. The top section displays the intro given to the user before beginning the lab, while the bottom screen shows an example problem that a student will be asked to complete. In this case, the user is being asked to fill out a truth table, and click on the check button in the bottom right to submit the answer.

2.3.3. Lab 2 UI



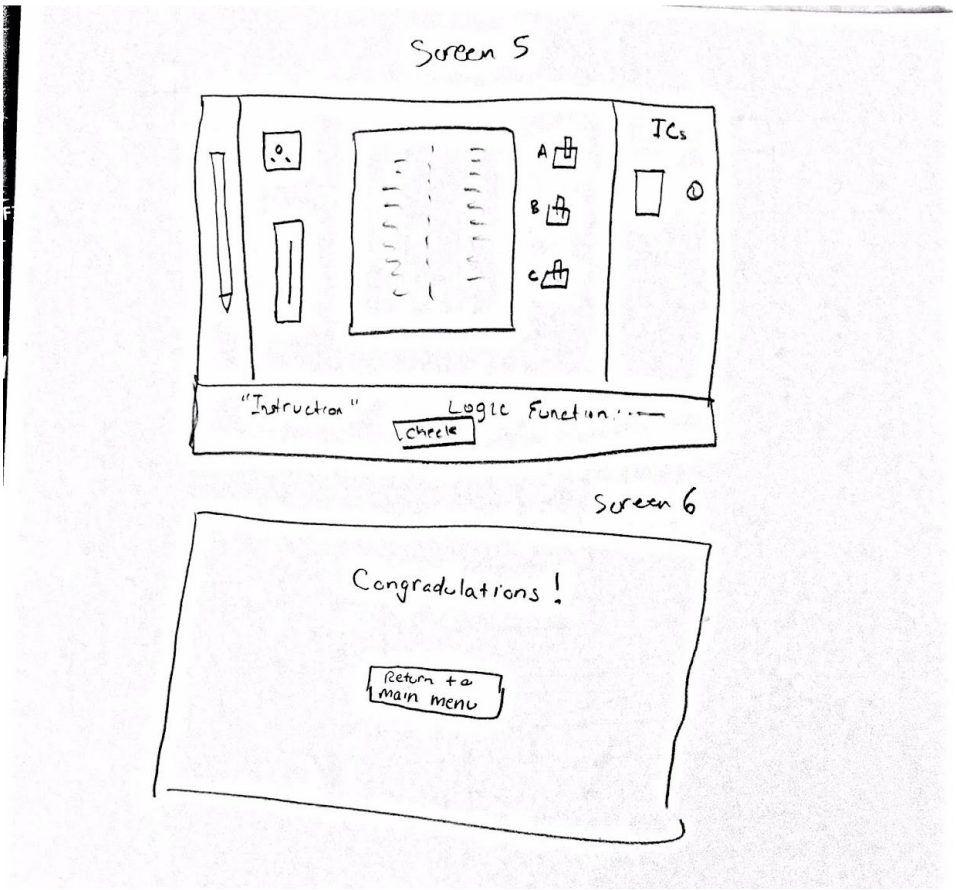
This interface, similar to the one from lab 1, is the interface for lab 2. In this lab, the user will have to fill out a Karnaugh map (top 2 screens), and construct a full adder (bottom screen) with the gates on the right and the protoboard on the left.

2.3.4. Lab 3 UI



This is the interface for lab 3, similar to the previous one, with the protoboard and chips to build the desired circuit.

2.3.5. Completed Lab UI



This is our final interface, displayed after the completion of a lab. If the user is correct, the image on the bottom will show, displaying a congratulations message.

3. Functional Requirements Specification

3.1. Stakeholders

The stakeholders of our system are Students and Instructors (Professors/Teachers).

- Student
    - The student will need to be able to complete assigned lab reports and must be able to communicate any questions they have to their teacher. They should also be responsible for making sure the cheat detector (Camera) detects them taking the laboratory.

- Instructor (Professor, Teacher)
    - The teacher will need to be able to enhance the learning of the students by viewing statistics and grades to see what should be focused on. They will be able to communicate with the students, check the grades for each students that have completed the labs, and be able to manage any new or existing users.

3.2. Actors and Goals

- Student (Initiating):
    - to perform digital logic experiments and get results at the end of each labs using logic equipment such as encoders, decoders, multiplexers … etc.
- Teacher (Initiating):
    - to present assignments (labs) and obtain grades for all the students in the class. Also they should be able to have live interactions with the students.
- Database (Participating):
    - SQL relational database that keeps all the data, including usernames, passwords, chat, and grades.
- Camera (Participating):
    - Facilitates the cheat detector to insure that students are doing their own assignments.

3.3. Use Cases

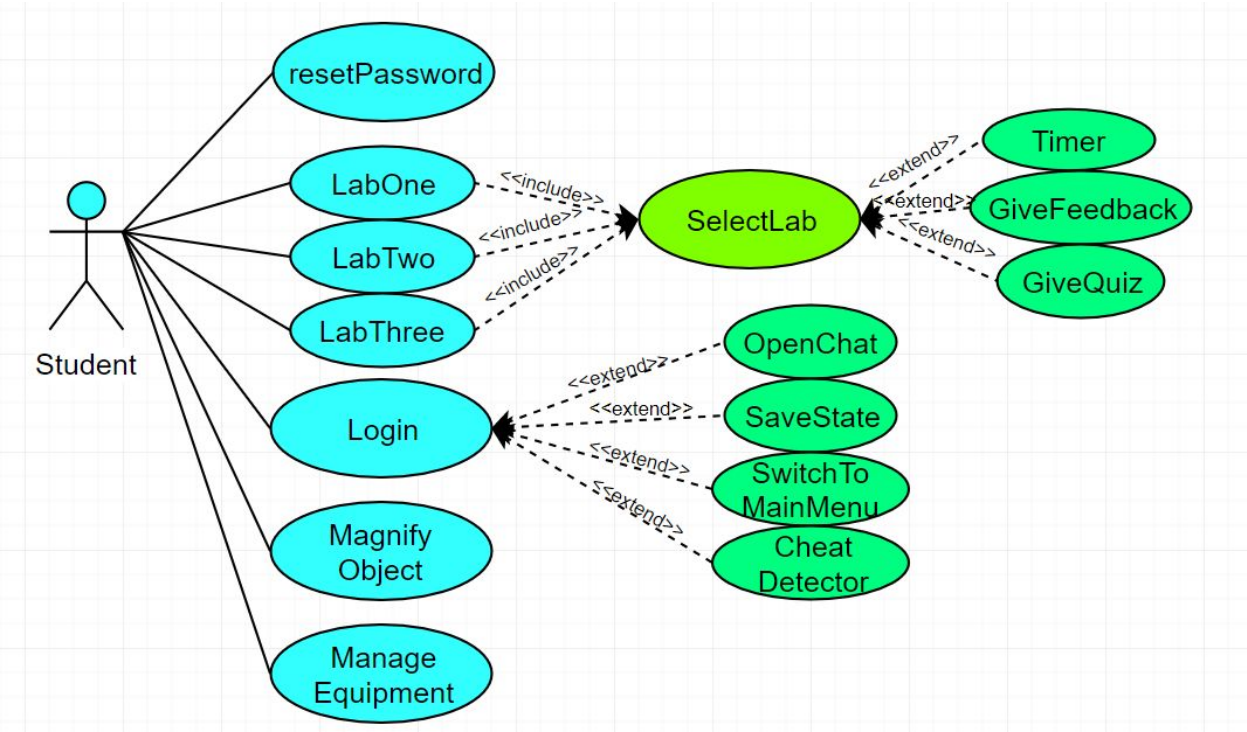
3.3.1. Casual Description

Use Case	Name	Description	Requirement
UC-1	manageEquipment	Allows the user to get equipment from a drop down menu and place them in the lab, connect them logically, and or delete them.	REQ1, REQ2, REQ3, REQ4, REQ6, REQ8 REQ9, REQ17, REQ18, REQ19,REQ22, REQ23, REQ24, REQ27, REQ32
UC-2	finishAndCheck	Checks the output for all logical input cases and compares with the correct answers	REQ5, REQ22, REQ31, REQ33
UC-3	login	Logs the user or instructor into the system to their respective views.	REQ38
UC-4	manageUsers	Allows the admin to add a user with a set password, and delete users.	REQ39, REQ40
UC-5	grades	Grading subsystem for instructor to view or delete grades. The instructor can also choose the download the grades in a markup language.	REQ8, REQ13, REQ15
UC-6	selectLab	Allows students to select a specific lab and open it up to the screen. Allows students to select lab 1, lab 2, and lab 3.	REQ41
UC-7	magnifyEquipment	Allows student to obtain additional information on different objects	REQ10
UC-8	timer	Counts down to let the student know how much time is remaining to either grade them on their performance, or give them suggestions on how to perform better.	REQ7

