The Pill^3

Liam Kenney, Oriana Alcala, Fernando A. Oriundo, Jordan Schneider

University of Central Florida, Department of Computer and Electrical Engineering, Orlando, Florida, 32816, U.S.A.

***Abstract*— This paper demonstrates the electronic hardware and software design of the Pill^3. The Pill^3 is an automated pill dispenser designed to eliminate the inconsistencies of people and act as a reminder as to when the user should take their prescribed medication and/or vitamins. The Pill^3 provides the user with multiple notification systems through the use of an OLED display and speaker integrated in the device, as well as a smartphone application.**

***Keywords*— Stepper Motor, Microcontroller, 3D printer, PCB , OLED Display, Firestore, IoT.**

I. Introduction

One of the biggest problems in the medical industry is the inconsistency with which patients follow their doctor's instructions. Incorrect dosages, forgetfulness, and substance abuse are among the issues that surround prescription medication and greatly affect the health of millions of patients. In the United States, non-adherence rates for chronic conditions like diabetes are as high as 50%, and cause around 125,000 deaths per year. To help alleviate these issues, there are many devices in the market available to help patients keep track of prescriptions, such as plastic pill containers that are separated by weekday, or automatic pill dispensers that are refilled via pharmacy delivered trays. These devices, while helpful, still have issues, primarily of which are the price and functionality. The two readily available market models revolve around monthly subscription payments for access to an app or a one time purchase of a device, which increases the price of the overall product.

To solve the previously mentioned issues, we decided to create the Pill^3. The Pill^3 is an automated pill dispensing device that would alert the consumer when it is time to take their medication based on the schedule provided by them. This device will be affordably produced, not requiring expensive components or costly server maintenance, it will strive to deliver a streamlined setup and refill experience that makes it both simple to maintain and easy for patients or caregivers to operate. A smart refill compartment makes adding new medication a breeze. The sleek design helps to minimize fault points making it reliable, and the app interface provides a friendly environment to use for people of all ages. In addition, the Pill^3 has an OLED display and a speaker that will also notify the user when pills have been dispensed in the scenario that they don't have their smartphone close by.

II. Motivation

The motivation for the Pill^3 was derived from our own personal lives and the issues many Americans around the United States face on a daily basis. These issues are especially prevalent in elderly patients and those with cognitive diseases where it can be difficult for them to remember if they've taken their medicine, or even know how much they're supposed to have. The Pill^3 seeks to create a tool patients can use to accurately track prescription dosage, timetables, and that will dispense the correct amount of medication according to the information given.

III. S*pecifications*

The specifications for the Pill^3 can be divided into two sections, the hardware and software components. For the hardware, the design and components will be discussed, and as for software, we will discuss the functions programmed on the MCU as well as the app development of the Pill^3. Below is an overview of what each component brings to our device.

A. Hardware Specifications

The hardware involves both the electrical and mechanical aspects of the Pill^3. It specifies what components will be on the PCB , how it is powered, the measurements and the functionality of the overall device.

- The device occupies a space within 1.5 ft^3
- The device will contain 30 days of medication with pills sized at 14 mm
- The device will have three dispersal systems
- The device will be powered via AC/DC converter
- The device will contain a microcontroller
  - Used to manage in device systems
    - Time
    - Notifications
    - Mechanical interfaces
- The device will contain a speaker capable of producing 8 kHz sounds for audio cues, which is a safe frequency for the user to be exposed to
- The device will contain a 30x2mm display for visual information and cues
- The device will contain 3 IR sensors to detect movement of pills
- The device will contain an apparatus to dispense medicine into a cup
- The device will have a pressure sensor that can tell there is a cup to dispense the medicine
- The device will contain a temperature sensor that will detect if the device is overheating
- The device will have a switch made for testing purposes only

B. Software Specifications

The software will involve the functionality of the mobile application and the MCU. The app will act as one of the main forms of notification as to when the pills are dispensed and when it is time to take their pills.

