ECE 445

Spring 2025

Final Report

Water Quality Monitoring System

Group #63

Team members:

Jackie Fang (jackief3@illinois.edu)

Haokai Liu (haokail2@illinois.edu)

Harrison Griggs (hgriggs2@illinois.edu)

TA:

Rui Gong

Professor:

Viktor Gruev

Abstract

This project presents a low-cost, IoT-based water quality monitoring system designed for real-time measurement of key parameters such as temperature and turbidity. The system is built around an ESP32 microcontroller and a custom PCB that integrates sensor inputs, power management, and wireless communication. Data is collected and transmitted to a laptop for monitoring. While temperature and turbidity measurements were successfully implemented, some sensors could not be tested due to PCB design issues.

Contents

1 Introduction

1.1 Problem

Water pollution poses a growing global concern, particularly in controlled aquatic environments such as fish tanks and aquaculture systems. Sources of pollution such as industrial waste, agricultural runoff, and inadequate infrastructure contribute to the degradation of freshwater ecosystems. According to the United States Environmental Protection Agency (EPA), nutrient pollution from agricultural runoff is among the most widespread and challenging environmental threats to freshwater ecosystems [10]. These pollutants can have devastating effects on aquatic life, particularly in controlled environments such as fish tanks and fish farms, where maintaining optimal water quality is critical for the health and survival of fish populations.

Traditional methods of monitoring water quality, such as manual sampling and laboratory testing, are time-consuming, labor-intensive, and often fail to provide real-time data. This delay in detecting contamination can lead to irreversible damage to aquatic ecosystems and economic losses for fish farmers. For example, the Food and Agriculture Organization (FAO) highlights that poor water quality is a leading cause of fish mortality in aquaculture, resulting in significant financial losses for farmers [4]. Furthermore, the lack of affordable and scalable solutions for real-time water quality monitoring exacerbates the problem, especially in remote or resource-constrained areas. Without timely intervention, pollutants can accumulate to dangerous levels, leading to fish mortality, ecosystem imbalance, and potential risks to human health if contaminated water is consumed. Therefore, there is an urgent need for an efficient, cost-effective, and scalable system that can continuously monitor water quality parameters and provide actionable insights to prevent contamination and ensure the safety of aquatic environments.

1.2 Solution

To address these challenges, we propose an IoT-based water quality monitoring system designed to provide real-time, actionable insights into maintaining water quality. Our solution features a custom PCB that integrates the ESP32 microcontroller, sensors for pH, turbidity, temperature, dissolved oxygen, and power/communication circuits, ensuring a compact and reliable design. The system measures and estimates critical water parameters in real time and transmits data wirelessly to a cloud dashboard for remote monitoring. Additionally, the system will be designed to be low-cost, portable, and scalable, making it suitable for fish tanks. By combining affordability, real-time data, and ease of use, our solution empowers communities to monitor water quality proactively and prevent contamination risks.

1.3 Overview

The project was not divided up into 3 equal parts during the semester for each group member to work on. Everybody worked on every part of the project a little. Although as the semester went on we focused on specific roles. Jackie was responsible for a lot of the schematic design and soldering, Haokai worked on the schematic design and soldering, Harrison Griggs worked on the code in the ESP32 microcontroller and the communication of the device. There were no block level changes throughout the semester and Figure 1 shown below is our block level diagram.

