AnomAIy: Edge-Level Anomaly Detection for Liquid-Cooled Computing Systems

Kristina Huang, Jacob Baer, and Aidan Blazevich

Department of Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—Downtime in liquid-cooled servers and PCs is costly, making early detection of system degradation critical to maintaining reliability and extending lifespan. Gradual faults such as blockages in the cooling loop reduce the efficiency of power dissipation, and degradation in the Voltage Regulator Module (VRM) can lead to cascading failure and system shutdown. This paper presents an anomaly detection framework for identifying early signs of degradation in liquid-cooled systems. To implement this, a hybrid regression and autoencoder model predicts expected power dissipation from sensor data and detects real-time anomalies to support preventative maintenance and improve long-term reliability.**

***Index Terms*—Anomaly Detection, Autoencoder, Edge Computing, Liquid Cooling, Regression Model, Server Cooling Systems, System Health Monitoring**

I. Introduction

Liquid cooling has become an increasingly common solution for managing power dissipation in computing systems due to its higher heat transfer efficiency compared to air cooling [22]. As server workloads and AI applications continue to demand more consistent performance, thermal management has become a key reliability concern. To address this challenge, this project targets server operators, data center managers, and PC system builders who rely on liquid cooling to maintain stable operation under continuous and varying power workloads.

Existing liquid-cooling systems rely on temperature thresholds and PID control loops to maintain target temperatures [11], [15]. These threshold-based alarms cannot detect inefficiencies and deterioration that develop over time, such as partial blockages in the cooling loop from coolant deposits or reduced efficiency in the Voltage Regulator Module (VRM) due to phase loss [38]. More heat builds up in the system when a VRM inefficiency increases power loss or a blockage reduces cooling effectiveness. Once temperatures approach the threshold, the system compensates by increasing fan and pump speeds, masking the underlying problem. In doing so, the system remains apparently stable even as energy consumption rises and mechanical components operate under greater stress.

First, persistent high operating temperatures reduce overall system reliability and lifetime. Higher thermal load on processors, power delivery circuits, and other electronics accelerates aging and increases the likelihood of failure [32], [35]. Even moderate increases in temperature over time shorten the useful life of the system and raise the risk of downtime. In always-on systems such as servers, even brief downtime can disrupt operations and lead to costly delays [36].

Second, increased heat generation makes it harder for the cooling system to maintain its temperature setpoint. As temperatures rise, the radiator and surrounding air approach thermal equilibrium, reducing the temperature difference that drives heat transfer. This lowers the overall heat transfer rate according to (1):

$$\dot{Q} = \dot{m} c_p \Delta T \tag{1}$$

where $\dot{Q}$ is the heat transfer rate (W), $\dot{m}$ is the mass flow rate of coolant (kg/s), c_p is the specific heat capacity of coolant (J/kg·°C), and ΔT is temperature rise across the component or loop (°C) [6]. Reduced cooling efficiency raises CPU temperatures, so the buffer between current and maximum safe operating temperature shrinks. With less margin available, temporary workload spikes or changes in ambient temperature can trigger thermal throttling, where the processor automatically reduces frequency and power to prevent overheating [18]. This leads to more frequent performance fluctuations and unstable behavior under load.

Finally, these hidden inefficiencies can lead to cascading failures in power delivery systems. For example, if one phase of a multiphase VRM begins to degrade, the remaining phases must supply more current to maintain output [23]. This added current increases power loss and thermal stress on the remaining phases, accelerating further degradation until eventual phase loss or complete VRM failure. In many cases, compensating fan and pump speeds delay visible symptoms, allowing the system to appear stable until the fault becomes severe. Prolonged high fan and pump speeds also shorten component lifespan and increase maintenance needs. Elevated bearing temperature and sustained high rotational speed cause exponential reductions in fan life, leading to earlier mechanical failure and decreased reliability of the cooling system [31].

Detecting inefficiencies early enables scheduled maintenance, extending service intervals, improving system reliability, and reducing the risk of costly unplanned downtime. This project aims to implement a preventative maintenance alert system that identifies such faults early and accurately.

II. Use-Case Requirements

For the success of the proposed maintenance alert system, we outline the following quantitative use-case requirements.

1. Abnormal Power Detection

An anomaly alert must be issued when the measured power differs from the predicted power by more than 10 % for at least 30 seconds. This requirement allows the system to detect reduced power delivery efficiency, such as a partially degraded VRM phase, without triggering false alerts caused by short spikes in CPU activity or small variations from sensor noise.

In a 65 W class CPU operating around a typical 60 W load, a 10 % mismatch corresponds to approximately 6 W of additional power loss. With a typical CPU water block thermal resistance of 0.1-0.2 °C/W [1], this results in a 0.6-1.2 °C rise in junction temperature. Even a small, sustained temperature rise of 1 °C has been shown to shorten component lifetime by about 1.3-1.5 years based on the Arrhenius acceleration model [37]. This supports the 10 % mismatch threshold as a meaningful indicator of system degradation.

The 30 second persistence requirement is based on the thermal response time of the cooling loop. Using a first-order RC model, the loop's temperature rise after a small abnormal power step was calculated. With a typical radiator thermal resistance R_{rad} of 0.172 °C /W and an estimated coolant volume of 310 mL, the calculated thermal capacitance C_{th} (2) was 1.296 kJ/°C, yielding a time constant τ (3) of 222.9 seconds. The following equations were used to calculate these quantities:

$$C_{\text{th}} = \rho V c_p \tag{2}$$

where ρ is the coolant density, V is the coolant volume, and c_p is the specific heat.

$$\tau = R_{\text{rad}} C_{\text{th}} \tag{3}$$

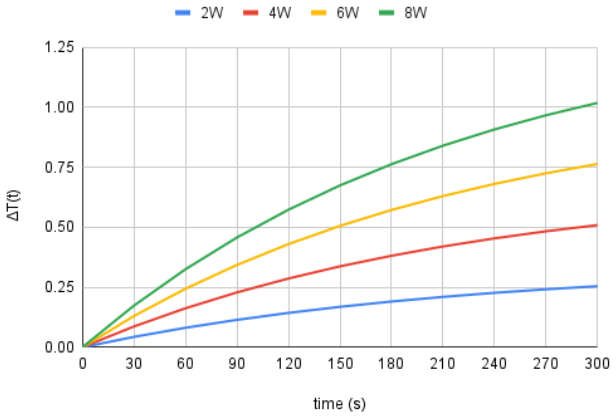

Fig. 1. Calculated bulk coolant temperature rise for injected power anomalies of 2 W, 4 W, 6 W, and 8 W using a first-order RC thermal model (4).

The first-order step response of coolant temperature rise was modeled according to the following equation:

$$\Delta T(t) = \Delta T_{\infty}\left(1 - e^{-t/\tau}\right) \tag{4}$$

where ΔT_{∞} is the final steady state temperature rise. Fig. *1* shows that for an injected anomaly of 8 W, the bulk coolant temperature increases by only 0.173 °C after 30 seconds which is sufficient to exceed normal temperature jitter in the loop. Requiring the mismatch to persist for at least 30 seconds ensures that alerts are generated only after the temperature change becomes distinguishable from short-term fluctuations caused by flow variability or minor load changes.

2. Abnormal Flow Detection

An anomaly alert must be issued when the coolant flow rate decreases by more than 20 % from its normal operating value for at least 30 seconds. This requirement ensures that the system can detect partial blockages in the cooling loop that reduce cooling performance.

A 20 % decrease in flow rate lowers the rate at which heat is carried from components to the radiator. From (1), the rate of heat removal depends on mass flow rate, specific heat capacity, and temperature difference between the coolant and the radiator. When flow decreases, less heat is transferred per second, causing higher steady state coolant and component temperatures. For water, which has a specific heat capacity of approximately 4.18 J/kg·°C, a 20 % reduction in flow rate at the same power load can raise coolant temperature by several degrees [26], accelerating component wear and lowering cooling efficiency as shown in Section II.1. The 30 second persistence requirement ensures that short fluctuations in pump speed or sensor readings do not trigger false alerts, consistent with the thermal response characteristics discussed in Section II.1.

3. Accurate Alerts

The anomaly detection system must maintain a False Positive Rate (FPR) below 5 % and a False Negative Rate (FNR) below 5 %. This ensures that alerts reflect true system degradation rather than normal variations in workload or measurement noise. False positives lead to unnecessary alerts while false negatives allow faults to go undetected, increasing the risk of unplanned downtime. In "RADS: Real-time Anomaly Detection System for Cloud Data Centres," the authors report a low FPR of 0-3 %, demonstrating that maintaining FPR below 5 % helps preserve system reliability [5]. Keeping rates within this range minimizes false alarms that can reduce trust in the system while ensuring that genuine degradation events are consistently detected.

4. Timely Alerts

An anomaly alert must be issued within 1 second after the 30 second persistence period has ended. This ensures that once an abnormal condition has lasted long enough for temperature changes to become measurable and distinguishable from normal short-term fluctuations, the system promptly notifies the operator. Although the system is intended for preventative maintenance, a 1 second limit demonstrates real-time responsiveness from the model once a fault is confirmed, allowing sensor data, model outputs, and alert logs to remain

synchronized. This enables operators to identify exactly when a fault occurred and correlate it with other system events.

III. Architecture and/or Principle of Operation

The system architecture consists of two main subsystems: a physical testbed that simulates a liquid-cooled computing system and a model trained on data collected from the testbed to identify normal versus abnormal cooling behavior. A detailed overview of the interconnections of our system is provided in Appendix, Fig. 10.

