A Lab-on-a-Drone for Turbidimetric and Nephelometric Analysis of Environmental Water Samples with Real-Time Data Transmission

Jackson Rodrigo dos Santos Silva[a], Vinícius Alcântara de Carvalho[a], Gabriel Bercley de Lima Vitorino[a], João Paulo Barbosa de Almeida[a], Alldebarã Fausto Ferreira[a], Willian Toito Suarez[b], Elton de Lima Borges[a], Vagner Bezerra dos Santos[a]*

[a]LIA[3] - (Laboratory of Instrumentation and Automation in Applied Analytics) of the Federal University of Pernambuco - Recife-PE, 50740-560, Brazil.
[b] Department of Chemistry, Federal University of Viçosa, Viçosa, MG, Brazil
*Corresponding author.
vagner.bsantos@ufpe.br / Av. Jornalista Anibal Fernandes, s/n, Cidade Universitária, Recife, PE, 50740-560, Brazil

Supplementary Information

Equations:

$$LOD = \frac{(3 \times SD)}{m} \qquad Equation\ S1$$

SD is the Standard deviation, and m is the slope of the analytical curve

$$LOQ = \frac{(10 \times SD)}{m} \qquad Equation\ S2$$

$$RSD\ (\%) = \left(\frac{SD}{A}\right) \times 100\% \qquad Equation\ S3$$

A is an average

$$RE\ (\%) = \left(\frac{X_{Proposed} - X_{Reference}}{X_{Reference}}\right) \times 100\% \qquad Equation\ S4$$

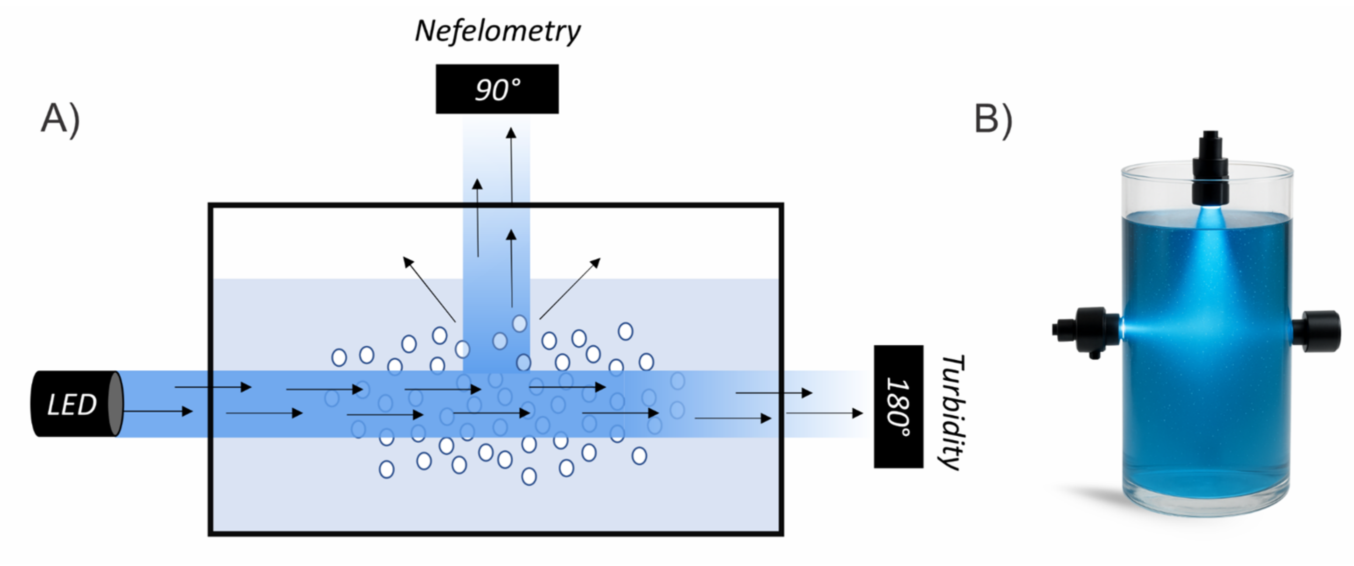

Figure S1. (A) Operating principle of an optical turbidimeter, illustrating nephelometric and turbidimetric measurement modes; and (B) operating principle of the developed prototype.

