`“Phil”  
SDS Report

By *DrinkTank*

Aaron Riggs, Andre Manz, Chris Baierski, Wesley Scott

**TABLE OF CONTENTS**

[**0.1 Recap**](#_bmk3hp7exfoo) **3**

[**1.0 Introduction**](#_lq5t60pgnxkb) **3**

[1.1 Purpose of this document](#_qj08n67dry7t) 3

[1.2 Scope of the development project](#_pgjiz57543cm) 4

[1.3 Software Context](#_847qnpoep10m) 4

**2.0 Overview of Software Components** **5**

2[.1](#_qj08n67dry7t) Structure and relationships 5

2[.2](#_pgjiz57543cm) Database design 5

**3.0 Architectural and component-level design 5**

3[.1](#_qj08n67dry7t) System structure 5

3[.2](#_pgjiz57543cm) Description for components 5

**4.0 User Interface Design 6**

4[.1](#_qj08n67dry7t) Description of the user interface 6

4[.2](#_pgjiz57543cm) Interface design rules 6

4.3 Components available 7

**5.0 Restrictions, limitations, and constraints 7**

**6.0 Testing Issues 7**

6[.1](#_qj08n67dry7t) Testing issues 7

6[.2](#_pgjiz57543cm) Classes of tests 7

[**7.0 UML Diagrams**](#_7am5mbbaa1nd) **8**

[7.1 Use Case Diagrams](#_yta6mnlndiwf) 8

[7.2 Sequence Diagrams](#_e2o7n95o6ra7) 10

[**8.0 Low Fidelity Prototypes**](#_q2uz2apcet3t) **11**

[8.1 GUI Prototype](#_gzvcy968p3m8) 11

8.2 Hardware Prototype 11

## **0.1 Recap**

“Phil” will be built with several main functionalities in mind, all with user-friendly design as a main priority. The following functions will be listed in order of their importance, with the highest priority appearing at the top. “Phil” will:

1. Pour a variety of drinks, in their standard ratios
2. Allow the user to select drinks via user interface
3. Allow for easy bottle replacement
4. Monitor and store the details of the drinks that have been poured
5. Include an intuitive way to add and remove drinks from the ‘menu’
6. Have an admin area that allows for backend management
7. Keep track of the amount of drinks made
8. Perform business logic on “Phil”’s usage (profits made, usage statistics, etc.)

As an automated bartender, “Phil” clearly needs to be able to pour drinks. “Phil’s” programming and hardware will be finely tuned to ensure we meet our project’s motivations; the need for the perfect pour ratios every time. In addition, there isn’t much fun in a bartender that makes drinks you don’t ask for. To avoid this, users will be able to order from a list of available drinks. They will select from a screen that shows an intuitive menu. Since we want “Phil” to last a lifetime, there needs to be a way to easily replace bottles once they have been emptied. This will allow our system to be a timeless addition to any bar, without an inherent expiration date.

As any owner of nearly any business will tell, it is important to know what is happening in your establishment. This is no different when it comes to the drinks industry. Because of that, “Phil” will keep track of and display information such as what drinks have been poured and their amounts. Also, as “Phil” moves from location to location, or as new drink recipes are made, there will be a need to add and remove drinks from his repertoire. Finally, we will include an admin area that will allow admin (us the creators, and other selected individuals) to perform checks and maintenance on “Phil’s” systems.

With these functionalities making up the basis of “Phil’s” systems, we aim to tackle our project’s motivations of helping to reduce pouring errors found in bars across the nation. “Phil” will be great addition to any restaurant, bar, or party; reducing costs, time costs, and drink inconsistencies.

## **1.0 Introduction**

### **1.1 Purpose of this document**

This document's purpose is to provide a high-level design framework around which to build PHIL (Please Have Infinite Liquor). It also provides a list of requirements against which to test the final project and determine whether we were able to successfully implement the system according to design.

