An Automated Approach to External Ventricular Drains

Design Document

University of Illinois Urbana-Champaign

ECE 445 - Spring 2025

Project # 66

Professor: Yang Zhao

TA: Jason Jung

Ralph Nathan, David Kapelyan, Isiah Lashley

1. Introduction

1.1. Problem

The brain's ventricular system continuously produces and absorbs CSF, with an estimated turnover of 450–500 cc per day and a total volume of approximately 150 cc at any given time. External Ventricular Drains (EVDs) are commonly used for short-term management of Intracranial pressure (ICP) to facilitate CSF drainage by utilizing gravity, with the drainage system zeroed at the external auditory meatus (EAM) to maintain accurate pressure measurements. However, EVD management is highly sensitive to positioning errors. Dr. Suguna Pappu, MD, PhD, has reported numerous cases where misalignment resulted in excessive drainage—up to 40 cc instead of the intended 10 cc (Normal ICP is typically 15–17 mmHg, and drainage is typically set at 10 cc/hour)—due to improper zeroing. When the EVD is positioned too low, large and rapid CSF loss can occur, sometimes exceeding 60–80 cc within minutes, which could lead to intracranial hemorrhage, brainstem herniation, and fatal outcomes [1]. Frequent patient movements—such as turning, transport, bathing, coughing, or even routine repositioning—necessitate clamping and re-zeroing the system. This disrupts continuous drainage and increases the risk of overdrainage or underdrainage due to improper recalibration. Outside specialized neurocritical care units, staff unfamiliar with EVD management face a heightened risk of errors, including failing to re-zero properly or inadvertently lowering the drain.

1.2. Solution

To ensure stable readings, reduce reliance on manual intervention and enhance patient safety, our proposed system automates CSF drainage by incorporating pressure and flow sensors that communicate with a microcontroller to control a solenoid valve. The

system drains CSF only when intracranial pressure (ICP) exceeds a user-set threshold, preventing overdrainage. Since ICP fluctuates due to patient movement, sneezing, or other factors, the pressure control mechanism ensures that ICP does not drop too low, while the flow control system regulates the drainage rate. A pressure sensor continuously monitors ICP, triggering the solenoid valve to open when pressure exceeds the threshold and close when it returns to a safe level. If the flow sensor detects a rate higher than the user-set limit, the solenoid valve will close and reopen only upon manual input, typically when the nurse empties the collection bag. In our proposed system, the EVD will be positioned level with the external auditory meatus (EAM) and secured to the mastoid bone as a fixed zero point, eliminating the need for manual zeroing.

1.3. Visual Aid

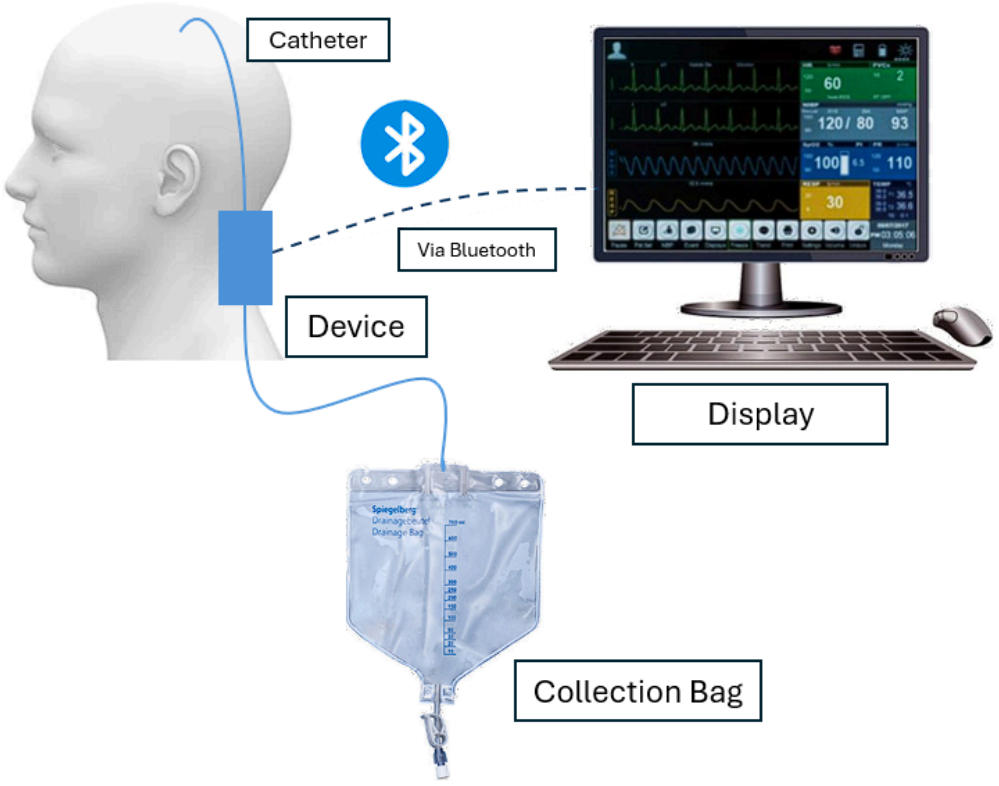

Figure 1: Visual aid of physical design and placement

To eliminate the need for manually zeroing the pressure sensor by adjusting the system's level relative to the external auditory meatus (EAM), as required in the current EVD setup, we will secure our device to the patient's mastoid bone as a fixed zero point, as shown in Figure 1. The device will include a tube connected to a collection bag for CSF drainage. Additionally, it will communicate with a graphical user interface (GUI) via Bluetooth, allowing users to input desired pressure levels and flow rates, manually reset the solenoid valve, and monitor real-time pressure and flow data. The GUI will also display graphs, an alarm system for abnormal readings, and data logs for physician review.

