SYSC3010 | Computer Systems Development Project

# The Mocktender Mocktail Mixing Machine

# Project Proposal


*Figure 1. Example of mocktail drinks to be made by the Mocktender machine [1].*

## Group L3-G8

Matt Reid, 101140593
Ethan Bradley, 101158848
Duncan MacLeod, 101160585

TA: Roger Selzler

February 8th, 2023

# Table of Contents

# 1 Introduction

The purpose of this document is to respond to the SYSC 3010 course requirement to produce a project proposal for our group, L3-G8's project, The Mocktender Mocktail Mixing Machine. It is an IoT (Internet of Things) automated mocktail mixing machine. This document outlines the motivation, objectives, and goals behind the system as well as the project flow including use cases, a deployment diagram, and a flow chart.

## 1.1 Background

A mocktail is a non-alcoholic drink that captures the essence of a cocktail without the downsides of alcohol consumption. From work offices where you want to stay sharp while having fun drinks for lunch to wanting a healthier alternative to drinking cocktails at home, there are lots of reasons to enjoy a mocktail. As more people look towards health and wellness trends, the number of people drinking non-alcoholic beverages is rapidly increasing. A NielsenIQ study in October 2021 found that non-alcoholic beverage sales had gone up 33.2% in 12 months to a total market of $331 million [2]. In the United States, 30% of people who are of drinking age do not drink alcohol [3], and according to a 2021 survey by Statistics Canada, 1 in 5 Canadians said they have been drinking less than before the COVID-19 pandemic [4].

The at home cocktail drinking market has also grown significantly during the COVID-19 pandemic, with one in four people now spending more on home drinking than they did before [5]. The cost of a bartender for a private even can cost upwards of $50 per hour [6]. To provide the ease of having drinks at home or private events without needing to buy equipment and learn how to mix them yourself or paying a bartender, machines such as *The Bartesian* allow the creation of at home cocktails, mocktails, and margaritas using fillable drink containers and pods [7]. This device allows you to fill containers, set strengths, and creates a wide array of drinks, however it is restricted to drinks that are set by the pods and does not allow recipe creation or remote container liquid level tracking. Our solution will provide the ability to use a remote web interface to create new recipes, view real-time liquid levels, and admin access for usage at events.

## 1.2 Motivation

For mocktail drinkers that want to enjoy a drink at home, the time spent measuring and mixing the drinks can be prohibitive. For private event managers, the cost of hiring a bartender to make drinks to impress their client can be enormous. To help reduce the time strain of making mocktails, and remove the need to hire a bartender, *The Mocktender Mocktail Mixing Machine* is an IoT machine perfect for the at home mocktail consumer, and an event manager looking to impress their guests. For at home users, the recipe creation system allows users to specify their own recipe and measurements to make them the perfect mocktail in seconds with minimal mess. For event managers, the web interface allows remote tracking of liquid amounts to know when it needs filling, and it removes the cost of hiring a bartender to create mocktails.

## 1.3 Project Objective

The objective of the project is to create an automated mocktail mixing machine that provides a web interface to allow real-time liquid level monitoring, and recipe creation. It replaces the need for at home mixing equipment, and bartenders at events to eliminate the inflated cost of making mixed drinks. It will provide timely mocktails from specified recipes, allowing for easy access to healthier drinks and less time spent mixing with more time spent drinking.

### 1.4 Specific Goals

To create an easy to use and fully functional mocktail mixing machine, The Mocktender has the following goals in order of descending priority:

**Automatic Liquid Mixing:** The machine will automatically mix liquids together based on defined recipes in a database. The database will contain recipes that specify liquid percentages, and the Raspberry pi will use a pump to dispense the correct amount of each liquid into a cup.

**Remote Liquid Level Monitoring:** The machine will allow users to view the current liquid levels through an online web interface. An ultrasonic sensor will be used to provide real-time updates of the liquid levels. Notifications will be sent to registered users that liquid levels are low.

**Recipe Creation and Storage:** The machine will allow users to create new recipes in the web interface and store them in the database. Recipes consist of liquid names and the percentage that should be put in the cup.

**Remote Dispensing Access:** The machine will allow users to access the drink pouring menu remotely on the web interface. GUI (Graphical User Interface) buttons will be available to dispense selected drink recipes.

**Administrative Access:** The machine will allow an admin user to login to the web interface and lock the machine from use or modify the liquid types that are available in the machine. There will also be a clean function allowing for a liquid to be emptied from the machine and water to be used to flush out the pipes.

## 2 System Design

### 2.1 System Overview Diagram

The Mocktender consists of three main subsystems with an additional cloud infrastructure, which is illustrated by Figure 2 below. It may appear confusing at first that Figure 2 includes two instances of a UI and a database. The local services function as an endpoint for the user to interact with directly. Through the local UI, an owner can configure Keycloak authentication to allow access to the cloud infrastructure. The purpose of the cloud UI and database is for security and reliability purposes. For example, the cloud infrastructure would allow the user to use the web interface even when the Mocktender is not active and not running anything locally. One function of the cloud UI is the ability to activate the Mocktender and interact with it remotely.

The three subsystems not on the cloud are each managed by a different Raspberry Pi. The first Raspberry Pi is the “System Controller,” which is responsible for hosting the local infrastructure and facilitating communication between subsystems. Since the System Controller hosts the user endpoints, it effectively acts as an adapter proxy between the other entities which include the user, the local UI and database, the cloud, and the other subsystems. Aside from the web infrastructure, the Mocktender will also be physically equipped with an LCD display and a joystick for the user to operate the machine directly instead of through a web UI. The LCD display will be controlled by a different Raspberry Pi called the “Output Controller.” The Output Controller is responsible for controlling/managing every output peripheral in the system, which comes down to the LCD display and the pumps for pouring the liquid. The last subsystem is called the “Input Controller” which manages the inputs to the system and feeds any significant events to the System Controller.