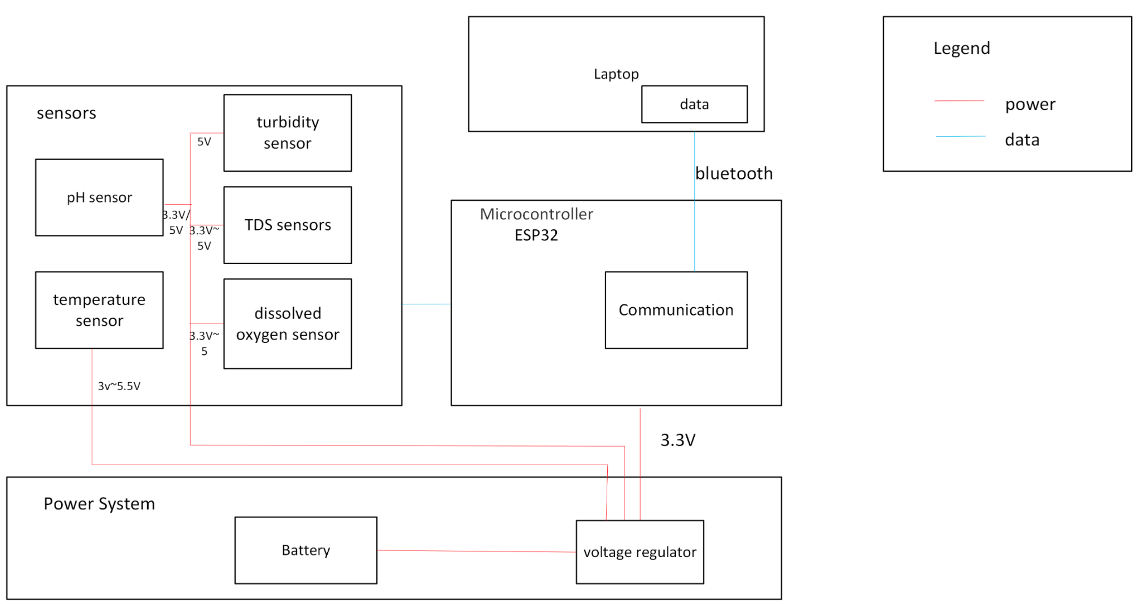

Figure 1: Block Diagram

2. Design

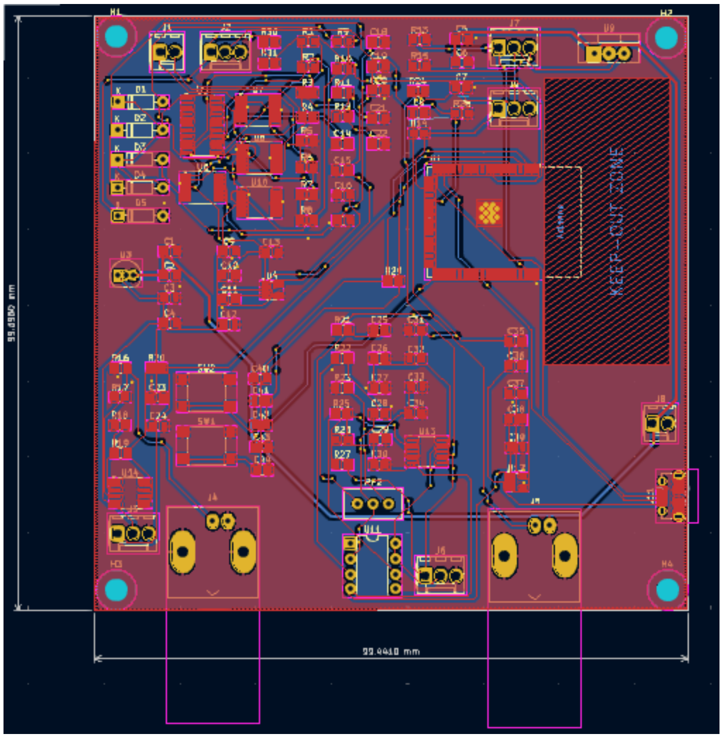

Figure 2: PCB Design

2.1 Design Procedure

The design of our IoT-based water quality monitoring system shown in Figure 2 was guided by three primary goals: accuracy, reliability, and affordability. To meet these goals, the system architecture was divided into four major blocks: microcontroller and computation, sensor array, communication, and power regulation. Each subsystem was evaluated based on available alternatives, and final selections were made through consideration of technical feasibility, cost constraints, and performance trade-offs. One key area of design flexibility is in the sensor subsystem. Although we selected sensors for pH, turbidity, temperature, total dissolved solids (TDS), and dissolved oxygen (DO), the system architecture supports substitution or expansion with other water quality sensors, provided they have similar input/output characteristics. For example, instead of the TDS sensor, it can be replaced by an electrical conductivity sensor using the same circuit designed on the pcb. Similarly, an additional sensor would be a chlorine sensor for environments where disinfectant monitoring is critical. The only requirement is that any additional or alternative sensor must operate within the system's input and output voltage. Another alternative approach to the design would be at the circuit level of the design. Things such as different component values and functionality for voltage dividers, operational amplifiers, etc. The reason we chose these specific sensor set is because this set is tailored for aquatic environments. Furthermore, the component selection is mainly due to availability and manufacturer recommendations.

2.2.1 Microcontroller Subsystem

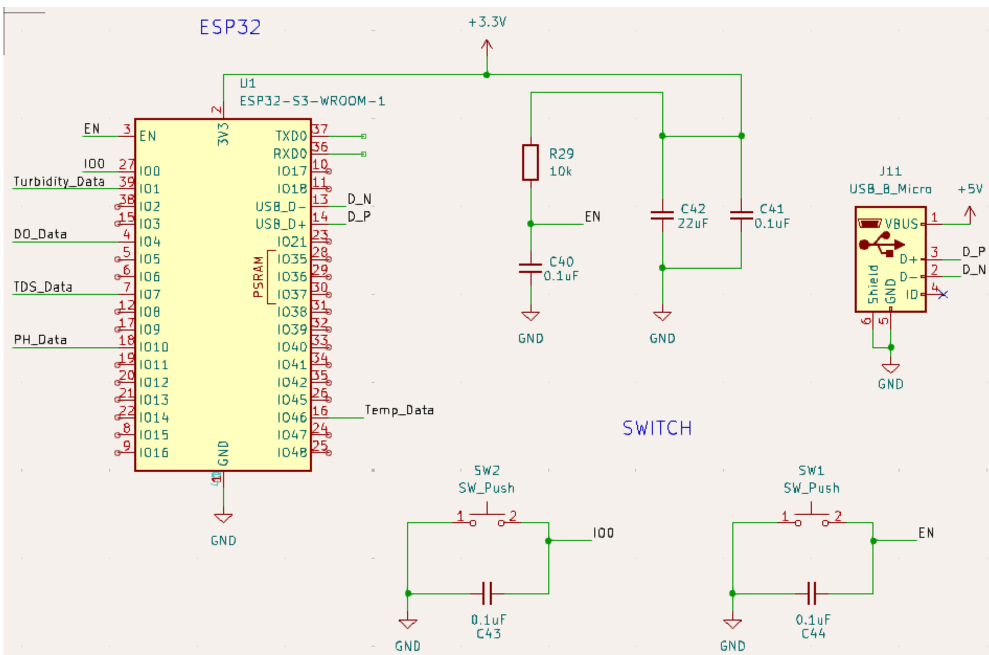

Figure 3: ESP32 Schematic

The ESP32 microcontroller is the central processing unit in the water quality monitoring system, responsible for acquiring, processing, and transmitting data from the sensors. As the core of the system, it has a crucial role in meeting the high-level requirements, particularly the need for real-time data transmission and reliable performance. The ESP32, shown in Figure 3, interfaces with the sensor array, power subsystem, and communication subsystem to ensure that all components operate efficiently. It collects analog and digital signals from the pH, turbidity, temperature, and TDS sensors, converting them into digital values using its ADC. It also optimizes power usage for the battery, ensuring efficient operation with the power subsystem. [3] For communication, the ESP32 supports Bluetooth for data transmission. Bluetooth allows for short-range communication, ideal for local monitoring or connecting to mobile devices or a centralized hub. The ESP32 transmits sensor data to the mobile app or cloud dashboard via Bluetooth, ensuring continuous monitoring. This capability is essential for meeting the real-time data transmission requirement, where data must be sent to the cloud dashboard with less than 5% packet loss.

