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

Detailed Design Specification
CSE 4317: Senior Design II
Summer 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	6.14.2021	JYJ	document creation
0.2	6.28.2021	ALL	complete draft
1.0	6.28.2021	ALL	official release

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 sensors and micro-controllers like the ESP32 which is then hosted to a local website or an app interface.

The user should expect to input desired commands, controls, and specific settings such as temperature and length of time by a website that is easily acccessible through a computer or phone. The user can expect that whichever temperature they set for their desired

2 System Overview

The Brew System layer is the core of the Back Burner Brew. The goal of the project is to automate the brewing process, and although the system itself is very simple, the steps required to brew a proper beer is time consuming and requires a large amount of attention from the brewer. The user interface layer will ask the brewer to enter their desired temperature and time for each step of the brewing process. The analog and digital components will send data to the digital components. This data will contain the temperatures of the Hot Liqour Tank (HLT) and the mash tun, once the data is receieved by the digital components, the Web server layer will store that data into a cloud database and that data will be used to determine when a heating element or pump will turn on.

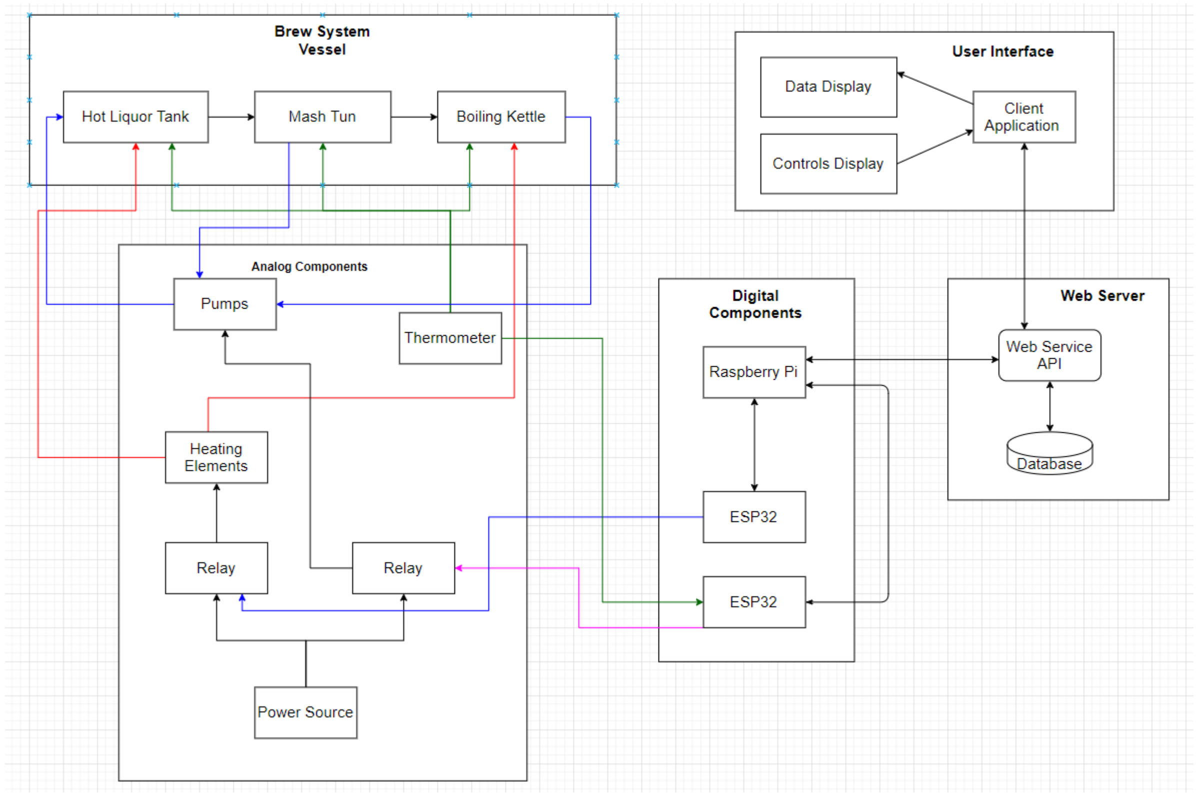

Figure 1: System Overview

3 Brew System Vessel Layer Subsystems

This section is completely depended upon the hardware. It is composed of only hardware all combined together to form a suitable environment for brewing. All the hardware must be connected properly with the system to ensure quality beverages.

3.1 Layer Hardware

The most important hardware required for this layer are three vessels/kettles, probably of 5 gallon each. Each vessel is connected to the 330 GPH Low Suction Electric pumps with suction hose to pump the liquid out of the vessels and DS18B20 Thermometer Temperature Sensor Probe in order to keep track of the temperature in each vessel.

