**Smart Medicine Dispensing System using IoT**

## Introduction

Our smart medicine dispensing system is targeted on users who regularly take drugs or vitamin supplements or nurses who take care of the older or patients. The medicine box is programmable that allows nurses or users to specify the pill quantity to take and the serve time for each day. The smart medicine box contains seven separate sub-boxes. Therefore, nurses or users can set information for seven different pills. When the pill quantity and serve time has been set, the medicine box will remind users or patients to take pills using sound and light. The specific number of pills needs to be taken will be displayed by a seven segment led display placed on the corresponding box. Compared with the traditional pillbox that requires users or nurses to load the box every day or every week. Our project would significantly release nurses or users’ burden on frequently preloading pills for patients or users.

**Introduction to IoT**

### IoT devices

Devices are generally made up of a circuit board with sensors attached that connect to the internet. Many devices communicate via a Wi-Fi chip. Here are some examples of IoT devices:

* pressure sensors on a remote oil pump
* temperature and humidity sensors in an air-conditioning unit
* accelerometers in an elevator
* presence sensors in a room

Two devices that are frequently used for prototyping are the basic MX Chip IoT Devkit from Microsoft and Raspberry PI devices. The MX Chip Devkit has sensors built in for temperature, pressure, humidity, as well as a gyroscope and accelerometer, a magnetometer and a Wi-Fi chip. Raspberry PI is an IoT device to which you can attach many different kinds of sensors, so you can select exactly what you need for your scenario.

The IoT Device SDKs enable you to build apps that run on your devices so they can perform the tasks you need. With the SDKs, you can send telemetry to your IoT hub, receive messages and updates from the IoT Hub, and so on.

### Communication

Your device can communicate with back-end services in both directions. Here are some examples of ways that the device can communicate with the back-end solution.

#### Examples

* Your device may send temperature from a mobile refrigeration truck every 5 minutes to an IoT Hub.
* The back-end service can ask the device to send telemetry more frequently to help diagnose a problem.
* Your device can send alerts based on the values read from its sensors. For example, if monitoring a batch reactor in a chemical plant, you may want to send an alert when the temperatures exceeds a certain value.
* Your device can send information to a dashboard for viewing by human operators. For example, a control room in a refinery may show the temperature and pressure of each pipe, as well as the volume flowing through that pipe, allowing the operators to watch it.

These tasks, and more, can be implemented using the IoT Device SDKs.

#### Connection Considerations

Connecting devices securely and reliably is often the biggest challenge in IoT solutions. This is because IoT devices have different characteristics when compared to other clients such as browsers and mobile apps. Specifically, IoT devices:

* Are often embedded systems with no human operator (unlike a phone).
* Can be deployed in remote locations, where physical access is expensive.
* May only be reachable through the solution back end. There is no other way to interact with the device.
* May have limited power and processing resources.
* May have intermittent, slow, or expensive network connectivity.
* May need to use proprietary, custom, or industry-specific application protocols.

### Back-end services

Here are some of the functions a back-end service can provide.

* Receiving telemetry at scale from your devices, and determining how to process and store that data.
* Analyzing the telemetry to provide insights, either in real time or after the fact.
* Sending commands from the cloud to a specific device.
* Provisioning devices and control which devices can connect to your infrastructure.
* Control the state of your devices and monitor their activities.

For example, in a predictive maintenance scenario, the cloud back end stores historical telemetry. The solution uses this data to identify potential anomalous behavior on specific pumps before they cause a real problem. Using data analytics, it can identify that the preventative solution is to send a command back to the device to take a corrective action. This process generates an automated feedback loop between the device and the cloud that greatly increases the solution efficiency.

## An IoT example

Here is an example of how one company used IoT to save millions of dollars.

There is a huge cattle ranch with hundreds of thousands of cows. It's a big deal to keep track of that many cows, and know how they're doing, and requires a lot of driving around. They attached sensors to every single cow, sending information such as the GPS coordinates and temperature to a back-end service to be written to a database.

Then they have an analytical service that scans the incoming data and analyzes the data for each cow to check questions like the following:

* Is the cow running a temperature? How long has the cow been running a temperature? If it has been longer than a day, get the GPS coordinates and go find the cow, and if appropriate, treat it with antibiotics.
* Is the cow in the same place for more than a day? If so, get the GPS coordinates and go find the cow. Has the cow fallen off of a cliff? Is the cow injured? Does the cow need help?

Implementing this IoT solution made it possible for the company to check and treat the cows quickly, and cut down on the amount of time they had to spend driving around checking on their animals, saving them a lot of money.

## IoT Services

There are several IoT-related services in Azure and it can be confusing to figure out which one you want to use. Some, such as IoT Central and the IoT solution accelerators, provide templates to help you create your own solution and get started quickly. You can also fully develop your own solutions using other services available -- it all depends on how much help you want, and how much control. Here is a list of the services available, as well as what you may use them for.

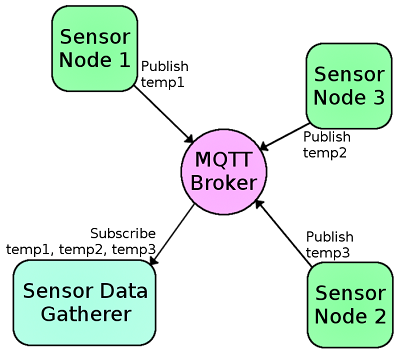
1. **IoT Central**: This is a SaaS solution that helps you connect, monitor, and manage your IoT devices. To start, you select a template for your device type and create and test a basic IoT Central application that the operators of the device will use. The IoT Central application will also enable you to monitor the devices and provision new devices. This service is for straightforward solutions that don't require deep service customization.
2. **IoT solution accelerators**: This is a collection of PaaS solutions you can use to accelerate your development of an IoT solution. You start with a provided IoT solution and then fully customize that solution to your requirements. You need Java or .NET skills to customize the back-end, and JavaScript skills to customize the visualization.
3. **IoT Hub**: This service allows you to connect from your devices to an IoT hub, and monitor and control billions of IoT devices. This is especially useful if you need bi-directional communication between your IoT devices and your back end. This is the underlying service for IoT Central and IoT solution accelerators.
4. [**IoT Hub Device Provisioning Service**](https://docs.microsoft.com/en-us/azure/iot-dps/): This is a helper service for IoT Hub that you can use to provision devices to your IoT hub securely. With this service, you can easily provision millions of devices rapidly, rather than provisioning them one by one.
5. [**IoT Edge**](https://docs.microsoft.com/en-us/azure/iot-edge/): This service builds on top of IoT Hub. It can be used to analyze data on the IoT devices rather than in the cloud. By moving parts of your workload to the edge, fewer messages need to be sent to the cloud.
6. [**Digital Twins**](https://docs.microsoft.com/en-us/azure/digital-twins/index): This service enables you to create comprehensive models of the physical environment. You can model the relationships and interactions between people, spaces, and devices. For example, you can predict maintenance needs for a factory, analyze real-time energy requirements for an electrical grid, or optimize the use of available space for an office.
7. [**Time Series Insights**](https://docs.microsoft.com/en-us/azure/time-series-insights): This service enables you to store, visualize, and query large amounts of time series data generated by IoT devices. You can use this service with IoT Hub.
8. [**Maps**](https://docs.microsoft.com/en-us/azure/azure-maps): This service provides geographic information to web and mobile applications. There is a full set of REST APIs as well as a web-based JavaScript control that can be used to create flexible applications that work on desktop or mobile applications for both Apple and Windows devices.

**Introduction to MQTT**

In a nutshell, MQTT uses your existing Internet home network to send messages to your IoT devices and respond to those messages.

MQTT Nodes

**Brief History**



MQTT (Message Queuing Telemetry Transport) is a publish/subscribe messaging protocol that works on top of the TCP/IP protocol. The first version of the protocol was developed by Andy Stanford-Clark of IBM and Arlen Nipper of Cirrus Link in 1999. What makes MQTT faster than say sending HTTP requests with your IoT device is MQTT messages can be as small as 2 bytes, whereas HTTP requires headers which contains a lot of information that other devices might not care about. Also, if you have multiple devices waiting for a request with HTTP, you'll need to send a POST action to each client. With MQTT, when a server receives information from one client, it will automatically distribute that information to each of the interested clients.

**The Basics**

Before you learn how to build a MQTT network, it will help to understand some of the jargon that's used and how each piece fits together to create your network.

Broker - The broker is the server that distributes the information to the interested clients connected to the server.

Client - The device that connects to broker to send or receive information.

Topic - The name that the message is about. Clients publish, subscribe, or do both to a topic.

Publish - Clients that send information to the broker to distribute to interested clients based on the topic name.

Subscribe - Clients tell the broker which topic(s) they're interested in. When a client subscribes to a topic, any message published to the broker is distributed to the subscribers of that topic. Clients can also unsubscribe to stop receiving messages from the broker about that topic.