2.2.2 Sensor Array

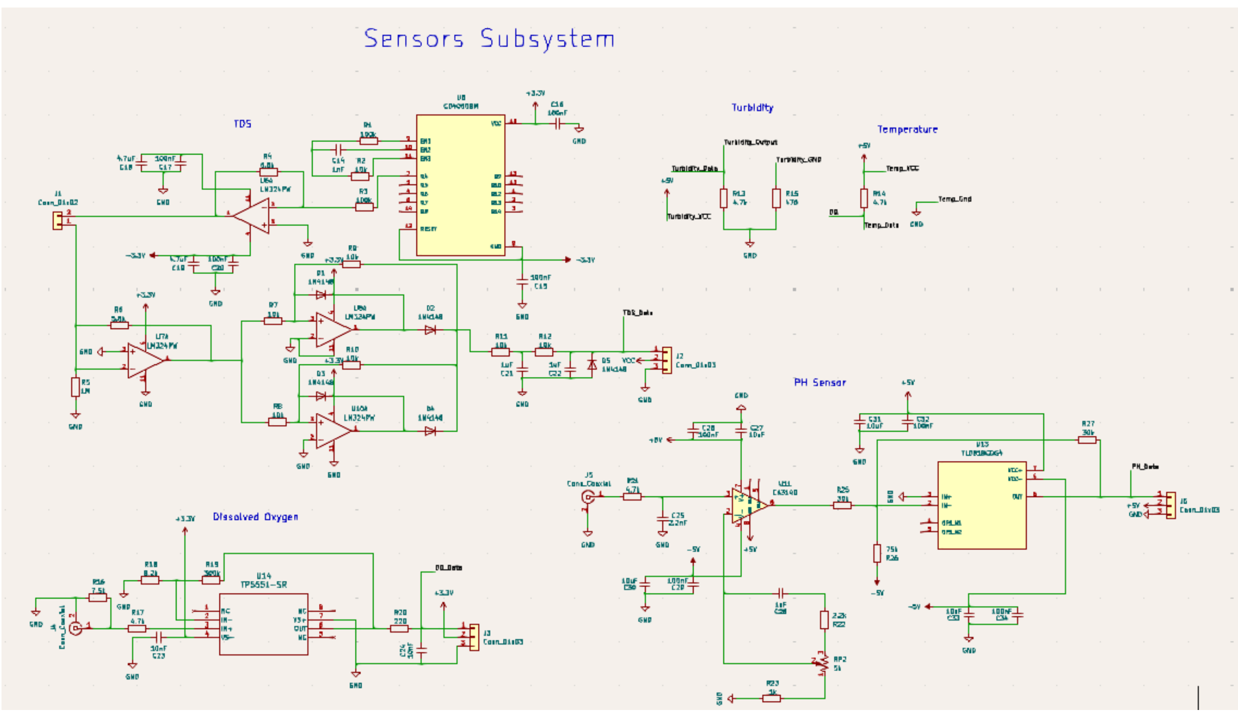

Figure 4: Sensor Subsystem

The Sensor Array Subsystem shown in Figure 4 is responsible for collecting real-time water quality data by measuring key parameters such as pH, turbidity, temperature, dissolved oxygen, and total dissolved solids (TDS). The Sensor Array Subsystem plays a key role in the overall design by providing the data input necessary for real-time monitoring. The sensors convert physical and chemical water parameters into analog or digital signals, which are then processed by the ESP32 microcontroller. These data points contribute directly to the system's ability to meet high-level requirements, including accurate water quality measurement and real-time data transmission for effective monitoring and early detection of contamination risks. The sensors used in the final implementation include the 5016-SRV-PH-ND for pH measurement, the TSD-10 for turbidity, the DS18B20 for temperature, the SEN0244 for TDS, and the DO-BAT for dissolved oxygen. The sensor array provides analog and digital outputs to the ESP32 microcontroller for data acquisition and processing. The microcontroller collects these data points, applies calibration algorithms, and transmits the processed information to the cloud dashboard for remote monitoring. The sensors are powered by the voltage regulator, which provides stable voltage levels to ensure the sensors operate within their specified ranges.

2.2.2.1 ESP32

To ensure accurate digital conversion of analog signals, the ESP32's 12-bit ADC converts voltages from 0 to 3.3 V into 4096 discrete levels [3]. For example, the turbidity and TDS sensors produce output voltages which are first scaled using the ADC conversion formula in Equation (1):

$$V = \text{sensorValue} \times \left(\frac{3.3}{4095.0} \right) \tag{1}$$

2.2.2.2 TDS sensor

For the TDS sensor, this voltage must be temperature-compensated. The system uses the temperature reading from the DS18B20 to calculate a compensation coefficient in Equation (2):

$$\text{compensationCoeff} = 1 + 0.02 \times (T - 25)$$

$$\text{compVoltage} = \frac{V}{\text{compensationCoeff}}$$

$$\text{TDS (ppm)} = \left(133.42 \times \text{compVoltage}^3 - 255.86 \times \text{compVoltage}^2 + 857.39 \times \text{compVoltage}\right) \times 0.5 \tag{2}$$

This polynomial approximation, derived from the sensor manufacturer's guidance, enables accurate estimation of TDS within ±5% of a reference meter.