- ❖ The device will connect to a phone app via 2.4 GHz wifi
- ❖ The app will send push notifications when medication is dispensed
- ❖ The app will send push notifications once it is time for the user to take the medication
- ❖ The user will be able to set when pills should be dispensed via app
- ❖ The app is connected to an online Firestore database which in turn is connected to the Pill^3 MCU to guarantee the full functionality of the device.
- ❖ The MCU should be able to get the signal from the app and display it in both phone screen and the actual device screen
- ❖ The MCU should be able to sense when the medication was dispensed
- ❖ The MCU should be able to determine when the cup is placed in dispensing area
- ❖ The MCU should be able to determine when to dispense the pill based on scheduled time
- ❖ The MCU should be able to activate the motors when it's time to dispense the pill

IV. Hardware System Design

The Pill^3 consists of two subsystems. The first subsystem pertains to the electronic hardware and mechanical aspect of the device. This is split into three main parts; the storage container, the medicine dispersal motor, and the medicine IR sensor. When these components are working in tandem it is able to store, dispense, and determine how much medication is left in the Pill^3. The second subsystem pertains to the user notification system. Its main purpose is to alert and notify the user of the status of the Pill^3 through the use of visual, audio and smartphone notifications. This section will discuss the uses and inner workings of these components in further detail.

A. Dispersal System

For our device to be successful, it must be able to take in a number of pills and dispense it at the time set by the user via the mobile application. In order to store and contain a months supply of prescribed medication and vitamins, we decided to make the Pill^3 a 1.5 ft^3 chassis. Inside the chassis, there are three 4 inch tall cylindrical PVC pipes with an interior diameter of 4.5 inches. Within these pipes are 3 disks made from 3D printed filament with a 4.4 inch diameter, leaving a gap of 0.1 inches between the disks and the wall of the cylinder. This is so the disks can fit snugly within the cylinder and still have room to rotate. In the center of the bottom disk there will be a hole for the motor shaft. This will be connected to the disk in the cylinder and to the stepper motor at the bottom of the cylinder, thus allowing the disk to be able to rotate independently of the cylinder. Connected to the cylinder at a level 0.05 inches above the disk will also be the feeding tract layer which is a wall of filament that is placed slightly above the disk and covers the left half of the disk, separating the disk into two halves, the storage side (right side) and the sorting side (left side). At the top of the storage side there will be a curved edge leading to a small opening. This opening connects to a linear path from the storage side of the disk to the sorting side. Once on the sorting side of the feeding tract, the linear path curves towards the bottom of the disk and begins to split. When it is time to dispense the pills, the stepper motors will activate, spinning the pills onto the sorting part of the disk and dispensing them. An IR sensor is attached outside of the cylinder to count the number of pills dispensed. When the correct numbers of pills are dispensed, a signal is sent to the motors to stop and the user is then notified via push notification and given an audio and visual alert to take their medication.

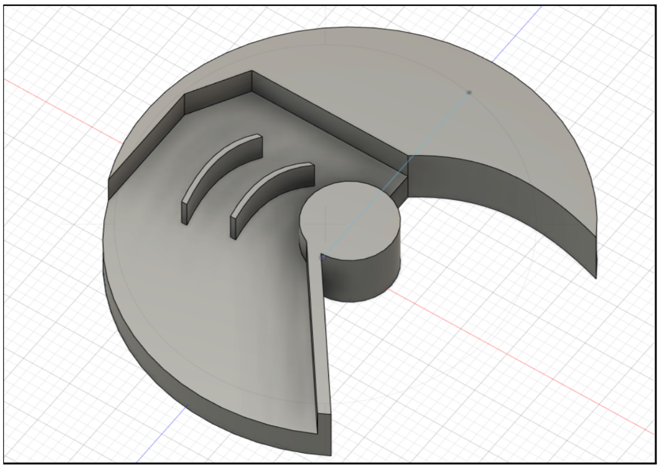

Figure 1 : Sorting Disk

B. Power Supply

In order for our device to operate, we require a constant source of power. The Pill^3 is meant to be a stationary, at-home device. The use of batteries or solar panels can tend to cause unnecessary problems as changing batteries can be an issue for people with joint problems and solar power wouldn't provide the necessary power for the components. For this reason, we opted to use AC power from a wall outlet. Standard household outlets generate 120 V and 60 Hz AC of electricity. Due to this, we decided to use an AC/DC converter that will convert the voltage generated by the household to 12 DC V and provide a current of 3A.

