An Intelligent IoT-Enabled Smart Grid with Edge AI, Cloud-Assisted Features and Multilingual Consumer Support

Ruturaj Parab[1], Sphurti Asawa[2], Prasanna Pandharikar[3]
[1,2,3]*Electronics and Telecommunication Engineering, Sardar Patel Institute of Technology Mumbai, India*

***Abstract*—Modernizing electrical distribution networks re-quires accessible monitoring, timely analytics, and dependable communication between consumers and utility-side stakeholders. This paper presents an IoT-enabled smart energy monitoring prototype combining ESP32-based metering, machine-learning analysis, cloud-assisted data services, and multilingual consumer support. The sensing layer uses a PZEM-004T energy-metering module and an ACS712 current sensor. For historical demand forecasting, a LightGBM regressor is trained on the public PJM AEP hourly electricity-consumption dataset using calendar, lag, and rolling-statistic features. An Isolation Forest model is trained separately to identify unusual multivariate operating patterns. The supplied training workflow exports the LightGBM model and preprocessing parameters for subsequent embedded integration; execution of the exported model on the ESP32-S3 is treated as a deployment step requiring separate validation. MQTT over Wi-Fi and Firebase Realtime Database provide the communication and application data path for dashboards, role-based access, alerts, and reports. The consumer interface also supports seven-language dashboard and alert translation and an AI-assisted chatbot. The paper specifies the training configuration available from the supplied code and distinguishes it from hardware measurements and performance results that require recorded experiment outputs.**

***Index Terms*—Smart grid, Internet of Things, smart energy meter, ESP32, LightGBM, Isolation Forest, time-series forecasting, anomaly detection, Firebase, MQTT, multilingual interface.**

I. INTRODUCTION

The evolution of conventional electrical grids toward smart grids is driven by the need for better monitoring, efficient energy use, and timely response to changing demand. Smart meters and connected sensors provide granular measurements that support consumer awareness, utility monitoring, and demand-side management. IoT architectures connect these devices to local processing nodes and cloud platforms, enabling bidirectional communication between consumers and operators [1]– [3].

Cloud platforms simplify aggregation and remote access, but cloud-dependent decision making is exposed to network latency and connectivity interruptions. Edge computing complements the cloud by performing selected processing close to the data source. In this work, LightGBM is evaluated as a tree-ensemble forecasting method on engineered hourly demand features, while an Isolation Forest provides unsupervised anomaly scoring. The training scripts export model artifacts for a subsequent embedded-inference integration step.

This work implements an integrated prototype with four functions: (i) electrical parameter acquisition, (ii) edge-based energy prediction and anomaly analysis, (iii) cloud-assisted monitoring and reporting, and (iv) multilingual consumer interaction through a dashboard and chatbot.

The main contributions are:

- An ESP32-based metering arrangement using PZEM-004T and ACS712 sensing components.
- A machine-learning pipeline using LightGBM for next-hour demand forecasting and Isolation Forest for anomaly detection, with model artifacts exported for integration.
- A Wi-Fi/MQTT/Firebase data path supporting dashboard updates, access-controlled views, alerts, and reports.
- A consumer interface with seven-language translation and an AI-assisted support chatbot.

The rest of the paper is organized as follows. Section II reviews related work. Section III presents the system architecture, and Section IV describes the hardware and firmware. Section V covers edge forecasting and anomaly detection, and Section VI the cloud and consumer services. Section VII reports experimental results, Section VIII discusses limitations and future work, and Section IX concludes.

A. Problem Statement

Conventional monitoring architectures face challenges related to communication delay, limited predictive capability, interoperability, deployment cost, and the availability of consumer-oriented information. A system that depends entirely on a remote cloud service can also lose analytics when connectivity is interrupted. This project addresses these concerns by combining local measurement and inference with cloud-assisted services.

B. Research Objectives

The objectives are to:

1) Develop an IoT sensing and edge-processing architecture using ESP32-class controllers.
2) Implement next-hour demand forecasting using Light-GBM and evaluate a separate Isolation Forest anomaly detector.
3) Provide monitoring, visualization, reporting, and alert services through Firebase.
4) Support multilingual dashboard content and consumer interaction.
5) Evaluate the prototype using forecasting and system-

C. Scope and Limitations

The prototype targets residential and small-scale energy monitoring, embedded inference, and consumer-facing cloud services. It is not presented as a certified revenue meter, a utility-grade protection system, or a field-validated re-placement for SCADA. Sensor accuracy, model performance, network range, device memory, and reliability depend on installation and test conditions.

