

Design: Guardian Trace

Yacob Alkarim

April 2026

1 Introduction

The increasing number of elderly individuals living with dementia has created a growing demand for technologies that improve patient safety and caregiver communication. People with dementia are particularly vulnerable to wandering and becoming lost, placing both the individual and their caregivers under significant stress. Existing products such as AirTags and Smart Tags provide basic tracking functions; however, they often lack features specifically designed for vulnerable users requiring continuous monitoring and communication support (Moon et al.2019).

This project, titled Guardian Trace, aims to develop a wearable GPS tracking and communication device designed to support individuals who are prone to wandering. The primary objective of the system is to provide real-time location tracking while improving communication between caregivers and patients. The device uses an Arduino Nano microcontroller integrated with a GPS module and a SIM800L EVB GSM communication module to collect and transmit location data to a mobile application.

A major focus of the project was creating a compact, lightweight, and cost-effective design suitable for everyday use. Additional objectives included selecting reliable components, designing a protective enclosure, and ensuring compatibility between the hardware and software systems. The project also investigated important engineering challenges such as GSM network reliability, power management, and system integration.

2 Team Members Roles

Yacob Alkarim - Hardware and project manager

As the hardware lead, I was responsible for integrating and validating the core components of the GPS tracking system, including the Arduino Nano, SIM800L EVB GSM module, and GPS module. I configured serial communication between devices, ensured proper wiring, and designed a stable power setup using external power sources and voltage regulation to meet the SIM800L's high current requirements. I would then work with the Amani and Nariman to make sure that the app was working with the gps tracking module. And with Nariman as the lead design I was helping in seeing if the design was meeting our specification and that the design wouldn't hinder the components in working well together.

Amani -

Nariman - The Guardian Trace project was completed as a collaborative team project in which each member had specific responsibilities. Yacob Alkarim was responsible for project management and hardware development, including coordinating tasks and supporting hardware integration. Amani Mohsin focused on software development and project documentation, including collaboration planning and application research. My

role was focused on 3D design and health and safety management. My responsibility was to design a practical enclosure that could securely contain all electronic components while remaining comfortable, accessible, and safe for everyday use. This involved measuring components, planning the internal layout, creating AutoCAD models, producing prototypes, and evaluating usability. I also considered several safety factors throughout the design process, including wiring organisation, short-circuit prevention, battery protection, rounded edges, and accessibility for vulnerable users.

3 Hardware information

Most aspects of the previous report remain unchanged; however, several modifications have been made in an effort to better achieve the project's primary objective

3.1 New parts

Sim800L EVB Data transmits module. This module works off a GSM (Global System for Mobile Communication). Which then divides signal frequencies into multiple time slots, allowing many users to share a single channel via multiplexing. The sim800L EVB functions to send sensor data in JSON format to a server with MQTT protocol (Sugiyanti, 2019).

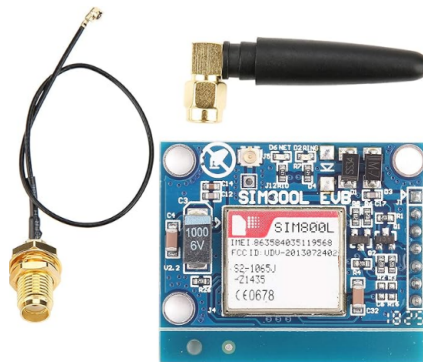


Figure 1: Sim800L EVB

The sim800L EVB works by using sending data through 2G data. The reason I went ahead with was because I was using a 2G sim from Vodafone to transmit the data. When doing research on whether the Vodafone still has 2G antennas connect in Swansea. It showed me a map showing were 2G is still available in Swansea.



Figure 2: Vodafone 2G Coverage in swansea

The reason I didn't go with a simple sim800L and rather with a sim800L EVB is working with the SIM800L presents several technical challenges due to its strict power requirements and sensitivity to power instability. The module requires a stable operating voltage between 3.4V and 4.4V and can draw current peaks approaching 2A during GSM transmission bursts, making it difficult to power reliably using standard micro-controller supplies or USB sources (SIMCom, 2014).

3.2 Removed parts

The SD card reader was initially intended to provide a way to review the tracker's recorded locations. However, due to limited understanding during the planning stage, it was not fully considered that the SD card would need to be physically removed and the stored data manually accessed, for example through Google Maps. This approach did not align with the project's primary objective of active, real-time tracking, as it would require locating the device, removing the SD card, and extracting the data before the tracking information could be reviewed.

