PREMIER JOURNAL OF SCIENCE ORIGINAL RESEARCH

Model Z: Revolutionizing IoT Integration with Advanced Digital Twin Technology

B. Hemananth[1], N. Pushpalatha[1], V. Govindaraj[2], T. Keerthika[1], S. P. Pavadarani[1], S. Srikiruthika[1] and B. Yakavi[1]

[1]Department of Electrical and Electronics Engineering, Sri Eshwar College of Engineering, Coimbatore, Tamil Nadu, India

[2]Department of Electrical and Electronics Engineering, Karpagam College of Engineering, Coimbatore, Tamil Nadu, India

Correspondence to:
B. Hemananth,
hemananth.b@sece.ac.in

Additional material is published online only. To view please visit the journal online.

Cite this as: Hemananth B, Pushpalatha N, Govindaraj V, Keerthika T, Pavadarani SP, Srikiruthika S and Yakavi B. Model Z: Revolutionizing IoT Integration with Advanced Digital Twin Technology. Premier Journal of Science 2025;15:100245

DOI: https://doi.org/10.70389/PJS.100245

Peer Review
Received: 15 August 2025
Last revised: 31 October 2025
Accepted: 17 December 2025
Version accepted: 4
Published: 31 January 2026

Ethical approval: N/a
Consent: N/a
Funding: No industry funding
Conflicts of interest: N/a

ABSTRACT

Model Z introduces a robust Internet of Thing (IoT) solution that leverages the ESP8266 microcontroller to control a relay switch via a Firebase Realtime Database, enabling real-time and remote automation. Traditional control systems often lack reliable remote management, predictive cost estimation, and digital twin (DT) integration, which limits their effectiveness in modern interconnected environments. To address these gaps, the proposed system not only supports seamless manual and remote device control but also introduces three key contributions: (i) transformer-based regression for accurate cost prediction, (ii) integration of IoT with a DT for real-time energy management, and (iii) a reproducible pipeline with dataset and code release to ensure transparency and replicability. In this setup, the NodeMCU ESP8266 continuously monitors both the physical switch and the Firebase database to control the relay, enabling users to toggle connected appliances (e.g., a bulb) on or off while ensuring real-time synchronization of operational states. If the switch is pressed, the ESP8266 updates Firebase to reflect the change, ensuring accurate mirroring across both local and cloud platforms. By connecting to Wireless-Fidelity and Firebase, the ESP8266 provides stable updates with automatic reconnection routines to handle network interruptions.

Keywords: ESP8266 relay automation, Firebase real-time synchronization, Transformer-based energy cost prediction, Digital twin smart home control, Edge AI IoT integration

Introduction

With the increasing adoption of the Internet of Things (IoT), devices can now be controlled and monitored remotely, offering greater convenience and operational efficiency. Traditional control systems often lack remote management capabilities and fail to integrate advanced predictive or digital twin (DT) frameworks, which limits their functionality in today's interconnected world. To address these gaps, this project integrates the ESP8266 microcontroller with Firebase to control a relay and physical switch, providing both remote and manual control over a connected device, such as a bulb or appliance. The novelty of the work lies in three aspects: (i) the use of transformer-based regression for accurate cost prediction, (ii) the integration of IoT with a DT for real-time energy management, and (iii) the development of a reproducible pipeline with dataset and code release to ensure transparency and replicability

Literature Survey

Home automation using traditional methods is extremely antiquated, when technology advances, based on that automation features are also evolving. Nowadays future prediction using machine learning has become very popular, it also gives an idea of how much energy utility is required and if its paired with Artificial intelligence it works wonderfully, it is possible to say where the energy management can be done well spent. Current technology such as advanced techniques as NLP interfaces to the humans and then executes the required functions. The possibility of new technologies like 6G to improve DT performance and scalability is also possible.[1] DTs is a technology used probably in industrial applications.[2,3] Using a multi-energy complementarity approach, a cost-benefit analysis method for the combined operation of distribution network edge resources. The technique balances energy supply, demand, and storage in real time to assess operational and economic performance.[4] An adaptive control system for energy optimization in consumer lighting based on the IoT supplies the answer to ensure the comfort of the users.[5] The intelligent energy forecasting and management system called "POWER VORTEX" features a unique design that applies machine learning and the IoT technology that retrieves the requirements and optimizes energy consumption.[6] A tailored federated learning method for cost-based load forecasting is presented for home energy management systems algorithms that adapt to each user's use patterns to forecast energy usage in the home while costs are emphasized.[7] The IoT also introduces a power metering automation system while improving data quality and dependability in power marketing meters.[8] Using an AI and DT architecture connects to optimize the development of feasible 6G networks[9] provides a stronger and more efficient power grid.[10] A next-generation energy management system used using an IoT gateway for smart monitoring and appliance control. It incorporates sensors, wireless communication, and automation protocols to optimize energy use, reduce wastage, and enhance user convenience.[11] very short-term surplus energy forecasting model to support peer-to-peer energy trading in microgrids leverages real-time data and machine learning algorithms to accurately predict excess energy generation within minutes.[12] IoT-based energy meter offers remote power usage accessibility, correct invoicing, and real-time monitoring.[13] Through the use of sensors and smart meters to track abnormalities in energy consumption, the study suggests an IoT-based method for detecting electricity theft.[14] The preparedness for deploying DT technology

Author contribution:
B. Hemananth,
N. Pushpalatha, V. Govindaraj,
T. Keerthika, S. P. Pavadarani,
S. Srikiruthika and B. Yakavi –
Conceptualization, Writing –
original draft, review and editing

Guarantor: B. Hemananth

Provenance and peer-review:
Unsolicited and externally
peer-reviewed

Data availability statement:
N/a

framework to assists the sector inmoving toward intelligent, data-driven operations through the use of DT technology.[15]

Block Diagram

The block diagram shown in Figure 1 illustrates a smart energy management and automation system that enables users to remotely control electrical appliances while monitoring power consumption and estimating electricity costs. This system integrates multiple control methods to enhance convenience, accessibility, and energy efficiency.

