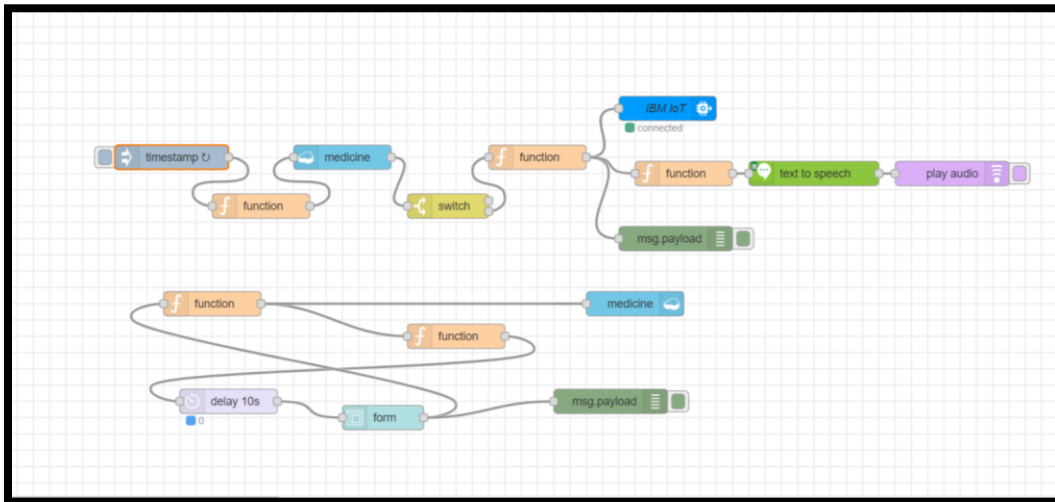


Date	08 November 2022
Team ID	PNT2022TMID54420
Project Name	PERSONAL ASSISTANCE FOR SENIOR WHO ARE SELF-RELIANT

SPRINT-2

CREATE NODE-RED FORM



ADDING IOT CREDENTIALS

The image shows the Node-RED interface with the 'Properties' panel open for a function node. The 'On Message' tab is selected, and the following JavaScript code is entered in the editor:

```

1 var d=new Date()
2 var utc=d.getTime()+d.getTimezoneOffset()*60000;
3 var offset = 5.5;
4 newDate = new Date(utc+(3600000*offset));
5 var n =newDate.toISOString()
6 var date = n.slice(0,10)
7 var time=n.slice(11,16)
8 global.set("time",time)
9 msg.payload-date+" "+time
10 return msg;

```

The 'Enabled' checkbox is checked at the bottom of the panel.