2.2.2.3 PH Sensor

The pH sensor output is similarly processed using a linear mapping equation shown in Equation (3) based on its datasheet:

$$\mathrm{pH} = 7 + \frac{(V - 2.5)}{-0.18} \tag{3}$$

This formula assumes that a voltage of 2.5 V corresponds to a neutral pH of 7, with a sensitivity of approximately −0.18 V per unit change in pH [8]. The ESP32 microcontroller samples this analog voltage using its 12-bit ADC, which maps the input range of 0 to 3.3 V to digital values from 0 to 4095. The voltage resolution of the ADC shown in Equation (4) is therefore:

$$\Delta V = \frac{3.3\ \mathrm{V}}{4095} \approx 0.000805\ \mathrm{V} \tag{4}$$

Applying this to the pH equation, the smallest resolvable pH difference is shown in Equation (5):

$$\Delta \mathrm{pH} = \frac{0.000805}{0.18} \approx 0.0045 \tag{5}$$

This resolution ensures sufficient precision for detecting small changes in pH within the target ±5% accuracy requirement. To maintain reliability, the pH sensor is calibrated using standard buffer solutions (pH 4.0, 7.0, and 10.0). Temperature compensation is applied using readings from the DS18B20 temperature sensor, and additional filtering is performed in software using a moving average technique to reduce measurement noise.

2.2.2.4 Turbidity Sensor

Our system uses an analog turbidity sensor (e.g., TSD-10) that outputs a voltage proportional to the water turbidity level. The voltage is read by the ESP32's ADC, which converts it into a digital value. The turbidity sensor has analog output in the range of 0 ~ 4.5v. We use Equation

(6) to convert analog output to turbidity value. This process is done by the ESP32 microcontroller.

$$V = \text{sensorValue} \times \left(\frac{3.3}{4095}\right) \tag{6}$$

2.2.3 Communication:

The Communication Subsystem enables data transmission, remote access, and cloud integration for the water quality monitoring system. This ensures real-time monitoring and data storage for further analysis. The Communication Subsystem connects the sensor array to the cloud-based platform, allowing users to remotely monitor and analyze water quality data. This subsystem supports real-time transmission of data collected from the sensors, ensuring that water quality conditions are always up to date and available for assessment. By transmitting data to a cloud server, the system provides the flexibility for remote access, enhancing the scalability and portability of the monitoring system. It also facilitates easy integration with mobile apps and other monitoring tools for end-users. ESP32 Built-in Bluetooth. UART Header for Programming (Through-hole pins). IoT Connectivity: ESP32 for Wi-Fi or LoRa module for long-range communication. Cloud Integration: Data sent to AWS IoT/ThingSpeak for storage and analysis. The ESP32 microcontroller serves as the interface point between the sensor array and the communication subsystem. It collects sensor data, processes it, and sends it to the cloud platform. The Bluetooth communication is handled by the ESP32's internal module, allowing the system to transmit data wirelessly to the mobile device or central hub. The sensor array provides the raw data that is processed by the ESP32 and transmitted through the communication subsystem. The data collected by the sensors is sent as digital packets to the cloud or Bluetooth-enabled device for analysis.

2.2.4 Power System

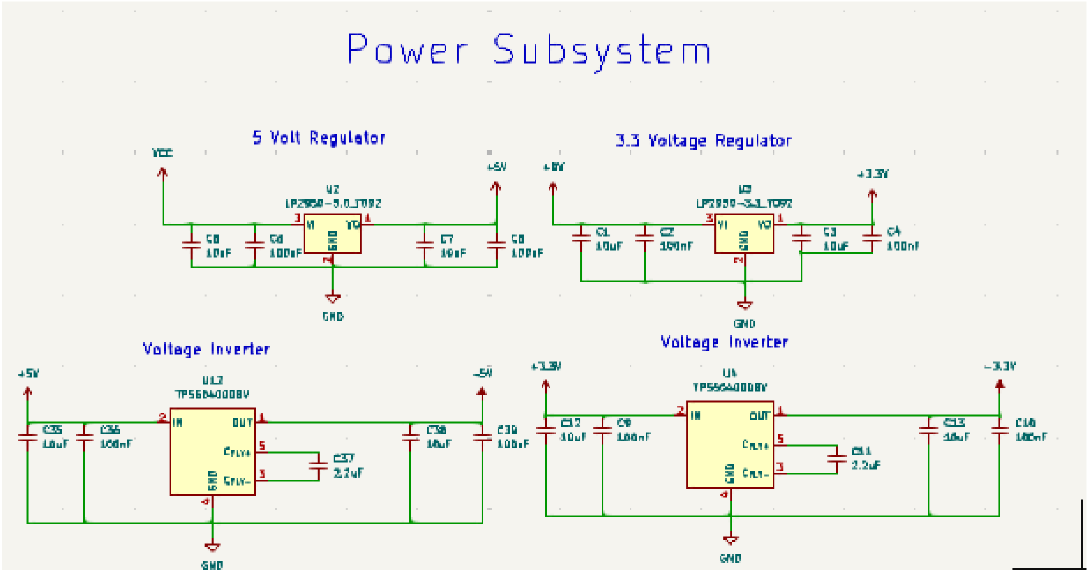

Figure 5: Power Subsystem

The Power Subsystem shown in Figure 5 ensures a stable and reliable energy supply for the water quality monitoring system, supporting battery-powered operation: external to PCB, connected via through-hole terminal block, Wide traces for high-current paths. The power subsystem is critical for maintaining system stability and longevity. It must provide a regulated power supply to all subsystems while ensuring that energy consumption is optimized for long-term operation. Since the system is designed for continuous water quality monitoring, the power subsystem must meet the high-level requirement of enabling the system to operate for at least 24 hours on a fully charged battery. Voltage Regulator (Through-hole for easy soldering). The ESP32 requires a stable 3.3V power supply to function correctly. The power subsystem provides regulated 3.3V output to prevent voltage fluctuations that could cause system instability. The pH, turbidity, TDS, temperature, and DO sensors require 3V, 3.3V or 5V to function. The power subsystem ensures that each sensor receives the appropriate voltage. The voltage regulator distributes power efficiently to avoid voltage drops that could affect sensor accuracy. Bluetooth communication requires bursts of power during data transmission. The power subsystem ensures that the system can handle power spikes when the ESP32 is actively transmitting data while maintaining efficiency during idle periods.

