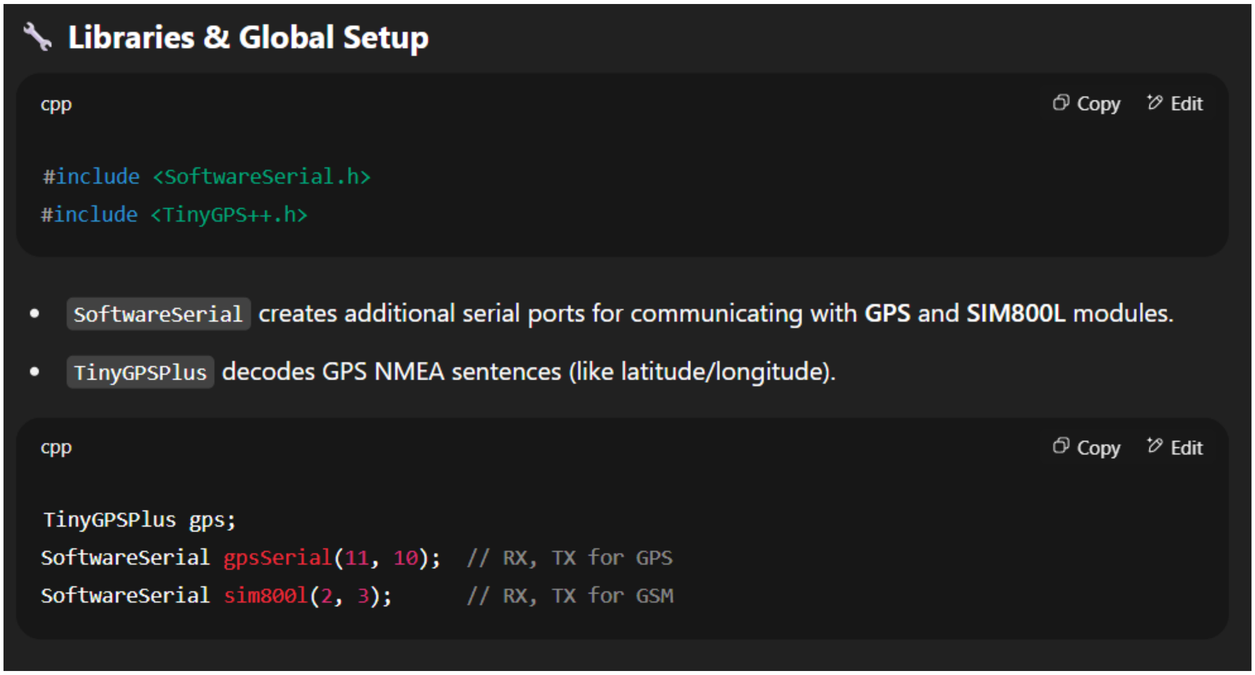
