

Smart Aquarium Maintenance System with Solar-Powered Energy Supply

Y. Divayana^{1*}, Linawati², I. G. A. P. Raka Agung³, I. G. N. Janardana⁴, D. C. Khrisne⁵, D. S. Saputro⁶, S. Halim⁷, R. B. Kurniawan⁸

^{1,2,3,4,5,6,7,8} Department of Electrical Engineering, Faculty of Engineering, Universitas Udayana, Indonesia

ARTICLE INFO

Article history:

Initial submission

05-09-2026

Received in revised form

17-09-2026

Accepted 19-09-2026

Available online 30-09-2026

Keywords:

Smart Aquarium System,
Internet of Things, ESP32,
Water Quality Monitoring,
Solar Power System

DOI:

<https://doi.org/10.59356/smart-techno.v8i2.246>

ABSTRACT

Aquarium maintenance needed constant monitoring of water quality and equipment condition to keep fish and other aquatic life healthy. Manual maintenance was often inefficient, requiring routine checks of several parameters at once. This study designed and built a smart aquarium maintenance system that monitored water quality, controlled equipment automatically, and drew its power from a renewable energy source. The system used an ESP32 S3 microcontroller, integrated with pH, water temperature, turbidity, and ultrasonic sensors that monitored aquarium conditions continuously in real time. A water pump, relay module, servo motor, Peltier-based cooler, and heater worked together to keep the aquarium environment within range. A solar power plant charged the battery through an MPPT controller, so the system ran without adding to household electricity use. The Android application, named AquaKeep, connected to the hardware through Firebase Realtime Database and let the user monitor and control the system remotely. Repeated testing showed that the temperature, pH, water-level, and turbidity sensors reached average errors of 0.38%, 0.87%, 1.15%, and 1.11%, all within the 5% threshold set for this study. The automatic feeding system

succeeded in all eight tested scenarios. The cooling rate depended on the initial water temperature, ranging from 1.08°C to 2.20°C every 10 minutes. Over 24 hours of typical operation, the system averaged 9.6 W, and separately sustained 17 hours without sunlight under a heavier simulated load. These results showed that the developed system worked accurately, automatically, and sustainably in keeping the aquarium environment stable.

1. INTRODUCTION

Many people keep ornamental freshwater fish as a hobby, and they use a wide range of tools and tank designs to do so. Pet fish kept in aquarium tanks need to be properly set up and maintained, or the fish will suffer a short and unpleasant life (Lin & Tseng, 2019). Fish health depends on close monitoring of water conditions such as pH, which for most ornamental fish should stay between 6 and 8 (Patro et al., 2023). Leftover food and fish waste build up in the tank over time, and this raises the water's turbidity, which can shift the water chemistry away from the range fish need. Poor water quality can stress the fish or even kill them (Ayon et al., 2026), and an unstable water temperature can also disturb their metabolism and behavior (Patro et al., 2023).

A well-maintained aquarium keeps fish alive and healthy, but most owners still maintain their tanks manually or with a simple water filter. This routine takes considerable time and effort, which becomes a problem for owners with a busy schedule and often leads to irregular maintenance. IoT-based aquarium tools exist as an alternative, but the ones with advanced features tend to be expensive. In addition, a tool that combines many maintenance components in one unit draws a large amount of electrical power, and running such a device continuously adds noticeably to household electricity consumption (Utami et al., 2024).

The tools reviewed in the next section each cover only part of this need, and one prior system combines several of these functions in a single unit. This study built on that gap. It designed and built a smart, IoT-based aquarium maintenance system with a solar power supply, adding two-way temperature regulation and pH-based alerting alongside the water-

quality and feeding functions already explored elsewhere. The system provided automatic scheduled feeding, feed-level and water-level monitoring, water temperature regulation, and pH and turbidity display. A companion Android application showed and controlled every function above.

2. LITERATURE REVIEW

Several existing studies point toward the components this system needed. Ayon et al. (2026) developed an ESP32-based smart aquarium system that integrated pH, TDS, temperature, and turbidity sensors with a servo-driven feeder and a water pump; the mechanical components, including the feeder, reached over 97% success across repeated test runs, and the sensor suite reached 96% average accuracy in a 72-hour trial with a 10-liter tank, though the source's own per-sensor breakdown shows this figure is led by temperature (96.5%), with the water-quality sensors alone (pH, TDS, turbidity) averaging closer to 90-92%. Hakim et al. (2019) characterized a SEN0189 turbidity sensor, the same sensor used in this study, against suspended-sediment samples and found a stable, near-linear voltage response across a turbidity range of about 1.9 to 1012 NTU. Widyati et al. (2025) built a combined automatic feeder and temperature-control system using a DS18B20 sensor, a load cell, and an ultrasonic sensor, with an ESP32 for sensing and an Arduino Uno driving the relay, cooling system, heater, and servo motor. Mohd Jais et al. (2024) improved the accuracy of low-cost pH, temperature, dissolved-oxygen, ammonia, and salinity sensors in an aquaculture tank through linear-regression calibration against a professional reference meter, reaching relative errors between 0.27% and 4%. Lin and Tseng (2019) proposed the FishTalk system, built on the IoTtalk platform, which drives aquarium actuators from sensor data in real time and includes an intelligent feeding mechanism designed to avoid over- or under-feeding while letting the owner watch the feeding remotely. Rebecchi et al. (2024) reported a quadratic relationship, $NTU = -1120.4V^2 + 5742.3V - 4352.9$, between voltage and turbidity for a household greywater-reuse system, though the authors note that this curve was adapted from a different, unspecified turbidity sensor rather than derived from their own. Madububa et al. (2020) compared MPPT and PWM solar charge controllers and measured an experimental efficiency of 91.1% for MPPT against 70.4% for PWM, a gain of about 30%. Closest to the present design, Chiagunye et al. (2025) built a solar-powered ESP32 system with a 30 W panel, a lead-acid battery, and a charge controller, combining scheduled feeding with turbidity-triggered draining and refilling through two relay-controlled pumps, and reported average sensor deviations under 2% together with a 100% success rate in restoring water clarity across repeated trials.

Table 1 places these systems side by side. Chiagunye et al. (2025) comes closest to a fully integrated design, combining solar power, automated feeding, and turbidity-triggered water changes in one ESP32-based platform. However, it does not regulate temperature, does not alert on pH, and relies on a fixed-threshold Google Sheets interface rather than a dedicated mobile application. Based on this remaining gap, the present study designed and built an IoT-based aquarium maintenance system that adds two-way temperature regulation, pH-based alerting, and a dedicated Android application on top of a self-sustaining solar power supply.

