Alfred: WiFi-Enabled Automated Mixed Drink Maker

Patrick Barron, CSE, Chris Wong, CSE, Ben Ivaldi, EE, and John Fouad, CSE

***Abstract*—College towns and major cities are often filled with overpopulated bars. Alfred helps alleviate congestion while also expediting the process of buying a drink by allowing customers to order a drink through their mobile device. When a drink is ordered by the customer, that drink choice is sent to a controller, which then rotates a base holding 8 cups. The correct cup is rotated to a certain position where a drink is dispensed with the correct proportions, and then rotated again to the dispensing door. The customer will receive a personal, four-digit identification code that he/she/they will be able to enter at Alfred, and the dispensing door will vend their drink to the customer. Alfred will pause its execution until the drink is removed by the customer, then continue its execution.**

I. Introduction

HAVE you ever been at a bar and faced the struggle of waiting in an overcrowded line to order a simple mixed-drink? Or have you ever complained upon receiving the wrong drink at a bar, or a drink that is too strong or not strong enough? Alfred will allow users to order simple mixed-drinks from their mobile device at the bar without having to wait for the bartender or having to deal with pouring errors or receiving the wrong drink.

There have been inventions [11, 12] that are somewhat similar to our design but none of those have the fully automated functionality that Alfred contains. Also, similar designs that are used in society today such as the "Coca-Cola's Freestyle are available on lease for $320 per month" which is very expensive compared to our $500 dollar budget to make Alfred [1, 14]. Our design will also save people time in crowded bars, and in today's society time is very valuable to people. Our design will allow customers to spend more time with their friends, co-workers, family etc. instead of wasting their time waiting to order a drink which will increase their enjoyment of the bar.

Our requirements were developed based on the size of our design. Alfred will be able to be placed on a bar and the bartender will only have to insert cups of ice and bottles of the drink choices into the machine periodically. Alfred will be powered by plugging the system into a standard outlet. The requirements are also geared towards the customers as they will have to wait limited time to receive a drink and can order from their mobile device. Table I shows a list of specifications.

Table 1: List of requirements and specifications

Requirement	Specification
Pour a mixed drink	Correct portions and completed in under 2 minutes
Multiple drink options	Bartender can insert choice of alcohol (750mL) and mixers into dispensers. Choice of 4 different drinks
Minimizes spilling	Spills less than 5% of drink
Online ordering	User orders through mobile website
Minimizes transaction costs	Tab system for ordering
Drink served to correct customer	Serving door only opens when id code is presented
Simultaneous pouring of liquids	15.9" rotating base with 8 cups of ice and 6 pumps
Failsafe: detects positioning of base and cups	Sensors to make sure cup is removed before closing the serving door. Sensor used to align base

Figure 1: Alfred Front View

P. Barron from Bridgewater, MA (e-mail: pbarron@ umass.edu).
C. Wong from Norwood, MA (e-mail: cmwong@ umass.edu).
B. Ivaldi from Bridgewater, MA (e-mail: bivaldi@ umass.edu).
J. Fouad from Framingham, MA (e-mail: jfouad@ umass.edu).