QoS - Quality of Service. Each connection can specify a quality of service to the broker with an integer value ranging from 0-2. The QoS does not affect the handling of the TCP data transmissions, only between the MQTT clients. Note: In the examples later on, we'll only be using QoS 0.

0 specifies at most once, or once and only once without requiring an acknowledgment of delivery. This is often refered to as fire and forget.

1 specifies at least once. The message is sent multiple times until an acknowledgment is received, known otherwise as acknowledged delivery.

2 specifies exactly once. The sender and receiver clients use a two level handshake to ensure only one copy of the message is received, known as assured delivery.

**How MQTT Works**

As mentioned in the introduction, MQTT is a publish/subcribe messaging protocol. Clients will connect to the network, which can subscribe or publish to a topic. When a client publishes to a topic, the data is sent to the broker, which then is distributed to all the clients that are subscribed to that topic.

Topics are arranged in a directory-like structure. A topic might be "LivingRoom", or "LivingRoom/Light" if you have multiple clients within that parent topic. The subscriber client will listen for incoming messages from the subscribed topic and react to what was published to that topic, such as "on" or "off". Clients can subscribe to one topic and publish to another as well. If the client subscribes to "LivingRoom/Light", it might also want to publish to another topic like "LivingRoom/Light/State" so that other clients can monitor the state of that light.

Now that we understand the theory of how MQTT works, lets build a quick and easy example with a Raspberry Pi and ESP32 Thing boards to see it working in action. We'll start by setting up the broker and running a quick test to make sure it's working correctly.

**Literature Survey**

Ilkko et al4proposed UbiPILL A Medicine Dose Controller of Ubiquitous Home Environment (2009), Home automation and wireless sensor network which have enhancing the quality of life by providing security, information and comfort. Here had discuss a centric home server with three main roles: use of existing Interfaces on registered systems for remote monitoring and Control, serving the surrounding system as a data gateway and Providing content adaptive user interfaces enhanced by Belongings of end-user client devices, the ubipill device had implemented to remind people for elder and for monitoring purposes ubipill and home server have been design to reliably monitor the medicine box activity by web browser. Kliem et al5 proposed Security and communication architecture for networked medical devices in mobility-aware eHealth environments (2012), Telemedicine concept is cost efficient and location autonomous monitoring system, the suitable and secured medical data can be transferred with different devices with attention towards security and privacy issue. Emergency situations need on the flutter network integration and data transmission fluctuating from domains like patient home, medical practices, ambulances and, hospitals, where each domain may parallel to a different authority so, mobility aware approach allowing out of the box medical device integration and authentication, and simultaneously fulfilling the typical security and privacy requirements of e-health environments. Parida et al3proposed Application of RFID Technology for In-House Drug Management System (2012), RFID based technology have used to make drug management system, in this tracking of medicine can be done including emergency or regular medicine with or without RFID tag the HF tag have assigning the user and by employing RFID reader along with camera and web based system to track the user. This system can be beneficial for the old age, less educated people. Clifton et al2 A Self-powering Wireless Environment Monitoring System Using Soil Energy, proposed A large-scale clinical validation of an integrated monitoring system in the emergency department(2013),In the integrated patient monitoring which include electronic patient data which generally have more amount challenges to acquire cope with artefact data with the help of algorithm, analyzing and communicating the resultant data for reporting to clinician, here in this demonstrated the machine learning technology embedded within healthcare information system which provide clinical benefits for improving patient outcomes in busy environments. Hamida et al6 proposed towards efficient and secure in-home wearable insomnia monitoring and diagnosis system (2013), Due to the evolution in technology it is now possible to specific timing monitoring here delivers an experimental estimation of communication and security protocols that can be used in in-home sleep monitoring and health care and highlights the most proper protocol in terms of security and overhead. Design Procedures are then derived for the distribution of effective in-home patients monitoring systems. Ray et al7proposed Home Health Hub Internet of Things (H3IoT)( 2014) , Health is vital part of life and it is quite necessary to give priority health related issue in which digitization helpful by using number of devices through the concept of IOT but due to heterogeneity and interoperability the concept of digitization for health care is neglected, here in this the best focus given to architecture framework for human health hub which have envision of usage of real life implementation. Shivakumar et al8 proposed Design of vital sign monitor based on wireless sensor networks and telemedicine technology(2014), Vital sign monitor can be implemented with Bluetooth technology which is embedded with sensor, the transmitter will include the application oriented smart phone enable with 3G or IEEE 802.11 i.e. wi fi based transmission. The data from transmitter will be sending to cloud for centralized monitoring takes place; the expert in remote place can view all patient data and in case of emergency can take appropriate action.Ajmal Sawand et al1proposed Multidisciplinary approaches to achieving efficient and trustworthy eHealth monitoring systems(2014),The technological merging between IOT, wireless body area network and cloud computing have vital contribution in e health care which improve the quality of medical care, basically patient centric monitoring play a role in e health care services which involve medical data collection, aggregation, data transmission and data analysis here entire monitoring lifecycle and essential services component have discus as well as design challenges in designing the quality and patient centric monitoring scheme along with potential solution. Huang et al8 proposed the intelligent pill box—Design and implementation (2014), the implementation of pill box has proposed by keeping the problems of old age people in mind to provide full medication safety. The pill box will remind the patient about timing by doing this drug abusing can be controlled. Al-Majeed et al10proposed Home telehealth by Internet of Things (IoT) (2015), The real time monitoring can be possible through IOT which helps in development of low cost medical sensing, communication and analytic devices Samir V. Zanjal and Girish. R. Talmale / Procedia Computer Science 78 ( 2016 ) 471 – 476 which make quality of life, in case of density of messages there is fear of information degradation but by using proper algorithm we can resolve the problem and can make the low cost imaging, sensing and human computer interaction technology. Lin et al9 proposed A Self-powering Wireless Environment Monitoring System Using Soil Energy (2015),The monitoring system can uses the self-powering wireless environment with the help of renewable energy which can be beneficial in remote places where the power problem in wide manner, in this the system have demonstrated which will uses soil energy with carbon, zink electrodes.Moga et al11 proposed Embedded platform for Web-based monitoring and control of a smart home (2015), Present the low cost embedded platform for web based monitoring and controlling and the platform consist of distributed sensing and control network and touch screen to easy use interface to the user and remote web based access.

Taking Medicine at right time in proper amount will lead towards the faster recovery. In reality what happens is that, they get their prescribed medication but fail to follow their health care professional’s instructions. Many people while taking prescribed medication do not follow their doctor's’ instructions. Some common reasons for this are People may start feeling better and decide to not finish all of the medication .In some cases people think this medication is not working and they may not notice an improvement in their symptoms right away and may stop taking the medication. In some other cases people try to save money and they think that these medications are expensive, and people may skip doses or take less. According to World Health Organization, over 80% of the people above the age of 60 years are prescribed medicines that are to be administered 2 - 4 times a day. With the increase in Cardiovascular diseases and Diabetes among the peer group regular medicine administration has become a necessity. But among this another 40-60% is having the issues related to forgetting the taking of medicines at right time. We found several different pillbox products available in the market. The cheapest one was the traditional pillbox, which contained seven boxes for seven different days of a week. Such pillbox normally cost around 200 INR. However, user had to load the pills to the boxes every week. At the time of loading pills in the box, people many time mix their pills with other pills in the same box would increase the risk of making mistakes. Another type of pillbox in the market is also available, which had the sound reminder, and was able to remind the user to take medicine at user specified time but it only remind the user at once a day. The costs of this type of pillbox were so high about 1000 INR, Therefore, we think it was necessary to build a cheap and functional smart Medicine box that could bring more convenience for the user. We then defined the specifications of our device based on the user needs. From the literature cited, the research proposed an idea of Smart Medicine Box that will adapt the features of time tracking and alarm triggering Additionally, as compared to the existing system, It will remind the user to take medicine not for once per day but thrice per day.

## I. High Level Design

### 1.1Motivation

The specifications of our device based on the user needs. The device should be able to generate loud sound so that even people with impaired hearing were able to hear it. The device should demonstrate ease of use. In order to help user remember the number of pills they need to take, we also used android application that displays to indicate the number of pills the user need to take. Based on these specifications, we designed a high-level block diagram (figure below) to demonstrate the overall design of our device.

#### 1.2 Existing Products

We found several different pillbox products available in the market. The cheapest one was the traditional pillbox, which contained seven boxes for seven different days of a week. Such pillbox normally cost below $10. However, user had to load the pills to the boxes every week. Mixing different pills in the same box would increase the risk of making mistakes. We also found another type of pillbox, which had the sound reminder, and was able to remind the user to take medicine at user specified time. However, the users still have to put different kinds of pills in the same box, and reload the boxes every week. Additionally, It could only remind the user to take pills once a day. The average costs of this type of pillbox were about $50, which was still expensive than ours. Therefore, we think it was necessary to build a cheap and functional smart pillbox that could bring more convenience for the user.