1. Testbed Subsystem

The testbed subsystem simulates a computer or server liquid-cooling system, circulating coolant through a controlled loop with adjustable liquid flow and airflow for heat dissipation. This enables evaluation of thermal behavior under both normal and injected fault conditions. The subsystem consists of five functional components: the water loop, heater complex, fault injection assembly, sensor network, and power distribution network.

The water loop circulates coolant through a reservoir, pump, CPU water block, and radiator with a fan. These components are the same types commonly used in custom PC liquid-cooling systems, ensuring realistic thermal and flow behavior [12]. The pump controls coolant flow to carry heat away from the CPU block to the radiator while the fan removes heat from the coolant to ambient air. Together, these components determine the system's overall cooling capacity and temperature response.

The heater complex simulates power dissipation from a CPU and VRM. Two power resistors represent these heat sources, each driven by a solid-state relay (SSR) that is PWM duty-cycle controlled by the Pi Pico to adjust the simulated power load. Varying heater output allows the system to simulate different computational loads for data collection.

The fault injection assembly introduces controlled faults to evaluate the model's ability to detect anomalies and meet the use-case requirements. Two types of faults are implemented:

1. Flow restriction is done by partially closing a servo-controlled valve to reduce coolant flow.
2. Hidden power injection is achieved by increasing VRM heater power to simulate reduced power conversion efficiency.

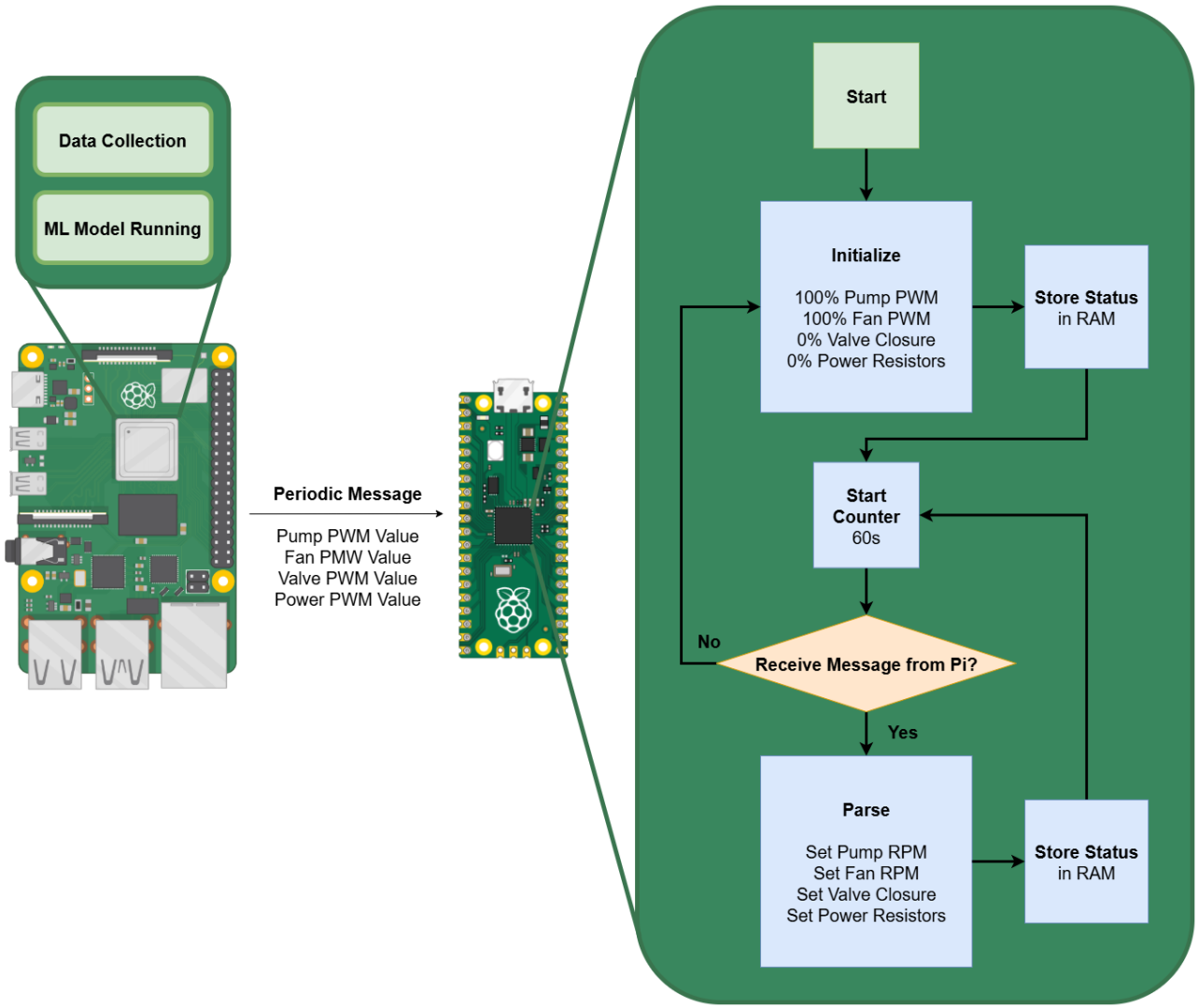

Fig. 2. System architecture showing communication between the compute unit (Pi 5) and control unit (Pico).

The sensor network consists of air temperature sensors at the radiator's inlet and outlet, a coolant temperature sensor at the reservoir, and tachometer feedback from the fan and pump. These measurements are used to infer power. All data are collected and timestamped by the compute unit, a Raspberry Pi 5 (Pi 5), and stored in InfluxDB for model input and anomaly detection.

The power distribution network provides regulated voltage rails for the components. A combination of power supplies, SSRs, and DC-DC buck converters delivers the required voltages for the heaters, pump and fan, and servo motors.

The system architecture, shown in Fig. 2, illustrates communication between the compute unit and control unit. The Pi 5 handles data collection, model execution, and the alert system while the Pico receives periodic heartbeat messages containing PWM commands for the pump, fan, servo valve, and heater power levels. On startup, the Pico initializes to a predefined safe operating state and continuously updates actuator outputs based on the latest heartbeat from the Pi 5. If no heartbeat is received, the Pico reverts to its safe state, maximum pump and fan speed, valve fully open, and heat power disabled to maintain cooling and prevent thermal runaway.

2. Model Subsystem

The model subsystem performs real-time anomaly detection using a regression model to preprocess data collected from the testbed and an autoencoder to detect anomalies.

During training, the regression model learns to predict expected power dissipation based on sensor inputs such as coolant temperature, radiator inlet and outlet air temperatures, and pump and fan speeds. The difference between predicted and measured power, or the residual, quantifies deviations from actual measured power.

An autoencoder is then trained on the residual, the percent deviation from actual power, and 30 second window-based features such as the mean, maximum, standard deviation, and slope. The autoencoder compresses these features through an encoder and reconstructs them through a decoder. Normal inputs produce low reconstruction error while anomalies produce a high reconstruction error. A threshold is determined during training to establish the reconstruction error that constitutes an anomaly flag [28].

The overall model workflow is shown in Fig. 3, illustrating the flow from regression-based power prediction and residual analysis to unsupervised anomaly detection and classification.

Once trained, the models are deployed on the Pi 5 for real-time inference. As the system operates, the Pi 5 continuously computes the residual between measured and predicted power and evaluates it using the trained autoencoder. If the reconstruction error exceeds a defined threshold, the model flags the anomaly and classifies it based on temperature response patterns: flow restrictions raise coolant temperature with little radiator outlet change, while power mismatches increase both coolant and radiator outlet temperatures.

Once the model confirms an anomaly, the result is immediately displayed on the Pi 5 terminal and recorded in a log file. This allows operators to see alerts in real time and review afterward to determine what conditions caused the fault.

IV. Design Requirements

The design requirements translate the use-case requirements from Section II into quantifiable metrics for both the hardware testbed and the anomaly detection model. Because the proposed system must first simulate realistic thermal behavior and collect representative data to train the anomaly detection model, the design requirements are divided into two categories. Testbed requirements (1 and 2) define the specifications of the components needed to simulate both normal and fault behavior of the PC or server, ensuring consistent, high-fidelity training and validation data. Model requirements (3 and 4) define the performance and alert generation criteria necessary for accurate, real-time anomaly detection once the model is deployed.

1. Abnormal Power Detection

To meet the abnormal power detection use-case, the testbed must simulate realistic CPU and VRM power dissipation levels and faulty VRM levels.

1.1 CPU and VRM Operating Range

The CPU heater must operate between 20-80 W, spanning idle to full load conditions for 65 W class CPUs. This range is based on reviews of commercial desktop processors such as the Intel Core i5-12400 and AMD Ryzen 5 7600 which report comparable power draw across typical workloads [19], [33].