3. Design Verification

The completed system was verified against the core functional objectives of the project, which focused on real-time environmental data acquisition, reliable wireless communication, and sustained autonomous operation. Testing was performed at the subsystem level and is summarized below. The full Requirement and Verification Table is provided in Appendix B.

3.1 ESP32 Verification

The ESP32 microcontroller operated within its specified voltage range (3.0–3.6 V), verified using a digital multimeter at the VCC pin. It successfully acquired and processed data from all connected sensors, with logged data showing ≥95% accuracy where applicable. Simultaneous sensor polling over a continuous 10-minute window resulted in no dropped readings or data corruption. Additionally, the ESP32 was able to transmit sensor data at 30-minute intervals over a 24-hour test.

3.2 Sensor Subsystem Verification

Among the five sensors integrated into the system, only the DS18B20 temperature sensor passed all verification criteria, maintaining accuracy within ±0.5°C across the tested range. The pH, turbidity, and TDS sensors failed verification due to signal integrity issues introduced by the custom PCB design. Despite producing voltage outputs, the readings from these sensors did not correspond to expected values based on known calibration standards. This suggests potential layout, grounding, or analog path problems. The dissolved oxygen sensor was excluded from testing due to unavailability of an affordable working module. However, interference and cross-talk testing confirmed that sensor readings, where available, remained stable and unaffected by simultaneous operation of other sensors.

3.3 Communication Subsystem Verification

Bluetooth communication between the ESP32 and a laptop application was established reliably within five seconds, passing the connection latency test. UART communication for serial debugging and firmware flashing also functioned without issue. Furthermore, the ESP32 was successfully tested for offline data buffering, using local storage to preserve sensor data during network outages and retransmitting the data when connectivity was restored.

3.4 Power Subsystem Verification

The power system was validated for continuous operation and fault tolerance. The voltage regulator provided stable current at 3.0 V, 3.3 V, and 5.0 V, supplying over 500 mA without voltage deviation beyond ±0.1 V. The fully assembled system ran for 26 continuous hours on a single battery charge, exceeding the intended 24-hour battery life goal. The circuit also passed the short-circuit protection test, with the regulator entering a safe mode during fault conditions and resuming normal operation without damage.

3.5 Verification Summary

Of the major functional requirements tested, the majority were verified successfully. Notable exceptions were found in the sensor subsystem, where pH, turbidity, and TDS sensors failed to produce valid outputs, and a dissolved oxygen sensor was unavailable. These issues are documented in Appendix B and will be addressed in future hardware revisions. Nonetheless, the overall system architecture such as control, communication, power, and data handling performed as intended, confirming the viability of the design.

4 Costs

4.1 Bill of Materials

4.1.1 Sensors

Description	Part #	Reference Sheet	Purchase Link	Cost
pH Sensor	KIB-87	pH Sensor	pH Sensor	$17.99
Total Dissolved Solids Sensor	ef74534f-275e-4c61-8060-b8eec	Total Dissolved Solids Sensor	Total Dissolved Solids Sensor	$4.26

	5ca4011			
Total Dissolved Oxygen Sensor	DO-BTA	Total Dissolved Oxygen Sensor	Total Dissolved Oxygen Sensor	$24.99
Turbidity Sensor	TSD-10	Turbidity Sensor	Turbidity Sensor	$8.96
Temperature Sensor	DS18B20 LM35DZ	Temperature Sensor	Temperature Sensor	$9.95
Total Cost				**$66.15**

4.1.2 Resistors and Capacitors

Description	Manufacturer	Quantity	Unit Price	Cost
10k Ohms resistor	Stackpole Electronics Inc	7	$0.10	$0.70
4.7k Ohms resistor	Stackpole Electronics Inc	4	$0.10	$0.40
6.8k Ohms resistor	Stackpole Electronics Inc	1	$0.10	$0.10
100k Ohms resistor	Stackpole Electronics Inc	2	$0.10	$0.10
5.6k Ohms resistor	Stackpole Electronics Inc	1	$0.10	$0.10

7.5k Ohms resistor	Stackpole Electronics Inc	1	$0.10	$0.10
8.2k Ohms resistor	Stackpole Electronics Inc	1	$0.10	$0.10
300k Ohms resistor	Stackpole Electronics Inc	1	$0.10	$0.10
30k Ohms resistor	Stackpole Electronics Inc	1	$0.10	$0.10
75k Ohms resistor	Stackpole Electronics Inc	1	$0.10	$0.10
470 Ohms resistor	Stackpole Electronics Inc	1	$0.10	$0.10
220 Ohms resistor	Stackpole Electronics Inc	1	$0.88	$0.88
1M Ohms resistor	Stackpole Electronics Inc	1	$2.67	$2.67
4.7 uF capacitor	Cal-Chip Electronics, Inc.	2	$0.0045	$0.0090
100 nF capacitor	Cal-Chip Electronics, Inc.	16	$0.0010	$0.0160
1 nF capacitor	Cal-Chip Electronics, Inc.	1	$0.0015	$0.0015
10 uF capacitor	Cal-Chip Electronics, Inc.	14	$0.0090	$0.1260

2.2 uF capacitor	Cal-Chip Electronics, Inc.	2	$0.0040	$0.0080
10 nF capacitor	Cal-Chip Electronics, Inc.	2	$0.0013	$0.0026
1 uF capacitor	Cal-Chip Electronics, Inc	2	$0.0020	$0.0040
2.2 nF capacitor	Cal-Chip Electronics, Inc.	1	$0.0021	$0.0021
Total Cost				**$5.72**