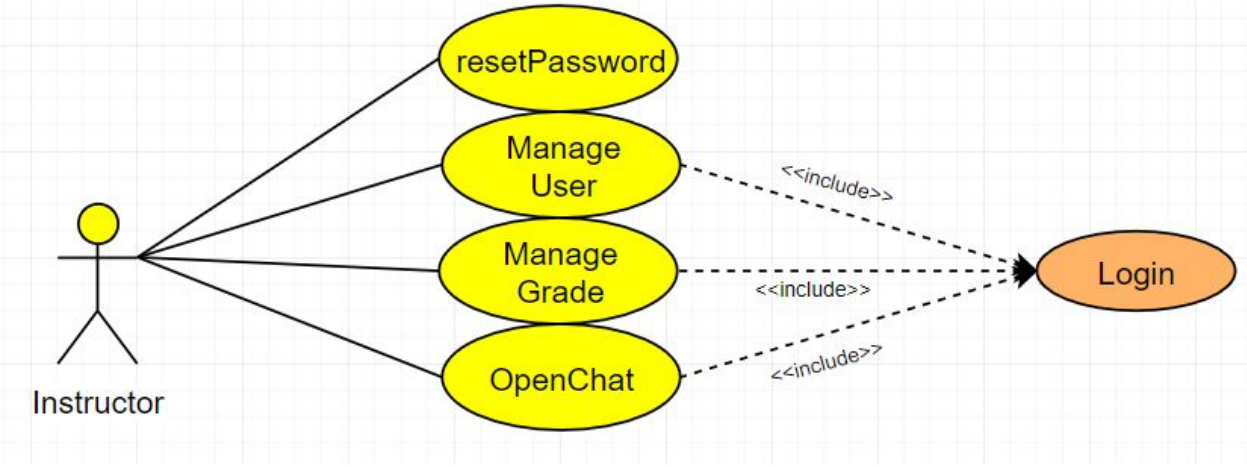
UC-9	saveState	Saves state of the lab if the student shuts the program down or changes lab.	REQ42
UC-10	switchToMainMenu	Allows the user to stop their current activity and go back to the main menu.	REQ25
UC-11	openChat	Allows anyone to open and speak in the program chat or contact instructor	REQ14
UC-12	giveFeedback	Give the students achievement starts for getting good grades, or let them know if they were incorrect	REQ8, REQ11, REQ13, REQ35
UC-13	cheatDetector	Allows instructors to track students and identify cheating with the camera.	REQ12
UC-14	giveQuiz	gives the student pre and post quizzes. May ask to fill out karnaugh maps and truth tables.	REQ16, REQ21, REQ26, REQ27, REQ28, REQ29, REQ30, REQ34
UC-15	labOne	Perform laboratory one which also acts as a tutorial to the students on how to perform the labs.	REQ 17, REQ 18, REQ 19, REQ 21, REQ 22, REQ 23, REQ 24
UC-16	labTwo	Perform laboratory two which lets student implement a ‘full adder’	REQ 26, REQ 27, REQ 28, REQ 29, REQ 30, REQ 31
UC-17	labThree	Perform laboratory three which lets student implement a ‘binary to gray’ converter	REQ 32, REQ 33, REQ 34

3.3.2. Use Case Diagram

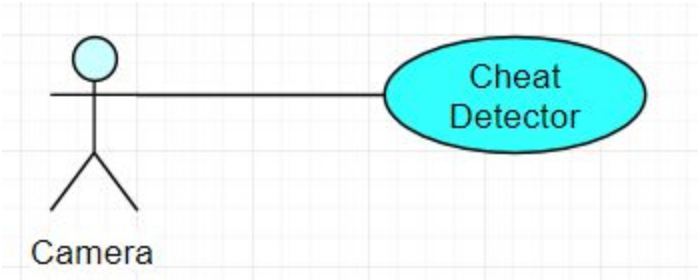
3.3.2.1. Student:



3.3.2.2. Instructor:



3.3.2.3. Camera:



3.3.3. Traceability Matrix

	PW	UC-1	UC-2	UC-3	UC-4	UC-5	UC-6	UC-7	UC-8	UC-9	UC-10	UC-11	UC-12	UC-13	UC-14	UC-15	UC-16	UC-17
REQ1	5	X																
REQ2	2	X																
REQ3	3	X																
REQ4	5	X																
REQ5	4		X															
REQ6	2	X																
REQ7	2								X									
REQ8	2	X				X							X					
REQ9	5	X																
REQ10	2							X										
REQ11	1												X					
REQ12	2													X				
REQ13	3					X							X					
REQ14	1											X						
REQ15	1					X												
REQ16	3														X			
REQ17	4	X														X		
REQ18	5	X														X		
REQ19	3	X														X		
REQ21	5														X	X		
REQ22	3	X	X													X		
REQ23	5	X														X		
REQ24	4	X																
REQ25	1										X							
REQ26	3														X		X	
REQ27	2	X													X		X	
REQ28	3														X		X	
REQ29	2														X		X	
REQ30	4														X		X	
REQ31	5		X														X	
REQ32	5	X																X
REQ33	5		X															X

	PW	UC-1	UC-2	UC-3	UC-4	UC-5	UC-6	UC-7	UC-8	UC-9	UC-10	UC-11	UC-12	UC-13	UC-14	UC-15	UC-16	UC-17
REQ34	5														X			X
REQ35	2												X					
REQ38	3			X														
REQ39	3				X													
REQ40	1				X													
REQ41	5						X											
REQ42	2									X								
MAX PW	5	5	5	3	3	3	5	2	2	2	1	1	3	2	5	5	5	5
TOTAL PW	121	55	17	3	4	6	5	2	2	2	1	1	8	2	27	25	19	15

3.3.4. Fully-Dressed Description

3.3.4.1. UC-1 (manageEquipment):

Use Case UC-1:	manageEquipment
Requirements:	REQ1, REQ2, REQ3, REQ4, REQ6, REQ8 REQ9, REQ17, REQ18, REQ19,REQ22, REQ23, REQ24, REQ27, REQ32
Initiating Actor:	Student
Actor's Goals:	To open an items menu and select a logic device to place on the protoboard while constructing their circuit
Preconditions:	The student must be actively completing a lab, where they will be presented with the option to select/move logic devices.
Postconditions:	Once completed the student will be able to see the item in its correct place on the protoboard, and will be able to put other logic devices around it in order to complete the circuit
Flow of events:	
→The user clicks manage equipment button, and selected an equipment he/she wants. ←The equipment will follow user's mouse. →The user clicks the destination on a protoboard that they want to place the equipment to. ←The equipment stops at the destination.	

3.3.4.2. UC-2 (finishAndCheck):

Use Case UC-2:	finishAndCheck
Requirements:	REQ5, REQ22, REQ31, REQ33
Initiating Actor:	Student
Actor's Goals:	To submit their constructed circuit for grading
Preconditions:	The student must have completed the circuit for a particular lab, at which point they can click a button to submit their work for grading
Postconditions:	The student will have submitted his or her work, and the system will compare the student's answers to the correct solution, assigning grades accordingly.
Flow of events:	
→The user moves clicks and drags pieces to the protoboard and constructs the circuit ←The system responds, displaying the user's completed circuit →The user clicks the finish button ←The system moves the student to a completion screen, and compares the student's circuit to the correct version in order to compute the grade	

3.3.4.3. UC-5 (grades):

Use Case UC-5:	grades
Requirements:	REQ8, REQ13, REQ15
Initiating Actor:	Student/Instructor
Actor's Goals:	To view grades from completed labs, and in the case of the instructor, to document the grades for each student
Preconditions:	Labs must be completed for a grade to be assigned.
Postconditions:	After grades are assigned, both students and instructors will be able to view them.
Flow of events:	
Student: →The student completes and clicked finish button. ←The system instantly gives the corresponding grade of the student. →The student completes the post quiz and presses submit ←The system assigns the student the post-quiz grade. Instructor: →The instructor logins ←The system presents the instructor with a) Download Grade b) Display All Grades buttons →The instructor clicks a) Download Grade b) Display All Grades → a) A pdf version of students' grade is downloaded b) All students' grades are displayed	

3.3.4.4. UC-14 (giveQuiz):

Use Case UC-14:	giveQuiz
Requirements:	REQ16, REQ21, REQ26, REQ27, REQ28, REQ29, REQ30, REQ34
Initiating Actor:	student
Actor's Goals:	The student will be required to take a quiz after the completion of each lab that will test their understanding of the work that was just done.
Preconditions:	The student must have completed the lab
Postconditions:	The student will be asked to complete a quiz, consisting of a mix of multiple choice questions, Karnaugh maps, and truth tables. Upon completion, they will be able to submit the quiz for grading
Flow of events:	
→The student submits their completed lab for grading ←The system responds by directing the user to a post-lab quiz if the lab was completed →The user completes the quiz by filling out necessary information →The user submits for grading by pressing submit ←The system compares the users answers to the correct answers and assigns grades accordingly	

3.3.4.5. UC-15 (labOne):

Use Case UC-15:	labOne
Requirements:	REQ 17, REQ 18, REQ 19, REQ 21, REQ 22, REQ 23, REQ 24
Initiating Actor:	Student
Actor's Goals:	Perform the lab (Introduction to Hardware) and the quizes included within it and receive a grade in the end.
Preconditions:	Student must have logged in and not have finished this particular lab prior to starting it.
Postconditions:	Student will have finished all the requirements to finish this lab, including the quiz, and have received a grade for it.
Flow of events:	
→The student clicks on the 'Laboratory One' button. ←The system switches the scene to the introduction to lab one and presents a pre-quiz to prepare the student →The student enters the required answers correctly ← The system starts the lab and gives the user the option to use any devices available in the system that they wish → The student can use the 'magnifying glass' object to inspect particular equipment ← The system shows the student more information about the particular device. → The student configures the devices in the protoboard as they wish with set outputs designated in a predefined area and clicks finish after he or she is finished. ← The system checks the output with the required output to ensure that the user has completed the lab correctly. The system will require the student to use an Inverter, And gate, and Or gate with the correct output on the designated LEDs to pass the experiment correctly. The system then presents the student with a post-quiz which is graded. → The student takes the post-quiz, answers the questions, and presses submit. ← The system checks the answer and assigns the grade accordingly.	

