```
Programa ESP32 con pantalla y sensor Turbiedad

#include <TFT_eSPI.h>
#include <Adafruit_MQTT.h>
#include <Adafruit_MQTT_Client.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include <time.h>  // Para obtener la hora desde un servidor NTP

// Configuración de la pantalla TFT
TFT_eSPI tft = TFT_eSPI();  // Configurar la pantalla TFT

// Configuración de WiFi y Adafruit IO
#define WLAN_SSID       "Familia Baretta* 2.4GHz"
#define WLAN_PASS       "jesusesvida"
#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883
#define AIO_USERNAME    "Charly76"
#define AIO_KEY         "aio_peET13z39trG2TzxrBPVyNzC0qm9"

WiFiClient client;
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT,
AIO_USERNAME, AIO_KEY);

// Feeds de Adafruit IO
Adafruit_MQTT_Subscribe pressureFeed = Adafruit_MQTT_Subscribe(&mqtt,
AIO_USERNAME "/feeds/presion");
Adafruit_MQTT_Subscribe temperatureFeed =
Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/temperatura");
Adafruit_MQTT_Subscribe conductivityFeed =
Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/conductividad");
Adafruit_MQTT_Publish turbidityFeed = Adafruit_MQTT_Publish(&mqtt,
AIO_USERNAME "/feeds/turbiedad");

// Configuración de ThingSpeak
const char* thingspeak_server = "api.thingspeak.com";  // Servidor de
ThingSpeak
const char* api_key = "XHHZDQFO5NNW6ZN4";  // Reemplaza "TU_API_KEY"
con tu clave API de ThingSpeak

// Desfase horario para Argentina (UTC-3)
const long gmtOffset_sec = -3 * 3600;
```

```
const int daylightOffset_sec = 0;  // Argentina no usa horario de
verano

// Configuración de la hora desde un servidor NTP
void setupTime() {
  configTime(gmtOffset_sec, daylightOffset_sec, "pool.ntp.org",
"time.nist.gov");
  Serial.println("Esperando la sincronización de tiempo...");
  delay(2000);  // Espera para la sincronización
}

// Función para obtener la hora y fecha actual
String getTime() {
  time_t now = time(nullptr);
  struct tm* timeinfo = localtime(&now);
  char buffer[25];
  strftime(buffer, sizeof(buffer), "%Y-%m-%d %H:%M:%S", timeinfo);
  return String(buffer);
}

void setup() {
  // Inicializar la pantalla TFT
  tft.init();
  tft.setRotation(2);
  tft.fillScreen(TFT_BLACK);

  // Mostrar título
  tft.setTextColor(TFT_RED, TFT_WHITE);
  tft.setTextSize(2);
  tft.fillRect(0, 0, 240, 60, TFT_WHITE);
  tft.setCursor(10, 10);
  tft.println("Control de Calidad");
  tft.setCursor(10, 30);
  tft.println("     del Agua");

  // Conectar a WiFi
  Serial.begin(115200);
  Serial.print("Conectando a ");
  Serial.println(WLAN_SSID);
  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
```

```
  }
  Serial.println();
  Serial.println("WiFi conectado");

  // Configurar la hora
  setupTime();

  // Conectar a Adafruit IO
  mqtt.subscribe(&pressureFeed);
  mqtt.subscribe(&temperatureFeed);
  mqtt.subscribe(&conductivityFeed);
}

void loop() {
  // Conexión a Adafruit IO
  MQTT_connect();

  // Comprobar si hay nuevos datos
  Adafruit_MQTT_Subscribe *subscription;
  while ((subscription = mqtt.readSubscription(5000))) {
    if (subscription == &pressureFeed) {
      float pressure = atof((char *)pressureFeed.lastread);
      mostrarDatos("Presion", pressure, "BAR", 80, TFT_PURPLE);
    }

    if (subscription == &temperatureFeed) {
      float temperature = atof((char *)temperatureFeed.lastread);
      mostrarDatos("Temp", temperature, "°C", 120, TFT_BLUE);
    }

    if (subscription == &conductivityFeed) {
      float conductivity = atof((char *)conductivityFeed.lastread);
      mostrarDatos("Cond", conductivity, "µS", 160, TFT_GREEN);
    }
  }

  // Leer el valor del sensor de turbiedad
  int sensorValue = analogRead(34);
  float turbidity = map(sensorValue, 0, 4095, 0, 100) / 100.0;  //
Ajustar según la calibración

  // Publicar el valor de turbidez
  turbidityFeed.publish(turbidity);
```

```
  sendToThingSpeak(turbidity, "field3");

  // Mostrar turbidez en la pantalla TFT
  mostrarDatos("Turb", turbidity, "NTU", 200, TFT_RED);

  // Mostrar hora y fecha en la pantalla TFT
  String currentTime = getTime();
  String currentDate = currentTime.substring(0, 10);
  String currentHour = currentTime.substring(11);

  tft.fillRect(0, 280, 240, 30, TFT_DARKGREY);  // Fondo para la hora
  tft.fillRect(0, 240, 240, 30, TFT_NAVY);      // Fondo para la fecha

  tft.setTextColor(TFT_WHITE, TFT_DARKGREY);
  tft.setTextSize(2);
  tft.setCursor(10, 290);
  tft.print("Hora: ");
  tft.println(currentHour);

  tft.setTextColor(TFT_WHITE, TFT_NAVY);
  tft.setTextSize(2);
  tft.setCursor(10, 250);
  tft.print("Fecha: ");
  tft.println(currentDate);

  delay(30000);  // Pausa antes de la siguiente lectura
}

// Función para mostrar los datos en la pantalla
void mostrarDatos(String etiqueta, float valor, String unidad, int
posY, uint16_t color) {
  tft.fillRect(0, posY, 240, 30, color);
  tft.setTextColor(TFT_WHITE, color);
  tft.setTextSize(2);
  tft.setCursor(10, posY + 5);
  tft.printf("%s: %.2f %s", etiqueta.c_str(), valor, unidad.c_str());
}

// Función para conectar a Adafruit IO
void MQTT_connect() {
  int8_t ret;
  if (mqtt.connected()) {
    return;
```

```
  }

  Serial.print("Conectando a Adafruit IO... ");

  while ((ret = mqtt.connect()) != 0) {
    Serial.println(mqtt.connectErrorString(ret));
    Serial.println("Reconectando en 5 segundos...");
    delay(5000);
  }

  Serial.println("Conectado a Adafruit IO");
}

// Función para enviar datos a ThingSpeak
void sendToThingSpeak(float value, String field) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    String url = "http://api.thingspeak.com/update?api_key=" +
String(api_key) + "&" + field + "=" + String(value);
    http.begin(url);
    int httpCode = http.GET();
    if (httpCode > 0) {
      String payload = http.getString();
      Serial.println(payload);
    } else {
      Serial.println("Error en la conexión HTTP");
    }
    http.end();
  }
}
```