Power Rack Manager

ECE 445 Design Document

Derek Niess, John Quinn and Bartu Alp
Group 47
TA: Madison Hedlund

2/27/20

1. Introduction

1.1 Problem and Solution Overview

Power racks are a crucial component to a person's workout as you are able to partake in a multitude of weightlifting exercises. Every gym consists of usually multiple power racks that come equipped with a bench, a barbell and weights. Some of the exercises that you can accomplish with a power rack include squatting, benching, deadlifts, and cleans. Although these are the most common exercises that you are able to do utilizing a power rack, many more exist as well. Oftentimes when people show up at the gym, most, if not all, of the power racks are in use due to their highly versatile nature. People will then have to wait an unknown amount of time for the people that are using them to complete their exercises. People also often do not know whether or not there is a line to wait in to use a particular power rack because they are off doing other exercises. College students suffer through this problem the most as college gyms are often packed with people and lines form quickly to use the power racks. Although this problem has a significant effect on college students and their busy lives, it is not unique to them. Users of any gym have no doubt run into this issue at one time or another.

Our solution is a power rack managing device that consists of a system in which you are able to put your name in a queue to use a particular power rack. A LCD display screen as well as a mobile application will display the current user of each power rack as well as the next three people in line waiting to use the power rack. It will also display a color to indicate whether the power rack is being used or not. A red color will indicate that the power rack is currently in use while a green color will indicate that no one is currently using the power rack. There will be a button attached to the power rack that a user will press when they begin using the power rack or when they have completed their exercise(s) on that particular power rack. When the button is pressed, signifying that a user has finished with a power rack, the queue will update on the LCD screen as well as the app. An app notification will also be sent whenever the queue changes for the particular power rack that you are waiting for. There will also be an ultrasonic motion sensor that will sense whenever no one is using the power rack and if no one has been using it for two minutes, it will update the queue accordingly. This is to safeguard against people forgetting to press the button that signifies that they are no longer using the power rack.

For most weightlifters and people who exercise in general, power racks are critical for a successful workout. The versatility that they offer is unmatched by any other weightlifting equipment. Because power racks allow users to perform a wide variety of exercises, they are in extremely high demand. The issue is that gyms only have a limited number of power racks and during busier hours, lines form with people waiting to use the power racks. Sometimes, people have to wait for an absurd amount of time to be able to use a power rack. Personally, we have all had to wait 20 minutes or longer to be able to use a power rack. Waiting an extended period of time can disrupt the routine and rhythm of a person’s workout and negatively affect the overall workout. Also, when longer lines form, confusion ensues regarding who is next in line to use a power rack. This occurs because rarely does someone sit around and wait to use a power rack. People usually tell the person(s) currently using the power rack that they are next in line before moving on to do another exercise while waiting their turn. This causes confusion on who is next in line. The Power Rack Manager can solve all of these issues by showing the queue for each power rack on a mobile application as well as an LCD display. It can also allow you to add your name to the queue and signify when you are done with the power rack while updating the queue accordingly.

1.2 Visual Aid

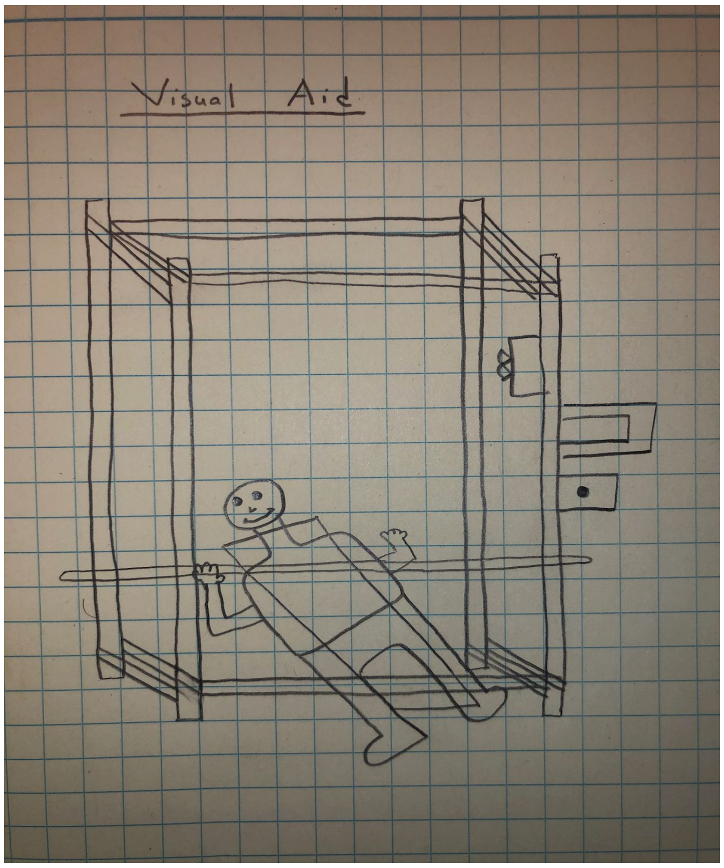

Figure 1. Power Rack Manager Visual Aid

The above picture depicts the image of a user who is currently doing the bench press movement. The sensors mounted on the right hand side detects the movement and classifies this rack as occupied. The LCD display shows the name of the current user as well as put a red light to accentuate that it is occupied. Below the LCD display there is the user button which is already pressed, notifying the system that the rack is in use.

1.3 High-Level Requirements List

- The mobile application must display the current queue list of every weight rack in the user's gym and the display must be refreshed within 5 seconds in the event of an update to any queue list.
- Once an occupied weight rack becomes vacant, the queue list of that weight rack should update accordingly (if there is a line), whether the occupancy button is pressed or once the ultrasonic motion sensor detects no movement within a 2 minute interval.
- The mobile application must notify the users waiting in the queue within 15 seconds when their chosen rack becomes available for them.