C. Calculations

Based on the selected AC/DC converter, calculations were needed before the implementation of the PCB design. Due to the use of low power modes, all the components present in the PCB design won't be active at the same time. Because of this, only the components used during the dispensing and notification process will be taken into account when making the calculations. The values to be considered in this process are the current drawn from each component. The idea is to get the sum of all of the active components and its result should be less than the net current value provided by the source, in this case the AC/DC converter. The calculations are shown below:

$$I_{net} > \sum I_{drawn}$$

$$I_{drawn} = 0.02 + 0.35 + 0.001 + 0.000430 + 0.0000081 + 2 + (0.007) * 20$$

$$I_{drawn} = 2.5114381$$

$$3\,A > 2.51\,A$$

As we can see in the calculation above, if we take into account all active components that are drawing current, the overall amount drawn is still less than the value provided by the source. This calculation was completed before the design was implemented in order to avoid a voltage drop in the power supply caused by overloading. Our main concern regarding this was the stepper motor controllers, as they are the ones that draw the most current.

In the table below it shows the voltage and current requirements for each component that will be placed in the Pill[3]. The values present in the table were taken in account during the calculations and design:

Table 1: Voltage requirement for hardware

Hardware	Voltage Requirement	Current Requirement
Arduino Iot 33	7 V - 21 V	7 mA per I/O pin
Adafruit NEMA-17 Motor	12 V	0.35 A
Obstacle avoidance IR Sensor	3.3 V - 5 V	20mA
Pressure Sensor	5 V	1 mA
MakerFocus Display	3.3 V - 5 V	430 uA
LMT85LP Temp. Sensor	3.3 V - 5 V	8.1 uA
Controller	1V - 36V	2A

Table 1 shows the current and voltage requirement of each component. As seen here, only the MCU and stepper motors will be functioning under 12 DC V while the other components will run with a 5 DC V. This was taken into account when designing the PCB. Since the AC/DC converter already generates the 12 DC V, it will be directly connected to the stepper motors and MCU via PCB. The other components require 5 DC V and will acquire this through a singular voltage regulator, regulating the 12 DC V to 5 DC V. While multiple regulators could have been implemented to lessen the load from our components, we are using low power modes in our code and each component consumes very little power. Due to the nature of the device, not all the components present in the Pill[3] will run at the same time, there will be a time gap between motors and sensors. The basic setup of our PCB can be seen in Figure 1.

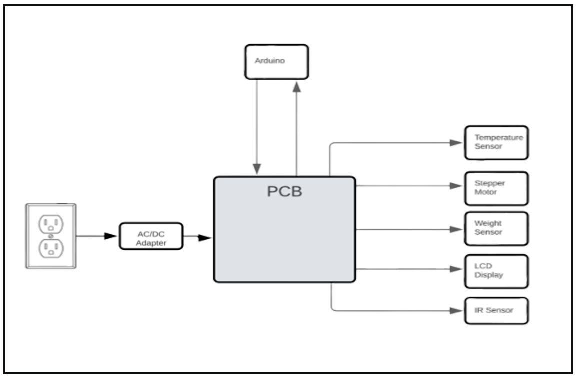

Figure 2 : Hardware Block Diagram

D. Motor

To dispense the medicine held in the storage container, an electric stepper motor with low speed and high torque is used to rotate the sorting/storage disk. It is attached to the sorting disk, allowing it to rotate along the bottom of the storage container. A stepper motor has both a full 360 degree range of motion and precise control over stop, start, and location. In order to operate the stepper motors, a signal is sent through the PCB, from the digital pins of the MCU, allowing it to activate when it is time to dispense the pills. This subsystem uses a passive feeding tract to determine a medicine's size, so it is able to space them appropriately. Since the motor does not need to move very quickly, the device is designed to only allow one pill to be dispensed at a time, so a slower motor allows adequate time for the sensor to count as the pills are dispensed. For the motor to properly communicate with the MCU a controller was added for each of the motors in the PCB, the idea of these controllers is to provide step and direction outputs to the driver. For the Pill[3] the controller will be utilized to control the acceleration and the steps per second. This controller also provides an interface to and control any other external signals.