3.3.4.6. UC-16 (labTwo):

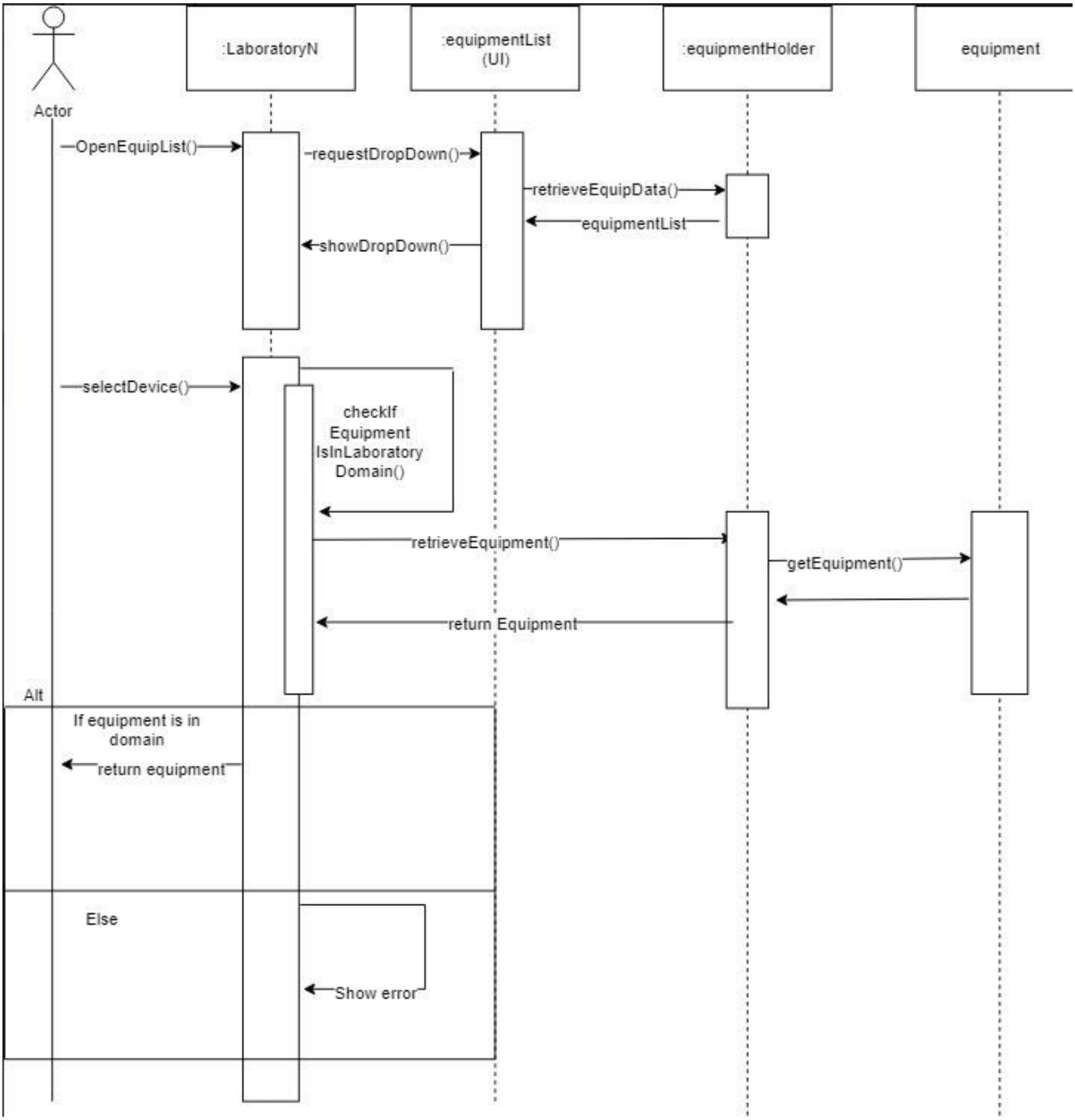
Use Case UC-16:	labTwo
Requirements:	REQ 26, REQ 27, REQ 28, REQ 29, REQ 30, REQ 31
Initiating Actor:	Student
Actor's Goals:	Perform the lab (Combinational SSI Circuits) and the quizes included within it and receive a grade in the end.
Preconditions:	Student must have logged in and not have finished this particular lab prior to starting it.
Postconditions:	Student will have finished all the requirements to finish this lab, including the quiz, and have received a grade for it.
Flow of events:	
→The student clicks on the 'Laboratory Two' button. ←The system switches the scene to the introduction to lab one and presents a pre-quiz to prepare the student →The student enters the required answers correctly ← The system starts the lab and gives the user the option to use any devices available in the system that they wish → The student can use the 'magnifying glass' object to inspect particular equipment ← The system shows the student more information about the particular device. → The student configures the devices in the protoboard as they wish with set outputs designated in a predefined area and clicks finish after he or she is finished. ← The system checks the output with the required output to ensure that the user has completed the lab correctly. The system will require the student to use an Inverter, NAND gates, and Or gate to design a Full Adder with the correct output on the designated LEDs to pass the experiment correctly. The system then presents the student with a post-quiz which is graded. → The student takes the post-quiz, answers the questions, and presses submit. ← The system checks the answer and assigns the grade accordingly.	

3.3.4.7. UC-17 (labThree):

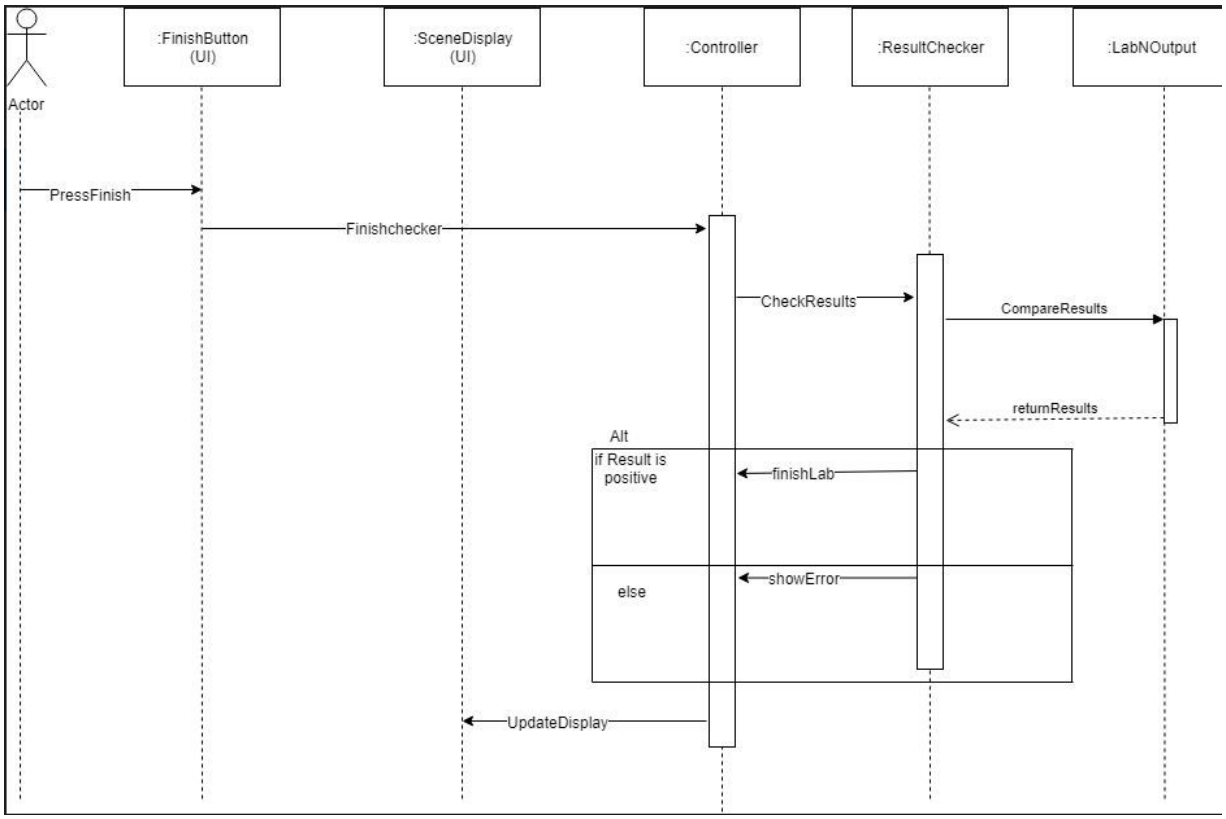
Use Case UC-17:	labThree
Requirements:	REQ 32, REQ 33, REQ 34
Initiating Actor:	Student
Actor's Goals:	Perform the lab (Combinational MSI Circuits) and the quizes included within it and receive a grade in the end.
Preconditions:	Student must have logged in and not have finished this particular lab prior to starting it.
Postconditions:	Student will have finished all the requirements to finish this lab, including the quiz, and have received a grade for it.
Flow of events:	
→The student clicks on the 'Laboratory Three' button. ←The system switches the scene to the introduction to lab one and presents a pre-quiz to prepare the student →The student enters the required answers correctly. ← The system starts the lab and gives the user the option to use any devices available in the system that they wish → The student configures the devices in the protoboard as they wish with set outputs designated in a predefined area and clicks finish after he or she is finished. ← The system checks the output with the required output to ensure that the user has completed the lab correctly with XOR gates, and MUXes to implement a grey to binary decoder. The system then presents the student with a post-quiz which is graded. → The student takes the post-quiz and answers the questions. ← The system checks the answer and assigns the grade accordingly.	