2. Design

2.1 Block Diagram

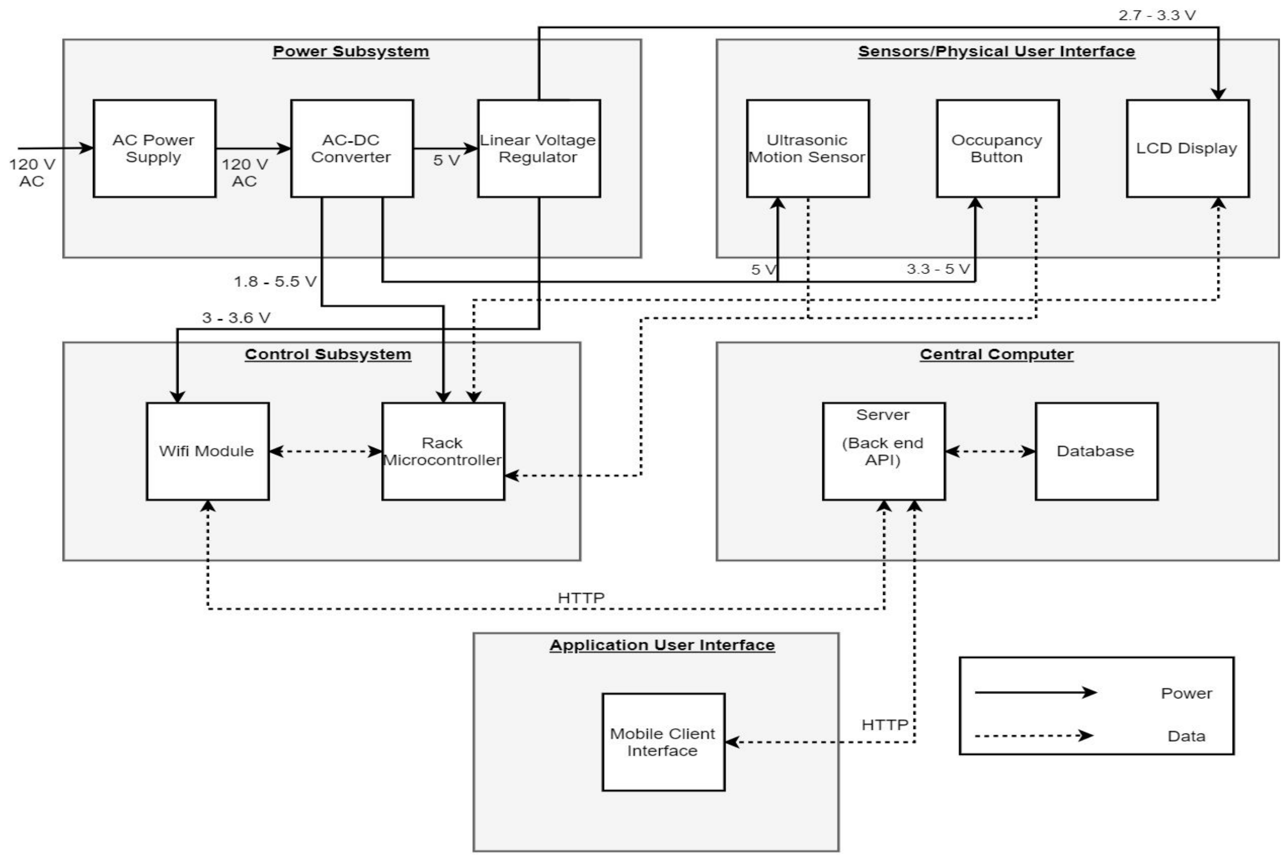

Figure 2. Power Rack Manager Block Diagram

The overall block diagram for the Power Rack Manager is shown in figure 2. The design consists of five subsystems. The power subsystem is responsible for providing adequate power to the components within the sensors/physical user interface subsystem and the control subsystem. The sensors/physical user interface subsystem is responsible for gathering data to be sent to the control subsystem for processing as well as receiving data from the rack microcontroller to be displayed on the LCD display. The central computer interface is responsible for handling and storing the processed data from the control subsystem and transmitting this data to the mobile client interface within the application user interface subsystem. The application user interface subsystem is responsible for receiving and displaying the correct data on user's mobile devices as well as communicating the displayed information to the server within the central computer subsystem.

2.2 Physical Design

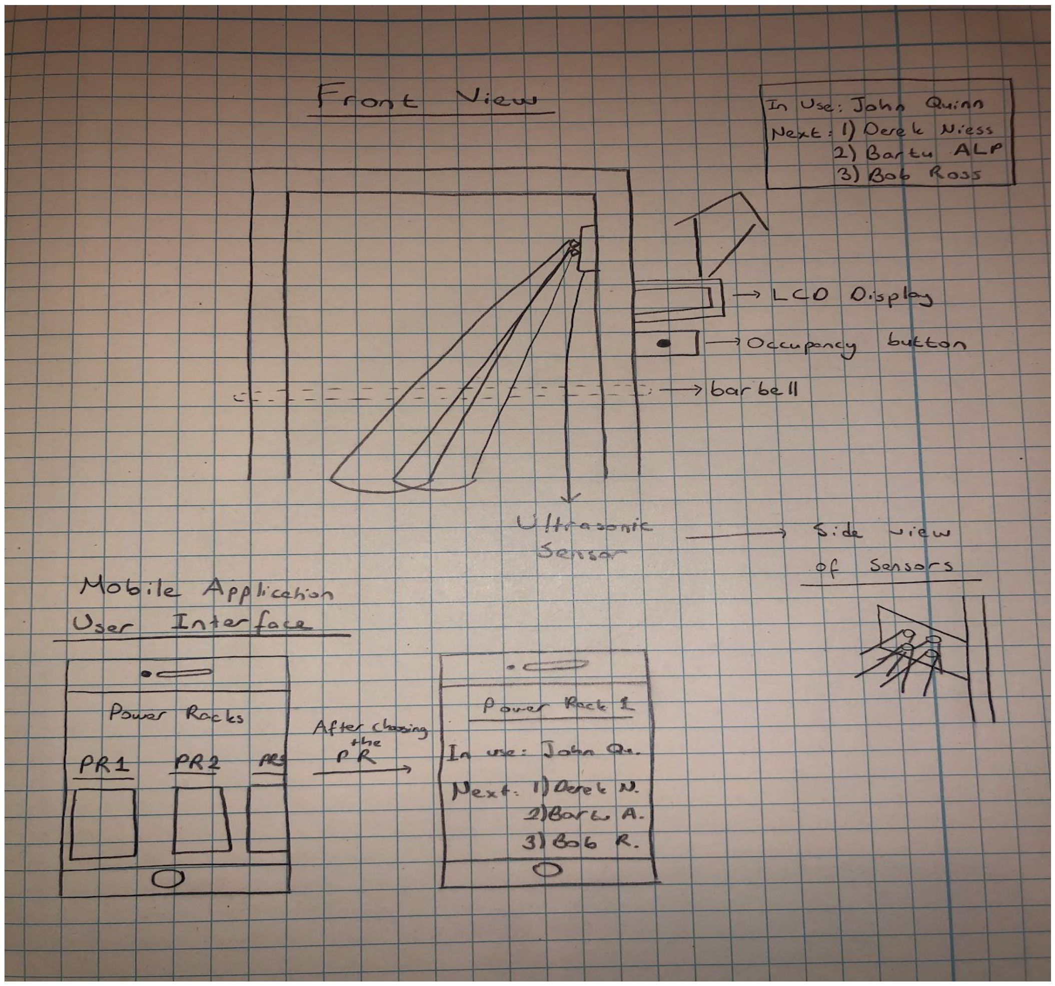

Figure 3. Power Rack Manager Physical Design

In Figure 3, we can see the detailed description of our physical design. All the components are named as well as a sample of the user interface is drawn. The four sensor subsystem can be seen from both the front and the side views.

2.3 Subsystems

2.3.1 AC/DC Converter

Since many components of the Power Rack Manager require a DC power supply, we will require an AC/DC converter to convert the standard wall outlet 120 V AC to a suitable DC level for operation. The reason that we chose a 120 V AC power supply is so that each Power Rack Manager will have a reliable power supply and will not have to depend on charging and using a battery. This AC/DC converter must be able to sustain an adequate amount of DC current to successfully operate all the required components as well. The AC/DC converter that we will be using is the IRM-15-5 from Mean Well USA Inc. This particular AC/DC converter can take in a range 85 to 264 V AC and produces a 5 V DC voltage and is able to supply up to 3 A of DC current. The input range is more than adequate to handle the 120 V AC supply drawn from a standard wall outlet. We chose an AC/DC converter that is able to produce 5 V DC because three of our components have a suitable operating voltage of 5 V DC [6]. The ultrasonic sensors require a 5 V DC supply voltage and require an operating current of 15 mA each. The occupancy button requires a supply voltage of 3.3 V to 5 V DC as well. In addition, the microcontroller that we will be using has an operating voltage of 1.8 V to 5.5 V DC and requires an operating current of 9 mA at a supply voltage of 5 V. These three components will be wired directly to the 5 V DC output of the AC/DC converter and will use this voltage as their supply. The maximum current that we require to operate our components is 269 mA given that the LCD display requires 120 mA and the wifi module requires an average of 80 mA. So the current rating of 3 A for this AC/DC converter is more than suitable to meet all of our power needs for the Power Rack Manager. Since the LCD display that we will be using requires an operating voltage of 2.7 to 3.3 V DC, the 5 V output of the AC/DC converter will also be wired to a voltage regulator that will drop this voltage within the 2.7 to 3.3 V range for optimal operation of the LCD display. The wifi module also requires an operating voltage less than the 5 V generated by the converter as it operates within a voltage range of 3 to 3.6 V. This wifi module will also be wired to the same voltage regulator as the LCD display.