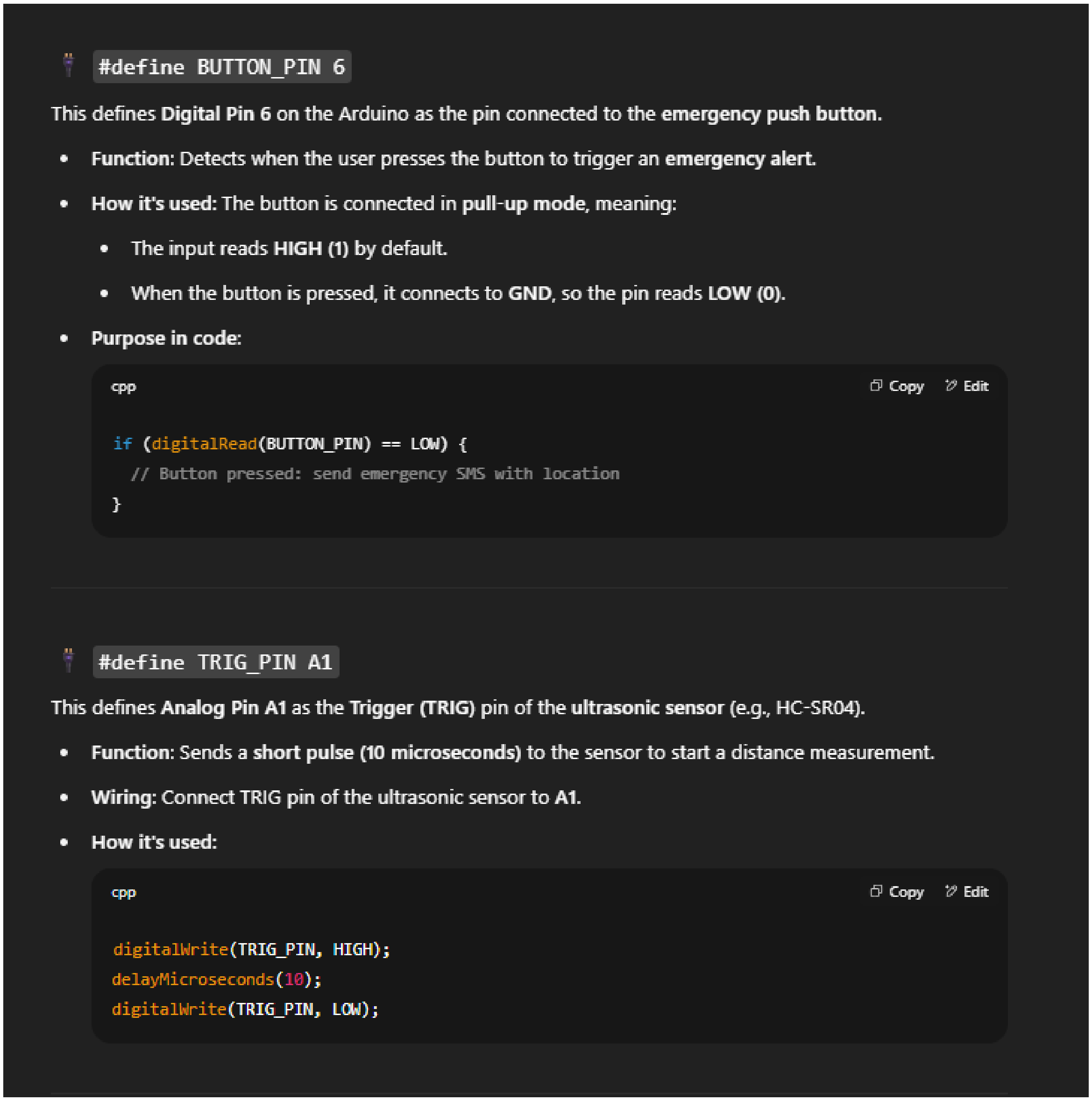