Table 1. State-of-the-art comparison of aquarium/aquaculture IoT systems

System	Monitored Parameters	Actuators	IoT/Cloud Platform	Automation	Energy Source	Calibration Procedure	Test Duration	Reported Performance
FishTalk (Lin & Tseng, 2019)	Temperature, turbidity (other sensors not detailed in the abstract)	Feeder, pump, heater, fan (generic multi-actuator set)	IoTtalk platform	Real-time actuator control; adaptive feeding to avoid over/under-feeding	Not stated explicitly	Not detailed	Not stated explicitly	Feeding mechanism avoids over/under-feeding (no quantitative figure given)
Ayon et al. (2026)	pH, TDS, temperature, turbidity	Servo feeder, water pump	ESP32 + Blynk	Monitoring, alerting, and automated feeding	Not stated explicitly	Basic sensor function testing	72 hours, 10-liter tank, 5 goldfish	96% average accuracy; sensor >97% mechanical success

System	Monitored Parameters	Actuators	IoT/Cloud Platform	Automation	Energy Source	Calibration Procedure	Test Duration	Reported Performance
Widyati et al. (2025)	Temperature (DS18B20), feed weight (load cell), water level	Automatic feeder, heater, cooling system (relay via Arduino Uno)	ESP32 + Arduino Uno	Automatic temperature and feeding control	Not stated explicitly	Sensor reading adjustment	Not stated explicitly	Temperature and feeding control functioned as intended (no error figure detailed)
Mohd Jais et al. (2024)	Temperature, pH, dissolved oxygen, ammonia, salinity	None (monitoring only)	Arduino Uno + ESP8266	Monitoring and logging only, no actuator control	Not stated explicitly	Linear regression vs. professional reference meter	Aquaculture tank trial (duration not detailed in available text)	Relative error 0.27%–4.0%
Chiagunye et al. (2025)	Temperature (DS18B20), pH, turbidity	Servo feeder (12 h interval), 2 relay-controlled pumps (drain/refill)	ESP32 + Google Sheets (HTTP)	Scheduled feeding + turbidity-triggered draining/refilling	Standalone PV: 30 W panel + charge controller + lead-acid battery	Not detailed	Real-world field test; 7 turbidity-triggered events recorded	Average deviation <2% (temperature/pH/turbidity); 100% success restoring water clarity
AquaKeep (proposed)	Temperature, pH, turbidity, water level (IP67 ultrasonic), feed level	Servo feeder, Peltier cooler, heater, drain pump, refill pump	ESP32-S3 + Firebase Realtime DB + Android app	4 integrated control loops: turbidity-based drain/refill, two-way temperature regulation, pH alerting, scheduled feeding	Standalone PV: 100 Wp panel + MPPT + 70 Ah LiFePO4 battery	Temp./pH/water-level vs. calibrated reference instruments (5 reps/condition); turbidity linear regression, R ² = 0.9999 (40 readings)	5 reps/condition (temp., pH, water level, turbidity); 24 h continuous load logging; 5-day validation across varied weather	Sensor errors 0.38/0.87/1.15/1.11%; 100% feeding success (8 scenarios); cooling 1.08–2.20°C/10min; 230.7 Wh/day typical load; 17 h heavy-load autonomy

3. SYSTEM DESIGN

System Overview

Figure 1 shows the overall system architecture, built around a sensor-cloud-application connection pattern common to IoT systems (Uviase & Kotonya, 2018; Abuseta, 2024). An ESP32 S3 microcontroller worked as the central controller. It was chosen for its built-in Wi-Fi, which let it exchange data in real time with Firebase Realtime Database as the cloud platform between the hardware and the Android application. The ESP32 S3 offered 38 pins, 28 GPIO lines, a 12-bit 8-channel ADC, 16 PWM channels, a clock speed of up to 240 MHz, 520 KB of SRAM, and 4 MB of flash storage, running at a 3.3 V operating voltage.

On the monitoring side, the system read water temperature with a DS18B20 sensor (measurement range -55°C to +125°C, accuracy ±0.5°C between -10°C and +85°C), water acidity with a pH-4502C sensor (0-14 pH range, ±0.01 pH accuracy), and turbidity with a Keystudio turbidity sensor (0-4550 NTU range). A waterproof, IP67-rated A02YYUW ultrasonic sensor measured the water level (range 3-450 cm, 1 mm resolution, UART interface at 9600 bps). An HC-SR04 ultrasonic sensor measured the remaining feed level. The ESP32 read all of these sensors periodically and sent the data to Firebase. The Android application then displayed this data in real time.

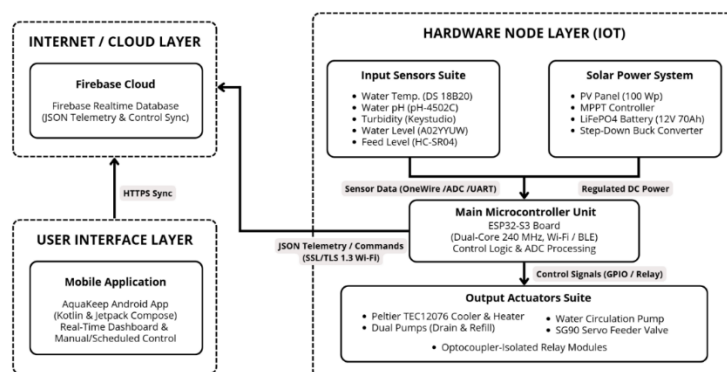


Figure 1. Overall system architecture

On the control side, a relay module switched a thermoelectric (Peltier TEC12076) cooling module and a heater to keep the water temperature within range. A pump handled water circulation, and a servo motor (SG90) opened the feed valve on a fixed schedule or on a manual command from the application. The system drew its power from a solar power plant. A 100 Wp solar panel charged a 70 Ah, 12 V LiFePO4 battery through an MPPT charge controller. The battery then supplied the whole system through a step-down module, so the system could run continuously without adding to the household electrical load.

The system managed four coordinated control loops. First, a turbidity-based draining loop used the turbidity sensor to check water clarity. Once the water was classified as turbid, the ESP32 activated a relay connected to a draining pump. As the water level dropped, the A02YYUW sensor then triggered a second relay to refill the tank with clean water. Second, a temperature-control loop kept a filter pump running continuously through a copper coil attached to the Peltier module. When the DS18B20 sensor reported a temperature rise, the ESP32 activated the relay connected to the Peltier module to cool the water back down. The heater engaged through the same logic whenever the temperature dropped too low. Third, a pH-monitoring loop compared the pH-4502C reading against the 6-8 target range. It pushed an alert to the mobile application whenever the reading fell outside that range. Fourth, a scheduled feeding loop combined the HC-SR04 feed-level sensor with an onboard RTC module. The ESP32 compared the current time against a user-defined feeding schedule. Once they matched, it activated the SG90 servo to release feed, while the HC-SR04 sensor tracked the remaining supply and notified the application once it ran low.

