Descripción

Un manual donde aprenderá el uso del sistema de manera más técnica y precisa.

Manual técnico Majinsack

“Persigue todos tus deseos.”

**ÍNDICE**

1. Objetivos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4
   1. Contenido\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

1. Alcance\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

1. Requisitos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4
   1. Hardware \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4
   2. Software\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4

1. Herramientas utilizadas y recomendadas\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

1. Ensambre\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

5.1 NodeMCU\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5

5.2 Módulo GPS Neo-6m\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6

* 1. Sensor de Agua\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6

6.Análisis\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

6.1 Introducción\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

6.2 Propósito\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

6.3 Alcance\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

6.4 Definiciones y Acrónimos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

7.Desarrollo\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

7.1 Definición de librerías\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7

7.2 Definición de variables\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8

7.3 Función Setup \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8

7.4 Función Loop \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_9

7.4.1 Latitud y Longitud \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_9

7.4.2 Fecha \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_9

7.4.3 Tiempo\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_10

7.4.4 Conexión HTTP \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11

8. Aviso\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11

9. Diagrama del sistema (Componentes) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_12

## 1. Objetivos

El propósito de este documento es informar de forma específica para lograr un correcto mantenimiento del sistema.

Principalmente el documento está orientado a guiar al usuario, informarle sobre su desarrollo, el proceso de instalación, seguimiento de código, etc.

### 1.1 Contenido

* Instalación del sistema.
* Vista del código para un mejor mantenimiento.
* Requisitos para la ejecución de dicho sistema.

## 2. Alcance

Este documento está dirigido principalmente a Técnicos, Programadores e Ingenieros Electrónicos.

Saberes requeridos: Conocimientos en Hardware y Software.

## 3. Requisitos

**3.1** HARDWARE:

* NodeMCU v3
* Móulo GPS Neo-6m
* Sensor de Lluvia}
* 10 cables macho-hembra
* 10 cables hembra-hembra
* Soldador
* Estaño
* Protoboard
* PowerBank

**3.2** SOFWARE:

* Arduino
* Visual Studio Code

## 4. Herramientas utilizadas y recomendadas

* Navegador web: estos pueden ser Opera GX, Google Chrome, Microsoft Edge, Mozilla Firefox, etc.: Para pruebas y testeos del sistema.
* Arduino IDE 2.0: Para subir el código a la placa.
* Librerías para el IDE de Arduino: TinyGPS++, SoftwareSerial, ESP8266WiFi y ESP8266HTTPClient.
* Página web de diseño 3d, Tinkercad: Para el desarrollo y creación de la estructura final del sistema.
* Protoboard: Para ejecutar pruebas y asegurar el perfecto funcionamiento del producto.

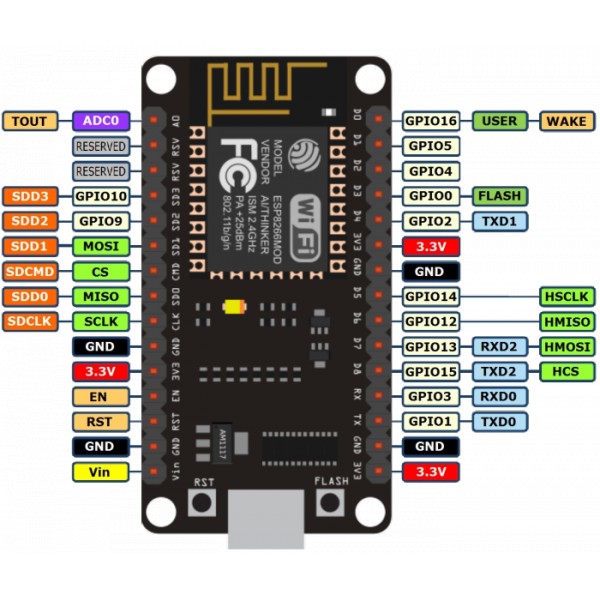
## 5. Ensamble

Para el correcto ensamble del sistema, tenemos que seguir los siguientes pasos:

(No se deben colocar resistencias, las que están colocadas simulan, junto a la placa nombrada GY-NEO6MV3, el circuito integrado en el Módulo GPS Neo-6m).

