```
// Βιβλιοθήκες
#include <WiFiS3.h>
#include <DHT.h>
#include "ThingSpeak.h"
#include <Adafruit_GFX.h>    // Core graphics library
#include <Adafruit_ST7789.h> // Hardware-specific library for ST7789
#include <SPI.h>             // Arduino SPI library
#include <Adafruit_GFX.h>    // Core graphics library

// Ορισμός αντικειμένου οθόνης
#define TFT_CS    10
#define TFT_DC     7
#define TFT_RST    8
Adafruit_ST7789 tft = Adafruit_ST7789(TFT_CS, TFT_DC, TFT_RST);

// Ορισμός αντικειμένων για WiFi & DHT11
WiFiClient client;
DHT dht11(2, DHT11);

// Ορισμός παραμέτρων σύνδεσης για WiFi & ThingSpeak
char ssid[] = "1gktoumpas";
char pass[] = "tsfd1387";
unsigned long myChannelNumber = 3272920;
const char* myWriteAPIKey = "P3UL5BT8URFA72G5";
int status = WL_IDLE_STATUS;

// Ορισμός μεταβλητών για τον έλεγχο του χρόνου μετάδοσης
unsigned long lastReadTime = 0;
unsigned long lastUploadTime = 0;

// Ορισμός μεταβλητών για τα δεδομένα καταγραφής από τους αισθητήρες
float humidity = 0;
float temperature = 0;
float co2In = 0;
float co2Out = 0;
int count = 1;

// Συνάρτηση σύνδεσης στο Διαδίκτυο
void connectToWiFi() {
  if (WiFi.status() == WL_NO_MODULE) {
    Serial.println("Η σύνδεση στο Διαδίκτυο απέτυχε!!!");
    while (true) ;
  }

  // Προσπάθεια σύνδεσης στο Διαδίκτυο
  while (status != WL_CONNECTED) {
    Serial.print("Προσπάθεια σύνδεσης στο Διαδίκτυο SSID: ");
    Serial.println(ssid);
    status = WiFi.begin(ssid, pass);
    delay(10000);
  }
```

```
  // Στοιχεία σύνδεσης στο Διαδίκτυο
  Serial.print("Σύνδεση στο Διαδίκτυο -->");
  Serial.print("WiFi Status: ");
  Serial.print(WiFi.status());
  Serial.print("   |   SSID: ");
  Serial.println(WiFi.SSID());
}

// Σύνάρτηση επανασύνδεσης στο Διαδίκτυο, αν αποσυνδεθεί
void reconnectWiFi() {
  if (WiFi.status() != WL_CONNECTED) {
    Serial.println("Επανασύνδεση στο Διαδίκτυο....");
    connectToWiFi();
  }
}

// Συνάρτηση αποστολής δεδομένων στο ThingSpeak
void updateThingSpeak() {
  ThingSpeak.setField(1, humidity);
  ThingSpeak.setField(2, temperature);
  ThingSpeak.setField(3, co2In);
  ThingSpeak.setField(4, co2Out);
  int response = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
}

// Συνάρτηση ανάγνωσης δεδομένων από τους αισθητήρες
void readSensorData() {
  humidity = dht11.readHumidity();
  temperature = dht11.readTemperature();
  co2In = analogRead(A0);
  co2Out = analogRead(A1);
```

```
 // Έλεγχος ορθών τιμών δεδομένων από τους αισθητήρες και εμφάνιση στην οθόνη
  if (not (isnan(humidity) || isnan(temperature)) || isnan(co2In) || isnan(co2Out)) {
    Serial.print("Humidity: ");
    Serial.print(humidity);
    Serial.print("% | Temperature: ");
    Serial.print(temperature);
    Serial.print("°C");
    Serial.print(" | co2In: ");
    Serial.print(co2In);
    Serial.print(" | co2Out: ");
    Serial.println(co2Out);

    tft.fillScreen(ST77XX_GREEN);
    tft.setCursor(35, 30);
    tft.setTextColor(ST77XX_BLUE);
    tft.setTextSize(3);
    tft.println("MEASUREMENTS");
    tft.println("");
    tft.print("Humidity:");
    tft.println(humidity);
    tft.print("Temperature:");
    tft.println(temperature);
    tft.print("CO2 in:");
    tft.println(co2In);
    tft.print("CO2 out:");
    tft.println(co2Out);
    tft.println("");
    tft.print("count:");
    tft.println(count);
  }
}


void setup() {
  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(LED_BUILTIN, LOW);

  Serial.begin(9600);               // Καθορισμός ταχύτητας σειριακής επικοινωνίας με τον υπολογιστή
  tft.init(240, 320, SPI_MODE2);    // Ορισμός ανάλυσης οθόνης σε 240X320
  tft.setRotation(3);               // Ορισμός προσανατολισμού οθόνης
  tft.setTextWrap(false);           // Χωρίς αναδίπλωση κειμένου
  tft.fillScreen(ST77XX_GREEN);     // Ορισμός φόντου στην οθόνη

  connectToWiFi();                  // Αρχική σύνδεση στο Διαδίκτυο
  delay(20000);
  ThingSpeak.begin(client);         // Αρχικοποίηση της σύνδεσης με το ThingSpeak
  dht11.begin();                    // Αρχικοποίηση του αισθητήρα DHT11
}
```

```
void loop() {
  reconnectWiFi();                          // Επανασύνδεση στο Διαδίκτυο, αν χρειάζεται

  unsigned long currentTime = millis();          // Αρχικοποίηση του χρόνου αποστολής δεδομένων

  if (currentTime - lastReadTime >= 60000) {      // Ανάγνωση τιμών από τους αισθητήρες κάθε 30
δευτερόλεπτα
    readSensorData();                        // Διάβασμα δεδομένων από τους αισθητήρες
    updateThingSpeak();                        // Αποστολή δεδομένων στο ThingSpeak
    count=count+1;

    lastReadTime = currentTime;                 // Ορισμός του χρόνο αποστολής των δεδομένων
  }
}
```