Power System Design

The daily electrical load of the system was calculated component by component. This covered the ESP32 S3 controller, all five sensors, the relay module, the Peltier module, the water pump, the step-down converter, and the MPPT controller, using (1) (Habeeb et al., 2023):

$$\text{Daily Load (Wh)} = \text{Power (W)} \times \text{Duration of Use (h)} \quad (1)$$

This calculation gave a total daily energy consumption of 481.34 Wh. Because the system drew under 1000 W, a 20% safety margin was added in place of a detailed loss calculation, using (2) (Habeeb et al., 2023):

$$\text{Total Energy (Wh)} = \text{Daily Energy (Wh)} + [\text{Daily Energy (Wh)} \times \text{System Loss (\%)}] \quad (2)$$

Applying this margin to the 481.34 Wh daily load gave a total required energy of 577.61 Wh per day. The equivalent sun hours, or Peak Sun Hour (PSH), were estimated from the annual Global Horizontal Irradiance (GHI) for South Denpasar, 2088.3 kWh/m² per year, using (3) (Mertens, 2014):

$$\text{PSH (h)} = \text{Annual Global Horizontal Irradiance (kWh/m}^2\text{)} / (\text{Standard Test Condition} \times 365 \text{ days}) \quad (3)$$

This gave a PSH value of 5.72 hours for the installation site. The number of solar panels needed was then found using (4) (Habeeb et al., 2023):

$$\text{Number of Panels} = \text{Total Energy (Wh)} / [\text{Panel Capacity (Wp)} \times \text{Optimal Sunlight Hours (h)}] \quad (4)$$

With a 100 Wp panel and a PSH of 5.72 hours, this calculation returned 1.01, so a single 100 Wp solar panel was used to meet the system's daily energy demand. Finally, the minimum battery capacity was determined using (5) (Habeeb et al., 2023):

$$C = E_d / (V_s \times DOD) \quad (5)$$

where C is the battery capacity in ampere-hours, Ed is the daily energy consumption in Wh, Vs is the battery voltage, and DOD is the maximum allowable depth of discharge. With Ed = 577.61 Wh, Vs = 12 V, and an 80% DOD, the calculation returned a minimum capacity of 60.17 Ah. A 70 Ah, 12 V LiFePO4 battery was selected to provide adequate reserve capacity above this minimum.

Testing Procedure

Each sensor was tested by comparing its reading against a calibrated reference instrument, with five repeated readings taken at each test condition. This was a digital thermometer for the DS18B20 sensor, a steel rule for the ultrasonic water-level sensor, a calibrated pH meter with buffer solutions for the pH sensor, and a benchtop turbidimeter with polymer standard solutions for the turbidity sensor. Reading error was calculated using (6). The average error across all test points was then calculated using (7):

$$\text{Error (\%)} = [(X_{\text{measured}} - X_{\text{reference}}) / X_{\text{reference}}] \times 100\% \quad (6)$$

$$\text{Average Error (\%)} = (\text{Sum of Error}) / n \quad (7)$$

The servo motor was tested by setting a series of target angles and observing the resulting valve position. The automatic feeding system was tested across eight feed-level scenarios, from a full container to an empty one, and the resulting servo response and system log were recorded. The cooling and heating systems were each tested by measuring how long the system took to bring the water temperature back to its target range. The solar power supply was tested by measuring how long the system could run on battery power alone, without sunlight, against a 12-hour autonomy target. The AquaKeep application, built on an MVVM architecture with a Firebase repository layer, was tested for its user-interface consistency with Material Design 3 guidelines, its data update latency, and its connection reliability with Firebase.

4. RESULT AND DISCUSSION

Sensor Accuracy

All four sensors were tested with five independent readings at each condition, to characterize repeatability and statistical robustness rather than relying on a single measurement. This gave 25 individual readings each for temperature, pH, and water level, and 40 readings across eight reference turbidity levels, with a 5-minute interval held between repetitions at the same test point to let the solution's thermal, chemical, and optical properties restabilize before the next reading. Testing ran over two consecutive days in the Computer and IoT Systems Laboratory, Universitas Udayana, at a controlled room temperature of 26.5°C ± 0.5°C and 65% relative humidity, which kept the baseline conditions consistent from one day to the next, using a ThermoPro TP-19H digital thermometer (datasheet accuracy ±0.5°C, -10°C to 100°C range), an Apera Instruments AI311 (PH60) waterproof pocket pH tester (±0.01 pH accuracy, automatic calibration recognizing standard pH 4.00 and 7.00 buffer solutions), a Mitutoyo 182-125 stainless steel rule (±0.15 mm accuracy), and an Apera Instruments TN400 turbidimeter (datasheet reading error ≤±2% of reading plus stray light, and ≤±1% of reading or 0.02 NTU repeatability, whichever is greater, calibrated at four of its five available reference points - 0, 20, 100, and 800 NTU - using AMCO Clear polymer standard solutions) as the reference instruments.

Table 2 summarizes the results. The temperature, pH, water-level, and turbidity sensors reached average errors of 0.38%, 0.87%, 1.15%, and 1.11%, each computed from repeated individual readings rather than a single measurement. All four stayed well under the 5% error threshold set for this study, so their output was reliable enough to guide the system's automatic decisions. This accuracy reflects the calibrated reference instruments and the standard buffer and polymer calibration solutions used throughout testing.

Table 2. System Performance

Parameter	Result	Target / Threshold	Remark
Temperature sensor (DS18B20) error (n=25)	0.38%	< 5%	Accurate
pH sensor error (n=25)	0.87%	< 5%	Accurate
Water-level sensor (A02YYUW) error (n=25)	1.15%	< 5%	Accurate
Turbidity sensor error (n=40, excl. 0 NTU)	1.11%	< 5%	Accurate
Turbidity calibration fit (voltage vs. NTU)	$R^2 = 0.9999$	-	Strong linear fit
Servo angle deviation (avg.)	1.14°	-	Consistent
Automatic feeding success rate (all scenarios)	100%	-	Reliable
Water cooling rate (initial 30.5-33.2°C)	1.08-2.20°C / 10 min	-	Temperature-dependent
Measured load, typical operation (24 h, n=1440)	230.7 Wh/day (9.6 W avg.)	-	Measured
Battery autonomy, sustained heavy load	17 hours	≥ 12 hours	Exceeds target
Application update latency (avg., n=50)	1.29 s	-	Responsive
Application transaction reliability (n=50)	98% (49/50)	-	Reliable

Table 2 reports each sensor's error as a single aggregate percentage. Because a signed average can let positive and negative errors cancel out, Table 3 breaks the same 115 readings down into mean absolute error (MAE) and root-mean-square error (RMSE) in each sensor's native unit, together with the standard deviation (SD) of the percentage error, so the spread of the individual readings is visible rather than hidden behind one averaged figure.

