# ECE Capstone Project

Authors: Aryan Chordia, Thomas DeLauro, Tyler Duic

*18500 ECE Design Experience, Team E2*

*Electrical and Computer Engineering, Carnegie Mellon University*

*{aryanc, tdelauro, tduic}@andrew.cmu.edu*

***Abstract* — We will be creating an automated drink-making machine, capable of making a variety of different cocktails and mixed drinks. The user will be able to use a phone application (iOS) to place these drink orders and receive their drinks in a matter of seconds. The user will only be required to place the glass on the platform of the machine and then use the phone application to select the drink of their choice. The alcohols and mixers must be pre purchased and placed into the relevant holders on the machine itself.**

***Index Terms*—Bluetooth Low Energy, DC Barrel Jack, Elegoo Mega, iOS, Internet Of Things, Peristaltic Pumps, Stepper Motor, Swift, Timing Belt**

## I. Introduction

Everybody's lives during the COVID-19 pandemic have changed drastically when compared to pre-coronavirus times. Due to the nature of the disease, individuals everywhere have been social distancing in an effort to decrease the spread of the disease. Bars and gatherings everywhere have closed down to comply with social distancing regulations. Now more than ever, individuals are staying home and spending most of their time with their families and close friends and often pass the time by enjoying their favorite drink right at home. Alcohol sales in the United States are at an all time high, up close to 250% from 2019 to 2020. We wanted to build something that provides users with a fun and interactive way to mix and create their drinks.

The drink maker that we have designed will be able to make a multitude of drinks with up to 4 different liquids. The drinks will be mixed after the completion of the pouring process, a feature that many other bartender robots lack. Our system will require the user to simply put their glass on the platform and select which drink and size they desire, all from a phone application. The machine will begin dispensing the liquids within 3 seconds of the user request and will execute the drink order with 100% accuracy without spilling a drop.

## II. Design Requirements

### *A. Phone Application Communication*

The phone application must be able to maintain the bluetooth connection with the module and hence the system at all times the user is logged in. In the case that there is a power cut and the bluetooth module is turned off the users current drink progress and history must be saved and protected. The bluetooth module must be able to correctly relay the users request to the machine without any corruption of data to ensure there is 100% accuracy with the users drink request and the drink the machine makes.

### *A. User Feedback*

The user must be able to place their drink order using the phone application and the drink must be prepared once the user places a glass on the platform. The application must notify the user if a certain liquid needs to be refilled as well as only allow the user to choose a drink order if it is possible to make with the existing liquid levels. The user must be notified if there are pending drink orders on the machine and should get a notification when it is their turn on the queue. The application must provide a monthly report to the user on their drinking habits, which will include metrics such as calories consumed, favorite drinks/alcohols and new drink suggestions based on current consumption patterns.

### *B. Drink Making Process*

During the making of the drink the glass will be placed on a moving belt and since there will be liquids in the glass, we must ensure that there is zero spillage. There will also be a rod to mix the drink once all the liquids that comprise the drink are poured in. Hence we must ensure that when this rod stirs the liquids in the glass, there is no spillage while also making sure that the strength of the stir is good enough to attain the desired taste for the drink.

### *C. Physical Structure*

There will be a wooden structure that will encase all the drink bottles as well as the silicone tubing to connect to the dispenser. This structure must be strong enough to hold at a minimum four bottles each 750ml in volume. As well as this, the mechanism to replace the bottles when they are finished must be effortless and highly intuitive to the users. The application will notify the users when the bottles need to be replaced.

### *D. Drink Quality*

When it comes to the quality of the drinks produced from our machine, the drinks must have the precise amount of liquids as defined by the recipe for the drink. The drinks must also be correct in terms of the user it belongs to and hence the queue integrity must never be violated. The drinks must not be prepared if the liquids needed for the drink are not available in the required quantities as defined by the recipe for the drink.

## III. Architecture and/or Principle of Operation

We are building our iOS application using Swift with Xcode as the IDE. Swift provides error prevention and speed and was developed by Apple specifically for creating applications such as ours. Xcode provides a development environment specifically curated for creating iOS and MacOS applications and allows the developer to preview their application by simply connecting an iPhone to their Mac.

Upon entering the application, the user will be prompted to create a profile and connect to the Bluetooth module used by our machine. Within the application, the user will be presented with options to repeat their last order, request a new drink, and check their drink history and statistics of drinks they have ordered in the past. Upon ordering a drink, a signal containing the drink order and an identification key for the user will be sent over Bluetooth Low Energy to an *ELEGOO Mega,* using a *DSD Tech HM-10 Bluetooth Module* to facilitate the communication.