Table 1. RV Table for AC/DC Converter

Requirements	Verification
1. The AC/DC converter must be able to take in an AC supply voltage of approximately 120 V and convert this to a DC range of 4.5 - 5 V for powering of necessary components. 2. The AC/DC converter must be able to sustain a maximum DC current of 200 mA 3. The AC/DC converter must be	1. A. Connect the input of the AC/DC converter to a standard 120 V AC wall outlet. B. Connect the output of the AC/DC converter to an electronic load with a fixed current of 200 mA. C. Probe the voltage across the electronic load and ensure that the voltage across it is between

able to operate at a temperature of under 45 ℃ to ensure thermal safety.	4.5 and 5 V. 2. Same procedure as steps 1.A-C. 3. During steps 1.A-C use an IR thermometer and ensure the operating temperature of the converter is less than 45 ℃.

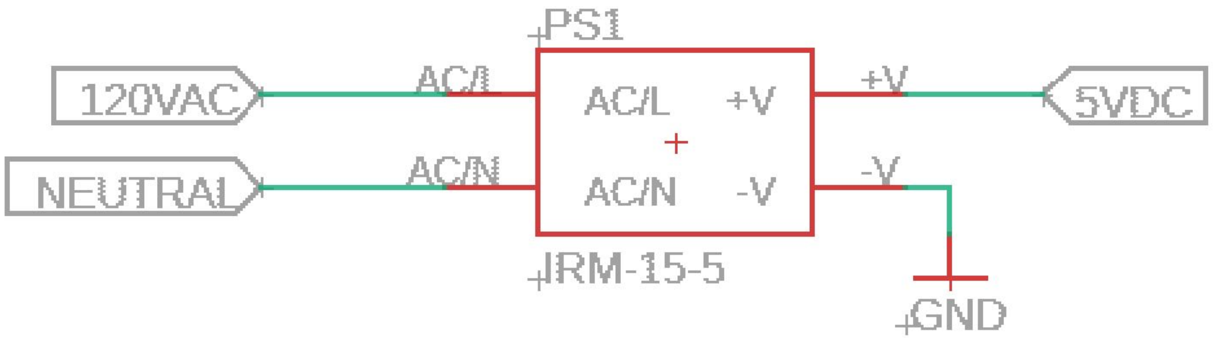

Figure 4. AC/DC Converter Schematic

2.3.2 Voltage Regulator

Because the LCD display and the wifi module of the Power Rack Manager require a lower operating voltage than the 5 V DC generated by the AC/DC converter, we will require a voltage regulator to adequately supply these components. The LCD display requires an operating voltage between 2.7 V and 3.3 V while the wifi module requires an operating voltage between 3.0 V and 3.6 V. We will use a voltage regulator that is able to adequately supply both of these components. The input of the voltage regulator will be 5 V DC from the AC/DC converter and the output will be in the range of 3.0 - 3.3 V DC and will be fed directly to the LCD display and wifi module to adequately power them. We will be using the ADP123 voltage regulator from Analog Devices Inc. This particular voltage regulator is able to take in an input voltage of 2.3 V to 5.5 V DC and produces a fixed output voltage that depends on the resistors attached to the output [7]. The output voltage is defined by Vout = 0.5V(1 + R1/R2) [7]. To obtain an output voltage of 3 V we will use a value of 250 Ω for R1 and 50 Ω for R2. This input voltage range is perfect as the AC/DC converter produces a voltage of 5 V and this output will be the input of the ADP123 voltage regulator. The output voltage of the regulator is within the operating range of 3.0 V to 3.3 V for both the LCD display and wifi module and this will be sufficient to power both components. The maximum current rating of the ADP123 is 300 mA and therefore will be able to accommodate the 120 mA current required by the LCD display and the average of 80 mA required by the wifi module.

Table 2. RV Table for Voltage Regulator

Requirements	Verification
1. For an input voltage of 4.5 V to 5 V, output a voltage between 3.0 V and 3.3 V at a maximum current of 220 mA. 2. Ensure that the operating temperature remains under 45 ℃.	1. A. Connect a DC power supply set between 4.5 and 5 V to the input of the voltage regulator. B. Connect an electronic load to the output of the voltage regulator with a fixed current setting of 220 mA. C. Meter the output of the voltage regulator and ensure that this output is between 3.0 V and 3.3 V. 2. During steps 1.A-C use an IR thermometer to make sure that the voltage regulator's operating temperature remains under 45 ℃.

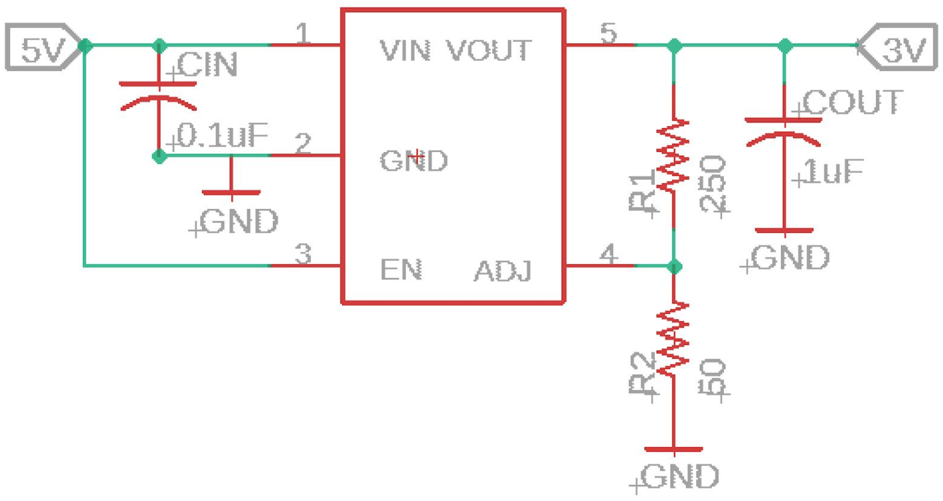

Figure 5. Voltage Regulator Schematic

2.3.3 Ultrasonic Motion Sensor

A great deal of responsibility is placed on the ultrasonic motion sensor in order for the Power Rack Manager to operatore successfully. The ultrasonic motion sensor is responsible for detecting whether or not there is movement within the vicinity of each power rack in order to determine whether it is in use. The ultrasonic motion sensor will be powered directly from the AC/DC converter as the voltage generated falls within the ideal operating range of the sensor. The Ultrasonic sensor will also transfer data to the microcontroller for further processing in determining whether a specific power rack is occupied. Because motion detection must be performed on a continuous basis, it must be able to transmit data to the rack microcontroller at a quick rate for processing. The ultrasonic motion sensor works by transmitting ultrasonic sound waves (high frequency) and, when this sound wave is reflected off of an object, a receiver determines if there are changes in the ultrasonic sounds waves in determining if there is motion. The ultrasonic sensor is able to detect the distance of an object and, when this distance changes (speed at which the ultrasonic sound wave is received), motion is detected and the appropriate signal is set which will be received by the microcontroller [3]. We will require four ultrasonic sensors to provide full coverage of the power rack area with no blind spots. The ultrasonic sensor that we will be using is from Adafruit Industries LLC and the part number is 3942. This ultrasonic sensor both transmits ultrasonic sound and receives the reflected ultrasonic waves in determining if there is motion within the vicinity. This particular sensor requires a DC power and logic supply of 5 V and draws a current of 15 mA during measurement. The output of the sensor is an analog logic voltage that ranges from 0 V to 5 V [8]. This output will be fed directly to the ATmega328 microcontroller to one of its analog I/O pins for further analysis in determining if motion was detected. We chose this particular sensor because it is able to detect motion up to 4 meters away and has a measurement angle of 15 degrees [8]. These measurements allow us to minimize the number of required ultrasonic sensors per power rack as we will only require four sensors.