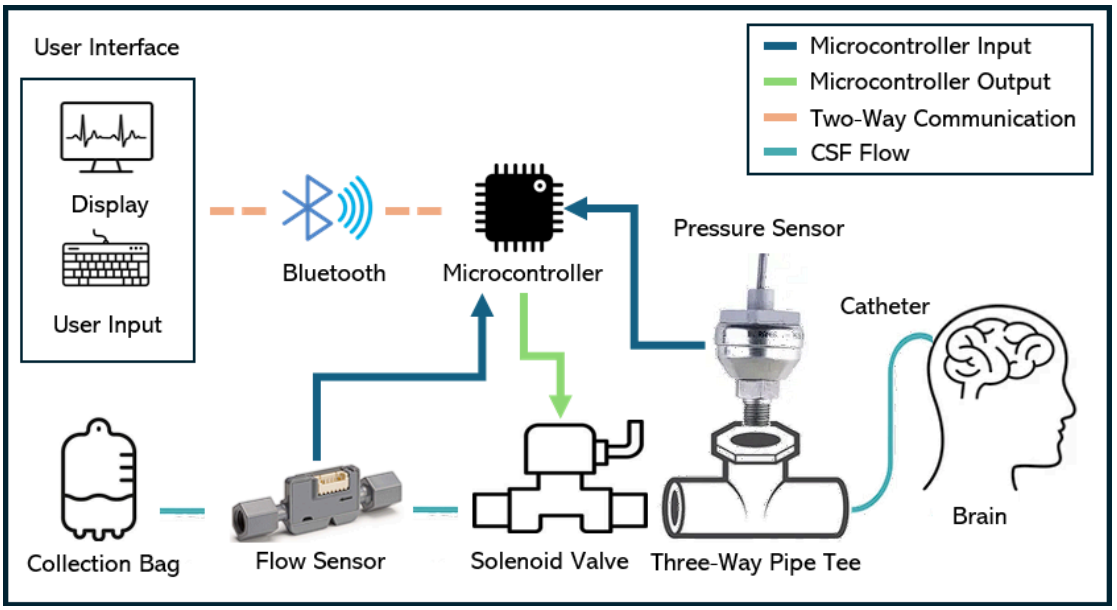

Figure 2: Visual aid of overall system design

Our design features a three-way pipe tee to which the catheter is connected, with a pressure sensor mounted on top, perpendicular to the CSF flow, and a solenoid valve at the opposite end, which remains closed by default in the absence of a driving voltage. As CSF fills the tee, the pressure sensor, in direct contact with the fluid, measures the CSF pressure, corresponding to the patient's intracranial pressure (ICP), and transmits the data to a microcontroller. If the detected pressure exceeds the

maximum user-set value, the microcontroller activates the solenoid valve to allow CSF drainage. Once the pressure drops to the minimum user-set threshold, the microcontroller signals the solenoid valve to close, halting drainage.

CSF flows through the solenoid valve to a flow sensor via a tube before reaching a collection bag. The flow sensor monitors the drainage rate and transmits the data to the microcontroller. If the flow rate exceeds a user-set limit, typically 10 cc/hour, the microcontroller signals the solenoid valve to close, stopping drainage until manually reopened—typically when the nurse empties the collection bag—to ensure controlled CSF removal.

1.4. High-Level Requirements

For our project to be successful, the system must meet three key requirements:

1. Automated Flow Regulation: The system must autonomously control CSF drainage via a solenoid valve. The valve should open when the intracranial pressure (ICP), measured by a pressure sensor, exceeds a user-set threshold (e.g., 16 mmHg) and close when it reaches a maximum user-set limit (e.g., 100 mmHg).
2. Flow Rate Sensing: The system must limit the flow rate to 10 ccs per hour, using a flow rate sensor to monitor and adjust drainage via the solenoid valve accordingly.
3. Real-Time Monitoring: The microcontroller must support Bluetooth communication with a display monitor, providing near real-time data with a maximum delay of 10-30 seconds (as approved by Dr. Pappu)