We are using an *ELEGOO Mega* as our microcontroller to control all the movements and signals to the motors. The phone application will trigger a programmed response within the microcontroller based on the user request. For example, if the user requests a drink, the request will be sent to the microcontroller using the *DSD Tech HM-10 Bluetooth Module*. Upon receiving the signal from the Bluetooth module, the microcontroller will then trigger our motors to move the glass to the dispensing station. The servo and peristaltic tubes will dispense the required amount of each liquid for the drink into the glass (these individual motors are also controlled by the microcontroller.) Once all the required liquids have been poured into the glass, the platform will move the glass to the mixing station. Here, a stirrer will enter the glass and stir the combination of liquids to create the desired taste of the drink.

## IV. Design Trade Studies

### *A. ELEGOO Mega*

We decided to use the *ELEGOO Mega* in our design for a multitude of reasons. Firstly, the *ELEGOO Mega* is compatible with the Arduino IDE. This allows us to program the *ELEGOO Mega* to control the hardware components in the same manner as we would an Arduino. The *ELEGOO Mega* also offers us enough GPIO ports to control our components, something that other microcontrollers we considered lacked. Overall, this was the most cost effective microprocessor available that checked all of the boxes we had.

### *B. Servo-Controlled Mixing System*

The decision to include this sub system in our design came after realizing that this feature was missing from every other bartender robot we came across during our initial research. Ensuring that a created drink is properly mixed is one of the most important parts of the drink-making process.

We decided to implement this system using two different servos; a standard *SG90 9g Micro Servo* and a *Continuous Rotation Servo.* This solution was the most compact and efficient way we could formulate to create a mixing contraction for our system. Our choice to use two different servos stems from the use case of each of them; the *Continuous Rotation Servo* was necessary to perform the stirring action whereas the *SG90 9g Micro Servo* is only needed to lower and raise the platform.

### *C. Linear Sliding Platform*

The *NEMA 17 Stepper Motor* seems to be the best option for implementing a small-scale sliding platform. This stepper motor can generate more than enough torque to move the glass's platform. The choice of timing belt and timing belt pulley are a cheap and effective option to slide the platform. Because we are using the *NEMA 17 Stepper Motor,* it will also be very easy for us to position the platform where we want it to be by simply counting the number of steps the motor takes.

We decided to use a timing belt and timing belt pulley to linearize the motion as this was the most common option amongst other designs that we researched.

### *D. Peristaltic Pumps*

We chose to use these pumps to pump the liquid from the bottles and dispense into the drink due to the ease of integration with the *ELEGOO Mega*. These pumps will also allow us to pump precise amounts of the desired liquid into the silicone tubes for the drink. We initially wanted to use bottle dispensers however, we chose to use these pumps instead as the dispensers world require additional motors to activate them and this would add complexity which is unnecessary. As well as this the pumps allow for much more precise measurements of each liquid used in the drink.

### *D. Bluetooth Communication*

While it adds a layer of complexity to the project as compared to creating a web application which could communicate with the *ELEGOO Mega*, we chose Bluetooth because it allows for a faster, more seamless connection between the user placing a drink request and the request being received and processed. Using Bluetooth also ensures drink requests will only be placed when near the machine as you must be close-by in order to establish a connection. We specifically chose to use the *DSD Tech HM-10 Bluetooth Module* because it comes with a shield to prevent disturbance and supports the protocols used by iOS for BLE communication.

### *E. Swift (Xcode)*

We will build our application using the swift programming language as this will make it compatible with any iOS device and we will use Xcode's built in features to help us design the user interface of the application. Swift was the natural choice for iOS development as it was developed by Apple. Additionally, the language is specifically designed to prevent errors and, as the name suggests, runs faster than both Objective-C and Python. We are using Xcode specifically because it provides previews of the application on any iOS platform, which will make debugging and testing a much smoother process.

## V. System Description

The project consists of a hardware, software, and mechanical component. The mechanical component will house the hardware and be responsible for the pouring of liquids and stirring of completed drink requests. On the hardware side, an *ELEGOO Mega* will be responsible for controlling the motors and pumps which regulate the dispensing of liquids and the position of a moving platform. The software will consist of an app on which users will place their drink requests and they will subsequently be transmitted to the *ELEGOO Mega* over Bluetooth Low Energy.