3.4. System Sequence Diagrams

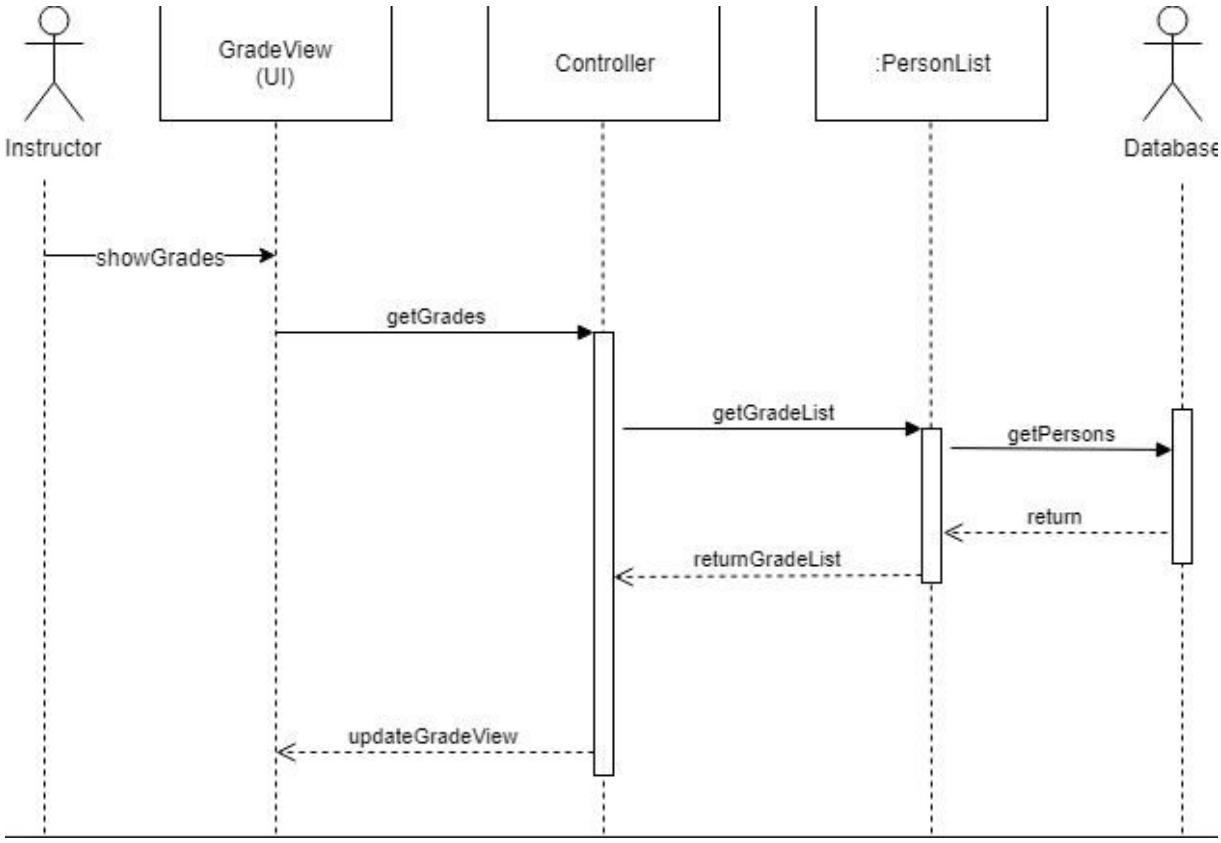
3.4.1. UC-1 (manageEquipment)



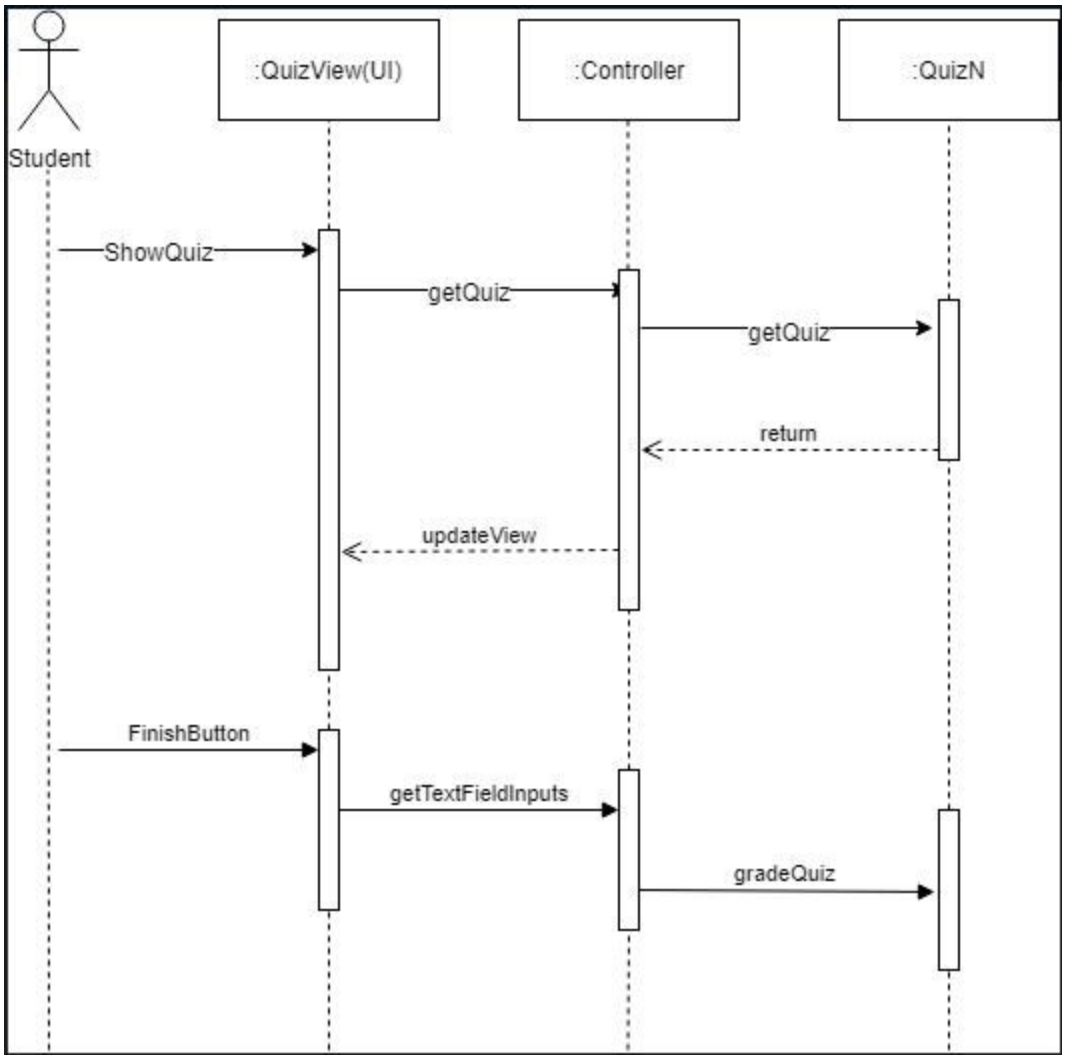
3.4.2. UC-2 (finishAndCheck)