Table 3. RV Table for Ultrasonic Sensor

Requirements	Verification
1. The ultrasonic sensors must be able to transfer analog data to an I/O pin of the microcontroller at a rate of 2 seconds or under so that the microcontroller can process this data quickly enough to determine if motion is detected. 2. The ultrasonic sensors must be able to detect change in distances of a minimum of 5 inches in order to be able to successfully detect small movements. 3. The ultrasonic sensors must be able to detect motion up to a maximum distance of 3 meters. 4. The ultrasonic sensors must be able to operate at temperatures of under 45 ℃ to maintain thermal stability.	1. A. Power an ultrasonic sensor with a DC supply of 5 V for both the logic and power supply. B. Connect the analog output to an I/O pin of any microcontroller, preferably the arduino uno. C. Measure a distance of 3 meters away from the sensor and move approximately 5 inches towards the sensor. D. See if the analog logic signal is received by the microcontroller in under two seconds and that it is able to detect this small movement. 2. Same procedure as steps 1.A-D. 3. Same procedure as steps 1.A-D. 4. During steps 1.A-D use an IR thermometer to ensure that the temperature of the sensor remains under 45 ℃.

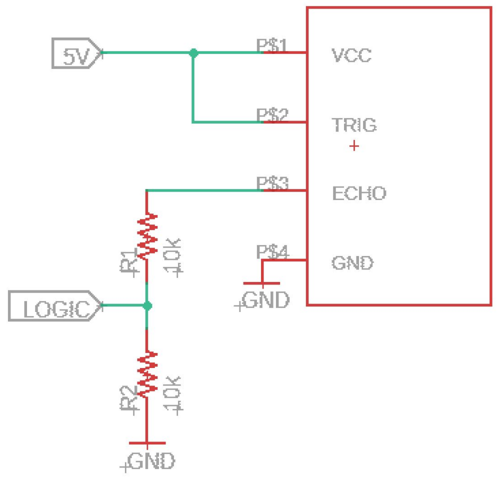

Figure 6. Ultrasonic Sensor Schematic

2.3.4 Occupancy Button

The primary method of occupancy detection will be a button attached to each power rack that each user will press when they begin using a particular power rack and will also press when they have completed their usage of a power rack. This occupancy button will be powered from the power subsystem and, more specifically, from the voltage regulator as the occupancy button requires a DC supply voltage that is lower than the generator voltage from the AC/DC power converter. More specifically, the push button will require a supply voltage of 3.3 to 5 V DC [9]. We will be using the KS0029 digital push button as our occupancy button. The occupancy button will also be in communication with the rack microcontroller to send data for processing. This push button has a digital interface (SPI) that determines whether the button has been pressed or not [9]. We chose this particular button because it is compatible with the arduino uno and we will be using the ATmega328 microcontroller which is the microcontroller that the arduino uno uses. Data will be received by the microcontroller through the SPI interface from the occupancy button to signify when the button is pressed in order to check whether or not the power rack is in use at that specific time. The occupancy button will work together with the ultrasonic motion sensor to accurately determine the usage of each power rack.

Table 4. RV Table for Occupancy Button

Requirements	Verification
1. The occupancy button must be	1. A. Power the occupancy button

able to transfer data through SPI protocol at a rate of 2 seconds or under so that the microcontroller can process this data and determine if movement is detected. 2. The occupancy button needs to operate at under 45 ℃ for safe thermal operation	using a DC voltage between 3.3 V and 5 V B. Connect the SPI digital output to any microcontroller, arduino uno preferred C. Press the occupancy button and see if the digital signal is received by the microcontroller in 2 seconds or under. 2. A. In steps 1.A-C use an IR thermometer to ensure that the operating temperature remains under 45 ℃.

2.3.5 LCD Display

The input of the LCD display is the rack microcontroller. According to the signal it receives from the rack microcontroller, it will either output a green or a red color, green implying vacancy while red implying occupied. In addition to this, the LCD display will also show its users, the number of people waiting in the queue for that specific rack so that people in the gym can plan ahead without the need of checking the app. Right below the number of people waiting for that specific rack, the LCD display will also show the names of the top three people waiting in the queue. So, the LCD display will have a color display, a number display and also provide the names of the top three people who are about to use the rack. The LCD display gives a physical display of the queue as well as letting people know if the current power rack is in use. These will be mounted on each power rack and easily visible to the user. The LCD display will be powered by the power subsystem from the voltage regulator as the DC voltage generated by the AC/DC generator is too high to adequately power the LCD display. The LCD display will have communication with the rack microcontroller so that the LCD display is able to receive the correct data. We will be using the EA DOGXL240N-7 LCD display due to its size and digital interface versatility. Users must be able to clearly see the LCD display from approximately three feet away. This particular LCD display is 94 by 67 mm and based on these dimensions, will be clearly visible from three feet away [10]. Ideally, we would have chosen a larger LCD display so that it would be visible from distances greater than three feet, but larger displays are typically expensive at greater than $200. This particular LCD display has three wire and four wire SPI digital interfaces as well as an I²C digital interface [10]. We will be using the three wire SPI digital interface in this case to receive display data from the ATmega328 microcontroller. This particular LCD display requires a DC supply voltage of 2.7 V to 3.3 V while requiring a supply current of up to 120 mA [10]. This voltage range will be provided by a voltage regulator to adequately power the LCD display.

Table 5. RV Table for LCD Display

Requirements	Verification
1. The LCD display must have a refresh rate of 3 seconds or quicker to update both the vacant/occupied display as well as updating the names waiting in the queue. 2. The names in the queue must be clearly visible and readable from a maximum of 3 feet away. 3. The LCD display must operate at a temperature of under 45 ℃ for thermal safety.	1. A. Connect the LCD display to a DC supply voltage between 2.7 V and 3.3 V. B. Connect the LCD display using the 3 wire SPI interface to any microcontroller, preferably an arduino uno. C. Transmit data from the microcontroller to be displayed on the LCD display and ensure that this data is correctly displayed within 3 seconds. 2. During step 1.C, stand 3 feet away from the LCD display and ensure that you can clearly read what is being displayed. 3. When steps 1.A-C are being completed, use an IR thermometer to measure the operating temperature and make sure that it is under 45℃.

2.3.6 Rack Microcontroller

The MCU will act as the primary source for handling communication. It will read raw sensor data from the button and ultrasonic sensor via manually assigned IO pins, write to the LED display via SPI, and make requests and receive responses from the remote server housed by a central computer. The microcontroller will also be powered from the power subsystem and the AC/DC converter as the voltage generated by the AC/DC converter is suitable for microcontroller operation. The microcontroller that we will be using is the ATmega328 microcontroller. This microcontroller requires an operating voltage between 1.8 V and 5.5 V and draws a current of 9 mA at 15 MHz operation and a supply voltage of 5 V [11]. The microcontroller will be wired directly to the 5 V output of the AC/DC converter for its power supply. The microcontroller will be handling quite a bit of data as it interfaces with the occupancy button, the two ultrasonic sensors, the LCD display and the wifi module. The reason why we chose this particular microcontroller is because it is sufficiently fast enough to meet all of our requirements as we will be operating it at 15 MHz and it has a multitude of communication protocols to interface with all of the necessary components. The microcontroller will receive data from with the occupancy button through SPI protocol and will also receive data from the ultrasonic sensors through an analog I/O port. It will send data to the LCD through an SPI interface.

Table 6. RV Table for Rack Microcontroller

Requirements	Verification
1. The microcontroller must be able to read and write via SPI at speeds more than 4.5 Mbps 2. The microcontroller must be able to read data from the I/O pin of the ultrasonic sensors	1. a. To test SPI speeds, first connect the microcontroller to PuTTY terminal b. Connecting the microcontroller to FT4222, a USB-SPI bridge c. Complete the setup of the terminal to 4.5Mbaud d. Send 100 characters e. Echo them back f. Verify that they match 2. a. To test I/O reads, first write a simple script that will send a pulse via the input pin to the sensor for projecting an ultrasonic frequency and poll the output pin for a distance measurement. The script should write

	this value to an empty file on flash memory. b. Connect the microcontroller to FT4222, a USB-UART bridge. c. Connect the microcontroller to PuTTY terminal d. Open the file mentioned in 2.a via the cat command. If a number is displayed in the terminal, then the read was successful.