The VRM heater must cover a 1-30 W range. Under normal (no phase-loss) conditions, VRMs achieve 85-95 % efficiency

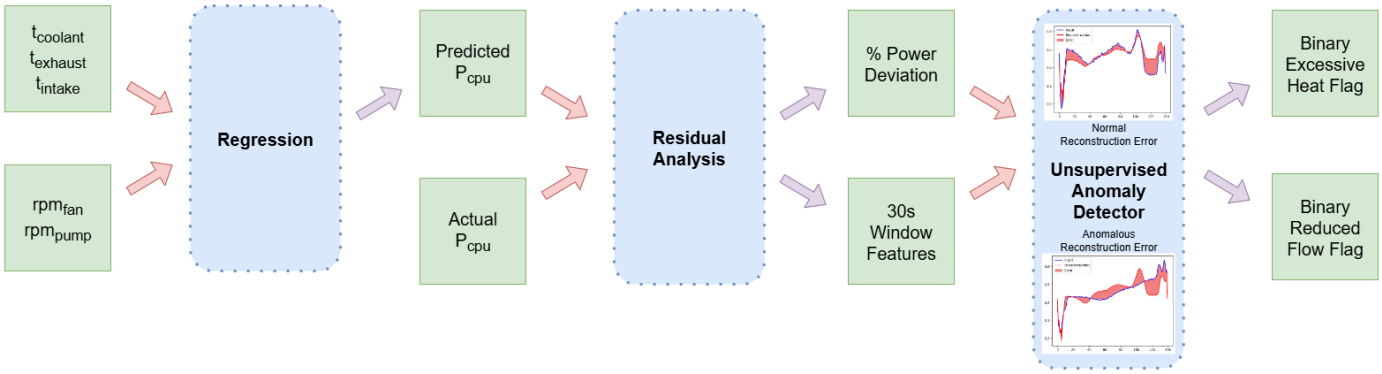

Fig. 3. Machine learning model workflow for anomaly detection.

for 20-80 W CPU loads [35]. An additional 10 % efficiency loss is modeled to simulate a degraded phase or reduced efficiency fault. The VRM loss fraction L is defined by the standard efficiency-loss equation for DC-DC converters [34]:

$$L = \left(\frac{1}{\eta} - 1\right) P_{\text{CPU}} \tag{5}$$

where L is the conversion loss (W), η is VRM efficiency, and P_{CPU} is CPU load power (W).

Under normal operating conditions, efficiencies from 85-95 % produce conversion losses ranging from approximately 1.05 W at $P_{\text{CPU}} = 20\,W$, $\eta = 0.95$ to 14.1 W at $P_{\text{CPU}} = 80\,W$, $\eta = 0.85$. With an additional 10 % efficiency loss to model the fault condition, the conversion loss rises to approximately 26.7 W at $P_{\text{CPU}} = 80\,W$, $\eta = 0.75$. Therefore, the VRM heater must cover a 1-30 W range, providing sufficient headroom for both normal and degraded VRM behavior.

To accurately simulate both normal and fault loads, the heater control system must maintain a command accuracy within 2 % RMS across the 20-80 W CPU load range. This ensures that the applied heater power matches the intended load and that control uncertainty does not interfere with the 10 % deviation threshold used for anomaly detection. A 2 % RMS error provides a five times separation between command uncertainty and the 10 % detection threshold, as shown by the signal-to-noise ratio (SNR) equation [9]:

$$\text{SNR} = \frac{0.10P}{0.02P} = 5 \tag{6}$$

Maintaining this ratio minimizes the likelihood of false positives from PWM inaccuracy. Across the 20-80 W range, a 2 % error corresponds to only 0.4-1.6 W, providing consistent repeatable loads for reliable model validation.

1.2 Pump and Fan Control Accuracy

To ensure realistic cooling behavior during data collection, the pump and fan must be PWM controllable with less than 2 % deviation between commanded and measured RPM. PWM control allows adjustment of flow and airflow rates to match real cooling loop conditions [13], [27]. Without it, overcooling could flatten temperature gradients, preventing sensors from capturing measurable thermal responses to injected faults.

Maintaining tight PWM control accuracy ensures that the system's thermal response remains constant between runs, so any measured power mismatch reflects true anomalies rather than variations in cooling performance. The rate of heat removal from the coolant is given by:

$$Q = h\,A\,(T_{\text{surface}} - T_{\infty}) \tag{7}$$

where Q is the heat removal rate (W), h is the convective heat transfer coefficient (W/m²·K), A is the effective heat exchange area (m²), and $T_{\text{surface}} - T_{\infty}$ is the temperature difference between the heated surface and ambient air.

For forced convection heat transfer, h scales approximately with the square root of the flow velocity [6]. Since pump and fan RPM are proportional to flow velocity, the resulting relationship between RPM and h is:

$$\frac{h_2}{h_1} = \left(\frac{\text{RPM}_2}{\text{RPM}_1}\right)^{0.5} \tag{8}$$

A 2 % change in RPM therefore changes h by only about 1 % which in turn changes Q by 1 %. This shift in cooling capacity is one-tenth of the 10 % power mismatch threshold defined in the use-case requirements, ensuring that cooling drift does not cause or obscure anomaly detection.

To achieve this precision, the pump and fan PWM control signals operate at a carrier frequency above 25 kHz. This allows the fan and pump to maintain steady RPM within 2 % accuracy for consistent heat-transfer conditions.

2. Abnormal Flow Detection

To meet the abnormal flow detection use-case, the test bed's servo-controlled valve must be capable of restricting flow rate between 0-30 % in discrete 5 % increments with ±2 % repeatability. This requirement ensures that the system can reliably generate calibrated and repeatable reductions in flow rate to train and validate the anomaly detection model. The 0-30 % range provides sufficient margin around the 20 % target restriction while extending to 25-30% for calibration and margin testing. Reductions below 20 % need to be tested to verify that the model does not trigger an anomaly flag on minor coolant flow variations. The 5 % incremental step size offers a practical balance between resolution and control stability, and the ±2 % repeatability ensures consistent fault injection and reliable comparison across tests.

3. Accurate Alerts

To meet the accurate alert use-case requirement, the anomaly detection model must maintain a FPR below 5 % and a FNR below 5 %. Two design requirements were derived for the autoencoder to satisfy this use-case requirement.

3.1 Anomaly Detection Accuracy

The target FPR and FNR below 5 % follows directly from the use-case requirement and defines the quantitative accuracy the model must achieve during validation. For the autoencoder-based anomaly detector, these metrics are determined by selecting the reconstruction error threshold that balances detection sensitivity and false-alarm probability.

3.2 Regression Model Accuracy

The regression model that predicts expected power from temperature and RPM inputs must achieve a Root Mean Square Error (RMSE) of less than 2 % to ensure that the predicted power P_{pred} closely matches the measured power P_{meas} by (9):

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(P_{\text{pred},i} - P_{\text{meas},i}\right)^2} < 0.02\,P_{\text{meas,avg}} \tag{9}$$

This requirement applies to the trained regression model's accuracy during normal operation and is distinct from the hardware power control accuracy defined in Section 1.1.

Keeping RMSE below 2 % ensures that the model

uncertainty remains small compared to the 10 % power deviation threshold in Section II.1, so the residual input to the autoencoder reflects true deviations rather than regression error. For example, at an 80 W load, a 2 % RMSE corresponds to ±1.6 W average prediction error, well below the 12.6 W mismatch threshold from Section 1.1. This allows the anomaly alerts to indicate real power deviations rather than prediction noise.

4. Timely Alerts

To meet the use-case requirement of issuing an anomaly alert within 1 second from the fault, the total latency budget is divided among three sequential processes: regression inference, anomaly detection inference, and alert transmission.

4.1 Regression Inference Latency

The regression model predicting expected power from temperature and RPM features must complete inference within 300 ms. This time is based on prior benchmarks showing that the average inference time of several neural network models achieve inference times of below 300 ms on the Pi 5 [2]. Since our regression model uses a low-complexity model and only a few features, coolant temperature, radiator air temperatures, and RPM values, this inference time should be achievable.

4.2 Autoencoder Inference Latency

The anomaly detection autoencoder must complete execution within 650 ms. Together with the regression time, this keeps total model side latency under 950 ms, leaving margin for alert transmission. This budget is supported by prior work where a similar autoencoder-based anomaly detection framework achieved real-time inference latency below 650 ms with more features [30]. Thus, the 650 ms limit is reasonable and gives margin for runtime overhead in real deployment.

V. Design Trade Studies

1. Power Simulation Methods

Two approaches were evaluated for generating a controllable and repeatable thermal load within the cooling loop: using an actual PC with a CPU and VRM as the heat source and using a resistive load to emulate the same power dissipation characteristics under controlled conditions.

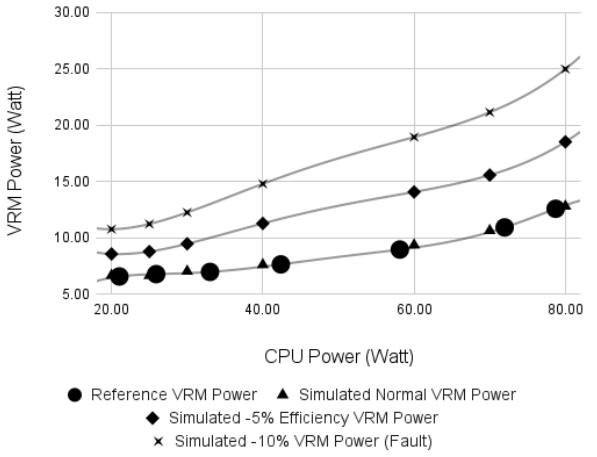

Fig. 4. VRM power under different CPU loads.

1.1 Actual PC Approach

In this approach, the CPU and VRM act as natural heat sources by running high-intensity computational workloads. Tools such as Prime95, AIDA64, and IntelBurnTest stress the CPU through continuous calculations like prime generation and floating-point operations, driving utilization near maximum. Thermal power can be modulated by adjusting test parameters.

This method produces realistic power and temperature behavior, capturing transient effects, VRM switching losses, and load dynamics. However, it is unsuitable for controlled or repeatable fault experiments. Simulating VRM degradation such as partial phase failure risks irreversible hardware damage, preventing consistent data collection and model retraining. Thus, the method is impractical and cost-prohibitive for experimental use.