3.2 Layer Operating System

There is no operating system involved in this layer. However, the temperature data and the pump of this layer is controlled by the Arduino Uno micro controller which is in different layer.

3.3 Layer Software Dependencies

Since this is completely hardware dependent layer, there is no any software dependencies.

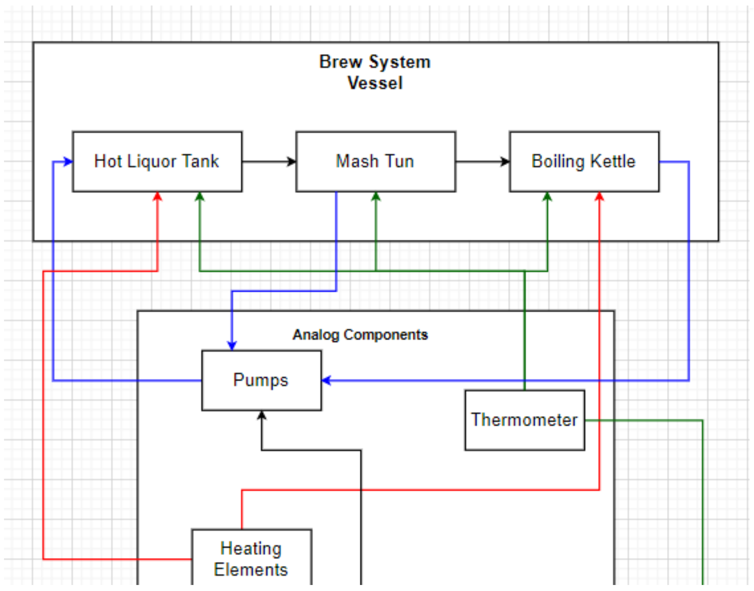

Figure 2: Brew System Vessel Layer Representation

3.4 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 simply a vessel hardware used for boiling purpose.

3.4.1 Subsystem Hardware

Apart from the hardware mentioned above, this subsystem have one more hardware involved which is a Kettle Heating Element.

3.4.2 Subsystem Operating System

No Operating System is involved in this subsection.

3.4.3 Subsystem Software Dependencies

The Heating Element is depended on the input provided by the user on the basis of temperature as the water is only heated to that temperature. So, this subsystem require input from the user/brewer to properly function.

3.4.4 Subsystem Programming Languages

In order to provide the input, the user is using Arduino programming language for Arduino Uno.

3.4.5 Subsystem Data Structures

No specific data structures are being used in this subsystem. It is plain number being passed in and out of this subsystem.

3.4.6 Subsystem Data Processing

The thermometer temperature sensor probe reads the data and simply passes it to micro controller and the micro controller uses that data in order to raise the temperature of water by turning on or off the heating element.

3.5 Mash Tun Subsystem

The main purpose of Mash Tun subsystem is to mix the hot water with grains to produce wort. This subsystem is simply a vessel hardware to separate mash and wort.

3.5.1 Subsystem Hardware

There are no separate hardware used in this subsystem.

3.5.2 Subsystem Operating System

No Operating System is involved in this subsection.

3.5.3 Subsystem Software Dependencies

The hardware used in this subsystem are not software dependencies.

3.5.4 Subsystem Programming Languages

No programming languages are used.

3.5.5 Subsystem Data Structures

No specific data structures are being used in this subsystem. there is just a float value being passed out of this subsystem.

3.5.6 Subsystem Data Processing

The thermometer temperature sensor probe reads the data and simply passes it to micro controller.

3.6 Boiling Kettle Subsystem

The main purpose of Boiling Kettle subsystem is to boil wort for the required amount of time. This subsystem is simply the final vessel hardware used to produce beverage.

3.6.1 Subsystem Hardware

Apart from the hardware mentioned above, this subsystem have more hardware involved like a Kettle Heating Element and a NY Brew Supply copper wort chiller.

3.6.2 Subsystem Operating System

No Operating System is involved in this subsection.

3.6.3 Subsystem Software Dependencies

The software dependencies is similar to Hot Liquor Tank Subsystem. The Heating Element is depended on the input provided by the user on the basis of temperature as the wort is only heated to that temperature.

3.6.4 Subsystem Programming Languages

In order to provide the input, the user is using Arduino programming language for Arduino Uno.

3.6.5 Subsystem Data Structures

No specific data structures are being used in this subsystem. It is plain number being passed in and out of this subsystem.