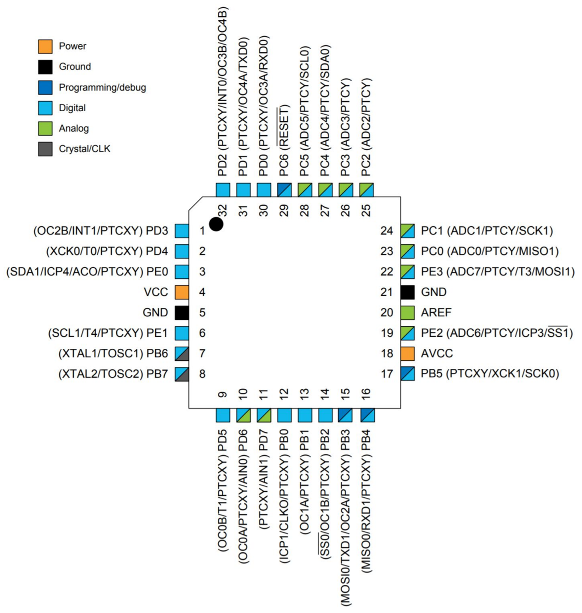

Figure 7. Microcontroller Pinout

2.3.7 Wifi Module

In order for the MCU (microcontroller) to communicate with the mobile user application and server, it needs a wireless connection to transfer data through. The wifi module allows that very connection to be made. As such, its function is entirely dependent upon the function of the MCU and should be critiqued with respect to the MCU. The wifi module that we will be using is the ESP8266 due to its compatible nature with the arduino microcontroller

(ATmega328). This wifi module requires an operating voltage of 3 V to 3.3 V and an average operating current of 80 mA [12]. The input voltage will be wired to the output of the voltage regulator since the voltage regulator is equipped to supply 3 V as well as an adequate current to this component.

Table 7. RV Table for Wifi Module

Requirements	Verification
1. When the microcontroller executes an API call, the server must receive the same HTTP request within 5 seconds.	1. API calls made in the application running on the MCU will be preceded with a universal time stamp and logged on a console file. The universal time will be logged on the server upon any API call being received. Subtract the timestamp on MCU console file from the timestamp on server to get the time taken to send and receive the HTTP request.

2.3.8 Server/Database/Mobile App

The backend of our application is composed of a server and a database, where the server will be actively polling for incoming API requests and the database will house the current queue list of each registered power rack. Once the server receives an API request, the function(s) associated with that request are executed, and the server will then either query or modify the database according to that request. The frontend is a mobile application generated using SwiftUI that will display the current data on the database for a particular gym.

Table 8. RV Table for Server/Database/Mobile App

Requirements	Verification
1. The server must be able to execute NoSQL statements that update or query the database with a successful return value. 2. The mobile application must	1. a. Using the terminal command line interface for the NoSQL database, insert a temporary table and populate it with fake data.

display the updated queue list data of a single weight rack within 10 seconds of the list being updated in the database.	b. Write a simple HTTP GET method in the backend application that, when called, connects to the database and executes a NoSQL statement querying the temporary table and returns a response object with all of the documents in the temporary table. c. Run the server using the runserver command. d. Use Postman to make the same GET request in 1.b and note the response object. If the object contains the data in the temporary table, the server is properly connected to the database. 2. a. Write a method in the SwiftUI that polls the server for updates to the temporary table mentioned in 1.a and displays the data of the response it receives. The timestamp of each response will also be displayed. b. Add code in the backend application for a POST method that executes an UPDATE statement that increments the values in the temporary table. It should record the timestamp of a successful update to the database and return this

	timestamp as the response object. c. Run the server using the runserver command. d. Use Postman to make a POST request that will update the documents in the temporary table by incrementing the values. e. Check the phone running the SwiftUI to see what the two timestamps displayed are. If they are within 10 seconds of each other, the app refresh time is sufficient.

2.4 Tolerance Analysis

The reliability of our design depends primarily upon the accuracy of the ultrasonic sensors. More specifically, the fusion of sensor data must sufficiently explain the state of the system's environment. We must start by describing what the environment is and the assumptions of this environment. The environment our system is designed to function properly in is any section of a gym with barbell power cages or power racks. We are not considering smith machines. That said, our design would likely function properly if applied to a smith machine. Given this detail, we note that an olympic barbell, the most popular type of barbell, is 52 inches between sleeves, and we can assume that racks compatible with this barbell are anywhere from 48 inches to 50 inches in width, 40 inches to 50 inches in depth, and always at least 80 inches in height [4]. Our last assumption is that a user is deemed to be using a rack if they are within a radius of half of the width of the rack centered at the midpoint of where the barbell is mounted. We reason that this is practical because the user will be performing reps close to where the rack arms are, and the user will be re-racking the barbell after each set anyway.

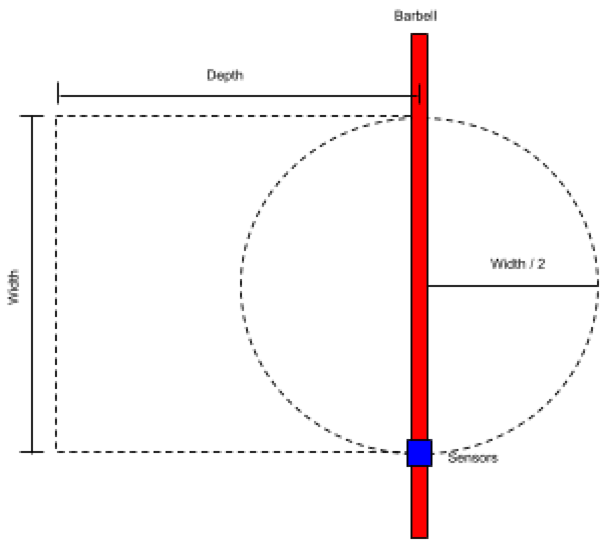

Figure 8: The dotted circle indicates the necessary area to detect movement within relative to the rack to determine occupancy.

In order to determine how many ultrasonic sensors to use to cover the circular area shown in figure 8, we must calculate the effective surface area measureable from a single ultrasonic sensor. We start with the assumption the sensor packet will be at a height of 80 inches, which any rack should be able to accommodate. If we ignore the added surface area caused by generating a conic beam at an angle and instead assume that the surface area forms a perfect circle--and we can make this assumption to no negative effect because an angled conic beam is always larger than a perfectly perpendicular beam--then we can calculate the radius of the coverage of a single ultrasonic sensor using the following equation:

$$r = h\,tan(\theta_{beam}) \tag{1}$$

Using equation 1, we input a height h of 80 inches and a beam angle of 15 degrees, as mentioned in the ultrasonic sensor datasheet, to find that a single ultrasonic sensor generates a minimum effective coverage with a radius of 21.44 inches [8].

