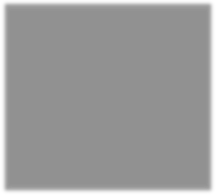
Established – 1961 Subject: \_\_\_\_\_\_\_\_\_\_\_

**SEVA SADAN’S**

**OF**



**ARTS, SCIENCE & COMMERCE ULHASNAGAR – 421 003**



**CERTIFICATE**

This is to certify that Mr./Ms. Simran nadeem khan  
of S.Y. Computer Science (SYCS) Roll No. \_\_\_\_\_\_\_\_2524021\_\_\_\_\_\_\_\_\_\_ has satisfactorily completed The Internet Of Thing Mini Project entitled\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_Women’s Safety Tracking Device\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

during the academic year 2025 – 2026, as a part of the practical requirement. The project work is found to be satisfactory and is approved for submission.

**PROF. INCHARGE HEAD OF DEPT**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr.no** | **Chapters** | **Page No.** |
| **1** | **Introduction** | **01-04** |
| **2** | **Requirement specification** | **05-06** |
| **3** | **System Design** | **07-16** |
| **4** | **Implementation** | **17-40** |
| **5** | **System Testing and Result** | **41-51** |
| **6** | **Future Scope and Conclusion** | **52-53** |
| **7** | **References** | **54-55** |
| **8** | **Glossary** | **56-57** |

**INTRODUCTION**

**Women Safety Tracking Device**

Women safety is a major concern in modern society due to the rising number of harassment and crime cases. Quick access to emergency help is essential to ensure personal security and reduce response time during dangerous situations. With the advancement of Internet of Things (IoT) technology, smart safety devices can provide instant alerts and real-time location tracking. The Women Safety Tracking Device is designed as a portable emergency system that enables users to send SOS alerts with their live location.

The device uses an ESP8266 microcontroller integrated with a GPS module to capture accurate geographical coordinates. A push button is provided to trigger emergency alerts quickly and easily during critical situations. Once activated, the system sends an SOS message with a Google Maps link to trusted contacts using Telegram via WiFi connectivity. An OLED display shows system status such as WiFi connection and alert confirmation. The device is compact, cost-effective, and easy to use, making it suitable for daily safety purposes. Overall, this project aims to enhance personal security by providing a reliable and efficient emergency tracking solution.

**MOTIVATION**

The motivation behind this project arises from the increasing concerns regarding women safety in public and private spaces. Many existing safety solutions rely on smartphones, which may not be easily accessible during emergencies. There is a need for a simple and quick alert system that can send real-time location information with minimal effort. Advances in IoT and GPS technologies make it possible to develop compact and reliable safety devices. This project aims to provide a low-cost portable solution that ensures instant SOS alerts and faster emergency response. Ultimately, the device is intended to improve confidence, independence, and personal security for women

**PROBLEM STATEMENT**

Women frequently encounter unsafe situations where immediate assistance is required but not readily available. Many existing safety solutions rely on smartphones, which may be difficult to access during emergencies or panic conditions. Delays in sending accurate real-time location information can further reduce the chances of timely help. There is also a lack of simple, portable, and dedicated emergency alert devices for personal safety. Hence, a reliable system is needed that can instantly send SOS alerts along with precise location tracking.

**How it works :**

The device is powered on and the ESP8266 connects to a WiFi network. The GPS module continuously receives satellite signals to determine the user’s location. When the emergency push button is pressed, the system captures the current latitude and longitude. The device then generates a Google Maps link using the obtained coordinates. An SOS alert message containing the location link is sent to predefined contacts via Telegram. Finally, the OLED display shows confirmation that the emergency alert has been successfully transmitted.

**Key Features**

1. **One-Button SOS Activation:** Allows users to trigger emergency alerts instantly with a single press.
2. **Real-Time GPS Tracking:** Captures accurate latitude and longitude for precise location sharing.
3. **Instant Telegram Alert:** Sends SOS message with Google Maps link to trusted contacts.
4. **WiFi Connectivity:** Enables fast and reliable communication through internet-based messaging.
5. **OLED Status Display:** Shows system updates such as WiFi connection, GPS status, and alert confirmation.
6. **Portable and Compact Design:** Small and lightweight device suitable for daily carrying or wearable use.
7. **Low-Cost and Easy to Use:** Designed with affordable components and simple operation for practical deployment.

**Scope of the Project**

1. The device can be used as a personal safety tool for women to send instant emergency alerts with live location.
2. It can be extended for safety monitoring of students, elderly individuals, and travelers.
3. The system can be integrated with GSM or mobile apps for wider communication without WiFi dependency.
4. Future enhancements may include wearable designs such as smart bands or jewelry for better portability.
5. The project provides a foundation for advanced safety systems using IoT, sensors, and AI-based threat detection.

**OBJECTIVES**

The main objective of this project is to design a portable women safety device that can send instant SOS alerts during emergency situations. It aims to provide real-time location tracking using a GPS module to help responders reach the user quickly. Another objective is to develop a simple one-button activation system for fast and easy use under stress. The project also focuses on integrating WiFi and Telegram communication for reliable alert transmission. Additionally, it seeks to display system status through an OLED screen for user awareness. Overall, the device aims to enhance personal security with a low-cost and efficient safety solution.

**Key Goals**

1. To develop a reliable emergency alert system that can be activated quickly during unsafe situations.
2. To provide accurate real-time location tracking using GPS technology.
3. To ensure instant transmission of SOS alerts to trusted contacts through wireless communication.
4. To design a compact, portable, and user-friendly safety device for daily use.
5. To create a low-cost and efficient solution that enhances personal security and emergency response.

**REQUIREMENT SPECIFICATION**

**Hardware Requirements**

| **Sr. No** | **Component Name** | **Quantity** | **Description** |
| --- | --- | --- | --- |
| 1 | ESP8266 NodeMCU | 1 | Main microcontroller used for WiFi communication and system control |
| 2 | GPS Module | 1 | Provides real-time latitude and longitude location tracking |
| 3 | OLED Display | 1 | Displays system status such as WiFi connection and SOS confirmation |
| 4 | Push Button | 1 | Used to trigger emergency SOS alert |
| 5 | Connecting Wires | Few | Used for electrical connections between components |
| 6 | Breadboard and PCB | 1 | Used for assembling circuit connections |

### ****Software Requirements Table — Women Safety Tracking Device****

| **Sr. No** | **Software / Library** | **Purpose** |
| --- | --- | --- |
| 1 | Arduino IDE | Used to write, compile, and upload code to ESP8266 |
| 2 | ESP8266WiFi Library | Enables WiFi connectivity for internet communication |
| 3 | Universal Telegram Bot Library | Sends SOS alert messages through Telegram |
| 4 | TinyGPS++ Library | Processes GPS data to obtain latitude and longitude |
| 5 | ArduinoJson Library | Handles JSON data required for Telegram communication |
| 6 | Wire Library | Supports I2C communication with OLED display |
| 7 | Adafruit SSD1306 Library | Controls OLED display operations |
| 8 | Adafruit GFX Library | Provides graphics support for OLED text and shapes |

**SYSTEM DESIGN**

**Block Diagram**

Push Button

(sos trigger)

ESP8266

Microcontroller

GSM

Module

OLED

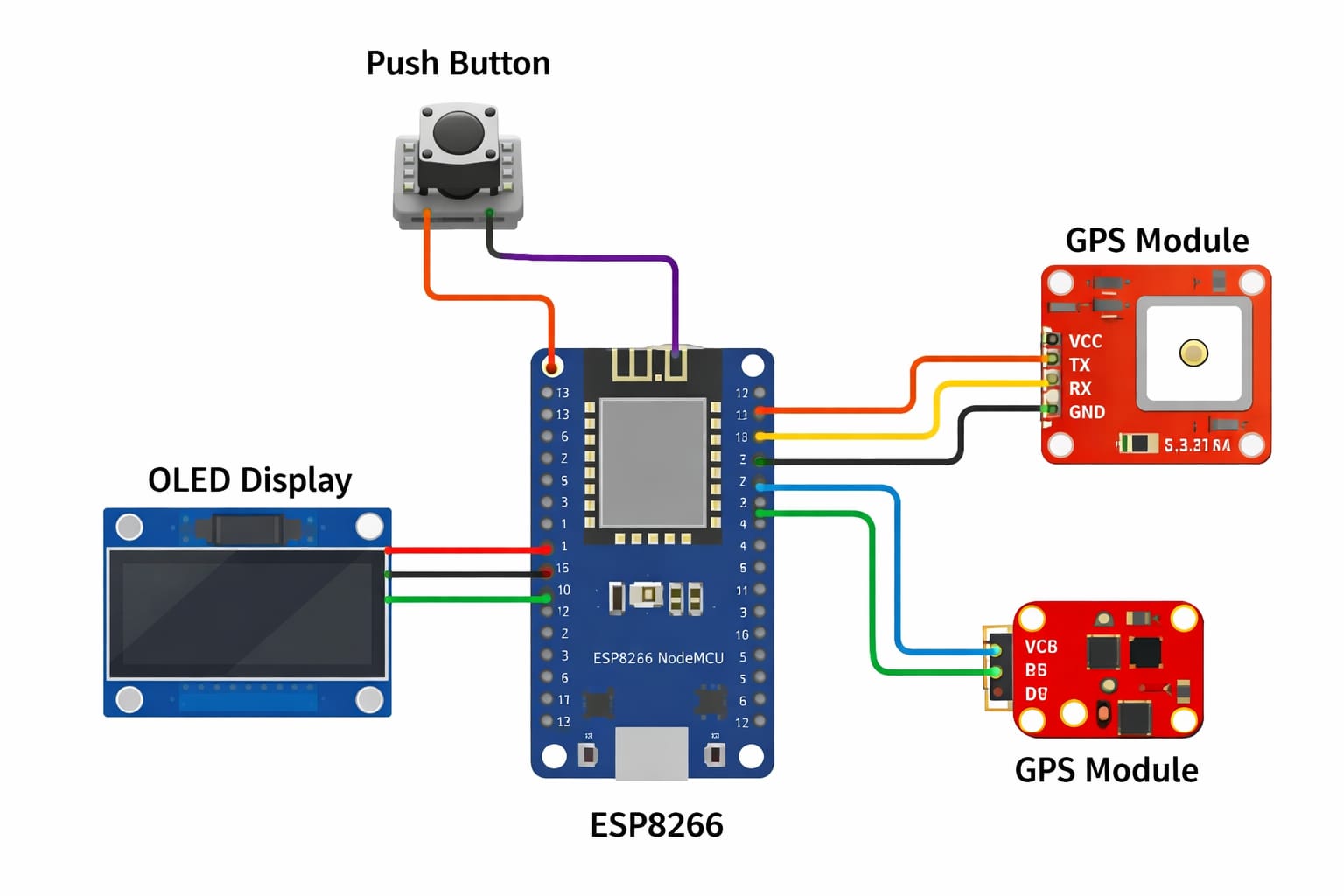
Display

Emergency

Contact/User

The push button acts as an emergency trigger that sends a signal to the ESP8266 microcontroller when pressed. The ESP8266 processes this input and activates the GSM module to send an SOS alert message to predefined contacts. Simultaneously, the OLED display provides system status updates such as SOS activation and message confirmation. The GSM module enables wireless communication, ensuring that emergency alerts are delivered instantly to the user’s contacts.

**Circuit Diagram :**

****

The circuit diagram illustrates the connection between the ESP8266 microcontroller, GPS module, OLED display, and push button used in the women safety tracking device. The ESP8266 acts as the central controller that manages all system operations and communication between components. The OLED display is connected to the ESP8266 through I2C pins (D1 and D2) to provide real-time status information such as WiFi connectivity, GPS signal, and SOS alert confirmation.

The GPS module is interfaced with the ESP8266 using serial communication pins (D5 and D6) to transmit latitude and longitude data required for location tracking. The push button is connected to digital pin D7 and ground, functioning as an emergency trigger that activates the SOS alert when pressed. Power is supplied to all components through the ESP8266’s regulated 3.3V and ground pins, ensuring stable operation of the device.

Overall, the circuit ensures seamless integration of input, processing, display, and communication modules to enable instant emergency alert functionality with real-time location tracking.

**Key Steps of Circuit Diagram:**

 **ESP8266 Setup:** Connect the ESP8266 as the main controller that will process inputs and manage communication between all components.

 **OLED Display Connection:** Interface the OLED display with ESP8266 using I2C pins (D1 for SCL and D2 for SDA) to display system status messages.

 **GPS Module Connection:** Connect the GPS module to ESP8266 using serial pins (D5 and D6) to receive real-time latitude and longitude data.

 **Push Button Connection:** Attach the push button to digital pin D7 and ground, enabling it to act as an emergency SOS trigger.

 **Power Supply Integration:** Provide power to all components through ESP8266’s 3.3V and GND pins to ensure stable circuit operation.

 **Clear Visualization:** It provides a clear representation of how electronic components are connected within the system.

**Benefits of circuit diagram :**

 **Easy Troubleshooting:** Helps identify wiring errors and faults before or during implementation.

 **Simplifies Assembly:** Guides users in correctly assembling components and making proper connections.

 **Documentation:** Serves as a reference for future maintenance, modification, or replication of the project.

 **Improves Understanding:** Helps students and developers understand the working and interaction of different components in the circuit.