### *A. Enclosure*

The hardware and liquids will be encased in a wooden structure (see Figure 1). This structure will consist of two major components. This first of these components will contain the hardware for the Glass Movement mechanism and support both the Glass Movement and Stirring mechanisms. The second component will house the liquids from which each drink will be made. The liquids will be contained within the enclosure and will be accessible from the back of the structure via a sliding door. The labels on the liquids will be visible from the front of the structure through a plexiglass window, while the tubes and peristaltic pumps will be hidden from view as they will be kept above the section made visible by the plexiglass.

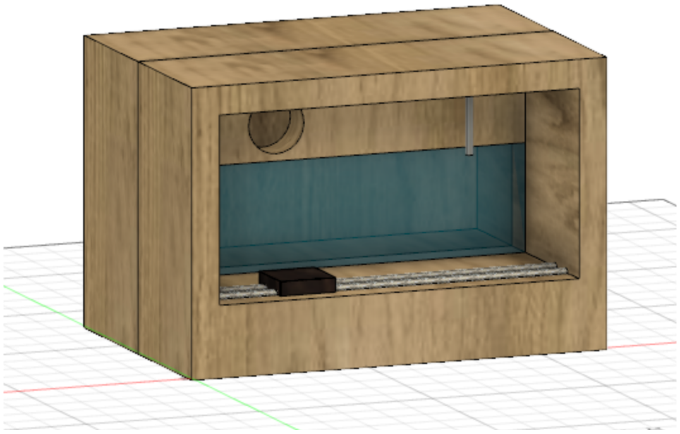


Figure 1: Enclosure CAD

### *B. Glass Movement*

Once the liquids have finished being dispensed into the glass, the glass must be moved under the stirring station to ensure the beverage is sufficiently mixed. This will be accomplished by having the *ELEGOO Mega* send commands to our *NEMA 17 Stepper Motor*. These commands consist of specifying the direction of rotation via the DIR input and whether or not the motor should rotate via the STEP input.

Those signals will propagate through a *DRV8825 Stepper Motor Driver* and then finally arrive at the *NEMA 17 Stepper Motor*. The stepper motor will be attached to a *6mm Timing Belt* and *Timing Belt Pulley*. These will be connected to a wooden platform and will allow the platform to move using *Linear Motion Rods*. The distance the platform must move will always be the same, so there is no need to calculate how far the platform has moved or where the platform currently is once we perform our initial testing and observe what amount of rotations result in the platform moving the correct distance. The connections between these components and the *ELEGOO Mega* can be found in Figure 2.

### *C. Liquid Dispensing and Stirring*

Immediately following the arrival of a drink request via bluetooth, the liquid dispensing process begins. The *ELEGOO Mega* program will process the request and compute which liquids need to be dispensed as well as their respective amounts. The *ELEGOO Mega* will then send an integer value between 0-255 along the GPIO port used to control a specific peristaltic pump. The output value 0-255 determines the flow rate of the liquid, with 0 indicating 0 mL/min and 255 indicating 100mL/min. This signal travels through a *TA7291 Bridge Driver*, included to help control the peristaltic pump's input voltage. Lastly, the signal will arrive at one of the four *12V Gikfun Peristaltic Pumps*, which will all be connected to a 12V DC power supply. We can use the signal history as well as the amount of time the pump spends dispensing to calculate the total amount of liquid dispensed using Equation 1.

$$Amount\ dispensed\ =\ \sum_{i=1}^{n} x_i \left(\frac{1}{255}\right) \left(\frac{5}{3} * 10^{-3} \frac{mL}{mS}\right)$$

Where
$x_i$ = the ELEGOO output as the $i^{th}$ millisecond of dispensing
$n$ = # of milliseconds spent pouring

Equation 1. Amount of liquid dispensed from a particular *12V Gikfun Peristaltic Pump*

The program will use this calculated amount from (1) to determine how much longer the *12V Gikfun Peristaltic Pump* should dispense the liquid according to the drink recipe. This information is also sent back to the phone application to keep track of how much liquid is left in each bottle.

Once all of the liquids have been dispensed into the glass, the platform will move to the mixing portion of the system. This consists of a *Continuous Rotation Servo* that will be