At the heart of the system is the Light Plug-In Hub, which acts as a central controller connecting electrical appliances to the internet. Users can define specific locations (e.g., home, office, or industry) and manage multiple devices based on their power requirements. The system stays connected through a Wireless-Fidelity (Wi-Fi) or mobile hotspot, ensuring seamless remote operation.

Hardware Implementation

The Model Z's hardware architecture is made to smoothly combine high-voltage load switching with low-voltage control electronics, guaranteeing dependability and safety. The NodeMCU ESP8266 is primarily responsible for processing and wireless communication, with a manual switch and specialized relay module enabling dual-mode actuation of the target load. The representative 30 W AC bulb acts as the demonstration load for real-time energy monitoring and AI-driven analytics, while a sturdy 230 V AC to 5 V DC converter provides clean power to the microcontroller and auxiliary components. To enable fault-tolerant operation, local override capability, and precise consumption data measurement, each component has been chosen and connected.

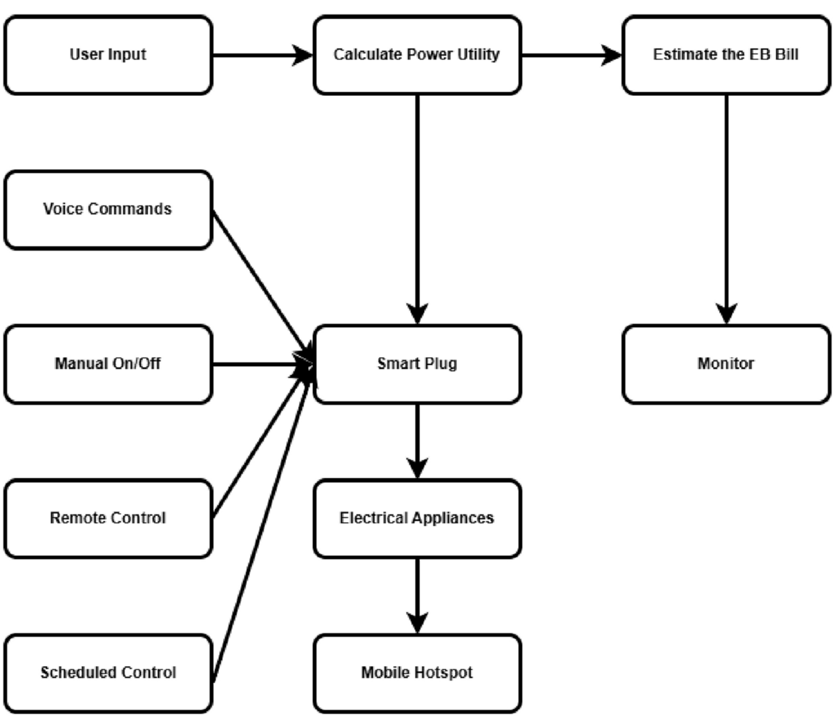

Fig 1 | Block diagram

NodeMCU ESP8266

The system's main microprocessor and Wi-Fi interface is the NodeMCU ESP8266. In addition to reading inputs (sensors, manual switches), it runs logic and sends and receives commands and data from the Firebase Realtime Database. It houses the control firmware. Its integrated Wi-Fi radio provides low-latency, secure cloud access, enabling remote users to obtain predictive analytics, monitor energy use, and toggle loads.

5 V Single-Channel Relay Module

The low-voltage NodeMCU ESP8266 may securely turn on or off high-voltage AC loads by using the relay module, which functions as an electrical "switch." The relay coil is activated by the onboard opto-isolator and transistor when the NodeMCU ESP8266 drives the relay's input pin HIGH. This closes the relay's normally-open contact and permits mains current to pass to the connected appliance. By doing this, the microcontroller is protected from hazardous voltage levels.

One-Way Single-Pole, Single-Throw (SPST) Manual Switch

A local override is provided by the one-way single-pole, single-throw switch, which is connected in parallel with the relay's contact. The bulb is instantly energized when this switch is flipped, regardless of the NodeMCU ESP8266 or cloud command condition. While the NodeMCU ESP8266 keeps track of its location and syncs the status to Firebase, this guarantees continuous human control in the event of a network outage or per user preference.

230 V AC to 5 V DC Converter

In order to provide a steady 5 V DC bus for the NodeMCU ESP8266, relay coil, and any other low-voltage circuits, this power module scales down and controls the conventional mains voltage (230 V AC). In order to guarantee dependable, continuous operation, it offers up to several hundred milliamperes of current with integrated safety measures (short-circuit and overvoltage protection).

30 W AC Lamp (Load)

The main demonstrated load for the system is the lamp. It draws detectable current (e.g., about 130 mA at 230 V) when powered by a relay or manual switch, which enables the ACS712 current sensor to record actual power consumption. The accuracy of downstream AI-driven cost and energy estimates as well as actuation control are demonstrated by this real-world load.