**When to Use a Circuit Diagram** ?

 **Design Phase:** While planning an electronic project to visualize component connections before implementation.

 **Assembly Stage:** During hardware setup to ensure accurate wiring and correct component placement.

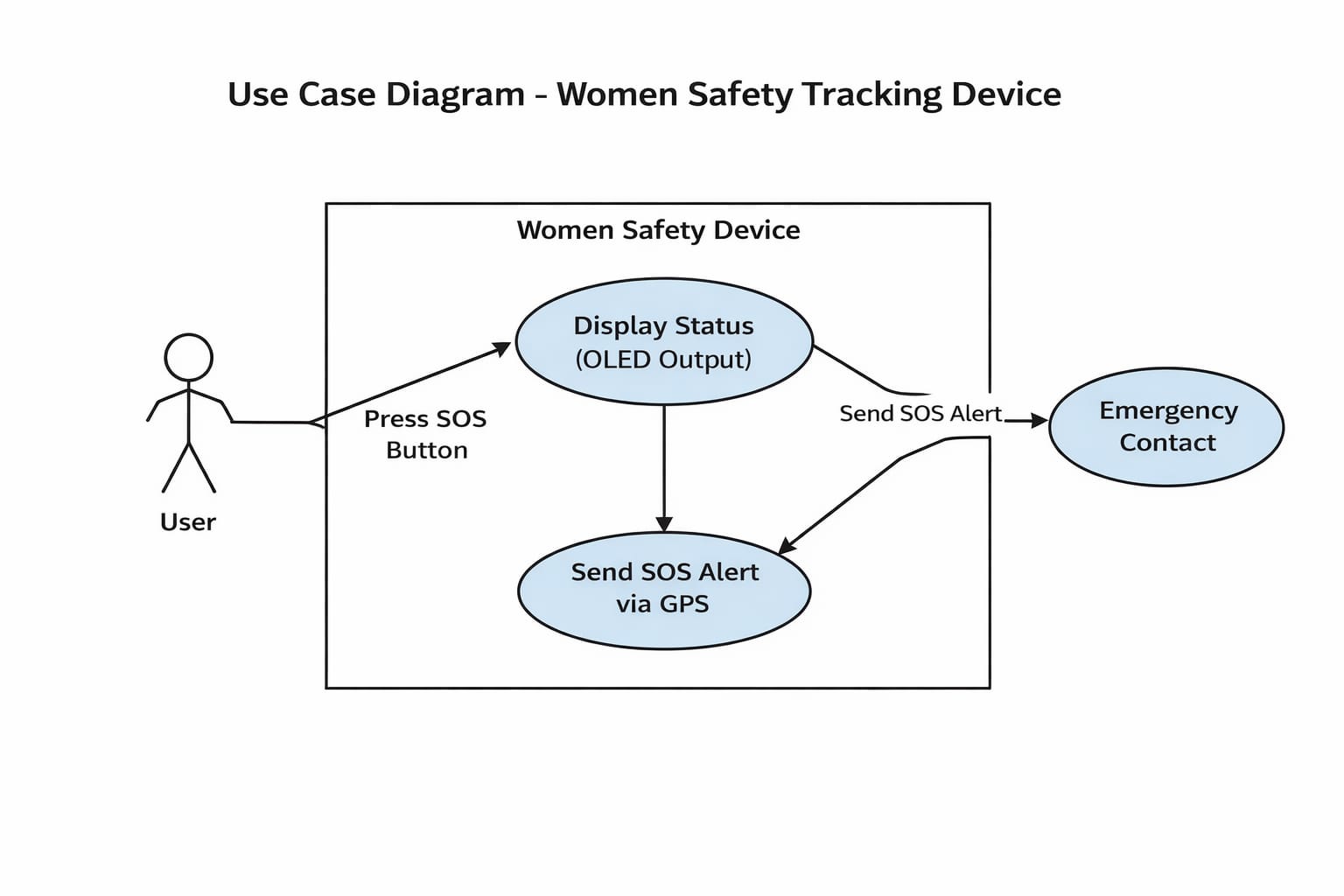
 **Troubleshooting:** To identify faults, loose connections, or wiring mistakes in the circuit.

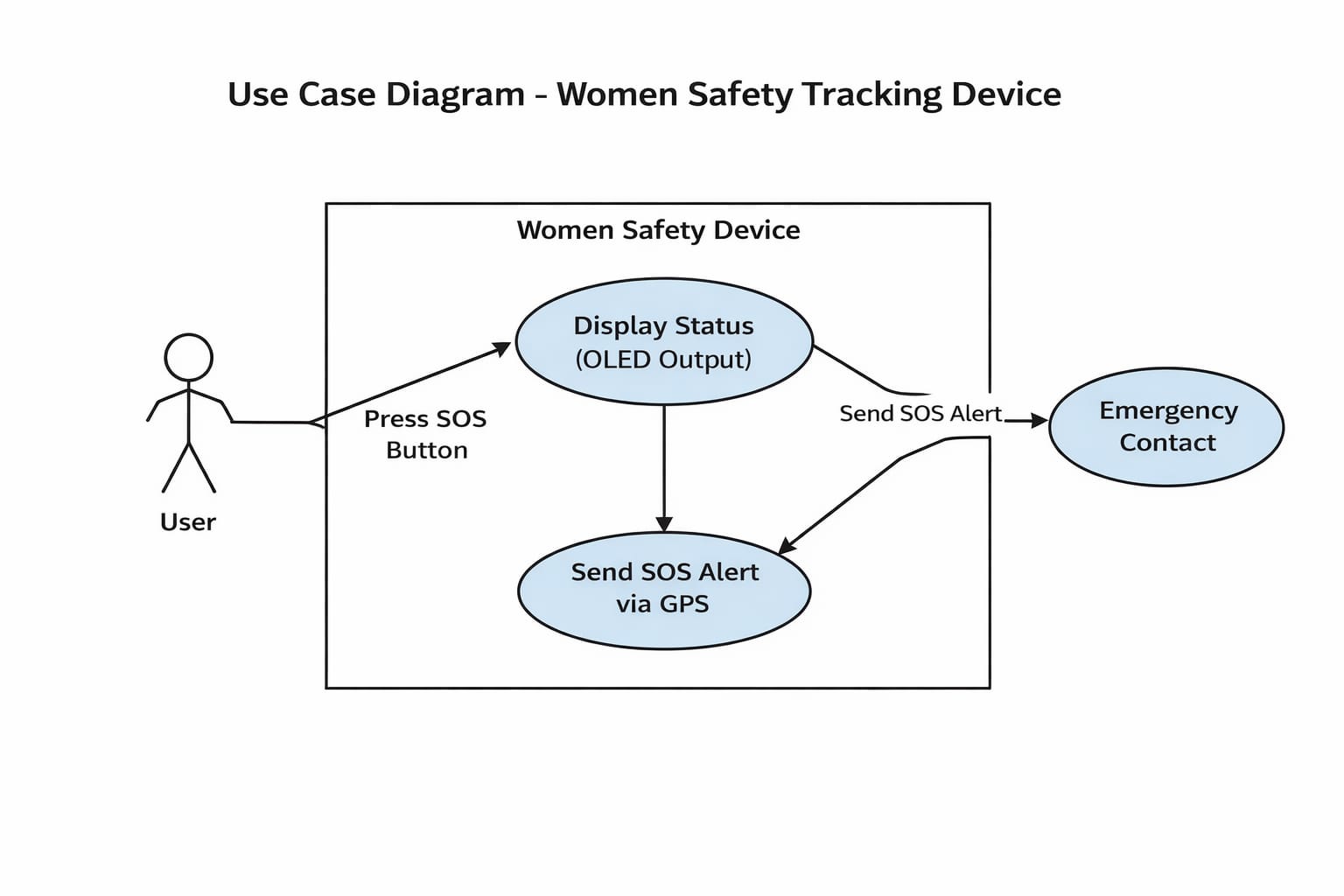
 **Documentation:** For recording the project design for future reference, sharing, or replication.

 **Educational Purpose:** To help students understand circuit operation and component interaction clearly.

**Circuit Model Testing Methods** :

1. **Simulation Testing:** Use software tools (such as Proteus or Tinkercad) to simulate the circuit before physical implementation and verify its functionality.
2. **Continuity Testing:** Check wiring connections using a multimeter to ensure there are no open circuits or short circuits.
3. **Functional Testing:** Power the circuit and verify that each component (ESP8266, GPS, OLED, push button) operates as expected.
4. **Load Testing:** Evaluate circuit performance under actual operating conditions to ensure stability and reliability.
5. **Debugging:** Identify and resolve issues by examining code, connections, and component behavior during operation.

**Use Case Diagram : Bottom of Form**

****

The use case diagram represents the interaction between the user, the women safety tracking device, and the emergency contact. The primary actor in the system is the user, who activates the device by pressing the SOS button during an emergency situation. Once triggered, the device performs internal operations such as processing the input and displaying system status on the OLED screen.

The device then captures the user’s location using the GPS module and sends an SOS alert message to predefined emergency contacts. The emergency contact acts as the secondary actor who receives the alert and can provide immediate assistance. This diagram highlights the main functionalities of the system, including SOS activation, status display, and emergency alert transmission, ensuring a clear understanding of system behavior and user interaction.

**purpose of use case diagram**

1. **Visualizing System Interaction:** It shows how users and external entities interact with the system and its functionalities.
2. **Identifying Requirements:** Helps define and understand the functional requirements of the system clearly.
3. **Improving Communication:** Provides an easy way for developers, stakeholders, and users to understand system behavior.
4. **Guiding System Design:** Assists in planning system features based on user needs and interactions.
5. **Supporting Testing:** Helps create test cases by outlining all possible user actions and system responses.
6.  **Clarifying User Roles:** Helps identify different actors involved in the system and their responsibilities.
7.  **Simplifying Complex Systems:** Breaks down system functionalities into manageable use cases for easier understanding.
8.  **Documentation Support:** Serves as useful documentation for future development, maintenance, and system enhancements.

**Activity Diagram :**

****

**Description :**

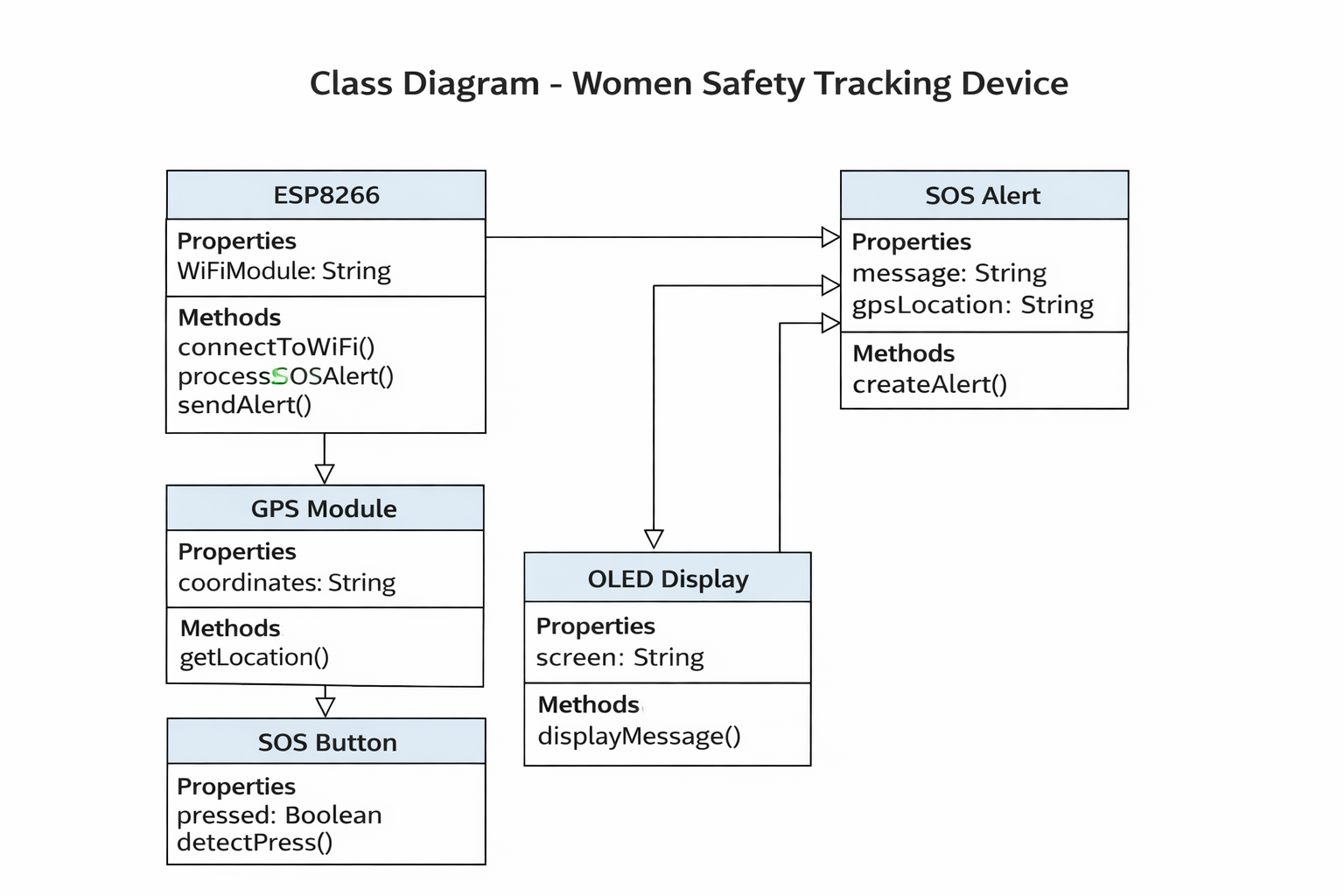
The activity diagram represents the workflow of the women safety tracking device from start to finish. The process begins when the system is powered on and continuously monitors the SOS push button for emergency activation. When the user presses the button, the device detects the input and displays an SOS activation message on the OLED screen.

Next, the system captures the user’s real-time location using the GPS module and processes the obtained coordinates. The ESP8266 then generates an SOS alert containing the location information and sends it to predefined emergency contacts through wireless communication. Upon receiving the alert, the emergency contact initiates a response to provide assistance. Finally, the process ends after successful alert transmission and confirmation, completing the emergency workflow of the system.

**Purpose of Activity Diagrams :**

1. **Visualizing Workflow:** Helps represent the step-by-step flow of activities and processes within the system.
2. **Understanding System Behavior:** Shows how different actions and decisions occur during system operation.
3. **Identifying Parallel Activities:** Helps illustrate tasks that can occur simultaneously in the system.
4. **Improving Communication:** Provides a clear visual representation that is easy for developers and stakeholders to understand.
5. **Supporting System Design:** Assists in identifying process gaps and improving system efficiency before implementation.
6. **Documentation:** Serves as useful documentation for future maintenance and system enhancements.
7. **Testing and Validation:** Helps derive test cases by outlining expected process flows and outcomes.