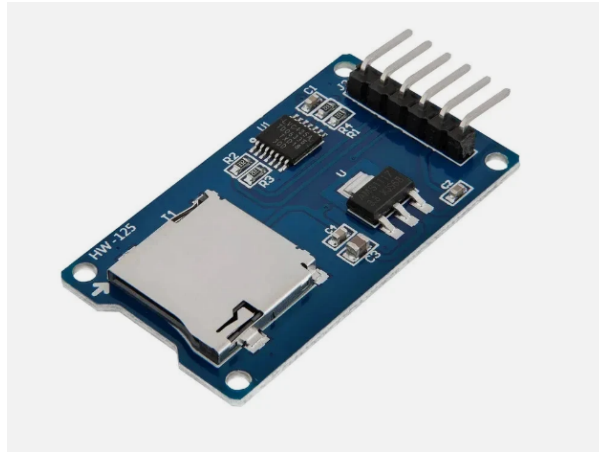


Figure 3: SD card reader

4 Developing Arduino Hardware

Description:

In the section of the report I will be going over how the GPS tracking unit was created any alteration from the semester 1 report. And any changes that came about and how it was handled.

4.1 Creating the GPS tracking unit

Explanation of the code is in the Appendix.

Figure 2 shows the data the GPS tracking module is collecting. Note there is something called "GPS drift" this is when the GPS module isn't actually detecting motion. It is however recalculating its position every second. And due to the inherent errors in the signal, the calculated position "jumps" slightly around the true location, Resulting in different latitude, longitude, speed and altitude.

When trying to make the GPS tracking module what I was trying to accomplish was read GPS data then send this data via the SIM800L EVB to the app and then be able to do the active tracking. The first of many things was that the SIM800L EVB was giving my random characters in the serial monitor to fix this I had to put in the correct baud rate which was 115200. To check this I used a list of AT commands that I would be able to speak to and gather data from the SIM800L (Das, Abhishek, et al (2023)). At first I put the AT command, AT and it would give me a response of OK. This would allow me to confirm that the SIM800L EVB was able to communicate to the Arduino Nano.

My next idea into why there might have been an issue was the power. At first I was connecting it with a USB port and that I was using it while the GU-T7. So, I separated the components to check that and there was still nothing happening. However, looking on the data sheet of the SIM800L EVB I noticed that it was very power hungry needing a pulse of 2 Ah. While the Nano was only able to supply it with around 50mAh. I tried

```

1  #include "TinyGPS++.h"      // Softwear library
2  #include "SoftwareSerial.h"
3
4  SoftwareSerial serial_connection(3, 4); // RX=pin 3, TX=pin 4
5  TinyGPSPlus gps;           // Note: Class name has ONE plus sign
6
7  void setup()
8  {
9      Serial.begin(9600);
10     serial_connection.begin(9600);
11     Serial.println("GPS Start");
12 }
13
14 void loop()
15 {
16     while(serial_connection.available())
17     {
18         gps.encode(serial_connection.read());
19     }
20
21     if(gps.location.isUpdated())
22     {
23         Serial.println("Satellite Count:");
24         Serial.println(gps.satellites.value());
25         Serial.println("Latitude:");
26         Serial.println(gps.location.lat(), 6);
27         Serial.println("Longitude:");
28         Serial.println(gps.location.lng(), 6);
29         Serial.println("Speed MPH:");
30         Serial.println(gps.speed.mph());
31         Serial.println("Altitude Feet:");
32         Serial.println(gps.altitude.feet());
33         Serial.println("");
34     }
35 }
36

```

Figure 4: GPS-tracking code

this however nothing seem to change it was still not connecting to anything and was able to send the data. And i was able to use tinkercad to check if there was and issue with my power as tinkercad doesn't have sim800L EVB featurer in it.

After some tinkering I noticed that the i had placed into of sim800L EVB was a 5G/4G sim and wasn't able to go to 2G network I tested this by putting in the command AT+CPIN? this would read out as READY meaning the sim was operate. After time I got hold of a LEBARA sim card that was able to support 2G networks. However this only yielded me the same results. To check with-in the arduino software the command was AT+READY this would give me a result of READY. This would check that the sim

```
Satellite Count:
7
Latitude:
51.634136
Longitude:
-3.977529
Speed MPH:
0.46
Altitude Feet:
375.00
```

Figure 5: TRacking Data

```
Output  Serial Monitor X
Message (Enter to send message to 'Arduino Nano' on 'COM3')
AT
OKC
```

Figure 6: AT command

wasn't the issue.