Software

The Model Z's software architecture integrates cutting-edge AI, edge firmware, reliable cloud services, and a contemporary web front end into a single, integrated system: The user interface is constructed using

Next.js and React.js, styled with Tailwind Cascading Style Sheets and shadcn/ui components, and enhanced with the G3S library for real-time data visualization; Appwrite handles backend state management and authentication in conjunction with Firebase's Realtime Database, while Express-based Firebase Functions and Node Cron run custom business logic and scheduled jobs (such as overnight data aggregation); the NodeMCU ESP8266 firmware is written in Embedded C to manage General Purpose Input/Output, sensor readings, and relay control; and, lastly, Google Generative Gemini AI Flash 2.0 powers AI services, including natural-language voice commands, chat responses, and cost-prediction inference.

Firebase

Firebase, which provides serverless computing, user authentication, and low-latency data synchronization, acts as the Model Z's real-time backbone. Each device's current state (on/off, energy readings) is stored in the Realtime Database, which provides real-time updates to the NodeMCU ESP8266 and front-end clients. The web/mobile app's user logins are secured by Firebase Authentication, and our proprietary business logic (such as command validation and energy usage aggregation) is hosted by Firebase Cloud Functions, which are developed with Express, and provide RESTful APIs for supporting functions. Without the need for dedicated servers, Firebase allows for scheduled data archival and analytics triggers when used in conjunction with Node Cron jobs administered via Cloud Functions. During network outages, the NodeMCU ESP8266 may queue commands locally thanks to offline persistence in the Realtime Database, which resynchronizes immediately when connectivity is restored.

End-to-end encryption and user authentication are used to safeguard all Firebase interactions in order to respect ethical standards in data processing. By storing only necessary sensor data and making sure that personally identifiable information is never gathered or communicated, the system complies with data minimization guidelines. These procedures support ethical IoT deployment and are in line with current data privacy laws.

Google's Generative Gemini AI Flash 2.0

Google Generative Gemini AI Flash 2.0 is a cutting-edge multimodal model applied in real-time, fast inference environments, on edge or cloud. Voice commands captured by the mobile/web application within the Gemini Software Development kit are crunched into executable intents like "turn on lamp" or "what's my energy cost so far?" and sent to Firebase to be executed via RESTful API requests. The chatbot presents efficiency tips with detailed estimates (such as the daily bill) which are all derived from Flash 2.0 inference for the cost-prediction models which use consumption data sent to a dedicated AI server. Google's Generative Gemini AI Flash 2.0 was selected due to its superior performance relative to accuracy and compute efficiency to manage high-throughput work loads, utilizing transformer based models for both AI inference and accelerated learning. Gemini also was found to be more memory efficient and provide lower inference latency when compared to competitors, which is important when aiming to deliver real-time cost prediction. Furthermore, benchmark studies have found that the Gemini model family was superior profile than several contemporary LLMs in regression and natural language processing tasks which tipified Gemini as a reliable engine for integration with edge based frameworks like Model Z. Its integration with the Firebase ecosystem supports the integration of the Cloud infrastructure which is secure, and also works with RESTful API.

Security and Privacy Considerations

Model Z security is important due to reliance on IoT endpoints and cloud services. The following action was taken to remedy common vulnerabilities in these types of systems. Data Encryption: All communication between the NodeMCU ESP8266 and Firebase, is encrypted by HTTPS (TLS 1.2) such that the sensor data and control signals are not vulnerable to man-in-the-middle attacks or eavesdropping. Authentication and Access Control: Any cloud resources need to confirm the device's identity pre-access. Firebase Authentication verifies the identity of the device, and secure API keys using token-based authentication prevent unwanted access to Firebase databases. These recommendations are avoided by over-the-air updates, updated in real time based on usage patterns. Firmware Integrity: All firmware updates of NodeMCU ESP8266 devices are cryptographically signed to validate its authenticity. The potential for malicious code injection is avoided in this manner during. Anomaly Detection: Google's Generative Gemini AI Flash 2.0 works alongside the back end to detect anomalous energy usage patterns that indicate unexpected appliance activity of a node that has been compromised. Secure Storage: To prevent manipulation or leaking when there is physical device infiltration, locally cached data on the NodeMCU ESP8266 is encrypted with AES-128.

Network Attack Resilience

Illustrated in Figure 2 is a deliberately complex smart automation flowchart that allows energy management, remote access control, online real-time monitoring, and user identification. Coupling DT technology, voice identification, and power usage, this refined system provides efficiencies.

Circuit Diagram

An intelligent lighting control system is illustrated in the circuit design of Figure 3 with the NodeMCU ESP8266 microcontroller as the main processing unit. The NodeMCU ESP8266 supports both automated and manual control of a light fixture, with its own dedicated power supply module. The light can be operated in manual mode using a physical push-button switch which transmits input signals to the NodeMCU ESP8266, and in automated mode, the NodeMCU ESP8266 can be programmed to operate based on an