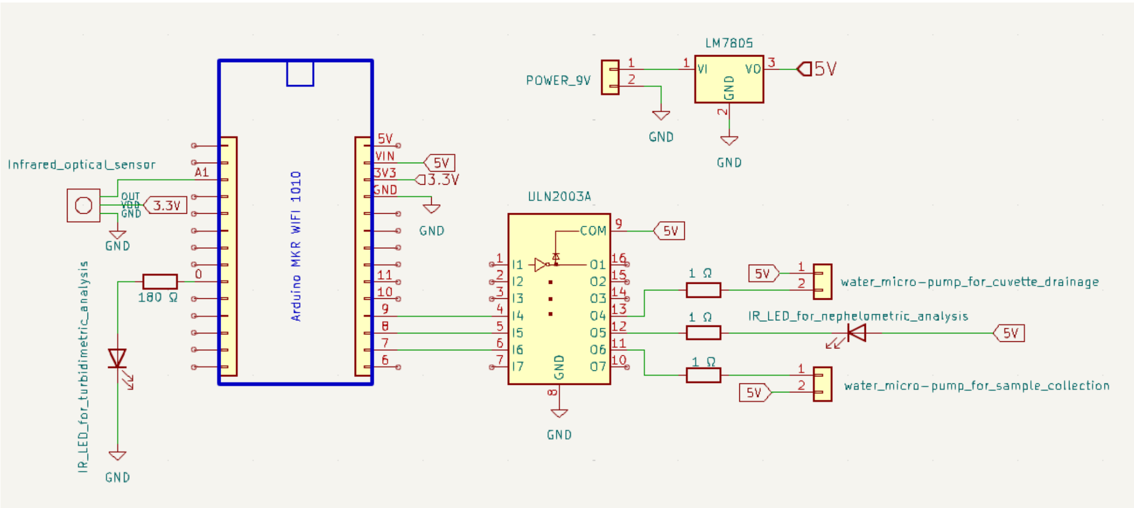

Figure S2: Schematic representation of the electronic circuit of the developed turbidimeter/nephelometer.