**Class Diagram :**

****

The class diagram represents the structural design of the women safety tracking device by illustrating different system components as classes and showing their relationships. The ESP8266 class acts as the central controller that manages system operations such as connecting to WiFi, processing SOS alerts, and sending emergency notifications. The GPS module class provides location data through its coordinate property and location retrieval method, which is used by the ESP8266 to determine the user’s position.

The OLED display class is responsible for presenting system status messages such as SOS activation and alert confirmation using its display function. The SOS button class represents the emergency input component, containing properties related to button state and methods for detecting button press events. Additionally, the SOS alert class stores alert information including the message and GPS location, and provides a method for creating and managing emergency alerts.

Overall, the diagram highlights how different classes interact and collaborate to enable SOS detection, location tracking, alert generation, and user feedback, ensuring efficient system functionality.

**Purpose of Class Diagrams :**

1. **Modeling System Structure:** Helps represent the static structure of the system by showing classes, attributes, and methods.
2. **Understanding Relationships:** Illustrates relationships and interactions between different components of the system.
3. **Design Blueprint:** Acts as a guide for implementing the system in hardware and software development.
4. **Improving Communication:** Provides a clear visual representation for developers and stakeholders to understand system architecture.
5. **Supporting Code Development:** Helps programmers organize classes and methods before coding.
6. **Documentation:** Serves as useful documentation for system maintenance and future upgrades.
7. **Simplifying Complex Systems:** Breaks down complex systems into manageable classes for better understanding.

The Gantt chart represents the timeline and scheduling of activities involved in the development of the Women Safety Tracking Device project. It visually illustrates the sequence, duration, and overlap of different project tasks from initiation to completion. The project began with planning and material gathering, followed by assembling the hardware components and installing the Arduino with proper wiring.

Subsequently, programming of the Arduino was carried out to implement system functionality, including SOS alert generation and location tracking. Testing and debugging were performed to identify and resolve errors, ensuring reliable system performance. Finally, adjustments and refinements were completed to improve device stability and usability before project completion.

Overall, the Gantt chart provides a clear overview of project progress, helps track task dependencies, and ensures efficient time management throughout the development process.

**Advantages of gantt chart**

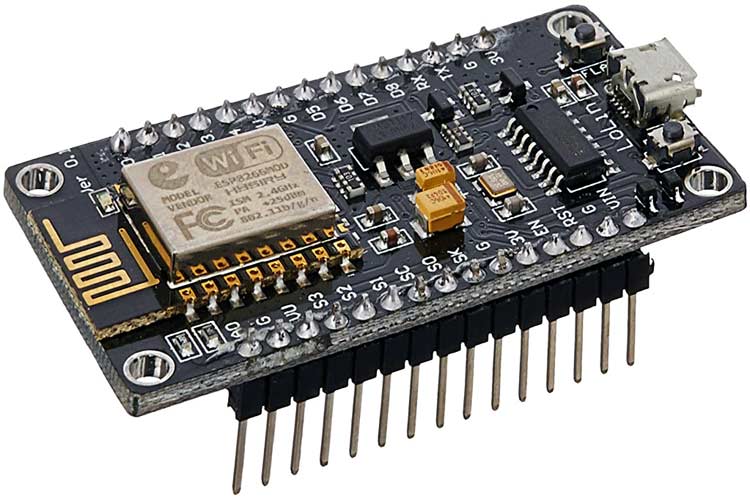
1. **Clear Project Timeline:** Provides a visual representation of project tasks along with their start and end dates.
2. **Better Time Management:** Helps in planning and scheduling activities efficiently to avoid delays.
3. **Task Tracking:** Enables monitoring of project progress and identification of completed and pending tasks.
4. **Improved Coordination:** Helps team members understand task dependencies and workflow sequence.
5. **Easy Communication:** Offers a simple visual tool to explain project status to stakeholders and supervisors.
6. **Resource Planning:** Assists in allocating resources effectively for different project phases.
7. **Identifying Overlaps:** Helps detect overlapping tasks and manage workload distribution properly.