II. Design

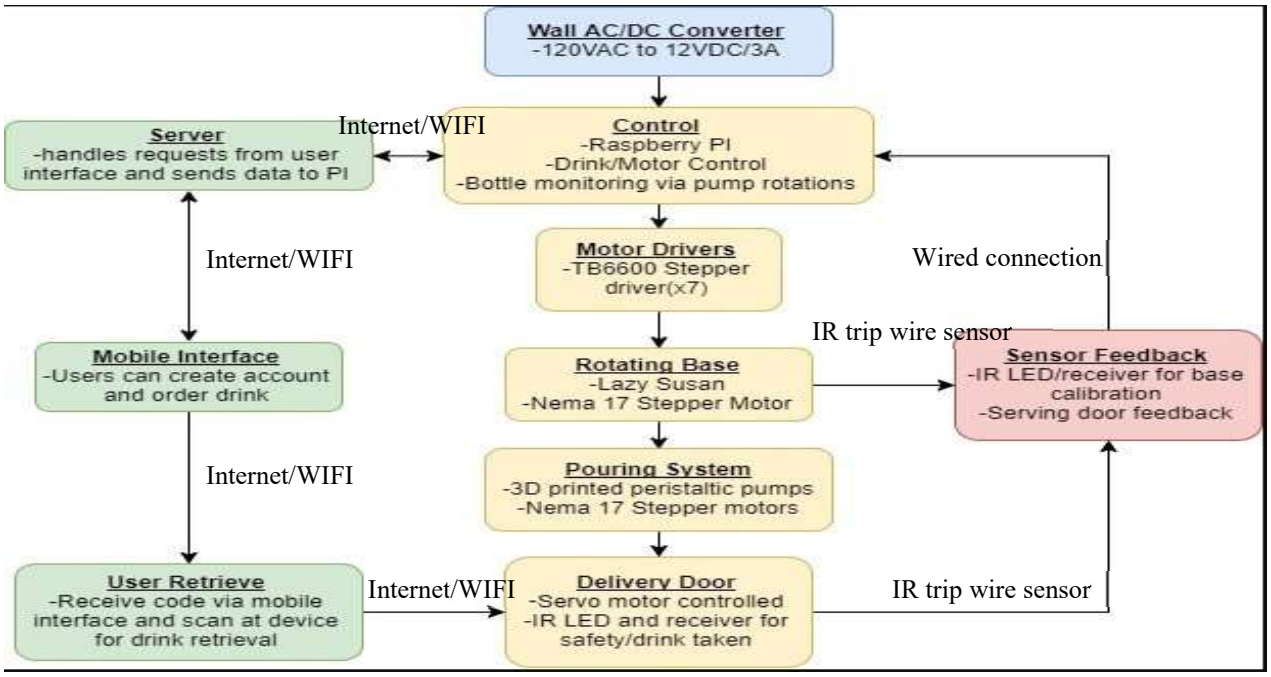


Figure 2: Block Diagram

A. Overview

For Alfred to function properly, it will need multiple motors and controllers. The use of a Raspberry Pi was needed to get Alfred to dispense beverages with the correct proportions. An "order" button is pressed on our website, which then initiates Alfred to begin creating a drink. Seven stepper motors are used for the prototype: six for pumping the liquids and one for rotating the base.

A serving door is implemented that has a laser and photoresistor, one connected to each side – this lets the system know when the cup is removed from the rotating base so that the serving door can shut again and execution of the machine can continue. There is also a laser and photoresistor for the rotating base to make sure the base is calibrated in the correct position i.e. under the correct pumps. The user is able to select a specific drink that they want made, and Alfred will dispense the correct drink with the correct proportions. The user will receive a specific, four-digit identification number that they will be able to enter at a keypad and receive their drink.

According to Figure 2, the power coming from the wall outlet goes into the Raspberry Pi. From there, the Pi controls the motor drivers which in turn control the stepper motors and servo motor responsible for the rotating base, the pouring system, and the delivery door. The Pi also communicates with the server which handles requests from the user (mobile) interface, where users can select the specific drink they would like to order. A code will then be sent to the user which will allow them to retrieve their drink. Once the delivery door opens, the photoresistor will let the Pi know when the cup has been removed, allowing it to close. Once a drink has been successfully retrieved by the user, Alfred will continue making drinks and the process starts all over again.

Other alternatives we considered were Chinese peristaltic pumps, but their flow rate was too slow [13]. We also looked into mechanical shot fillers that perfectly pre-filled a shot, but we would have to create another mechanical arm that pushes the shot filler up.