1.2 Resistive Load Approach

The second approach uses power resistors to simulate the heat generated by the CPU-VRM power stages. By controlling the average voltage via PWM duty cycle, power dissipation can be adjusted to represent different load levels or efficiency losses matching real VRM behavior. This method means faults can be simulated without hardware damage, load conditions are repeatable across runs for consistent data collection, and efficiency losses can be programmed to represent varying degrees of VRM degradation for anomaly detection.

As shown in Fig. 4 and Fig. 5, the resistors can be configured to closely match a VRM efficiency curve obtained from manufacturer data sheets [35]. Fig. 4 shows how simulated VRM power output increases with CPU load under different CPU loads, and Fig. 5 illustrates the corresponding efficiency curves. The -5 % and -10 % curves simulate minor and major degradation cases, respectively, providing clear separation between normal and faulted operation for data labeling. The resistive load method was selected as the approach for this system because it is more practical and repeatable for replicating degradation faults.

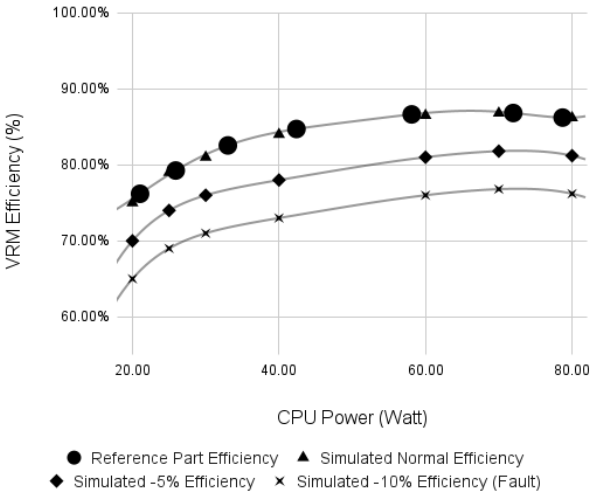

Fig. 5. VRM efficiency curves under different CPU loads.

2. Loop Construction and Control

To accurately simulate the behavior of a real-world liquid-cooling system, the test loop must replicate both the heat transfer and flow characteristics found in PC cooling applications. Two construction approaches were evaluated: using a sealed all-in-one (AIO) cooler and building a custom water-cooling loop from individual components.

2.1 AIO Cooler Approach

An AIO cooler integrates a pump, radiator, tubing, and water block into a compact, factory-sealed unit. AIOs are widely used in desktop PCs and small form factor servers because they provide simple, efficient self-contained cooling without user assembly [10]. However, the sealed configuration prevents internal access for installing sensors or injected controlled faults such as partial blockages without disassembling and permanently altering the device which introduces the risk of leakage. Key parameters, like flow rate, total fluid volume, and effective loop heat capacity also cannot be measured directly, limiting its usefulness for thermal profiling and fault testing.

2.2 Custom Loop Approach

A custom water-cooling loop was chosen to provide full control over thermal and flow parameters. It uses standard PC components: a DC pump, radiator, reservoir, copper water block, and flexible tubing. The configuration allows measurement and tuning of key parameters:

1. Pump flow rate which determines the convective heat transfer through the water block and radiator.
2. Radiator thermal resistance is determined from the radiator fin surface area, fan speed, and airflow characteristics.
3. Total fluid volume which defines the system's transient temperature response.

With these parameters defined, the total heat capacity of the loop can be estimated using (2) and the temperature response to a step change in input power can be modeled using (4). Together, these relations establish a basis for correlating CPU or VRM power fluctuations with measurable temperature changes. This configuration reproduces realistic single-CPU cooling conditions while providing a controllable testbed for collecting labeled data under both normal and fault states.

3. Flow Reduction Simulation Method

To simulate partial flow restriction in the water loop, two approaches were evaluated: mechanically pinching the tubing to reduce its cross-sectional area and installing a controllable valve to regulate flow.

The tube pinching method is simple to implement but has poor precision and repeatability. Flexible silicone tubing can deform inconsistently under pressure, and its stiffness changes over time with heat exposure and aging. Small variations in applied force can cause large differences in actual flow rate which makes it difficult to achieve consistent levels of restriction between tests. Furthermore, repeated pinching can weaken the tubing and introduce leaks over time.

The servo-controlled valve provides a more accurate and repeatable solution for simulating partial flow restriction. The valve position can be precisely adjusted using PWM control. By turning the valve to defined angles, the flow rate can be reduced by known percentages relative to the fully open position. This approach offers greater accuracy, stability, repeatability, and long-term durability compared to the tube pinching method.

4. Processors

Processor selection focused on balancing computing power with reliable control. The Raspberry Pi 5 was chosen as the main controller because it can easily handle data collection and run the machine learning model with its strong processing capability [2]. Other small computers such as the NVIDIA Jetson Nano or BeagleBone Black were considered, but they are either more expensive, consume more power, or require additional setup for integration.

Since the Pi 5 runs a normal Linux operating system, it cannot guarantee precise timing for hardware control. To provide deterministic operation, a Raspberry Pi Pico microcontroller is used alongside the Pi 5. The Pico runs code directly on its chip and generates the PWM signals for the pump, fan, and valve without an operating system, so its timing is predictable and consistent [8]. Alternatives such as the Arduino Uno or ESP32 were also considered, but the Pico offered higher PWM resolution, simple UART communication, and lower cost.

5. Sensors

Accurate temperature sensing is essential for the anomaly detection model training. The system requires sensors that can detect small temperature changes, fit into the physical layout of the cooling loop, and have long-term reliability under continuous operation.

For measuring the radiator inlet and outlet air temperature, which are the main features used for power prediction, a temperature sensor with enough resolution to measure the small temperature response outlined in Section II.1 was required. The TMP117 provides ±0.1 °C across the 20-50 °C range and communicates using the I²C interface, allowing multiple sensors to share one connection line to the Pi 5. Compared with other I²C sensors such as the LM75A, which offers only ±0.2 °C accuracy, the TMP117 achieves much higher precision while remaining cost effective. Unlike analog sensors such as thermocouples or thermistors, the TMP117 produces a direct digital reading and does not require an ADC. This reduces wiring complexity and prevents signal drift from cable length or electrical noise. Overall, the TMP117 provides the best balance of accuracy, reliability, and cost for this application.

A separate coolant temperature sensor is also installed at the reservoir port using a standard G1/4 fitting for compatibility with common water-cooling components. This is an analog sensor (NTC-based probe, ±1 °C typical accuracy) used as a supplementary reading of bulk coolant temperature.

Since this channel is analog, a few low-cost ADCs were considered to connect to the Pi 5 which does not have a built-in ADC. Coolant temperature changes slowly, so very high sample rates are unnecessary. The key factors for selecting an ADC were easy integration with the I²C temperature sensors and low noise. Table 1 shows the ADC options considered with

their specifications. The ADS1115 was chosen because it integrates easily with the other I²C sensors in the Qwiic ecosystem and includes a differential input mode for noise rejection.

Table 1. ADC Comparison

Parameters	Options		
	MCP3008	***ADS1015***	***ADS1115***
Sampling Rate	200 kHz	3.3 kHz	0.86 kHz
Sampling Depth	10 bit	12 bit	16 bit
Channel Count	8	4	4

6. Model Selection

Several types of models are commonly used for anomaly detection, including regression, decision tree, random forest, isolation forest, and autoencoder. Each model was evaluated based on its ability to detect gradual degradation and accurately flag anomalies on correlated sensor data.

6.1 Tree-based Models

Tree-based models such as decision tree, random forest, and isolation forest are widely used for anomaly detection due to their accuracy on static datasets [17], [24], [29]. However, they classify each sample independently and cannot capture gradual or time-dependent changes. In this system, degradation such as partial cooling blockages or reduced VRM efficiency develops slowly over time, producing small correlated drifts in temperature and power. Tree-based models lack temporal awareness, so these drifts are often treated as normal variation until the fault becomes severe. Thus, these do not fulfill our use case of early detection of gradual degradation.

6.2 Autoencoder

Autoencoders achieve high accuracy on multivariate anomaly detection tasks, up to 99.37% [30], and perform especially well on contextual and temporal data [7], [20]. They learn the normal relationships between temperature and power over time, reconstructing expected behavior and identifying anomalies when the reconstruction error exceeds a defined threshold. By tuning this threshold appropriately, the autoencoder can detect small deviation that indicate gradual system degradation. Thus, autoencoder fulfills our use case of detecting slow, progressive degradation over time.

6.3 Regression

Although the autoencoder effectively captures gradual deviations, the system's sensor data are strongly correlated through predictable thermal relationships defined in (1). Such correlations can lead to false positives when normal fluctuations, such as transient power spikes, are misclassified as faults [4]. To improve alert precision and stability, the system uses a hybrid regression and autoencoder model. The regression stage preprocess data by predicting power from the sensor data, and the autoencoder evaluates the residual between measured and predicted power to detect degradation. This hybrid approach reduces false positives from correlated data and improves response times [4].

Elastic Net regression is used instead of L1 (Lasso) or L2 (Ridge) regularization because correlated inputs, such as temperature and power that change together according to (1), make the model prone to overfitting, especially since testing can only cover a limited range of operating conditions. L1 forces some coefficients towards zero to remove weak predictors while L2 distributes weight more evenly across related inputs to avoid instability. Elastic Net combines both effects by applying a weighted mix of L1 and L2 penalties during training. As a result, Elastic Net makes the regression model generalize better to unseen data.