**SYSTEM IMPLEMENTATION**

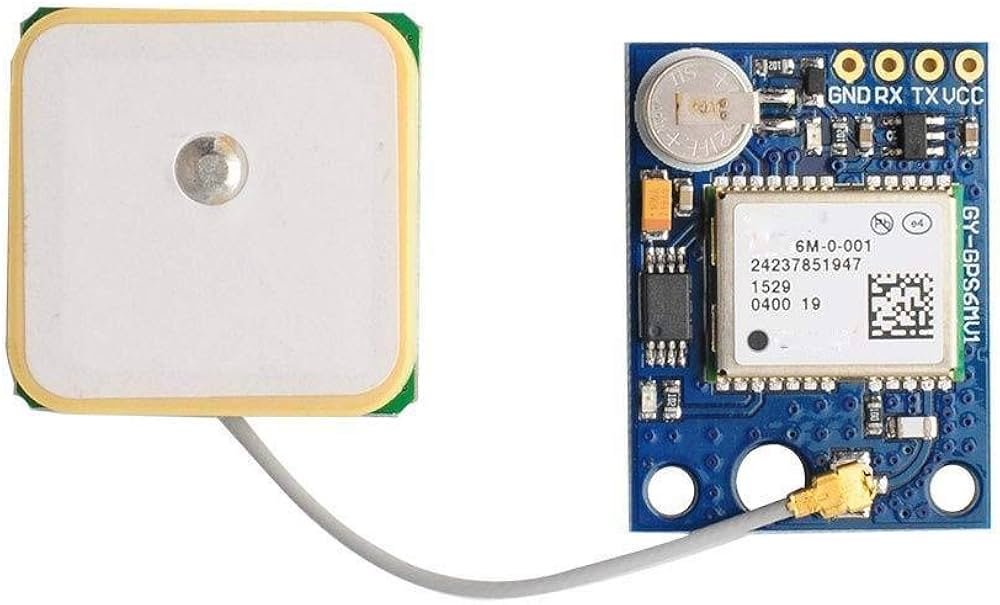
**Components & Assembling:**

**Step 1:**

Firstly, identify these components



# NodeMCU ESP8266



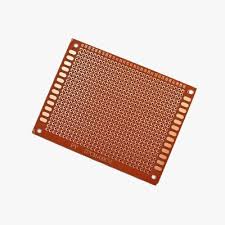
# NEO-6M GPS Module



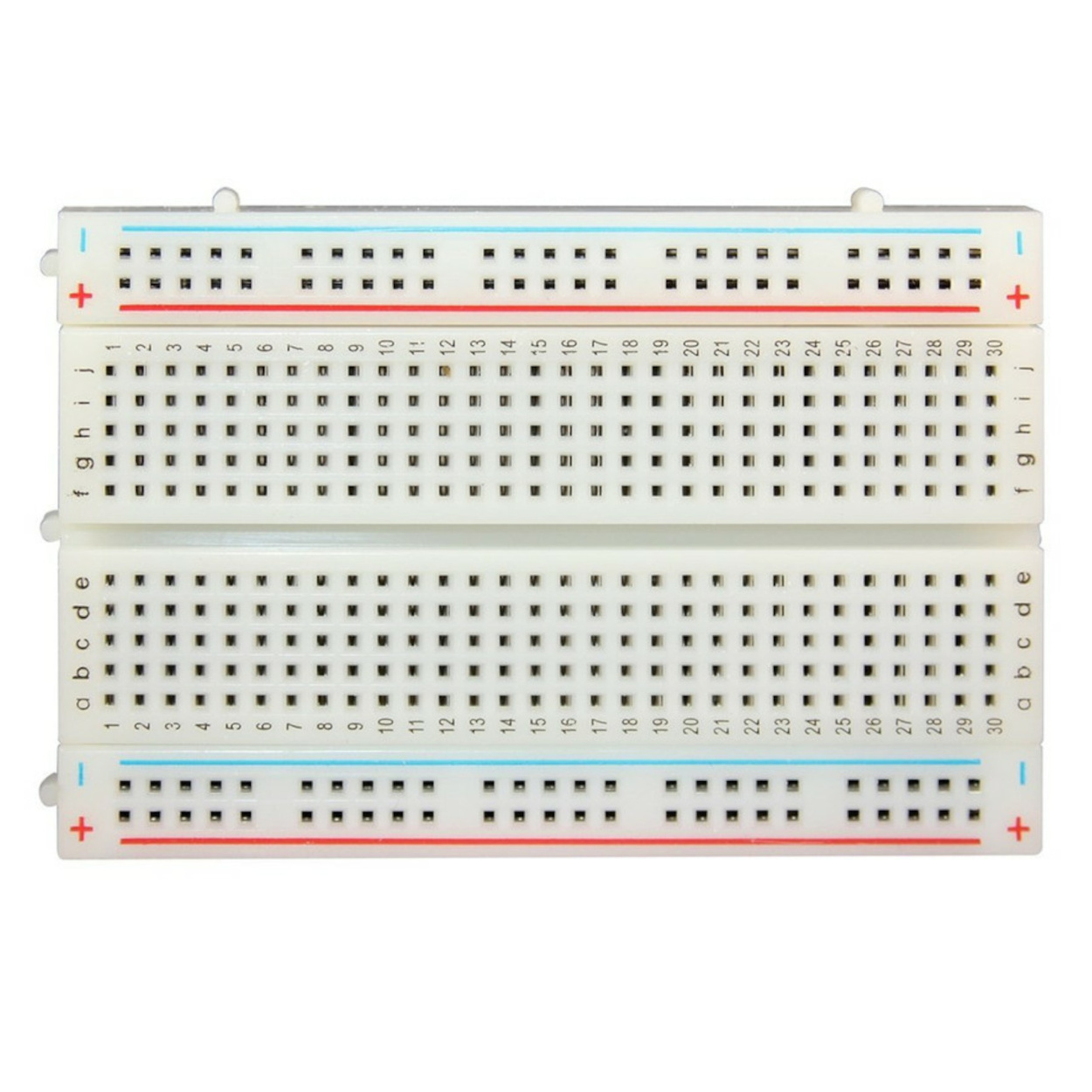
OLED Display



Push Button



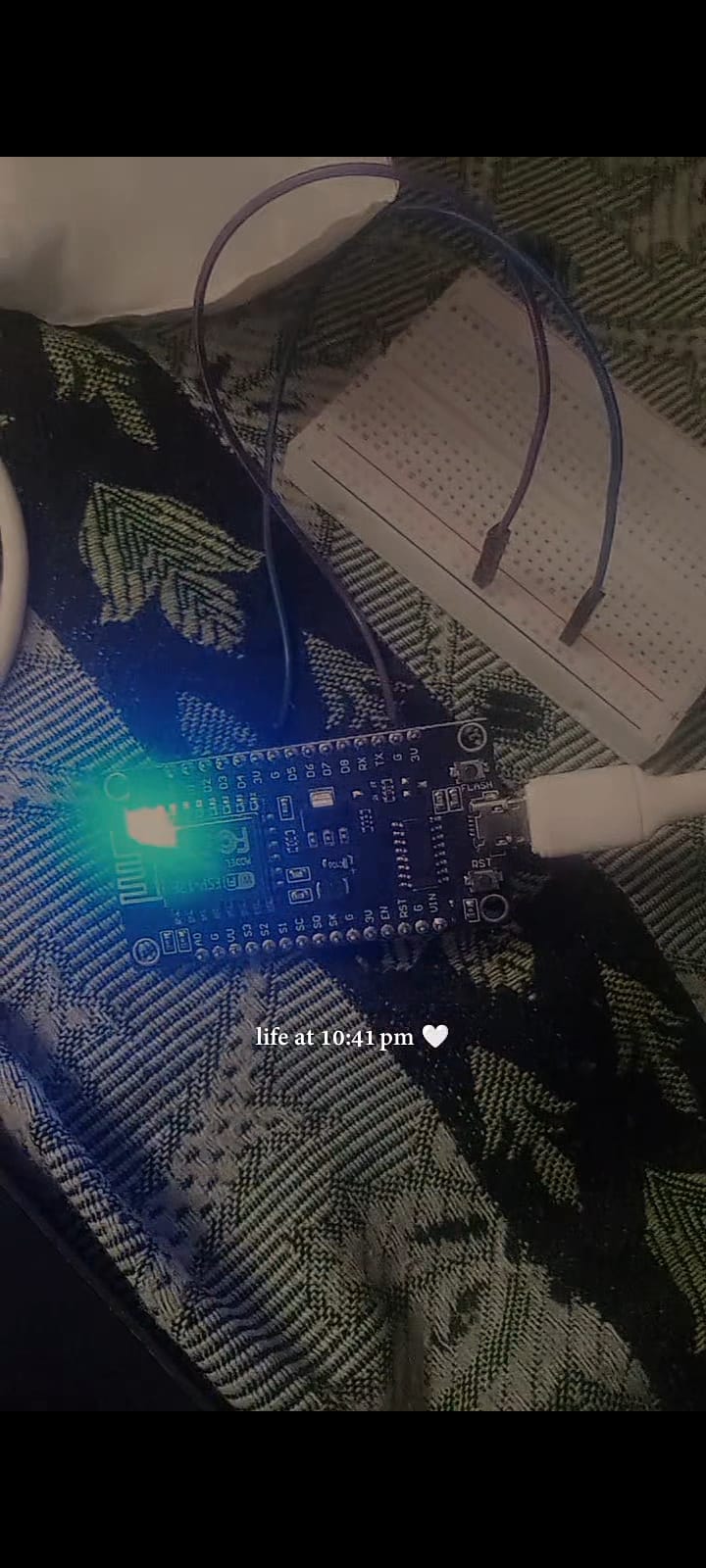
PCB



BreadBoard

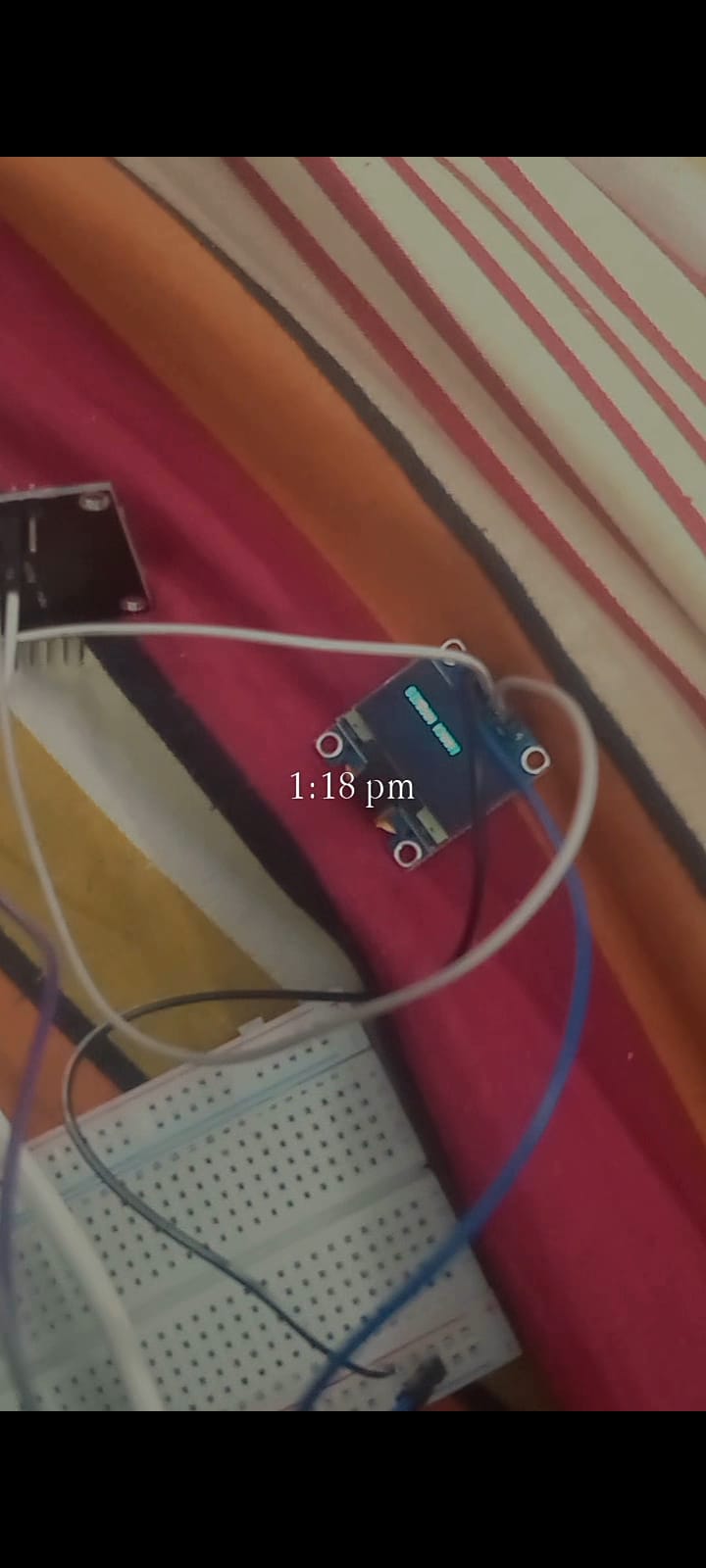
**Step 2:**

checking connection



Checking if the esp is working or not

| **ESP8266 Pin** | **Breadboard Rail** |
| --- | --- |
| 3.3V | Red (+) rail |
| GND | Blue (–) rail |



Checking oled is working or not

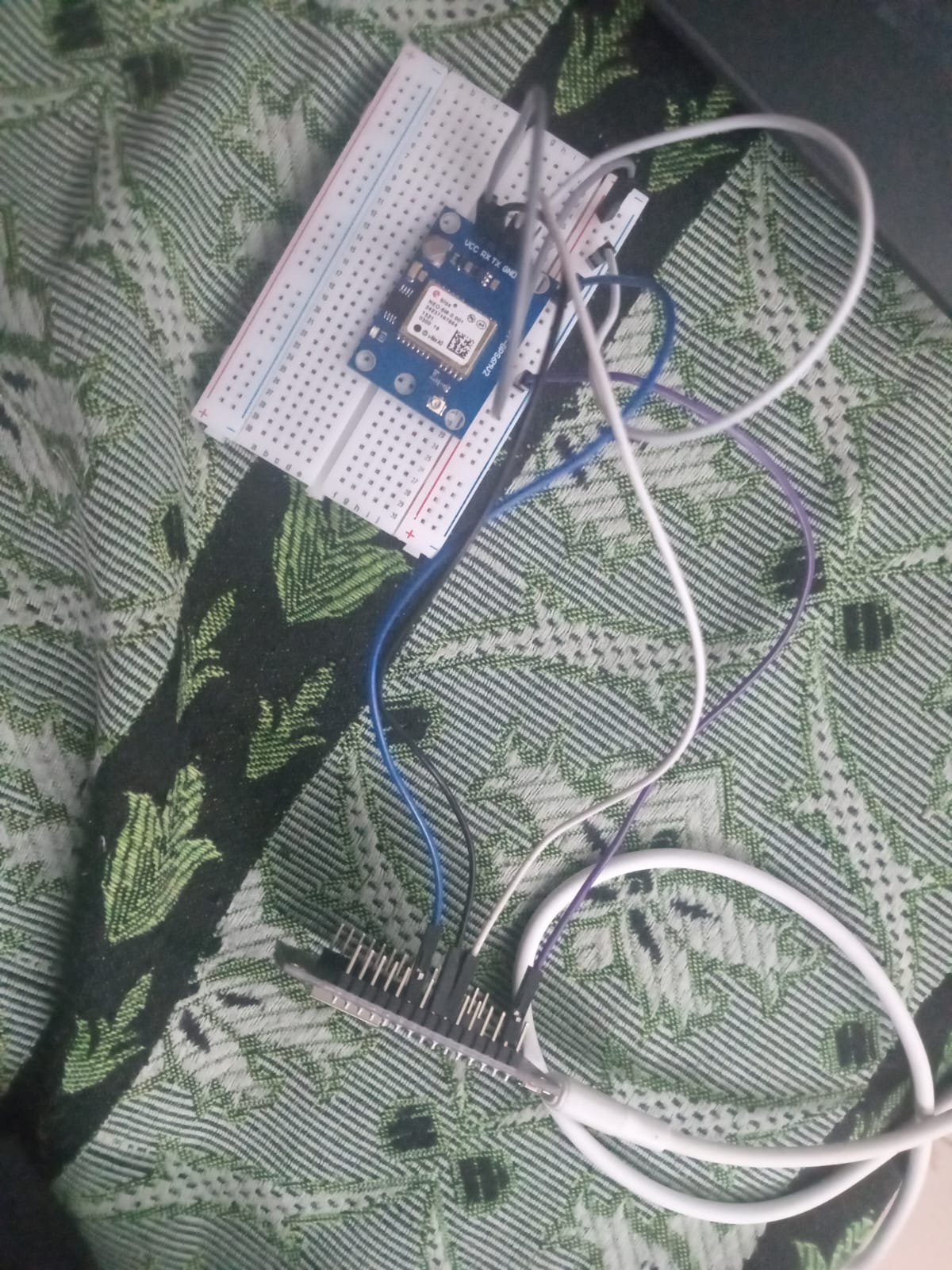
| **OLED Pin** | **Breadboard Rail** |
| --- | --- |
| VCC | Red (+) rail |
| GND | Blue (–) rail |

👉 Rails must be connected to ESP8266 3.3V & GND

| **OLED Pin** | **ESP8266 Pin** |
| --- | --- |

|  |  |
| --- | --- |
| SDA | D2 (GPIO4) |

|  |  |
| --- | --- |
| SCL | D1 (GPIO5) |



Checking gps is working or not

| **GPS Pin** | **ESP8266 Pin** |
| --- | --- |

|  |  |
| --- | --- |
| VCC | 3.3V |

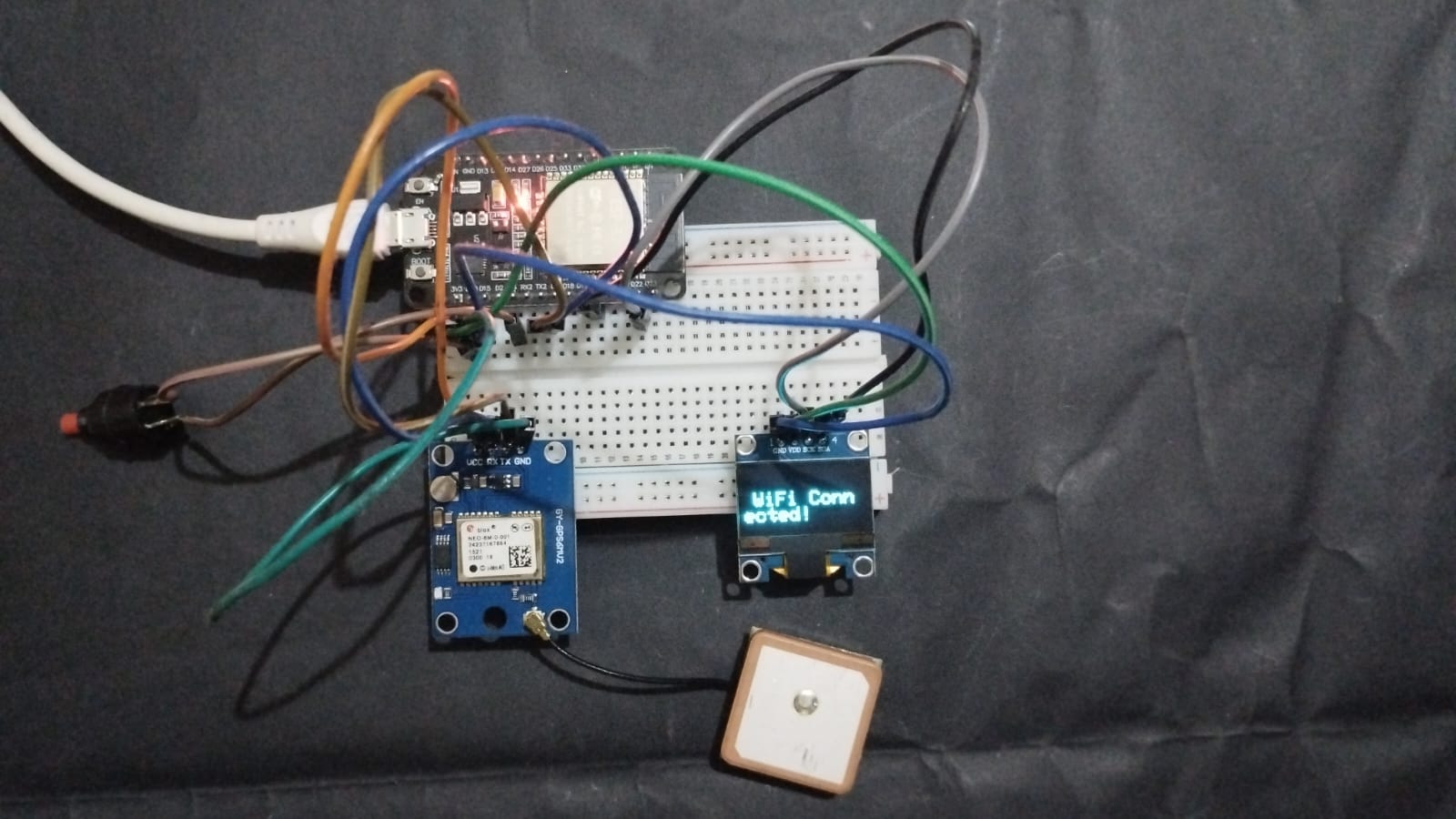
|  |  |
| --- | --- |
| GND | GND |

|  |  |
| --- | --- |
| TX | D5 (GPIO14) |

|  |  |
| --- | --- |
| RX | D6 (GPIO12) |

**Step 3:**

Connecting all components in the breadboard

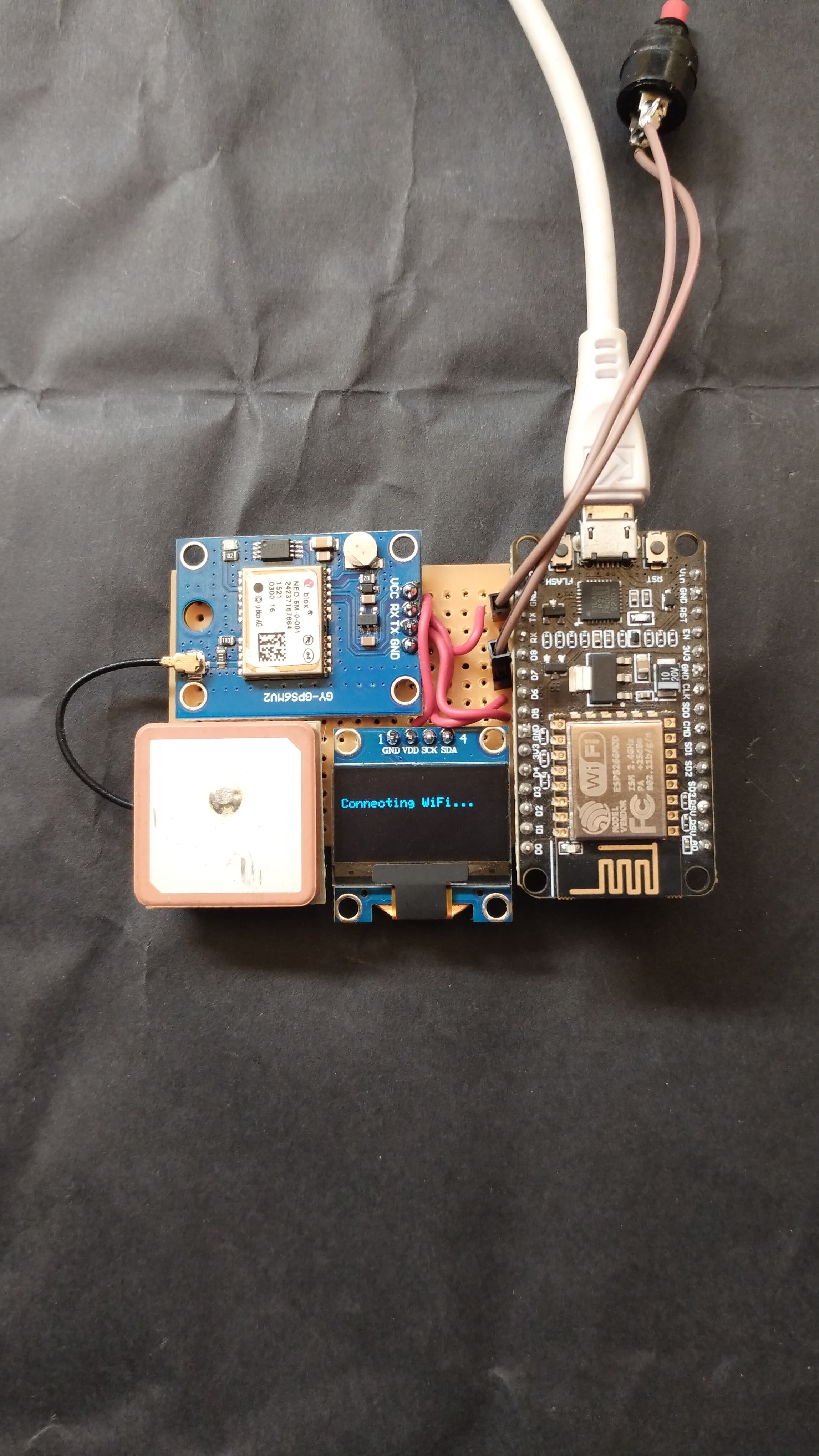


In the breadboard I have also added a push button

| **Push Button Pin** | **ESP8266 Pin** |
| --- | --- |
| One side | D7 (GPIO13) |
| Other side | GND |