1.3 High level Block diagram

ESP8266-12E

(NODEMCU)

LED Indicator

WiFi Router

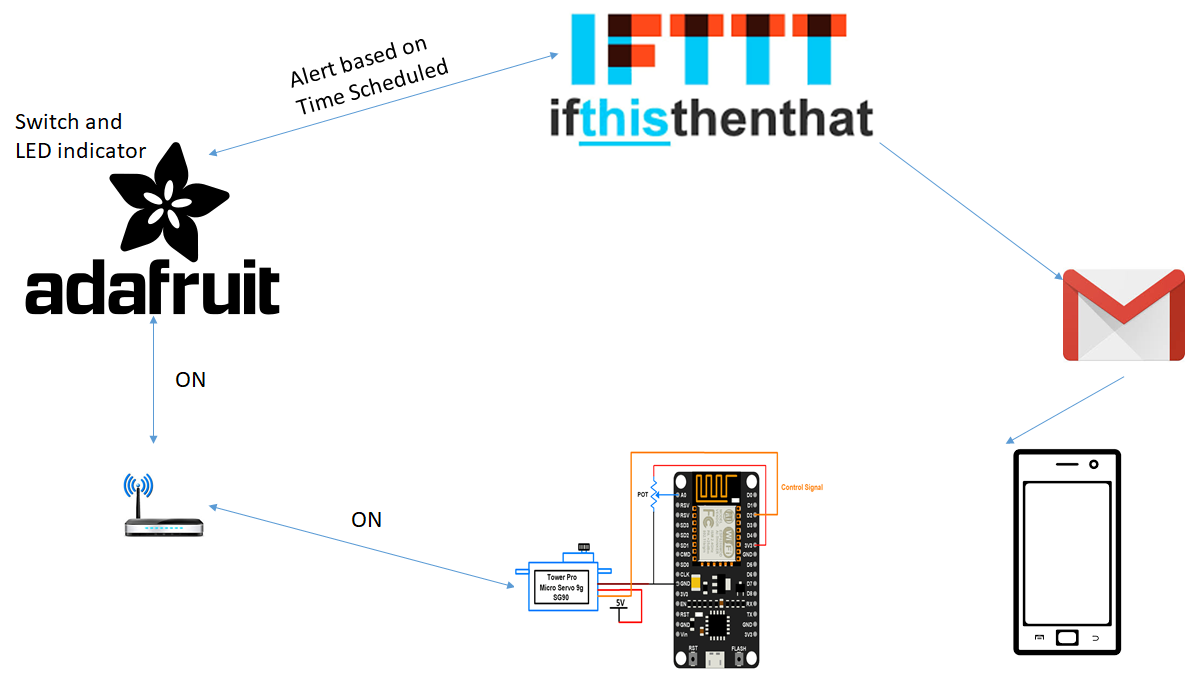
## Figure: High level block diagram

NodeMCU which is a wifi enabled microcontroller will be connected to servo motor, LED and buzzer for visual and audio alerts. NodeMCU is connected to Adafruit platform and will be always subscribed to Switch widget in Adafruit. IFTTT is another IoT platform where the scheduled alarms for medicine will be triggered based on patient schedule.

Patient or care taker configures the time of medicine in IFTTT platform. When the scheduled time reaches an Email,SMS will be sent to patient mobile and an Web pages Alert LED turns red. LED and buzzer attached NodeMCU will start alerting the user.

As the user figures out the alert, patient needs to turn off the alert on web app thereby turning of LED and buzzer attached to device. After alarm is shut off by user the servo motor mechanism attached to NodeMCU rotates and releasing specified dosages.

**Connectivity Diagram**

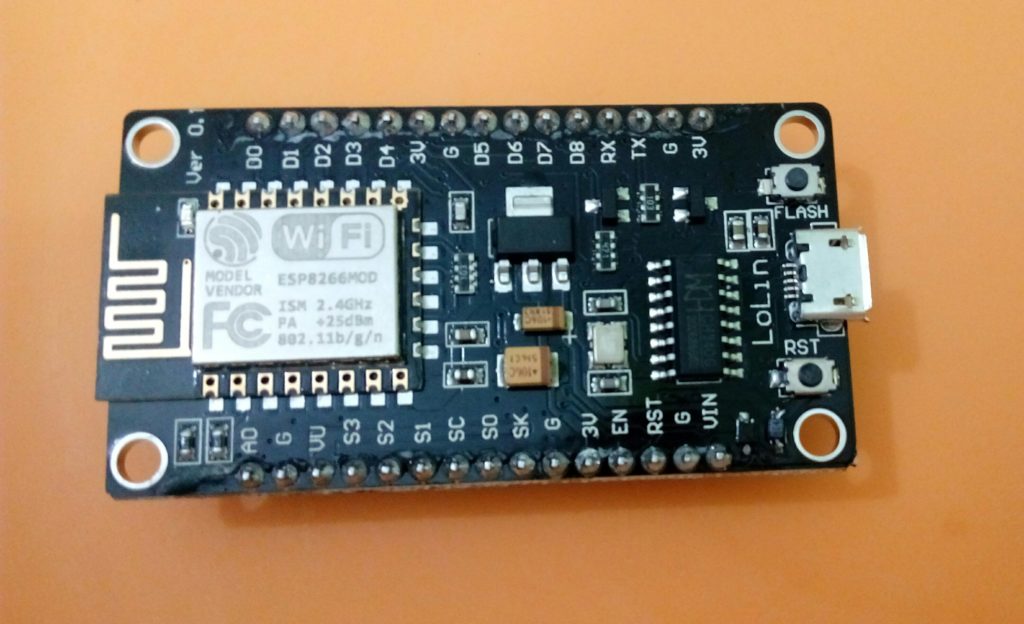
****

## III. Hardware Design

### 3.1 Embedded Board

NodeMcu

NodeMCU is an open source IoT platform. Which includes firmware which runs on the ESP8266 Wi-Fi Module from Espressif Systems,and hardware which is based on the ESP-12 module. The term “NodeMCU” by default refers to the firmware rather than the dev kits. NodeMCU firmware was developed so that AT commands can be replaced with Lua scripting making the life of developers easier. So it would be redundant to use AT commands again in NodeMCU.



### 3.2 Buzzer

### A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke.Buzzer is an integrated structure of electronic transducers, DC power supply,widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers and other electronic products for sound devices. Active buzzer 5V Rated power can be directly connected to a continuous sound, this section dedicated sensor expansion module and the board in combination, can complete a simple circuit design, to "plug and play.



Buzzer module used in our project

**3.3 Servo Motor**

Servo implies an error sensing feedback control which is utilized to correct the performance of a system. It also requires a generally sophisticated controller, often a dedicated module designed particularly for use with servomotors. Servo motors are DC motors that allows for precise control of angular position. They are actually DC motors whose speed is slowly lowered by the gears. The servo motors usually have a revolution cutoff from 90° to 180°. A few servo motors also have revolution cutoff of 360° or more. But servo motors do not rotate constantly. Their rotation is limited in between the fixed angles.

The servo motor is actually an assembly of four things: a normal DC motor, a gear reduction unit, a position-sensing device and a control circuit. The DC motor is connected with a gear mechanism which provides feedback to a position sensor which is mostly a potentiometer. From the gear box, the output of the motor is delivered via servo spline to the servo arm. For standard servo motors, the gear is normally made up of plastic whereas for high power servos, the gear is made up of metal.

A servo motor consists of three wires- a black wire connected to ground, a white/yellow wire connected to control unit and a red wire connected to power supply.

The function of the servo motor is to receive a control signal that represents a desired output position of the servo shaft and apply power to its DC motor until its shaft turns to that position.

It uses the position sensing device to figure out the rotational position of the shaft, so it knows which way the motor must turn to move the shaft to the instructed position. The shaft commonly does not rotate freely around similar to a DC motor, however rather can just turn 200 degrees.



Servo Motor

Servo Motor

From the position of the rotor, a rotating magnetic field is created to efficiently generate toque. Current flows in the winding to create a rotating magnetic field. The shaft transmits the motor output power. The load is driven through the transfer mechanism. A high-function rare earth or other permanent magnet is positioned externally to the shaft. The optical encoder always watches the number of rotations and the position of the shaft.

Working of a Servo Motor

The Servo Motor basically consists of a DC Motor, a Gear system, a position sensor and a control circuit. The DC motors get powered from a battery and run at high speed and low torque. The Gear and shaft assembly connected to the DC motors lower this speed into sufficient speed and higher torque. The position sensor senses the position of the shaft from its definite position and feeds the information to the control circuit. The control circuit accordingly decodes the signals from the position sensor and compares the actual position of the motors with the desired position and accordingly controls the direction of rotation of the DC motor to get the required position. The Servo Motor generally requires DC supply of 4.8V to 6 V.