VI. System Implementation

As described in Section III, the system is implemented as two subsystems: the testbed hardware which physically simulates the liquid-cooling environment and the model subsystem which performs the anomaly detection. The Pi 5 serves as the central compute and data acquisition node, and the Pico functions as the controller for the testbed hardware.

1. Testbed Subsystem

The testbed subsystem implements the physical cooling loop used to collect simulation data for model training and to validate anomaly detection performance. It includes the water loop, heater complex, fault injection assembly, sensor network, and power distribution network. The water loop, heater complex, and fault injection subsystems are controlled by the Pico, which receives UART commands from the Pi 5 specifying PWM values for the fan, pump, and SSRs.

1.1 Water Loop

The cooling loop is assembled from standard PC liquid-cooling components to ensure realistic thermal performance and maintainability. A 12 V pump circulates coolant from an acrylic reservoir through a copper CPU water block and into a 120 mm radiator. A 12 V fan mounted on the radiator dissipates heat from the coolant to ambient air. Both the pump and fan receive PWM control signals from the Pico. Coolant flows through 6 mm polyurethane tubing secured with clamps and brass fittings. The assembled loop holds approximately 300 mL of coolant, selected to match the modeled thermal time constant derived in Section II.1. A SolidWorks model of the assembled loop is shown in Fig. 6, illustrating component placement.

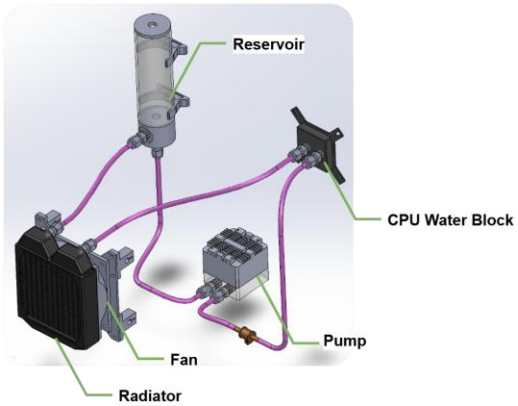

Fig. 6. SolidWorks model of the water loop.

1.2 Heater Complex

Two power resistors simulate CPU and VRM power dissipation, as shown in Fig. 7. Each 5 Ω, 100 W resistor is mounted to an aluminum spreader plate for power dissipation. Two SSRs drive the resistors, and the load is varied by PWM control from the Pico to the SSRs. The CPU heater operates between 20-80 W, and the VRM heater operates between 1-30 W. The Pico receives PWM commands from the Pi 5 to control the SSRs for each programmed load step on the resistors.

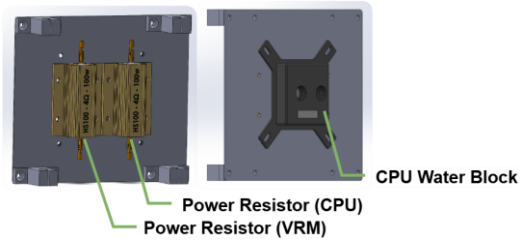

Fig. 7. SolidWorks model of the heater complex.

1.3 Fault Injection Assembly

Two types of faults are implemented: flow restriction and power mismatch. For the first, the Pico uses PWM to control two 5 V servo motors that are attached to an in-line valve (Fig. 8), allowing partial closure of the valve to restrict flow. To map the PWM control commands to actual flow rate reductions, a flow meter will be used during calibration to determine the correspondence between PWM value and percentage of flow rate reduction.

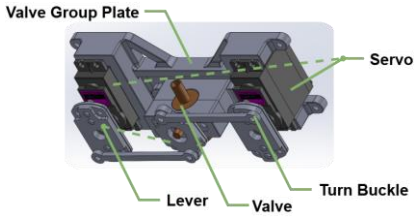

Fig. 8. SolidWorks model of the flow restriction fault injection assembly.

For power mismatch faults, the Pico adjusts the PWM duty cycle for the VRM SSR-resistor pair to produce an injected hidden power that is not reflected in the measured power, creating a controlled mismatch between measured and predicted power. Both mechanisms are initiated by the Pi 5 through serial UART command sequences.

1.4 Sensor Network

As shown in Appendix, Fig. 10, all sensors connect to the Pi 5 via I²C for synchronized data acquisition without interfering with UART communication with the Pico. A Qwiic shim adds convenient access to the 3.3 V, I²C, and ground lines. Two TMP117 temperature sensor boards, mounted at the radiator inlet and outlet, provide ±0.1 °C accuracy, and an in-line analog sensor at the reservoir measures coolant temperature. An ADS1115 ADC, daisy-chained with the TMP117s over I²C, digitizes the analog signal. Fan and pump tachometers are read through GPIO pins to monitor RPM. All sensor data are logged locally in InfluxDB on the Pi 5 for model training and validation.

1.5 Power Distribution and Mechanical Structure

The system is powered by a 24 V DC supply connected to a wall outlet. The Pi 5 has its own power input from wall adapters, and the Pico is powered through the Pi 5's 5V rail. A 24 V to 12 V DC-DC buck converter powers the pump and fan, and a separate 24 V to 5 V converter powers the servos. Each converter output is distributed through WAGO terminals to create shared power rails for components operating at the same voltage. The 24 V supply also powers the SSRs that drive heater resistors during load variation. Components are mounted on custom cut plates secured to a modular frame of aluminum rods and connectors as shown in Fig. 9.

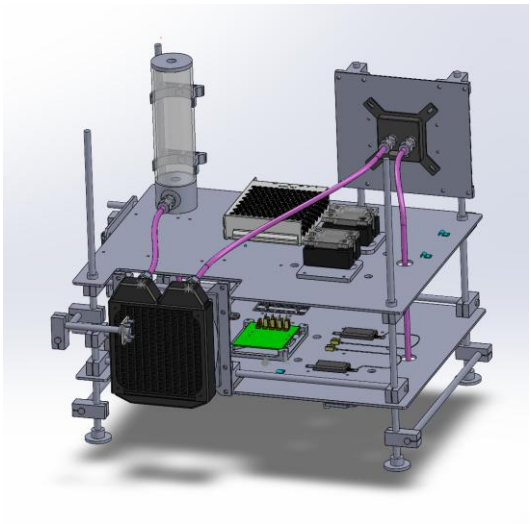

Fig. 9. SolidWorks model of hardware system.

2. Model Subsystem

The model subsystem performs on-device inference for real-time anomaly detection. It consists of a regression model for power prediction, residual and feature computation, and an autoencoder trained on normal operation data. The Pi 5 handles all model computation and evaluation locally to maintain low latency and continuous operation.

3.1 Regression Model

The regression model is implemented in Python using scikit-learn's Elastic Net regressor. It is trained using data exported from InfluxDB and collected from the testbed operating under normal conditions. Each feature, coolant temperature, radiator inlet and outlet air temperature, and pump and fan speeds is standardized before training. Model coefficients are obtained through five-fold cross-validation to minimize mean-squared error and prevent overfitting. During operation, predicted power values are logged in InfluxDB along with the sensor data and commanded heater power from the Pi 5.

3.2 Residual and 30-Second Window Feature Extraction

During operation, the Pi 5 computes the residual given by:

$$r = \frac{P_{\text{meas}} - P_{\text{pred}}}{P_{\text{meas}}} \tag{10}$$

and continuously buffers the most recent 30 seconds of residual

data. A Python routine maintains this sliding window and computes four summary features every 5 seconds, mean, maximum, standard deviation, and slope. These features are appended to a rolling data frame and passed to the autoencoder input layer. All calculations are performed on the Pi 5 in the same process as the regression model to minimize overhead.

3.3 Autoencoder

The autoencoder is trained on residual features collected every 5 seconds during normal testbed operation, providing approximately 17,000 sample per day. It takes in five input features, the most recent residual along with the mean, maximum, standard deviation, and slope of the residual over a 30-second window, and learns to reproduce these same values as its output. Inside the network, several connected layers first compress the input information into a smaller internal representation, encoding, and then reconstruct it back to the original form, decoding. The difference between the input and output is the reconstruction error which measures how much the current operating condition deviates from normal behavior.

After training, the reconstruction error is recorded for many normal samples across a range of conditions. The average reconstruction error plus three standard deviations is used as the threshold for normal operation, corresponding to approximately the 99.7 % confidence interval of the training data [25]. This threshold ensures that only statistically significant deviations are flagged as anomalies. The threshold value is stored in a configuration file on the Pi 5 and loaded when the program starts.

At runtime, the Pi 5 evaluates new data every 5 seconds, matching the sensor sampling rate. Each inference cycle uses the most recent 30 seconds of residual data to generate an updated reconstruction error. Since the 30-second window already smooths transient noise, a reconstruction error above the threshold is immediately considered anomalous.

Once an anomaly is detected, the Pi 5 classifies the fault type based on temperature response patterns. The program compares the rate of change of the coolant temperature and the radiator outlet air temperature over the last 60 seconds. If the coolant temperature rises faster while the outlet air temperature remains relatively steady, the anomaly is labeled a flow restriction fault. If both temperatures rise together, it is labeled a power mismatch fault since additional heat is entering the loop. Each detection cycle generates a small JSON packet containing the timestamp, anomaly flag, and fault label. This packet is sent to the alert subsystem for immediate logging or alert notification.

When an anomaly is detected, the model writes an entry to a local log file and prints an alert to the Pi 5 terminal. Each log entry includes the timestamp, fault label, measured and predicted power, residual value, and anomaly flag. These outputs provide enough detail to verify the model's decision and analyze system behavior during fault conditions.