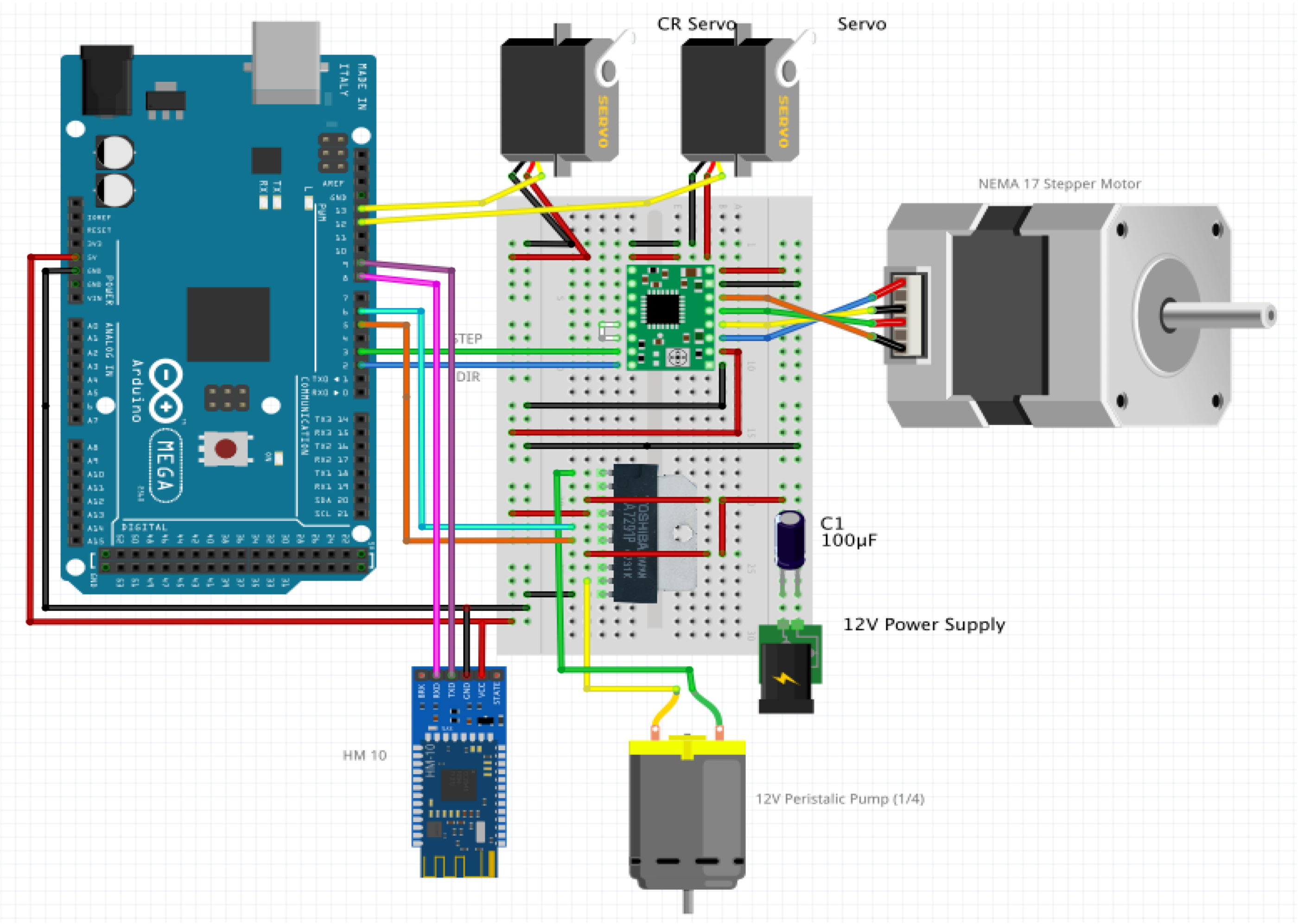


Figure 2: Hardware Breadboard Diagram

attached to a platform. This platform will be lowered and raised by a *SG90 9g Micro Servo*. The *Micro Servo* and *Continuous Rotation Servo* will be controlled via the Servo.h Arduino library. The connections between these components and the *ELEGOO Mega* can be found in Figure 2.

### D. *Bluetooth Communication*

We will be using Bluetooth Low Energy to establish the connection and enable communication between the phone application and the *ELEGOO Mega* via a *DSD Tech HM-10 Bluetooth Module*. This module is readily available and uses Bluetooth 4.0 technology and will establish all wireless data communication between the drink-making machine and the phone application. We will be using the communications to notify the machine when a user requests a drink and the type of drink requested. These communications will also notify the user when it is their turn to place their glass on the moving platform and when their drink has been created.