E. IR Sensor

To determine if a pill has been dispensed through the sorting disk, an infrared sensor is used to track pills leaving the feeding tract layer of the disk. The infrared sensor powered through the PCB receiving 5 DC V is positioned outside of the cylinder, aimed perpendicular to the exit of the container. While there is no object directly in front of the sensor or if there is a strip of black material in front of the sensor, the infrared light emitted from the clear-colored emitter diode has no surface to reflect off of. This means that the dark-colored detector diode of the sensor does not observe any IR light. Once a pill crosses in between the sensor and the black material, the infrared light now has a surface to reflect off of and the sensor is able to detect the change in status, sending a signal back to the MCU.

The change in status is used to both accurately count the number of pills distributed and quickly and reliably inform the system of when this change occurs. Due to the nature of the storage device itself, it is difficult to determine the number of pills in the system. As such, the sensors will have a timer associated with them, and if a pill has not crossed in front of the sensor before the time is up, the device will alert the user that there are no more pills in that cylinder. The second most important quality of the infrared sensor's position is its role in stopping the rotating disk. Once a pill is counted, the device will be able to receive the information from the infrared sensor and send the command to stop the stepper motor within 0.5 seconds, to prevent any unwanted extra pills from being distributed.

F. OLED Display

As a form to notify the user when their pills have been dispensed, we implemented an OLED Display. The OLED display is a 30x2mm screen and requires 5 DC V to operate. It is connected to a voltage regulator through the PCB to receive its needed power. We soldered header pins onto the PCB and wires to make the connection sturdy. The display is activated when the MCU sends a signal through the PCB. When the signal is received, a visual alert is displayed 3 inches above the cup. It is mounted onto the chassis and can be clearly seen by the user. The OLED acts as a secondary form of notification if the user does not have his/her smartphone with them at the moment of dispensing. It is also in place if the user has trouble hearing and/or misses the other notifications that are sent out through the smartphone application and audio cue. While not dispensing pills it cycles through several idle screens, providing the user with information about the time and device statuses like temperature.

G. Audio Alert

For a tertiary form of notification, the Pill^3 has an audio alert which functions like a cellphone ringtone. This, along with the OLED and the smartphone application form the completed notification system. The speaker is also inexpensive and not difficult to implement either since it is a simple hardware device that can communicate easily to the microcontroller. The speaker does not require power for it to operate, so it is directly connected to the MCU through the PCB. It is mounted at the top left of the chassis and produces short high frequency sounds which have been tuned to musical notes. It is implemented if the user is hard of seeing and/or misses the push notification from their smartphone.

H. Temperature Sensor

The temperature sensor in the device is used to monitor the heat dissipation of the PCB and components attached. It is used solely for testing purposes. While testing, we were able to see that not much heat is produced from our components. The temperature sensor requires only 5 DC V to function. It gets its power from the voltage regulator that is connected to the AC/DC converter through the PCB. While testing, the MCU would send a signal to the temperature sensor and record any irregularities that may occur when the Pill^3 is in operation. It was mounted inside the chassis, alongside the MCU, PCB and stepper motors. As a safety precaution, if the temperature sensor ever reaches above a threshold of 85 degrees, the device will notify the user that it is an unsafe location to store pills, and will turn off the microcontroller to prevent possible damage to electrical components.

I. Pressure Sensor

The pressure sensor is utilized to detect when there is a cup in the dispensing area. The idea of the sensor is that if there is not a cup in the dispensing area, no pills will be dispensed. This helps avoid the pills being dispensed to the floor or getting lost inside the Pill^3. This sensor is connected to the PCB in which it obtains a voltage of 5 DC V in order to function. The sensor activates when there is a change in resistance. This causes a change in voltage and in turn will send a signal to the MCU to notify a cup has been placed and pills can be dispensed.

J. MCU

The Pill^3 contains an Arduino Nano Iot 33, this device is the one in charge of all the components in the PCB. This MCU has a total of 30 pins that all previously mentioned components are connected to. Since this board is able to connect to the established app through wifi, the user is able to send the information of their pill schedule as well as the quantity of pills to be stored in the Pill^3 through Firestore to the MCU. This board is able to send the proper commands to the motors once it's time for a specific pill to be dispensed. Once the instruction has been sent, the MCU will be waiting for the IR sensor to detect if the pill left the container. If the pill successfully left the container, the user will be notified that the pill is in the cup. The MCU is able to do all this as long as the cup has been placed in the corresponding area.