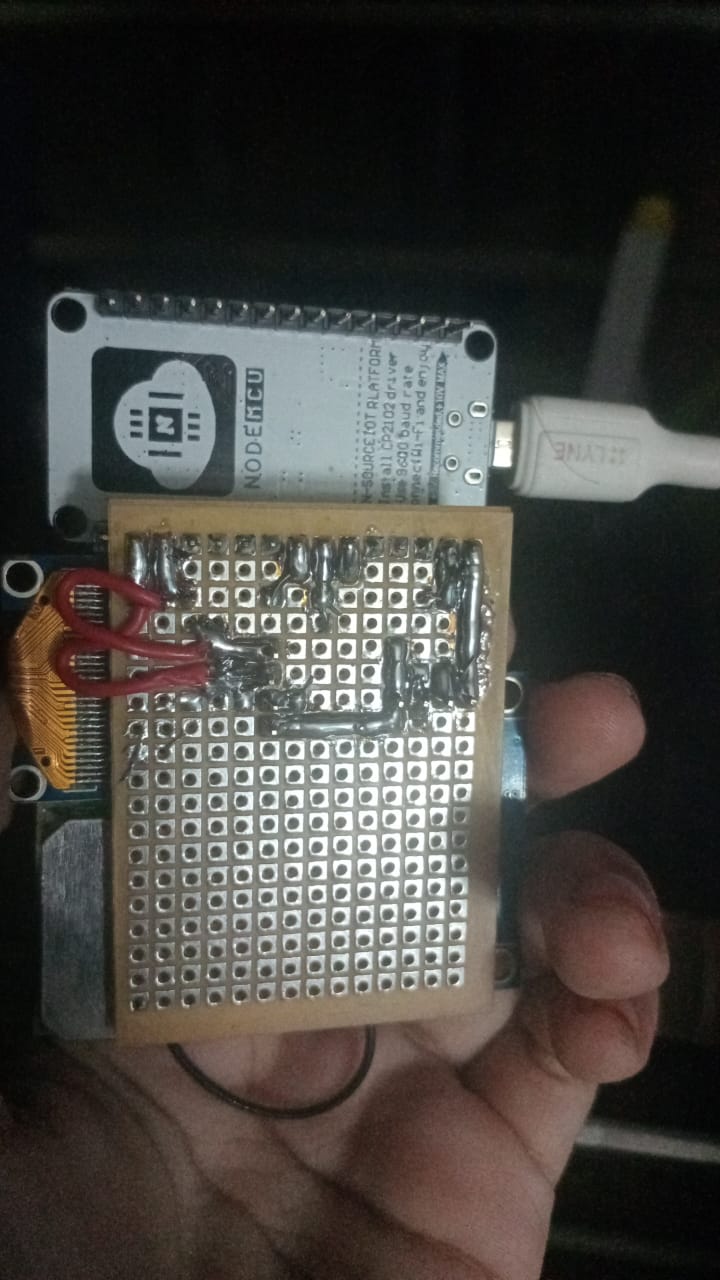
**Step 4:**

Connecting all the modules in pcb



**Step 5:**

Soldering them



**Step 6:**

So, let’s create the program for this project. .It is as follows.

#include <ESP8266WiFi.h>

#include <WiFiClientSecure.h>

#include <UniversalTelegramBot.h>

#include <ArduinoJson.h>

#include <TinyGPS++.h>

#include <SoftwareSerial.h>

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

/\*\*\*\*\*\*\*\* WIFI \*\*\*\*\*\*\*\*/

const char\* ssid = "simran";

const char\* password = "12345678";

/\*\*\*\*\*\*\*\* TELEGRAM \*\*\*\*\*\*\*\*/

#define BOT\_TOKEN "7981008968:AAFRm144vcIJnjn4NyMUpwz8x94oxwOWfjI"

#define CHAT\_ID "8318395880"

WiFiClientSecure client;

UniversalTelegramBot bot(BOT\_TOKEN, client);

/\*\*\*\*\*\*\*\* GPS \*\*\*\*\*\*\*\*/

TinyGPSPlus gps;

SoftwareSerial gpsSerial(D5, D6); // RX, TX

/\*\*\*\*\*\*\*\* OLED \*\*\*\*\*\*\*\*/

#define SCREEN\_WIDTH 128

#define SCREEN\_HEIGHT 64

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, -1);

/\*\*\*\*\*\*\*\* PUSH BUTTON \*\*\*\*\*\*\*\*/

#define BUTTON\_PIN D7 // 🔴 Changed from D3 to D7

bool buttonPressed = false;

unsigned long lastPressTime = 0;

void setup() {

Serial.begin(115200);

delay(1000);

pinMode(BUTTON\_PIN, INPUT\_PULLUP);

gpsSerial.begin(9600);

Wire.begin(D2, D1);

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

showStartupScreen();

connectWiFi();

client.setInsecure(); // Required for Telegram

}

/\* ------------------ LOOP ------------------ \*/

void loop() {

while (gpsSerial.available()) {

gps.encode(gpsSerial.read());

}

if (digitalRead(BUTTON\_PIN) == LOW && millis() - lastPressTime > 1000) {

lastPressTime = millis();

sendSOS();

}

}

/\* ------------------ FUNCTIONS ------------------ \*/

void showStartupScreen() {

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(10, 20);

display.println("Women Safety SOS");

display.setCursor(25, 35);

display.println("System Starting...");

display.display();

delay(2000);

}

void connectWiFi() {

display.clearDisplay();

display.setCursor(0, 20);

display.println("Connecting WiFi...");

display.display();

WiFi.begin(ssid, password);

int timeout = 0;

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

timeout++;

if (timeout > 40) { // 20 sec timeout

display.clearDisplay();

display.setCursor(10, 25);

display.println("WiFi Failed!");

display.display();

Serial.println("\nWiFi Failed");

return;

}

}

Serial.println("\nWiFi Connected!");

Serial.println(WiFi.localIP());

display.clearDisplay();

display.setCursor(10, 25);

display.println("WiFi Connected!");

display.display();

delay(2000);

}

void sendSOS() {

display.clearDisplay();

display.setCursor(15, 25);

display.println("Getting GPS...");

display.display();

unsigned long start = millis();

while (!gps.location.isValid() && millis() - start < 10000) {

while (gpsSerial.available()) {

gps.encode(gpsSerial.read());

}

}

if (!gps.location.isValid()) {

display.clearDisplay();

display.setCursor(10, 25);

display.println("GPS Not Fixed!");

display.display();

delay(2000);

return;

}

float lat = gps.location.lat();

float lng = gps.location.lng();

display.clearDisplay();

display.setCursor(10, 20);

display.println("Sending SOS...");

display.display();

String message = "🚨 WOMEN SAFETY SOS ALERT!\n\n";

message += "Emergency Help Needed!\n\n";

message += "Location:\n";

message += "https://maps.google.com/?q=";

message += String(lat, 6);

message += ",";

message += String(lng, 6);

bool sent = bot.sendMessage(CHAT\_ID, message, "");

display.clearDisplay();

display.setCursor(10, 25);

if (sent) {

display.println("Location Sent!");

Serial.println("Message Sent");

} else {

display.println("Send Failed!");

Serial.println("Telegram Failed");

}

display.display();

delay(3000);

display.clearDisplay();

display.setCursor(20, 25);

display.println("System Ready");

display.display();

}

**Step 7:**

Perfect 👍 I’ll explain your code in **easy English + line-by-line** so you can use it in viva and report.

# ⭐ ****1️⃣ Libraries****

#include <ESP8266WiFi.h>

👉 Enables WiFi connection for ESP8266

#include <WiFiClientSecure.h>

👉 Allows secure HTTPS communication (needed for Telegram)

#include <UniversalTelegramBot.h>

👉 Library to send Telegram messages

#include <ArduinoJson.h>

👉 Helps Telegram library handle JSON data

#include <TinyGPS++.h>

👉 Reads and processes GPS data

#include <SoftwareSerial.h>

👉 Creates extra serial pins for GPS communication

#include <Wire.h>

👉 Enables I2C communication (used by OLED)

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

👉 Used to control OLED display

# ⭐ ****2️⃣ WiFi credentials****

const char\* ssid = "simran";

const char\* password = "12345678";

👉 Stores WiFi name and password

# ⭐ ****3️⃣ Telegram details****

#define BOT\_TOKEN "...."

#define CHAT\_ID "...."

👉 Bot token identifies your Telegram bot  
👉 Chat ID is where alert will be sent

WiFiClientSecure client;

UniversalTelegramBot bot(BOT\_TOKEN, client);

👉 Creates Telegram bot object

# ⭐ ****4️⃣ GPS setup****

TinyGPSPlus gps;

👉 Creates GPS object

SoftwareSerial gpsSerial(D5, D6);

👉 GPS connected to D5 (RX) and D6 (TX)

# ⭐ ****5️⃣ OLED setup****

#define SCREEN\_WIDTH 128

#define SCREEN\_HEIGHT 64

👉 OLED resolution

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, -1);

👉 Creates OLED display object

# ⭐ ****6️⃣ Push button****

#define BUTTON\_PIN D7

👉 SOS button connected to D7

bool buttonPressed = false;

unsigned long lastPressTime = 0;

👉 Used for debounce (avoid multiple presses)

# ⭐ ****7️⃣ Setup function****

Serial.begin(115200);

delay(1000);

👉 Starts serial monitor

pinMode(BUTTON\_PIN, INPUT\_PULLUP);

👉 Button uses internal pull-up resistor

gpsSerial.begin(9600);

👉 Starts GPS communication

Wire.begin(D2, D1);

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

👉 Initializes OLED

showStartupScreen();

👉 Displays startup message

connectWiFi();

👉 Connects to WiFi

client.setInsecure();

👉 Allows HTTPS without certificate

# ⭐ ****8️⃣ Loop function****

while (gpsSerial.available()) {

gps.encode(gpsSerial.read());

}

👉 Continuously reads GPS data

if (digitalRead(BUTTON\_PIN) == LOW && millis() - lastPressTime > 1000)

👉 Detects SOS button press

lastPressTime = millis();

sendSOS();

👉 Calls SOS function

# ⭐ ****9️⃣ showStartupScreen()****

Displays:  
👉 Women Safety SOS  
👉 System Starting

# ⭐ ****🔟 connectWiFi()****

Shows “Connecting WiFi” on OLED  
Attempts connection  
If success → displays “WiFi Connected”  
If fail → shows “WiFi Failed”

# ⭐ ****1️⃣1️⃣ sendSOS()****

### 👉 Step 1 — Getting GPS

Shows “Getting GPS”  
Waits up to 10 sec for GPS fix

### 👉 Step 2 — If GPS not fixed

Shows “GPS Not Fixed”

### 👉 Step 3 — If GPS fixed

float lat = gps.location.lat();

float lng = gps.location.lng();

👉 Gets latitude & longitude

### 👉 Step 4 — Sending alert

Shows “Sending SOS”

String message = "🚨 WOMEN SAFETY SOS ALERT!";

👉 Creates SOS message

message += "https://maps.google.com/?q=" + lat + "," + lng;

👉 Creates Google Maps link

### 👉 Step 5 — Send Telegram

bool sent = bot.sendMessage(CHAT\_ID, message, "");