Moving forward the idea of it not being able to connect to any 2G networks was the probable cause. To check if it was a connection is there was a command of AT+CSQ this would show how strong there signal is. This would give me random result varying from 13 to 0. I tried to leave the application for around 2 hours maybe during that time allowing it to connect to something. However it would lead to the same problem. To know if the sim800L EVB was connected to a antenna there would be a red light on the module if the red light was to flash in increments of 1 second that means its either looking or on a weak signal. However if the red light was to flash in increments of 3 second this mean that the signal was strong and you were able to transmit data with it.

```
Output  Serial Monitor X
Message (Enter to send message to 'Arduino Nano' on 'COM3')
AT+CSQ
+CSQ: 7,0
OK
```

Figure 7: AT+CSQ results

Another command that I tried to use was the AT+CREG? this would allow me to see if the signal was connected to anything in the first place. However i was only ever

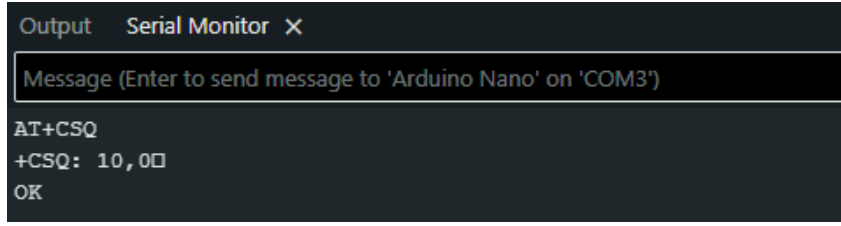


Figure 8: AT+CSQ results

getting +CREG:0,0. This mean that it was not registering to a network.

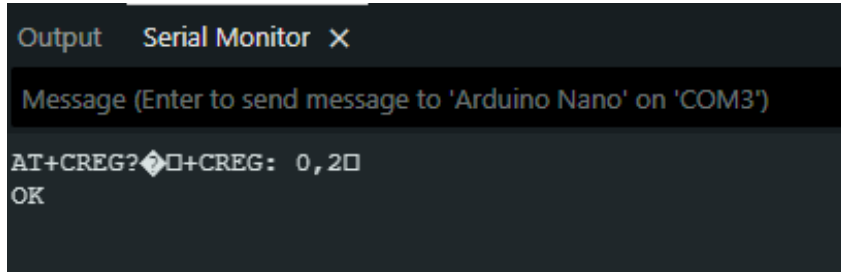


Figure 9: AT+CREG? results

After testing everything the root cause was identified as after testing Sim was working the sim800L was working and the power was acceptable. The real issue was that there was weak or inconsistent 2G signal in my location. What i was able to learn from this that no amount of coding or wiring fixes poor signal.

After more research I assume the issue might've been that there wasn't enough current being supplied to the sim800L EVB. So I got a battery that could supply 7.4V and 2600mAh, however when testing all that would happen would that there would be increase in duration from the sim800L EVB from just re-cycling the reconnecting process. The system would turn on then flash 3 time very quickly (under 1 second), then stay on a bit longer and then just repeat the cycle. This would mean that power source cannot maintain a stable voltage during these transmission bursts, the module may repeatedly restart the network registration process without successfully connecting. In addition, GSM-based embedded systems are known to experience communication instability and registration problems when operating under under voltage conditions or poor network availability, which can significantly reduce reliability in practical applications (Okoronkwo et al., 2013). Therefore, the repeated flashing behavior observed on the SIM800L EVB most likely indicates that the module was unable to establish and maintain network registration due to electrical instability or communication constraints associated with GSM connectivity.

Looking at the information provided by the UK parliament, on the information of switching of 2G and 3G in the UK. It states that 2G is still widely available in the UK and doesn't have a set timetable on switching of the 2G network. With this information