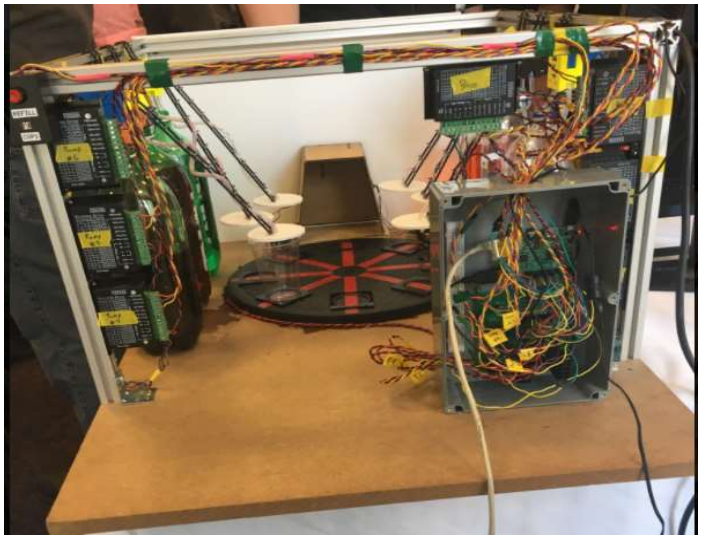


Figure 3: Alfred back view

B. Rotating Base and Calibrating Laser + Photoresistor

The rotating base is a Lazy Susan that is screwed into a wooden platform. A stepper motor driven by a motor driver has a rubber wheel attached to the end of it, and when the motor rotates, the wheel rotates [3, 4]. This wheel is placed on the base. As a result, the base rotates synchronously with the wheel but with a larger turn radius. Code written on the Pi is controlling the rotating base to move to certain positions based

on where we want a cup to be rotated to. Six specific drink nozzles are located in certain areas around the base. The base rotates a certain number of steps to a specific position to receive the correct liquid. The calibrating laser and photoresistor ensures that we know exactly where the base is at all times [6]. The photoresistor is attached to the wooden base, while the laser is suspended on top of it, facing directly downwards at the photoresistor. There is a small rubber square attached to the side of the rotating base. While Alfred is operating, the photoresistor is continually receiving light from the laser, but at some point while the base is rotating the rubber square impedes the light hitting the resistor. At this moment, we know that the base is aligned properly. This calibration is to ensure that error does not accumulate from minute degree inconsistencies in the rotation of the base/stepper motor. Once the laser/photoresistor is lined up, we can continue rotation of the base knowing the exact location of each cup. Calibration is done after all eight cups have been taken and replaced with eight new ones. Knowledge learned in Software Intensive Engineering and Computer Systems Lab 1&2 allowed us to control the Arduino, control I/O, and control the motors. We will implement a test and check method that ensures the correct functions of the laser and photoresistor, and that the rotating base moves to the correct position via the shortest path.

Figure 4: Alfred Bird's Eye View

C. Serving Door and Laser + Photoresistor

A serving door is at the forefront of Alfred which opens up once a customer enters their unique, four-digit identification code. This door is in the shape of a trapezoidal prism with two faces missing (see Figures 5 & 6). If one was looking directly at the front face of the door, the bottom face and the back face will be the ones missing. The door is rotated back 90 degrees with the help of a servo motor, so that the front face is now on top. This door rotates around the final cup position on the base, which will allow the customer to reach in and grab their drink without being able to access the other drinks in the machine. In the two side faces of the door, there is another laser and photoresistor combo that lets us know whether the completed cup has been removed or not. The light from the laser is hitting the photoresistor while the door is closed, but once it opens around a cup, the light is refracted because of the liquid in the cup and is no longer hitting the photoresistor. When the cup is removed, the laser hits the photoresistor once again and the system knows that it is safe to close the door. Alfred then continues execution. Knowledge obtained from Software Intensive Engineering and Computer Systems Lab 1&2 allows us to control the serving door. The servo motor and the communication from the photoresistor will be controlled by the Raspberry Pi. We changed the original design of the door from a cube to a trapezoidal prism to accommodate the minimal area we had to drop the door down between the adjacent cups in the machine.

