- .... .   . -.-. --- ... -.-- ... - . --
.. --- - -...- .-- . -... -...- .- .. / ..-. --- .-. / -..- .-.

-... -.-- / -- .- - - . --- / -- --- .-. .. -.-. --- -. .. / -.--. .-- .. - .... / -.-. --- .-.. .-.. .- -... --- .-. .- - .. --- -. /
--- ..-. / .-.. ..- .. --.. / ...- . .-.. .... --- -.--.-

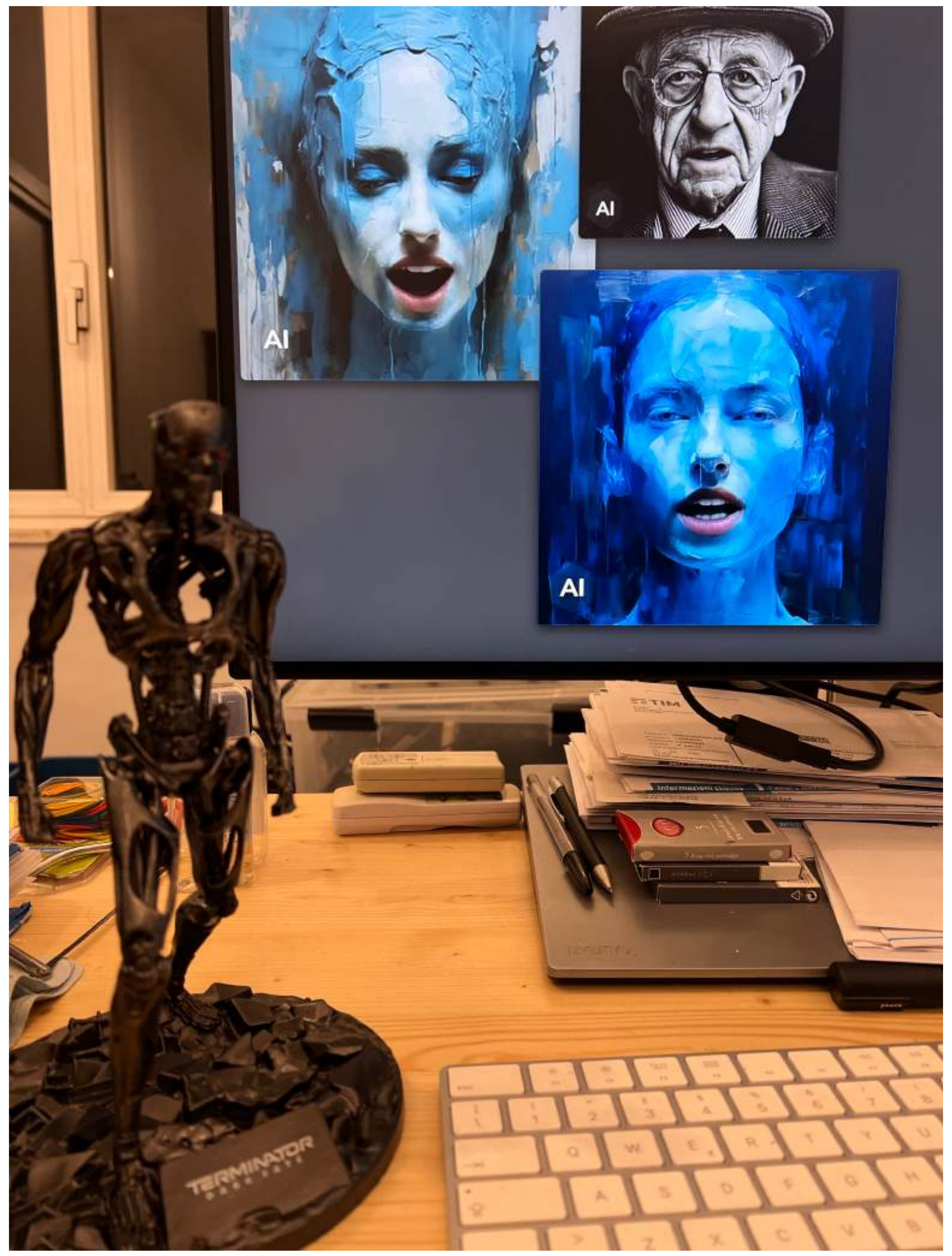
The Ecosystem IOT-WEB-AI for XR

by Matteo Moriconi (VFXRIO)
(with collaboration of Prof. Luiz Velho - IMPA)

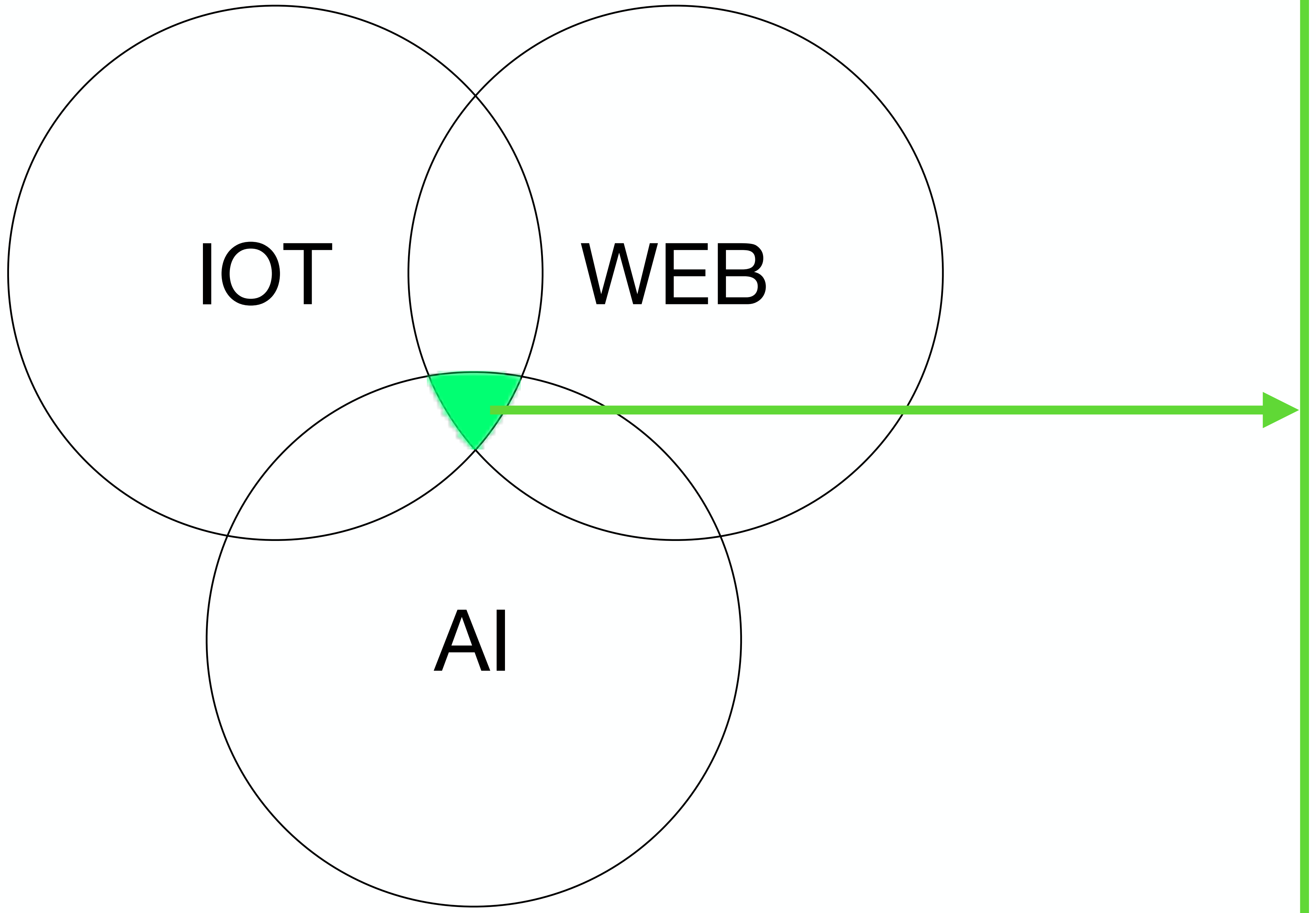
THE ECOSYSTEM
IOT-WEB-AI for XR

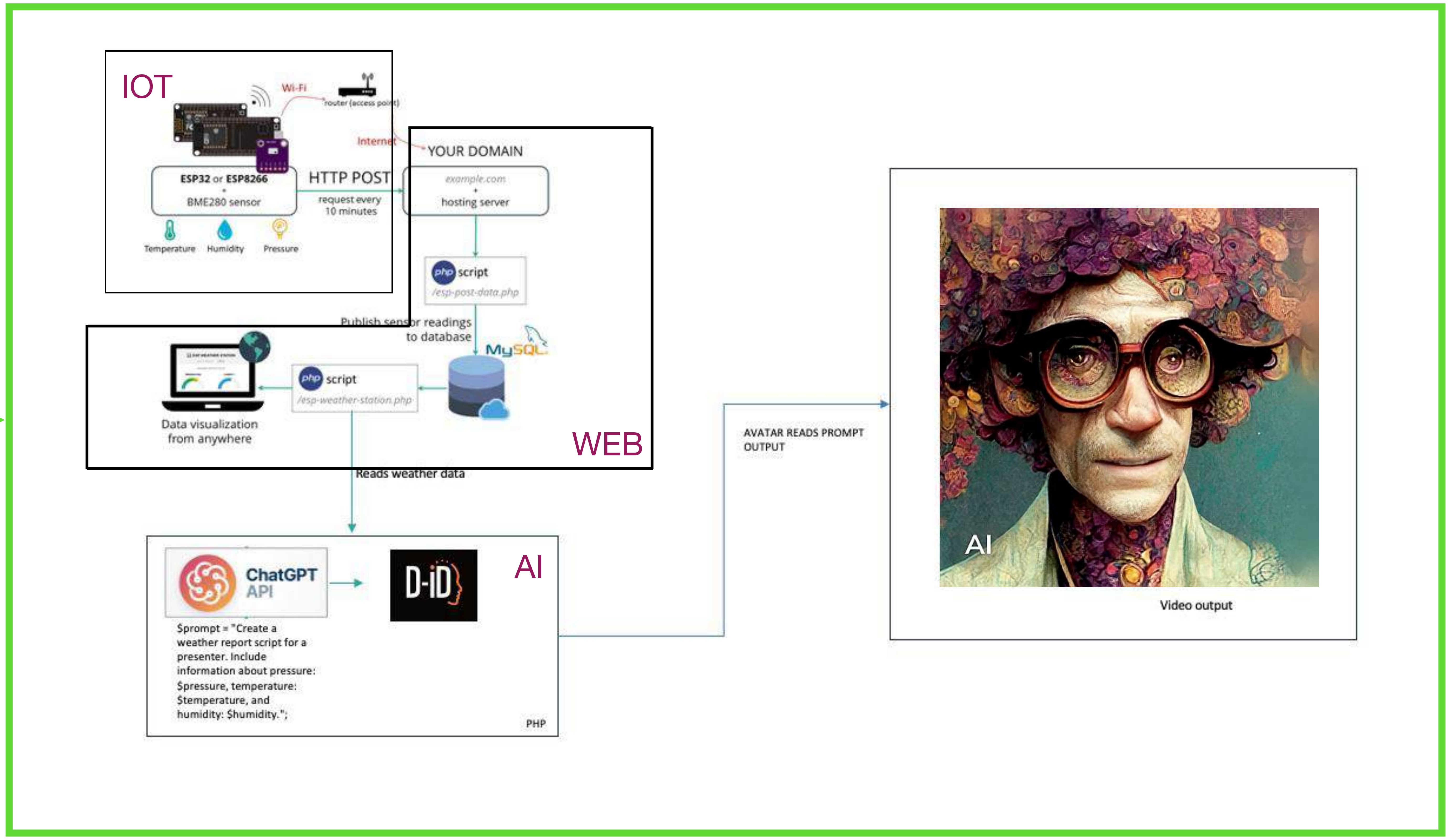
Pesquisa sobre a convergência: AI, Web e IoT para XR.

- Widget:
  - The New Weather Channel



Generative Weather Man





```
$prompt = "Create a
weather report script for a
presenter. Include
information about pressure:
$pressure, temperature:
$temperature, and
humidity: $humidity.";
```

IOT - WEB - AI

Recursos Utilizados

STEP1 - IOT: **DATA - Arduino IDE** (Integrated Development Environment.)

STEP2 - WEB: **PHP script** (Hypertext Preprocessor), **HTTPS** (Hypertext Transfer Protocol Secure) for data transport e **MySQL** (My Structured Query Language.) data storage

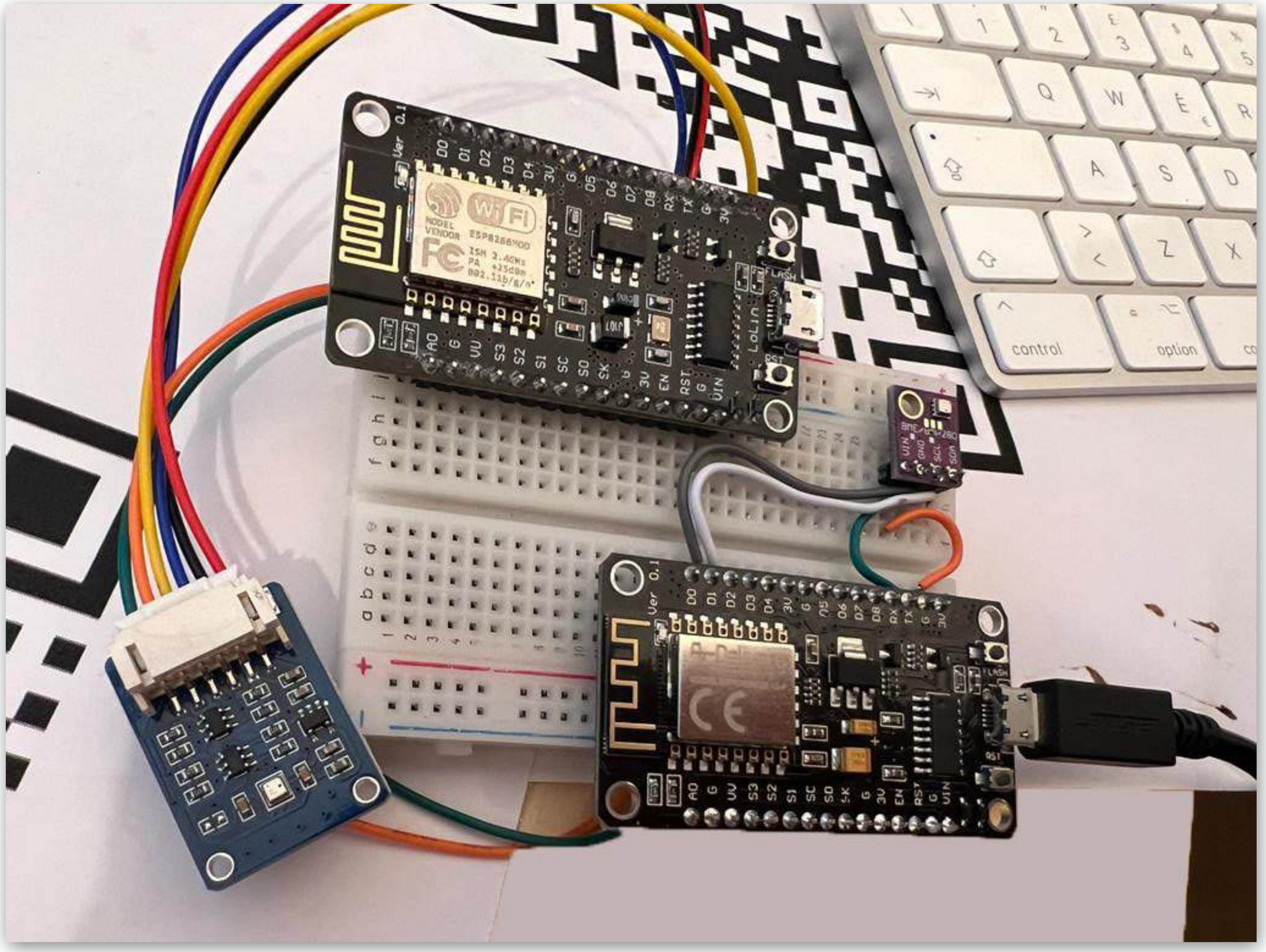
STEP3 - AI: **CHAT GPT API** (Chat Generative Pre-trained Transformer.) - interface cognitiva para a leitura de dados compreensíveis a humanos. **D-iD** para geracão de video. API (Application Programming Interface.)