Controlling a Servo Motor

A servo motor is controlled by controlling its position using Pulse Width Modulation Technique. The width of the pulse applied to the motor is varied and send for a fixed amount of time.

The pulse width determines the angular position of the servo motor. For example a pulse width of 1 ms causes a angular position of 0 degrees, whereas a pulse width of 2 ms causes a angular width of 180 degrees.

Advantages:

If a heavy load is placed on the motor, the driver will increase the current to the motor coil as it attempts to rotate the motor. Basically, there is no out-of-step condition.

High-speed operation is possible.

Disadvantages:

Since the servomotor tries to rotate according to the command pulses, but lags behind, it is not suitable for precision control of rotation.

Higher cost.

When stopped, the motor’s rotor continues to move back and forth one pulse, so that it is not suitable if you need to prevent vibration

7 Applications of Servo Motors

Servomotors are used in applications requiring rapid variations in speed without the motor getting overheated.

In Industries they are used in machine tools, packaging, factory automation, material handling, printing converting, assembly lines, and many other demanding applications robotics, CNC machinery or automated manufacturing.

They are also used in radio controlled airplanes to control the positioning and movement of elevators.

They are used in robots because of their smooth switching on and off and accurate positioning.

They are also used by aerospace industry to maintain hydraulic fluid in their hydraulic systems.

They are used in many radio controlled toys.

They are used in electronic devices such as DVDs or Blue ray Disc players to extend or replay the disc trays.

They are also being used in automobiles to maintain the speed of vehicles.

**3.4 LED**

LEDs create light by electroluminescence in a semiconductor material. Electroluminescence is the phenomenon of a material emitting light when electric current or an electric field is passed through it - this happens when electrons are sent through the material and fill electron holes. An electron hole exists where an atom lacks electrons (negatively charged) and therefore has a positive charge. Semiconductor materials like germanium or silicon can be "doped" to create and control the number of electron holes. Doping is the adding of other elements to the semiconductor material to change its properties. By doping a semiconductor, you can make two separate types of semiconductors in the same crystal. The boundary between the two types is called a p-n junction. The junction only allows current to pass through it one way; this is why they are used as diodes. LEDs are made using p-n junctions. As electrons pass through one crystal to the other they fill electron holes. They emit photons (light). This is also how the semiconductor laser works.

**3.5 IoT Platform**

**Two IoT platfroms will be used here, They are Adafruit and IFTTT**

**Adafruit**

MQTT, or message queue telemetry transport, is a protocol for device communication that Adafruit IO supports. Using a MQTT library or client you can publish and subscribe to a feed to send and receive feed data.

If you aren't familiar with MQTT check out this introduction from the HiveMQ blog. All of the subsequent posts in the MQTT essentials series are great and worth reading too.

To use the MQTT API that Adafruit IO exposes you'll need a MQTT client library. For Python, Node.js, and Arduino you can use Adafruit's IO client libraries as they include support for MQTT (see the client libraries section). For other languages or platforms look for a MQTT library that ideally supports the MQTT 3.1.1 protocol.

Connection Details

You will want to use the following details to connect a MQTT client to Adafruit IO:

Host: io.adafruit.com

Port: 1883 or 8883 (for SSL encrypted connection)

Username: your Adafruit account username (see the accounts.adafruit.com page here to find yours)

Password: your Adafruit IO key (click the AIO Key button on a dashboard to find the key)

We strongly recommend using SSL if your MQTT client allows it.

If the MQTT library requires that you set a client ID then use a unique value like a random GUID. Most MQTT libraries handle setting the client ID to a random value automatically though.

Topics

Adafruit IO's MQTT API exposes feed data using special topics. You can publish a new value for a feed to its topic, or you can subscribe to a feed's topic to be notified when the feed has a new value. Any one of the following topic forms is valid for a feed:

(username)/feeds/(feed name or key)

(username)/f/(feed name or key)

And for groups:

(username)/groups/(group name or key)

(username)/g/(group name or key)

Where (username) is your Adafruit IO username (the same as specified when connecting to the MQTT server) and (feed name or key) is the feed's name or key. The smaller '/f/' path is provided as a convenience for small embedded clients that need to save memory.

Check out our guide to Feed Naming for the full details.

For example if your username is mosfet and you're accessing a feed called Photocell One (which has a Key of photocell-one) you can use any of these paths:

mosfet/feeds/Photocell One

mosfet/f/Photocell One

mosfet/feeds/photocell-one

mosfet/f/photocell-one

In order to reduce confusion, however, we always recommend using the Key to refer to a specific feed or group in code.

To append a new value to a feed perform a MQTT publish against the feed topic and provide the new feed value as the payload of the request.

To be notified of a change in a feed perform a MQTT subscribe to the feed topic. When a new value is added to the feed then the Adafruit IO MQTT server will send a notification with the new feed value in the payload.

You can also subscribe to the parent 'feeds' path to be notified when any owned feed changes using MQTT's # wildcard character. For example the mosfet user could subscribe to either:

mosfet/feeds/#

mosfet/f/#

Once subscribed to the path above any change to a feed owned by mosfet will be sent to the MQTT client. The topic will specify the feed that was updated, and the payload will have the new value.

Be aware the MQTT server sends feed updates on all possible paths for a specific feed. For example, subscribing to mosfet/f/# and publishing to mosfet/f/photocell-one would produce messages from: mosfet/f/photocell-one, mosfet/f/photocell-one/json, and mosfet/f/photocell-one/csv; each referring to the same updated value. To reduce noise, make sure to grab the specific topic of the feed / format you're interested in and change your subscription to that.

PLEASE NOTE: as we adjust which identifiers we use for Feeds internally, the feed updates you receive when using a wildcard will include but may not be limited to the ones shown above.

If you'd like to avoid the formatted feeds ("/json" and "/csv" topics) but still see all the feeds you're publishing to, you can use MQTT's + wildcard in place of #. In this case, subscribing to mosfet/f/+ would produce output on mosfet/f/photocell-one, but not mosfet/f/photocell-one/json.

Publish QoS Levels

One feature of MQTT is the ability to specify a QoS, or quality of service, level when publishing feed data. This allows an application to confirm that its data has been sucessfully published. If you aren't familiar with MQTT QoS levels be sure to read this great blog post explaining their meaning.

For publishing feed values the Adafruit IO MQTT API supports QoS level 0 (at most once) and 1 (at least once) only. QoS level 2 (exactly once) is not currently supported.

Rate Limit

Adafruit IO's MQTT server imposes a rate limit to prevent excessive load on the service. If a user performs too many publish actions in a short period of time then some of the publish requests might be rejected. The current rate limit is at most 1 request per second (or 60 requests within 60 seconds).

If you exceed this limit, a notice will be sent to the (username)/throttle topic. You can subscribe to the topic if you wish to know when the Adafruit IO rate limit has been exceeded for your user account.

This limit applies to all connections so if you have multiple devices or clients publishing data be sure to delay their updates enough that the total rate is below 2 requests/second.

Data Format

There are a few ways to send data to our MQTT API if you're writing your own client library.

The simplest way to send values to an IO Feed topic is to just send the value. For example, a temperature sensor is going to produce numeric values like 22.587. If you're sending to mosfet/feeds/photocell-one you can use the raw number or a string. That means either 22.587 or "22.587" will be accepted as a numeric value. Adafruit IO does its best to treat data as numeric values so that we can show you your data as a chart on an Adafruit IO dashboard and through our Charting API.

Data with Location

To tag your data with a location value, you'll either need to wrap it in a JSON object first or send it to the special /csv formatted MQTT topic.

**IFTTT**

"If This Then That" (IFTTT) is a web service that allows you to connect other web services, apps and devices to each other to automate simple tasks. For instance, like many people you probably have receipts sprinkled through your email. Perhaps you would like to keep these in one place. By connecting the IFTTT Gmail service to the Evernote service, IFTTT can automatically transfer any emails containing the words "receipt"" or "order" to an Evernote folder. These trigger - action combinations are called applets and there are hundreds of pre-existing ones for you to choose from - or you can make your own. At the time of writing IFTTT had implemented over 600 channels (see Figure 1 for a some of the better known services), and more are coming online all the time, so the possibilities are substantial.



Figure 1: Some of the better known services to which IFTTT can connect.

IFTTT is ideal for using with unforgettable.me. Whenever an IFTTT action fires, you can store the fact away in unforgettable.me, so that you have a permanent record that is searchable and visualizable. For instance, you can:

1. Record outlook calendar events when they occur
2. Record incoming Gmails and Office 365 emails
3. Record incoming and outgoing phone calls
4. Record incoming and outgoing SMSs
5. Record your weight and BMI when you step on your Fitbit Aria scales
6. Record sleep and steps from Fitbit watches
7. Record the main news headlines from National Public Radio
8. Store photos you take on Android or iOS devices
9. Record your moods through Jawbone UP

In this tutorial, you will learn to register an IFTTT account, connect unforgettable.me applets and put Do buttons on your phone so that you can register events that cannot be captured automatically. Note that IFTTT is constantly evolving, so some of the screens may look a little different from the ones we show in this tutorial. The same general sequence and functionality should be present, however. Also, note that some Applets may record your GPS location if you turn them on.