### **1.2 Scope of the development project**

PHIL is a automated bartender is designed to fix one of the largest problems that the food and specifically the beverage industry has, over pouring. This device PHIL will allow users to pour drinks with strict portion control as well as keep track of all drinks created and served, this will better keep track of the resources used and more importantly not used. PHIL has a Raspberry PI as the brain of the project doing all of the control and use computations as well as running the display and GUI. PHIL will be able to be placed in many locations and serve many types of clients. PHIL will run on python, that is to say that the GUI and backend is python based. The essential uses are that the user of PHIL will input the drink they want via the built in touchscreen on the RPi through the GUI and the device will output the drink requested. There will be an admin menu available through a login portal in the GUI that will allow the owner of PHIL to see the drinks being made and the amount of servable material left in the device. We would like to but have doubts about some additional things that we would like to accomplish over the course of the semester like having PHIL use an automated voice assistant, having a companion app to make drinks remotely and having an automated cleaning system built in.

### **1.3 Software Context**

This device is meant to be put in bars and restaurants that have issues with overpouring, the goal of the device is to more or less be a bartender for simple drinks so that the person bartenders can focus on not only the more complicated drinks but the customer leading to a better customer experience and more profit for the establishment.

**1.4 Major constraints**

PHIL’s major constraints will be based on how the establishment intends to use the device. PHIL cannot be used without supervision as it does not do age verification on it’s own, PHIL cannot be poorly maintained for food safely reasons and PHIL cannot be used with liquids that it was not designed to use.

**2.0 Overview of Software Components**

PHIL consists of a variety of software components:

**2.0.1 Raspbian:** grants access to the RPI’s hardware features.

**2.0.2 Python Libraries:**

Gpio: using LED for interfacing with pumps

TKinter: for GUI design

Time: using sleep() for pausing pump code for set period of time in order to

pour liquid

**2.0.3 Arduino Interfacing:** using Arduino C library for switching pins from high to low

**2.0.4 Business Logic**

PHIL’s onboard logging application utilizes simple Python functions to calculate

and store the number of drinks PHIL poured, the time at which they were

poured, the total frequency of each drink that is poured, and the expected amount of money made in various time ranges.

**2.1 Structure and Relationships**

The listed technologies in section 2.0 are very minimally coupled, resulting in little

chance for programmatic error. Each time Phil pours a drink, the RPI brain will both:

**2.1.1** Signal the Arduino to set the necessary pins connected to the pumps to ON.

**2.1.2** Store data from 2.0.4 in SQLite database in addition to the stats tracking page.

**2.2 Database Design**

SQLite DB Schema:

**2.2.1** Drinks(int drinkId, config)

**2.2.2** Sales(saleId, drinkId, timestamp)

**3.0 Architectural and component-level design**

PHIL uses a raspberry as the “brain” of the device, this is connected to a touch screen and an arduino MCU, the user will input into the touch screen and the RPi will output to the arduino. The arduino UNO is connected to the RPI on one side and the relays on the other, this is used to take 3.3v signal from the RPI and convert it to a 5v signal that the relays can use. The relays are used to pass the proper voltage from the power supply to the pumps and valves that control the flow of liquids without these device having to pull power from the RPI or arduino themselves. All of this is controlled by the GUI described in section 4.

**3.1 System Structure**

The structure of the system is relatively simple on the scale that we are working, PHIL has the RPI a brain and components that are attached to the RPI that do all of the physical work that needs to be done. Within the RPI we have a set of python files that are the main function of the device, these files run the GPIO and logs that make up PHIL’s main operational requirements.

**3.2 Description for Components**

**3.2.1** 12V liquid Pump, These pumps working on relays 1 and 2 have a slow but very consistent action that allows for precise measurements of liquid to be released at a constant rate so we use them for pouring the liquor into the glass.

**3.2.2** 9v air pump, This pump along with the valve (3.2.3) control the flow of soda. This is not as precise of a measurement but also does not have to be because the cost of soda in the industry is minimal.

**3.2.3** 12v valve, this valve is meant to limit the flow of soda after the air pump has disengaged. The soda because of carbonation flows on it’s own even with the air pump off and this valve is meant to stop that issue.

