# SMART MEDICINE REMINDER FOR ELDERLY PEOPLE

# ABSTRACT

Sometimes elderly people forget to take the medicine at the required time of medicines. And sometimes they also forget which medicine they have to take at required time. And it is difficult for Doctor/Compounder/ care takers to monitor patients around the clock. To avoid this problem, we have made this medicine reminder system. An app is build for the user (care taker) which enables him to set the time at which the person has to consume the tablet. The time that is set is received by our Device Node MCU and stores the data in EEPROM through IOP Address RTC clock module is used for comparison of time set by the user. If both gets matched then LED placed at particular slot will glow and a sound is generated by Buzzer.

**PROBLEM STATEMENT:**

When it comes to our elder ones, we always want to stay them healthy and fit. But what will happen if they get ill and forget to take medicine on time. We would be worried, right? At hospitals, there are many patients and it is difficult to remind every patient to take medicine on time. The traditional ways require human efforts to remind to take medicines on time. The digital era doesn’t follow that and we can use machines to do that. The application of Smart Medicine Reminder is very wide and can be used by patients at home, doctors at hospitals and at many other places. When it comes to reminding, there can be many ways to remind it like Show it on a display, Send notification on email or Phone, Using mobile apps, Buzz alarm, Using Bluetooth/ Wi-Fi, Get a call, Remind for next medicine time while reminding current time.

We can combine ways depending upon the need. To keep things simple here we made a simple Medicine Reminder Using Node MCU developed mobile application such we can access and set reminder from anywhere which reminds to take medicines 1 or 2 or 3 times a day. The time slot can be selected using push buttons. Also, it shows current Date and Time.

The remarkable problem is that patients forget to take the proper medicines in proper proportion and in proper time. Medication adherence, which refers to the degree or extent to which a patient takes the right medication at the right time according to a doctor’s prescription, has recently emerged as a serious issue because many studies have reported that non-adherence may critically affect the patient, So we are introducing an Android application whose objective is to remind the patients of their dosage timings through Alarm Ringing system so that they can stay fit and healthy. It is life-saving, money saving and time saving application which is easy to use and provides a good user interface.

**PROJECT WORKING PROCESS:**

The main idea of project is to first remind the patients to take medicine on time. For this using Node MCU we developed mobile application such we can access and set reminder from anywhere which reminds to take medicines 1 or 2 or 3 times a day. The time slot can be selected using push buttons. Also, it shows current Date and Time. The mobile app can be accessed from anywhere but the reminder device should be with the patient. This mobile application is simple such that it provides a better look and feel to the user. The working process of this project is simple because here we can simply open the mobile application and click on the ON button, once the button is on it will automatically sets reminder. Here the reminder in set by the node red editor stores in the cloud. The time that is set is received by our Device Node MCU and stores the data in EEPROM through IOP Address RTC clock module is used for comparison of time set by the user. If both gets matched then LED placed at particular slot will glow and a sound is generated by Buzzer, shows a display to the patient to take medicine, once the patient take medicine pushes the button, the alarm offs and cloud stores the data.

**COMPONENTS REQUIRED**

* Node MCU
* OLED
* Buzzer
* Basic Shield
* Connecting wires

**Node MCU:**

The Node MCU devkit is a development board with the ESP8266 mounted on it. It also has a USB to Serial convertor chip on board. This removes the need of the FTDI USB to Serial Converter. Also, it has a voltage converter on board for converting the 5V supplied by the USB port to 3.3V input required by the ESP8266. So all you have to do is plug the USB cable from the computer right into the micro USB slot of the Node MCU dev. board, and you can start with your ESP8266 programming / prototyping. The Node MCU provides easier way to program the ESP8266 module.

**ESP8266:**

The ESP8266 is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller.ESP8266 offers a complete and self-contained Wi-Fi networking solution, allowing it to either host the application or to offload all Wi-Fi networking functions from another application processor. When ESP8266 hosts the application, and when it is the only application processor in the device, it is able to boot up directly from an external flash. It has integrated cache to improve the performance of the system in such applications, and to minimize the memory requirements.