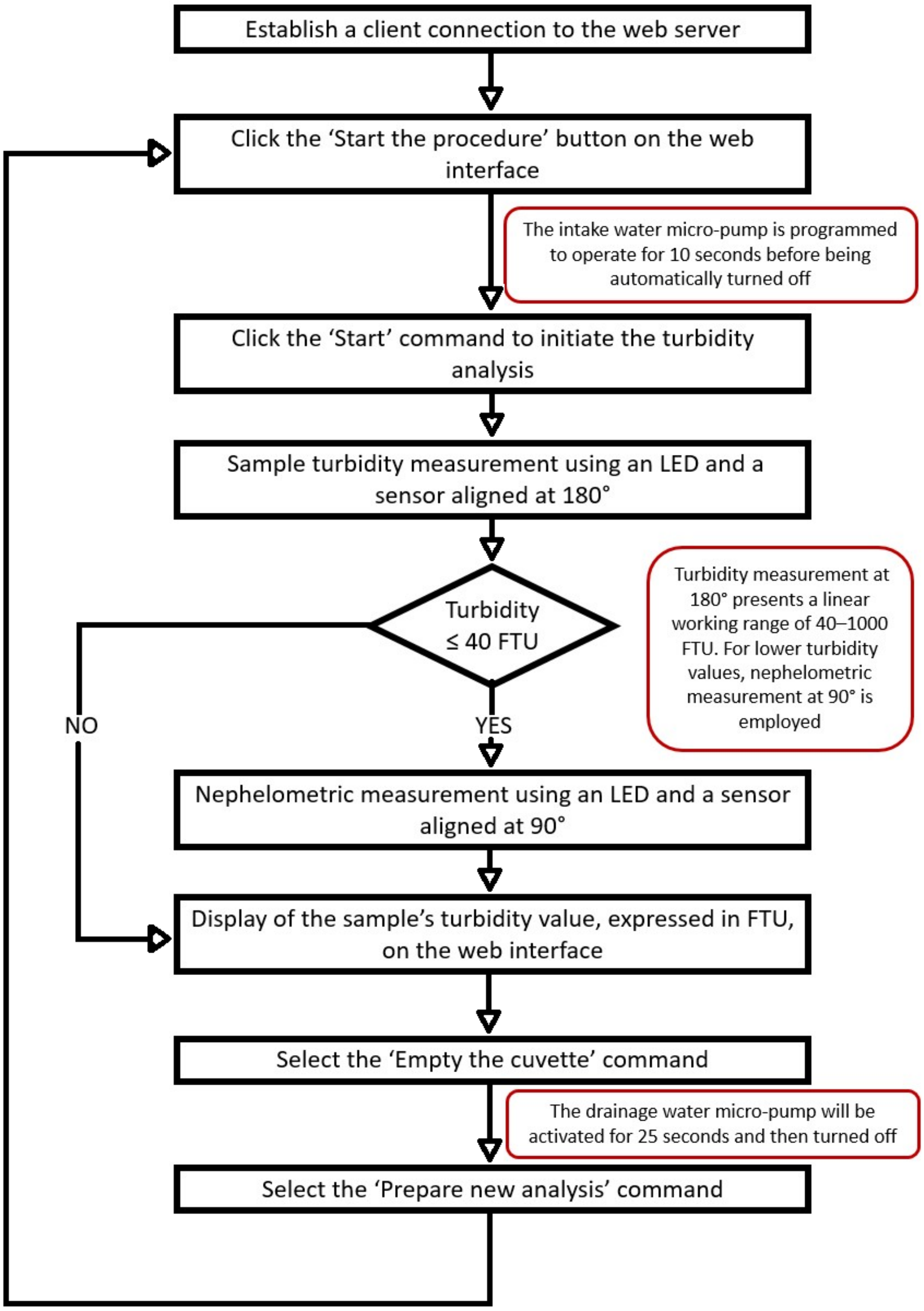

Figure S3: Flowchart illustrating the operation of the algorithm for performing water sampling and turbidity analysis at multiple sampling points.

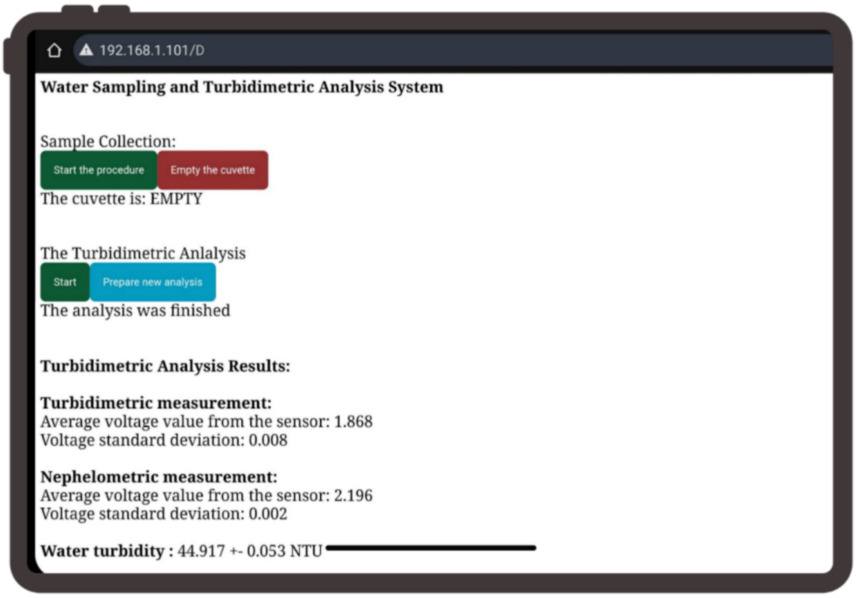

Figure S4. Smartphone interface displaying real-time measurement results.