### Registering for IFTTT

The first step is to register for an IFTTT account. Navigate to IFTTT.com and click on the sign up button in the top right hand corner. You can use your Google or Facebook sign ups or click through to register with IFTTT independently (see Figure 2).

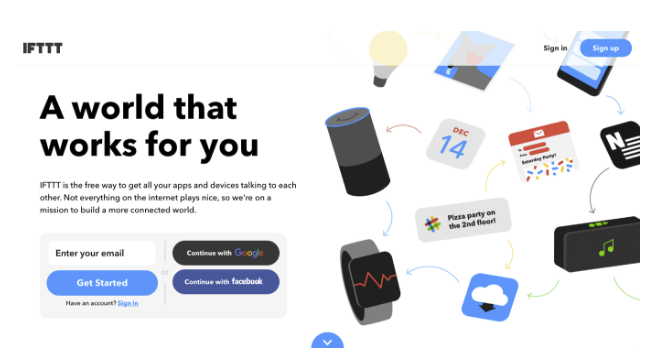


Figure 2: The IFTTT sign up page

### Connecting Unforgettable.Me

Now that you have an IFTTT account you are ready to connect to unforgettable.me. Make sure you are logged in, click on the search button at the top of the page, and type in unforgettable. The service as well as some applets will appear (see Figure 3).

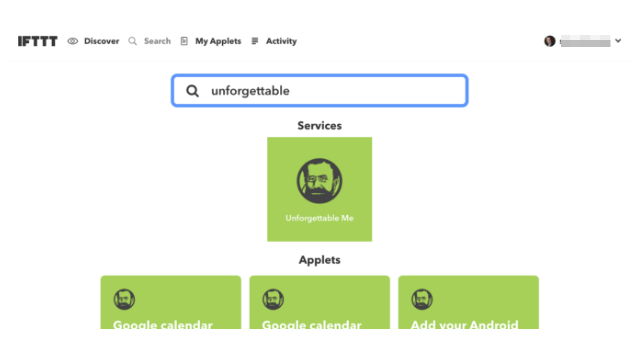


Figure 3: Searching for the unforgettable.me service.

Click on the icon under services and you will be taken to the connection page (see Figure 4).

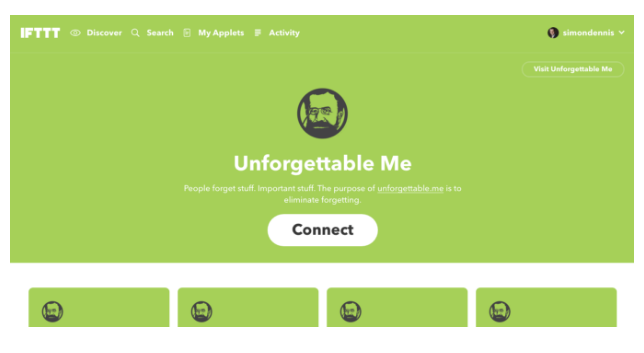


Figure 4: Connect the unforgettable.me service.

Click on Connect and you should see a page like that in Figure 5. Click Authorize and you will be prompted for your unforgettable.me username and password. Type these in and the connection will be created. If you don’t have an unforgetable.me account as yet, stop and go back to the "unforgettable.me" tutorial.

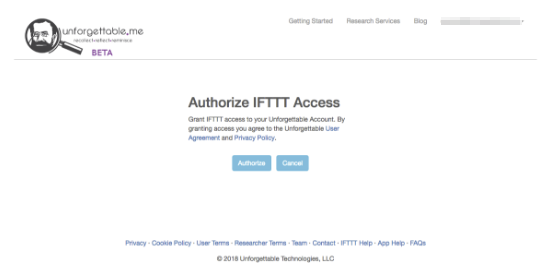


Figure 5: Click authorize to create the connection to unforgettable.me.

### Connecting Existing Applets

Now that you are connected to unforgettable.me you can start to turn on applets. Search for unforgettable again and you should see applets like the ones in Figure 6.

We are going to enable the NPR Stories applet in this tutorial. You may wonder why one would want to track news stories as they are not personal data. However, including them can be very useful when trying to retrieve a specific event. People often organize their memories on the basis of world events. For instance, I can remember that I was in Canberra at a forum when the September 11th attacks occurred. While I would be unlikely to be able to retrieve the date at which I was at the forum under other circumstances, the connection to an event of historical significance makes the task trivial in this case. By retaining the NPR stories, one can use the unforgettable.me system to leverage this kind of knowledge. It's also just fun to reminisce about these events.

To connect the NPR Stories applet, keep scrolling down until you find it and then click.

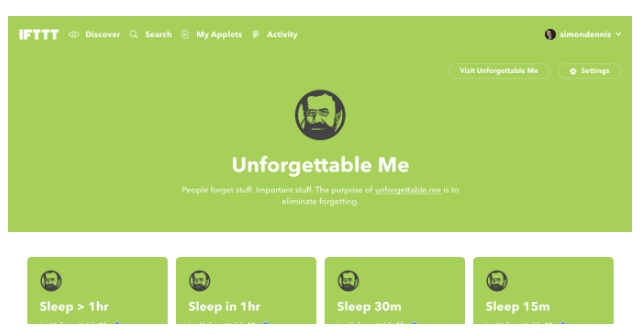


Figure 6: Scroll down to see the pre-existing unforgettabel.me applets.

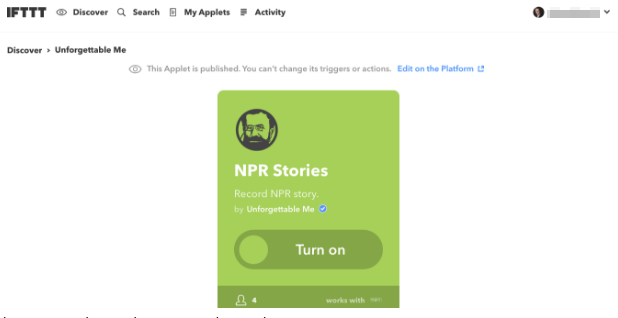


Figure 7: Turning on the NPR Stories applet.

You should now see the switch for the NPR Stories applet. Note you can turn the applet on and off whenever you like. Turn it on now

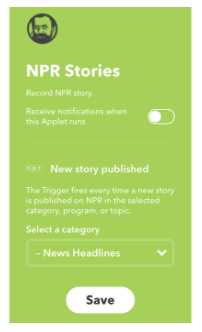


Figure 8: Connect to NPR.

Because you do not need an account with NPR to access their services, no authorization dialog is necessary. You can choose whether you want to receive notifications each time the applet runs. As NPR Stories come through quite often it can become annoying if you turn on notifications for this applet, so we will leave them off. You can also select which stories will be sent to unforgettable.me. There are many categories and topics to choose from. We will leave the selector on News Headlines for now.

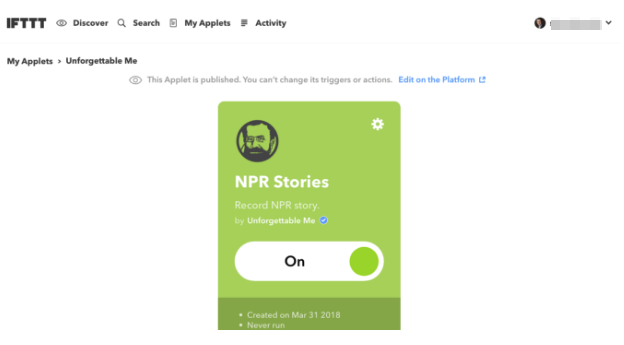


Figure 9: Switch on the NPR applet.

You should now be connected (see Figure 9). Congratulations! You have connected your first applet.

As an exercise find five more applets that you think would be interesting to track and connect them to your account.

We’d like to draw your attention to two applets that can be particularly helpful in some circumstances. Tracking phone calls and texts provides very useful information about the nature of your communication with others. If you are an Android user, there are applets to log incoming and outgoing calls and texts. However, you may be concerned about retaining very private information like who you are calling and the contents of texts in the system. Furthermore, if you are asking others to track their calls and/or texts you may not want to ask them to retain this kind of information. Nonetheless, it can be useful to be able to track patterns communication.

For instance, we are currently running a study in which we are tracking individuals with bipolar spectrum disorder. A potentially important indicator of their current mental state is the degree to which they are communicating with others. We did not want to ask them divulge who these people were, however.

In situations like these, one can use the obfuscated phone call and text applets. These applets retain the time at which calls or texts were made but convert the sender or receiver information into a code. The same code will appear whenever a call or text is made to or received from that individual. However, the code cannot be used to reconstruct the sender or receiver’s details. So, you can retain the number of unique people that you are communicating with and the pattern of calls to those people, without compromising privacy. Furthermore, the obfuscated text applets do not retain the contents of texts.