### E. *User Interface*

The user interface will have a login/register page to allow users to sign up or login. Following this step we will then have a page where users can connect to the nearest bar tending machine and see the available drink options for that specific machine. The user will then be able to choose the drink of their choice and the application will prompt them to place a certain glass size on the platform. Once the user satisfies this requirement they will click to begin the drink making process. Once the drink is complete the user will be prompted to take the glass off the platform. In the case that the machine has a queue of drinks yet to be completed the user will be able to choose the drink of their choice and then will have to wait in the queue until it is their turn to place the glass on the platform. The application will notify the user when it is their turn to place the glass on the platform. The user will also be able to track their drinking history through the platform. Each completed drink the user requests will be recorded and

## VI. Project Management

### A. *Schedule*

The Gantt chart is split into four sections: Deliverables, Logistics, Design and Implementation. Deliverables and logistics are shared among the team and as such all of us must be attentive in order to complete the requirements. The Design portion of the Gantt chart focuses mainly on the creation of the enclosure which will house the liquids and the hardware components and the building of the timing belt which will move the glass to the correct locations. Finally, the Implementation part of the Gantt chart covers the piecing together of all the hardware, software and mechanical components.

We have spent the past few weeks finalizing our design and will begin to build the project in the coming weeks. After familiarizing ourselves with the modules and components we are using we will begin piecing them together. We have scheduled the creation of the components such that we will be left with two weeks after the planned completion to test and ensure our system is working properly. This will allow us to refine parts of the system that may have been more difficult than anticipated or order parts should we need them.