Figure S5: (A) Portable photovoltaic system used to power auxiliary equipment in the field. (B) Portable Wi-Fi router used to create the local network.

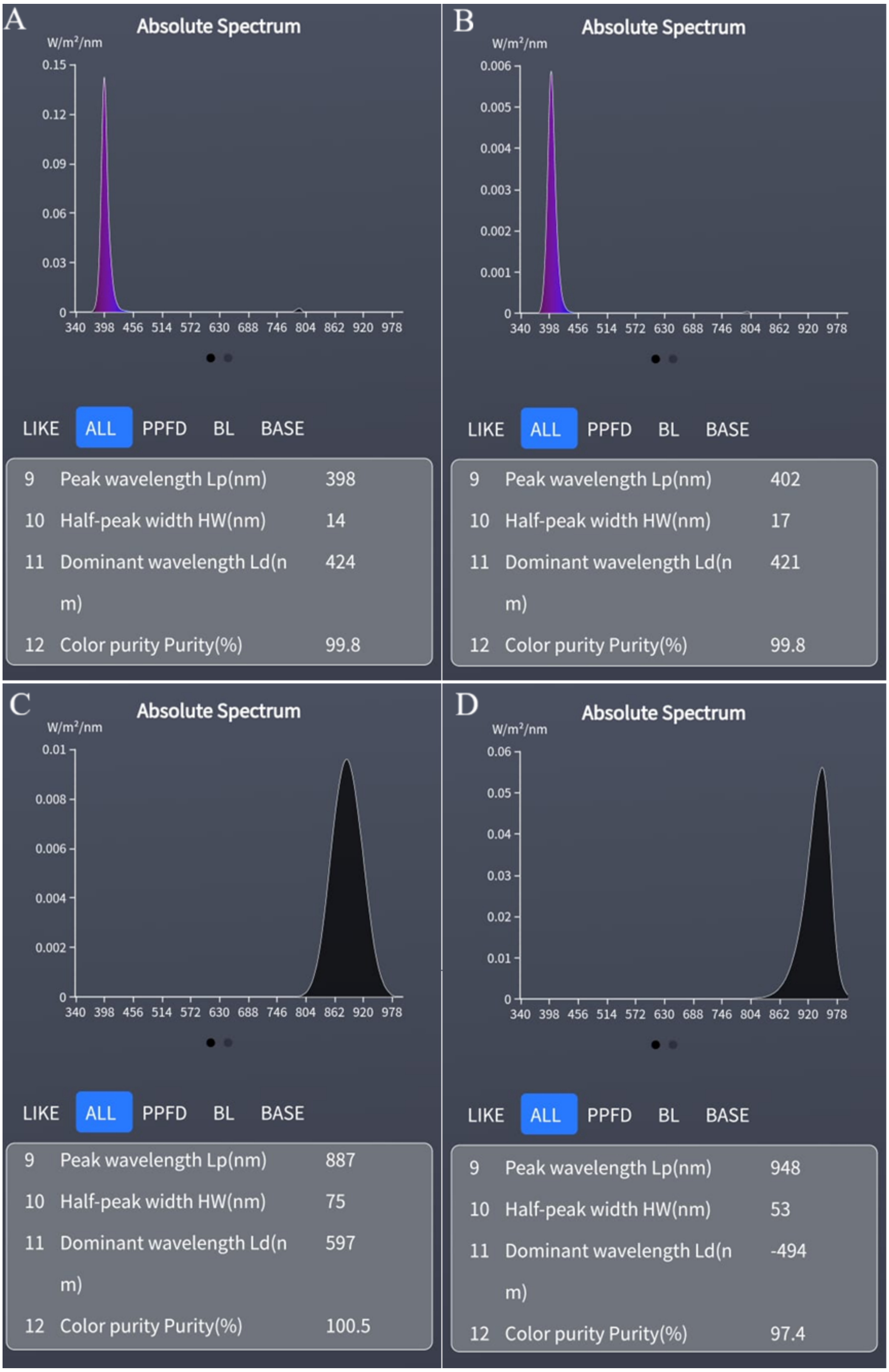
A
Absolute Spectrum
W/m²/nm
0.15
0.12
0.09
0.06
0.03
0
340 398 456 514 572 630 688 746 804 862 920 978
LIKE ALL PPFD BL BASE
9 Peak wavelength Lp(nm) 398
10 Half-peak width HW(nm) 14
11 Dominant wavelength Ld(nm) 424
12 Color purity Purity(%) 99.8
B
Absolute Spectrum
W/m²/nm
0.006
0.005
0.004
0.003
0.002
0.001
0
340 398 456 514 572 630 688 746 804 862 920 978
LIKE ALL PPFD BL BASE
9 Peak wavelength Lp(nm) 402
10 Half-peak width HW(nm) 17
11 Dominant wavelength Ld(nm) 421
12 Color purity Purity(%) 99.8
C
Absolute Spectrum
W/m²/nm
0.01
0.008
0.006
0.004
0.002
0
340 398 456 514 572 630 688 746 804 862 920 978
LIKE ALL PPFD BL BASE
9 Peak wavelength Lp(nm) 887
10 Half-peak width HW(nm) 75
11 Dominant wavelength Ld(nm) 597
12 Color purity Purity(%) 100.5
D
Absolute Spectrum
W/m²/nm
0.06
0.05
0.04
0.03
0.02
0.01
0
340 398 456 514 572 630 688 746 804 862 920 978
LIKE ALL PPFD BL BASE
9 Peak wavelength Lp(nm) 948
10 Half-peak width HW(nm) 53
11 Dominant wavelength Ld(nm) -494
12 Color purity Purity(%) 97.4