Figure 10: sim800L EVB no blink

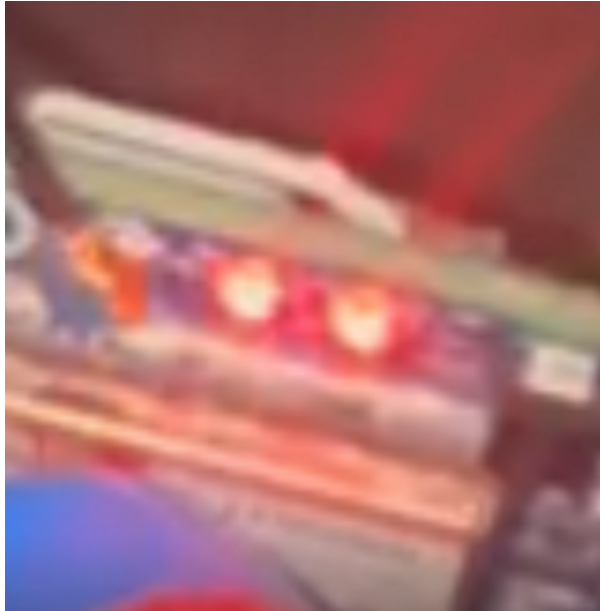


Figure 11: Sim800L EVB blinking

and the information provided in figure 2 on the network covers in Swansea. The only real issue would seem to be the power instability. I think the most likely root cause of failure was that the SIM800L was not receiving a stable enough power supply to handle the high current bursts required during transmission. This caused the module to repeatedly reset and prevented it from reliably registering to the network. To fix this the best course of action would be finding the right way in powering it with enough voltage and current. Or, probably update the GSM model to a 4G capable model.

5 Final design



Figure 12: Final design

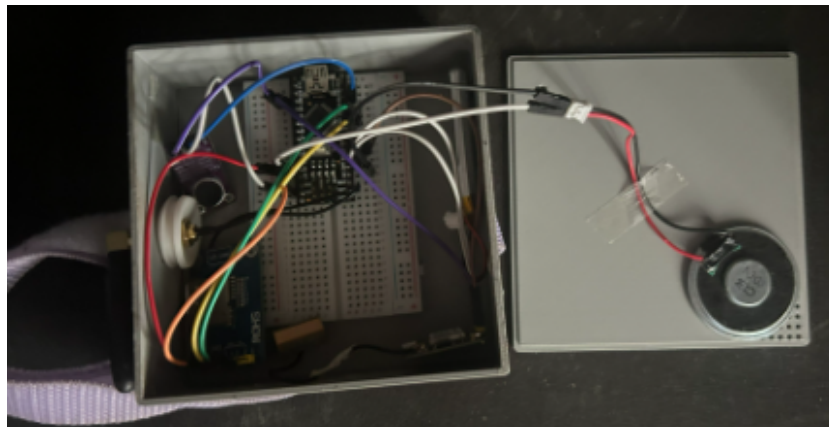


Figure 13: Final design inside

With-in the final design i try to discuss the best way in installing the all components into the box with Nariman. we needed the antenna on the outside for the best connectivity. Due to all the components being so close together we were concerned about over heating. And the main issue we concerned with is that the MAX9814 microphone module would pick up noise from the speaker. All challenges are considered with-in the Narmian report.

6 Evaluation

The Guardian Trace project was developed to address the growing issue of dementia patients wandering and becoming lost, which can place vulnerable individuals at significant risk while also creating stress for caregivers. The project aimed to design a wearable GPS tracking and communication device capable of providing real-time location tracking through the integration of an Arduino Nano, GT-U7 GPS module, and SIM800L EVB GSM communication module.

6.1 Quality of the product.

The quality of the development process was also improved through continuous testing and troubleshooting. Throughout the project, different issues involving serial communication, GSM connectivity, and power management were investigated using AT commands and hardware testing methods. This allowed faults to be identified more effectively and helped improve my understanding of embedded systems integration.

6.2 Reliability of the product

The reliability of the product became one of the biggest challenges during development. The GPS module itself was able to collect and display location data correctly, including latitude, longitude, altitude, speed, and satellite count. However, the reliability of the communication system was inconsistent due to issues with the SIM800L EVB module. Throughout testing, the module repeatedly struggled to connect and stay registered on the GSM network. In many cases, the module would continuously restart during transmission attempts, preventing stable real-time tracking.

7 Hardware

8 Hardware

Item	Qty	Cost	Notes / References	Weight
Arduino Nano	1	£10–£20	Controller for the whole operating system.	7g
GU-T7 GPS Module	1	£6	Tracking unit for the product.	5.7g
SIM800L EVB	1	£16	GSM/GPRS communication module for cellular connectivity.	1.35g
MAX9814 Microphone Module	1	£5–£8	Microphone unit.	2.5g
Speaker 4Ω–8Ω	1	£2–£4	Audio output connected to the Arduino Nano.	13g
LED Light	1	£2–£4	Used for the SOS light.	0.5g
3.7V 500mAh LiPo Battery	1	£6–£10	Power source for the product.	12g
App	1	£25	Mobile application for user interface and communication.	0g
Total		£72–£93		42.05g

Table 1: Hardware components and associated costs.