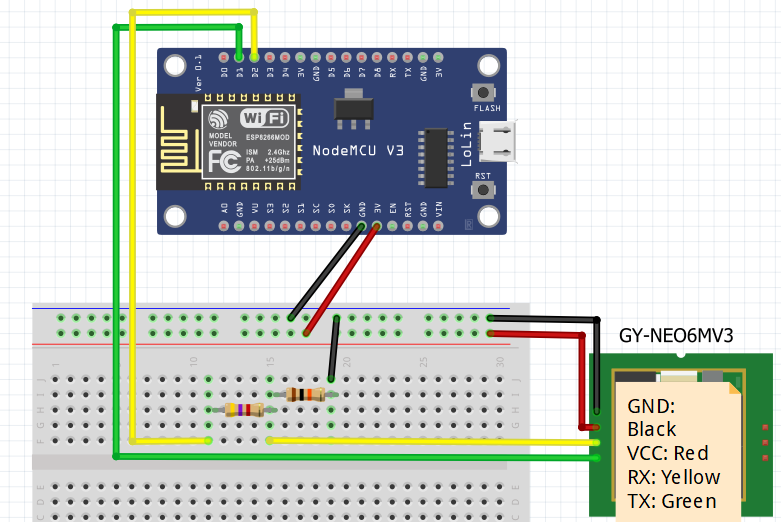
**5.1** NodeMCU

Especificación de pines de la placa NodeMCU



**5.2** Módulo GPS Neo-6m

* El Módulo GPS Neo-6m irá conectado a la placa NodeMCU, el Pin GND del Módulo GPS se conecta directamente al GND de la NodeMCU.
* El pin Vcc del Módulo GPS va conectado directamente a un pin de 3v.
* El pin RX se conecta al GPIO4 o D2 de la NodeMCU.
* El pin TX se conecta directamente al pin GPIO5 o D1 de la NodeMCU.



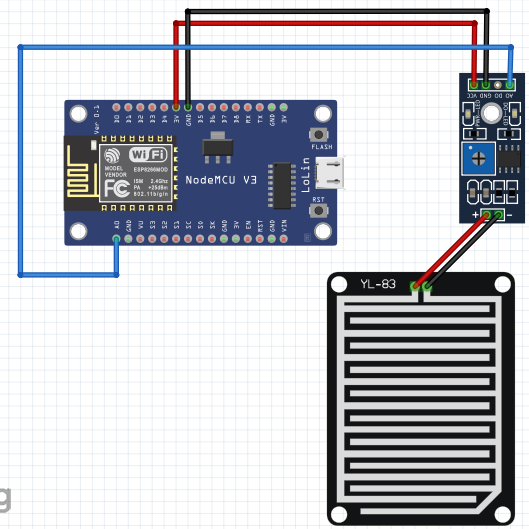
**5.3** Sensor de Agua

- El detector de agua (la placa negra cubierta por una fina lámina de aluminio) va conectado directamente a su placa de control, conectado el positivo con el positivo y el negativo con el negativo.

- EL pin GND de la placa de control va conectado directamente al pin GND de la NodeMCU.

- El pin VCC de la placa de control va conectado directamente a un pin de 3V de la NodeMCU.

- El pin analógico de la placa de control va conectado al pin A0 de la NodeMCU.



Una vez completado este procedimiento, el sistema comenzara a funcionar una vez que el usuario conecte el PowerBank a la placa NodeMCU para suministrarle energía.