3.4.3. UC-5 (grades)

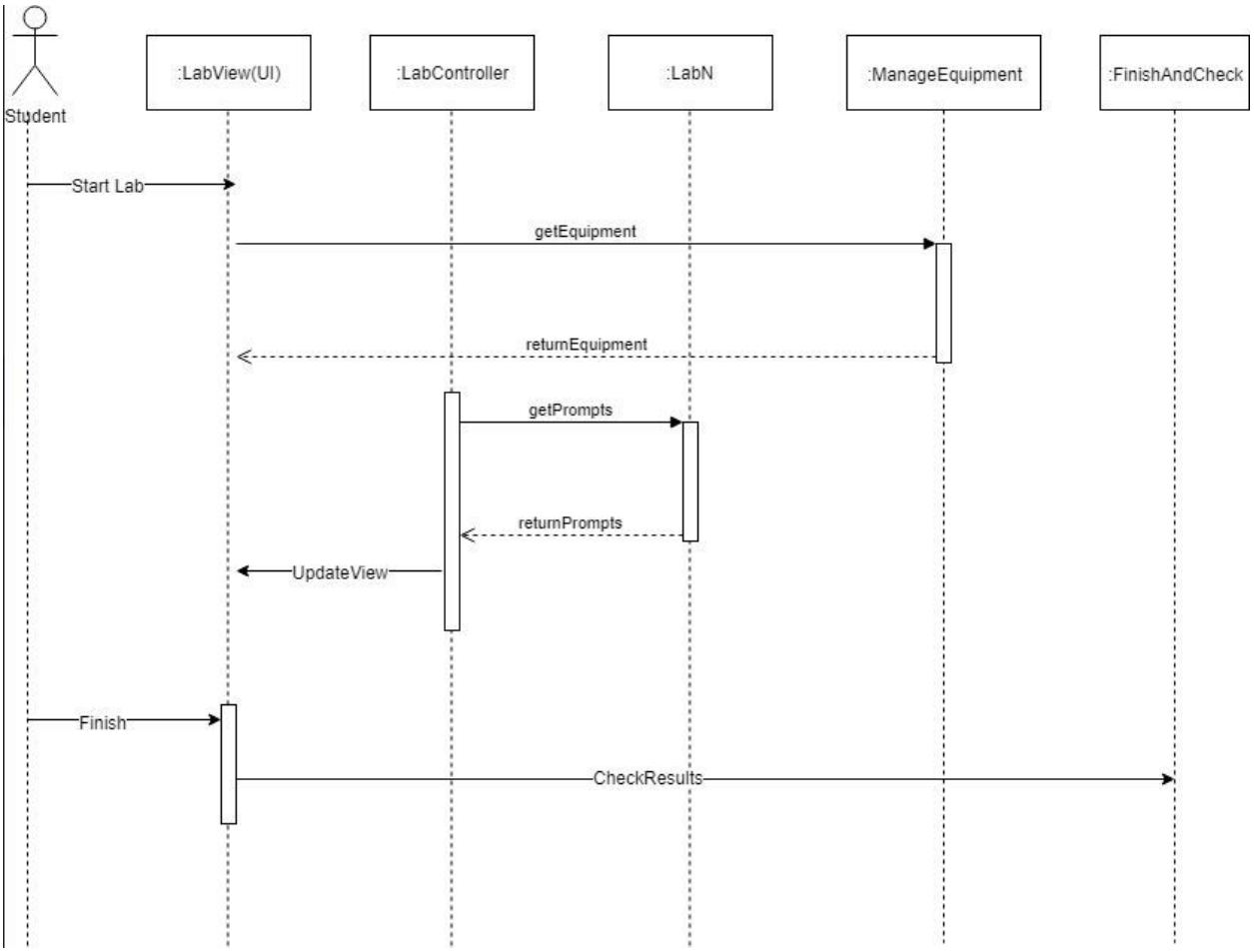


3.4.4. UC-14 (giveQuiz)



3.4.5. UC-15 - UC-17 (labOne, labTwo, labThree)

The system sequence diagram below demonstrates the general sequence for all three labs as they follow a very similar pattern:



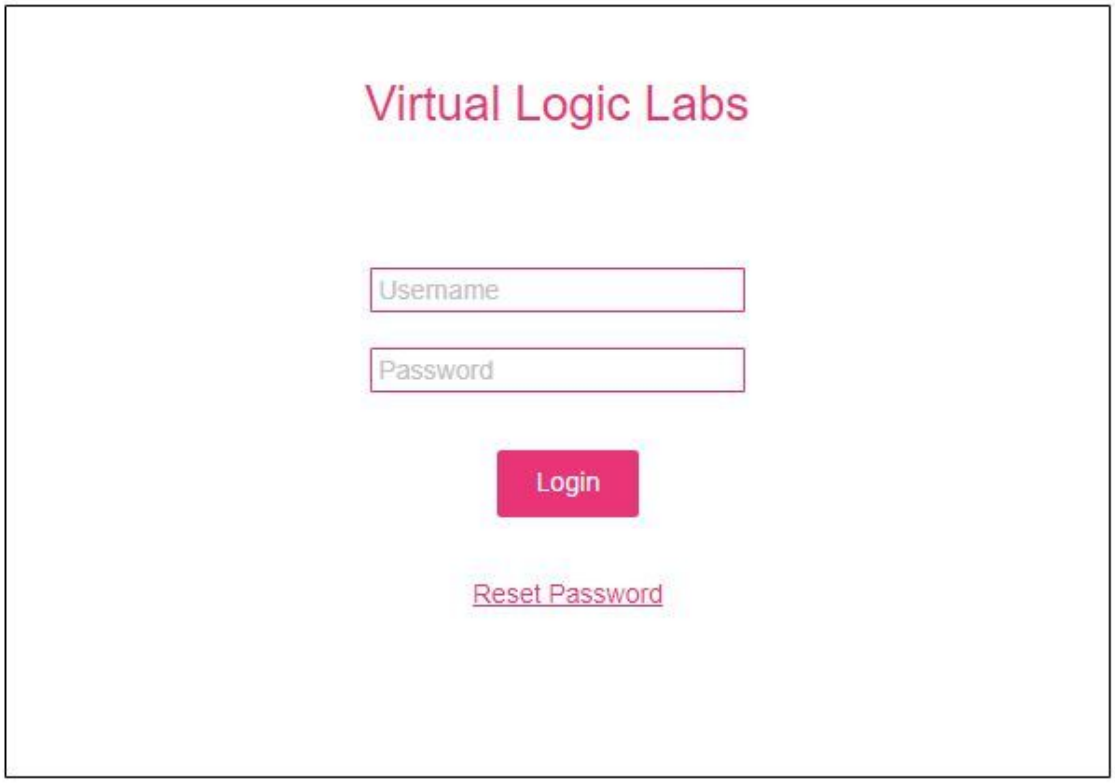
4. User Interface Specification

In this section, we decided to show more than the fully dressed cases above for the UI specification due to our application being graphics-heavy.

4.1. Preliminary Design

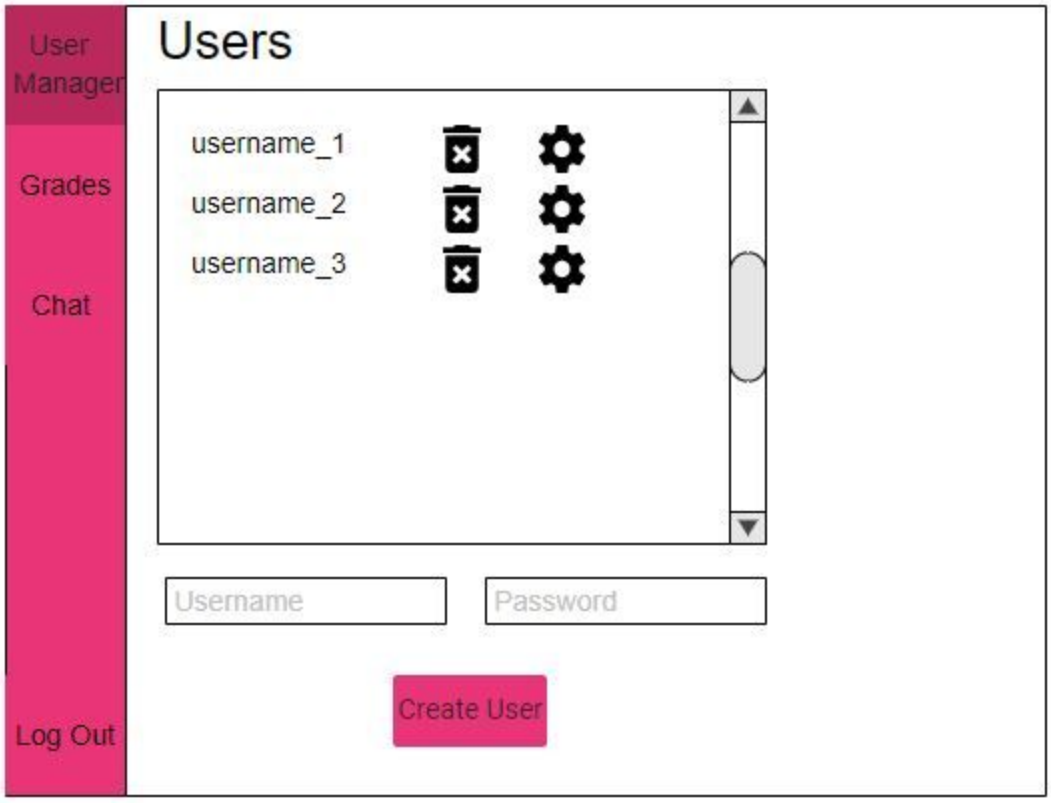
4.1.1. UC-3

- Opening UI:
  - Prompt the user for login to log into either Administrator mode or Student mode based on the username entered. The login initiates after the user pressed login.



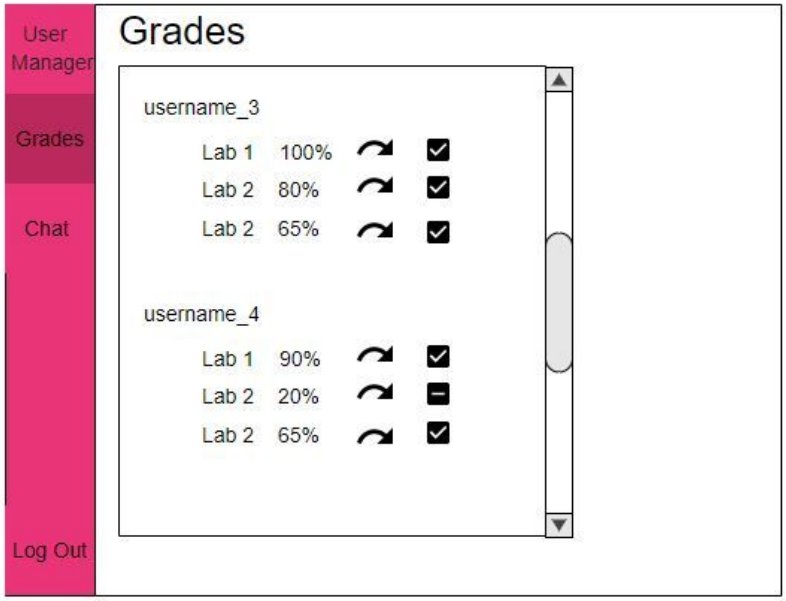
4.1.2. UC-4

- Administrator Subsystem View: User Manager
    - This is where the list of all the users in the entire system is. The instructor can go to settings to edit the password, or delete the user completely. In addition to that, the instructor can also add users with preset passwords. This system was designed oriented to the instructor as they are likely the ones to assigns students to do these labs, thereby preventing random users signing up to make an account.