Figure 5: Serving Door

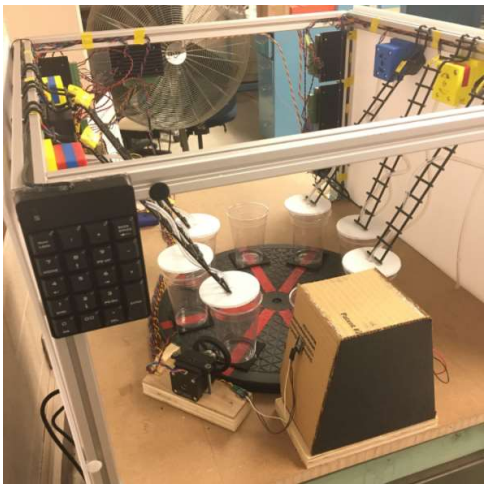


Figure 6: Serving Door with Keypad

D. Pouring System

This subsystem of Alfred is responsible for the liquid distribution. The main design goals of this system are speed, accuracy, and affordability. In order to meet these goals we used an open source 3D model of a peristaltic pump, designed by silisand[1],[2] that is driven by a Nema 17 stepper motor [3]. By doing so, we were able to utilize the unique properties of these types of pumps, without paying the high market cost for them. These allow for high precision liquid distribution, without having liquid touch any mechanical or electrical components.

Because these motors use a significant amount of power, they each require a motor driver. The drivers chosen for these motors was the TB6600 [4]. These drivers allow for simple control signals and variable microstepping if needed, but we found that the normal 200 step per rotation of the motors themselves was sufficient.

In addition to the drivers, the pouring system also required 3.3V to 5V logic level converters for each driver. This is because the drivers require 5V signals to operate, while the Raspberry Pi I/O pins have a logic level high of 3.3V. Because of this, we created a PCB of transistor amplifiers to increase the signal voltage. The knowledge gained for Electronics 1 and 2 was sufficient in designing this, and the knowledge gained from Computer Systems Lab 1 allowed for the digital control of the motors.

After printing, building, and wiring our preliminary pumps, we found that they did not have enough holding strength for the amount of liquid required for our design without dripping. Since dripping liquid into cups without command is not desired, and could be a hazard, we needed to solve this. By implementing code to run the pumps backwards after distribution to the designated cup was complete, the tube line is cleared and no liquid is available to drip into the cup.

Overall, the use of these printed pumps in conjunction with the drivers have allowed for quick and accurate distribution of liquid. In comparison to the initial Chinese peristaltic pumps we bought, which had a flow rate of about 2.7 fl oz/min, our printed pumps have a flow rate of about 12.5 fl oz/min, which greatly improves our production time [13].

E. Mobile Interface and User Retrieve (Keypad)

The mobile interface for Alfred consists of a mobile website that is accessible from a smartphone and enables the user to create an account. Upon creation of the account the user will then be able to place orders from a choice of 4 mixed drinks. From there the users order will be sent through the control system and initiate the drink making/pouring process. When the users drink is complete, they will then receive a text message with a code and can proceed to use their code to retrieve their drink.

The website uses HTML, JS, and CSS code for the user interface side of the website. We used Django to implement the website while coding in Python and linked our database and Redis queue to an Amazon server and we will explain this more in the next section. This code allows users to create an account and link their account with the sites database so they can proceed to order drinks. For the ordering itself, code is used to form a selection menu where the user can click which drink they want and then click a separate place order button. The website is very user friendly as we added drop down menus for drink selection and also enabled a passwords reset for user accounts that sends them information about changing their password.

We use our gained knowledge from Software Intensive Engineering with regards to coding in HTML and also making websites user friendly. Going forward we will continue to learn more JS and CSS to possibly improve the interface and also learn more about working with the front-end and the database to store the users account information. To design and test we checked our code files on a web browser to see exactly how the site looks and also we will test the ordering and account portions by going to the site and placing orders and creating accounts respectively. To analyze these results we made sure the website looks nice on the mobile device and is capable of creating an account and then placing a drink order.

Upon completion of the drink being poured, the user will receive a personal identification code in the form of a text message to then go and retrieve their drink. To use this block we will start by sending the user a personal code that is a number sent to their phone using Twilio and then the users will be able to use that code to go up to Alfred and get their drink. We implemented this with a simple USB Keypad where the users enter their personal code and then press enter and their drink will rotate to the door and be served to the customer. From there we would have liked to advance our project to send the user a unique QR code or bar code that they will be able to scan when they go to retrieve their drink, but it remained to be too expensive for our budget and we ran out of time.