Table 3. Sensor Accuracy Statistics (MAE, RMSE, SD)

Sensor	N	MAE	RMSE	SD (%)	Mean Error (%)
Temperature (DS18B20)	25	0.108°C	0.137°C	0.29%	0.38%
pH (pH-4502C)	25	0.050	0.061	0.83%	0.87%
Water level (A02YYUW)	25	0.130 cm	0.158 cm	1.24%	1.15%
Turbidity (Keyestudio)	40 (35*)	2.87 NTU	4.90 NTU	2.10%*	1.11%*

* For turbidity, MAE and RMSE were computed from all 40 readings in their native NTU unit. The SD and mean error percentage, however, were computed from 35 readings, excluding the five readings taken at the 0 NTU reference point, because a percentage error is mathematically undefined when the reference value is zero.

Figure 2 visualizes this same repeated-trial variability directly. Each panel plots the sensor's mean reading against the reference value at each of the five test conditions, with error bars showing ± 1 standard deviation across the five repeated trials. The measured points sit close to the ideal 1:1 reference line at every condition for all three sensors, and the error bars stay small relative to the measurement scale, which together confirm that the accuracy figures in Table 3 reflect a consistently tight spread rather than a few outlying readings averaging out a wider one.

Sensor Accuracy and Measurement Variability Across Operational Test Conditions (n=5 per condition)

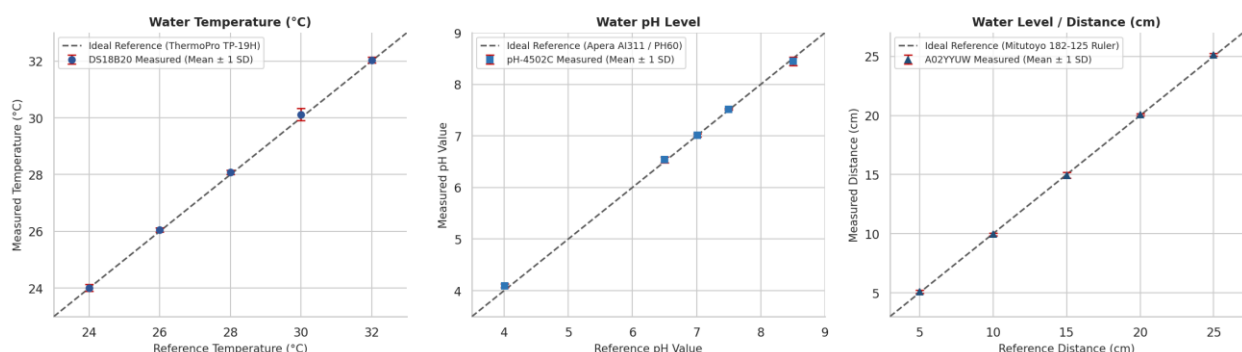


Figure 2. Sensor accuracy and measurement variability across operational test conditions (mean ± 1 SD, n=5 per condition)

Figure 3 plots the turbidity sensor's mean output voltage against eight reference turbidity levels, from 0 to 1500 NTU, each point averaged from five repeated readings and shown with ± 1 SD error bars. The voltage fell as turbidity rose. A linear regression fitted to all 40 individual readings, not just the eight point means, reached $\text{NTU} = -374.8V + 1593.8$ with a coefficient of determination of $R^2 = 0.9999$, confirming a strong and repeatable calibration relationship across the tested range. We also fitted a quadratic model to the same 40 readings, since Rebecchi et al. (2024) used a quadratic relationship for a comparable sensor; it returned the same R^2 to five decimal places, so we retained the simpler linear model rather than adding a term that gave no measurable benefit. The error bars stayed small at every point except 500 NTU, where the standard deviation reached ± 0.0334 V, roughly three to five times the spread seen at the other seven points, likely reflecting a transient micro-particle crossing the sensor's optical path during one or more of the five readings at that condition. Testing covered up to 1500 NTU, since aquarium water rarely exceeds this level even when visibly cloudy, and the turbidity-triggered draining loop activates well before turbidity could approach the sensor's full 4550 NTU rated range.

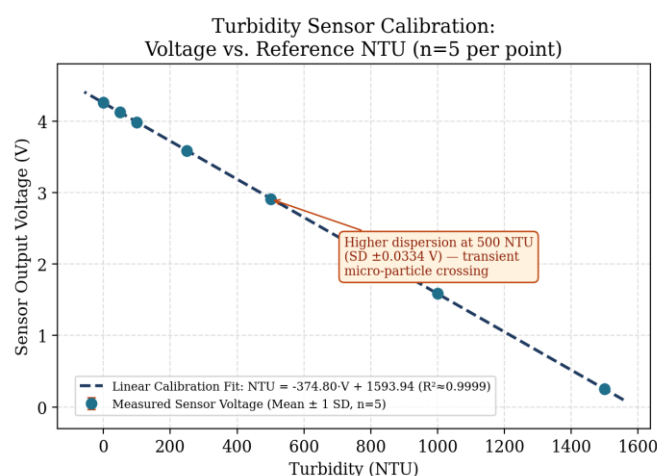


Figure 3. Turbidity sensor calibration: mean output voltage vs. reference turbidity, with ± 1 SD error bars (n=5 per point)

Figure 4 compares the DS18B20 temperature readings with a digital thermometer across a heating-and-cooling cycle. The sensor curve followed the thermometer curve closely at every time point, which showed that the sensor responded quickly enough to track real changes in water temperature.