K. PCB

The PCB was designed multiple times in order to enhance the original design. During this time the components remained the same, as for the MCU originally selected, we decided to change due to some drawbacks faced with the overall design. The PCB shows the following connections to the MCU:

- Motors & Controllers
- Weight sensor
- OLED
- Speaker
- IR Sensors
- Temperature sensor

The PCB takes an input voltage of 12 DC V, this voltage is directly used by the motors and the MCU. For the small components in the PCB, a voltage regulator was included. This regulator takes 12 DC V from the source and steps it down to 5 V for all of the other small components. The AC/DC converter that is connected to the PCB is able to prove the needed voltage and a current of 3 amperes. A switch was also included in the PCB design to facilitate the testing process, lastly we made sure the PCB has the mounting holes in order to place it inside the Pill^3 without any inconvenience.

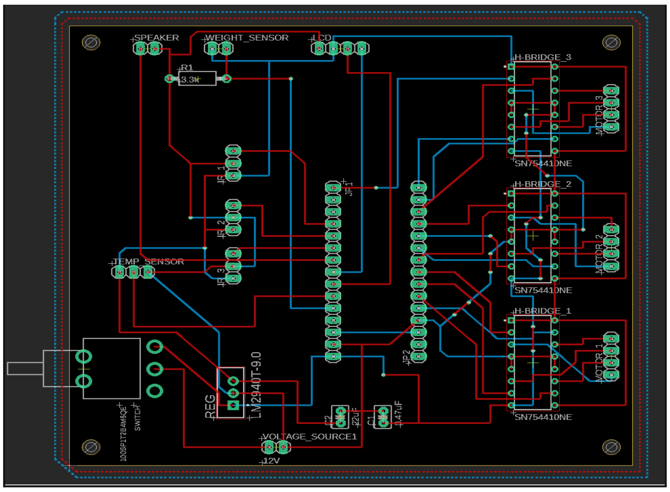

Figure 3: PCB Schematic

L. Chassis

For the Pill^3, we had a plethora of options to choose from when it came to the chassis. Filament, wood, metal and acrylic were amongst the top choices for our device. However, each type of material had their own difficulties. Metal and acrylic proved to be too expensive as well as being difficult to shape since we were not equipped with the right tools. Filament was difficult to use for a device of this stature as it was far too brittle and prone to breaking. Additionally, due to the planned size of the device the outer walls would have to be printed in multiple pieces, further reducing stability. Due to these factors we opted to use plywood, as it was inexpensive, solid, and easy to mold into the shapes we needed. We cut each piece of wood to be a 1.5 x 1.5ft structure.

Figure 4: Exterior of Chassis

For the exterior of the Pill^3, we decided to drill the bottom and three sides of the chassis to each other. The right, left and bottom parts were untouched as it didn't need to be modified. The front had sections cut off of it to make room for the cup and to mount the OLED display and speaker. This can be seen in figure 2. As seen in the figure below, the top of the chassis has three cylindrical holes cut into it. This is where the pills will be placed by the user. Instead of drilling this onto the rest of the chassis, we decided to add hinges to the back and top of the chassis to have quick and easy access to the rest of the components. These hinges will not be present in the final version of this product and are only for showcase purposes.