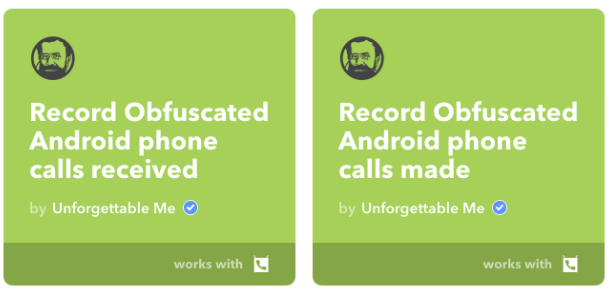


Figure 10: Obfuscated phone call and SMS applets allow you to record calls and SMS without retaining the actual numbers, senders or contents.

### Adding DO Buttons

The services that we have highlighted so far have been focused on the automatic recording of events - so called passive experience sampling. There are many actions (e.g. taking your own mug to the coffee shop to avoid the use of a throw away cup) and subjective states (e.g. happy, tired, frustrated) that require the intervention of the user to track. IFTTT provides an easy way to capture these kinds of events

Figure 11 shows a set of Do buttons on a smartphone. Pressing the icon records the kind of event, the time at which it occurred and the GPS location. There are many applets devoted to tracking these events. Figure 12 shows an event calendar showing the times I have clicked on the "tired" button. If you look closely you will see that there are patterns to the events.

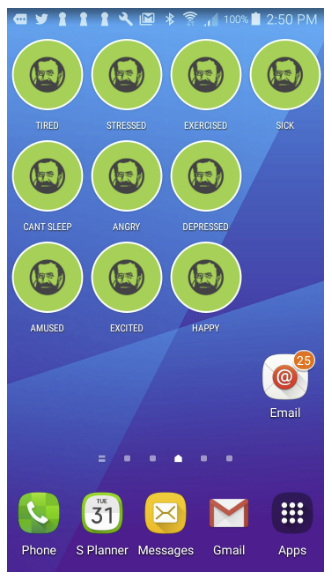


Figure 11: Do buttons on a smartphone.

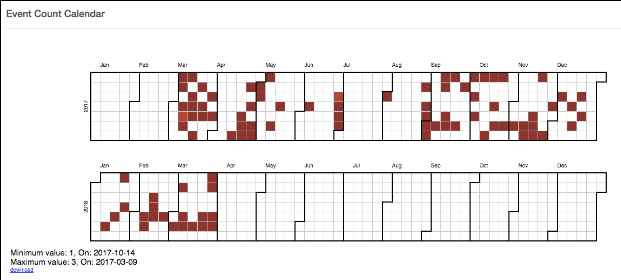


Figure 12: When I was tired.

To begin with you must connect the Do button applets that you wish to add to your phone through the IFTTT site using the procedures we outlined above.

Now, install the IFTTT app on your phone. There are both Android and iOS versions of the app. We will describe the procedure for adding Android Do buttons in this tutorial. There is an equivalent procedure for iOS devices (see below).

### Android procedure for adding DO buttons

Once you have the app installed, open it and go to settings (Figure 13a) and widgets (Figure 13b). You will see a list of the Do applets you have connected. Click on one and make sure it is turned on. If, at any stage you want to turn off applet you can so so either from the website or here in the app (Figure 13c).

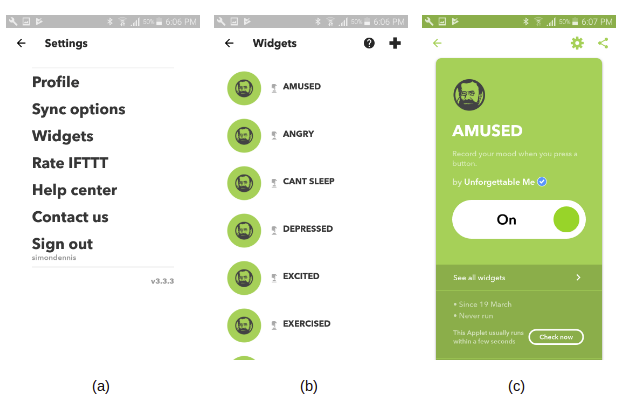


Figure 13: Turning on the button applets.



Figure 14: Creating the button.

We have now turned the applet on, but we still need to add the button to the screen so it can be pressed. Long tap the home screen until it looks like Figure 14a. Now tap Widgets. Select the widgets option at the bottom of the screen and you should see a screen like Figure 14b. Find the IFTTT widgets (you may have to scroll) and select it. You should see a screen like Figure 14c. Choose the circle widget on the right hand side.

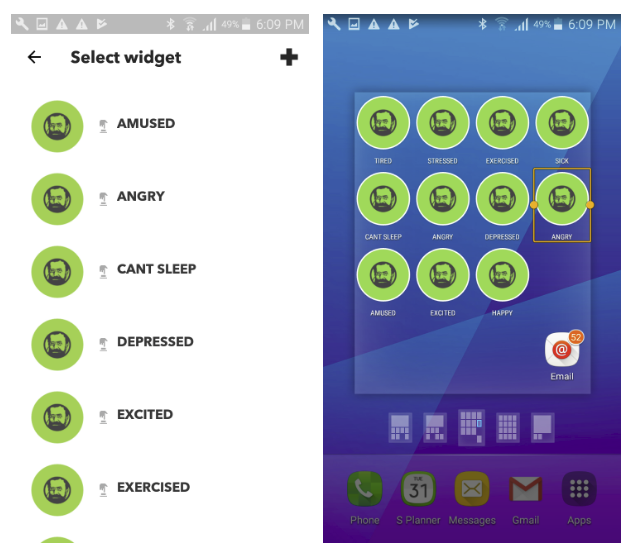


Figure 15: Place the widget on your screen.

Now you will see a screen like Figure 15a. Choose the Do button that you want to add and hold. The screen will appear as in Figure 15b. Now you can place the button where you would like it. And then you are done.

To test the Do button press it and then go into your unforgettable.me account. It might take a few minutes, but if you put \* in the bar and search you should see an event like that in Figure 16.

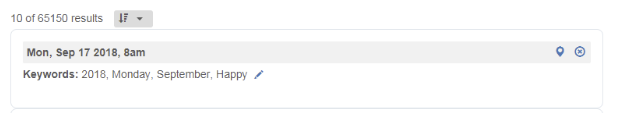


Figure 16: A Happy Do button event.

In the next few hours, a Streetview image will be added go you can see where the event occurred. Then the event will look like that in Figure 17.

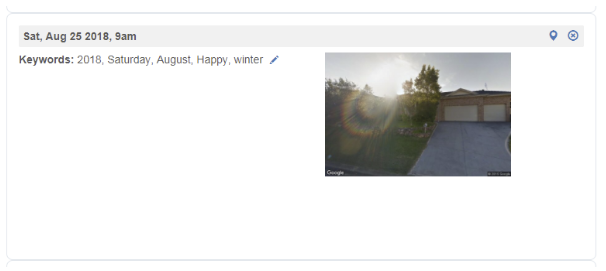


Figure 17: An augmented Happy Do button event.

As an exercise find another 4 buttons to add to your screen and test them out.

### iOS procedure for adding DO buttons

Start off by opening the IFTTT app that you have install from the App Store. On the discover tab you will see a large magnifying glass in the top right of the screen, tap the magnifying glass icon (see Figure 18).

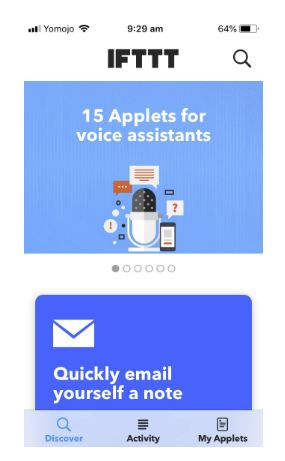


Figure 18: Open the IFTTT App.

Some examples of Unforgettable Do buttons are Happy, Tired, Amused, and so on that record your state at given time. Here we are going to add "Headache" to track when we having headaches.

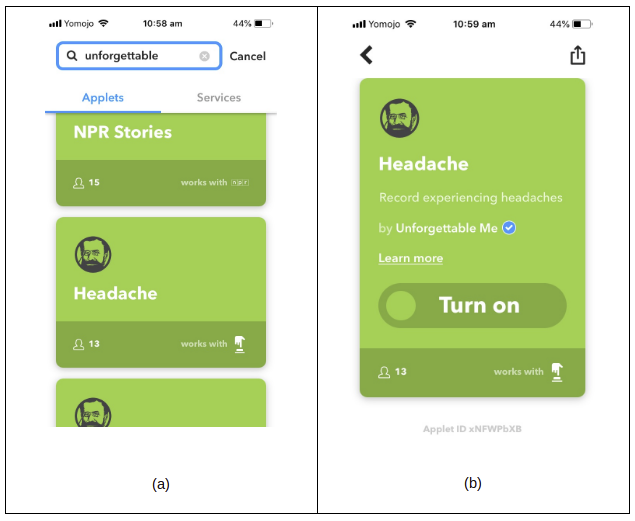


Figure 19: Find and select the Headache button.