New Weather Channel

by Matteo Moriconi
(with collaboration of Luiz Velho)

IOT: Temperature / Humidity Sensor

- NodeMCU ESP8266 ESP-12F WiFi
- Waveshare BME280 Environmental Sensor



- NodeMCU é uma plataforma IoT de código aberto e baixo custo;

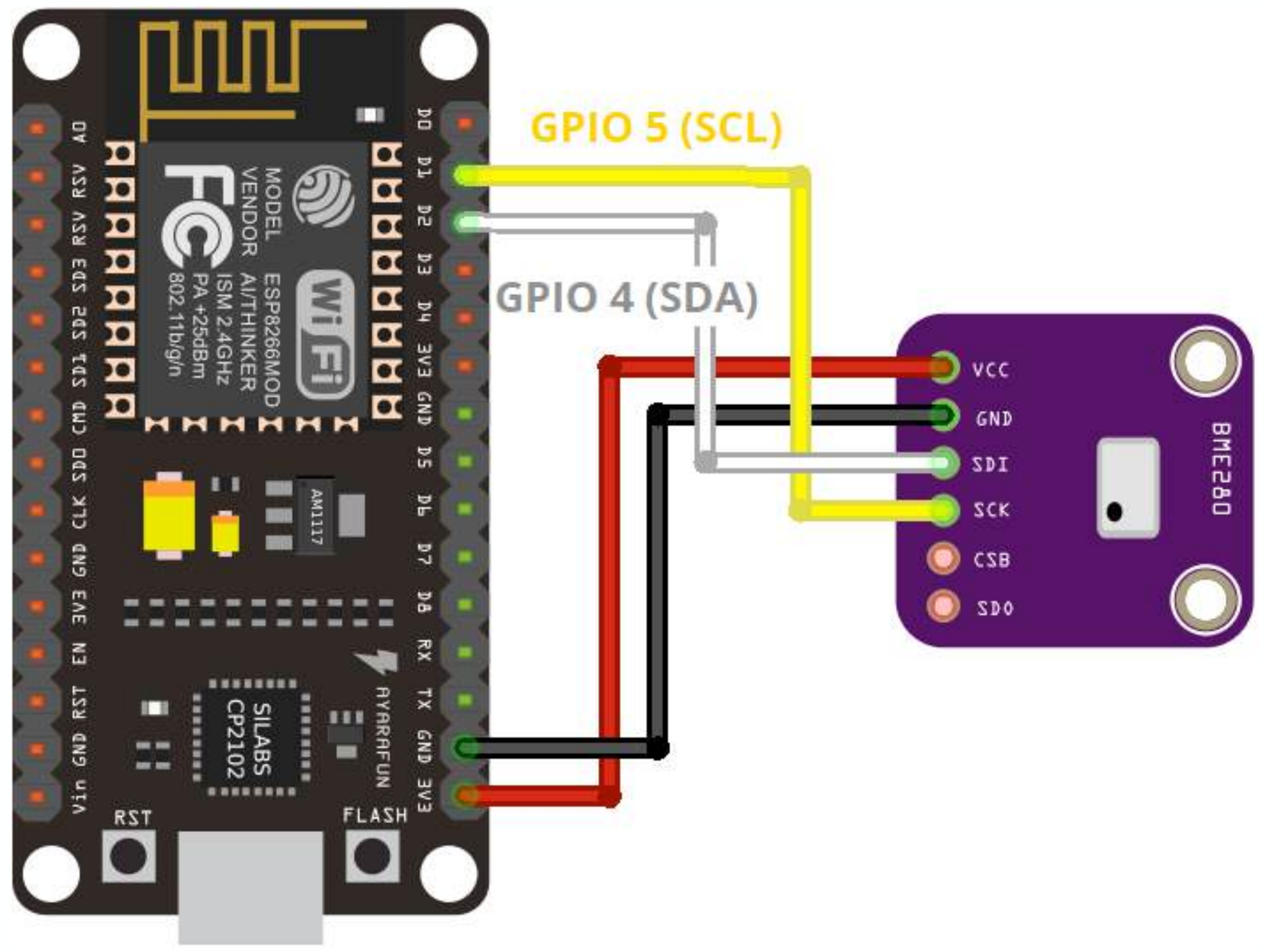
- NodeMCU é um firmware de código aberto para o qual estão disponíveis designs de placas de prototipagem de código aberto;

- O firmware utiliza a linguagem de script Lua. Ele é baseado no projeto eLua;

- NodeMCU fornece acesso ao GPIO (Entrada/ Saída de Propósito Geral) e uma tabela de mapeamento de pinos faz parte da documentação da API

IOT: Setup - BME280 wiring to ESP8266

- NodeMCU ESP8266 ESP-12F WiFi
- BME 280

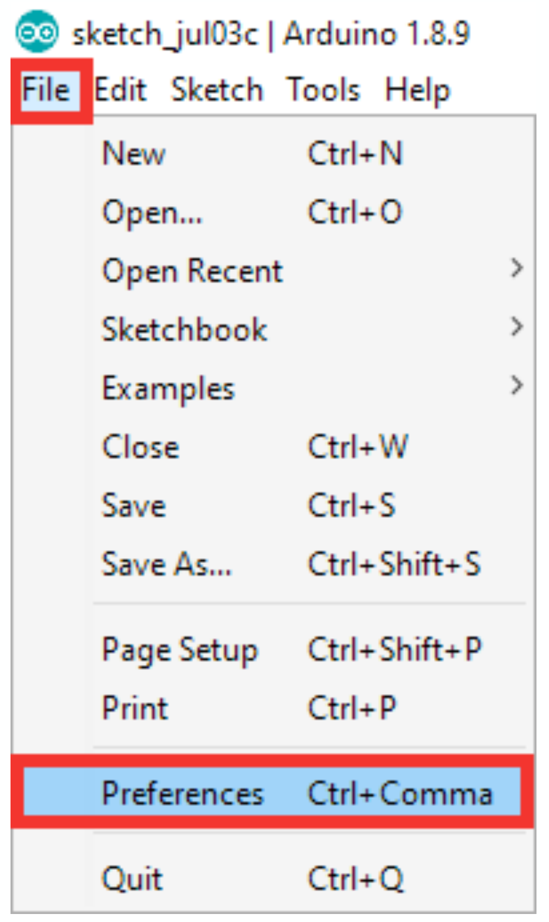


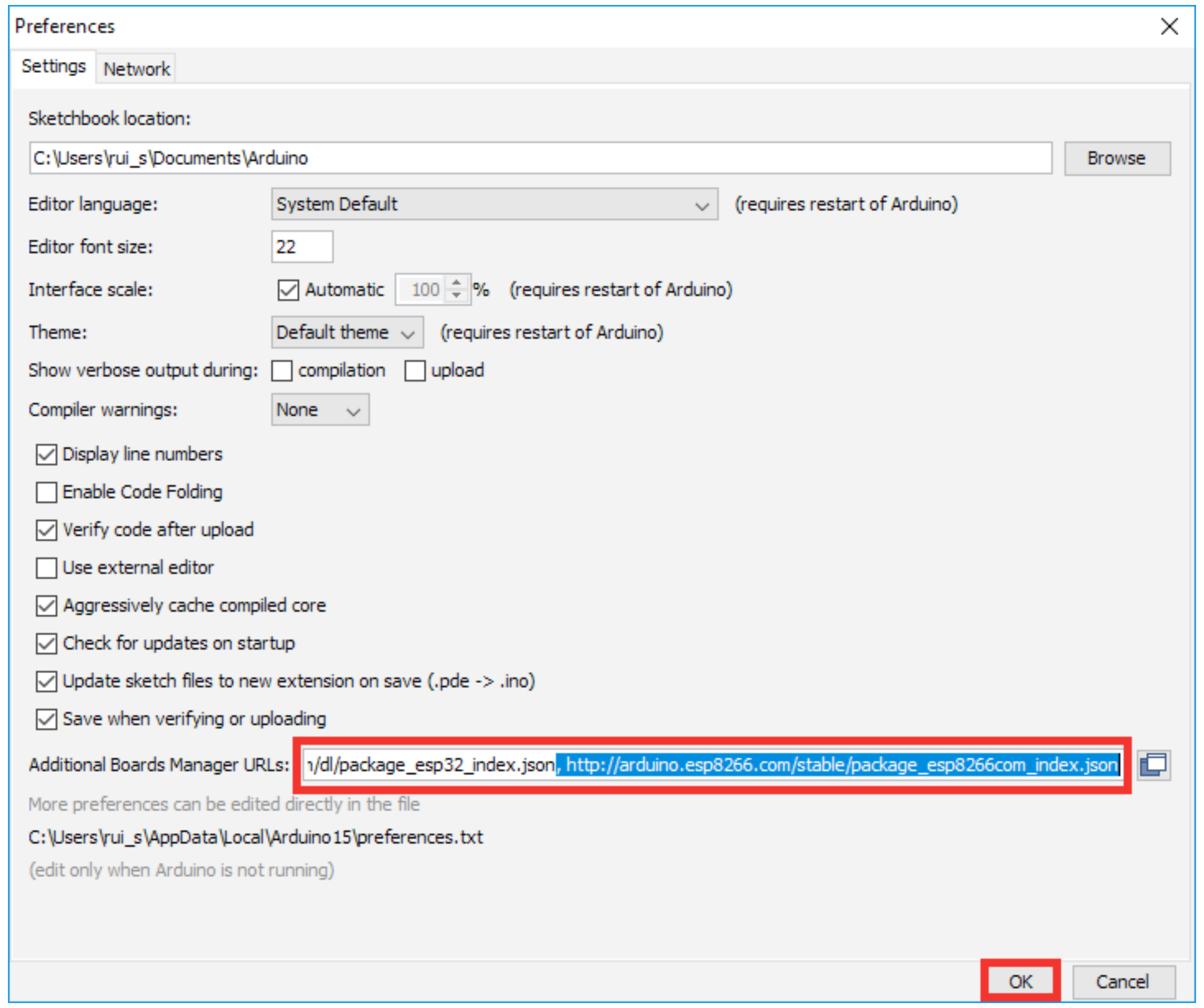
GPIO 5 (D1): SCL (SCK): Isso significa que o pino GPIO 5 está sendo usado para a função de Linha de Relógio Serial (SCL) ou Relógio Serial (SCK). SCL é tipicamente utilizado em protocolos de comunicação como I2C (Inter-Integrated Circuit) ou SPI (Serial Peripheral Interface) para sincronizar a transferência de dados entre dispositivos.

GPIO 4 (D2): SDA (SDI): Isso significa que o pino GPIO 4 está sendo usado para a função de Linha de Dados Serial (SDA) ou Entrada de Dados Serial (SDI). SDA também é comumente usada em protocolos de comunicação como I2C para transferir dados entre dispositivos.