**3.2.4** dell power supply, this device is to supply power to PHIL from a 120v AC power source and provide multiple DC voltages for pumps and devices used across the implementation of PHIL.

**3.2.5** relay board, This device controlled by the arduino allows 12v devices to be supplied of cut off from 12v power controlled by a 5v power source. This device makes it so that we can control 12v devices without blowing out the power supply on the RPI and arduino.

**3.2.6** arduino UNO, This device is just a signal passthrough from the RPI to the relay board, effectively boosting the RPI 3.3v to a 5v signal.

**3.2.7** Raspberry PI, The Brain. This device controls all we know to be PHIL, all of the user input through the touch screen and all of the output to the pumps and valves goes through and gets processed by PHIL. This device is the main component, it is internet connected and has input and output that is necessary for the entire operation of the device. This allows not only remote access in case something goes wrong but also if we wanted to develop a companion application this would be the way that we would implement that.

**3.2.8** RPI Touch Screen, This is included so that the user does not have to use a keyboard and mouse to control PHIL. This component is required for the user experience to work as intended.

**3.2.9** Hoses, These are just hoses so that the liquid can run from the pumps to the dispenser.

**4.0 User interface design**

PHIL will utilized a intuitive Graphical User Interface (GUI) that allows for the selection of drinks, as well as for administrators to modify and monitor PHIL’s drinks.

**4.1 Description of the user interface**

The user will be presented with a screen with buttons labeled with various drink names. By clicking on a button, PHIL will pour that drink. If the user is an admin, they can also click the Admin button to sign in. This will allow them to add, remove, and customize drinks, as well as view PHIL’s drink logs.

**4.1.1 Objects and actions**

All objects are imported from Tkinter library, which are called widgets. The widgets include, but are not limited to, buttons and message boxes.

**4.2 Interface design rules**

While designing the GUI, the focus of design will be on:

* Consistency
  + The GUI will follow the same theme and design on each screen
* Feedback
  + The user will be notified of what is happening and what each action they take does.
* Error Handling
  + The user will be presented with readable error messages if errors occur.

**4.3 Components available**

Any and all Tkinter for Python 3.5 widgets are available.

**5.0 Restrictions, limitations, and constraints**

The main design constraints are based on the physics of how liquid flows in the tubes and such that we are using the RPI instead of a full blown integrated development platform. The tubes and adapters can be a pain when it comes to disrupting the flow of soda and causing carbonation, this can cause some issues but of this we have not explored as much. The issues with the raspberry PI are mainly power distribution causing unnecessary complications like having to use the arduino and relays to run simple 12v devices.

**6.0 Testing Issues**

This testing is designed to show any leaks, flaws and vulnerabilities in the system. We will be testing for log accuracy, leaks and drink ratio correctness for the majority of the test.

**6.1 Classes of tests**

The testing of PHIL will consist of running him in an environment that would be a small simulation of the conditions that PHIL is designed to operate under, this would most likely be a small gathering of the work group plus another 3-4 people ordering drinks from the device and us recording the drinks made and operational conditions.