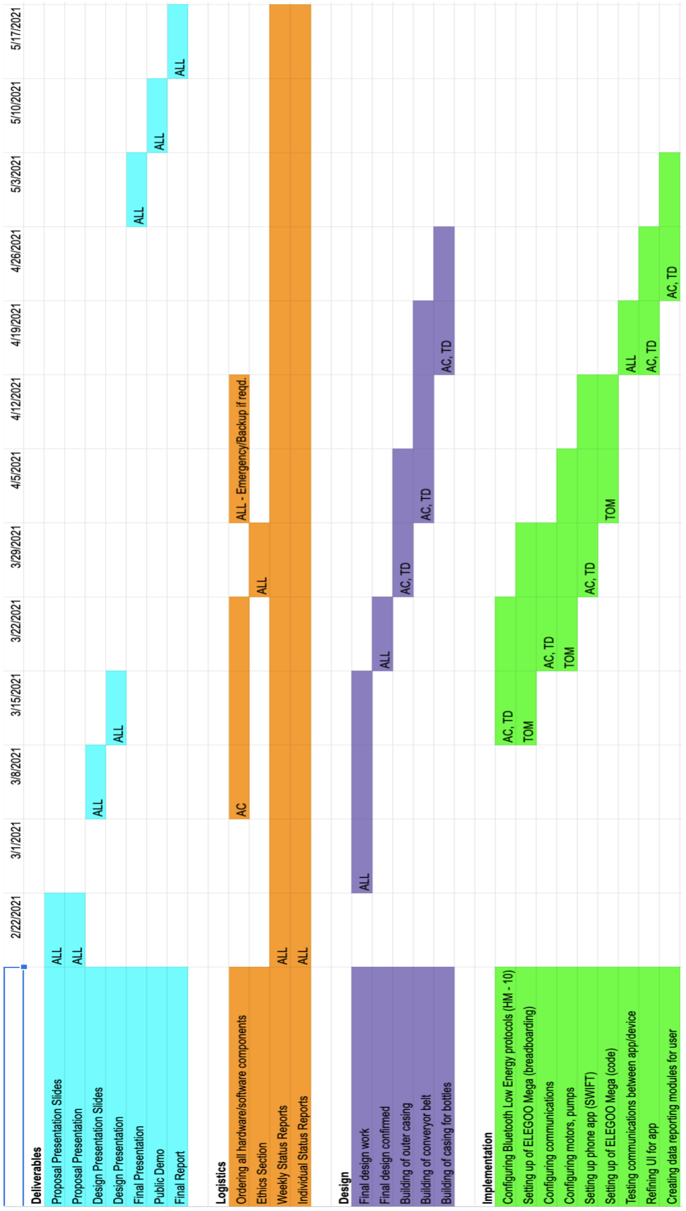


Figure 3: Gantt Chart Schedule

### B. *Team Member Responsibilities*

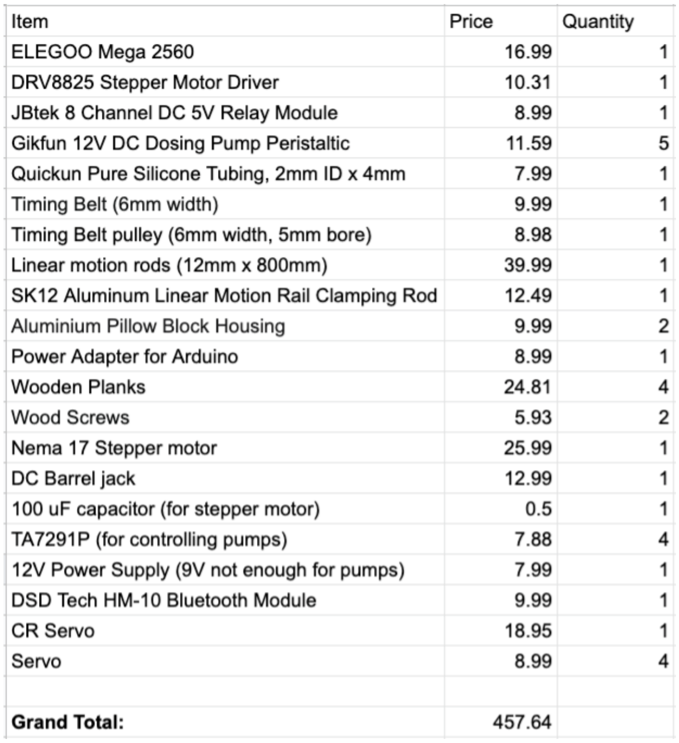
Thomas - Write *ELEGOO Mega* program to control all necessary hardware components. Assist Tyler with building the enclosure and integrating the enclosure with the hardware and working with the entire team to help connect our phone application with the *ELEGOO Mega*. Some app development if needed.

Tyler - Build enclosure to house hardware components and bottles of liquid. Work with Tom to ensure the structure is built in a way that allows the hardware components to reach where they need. Build iOS application and establish BLE connection between application and *DSD Tech HM-10 Bluetooth Module*.

Aryan - Handle all aspects of project management in terms of ordering materials/maintaining a master bill of materials and ensuring the team stays within budget as well as updating the Gantt chart. Ensure the team stays on track for deadlines and completion. Work with Tyler in building the iOS application and integrating the bluetooth connectivity module between phone and machine. Design layouts and create implementations for iOS application pages.

### C. *Budget*

Table 1 presents all the materials that were used in the creation of this project. We were given a budget of $600.00 and have spent just above $450.00 at the moment.



| Item | Price | Quantity |
|---|---|---|
| ELEGOO Mega 2560 | 16.99 | 1 |
| DRV8825 Stepper Motor Driver | 10.31 | 1 |
| JBtek 8 Channel DC 5V Relay Module | 8.99 | 1 |
| Gikfun 12V DC Dosing Pump Peristaltic | 11.59 | 5 |
| Quickun Pure Silicone Tubing, 2mm ID x 4mm | 7.99 | 1 |
| Timing Belt (6mm width) | 9.99 | 1 |
| Timing Belt pulley (6mm width, 5mm bore) | 8.98 | 1 |
| Linear motion rods (12mm x 800mm) | 39.99 | 1 |
| SK12 Aluminum Linear Motion Rail Clamping Rod | 12.49 | 1 |
| Aluminium Pillow Block Housing | 9.99 | 2 |
| Power Adapter for Arduino | 8.99 | 1 |
| Wooden Planks | 24.81 | 4 |
| Wood Screws | 5.93 | 2 |
| Nema 17 Stepper motor | 25.99 | 1 |
| DC Barrel jack | 12.99 | 1 |
| 100 uF capacitor (for stepper motor) | 0.5 | 1 |
| TA7291P (for controlling pumps) | 7.88 | 4 |
| 12V Power Supply (9V not enough for pumps) | 7.99 | 1 |
| DSD Tech HM-10 Bluetooth Module | 9.99 | 1 |
| CR Servo | 18.95 | 1 |
| Servo | 8.99 | 4 |
| | | |
| **Grand Total:** | 457.64 | |

Table 1: Master Budget

### D. *Risk Management*

We have identified several risk factors for our project. Firstly, we must ensure that there is enough liquid in each bottle for the requested drink to be dispensed. To rectify this issue we will be keeping track of the drinks requested, more specifically the amounts of each liquid used in the creation of said drinks, and comparing the amount used to the amount present to begin with. Upon replacing a bottle the user will be prompted to designate the amount of liquid present so the system is able to identify when it should be refilled.

Additionally, we expect multiple users to attempt to use the machine at once. In order to combat any possible problems that may arise as a result, we will establish a queueing system. The user will be notified upon being placed in the queue and receive another notification when it is their turn to place a glass on the platform.

