SafeStep: Smart White Cane Attachment for Audio + Haptic Navigation and Emergency Alerts

ECE 445 Design Document - Spring 2026

Project # 15

Arsalan Ahmad, Eraad Ahmed, Abdulrahman Almana

Professor: Arne Woosley

Fliflet

TA: Abdullah Alawad

1 Introduction

1.1 Problem

White canes provide reliable obstacle detection and remain one of the most trusted mobility tools for blind and low-vision individuals. However, while they are effective at identifying immediate obstacles, they do not provide route-level navigation that helps a user move efficiently toward a destination. Navigating unfamiliar environments such as university campuses, public transportation areas, or busy urban streets can therefore become slow and mentally demanding. Users often need to frequently stop to reorient themselves, ask for assistance, or rely on smartphone navigation that is not designed to integrate naturally with cane-based mobility.

Although smartphone navigation applications can provide directions, they operate independently from the white cane and do not directly translate navigation information into physical feedback that aligns with how users already travel. Most systems depend on a single form of feedback, which may not suit every user or situation. This creates a gap between traditional mobility aids, which are simple and reliable, and modern navigation technologies that are not fully integrated into cane-based movement.

Personal safety is another concern during independent travel. Falls or disorientation can occur due to uneven terrain, unexpected obstacles, or unfamiliar surroundings. In many cases, there is no automatic mechanism to notify others if assistance is needed, which can delay assistance during emergencies. These limitations highlight the need for a solution that enhances navigation guidance and safety while preserving the familiarity and reliability of a standard white cane rather than replacing it.

1.2 Solution

We propose a modular smart attachment that mounts onto a standard white cane to improve navigation and safety without replacing the cane's core purpose. The attachment will connect via Bluetooth to a user's phone and headphones to support clear spoken directions, and it will also provide vibration-based cues for users who need non-audio feedback. The vibrations cues will convey information such as turns, continuing straight, and arrival based on a set of vibrations. In addition to navigation support, the attachment

will also allow personal safety though fall detection and emergency alert functionality. When a fall is detected the module will send a SMS message to an emergency contact notifying them about the fall and the user's last known location. This allows the user to get assistance quickly in the event of an emergency.

1.3 Visual Aid

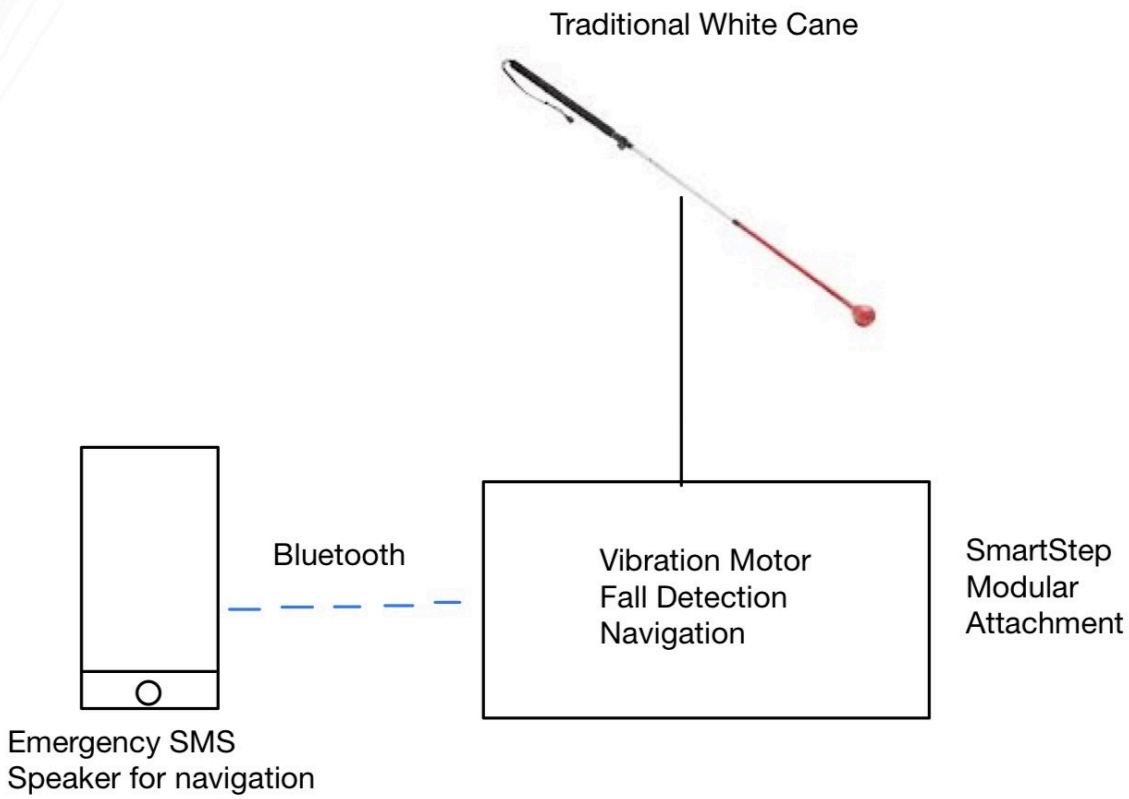

Figure 1: Smart Cane Attachment Visual Aid

1.4 High Level Requirements

To consider our project successful, our modular cane attachment must fulfill the following.