II. RELATED WORK

A. IoT Architectures for Smart Grids

Al-Turjman and Abujubbeh discuss IoT-enabled smart grids and the role of smart meters as edge devices that support real-time collection, bidirectional communication, and demand response [1]. Pau *et al.* describe a cloud-based smart-metering infrastructure for distribution-grid services and automation [2]. Together, these studies motivate an architecture in which meter data is collected at the edge and exposed to higher-level services.

B. Impact and Sustainability

Hossein Motlagh *et al.* identify IoT as an enabler of de-centralized energy systems, including predictive maintenance and peer-to-peer energy trading [3]. Bibri frames sensor-based analytics as a foundation for smart sustainable cities, linking energy monitoring to environmental goals [4].

C. Machine Learning for Forecasting and Anomaly Detection

Ke *et al.* introduced LightGBM, a gradient-boosting frame-work designed for efficient decision-tree learning [7]. Its tree-based structure is suitable for tabular features such as calendar variables, lagged demand, and rolling statistics. Liu *et al.* proposed Isolation Forest, an unsupervised method that identifies anomalies through shorter isolation paths in randomly constructed trees [8]. These methods are used here for dis-tinct tasks: one-step-ahead demand regression and multivariate anomaly scoring.

D. Security and Privacy

Large IoT deployments in critical infrastructure raise security concerns. Ullah *et al.* survey secure data aggregation and identify encryption, authentication, and secure routing as essential [5]; although their setting is healthcare, the findings apply to metering. Mocrii *et al.* identify privacy risks and de-vice vulnerabilities as major barriers in IoT-based smart homes [6]. These studies motivate authenticated access, database-level authorization, and secure transport in the present design.

Table I Architectural Characteristics of Cloud-Centric Monitoring and The Proposed Edge-Assisted Prototype (Qualitative).

Aspect	Cloud-centric	Proposed prototype
Forecast execution	Remote/cloud training workflow	LightGBM model exported for subsequent edge integration

Link interruption	Remote analytics typically unavailable	Local acquisition and inference continue; records buffered and forwarded later
Consumer access	Vendor-specific portal	Role-based PWA with translated UI and chatbot
Model representation	Depends on deployment	LightGBM text model and fitted scaler exported
Remote management	Varies by vendor	MQTT downlink commands and OTA updates

E. Research Gap

Existing work demonstrates the value of smart metering, cloud coordination, and embedded inference. This project contributes an integrated implementation that combines low-cost metering hardware, edge forecasting, a cloud dashboard, automated reports, multilingual support, and a consumer chat-bot. Table I summarizes the architectural differences from a cloud-centric design; it is a qualitative comparison, not a measured benchmark.

III. PROPOSED SYSTEM ARCHITECTURE

The system comprises sensing, edge-processing, cloud-service, and consumer-interface layers. Fig. 1 shows the logical data path.

A. Sensing Layer

The PZEM-004T provides primary electrical measurements: voltage, current, power, and accumulated energy. The ACS712 adds a second current-sensing channel used for current-related analysis. Because the two devices have different interfaces and error characteristics, their readings are calibrated and interpreted according to their respective specifications. Split-core current transformers can be used to observe bidirectional flow for net-metering scenarios with local generation, and galvanic isolation is included in the sensor circuitry for safe high-voltage measurement.

B. Edge-Processing Layer

The supplied machine-learning workflow is trained in Python and exports a LightGBM Booster text file together with its StandardScaler parameters. The forecasting features comprise hour, day of week, month, weekend indicator, 24 hourly demand lags, and six-hour rolling mean and standard deviation. The next-hour target is formed by shifting demand by one hour.

The supplied script does not itself demonstrate that the exported LightGBM model executes on the ESP32-S3. A compatible inference implementation, memory profiling, and on-device latency measurement are therefore required before claiming embedded execution.