## 6. Análisis

**6.1** Introducción

Este sistema ha sido analizado, diseñado, desarrollado y sometido a pruebas de diferentes tipos y al mismo tiempo dando soporte en cualquier fallo que pudiera haber en este, con el fin de que se brinde la mejor calidad y rendimiento posible a cualquier persona que quiera tener acceso a este.

### 6.2 Propósito

En esta sección se hablará de cómo se planteó y elaboró el sistema para evitar cualquier confusión que se tenga en base al desarrollo del mismo.

### 6.3 Alcance

MajinSack surge a partir de la problemática de los niños y personas que se pierden, ofreciendo el seguimiento del usuario de forma cómoda, poco perceptible y eficaz.

Se ha creado para cubrir la necesidad de ciertas personas y así brindar alivio o reducir la preocupación, sobre todo hacia los padres, esta necesidad es la ubicación de la persona.

Así logramos obtener la ubicación a tiempo real de la persona, para todo tipo de persona que lo necesite, siendo cómodo y fácil de llevar brindando, además, detección de agua y una entrada para cargar el celular.

### 6.4 Definiciones y acrónimos

MajinSack: Majin (Japonés) => Genio | Sack (abreviación del Inglés: rucksack) => Mochila

MajinSack: Mochila Genio

**7. Desarrollo**

A continuación, se va a proporcionar el código de Arduino del sistema.

**7.1** Definición de librerías

//---------Librerías------------

#include <TinyGPS++.h> // Librería para el módulo GPS.

#include <SoftwareSerial.h> //librerías necesarias para la conexión WiFi

#include <ESP8266WiFi.h> //       ""

#include <ESP8266HTTPClient.h>//       ""

**7.2** Definición de Variables

//---------Variables------------

#define PIN\_ANALOG\_RAIN\_SENSOR A0  //Entrada analógica para la señal del sensor lluvia

TinyGPSPlus gps;  //Objeto de la librería TinyGPS++

SoftwareSerial ss(4, 5); //Conexión serial del GPS a los pines GPIO4 y 5 de la placa NodeMCU

const char\* ssid = "Telecentro-Galarza";//Nombre de la red WiFi

const char\* password = "0111171227";//Contraseña de la red WiFi

String serverName = "http://localhost:4000"; //Nombre del Servidor de la página

float latitude , longitude; //Definición de las variables dónde se almacenarán la longitud y latitud

int year , month , date, hour , minute , second; //Variables donde se almacenarán los datos del Año, mes, día, hora, minuto y segundo

String date\_str , time\_str , lat\_str , lng\_str; //Definición de las variables dónde se almacenarán la fecha, la hora, la latitud y la longitud

int pm; //Definición de la variable PM

WiFiServer server(80);  //Servidor WiFi

**7.3** Función Setup

void setup()

{

  Serial.begin(115200);//inicio de la comunicación serie

  ss.begin(9600);//inicio de la comunicación con el GPS

  Serial.println();

  Serial.print("Connecting to ");

  Serial.println(ssid);

  WiFi.begin(ssid, password); //Conectando al WiFi

  while (WiFi.status() != WL\_CONNECTED)// Mientras no está conectado a una red

  {

    delay(500);

    Serial.print("."); //Imprime "...."

  }

  Serial.println("");

  Serial.println("WiFi connected");

  server.begin();

  Serial.println("Server started");

  Serial.println(WiFi.localIP());  // Imprimir la IP

}

**7.4** Función Loop

**7.4.1** Latitud y longitud

void loop(){

while (ss.available() > 0) //Cuando los datos están disponibles

    if (gps.encode(ss.read())) //Leer datos del GPS

    {

      if (gps.location.isValid()) //Si los datos de la localización que el gps lee son correctos

      {

        latitude = gps.location.lat();

        lat\_str = String(latitude , 6); // Valor de la latitud en una cadena de caracteres

        longitude = gps.location.lng();

        lng\_str = String(longitude , 6);  // Valor de la longitud en una cadena de caracteres