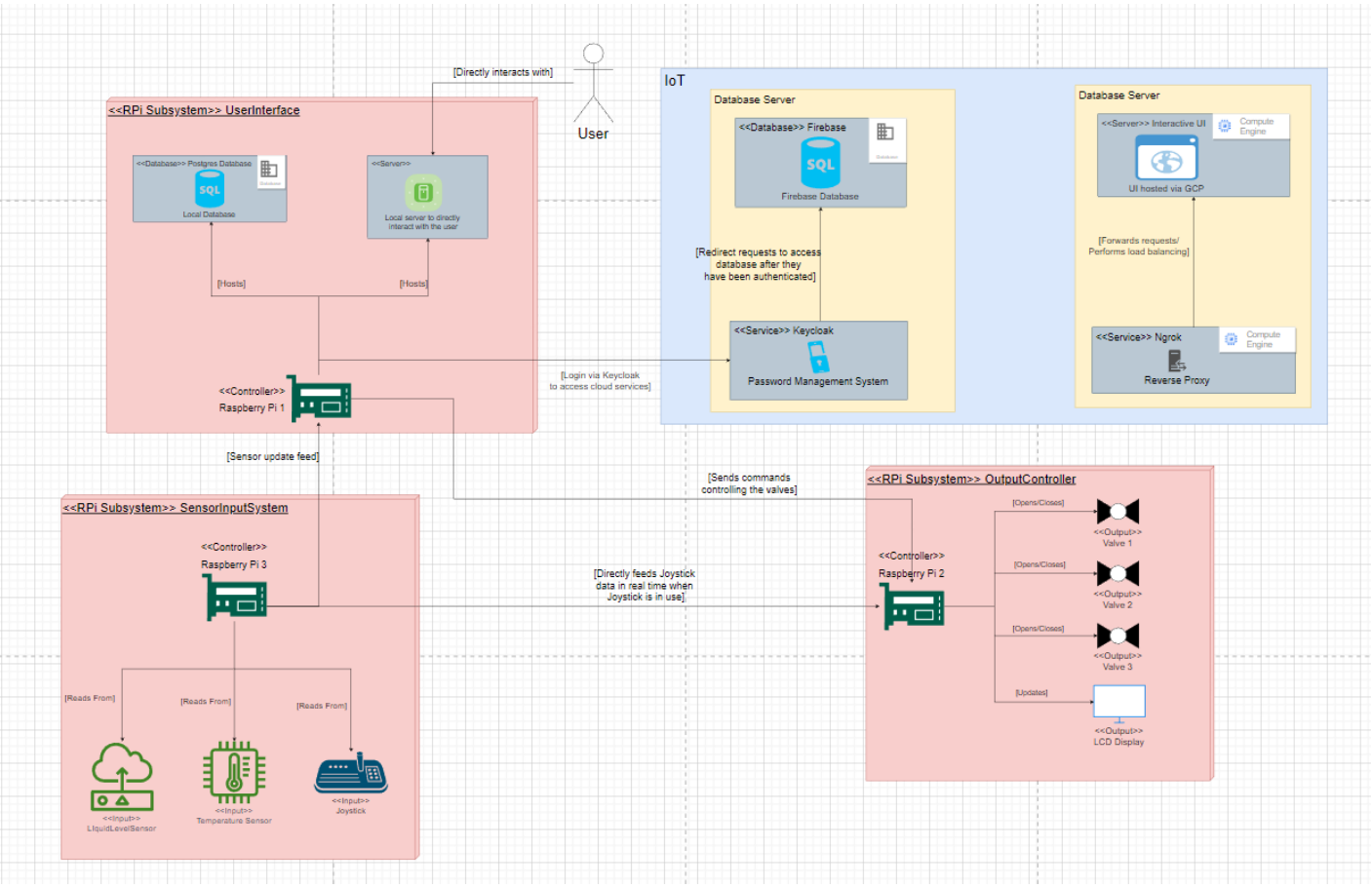


*Figure 2. Deployment diagram of The Mocktender Mocktail Mixing Machine*

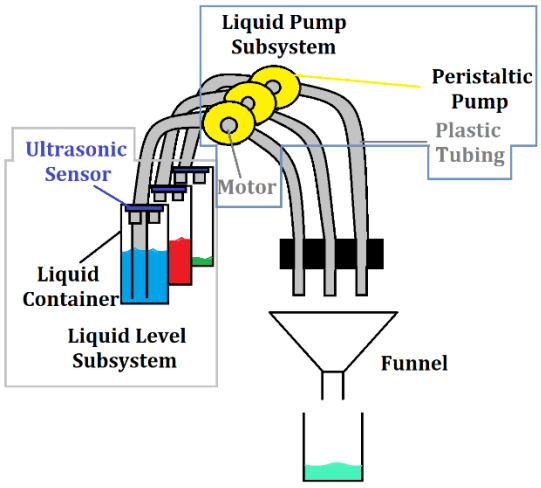


*Figure 3. Rough Sketch of Internal Systems*

### 2.1.1 Communication Protocols

The liquid level sensors and the pump controllers can be communicated with using GPIO as they operate on digital signals. Similarly, the LCD display can also be controlled using GPIO. The pump controllers will use stepper motors, and as such require a stepper motor driver to operate properly. The

temperature sensor on the SenseHat can be read from using the I2C bus on the Raspberry Pi. Manual controls on the frame of the machine also produce digital signals, such as button presses. The Joystick produces analog signals, so it will be connected to the computer device using an Analog to Digital converter.

A Raspberry Pi node will upload to and read data from the real-time database. This node will communicate this data between nodes using an SPI connection, allowing for a concurrent 2-way communication stream. The Raspberry Pi will issue and receive commands from the other devices on the SPI bus, allowing the system to react quickly in real-time in response to unforeseen stimuli.

## 2.2 Component Details

The Mocktender will by comprised of 5 main systems. A local and cloud-based database will be used to store recipes that users create on the front-end User Interface. The front-end will also allow them to login and monitor the liquid levels in the systems and will allow them to be notified when the Liquid Level sensor system detects that the amount of fluid remaining in a container is running low. The user can use the Manual user input system or the Front-end User Interface to activate the Drink Pump system to dispense their drink.

### 2.2.1 Real-time and Local Databases

The first component proposed is the database model. As seen in Figure 2, the Mocktender will have both a local instance and a cloud instance, which will be synchronized. In the case that changes are made to the cloud database when the local database is offline, a change log file is generated and stored in a cloud VM, and the location of the log file is recorded in the database table “DatabaseChangeLogRef.” Permission to view/edit certain tables will be managed via Keycloak. The database schema can be split into a few categories, which are colour coded as shown in Figure 4. Database tables in red store data that is primarily used for the Mocktender’s main function, creating mixed drinks. Database tables in purple and the Keycloak table are related to user information and authentication. Tables in green are used for synchronizing the local and cloud instances.