TABLE II Technical Capabilities Matrix Highlighting the Unique Combination of Real-Time Processing, Edge Intelligence, And Offline Operation in The Proposed System.

Technical Capability	Traditional SCADA	Cloud-based IoT	Proposed System
Real-time Processing	Limited	Limited (Cloud Dependent)	Full (Local)
Predictive Analytics	No	Yes (Cloud-based)	Yes (Edge-based)
Demand Response	Manual	Semi-Automated	Fully Automated
Fault Detection	Yes	Yes	Yes (ML-based)
Energy Optimization	Basic	Advanced	Advanced (LSTM)
Grid Stability Monitoring	Yes	Yes	Yes
Renewable Energy Integration	No	Yes	Yes
Cybersecurity Features	Moderate (Proprietary)	High (SSL/TLS)	Very High (Local + Encryption)
Edge Intelligence	None	Limited (Cloud)	Full (ESP32-S3)
Offline Operation	No	No	Yes (Local Cache)
Wireless Communication Range	Limited	Very High (WiFi/4G/5G)	High (WiFi 2.4/5 GHz)
Network Bandwidth Required	High (Dedicated lines)	Moderate (1–10 Mbps)	Low (Optimized)

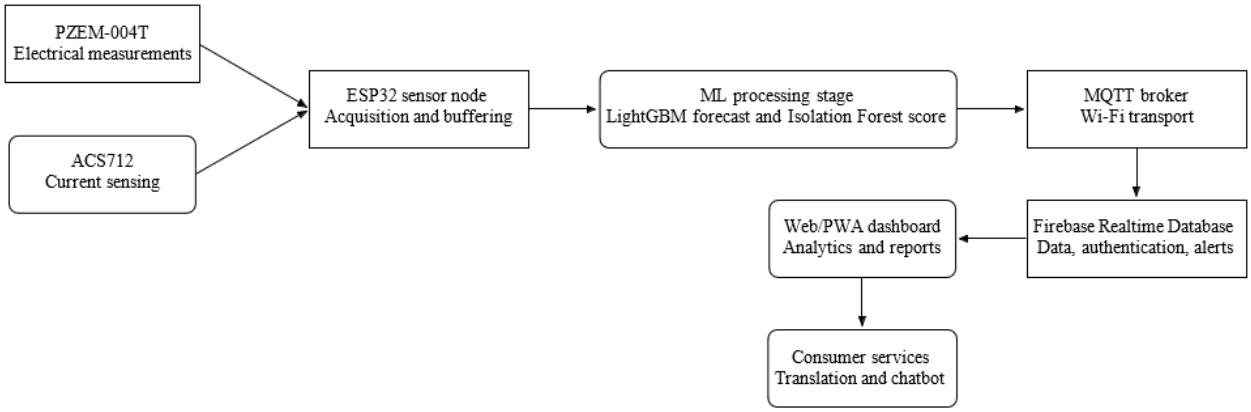


Fig. 1. Logical architecture of the implemented prototype.

C. Cloud and Application Layer

The edge node publishes processed measurements and event data through MQTT [9] over Wi-Fi. Firebase Real-time Database [10] stores synchronized records for dashboard access, and Firebase Authentication with database security rules [11] provides user authentication and access control. The dashboard presents current measurements, historical trends, prediction overlays, node status, and alerts.

IV. HARDWARE AND FIRMWARE IMPLEMENTATION

A. Controller and Measurement Components

Table III lists the principal components. An OLED displays provides local visibility of voltage, current, and cumulative energy, and a regulated supply provides the low-voltage rails.

TABLE III Principal Hardware Components and Their Roles.