F. Server and Client Block

The purpose of this block is to allow for communication between each of the major components of Alfred to control the system as a whole. For quick implementation and testing, CPython was initially used to prototype the server and client [7]. However, with the scaling of the system to include multiple processes, the CPython was transcribed to C since CPython's Global Interpreter Lock prevents threads with shared memory from operating in parallel. The server and client use sockets and TCP to communicate via the internet. However, to increase the scalability of our design, we revamped the server to utilize a Django server with an attached sqlite3 database and a Redis queue for communication with the embedded client. The Django framework allows for dynamic generation of the webpages that we send to the client's mobile device. The sqlite3 database stores all of the information that we would like to keep, including user information like username, hashed password, email, and phone number; along with administrator details like beverages available on each machine and drink receipts. The Redis queue was used to interface sending messages between the server and client. By using this technology, a bar using our product could own multiple Alfred machines and the drink could be sent openly

on the queue and pulled from any of the machines, allowing for scalability. The server was hosted remotely with Amazon Web Services and the client will run on a Raspberry Pi 3 Model B [9]. The Raspberry Pi's Broadcom BCM2837 ARM Cortex-A53 is a 4-core, 2-way superscalar processor which should enable it with the ability to run multiple process threads at the same time [8, 10]. This is useful since the complexity of maintaining communication with the server and processing the information to control the system will require a multithreaded solution for the fastest performance.

The server and client block is very software heavy. Therefore, there were many techniques that were used from the more software-based classes. Data Structures and Algorithms was useful for determining how the data should be saved and processed in an efficient way. Introduction to Computation and Cryptography aided in the selection of a mathematical ring using modular arithmetic to provide a mathematical model that the client could interpret. Computer Systems Lab I and II helped with mapping the operating system's address space to the physical addresses that control the general purpose input output pins on the Raspberry Pi's board. Computer Networks and the Internet assisted in understanding and implementing the internet protocols. Software Intensive Engineering helped with controlling multiple thread resources so that there are no errors or deadlock between the process threads. Computer Architecture aided with choosing a processor that would fit our needs for the project.

While the prior knowledge that we had helped with this block, we still needed to learn specific hardware-software interactions for the Raspberry Pi. Also, we need to learn more about developing a solution that could be ported to industryscale tools. For example, hosting our server on someone else's server and using the Django framework for dynamic webpage generation.

To design and test this block, the software will have to be run under realistic situations and workloads. The remote server will need to serve webpages and communicate with a reasonable number of clients. Also, the client will need to handle a reasonable number of drink requests. In order to test the performance, we will put the system under realistic to heavy workloads and then measure the system's response time, utilization, processing time, and task completion time. All of these metrics can be analyzed to see if the system is performing better or worse. Lower response times, processing times, and task completion times mean that the system is performing better. Utilization is harder to analyze. While it is preferred that the system is being utilized a lot, which could mean that the system is working to the maximum potential, having too much utilization could result in overworking the system and causing rapid deterioration of the hardware, and it could also mean that there is a bottleneck in the system.

G. PCB

The purpose of this block is to route all of the signals and power to the correct components. We used the Eagle CAD tools to generate the necessary Gerber and Excellon files to send to the fabrication company. In addition, we used the datasheet of NXP's GTL2002 chip to properly design a system on the PCB that was capable translating the 3.3V logic level of the Raspberry Pi to the 5V logic level used by the motor drivers.

While we did not develop a layout, since the fabrication of the PCB required using computer aided design, our experience using industry tools for CAD in our VLS design class. The soldering skills that we learned during Electronics I helped in assembling the board after it was fabricated.

