Department of Computer Science & Engineering
The University of Texas at Arlington

Architectural Design Specifications
CSE 4316: Senior Design I
Spring 2021

Back
Burner
Brew

Team Friendship
Back Burner Brew

Luke Brown
Marcos Juarez Casillas
Ju Young Isa Jung
Sujan Dumaru
Sunghwa Cho
Matthew Francis Schultz

Revision History

Revision	Date	Author(s)	Description
0.1	10.01.2015	GH	document creation
0.2	10.05.2015	AT, GH	complete draft
0.3	10.12.2015	AT, GH	release candidate 1
1.0	10.20.2015	AT, GH, CB	official release
1.1	10.31.2015	AL	added design review requests

Contents

List of Figures

List of Tables

1 Introduction

The "Back Burner Brew" is built with the sole purpose of brewing large batch of beer in the home environment. This product provides home brewers with a low-cost electric home brewing system that allow them to have precise control over the brewing process. The brewing process can be automated with the help of micro-controllers like the ESP32 which is then hosted to a local website or an app interface.

ESP32 is a micro-controller that can receive data such as current temperature of the water or mash from the heat sensors located inside the kettles which can be converted to either analog or digital input. The heating coil can be controled using the input from the user as per their desired either to increase or to decrease the temperature. The electric pump can also be controlled by the user through micro-controllers to regulate the flow of the water in the kettles. The user will be able to communicate with the brewing system through a web interface or app interface.

The user should expect to input desired commands, controls, and specific settings such as temperature and length of time by a easily accessible touchscreen. The touchscreen will be attached to a Raspberry Pi that will handle communications between the user and the various sensors and heating elements. The user can expect that whichever temperature they set for their desired application, that the temperature will remain constant.

The intended audiences for this product would be home brewers or person interested in brewing beer only. Provided that the user manual would be present in the product, any person who wants to brew beer in his local environment can easily use this product. This product is made focusing on how effortless can the brewing process gets simply with the use of micro-controller.

2 System Overview

This section will describe the overall structure of the Back Burner Brew. The Brew System Vessel layer is where the brewing process will take place and could function as its own brewing system without the automation. All the other layers are built on top of the Brew System Vessel layer and will be used to automate the brewing process.

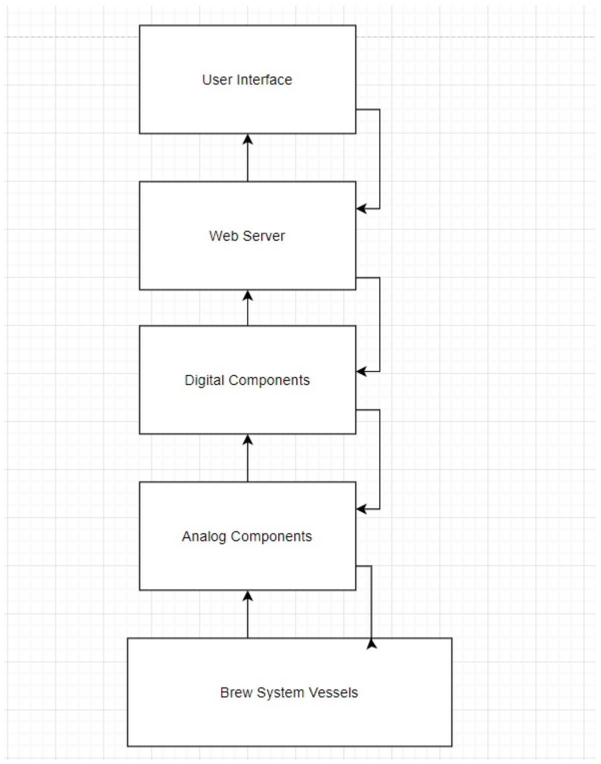

Figure 1: Simple ADS

2.1 Brew System Vessel Layer

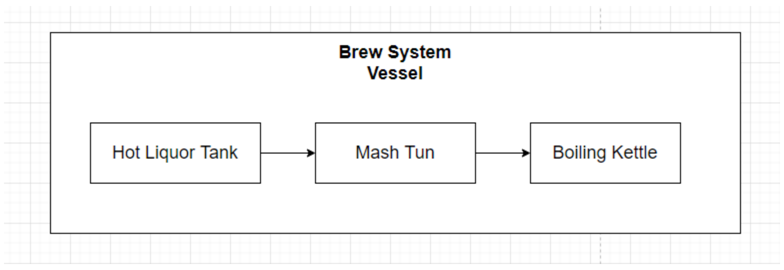

Figure 2: Brew System Vessel Layer

The Brew System Vessel Layer is where the brewing process will take place and does not have any software involved. This layer would be enough for a brewer to make their own beer without any automation. This layer is comprised of the mash tun, hot liquor tank(HLT), and the brewing kettle. The HLT will control the temperature of the liquid inside the mash tun by heating the liquid through a coil inside the HLT. The mash tun is where the mashing process begins and the grain is added to the water.