2. Design

2.1. Block Diagram

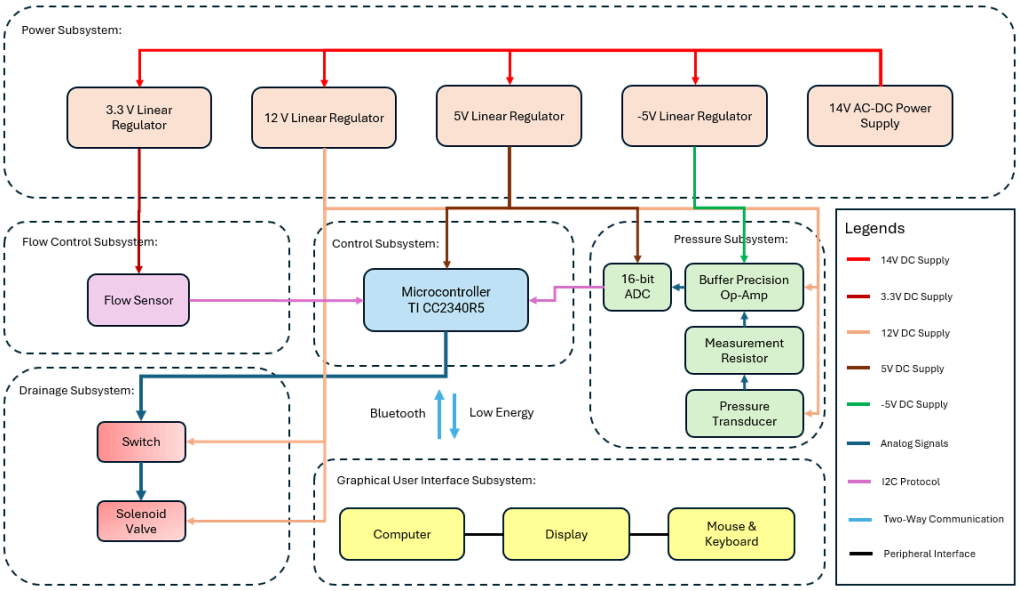

Figure 3: Block Diagram of the Automated EVD System

2.2. Physical Design

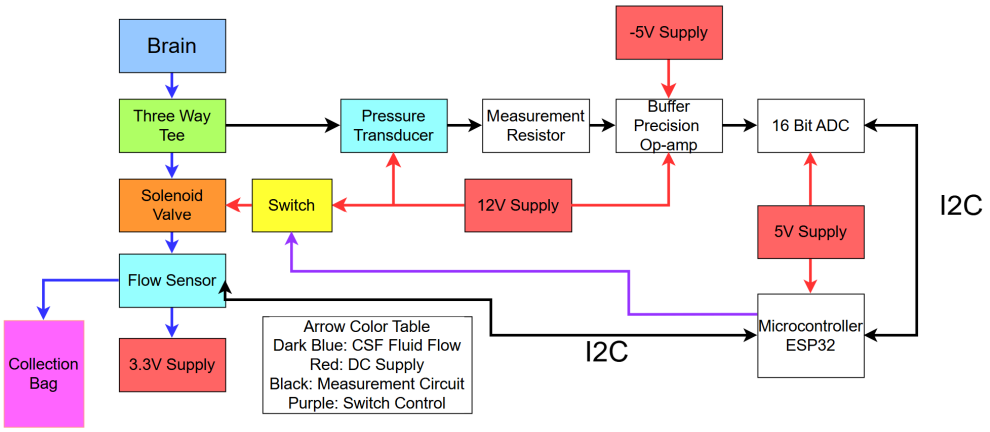

Figure 4: Physical Design Diagram of the Automated EVD System

The mechanical design of the system consists of several stages. The first stage includes a three-way tee connected to the input, a pressure transducer, and a solenoid valve. The solenoid valve is then connected to a flow sensor, which is

inline with the collection bag. By placing the flow sensor at the valve's output, we ensure that we measure only CSF outflow, preventing any interference from the initial fluid buildup in the first stage.

The pressure measurement circuit also consists of several stages. The pressure transducer is connected to a 200Ω resistor, which converts its output current signal into a voltage measurement. An op-amp in a unity gain configuration serves as a buffer to prevent issues when driving the analog-to-digital converter (ADC). The ADC then outputs an I²C signal to the microcontroller. The use of an external ADC improves measurement precision by providing higher resolution than the microcontroller's internal ADC and avoiding internal noise issues. Meanwhile, the flow sensor also outputs an I²C signal and is connected directly to the microcontroller.

2.3. Subsystem Description & Requirements

2.3.1. Microcontroller Subsystem

The Microcontroller Subsystem serves as the central processing unit, managing sensor inputs and control outputs to enable automated, real-time regulation of CSF drainage. The pressure sensor outputs an analog voltage, which is converted to a digital signal via an external Analog-to-Digital Converter (ADC). The microcontroller reads the ADC output via the I²C protocol, translating it into pressure readings. It also receives flow rate data from the flow sensor and processes both inputs to determine appropriate valve actuation. If the pressure or flow rate exceeds user-set thresholds, the microcontroller outputs a voltage signal to drive the solenoid valve,

controlling CSF drainage. Additionally, it interfaces with the Graphical User Interface (GUI) via Bluetooth, transmitting real-time pressure and flow data. We are utilizing the Espressif ESP32-WROOM-32 microcontroller, which features built-in Wi-Fi and Bluetooth, making it well-suited for wireless communication. The ESP32's integrated RF transceiver ensures reliable data transmission for seamless system monitoring and control.

Requirements	Verification
● Must support I²C and Bluetooth communication for sensor data acquisition and real-time transmission.	**Equipment:** A computer will be necessary **Test Procedures:** Connect the pressure transducer to the ADC circuit and wire the ADC module to the ESP32 microcontroller using I²C protocol (SDA and SCL lines). Upload a test firmware to the ESP32 that continuously reads pressure sensor data via I²C and transmits the acquired data via Bluetooth. Use the USB Logic Analyzer to verify that the ESP32 generates appropriate I²C start conditions, sends the device address, and correctly reads the pressure data bytes from the ADC. Use the laptop and Bluetooth communication software (Bleak) to confirm that pressure readings are transmitted via Bluetooth at 1 Hz or faster. Repeat the procedure with the Sensirion flow sensor, verifying I²C communication and Bluetooth data transmission. **Presentation of Results:** Table of I²C addresses, data packets, and

	timestamps captured by the logic analyzer. Screenshot of Bluetooth data received on the GUI. Annotated wiring diagram of test setup.