IOT: Setup - Installing Libraries

- NodeMCU ESP8266 ESP-12F WiFi





Sketchbook location: C:\Users\rui_s\Documents\Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size: 22

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Theme: Default theme (requires restart of Arduino)

Show verbose output during: ☐ compilation ☐ upload

Compiler warnings: None

☑ Display line numbers

☐ Enable Code Folding

☑ Verify code after upload

☐ Use external editor

☑ Aggressively cache compiled core

☑ Check for updates on startup

☑ Update sketch files to new extension on save (.pde -> .ino)

☑ Save when verifying or uploading

Additional Boards Manager URLs: n/dl/package_esp32_index.json, http://arduino.esp8266.com/stable/package_esp8266com_index.json

More preferences can be edited directly in the file

C:\Users\rui_s\AppData\Local\Arduino15\preferences.txt

(edit only when Arduino is not running)

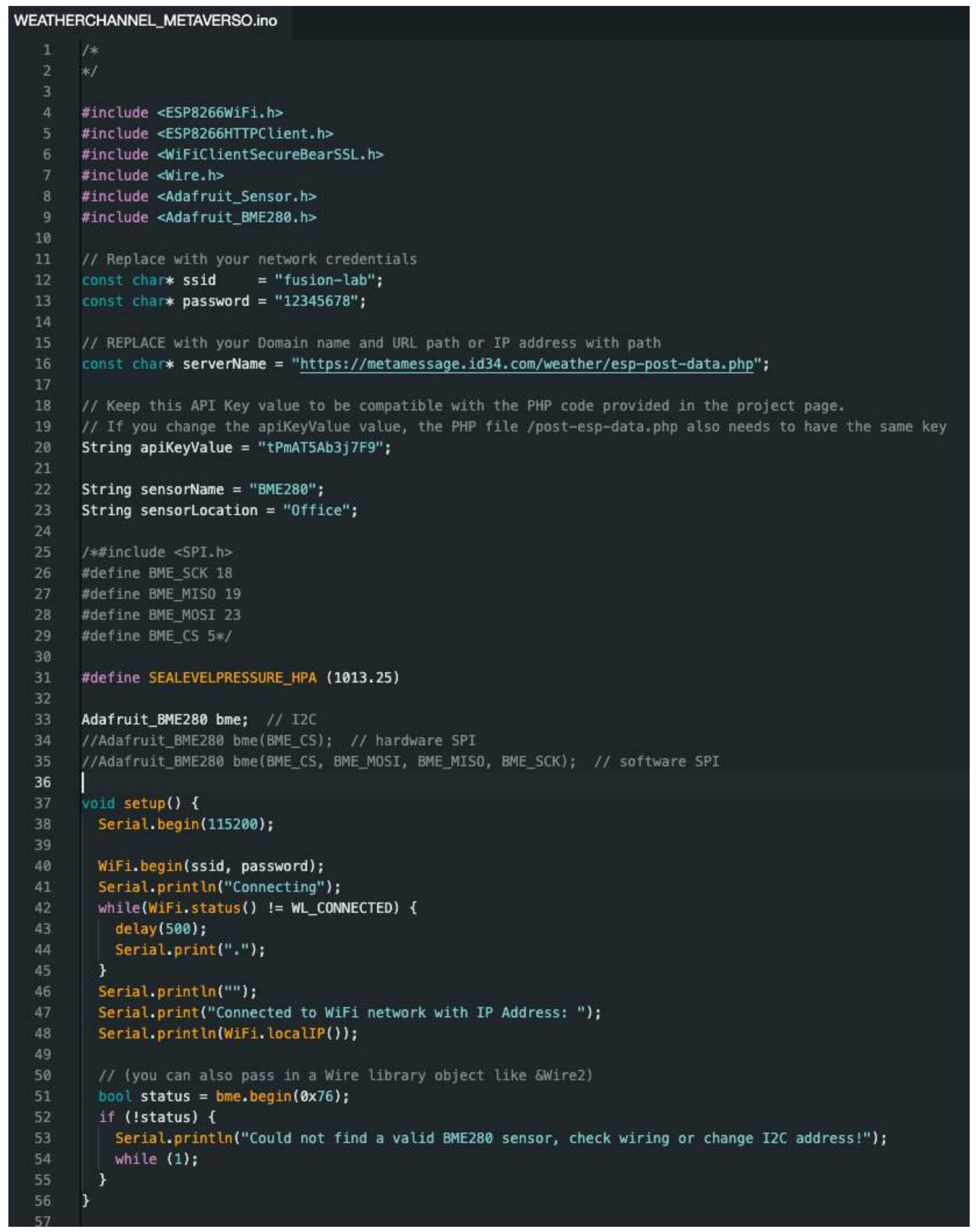
1 - Install ESP8266 Add-on in Arduino IDE

2 - Enter http://arduino.esp8266.com/stable/package_esp8266com_index.json into the “Additional Boards Manager URLs” field as shown in the figure below. Then, click the “OK” button

3 - Open the Boards Manager. Go to Tools > Board > Boards Manager… Search for ESP8266 and press install button for the “ESP8266 by ESP8266 Community

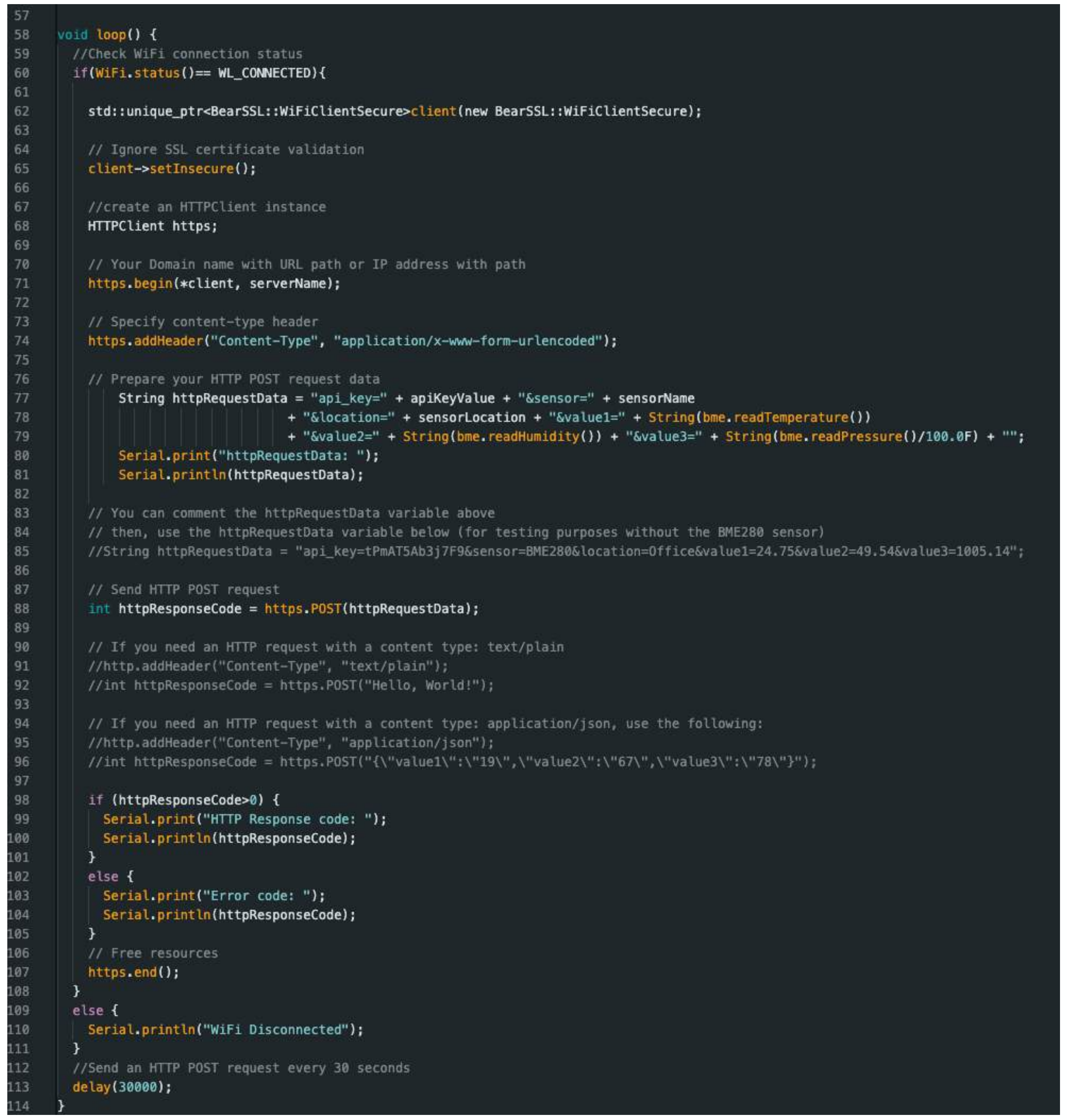
IOT: Setup - CODING

- NodeMCU ESP8266 ESP-12F WiFi



WEATHERCHANNEL_METAVERSO.ino

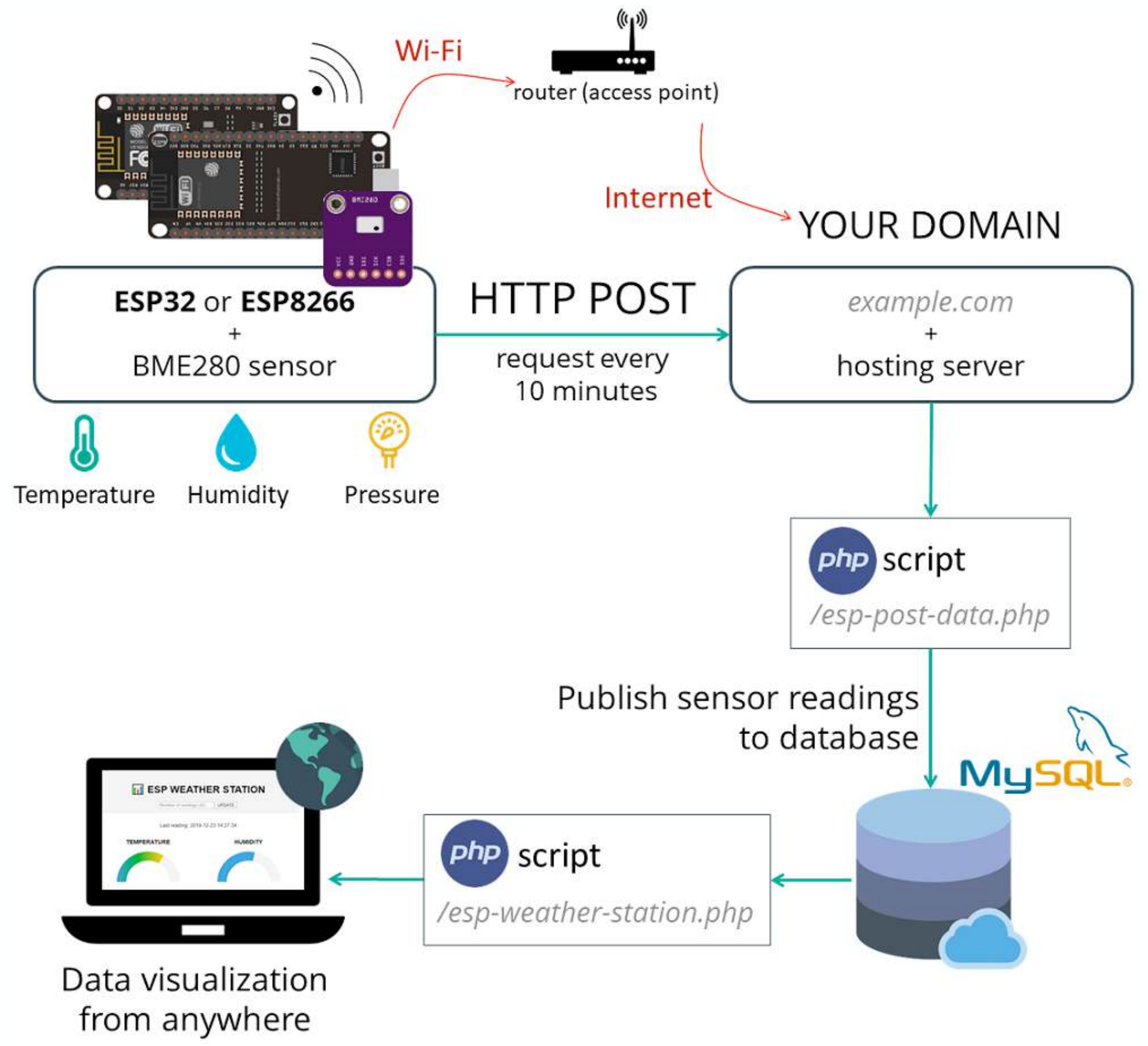
```cpp
/*
*/

#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClientSecureBearSSL.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>

// Replace with your network credentials
const char* ssid     = "fusion-lab";
const char* password = "12345678";

// REPLACE with your Domain name and URL path or IP address with path
const char* serverName = "https://metamessage.id34.com/weather/esp-post-data.php";

// Keep this API Key value to be compatible with the PHP code provided in the project page.
// If you change the apiKeyValue value, the PHP file /post-esp-data.php also needs to have the same key
String apiKeyValue = "tPmAT5Ab3j7F9";

String sensorName = "BME280";
String sensorLocation = "Office";

/*#include <SPI.h>
#define BME_SCK 18
#define BME_MISO 19
#define BME_MOSI 23
#define BME_CS 5*/

#define SEALEVELPRESSURE_HPA (1013.25)

Adafruit_BME280 bme;  // I2C
//Adafruit_BME280 bme(BME_CS);  // hardware SPI
//Adafruit_BME280 bme(BME_CS, BME_MOSI, BME_MISO, BME_SCK);  // software SPI

void setup() {
  Serial.begin(115200);

  WiFi.begin(ssid, password);
  Serial.println("Connecting");
  while(WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to WiFi network with IP Address: ");
  Serial.println(WiFi.localIP());

  // (you can also pass in a Wire library object like &Wire2)
  bool status = bme.begin(0x76);
  if (!status) {
    Serial.println("Could not find a valid BME280 sensor, check wiring or change I2C address!");
    while (1);
  }
}

```



```cpp

void loop() {
  //Check WiFi connection status
  if(WiFi.status()== WL_CONNECTED){

    std::unique_ptr<BearSSL::WiFiClientSecure>client(new BearSSL::WiFiClientSecure);

    // Ignore SSL certificate validation
    client->setInsecure();

    //create an HTTPClient instance
    HTTPClient https;

    // Your Domain name with URL path or IP address with path
    https.begin(*client, serverName);

    // Specify content-type header
    https.addHeader("Content-Type", "application/x-www-form-urlencoded");

    // Prepare your HTTP POST request data
      String httpRequestData = "api_key=" + apiKeyValue + "&sensor=" + sensorName
                            + "&location=" + sensorLocation + "&value1=" + String(bme.readTemperature())
                            + "&value2=" + String(bme.readHumidity()) + "&value3=" + String(bme.readPressure()/100.0F) + "";
      Serial.print("httpRequestData: ");
      Serial.println(httpRequestData);

    // You can comment the httpRequestData variable above
    // then, use the httpRequestData variable below (for testing purposes without the BME280 sensor)
    //String httpRequestData = "api_key=tPmAT5Ab3j7F9&sensor=BME280&location=Office&value1=24.75&value2=49.54&value3=1005.14";

    // Send HTTP POST request
    int httpResponseCode = https.POST(httpRequestData);

    // If you need an HTTP request with a content type: text/plain
    //http.addHeader("Content-Type", "text/plain");
    //int httpResponseCode = https.POST("Hello, World!");

    // If you need an HTTP request with a content type: application/json, use the following:
    //http.addHeader("Content-Type", "application/json");
    //int httpResponseCode = https.POST("{\"value1\":\"19\",\"value2\":\"67\",\"value3\":\"78\"}");

    if (httpResponseCode>0) {
      Serial.print("HTTP Response code: ");
      Serial.println(httpResponseCode);
    }
    else {
      Serial.print("Error code: ");
      Serial.println(httpResponseCode);
    }
    // Free resources
    https.end();
  }
  else {
    Serial.println("WiFi Disconnected");
  }
  //Send an HTTP POST request every 30 seconds
  delay(30000);
}
```