## VII. Related Work

There are numerous DIY bartender bots in people's homes across the world. Many of these related projects are available online with corresponding videos, and in some cases even tutorials. Because of the plentiful design choices available while planning and building a bartender robot, no two home bartender robots are the same. For instance, several similar bartender robots have a more minimalist enclosure. Some robots take user input from phone applications while others have controls directly on the system. Some are even designed with the bottles flipped upside down.

Nearly every design we came across while performing our initial research had each beverage at different pouring locations. This meant that the platform the glass sat on had to move each time the requested drink required multiple liquids to make. We initially decided this approach, but then later realised that this additional movement was unnecessary given that our design uses plastic tubing and peristaltic pumps. Nonetheless, we did end up including a moving platform in our implementation and drew inspiration of how to do so from several of these projects.

A number of the designs we found while researching had the user place drink requests using a touch screen attached to the machine. We believe a phone application provides a more enjoyable experience and allows us to customize the user experience more easily. With an application we are able to provide the user with personalized drink history and statistics.

Initially we wanted to have a built-in user console on the machine for the user to interact with and choose their drinks. However, when researching for the project we came across a similar automated bartender which placed orders through a web application and was able to register different users. Hence we decided to have a phone application to connect and interact with the drink-making machine (over bluetooth) so as to have the same functionality. We also realized we can create interesting analytical insights to provide to our users on their drinking habits and include helpful metrics such as calories and alcohols consumed and deliver this information to the users to improve their overall experience.