The boiling kettle is where the liquid will be boiled for a set amount of time and then sent to a chiller and fermentation kettle.

2.2 Analog Components Layer

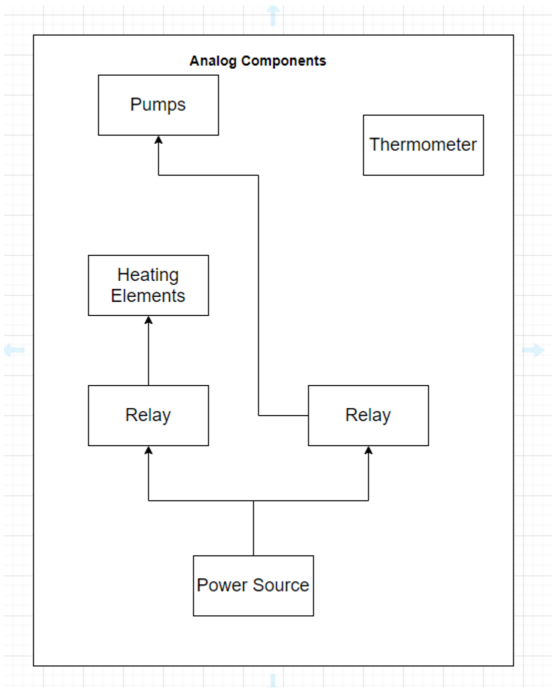

Figure 3: Analog Components Layer

The Analog Components Layer is comprised of pumps, thermometers, relays, heating element, and the power source. The pumps are controlled by the relays and will be activated when it is time for the liquid inside the kettles to be moved to the next stage of the brewing process. The pumps and heating elements will be controlled by the ESP32's which will send a signal to the relays to activate the components. The thermometers will send data to the ESP32's which communicate to the Raspberry Pi.

2.3 Digital Components Layer

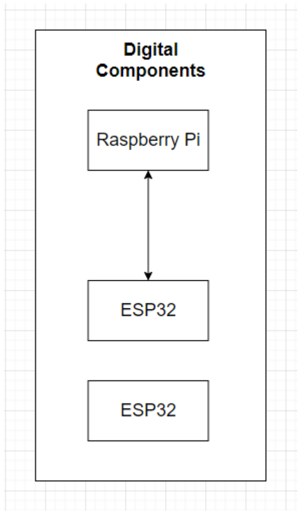

Figure 4: Digital Components Layer

The Digital Components Layer is made up of the ESP32's and the Raspberry Pi. The ESP32 will receive data from the thermometer and will send the data to the Raspberry Pi. The Raspberry Pi will send the data to the web server and any instructions recieved from the web server will be sent to the ESP32's. The ESP32's will send a signal to the relay and activate the pumps and heating elements depending on which step of the brewing process it is in.

2.4 Web Server Layer

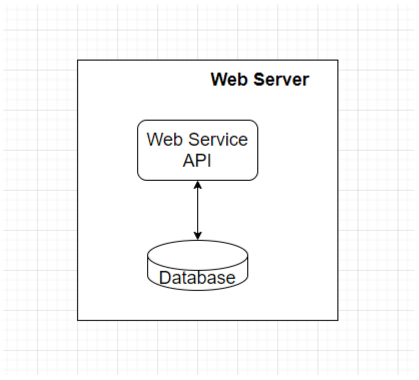

Figure 5: Web Server Layer

The Web Server Layer will be where all the data will be stored and instructions will be sent out from. The web server will receieve data from the Raspberry Pi and will store the data in a database. The web server will then decide what actions the system must take in order to meet the criteria set by the brewer. The two primary actions that web server will control is activating the heating elements and the pumps. Conditions such as water temperature and the amount of time spent on a specific step of the brewing process will be receieved from the User Interface Layer and will be maintained by the web server.

2.5 User Interface Layer

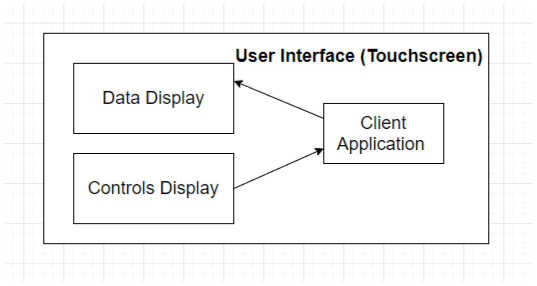

Figure 6: User Interface Layer

The User Interface Layer will be where the user can set the conditions of the brewing system, as well as see the temperatures of the different kettles and the time remaining on the current step in the brewing process. The brewer will have a user interface which will allow them to set the temperatures of the HLT and the mash tun. The boiling kettle will not be set a specific temperature, instead the heating element will be set to a power output set by the brewer. Once the brewer has entered their desired conditions, this data will be sent to the web server through the client applciation.