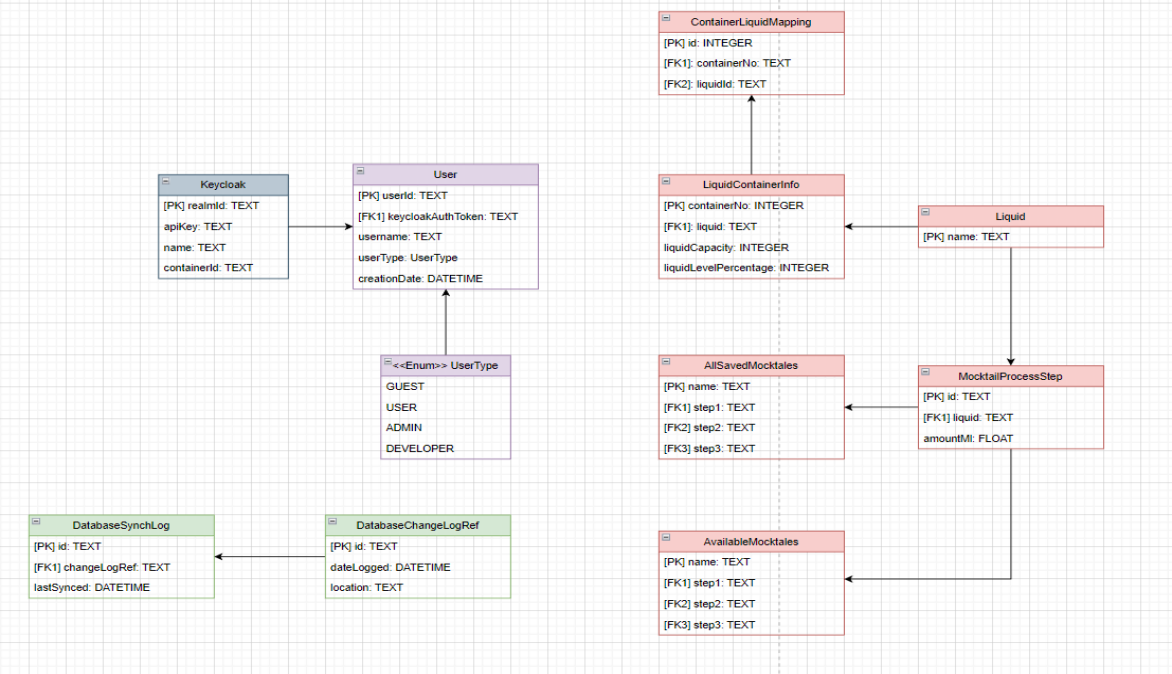


*Figure 4. Database Schematic of The Mocktender*

### 2.2.2 Drink Pump System

The second component proposed for the Mocktender is the Drink Pump System. To have precise control over how well a drink is mixed, each of the 3 liquids added to the drink needs to have its own liquid pump module. A liquid will be stored in a container until GPIO signals are sent to a stepper motor, which will be used to implement a liquid pump and can be see below in Figure 5. As even small motors are capable of drawing hundreds of milliamps [8], it is essential that there is sufficient power distribution to ensure that the pumps do not become unresponsive. This is modelled in Figure 5 as an additional power bank connected only to the stepper motors.

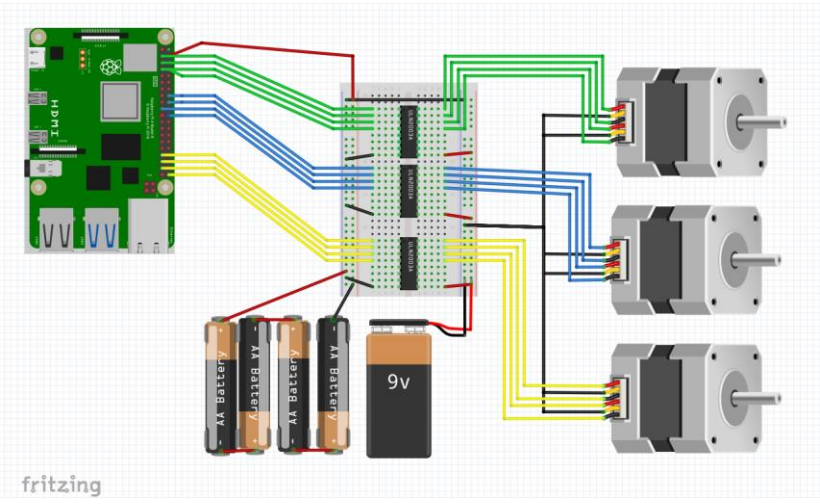


*Figure 5. Drink Pump schematic view of the OutputController subsystem*

### 2.2.3 Manual User Input System

The third proposed system is the Manual User Input system. As the Mocktender is a drink mixing machine, it should not require an internet connection to be used. As such, a physical input system will be included in the system. Figure 6 below shows a computer device interfacing with an LCD screen as well as a rotary encoder with an internal button. Options will be shown on the LCD for a user to select their drink preferences, and the encoder can be used to confirm their selection. If the user has credentials to input, their recipes will be saved to a local database in the system which will be uploaded to the cloud database when a connection becomes available. The LCD module can be controlled using GPIO or I2C depending on the hardware available. The rotary encoder does not require analog-to-digital conversion which makes it the ideal choice for hardware input with a Raspberry Pi.