VII. Test, Verification and Validation

To meet the quantitative specifications defined in Section II and IV, this section outlines the testing methods used to verify and validate the system implementation. The validation plan is divided into tests for the hardware design requirements and tests for the model and alert system requirements.

1. Testbed Requirements Tests

1.1 Heater Control Accuracy

To verify that the CPU and VRM heaters maintain power control accuracy within 2 % RMs, both channels are swept across their respective operating ranges: 20-80 W for the CPU heater and 1-30 W for the VRM heater. For each commanded PWM level, the voltage is measured using a digital multimeter (DMM). The resistance of each resistor is measured beforehand using the same DMM to ensure consistent calibration. The instantaneous power at each setting is then computed as:

$$P = \frac{V^2}{R} \tag{11}$$

where V is the measured voltage across the resistor and R is its measured resistance.

The expected heater power from the Pi 5's PWM command is compared against the calculated P value from measurements. The test passes if the RMS deviation between commanded and measured power is less than 2 % across the full range, confirming that the power control system maintains sufficient accuracy to prevent control noise from interfering with the 10 % anomaly detection threshold.

1.2 Pump and Fan Control Accuracy

The pump and fan are swept from 20 % to 100 % PWM duty in 5 % increments. For each step, the Pi 5 records steady-state RPM from the tachometer output over a 10 second window. The measured RPMs are compared to the commanded values to verify control accuracy within ±2 %. The test passes if all readings remain within this tolerance, confirming that the PWM control maintains stable and repeatable flow and airflow during operation.

1.3 Valve Control Accuracy

The servo-controlled valve is tested from 0 % to 30 % restriction in 5 % increments. Actual flow rate is measured using an inline flowmeter during this test for each setting, and each position is repeated three times to evaluate repeatability. A lookup table mapping PWM command to flow rate reduction percentage is generated from these measurements. The test passes if flow restriction error and repeatability are both within ±2 %, verifying that the valve can reliably produce calibrated, repeatable flow faults.

2. Model Validation Tests

2.1 Abnormal Power Detection Validation

To validate abnormal power detection under realistic operating conditions, the system is tested across multiple CPU heater workloads ranging from 20-80 W. For each test, the VRM heater power is increased by +10 % relative to the corresponding CPU load to simulate degraded efficiency faults. The workloads include steady-state levels as well as dynamic usage patterns such as step changes, ramp-up and ramp-down transitions, and short idle-to-load spikes to reflect real processor

behavior.

The Pi 5 logs measured and predicted power continuously, recording the time of fault injection and the alert trigger. The test passes if the model detects a sustained greater than 10 % deviation for at least 30 seconds across all tested workloads and issues an alert within 1 second of the threshold crossing. This verifies that the anomaly detection pipeline reliably distinguishes true power and mismatches from normal workload transients and meets both the magnitude and timing requirements defined in the use-case specification.

2.2 Abnormal Flow Detection Validation

To validate the system's ability to detect blockages in the coolant loop, tests are conducted with the CPU and VRM heaters operating under a range of workloads. To isolate the fault, the VRM will be kept at normal efficiency. For each test, the servo valve applies a 20 % flow restriction, determined from the PWM to flow rate lookup table. The restriction is applied both during steady-state operation and during load transitions such as ramp-up, ramp-down, and idle-to-load conditions.

The Pi 5 records the time of fault initiation and the alert trigger for each trial. The test passes if the model detects a sustained greater than 20 % flow reduction that persists for at least 30 seconds and raises a flow fault alert within 1 second of threshold persistence.

2.3 Model Accuracy Validation

To verify regression model accuracy, predicted power values are logged along with measured heater power during fault-free runs across varying workloads as described in Section 2.1. RMSE (9) is computed from the measured and predicted power, and the test passes if RMSE is less than 2 % of the average measured power across all runs.

To validate anomaly detection accuracy, the autoencoder is evaluated using logged datasets from fault-free and fault-injected runs. The data is divided into 30 second windows, and for each window the model outputs an anomaly flag and anomaly label. These predicted flags and anomaly labels are then compared against the know true anomaly flags and fault labels for the same time intervals. The FPR and FNR are computed from this comparison, and the test passes if both remain below 5 %.

2.4 Latency Validation

End-to-end latency is measured from the time the regression model computes the predicted power corresponding to the 30th second of a sustained fault to the time the alert message is displayed in the Pi 5 terminal. This captures the delay of the detection and alert pipeline once the 30 second persistence requirement has been satisfied.

Time stamps for the final regression computation and the alert display are logged automatically. The total latency is the difference between the two. The test passes if the average latency across power and flow fault trials remains below 1 second, confirming that the combined inference and alert transmission processes meet the real-time performance requirement.

VIII. Project Management

1. Schedule

The overall project schedule is shown in Fig. 11 in the Appendix. The project is divided into six phases. The first four include physical construction, research, data collection software development, and valve and PWM control development. These can run in parallel with minimal dependencies to split work efficiently. Once the testbed and control systems are completed, the final two phases, data collection and model bring-up, proceed sequentially. These phases rely on the completed hardware to generate datasets for training and validating the anomaly detection model.

2. Team Member Responsibilities

The project is divided as follows:

- Kristina is responsible for hardware integration, CAD modeling, and testbed assembly, and will also lead anomaly detection model development
- Jacob is in charge of Pi 5 software development, including sensor data collection, database integration, and alert system implementation. He will also lead data collection.
- Aidan will develop and test the valve control and PWM control code, ensuring accuracy and precision for the heater complex, fan, pump, and servo valves.

All team members will collaborate on system integration, debugging, and the final ML model bring-up.

3. Bill of Materials and Budget

The complete Bill of Materials is provided in Table 2.

4. TechSpark Use Plan

We do not plan to use TechSpark for this project.

5. Risk Mitigation Plans

One risk is that the combined anomaly detection model may not meet the FPR and FNR targets because of differing thermal patterns between power and flow faults. If this occurs, the system will use two separate autoencoder-based models: one trained specifically to detect abnormal power mismatches and another to detect abnormal flow restrictions. This separation allows each model to learn feature patterns relevant to its fault type and apply individually tuned reconstruction error thresholds to improve detection accuracy and reduce misclassification.

Another risk is that the fan and pump could overcool the system which would be an inaccurate simulation of a real PC or server. Temperature changes from injected faults may become too small to detect reliably as a result. To address this, fan and pump speeds will be reduced during calibration to increase the coolant temperature rise and better simulate real PC and server thermal conditions.

Fault injection and high-load testing pose risks of overheating, coolant leakage, or thermal damage if control signals fail. The Pico includes a heartbeat safety routine that continuously monitors communication with the Pi 5. If communication is lost, the Pico automatically drives the pump

and fan to maximum speed, opens the valve fully, and disables both heaters, returning the system to a safe state.

IX. Related Work

1. IoT Module for Vacuum Pump Preventative Maintenance [14]

This project develops an IoT module designed to attach directly to vacuum pumps for predictive maintenance. The system collects vibration, acoustic, and temperature data to identify abnormal performance patterns that could indicate mechanical wear or impending failure. It represents a growing trend in integrating IoT hardware with machine learning to improve equipment reliability and operational efficiency. Compared to our project, their system depends heavily on the deployment of physical sensors and embedded hardware for data acquisition. Our project's main goals are automatic alerting and data-driven anomaly identification without the need for an add-on module. While monitoring the health of physical systems is a common goal of both projects, ours places more emphasis on using existing sensors to monitor the system rather than integrating additional sensors.

2. Autoencoder Based Anomaly Detection and Explained Fault Localization in Industrial Cooling Systems [16]

This study examines the ability of autoencoders to locate and detect issues in extensive industrial cooling systems. The system can identify instances in which specific sensor readings substantially vary from typical operating behavior by calculating the reconstruction error between observed and predicted data. Our project and this method are very similar in that they both use feature reconstruction and unsupervised learning to identify anomalies in complex data. Our effort focuses on flexible anomaly detection in a controlled testbed setting, whereas their work focuses on high-dimensional industrial cooling systems. This is the primary difference in application scope. Autoencoders are a useful tool for identifying abnormalities in physical systems that have multiple interconnected variables.

3. Industrial IoT System for Pump Condition Monitoring [21]

This project offers an industrial Internet of Things framework for monitoring on mechanical pump health. To anticipate possible mechanical deterioration before failure happens, the system gathers temperature and vibration data in real time and uses signal analysis algorithms. By fusing sensor input with cloud-based data analytics, its design prioritizes scalability and dependability in industrial settings. Although the general objective of predictive maintenance is the same as ours, its implementation is different. In contrast to their work, which relies on physical sensors and rule-based signal processing, our study uses machine learning algorithms to automatically identify deviations and learn typical system behavior. While both methods highlight the value of proactive monitoring, our research uses AI models rather than fixed thresholds to increase intelligence and adaptability.

X. Summary

This project demonstrates a preventative maintenance alert system that detects early signs of degradation in liquid-cooled PCs and servers before critical temperature thresholds are reached. Using a hybrid regression and autoencoder model, the system identifies gradual changes in power dissipation efficiency that indicate coolant blockages as well as hidden increases in power loss that signal VRM degradation. By analyzing real-time thermal and power data, it provides early warnings that allow operators to perform maintenance during planned service windows instead of after a server outage.