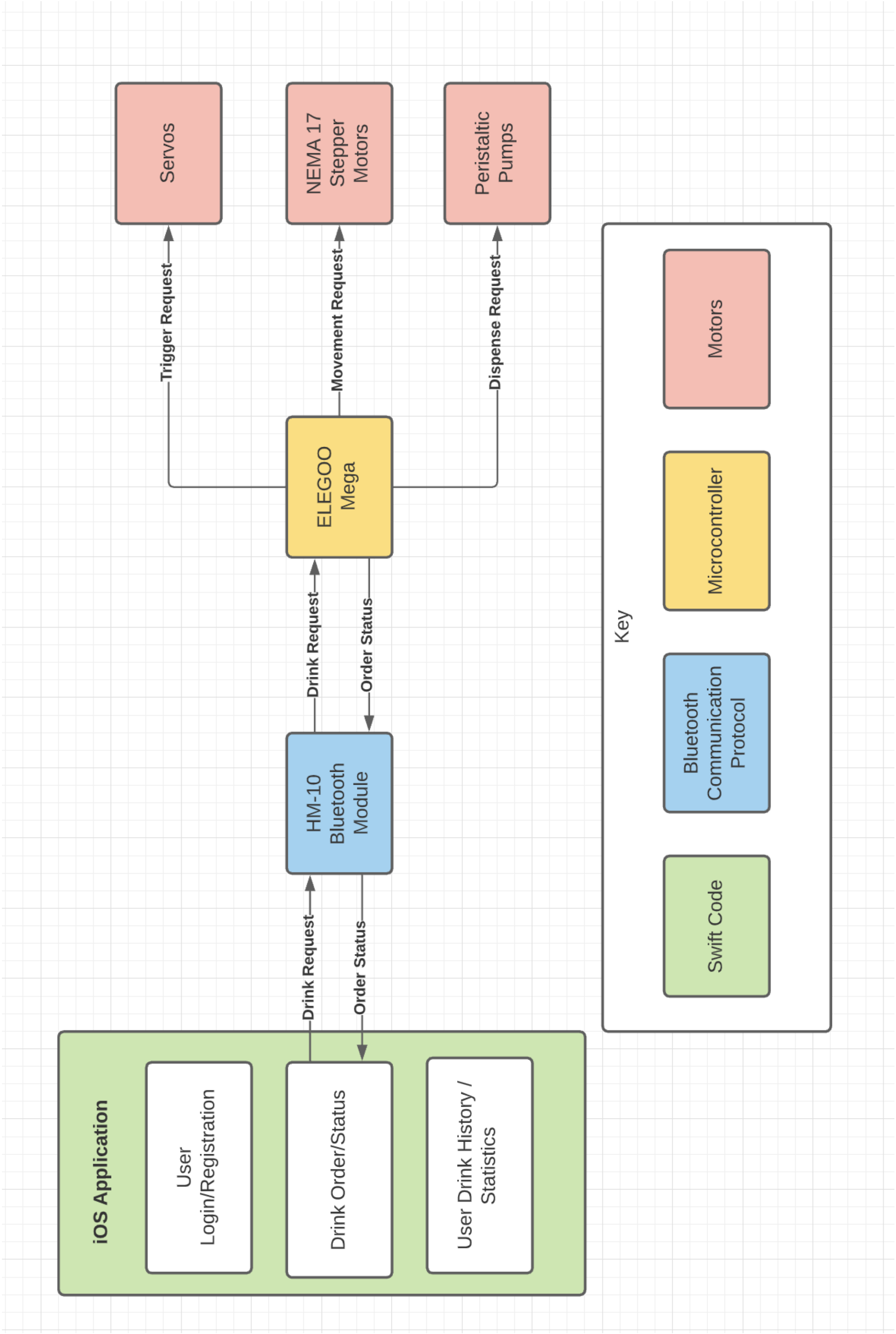


Figure 4: Software System Diagram

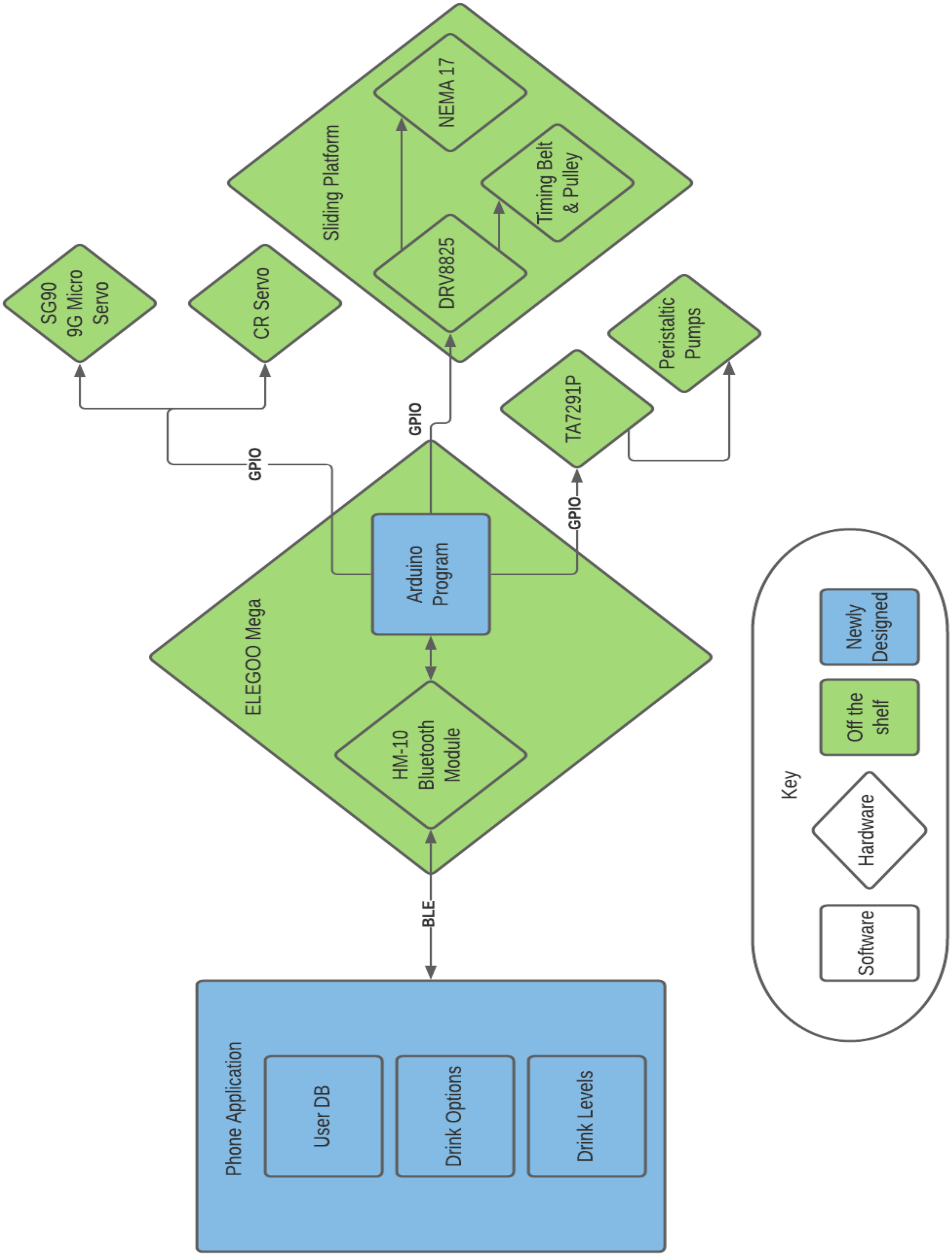


Figure 5 - System Block Diagram