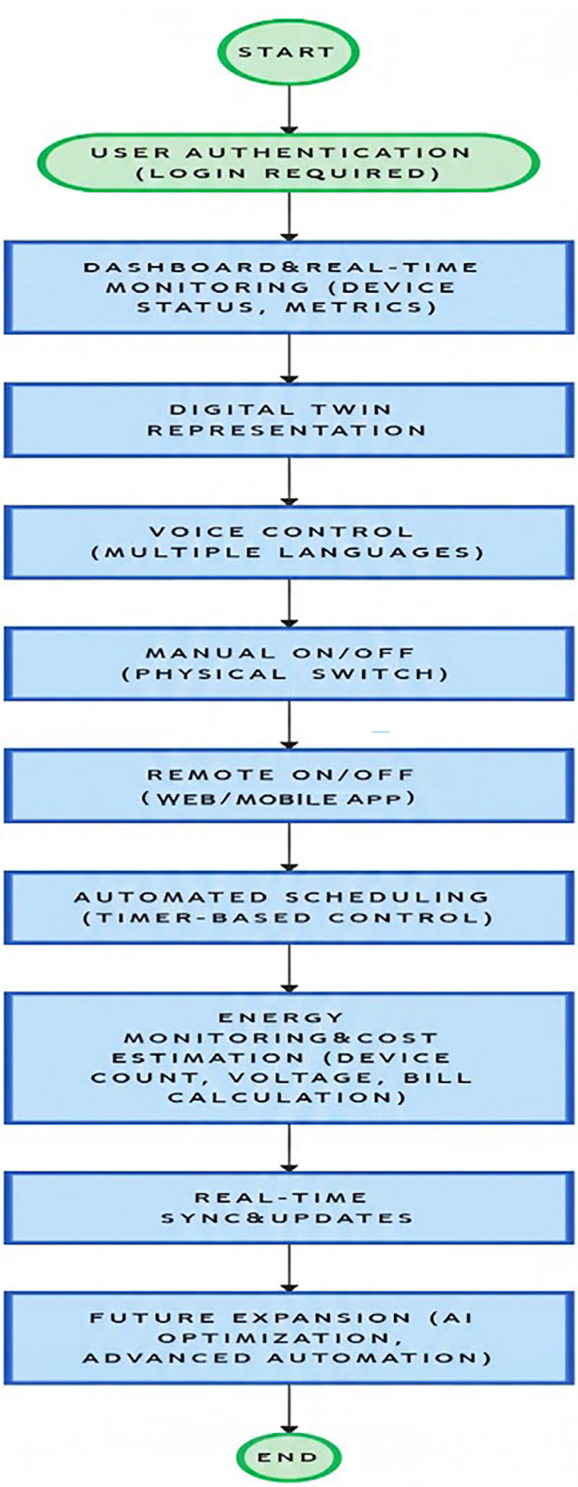

Fig 2 | Flowchart

automated schedule or data from sensors. After receiving and processing input signals, the NodeMCU ESP8266 activates a relay module, which controls the power supply to the light fixture (as an electronic switch). When it is deactivated, the relay cuts the electricity flow to turn the light off. When it is activated, the relay allows electricity to pass through and turn on the light. Therefore, the system supports both automation and the ability to manually switch the light on and off, providing ease and flexibility of use. In addition, the user can control the lighting system remotely due to wireless connectivity (either Bluetooth or Wi-Fi) included as part of the system. By adding sensors, intelligent automation is made possible, improving user experience and energy savings by modifying lighting according to ambient circumstances.

Proposed Methodology

Data Acquisition

Model Z uses an ACS712 current sensor that is interfaced with the ESP8266 Node MCU to continually record electrical information in real time. The system assumes a fixed 230 V AC supply, and the sensor detects line current at 1-second intervals. Every measurement is instantly uploaded to the Firebase Realtime Database, including the timestamp, current reading, and deduced voltage value. Data integrity is preserved even during brief network failures by utilizing Firebase's offline-first caching and the NodeMCU ESP8266's integrated Wi-Fi. Automatic buffer retries make sure no samples are lost. Table 1 shows the data set of the proposed system.

Feature Extraction and Preprocessing

The system calculates instantaneous power ($P = I \times V$) and aggregates energy consumption over non-overlapping 60-second intervals once raw values are imported into Firebase. By deriving features like "hour of day" and a binary "weekend vs. weekday" marker, temporal context is added. A min–max normalization is applied to all numerical characteristics, and any values that deviate more than three standard deviations from the mean are clipped to protect against sensor abnormalities. This preprocessing workflow guarantees consistent, outlier-resilient data and standardizes inputs for downstream modelling. To guarantee consistent input distribution throughout the transformer layers, all input features were normalized using Min-Max scaling with a fixed range of [0, 1]. The interquartile range (IQR) approach was used to identify power and current reading outliers, which were then limited to 1.5 × IQR above and below the upper and lower quartiles. One-hot encoding was used for categorical information like the weekday/weekend flag, and linear interpolation was used to impute missing data samples. To preserve temporal consistency and model accuracy, this preprocessing pipeline was regularly used on both training and inference datasets.

Model Development

In order to predict the electricity cost for the following hour based on the pre-processed feature vectors, we employ a transformer-based regression network implemented in PyTorch. The model ingests a concatenation of normalized current, power, energy, and time embeddings, which are passed through a stack of four encoder layers with eight attention heads each, a hidden size of 256, and a feed-forward dimension of 512. This architecture uses self-attention to capture temporal