As shown in Figure 19a, type "Unforgettable" into the search box and then scroll through "Applets" results until you find the "Headache" button we want to add and then click on it. You now want to click "Turn on" to start using the Do button (see Figure 19b). A panel will be displayed prompting you to configure the widget (button), you can select "Not now" (see Figure 20).

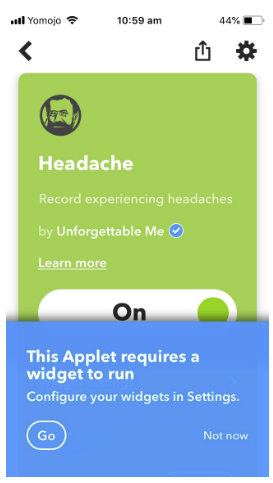


Figure 20: You will be asked to configure the widget.

Next we need to enable the IFTTT in the Notification Center. Close the IFTTT app, and then swipe left until you see the Notification Center shown in Figure 21a.

Touch on "Edit" to add the IFTTT to the Notification Center. You will see a list of Apps that can be added. Click the plus sign next to the IFTTT app and then click the "Done" button in the top right of the display (see Figure 21b).

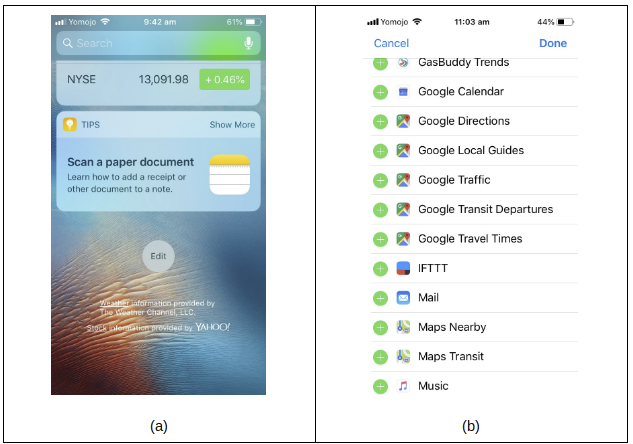


Figure 21: Add the IFTTT panel to the Notification Center.

You should now have an IFTTT panel in your Notifications Center (see Figure 22). Any additional Unforgettable Me buttons that you enable in the future should now just appear here. You can click on the buttons in this panel to record your current state or moods to Unforgettable Me.

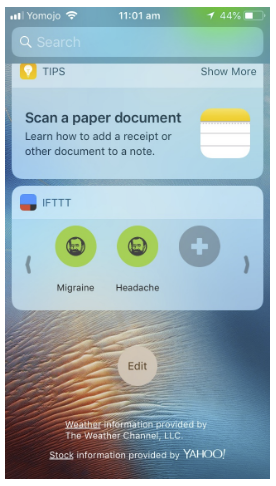


Figure 22: The IFTTT tab in the Notifications Center

## IV. Results

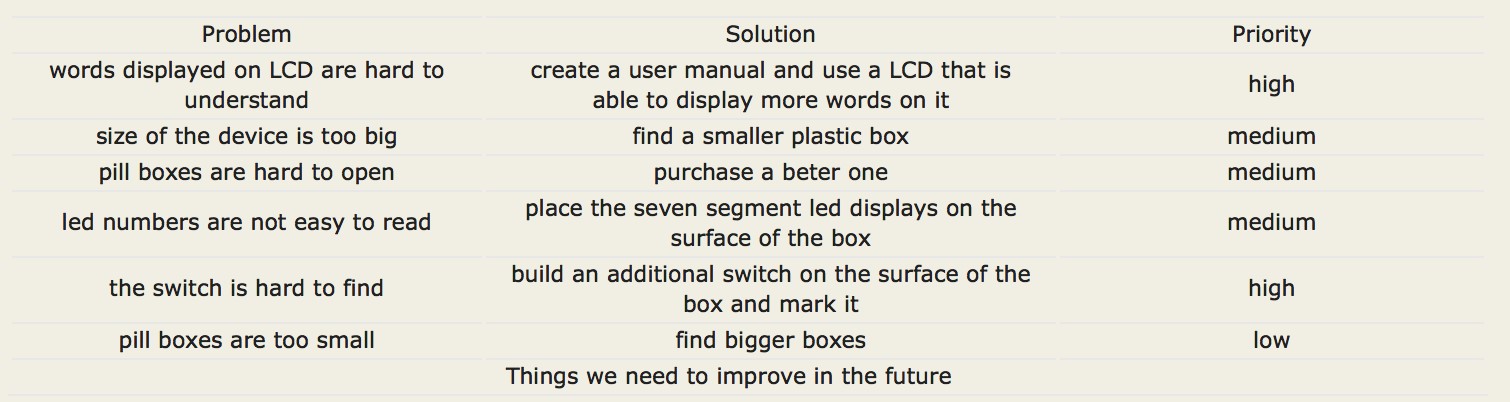
### 4.1 Over Performance

The overall performance of our device was satisfied. The device was packaged in a paper box, which was a little bit large for the device. However, it was able to cover all the messy wires and electrical components, so that the user would not be distracted. Moreover, the paper box also provided certain isolation ability to lower the electrical shock risks. The lcd module and keypad were mounted on the surface of the paper box. When the power is on, the lcd would display characters with the gentle yellow backlight, which allows the user to recognize the characters on the screen even in dark environment. With the implementation of statemachine, the keypad responded promptly and accurately when we pressed the buttons. The long press and backspace features went well during the initialization stage. The seven units of seven segment led displays were placed inside the box since they were embedded on the breadboard. We filled some hard sponge under the breadboard, so that the led displays were close to the surface of the paper box. The light intensity of led displays was satisfied, that the number displayed can be easily recognized even when lab's fluorescent lamps were all on. During the test, we found that the light intensity for some led displays was a little different than the others, but this would not affect the users to recognize the numbers displayed. The speaker was able to produce clear and loud synthesized sound when the comparison stage was triggered. When the "#" key was pressed, the sound stopped, and we were able to see led displays show corresponding numbers without any flicker.

### 4.2 User Evaluation (Usability)

Since our smart pillbox is intended to be used by the user who does not have any electrical engineering background or is not able to operate a complicate system. Therefore, the user evaluation is crucial for our project in terms of future improvements. We planned to do two rounds of user evaluation; first round is focused on collecting feedbacks from people who have strong learning ability and set the pill information by themselves. Second round of evaluation is focused on collecting feedbacks from users who does not have strong learning ability and need others' help to set the pill information.