Table 1: Microcontroller Subsystem - Requirements & Verification

2.3.2. Power Subsystem

The Power System Circuit ensures a stable and reliable power supply to all subsystems, enabling continuous operation of the microcontroller, sensors, solenoid valve, external ADC, and other measurement circuitry. Given the medical nature of the system, power reliability is crucial.

We will use four linear voltage regulators to step down a 14V input voltage from an external power supply connected to a wall outlet. A 3.3V linear regulator will power the flow rate sensor, while a 12V linear regulator will supply the buffer op-amp, pressure sensor, and solenoid. The solenoid consumes 6W of power, drawing 500mA of current; therefore, we will use a 12V 1.5A regulator from Analog Devices. Additionally, a 5V linear regulator will power the external ADC and ESP32 microcontroller, and a -5V linear regulator will supply the negative voltage rail of the buffer op-amp.

We chose linear regulators over switching regulators, such as buck converters, because minimizing noise on the board is our primary concern. Switching regulators introduce ripple noise on their output, which can pollute the ground plane. This noise could significantly reduce the precision and accuracy of our

ADC, which is critical for meeting the design specifications. Since our system is not battery-powered, power efficiency is not a limiting factor, making linear regulators the preferred choice for ensuring clean and stable power delivery.

Requirements	Verification
● Must supply at least 25mA continuously at 5V ± 0.1V to support all components.	**1. Equipment:** Digital Multimeter Oscilloscope Electronic Load ESP32 External ADC **2. Test Procedures:** **Voltage Output Verification:** Connect the 14V power supply to the input of the 5V linear regulator. Use a DMM to measure the output voltage of the regulator at no load and under a 25mA load. Ensure the voltage remains within the 5V ± 0.1V tolerance range. **Current Supply Verification:** Connect an adjustable electronic load to draw 25mA from the 5V regulator. Gradually increase the load and verify that the voltage remains stable until reaching the maximum expected system load. Record the voltage and current values. **3. Presentation of Results:** Results will be presented in a table format showing measured voltage at different load conditions. Oscilloscope screenshots will capture

	voltage stability over time.
• Must have low dropout voltage regulators to maintain stable operation	**1. Equipment:** Digital Multimeter Oscilloscope **2. Test Procedures:** Gradually reduce the input voltage while monitoring the output voltage of each regulator. Record the minimum input voltage at which the regulator maintains stable operation within specification. Compare the measured dropout voltage with the regulator's datasheet values**.** **3. Presentation of Results:** Results will be tabulated showing dropout voltage measurements for each regulator. Graphs may be included to illustrate voltage regulation behavior under varying input conditions.

Table 2: Power Subsystem - Requirements & Verification

2.3.3. Drainage Subsystem (Solenoid Valve)

The Solenoid Valve regulates CSF flow based on pressure readings, ensuring automated drainage only when necessary. It opens when intracranial pressure (ICP) exceeds the user-set maximum threshold and closes when pressure returns to the minimum user-set threshold, preventing overdrainage.

Our solenoid valve is normally closed (NC), meaning it remains closed by default. In the event of a sudden power loss, the valve will automatically close, preventing

uncontrolled drainage. The solenoid operates on a 12V DC power supply with a power rating of 14W, requiring an input current of 500mA.

A switch will be placed between the 12V rail and the solenoid input, controlled by an output signal from the microcontroller. This allows the system to toggle the solenoid valve on or off as needed for precise CSF flow regulation.

Requirements	Verification
● The solenoid must have a response time less than 1 second when receiving a HIGH/Low signal.	**1. Equipment:** Oscilloscope Function Generation **2. Test Procedures:** Connect a function generator to the solenoid switch. Use a square wave with the pulsing frequency being sufficiently low such that both the “on” and “off” duration is greater than 500ms. Place a shunt resistor between the solenoid and the switch output. Measure the voltage across the shunt resistor. There should be a voltage spike when it is turned on/off as it is an inductive load. Record the response times over multiple trials. **3. Presentation of Results:** Table of input signal timestamps and measured solenoid response time for both opening and closing. Oscilloscope screenshots showing the voltage change when the solenoid is opening and closing.. Summary report verifying whether the 1 second response time requirement is met.

Table 3: Drainage Subsystem - Requirements & Verification

2.3.4. Pressure Subsystem

The Pressure Transducer measures intracranial pressure (ICP) within the CSF drainage catheter and provides real-time data to the microcontroller, which determines whether drainage is required.

We are using the Cole-Parmer EW-68075-03 pressure transducer, which has a measurement range of 0 to 2 psi (0 to 103.43 mmHg) and an output current of 4–20mA, proportional to the pressure measurement. It features a high accuracy of ±0.16mA and will be powered by a 12V linear regulator, though it supports an input voltage range of 9–30V DC.