3.6.6 Subsystem Data Processing

The thermometer temperature sensor probe reads the data and simply passes it to micro controller and the micro controller uses that data in order to raise the temperature of water by turning on or off the heating element. After the desired time is reached, the wort is sent to chiller which then cool downs to form beverages.

4 Analog Components Layer Subsystem

This layer is very hardware dependent, to the extent to were all software control is done elsewhere in the Digital Layer. The components are used by the Digital component layer to control the flow and temperature of the Brew System.

4.1 Layer Hardware

The hardware layer will contain 2 relays, a heating element, the specific amount and model is subject to change on testing. It will have a 330 GPH low suction electric pump, and a DS18B20 thermometer temperature sensor probe.

4.2 Layer Operating System

The hardware layer will have no operating system. As the hardware layer will be manipulated by the Digital Components subsystem

4.3 Layer Software Dependencies

All control dependencies will be offset on the Digital Components subsystem as the relays will be controlled through power management from the Digital Components layer

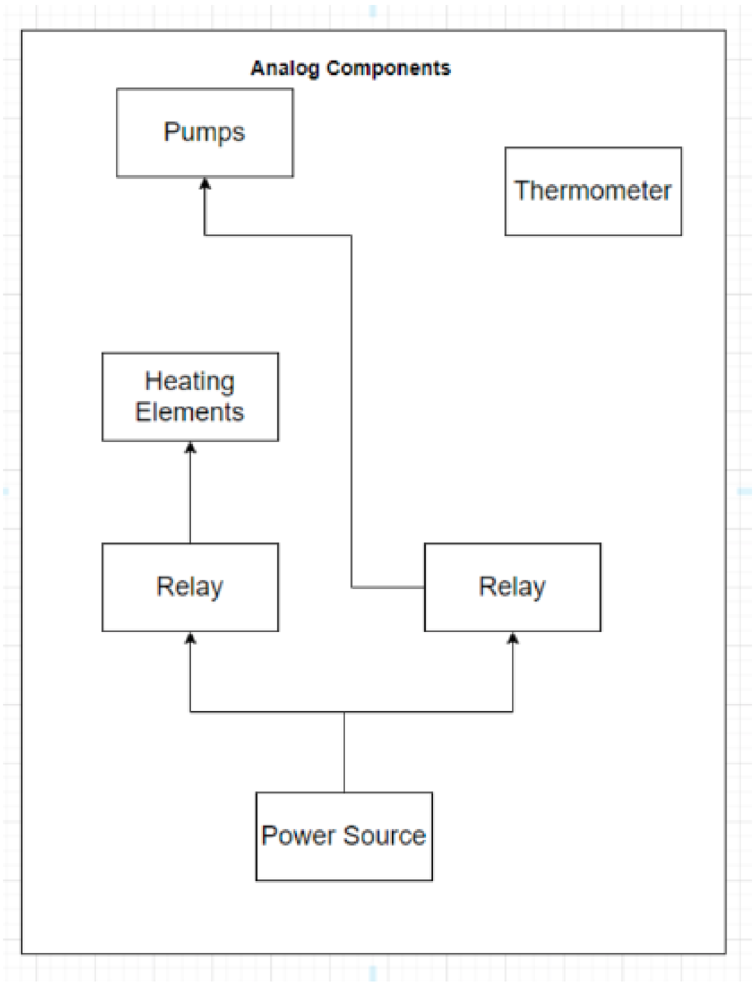

Figure 3: Analog System Vessel Layer Representation

4.4 Heating Elements Subsystem

This subsystem is used for temperature control of Hot liquor tank and the Boiling Kettle. Specifically the subsystem contains the heating elements and the Relay responsible for managing temperature control.

4.4.1 Subsystem Hardware

This subsystem will have heating elements, and a control relay.

4.4.2 Subsystem Operating System

This subsystem will have no operating system.

4.4.3 Subsystem Software Dependencies

There is no software dependencies.

4.4.4 Subsystem Programming Languages

There is none all programming and control will be done by the digital components layer.

4.4.5 Subsystem Data Structures

This will be the case for relay control we should be using as simple of a Data structure as possible because anything more is unnecessary because the relay just need a on or off signal.

4.4.6 Subsystem Data Processing

For relay power control we should do a decent amount of testing to judge the delay that we are operating on for temperature management.

4.5 Pump Control Subsystem

This subsystem will be used for flow control of the entire system.

4.5.1 Subsystem Hardware

This subsystem will contain the pumps, and a control relay.

4.5.2 Subsystem Operating System

This subsystem will have no operating system.

4.5.3 Subsystem Software Dependencies

There is no software dependencies.