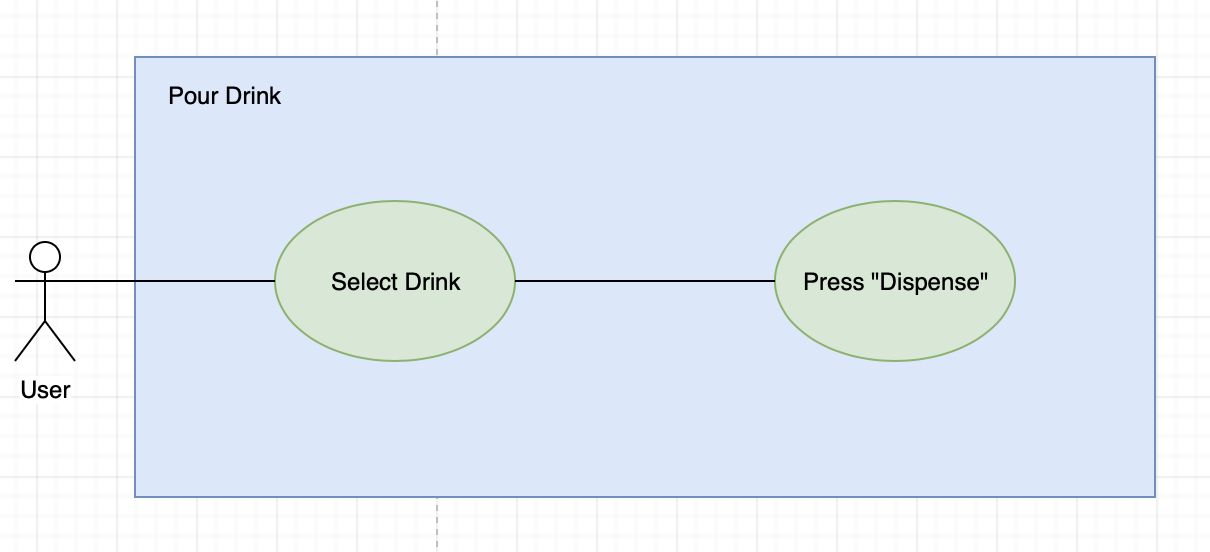
**6.2 Identification of critical components**

The main parts of PHIL that will be under scrutiny during testing will be the accuracy of the pour amount as it pertains to the amount of liquor poured and the script for PHIL to log all of the drinks that he has made over the course of the night. These are the testing parameters that we chose because we have designed the entire system to make bars and restaurants more profitable.