👉 Sends alert

### 👉 Step 6 — Confirmation

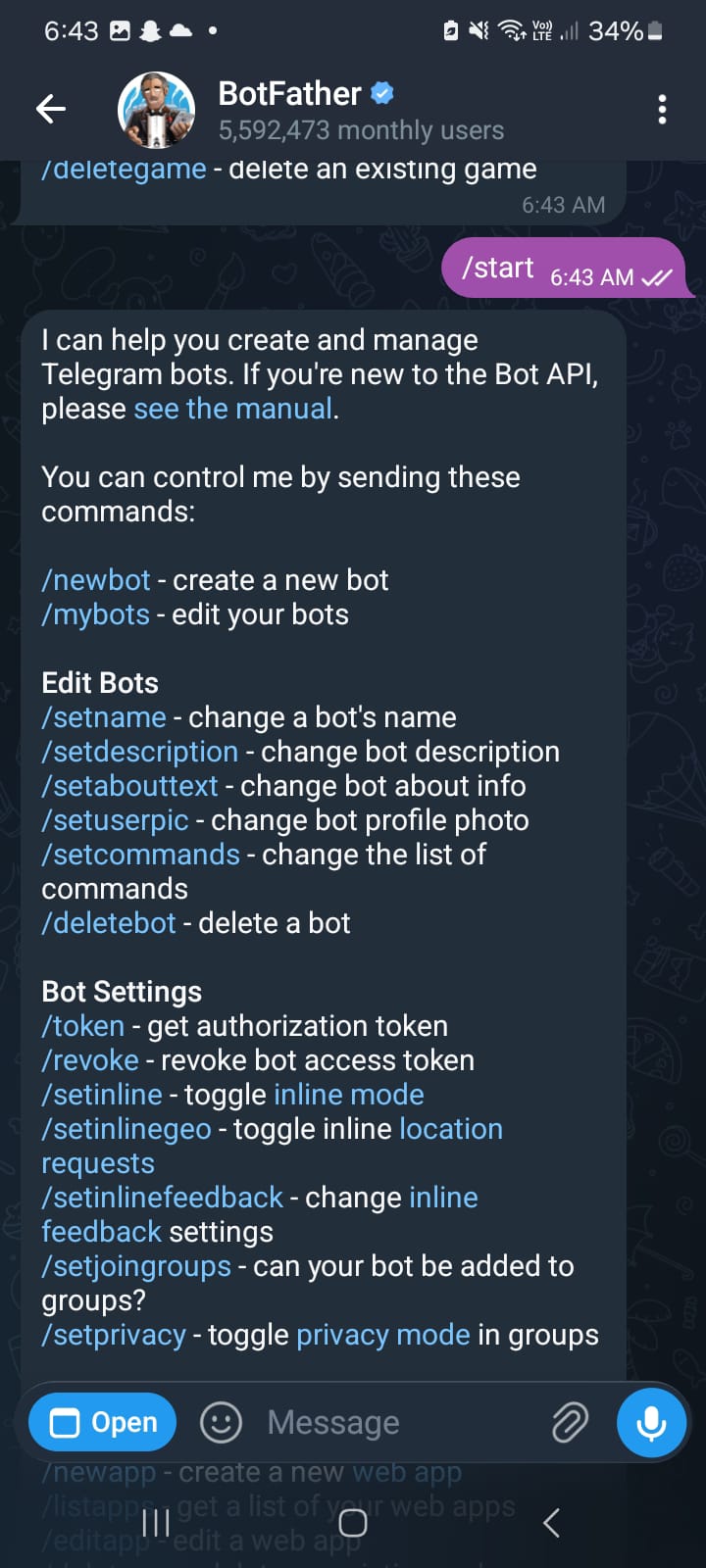
If sent → “Location Sent”  
Else → “Send Failed”

**SYSTEM TESTING And RESUlT**

**Step 8:**

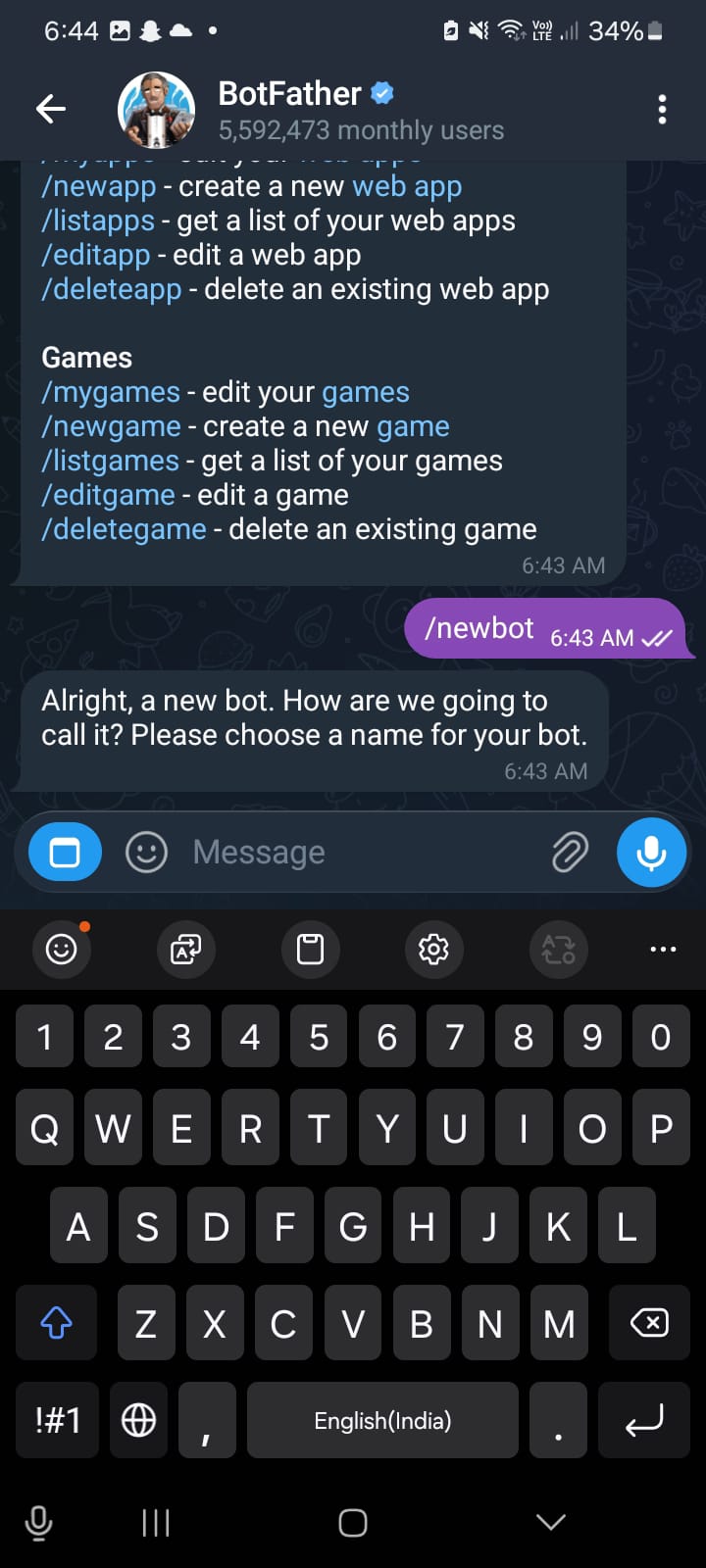
Setup in telegram:

# STEP 1 — Create Telegram Bot

1️⃣ Open Telegram  
2️⃣ Search 👉 **@BotFather**  
3️⃣ Start chat  
:

5️⃣ Then type:

/newbot



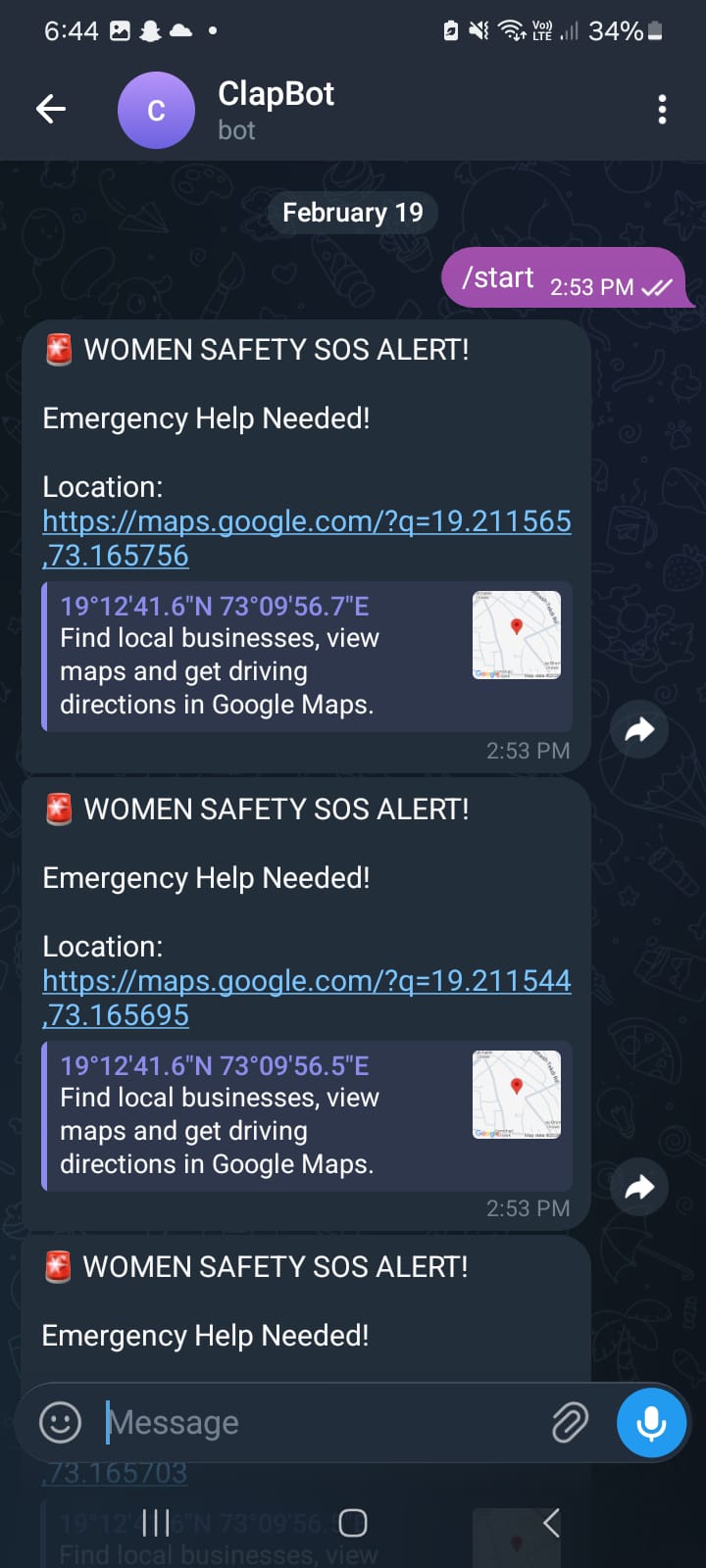
6️⃣ Enter:

* Bot name (any)
* Username (must end with **bot**)  
  Example: ClapBot

👉 BotFather will give **BOT TOKEN**

⭐ Copy this token → paste in code:

#define BOT\_TOKEN "your\_token"



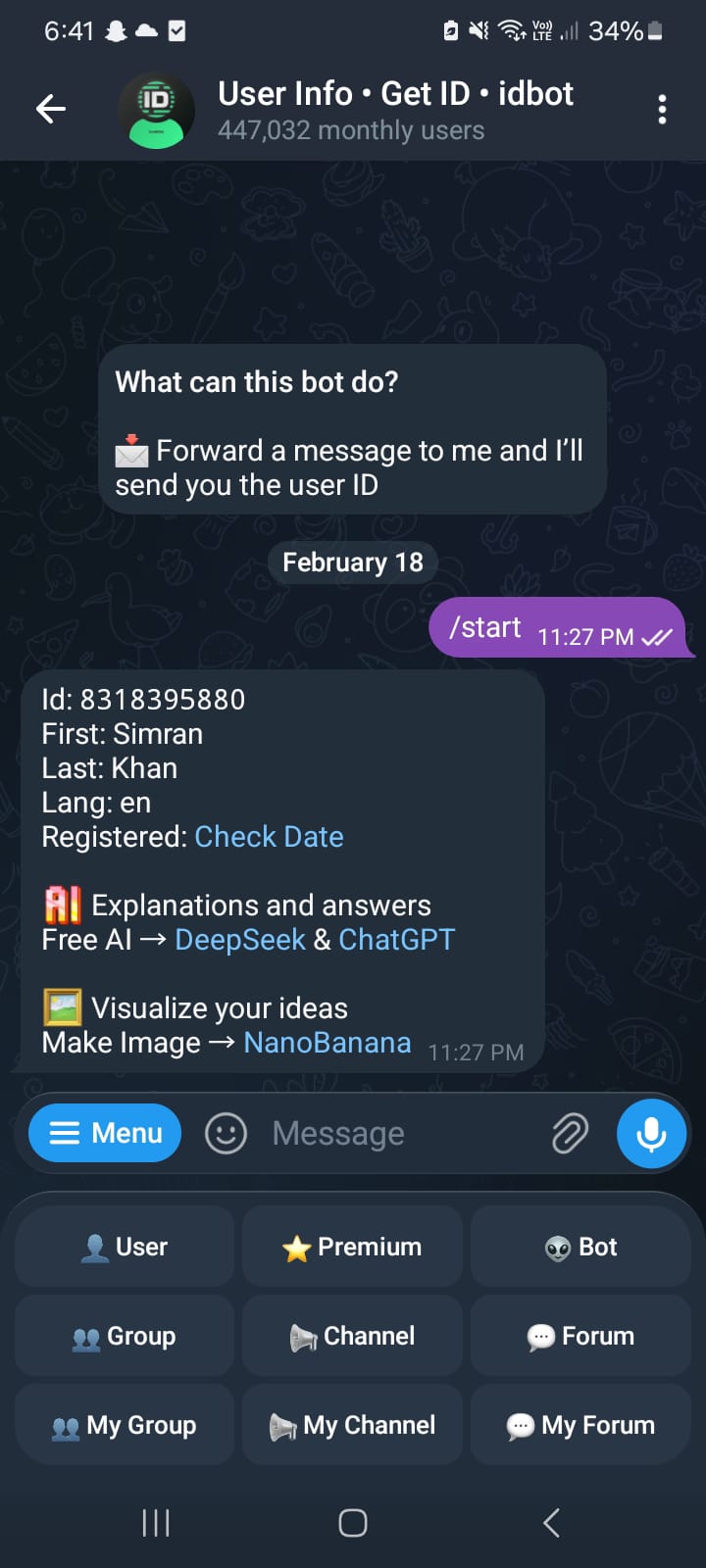
# STEP 2 — Get Chat ID

Now you need your Telegram **CHAT ID**

1️⃣ Search 👉 **@userinfobot**  
2️⃣ Click start  
3️⃣ It will show your **Chat ID**

⭐ Copy and paste:

#define CHAT\_ID "your\_chat\_id"