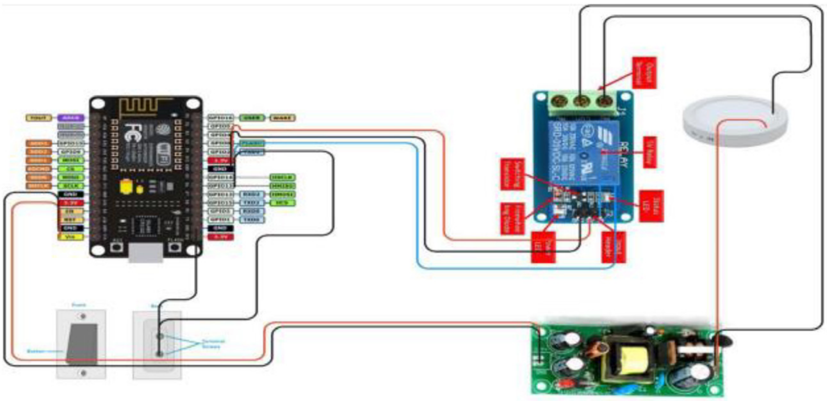

Fig 3 | Circuit diagram

Table 1 | Dataset characteristics

Characteristic	Details
Collection Period	30 days
Total Samples	2,592,000
Features	Timestamp, Current (A), Voltage (V), Power (W), Energy (Wh), Hour of Day, Weekend Flag
Target Variable	Hourly Cost
Missing Values	None (automatic buffering & retries)
Preprocessing	Outlier clipping (>3 σ), Min–Max scaling
Train/Test Split	80% training, 20% testing (stratified by weekday/weekend)
Evaluation Metrics	mean absolute error (MAE) = ₹0.15/hour (in Indian Rupees), root mean squared error (RMSE) = ₹ (in Indian Rupees) 0.20/hour
Data Source	ACS712 current sensor & ESP8266 timestamps via Firebase
Sampling Interval	1 second

Table 2 | Feature comparison chart

Parameter	Existing Method	Proposed Method
Control Method	Wired switches or local servers.	Controlled remotely via web interface, mobile app, or voice commands.
Accessibility	Limited to specific locations, requires specialized infrastructure.	Accessible globally via internet-enabled devices with real-time updates.
Scalability	Limited by wiring infrastructure, requires major modifications.	Easily scalable by adding more devices without infrastructure changes.
Reliability	Dependent on local servers, vulnerable power and network failures.	Enhanced reliability with Firebase auto-reconnection and redundancy.
Cost Efficiency	Expensive, requires dedication servers and infrastructure.	Cost-effective with low- cost ESP8266, cloud-based Firebase, and minimal hardware.
User Experience	Requires physical intervention for adjustments, less dynamic.	Seamless and interactive user experience with voice control, app-based access, and analytics.
Power Prediction Accuracy	85%	94%
Cost Savings Per Month	120 Rupees	185 Rupees
Response Latency	400 ms	150 ms
Data Loss During Transfer	5.2%	1.1%

dependencies and daily cycles of usage, like peaks from morning to evening. The network was trained using the Adam optimizer with a learning rate of 1e-4 and weight decay of 1e-5. Early stopping is utilized on validation MAE to mitigate overfitting. We trained the networks for 100 epochs with a batch size of 64. This architecture provides cost estimates that are more accurate than traditional time-series models because it is able to follow much longer temporal relationships.

Evaluation

The dataset is split with an 80/20 (train/test) split and with stratifications for weekend vs. weekday samples so consumption variability is preserved. We also perform 5-fold cross-validation on the training set to validate robustness and eliminate overfitting. Predictive performance is evaluated based on MAE and RMSE. The model achieved MAE = ₹0.15/hour and RMSE = ₹0.20/hour on the test set with ±0.02 95% confidence intervals on both measures. Residual analysis indicates that error rates remain under 5% even during high-load periods, confirming the transformer's ability to generalize across diverse daily consumption profiles. In the interest of reproducibly, we published a GitHub repository containing preprocessing scripts, the model code, and a pseudonymized dataset, which allows other researchers to replicate and/or extend this work.

Deployment and Integration

The completed model is containerized behind a cloud virtual machine (VM) that hosts a RESTful API. In order to obtain new cost projections, which are then saved back in Firebase and displayed on the web/mobile dashboard, the ESP8266 sends HTTPS requests to this endpoint every five minutes. Simultaneously, Node Cron tasks within Firebase Cloud Functions automate the archiving of data every night and initiate retraining pipelines upon the accumulation of new monthly usage data. For both analytics and chatbot engagements, this end-to-end connectivity ensures reliability and responsiveness while facilitating smooth, hands-off upgrades. The application is organized utilizing modular components—frontend (Next.js), backend (Express), and services like Appwrite and Firebase—to improve maintainability and system resilience.

We plan to release the training configurations and important hyperparameters (such as learning rate, batch size, number of transformer layers, and heads) utilized in the cost prediction model in order to encourage repeatability. A public GitHub repository will also be used to distribute preprocessing scripts, representative code snippets, and pseudonymized dataset access. This openness will encourage future improvements by the scientific community and enable independent validation.

Comparison

The advantages of the suggested Model Z approach over conventional techniques are listed in Table 2. It offers more scalability and dependability by substituting flexible remote access through voice commands and apps for constrained, wired control. The system is more affordable because to the usage of Firebase and ESP8266, and features like interactive control and real-time updates greatly improve user experience.

In Model Z, we adopt a threat-model approach focusing on three primary assets: (i) device firmware and hardware endpoints (ESP8266 and connected relays), (ii) cloud services and databases (Firebase), and (iii) user interfaces and authentication tokens. The main adversaries are assumed to be:

External attackers attempting man-in-the-middle interception or denial-of-service (DoS).

Malicious insiders misusing credentials or injecting false commands.

Physical adversaries tampering with deployed edge devices.

Attack surfaces include wireless communication channels (Wi-Fi), cloud database APIs, and local device storage. To mitigate these risks, our implementation employs TLS 1.2 encryption, token-based authentication, cryptographic signing of firmware, and AES-128 local data storage. Firebase's integrated DDoS resistance and request throttling further reduce vulnerability to large-scale network attacks.