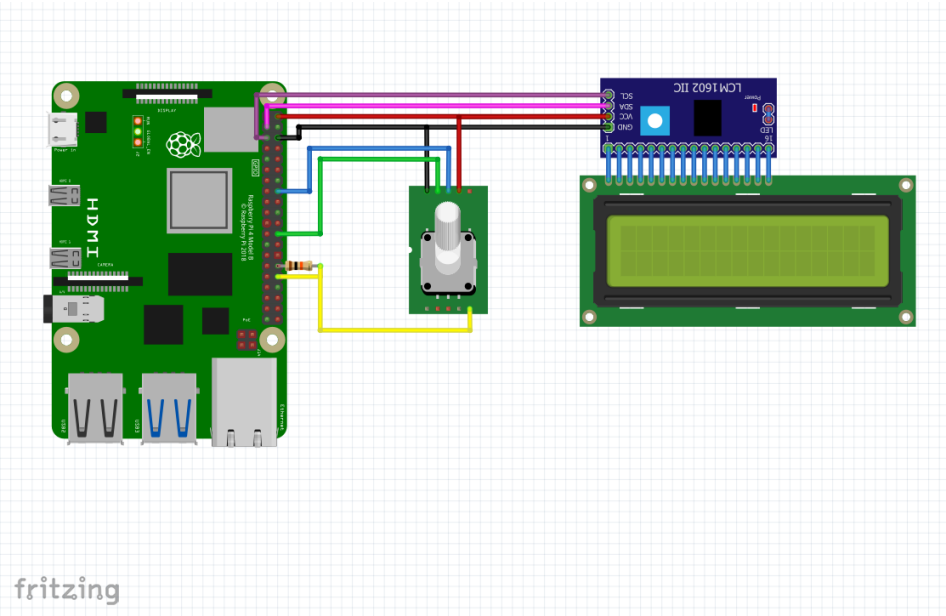


*Figure 6. Schematic view for the manual user input system*

### 2.2.4 Liquid Level Sensor System

The fourth proposed system for the Mocktender is the Liquid Level Sensor. The system will use ultrasonic sensors to detect the how far away the level of a liquid is from the lid of a container. The ultrasonic sensors can be controlled using GPIO and the data measured from the sensors will be used to graphically display the liquid level on the front-end graphical user interface. The computer reading the sensor data will communicate with the computer interfacing with the real-time database to ensure that there is always updated liquid level data on the front-end. Figure 7 below shows how the ultrasonic sensors will connect to a computer device. Each Ultrasonic sensor requires 2 pins, 1 for triggering the sensor and 1 for reading the data as a digital data stream.

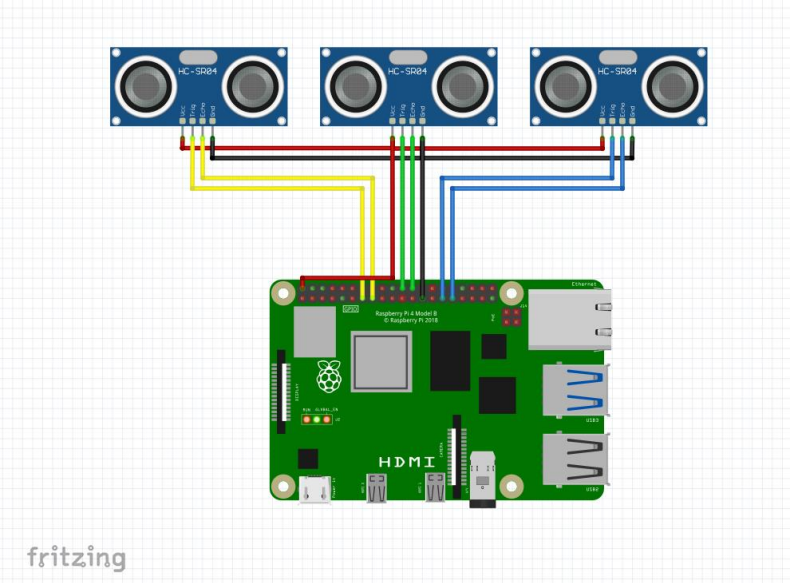


*Figure 7. Schematic view for the liquid level system, managed by the Input Subsystem*

### 2.2.5 Front-end User Interface

The fifth component being proposed for the Mocktender is the Front-end user interface. The Front-end will be hosted on a webserver on one of the Raspberry Pi's in the system and will be the main source of interaction between the system and the end-user. It will be built using React web design principles to provide an easy-to-use interface to the user. Figure 8 below shows a model of a potential use-flow of the front-end, using links to navigate to different webpages that organize related tasks into one spot. The user will be able to view their Mocktender's current liquid levels in real-time and create Recipes based on the current combination of liquids in the system. The user will also be able to subscribe to notifications using their preferred method (e.g., Email, SMS, Push notifications). One user's information will be hidden from another user by requiring them to login with a password.