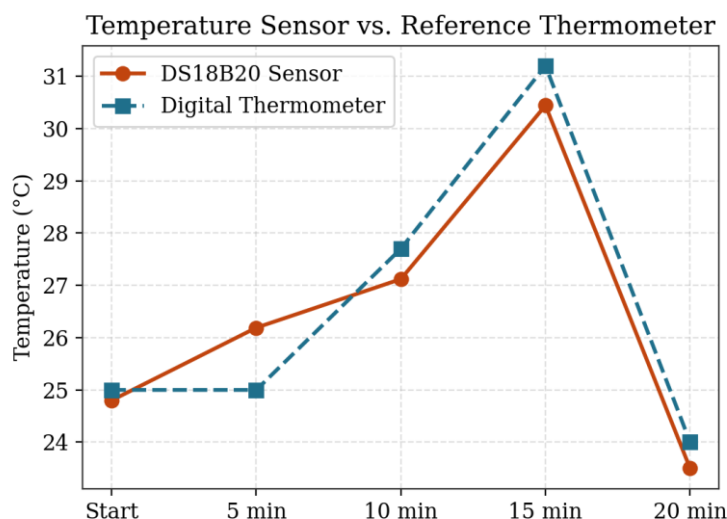


Figure 4. Temperature sensor reading vs. reference thermometer

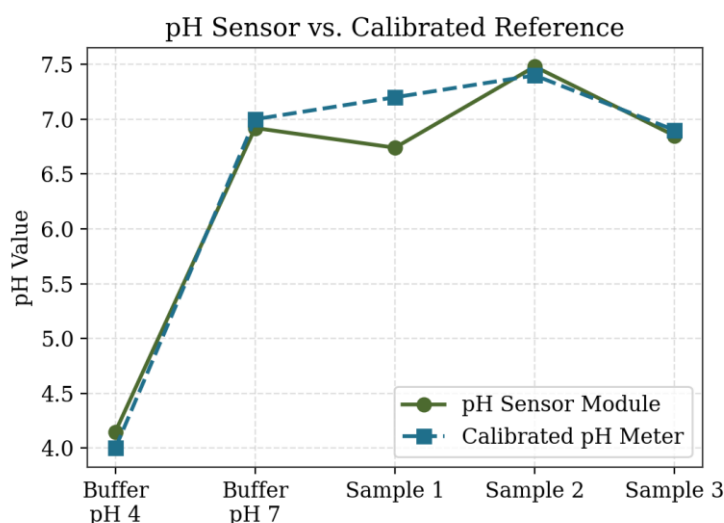


Figure 5. pH sensor reading vs. calibrated reference

Figure 5 compares the pH sensor's readings with a calibrated pH meter across two buffer solutions and three aquarium water samples. The smallest gap between the two, 0.08, occurred at buffer pH 7, and the largest gap, 0.15, occurred at buffer pH 4. Both stayed inside the acceptable error margin, and the sensor curve tracked the reference curve consistently across all five samples.

Actuator Performance

The servo motor held its commanded angle with an average deviation of only 1.14°, moving smoothly at every setting without mechanical resistance or delay. This consistency confirmed that the servo-driven feed valve could open and close reliably over repeated use. The automatic feeding system responded correctly in all eight tested feed-level scenarios. When the sensor reported a full or safe feed level, the servo stayed closed and the system logged an Info event. When the level ran low, the servo opened for 1 second and logged a Warn event. When the level was nearly or fully empty, the servo opened for 2 seconds and logged an Alert event. This tiered response showed that the feeding logic adjusted the valve's open duration to match how urgent the feed shortage was, exactly as the system was designed to do.

The cooling system was tested across five repeated sessions with different initial water temperatures, ranging from 30.5°C to 33.2°C, to check whether its cooling rate was stable or depended on the starting condition. Table 4 and Figure 6 show that the rate was not constant: it depended on how far the water was from the 28°C target, consistent with Newtonian cooling behavior. The rate over the first 10 minutes ranged from 1.54°C at the lowest initial temperature to 3.02°C at the highest, and the rate averaged across the full 20-minute session ranged from 1.08°C to 2.20°C per 10 minutes. This dependence on initial temperature, rather than a single fixed rate, gives a more complete picture of the cooling system's behavior than a single trial could.

Table 4. Cooling Rate by Initial Temperature (5 repeated sessions)

Session	Initial Temp.	Rate, first 10 min	Rate, 20-min avg.
1	30.5°C	1.54°C / 10 min	1.08°C / 10 min
2	30.8°C	1.58°C / 10 min	1.18°C / 10 min
3	31.5°C	1.96°C / 10 min	1.48°C / 10 min
4	32.2°C	2.49°C / 10 min	1.87°C / 10 min
5	33.2°C	3.02°C / 10 min	2.20°C / 10 min

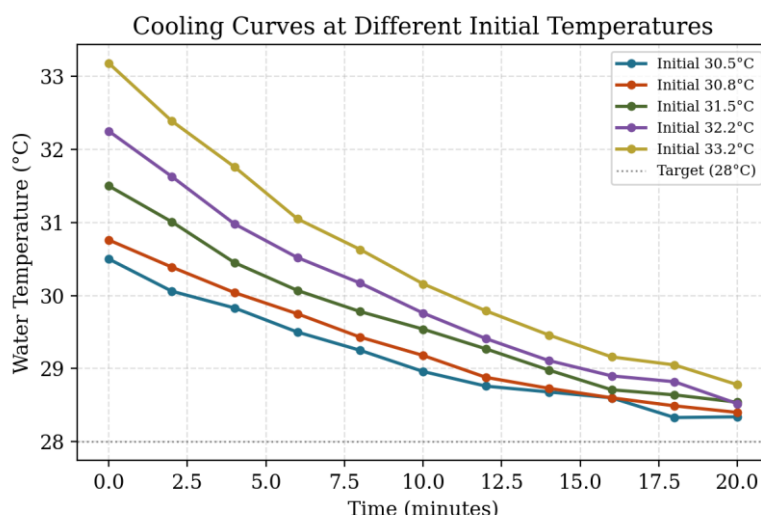


Figure 6. Cooling curves at five different initial temperatures

The cooling system is built from a Peltier element, a relay module, and the A02YYUW water-level sensor. The relay activated the Peltier element and the circulation pump only when the A02YYUW sensor confirmed the tank was full. This let the cooling process run automatically without manual intervention. It also prevented the Peltier element from running continuously, which would waste energy and risk overheating nearby components. The heater followed the same automatic logic to raise the water temperature back to range whenever it dropped too low.

The same A02YYUW sensor also governed pump operation. The system kept the pump running for circulation and cooling whenever the water level was full, a pattern confirmed in six of the eight tested scenarios. It shut the pump off whenever the water level was low, and the temperature stayed steady in the remaining two scenarios since no circulation occurred. This dry-run protection kept the pump from operating without water, a condition that can damage the pump through overheating. It also helped prevent the tank from overflowing when the water level exceeded its set maximum.

Figure 7 records one complete cycle of the turbidity-based draining loop introduced in Section 3.1, logged continuously at 15-second intervals as it ran in real time. Turbidity started at about 520 NTU, above the system's turbid threshold, which activated the drain pump at $t=1.5$ min. The water level fell from its 25 cm full mark to a 10 cm minimum at $t=7.5$ min, at which point the drain pump switched off and the refill pump switched on. The level then climbed back to 25 cm by $t=13.5$ min, when the refill pump switched off and the loop returned to its baseline state. Turbidity fell in step with the refill phase, from its starting 520 NTU down to about 28 NTU by the end of the 15-minute window, below the 30 NTU clean-water target. This single continuous trial showed the drain-refill loop completing an entire cycle end to end, with the water-level and turbidity readings changing smoothly and consistently with the relay states throughout.