4.1.3 Other

Description	Part #	Quantity	Cost per Unit
Diode	1N4148	5	$0.1
Microcontroller	ESP32	1	$5.92
Potentiometer	3296W-1-502	1	$2.90
BNC Connector	031-5540	2	$6.30
XH connector	xh2.54-2p	1	$0.95
3 Position Female Dupont	S7036-ND	5	0.266
Clock Generator	CD4060	1	0.564

Op-Amp	CD3140E	1	$3.70
Op-Amp	TP5551-SR	1	$3
Op-Amp	LM324	4	$0.01
Op-Amp	TL081CDR	1	$0.067
Charge Pump Inverter	TPS60400DBVR	2	$1
Linear Voltage Regulator	LP2950CZ	2	$1.50
Total Cost			**$36.571**

4.2 Cost Analysis

The total cost above for all the components is $81.287. For the labor cost, we can expect:

$$\$45\, per\ hour\ \times\ 15\, hours\ per\ week\ \times\ 12\, weeks\ \times\ 2.5\ =\ \$20250$$

The total labor cost for all team members is:

$$\$20250\ \times\ 3\ =\ \$60750$$

The total cost for this project is:

$$\$60750\ +\ \$81.287\ =\ \$60831.287\ \approx\ \$60831.29$$

5 Conclusions

5.1 Accomplishment

In this project, we successfully designed and partially implemented an IoT-based water quality monitoring system. We were able to integrate and test the temperature and turbidity sensors, collect real-time data, and visualize it through a computer interface. Our custom PCB was built and powered correctly, and wireless communication via the ESP32 was verified.

5.2 Uncertainties

Some aspects of the system could not be fully tested due to issues in the PCB design. As a result, pH and TDS measurements could not be validated. There is also uncertainty in the long-term accuracy of sensor readings in natural environments, especially without calibration or environmental protection.

5.3 Ethical considerations

Our project aligns with the IEEE Codes of Ethics by promoting environmental sustainability and public welfare. Monitoring water quality helps prevent contamination, protect aquatic life, and support healthy ecosystems. Ethically, we recognize the importance of accuracy and reliability in environmental sensing systems, especially when data may inform real-world decisions. We also ensured that our system design avoids harm, follows safety guidelines, and respects data privacy by keeping monitoring local.

5.4 Further work

For future development, we recommend redesigning the PCB to fix layout errors, adding test points, and using modular connectors for easier debugging. Additional sensors should be integrated and tested one at a time to confirm full functionality. We also suggest using a better designed case to protect PCB and a better probe to combine all the sensors together, and building a mobile or web-based dashboard for remote data access.

6 References

[1] Dfrobot. “”Turbidity Sensor SKU: Sen0189”.” (2017). [Online]. Available: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/2555/SEN0189_Web.pdf (visited on 02/12/2025)

[2] Dfrobot. “”Gravity: Analog TDS Sensor / Meter For Arduino SKU: SEN0244”.” (n.d). [Online]. Available: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/2309/SEN0244_Web.pdf (visited on 02/12/2025)

[3] Espressif. “”ESP32 Series Datasheet Version 4.8“.” (2025). [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf (visited on 02/12/2025)

[4] Food and Agriculture Organization (FAO). “Aquaculture Challenges.” [Online]. Available: https://www.fao.org/fishery/en (visited on 03/03/2025).

[5] Honeywell. “”Temperature Sensors”.” (2001). [Online]. Available: https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/4845/DS_1482_TD%20Series.pdf (visited on 02/12/2025)

[6] IEEE. “”IEEE Code of Ethics”.” (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 02/13/2025).

[7] NanJing Top Power ASIC Corp. “”TP4056 1A Standalone Linear Li-lon Battery Charger with Thermal Regulation in SOP-8”.” (n.d). [Online]. https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/TP4056.pdf (visited on 02/12/2025)

[8] Press, J., & Press, N. ““‘Analog PH sensor / meter - atlas scientific”.” (2024), [Online]. Available: https://files.atlas-scientific.com/Gravity-pH-datasheet.pdf (visited on 02/12/2025)

[9] Press, J., & Press, N. ““EZO-DO Embedded Dissolved Oxygen Circuit - atlas scientific”.” (2024), [Online]. Available: https://files.atlas-scientific.com/DO_EZO_Datasheet.pdf (visited on 02/12/2025)

[10] United States Environmental Protection Agency (EPA). “Nutrient Pollution: The Problem.” [Online]. Available: https://www.epa.gov/nutrientpollution/problem (visited on 03/03/2025).

Appendix A Abbreviations

Unit or Term	Symbol or Abbreviation
alternating current	ac
American wire gauge	AWG
ampere	A
ampere-hour	Ah
amplitude modulation	AM
angstrom	Å
antilogarithm	antilog
atomic mass unit (unified)	u
audio frequency	AF
automatic frequency control	AFC
automatic gain control	AGC
automatic volume control	AVC
average	avg
backward-wave oscillator	BWO
bar	bar
barn	b
beat-frequency oscillator	BFO
bel	B
billion electronvolts*	BeV
binary coded decimal	BCD
bit	b
British thermal unit	Btu
byte	B
calorie	cal
candela	cd
candela per square foot	cd/ft^2
candela per square meter	cd/m^2
cathode-ray oscilloscope	CRO
cathode-ray tube	CRT
centimeter	cm
centimeter-gram-second	CGS
circular mil	cmil
continuous wave	CW
coulomb	C
cubic centimeter	cm^3
cubic foot per minute	ft^3/min
cubic meter	m^3
cubic meter per second	m^3/s
curie	Ci
cycle per second	Hz
decibel	dB
decibel referred to one milliwatt	dBm
degree Celsius	°C
degree Fahrenheit	°F
degree Kelvin**	K
degree (plane angle)	...°
degree Rankine	°R
degree (temperature interval or difference)	deg
diameter	diam
direct current	dc
double sideband	DSB
dyne	dyn
electrocardiograph	EKG
electroencephalograph	EEG
electromagnetic compatibility	EMC
electromagnetic unit	EMU

*Deprecated: use gigaelectronvolt (GeV).
**Preferably called simply *kelvin*.