3 Subsystem Definitions & Data Flow

The Back Burner Brew system will consist of five different subsystems: brew system vessels, analog components, digital components, web server, and the user interface. The brew system vessels will be operated through the analog components. The analog components will send data to digital components. The analog components will be controlled by the digital components based on what is input to the user interface or certain conditions that are predefined such as keep water a certain temperature. The web server will be the intermediary between the user interface and the digital components. The web server will also store all data associated with the brewing process such as temperature or time spent on specific tasks.

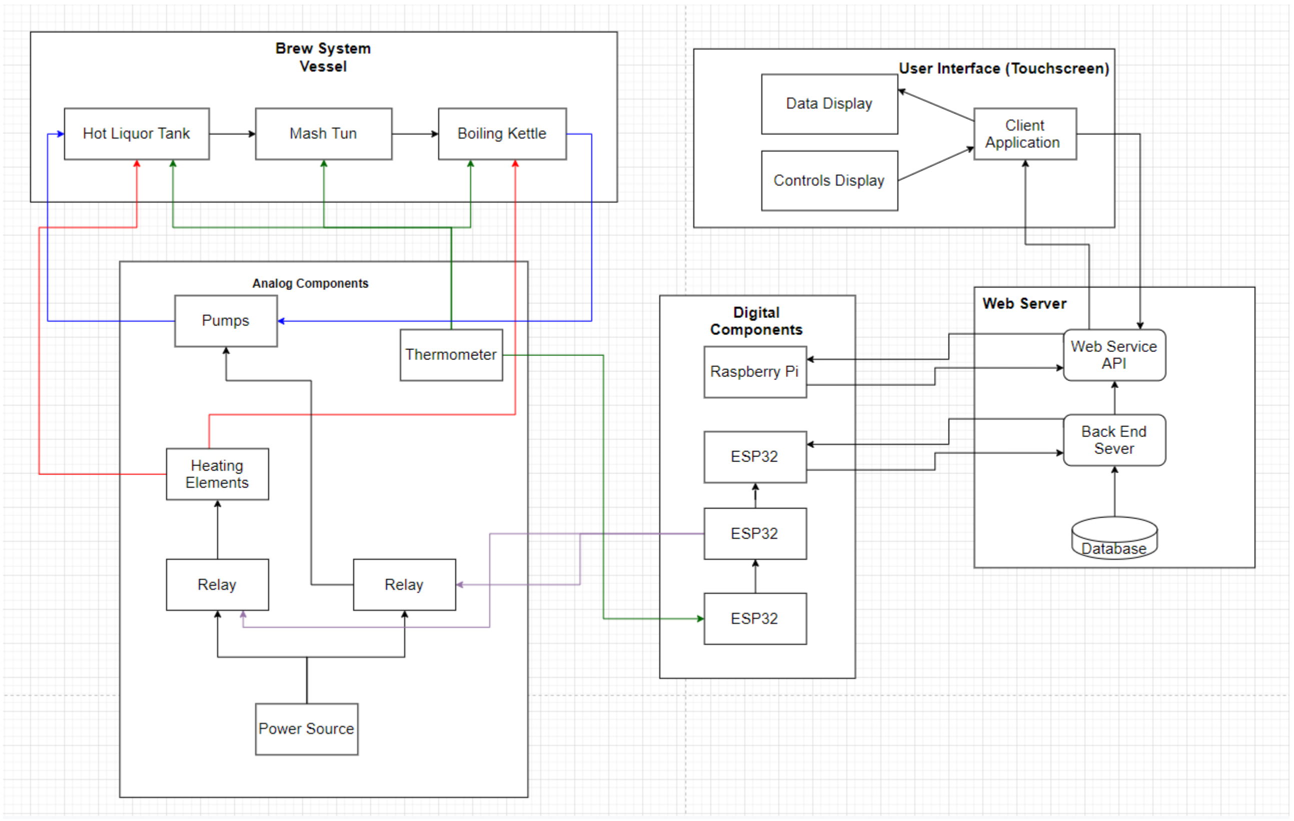

Figure 7: Subsystem Diagram for the Back Burner Brew device.

4 Brew System Vessel Layer Subsystems

The Brewing Vessel layer consists of three major components which plays a vital role in each part for maintaining a good brew. This is the initial phase and the most important layer which should be observed properly. Any mistake made on this layer can drastically affects the taste of beverages. Brew System Vessel layer is directly connected to the some of subsystems from the Analog Components layer which help to regulate the Brewing process. "Brew in A Bag", an alternative for this system was considered which would only contains two subsystems; however, the system chosen in this design provided users with better control and evaluation on overall brewing process. The three subsystems of this layer are explained below:

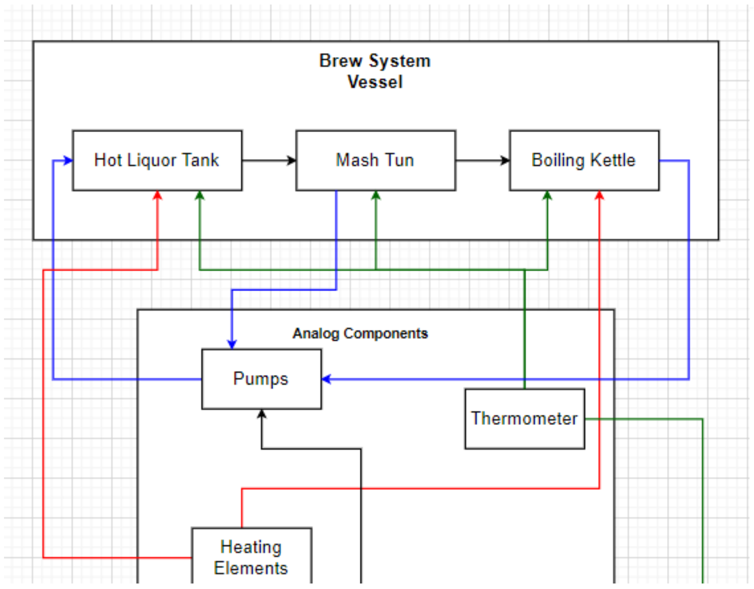

Figure 8: Brew System Vessel Layer Representation

4.1 Hot Liquor Tank Subsystem (HLT)

The main purpose of Hot Liquor Tank subsystem is to monitor and heat the water to the desired temperature. This subsystem is directly connected to three subsystems from Analog Components layer which are pumps, thermometer, and heating elements.

4.1.1 Assumptions

In the initial phase, the vessel should be filled with water by the brewer. The subsystem is controlled by the micro-controller based on the current temperature of the water measured from thermometer and sent through micro-controller to the user.

4.1.2 Responsibilities

The Hot Liquor Tank subsystem is responsible for providing information such as current temperature of the water to the brewer, current status of pump and heating elements. Then the subsystem should heat the water to the desired temperature as per the brewer wants. When the water is reached to desired temperature, then it is responsible to pass the water to the Mash Tun subsystem.

4.1.3 Subsystem Interfaces

Table 2: Hot Liquor Tank Subsystem interfaces

Description	Inputs	Outputs
Thermometer Subsystem	User input to display temperature	Current Temperature of the water
Pump Subsystem	User input collected from the micro controller	Open/Close the pump based on the input
Heating Element Subsystem	User input in temperature collected from the micro controller	Turn on/off heating elements in order to reach user desired temperature

4.2 Mash Tun Subsystem

The main purpose of Mash Tun subsystem is to mix the hot water with grains to produce wort. This subsystem is directly connected to Thermometer and Pump subsystems from Analog Components layer which is used to provide temperature of the mash.

4.2.1 Assumptions

When the Mash Tun is filled with hot water from the HLT, then the Mash Tun is manually loaded with the crushed grains. The user is supposed to set the temperature and time through web or app interface he want the mash to be in this subsystem. If the temperature inside the Mash Tun reached to the different temperature than the user set, the water is sent back to HLT and heated. The mash is rinse several time to rinse the sugar from the mash which is then passed to Boiling Kettle subsystem.

4.2.2 Responsibilities

The Mash Tun is responsible to remove sugars, mostly maltose, sucrose and maltotriose from the crushed grains to produce the final product called wort, also known as un-fermented beer. The Mash Tun should provide temperature of the current mash to the brewer. The Mash Tun is responsible to pump out the water to HLT Subsystem when it doesn't maintain required temperature. And when the water is heated enough, then it is again passed to Mash Tun subsystem. By this way, the mash is rinse to get the sugars out. The wort obtained is then passed to Boiling Kettle subsystem in this layer.

4.2.3 Subsystem Interfaces

Table 3: Mash Tun Subsystem interfaces

Description	Inputs	Outputs
Thermometer Subsystem	User input to display and set temperature	Current Temperature of the mash
Pump Subsystem	User input collected from the micro controller	Open/Close the pump based on the temperature of the mash

4.3 Boiling Kettle Subsystem

The main purpose of Boiling Kettle subsystem is to boil wort for the required amount of time. This subsystem is directly connected to Thermometer, Pump, and Heating elements subsystems from Analog Components layer which is used to maintain the amount of heat provided to the wort.

4.3.1 Assumptions

The Boiling Kettle subsystem is almost an automated process. The brewer have to add some hops after each beer cycle in the wort as needed because it add bitterness to the beer. The brewer must be careful with adding appropriate amount of hops as too much can ruin the taste and very little won't bring any taste to beer.

4.3.2 Responsibilities

The Boiling Kettle is only responsible for heating the wort with appropriate heat for the appropriate amount of time. The wort should be heated properly to maintain the good flavor of beer. Once the boil is done, the Boiling Kettle subsystem is responsible for sending beer to the chiller for fermentation.