      }

**7.4.2** Fecha

if (gps.date.isValid()) //Si los datos (día, mes y año) que el gps lee son válidos

      {

        date\_str = "";

        date = gps.date.day();

        month = gps.date.month();

        year = gps.date.year();

        if (date < 10)

          date\_str = '0';

        date\_str += String(date); // Valor del día guardado en una cadena de caracteres

        date\_str += " - ";

        if (month < 10)

          date\_str += '0';

        date\_str += String(month); // Valor del mes guardado en una cadena de caracteres

        date\_str += " - ";

        if (year < 10)

          date\_str += '0';

        date\_str += String(year); // Valor del Año guardado en una cadena de caracteres

      }

**7.4.3** Tiempo

if (gps.time.isValid())  //Si el tiempo que el gps está leyendo es válido

      {

        time\_str = "";

        hour = gps.time.hour();

        minute = gps.time.minute();

        second = gps.time.second();

        if (minute > 59) // reseteo de minutos si los minutos pasan de 59 (60) se resetea, entonces nos muestra el minuto "0"

        {

          minute = minute - 60;

          hour = hour + 1;

        }

        hour = (hour - 3) ;  //Argentina tiene un formato UTC o GMT de -3 horas

        if (hour > 23)

          hour = hour - 24;   // reseteo de horario si la hora es más de 23 (24) se resetea, entonces nos muestra la hora "0"

        if (hour >= 12)  // Saber si es PM o AM

          pm = 1;

        else

          pm = 0;

        hour = hour % 12;

        if (hour < 10)

          time\_str = '0';

        time\_str += String(hour); //Valor de horas guardados en una cadana de caracteres.

        time\_str += " : ";

        if (minute < 10)

          time\_str += '0';

        time\_str += String(minute);//Valor de minutos guardados en una cadana de caracteres.

        time\_str += " : ";

        if (second < 10)

          time\_str += '0';

        time\_str += String(second); //Valor de segundos guardados en una cadana de caracteres

        if (pm == 1)

          time\_str += " PM ";

        else

          time\_str += " AM ";

      }

    }

**7.4.4** Conexión HTTP

WiFiClient client = server.available(); // Pedimos el estado del cliente

  if (!client)  //Preguntamos si está desconectado

  {

    return; //Enviamos nada

  }

  HTTPClient http; //Pedimos una conexión HTTP

  int sensorValue = analogRead(PIN\_ANALOG\_RAIN\_SENSOR); // Leer datos del puerto analógico (sensor de agua)

  //Creamos una ruta (http://localhost:4000/GPS?fecha=\*Una fecha\*&tiempo=\*un horario\*&latitud=\*una latitud\*&longitud=\*una longitud\*&agua=\*detección de agua\*)

  String serverPath = serverName + "/GPS?fecha=" + date\_str + "&tiempo=" + time\_str + "&latitud=" + lat\_str + "&longitud=" + lng\_str + "&agua=" + sensorValue;

  http.begin(client, serverPath.c\_str()); //iniciamos la conexión HTTP

  int httpResponseCode = http.GET(); //Creamos una variable que almacene una petición de tipo GET

if (httpResponseCode>0) { //si la conexión existe continúa con el proceso

    Serial.print("HTTP Response code: ");

    Serial.println(httpResponseCode);

    String payload = http.getString();

    Serial.println(payload);

  }

  else { //si la conexión no existe retorna un error

    Serial.print("Error code: ");

    Serial.println(httpResponseCode);

  }

  http.end(); //terminamos la conexión HTTP

  delay(5000); // Retardo entre mediciones

}

## 8. Aviso

El código de la página web no está disponible en este documento, ya que con su gran cantidad de código no es eficiente escribirlo en un documento.

Pero siempre tiene el documento “Código del sistema”, el cual contiene una carpeta llamada “Página” y otra llamada “MajinSack” donde podrá ver el código de la página web.

## 9. Diagrama electrónico del Sistema