Figure S6. Emission spectra of the LEDs used for turbidimetric measurements (A and C). Emission spectra of the LEDs used for nephelometric measurements (B and D).

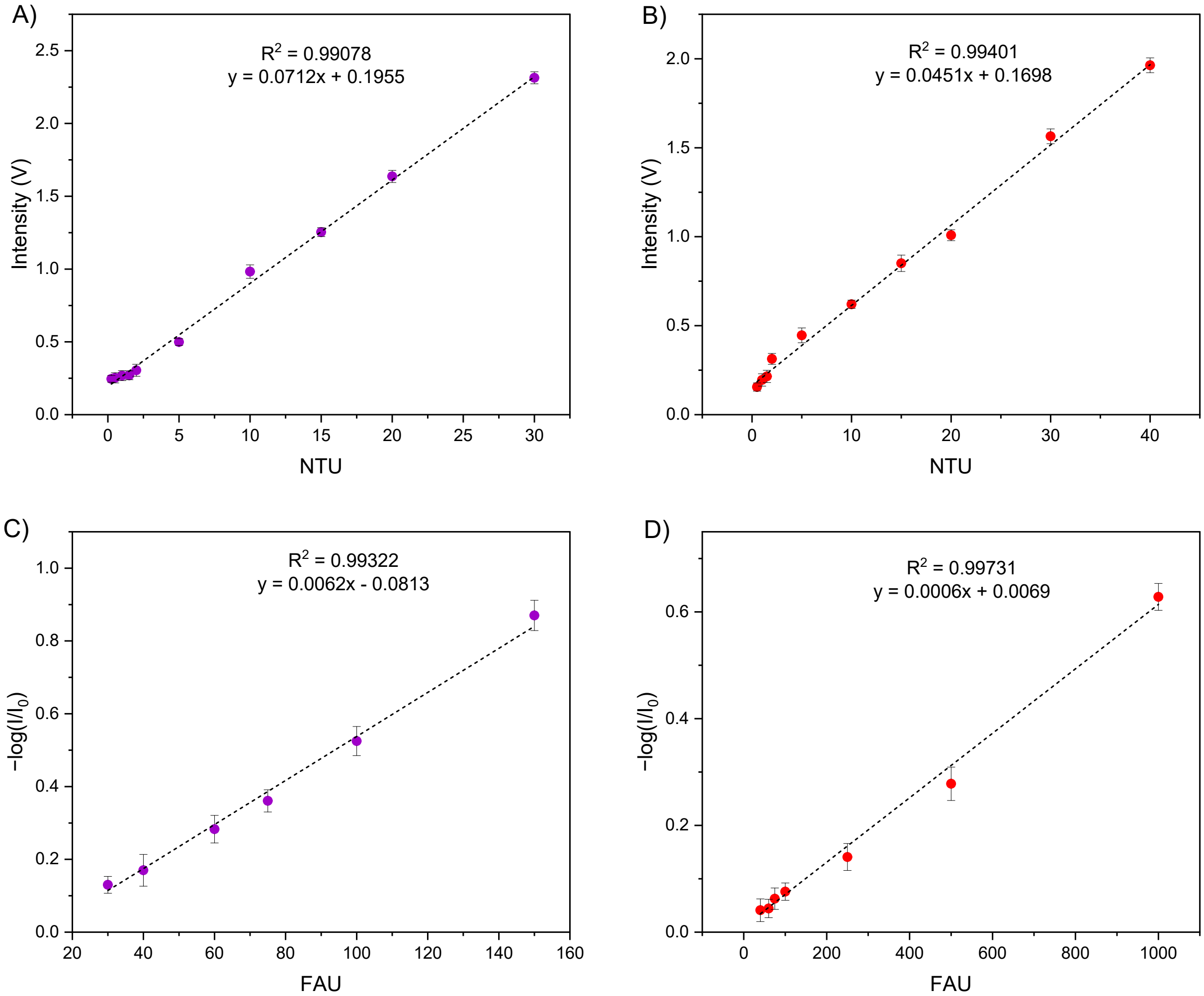

Figure S7. Analytical curve for Nephelometry at 405 nm (A) and 890 nm (B) and Turbidimetry at 405 nm (C) and 890 (D). All data were obtained in triplicate (n = 3).

Table S1. Comprehensive code for the control of a water-sampling and turbidity-analysis system.

```
#include <Arduino.h>
#include <WiFiNINA.h>
#include <Math.h>


char ssid[] = "Wi-Fi network SSID";
char pass[] = "Password";

WiFiServer server(80);


WiFiClient client = server.available();


//Variable declaration section
int LED=0;
int LEDnef=8;
float dados;
float dadosnef;
float tensao;
float tensaonef;
int quantidadedemedidas=100;
float medicoes[120];
float medicoesnef[120];
int pulsosdeluz=0;
int rep=0;
int repnef=0;
double media=0;
double medianef=0;
double desvio=0;
double desvionef=0;
double desviototal=0;
double desviototalnef=0;
int cd =0;
int analisefinalizada=1;
int zerar=0;
float NTU=0;
float NTUnef=80;
float NTUdesvio=0;
float NTUdesvionef=0;



// Sampling variables
int microbomba_c=7; //water inlet micro-pump
int microbomba_e=9; // Micro-pump for water drainage
int coleta_ativar=0;
int coleta_desativar=0;
int vazio=0;
int esvaziar=0;
float tempo_c=10000;
float tempo_e=25000;




void setup() {
  pinMode(microbomba_e, OUTPUT);
  pinMode(microbomba_c, OUTPUT);
  pinMode(LED, OUTPUT);
  pinMode(LEDnef, OUTPUT);
  digitalWrite(LED, LOW);
  digitalWrite(LEDnef, LOW);

Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, pass);
  while (WiFi.status() != WL_CONNECTED) {
      delay(100);
      Serial.print(".");
```



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



  Serial.println("IP address: ");

Serial.println(WiFi.localIP());



  server.begin();


}


void loop() {

  client = server.available();
if (client) {
    Serial.println("new client");
    String currentLine = "";
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        Serial.write(c);
        if (c == '\n') {
          if (currentLine.length() == 0) {
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println();
            client.print("<b>Water Sampling and Turbidimetric Analysis System</b>");
            client.print("<br><br><br>");