4.5.4 Subsystem Programming Languages

There is none all programming and control will be done by the digital components layer.

4.5.5 Subsystem Data Structures

This will be the case for relay control we should be using as simple of a Data structure as possible because anything more is unnecessary because the relay just need a on or off signal.

4.5.6 Subsystem Data Processing

For relay power control we should do a decent amount of testing to judge the delay that we are operating on for temperature management.

4.6 Power Control Subsystem

This subsystem will be controlling the electric current to the relays and the Heating Elements.

4.6.1 Subsystem Hardware

This subsystem contains all relays ,the Power source and the Heating Elements.

4.6.2 Subsystem Operating System

This subsystem will have no operating system.

4.6.3 Subsystem Software Dependencies

There is no software dependencies.

4.6.4 Subsystem Programming Languages

There is none all programming and control will be done by the digital components layer.

4.6.5 Subsystem Data Structures

The data structure will be similar to the others in testing and requirements.

4.6.6 Subsystem Data Processing

For data processing we should be stuttering the on and off states of the relays to simulate a more precise control of the power flow.

4.7 Thermometer Control Subsystem

This subsystem will be providing the power to the thermometer

4.7.1 Subsystem Hardware

The subsystem contains the power supply and the Thermometer.

4.7.2 Subsystem Operating System

This subsystem will have no operating system.

4.7.3 Subsystem Software Dependencies

There is no software dependencies.

4.7.4 Subsystem Programming Languages

There is none all programming and control will be done by the digital components layer.

4.7.5 Subsystem Data Structures

None you will simply be providing power

4.7.6 Subsystem Data Processing

None you will simply be providing power

5 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.

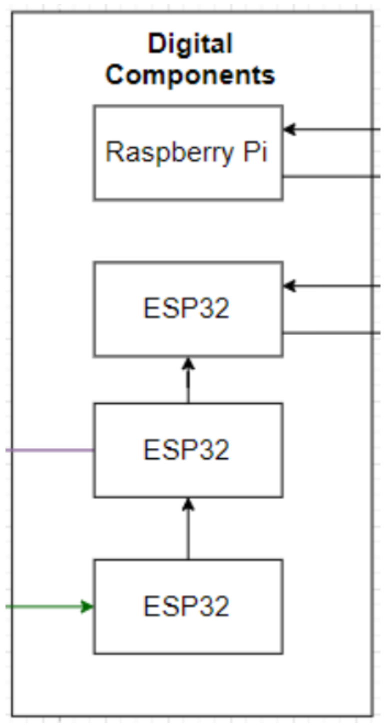

Figure 4: Digital Components Subsystem

5.1 Digital Components Layer Hardware

The hardware used in the digital components subsystem will made up of two different types of electronics. The computer that acts as the arbiter of tasks will be a Raspberry Pi 3b+. The microcontrollers used are ESP32s.

5.2 Digital Components Layer Operating System

The Raspberry Pi 3b+ will run Raspbian OS. There will not be an operating system on the microcontrollers.

5.3 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.

5.3.1 Subsystem Hardware

The hardware involved in this layer will consist only of a Raspberry Pi 3b+.

5.3.2 Subsystem Operating System

The operating system used will be Raspbian OS.

5.3.3 Subsystem Software Dependencies

This subsystem will use the Mosquitto MQTT Message Broker.

5.3.4 Subsystem Programming Languages

This subsystem will at the most use a scripting language such as BASH.

5.4 Temperature Sensor 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.

5.4.1 Subsystem Hardware

The hardware involved in this layer will consist of a microcontroller ESP32 and a temperature sensor MAX6675.

5.4.2 Subsystem Operating System

None will be used for this subsystem.

5.4.3 Subsystem Software Dependencies

Arduino libraries will be used for the temperature sensor.

5.4.4 Subsystem Programming Languages

C++ will be used to program the ESP32.

5.4.5 Subsystem Data Structures

The ESP32 will read data from the temperature sensor, and send it to the Raspberry Pi for a decision to be made. The data will be passed via the MQTT protocol.

5.5 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.

5.5.1 Subsystem Hardware

The hardware involved will be an ESP32 and a 120V heating element.

5.5.2 Subsystem Operating System

None will be used for this subsystem.

5.5.3 Subsystem Software Dependencies

Arduino MQTT libraries for the ESP32.

5.5.4 Subsystem Programming Languages

C++ will be used to program the ESP32.

5.5.5 Subsystem Data Structures

Commands will be sent to the ESP32 via MQTT messages.

5.6 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.

5.6.1 Subsystem Hardware

The hardware for this subsystem will consist of a single ESP32.

5.6.2 Subsystem Operating System

None will be used for this subsystem.