🔧 Libraries & Global Setup

```cpp
#include <SoftwareSerial.h>
#include <TinyGPS++.h>
```

- `SoftwareSerial` creates additional serial ports for communicating with **GPS** and **SIM800L** modules.
- `TinyGPSPlus` decodes GPS NMEA sentences (like latitude/longitude).

```cpp
TinyGPSPlus gps;
SoftwareSerial gpsSerial(11, 10);  // RX, TX for GPS
SoftwareSerial sim800l(2, 3);      // RX, TX for GSM
```



`#define BUTTON_PIN 6`

This defines **Digital Pin 6** on the Arduino as the pin connected to the **emergency push button.**

- **Function:** Detects when the user presses the button to trigger an **emergency alert.**
- **How it's used:** The button is connected in **pull-up mode**, meaning:
  - The input reads **HIGH (1)** by default.
  - When the button is pressed, it connects to **GND**, so the pin reads **LOW (0)**.
- **Purpose in code:**

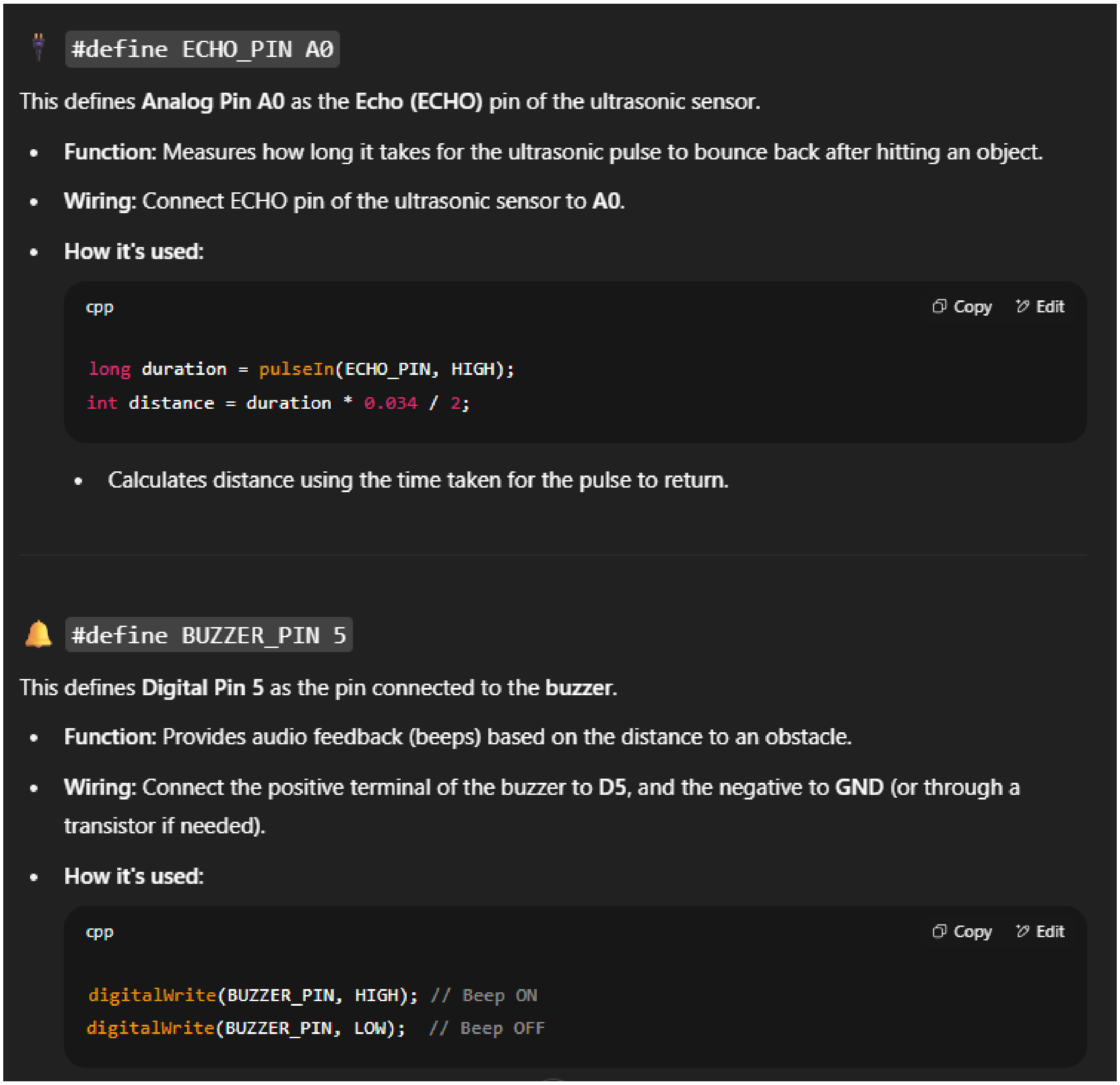
```cpp
if (digitalRead(BUTTON_PIN) == LOW) {
  // Button pressed: send emergency SMS with location
}
```

`#define TRIG_PIN A1`

This defines **Analog Pin A1** as the **Trigger (TRIG)** pin of the **ultrasonic sensor** (e.g., HC-SR04).

- **Function:** Sends a **short pulse (10 microseconds)** to the sensor to start a distance measurement.
- **Wiring:** Connect TRIG pin of the ultrasonic sensor to **A1**.
- **How it's used:**

```cpp
digitalWrite(TRIG_PIN, HIGH);
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);
```



🔌 `#define ECHO_PIN A0`

This defines **Analog Pin A0** as the **Echo (ECHO)** pin of the ultrasonic sensor.

- **Function:** Measures how long it takes for the ultrasonic pulse to bounce back after hitting an object.
- **Wiring:** Connect ECHO pin of the ultrasonic sensor to **A0**.
- **How it's used:**

```cpp
long duration = pulseIn(ECHO_PIN, HIGH);
int distance = duration * 0.034 / 2;
```