4.3.3 Subsystem Interfaces

Table 4: Boiling Kettle Subsystem interfaces

Description	Inputs	Outputs
Thermometer Subsystem	User input to display temperature	Current Temperature of the wort
Pump Subsystem	User input collected from the micro controller	Open/Close the pump based on the input
Heating Element Subsystem	User input in temperature collected from the micro controller	Turn on/off heating elements in order to reach user desired temperature

5 Analog Components Layer Subsystems

In this section, the layer is described in some detail in terms of its specific subsystems. Describe each of the layers and its subsystems in a separate chapter/major subsection of this document. The content of each subsystem description should be similar. Include in this section any special considerations and/or trade-offs considered for the approach you have chosen.

5.1 Subsystem 1

This section should be a general description of a particular subsystem for the given layer. For most subsystems, an extract of the architectural block diagram with data flows is useful. This should consist of the subsystem being described and those subsystems with which it communicates.

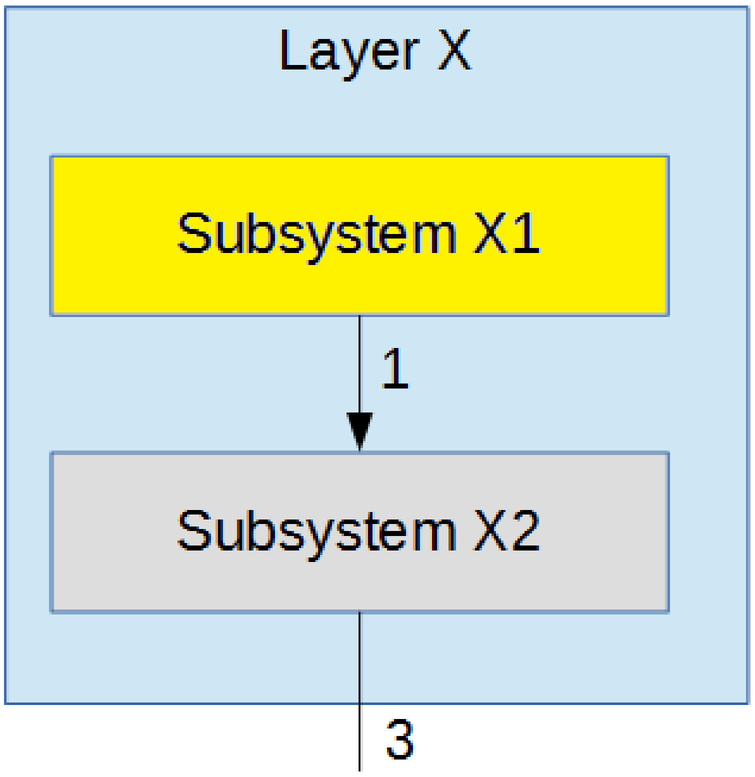

Figure 9: Example subsystem description diagram

5.1.1 Assumptions

Any assumptions made in the definition of the subsystem should be listed and described. Pay particular attention to assumptions concerning interfaces and interactions with other layers.

5.1.2 Responsibilities

Each of the responsibilities/features/functions/services of the subsystem as identified in the architectural summary must be expanded to more detailed responsibilities. These responsibilities form the basis for the identification of the finer-grained responsibilities of the layer's internal subsystems. Clearly describe what each subsystem does.

5.1.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here. Create a table with an entry for each labelled interface that connects to this subsystem. For each entry, describe any incoming and outgoing data elements will pass through this interface.

Table 5: Subsystem interfaces

ID	Description	Inputs	Outputs
#xx	Description of the interface/bus	input 1 input 2	output 1
#xx	Description of the interface/bus	N/A	output 1

5.2 SUBSYSTEM 2

Repeat for each subsystem

5.3 SUBSYSTEM 3

Repeat for each subsystem

6 Digital Components Layer Subsystems

The digital components layer will consist of 4 devices. The first is a Raspberry PI. This will act as the main computer of the system. It will the majority of the data both from the server, the UI, and the microcontrollers. The other 3 components will all be ESP32 microcontrollers, but they will each serve a different purpose. An ESP32 will receive information from a thermometer sensor and relay it to the heat control ESP32. Based on what the temperature is, the heat control ESP32 will trigger the relay to activate or deactivate the heating element. The final ESP32 will be used to trigger the relay to turn on the pump based on what the message broker instructs it to do.