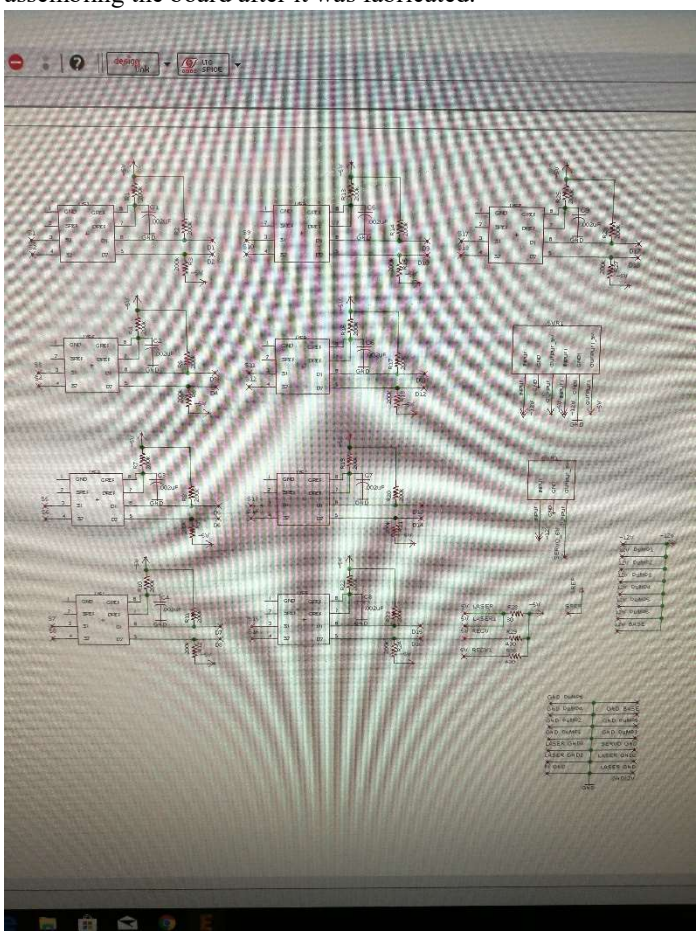
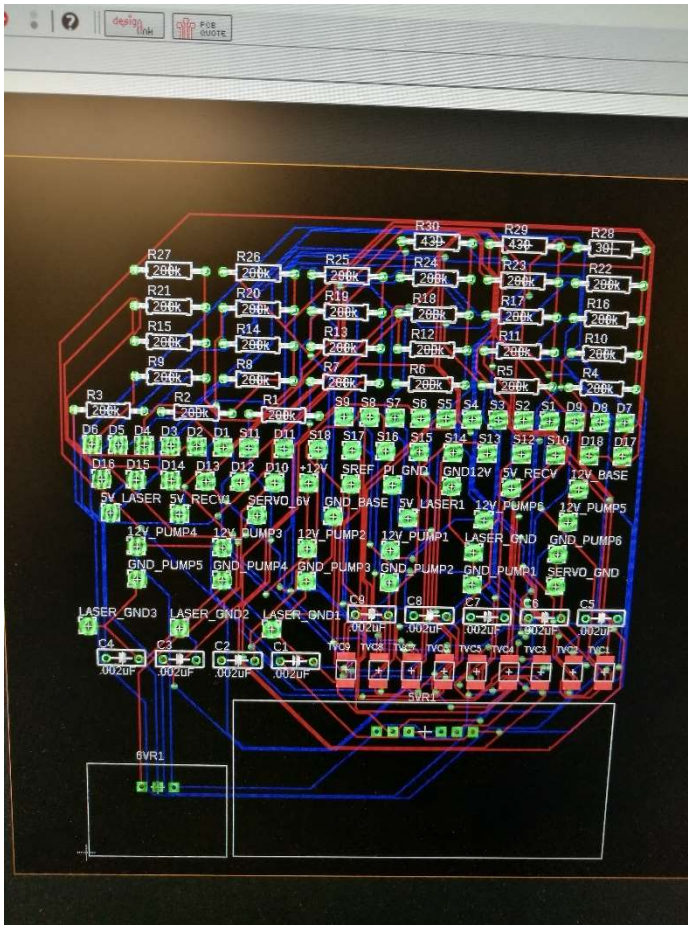


Figure 7 (Above): PCB Schematic & 8 (Next Page): PCB Layout



III. Project Management

By following our deadlines and working together with clear communication, each component was integrated to work in harmony with the new implementation of the serving door, feedback sensors, PCB, and personal identifier; also, the old implementation was scaled and updated to support the simultaneous pouring of 6 drinks at the same time without a microcontroller or a local host.