            // Operation of the sample collection and cuvette drainage system
            client.print("Sample Collection:");
            client.print("<br>");
            client.print("<a href=\"/H\"><button style='background: #0C5934; border-radius: 8px; padding: 20px; cursor: pointer; color: #fff; border:
none; font-size: 16px;'>Start the procedure</button></a>");
            client.print("<a href=\"/D\"><button style='background: #962F2F; border-radius: 8px; padding: 20px; cursor: pointer; color: #fff; border:
none; font-size: 16px;'>Empty the cuvette</button></a>");
            client.print("<br>");
            client.print("The cuvette is: ");
           if((coleta_ativar==1)&&(coleta_desativar==0)){
            digitalWrite(microbomba_c,HIGH);
            delay(tempo_c);
            digitalWrite(microbomba_c,LOW);
            coleta_desativar=1;

           }
           if(coleta_desativar==1){
           client.print(" FUll");
           client.print("<br>");
           }
           if((esvaziar==1)&&(vazio==0)){
            digitalWrite(microbomba_c,HIGH);
            digitalWrite(microbomba_e, HIGH);
            delay(tempo_e);
            digitalWrite(microbomba_c,LOW);
            digitalWrite(microbomba_e,LOW);
            vazio=1;
           }
           if(vazio==1){
            client.print(" EMPTY");
            client.print("<br>");
           }
            client.print("<br>");
            client.print("<br>");


            // Sample Turbidity Analysis
            client.print("The Turbidimetric Analysis");
            client.print("<br>");
            client.print("<a href=\"/E\"><button style='background: #0C5934; border-radius: 8px; padding: 20px; cursor: pointer; color: #fff; border:
none; font-size: 16px;'>Start</button></a>");
             client.print("<a href=\"/N\"><button style='background: #069cc2; border-radius: 8px; padding: 20px; cursor: pointer; color: #fff; border:
none; font-size: 16px;'>Prepare new analysis</button></a>");
            client.print("<br>");


            if((pulsosdeluz ==0)&&(analisefinalizada==1)){
              media=0;
```

```
            medianef=0;
            desvio=0;
            desvionef=0;
            desviototal=0;
            desviototalnef=0;



            NTU=0;
            NTUnef=0;
            NTUdesvio=0;
            NTUdesvionef=0;
            client.print("Click the 'start' button to start another analysis");
            client.print("<br>");

if(zerar==1){
             for(int zerando =0; zerando<quantidadedemedidas; zerando++){
               medicoes[zerando]=0;
               medicoesnef[zerando]=0;
               }
               zerar=0;
             }
          }
           if ((pulsosdeluz ==1)&&(analisefinalizada==1)){


            // Sample turbidity measurement using an LED and a sensor aligned at 180°
             digitalWrite(LED, HIGH);
            for (rep=0; rep<quantidadedemedidas; rep++){
              if(rep==0){
                delay(100);
              }
              delay(5);
              dados=analogRead(A1);
              tensao=(dados*3.3)/1023;
              medicoes[rep]=tensao;
              media= media+medicoes[rep];
            }

            media= (media/quantidadedemedidas);
            for(int cd =0; cd<quantidadedemedidas; cd++){
              float diferenca=(medicoes[cd]- media);
              desvio = desvio + pow(diferenca,2);
            }
            desviototal=(desvio/quantidadedemedidas);
            desviototal=pow(desviototal,(0.5));


            digitalWrite(LED, LOW);

          //Resulting equation of the calibration curve for turbidity measurement using an LED and a sensor aligned at 180°.
           NTU=((-714.286)*media)+1275.428;
           NTUdesvio=((-714.286)*desviototal);
           NTUdesvio=pow((pow(NTUdesvio,2)),(0.5));


           delay(100);