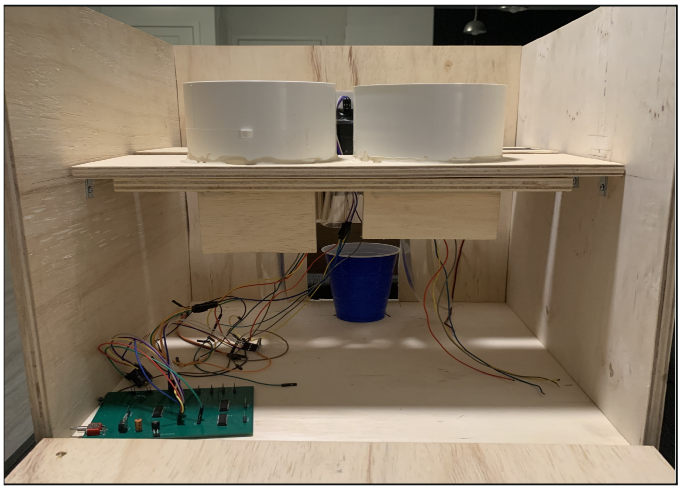

Figure 5: Interior of Chassis

For the interior of the chassis we added a shelf to hold our dispersal system and stepper motors. This can be seen in figure 2. The PCB will be placed in the back right of the chassis and the wires will be organized throughout the inside to avoid confusion and disarray. By having a big area to work with, it is ensured that the correct pills can be dispensed and a large number of pills can be stored in our device.

V. Software System Design

We begin with having the microcontroller be the link that combines both our hardware and software diagrams. The phone, where the user will input their information, sends that information to Firestore, an online database where all data on the pills will be stored. From the database information regarding a prescription (when to dispense the pills, what storage chamber they're in, and how many pills to dispense at a time) is sent to the microcontroller on board the device. The microcontroller and smartphone application then independently keep track of when the pills are dispensed and take the appropriate actions when that time arrives.

For example, if a user wants to set a time that they want a certain pill to dispense, that is not something that has to be sent directly to the microcontroller, rather it has to be transferred through the backend to the database to store/update the times a user wants a pill time updated.

When the user loads the phone, it will have to request information from the database which will run through the controller module. Once the application boots up, it can request the information necessary, then display that information such as times already set for pills to dispense, what pills at what times have been dispensed, etc.

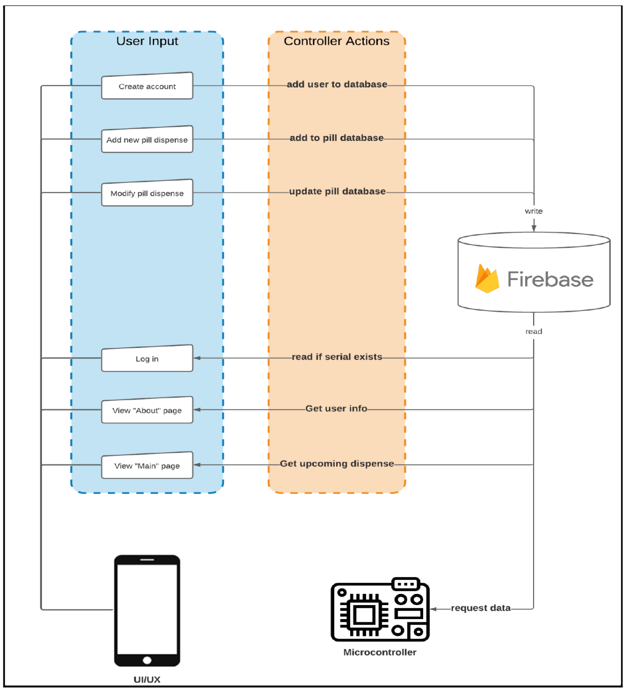

Figure 6 : Software Block Diagram

Firebase by Google is the technology being used for data storage in Pill[3]. Firebase uses a NoSQL database storage method which stores data as JSON objects. This can be leveraged to efficiently store user data and easily store and fetch the information necessary for the application. Specifically in regards to data storage in Firebase, we will be using Firestore, which allows for this simple storage method of essentially cloud-stored JSON files. Using Firestore allows for structured documents compared to Real Time Database's simple JSON-tree.

While thinking about database design, we first need to acknowledge elements we need to have stored:

- Device's serial number
- User's email
- User's name
- Pill dispensing
  - What pill
  - What slot to dispense from
  - Dosage
  - When to dispense
    - What days to dispense
    - What times to dispense on each day

With this information, we know that we have to have our users enter data for pill identification, how many pills to dispense, what days to dispense them, and what times to dispense them. The UI for this can be as simple as text fields for name and number of pills to dispense, while days can be done using a simple multi-selector bootstrap from react-native-elements. However, what times to choose is slightly more complicated; we opted for a time-selection tool to make the UX easier. This way, we wouldn't have to worry about the different ways people can write time (1pm, 1:00pm, 13:00, 1300). This allows for less user-error and a better user experience when choosing a time.