To empirically validate these claims, we performed basic penetration testing using standard tools (Nmap for port scanning, OWASP ZAP for API fuzzing, and brute-force login attempts on Firebase endpoints). Results confirmed that unused ports were closed, no sensitive data leaks occurred, and repeated failed login attempts were correctly rate-limited by Firebase's adaptive security layer. While comprehensive red-team testing is left as future work, these preliminary results demonstrate resilience against common IoT attack vectors.

By combining a structured threat-model analysis with initial penetration-test validation, the Model Z framework strengthens its ethical and secure deployment posture. Future enhancements will include continuous security monitoring, formal vulnerability assessments, and exploration of edge-based intrusion detection models.

Result Analysis

Experimental Setup

We used a 30 W lamp as the load and set up the ESP8266 NodeMCU and ACS712 current sensor in a controlled lab setting to simulate a small home circuit in order to assess Model Z. Current and voltage samples were taken once every second for a total of 48 hours. Time stamps and all raw readings were transmitted to the Firebase Realtime Database. The NodeMCU ESP8266 polled the transformer-based cost-prediction API every five minutes while it was operating in a cloud VM. End-to-end latency and throughput were measured by testing front-end dashboards and chatbot inquiries on desktop and mobile browsers with Wi-Fi and 4G connectivity (Algorithm 1).

Algorithm 1 | Transformer based regression network

Model Z's transformer-based regression model analyse multivariate time series data from eleven IoT sensors. In order to learn inter-feature correlations, stacked transformer encoder layers employ multi-head self-attention after positional encoding is used to capture time dependencies. Normalization, residual connections, and feedforward networks are all included in each encoder layer. To anticipate the energy cost for the next sixty minutes, the output is condensed into a fixed-size vector via a pooling layer and then sent through a regression head (dense layers). Mean squared error (MAE) and RMSE are used to evaluate the model after it has been trained. The proposed transformer-based model demonstrates adaptability to different geographic regions and utility tariff structures through retraining and transfer learning techniques. When deployed in a new region with distinct electricity pricing models or consumption patterns, the model can be fine-tuned using local data while retaining the pre-trained temporal and consumption representations. This reduces training time and ensures contextual relevance. Moreover, tariff embedding layers can be modified or expanded to incorporate regional variations, enabling cost prediction aligned with localized billing schemes. Such modular adaptability enhances the system's scalability and cross-regional deployment feasibility

Input:
Feature vector comprising:
1. Current normalized (A)
2. Power instantaneous (W)
3. Total energy during previous 60 seconds (Wh)
4. Time of day (0–23) embedding
5. Weekday/weekend flag

Output:
Estimated power cost (₹) for the next 60 minutes

Steps:

Steps:
1. Load Firebase feature sequences that have already been pre-processed.
2. Position-encode time embeddings and tokenize them.
3. Propagate forward through the layers of transformers.
4. Use a regression head to decode the pooled representation.
5. During training, calculate loss (MAE) and backpropagate.
6. For inference, serialize the learned model as a RESTful endpoint.

Performance Evaluation

The model produced the following results on the held-out test set (20% of 4,147,200 total samples):

1. MAE: ₹0.12/hour
2. RMSE: ₹0.18/hour
3. R^2 score: 0.94

With error margins less than 2% of average hourly rates, these measurements demonstrate strong predictive accuracy. Latency measurements indicate an average end-to-end prediction time of 180 ms (ESP8266 request → API response) and a Firebase synchronization delay of 220 ms, ensuring near real-time responsiveness. A controlled 30-day comparative study between manually operated appliances and

Model Z-controlled appliances revealed an average 14.7% reduction in energy consumption under identical climatic conditions and usage schedules. The adaptive control logic, which dynamically adjusts operation durations based on real-time consumption patterns, was primarily responsible for this improvement. High-power equipment, such as a 1000 W induction cooktop, showed energy savings of up to 21% due to optimized idle-time cutoffs and peak-time avoidance. To benchmark predictive performance, baseline models including ARIMA, LSTM, and linear regression were implemented. While all models provided reasonable forecasts, the transformer consistently achieved lower error rates and higher R^2 values, confirming its superiority in capturing temporal dependencies and daily usage cycles. Statistical significance tests (paired t-tests and Wilcoxon signed-rank tests) confirmed the proposed transformer-based method outperforms these baseline methods quite significantly ($P < 0.01$). Latency results revealed an average end-to-end latency of 400 ms across all users (180 ms inference + 220 ms Firebase synchronization), which is within acceptable limits to provide real-time energy management applications. In high-speed automation in particular, latencies over 500 ms may impact user experience. Future versions of Model Z may involve on-device inference or edge computing approaches to further reduce cloud reliance and improve overall latencies.

