Team ID: PNT2022TMID27560

Project Title: Safety Gadget for child safety monitoring and notification

Delivery Plan Sprint - 2

Sprint 2 is about LOGIN and NOTIFICATION of the IoT device in Parent's Web Application for getting information about Child's Status.

LOGIN:

This Coding is to build the login page of the parent's application to get information about the child's condition.

Coding:

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-
scale=1">
<title> Login Page </title>
<style>
Body {
font-family: Calibri, Helvetica, sans-serif; background-
color:#9FE2BF;
}
button {
background-color: #9FE2BF;
width: 100%;
color: black;
padding: 15px;
```

```
margin: 10px 0px;
border: none;
cursor: pointer;}
form {
border: 3px solid #f1f1f1;
}
input[type=text], input[type=password]
{
width: 100%;
margin: 8px 0;
padding: 12px 20px;
display: inline-block;
border: 2px white;
box-sizing:
border-box;
}
button:hover {
opacity: 0.7;
}
.cancelbtn {
width: auto;
padding: 10px 18px;
margin: 10px 5px;
}
.container {
padding: 25px;
background-color: #CCCCFF;
}
</style> </head>
<body>
<center> <h1> Login Form </h1> </center>
<form>
<div class="container">
```

<label>Device ID/Number: </label>
<input type="password" placeholder="Enter Password"
name="password" required>
<label>E-Mail : </label>
<input type="text" placeholder="Enter Username"
name="username" required>
<label>Password : </label><input type="password" placeholder="Enter Password"
name="password" required>
<button type="submit">Login</button>
<button class="loginBtn loginBtn--facebook">Login with
Facebook.</button>
<button class="loginBtn loginBtn--google">Login with Google.</button>
<input type="checkbox" checked="checked"> Remember me
<button type="button" class="cancelbtn"> Cancel</button> Forgot <a
href="#"> password?

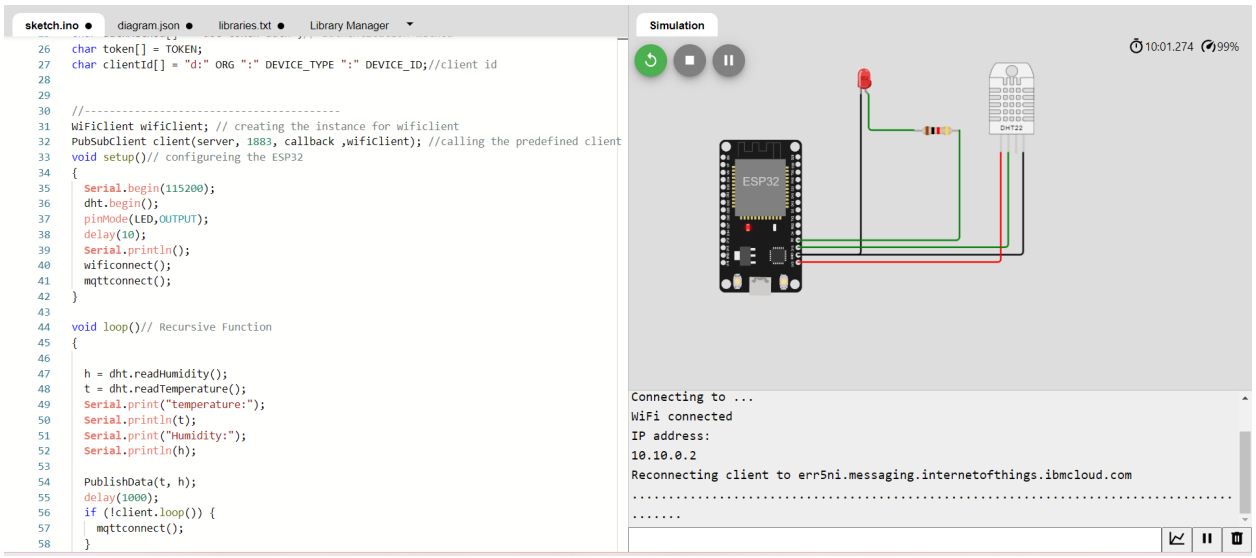
</form>
</body>
</html>
NOTIFICATION:
This coding will make connection between IoT Device & Parent's application.
When the child cross across the geofence message will be notified on parent's
application .
Coding:
#include<WiFi.h>//library for wifi
#include<PubSubClient.h>//library for MQTT
void callback(char* subscribetopic, byte* payload,unsigned int payloadlength);
//---------------credentials of IBM Account------
#define ORG "45z3o2"// IBM ORGANIZATION ID
#define DEVICE_TYPE "ESP32_Controller"//DEVICE TYPE MENTIONED IN IOT
WATSON
PLATFORM #define DEVICE_ID "bme2"//DEVICE ID MENTIONED IN IOT WATSON
PLATFORM