The approach establishes a framework for integrating predictive thermal diagnostics into cooling systems by leveraging existing temperature sensors and RPM feedback. Remaining work includes data collection, training the model on normal and fault conditions, validating its performance, and refining detection thresholds for consistent accuracy across workloads. Once completed, the prototype will demonstrate a practical method for anomaly detection in PC and server thermal management.

Glossary of Acronyms

ADC – Analog-to-Digital Converter
AIO – All-In-One
AUC – Area Under the Curve
BMC – Baseboard Management Controller
CPU – Central Processing Unit
DC-DC – Direct Current to Direct Current (converter)
DMM – Digital Multimeter
FNR – False Negative Rate
FPR – False Positive Rate
GPIO – General Purpose Input/Output
I²C – Inter-Integrated Circuit
JSON – JavaScript Object Notation
ML – Machine Learning
PID – Proportional Integral Differential
Pi 5 – Raspberry Pi 5
Pico – Raspberry Pi Pico
PWM – Pulse Width Modulation
Qwiic – Quick Interface for I²C Connection (System by SparkFun)
RMSE – Root Mean Square Error
RPM – Revolutions Per Minute
SSR – Solid-State Relay
UART – Universal Asynchronous Receiver-Transmitter
VRM – Voltage Regulator Module

References

[1] Advanced Thermal Solutions, “ATS Liquid Cooling eBook Select Technical Articles on Liquid Cooling and its Various Roles in the Thermal Management of Electronics,” QATS. Available: https://www.qats.com/cms/wp-content/uploads/Liquid-Cooling-eBook2.pdf

[2] A. Allan, “Benchmarking Raspberry Pi 5,” *Raspberry Pi*, Oct. 20, 2023. https://www.raspberrypi.com/news/benchmarking-raspberry-pi-5/

[3] B. Azkaei, K. C. Joshi, and G. Exarchakos, “Machine Learning-Driven Anomaly Detection for 5G O-RAN Performance Metrics,” *IEEE INFOCOM 2025 - IEEE Conference on Computer Communications*

Workshops (INFOCOM WKSHPS), pp. 1–6, May 2025, doi: https://doi.org/10.1109/INFOCOMWKSHPS65812.2025.11152997.
[4] T. Baranwal, A. Das, S. Varada, S. Das, and M. R. Haider, "Machine learning-based anomaly detection of correlated sensor data: An integrated principal component analysis-autoencoder approach," 2025. doi: https://doi.org/10.1109/LANMAN66415.2025.11154513.
[5] S. Barbhuiya, Z. Papazachos, P. Kilpatrick, and D. Nikolopoulos, "RADS: Real-time Anomaly Detection System for Cloud Data Centres," Nov. 2018. Accessed: Oct. 07, 2025. [Online]. Available: https://arxiv.org/pdf/1811.04481
[6] T. L. Bergman, *Fundamentals of heat and mass transfer: Theodore l. Bergman ... [et al.].* Wiley, 2017. Available: https://books.google.com/books?id=OViz0AEACAAJ
[7] M. Braei and S. Wagner, "Anomaly detection in univariate time-series: A survey on the state-of-the-art," Apr. 2020, doi: https://doi.org/10.48550/arXiv.2004.00433.
[8] J. Butts, "7 superpowers of a Raspberry Pi Pico that beat the regular Pi," *XDA*, Feb. 07, 2025. https://www.xda-developers.com/7-superpowers-of-a-raspberry-pi-pico-that-beat-the-regular-pi/ (accessed Oct. 07, 2025).
[9] Cadence PCB Solutions, "What is Signal to Noise Ratio and How to calculate it?," *Cadence*, 2023. https://resources.pcb.cadence.com/blog/2020-what-is-signal-to-noise-ratio-and-how-to-calculate-it
[10] Corsair Gaming, "AIOs vs Custom Cooling: Which is better?," *Corsair*, Feb. 27, 2025. https://www.corsair.com/us/en/explorer/diy-builder/custom-cooling/aios-vs-custom-cooling-which-is-better/?srsltid=AfmBOoqya4ifCpr-06flTmZ32-kLRk16-NrOELtz8ozdEUVOeTSyd5pl (accessed Oct. 07, 2025).
[11] Dell, "PowerEdge Servers Error and Event Messages Reference Guide | Dell European Distribution Business," *Dell*, 2018. https://www.dell.com/support/manuals/en-ed/poweredge-t560/error_event_message_guide_b/thrmthermal-event-messages?guid=guid-5f92b138-10fd-4e73-9e2f-eae4e6f53c7e&lang=en-us (accessed Oct. 07, 2025).
[12] EKWB, "Custom CPU Loop," *EKWB*, Aug. 18, 2021. https://www.ekwb.com/solutions/custom-loop/#needed-components (accessed Oct. 07, 2025).
[13] EKWB, "EK-D5 PWM G2 Motor (12V DC PWM Pump Motor)," *EKWB*, 2025. https://www.ekwb.com/shop/ek-d5-pwm-g2-motor-12v-dc-pwm-pump-motor?srsltid=AfmBOor-VThyuhWKKJLnjKZ7HyHAnfcVi6aTsEZjXuRUY_bDfmWiiQfy (accessed Oct. 07, 2025).
[14] G. Fedder, "IoT Module for Vacuum Pump Preventative Maintenance," *Carnegie Mellon University Electrical & Computer Engineering Student Project Tracker*. https://spt.apps.ece.cmu.edu/project/1581
[15] N. Hidayat, R. F. Iskandar, and M. Rokhmat, "Personal computer temperature control using PID control based liquid cooling system," 2015, pp. 169–171. doi: https://doi.org/10.1109/ICACOMIT.2015.7440199.
[16] S. Holly *et al.*, "Autoencoder based Anomaly Detection and Explained Fault Localization in Industrial Cooling Systems," *arXiv.org*, 2022. https://arxiv.org/abs/2210.08011 (accessed Oct. 07, 2025).
[17] A. Huč, J. Šalej, and M. Trebar, "Analysis of Machine Learning Algorithms for Anomaly Detection on Edge Devices," *Sensors*, vol. 21, no. 14, p. 4946, Jul. 2021, doi: https://doi.org/10.3390/s21144946.
[18] Intel, "Information about Temperature for Intel® Processors," *Intel*. https://www.intel.com/content/www/us/en/support/articles/000005597/processors.html
[19] B. Justice, "Intel Core i5-12400 CPU Performance Review," *The FPS Review*, Feb. 22, 2022. https://www.thefpsreview.com/2022/02/22/intel-core-i5-12400-cpu-performance-review/9/ (accessed Oct. 07, 2025).
[20] J. R. Ky, B. Mathieu, A. Lahmadi, and R. Boutaba, "ML Models for Detecting QoE Degradation in Low-Latency Applications: A Cloud-Gaming Case Study," in IEEE Transactions on Network and Service Management, vol. 20, no. 3, pp. 2295-2308, Sept. 2023, doi: 10.1109/TNSM.2023.3293806.
[21] Y. Lee, C. Kim, and S. J. Hong, "Industrial Internet of Things for Condition Monitoring and Diagnosis of Dry Vacuum Pumps in Atomic Layer Deposition Equipment," *Electronics*, vol. 11, no. 3, p. 375, Jan. 2022, doi: https://doi.org/10.3390/electronics11030375.
[22] Lenovo, "Liquid Cooling vs. Air Cooling: Which Option is Best for You? | Lenovo US," *Lenovo*, 2021. https://www.lenovo.com/us/en/glossary/liquid-cooling-vs-air-cooling/?orgRef=https (accessed Oct. 07, 2025).
[23] Lenovo, "VRM Explained: Protect Your PC's Vital Components | Lenovo US," *Lenovo*, 2021. https://www.lenovo.com/us/en/glossary/vrm/?orgRef=https (accessed Oct. 07, 2025).
[24] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," *2008 Eighth IEEE International Conference on Data Mining*, Dec. 2008, doi: https://doi.org/10.1109/icdm.2008.17.
[25] D. Mindrila and M. Phoebe, "Confidence Intervals," 2013. Available: https://www.westga.edu/academics/research/vrc/assets/docs/confidence_intervals_notes.pdf
[26] R. Ness, "Water Cooling Calculator - Ness Engineering Inc.," *Ness Engineering Inc.*, 2015. https://www.nessengr.com/technical-data/water-cooling/
[27] Noctua, "Noctua PWM specifications white paper," Noctua. Available: https://noctua.at/pub/media/wysiwyg/Noctua_PWM_specifications_white_paper.pdf
[28] Pier Paolo Ippolito, "Introduction to Autoencoders: From The Basics to Advanced Applications in PyTorch," *DataCamp*, Dec. 14, 2023. https://www.datacamp.com/tutorial/introduction-to-autoencoders
[29] A. K. Sah and V. K, "Anomaly-based intrusion detection in network traffic using machine learning: A comparative study of decision trees and random forests," 2024, pp. 1–7. doi: https://doi.org/10.1109/ICNWC60771.2024.10537451.
[30] O. A. Saleh and M. Cevik, "Secure edge-based smart grid communication using lightweight authentication modeling with autoencoders and real-world data," *Discover Computing*, vol. 28, no. 1, Jun. 2025, doi: https://doi.org/10.1007/s10791-025-09643-w.
[31] Sanyo Denki, "Understanding Fan Life," *Sanyo Denki*. https://www.sanyodenki.com/global/america/documents/Sanyo_Denki_America_Understanding_Fan_Life.pdf (accessed Oct. 07, 2025).
[32] S. Strutt, C. Kelley, H. Cisco, T. Singh, and Reuters, "Data Center Efficiency and IT Equipment Reliability at Wider Operating Temperature and Humidity Ranges," DOE, Jan. 2012. Accessed: Oct. 07, 2025. [Online]. Available: https://www.energy.gov/sites/prod/files/2013/12/f5/data_center_efficiency_and_reliabilit_at_wider_operating_ranges.pdf
[33] V. Subramaniam, "AMD Ryzen 5 7600 65 W Review: Midrange US$220 gaming sweet spot that outperforms Core i9-12900K and all Zen 3 CPUs in single-core," *Notebookcheck*, Apr. 27, 2023. https://www.notebookcheck.net/AMD-Ryzen-5-7600-65-W-Review-Midrange-US-220-gaming-sweet-spot-that-outperforms-Core-i9-12900K-and-all-Zen-3-CPUs-in-single-core.708080.0.html (accessed Oct. 07, 2025).
[34] TDK-Lambda Americas, "Efficiency Calculations for Power Converters," *TDK*, 2020. https://www.us.lambda.tdk.com/resources/blogs/20120905.html
[35] Texas Instruments, "TPS53647 4-Phase, D-CAP+, Step-Down, Buck Controller with NVM and PMBus™ Interface for ASIC Power and High-Current Point-of-Load," *Texas Instruments*, Feb. 2017. https://www.ti.com/lit/ds/symlink/tps53647.pdf?ts=1759773351728&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FTPS53647
[36] "Unitrends, "Downtime: Causes, Costs and How to Minimize It | Unitrends," *Unitrends*, Mar. 22, 2021. https://www.unitrends.com/blog/downtime-causes-costs-and-how-to-minimize-it/
[37] A. Webber, "Calculating Useful Lifetimes of Embedded Processors Calculating Useful Lifetimes of Embedded Processors," Texas Instruments, 2014. Accessed: Oct. 07, 2025. [Online]. Available: https://www.ti.com/lit/an/sprabx4b/sprabx4b.pdf?ts=1758537291456&ref_url=https
[38] J. Worch, K. Sado, A. R. J. Downey, J. Khan, and E. Santi, "Real-Time Blockage Detection and Autonomous Recovery in Liquid-Cooled Systems Using Digital Twins: *The Views Expressed are Those of the Author and do not Reflect the Official Policy or Position of the Department of Defense or the U.S. Government.*," *2025 IEEE Electric Ship Technologies Symposium (ESTS)*, pp. 543–549, Aug. 2025, doi: https://doi.org/10.1109/ests62818.2025.11152430.

