```
#include <ESP8266WiFi.h>
#include <ESP8266mDNS.h>
#include <WiFiUdp.h>
#include <ArduinoOTA.h>
#include <PubSubClient.h>
#include <BME280I2C.h>
#include <Wire.h>

BME280I2C bme;
float temperatur;

const char* ssid = "xxxx";
const char* password = "xxxx";
const char* topic="room/temperature1";
const char* topic2="room/hum1";
const char* topic3="room/pres1";
const char* mqtt_server = "192.168.1.201";
const char* clientname="ESP8266Client1";
const char* otahostname="myesp8266-12a";


 float temp(NAN), hum(NAN), pres(NAN);
  BME280::TempUnit tempUnit(BME280::TempUnit_Celsius);
  BME280::PresUnit presUnit(BME280::PresUnit_hPa);



// Initialiser esp clienten, skift evt.navne hvis du har flere kørende
WiFiClient espClient;
PubSubClient client(espClient);

// tids variabler til udmåling af pauser mellem aflæsningerne
long now = millis();
long lastMeasure = 0;


// Bruges til at forninde sifg med MQTT brokeren
void reconnect() {
  // Loop indtil vi har forbindelse
  while (!client.connected()) {
    Serial.print("Forsøger en MQTT forbindelse...");
    // Attempt to connectupdateup
    if (client.connect(clientname)) {
      Serial.println("forbindelse");
      } else {
      Serial.print("fejlede, returcode=");
      Serial.print(client.state());
       Serial.println(" prøv igen om 5 sekunder");
       // Went 5 sekunder
```

```
    delay(5000);
   }
  }
}

void setup() {
  Serial.begin(115200);
 // Først skal vi have forbindelse på wifi
 Serial.println();
 Serial.print("Forbinder til ");
 Serial.println(ssid);
 WiFi.mode(WIFI_STA);
 WiFi.begin(ssid, password);
 while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.print(".");
 }
 Serial.println("");
 Serial.print("WiFi forbundet - ESP IP adresse: ");
Serial.println(WiFi.localIP());
 // Port default for 8266
 // ArduinoOTA.setPort(8266);
 // Hostname default til esp8266-[ChipID]
ArduinoOTA.setHostname(otahostname);
 // Ingen authentication by default
 //ArduinoOTA.setPassword((const char *)"123");
  ArduinoOTA.onStart([]() {
   Serial.println("Start");
 });
 ArduinoOTA.onEnd([]() {
   Serial.println("\nEnd");
 });
 ArduinoOTA.onProgress([](unsigned int progress, unsigned int total) {
   Serial.printf("Progress: %u%%\r", (progress / (total / 100)));
 });
 ArduinoOTA.onError([](ota_error_t error) {
   Serial.printf("Error[%u]: ", error);
   if      (error == OTA_AUTH_ERROR   ) Serial.println("Auth fejlede");
   else if (error == OTA_BEGIN_ERROR  ) Serial.println("Start fejlede");
   else if (error == OTA_CONNECT_ERROR) Serial.println("Forbindelse fejlede");
   else if (error == OTA_RECEIVE_ERROR) Serial.println("Modtagelse fejlede");
   else if (error == OTA_END_ERROR    ) Serial.println("Afslutning fejlede");
 });
  ArduinoOTA.begin();
  Wire.begin();
  bme.begin();

 client.setServer(mqtt_server, 1883);
```

```
}

void loop() {
 ArduinoOTA.handle();

 if (WiFi.status() == WL_CONNECTED) {
   if (!client.connected()) {
     reconnect();
    }
  }

 if(!client.loop())
   client.connect(clientname);

 now = millis();
 // Publiser ny temperatur hver 15. sekund
 if (now - lastMeasure > 15000) {
   lastMeasure = now;
   bme.read(pres, temp, hum, tempUnit, presUnit);
   Serial.println(pres);
   Serial.println(temp);
   Serial.println(hum);
   Serial.println(tempUnit);
   Serial.println(presUnit);
   static char temperaturud[7];
   dtostrf(temp, 6, 2, temperaturud);
 // Publiser Temperatur
   client.publish(topic, temperaturud);
   Serial.println(temperaturud);
   dtostrf(hum, 6, 0, temperaturud);
// Publiser humidity
   client.publish(topic2, temperaturud);
   Serial.println(temperaturud);
   dtostrf(pres, 6, 0, temperaturud);
// Publiser Pressure
   client.publish(topic3, temperaturud);
Serial.println(temperaturud);
   }
}
```