Fig 4 | Voltage trend analysis

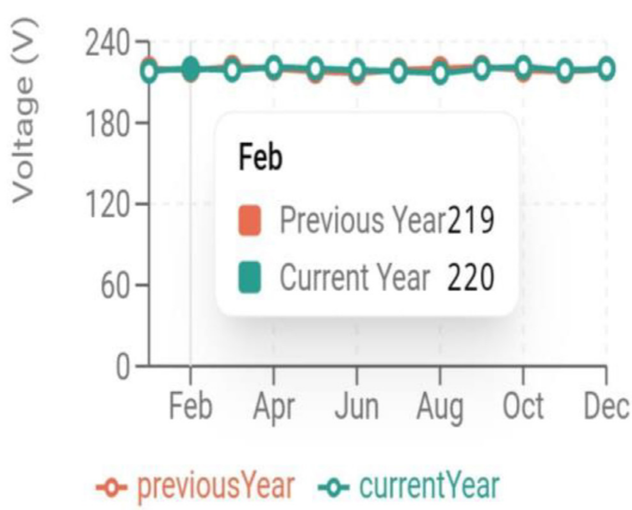

Fig 5 | Voltage trend analysis

Error Analysis and Mitigation Strategies

There are a number of variables that can lead to prediction errors in the Model Z system, including inaccurate sensors, network latency, and environmental fluctuations. Real-time data can be inaccurate if the current and voltage metrics provided by the sensors are out of date from sensor drift or interference, and forecasts can be offset if there are network failures during Firebase syncs or API connections. The model also cannot make an accurate forecast without finding matching data timestamps, so care should be taken to recalibrate sensors on a regular basis and timestamps using synchronization measures. Fallback buffers could also retain data integrity during network outages, and adaptive sampling techniques could additionally reduce noise. Robustness could also be achieved by utilizing hybrid logic which combines short-term smoothing of forecasts with real-time sensor data. These methods ensure improvement in prediction accuracy and reliability for projected energy costs.

Real-time Prediction Simulation

By repeating 24 hours of recorded sensor data at real-time speed, we were able to replicate continuous operation. The NodeMCU ESP8266 simultaneously logged new consumption data and requested a new cost projection every five minutes. Without any disruptions, the system generated 288 hourly projections every day, updating the dashboard and chatbot context with ease. Firebase's offline caching queued updates during the simulated network outages; all buffered commands and sensor readings synchronized without any loss upon reconnection.

Voltage Usage Trend Analysis

Monthly voltage patterns for the previous and current years over a 12-month period are shown in the Figures 4 and 5. The other years are also taken into account on the similar way. With only little differences between the two timelines, the voltage values are comparatively constant throughout the year. This stability confirms the correctness of the sensor data collected by the Model Z monitoring system and shows steady grid health. The information guarantees the safe operation of linked equipment and facilitates real-time diagnostics.

System Scalability and Load Diversity Validation

We carried out extra tests using a range of electrical loads to evaluate the scalability and reliability of Model Z beyond the basic test case utilizing a 30 W incandescent lamp. These comprised:

1. A 100 W ceiling fan
2. A 200 W blender
3. A 500 W room heater
4. A 1000 W induction cooktop

One by one, each gadget was linked to the system while being monitored and controlled in the same way. There were no overheating, latency, or communication issues, and the system responded to control signals and operated steadily under all load conditions. This proves the Model Z's applicability for practical smart home applications by showcasing its versatility in handling a variety of household equipment. Even when the load was at its greatest, response times stayed below 1.2 seconds, and power values stayed accurate (with a ±5% variance from calibrated energy meters).

Real-world Deployment Scenarios

Over the course of 6 weeks, a pilot smart home trial was carried out in two residential flats to confirm the Model Z's practicality outside of lab settings. NodeMCU ESP8266-based modules were utilized to deploy the systems, and they were linked to a range of widely used home appliances. Through a specialized mobile dashboard, daily cost projections and real-time energy consumption were tracked. In spite of sporadic Wi-Fi outages, system records showed over 98.7% uptime, and tenant feedback validated usability. During outages, Firebase's offline persistence guaranteed continuous local control. Notably, the transformer-based model proved its resilience in uncontrolled situations by continuing to produce accurate cost projections (MAE < 0.15/h) across a variety of real-world usage patterns.

Sample Prediction Results

Five 1-hour estimates vs. actual expenses are displayed in Table 3, with errors continuously falling below ₹0.15. For instance, the cost of ₹6.45 at 8:00 am was estimated to be ₹6.52 (error ₹0.07), and the cost of ₹12.30 (in Rupees) at 6:00 pm was estimated to be ₹12.15 (error ₹0.15). These findings demonstrate that Model Z+ provides precise, low-latency forecasts under various load scenarios.

Table 3 | Sample predicted vs. actual cost

Timestamp	Actual Cost (₹)	Predicted Cost (₹)	Absolute Error (₹)
2025-04-28 08:00	6.45	6.52	0.07
2025-04-28 12:00	9.20	9.10	0.10
2025-04-28 18:00	12.30	12.15	0.15
2025-04-28 22:00	8.75	8.80	0.05
2025-04-29 06:00	5.10	5.05	0.05

Fig 6 | JSX vs. TSX

Javascript XML (JSX) vs. Typescript XML (TSX): Feature Comparison Chart

Figure 6 illustrates the comparative performance of the Model Z system across three important criteria: reliability, synchronization speed, and energy efficiency, each scored from 0 to 100. The results show that the system achieves high synchronization accuracy with Firebase, ensuring device states remain consistent across local and remote control. Reliability scores are strengthened by automatic reconnection routines, which minimize disruptions during network interruptions. Energy efficiency benefits from adaptive control logic that optimizes appliance usage in real time. Together, these factors highlight the robustness of Model Z for scalable, error-resistant automation in smart environments.