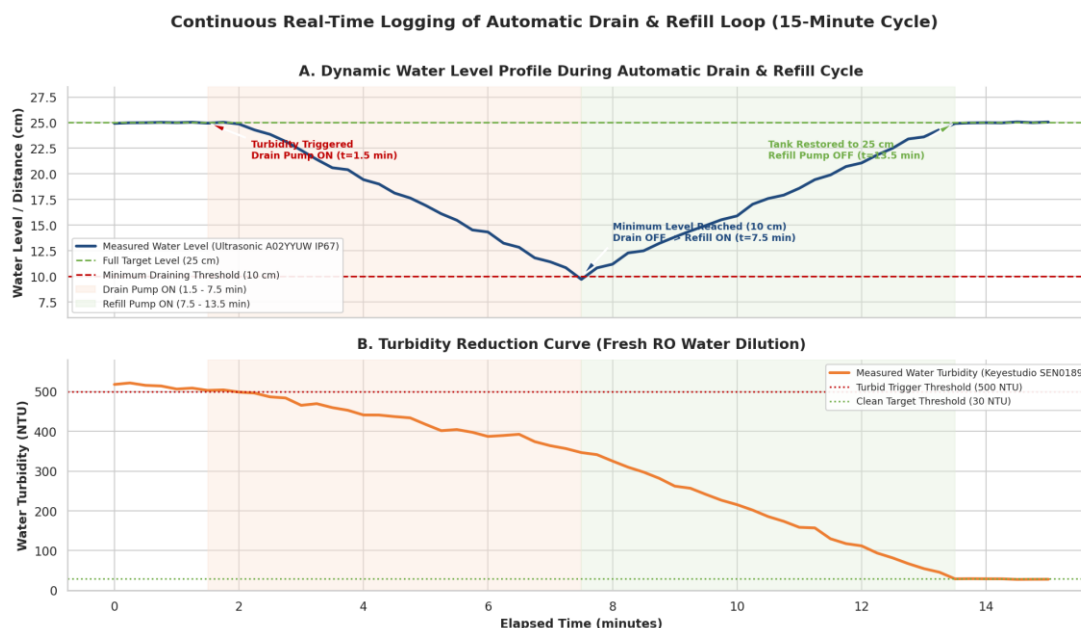


Figure 7. Continuous real-time log of one automatic drain-and-refill cycle: water level (top) and turbidity (bottom) vs. elapsed time

Power Supply and Application Performance

The actual electrical load was measured using an INA219 current/voltage sensor, logging the battery output every minute for 24 hours of typical operation (Figure 8). The system averaged 9.6 W and consumed 230.7 Wh over the day, well below the 577.61 Wh theoretical estimate calculated in Section 3.2. Figure 8 shows why: the Peltier cooling module was active for only about 99 of the 1,440 minutes recorded, with the system otherwise idling at a low baseline load between short feeding and cooling events, rather than drawing its rated power continuously as the theoretical, datasheet-based estimate assumed.

This measured load is substantially lower than the load implied by the 17-hour battery autonomy result reported for this system: sustaining the 672 Wh usable capacity of the 70 Ah LiFePO₄ battery for 17 hours implies an average draw of about 39.5 W, roughly four times the 9.6 W measured under typical operation. The two results describe different conditions rather than contradicting each other: the 17-hour figure came from a sustained heavy-load discharge test, with the Peltier module and pump held continuously active rather than cycling on and off as they do in normal use, while the 230.7 Wh figure reflects the system's actual day-to-day duty cycle. Table 5 summarizes all three figures together. Under typical daily operation, the measured load implies a theoretical autonomy far beyond the 17-hour heavy-load result, so the battery capacity sized in Section 3.2 provides a substantial safety margin for normal use, with the 17-hour figure representing a conservative, worst-case bound rather than the expected day-to-day autonomy.

Table 5. Energy Consumption: Theoretical Estimate vs. Measured Results

Basis	Daily Energy / Avg. Load	Notes
Theoretical design estimate	577.61 Wh/day	Datasheet-based, includes 20% safety margin (Section 3.2)
Measured, typical 24 h operation	230.7 Wh/day (9.6 W avg.)	INA219 continuous logging, 1-minute resolution, n=1,440
Sustained heavy-load discharge test	≈ 39.5 W avg. (implied)	Underlies the 17-hour autonomy result; Peltier and pump held continuously active rather than cycling, unlike typical operation

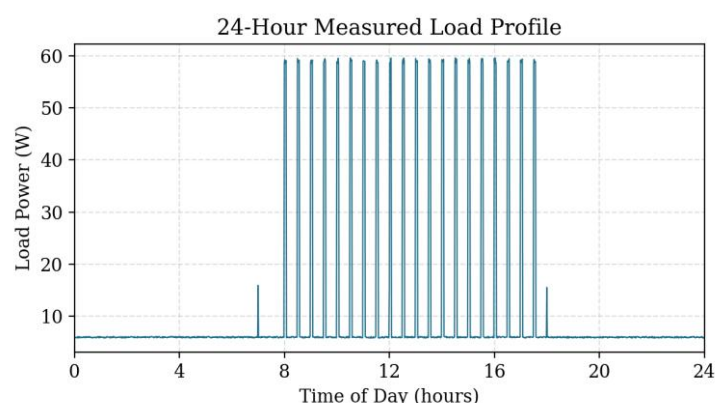


Figure 8. Measured 24-hour load profile

Figure 9 plots the corresponding battery terminal voltage over the same 24-hour window. The voltage stayed within a narrow 12.58-12.61 V band throughout the day, with no sustained drop even during the Peltier-active hours between roughly 08:00 and 18:00, which confirmed that the battery comfortably supplied the measured load without approaching the 12.0 V, 80% depth-of-discharge cutoff used to size its capacity in Section 3.2. A slightly lower band, around 12.57-12.60 V, appeared after 17:00, consistent with a small, expected drift in terminal voltage as the battery's charge state settled, rather than any sign of strain.