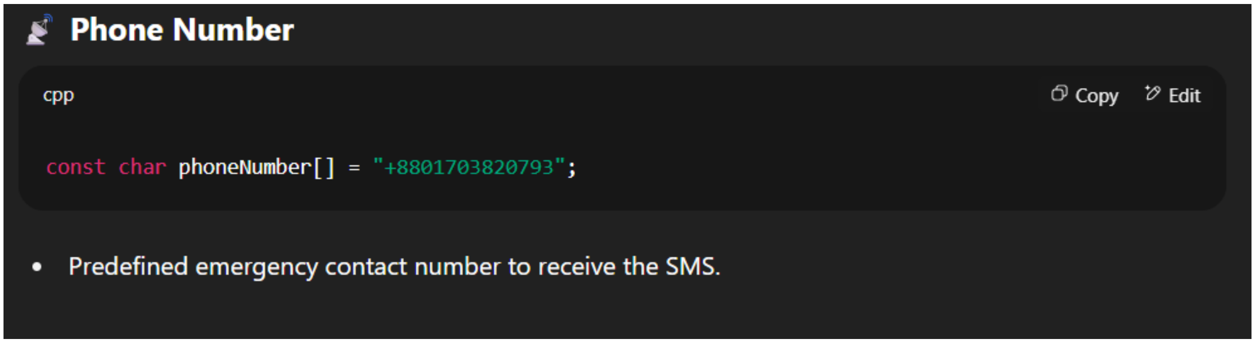
  - Calculates distance using the time taken for the pulse to return.

🔔 `#define BUZZER_PIN 5`

This defines **Digital Pin 5** as the pin connected to the **buzzer**.

- **Function:** Provides audio feedback (beeps) based on the distance to an obstacle.
- **Wiring:** Connect the positive terminal of the buzzer to **D5**, and the negative to **GND** (or through a transistor if needed).
- **How it's used:**

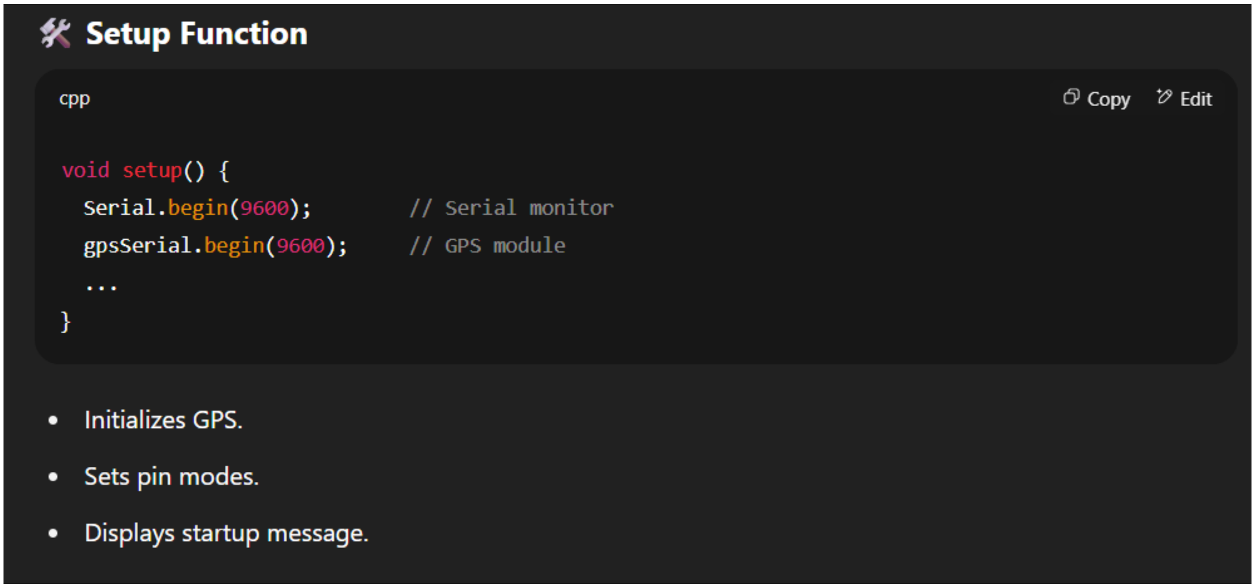
```cpp
digitalWrite(BUZZER_PIN, HIGH); // Beep ON
digitalWrite(BUZZER_PIN, LOW);  // Beep OFF
```