- The system must be able to provide clear navigation through audio and vibration cues to guide the user towards a destination as well as see 10 meters ahead to warn users of potential obstacles.
- The system must be able to detect falls and trigger an emergency alert to a pre-set emergency contact and give the user’s last known location.

- The system must connect to the user's phone via bluetooth and headphone to support navigation and emergency alerts.

2 Design

2.1 Block Diagram

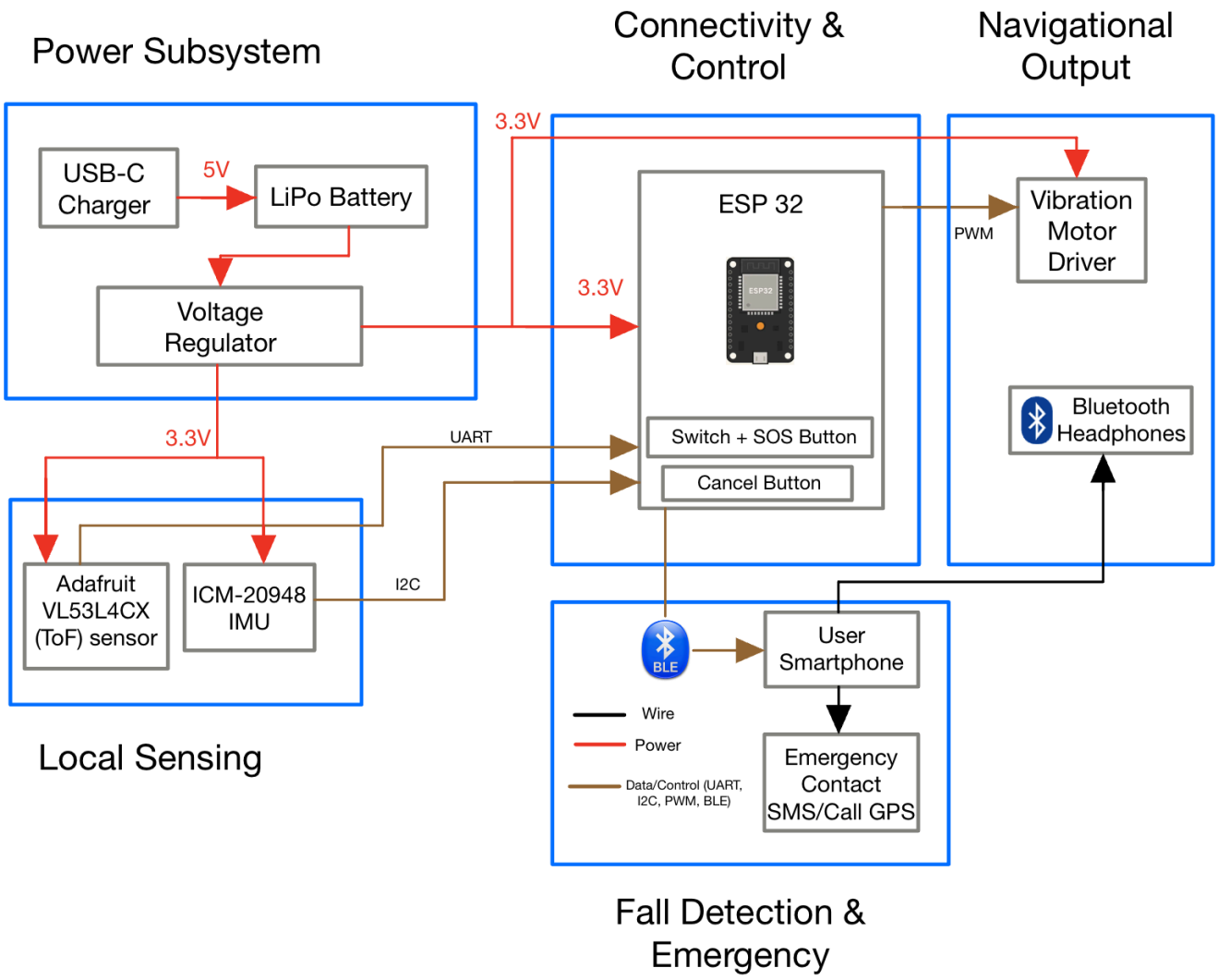

Figure 2: Block diagram providing top level overview of the components involved and their connectivity