For our backend, we used a simple controller schema where the application would just make calls to both a pill-service and user-service which would handle fetching and updating data in Firestore.

Below is an illustration of how it works:

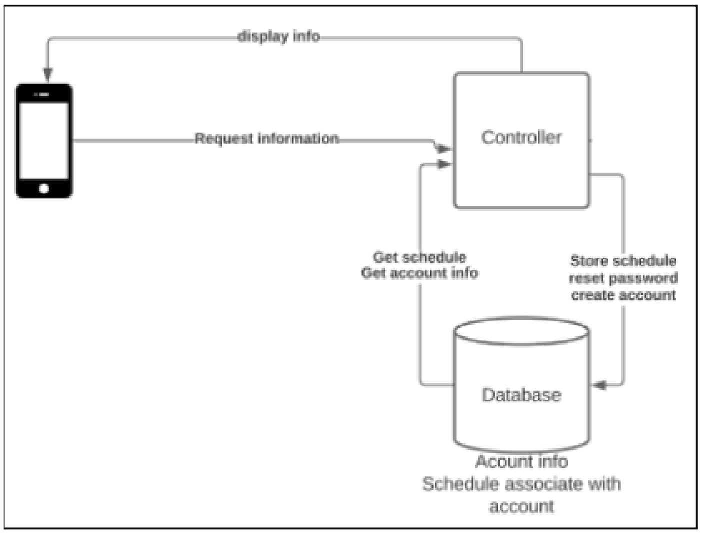

Figure 7 : Backend Software Block Diagram

The mobile application is written using React-Native so that it can be run on both iOS and Android devices. Some important libraries we are using are:

- firebase-app for communication with firestore
- react-native-redux for managing a global application state
- react-navigation for moving between different screens efficiently
- react-native-elements for our UI components' styling and bootstrap

The primary screen, the dashboard- is composed of a weekly calendar and the navigation bar which lets users see their information or add a new pill. All their pills currently stored will be visible from the weekly calendar. Another way of editing the pill storage is by holding down on the pill buttons; this allows users to either choose to delete or edit the pill entry.

The subsystems and software described above will be used in conjunction with each other to inform the consumer of any status changes or updates, and precisely deliver the medicine to the user at user-chosen times. To achieve all this, the subsystems need to be arranged and designed in a way that makes them both intuitive for the user to interact with and reliable in regards to storage conditions and timing.

While waiting for additional updates to the Firestore database or in between pill dispensings, the microcontroller is in a low power idle mode, waking up approximately once every 10 seconds to check the pill data and current time. While in this low power mode, the current draw drops from a peak of approximately 250 mA to as low as 6 mA at the MCU level. When a change is detected in the database, all pill information is downloaded to the microcontroller and added to pill fields on the board to be used in the later functions. If the microcontroller detects that the time to dispense a pill has arrived, it'll start up several functions and exit low power mode. The first function activated will be the user notification function, wherein a preselected jingle is played through the device's speaker and the OLED screen asks the user to place a cup on the dispensing pad. The controller will then wait until the weight sensor detects a cup before activating the second function, the pill dispensation function. This function rotates the motors quickly in increments of 10 degrees until the IR sensor associated with that motor's cylinder detects that a pill has been dispensed. This function is repeated according to the set number of pills needing to be dispensed and runs through each cylinder separately starting with the first one. After the pills have been dispensed and the user has taken them, a message is shown on the OLED screen and the device returns to its idle mode.

VI. Challenges

During the building of the Pill^3 we encounter certain challenges along the way. Although this device is functional in its current state and achieves all the goals laid out at the beginning of the project, there are still some things that could be further improved given more time and resources.