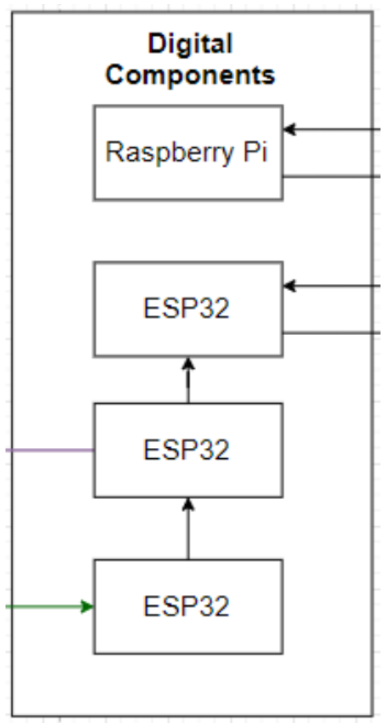

Figure 10: Digital Components Subsystem Description Diagram

6.1 Raspberry PI Subsystem

The Raspberry PI will host the web server and the MQTT client. The RBP will receive information from the web server about what the user inputed via the UI. It will then take that information and relay it to the microcontrollers using the MQTT protocol. The RBP will then receive sensor data from the microcontrollers through MQTT messages and send store the information in the web server.

6.1.1 Assumptions

The RBP will have a power supply. It will remain safe from the elements. It will be connected to the network. It will have unbroken communication with the web server. It will communicate with the microcontrollers through the MQTT protocol.

6.1.2 Responsibilities

It will receive data from the web server to determine how to operate the pump. It will instruct the micro-controller to activate or deactivate based on the information it receives. It will relay sensor information to the web server so that it can be displayed.

6.1.3 Subsystem Interfaces

Table 6: Raspberry Pi Subsystem interfaces

Description	Inputs	Outputs
Web Service API	Information from UI input by user	Sensor Data and status
ESP32	Sensor data from ESP32	Messages instructing microcontrollers what to do

6.2 Thermometer Sensor Subsytem

This microcontroller will monitor data from the thermometer sensor. It will then relay that information back to the message broker.

6.2.1 Assumptions

The microcontroller will have a power supply. It will remain safe from the elements. It will be connected to the network. It will be connected to the thermometer sensor. It will operate through MQTT protocol.

6.2.2 Responsibilities

It will monitor the data from the thermometer sensor and relay that information using the MQTT protocol.

6.2.3 Subsystem Interfaces

Table 7: Thermometer Sensor Subsystem interfaces

Description	Inputs	Outputs
Thermometer Sensor	Data from thermometer sensor	message to communicate sensor data

6.3 Heat Control Subsystem

The microcontroller will await messages from the message broker. The message broker will recieve information from the thermometer ESP32. If the temperature is too low, the message broker will send a message to this microcontroller to trigger the relay and allow power to flow to the heating element. If the temperature gets too high on the heating element, the message broker will tell this microcontroller to deactivate the heating element.

6.3.1 Assumptions

The microcontroller will have a power supply. It will remain safe from the elements. It will be connected to the network. It will be connected to the relay that controls power to the heating element. It will operate through MQTT protocol.

6.3.2 Responsibilities

It will activate or deactivate the relay that controls the power flow to the heating element based on what the message broker instructs it to do. It will send a status message back to the message broker.

6.3.3 Subsystem Interfaces

Table 8: Heat Control Subsystem interfaces

Description	Inputs	Outputs
Heating Element Relay	message from RBP	relay trigger and status message

6.4 Pump Control Subsystem

This microcontroller will await messages from the message broker. When the message broker sends it the message to turn on the pump, the microcontroller will trigger the relay to allow power to go to the pump. When the message broker sends the message to turn off the pump, the microcontroller will cease sending power to the relay, causing it to close, and power will not be supplied to the pump.

6.4.1 Assumptions

The microcontroller will have a power supply. It will remain safe from the elements. It will be connected to the network. It will be connected to the pump relay. It will operate through MQTT protocol.

6.4.2 Responsibilities

It will trigger the relay that will supply power to the pump based on the message it receives from the message broker. It will then return a status message to the message broker.

6.4.3 Subsystem Interfaces

Table 9: Pump Control Subsystem interfaces

Description	Inputs	Outputs
Pump Relay	message from RBP	relay trigger and status message

7 Web Server Layer Subsystems

The Web Server Layer will receive and store data gained from Raspberry Pi and provide web service API to front-end applications. Online or local databases will be used such as Google Firebase, Amazon RDS, Mysql, etc. Web service API will provide information to the devices when they call in order to meet the criteria set by the brewer. The web server should be able to get a recipe from a user and control heating elements and the pumps based on conditions of water temperature and the amount of time that a user set for brewing.

7.1 Web Service API

API is a software interface that allows two applications to interact with each other without any user involvement. Web service API will provide information extracted from the database to the user interface layer when certain APIs are called.