2.2 Physical Design

Figure 3: AI generated concept providing a rough visual of the final product

The concept above is a good (albeit not entirely accurate) physical design concept for the SafeStep white cane attachment. We plan to have the device essentially be the handle of the white cane, which the user grips when using. Inside this enclosure/handle is the PCB, electronics, and the haptic vibration motor. The ToF module will be installed on this enclosure in a way in which, when the cane is operated (facing away from the user towards the ground at a 45-degree angle, with arms fully extended), the ToF sensor will be facing directly forward. This allows us to keep connectivity quite good between our device and our phone, since it's closer to the body. It also allows the haptic sensory output, which is an essential part of this project. It also is the adequate area to mount the ToF sensor since we want realistic object detection that the user cannot detect with the white cane, which are large objects straight ahead 6-15 feet away.

2.3 Subsystem 1, Connectivity + Control

This subsystem is centered on the ESP32-S3 microcontroller module, which manages all onboard sensor processing, navigation logic, and wireless communication. This subsystem establishes a Bluetooth Low Energy (BLE) connection to the user's smartphone, receives navigation instructions from a phone app (taking navigation directions from APIs for google or apple maps), and translates those instructions into audio and haptic output patterns. It also collects local sensor data from the ToF distance sensor and IMU over I²C/UART, fuses this information, and decides when to trigger obstacle alerts, fall detection responses, emergency sms requests, and many other peripheral sets of information.

The microcontroller will run firmware that continuously polls the ToF sensor at approximately 50 Hz (ESP32 Clock) during operation and classifies the measured distance into safety zones (e.g., clear, caution, warning, danger). The same controller maintains the BLE link, monitors battery and charging status signals from the power subsystem, and coordinates system-level fault handling (e.g., degraded operation when the phone disconnects, low-battery warnings, or sensor failure). This subsystem therefore acts as the central controller that enforces high-level requirements such as reliable navigation cues, fall detection logic, and emergency alert initiation.

Table 1: Connectivity + Control Subsystem – Requirements & Verification

Requirements	**Verification**
The ESP32 control subsystem must successfully establish and maintain a BLE connection to a smartphone running the companion app within 10 seconds of power-on.	Power on the device with the app open on a smartphone. Measure the time from device boot to reported BLE connection in the app logs or serial debug output. Repeat at least 10 times and ensure that connection time is ≤ 10 s in all trials.

The control subsystem must receive navigation commands from the phone and drive audio and haptic outputs with total end-to-end latency ≤ 200 ms.	Use the app to send a timestamped navigation command (e.g., “turn left”) while logging the command send time on the phone and the alert trigger time on the ESP32 via serial output. Measure the time difference for at least 20 trials and confirm all measured latencies are ≤ 200 ms.
The control subsystem must poll the ToF sensor at 50 Hz (±10%) while active and update obstacle zones accordingly.	Add a debug counter that increments each time a ToF reading is taken. Over a 10-second interval, record the number of readings via serial output and verify that the measured sampling frequency is between 45 Hz and 55 Hz. Confirm that changes in measured distance result in corresponding zone changes in debug output.
The firmware must detect a BLE disconnection within 2 seconds and trigger a local warning (e.g., vibration pattern or buzzer) to alert the user.	With the device connected to the phone, abruptly disable Bluetooth or move the phone out of range. Measure the time between disconnection (as seen in the app or phone OS) and the start of the local warning using a stopwatch. Verify that the warning begins within 2 seconds for at least 5 disconnection trials.

2.4 Subsystem 2, Navigation Output (Audio + Haptics)

The Navigation Output subsystem is responsible for communicating navigation and warning information to

the user through vibration-based haptic cues and audio guidance delivered via a paired smartphone. The ESP32 generates distinct vibration patterns using a motor driver to indicate navigation directions such as left turns, right turns, continuing straight, or arrival at a destination. At the same time, route-level instructions from the smartphone's navigation application are delivered through a Bluetooth earbud or headphone.

By combining haptic and audio feedback, this subsystem allows users to receive guidance in a way that aligns with natural cane usage without requiring constant visual or manual interaction with a phone. Haptic cues provide immediate directional awareness, while audio offers higher-level route context. The subsystem must ensure that vibration signals are timely and distinguishable, and that audio guidance remains stable during use.

Table 2: Navigation Output (Audio + Haptics) – Requirements & Verification

Requirements	**Verification**
The system must begin a vibration cue within 0–0.75 s of receiving a navigation command.	Trigger a navigation command from firmware while recording with video. Measure time from command log to vibration onset for 10 trials and confirm latency is ≤ 0.75 s.
The system must support at least four distinct vibration patterns corresponding to navigation states.	Program four patterns and test recognition with 3 users over 5 randomized trials. Require ≥80% correct identification.
Audio navigation must remain connected within 10 m indoors without interruptions longer than 5 s over a 5-minute test.	Play navigation prompts while the user walks with the phone at 5–10 m distance. Log any dropouts; confirm none exceed 5 s.
Vibration cues must remain perceivable during	Conduct a 10-minute walking test with periodic

normal walking motion.	vibration cues and record perceived detections. Require perception in ≥90% of events.