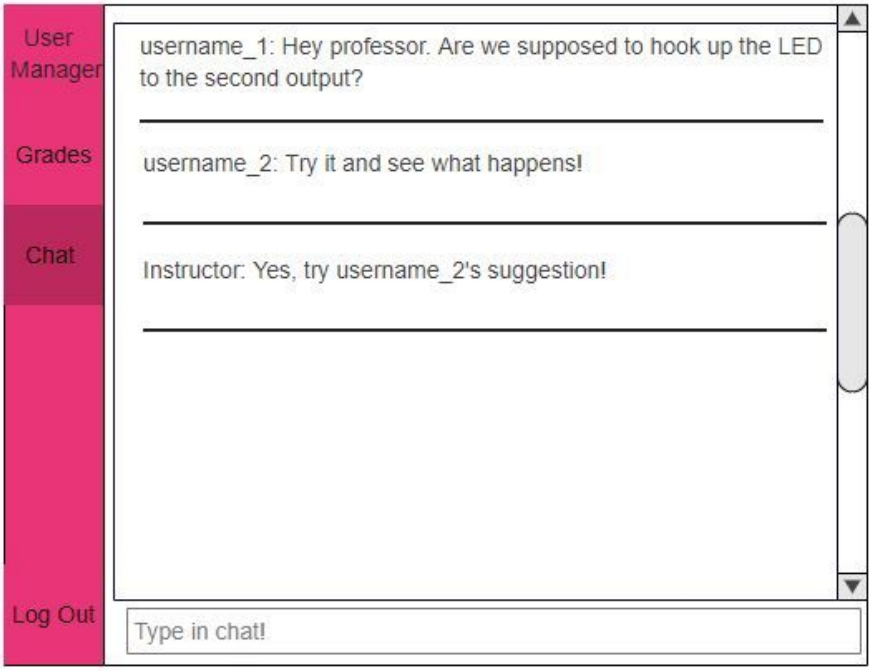
4.1.3. UC-5

- Administrator Subsystem View - Grades:
  - This is where the grades for every users lie. The scroll view has the list of every users, and the labs that each users performed with it's grade next to it. The instructor has the option to click the circular button next each lab grade to reset it back to 0% so the student will be able to attempt the lab again. The checkmark indicates that the student has completed the lab and that the system has verified that the student did not cheat in the lab.



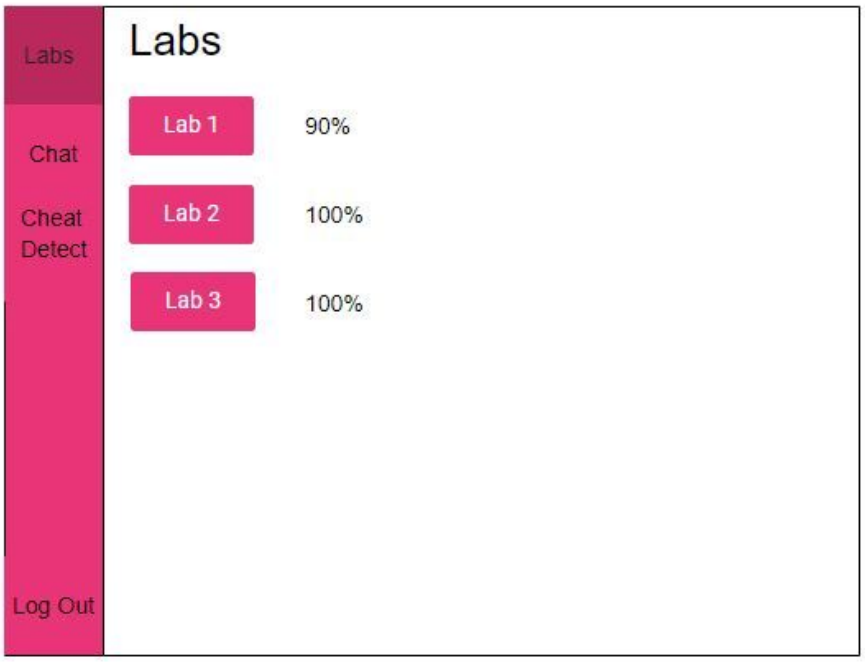
4.1.4. UC-11

- Administrator Subsystem View - Chat:
  - This is where the messaging system lies so the users and the instructor can interact with each other. The messages already written are in the top of the view, the messages being "typed" are in the bottom of the view. Pressing enter sends the message to the rest of the chat.



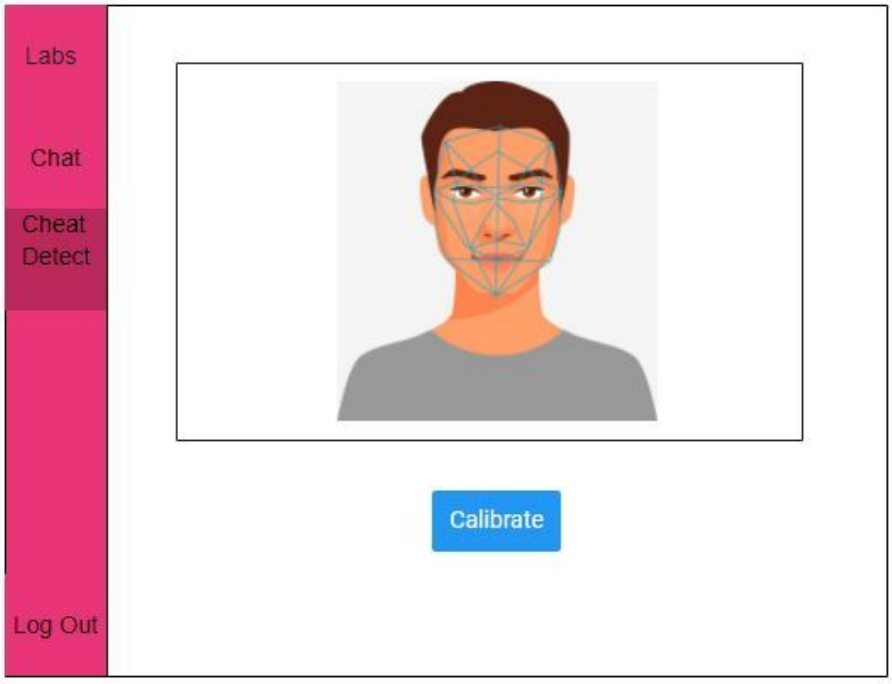
4.1.5. UC-6

- Student Subsystem View - lab selection:
  - The student/user selects the lab he or she wants to perform. If they have started the lab already, it presents how much of the lab they have completed already. Clicking on the lab opens it up for the student to perform it.



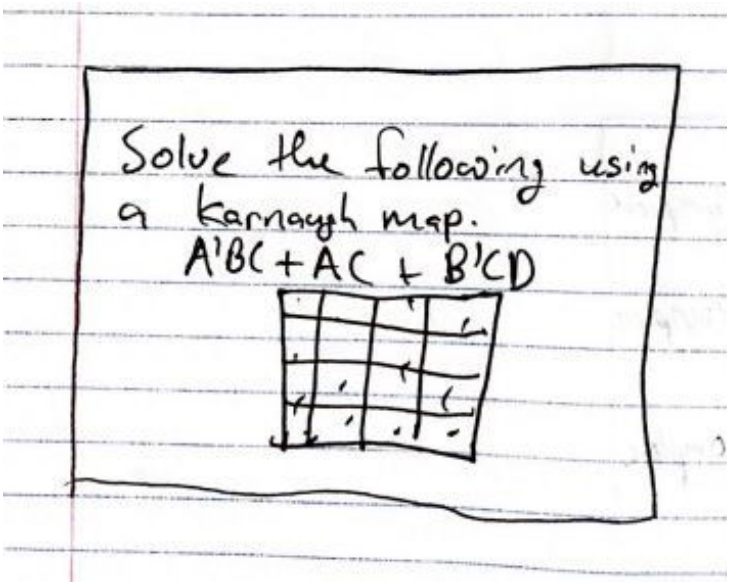
4.1.6. UC-13

- Student Subsystem - Camera Verification:
  - Takes a picture of the student to calibrate their face into the system. This is done to cross verify that the lab is being done by the same person that registered for it.