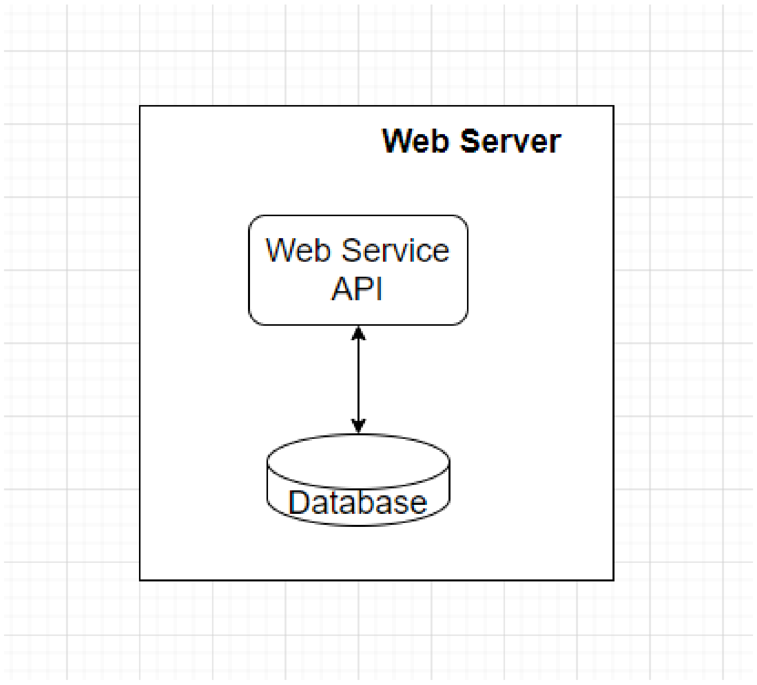

Figure 11: Example subsystem description diagram

7.1.1 Assumptions

- The Web service API supports HTTP/HTTPS protocol. is used for transferring data.
- Web service supports XML and JSON.
- Web service is RESTful (GET, POST, PUT and DELETE).

7.1.2 Responsibilities

Web Service API should establish stable connections with database and HTTP protocols to transfer data without data leak. Each corresponding APIs will extract accurate information from the database and provide it to other devices to help them to make the correct decision.

7.1.3 Subsystem Interfaces

Table 10: Subsystem interfaces

Description	Inputs	Outputs
API called from other devices	user input	Data

7.2 Database

A database is a data structure that stores organized information with multiple tables, which may each include several different fields. The brewing database may include tables for water temperature, recipe, etc. Each of these tables would have different fields that are relevant to the information stored in the table.

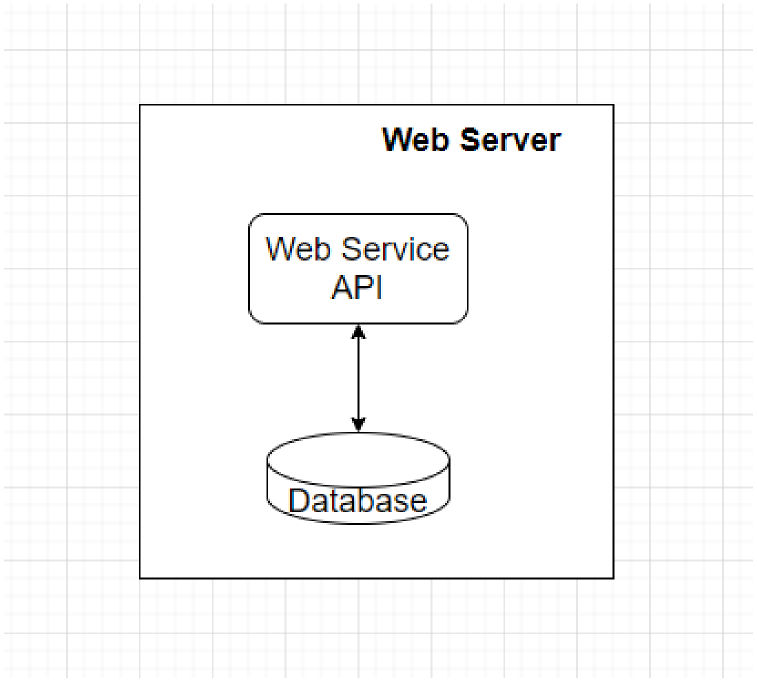

Figure 12: Example subsystem description diagram

7.2.1 Assumptions

- Database should establish a stable connection with Web service API.

7.2.2 Responsibilities

The database should be available at any time and have enough capacity to store data.

7.2.3 Subsystem Interfaces

Table 11: Subsystem interfaces

Description	Inputs	Outputs
Receive requests from API	query from API	Data

8 User Interface Layer Subsystems

The touchscreen device will retrieve application data from the web server to display information from the current brew. The display will also offer a Graphical User interface where the user can input commands to modify the current brew. The client application running on this device will then interpret the data, and either display it on the Data Display or send it back to the server as commands accordingly.

8.1 Data Display

The data display will display real-time information relevant to the current brew. The data display will retrieve this data from the web server.