2.5 Subsystem 3, Local Sensing (TOF + IMU)

This subsystem is responsible for motion sensing as well as environment sensing for obstacle detection. This subsystem is made up of the Adafruit VL53L4CX Time-of-Flight (ToF) distance sensor and the ICM-20948 9-DoF. These sensors are essential for distance calculations as well as fall detection using the gyroscope and accelerometer on the ICM chip.

The VL53L4CX ToF sensor measures the distance to objects directly in front of the user. It operates over a range of approximately 1mm to 6 meters and will communicate with the ESP32 sampled at 50Hz during operation.

The ICM-20948 IMU provides 3-axis accelerometer and 3-axis gyroscope data. The IMU communicates with the ESP32 and is sampled at 100Hz. Accelerometer data is used to detect sudden impacts, while gyroscope data supports motion state classification. Fall detection is implemented by detecting a high acceleration spike followed by a sustained period of low motion, indicating impact and inactivity.

This subsystem contributes directly to the project's high-level requirements by enabling:

- Real-time obstacle detection for safe navigation
- Reliable fall detection for emergency response

Table 3: Local Sensing (TOF + IMU) – Requirements & Verification

Requirements	**Verification**
The VL53L4CX sensor shall provide valid distance measurements within the operational range of 1 cm	Place a flat reflective target at fixed distances (0.2 m, 0.5 m, 1 m, 2 m, 3 m, 4 m) and log distance

to 5 meters.	measurements for 30 seconds at each position. Compare measured values to actual distances.
Distance measurement error shall not exceed ±10% within the 1 cm to 5 m operating range.	At each test distance, compute average measured distance and calculate percentage error relative to actual distance.
The IMU shall detect an acceleration spike of ≥ 2.5 g during simulated fall impact.	Perform controlled drop/impact tests and log peak acceleration values. Confirm spike detection exceeds defined threshold.

2.6 Subsystem 4, Fall Detection + Emergency Alerts

The Fall Detection + Emergency Alerts subsystem is responsible for identifying fall-like events using IMU motion data and initiating an emergency notification workflow through the user's smartphone. The subsystem runs fall detection logic on the ESP32 using accelerometer/gyroscope features (e.g., high acceleration peak, followed by orientation change and inactivity) to reduce false triggers from normal walking or cane taps. When a fall is detected, the system enters a short confirmation window where the user can cancel the alert using the Cancel button. If the alert is not canceled, or if the SOS button is pressed, the ESP32 sends a BLE trigger to the smartphone. The smartphone then initiates an emergency action (SMS/call) that includes the user's last known location. This subsystem directly supports the project's safety goal by reducing response time during emergencies while maintaining usability and minimizing false alarms.

In addition, the subsystem must remain reliable even when the device is used in real walking conditions. That means it must handle noisy IMU data, intermittent BLE connection issues, and user input timing. The subsystem must also provide clear user feedback (vibration and/or audio prompt through the phone) during

the cancel window so that a user knows an alert is about to be sent and can stop it if needed.

Table 4: Fall Detection + Emergency Alerts – Requirements & Verification

Requirements	Verification
Fall detection latency: When a staged fall event occurs, the system must detect the event and enter alert mode within 0–10 s of the fall impact.	Create a repeatable staged fall test: drop the device from a fixed height (0.5 meter) onto mat, then keep it still. Use video recording to time from impact to the start of the alert indication (vibration pattern and/or phone prompt). Record at least 10 trials and report detection times as a table (min/avg/max).
Cancel window duration: After fall detection, the system must provide a cancel window of 8–10 s during which pressing the Cancel button prevents the emergency alert from being sent.	Trigger a fall event, then press Cancel several times (t = 2 s, 6 s, 9 s into the window). Verify whether the alert is suppressed. Repeat for 3 trials per time point. Record results as a pass/fail table and confirm the window falls within 8–10 s.
False trigger rate (normal use): During normal walking with cane motion, false emergency triggers must be ≤ 1 per 20 minutes of continuous use.	Use the device during a continuous walking trial for 20 minutes with BLE enabled and sensing active. Log all “entered alert mode” events. Repeat 2 sessions and report total false triggers.
SOS button response: When the SOS button is	With the phone connected over BLE, press SOS

pressed, the system must send a BLE emergency trigger to the smartphone within 0–2 s.	while recording video showing the button press and the phone receiving the trigger/automation starting. Time the delay from press to phone confirmation. Perform 10 trials and gather min/avg/max.
BLE emergency trigger reliability: With the phone within 10 m indoors, the BLE emergency trigger must be successfully received and processed by the smartphone in ≥ 90% of attempts.	Place the phone 5–10 m away (same room / hallway). Send 10 SOS triggers and count how many result in the phone automation starting successfully.