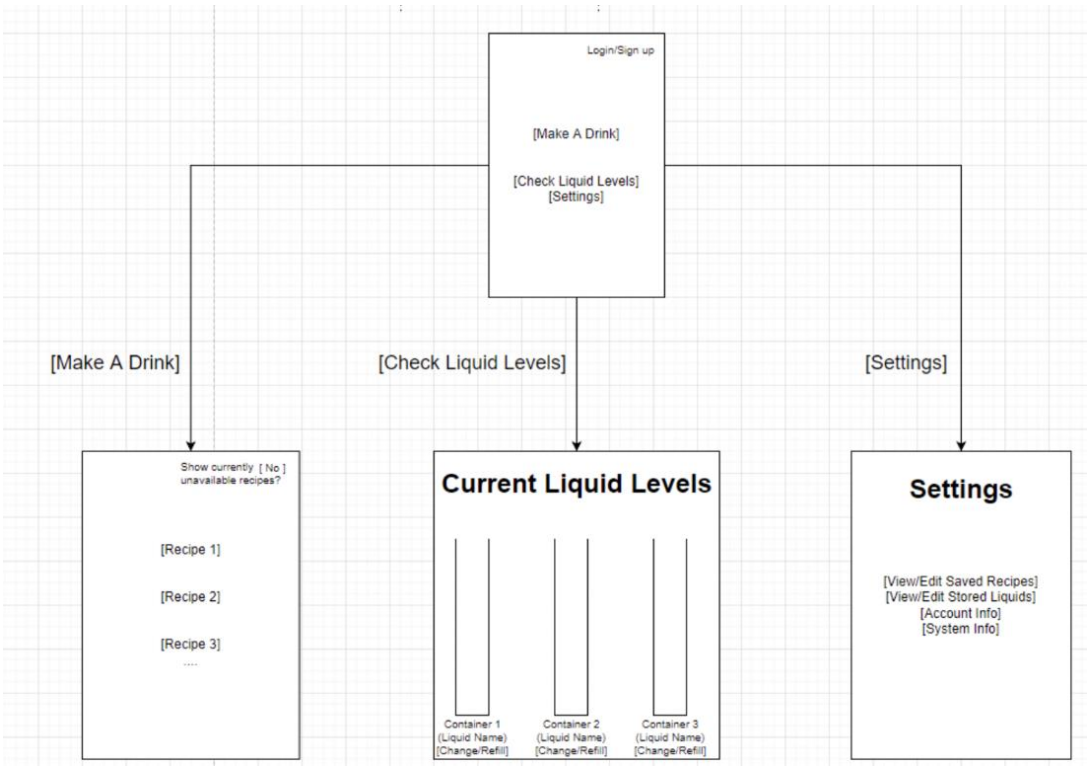


*Figure 8. Preliminary core page layout for the user interface*

## 2.3 Use Cases

### 2.3.1 Use Case 1: MakeDrink

The most basic use case demonstrating the process to make a drink is the MakeDrink use case. This can be done through the web interface and manually through the machine itself. Figure 9 demonstrates the use case of making a drink via interacting with the machine directly. The Mocktender will be equipped with a simple interface consisting of an LCD display and a joystick. The LCD display will show all the saved Mocktails that are currently available for pouring. The user will be able to use the joystick to select the drink they want to make. When the user confirms their choice, the SystemInterface will pour the drink by communicating with OutputController by signalling which pumps to stop and start. When the drink is done, the LCD display will say “Enjoy.”

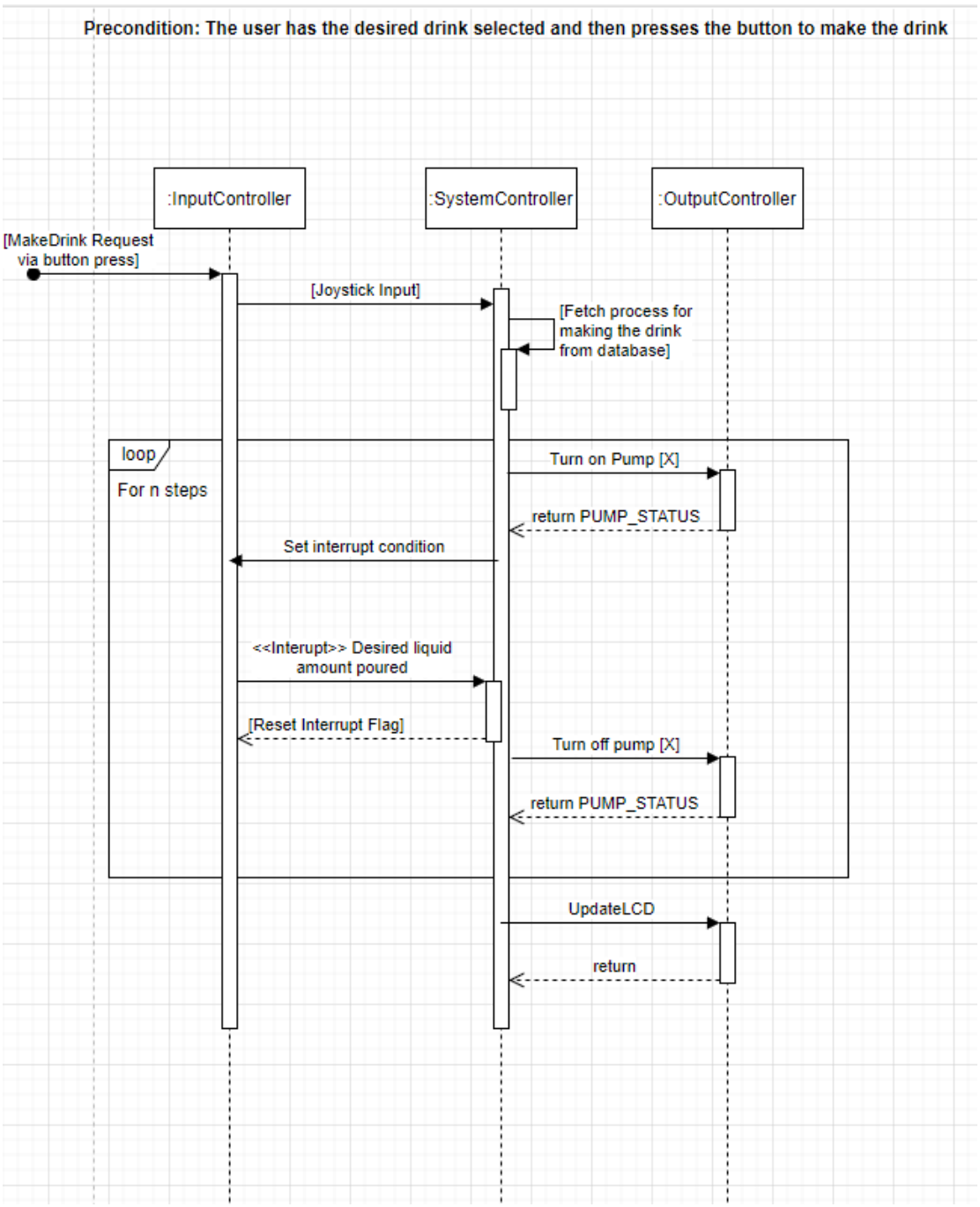


*Figure 9. Sequence diagram for MakeDrink use case*

### 2.3.2 Use Case 2: MakeRecipe

The MakeRecipe use case outlines the basic flow of the process to create a new mocktail recipe, which is done through the web interface. Figure 10 shows the process of interacting through the web interface to create a recipe. The user navigates through the web interface by going to “Settings” ->” Update UI” ->” View/Edit Saved Recipes.” From there, the user will be able to specify a sequence of drinks to pour to create their new mocktail. Once saved, the new mocktail is now available to make.