This transducer is a capacitive pressure sensor, meaning it measures pressure by detecting changes in capacitance across a flexible diaphragm. As the diaphragm deforms under pressure from the directly contacting CSF, the capacitance changes, allowing the sensor to determine the pressure exerted by the fluid. The transducer will be connected to a three-way tee, ensuring direct contact with the CSF, which is essential for accurate pressure readings. Given that pressure fluctuations as small as 1 mmHg (0.019 psi) need to be detected, a high-precision sensor is necessary.

To convert the transducer's current output into a voltage signal, a 200-ohm resistor will be used, ensuring that the maximum output voltage does not exceed 4V. This allows us to fully utilize the ADS1115 ADC's ±4.096V full-scale range. The voltage across the resistor (with one end connected to ground) will be fed into the ADC via a high-precision op-amp in a unity gain configuration, which buffers the signal. This setup provides a low-impedance input signal to the ADC, preventing issues related to interaction between the ADC and the measurement circuit.

Requirements	Verification
• Must have an accuracy of at least ±1 mmHg for precise pressure regulation.	**1. Equipment:** Mass Flow Controller Digital Multimeter (DMM) Cole-Parmer EW-68075-03 Pressure Transducer Oscilloscope Vacuum pump Needle valve Computer **2. Test Procedures:** The pressure within the tee will be set using a mass flow controller which will flow in an inert gas such as nitrogen or argon. In place of the solenoid the output of the tee will be connected to a vacuum pump. It is necessary to use a vacuum pump to bring down the pressure within the tee as the system is designed to measure pressures which are below atmospheric pressure. The base pressure will be controlled using a needle valve to ensure that we do not pump out more of the gas used than necessary as it will allow us to control our base pressure. Gas can be used in place of a fluid for measuring the verifying that our pressure transducer works as it is specified to be capable of measuring both gas and fluid pressure. We will measure the voltage at the output across the measurement resistor(200 ohms) to verify that the current measurement is accurately being measured. This will then be compared to the value being read by the microcontroller to verify that the active components of the measurement circuitry is working correctly on a computer. This process will be swept across the full pressure range(1-103 mmHg) of our pressure transducer to verify that it works correctly across the entire pressure range. It will be performed in 5 mmHg

	increments. **3. Presentation of Results:** Table of applied pressure vs. measured output voltage using DMM, and measured voltage on microcontroller. Calculated error values in mmHg. Graph of measured vs. expected pressure values.
• Must operate across the full range of the pressure transducer which is 1-103 mmHg	**1. Equipment:** Precision Pressure Calibrator Digital Multimeter Cole-Parmer EW-68075-03 Pressure Transducer 200-ohm Precision Resistor **2. Test Procedures:** Connect the pressure transducer to the precision pressure calibrator. Apply pressure values from 0 to 50 mmHg in 5 mmHg increments. Measure the transducer's output current at each pressure step using a DMM. Convert the current readings to voltage using the 200-ohm resistor. Verify that the output is within the expected range for each pressure level. **3. Presentation of Results:** Table of applied pressure vs. measured output voltage. Summary report confirming that the sensor operates within the required range.

Table 4: Pressure Subsystem - Requirements & Verification

2.3.5. Flow Control Subsystem

The Flow Rate Sensor monitors CSF drainage to prevent excessive flow (>10 ccs/hour). If the flow rate exceeds the user-set threshold, the microcontroller will pass a signal into a voltage controlled switch. The switch will be connected between the 12V voltage rail and the solenoid. The default stage of the switch when its input signal is low is to remain closed. When a voltage is applied to the control terminal the switch closes delivering voltage to the solenoid opening it.

We are using the Sensirion SLF3S-0600F, a low-power liquid flow sensor that operates based on differential pressure sensing. It outputs data as a 16-bit digital value via the I²C interface, where the flow rate (µL/min) is directly proportional to the output value. The sensor has a pressure measurement range of -600 Pa to +600 Pa and features high accuracy, with an offset error of ±0.5 Pa and a full-scale error of ±3 Pa, ensuring precise monitoring of CSF flow.

The SLF3S-0600F is typically powered by 3.3V, making it compatible with the ESP32 microcontroller. Its digital output via I²C eliminates the need for external ADC conversion, simplifying integration into the system.

Requirements	Verification
• Must use I²C protocol to interface with the microcontroller	**1. Equipment:** ESP32 Microcontroller Sensirion SLF3S-0600F Flow Sensor Logic Analyzer Oscilloscope Laptop with I²C communication software (Arduino IDE) **2. Test Procedures:** Connect the SLF3S-0600F sensor to the ESP32 microcontroller via I²C (SDA and SCL lines). Power the sensor with 3.3V and ensure proper grounding.

	Use a logic analyzer to capture I²C communication while requesting flow rate data from the sensor. Verify that the ESP32 correctly sends the sensor's I²C address and receives a 16-bit data response. Compare the received data with expected sensor output based on known test conditions. Check for proper acknowledgment signals after each transmitted byte. Repeat the test for different flow rates and confirm successful data acquisition. **3. Presentation of Results:** Logic analyzer waveform screenshots showing correct I²C communication. Table of transmitted and received I²C data values. Summary of any communication errors or missing acknowledgments.
• Must have a sampling rate of at least 1 sample/second to detect sudden flow changes.	**1. Equipment:** ESP32 Microcontroller Sensirion SLF3S-0600F Flow Sensor Logic Analyzer Timer Data logging software **2. Test Procedures:** Configure the ESP32 firmware to read flow rate data from the sensor every second. Start a stopwatch and simultaneously begin data logging. Observe the timestamps of received data packets. Verify that data is sampled at a rate of at least 1 sample per second for a duration of