The team works as a cohesive unit. Each team member has a specialty that they can apply to specific subsystems, and there is enough overlap in knowledge between each team member, which promotes joint solutions to problems that occur in any given subsystem.

John Fouad and Benjamin Ivaldi's specialties regard mechanical, power, sensor, and microcontroller devices. Patrick Barron's specialty lies in frontend development. Christopher Wong's specialty lies in backend development.

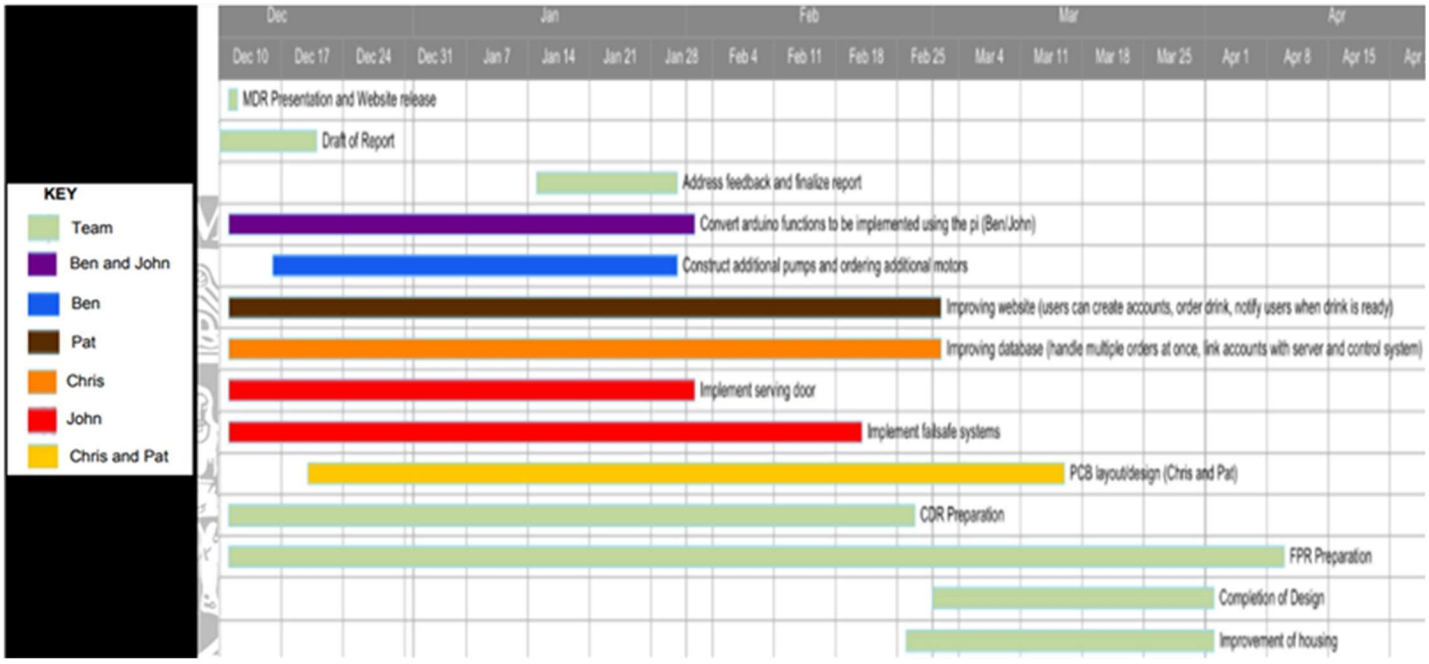
This collaboration is aided by the open communication within the team. Often, there are daily discussions between group members with weekly team meetings. In addition, there are weekly meetings with our advisor scheduled to check our progress and ensure that the project is progressing forwards.

IV. Conclusion

Since MDR, Alfred has been scaled-up and with the completion of the pump system, implementation of the serving door, and the addition of the ability of the website/server to intake a personalized drink order.