5.6.3 Subsystem Software Dependencies

Arduino MQTT libraries for the ESP32.

5.6.4 Subsystem Programming Languages

C++ will be used to program the ESP32.

5.6.5 Subsystem Data Structures

Commands will be sent to the ESP32 via MQTT messages.

6 Web Service Layer Subsystems

Since web service API and Database are hosted on Cloud service, no hardware is required. This section will discuss API tool platforms and databases.

6.1 Web Service API

API is an application interface that integrates two applications or systems to support the transfer of data to each other. Mulesoft is a platform where provides many convenient functions to build and deploy API. Using Mulesoft make it easier to connect MySQL.

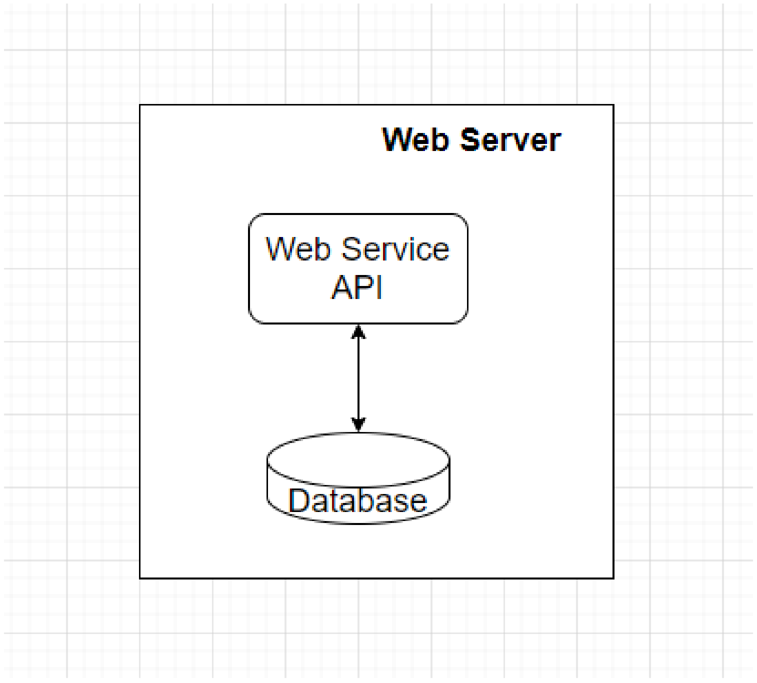

Figure 5: Example subsystem description diagram

6.1.1 Web Service API Software Dependencies

MuleSoft Exchange provides many dependencies and other APIs that are needed to build our API through a connector that prebuilds connectivity to an endpoint.

6.1.2 Web Service API Programming Languages

DataWeave is the MuleSoft expression language for accessing and transforming data that travels through a Mule app.

6.1.3 Web Service API Data Structures

MuleSoft Design Center has a function called API specification which helps to define incoming and output data format and creates automated documentation for the APIs. Through API specification, the general structure of API will be provided.

6.1.4 Web Service API Data Processing

API will listen to HTTP requests and based on a certain path, it will retrieve data from the database and return it to the user.

6.2 Database

The online database allows you to store sensor readings from the ESP32 and read from anywhere in the world by accessing your server domain.

6.2.1 Database Programming Languages

A query is used to communicate with the database.

6.2.2 Database Data Structures

There are some required tables that the database has to have such as temperature for each component and recipe.

6.2.3 Database Data Processing

The database will Run the query for requests and return the data to API.

7 User Interface Layer Subsystems

7.1 User Interface

The website used to control the system will be compatible with major web browsers and will be connected to a database and with the brewing system. Website is verified to work on Chrome for Windows 10, version 91.0.4472.124, and Chrome for MacOS Big Sur, version 91.0.4472.124

7.2 Layer Operating System

Website available on Windows 10 version 21H1 and MacOS Big Sur version 11.4

7.3 Layer Software Dependencies

The user interface components will depend on Mulesoft's Mule software, version 4.3.0

7.4 Website

The website provided to the user will have multiple responsabilities. First, it will display real-time information relevant to the user about the current brew. The website will also display a controls section for the user to interact with the brewing system. The information send to and from the website will be sent to the servers and then to the brewing system utilizing the above mentioned Mulesoft software.

7.4.1 Subsystem Operating System

No specific operating system required.

7.4.2 Subsystem Software Dependencies

The website and UI will utilize the React JS library

7.4.3 Subsystem Programming Languages

JavaScript will be used for creating the website.

8 Appendix A

Include any additional documents (CAD design, circuit schematics, etc) as an appendix as necessary.