Como o código funciona:
- Importe todas as bibliotecas necessárias para o funcionamento;
- Defina variáveis que você pode querer alterar (apiKeyValue, sensorName, sensorLocation);
- O apiKeyValue é apenas uma string aleatória que você pode modificar. É usado por motivos de segurança, para que apenas quem conhece sua chave de API possa publicar dados em seu banco de dados;
- Inicialize a comunicação serial para fins de depuração;
- Estabeleça uma conexão Wi-Fi com seu roteador;
- Inicialize o BME280 para obter leituras;
- Inicialize um cliente Wi-Fi seguro.
- Então, no loop(), é onde você faz a solicitação POST HTTP a cada 10 minutos com as leituras mais recentes do BME280:

WEB: Environment Readings



1 - Preparing Your MySQL Database

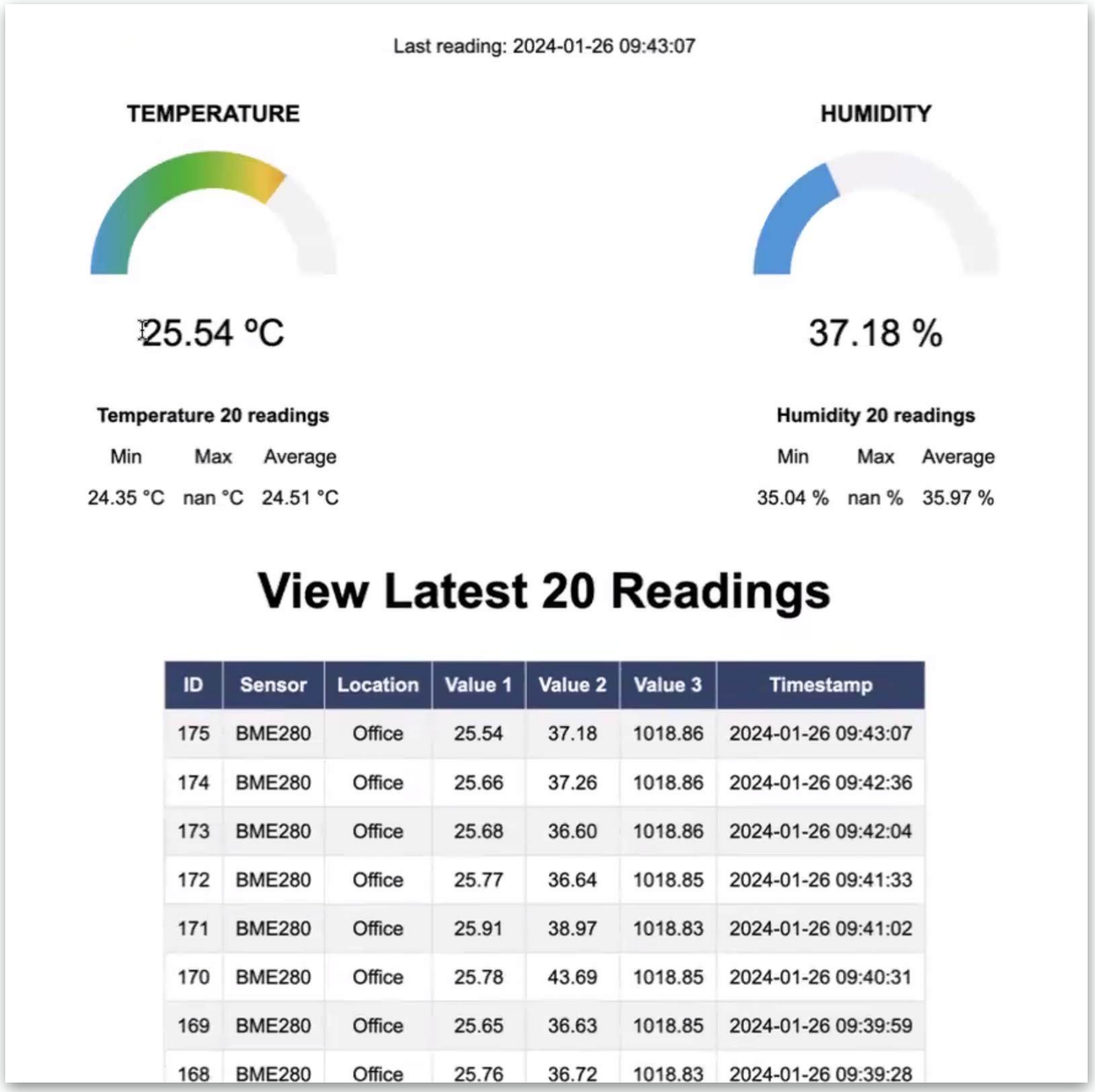
2 - PHP Script HTTP POST – Receive and Insert Data in MySQL Database (esp-post-data.php)

3 - PHP Script for Database Functions (esp-database.php)

Referência: https://randomnerdtutorials.com/cloud-weather-station-esp32-esp8266/

WEB: Environment Readings

- WebView



Last reading: 2024-01-26 09:43:07

TEMPERATURE

25.54 °C

Temperature 20 readings

Min	Max	Average
24.35 °C	nan °C	24.51 °C

HUMIDITY

37.18 %

Humidity 20 readings

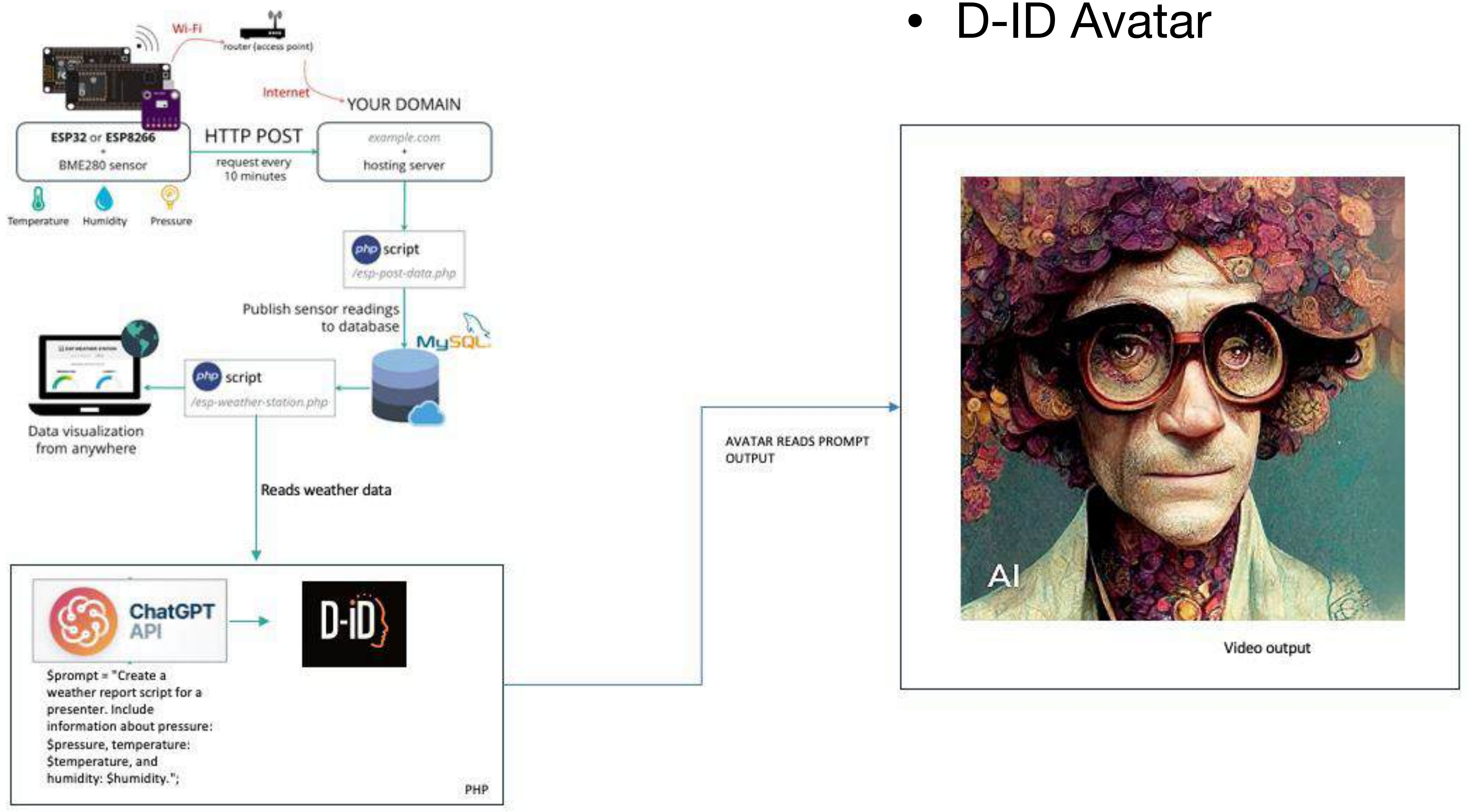
Min	Max	Average
35.04 %	nan %	35.97 %

View Latest 20 Readings

ID	Sensor	Location	Value 1	Value 2	Value 3	Timestamp
175	BME280	Office	25.54	37.18	1018.86	2024-01-26 09:43:07
174	BME280	Office	25.66	37.26	1018.86	2024-01-26 09:42:36
173	BME280	Office	25.68	36.60	1018.86	2024-01-26 09:42:04
172	BME280	Office	25.77	36.64	1018.85	2024-01-26 09:41:33
171	BME280	Office	25.91	38.97	1018.83	2024-01-26 09:41:02
170	BME280	Office	25.78	43.69	1018.85	2024-01-26 09:40:31
169	BME280	Office	25.65	36.63	1018.85	2024-01-26 09:39:59
168	BME280	Office	25.76	36.72	1018.83	2024-01-26 09:39:28

STEP3 - AI-WEB-IOT: Generative Weather Man

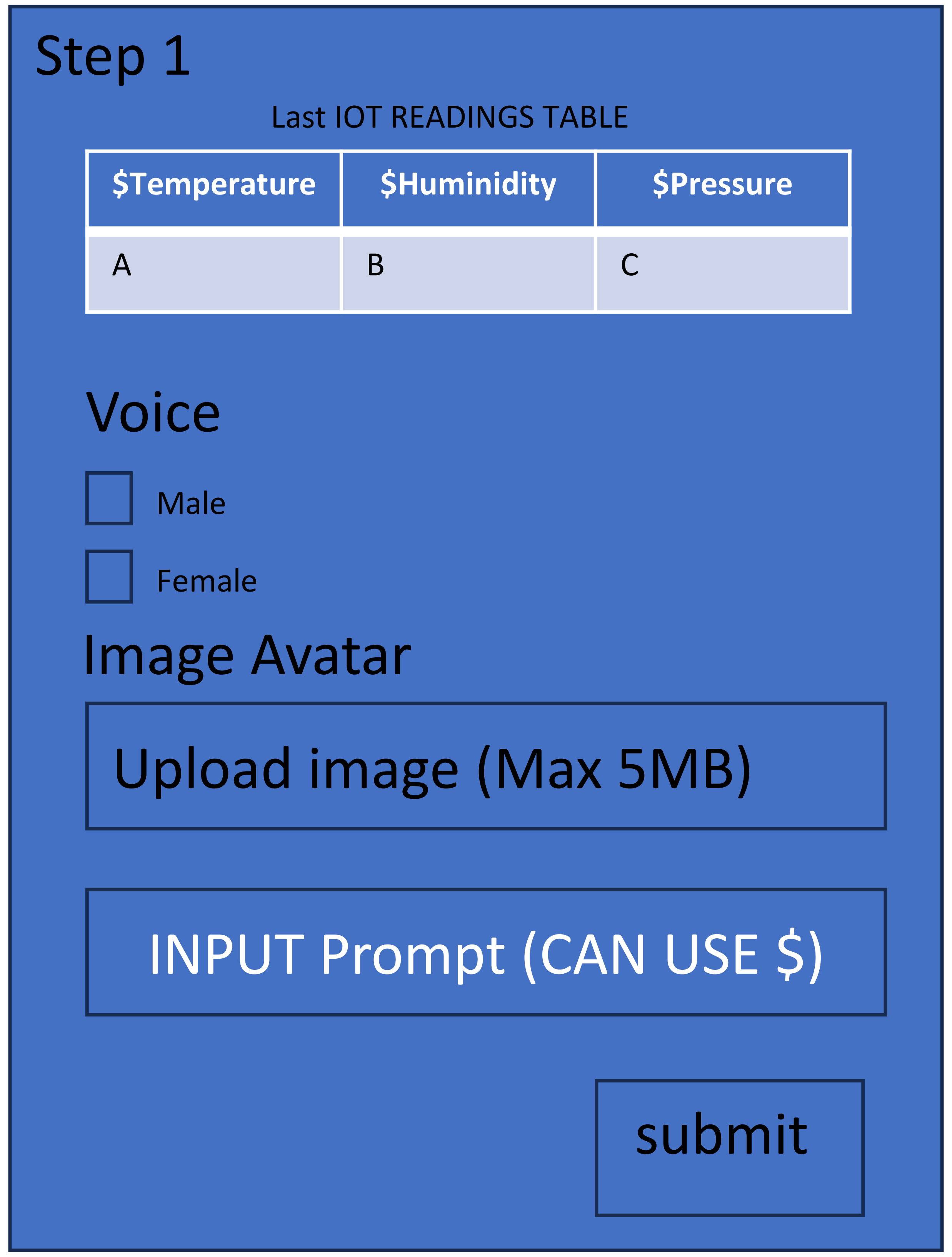
- D-ID Avatar



```
$prompt = "Create a
weather report script for a
presenter. Include
information about pressure:
$pressure, temperature:
$temperature, and
humidity: $humidity.";
```