With the knowledge of how much area a single ultrasonic sensor can cover, we can now determine how many ultrasonic sensors are needed to detect movement within the above radius around the barbell. This problem can be reduced and compared to a set of special covering problems that find the largest coverable square given an input number of unit circles [14]. To do so, we must first find the precise factor needed to relate the side of the covered square to the radius of the circle to account for the fact that the covering problems we will be comparing our own problem to are normalized:

$$s = f * r \tag{2}$$

Plugging in a radius of 21.44 inches and side length of 50 inches, we get a factor of 2.33. With this known, we can compare it to the following chart to determine which covering problem, defined by the input number of unit circles, will solve our own problem:

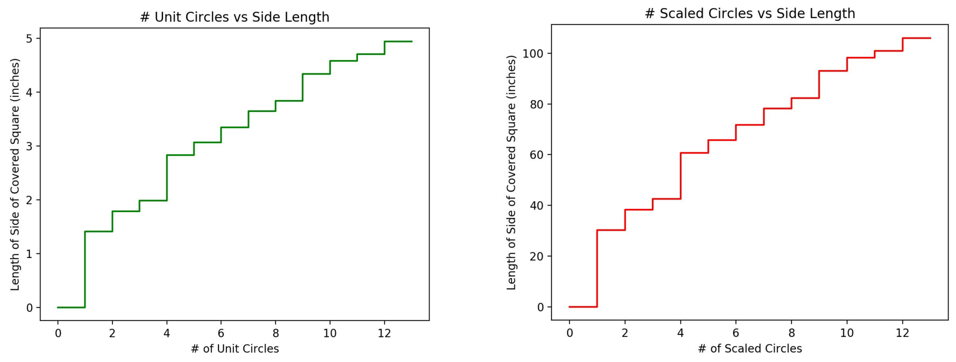

Figure 9: The plot on the left presents the relationship between the number of unit circles and the maximal side length of a square coverable by those circles. The plot on the right presents the same relationship, but in the case where the radius of the circle is not 1 inch, but 21.44 inches. Note that the two plots are step functions that are right continuous.

As one can see, we need a minimum of 4 circles to cover our target square. Given this knowledge and upon further investigation into the solution to the covering problem where the input is 4 circles, we can determine the coordinate distance of the where the center of the beam should be aimed at on the ground relative to the horizontal position of the sensor packet:

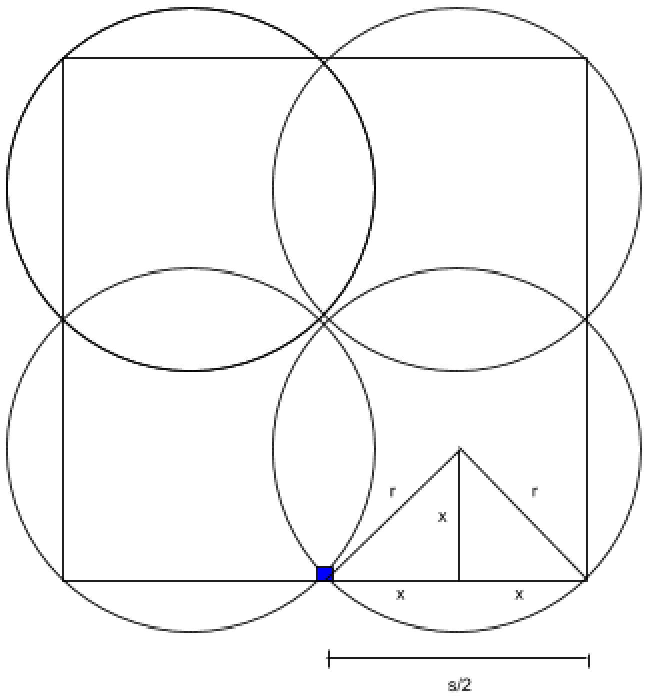

Figure 10: The square is just large enough to inscribe within it the critical circular area in which movement must be detected, so if the square is covered, then the critical circular area is also covered. Each circle has a radius of r inches. Each circle overlaps a corner of the square, and all of them join at the center of the square. The blue dot symbolizes where the sensor packet is located.

To find the (x, y) coordinates of the center of each circle, we must solve for x in figure 3. Using the pythagorean theorem, find that x is 15.16 inches. We can then plug this value into the below equations to get the coordinates:

$$(x,\ y)_{top\ left} = (-x,\ s-x) \qquad (x,\ y)_{top\ right} = (x,\ s-x) \tag{3}$$
$$(x,\ y)_{bottom\ left} = (-x,\ x) \qquad (x,\ y)_{bottom\ right} = (x,\ x)$$

The last step is to determine the angle to point each sensor at along the horizontal and vertical plane. For the horizontal plane, we calculate θ_H such that 0 degrees is perpendicular with the surface holding the sensors:

$$\theta_{H,\ bottom\ left} = arctan(x\,/-x) \qquad \theta_{H,\ bottom\ right} = -arctan(x\,/\,x) \tag{4}$$
$$\theta_{H,\ top\ left} = arctan((s-x)\,/-x) \qquad \theta_{H,\ top\ right} = -arctan((s-x)\,/\,x)$$

We determine θ_V for the vertical plane such that 0 degrees is horizontal with the ground. Since the height of the sensors will be 80 inches, we can calculate the vertical angles using the following equations:

$$\theta_{V,\,bottom\ left} = arctan(x / 80) - 90 \tag{5}$$
$$\theta_{V,\,bottom\ right} = arctan(x / 80) - 90$$
$$\theta_{V,\,top\ left} = arctan((s - x) / 80) - 90$$
$$\theta_{V,\,top\ right} = arctan((s - x) / 80) - 90$$

The two issues to be addressed in our analysis are:

1) Ensure that the inaccuracy of the sensors do not throw off the decision that movement is detected.
2) Ensure that the inaccuracy of the sensors do not throw off the decision that movement is detected.Ensure that the sensors cover the critical area of the power rack.

Issue one is trivial. Since we do not care about the distance measurement itself but the detection of any small change in this distance measurement, we can account for the inaccuracy of +/- 3mm of each sensor by filtering out any change in distance less than 6mm [8]. Regardless of the exercise, lifting a barbell requires a movement that far exceeds 6mm. Taking the average female upper arm length of 13.62 inches, or 345.9 mm, as a reference for what can reasonably be considered the minimal displacement an exercise would likely require (leg barbell exercises almost always require more movement than upper body exercises), the sensors should theoretically detect movement 57 times over.

For issue two, we can see that the problem boils down to properly calculating the orientation of the sensors and physically placing those sensors such that this angle is met. Let's go back to how we solved this when we reduced our problem to the previously mentioned covering problem with 4 circles. If our sensors are perfectly oriented in the way we calculated it, we'd have a square coverage with a side length of 2.828 * 21.44 = 60.63 inches. This creates a 60.63 - 50 = 10.63 inch buffer in the event the sensors were not properly oriented.

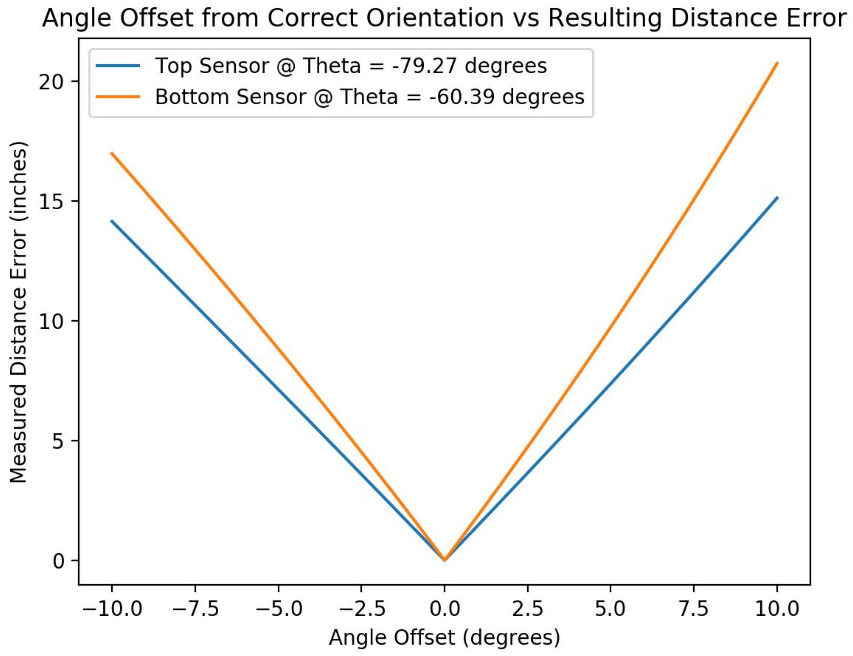

Figure 11: This is a plot showing the effect of an applied angle offset when physically attempting to orient a sensor correctly based on whether the sensor is meant to point at the center of a top circle or bottom circle in the previously mentioned covering problem. Note that an error from the theoretical distance between the sensor and targeted position on the ground is greater than or equal to 10.63 inches at an angle of roughly 5 degrees or higher.