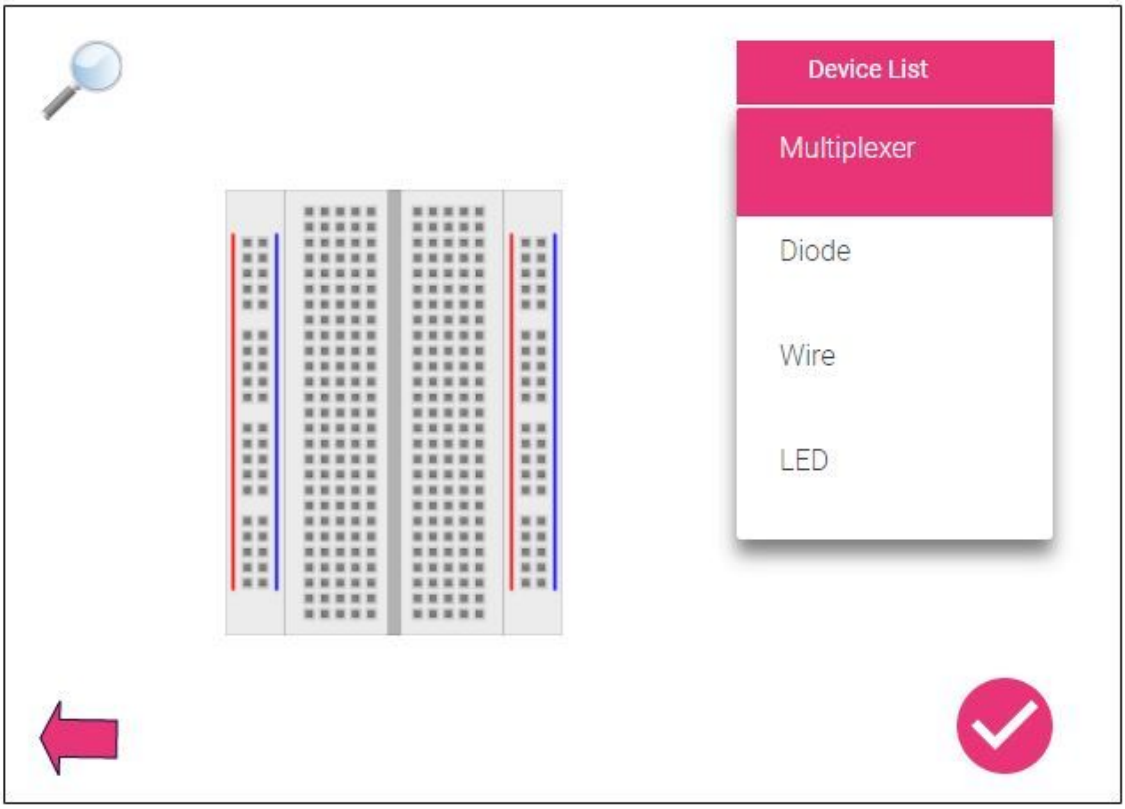
4.1.7. UC-14

- Student Subsystem - Lab Pre/Post Quiz example:
  - The student would have to interact with the karnaugh map, and the system would only allow the correct answer for the user to get through to the next part of the lab.



4.1.8. UC-1, UC-7, UC-15, UC-16, UC-17

- Student Subsystem - lab/manage equipment:
  - Below is a general example of what a lab is going to look like in addition to the equipment selection system.



4.2. User Effort Estimation

0. Log In - Total variable presses + 1/0 clicks
    a. Enter Username (variable length)
    b. Enter Password (variable length)
    c. Press enter/click log in

1. Manage Equipment - Total 3 Clicks
    a. Click on drop down menu that holds all the equipment
    b. Click on the equipment desired which places the equipment under the mouse
    c. Click anywhere in the view to place the equipment

2. Grades - Total 2 Clicks
Assumption: Instructor logs in
    a. Instructor clicks on the grades tab

3. Set Cheat Detector Camera - 2 Clicks
Assumption: Student is already logged in and camera exists
    a. Click on “ProctorTrack” (name to be changed) tab.
    b. Press calibrate and wait for software approval

4. Chat - Total 2 clicks + 1 press
Assumption: Student is already logged in
    a. Click on Chat tab
    b. Click on the text box on the bottom to enter text
    c. Press enter

5. Create User - Total 3 clicks
Assumption: Instructor is already logged in
    a. Click the User Management tab
    b. Enter username and password
    c. Click create user

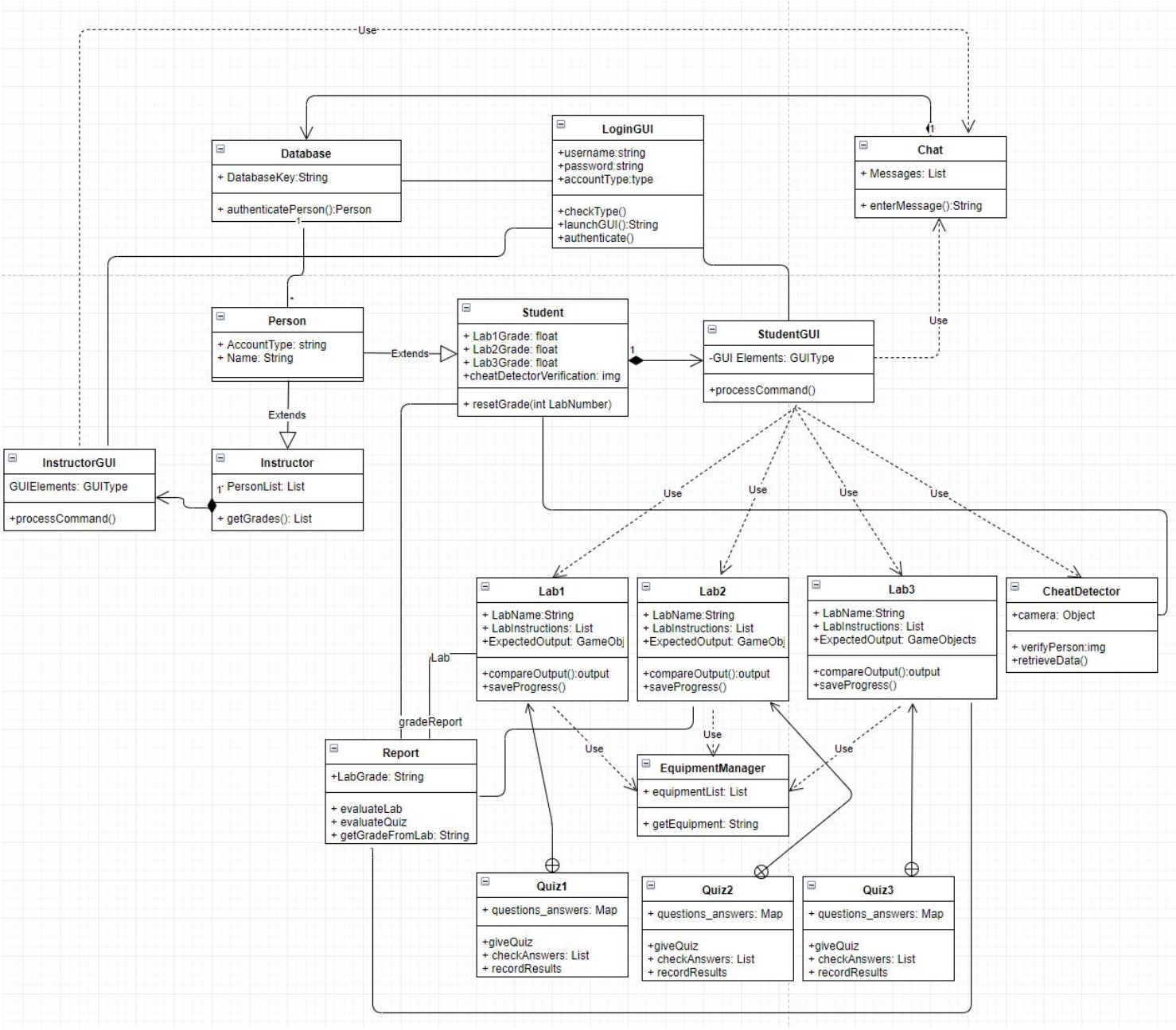
6. Select Lab - Total 2 clicks
Assumption: User is already logged in
    a. Click on the Labs tab
    b. Click on the lab of choice

7. Individual labs
    Implementation specific. Need to explore further.

8. Individual quizzes
    Implementation specific. Need to explore further.

5. Domain Analysis

5.1 Domain Model



5.1.1 Concept Definitions

Responsibility Description	Type	Concept Name
Provides the interface GUI for the actors to login to the application, the authentication is done through the database	K	LoginGUI
Allows the user to communicate with other users	D	Chat
Stores student account data	K	Person
Presents the user with lab 1 activity	D	Lab1
Presents the user with lab 2 activity	D	Lab2
Presents the user with lab 3 activity	D	Lab3
Provides user with tool to interact with the lab environment in order to perform the lab	D	EquipmentManager
Generates a summary of lab performance results	D	Report
Provides an interface for the student to interact with GUI and interacts with Lab classes to show the laboratory simulations	D	StudentGUI
Provides an interface for the instructor to interact with the GUI	D	InstructorGUI
Identifies if the user is cheating by using the camera	D	CheatDetector
Holds ‘Persons’ in an online database, and holds chat messages	D	Database
Administers a quiz to the student at the end of lab 1	D	Quiz1
Administers a quiz to the student at the end of lab 2	D	Quiz2
Administers a quiz to the student at the end of lab 3	D	Quiz3
Student that participates in the Laboratory Simulation portion of the application	D	Student
Instructor that views the grade and oversees the account creations in the application	D	Instructor