IoT/GPT Proof of Concept

Weather Channel – IOT – GPT - TALKING AVATAR
proof of concept



Step 1

Last IOT READINGS TABLE

$Temperature	$Huminidity	$Pressure
A	B	C

Voice

☐ Male

☐ Female

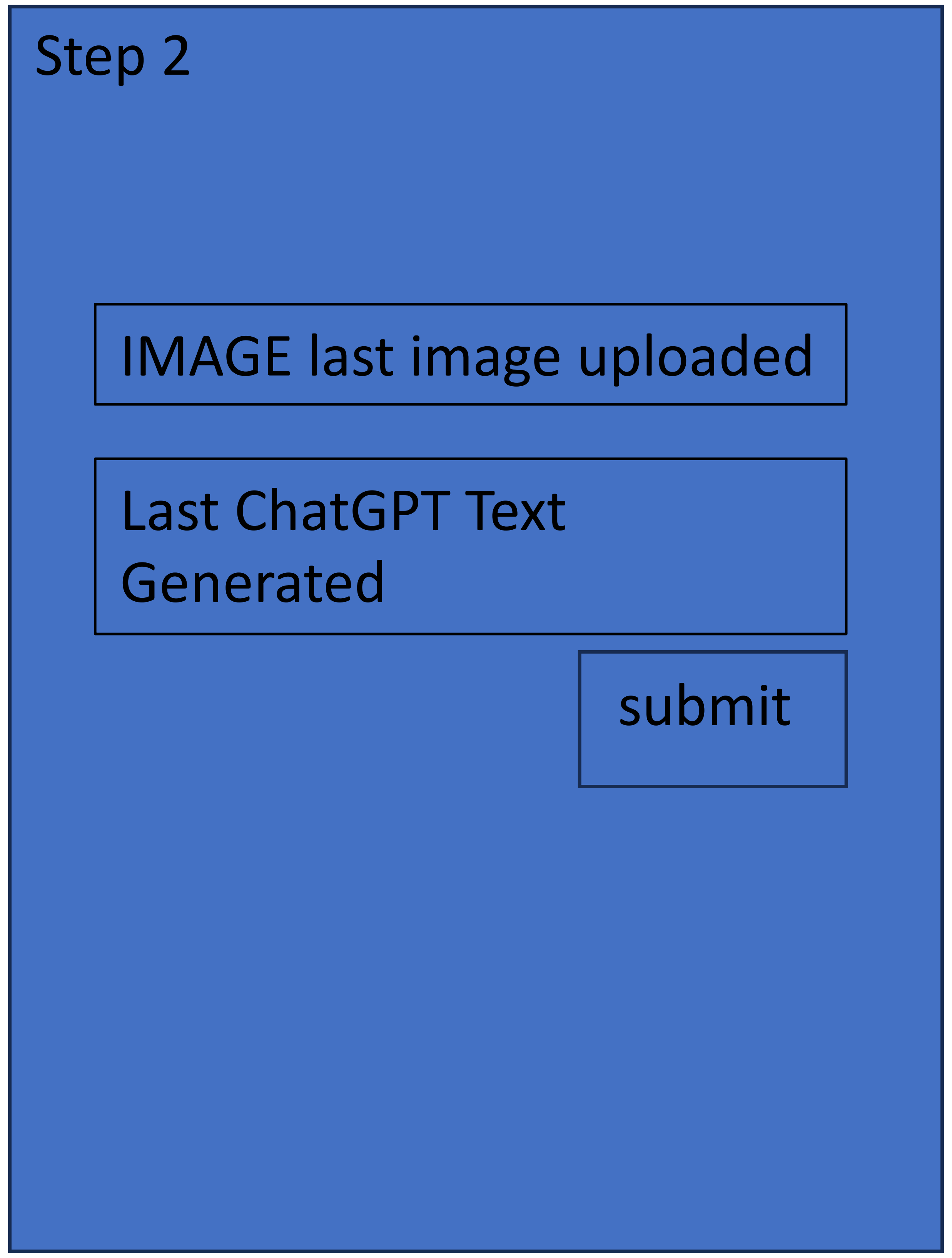
Image Avatar

Upload image (Max 5MB)

INPUT Prompt (CAN USE $)

submit

Store in the database
and uses ChatGPT API



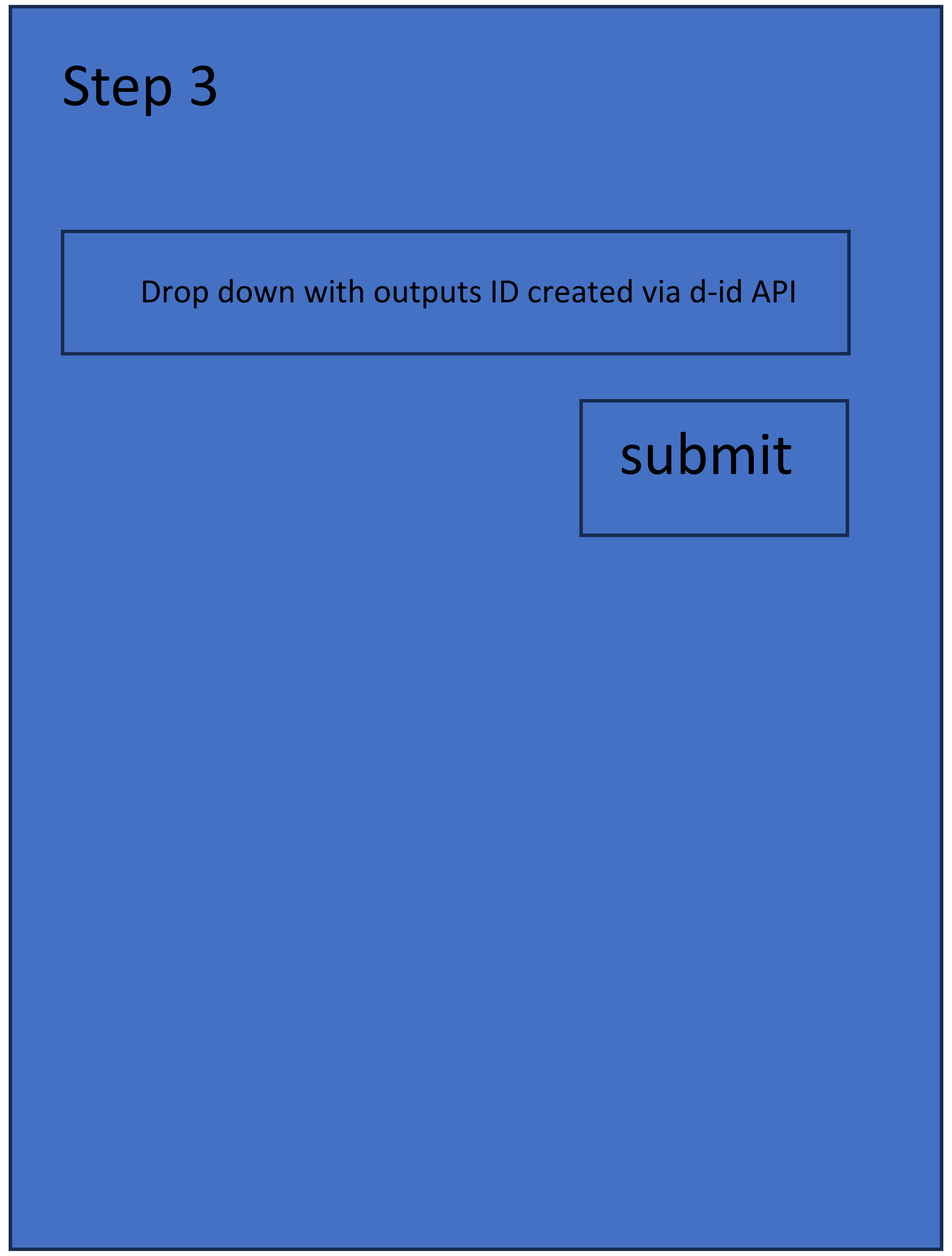
Step 2

IMAGE last image uploaded

Last ChatGPT Text Generated

submit

Submit Data to generate avatar to
https://www.d-id.com/ API



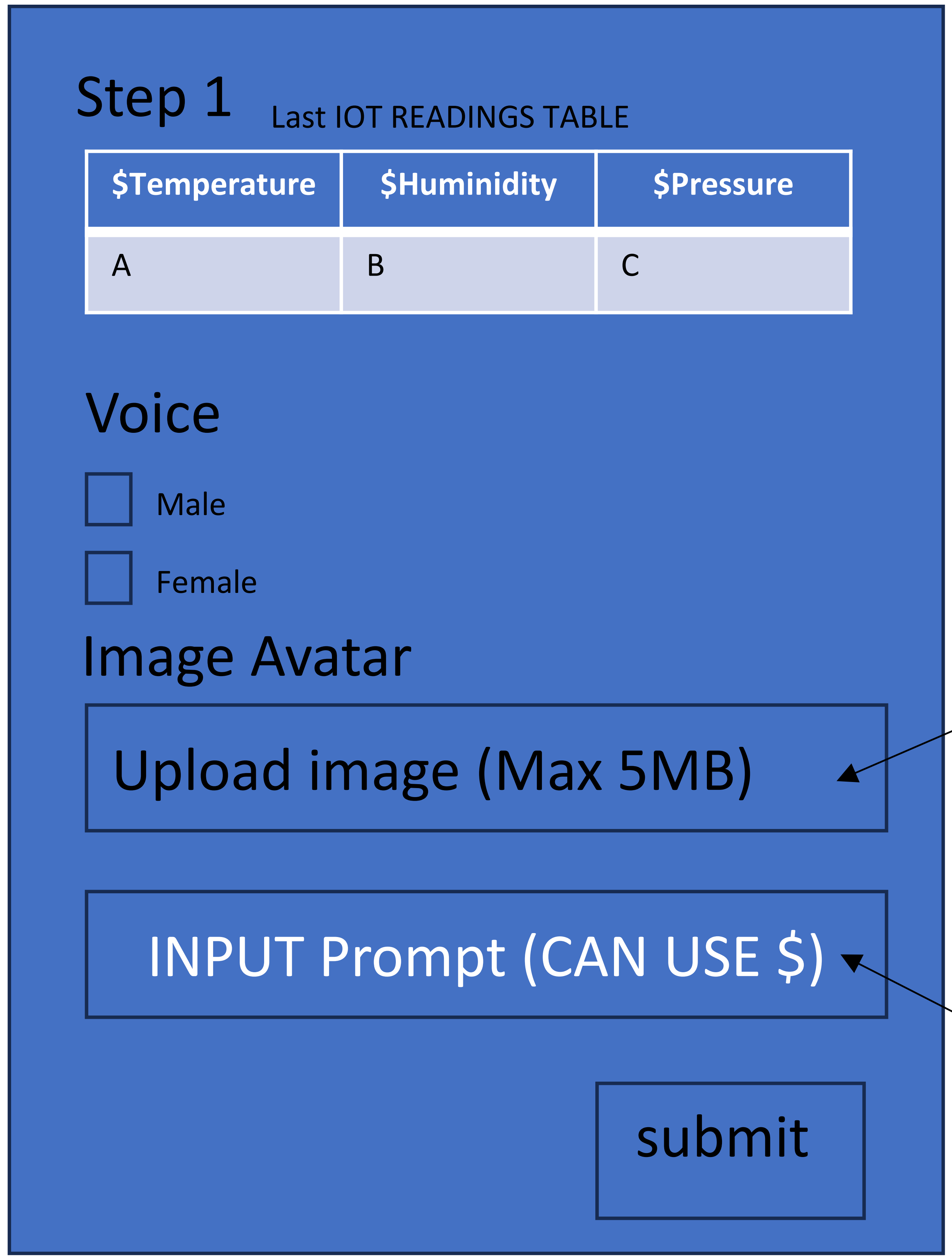
Step 3

Drop down with outputs ID created via d-id API

submit

When the user selects one of the ids
generated by d-id the page will display the
talking avatar

Step 1



Step 1 Last IOT READINGS TABLE

$Temperature	$Huminidity	$Pressure
A	B	C

Voice

- [ ] Male
- [ ] Female

Image Avatar

Upload image (Max 5MB)

INPUT Prompt (CAN USE $)

submit

Image source

$prompt = "Create a weather report text with lots of humor for a presenter. the name of the presneter Noah the Apocaplipic weather presenter Include information about temperature:$temperature degrees, humidity: $humidity%";

Step 2

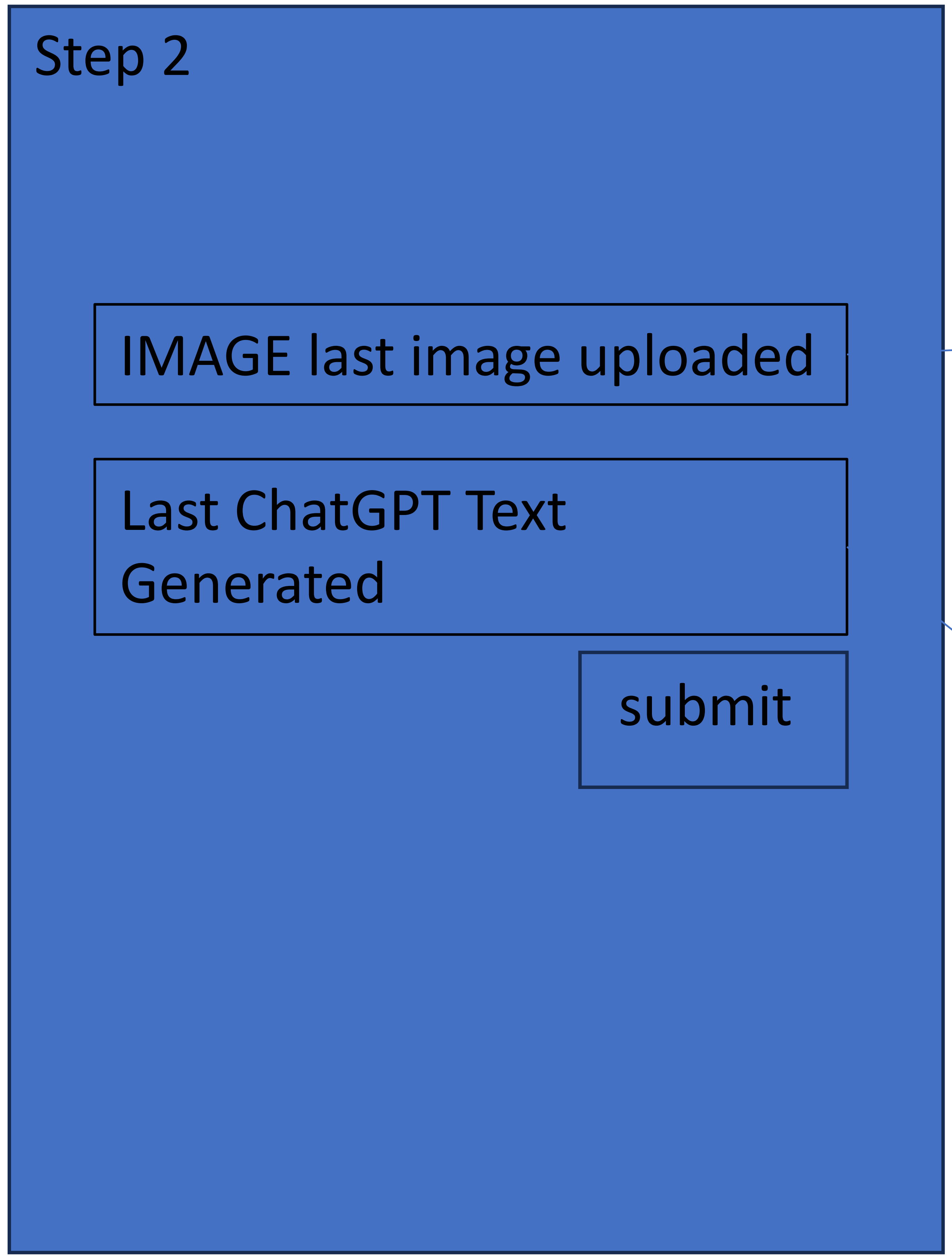
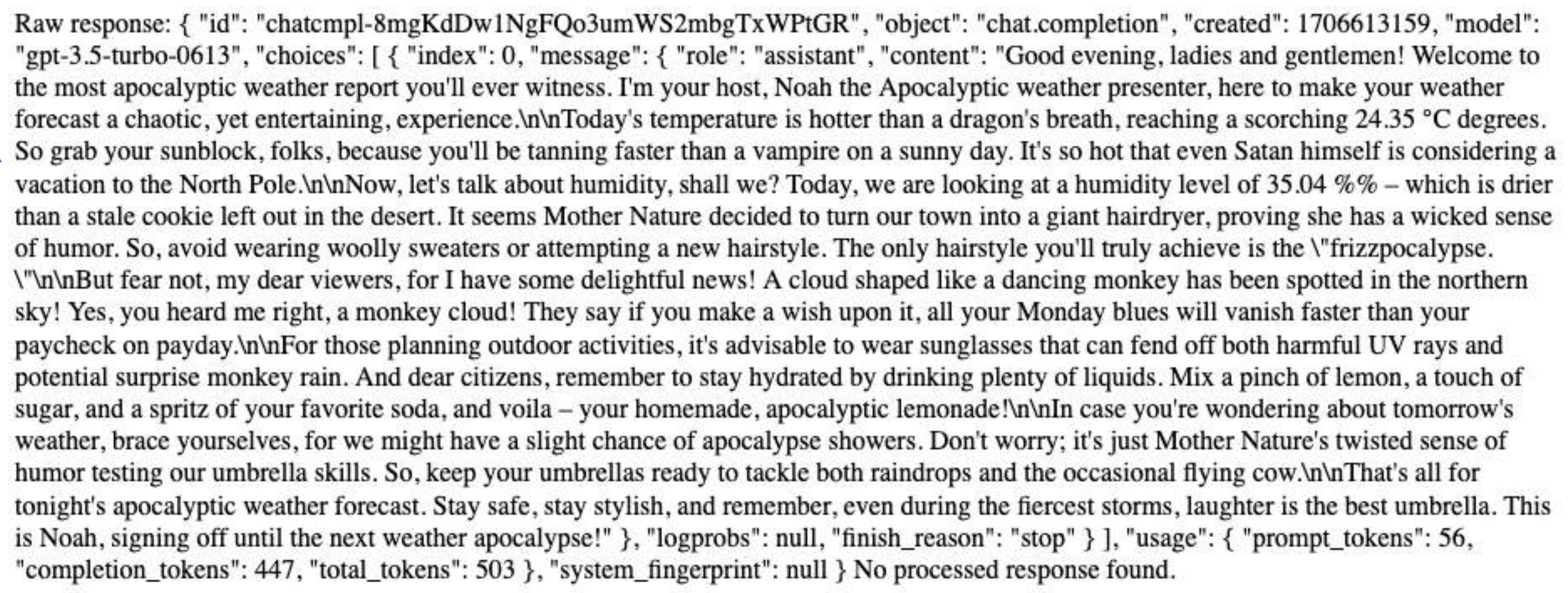