One system preventing the system from being fully ready for production is the full Twilio package. In a full production model, we would have purchased a Twilio subscription; however, since our model was only a prototype, we only had a free trial that only allowed for texting to registered phones.

Figure 9: Gantt Chart



Given more time and budget, we could have built another system to display the full scalability of our system design, having multiple machines pulling from the same queue to make drinks for a high volume of customers. In addition, we could have added a cup dispenser that would eliminate most of the bartender's responsibility for maintaining Alfred during operation.

V. ACKNOWLEDGEMENTS

Special thanks to our advisor Professor Andras Moritz and our evaluators Professor William Leonard and Professor Guangyu Xu. Also thanks to Professor Christopher Hollot, Professor Baird Soules, and Francis Caron.

REFERENCES

[1] Pullen, John Patrick. "5 Technologies Changing the Restaurant Industry." *NBCNews.com*, NBCUniversal News Group, 9 Sept. 2012, www.nbcnews.com/id/48959179/ns/businesssmall_business/t/technologies-changing-restaurantindustry/#.WjsylFWnHIV.

[2] Thingiverse.com. "Peristaltic Pump Improved for Nema 17 by Silisand." *By Silisand - Thingiverse*, www.thingiverse.com/thing:1134817.

[3] Clifford, Paul. "Stepper Motor Specifications, NEMA 17 1.8 Degree 200 Steps-per-Revolution Four-Phase Unipolar Permanent-Magnet StepperMotor." Find Controllers for Instrumentation and Automation at the Mosaic Industries Site, Mosaic Industries, Inc., www.mosaicindustries.com/embedded-systems/microcontroller-projects/steppermotors/specifications.

[4] "TB6600 Stepper Motor Driver SKU: DRI0043." DFRobot, 28 June 2017, www.dfrobot.com/wiki/index.php/TB6600_Stepper_Motor_Driver_SKU:_DRI0043

[5] "Ks0002 Keyestudio Mega 2560 R3 Development Board." Ks0002 Keyestudio Mega 2560 R3 Development Board - Keyestudio Wiki, Keyestudio, wiki.keyestudio.com/index.php/Ks0002_keyestudio_Mega_2560_R3_Development_Board.

[6] "IR Receiver Modules for Remote Control Systems." Vishay, www.vishay.com/docs/82491/tsop382.pdf.

[7] "Python/C API Reference Manual." Python/C API Reference Manual - Python 2.7.14 Documentation, Python, docs.python.org/2/capi/index.html.

[8] "ARM® Cortex® -A53 MPCore Processor." ARM, docs-apipeg.northeurope.cloudapp.azure.com/assets/ddi0500/g/DDI0500G_cort e x_a53_trm.pdf.

[9] "Raspberry Pi Schematic." RaspberryPi.org, www.raspberrypi.org/documentation/hardware/raspberrypi/schematics/Raspberry-Pi-3B-V1.2-Schematics.pdf

[10] "ARM® Cortex® -A53 MPCore Processor." ARM, infocenter.arm.com/help/topic/com.arm.doc.ddi0500d/DDI0500D_cortex_a53_r0p2_trm.pdf.

[11] Jamiep, and Instructables. "Build a Mobile Bar - BaR2D2." Instructables.com, Instructables, 8 Nov. 2017, www.instructables.com/id/Build-A-Mobile-Bar-BaR2D2/.

[12] Lomas, Natasha. "Barobot Is A Hackable Cocktail Mixing Robot." TechCrunch, TechCrunch, 20 May 2014, techcrunch.com/2014/05/20/barobot/.

[13] "Dosing Pump12V DC Peristaltic Liquid Pump Hose Pump Dosing Head for Aquarium Lab Analytical Water (Green)." Amazon.com, www.amazon.com/dp/B01HRPKBAE/ref=sspa_dk_detail_1?psc=1&pd_rd_i=B01HRPKBAE&pd_rd_wg=HgS0y&pd_rd_r=H11V96JM160276XH0KWT&pd_rd_w=q4xRM.

[14] "Coca-Cola FreeStyle." Coca-Cola, www.coca-colafreestyle.com/.