5.1.2 Association Definitions

Concept Pair	Association Description	Association Name
Student ↔ StudentGUI	Student passes request to StudentGUI to display student interface qualities	Conveys request
Instructor ↔ InstructorGUI	Instructor passes request to InstructorGUI to display instructor interface qualities	Conveys request
StudentGUI ↔ LoginGUI	Actor's interface signifies that a student is attempting to sign in.	Conveys request
InstructorGUI ↔ LoginGUI	Actor's interface signifies that an instructor is attempting to sign in.	Conveys request
LoginGUI ↔ StudentGUI	Login passes request to StudentGUI to trigger student interface from the Student class	Generates
LoginGUI↔InstructorGUI	Login passes request to InstructorGUI to trigger instructor interface from the Instructor class	Generates
Lab↔Equipment	Active Labs will allow students to interact with equipment as they build the necessary circuits	Generates
Lab1↔Quiz1	The completion of a lab 1 triggers a quiz that the student must complete to receive a grade	Generates
Lab2↔Quiz2	The completion of a lab 1 triggers a quiz that the student must complete to receive a grade	Generates
Lab3↔Quiz3	The completion of a lab 1 triggers a quiz that the student must complete to receive a grade	Generates
Lab↔CheatDetector	Active labs enable CheatDetector to identify if a student is using outside materials	Requests notify
CheatDetector↔Camera	CheatDetector accesses camera and uses it to monitor student as labs and quizzes are in progress	Provides data
StudentGUI↔CheatDetector	The GUI uses the CheatDetector to allow the student to calibrate his/her face and	Generates

	verify.	
Student↔Report	The student class uses the report to assign the student the respective lab's grade.	Provides data
Lab↔Report	The lab uses the report to generate the grade for the lab portion of the simulation	Generates
Quiz↔Report	After the quiz is finished, the quiz generates a grade report for the Quiz portion of the simulation	Generates
Chat↔StudentGUI	The chat functionality is displayed in the Student section of the GUI	Generates
Chat↔InstructorGUI	The chat functionality is displayed in the Instructor section of the GUI	Generates
Chat↔Database	The chat queries the database for new messages	Requests notify
CheatDetector↔Student	The cheat detector uses the image associated with the student to verify that the user is the correct student.	Requests notify
Student↔Database	Generate student class from the database	Generates
Instructor↔Database	Generate instructor class from the database	Generates

5.1.3 Attribute Definitions

Concept	Attributes	Attribute Description
Login	checkType()	Checks the user is a professor or a student
	authenticate()	Checks if the username and password match the record in the database
Lab	compareOutput()	Compares the output of the logic to the answer
	saveProgress()	Saves student's current work to database
Equipment	getEquipment()	Gets the equipment user wants ready to place
Report	EvaluateLab()	Evaluates the lab to generate a grade for the specified lab.
	EvaluateQuiz()	Evaluates the quiz to generate a grade for the specified quiz
	GetGradeFromLab()	Returns grade from the specified lab for the student class
StudentGUI	processCommand()	Processes any GUI commands given (generic name for future implementation)
InstructorGUI	processCommand()	Processes any GUI commands given (generic name for future implementation)
CheatDetector	retrieveData()	Retrieves camera footage
	verifyPerson()	Uses facial recognition points to determine if student is cheating
Quiz	giveQuiz()	Administers quiz to student

	recordResults()	Records results of quiz
	checkAnswers()	Checks the users answers with the recorded actual answers.
Database	authenticatePerson	Connects to the online database and authenticates the entered Username and Password with the data in the database.
LoginGUI	checkType	Checks the user's type
	launchGUI	Launches the respective GUI for the student or the instructor
	authenticate()	Calls on the database to authenticate the entered username and password
Student	resetGrade	Method to reset the specified lab grade from the Instructor
Instructor	getGrade	Method to get a list of students' grade
Chat	enterMessage	Allows user to enter a custom message to the chat

5.1.4 Traceability Matrix

	Domain Concepts																
Use Case	PW	LoginGUI	Chat	Person	Lab1	Lab2	Lab3	EquipmentManager	Report	StudentGUI	InstructorGUI	CheatDetector	Database	Quiz1	Quiz2	Quiz3	Student	Instructor
UC-1	55							X										
UC-2	17									X								
UC-3	3	X		X									X				X	X
UC-4	4												X					
UC-5	6										X							X
UC-6	5										X							
UC-7	2							X		X								
UC-8	2									X								
UC-9	2									X								
UC-10	1									X	X							
UC-11	1		X															
UC-12	8								X									
UC-13	2											X						
UC-14	27													X	X	X		
UC-15	25				X													
UC-16	19					X												
UC-17	15						X											

5.2 System Operation Contracts

Name: manageEquipment(partType, partLocation)
Responsibility: Control the movement of the pieces by responding to the user's click/drag
Cross-References: UC-1
Output: The system can introduce new object to the board and have them interact with each other
Pre-Conditions: Student clicks on and drags a piece to place on the protoboard
Post-Conditions:

- A new Equipment object is created and given the ability to interact with the protoboard
- Different parts interact with each other as they are moved around the board
- Location values for each object constantly change as they are moved around

Name: check();
Responsibility: To check to see if the solution that a student has submitted is correct
Cross-References: UC-2
Output: The system will either output a message saying that the user is correct, or display a grade and a description of the errors that were made
Pre-Conditions: The student finishes a task and presses the finish button
Post-Conditions:

- The student's answers are assigned to variables and compared to the values of the correct answers to test for equivalence
- Variable grade is assigned a value based on percentage of correct responses
- Test if grade equals 100 (perfect score), and will display congratulations message accordingly

Name: grade(labScore);
Responsibility: Receives the scores from each lab that the user completes and copies the grade into the gradebook for both students and teachers to view
Cross-References: UC-5

Output: The output to this is the grade that the student received, and it is available to both students and teachers
Pre-Conditions: The student has finished a lab and submitted his/her work for grading
Post-Conditions:

- Each student object has their grade variable assigned to them
- Grade becomes viewable by students and instructors

Name: quiz()
Responsibility: Give the student s a quiz upon the completion of each lab
Cross-References: UC-14
Output: A quiz is given to the user, which they must complete
Pre-Conditions: The user completes and submits a lab
Post-Conditions:

- The user is presented with a quiz, typically the image of a Karnaugh Map or Truth table that must be solved
- Empty spaces must be filled with values, which will be compared to the correct values after submission

Name: lab1()
Responsibility: To allow the user to perform tasks given to them in lab 1
Cross-References: UC-15
Output: Brings the user to a screen where they are able to complete lab 1
Pre-Conditions: The user must select “Lab 1” from the labs menu
Post-Conditions:

- The user is given the ability to complete lab 1
- System displays image of a truth table, which user must fill out correctly
- There is no circuit design for lab 1
- quiz() is called to give the user a quiz afterwords

Name: lab2()
Responsibility: To allow the user to perform tasks given to them in lab 2
Cross-References: UC-16
Output: Brings the user to a screen where they are able to complete lab 2
Pre-Conditions: The user must select “Lab 2” from the labs menu
Post-Conditions:

- The user is given the ability to use lab 2

- manageEquipment(partType, partLocation) is called often as the user moves pieces around the board
- quiz() is called to give the user a quiz afterwords

Name: lab3()
Responsibility: To allow the user to perform tasks given to them in lab 3
Cross-References: UC-17
Output: Brings the user to a screen where they are able to complete lab 1
Pre-Conditions: The user must select “Lab 3” from the labs menu
Post-Conditions:

The user is given the ability to use lab 2
manageEquipment(partType, partLocation) is called often as the user moves pieces around the board
quiz() is called to give the user a quiz afterwords

6. Plan of Work

The main infrastructure that will facilitate this project is the Unity game engine which is capable of easily rendering 2-Dimensional graphics in a developer friendly manner. The main programming language of choice will be C# with possible integration of others on smaller details. We also plan on using an open source and free database such as LiteDB or RavenDB to upload our grading reports, user data, and chat service.

The most important part of this project is to implement the individual labs to facilitate student learning. To do this, a smaller subset of our group members will implement the “game” logic and create assets needed to create the laboratory experience. This is likely to be the greater of the technical challenge. However, after the initial hurdle of designing a base game logic, creating the actual labs should be fairly straight forward as all the labs use the same concepts of putting together digital circuits.

The second part is to implement a graphical user interface that the administrators and students will interact with to create new users, login, take labs, chat, and view grades in. Parts of this need focus from individual team members but will be implemented by another small subgroup of team members. This part requires the additional specialization of database management which some of our team members possess.

Our team has a diverse set of skill sets and the willingness to learn new technology to tackle this issue. We all have experience in some variation of object oriented programming which is used to tackle majority of software related issues seen today. All of the members have some sort of experience in Java, C++, and Python based on past experiences. To unite all of these together, we found that C# provides a combination of all threes syntax, and underlying programming philosophies. Together, we all possess the required skill to tackle this issue, with one challenge: being able to communicate and work productively through the product development cycle.

7. References

Software Engineering Guide by Ivan Marsic
http://www.ece.rutgers.edu/~marsic/Teaching/SE/

DLD Lab Manual
http://www.ece.rutgers.edu/~marsic/Teaching/DLD/lab.html

Domain Analysis
http://www.site.uottawa.ca/~laganier/seg2500/domain

Previous project
http://www.ece.rutgers.edu/~marsic/books/SE/projects/ViBE/2012-g5-report3.pdf