           // Measurements using an LED and a sensor aligned at 90°
            if(NTU<=40){
            digitalWrite(LEDnef, HIGH);
            for (repnef=0; repnef<quantidadedemedidas; repnef++){
              if(repnef==0){
                delay(100);
              }
              delay(5);
              dadosnef=analogRead(A1);
              tensaonef=(dadosnef*3.3)/1023;
              medicoesnef[repnef]=tensaonef;
              medianef= medianef+medicoesnef[repnef];
            }
             digitalWrite(LEDnef, LOW);

            medianef= (medianef/quantidadedemedidas);
            for(int cdnef =0; cdnef<quantidadedemedidas; cdnef++){
              float diferencanef=(medicoesnef[cdnef]- medianef);
              desvionef = desvionef + pow(diferencanef,2);
            }
            desviototalnef=(desvionef/quantidadedemedidas);
            desviototalnef=pow(desviototalnef,(0.5));



           //Equation obtained from the calibration curve for nephelometric measurements performed with an LED and a sensor aligned at 90°
           NTU=(22.173*medianef)-3.7649;
```



```
            NTUdesvio=(22.173*desviototalnef);
             }


             pulsosdeluz =0;



             analisefinalizada=0;
           }
           if(analisefinalizada ==0){
           client.print("The analysis was finished");
           client.print("<br>");
           client.print("<br><br>");
           client.print("<b>Turbidimetric Analysis Results: </b>");
           client.print("<br><br>");



client.print("<b> Turbidimetric measurement: </b> <br>");
           client.print("Average voltage value from the sensor:  ");
           client.print(media,3);
           client.print("<br>");
           client.print("Voltage standard deviation:  ");
           client.print(desviototal,3);
           client.print("<br>");
           client.print("<br>");
           client.print("<b> Nephelometric measurement: </b> <br>");
           client.print("Average voltage value from the sensor:  ");
           client.print(medianef,3);
           client.print("<br>");
           client.print("Voltage standard deviation:  ");
           client.print(desviototalnef,3);
           client.print("<br><br>");
           client.print(" <b> Water turbidity : </b>  ");
           client.print(NTU,3);
           client.print(" +- ");
           client.print(NTUdesvio,3);
           client.print(" FTU ");
           client.print("<br>");
           client.print("<br>");
           client.println();



           }

           break;
         }
         else {
           currentLine = "";
         }
       }
       else if (c != '\r') {
         currentLine += c;
       }


       if (currentLine.endsWith("GET /H")) {
       coleta_ativar=1;
       vazio=0;
       esvaziar=0;
       }
       if (currentLine.endsWith("GET /D")) {
       coleta_desativar=0;
       coleta_ativar=0;
       esvaziar=1;
       }
       if (currentLine.endsWith("GET /E")) {
          pulsosdeluz =1;
       }
       if (currentLine.endsWith("GET /N")) {
           digitalWrite(LED, LOW);
           pulsosdeluz=0;
           analisefinalizada=1;
           zerar=1;
       }

     }
   }
   client.stop();
   Serial.println("client disconnected");
 }
}
```

Table S2. Principles used by the metric system of the AGREE analytical calculator

Principles	Criterion
1	Sample handling
2	Minimum sample size and number
3	In situ analysis
4	Low-energy operations and reduced reagent count
5	Automated and miniaturized method
6	No derivatization
7	Low generation of analytical waste
8	Multi-analytical or multi-parameter method
9	Minimal energy use
10	Reagents obtained from renewable sources
11	No toxic reagents
12	Increased operator safety

Table S3. Principles employed by the GAPI metric system.

Principles	Criterion
	Sample preparation
1	Collection
2	Preservation
3	Transport
4	Storage
5	Type of method
6	Scale of extraction
7	Solvents/reagentes used
8	Additional treatments
	Reagents and solvents
9	Amount
10	Health hazard
11	Safety hazard
	Instrumentation
12	Energy
13	Occupational hazard
14	Waste
15	Waste treatment

.

Table S4. Principles employed by the BADI metric system.

Principles	Criterion
1	Type of analysis
2	Single or multiple element analysis
3	Analytical technique
4	Simultaneous sample preparation
5	Sample preparation
6	Samples per hour
7	Reagents and materials
8	Preconcentration
9	Degree of automation
10	Sample quantity