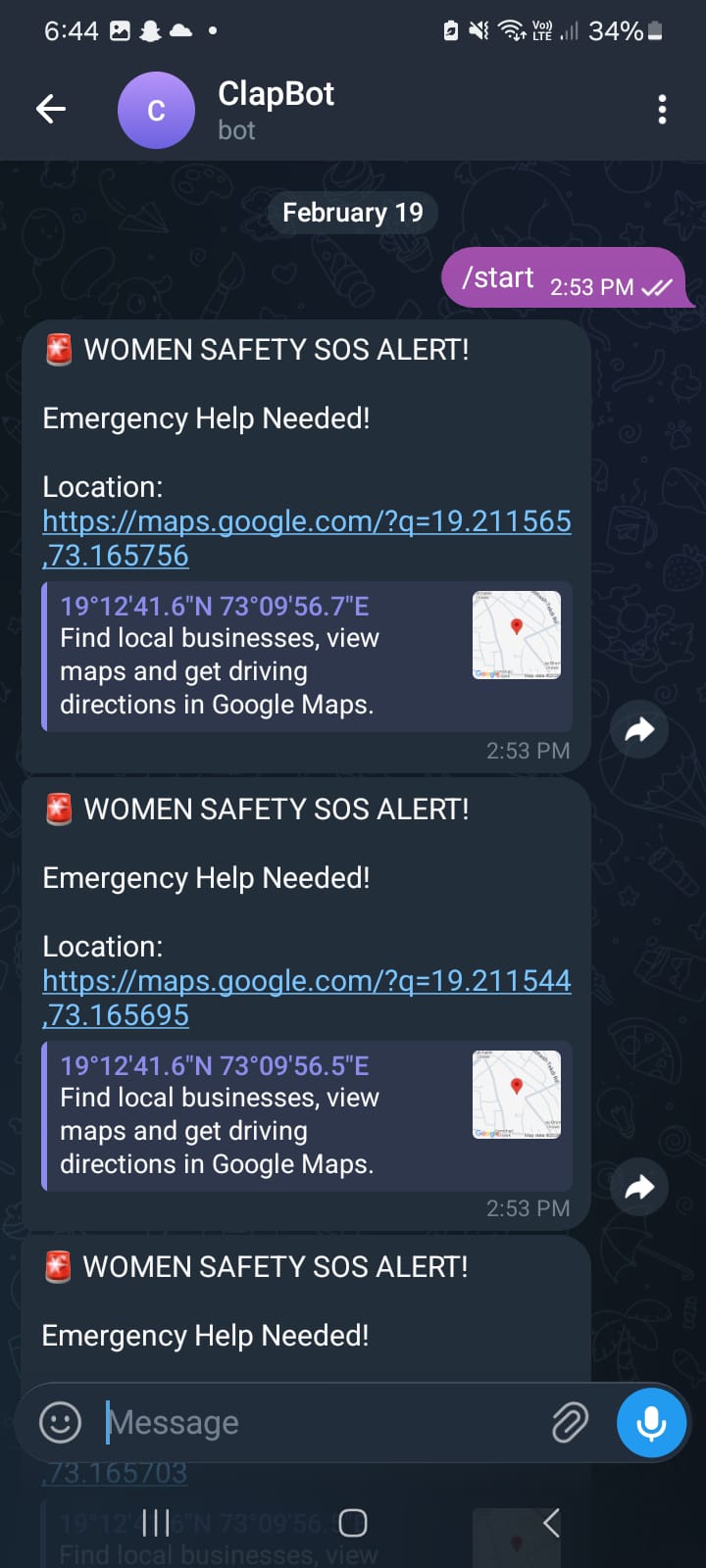


# STEP 3 — Start your bot

Before ESP can send message:

👉 Open your bot in Telegram  
👉 Click **START**

🚨 If you don’t press start → ESP cannot send messages



**Step 9:**

Install Arduino and Install libraries

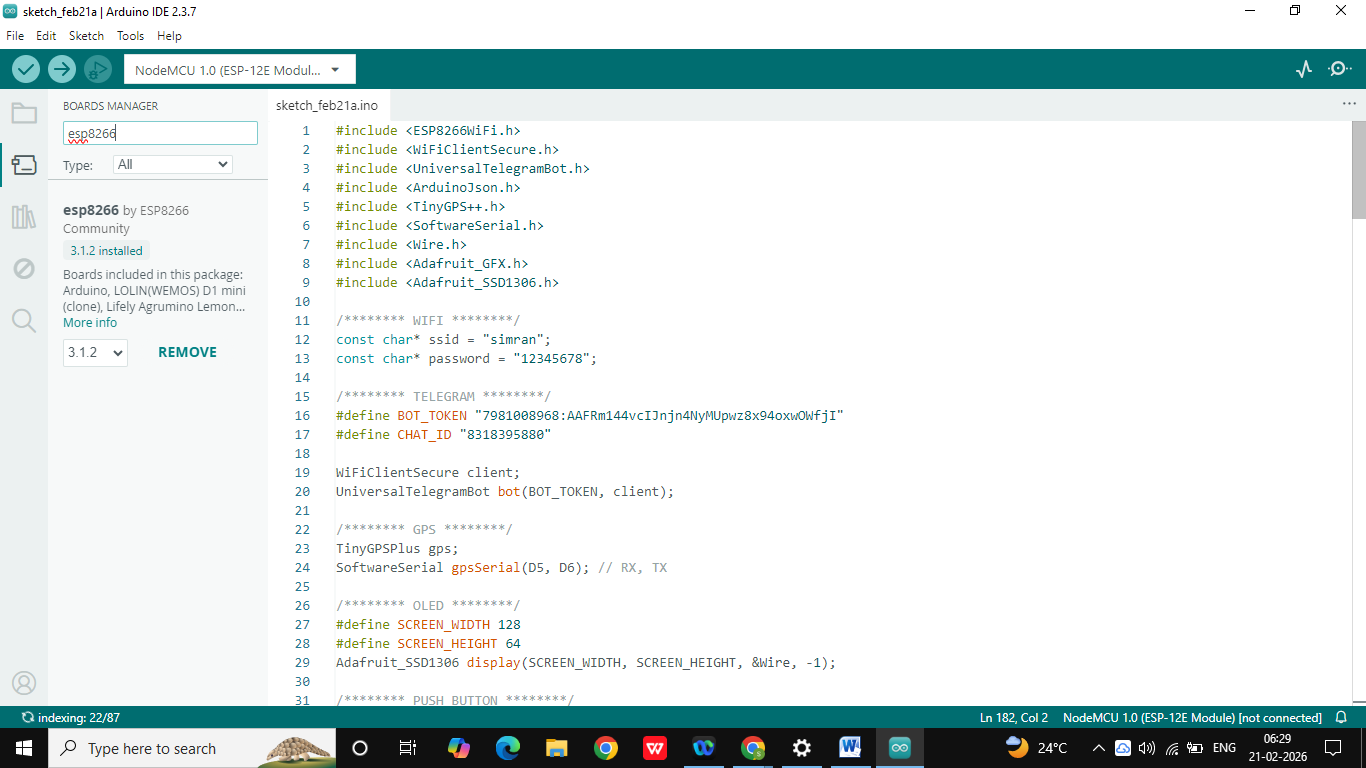
### 1. WiFi (ESP8266 core)

* ESP8266WiFi.h
* WiFiClientSecure.h

📌 **How:**  
Install **ESP8266 board package**, not a library.

* Go to **Tools → Board → Boards Manager**
* Search **ESP8266**
* Install **ESP8266 by ESP8266 Community**

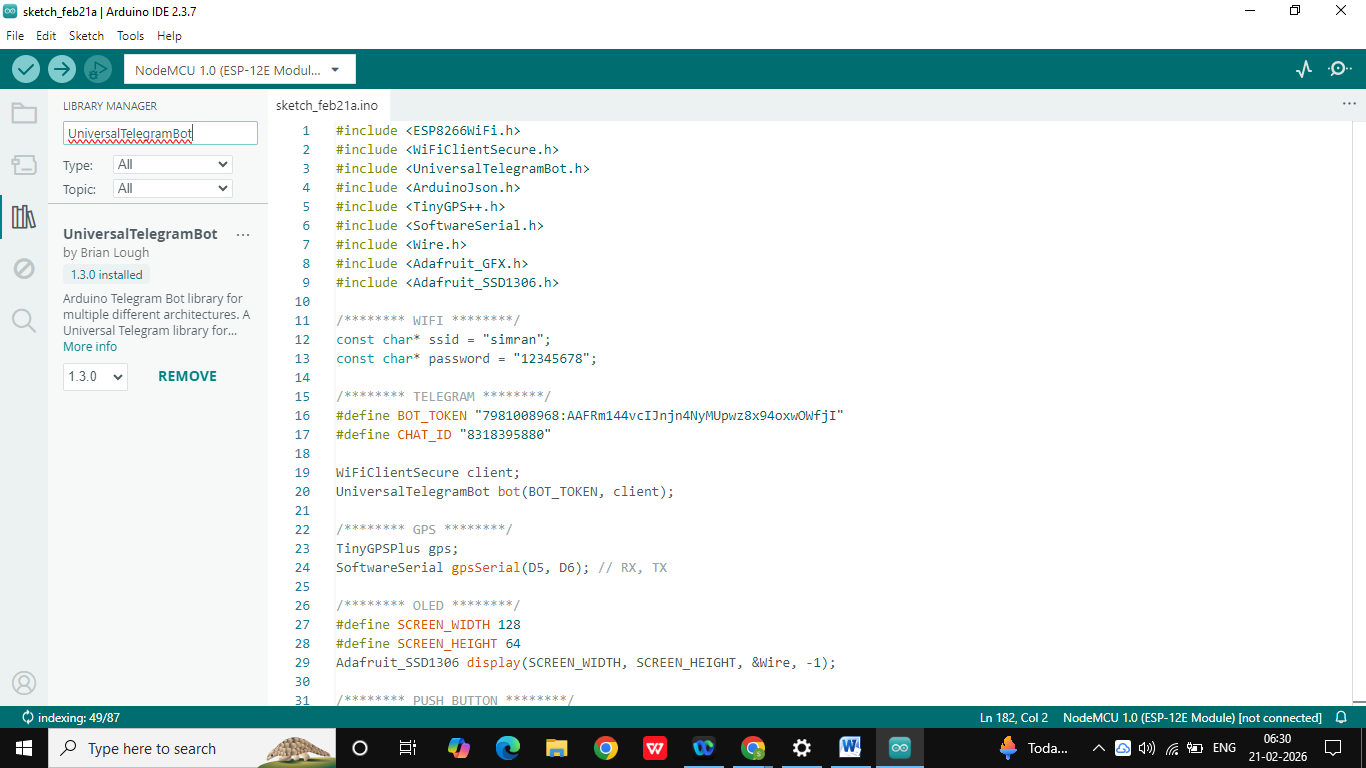
This automatically gives both WiFi libraries.