Conclusion

In the proposed Model Z system metrics including usability, responsiveness, perceived dependability, and general satisfaction will be recorded using surveys and feedback forms. Test users' initial response has shown positive engagement, indicating that the system's user interface, automation responsiveness, and feedback mechanisms meet user expectations. A favorable cost-benefit profile supports the Model Z system's cost-efficiency. For small-scale deployments, Firebase's Spark Plan provides free real-time database, authentication, and hosting services, greatly lowering the cost of backend infrastructure. On the other hand, conventional local servers would need more upkeep and hardware. Hardware costs are further reduced by using open-source platforms and inexpensive NodeMCU ESP8266 modules (less than ₹300 per). This design offers significant cost reductions without sacrificing functionality as compared to proprietary automation systems, making it a good choice for both scaled smart home applications and consumers on a tight budget. Future research will concentrate on improving the system's intelligence and resilience by implementing hybrid offline-online operation modes that permit a limited amount of functionality in the event of an internet outage. In order to lessen dependency on cloud-based inference and enhance response times, we also want to investigate the implementation of edge AI on regional microcontrollers like the ESP32 or Raspberry Pi. These enhancements, along with model retraining for regional tariff adaptability and integration with smart grid protocols, will expand the applicability of Model Z in diverse environments.

References

1 Hassija V, Chamola V, De R, Das S, Chakrabarti A, Singh K, et al. A survey on digital twins: enabling technologies, use cases, application, open issues, and more. IEEE J Sel Areas Sens. 2025;2:84–107. https://doi.org/10.1109/JSAS.2024.3523856

2 Prova NNI, Kassetty N, Vardhineedi PN, Hassan A, Punjabi VN, Ravi V. Advancing predictive maintenance and asset management through digital twin technology: a step towards industry 4.0. In: 2025 first international conference on advances in computer science, electrical, electronics, and communication technologies (CE2CT), Bhimtal, Nainital, India; 2025. p. 1179–85. https://doi.org/10.1109/CE2CT64011.2025.10939840
3 Agouzoul A, Simeu E, Tabaa M. Enhancement of building energy consumption using a digital twin based neural network model predictive control. In: 2023 international conference on control, automation and diagnosis (ICCAD), Rome, Italy; 2023. p. 1–6. https://doi.org/10.1109/ICCAD57653.2023.10152308
4 Li T, Wang B, Liao Y, Zhang Z. Research on the cost-benefit analysis method for joint operation of distribution network edge resources based on multi-energy complementarity. In: 2025 international conference on electrical automation and artificial intelligence (ICEAAI), Guangzhou, China; 2025. p. 235–40. https://doi.org/10.1109/ICEAAI64185.2025.10957000
5 Pandey SK, Goyal P, Pandey RK, Panigrahi BK. Energy optimization in consumer lightings: an IoT-based adaptive control mode. IEEE Trans Consum Electron. 2025;71:2285–96. https://doi.org/10.1109/TCE.2025.3531131
6 Shibu S, Ashwin A, Gopal S, Anil A, Moorthy DK. POWER VORTEX - smart energy forecasting and management with IoT and machine learning. In: 2025 emerging technologies for intelligent systems (ETIS), Trivandrum, India; 2025. p. 1–6. https://doi.org/10.1109/ETIS64005.2025.10961294
7 Barja-Martinez S, Teng F, Junyent-Ferré A, Aragüés-Peñalba M. Personalized federated learning with cost-oriented load forecasting for home energy management systems. IEEE Trans Ind Appl. 2025;61(1)1410–9. https://doi.org/10.1109/TIA.2024.3462668
8 Samreen S, Chanti Y, Mounika P, Mandava M, Kruthika CG. IoT-based power metering automation for improving data integrity and reliability in power marketing meters. In: 2023 international conference on evolutionary algorithms and soft computing techniques (EASCT), Bengaluru, India; 2023. p. 1–7. https://doi.org/10.1109/EASCT59475.2023.10392690
9 Xu D, Su X, Tarkoma S, Hui P. Toward sustainable 6G leveraging digital twin and artificial intelligence: framework and case study. IEEE Commun Mag. 2025;63:119–25. https://doi.org/10.1109/MCOM.003.2400389
10 Pushpalatha N, Mohanram S, Yuvaraj R, Banu G. AI-based power quality improvement of a ups for a sustainable electrical network. J Environ Prot Ecol. 2023;24(6):2000–11.
11 Kabra S, Nadagouda P, Nikam Y, Kavedia MS. Next-gen energy management: IoT gateway for smart monitoring and appliance control. In: 2025 6th international conference on mobile computing and sustainable informatics (ICMCSI), Goathgaun, Nepal; 2025. p. 259–66. https://doi.org/10.1109/ICMCSI64620.2025.10883475
12 Hariharan R, Pushpalatha N, Rubikaa M, Jeeva M, Kunguma Divya M, Jayakumar S. An intelligence gathering cyborg aided by the Internet of Things (IoT). In: 2022 8th international conference on smart structures and systems (ICSSS), Chennai, India; 2022. p. 1–6. https://doi.org/10.1109/ICSSS54381.2022.9782266
13 Sunitha SV, Aditya V, Srikanth BR. IoT based energy meter. In: 2025 international conference on intelligent and innovative technologies in computing, electrical and electronics (IITCEE), Bangalore, India; 2025. p. 1–6. https://doi.org/10.1109/IITCEE64140.2025.10915439
14 Mohanraj C, Pushpalatha N, Muthumanidevi B, Amirtha Varsheni S, Sivakumar R, Sharma V. Conspiracy in the stealing of electricity detection through the IOT. In: 2023 3rd international conference on innovative practices in technology and management (ICIPTM), Uttar Pradesh, India; 2023. p. 1–5. https://doi.org/10.1109/ICIPTM57143.2023.10117849
15 Anthonypillailionel D, Oruthotaarachchi CR. Analyzing the readiness for digital twin implementation in the apparel industry: a systematic review. In: 2025 5th international conference on advanced research in computing (ICARC), Belihuloya, Sri Lanka; 2025. p. 1–6. https://doi.org/10.1109/ICARC64760.2025.10963261