	at least 5 minutes. Introduce a rapid flow rate change and check if the sensor detects it within 1 second. Repeat the test under different conditions (low and high flow rates) to ensure reliability. **3. Presentation of Results:** Log file with time stamped flow rate readings. Graph showing measured flow rate over time. Summary of any instances where the sampling rate deviated from 1 Hz.
• Must measure flow rates from 0–10 ccs/hour with ±5% accuracy	**1. Equipment:** Sensirion SLF3S-0600F Flow Sensor Computer Micro Pipette Oscilloscope **2. Test Procedures:** Flow a set amount of fluid into the fluid sensor using a micropipette. The flow sensor is rated for 2 mL/min=33uL/sec. On the computer check that the flow rate being measured is accurate. This value will need to be the flow rate over time as the fluid will move through the sensor over a period of time not equal to a second. If the flow rate read on the computer is inaccurate, check using an oscilloscope capable of I2C decoding that the output value from the sensor is correct. Input different amounts of fluid using micropipettes to verify that the sensor works across a range of flow rates. Record the sensor output for each flow rate and compare it with the expected value.

	Cross-check the readings by measuring the actual collected volume over time using a digital scale. Calculate the percentage error and ensure it is within ±5% of the expected flow rate. Repeat the test multiple times to confirm consistency and reliability. **3. Presentation of Results:** Table of expected vs. measured flow rates with percentage error. Graph of measured flow rate vs. expected flow rate. Summary of any deviations beyond ±5% accuracy.

Table 5: Flow Control Subsystem - Requirements & Verification

2.3.6. Graphical User Interface Subsystem

The **Graphical User Interface (GUI)** provides an **interactive display** for medical professionals to **monitor intracranial pressure (ICP), flow rate, and valve status in real time**. It also **logs data** and **triggers alerts** for abnormal pressure levels, ensuring patient safety.

Since we are using the **Espressif ESP32-WROOM-32 microcontroller**, which features **built-in Wi-Fi and Bluetooth**, our GUI will communicate with it via **Bluetooth Low Energy (BLE)**. This connection allows the GUI to **read sensor data** and **send control commands** to the system.

The **GUI will be compatible with Windows, Linux, and macOS**, and will be developed using **Python** with frameworks such as **PyQt, Kivy** (for UI design), and **Bleak** (for BLE connectivity).

Requirements	Verification
• Must display real-time pressure and flow data with at most a 10–30-second delay	**1. Equipment:** Laptop ESP32 Bluetooth Low Energy (BLE) Pressure and flow sensors Timer **2. Test Procedures:** Establish BLE connection between the GUI and ESP32. Simulate pressure and flow changes using controlled inputs. Observe and measure the delay in data updates on the GUI. Repeat for multiple data points and average the results. **3. Presentation of Results:** Table comparing input changes vs. displayed updates with timestamps. Graph showing real-time data responsiveness.
• Must allow users to set custom pressure thresholds for automated control.	**1. Equipment:** Laptop ESP32 Pressure Sensor **2. Test Procedures:** Open GUI and navigate to threshold settings. Input different custom pressure values. Verify that the system responds appropriately **3. Presentation of Results:**

	Screenshot of GUI showing successful threshold updates. Table of input thresholds and corresponding system responses.
• Must provide visual alerts if pressure exceeds safe limits (>20 mmHg).	**1. Equipment:** Laptop ESP32 Pressure sensor **2. Test Procedures:** Simulate pressure readings above 20 mmHg. Observe if the GUI triggers a visual alert. Test multiple pressure values to confirm proper alert functionality **3. Presentation of Results:** Screenshots of alerts at various pressure levels. Table listing test pressures and whether alerts triggered.
• Must support Bluetooth 4.0 or higher for stable connectivity.	**1. Equipment:** Laptop ESP32 Bluetooth signal strength analyzer **2. Test Procedures:** Establish BLE connection and check connection stability. Measure signal strength and data transmission rate. Repeat tests at various distances and obstructions. **3. Presentation of Results:** Table of connection status and signal strength at different distances. Graph of data transmission reliability.

• If the Bluetooth connection is lost, it must reconnect within 5 seconds or notify the user.	**1. Equipment:** Laptop ESP32 BLE **2. Test Procedures:** Manually disconnect Bluetooth and observe reconnection time. Simulate interference and measure recovery time. Verify if GUI provides a notification if reconnection fails within 5 seconds. **3. Presentation of Results:** Table listing disconnection events and reconnection times. Screenshot of GUI notification message

Table 6: Graphical User Interface Subsystem - Requirements & Verification

2.4. Tolerance Analysis

2.4.1. External ADC

The Texas Instruments ADS1115 has a resolution of 16 bits, and we will use the ±4.096V input range. Since we are using a 200Ω measurement resistor and our pressure transducer has a current span of 16mA, the voltage step per mmHg is calculated as:

(0.016 A) * 200 Ω / 517.15 mmHg = 6.19 mV/mmHg.

Using this, we determine the number of bits corresponding to our voltage step. The resolution of an ADC is given by:

$(VH - VL) / 2^n = (4.096V - (-4.096V)) / 2^n = 8.192V / 2^n$

Setting the two equations equal to each other to solve for n:

$n = \log_2(8.192V / 6.19mV) = 10.36$ bits.