```
#define TOKEN "OKZ+q@JfPWDOd6wBTj"//Token String data3;
float dist;
//------------customize the above value-------char server[]=ORG
".messaging.internetofthings.ibmcloud.com";//server
name
char publishtopic[]="ultrasonic/evt/Data/fmt/json";/*topic name and
type of event perform and format
in which data to be send*/
char subscribetopic[]="ultrasonic/cmd/test/fmt/String";/*cmd REPRESENT
Command tupe 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
//
WiFiClient wifiClient;// creating an instance for wificlient
PubSubClient client(server, 1883 , callback , wifiClient);/*calling the predefined
client id by passing parameter like server id,portand wificredential*/
int LED =4;
int trig =5; int echo=18; void setup(){
Serial.begin(115200); pinMode(trig,OUTPUT); pinMode(echo,INPUT);
pinMode(LED,OUTPUT); delay(10); Serial.println(); wificonnect(); mqttconnect();
void loop() { digitalWrite(trig,LOW); digitalWrite(trig,HIGH);
delayMicroseconds(10); digitalWrite(trig,LOW);
float dur=pulseIn(echo,HIGH); float dist=(dur * 0.0343)/2; Serial.print("distance in
cm"); Serial.println(dist); PublishData(dist);
delay(1000);
if (!client.loop()){ mqttconnect();
}
}
/*.....retrieving to cloud.....
*/
void PublishData(float dist){ mqttconnect();//function call for connecting to ibm
```

```
/*creating the string in form of JSON to update the data to ibm cloud*/ String
object;if(dist<100)
{
digitalWrite(LED,HIGH); Serial.println("no object is near");
object="Near";
}
else
{
digitalWrite(LED,LOW); Serial.println("no object found"); object="No";
}
String payload="{\"distance\":"; payload +=dist;
payload +="," "\"object\":\""; payload += object;
payload += "\"}";
Serial.print("Sending payload: ");
Serial.println(payload);
if(client.publish(publishtopic, (char*) payload.c_str())){
Serial.println("Publish ok");/* if its successfully upload data on the
cloud then it will print publish ok in serial monitor or else it will print publish
failed*/
} else{
Serial.println("Publish failed");
}
}
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 definition for wificonnect
```

```
{
Serial.println(); Serial.print("Connecting to ");WiFi.begin("vivo 1816",
"taetae95",6);//PASSING THE WIFI CREDENTIALS TO
ESTABLISH 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("dta: "+ data3);
//if(data3=="Near")
//{
//Serial.println(data3);
//digitalWrite(LED,HIGH);
//}
//else //{
//Serial.println(data3);
//digitalWrite(LED,LOW);//} data3="";
}
```

Output:




```
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id


//-------------------------------------
WiFiClient wifiClient; // creating the instance for wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client
void setup()// configureing the ESP32
{
  Serial.begin(115200);
  dht.begin();
  pinMode(LED,OUTPUT);
  delay(10);
  Serial.println();
  wificonnect();
  mqttconnect();
}

void loop()// Recursive Function
{

  h = dht.readHumidity();
  t = dht.readTemperature();
  Serial.print("temperature:");
  Serial.println(t);
  Serial.print("Humidity:");
  Serial.println(h);

  PublishData(t, h);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
```