📡 Phone Number

```cpp
const char phoneNumber[] = "+8801703820793";
```

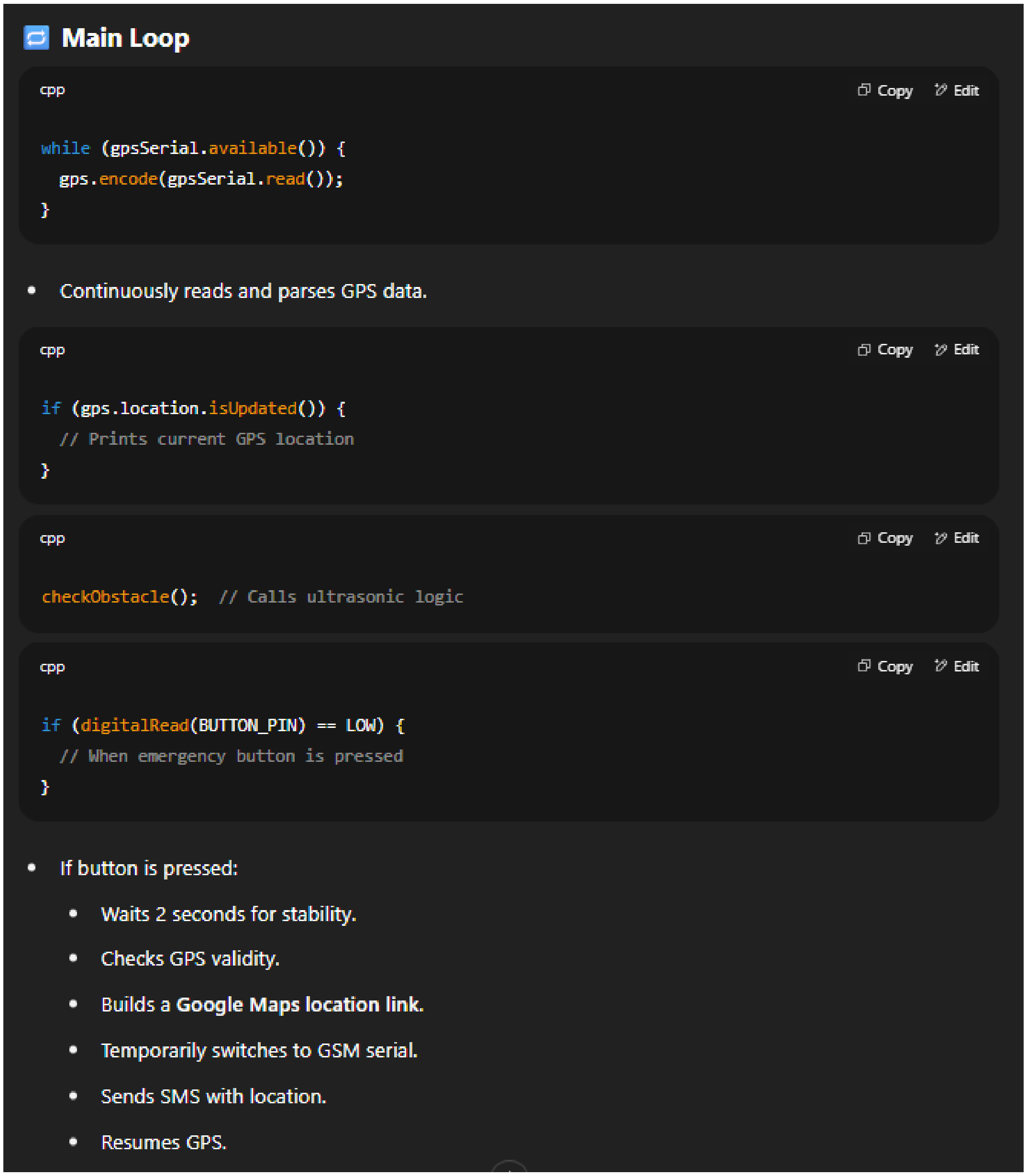
- Predefined emergency contact number to receive the SMS.



🛠️ Setup Function

```cpp
void setup() {
  Serial.begin(9600);         // Serial monitor
  gpsSerial.begin(9600);      // GPS module
  ...
}
```

- Initializes GPS.
- Sets pin modes.
- Displays startup message.



🔁 Main Loop

```cpp
while (gpsSerial.available()) {
  gps.encode(gpsSerial.read());
}
```