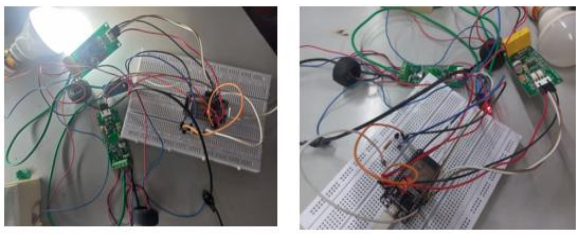
Component	Role
ESP32	Sensor-node controller: acquisition, buffering, Wi-Fi transport
ESP32-S3	Intended edge target; on-device LightGBM inference requires separate implementation and validation
PZEM-004T	Primary metering: voltage, current, power, energy
ACS712	Additional current-sensing channel
Split-core CT	Bidirectional current observation (net metering)
OLED display	Local readout of electrical parameters

B. Data Acquisition and Energy Calculation

The firmware acquires electrical measurements and attaches timestamps. When instantaneous power samples are available, energy over an interval is estimated by numerical integration:

$$E_{\mathrm{kWh}} = \frac{1}{3.6 \times 10^{6}} \sum_{i=1}^{N} P_i \Delta t_i \qquad (1)$$

where P_i is power in watts and Δt_i is the interval in seconds. If the PZEM-004T accumulated-energy register is used, that register is the meter's energy measurement and is not recomputed from the same readings without a defined reconciliation method.



(a) (b)

Fig. 2. Hardware prototype: (a) sensor node with ESP32 and PZEM-004T, (b) edge node and display assembly.

The PZEM-004T returns processed metering values over its communication interface; it is not equivalent to sampling raw voltage and current waveforms at 1 kHz. Waveform sampling with RMS and power-factor computation through the ACS712 and a separate voltage-sensing circuit is a distinct acquisition path and is treated as such. Calibration routines correct sensor offsets before values enter the analytics path.

C. Local Storage and Connectivity

Local flash storage (SPIFFS) buffers timestamped

records during connectivity interruptions, and buffered records are forwarded once the network returns. Processed data (voltage, current, power, energy) is published at 5 s intervals. MQTT carries telemetry and downlink commands on a per-device command topic, enabling remote configuration, over-the-air (OTA) firmware updates, and the push of new time-of-day tariff rates.

V. MACHINE-LEARNING FORECASTING AND ANOMALY DETECTION

A. Forecasting Pipeline

The forecasting script downloads the AEP hourly demand series from the PJM dataset repository and sorts observations chronologically. It creates four calendar features (hour, day of week, month, and weekend indicator), 24 lag features from the preceding hourly demand values, and two six-hour rolling statistics.

The target is the demand value one hour ahead. After rows containing missing lag or rolling values are removed, the feature matrix is split chronologically: the first 80% is used for training and the final 20% for testing. A StandardScaler is fitted on the training features and then applied to both partitions. The regressor is configured with 800 estimators, learning rate 0.05, maximum depth 10, 80 leaves, and random state 42.

The fitted Booster is exported in LightGBM text format, and the scaler is saved separately. Table V summarizes the configuration directly specified by the supplied code.

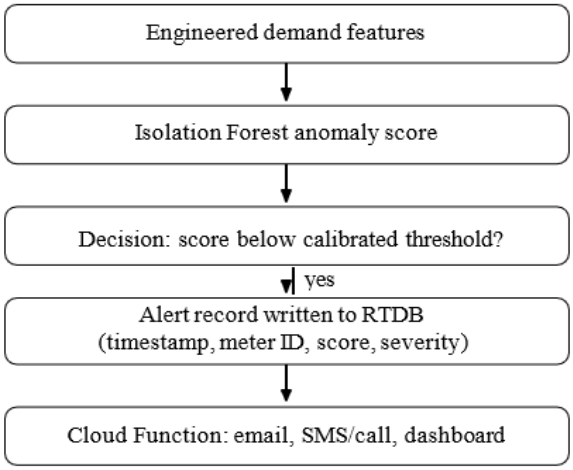


Fig. 3. Isolation Forest anomaly scoring and alert pipeline.

B. Evaluation Metrics

Forecast quality is measured with

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \quad (2)$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (3)$$

$$\mathrm{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (4)$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (5)$$

where y_i and $\hat{y}_i$ are actual and predicted values and $\bar{y}$ is the mean of the actual values. MAPE requires an explicit policy for zero-valued actual samples.

C. Anomaly Detection

The anomaly-detection script constructs a second feature set using calendar variables, selected demand lags (1, 2, 3, 6, 12, and 24 hours), and six-hour rolling mean and standard deviation.

It fits a StandardScaler and an Isolation Forest with 200 estimators, contamination parameter 0.001, and random state 42. The model assigns an anomaly score using the decision function and labels observations as normal (1) or anomalous (-1). The script derives a score threshold from the mean and standard deviation of scores labelled normal and exports this threshold with the scaler.