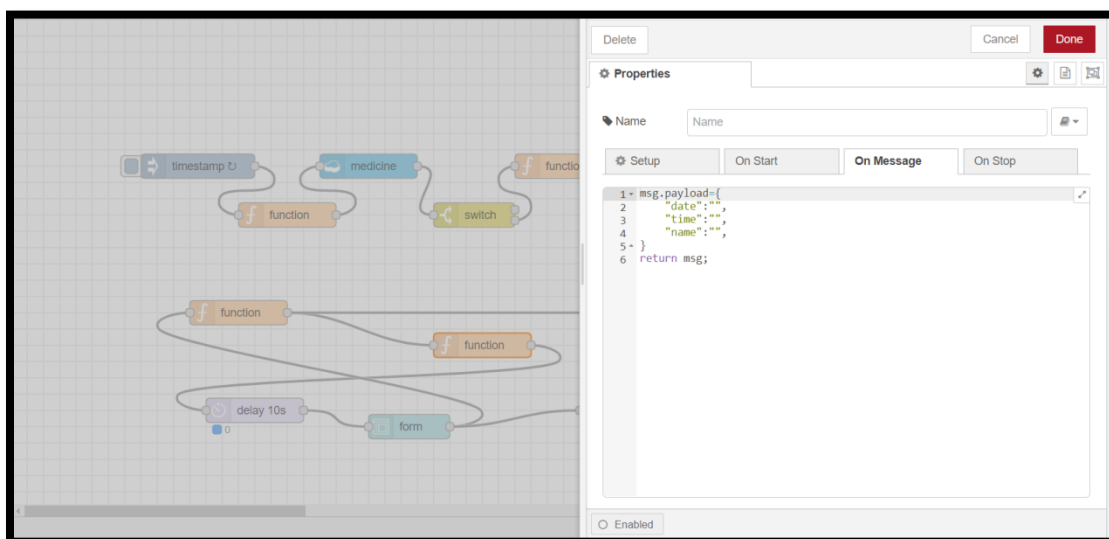
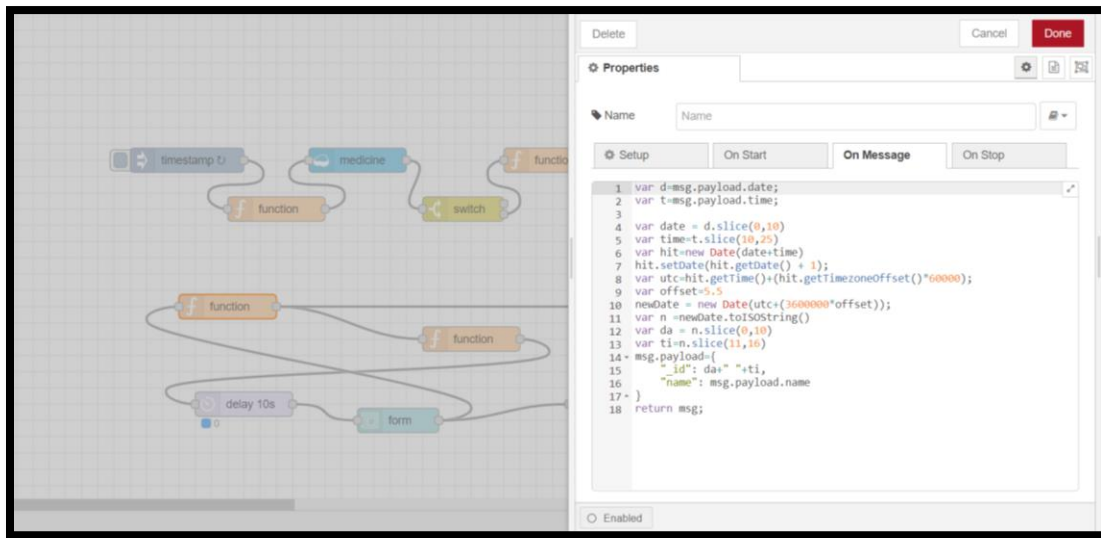
The screenshot shows a workflow editor with a sequence of nodes: **timestamp**, **medicine**, **function**, **switch**, **function**, **delay 10s**, **form**, and **msg.payload**. The **Properties** panel on the right is open for the **msg.payload** property. It shows a rule configuration with the following settings:

- Name:** Name
- Property:** msg.payload
- Rule:** is null → 1
- Rule:** otherwise → 2
- checking all rules:** checked
- recreate message sequences:** unchecked
- Enabled:** checked

The screenshot shows the same workflow editor, but the **Properties** panel on the right is now open for the **msg.payload** property, showing a configuration for an API Key, Device Type, Device Id, Command Type, Format, Data, QoS, Name, and Service. The settings are as follows:

- Authentication:** API Key
- API Key:** device api
- Output Type:** Device Command
- Device Type:** ESP
- Device Id:** ESP32
- Command Type:** command
- Format:** String
- Data:** 1
- QoS:** 0
- Name:** IBM IoT
- Service:** registered
- Enabled:** checked

FUNCTION TO GET MEDICINE DETAILS AND TIME



CODE FOR SIMULATION:

```

#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT
#include <LiquidCrystal_I2C.h>
#include "DHT.h" // Library for dht11
#define DHTPIN 15 // what pin we're connected to
#define DHTTYPE DHT11 // define type of sensor DHT 11
#define LED 2
DHT dht (DHTPIN, DHTTYPE); // creating the instance by passing pin and typr of
dht connected
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);