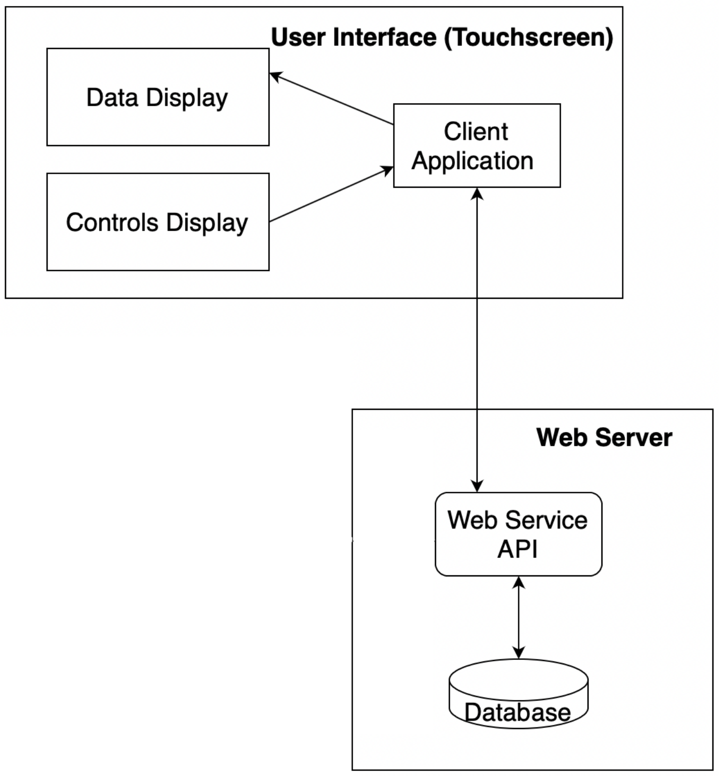

Figure 13: User Interface subsystem diagram

8.1.1 Assumptions

- The device will receive the data and interpret it correctly on the touchscreen.
- The data received from the server will be accurate and up-to-date.
- The device will have a stable connection to the server throughout the brew.

8.1.2 Responsibilities

The device will receive accurate and relevant information from the server and display it on the touchscreen for the user to make decisions on the current brew. This information will include current set temperature, length of time the temperature has been set, how much longer brew will stay at current temperature, and any other information that may be relevant.

8.1.3 Subsystem Interfaces

Table 12: Subsystem interfaces

Description	Inputs	Outputs
Data Display	Data from server	Brew information

8.2 Controls Display

The controls display on the touchscreen will allow the user to interact with and send commands to the server, which will relay the data to the brewing system.

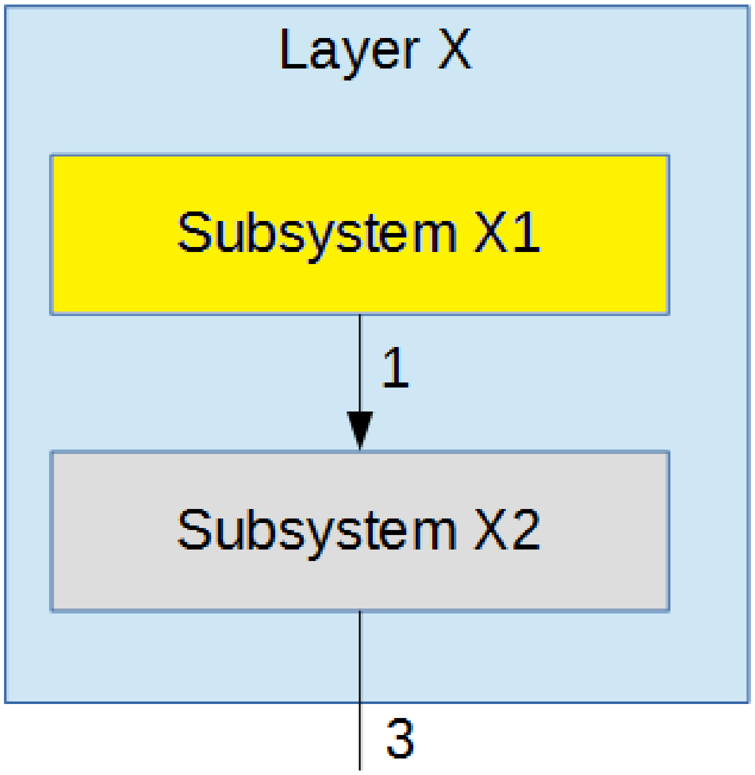

Figure 14: Example subsystem description diagram

8.2.1 Assumptions

- The client application will read user commands correctly and send them following the established protocol.
- The device will be quick in sending commands to the server.
- The device will have a stable connection to the server throughout the brew.

8.2.2 Responsibilities

The touchscreen will allow the user to enter commands such as set, increase, or decrease temperature and/or the length of time temperature is set. The interface will be intuitive and easy to use. The client application will then take the commands, and send them to the server following the set communcation protocol. Commands sent will update the information on the data display if applicable.

8.2.3 Subsystem Interfaces

Table 13: Subsystem interfaces

Description	Inputs	Outputs
Controls Display	User input	Data to server