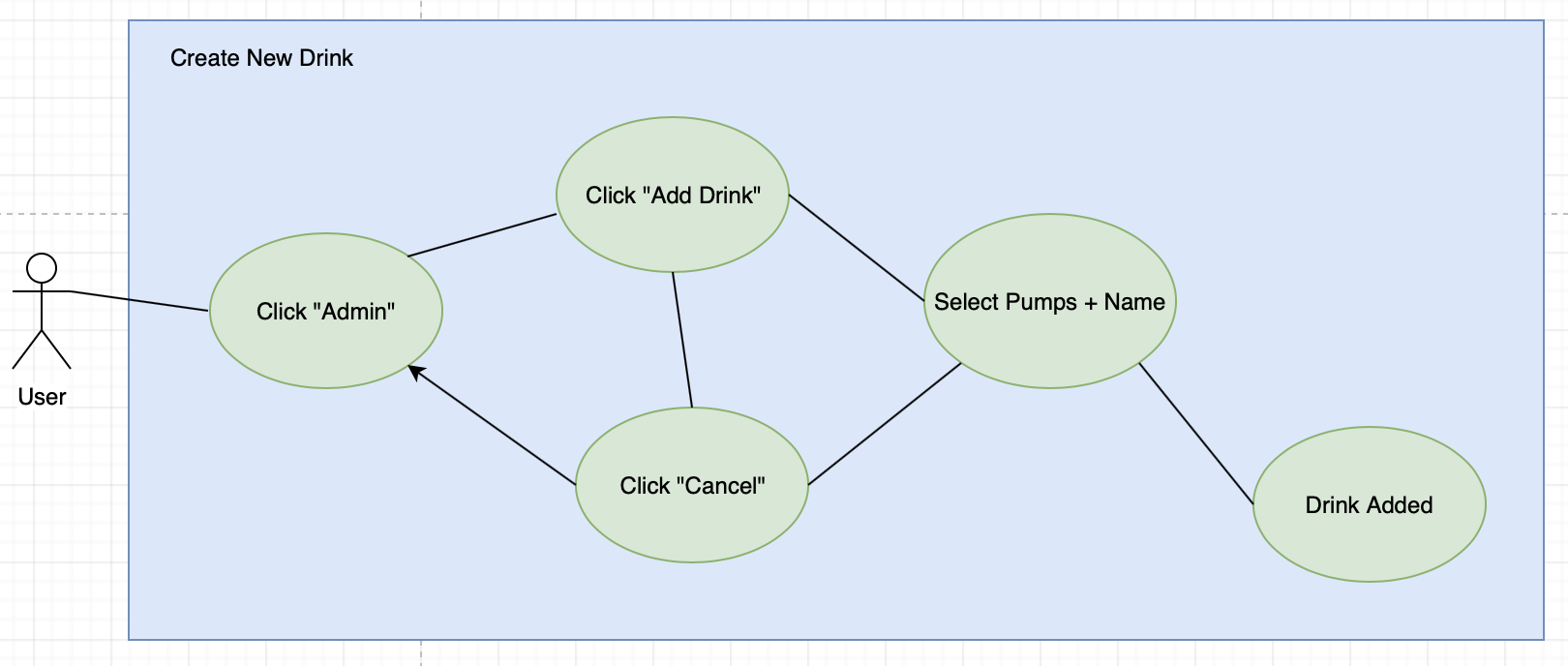
## **7.0 UML Diagrams**

### **7.1 Use Case Diagrams**

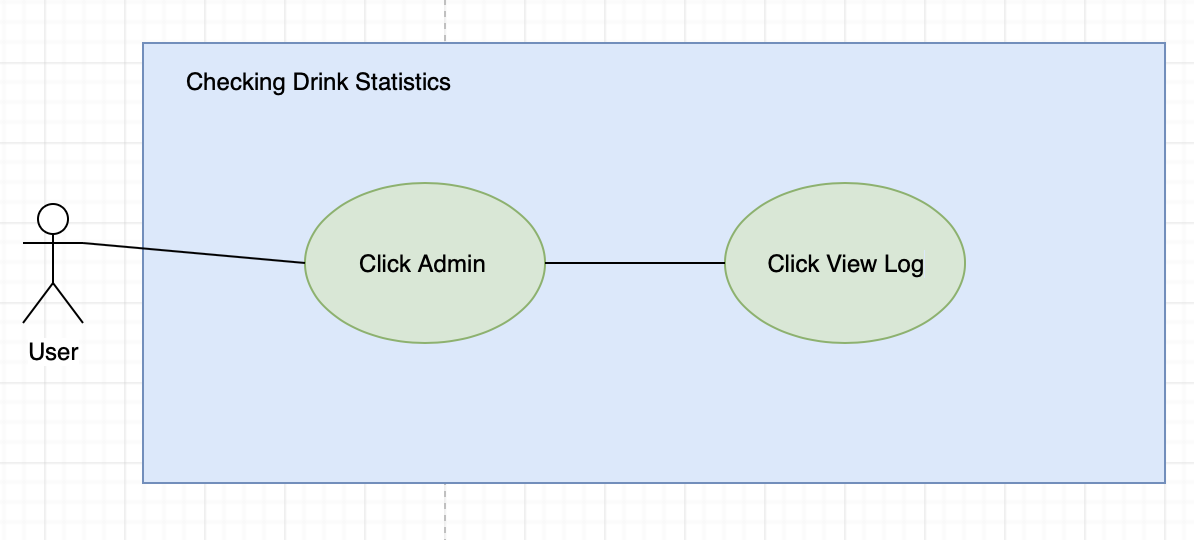
**7.1.1 Pour Drink**



**7.1.2 Create New Drink**



**7.1.3 Checking Drink Statistics**



### 

### 

### 

### 

### 

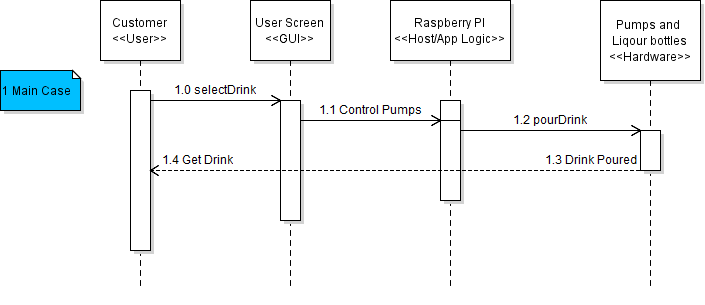