When making the first hardware table I stated I would be using the Neo-6M GPS module. When receiving our parts this one item works that was the exact same however it works exact same as the Neo-6M. As-well with making the design of the product we didn't use the push button we understood that the push button would give more problems to the safety of the design. And that we didn't need it.

9 Gantt chat

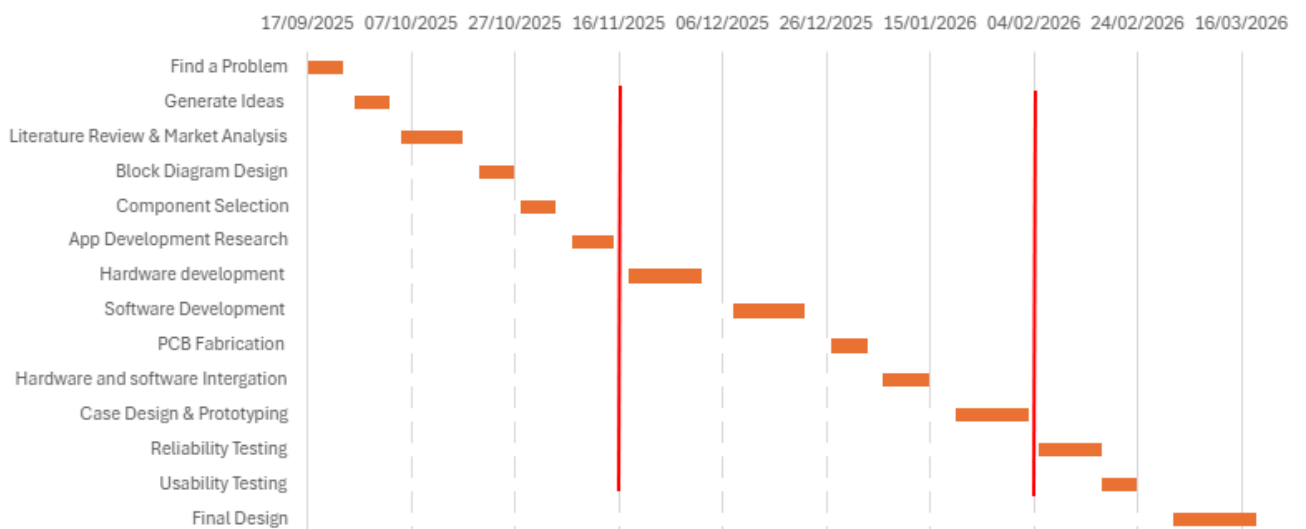


Figure 14: Gantt Chart

The project did not fully adhere to the original Gantt chart, as the design phase exceeded the planned duration by approximately one month. This delay was primarily due to an underestimation of the complexity involved in developing the system design, particularly in ensuring compatibility between hardware and software components. Additionally, multiple design iterations were required to refine the proposed solution, which further extended this phase.

As a result of this delay, subsequent stages, including development, integration, and testing, were pushed back and had to be completed within a reduced time frame. This ultimately impacted the overall progress of the project, and as a consequence, the final system was not completed to the full extent originally intended. This resulted in not being able to finish the final design before the second hard deadline and not be able to test the product.

9.1 Improvement for the future

During my project, I needed a module to transmit data via 4G/5G, but the £100 budget made this impossible—a SIM7600 would have consumed half the budget, as 4G/5G technology has not yet scaled down for low-cost projects. I chose 2G instead, believing it would work because UK coverage maps showed availability. However, when we tested components outdoors, signal failures occurred repeatedly. I felt frustrated and responsible, especially because I had tested parts individually rather than fully integrating and validating them earlier. Within the team, we divided tasks but failed to stress-test each other's riskiest assumptions collaboratively. This slowed our problem-solving significantly—we discovered issues in parallel rather than solving them together from the start. The critical lesson was that theory does not equal reality: just because Google says 2G exists does not mean it works reliably in the field. A better team approach would have been to identify our highest-risk assumption (real-world network coverage) and test it on day one, outdoors, with the whole group involved. We could have also explored sharing a SIM7600 module among team members or borrowing one temporarily rather than immediately settling for 2G. Looking back, I should have insisted that we integrate and test components as a complete system much sooner. For future group projects, I will create a shared "assumption validation" checklist and test the most critical conditions first such as signal strength in the actual deployment location—before committing to any component. This experience taught me that effective problem-solving requires both early validation and true team collaboration, not just dividing tasks and hoping everything works in theory.