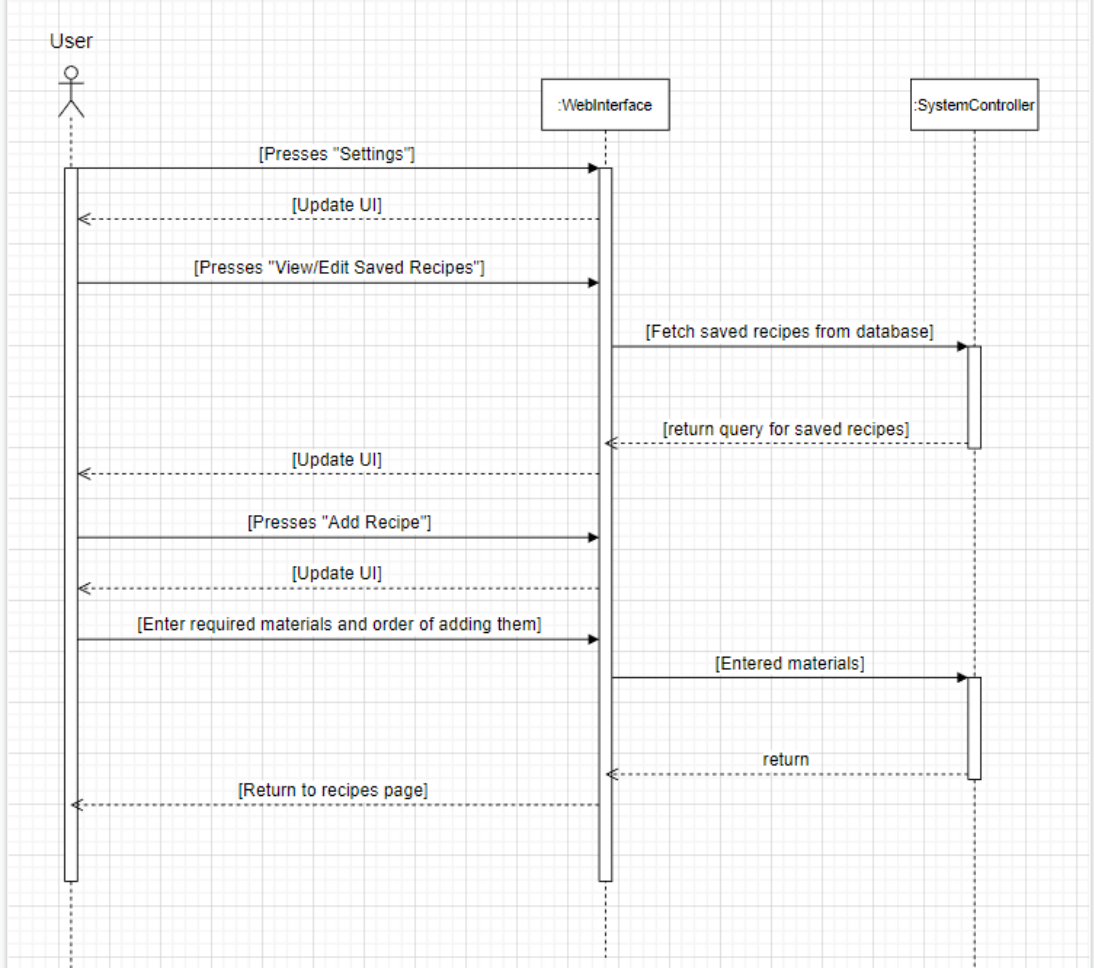


*Figure 10. Sequence Diagram for MakeRecipe use case*

### 2.3.3 Use Case 3: CheckLiquidLevel

The Mocktender is equipped with sensors to track how much liquid is left in each container, which the user can view through both the physical display and the web interface. The web interface is more comprehensive and easier to use, so Figure 11 demonstrates the use case for checking liquid levels via the web interface. The user must be an admin to view the liquid levels. In practice, an “admin” account would correspond to someone or a group of people who own the machine.

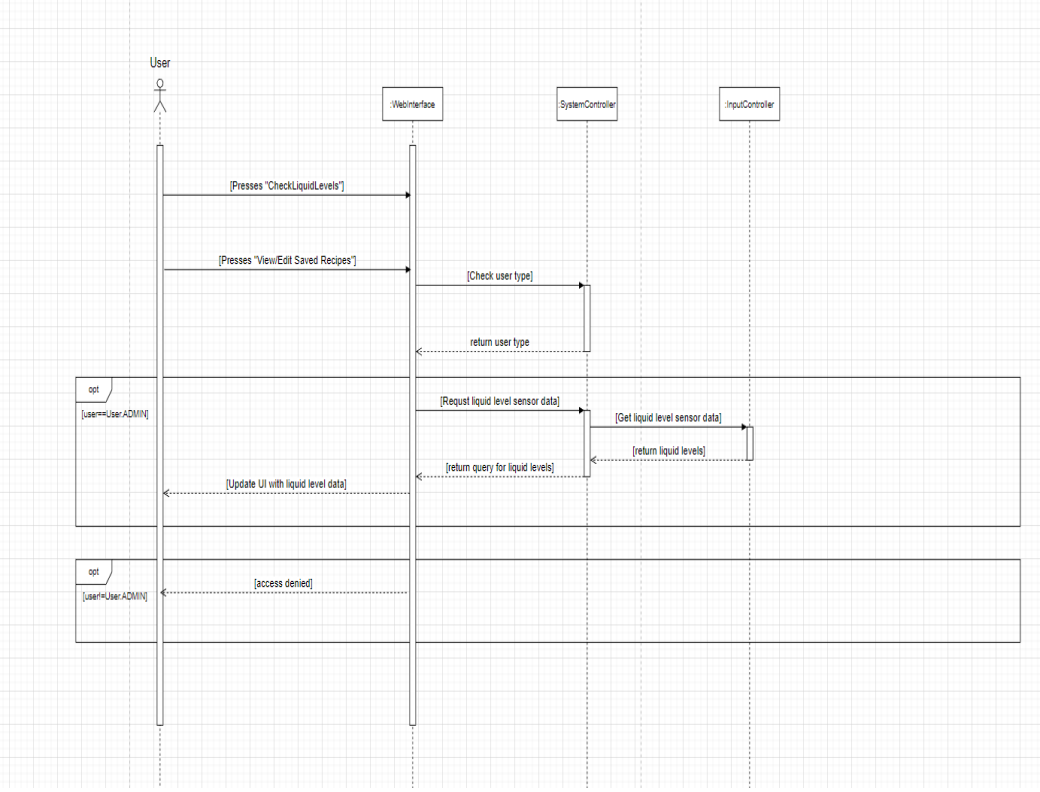


*Figure 10. Sequence diagram for CheckLiquidLevel use case*

# 3 Work Plan

## 3.1 The Project Team

*The Mocktender* Project team, L3-G8, is composed of Matt Reid, Ethan Bradley, and Duncan MacLeod. Matt is interested in communications between devices and has strengths in documentation and backend database development. Ethan likes to design and build circuits and has strengths in creating schematics and implementing hardware designs. Duncan enjoys Android app development and has strengths in frontend development and user interfaces.