Based on the table below, we can see that losing up to 5–6 bits of resolution still allows for sufficient accuracy in resolving pressure changes on the order of 1 mmHg.

Number of Bits	Resolution(mV)
16	0.125
15	0.25
14	0.5
13	1
12	2
11	4
10	8
9	16
8	32

Table 7: Relationship between ADC bit depth and voltage resolution

2.4.2. Pressure Transducer

The pressure transducer has a full scale accuracy range of ± 0.25%. This means that the full range accuracy of our output current is ± 16mA * 0.0025 = ± 40uA. This corresponds to a full range error of ± 0.258575 mmHg. This is an error of approximately 25.6 % if we are looking to achieve a pressure resolution of 1 mmHg. This is a significant level of change relative to our desired resolution which ideally can be corrected for by determining the current offset using a load line of a current vs pressure measurement. A potential issue that

may occur with this method of correction is the transducer being insufficiently linear to be able to correct for the error in this manner.

2.4.3. Flow Sensor

The Sensirion SLF3S-0600F flow sensor operates via differential pressure sensing, measuring flow rate as a 16-bit digital value via I²C, with a pressure range of -4.50 mmHg to +4.50 mmHg and an accuracy of ± 0.00375 mmHg (offset error) and ± 0.0225 mmHg (full-scale error). Since flow rate is proportional to differential pressure, any pressure error directly affects the flow rate measurement. To quantify this impact, we calculate the flow rate deviation caused by the offset error (± 0.00375 mmHg) using the formula: Flow Rate = k × Pressure, where k is the proportionality constant based on the sensor's calibration. Given that full-scale pressure (4.50 mmHg) corresponds to a full-scale flow rate (6000 µL/min), we calculate k as follows:

$$k = 6000\ \mu L/min / 4.50\ mmHg = 1333.33\ \mu L/min/mmHg$$

Thus, the flow rate deviation due to the offset error is:

$$(6000\ \mu L/min / 4.50\ mmHg) \times 0.00375\ mmHg = 5\ \mu L/min$$

This means the offset error introduces a ±5 µL/min shift in baseline readings. Similarly, at full-scale pressure (4.50 mmHg corresponding to 6000 µL/min flow rate), the ±0.0225 mmHg full-scale error results in a ±30 µL/min deviation. However, at the 10 cc/hour (100 µL/min) threshold, the error contribution is only:

(100 µL/min / 4.50 mmHg) × 0.0225 mmHg = 0.5 µL/min

which is negligible. The offset error (±0.00375 mmHg) contributes:

(100 µL/min / 4.50 mmHg) × 0.00375 mmHg = 0.083 µL/min

making it insignificant to system performance. In a worst-case scenario, an overestimated flow rate may cause the solenoid valve to close slightly early, restricting drainage, while an underestimated flow rate could delay valve closure, allowing brief overdrainage. However, given the sensor's high accuracy, these deviations are minimal and do not pose a risk to system reliability. The SLF3S-0600F meets the required precision for CSF flow monitoring, ensuring accurate flow detection and preventing excessive drainage.

3. Cost and Schedule

3.1. Bill of Materials Cost Breakdown

Component	Manufacturer	Part Number	Quantity	Unit cost($)	Links
Microcontroller	Espressif Systems	ESP32-WROOM-32E-H4	1	$5.06	Link
Pressure Transducer	Cole-Palmer	EW-68075-03	1	$241.82	Link
Flow rate sensor	Sensirion	SLF3S-0600F	1	$144.27	Link
Solenoid	Electric Solenoid Valves	RSC-2-12VDC	1	$17.67	Link
3.3V Linear Regulator	Analog Devices	LT1086CT-3.3#PBF	1	$7.93	Link
5V Linear Regulator	Analog Devices	LT1086CT-5#PBF	1	$7.20	Link
12V Linear Regulator	Analog Devices	LT1086CT-12#PBF	1	$7.20	Link
-5V Linear Regulator	Analog Devices	LT1964ES5-5#TRPBF	1	$5.74	Link

Precision Op-amp	Analog Devices	LTC1052CN8#PBF	1	$11.44	Link
18V 2A Dc Supply	XHUCHINMAL	N/A	1	$12.49	Link
16 Bit ADC	Texas Instruments	ADS1115IDGSR	1	$4.38	Link
200Ω Resistor ±0.01%	Stackpole Electronics Inc	RNCF1206TKY200R	1	$1.77	Link
1/4" NPT 304 Steel Tee	Taisher	N/A	1	$14.99	Link
1/4 NPT Male x 1/4"-28 UNF Male Adapter	McMaster Carr	1859N168	2	$14.87	Link
SLF3X Mounting Clamp	Sensirion AG	N/A	1	$4.58	Link
Voltage Switch	Texas Instruments	TPS22810DBVR	1	$0.49	Link
Heatsink for 12V Linear Regulator	Wakefield-Vette	240118ABHE22	1	$0.71	Link
			Total Cost	$513.20	

Figure 5: Parts List

3.2. Estimated Hours of Development:

All three members of our group are electrical engineering students. The average annual wage for electrical engineering students is $87,769, which equates to an hourly wage of $46. Assuming our work begins from the week the proposal was due and continues through the week before the mock demo, the total duration of work is 9 weeks.