1. Gel Pills: during the Pill^3 design stage, all kinds of medication were taken into account, however, there are pills that have different consistency. After further research we came to conclude that these kinds of pills are better stored in their original container. For the most part these pills tend to stick together due to gel material. If it is placed in the Pill^3 it can cause the pills to stick together, thus becoming unable to dispense.
2. Acrylic sheets: at the early stages of the design we considered acrylic sheets that would be cut to be part of the dispensing system, acting as the rotating and sorting disks. Due to pricing limitations regarding this resource and the significantly lengthier process of prototyping with the acrylic material, we decided to design a 3D model of the disks previously mentioned. This choice alone reduced the final price of the finished product by approximately $40.00 and allowed us more freedom with creating, testing, and adjusting prototypes. If we had stuck with the acrylic sheet, each prototype alone would have cost upwards of $20.00 and would have had twice the turnaround time that a 3D printed prototype did.
3. PCB iterations: Throughout the Pill^3 design we started with a different microcontroller, which was limited with the amount of digital pins but housed hardware that significantly simplified the digital connection to the database. To solve the limited number of pins available, we added a multiplexer. The idea was to have two of the three stepper motors connected to it, to then connect it to the MCU. While doing so, we encountered a problem where the stepper motors needed constant inputs that we could not replicate using the multiplexer. Due to these circumstances, we decided to re-design the PCB and change the MCU. The second design contained an Arduino nano IoT 33, which provides more analog and digital pins. After testing the design, it showed positive results. To enhance our final design we decided to print one more iteration with some pin configuration changes and mounting holes that previous versions did not have.

VII. Conclusion

The purpose of the Pill^3 is to provide an affordable and easy to use automated pill dispenser for the masses that would help eliminate the inconsistencies people have when taking their prescribed medication and/or vitamins. Its easy to use mobile application and a friendly environment made it so that anyone would be able to schedule when their medication needs to be dispensed and be notified as to when to take their medication. The visual, audio and smartphone notifications make it so you can never forget to take your needed pills.

With an overall design consisting of one MCU communicating with six hardware components via a PCB, mass-producing this device would come at a low cost. This is due to the simple, but effective design of our device. Throughout the process, we designed this device with the consumer in mind. Although we were able to create this, there could always be room for improvement. Possibilities for upgrading our device would be to include some extra features onto the hardware and software aspects of the Pill[3]. For the software, an index of side effects that a medication may have or possibly being able to communicate with an Alexa or Google home. For the hardware, reducing the size of the chassis might be more desired by the user so that the device can be placed mostly anywhere. Overall, we were able to implement our gathered engineering knowledge onto this device and came together to work as a team.

VIII. The Engineers

Liam Kenney is currently a senior Electrical Engineer at the University of Central Florida. He has interned at Qorvo as an electrical engineer and will be joining Qorvo as a Development Engineer in Summer 2022.

Oriana Alcala is currently a senior Electrical Engineer at the University of Central Florida. She has interned at JST Power Equipment, Nelson Engineering and currently works at Burns and Mcdonnell. She will be joining Burns and Mcdonnell soon after graduation.

Fernando A. Oriundo is currently a senior Electrical Engineer at the University of Central Florida. He has interned at CUMMINS INC, Nelson Engineering and will be joining Northrop Grumman as a Systems Engineer in their pathways program in August 2022. He hopes to get his MBA in the years to come.

Jordan Schneider is currently a senior Computer Engineer at the University of Central Florida. He interned at Amazon as a Software Engineer this past summer and will be joining Amazon as a Software Development Engineer soon after graduation.

Acknowledgment

The authors of this paper would like to thank Samuel M. Richie and Lei Wei for their feedback and guidance throughout this process.

We would also like to thank the committee for their time throughout this unprecedented time and the University of Central Florida for shaping us into being the next wave of engineers.

We would also like to express gratitude to the UCF engineering department for providing us the tools to be successful. The TI Innovation lab and Senior Design lab was a huge asset to our project. We will take the lessons learned from this experience and apply it to our future jobs as engineers.

References

[1] Connecting Arduino to Firebase to send & receive data. In: Arduino Project Hub. https://create.arduino.cc/projecthub/electropeak/connecting-arduino-to-firebase-to-send-receive-data-cd8805. Accessed 13 Nov 2021

[2] Health-Care Waste. In: World Health Organization.https://www.who.int/news-room/fact-sheets/detail/health -care-waste. Accessed 27 Oct 2021

[3] "Infection control: Buffs and grinding ... - oaktree products," *https://www.oaktreeproducts.com*. [Online].Available: https://www.oaktreeproducts.com/img/product/description/IC-Buffs%20and%20Grinding%20Stones%20for%20Modification.pdf. [Accessed: 05-Apr-2022].