### 3.1.1 Roles and Tasks

Table 1. Assignment of major tasks to team members

| **Major Task** | **Primary** | **Secondary** |
|---|---|---|
| Ultrasonic sensor water level data collection | Ethan & Matt | Duncan |
| Achieve communication between the three Raspberry Pi devices | Matt | Ethan |
| Create frontend user interface with admin login | Duncan | Matt |
| Construct machine frame and component housing | Ethan & Matt | Duncan |
| Create backend Firebase real-time database | Matt & Duncan | Ethan |
| Design circuit for electronic devices | Duncan & Ethan | Matt |
| Create pump mechanism for mixing drinks | Ethan | Matt |
| Integrate sensor data and pump mechanism with database | All Members | N/A |
| Connect all systems together | All Members | N/A |
| Perform full testing on all systems | All Members | N/A |

Table 1 outlines the assignment of major tasks of building *The Mocktender* machine to members of the team. The primary assignments were selected based on the strengths of team members and what they are interested in learning during the SYSC 3010 course.

For the sensor data collection, Ethan is experienced with connecting sensors, and Matt wanted to learn more about sensor connections. Matt will work on communication between devices and Duncan will work on the frontend as they play into their strengths. For building the machine frame and component housing, Ethan and Matt were selected as they both has experience with building frames for projects.

The database will be worked on by Duncan as he has previous work experience and Matt as he has not worked with databases previously and is interested in learning. For the pump mechanism, Ethan was chosen as he is strong with creating circuits and mechanisms so is a perfect fit for making pump mechanisms with stepper motors. Ethan is working on circuit design as it is his strength and Duncan is joining as it is a major learning interest.

The rest of the tasks including integration with the database, system connection, and testing are all being worked on by all team members due to the tasks being centered on the entire system at once and is beneficial for all team members to understand.

### 3.1.2 Teamwork Strategy

As a team we will work together in an Agile sprint-based workflow to ensure that deadlines are followed, and tasks are organized. We will have a meeting every week to discuss the last weeks tasks and completions and identify the new sprint items for the upcoming week. If there are any items not completed from the week before, the reasons why will be discussed, and it will be completed as soon as possible. Our group will communicate both in person for sprint meetings and throughout the week on Microsoft Teams.

When code is completed, it will be reviewed by the secondary team member assigned to each task in Table 1 using GitHub pull requests. GitHub will be used for all the code for our project as it allows for continuous sharing source code between members and organization of changes and issues that were made/solved. Throughout the development process, any issues identified will be added to GitHub's issue tracking system so that they are organized and can be referenced during the weekly sprint meetings.

### 3.1.3 What we will need to learn

*The Mocktender* implements technologies with various levels of understanding by our group members. During the project timeframe, we will need to learn front-end development and design, serial communications between devices, designing circuits with sensors/actuators, and integrating mechanical systems with electrical components.

Front-end development and design will be used to create the web interface. It will require learning JavaScript and React web design principles to create a functional and easy to use GUI.

Implementing a cloud-based infrastructure will require familiarization with the Google Cloud Platform and its functionalities, as well as the third-party services that typically accompany it, such as Keycloak and Ngrok.

Serial communications between devices will be used to send sensor data to the user interface subsystem and send recipe data to the pump control subsystem. It will require building on skills in using SPI communications between devices.

Learning to design circuits with sensors/actuators will be needed for building the ultrasonic water level sensor and transferring liquids using pumps. This will require learning soldering and integrating ports required by the ultrasonic sensor and pump motors.

For integrating mechanical systems and electrical components, we will need to learn protocols for sending electrical data to the mechanical systems.

## 3.2 Project Milestones

To structure the implementation and design of the Mocktail Machine, we have broken up our timeline into 7 milestones which can be seen in the table below.

Table 2. Project milestones with deadlines.

| Milestone # | Name | Description | Date |
|---|---|---|---|
| 1 | Uploading and Fetching Data | Computer devices can read from and write data to a real-time database independently of each other, which can then be accessed and displayed on an external device. | February 21st |
| 2 | Communication Between Nodes | Serial communication established between computer devices (I2C, SPI, etc.) to transfer data stored locally between nodes and issue commands. | February 28th |
| 3 | Measurement and Conditional Actuation | Liquid level of a fixed container can be measured accurately using sensors can uploaded to the database. Data can be used to turn on an LED when a threshold is reached. | March 3rd |
| 4 | Exposing Data to the World | Two additional containers with liquid level measuring will also have their data uploaded to the database. The liquid level will be displayed as text on a webserver | March 7th |
| 5 | Hardware Assembly | Assembled liquid dispensing system utilizing peristaltic pumps. Liquid level on webserver should be updated (as text) in real-time. | March 14th |
| 6 | Graphical User Interface | Webserver displays the volume of the liquid containers as an animation. Users can navigate between different webpages on the server. | March 21st |
| 7 | Controlling the Hardware System with UI | Webserver component can control all 3 liquid pumps implementing the drink mixing. Users can login and save customs mixtures locally and in the real-time database. | March 28th |

## 3.3 Schedule of Activities

Figure 12 below outlines the timeline of milestone activity completion as a Gantt chart over the 2 months working on our project. The milestones are outlined on the right, and the black lines represent when the activities will be worked on.