In order for issue two to fail, a sensor would have to be misoriented by an angle of at least 5 degrees from its intended angle, regardless of which circle area the sensor is meant to cover. Given that even the crudest inclinometer has an accuracy of at most 2 degrees, misorienting the sensors should not be a problem [13].

3. Project Differences

3.1 Overview

Our proposed solution to this problem is fairly different than the original proposed solution. The primary method of occupancy detection from the original solution is to use a large mat with load cells to determine if a power rack is in use. Our primary method of occupancy detection is simpler than this as we will just have a button that the user presses to signify that they are using the power rack as well as when they have finished. The original solution has an infrared sensor to detect movement near the power rack as a second method of occupancy detection. We will also be using a sensor, but we will be using an ultrasonic sensor instead of the originally proposed infrared sensor. The original solution also has a website as their interface to communicate whether a power rack is occupied or not. Our user interface is significantly more complex as we will have an LCD display on the power racks that will show the name of the person who is currently using the power rack as well as the next three people in line to use it. We will also have a color indicating the usage status of the power rack where red means that the power rack is currently in use and green means that the rack is available. This data will also be displayed in the mobile application where people will be able to see how long the queues are as well as adding their name to a particular power rack queue. Notifications will also be sent via mobile application whenever their name moves up in the queue. The LED display will also be updated in accordance with the mobile application queue. The fifth and final difference between the original solution and our solution is that we designed our communication network differently. They have a more centralized, hierarchical structure where they have a master controller that governs sensor controllers and reports to a central server. Our network is simpler. Instead of waiting for the data coming from multiple sensor controllers in order to communicate data with the server via a master controller, we removed the master controller and will just use a single controller per rack to communicate individually with the server. This means that if one controller responsible for a rack goes down, it won't affect the functioning of all of the other controllers. It removes unnecessary levels of communication and adds robustness to the entire network.

3.2 Analysis

Table 9. Project Differences

	Their Solution	**Our Solution**	**Why is it better? (Pros)**	**Trade-offs (Cons)**
1) Primary method of Occupancy Detection	Use of a large mat with load cells	User Button	Minimizes false detection possibility, simpler	Partial loss in autonomy
2) Second method of Occupancy Detection	Infrared Sensor	Ultrasonic Sensor	More reliable, infrared is tricky in dark indoor settings	Cost
3) User Interface	Website	LCD display on the rack & a mobile application	Removes the need for constant internet connection, simpler	No use through a computer
4) Queue Option	None	Addition of a queuing feature	Giving the users an option when all the racks are full	None
5) Communication Network	Centralized, hierarchical structure	No master controller, instead single controllers per rack	Simpler, robust, reliable	None

1) The main idea of the original design is to use a large mat with load cells under the power rack to determine if the power rack is in use. Even though it adds autonomy to the overall design, we believe that it is an unnecessary complexity that can have a way simpler and a more reliable solution. For this reason, we chose to add a user button to our overall rack design where pressing the button would mean that the rack is in use. By pressing, a red light will appear on the mounted LCD display, displaying that the rack is occupied. When the user is done with using the rack, he/she will simply press one more time to state that they are done using the equipment where the color of the LCD will turn green. In

the case when the user forgets to press the button when they are done using the rack, we will use our sensor data to determine if there is any movement within the rack. If no movement is detected in two minutes then the system will automatically change the state from occupied to vacant.

2) Both infrared and ultrasonic sensors are used for object detection and distance measuring. However, after consideration of both sensors we found out that instead of an infrared sensor, an ultrasonic sensor will be more appropriate for this project. First of all, infrared sensors are better for object detection in very short lengths (those that are less than a decimeter), while the ultrasonic sensors provide better accuracy in lengths over 1 meter [15] which is very suitable for our project. On top of this, while infrared is better in detecting softer objects, for materials like iron, steel (materials within the barbell) ultrasonic works better [16]. Lastly, since the working principle of infrared relies on light sources, the change of lights within the gym can create problems in detecting, however since ultrasonic uses sound waves, light won't cause a problem [15], making ultrasonic even a better choice for this project. Since cost is not an issue for this project we don't necessarily see a downside for choosing ultrasonic over infrared.

3) In the original design, the user interface consisted of a website design where users would be able to login and see the availability of the power racks. We believe this is highly ineffective since within the gyms no one will bring in their laptops and will check the current situation of the racks. However, everyone carries their phones with them while working out so we thought a mobile application might be more useful in this setting. In addition to transferring the user interface to a mobile app, we also thought about the people who do not like to carry their phones around them while working out. For those people, we added LCD displays on top of the power racks where people will be able to see the vacancy of that rack as well as the live queue for that specific rack.

4) We have added the queue option which was never mentioned in the original design. With the queue feature, users can get in line in the event when all the power racks are full. Throughout the gyms in the US, when all the racks are full, people need to communicate verbally by setting up their own queues in their heads. This sometimes leads to confusion and often creates problems among the users, debating whose turn it is to use the rack. We thought of this feature because we already had a mobile app. It will be possible to get in line for a specific rack through the app and users will be able to see the current situation of the queue through both the app and the LCD display.

5) Our design for the communication network is better than the original design since our design has the capability to work without any extra problem when more weight racks are added to the system. In the original design, each rack controller has to send an update about its current state to the master controller before the master controller can send it to the server in order to update the database with this new info. In our design, each rack controller is isolated and responsible for sending its own state to the server for updating the database. Hence, we can achieve asynchronous database updates. In addition, we can say that our design is more distributed because we don't have a central controller governing over a set of rack controllers. Ours is also more reliable because if one of their racks fails to send its state, their system will either stall waiting to send anything to the server because it hasn't received the state of the failed rack controller, or they will have to have to add complexity to their code that works around failed rack controllers. More complexity to a system historically introduces more uncertainty to the system, which I suppose would still be worse than having the simple design that we have. We will be using a single AWS EC2 instance running a Docker container that maintains an Apache Cassandra database and runs Apache HTTP Server. They are using two Google Firebase databases and a Google Cloud API that will interact with the databases. The difference between these two implementations is just a matter of preference. Each component is set up differently, but performs more or less the same. Lastly in our network design, we only need to consider the transfer rate of the WiFi module whereas the original design has the WiFi module and a wireless transceiver. The wireless transceiver won't really affect the overall functionality of the system, but ours is faster since there are less layers of communication to go through within the design. So overall, we believe that our design is more reliable, more distributed, faster and simpler.

4. Cost and Schedule

4.1 Cost Analysis

We need to make a few assumptions before we start the cost analysis of building the Power Rack Manager. One assumption is that the average salary of an electrical/computer engineer is $40 per hour and the number of work hours required a week is, on average, 20 hours. Because we need to shorten our timeframe for this project, the new timeframe is 8 weeks as opposed to the normal course length of 16 weeks. This means that each team member will need to roughly double the number of hours dedicated to the Power Rack Manager each week. With these assumptions, the labor costs for 3 member team can be calculated as

$$Labor\ costs = \$48,000 \ = 3 * \frac{\$40}{hr} * \frac{20\ hrs}{week} * 8\ weeks * 2.5$$

Some assumptions also need to be made regarding our PCB costs. The estimated PCB cost for the Power Rack Manager is approximately $40. We will also require installation of the ultrasonic sensors and LCD display on the power rack and this cost can be estimated at around $10.