2.7 Subsystem 5, Power

The power subsystem is responsible for providing safe and reliable electrical power to all electronics in the pcb, including the ESP32‑S3, ToF sensor, IMU, haptic motor, and other auxiliary components. The design uses a 12 V NiMH 24000 mAh battery pack as the primary energy source, which is regulated down to intermediate 5 V and logic-level 3.3 V rails using linear regulators (e.g., TPS7A2050 for 5 V and AP2112K for 3.3 V). A battery management and charging circuit based on an adjustable regulator (e.g., LM317) and protective components (fuse, appropriate resistors, and capacitors) will limit charging current and prevent overvoltage, in accordance with our ethical and safety guidelines in section 4 of this document.

This subsystem must support a few hours (trial system) of typical operation while the system performs continuous sensing, BLE communication, and intermittent haptic and audio feedback. It must also prevent unsafe battery conditions such as overcharge, over‑discharge, and short-circuit currents that could damage the device or harm the user. Visual, audio, and haptic indicators (LED, buzzer, and vibration) and will notify charging status and low‑battery conditions so the user or their non-visibility impaired loved ones can

recharge before the system shuts down; the firmware will also monitor the 3.3 V rail or a scaled battery voltage input to trigger low-battery warnings and, if necessary, an orderly shutdown.

Table 5: Power Subsystem – Requirements & Verification

Requirements	**Verification**
The power subsystem must provide regulated 5 V and 3.3 V rails within ±5% of nominal over the expected battery voltage range during no-load and full-load operation.	Connect the battery simulator or bench supply to the power input and sweep from fully charged (e.g., 14.4 V) down to the minimum operating voltage. At both no-load and maximum expected load, measure the 5 V and 3.3 V rails with a multimeter and confirm each remains within ±5% of nominal.
The device must operate for at least N hours (e.g., 2 hours) of typical use on a full battery charge.	Fully charge the battery, then run the system in a representative usage scenario (BLE connected, ToF/IMU active, periodic haptics/audio) until the system reaches its low-battery shutdown condition. Record elapsed time and verify that the runtime is ≥ N hours. Repeat for at least two charge–discharge cycles.
The charging circuit must safely charge the NiMH battery from a 5 V USB-C source, limiting current	Fully discharge the battery to the safe lower limit specified by the cell datasheet. Apply 5 V via

to the designed value and preventing overcharge.	USB-C and monitor charge current and battery voltage with a multimeter. Confirm that charge current never exceeds the design limit and that battery voltage remains within manufacturer specifications at end-of-charge. Verify that current tapers as expected and that charging stops or is reduced when the target voltage is reached.
The power subsystem must include over-current protection that limits fault current to a safe level in case of a short on the output rails.	Intentionally create a short circuit across the 5 V rail through a current-measuring instrument (e.g., multimeter in current mode) while monitoring current and component temperature. Verify that the fuse or protection element activates or the regulator current limit engages so that the current does not exceed the specified safe limit and that no components exceed their rated temperature by touch or IR thermometer.
The system must detect a low-battery condition and initiate a controlled shutdown or reduced-function mode before the battery reaches damaging over-discharge levels.	Slowly discharge the battery while the device is running and log battery or scaled voltage readings on the microcontroller. Confirm that when the battery voltage crosses the low-battery threshold, the firmware triggers a visible/audible warning and either disables nonessential loads or shuts down the system. Verify that the measured battery voltage at

	shutdown remains above the minimum safe voltage recommended by the battery datasheet.

2.8 Subsystem 6, Programming (ESP32 Firmware)

This subsystem is responsible for coordinating communication between all other subsystems using the ESP32 microcontroller and ensuring that the overall system meets the project's high-level requirements. The ESP32 firmware is the central processing unit of the device, as it manages sensor data, signal processing, decision-making, vibration output control, and Bluetooth communication with the user's phone.

The firmware will continuously read distance measurements from the Adafruit VL53L4CX Time-of-Flight (ToF) breakout sensor over the I²C interface to detect obstacles in front of the user. This sensor operates from 1 mm to 6 meters and will be sampled at 50 Hz during active operation. The measured distances from objects will be classified into defined safety zones (clear, caution, warning, and danger) to provide real-time environmental awareness. The software will ensure that the worst object detection and alert to the user will occur with a worst case latency of 120 ms to support safe navigation.

While reading the TOF sensor data the firmware will also continuously read data from the motion data from the ICM-20948 9-DoF IMU over I²C at a sampling rate of at least 100 Hz. The accelerometer and gyroscope data are processed using threshold-based logic to detect potential fall events. A fall is detected when there is a sudden jump in the acceleration of the user followed by a period of low motion, meaning the user has fallen.