We estimate spending an average of 10 hours per week on the project, resulting in $4,140 of labor cost per team member. The total project cost is calculated as:

$(3 \times 4{,}140) + 513.20 = 12{,}933.20$

Thus the total labor cost for the project is $12,933.20

3.3. Total Cost Analysis

The total cost of the project is the sum of the total labor cost and the total cost of components. Thus, the total cost of the project is $12,933.20 + $513.20 = $13,446.40.

3.4. Schedule

Week	Task	Team Member
2/24/2025	Prototype Parts Ordering	David
	Finalize System Design	Everyone
	Research on I2C protocol communication	Ralph
	Research on GUI Design	Isiah
3/3/2025	Parts Testing (Pressure Sensor, Microcontroller, and External ADC)	Everyone
	Establish connection between Microcontroller and External ADC	Ralph & David
	Complete Lab Access	Everyone
	Structure GUI Design	Isiah
3/10/2025	Start & Finalize PCB Design	Everyone
	Breadboard Demo	Everyone

	Begin Prototype Build	David & Ralph
	Start GUI Programming	Isiah
3/17/2025	Finalize GUI Draft	Isiah
	Research Flow Sensor Communication Protocols	Ralph & David
	Research Microcontroller Bluetooth Protocol	Ralph
3/24/2025	Test Pressure, Flow Sensor	David
	Finalize and Test Prototype	Everyone
	Debug GUI	Isiah & Ralph
3/31/2025	Order 2nd round of PCB (if needed)	Everyone
	Establish Bluetooth Communication with GUI and Microcontroller	Isiah & Ralph
	Debug Prototype Design	David
4/7/2025	Order 3rd round of PCB (if needed)	Everyone
	Start Final Product Assembly	Ralph & David
	Finalize GUI	Isiah
4/14/2025	Final Product Debug	Everyone
	Finalize End Product	Everyone
4/21/2025	Mock Demo	Everyone
4/28/2025	Final Demo	Everyone
	Mock Presentation	Everyone
5/5/2025	Final Presentation	Everyone

Figure 6: Project Schedule

4. Ethics and Safety

Although only a prototype, our External Ventricular Drain (EVD) Automation Project involves automating the drainage of cerebrospinal fluid (CSF). Any malfunction—such as excessive or insufficient drainage—could result in severe patient harm, including brain injury or death. As such, we will adhere to the 2025 IEEE Code of Ethics.

First, our team will ensure the safety and privacy of the public, as stated in IEEE Code 1.1. We are committed to incorporating multiple safety features to prevent failure-related hazards. For example, our solenoid valve control system will include fail-safe mechanisms to prevent excessive CSF drainage in case of microcontroller failure; in other words, the solenoid will automatically close when the voltage is cut. Additionally, an emergency manual bypass valve will be available in case of unexpected system failure, allowing medical professionals to take immediate control. Lastly, we will integrate alerts into our GUI to notify medical professionals of anomalies, such as pressure and flow readings exceeding user-determined ranges.

Regarding IEEE Code 1.5, our team is committed to seeking and accepting honest criticism of our technical work, maintaining transparency, and properly crediting contributions, especially within our course network, including our TA, Jason Jung, supervising professor, Dr. Yang Zhao, and Dr. Suguna Pappu. Any references from external sources used in the project will also be documented and credited accordingly.

Lastly, since our project involves electrical and medical components, we will adhere to university laboratory safety policies and state regulations concerning electrical safety and biohazard handling. Electrical safety is a primary concern, as our system involves microcontrollers, sensors, and solenoid valves that require careful handling.

All testing will be conducted in designated laboratory areas. While developing and testing our system, we will comply with ECEB Lab Safety standards for laboratory safety. Ensuring proper safety measures when working with sensitive medical devices and fluid-based systems. By following these safety protocols, we will ensure a secure and compliant environment for the successful development of our project.

Additionally, since our project involves a medical device prototype, we recognize the importance of adhering to FDA regulations and medical device standards. Although not yet intended for clinical use, our system design follows principles outlined in the FDA's guidance for Class II medical devices, including risk management, software validation, and quality control measures. We will also consider IEC 60601 standards for medical electrical equipment to ensure electrical safety and electromagnetic compatibility. Future iterations of the project may require FDA 510(k) premarket notification to verify substantial equivalence with existing legally marketed devices. By aligning with these medical regulations and considerations, we aim to develop a safe, reliable, and potentially certifiable system that meets rigorous healthcare standards.

References

[1] IEEE, “IEEE Code of Ethics,” ieee.org, https://www.ieee.org/about/corporate/governance/p7-8.html. Accessed 13 Feb. 2025.

[2] Bertuccio, Alessandro et al. “External Ventricular Drainage: A Practical Guide for Neuro-Anesthesiologists.” *Clinics and practice* vol. 13,1 219-229. 31 Jan. 2023, doi:10.3390/clinpract13010020

[3] ECE 445 Safety Guidelines, “Safety :: ECE 445 - Senior Design Laboratory”, https://courses.grainger.illinois.edu/ece445/guidelines/safety.asp. Accessed 13 Feb. 2025.

[4] FDA Class II Medical Devices, “FDA Class II medical devices. (n.d.).”, https://www.rimsys.io/blog/fda-class-ii-medical-devices. Accessed 6 March 2025.