**OLED:**

OLED (Organic Light Emitting Diodes) is a flat light emitting technology, made by placing a series of organic thin films between two conductors. When electrical current is applied, a bright light is emitted. OLEDs are emissive displays that do not require a backlight and so are thinner and more efficient than LCD displays.   
which do require a white backlight.OLED displays are not just thin and efficient - they provide the best image quality ever.

**Basic Shield:**

This Basic Shield can be interface with 5V or 3.3V logic microcontroller boards like Arduino, AVR, PIC, 8051, ARM etc. Basic Shield is very popular shield for interfacing of electronics component with microcontroller like LED’s, variable resistor, Push Button, LDR, buzzer etc.All components are arranged in a proper manner so that you can use it with your microcontroller to learn basic programming of microcontroller in your projects. This shield helps you how to deal with basic electronics component in your project.

It contains 8 LEDs (5 mm), 2 Push Button1 Potentiometer (100k),LDR (light detection resistor),Buzzer.

This shield is comfortable with 5v or 3.3VDC power supply, and easy to fix on board (chassis). All components have separate pin so that we can use them in our projects. Basic shield has been designed with Common anode network, means to glow led you have to connect LOW logic signal at led pins.

**THE FINAL CODE:**

#include <ESP8266WiFi.h>

#include <PubSubClient.h>

#include <Wire.h>

#include <Adafruit\_SSD1306.h>

#include <Adafruit\_GFX.h>

const char\* ssid = "Aradhya";

const char\* password = "87654321";

#define ORG "ybg17p"

#define DEVICE\_TYPE "nodemcu"

#define DEVICE\_ID "swathi"

#define TOKEN "7075606886"

#define SSD1306\_LCDHEIGHT 64

// OLED display TWI address

#define OLED\_ADDR 0x3C

Adafruit\_SSD1306 display(-1);

#if (SSD1306\_LCDHEIGHT != 64)

#error("Height incorrect, please fix Adafruit\_SSD1306.h!");

#endif

String command;

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char topic[] = "iot-2/cmd/home/fmt/String";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;

//Serial.println(clientID);

WiFiClient wifiClient;

void callback(char\* topic, byte\* payload, unsigned int payloadLength);

PubSubClient client(server, 1883, callback, wifiClient);

void setup() {

Serial.begin(115200);

Serial.println();

pinMode(D3,OUTPUT);

wifiConnect();

mqttConnect();

// initialize and clear display

display.begin(SSD1306\_SWITCHCAPVCC, OLED\_ADDR);

display.clearDisplay();

// display a line of text

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,10);

display.print("medicine reminder");

// update display with all of the above graphics

display.display();

}

void loop() {

if (!client.loop()) {

mqttConnect();

}

delay(100);

}

void wifiConnect() {

Serial.print("Connecting to "); Serial.print(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.print("nWiFi connected, IP address: "); Serial.println(WiFi.localIP());

}

void mqttConnect() {

if (!client.connected()) {

Serial.print("Reconnecting MQTT client to "); Serial.println(server);

while (!client.connect(clientId, authMethod, token)) {

Serial.print(".");

delay(500);

}

initManagedDevice();

Serial.println();

}

}

void initManagedDevice() {

if (client.subscribe(topic)) {

Serial.println("subscribe to cmd OK");

} else {

Serial.println("subscribe to cmd FAILED");

}

}

void callback(char\* topic, byte\* payload, unsigned int payloadLength) {

Serial.print("callback invoked for topic: "); Serial.println(topic);

for (int i = 0; i < payloadLength; i++) {

//Serial.println((char)payload[i]);

command += (char)payload[i];

}

Serial.println(command);

display.clearDisplay();

// display a line of text

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0,10);

display.print(command);

// update display with all of the above graphics

display.display();

digitalWrite(D3,HIGH);

delay(5000);

digitalWrite(D3,LOW);

}