### 2. Telegram bot

* **UniversalTelegramBot** → by Brian Lough

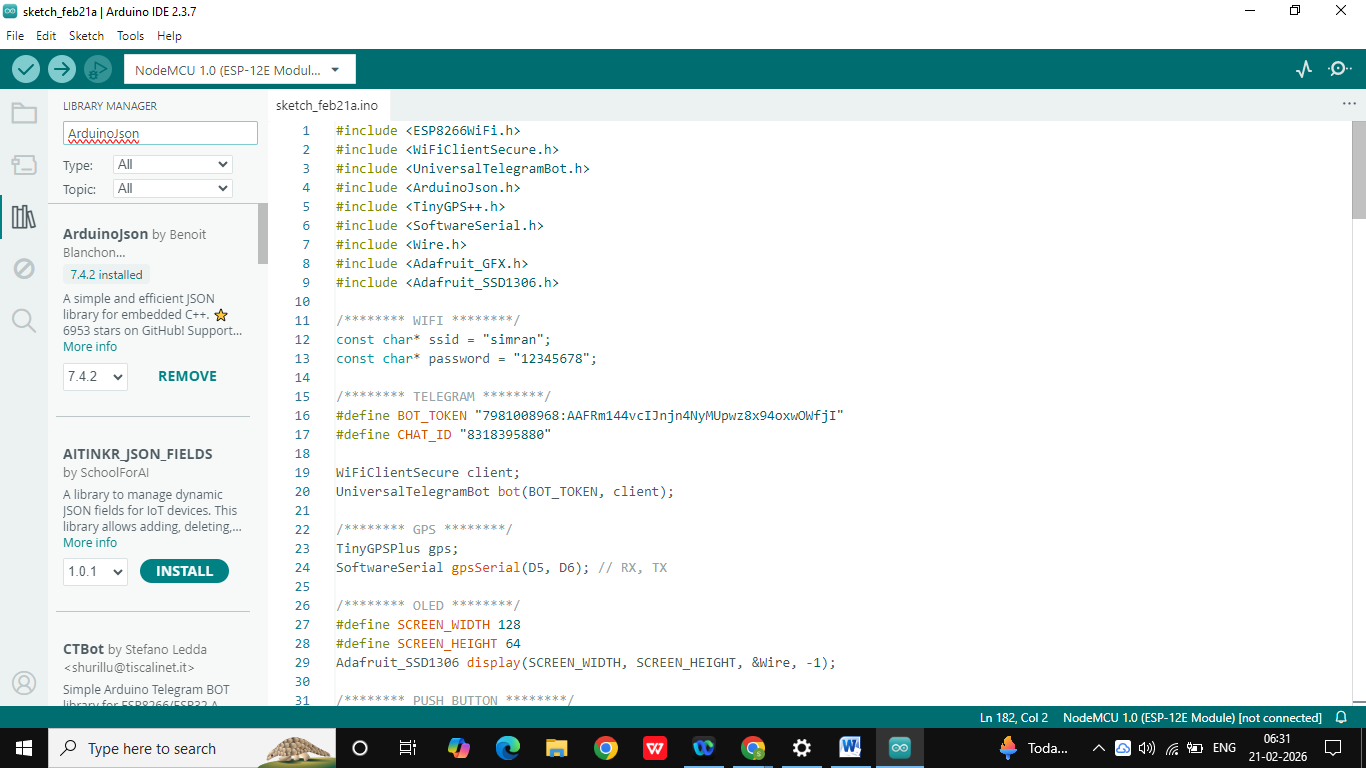
Search: UniversalTelegramBot



### 3. JSON parsing

* **ArduinoJson** → by Benoit Blanchon

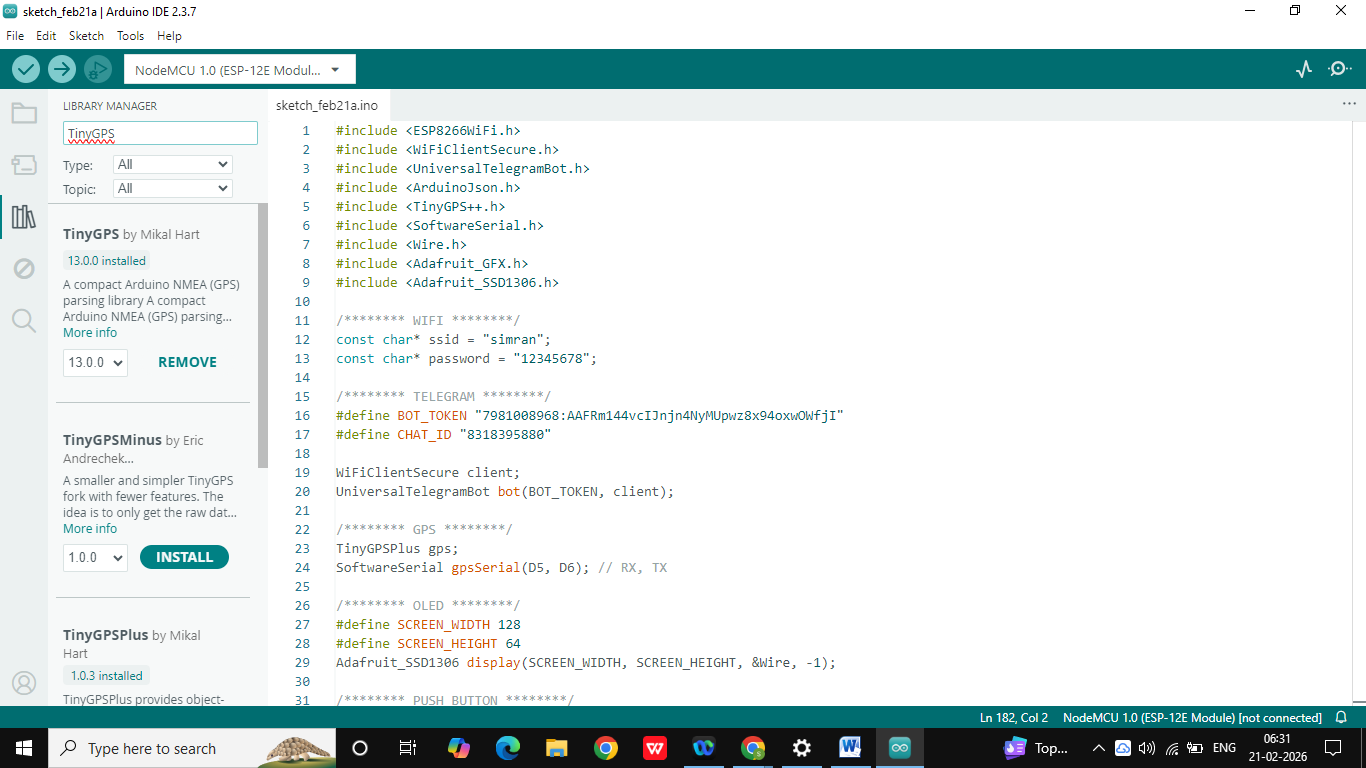
Search: ArduinoJson



### 4. GPS

* **TinyGPSPlus** → by Mikal Hart

Search: TinyGPSPlus



### 5. Software serial

* SoftwareSerial.h

📌 Already built-in (no need to install)

### 6. I2C communication

* Wire.h

📌 Built-in (no need to install)

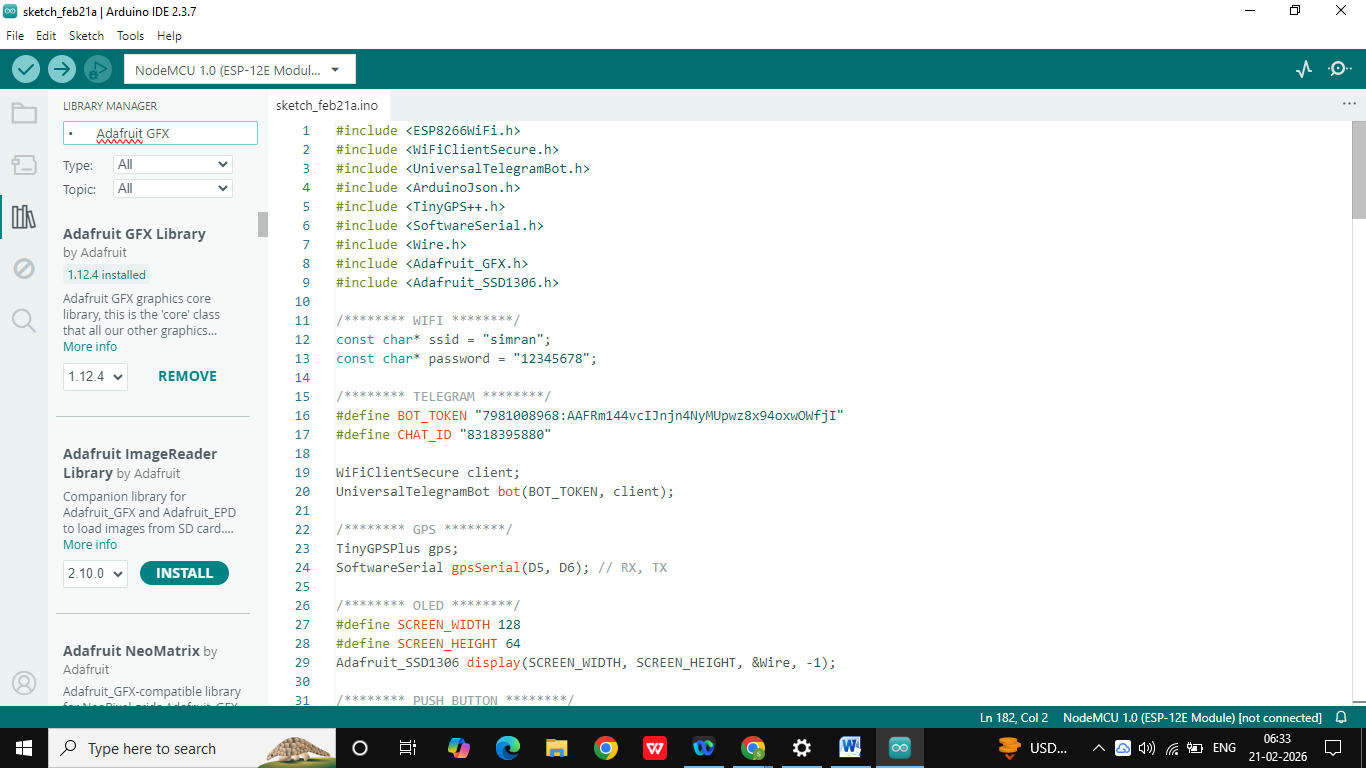
### 7. OLED display

* **Adafruit SSD1306**
* **Adafruit GFX Library** (dependency)

Search:

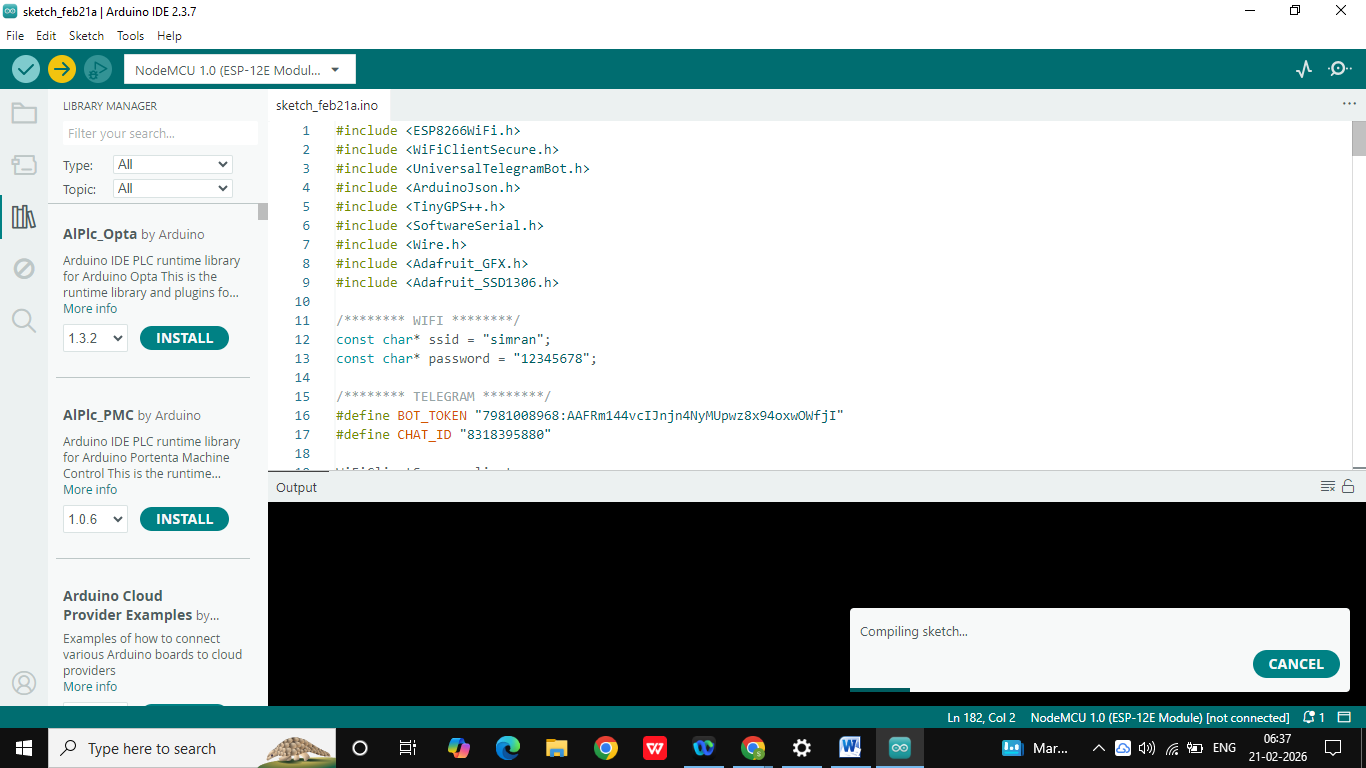
* Adafruit SSD1306
* Adafruit GFX





**Step 10:**

Upload the Code:



**Step 11:**

The oled screen will show

****

**FUTURE SCOPE AND CONCLUSION**

### ****Future Scope — Women Safety Tracking Device****

1. The system can be upgraded by integrating a GSM module to enable SOS alerts even without WiFi connectivity.
2. Wearable designs such as smart bands, pendants, or smart jewelry can be developed for better portability and convenience.
3. Additional sensors like accelerometer and fall detection modules can be added to automatically detect emergencies.
4. AI-based threat detection and voice activation features can be implemented to enhance safety capabilities.
5. A mobile application interface can be developed for real-time monitoring and device configuration.
6. Integration with police or emergency service databases can improve response time and effectiveness.

### ✅ ****Conclusion — Women Safety Tracking Device****

The Women Safety Tracking Device is a compact and reliable IoT-based solution designed to enhance personal security through instant SOS alerts and real-time location tracking. By utilizing ESP8266, GPS, OLED display, and a push button, the system enables quick emergency activation and accurate location sharing via Telegram. The device is easy to use, cost-effective, and portable, making it suitable for everyday safety purposes.

Testing results demonstrate that the system successfully sends emergency alerts with location information, ensuring faster response during critical situations. Although the device relies on WiFi connectivity, it provides a strong foundation for future improvements and advanced safety features. Overall, the project contributes to improving women safety by offering a practical and efficient emergency alert solution that promotes confidence and independence.

**REFERENCES**

### ****References — Women Safety Tracking Device****

1. Arduino Official Website — Documentation and resources for ESP8266 programming  
   <https://www.arduino.cc>
2. ESP8266 NodeMCU Documentation — Technical details and pin configuration  
   <https://nodemcu.readthedocs.io>
3. TinyGPS++ Library Documentation — GPS data processing and usage guide  
   <https://github.com/mikalhart/TinyGPSPlus>
4. Universal Telegram Bot Library — Telegram bot integration with ESP8266  
   <https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot>
5. Adafruit SSD1306 OLED Library — OLED display control and graphics support  
   <https://github.com/adafruit/Adafruit_SSD1306>
6. ArduinoJson Library Documentation — JSON handling for IoT communication  
   <https://arduinojson.org>
7. NEO-6M GPS Module Datasheet — GPS hardware specifications and working  
   https://components101.com/modules/gps-module

**GLOSSARY**

### ****Glossary — Women Safety Tracking Device****

1. **ESP8266:** A low-cost WiFi-enabled microcontroller used as the main controller for processing data and communication.
2. **GPS (Global Positioning System):** A satellite-based navigation system used to determine real-time geographic location.
3. **OLED Display:** A small screen used to show system status such as WiFi connection, GPS fix, and SOS confirmation.
4. **SOS Alert:** An emergency message sent to predefined contacts to request immediate help.
5. **IoT (Internet of Things):** A network of interconnected devices that communicate and exchange data over the internet.
6. **Telegram Bot:** An automated messaging service used to send SOS alerts and location information to contacts.
7. **Push Button:** An input device used to trigger the emergency alert in the system.
8. **Latitude and Longitude:** Geographic coordinates used to represent the exact location of the user.
9. **I2C Communication:** A communication protocol used to connect OLED display with the ESP8266 using two wires (SDA and SCL).
10. **Serial Communication:** A method of data transmission used by the GPS module to send location information to the microcontroller.