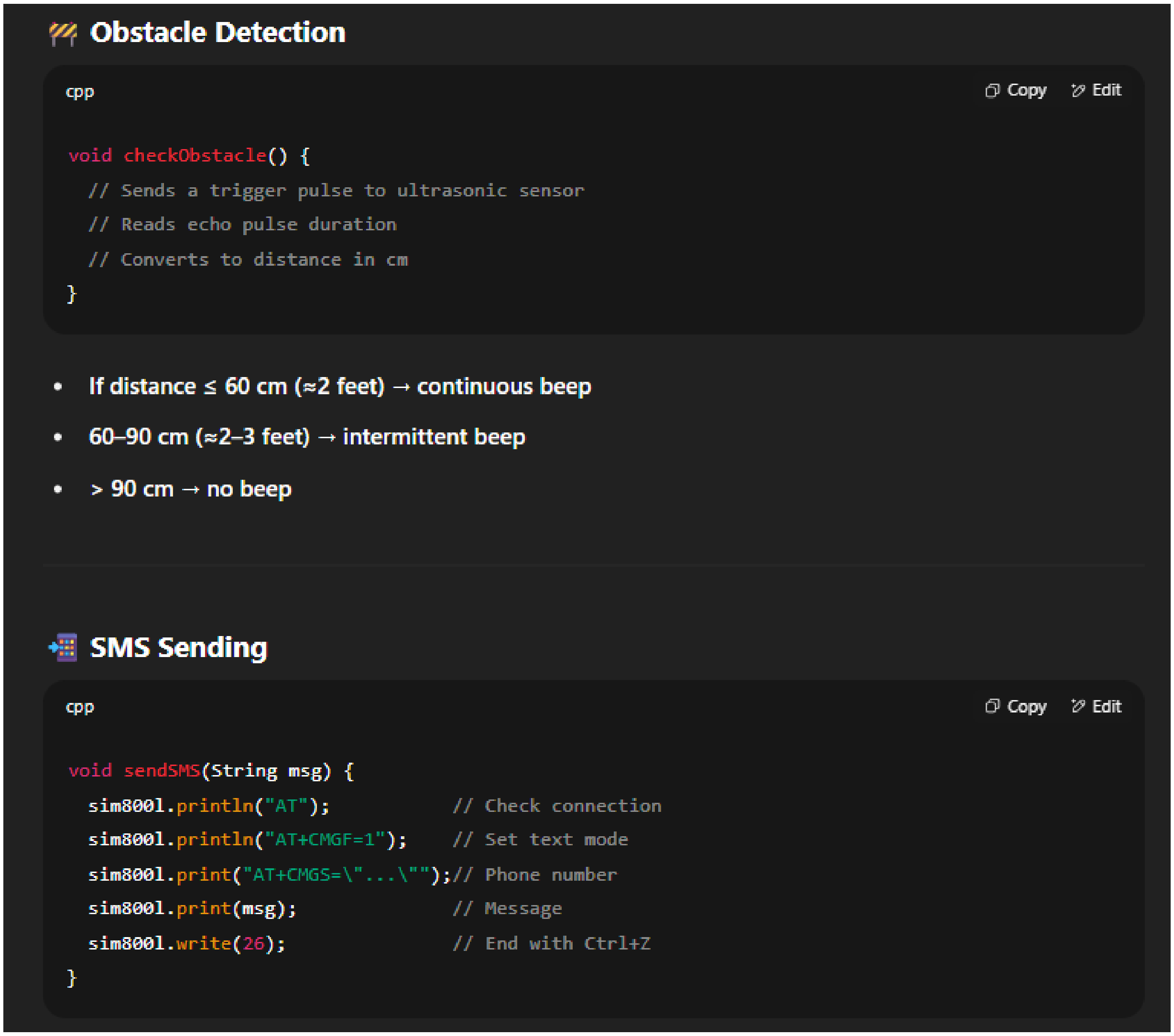
- Continuously reads and parses GPS data.

```cpp
if (gps.location.isUpdated()) {
  // Prints current GPS location
}
```

```cpp
checkObstacle();  // Calls ultrasonic logic
```

```cpp
if (digitalRead(BUTTON_PIN) == LOW) {
  // When emergency button is pressed
}
```

- If button is pressed:
  - Waits 2 seconds for stability.
  - Checks GPS validity.
  - Builds a **Google Maps location link.**
  - Temporarily switches to GSM serial.
  - Sends SMS with location.
  - Resumes GPS.



🚧 Obstacle Detection

```cpp
void checkObstacle() {
  // Sends a trigger pulse to ultrasonic sensor
  // Reads echo pulse duration
  // Converts to distance in cm
}
```

- If distance ≤ 60 cm (≈2 feet) → continuous beep
- 60–90 cm (≈2–3 feet) → intermittent beep
- > 90 cm → no beep

📲 SMS Sending

```cpp
void sendSMS(String msg) {
  sim800l.println("AT");           // Check connection
  sim800l.println("AT+CMGF=1");    // Set text mode
  sim800l.print("AT+CMGS=\"...\"");// Phone number
  sim800l.print(msg);              // Message
  sim800l.write(26);               // End with Ctrl+Z
}
```