Based on the obstacle classification and fall detection state, the ESP32 generates 200 Hz PWM signals to control the vibration motor driver. The firmware implements 4 vibrations for navigation, turn left, turn right, continue straight, and arrivals as well as a high priority continuous vibration for objects that are directly in front of the user.

Lastly, the subsystem maintains a Bluetooth Low Energy (BLE) connection with the user's smartphone with a target operating range of at least 10 meters line-of-sight. The bluetooth connection will

be used for emergency notifications, navigation, as well as headphone connection for audio navigation.

Table 6: Programming (ESP32 Firmware) – Requirements & Verification

Requirements	Verification
The ESP32 firmware shall read VL53L4CX distance data over I²C at ≥ 50 Hz during active operation.	Log timestamps for distance reads for 60 seconds and compute the average sample rate from the log.
The firmware shall read ICM-20948 IMU data over I²C at ≥ 100 Hz during active operation.	Log timestamps for IMU reads for 60 seconds and compute the average sample rate from the log.
The firmware shall update the vibration output based on the obstacle zone with worst-case detection-to-output latency ≤ 120 ms.	Timestamp the moment the firmware detects a zone change and measure time until the motor control pin changes using an oscilloscope/logic analyzer. Repeat across 100 threshold-crossing trials.
The firmware shall detect a suspected fall within 3 seconds of a simulated fall event using IMU data.	Perform controlled fall simulations (impact + inactivity) while logging IMU data and fall-state timestamps. Measure time from impact event to FALL_SUSPECTED assertion across multiple trials.
The firmware shall maintain a BLE connection to the smartphone at 10 m line-of-sight and reconnect automatically after a disconnect.	Conduct a 15-minute range test at 10 m line-of-sight while logging disconnects. Then force a disconnect (out of range), return to range, and measure reconnection time from link lost to connected.

The firmware shall generate a distinct emergency notification for automatic fall confirmation.	Capture BLE packets to confirm distinct event identifiers are transmitted.

2.9 Tolerance Analysis

One point of failure that could possibly pose a risk to the successful implementation of the project is the system latency in providing alerts, and the effect it has on stopping time. This can be crucial in very important scenarios, such as preventing collisions, notifying the user of crosswalks or other traffic controlling mechanisms, and the avoidance of dangerous hazards (ditches, streams, inclined platforms)

For this we can do a mathematical analysis of a potential scenario:

Imagine a user walking at a fast pace, and the Tof sensor on the device detects an object that is 2 meters (approximately 6.5 feet) away. We can derive a mathematically guided simulation to fit this scenario:

Using the Following Variables:

$Vwalk: \ walking\ speed,\ approximately\ 1.3\ m/s$

$Tsys: \ System\ Latency\ approximately\ 1.0\ second$

$Treact: \ Human\ Reaction\ time\ approximately\ 0.3\ second$

$Tmech: \ mechanical\ stop\ time,\ time\ to\ to\ decelerate\ to\ stopping\ speed\ approximately\ 0.5\ second$

Total Time:

$$Ttotal = Tsys + Treact + Tmech = 1.8\ seconds$$

Distance Calculations:

$$D(latency + reaction) = (Tsys + Treact) * Vwalk = (1.3\ s) * 1.3\ m/s = 1.69\ m$$

$$D(deceleration) = (Tmech) * Vwalk = (0.5s) * 1.3\,m/s = 0.65\ m$$

$$D(total) = 0.65 + 1.69 = 2.015\,m$$

If we analyze this crude simulation we see that we need 2.015 meters to come to a complete stop. However the object is 2 meters away. The simulation predicts a collision. That means it is possible that in real life testing, with delays and all, may not be able to be quick enough to stop the user from colliding with objects that are in very close proximity. We may need to come across some work arounds, like rerouting and collision avoidance, instead of alerting the user to come to a complete stop.

3 Cost and Schedule

3.1 Cost Analysis

Labor					
Team	Hourly Rate	Hrs/Wk	Weeks	Total Cost	
Arsalan Ahmad	$53/hr	10	14	$7,420.00	
Eraad Ahmed	$53/hr	10	14	$7,420.00	
Abdulrahman Almana	$53/hr	10	14	$7,420.00	
Supply Center	$85/hr	2	3	$510.00	
Total Labor Cost: **$22,770.00**					