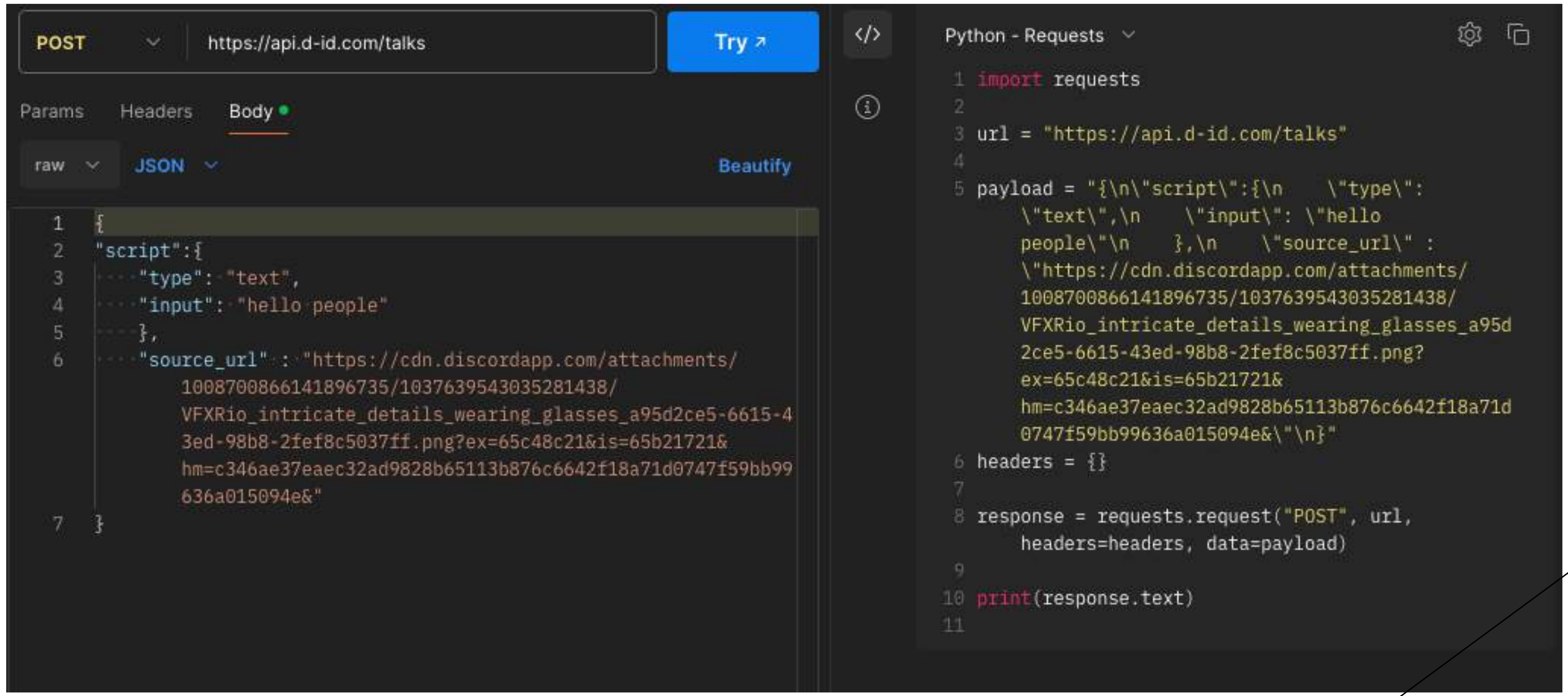
Image source


Raw response: { "id": "chatcmpl-8mgKdDw1NgFQo3umWS2mbgTxWPtGR", "object": "chat.completion", "created": 1706613159, "model": "gpt-3.5-turbo-0613", "choices": [{ "index": 0, "message": { "role": "assistant", "content": "Good evening, ladies and gentlemen! Welcome to the most apocalyptic weather report you'll ever witness. I'm your host, Noah the Apocalyptic weather presenter, here to make your weather forecast a chaotic, yet entertaining, experience.\n\nToday's temperature is hotter than a dragon's breath, reaching a scorching 24.35 °C degrees. So grab your sunblock, folks, because you'll be tanning faster than a vampire on a sunny day. It's so hot that even Satan himself is considering a vacation to the North Pole.\n\nNow, let's talk about humidity, shall we? Today, we are looking at a humidity level of 35.04 %% – which is drier than a stale cookie left out in the desert. It seems Mother Nature decided to turn our town into a giant hairdryer, proving she has a wicked sense of humor. So, avoid wearing woolly sweaters or attempting a new hairstyle. The only hairstyle you'll truly achieve is the \"frizzpocalypse. \"\n\nBut fear not, my dear viewers, for I have some delightful news! A cloud shaped like a dancing monkey has been spotted in the northern sky! Yes, you heard me right, a monkey cloud! They say if you make a wish upon it, all your Monday blues will vanish faster than your paycheck on payday.\n\nFor those planning outdoor activities, it's advisable to wear sunglasses that can fend off both harmful UV rays and potential surprise monkey rain. And dear citizens, remember to stay hydrated by drinking plenty of liquids. Mix a pinch of lemon, a touch of sugar, and a spritz of your favorite soda, and voila – your homemade, apocalyptic lemonade!\n\nIn case you're wondering about tomorrow's weather, brace yourselves, for we might have a slight chance of apocalypse showers. Don't worry; it's just Mother Nature's twisted sense of humor testing our umbrella skills. So, keep your umbrellas ready to tackle both raindrops and the occasional flying cow.\n\nThat's all for tonight's apocalyptic weather forecast. Stay safe, stay stylish, and remember, even during the fiercest storms, laughter is the best umbrella. This is Noah, signing off until the next weather apocalypse!" }, "logprobs": null, "finish_reason": "stop" }], "usage": { "prompt_tokens": 56, "completion_tokens": 447, "total_tokens": 503 }, "system_fingerprint": null } No processed response found.

Data stored in the database when submit generate a post To API D-ID

Step 3.1

API POST
https://www.d-id.com/

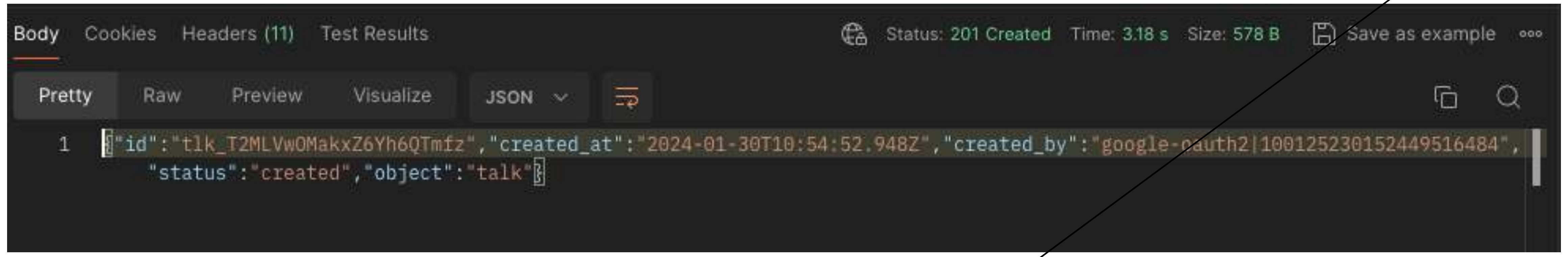


```
{
"script":{
    "type": "text",
    "input": "hello people"
    },
    "source_url" : "https://cdn.discordapp.com/attachments/
        1008700866141896735/1037639543035281438/
        VFXRio_intricate_details_wearing_glasses_a95d2ce5-6615-4
        3ed-98b8-2fef8c5037ff.png?ex=65c48c21&is=65b21721&
        hm=c346ae37eaec32ad9828b65113b876c6642f18a71d0747f59bb99
        636a015094e&"
}
```

```python
import requests

url = "https://api.d-id.com/talks"

payload = "{\n\"script\":{\n    \"type\":
    \"text\",\n    \"input\": \"hello
    people\"\n    },\n    \"source_url\" :
    \"https://cdn.discordapp.com/attachments/
    1008700866141896735/1037639543035281438/
    VFXRio_intricate_details_wearing_glasses_a95d
    2ce5-6615-43ed-98b8-2fef8c5037ff.png?
    ex=65c48c21&is=65b21721&
    hm=c346ae37eaec32ad9828b65113b876c6642f18a71d
    0747f59bb99636a015094e&\"\n}"
headers = {}

response = requests.request("POST", url,
    headers=headers, data=payload)

print(response.text)
```