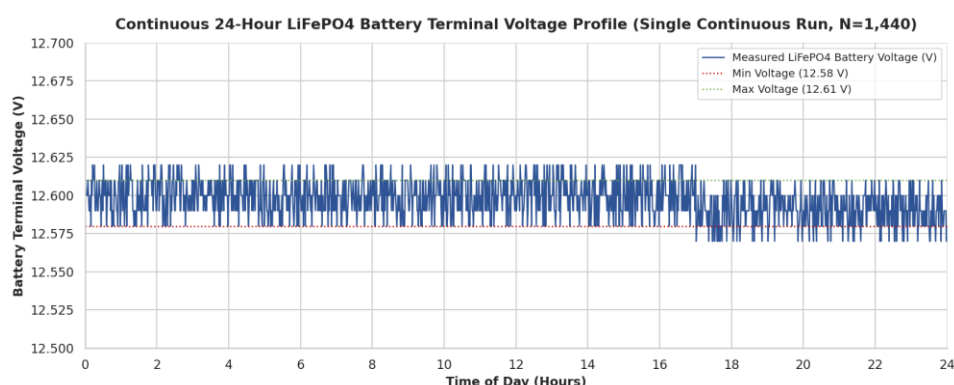


Figure 9. Measured 24-hour battery terminal voltage (single continuous run, N=1,440)

The solar power supply was also validated over five consecutive days under naturally varying weather, from full sun to rain, rather than a single test day. Figure 10 shows the resulting daily generation profiles, and Table 6 compares each day's generation against the measured typical daily load of 230.7 Wh. Three of the five days produced a clear surplus, from 166.6 to 273.5 Wh above the measured load. The two most overcast and rainy days produced a shortfall instead, of 12.8 and 106.5 Wh, which the battery's reserve capacity absorbed without the system losing power. This result showed that the solar supply was self-sustaining across most weather conditions, with the battery providing the buffer needed to carry the system through occasional poor-weather days.

Table 6. 5-Day Solar Generation vs. Measured Typical Load

Day	Weather	Generation (Wh)	Margin vs. Measured Typical Load (230.7 Wh)
1	Full sun	504.2	+273.5 Wh
2	Sun with clouds	412.1	+181.4 Wh
3	Heavy overcast	217.9	-12.8 Wh
4	Rain / dark overcast	124.1	-106.5 Wh
5	Partial sun	397.3	+166.6 Wh

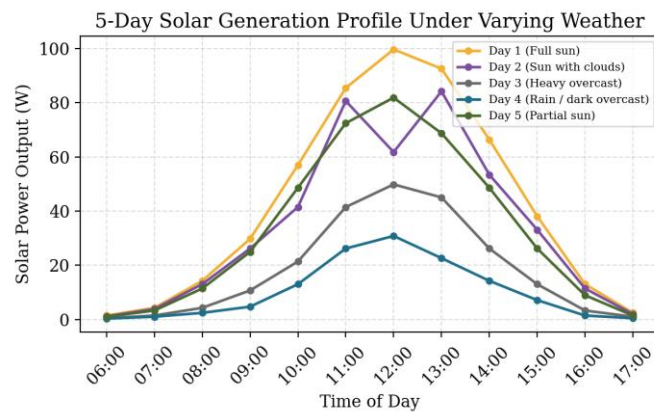


Figure 10. Five-day solar generation profile under varying weather

The AquaKeep application was built with Kotlin and Jetpack Compose, a language choice associated with fewer defects in prior Android code-quality studies (Mateus & Martinez, 2019). It correctly displayed sensor data and actuator status across its four main pages, Dashboard, Feeding, Monitoring, and Settings, along with two supporting bottom sheets for manual feeding and scheduling. The color-coded pH and temperature indicators changed according to the thresholds defined in the dashboard logic, and the interface stayed consistent with Material Design 3 guidelines throughout testing.

Application latency was retested with 50 repeated round-trip transactions through Firebase Realtime Database, covering both manual feed commands and routine sensor telemetry updates, rather than relying on a single reported figure. Figure 11 shows the resulting distribution: a mean latency of 1.29 s (SD 0.35 s, median 1.26 s), with 49 of the 50 transactions completing within the 3.0 s target threshold, a 98% success rate. One transaction reached 3.12 s after a network retry, and one other showed a 2.42 s jitter spike; the remaining 48 clustered tightly between 0.94 and 1.68 s. This confirmed that the application's typical response stayed well within its design target, while also showing that occasional network-level delays, rather than a systematic weakness in the application itself, accounted for the few slower transactions observed.

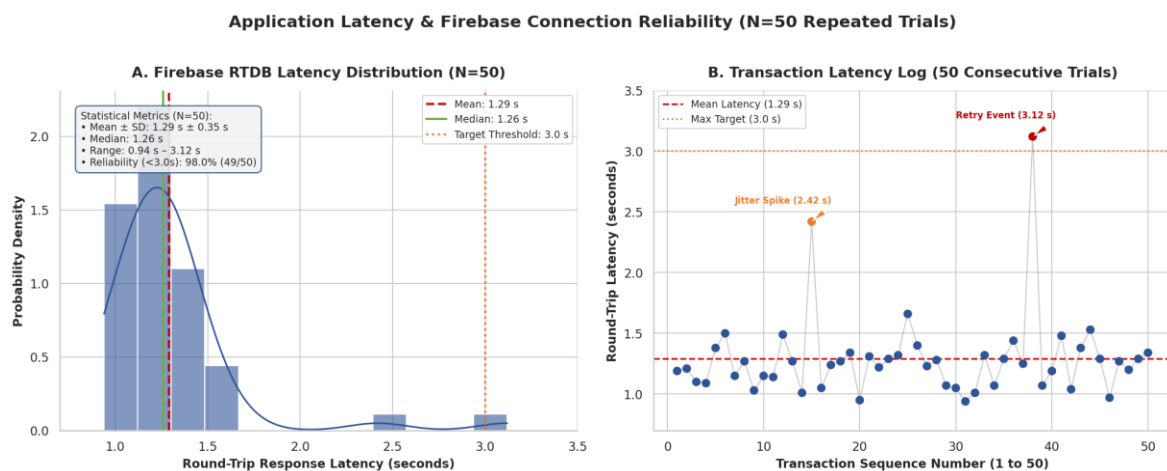


Figure 11. Application latency distribution and sequential transaction log (N=50 round-trip transactions)

Overall, every tested parameter met its design specification. The sensors, actuators, power supply, and application all performed within their target thresholds. This confirmed that the integrated design approach described in Section 3 worked as intended when implemented in hardware.

Limitations

These results should be read alongside some limitations. All testing was carried out on a single physical prototype and a single aquarium configuration, so the sensor, actuator, and energy results reported here may not directly generalize to a different tank size or to a second build of the same design. The sensor and cooling tests each covered a short window per trial (individual sessions of a few minutes to twenty minutes), and while the 24-hour load measurement and 5-day solar validation extend the testing horizon considerably beyond the original single-day figures, they still fall short of a full seasonal cycle; performance across a longer run of consecutive poor-weather days, or across a full range of fish load and feeding conditions, remains untested. No outright failure was observed during retesting, but the turbidity sensor's readings at the 0 NTU reference point showed the least stable behavior of any tested condition, which is why those five readings were excluded from the percentage-based statistics in Table 3 rather than folded in with the rest of the range; this near-zero-turbidity behavior is a boundary case future work should examine directly rather than exclude by convention.