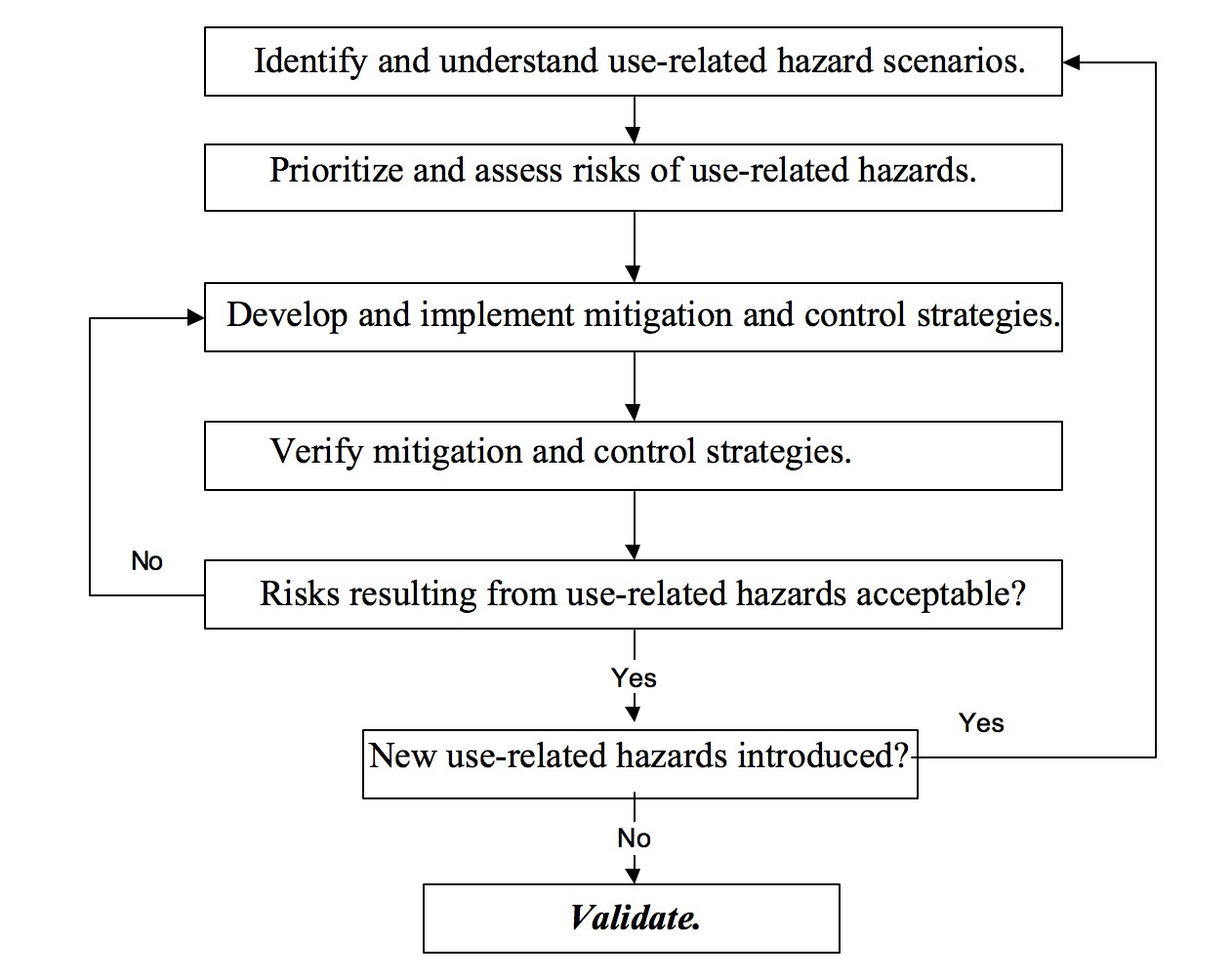
We invited some of our friends as our first round users to try our device and give us some feedbacks. Due to the limitation space for displaying characters on LCD, we were not able to display the instructions on the lcd screen. Therefore, users needed to have some quick learning on how to use the device. After around five minutes of learning, tester were able to set the real time clock, and pill information for each of the medicine box with out any difficulty. In the future, we would improve the user interface so that it can be more self-explanatory for the user to use. It would also be helpful to create a user manual to help the user to use the device. Besides that, our testers also thought the size of the paper box was a little big when compared with the actual pill boxes mounted on the top. Our testers also concerned that the boxes were hard to open and close. They also suggested that we should replace the pillboxes with larger ones. They also thought it would be better to place the seven segments led displays on the surface of the paper box, so that they could read the numbers displayed more easily. They also suggested that the switch of the microcontroller should also be placed outside the box and marked; since the real users might be have any knowledge about the circuitry. We created a table (see below) in order to list all the suggestions and comments from our testers.



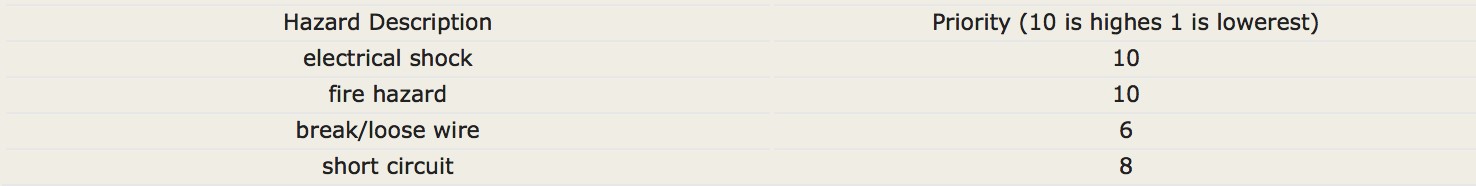
The priority level is rated based on how severe the problems would affect user to use the pillbox. Since our second round testers are mainly the older, we decided to conduct the evaluation during the winter break.

### 4.3 Safety

Our device is used to contain drugs, and it is defined as a class I device based on the definition of the medical device by FDA(Link). Therefore, safety is one of the most important factor need to be considered. We should identify all the possible risks and hazards before we build the device. Validation of all the safety factors is also essential after we finish build the device. We used the analysis structure (see figure below) described in Medical Device Use-Safety: Incorporating Human Factors Engineering into Risk Managementoublished (Link) by FDA to analyze the risks and hazards in our device.



In the identification of use-related hazard scenario stage, we identified the potential hazards based on device use description. We found several potential hazards that would occur during the use of the device (see table below).



Among all the hazards, we thought electrical shock and fire hazards should have the highest priority, since these two hazards would not only cause malfunction of the device but also have high risks of causing danger to the user. In order to mitigate the risks, we decided to cover the entire circuits in an insulated box. We also planned to use high burning point material such as metal or plastic for our box in order to lower the fire risk. However, due to the time limitation, we had to choose a paper box for our current device, and we didn't cover the entire circuitry. Therefore, we were still in the strategies implementation stage. However, for our future work, we would still stick to the analysis structure to carefully manage the risks for our device.

## V. Conclusions

### 5.1 Future Work

There are several aspects we need to work on our device in the future to meet the user needs. Firstly, we should develop strategies and modify the device based on the user evaluation results. This includes creating a user manual; choosing a larger lcd display; using a metal or plastic box to cover the entire circuitry; placing the switch and led displays on the surface of the box and using larger pill boxes.

We should also follow the risk analysis structure to analyze the potential risks and hazards as well as develop strategies to mitigate the risks.

**References:**

1. A. Sawand, S. Djahel, Z. Zhang, and F. Na. Multidisciplinary Approaches to Achieving Efficient and Trustworthy eHealth Monitoring Systems. Commun. China (ICCC), 2014 IEEE/CIC Int. Conf., pp. 187–192; 2014.

2. D. a. Clifton, D. Wong, L. Clifton, S. Wilson, R. Way, R. Pullinger, and L. Tarassenko. A large-scale clinical validation of an integrated monitoring system in the Emergency Department. IEEE J. Biomed. Heal. Informatics vol. 17, no. 4, pp. 835–842; 2013.

3. M. Parida, H.-C. Yang, S.-W. Jheng, and C.-J. Kuo.Application of RFID Technology for In-House Drug Management System.15th Int. Conf.Network-Based Inf. Syst., pp. 577–581; 2012.

4. L. Ilkko and J. Karppinen.UbiPILL A Medicine Dose Controller of Ubiquitous Home Environment. 2009 Third Int. Conf. Mob. Ubiquitous Comput. Syst. Serv. Technol., pp. 329–333; 2009.

5. A. Kliem, M. Hovestadt, and O. Kao.Security and Communication Architecture for Networked Medical Devices in Mobility-Aware eHealth environments,” 2012 IEEE First Int. Conf. Mob. Serv., pp. 112–114;2012.

6. S. T.-B. Hamida, E. Ben Hamida, B. Ahmed, and A. Abu-Dayya.Towards efficient and secure in-home wearable insomnia monitoring and diagnosis system. 13th IEEE Int. Conf. Bioinforma. Bioeng., pp. 1–6; 2013.

7. P. Ray.Home Health Hub Internet of Things (H 3 IoT): An architectural framework for monitoring health of elderly people.Sci. Eng. Manag. Res. pp. 3–5, 2014.

8. S. Huang, H. Chang, Y. Jhu, and G. Chen.The Intelligent Pill Box - Design and Implementation. pp. 235–236; 2014.

9. F.-T. Lin, Y.-C. Kuo, J.-C. Hsieh, H.-Y. Tsai, Y.-T. Liao, and H. C. Lee A Self-powering Wireless Environment Monitoring System Using Soil Energy. IEEE Sens. J., vol. 15, no. c, pp. 1–1; 2015.

10. S. S. Al-majeed.Home Telehealth by Internet of Things (IoT). pp. 609–613; 2015.

11. C. List, O. F. Authors, D. Moga, N. Stroia, D. Petreus, R. Moga, and R. A. Munteanu. Work Embedded Platform for Web-based Monitoring and Control of a Smart Home no. 53, pp. 1–3; 2015.

12. R. J. Rosati.Evaluation of Remote Monitoring in Home Health Care. in 2009 International Conference on eHealth, Telemedicine, and Social Medicine;2009, pp. 25–27.

13. J. E. Luzuriaga, J. C. Cano, C. Calafate, M. Perez, P. Boronat, and U. J. I, Handling Mobility in loT applications using the MQTT protocol.Internet Technol. Appl., pp. 245–250; 2015.

14. D. H. S. D. Privacy and I. A. Committee. Report No . 2006-02 The Use of RFID for Human Identity Verification I . Introduction and Executive Summary II . RFID Technology Overview.Technology, pp. 1–13;2006.

15. H. W. Wang, R. G. Lee, C. C. Hsiao, and G. Y. U. Hsieh. Active RFID system with cryptography and authentication mechanisms. J. Inf. Sci. Eng., vol. 26, pp. 1323–1344;2010