Parts					
Description	Manufacturer	Part Number	Quantity	Unit Cost	Total Cost
Microcontroller	Espressif	ESP32-S3-WROOM-1-N16R2	1	$6.13	$6.13
IMU Board	TDK/Adafruit	ICM-20948 9-DoF Breakout	1	$14.95	$14.95
TOF Board (Adafruit 6m)	Adafruit/STM	VL53L4CX Breakout	1	$14.95	$14.95
USB-to-UART Contr.	Silicon Labs	CP2102N-A02-G QFN28	1	$4.32	$4.32
Battery (12V NiMH)	Conrad Energy	12V 2400mAh Tamiya	1	$18.99	$18.99
Voltage Reg 5V	Texas Instr.	TPS7A2050PDBVR	1	$0.36	$0.36
Voltage Reg 3.3V	Diodes Inc.	AP2112K-3.3TRG1	1	$0.22	$0.22
Charging IC	Texas Instr.	LM317T	1	$0.56	$0.56
Battery Connector	Generic	Tamiya Connector	1	$2.49	$2.49
Fuse	Littelfuse	0451002.MRL	1	$0.50	$0.50
Vibrating Motor	Adafruit	1201 Mini Motor Disc	1	$1.95	$1.95
Piezo Buzzer	TDK	PS1440P02BT	1	$0.78	$0.78
USB-C Connector	Amphenol	USB-C Receptacle	1	$1.00	$1.00

Tactile Switches	Generic	Tactile SMD/TH	5	$0.40	$2.00
Red LED	Kingbright	LTST-C150CKT	1	$0.26	$0.26
Green LED	Kingbright	LTST-C171GKT	1	$0.12	$0.12
Resistor 10kΩ	Yageo	10kΩ (0603)	4	$0.05	$0.20
Resistor 499Ω	Yageo	499Ω (0603)	1	$0.05	$0.05
Resistor 240Ω	Yageo	240Ω (0603)	1	$0.05	$0.05
Resistor 1kΩ	Yageo	1kΩ (0603)	6	$0.05	$0.10
Resistor 100Ω	Yageo	100Ω (0603)	2	$0.05	$0.10
Resistor 330Ω	Yageo	330Ω (0603)	2	$0.05	$0.10
Resistor 150Ω	Yageo	330Ω (0603)	2	$0.05	$0.10
Capacitor 10µF	Murata	10µF (0603)	3	$0.10	$0.30
Capacitor 1µF	Murata	1µF (0603)	8	$0.08	$0.64
Capacitor 4.7µF	Murata	4.7µF (0603)	3	$0.08	$0.24
Capacitor 0.1µF	Murata	0.1µF (0603)	6	$0.08	$0.48
Capacitor 10nF	Murata	10nF (060	5	$0.04	$0.20
Total Part Cost: **$72.14**					
Excess Costs (15 percent rule) ≈ $10.80					
Tariffs ≈ $15.60					
PCB Cost ≈ $25.00					
Enclosure/Cane Materials ≈ $30					

Total Costs: $22923.54

3.2 Schedule

Week	Task	Person
February 22nd - February 28th	Order parts for prototyping and breadboard testing	Everyone
	Begin schematic design and component verification	Abdulrahman & Arsalan
	Research Tof sensor and IMU integration requirements	Eraad
	Complete Design Document submission	Everyone
March 1st - March 7th	Finalize PCB schematic and power subsystem design	Abdulrahman & Arsalan
	Configure ESP32 development environment	Eraad
	Test basic sensor communication on breadboard	Everyone
March 8th - March 13th	Prepare for Design Review and validate subsystem interfaces	Everyone
	Perform internal PCB audit and finalize PCB layout	Abdulrahman & Arsalan
	Submit first PCB order	Everyone

March 14th - March 22st	Spring Break (No scheduled projected work)	…
March 23th - March 28th	Assemble breadboard prototype (ESP32 and sensors)	Arsalan, Eraad
	Verify UART communication with Tof sensor	Eraad
	Verify I²C communication with IMU	Abdulrahman
March 29th - April 4th	Implement vibration motor driver control	Arsalan
	Develop vibration feedback patterns	Eraad
	Begin fall detection algorithm development	Abdulrahman
April 5th - April 11th	Integrate BLE communication with smartphone	Abdulrahman, Eraad
	Test navigation feedback system	Everyone
	Prepare Progress Demo prototype	Everyone
April 12th - April 18th	Assemble received PCB and validate power regulation	Everyone
	Perform full system integration testing	Everyone
April 19th - April 25th	Mock Demo preparation and system debugging	Everyone
	Optimize power usage and reliability testing	Eraad & Arsalan

April 26th - May 1st	Final system validation and testing	Everyone
	Final Demo and Presentation	Everyone

4 Ethics and Safety

4.1 Ethical Considerations

The main ethical considerations that must be kept in mind during the course of the project are the following: Public Safety, System Reliability, truthfulness in medical claims, data privacy and security, product testing and consent

4.1.1 Public Safety

In accordance with IEEE I.1[1], which mandates holding the safety, health, and welfare of the public paramount, our design should prioritize user safety above all else. Since visually impaired users will rely on our device to detect obstacles and request help, any failure could lead to physical injury or stranding. We must ensure that the "smart" attachment does not interfere with the cane's fundamental function as a reliable white cane; if the battery dies or the electronics malfunction, the cane must remain physically usable. Furthermore, reliance on automated phone shortcuts for emergency SOS alerts poses a safety risk if the phone is dead, disconnected, or if the operating system kills the background process. To mitigate this, we must clearly define the device as a secondary aid rather than a primary life-safety tool and design fail-safes that prioritize the user's immediate physical safety over complex electronic features.