```

```
//-----credentials of IBM Accounts-----
```

```
#define ORG "9a7os9"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "ESP32"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "LC!x?+V9etumdVMaSR" //Token
String data3="";
int buzz= 13;
```

```
//----- Customise the above values -----
```

```
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event
perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT
command type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth";// authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
LiquidCrystal_I2C lcd(0x27,16,2);
```

```
//-----
```

```
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the
predefined client id by passing parameter like server id,portand wificredential
void setup()// configureing the ESP32
```

```
{
  Serial.begin(115200);
  pinMode(LED,OUTPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}
```

```
void loop()// Recursive Function
```

```
{
  if (!client.loop()) {
    mqttconnect();
  }
}
```

```
/*.....retrieving to
```

```
Cloud. .... */
```

```
void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }

    initManagedDevice();
    Serial.println();
  }
}

void wificonnect() //function defination for wificonnect
{
  Serial.println(); Serial.print("Connecting
to");

  WiFi.begin("Wokwi-GUEST", "", 6); //passing the wifi credentials to establish
the connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

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

}

void initManagedDevice() {
  if (client.subscribe(subscribetopic)) {
    Serial.println((subscribetopic));
    Serial.println("subscribe to cmd OK");
  } else {
    Serial.println("subscribe to cmd FAILED");
  }
}

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
```

```

{

    Serial.print("callback invoked for topic: ");
    Serial.println(subscribetopic);
    for (int i = 0; i < payloadLength; i++) {
        //Serial.print((char)payload[i]);
        data3 += (char)payload[i];
    }

    Serial.println("Please take "+ data3);
    if(data3 != "")
    {
        lcd.init();
        lcd.print("Take"+ data3);

digitalWrite(LED,HIGH);
delay(20000);
digitalWrite(LED,LOW);

    }

    else
    {
digitalWrite(LED,LOW);

    }
    data3="";
}

```

esp32-dht22.ino
diagram.json
libraries.txt
Library Manager

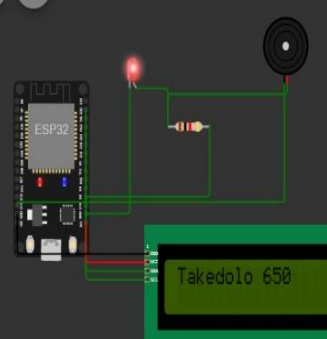
```

1  #include <WiFi.h> //library for wifi
2  #include <PubSubClient.h> //library for MQTT
3  #include <LiquidCrystal_I2C.h>
4  #include "DHT.h" // Library for dht11
5  #define DHTPIN 15 // what pin we're connected to
6  #define DHTTYPE DHT11 // define type of sensor DHT 11
7  #define LED 2
8  DHT dht (DHTPIN, DHTTYPE); // creating the instance by passing pin and type of
9  void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
10
11
12  //-----credentials of IBM Accounts-----
13
14  #define ORG "9a7os9" //IBM ORGANIZATION ID
15  #define DEVICE_TYPE "ESP" //Device type mentioned in IBM Watson IoT Platform
16  #define DEVICE_ID "ESP32" //Device ID mentioned in IBM Watson IoT Platform
17  #define TOKEN "LC!x?4V9etumdVMaSR" //Token
18  String data3="";
19  int buzz= 13;
20
21  //----- Customise the above values -----
22  char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
23  char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event
24  char subscribetopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT command
25  char authMethod[] = "use-token-auth"; // authentication method
26  char token[] = TOKEN;
27  char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
28  LiquidCrystal_I2C lcd(0x27,16,2);
29

```

Simulation

00:35.001 99%



iot-2/cmd/command/fmt/String
subscribe to cmd OK

callback invoked for topic: iot-2/cmd/command/fmt/String
Please take dolo 650
callback invoked for topic: iot-2/cmd/command/fmt/String
Please take dolo 650