Table 10. Parts Cost

Description	Manufacturer	Part #	Quantity	Cost
5 V AC/DC Converter	Mean Well USA Inc.	IRM-15-5	1	$10.22
3 V Voltage Regulator	Analog Devices Inc.	ADP123AUJZ	1	$1.11
Ultrasonic Motion Sensor	Adafruit Industries LLC.	HC-SR04	4	$3.95
Occupancy Push Button	Keyestudio	KS0029	1	$3.50

Rack Microcontroller	Microchip Technology	ATMEGA328	1	$1.34
LCD Display	Display Visions	EADOGXL240 N-7	1	$50.92
Wifi Module	Sparkfun	ESP8266	1	$6.95
PCB	N/A	N/A	1	~$40.00
Mounting Costs	ECE Department	N/A	N/A	~$10.00
Total				**$139.84**

Table 11. Cost Breakdown

Section	**Cost**
Labor	$48,000
Parts	$139.84
Total	**$48,139.84**

4.2 Schedule

Table 12. Schedule

Week	John	Derek	Bartu
1	Project selection	Differentiating the new project	Preparation of the RFA
2	Preparation of the Project Proposal	Parts selection	Preparation of the Project Proposal
3	Circuit schematics for the subsystems	Mathematical modeling	Preparation for the Design Document
4	Initial machine shop discussion	First design of the PCB layout	Designing the PCB layout
5	Power subsystem assembly	Initial software implementation	Sensor subsystem assembly with the PCB
6	Connection and implementation of all the subsystems	Server/Database testing	Testing & Verification
7	Initial outline of the final report	Debugging	Demonstration
8	Presentation	Presentation	Finishing the final report

5. Discussion of Ethics and Safety

Even though our project is not considered to be highly ethical at first glance, we believe that we are tackling a common problem for many sports enthusiasts. Our product intends to minimize the time lost in a gym as well as providing a fair distribution of equipment for all the gym members in the world. So, in other words, we are trying to create a product that would maximize the efficiency of our most important asset, time. We believe that a product/design that intends to create a fair environment for its users while minimizing the time they are losing is an ethical product.

To minimize the time lost for gym members, we are implementing an app that would notify its users about the vacancy of the racks in their gyms. This way users will be able to learn from the app whether or not there is an empty rack for them to workout. We will be totally honest and realistic while processing our sensor data to be as accurate as possible for each rack’s vacancy. We believe this fulfills the 3rd clause of IEEE Code of Ethics which states “to be honest and realistic in stating claims or estimates based on available data” [1].

In the case when all the racks are occupied, we will have a queueing system that would allow the users to get in line for a specific rack. This way users won’t lose time waiting for a rack to open up as they will be able to estimate when their rack will be vacant and schedule their time accordingly. In our experience, we have seen many individuals who waited for a rack to open up and when it was finally vacant, someone else just showed up and acted fast to steal their queue. Since our queuing system will provide an honest and fair way of equipment allocation, we fulfill the 8th clause of IEEE Code of Ethics which states “to treat fairly all persons and to not engage in acts of discrimination based on race, religion, gender, disability, age, national origin, sexual orientation, gender identity, or gender expression” [1].

For a project like this, we, as a group, considered all the possible safety hazards that could happen in the desired setting and took the necessary precautions and measures to make our design/product as safe as possible. This approach aligns with the 1st clause of the IEEE Code of Ethics which states “to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, and to disclose promptly factors that might endanger the public or the environment” [1]. For safety precautions we focused on two possible scenarios; our sensors sending out wrong signals to the control subsystem and having liquid spill over our sensor and power subsystems.

In the first scenario when our ultrasonic sensor sends out the wrong signal, our product won’t be able to determine the vacancy of the rack properly, resulting in the lost time of our customers. For this case, we are considering the sensor itself to fully function. Since the sensor is fully working, the only scenario where it can send a wrong signal is if the height of the rack makes the ultrasonic sensor out of the rack's reach. To tackle this possible safety hazard, we chose an ultrasonic sensor that can detect movement within 6.5 meters. We believe this is more than enough since there are not any gym racks that are higher than 6.5 meters.

For the second case, where a liquid spills over either the sensor or the power subsystem, there are possibilities of fire and getting shocked. Since gym is a place where the users sweat a lot while also consuming a lot of liquids whether it is water or something else, there is always the possibility of spillage. To minimize the possibility of this incident, the first thing we did was to place all the sensor subsystems on top of the rack. As we can see from the physical design drawing in figure 1, we placed the sensors away from most people's reach. However, since our product needs to be connected to a power outlet providing 120V, we also needed some protection close to the ground. This is why all of our wires/cables going to the power outlet will be covered with "Medium Capacity Cord Covers" [2].

Looking at the previous group's design, we also found a possible safety hazard. Since they implemented load cells below the rack, there was a possibility of the rack becoming unbalanced. Changing this with a button that is placed next to the sensor subsystem, cleared away the possibility of this incident.

We, as a team, believe that there is always room for improvement and will enhance our product/design by being open to any feedback and comment from our users. We believe this approach fulfills the 7th clause of the IEEE Code of Ethics which implies "to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others" [1].

6. Citations

[1] “IEEE Code of Ethics,” IEEE. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 03-Apr-2020].

[2] CableOrganizer.com, “Home,” Cord Cover Protection - Conceal Power & Data Cables. [Online]. Available: https://www.cableorganizer.com/cord-covers/. [Accessed: 04-Apr-2020].

[3] A. M. Zungeru, “Design and Development of an Ultrasonic Motion Detector,” International Journal of Security, Privacy and Trust Management, vol. 2, no. 1, pp. 1–13, 2013.

[4] E. Porter, “How big is a power rack? (Dimensions & 13 examples),” Trusty Spotter, 06-Mar-2019. [Online]. Available: https://trustyspotter.com/blog/power-rack-size/. [Accessed: 17-Apr-2020].

[5] Andrea, Andrea, The Transition: Average Body Measurements. [Online]. Available: http://andreaportman.tripod.com/averages.html. [Accessed: 17-Apr-2020].

[6] “Datasheet for IRM-15,” Mean Well. [Online]. Available: https://www.meanwell.com/Upload/PDF/IRM-15/IRM-15-SPEC.PDF.

[7] “Datasheet for ADP 122/123,” Analog Devices. [Online]. Available: https://www.mouser.com/datasheet/2/609/ADP122_123-1503483.pdf.

[8] “3942,” DigiKey. [Online]. Available: https://www.digikey.com/product-detail/en/adafruit-industries-llc/3942/1528-2711-ND/9658069. [Accessed: 17-Apr-2020].

[9] “Ks0029 keyestudio Digital Push Button,” Ks0029 keyestudio Digital Push Button - Keyestudio Wiki. [Online]. Available: https://wiki.keyestudio.com/Ks0029_keyestudio_Digital_Push_Button. [Accessed: 17-Apr-2020].

[10] “Datasheet for DOGXL240-7,” Electronic Assembly. [Online]. Available: https://www.lcd-module.com/fileadmin/eng/pdf/grafik/dogxl240-7e.pdf.

[11] “Datasheet for ATmega328PB,” Microchip. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/40001906C.pdf.

[12] “Datasheet for ESP8266,” SparkFun Electronics. [Online]. Available: https://cdn.sparkfun.com/datasheets/Wireless/WiFi/ESP8266ModuleV1.pdf.

[13] “Inclinometer,” Wikipedia, 10-Apr-2020. [Online]. Available: https://en.wikipedia.org/wiki/Inclinometer#Accuracy. [Accessed: 17-Apr-2020].

[14] Circles Covering Squares. [Online]. Available: https://www2.stetson.edu/~efriedma/circovsqu/. [Accessed: 17-Apr-2020].

[15] “RF Wireless World,” Difference between Ultrasonic sensor and Infrared sensor. [Online]. Available: https://www.rfwireless-world.com/Terminology/Ultrasonic-vs-Infrared.html. [Accessed: 18-Apr-2020].

[16] “Performance comparison of Infrared and Ultrasonic sensors for obstacles of different materials in vehicle/ robot navigation applications,” IOP Conference Series. [Online]. Available: https://iopscience.iop.org/article/10.1088/1757-899X/149/1/012141/pdf.