4.1.2 System Reliability

Reliability is critical for a navigation aid, as inconsistent feedback can be more dangerous than no feedback at all. Our use of a single-point Tof sensor (Adafruit VL53L4CX Time-of-Flight (ToF)) and an IMU (ICM-20948) introduces potential points of failure, such as the inability to detect glass doors, artifacts, or fast-moving objects. If the system produces false negatives, the user could collide with a hazard;

conversely, frequent false positives could degrade the usefulness of the device. To address this, we will need to perform rigorous testing of sensor limitations and implement software filtering to help remedy these fault points. We must also ensure the Bluetooth connection is robust, as a dropped connection during navigation instructions could leave a user disoriented in an unfamiliar location.

4.1.3 Truthfulness in Medical Claims

As per IEEE I.3[1], we must be honest and realistic in our claims, avoiding any misrepresentation of our project as a certified medical device. While the system includes "fall detection" and "emergency alerts," it is not a medical-grade monitor and has not undergone the rigorous validation required by agencies such as the FDA[4]. We must explicitly label the device as a "prototype device" or "assistive technology" rather than a medical safety product. Overstating the accuracy of any medical-related use cases would be deceptive and potentially dangerous, so we should transparently disclose these limitations in all documentation and user interfaces.

4.1.4 Data Privacy and Security

The collection and transmission of location data for the emergency alert system raises significant privacy concerns under IEEE I.1[1]. Our device transmits the user's last known GPS coordinates via SMS to a pre-set contact. If this data were intercepted or if the device were hacked, a malicious actor could track the movements of a vulnerable user. There is also a possibility that auxiliary data (such as health-related measurements, if ever added) could be intercepted and misused. To adhere to ethical standards regarding user privacy, we should minimize data collection, ensure that communications occur through a secure medium where possible, and ensure that users have full transparency and control over who receives their data and when it is sent.

4.1.5 Product Testing and Consent

Developing assistive technology for people with disabilities requires strict ethical adherence. We recognize that testing the device on ourselves (the engineering team) does not accurately represent the

experience of a blind user. However, testing on applicable groups without proper approval and safety oversight is unethical. Therefore, we will restrict our current testing to team members and simulation environments. If we plan to do user trials, we will need to obtain informed consent from participants.

4.2 Safety Considerations

The following safety considerations protect users and the design team from potential bodily harm: electrical and battery safety, mechanical and environmental safety, and laboratory and development safety.

4.2.1 Electrical and Battery Safety

The inclusion of a battery and microcontroller in a handheld device introduces specific electrical hazards. The primary risk is thermal runaway or battery explosion if a battery cell is punctured, overcharged, or short-circuited [2]. Since the device is held directly in the user's hand for extended periods, even minor overheating could cause discomfort or burns. To mitigate this, we must integrate a battery management system with proper protections [3]. Additionally, the circuitry must be secured in a waterproof casing to prevent any risk of electric shock, particularly if the device is used in wet conditions.

4.2.2 Mechanical and Environmental Safety

Mechanically, the attachment must be robust enough to withstand daily impact, wear, and tear. If the casing cracks, it could expose sharp edges or internal electronics, posing a physical danger to a user who navigates by touch. The design must also be water-resistant (aiming for an appropriate IP rating). Ergonomically, the attachment must not significantly alter the balance or weight of the cane, as this could cause wrist strain and discomfort for the user.

4.2.3 Lab and Development Safety

During the development phase, we will adhere to strict laboratory safety protocols to protect our team. We will follow course-staff guidelines on safe handling of lab equipment and design components. Product testing must be done in a controlled, safe environment, and a standard operating procedure will be established for testing the fall detection system so we do not injure ourselves while simulating falls.

5 Citations

[1] IEEE. (2026) Ieee code of ethics. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html

[2] J. Zhao, X. Feng, Q. Pang, M. Fowler, Y. Lian, M. Ouyang, and A. F. Burke, “Battery safety: Machine learning-based prognostics,” Progress in Energy and Combustion Science, vol. 102, p. 101142, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360128523000722

[3] S. Nayak, H. Jayalaxmi, M. Sathya, U. Karn, S. Tejas, and S. Roopa, “Implementation of a secure battery management system with safety features for lithium-based batteries,” Scientia. Technology, Science and Society, vol. 2, pp. 26–33, 06 2025.

[4] U.S. Food and Drug Administration, “Device registration and listing,” https://www.fda.gov/medical-devices/how-study-and-market-your-device/device-registration-and-listing, accessed: 2026-02-13.