Referência: https://www.postman.com/

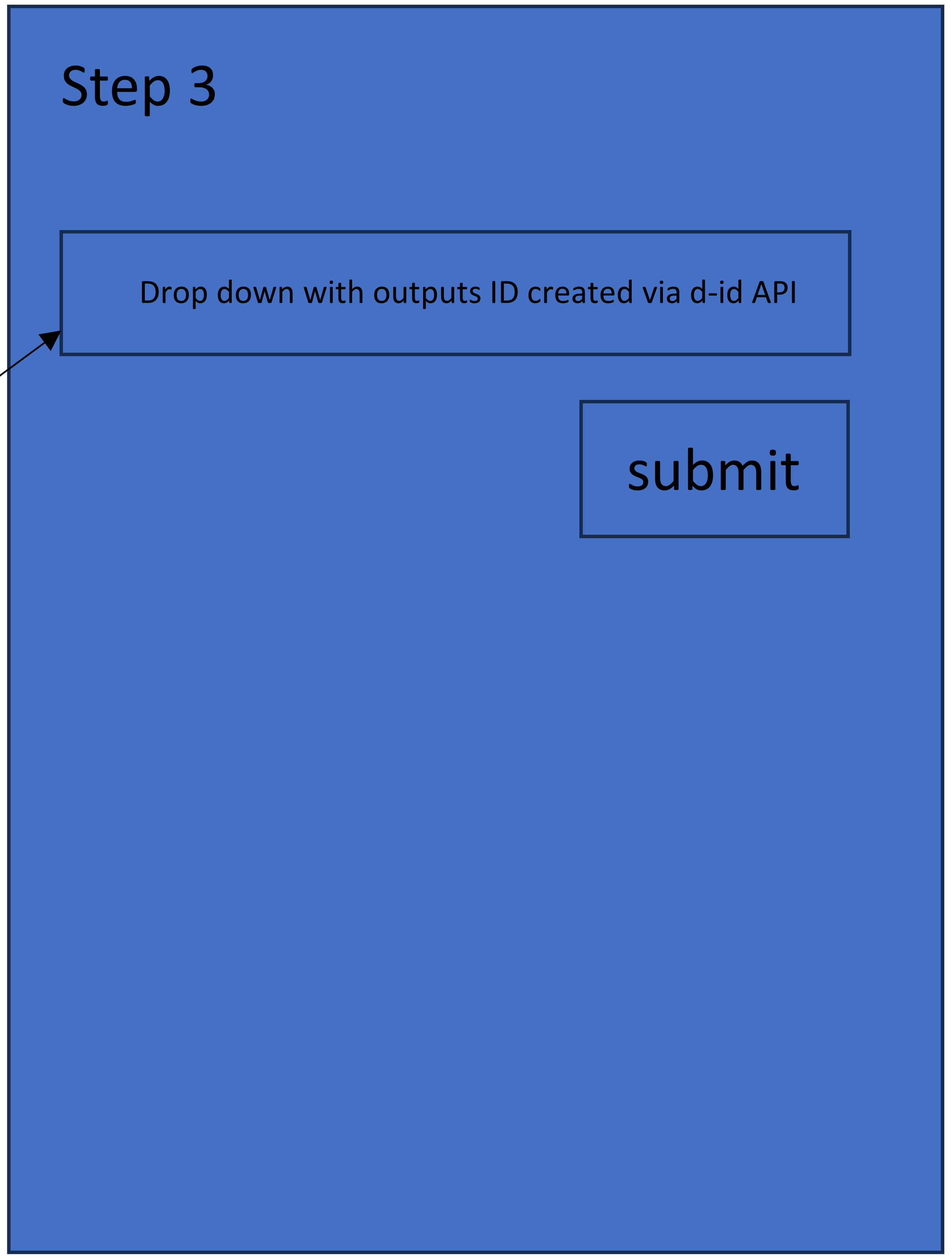
API POST: RESPONSE



Status: 201 Created Time: 3.18 s Size: 578 B

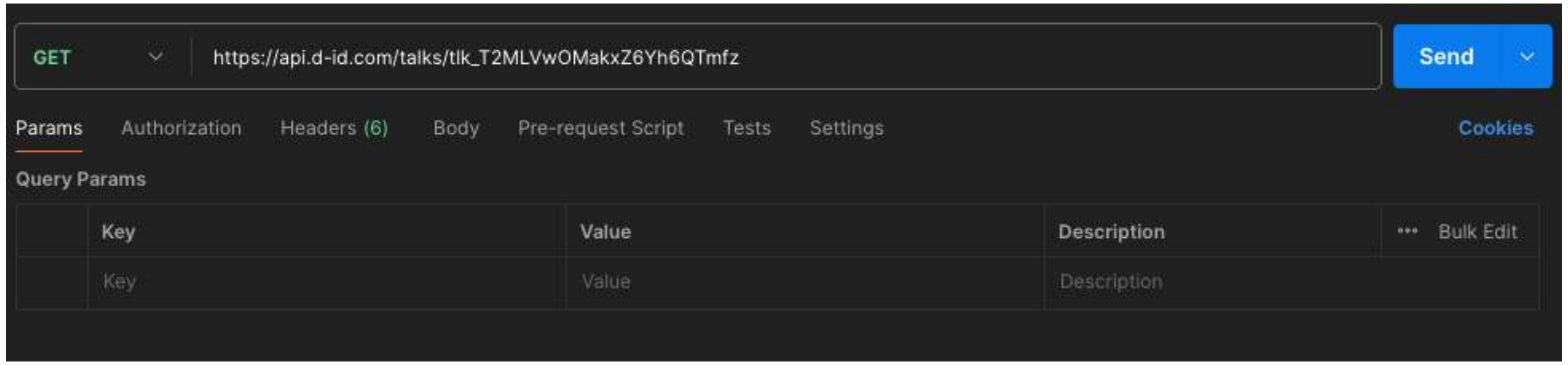
```
{"id":"tlk_T2MLVwOMakxZ6Yh6QTmfz","created_at":"2024-01-30T10:54:52.948Z","created_by":"google-oauth2|100125230152449516484",
    "status":"created","object":"talk"}
```

"id":"tlk_T2MLVwOMakxZ6Yh6QTmfz"



Step 3.2

API GET
https://www.d-id.com



GET https://api.d-id.com/talks/tlk_T2MLVwOMakxZ6Yh6QTmfz Send

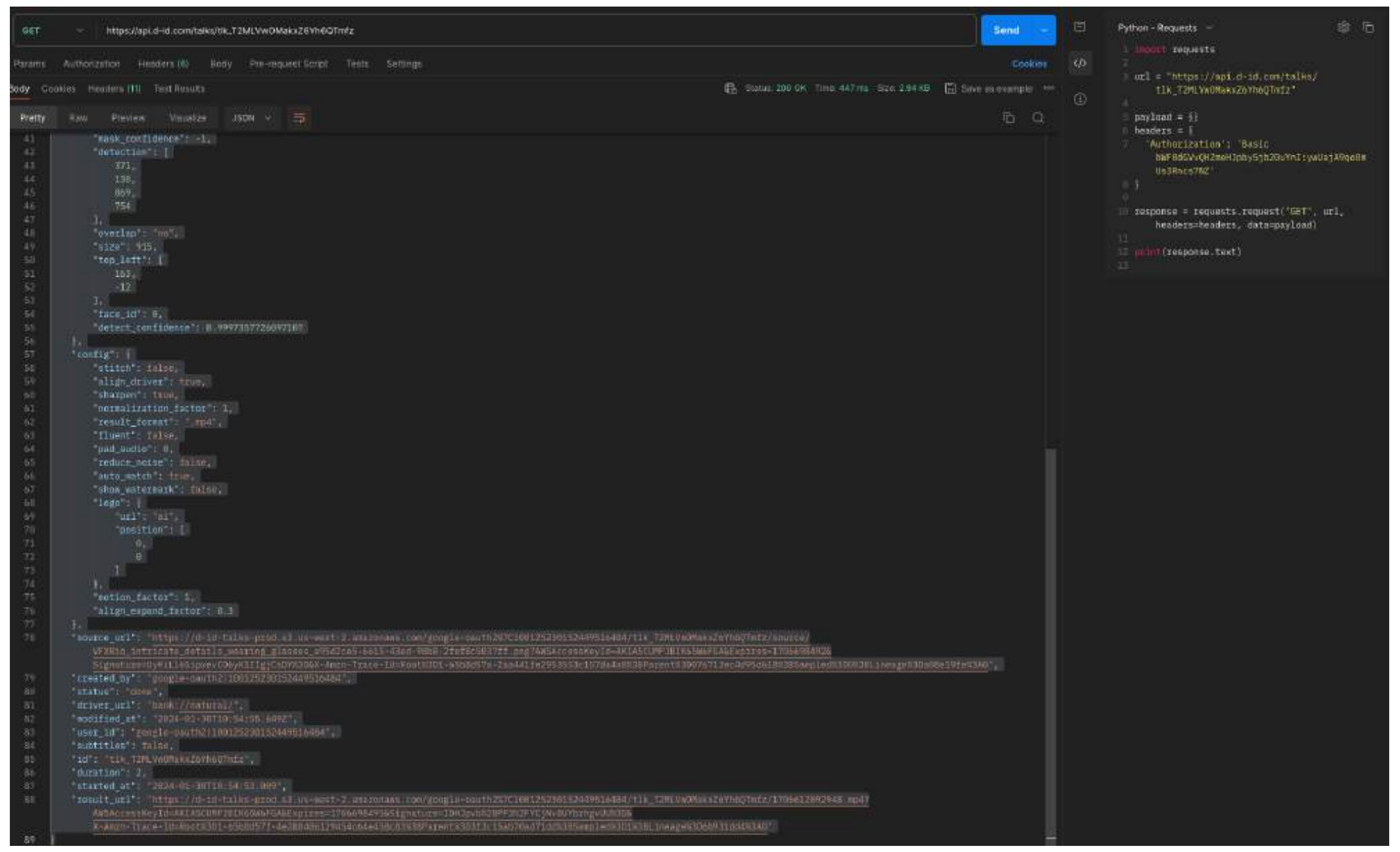
Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

Step 3

API GET: RESPONSE



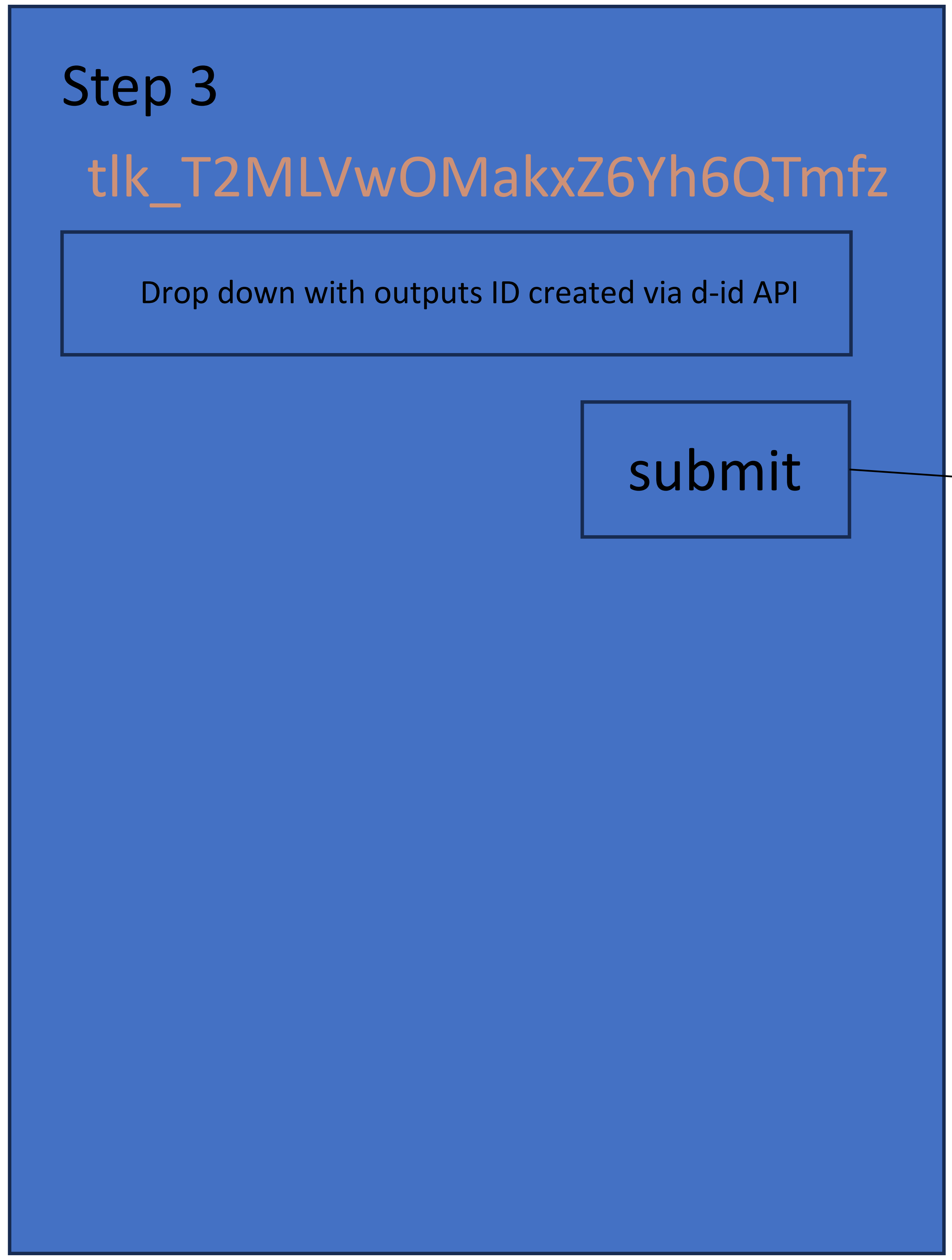
GET https://api.d-id.com/talks/tlk_T2MLVwOMakxZ6Yh6QTmfz Send

Status: 200 OK Time: 447 ms Size: 2.94 KB

```
        "mask_confidence": -1,
        "detection": [
            371,
            138,
            869,
            754
        ],
        "overlap": "no",
        "size": 915,
        "top_left": [
            163,
            -12
        ],
        "face_id": 0,
        "detect_confidence": 0.9997357726097107
    },
    "config": {
        "stitch": false,
        "align_driver": true,
        "sharpen": true,
        "normalization_factor": 1,
        "result_format": ".mp4",
        "fluent": false,
        "pad_audio": 0,
        "reduce_noise": false,
        "auto_match": true,
        "show_watermark": false,
        "logo": {
            "url": "ai",
            "position": [
                0,
                0
            ]
        },
        "motion_factor": 1,
        "align_expand_factor": 0.3
    },
    "source_url": "https://d-id-talks-prod.s3.us-west-2.amazonaws.com/google-oauth2%7C100125230152449516484/tlk_T2MLVwOMakxZ6Yh6QTmfz/source/[illegible]",
    "created_by": "google-oauth2|100125230152449516484",
    "status": "done",
    "driver_url": "bank://natural/",
    "modified_at": "2024-01-30T10:54:55.609Z",
    "user_id": "google-oauth2|100125230152449516484",
    "subtitles": false,
    "id": "tlk_T2MLVwOMakxZ6Yh6QTmfz",
    "duration": 2,
    "started_at": "2024-01-30T10:54:53.009",
    "result_url": "https://d-id-talks-prod.s3.us-west-2.amazonaws.com/google-oauth2%7C100125230152449516484/tlk_T2MLVwOMakxZ6Yh6QTmfz/1706612092948.mp4?[illegible]"
}
```

"result_url": "https://d-id-talks-prod.s3.us-west-2.amazonaws.com/google-oauth2%7C100125230152449516484/tlk_T2MLVwOMakxZ6Yh6QTmfz/1706612092948.mp4?AWSAccessKeyId=AKIA5CUMPJBIK65W6FGA&Expires=1706698495&Signature=IDHJpvb%2BPP3%2FYCjNv8UYbrhgvUU%3D&X-Amzn-Trace-Id=Root%3D1-65b8d57f-4e288406129454c64e458c83%3BParent%3D3f3c15ab70ad71dd%3BSampled%3D1%3BLineage%3D6b931dd4%3A0"