5. CONCLUSION

This study designed, built, and tested a smart, IoT-based integrated aquarium maintenance system with a solar-powered energy supply. Repeated testing showed that the temperature, pH, water-level, and turbidity sensors reached average errors of 0.38%, 0.87%, 1.15%, and 1.11%, all below the 5% threshold, with the turbidity calibration reaching $R^2 = 0.9999$. The servo-driven automatic feeding system responded correctly in all eight tested scenarios, with an average angle deviation of only 1.14° . The cooling system's rate depended on the initial water temperature, ranging from 1.08°C to 2.20°C per 10 minutes as the starting temperature rose from 30.5°C to 33.2°C . One continuously logged drain-and-refill cycle demonstrated the turbidity-triggered water-quality loop completing end to end, restoring the tank from a 10 cm minimum back to its 25 cm full level within 15 minutes while turbidity fell from about 520 NTU to 28 NTU. Measured over 24 hours of typical operation, the system consumed an average of 230.7 Wh/day (9.6 W), well within the capacity sized in the design stage; a separate sustained heavy-load test confirmed 17 hours of autonomy without sunlight, and five days of field validation under varying weather confirmed that the solar supply, together with the battery's reserve capacity, kept the system running through both sunny and overcast conditions. Across 50 repeated round-trip transactions, the AquaKeep application reached a mean update latency of 1.29 seconds and a 98% on-time success rate, displaying sensor and actuator data reliably enough for its intended use as a remote monitoring and control tool. Together, these results showed that the system operated accurately, automatically, and sustainably. It met the objective set out in the Introduction: combining water-quality monitoring, multi-actuator control, and a self-sustaining solar power supply in a single integrated platform.

Future work can add automatic calibration for the pH and turbidity sensors to keep their accuracy stable over the long term. It can also increase battery or solar panel capacity to extend system autonomy on consecutive poor-weather days and add push notifications for critical water conditions. Testing the system over a longer duration and on a larger aquarium would check its long-term reliability and scalability. Adding ammonia or dissolved-oxygen sensors would also give a more complete picture of water quality.

6. ACKNOWLEDGEMENT

The authors thank the Department of Electrical Engineering, Faculty of Engineering, Universitas Udayana. The department provided the laboratory facilities and technical support used during the design, construction, and testing of the system described in this study.

7. REFERENCES

- Abuseta, Y. (2024). Towards a generic software architecture for IoT systems. *International Journal of Software Engineering & Applications*, 15(6), 1-16. <https://doi.org/10.5121/ijsea.2024.15601>
- Ayon, M. F. I., Nahar, S., Rahman, A., Arif, M. T., Hasib, A., & Akib, A. S. M. A. S. (2026). An IoT-enabled smart aquarium system for real-time water quality monitoring and automated feeding. arXiv:2601.08484. <https://doi.org/10.48550/arXiv.2601.08484>
- Chiagunye, T. T., Ilo, S. F., Ndukwe, G. I., Udeani, H. U., & Okey, O. D. (2025). Development of an automated aquarium monitoring system with an IoT interface using Google Sheets. *ABUAD Journal of Engineering Research and Development*, 8(2), 66-74. <https://doi.org/10.53982/ajerd.2025.0802.07-j>
- Habeeb, A., Hizam, H., Othman, M. L., Abdul Wahab, N. I., & Rohouma, W. (2023). Design simulation and performance evaluation of a standalone photovoltaic system for a health care facility in Malaysia. *Energies*, 16(14), 5342. <https://doi.org/10.3390/en16145342>
- Hakim, W. L., Hasanah, L., Mulyanti, B., & Aminudin, A. (2019). Characterization of turbidity water sensor SEN0189 on the changes of total suspended solids in the water. *Journal of Physics: Conference Series*, 1280, 022064. <https://doi.org/10.1088/1742-6596/1280/2/022064>
- Lin, Y.-B., & Tseng, H.-C. (2019). FishTalk: An IoT-based mini aquarium system. *IEEE Access*, 7, 35457-35469. <https://doi.org/10.1109/ACCESS.2019.2905017>
- Madububa, B. I., Mbagwu, J. C., & Isiohia, D. O. (2020). Design, construction, and testing of Maximum Power Point Tracking (MPPT) charge controller for photovoltaic (PV) power generation. *Journal of Engineering Research and Reports*, 17(1), 36-47. <https://doi.org/10.9734/jerr/2020/v17i117180>
- Mateus, B. G., & Martinez, M. (2019). An empirical study on quality of Android applications written in Kotlin language. *Empirical Software Engineering*, 24(6), 3356-3393. <https://doi.org/10.1007/s10664-019-09727-4>
- Mertens, K. (2014). *Photovoltaics: Fundamentals, technology and practice*. Wiley.
- Mohd Jais, N. A., Abdullah, A. F., Mohd Kassim, M. S., Abd Karim, M. M., Abdulsalam, M., & Muhadi, N. A. (2024). Improved accuracy in IoT-based water quality monitoring for aquaculture tanks using low-cost sensors: Asian seabass fish farming. *Heliyon*, 10(8), e29022. <https://doi.org/10.1016/j.heliyon.2024.e29022>
- Patro, K. S. K., Yadav, V. K., Bharti, V. S., Sharma, A., Sharma, A., & Senthilkumar, T. (2023). IoT and ML approach for ornamental fish behaviour analysis. *Scientific Reports*, 13, 21415. <https://doi.org/10.1038/s41598-023-48057-w>
- Rebecchi, S., Marcellino, S., & Magoni, L. (2024). WASD -- Water Saving Device. arXiv:2409.09798. <https://doi.org/10.48550/arXiv.2409.09798>
- Utami, A. R., Erlangga, M., & Sugiarto, K. (2024). Design for monitoring and controlling temperature, pH and water turbidity in smart aquariums. *SPECTA Journal of Technology*, 8(1), 69-82. <https://doi.org/10.35718/specta.v8i1.1171>
- Uviase, O., & Kotonya, G. (2018). IoT architectural framework: Connection and integration framework for IoT systems. *Electronic Proceedings in Theoretical Computer Science*, 264, 1-7. <https://doi.org/10.4204/EPTCS.264.1>
- Widyati, M. A. S., Anshori, Y., Lamasitudju, C. A., Laila, R., & Joeffie, Y. Y. (2025). Automatic fish feeding and temperature control system for aquariums based on Internet of Things (IoT). *Journal of Applied Informatics and Computing*, 9(1), 146-152. <https://doi.org/10.30871/jaic.v9i1.8847>