### 

### 

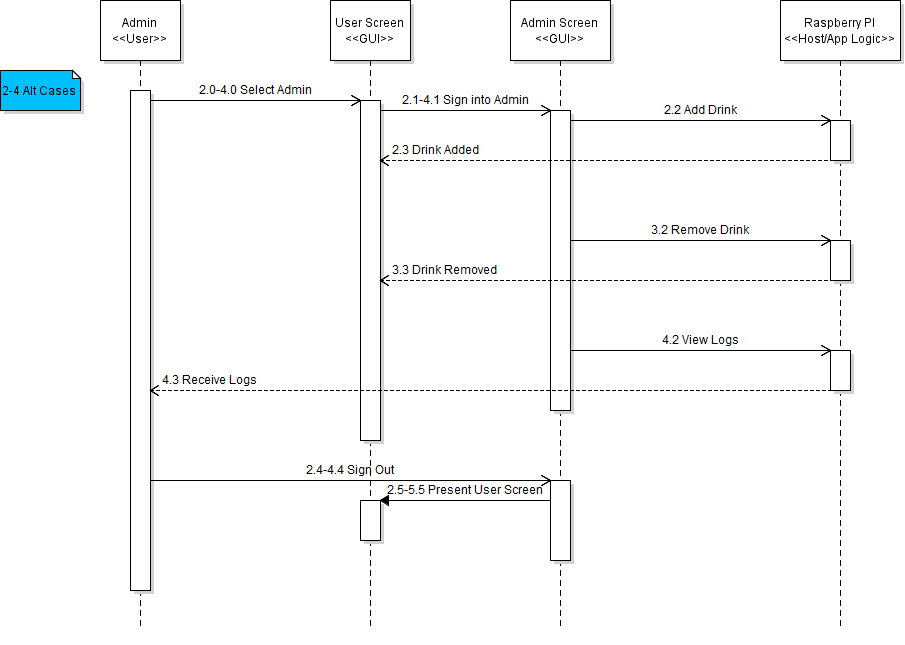
### 

### **7.2 Sequence Diagrams**

**7.2.1 Main Case**

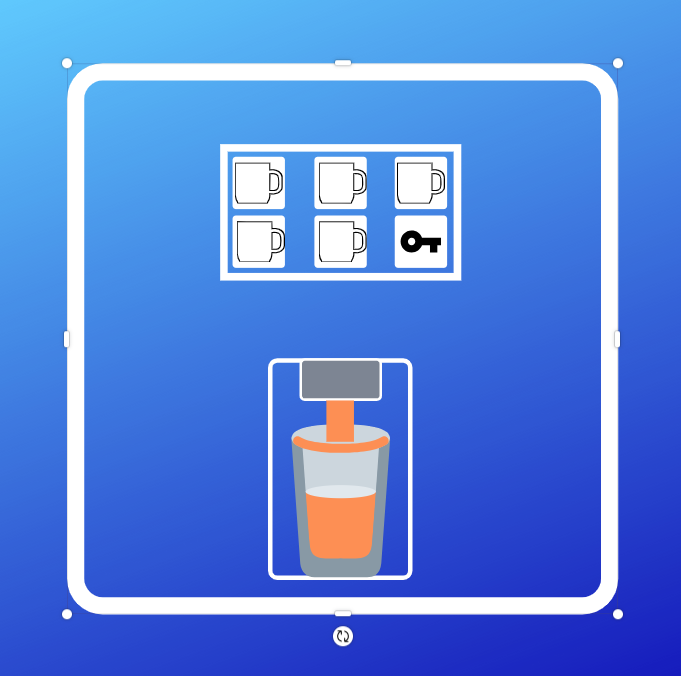


**7.2.2 Alternate Cases**



## **8.0 Low Fidelity Prototypes**

### **8.1 GUI Prototype**

  
**8.2 Hardware Prototype**