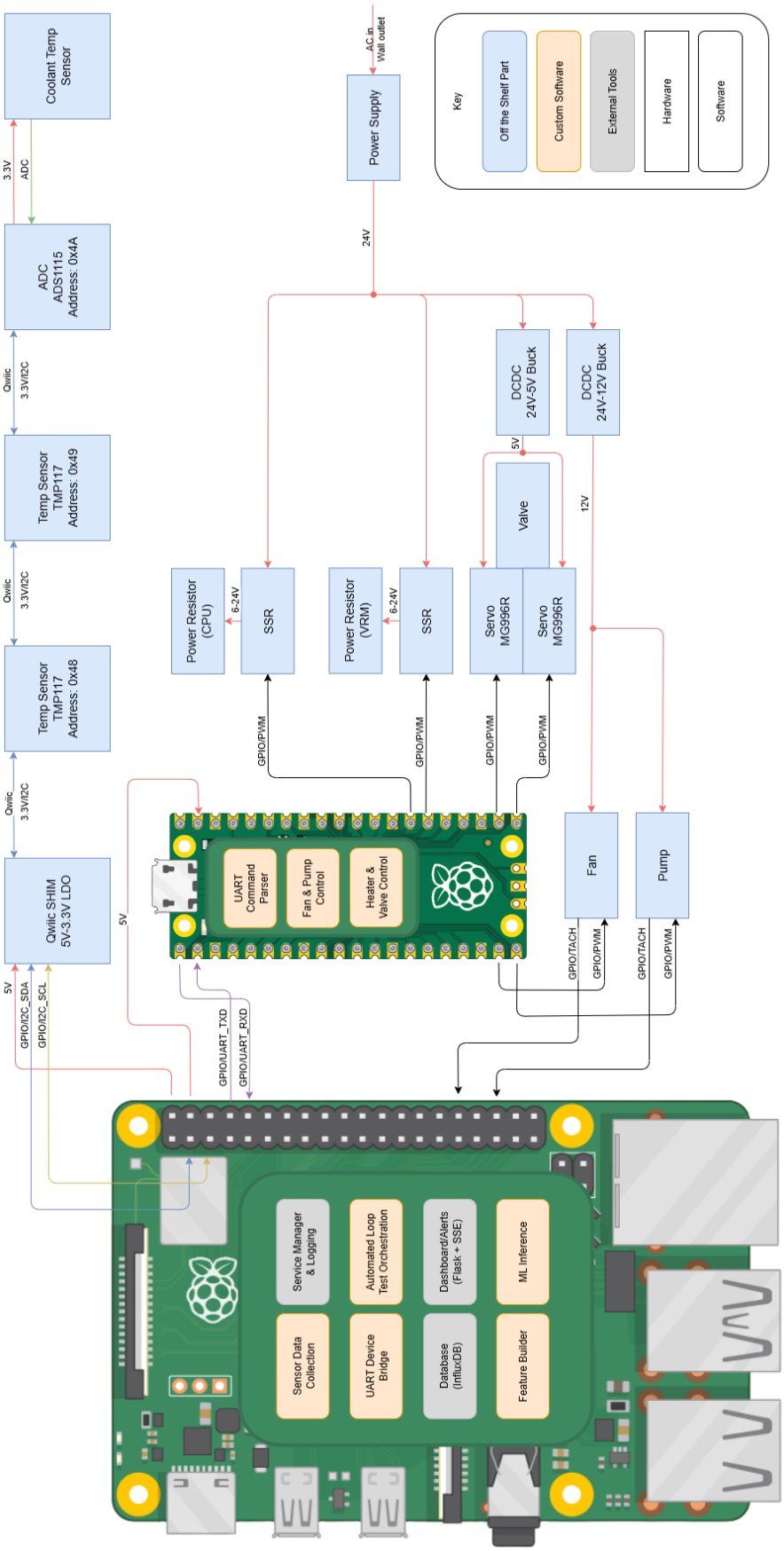

Fig. 10. Full system block diagram.

Fig. 11. Gantt Chart.

Table 2. Bill of Materials

Description	Model #	Manufacturer	Quantity	Cost @	Total
Raspberry Pi 5 8GB	SC1432	Raspberry Pi	1	$0.00	$0.00
Raspberry Pi Pico WH RP2040	SC0919	Raspberry Pi	1	$7.00	$7.00
RPi Pico Expansion Board	DFR0848	DFRobot	1	$4.90	$4.90
64GB MicroSD Card	SC0339L	Raspberry Pi	1	$0.00	$0.00
Qwiic Shim	15794	SparkFun Electronics	1	$1.95	$1.95
Pin Header	61300621821	Würth Elektronik	1	$0.37	$0.37
TMP117 Temperature Sensor	4821	Adafruit Industries LLC	2	$11.50	$23.00
Qwiic Cable Kit	15081	SparkFun Electronics	2	$0.00	$0.00
Servo Motor	MG996R	Deegoo	2	$0.00	$0.00
Silicone Tubing	-	YSIL	1	$10.99	$10.99
G1/4 Tube Fitting Adapter	-	Yosoo Health Gear	1	$7.72	$7.72
G1/4 Fitting Plug	LYSB01DVV5XNS-ELECTRNCS	BXQINLENX	1	$9.99	$9.99
G1/4 Temperature Sensor Fitting	TCWD-V1	Barrow	1	$13.99	$13.99
ADS1115 ADC	1085	Adafruit Industries LLC	1	$14.95	$14.95
Aluminum Radiator	-	Corsair	1	$59.99	$59.99
Fan	ACFAN00305A	Arctic	1	$8.49	$8.49
Water Pump	-	Sanpyl	1	$36.14	$36.14
Water Reservoir	-	Serounder	1	$18.17	$18.17
CPU Water Block	-	BXQINLENX	1	$15.98	$15.98
5 Ohm 100W Resistor	HS100 5R F	Ohmite	2	$12.37	$24.74
Solid State Relay	SSR-25DD	BlueStars	1	$18.99	$18.99
24V to 5V DC-DC Buck Converter	EA50-5V	Tobsun Electronics	1	$9.99	$9.99
24V to 12V DC-DC Buck Converter	EA120-12V	Tobsun Electronics	1	$9.39	$9.39
AC/DC Converter 24V	LRS-100-24	MEAN WELL USA Inc.	1	$15.00	$15.00
Cross Connector	-	uxcell	5	$0.00	$0.00
Panel Connector	LRSP-SK8-4P	VI-CHAN	2	$9.99	$19.98
Shaft Collar	ASDS-ZH128	Esedese	2	$12.99	$25.98
Flange Connector	-	daier	1	$7.99	$7.99
100mm Linear Motion Rod	-	Vigorous	2	$9.99	$19.98
200mm Linear Motion Rod	8DG	Generic	1	$0.00	$0.00
300mm Linear Motion Rod	8DG	Generic	1	$14.88	$14.88
400mm Linear Motion Rod	Rods8-400MM-4P	akkacm	2	$17.99	$35.98
Acrylic Floor Plate	-	Sculpteo	2	$22.82	$45.64
Aluminum Motherboard Plate	-	SendCutSend	1	$18.24	$18.24
				Grand Total	**$500.41**