Step 3.3



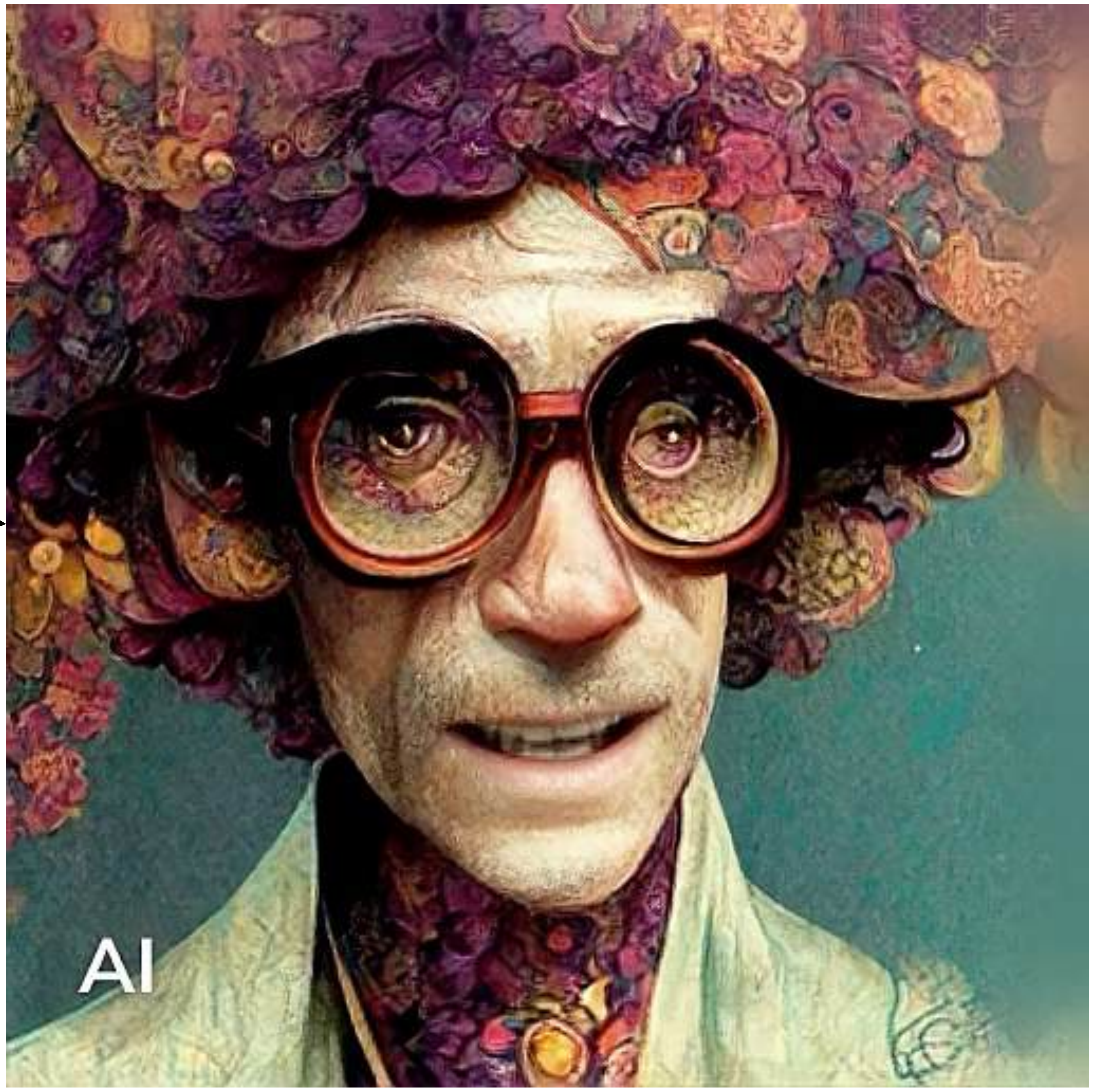
Step 3

tlk_T2MLVwOMakxZ6Yh6QTmfz

Drop down with outputs ID created via d-id API

submit

When submit visualize generated video

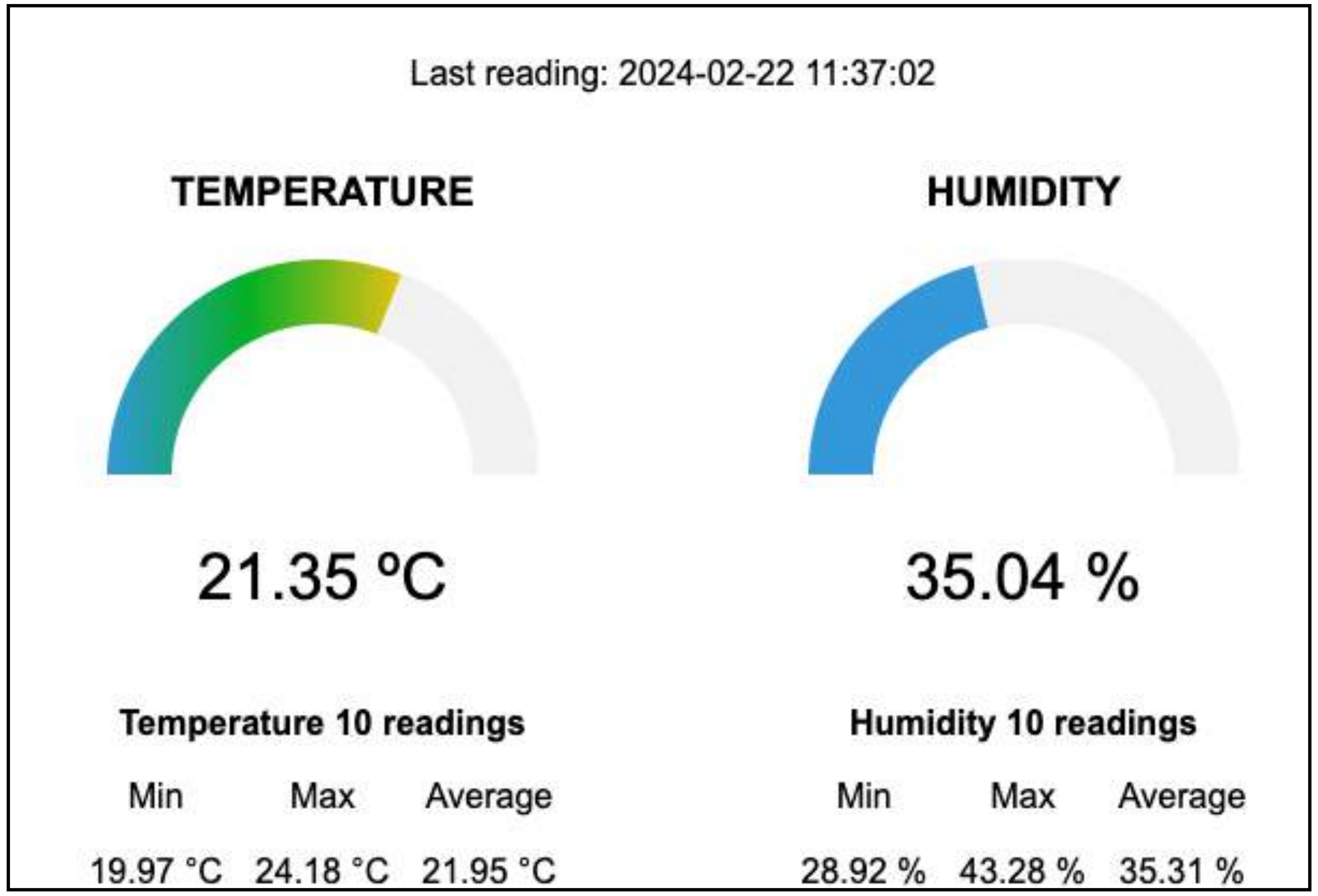


Video TALKING AVATAR

Celestia Borealis

reading the data from the IOT Weather Station

- D-ID


Last reading: 2024-02-22 11:37:02

TEMPERATURE

21.35 ºC

Temperature 10 readings

Min	Max	Average
19.97 °C	24.18 °C	21.95 °C

HUMIDITY

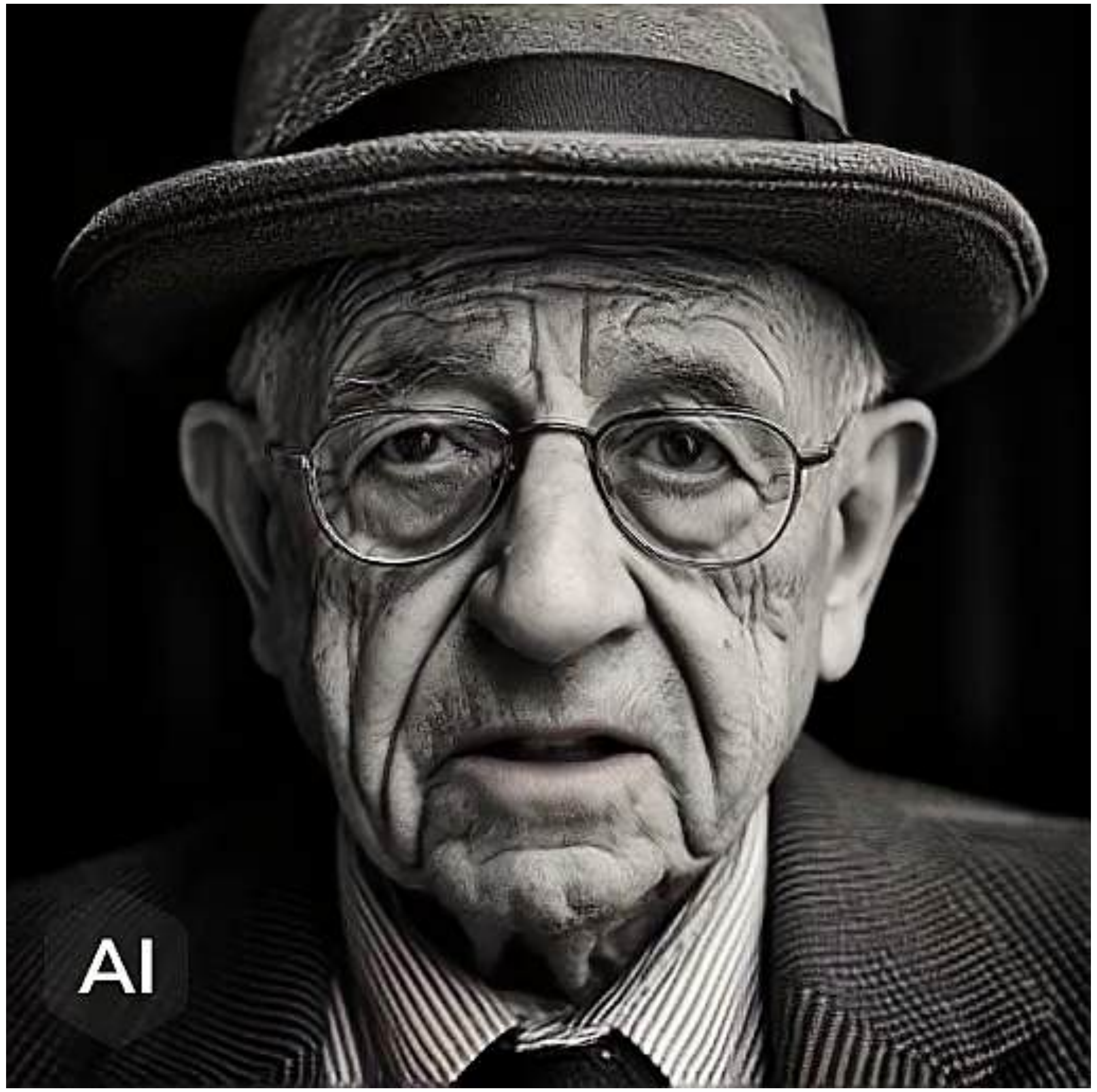
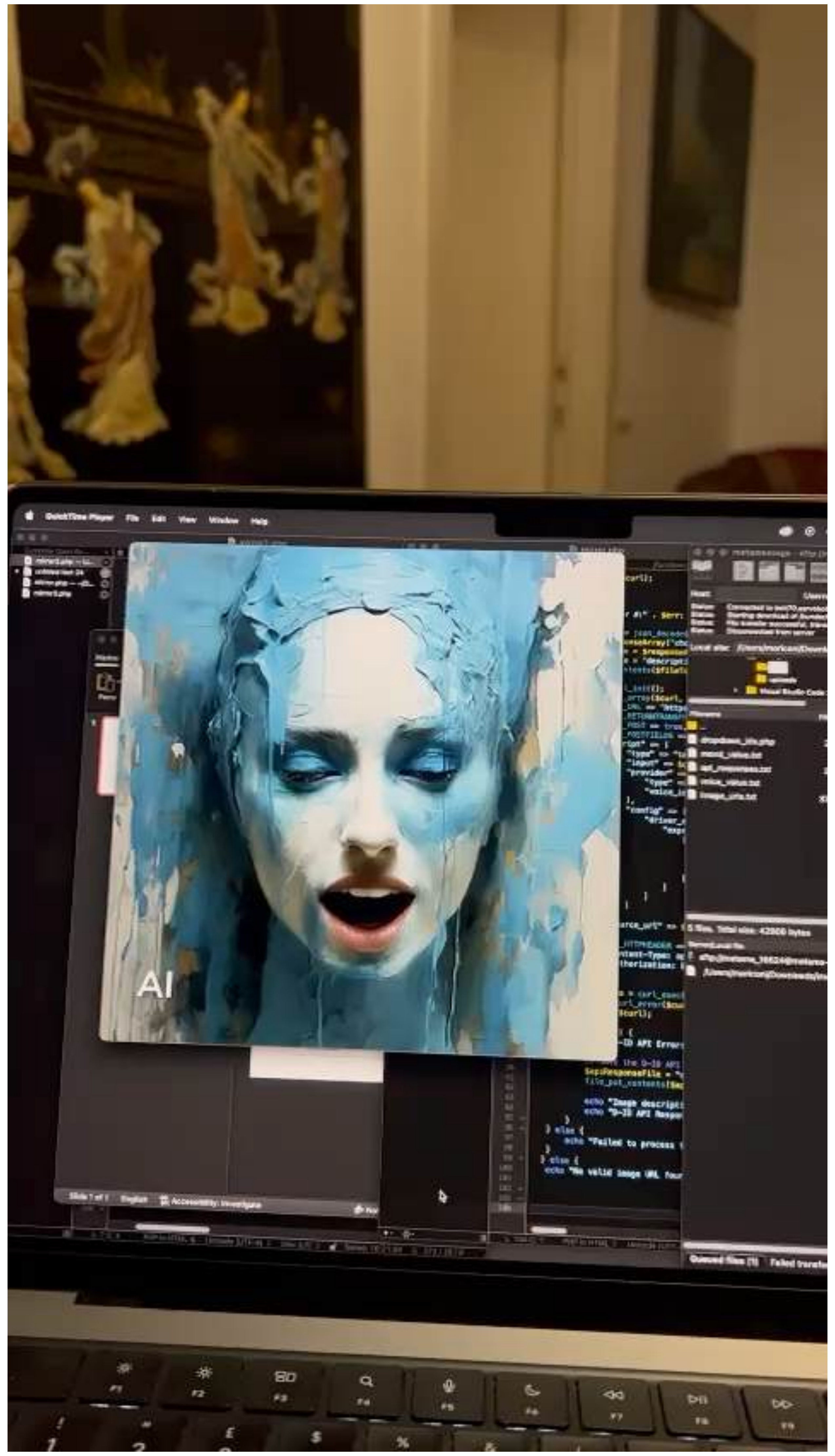
35.04 %

Humidity 10 readings

Min	Max	Average
28.92 %	43.28 %	35.31 %

Referências

- https://docs.d-id.com/reference/get-started
- https://web.postman.co
- https://platform.openai.com/docs/overview
- https://www.arduino.cc/en/software
- https://randomnerdtutorials.com/

AI

AI

AI

REAL-TIME DEMO