10 Appendix

10.1 Code explanation Appendix A

This code implements a complete GPS (Global Positioning System) data receiver and parser using an Arduino Nano micro controller and a GT-U7 GPS module. The system captures raw satellite data, converts it into human-readable location information, and displays it through a serial monitor interface. Allowing it to then be transferred into the app we have created or already existing maps like google map and other. The code leverages two specialized libraries to handle the complexities of GPS communication and data parsing.

The code begins by including two essential libraries. The TinyGPS++.h library provides functions for parsing NMEA (National Marine Electronics Association) sentences, which is the standard data format transmitted by GPS modules. Without this library, raw GPS data would appear as unintelligible strings of text. The SoftwareSerial.h library enables the creation of additional serial communication ports on standard digital pins, as the Arduino Nano has only one built-in hardware serial port that is already reserved for USB communication with the computer.

Line 4 creates a virtual serial port using digital pins 3 and 4 on the Arduino Nano. Pin 3 is configured as the receive pin (RX), which listens for incoming data from the GPS module. Pin 4 is configured as the transmit pin (TX), which sends data to the GPS module if needed.

The setup function runs once when the Arduino is powered on or reset. It performs three critical initialization tasks:

`Serial.begin(9600)` (line 9): Initializes the hardware serial port for communication with the computer's serial monitor. The 9600 parameter specifies the baud rate (9600 bits per second), which must match the baud rate setting in the serial monitor for proper communication.

`serial connection.begin(9600)` (line 10): Initializes the software serial port for communication with the GPS module. Most standard GPS modules, including the GT-U7, operate at 9600 baud by default. This establishes the communication handshake between the Arduino and the GPS device.

`Serial.println("GPS Start")` (line 11): Outputs a confirmation message to the serial monitor indicating that the system has successfully initialized. This serves as a diagnostic tool to verify that the Arduino is running and has reached the setup function.

`While(serial connection.available())` (line 16): This checks if any data has been received from the GPS module.

The last section of the code give us the most information that we can use to locate the GPS. This section displays all GPS information in a human-readable format on the serial

monitor or on app devices. I give is the Satellite count (number of satellites tracked), latitude/longitude (Position coordinates), speed (velocity in mile per hour) and altitude (Height above sea level).

11 References

I. Das, Abhishek, et al. “Smart Car Features Using Embedded Systems and IoT.” ArXiv.org, 2 Jan. 2021, arxiv.org/abs/2101.00496. Accessed 31 Oct. 2023.

II. Sugiyanti, Intan. “Design of ATM Crime Monitoring System Based on MQTT Protocol Using SIM800L and Arduino Mega 2560.” ResearchGate, July 2019, <https://www.researchgate.net/publication/334226643>.

IV. SIMCom (2015) SIM800H & SIM800L Hardware Design V2.02. Shanghai SIM-Com Wireless Solutions Ltd.

V. Okoronkwo, G. et al. (2013) An Improved GSM-Based Process Monitoring and Control in Embedded Systems. Available at: ResearchGate.

VI. Clark, Adam. “Switching off 2G and 3G in the UK.” House of Commons Library, 18 Nov. 2024,

VII. Moon, Nathan W, et al. “Designing Wearable Technologies for Users with Disabilities: Accessibility, Usability, and Connectivity Factors.” Journal of Rehabilitation and Assistive Technologies Engineering, vol. 6, Jan. 2019, p. 205566831986213, <https://doi.org/10.1177/2055668319862137>.

VIII. A. B. Agrawal et al., “A Smart Wearable for Emergency Response Using ESP32 and SIM7600,” in *2025 2nd International Conference on Electronic Circuits and Signaling Technologies (ICECST)*, Oct. 2025, pp. 793–796, doi: 10.1109/icecst66106.2025.11307591.

11.1 New part links

https://www.amazon.co.uk/SIM800L-Wireless-Extension-Antenna-Replacement/dp/B09BMPPLRY/ref=asc_df_B09BMPPLRY?mcid=90785f16b6e338b5880845ed91bfe753&tag=googshopuk-21&linkCode=df0&hvadid=697311939728&hvpos=&hvnetw=g&hvrand=186166977532848&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9192684&hvtargid=pla-1944980731386&psc=1&hvocijid=1861669775328488475-B09BMPPLRY-&hvexpln=0&gad_source=1

commonslibrary.parliament.uk/research-briefings/cbp-9959/