Unit or Term	Symbol or Abbreviation
electromotive force	EMF
electronvolt	eV
electrostatic unit	ESU
erg	erg
extra-high voltage	EHV
extremely high frequency	EHF
extremely low frequency	ELF
farad	F
field-effect transistor	FET
foot	ft
footlambert	FL
foot per minute	ft/min
foot per second	ft/s
foot-poundal	ft-pdl
foot pound-force	ft•lbf
frequency modulation	FM
frequency-shift keying	FSK
gallon	gal
gallon per minute	gal/min
gauss	G
gigacycle per second	Gc/s
gigaelectronvolt	GeV
gigahertz	GHz
gilbert	Gb
gram	g
henry	H
hertz	Hz
high frequency	HF
high voltage	HV
horsepower	hp
hour	h
inch	in
inch per second	in/s
inductance-capacitance	LC
infrared	IR
inside diameter	ID
intermediate frequency	IF
joule	J
joule per degree	J/deg
joule per kelvin	J/K
kilobit per second	kb/s
kilobyte	kB
kilocycle per second	kHz/s
kiloelectronvolt	keV
kilogauss	kG
kilogram	kg
kilogram-force	kgf
kilohertz	kHz
kilohm	kΩ
kilojoule	kJ
kilometer	km
kilometer per hour	km/h
kilovar	kvar
kilovolt	kV
kilovoltampere	kVA
kilowatt	kW

Unit or Term	Symbol or Abbreviation
kilowatthour	kWh
lambert	L
liter	l
liter per second	l/s
logarithm	log
logarithm, natural	ln
low frequency	LF
lumen	lm
lumen per square foot	lm/ft^2
lumen per square meter	lm/m^2
lumen per watt	lm/W
lumen-second	lm•s
lux	lx
magnetohydrodynamics	MHD
magnetomotive force	MMF
maxwell	Mx
medium frequency	MF
megacycle per second	MHz/s
megaelectronvolt	MeV
megahertz	MHz
megavolt	MV
megohm	MΩ
metal-oxide semiconductor	MOS
meter	m
microampere	μA
microfarad	μF
microgram	μg
microhenry	μH
micrometer	μm
micron†	μ
microsecond	μs
microsiemens	μS
microwatt	μW
mil	mil
mile per hour	mi/h
mile (statute)	mi
milliampere	mA
milligram	mg
millihenry	mH
milliliter	ml
millimeter	mm
millimeter of mercury, conventional	mmHg
millimicron‡	nm
millisecond	ms
millisiemens	mS
millivolt	mV
milliwatt	mW
minute (plane angle)	...'
minute (time)	min
nanoampere	nA
nanofarad	nF
nanometer	nm
nanosecond	ns
nanowatt	nW
nautical mile	nmi

†The name *micrometer* (μm) is preferred.
‡The name *nanometer* is preferred.

Unit or Term	Symbol or Abbreviation
neper	Np
newton	N
newton meter	N•m
newton per square meter	N/m^2
oersted	Oe
ohm	Ω
ounce (avoirdupois)	oz
outside diameter	OD
phase modulation	PM
picoampere	pA
picofarad	pF
picosecond	ps
picowatt	pW
pound	lb
poundal	pdl
pound-force	lbf
pound-force foot	lbf-ft
pound-force per square inch	lbf/in^2
pound per square inch§	psi
power factor	PF
private branch exchange	PBX
pulse-amplitude modulation	PAM
pulse code modulation	PCM
pulse count modulation	PCM
pulse duration modulation	PDM
pulse position modulation	PPM
pulse repetition frequency	PRF
pulse-repetition rate	PRR
pulse-time modulation	PTM
pulse-width modulation	PWM
radian	rad
radio frequency	RF
radio-frequency interference	RFI
resistance-capacitance	RC
resistance-inductance-capacitance	RLC
revolution per minute	r/min
revolution per second	r/s
roentgen	R
root-mean-square	rms
second (plane angle)	..."
second (time)	s
short wave	SW
siemens	S
signal-to-noise ratio	SNR
silicon controlled rectifier	SCR
single sideband	SSB
square foot	ft^2
square inch	in^2
square meter	m^2
square yard	yd^2
standing-wave ratio	SWR
steradian	sr
superhigh frequency	SHF
television	TV
television interference	TVI

§Although the use of the abbreviation psi is common, it is not recommended. See pound-force per square inch.

Unit or Term	Symbol or Abbreviation
tesla	T
thin-film transistor	TFT
transverse electric	TE
transverse electromagnetic	TEM
transverse magnetic	TM
traveling-wave tube	TWT
ultrahigh frequency	UHF
ultraviolet	UV
vacuum-tube voltmeter	VTVM
var	var
variable-frequency oscillator	VFO
very-high frequency	VHF
very-low frequency	VLF

Unit or Term	Symbol or Abbreviation
vestigial sideband	VSB
volt	V
voltage controlled oscillator	VCO
voltage standing-wave ratio	VSWR
voltampere	VA
volume unit	vu
watt	W
watthour	Wh
watt per steradian	W/sr
watt per steradian square meter	$W/(sr \cdot m^2)$
weber	Wb
yard	yd

Appendix B Requirements and Verification

ESP32:

Requirements	Verification	Verification Status (Y/N)
The ESP32 must operate within a voltage range of 3V-3.6V	Use a Digital Multimeter (DMM) to measure the voltage supplied to the ESP32 VCC pin.	Y
The ESP32 must correctly acquire and process data from all sensors with an accuracy of at least 95%.	1. Connect the ESP32 to a calibrated reference sensor (e.g., known pH buffer solution, precise temperature source). 2. Log sensor values from the ESP32 and compare against the reference. 3. Compute the error percentage and verify that accuracy is ≥ 95%. 4. Repeat the test at least 10 times for consistency.	Y
The ESP32 must be able to handle at least five sensor inputs simultaneously without data loss.	1. Connect five different sensors (e.g., pH, turbidity, temperature, TDS, dissolved oxygen) to the ESP32. 2. Simultaneously request data from all sensors. 3. Log the	N Was not able to implement all 5 sensors

	readings over a continuous 10-minute period. 4. Check that there are no dropped readings or data corruption in the logs.	
The ESP32 must transmit data every 30 minutes with less than 5% packet loss.	1. Configure the ESP32 to send sensor data to a cloud service every 30 minutes. 2. Monitor transmission logs for 24 hours (48 transmissions). 3. Count the number of successfully received packets and calculate the packet loss percentage. 4. Verify that packet loss ≤ 5% over the test period.	Y

Sensors:

Requirements	Verification	Verification status
The pH sensor must output a voltage that corresponds to pH values between 0 and 14	1. Prepare buffer solutions with pH values of 4, 7, and 10. 2. Use a DMM to measure the voltage output of the pH sensor and verify it falls within the expected range for	N Our pH sensor can’t get correct reading due to pcb design issue

	each buffer solution. 3. Record the voltage and check the linearity between pH values and voltage output.	
The turbidity sensor must measure water turbidity between 0 and 1000 NTU (Nephelometric Turbidity Units) with accuracy within ±5%.	1. Use a standard turbidity solution (e.g., 400 NTU). 2. Connect the turbidity sensor to the ESP32 and measure the turbidity. 3. Compare the sensor's output with the standard turbidity value and confirm that it is within ±5%. 4. Record results and check for consistency across multiple measurements.	N Our turbidity sensor can't get correct reading due to pcb design issue
The temperature sensor must measure water temperature with an accuracy of ±0.5°C over the range of 0°C to 50°C.	1. Place the temperature sensor (DS18B20) in water at known temperatures (e.g., 0°C, 25°C, 50°C). 2. Use a calibrated thermometer to verify the sensor's output. 3. Ensure that the sensor readings are within ±0.5°C of the calibrated temperature.	Y
The TDS sensor must measure Total Dissolved	1. Prepare a standard TDS solution (e.g., 500 ppm). 2.	N Our TDS sensor can't get

Solids (TDS) in the range of 0 to 1000 ppm with accuracy within ±5%.	Use the TDS sensor to measure the solution's TDS and compare it to the known reference value. 3. Ensure the readings fall within ±5% of the reference TDS. 4. Record the results and verify accuracy at multiple TDS concentrations.	correct reading due to pcb design issue
The dissolved oxygen (DO) sensor must measure dissolved oxygen concentration in the range of 0 to 20 mg/L with accuracy of ±2%.	1. Place the DO sensor in a saturated oxygen solution (e.g., water exposed to air). 2. Measure the DO concentration and compare it with the expected value from the manufacturer's specifications. 3. Ensure the sensor output is within ±2% of the expected value. 4. Record the DO readings and verify consistency with known concentrations.	N We couldn’t get a cheap DO sensor online
The sensor array must operate without significant interference or cross-talk between sensors.	1. Test the system by activating multiple sensors simultaneously. 2. Measure the output from each sensor and verify that each sensor provides a stable and	Y

	independent reading without significant deviations or interference from the others. 3. Record results and confirm that sensor outputs remain unaffected by simultaneous operation.	

Communication:

Requirements	Verification	Verification status
The ESP32 must establish a Bluetooth connection with the laptop application within 5 seconds of scanning.	1. Enable Bluetooth discovery mode on the ESP32. 2. Open the laptop application and scan for available Bluetooth devices. 3. Measure the time taken from scan initiation to successful connection. 4. Verify that the connection is established within ≤ 5 seconds.	Y
The ESP32 must support UART communication for debugging and external programming.	1. Connect the ESP32 to a PC via USB-to-serial adapter. 2. Open a serial terminal (e.g., PuTTY, Arduino Serial Monitor) and send test	Y

	commands. 3. Check if ESP32 properly transmits and receives messages over UART. 4. Use ESP32's UART to flash firmware and verify successful programming.	
The ESP32 must store data locally in case of network failure and retry transmission.	1. Simulate Wi-Fi/Bluetooth disconnection while ESP32 attempts to send data. 2. Verify that data is stored in ESP32's SPIFFS, SD card, or EEPROM. 3. Restore network connectivity and confirm that stored data is successfully transmitted to the cloud. 4. Validate that no data is lost in the process.	Y

Power:

Requirements	**Verification**	Verification status
The voltage regulator must supply at least 500mA continuously at 3V, 3.3V, and 5V to power the entire	1. Use a digital multimeter (DMM) or electronic load to measure the current drawn by the system at each voltage	Y

system.	level (3V, 3.3V, 5V). 2. Apply a constant load of 500mA to each output (3V, 3.3V, 5V) and verify that the voltage stays within the specified range (±0.1V). 3. Record the results and check for any significant drop in voltage or instability during the test.	
The system must operate for at least 24 hours on a fully charged battery.	1. Fully charge the battery and ensure it is connected to the system. 2. Run the system with all components operating (sensors, ESP32, communication) for normal operation. 3. Monitor the system and battery voltage every hour using a DMM to ensure it continues running for at least 24 hours without dropping below 3.3V. 4. Record the battery voltage periodically and confirm that the system functions for at least 24 hours under normal conditions.	Y
The power system must handle a sudden short circuit	1. Intentionally short the output terminals of the	Y

without damaging components.	voltage regulator for 5 seconds. 2. Confirm that the regulator enters protection mode (if applicable) and resumes normal operation after the short is removed. 3. Check for overheating or permanent damage to components. 4. Measure the temperature and current to ensure protection features are working as intended.	