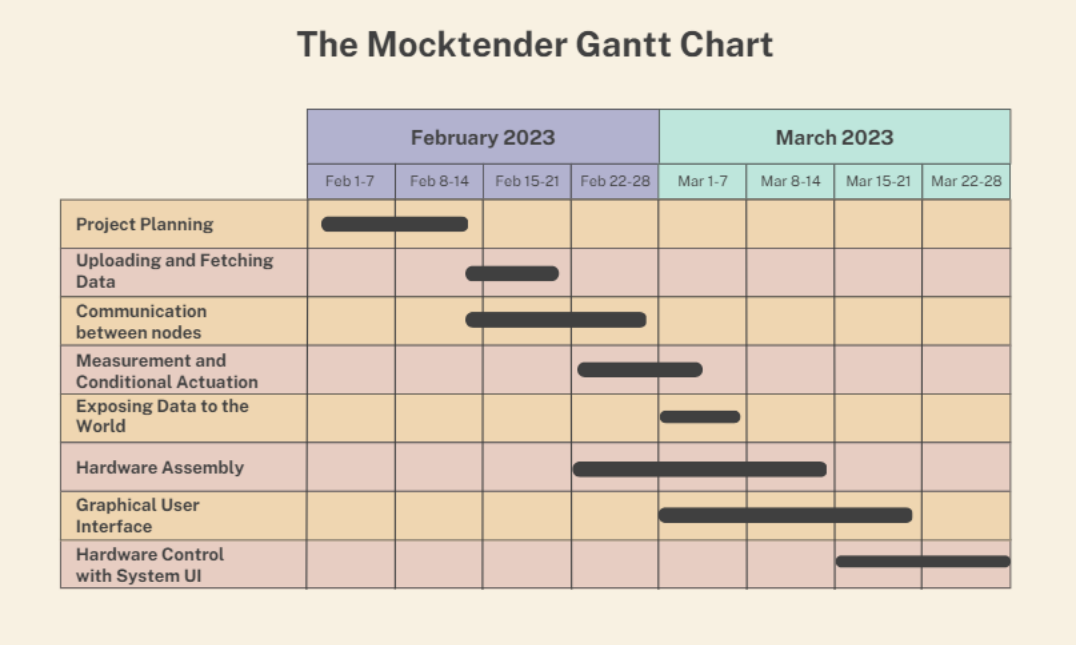


*Figure 11. Project Gantt Chart*

# 4 Project Requirements Checklist

Table 3. Project requirements and how they are met.

| **Requirement** | **Is it included in our project?** |
|---|---|
| Is there a computer per student in the group. | Yes, one Raspberry Pi will monitor the liquid levels, one Raspberry Pi will perform the liquid pumping, and one Raspberry Pi will provide the User Interface. |
| Is there at least one RPi in headless mode? | Yes, all 3 of the RPis are in headless mode. |
| Is there at least one hardware device per student in the group? | Yes, there are pumps, ultrasonic sensors, input buttons, and lcd display, and a temperature sensor. |
| Is there an actuator? | Yes, the pumps are actuators. |
| Is there a feedback loop? Interaction between input/output devices. | Yes, the pump system output device interacts with the user interface input. The liquid level system also outputs sensor data in a loop. |
| Is there a database? | Yes, there is a database to store the liquids in the machine and their levels, and the drink recipes. |
| Does the computer hosting the database have other responsibilities? | Yes, The Raspberry Pi hosting the database also hosts the web interface and takes user input to send to the other Pi devices. |

| | |
|---|---|
| Does the database contain at least two tables? | Yes, there will be a table for liquid levels and a table for drink recipes. |
| Is there a periodic timing loop? | Yes, a periodic timing loop will query the ultrasonic sensors in the liquid containers to obtain real time liquid levels for the web interface. |
| Is there some processing or analysis of the IoT data read? | Yes, the data from the ultrasonic sensor is used to find the volume of the liquid in the containers by processing how much of the container is empty. There will also be analysis showing which liquids are the most popular and how often they are used. |
| Are notifications sent (SMS, email, push notifications)? | Yes, notifications that liquid levels are low will be sent, and if the web interface is used, a notification will be sent that the drink is ready. |
| Is there at least one GUI running on a computer? | Yes, the web interface will be a bidirectional GUI, displaying liquid levels, and taking user input to create new recipes, or dispense drinks. |

# 5 Additional Hardware Required

Table 4. Additional hardware required for *The Mocktender* machine (excluding frame and housing)

| **Component Required** | **Use Description** | **Quantity** | **Cost** |
|---|---|---|---|
| Stepper Motor | Control pump mechanism | 2 | $15.18 |
| Stepper Motor Driver | Control pump mechanism | 2 | $2.48 |
| Ultrasonic Sensor | Liquid level detection | 1 | $10.01 |
| Plastic tubing | Used to transport liquid | 3m | $13.59 |

The total cost of hardware components that are not already owned or available to our group for free, as shown in Table 4 is $41.26.

# 6 References

[1] Colourful drinks, abstract, new, HD wallpaper. Peakpx. (n.d.). Retrieved February 2023, from https://www.peakpx.com/en/hd-wallpaper-desktop-gfuhh.

[2] Cox, K. (2021, October 12). Nielseniq: Newly Released Low & Non-alcoholic beverage trends. Wine Industry Advisor. Retrieved February 2023, from https://wineindustryadvisor.com/2021/10/12/nielseniq-newly-released-beverage-trends.

[3] Mocktails are gaining popularity. will meetings and events catch the trend? Meetings Today. (n.d.). Retrieved February 2023, from https://www.meetingstoday.com/articles/143402/mocktails-gaining-popularity-will-meetings-events-catch-trend.

[4] Turcotte, S. (2023, January 12). Here's why alcohol-free drinks are gaining in popularity. CTV Kitchener. Retrieved February 2023, from https://kitchener.ctvnews.ca/here-s-why-alcohol-free-drinks-are-gaining-in-popularity-1.6227268.

[5] Wells, H. (2021, June 14). Home drinking stirring up the cocktail market - CGA. CGA. Retrieved February 2023, from https://cgastrategy.com/home-drinking-stirring-up-the-cocktail-market/.

[6] What's the cost to hire a private bartender? National Bartending School. (2020, January 16). Retrieved February 2023, from https://www.nationalbartenders.com/bartender-school-questions/how-much-does-it-cost-to-hire-a-bartender-for-a-private-party/#:~:text=A%20bartender's%20basic%20party%20rate,The%20party's%20geographical%20location

[7] Bartesian.com official site - premium cocktails on demand. Bartesian. (n.d.). Retrieved February 2023, from https://bartesian.com/en-ca.

[8] Ada, L. (n.d.). Adafruit 16-channel PWM/Servo Hat & Bonnet for raspberry pi. Adafruit Learning System. Retrieved February 8, 2023, from https://learn.adafruit.com/adafruit-16-channel-pwm-servo-hat-for-raspberry-pi.