The supplied script evaluates the fitted detector on the same dataset used for fitting; its anomaly count is therefore descriptive and is not an independent estimate of detection accuracy. An anomaly flag indicates an unusual pattern, not proof of electricity theft.

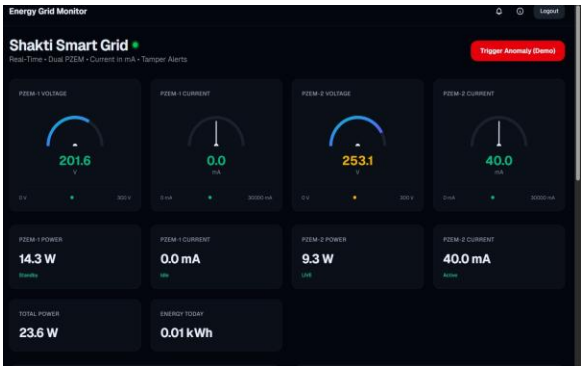
VI. CLOUD-ASSISTED FEATURES AND CONSUMER SERVICES

A. Firebase-Integrated Dashboard

The Firebase-backed dashboard enables multi-device access and role-based visibility.

The data pipeline is

Sensor node → ESP32-S3 edge node → MQTT broker→ Firebase RTDB → web/mobile dashboard.



(a)



(b)

Fig. 4. Firebase-based dashboard: (a) real-time monitoring view, (b) analytics and alert's view.

TABLE IV Roles and Permissions in The Cloud Dashboard.

Role	Permissions	Typical user
Administrator	Manage nodes, view system data, configure thresholds	Utility operator
Technician	View node health, analyze anomalies, support maintenance	Field engineer
Consumer	View own consumption, bills, and alerts	End user

Event-driven listeners update the interface when records change, without polling, and the implementation uses asynchronous writes with conflict-aware updates. The dashboard includes:

- Time-series graphs of voltage, current, active/reactive power, frequency, and power factor.
- Overlays of next-hour demand predictions produced by the LightGBM model, when forecast outputs are connected to the dashboard.
- Node-health metrics such as uptime, RSSI, and last-seen timestamps.
- Anomaly alerts with severity levels and contextual descriptions.

The front end is a Progressive Web App (PWA) installable on smartphones and desktops with offline-friendly behavior, built on the Firebase Web SDK with Chart.js or ApexCharts. Fig. 4 shows the interface.

B. Authentication and Role-Based Access

Firebase Authentication supports email–password, phone-based OTP, and OAuth sign-in. Role-based access control is implemented through custom claims and Realtime Database security rules [11], enforced at the database path level so that, for example, consumers can read only their own user-specific records (Table IV). Authorization is enforced by database rules or trusted server-side code, never by interface visibility alone.

C. Automated Report Generation

Automated reports turn consumption data into summaries for consumers and technicians, and can be generated daily, weekly, monthly, on demand, or when a significant anomaly is detected. Consumer reports include consumption plots, peak-demand periods, weekday–weekend patterns, energy-saving suggestions, and tariff-based bill estimates:

$$C = \sum_{i=1}^{N} E_i T(t_i) \qquad (6)$$

where E_i is energy consumed in interval i (kWh) and $T(t_i)$ is the applicable tariff per kWh. If power samples are used instead, the energy conversion and time units must be stated explicitly.

The workflow is:

1) Edge devices push aggregated usage statistics to Fire-base.
2) A Node.js Cloud Function is triggered by a schedule or event.
3) The function retrieves data and applies templates and summarization logic.
4) Reports are rendered in PDF/HTML and stored or emailed to users.

Technician reports add sensor-health indicators, node uptime, Wi-Fi RSSI trends, anomaly logs, suspected tampering events, and possible hardware faults.

D. Anomaly Alerts and Energy-Theft Indicators

The anomaly alerting workflow can use the Isolation Forest score and exported threshold to identify unusual feature combinations. The current training script demonstrates anomaly scoring and threshold export, but does not include labelled theft events, branch-current validation, or an independent evaluation of false-positive and false-negative rates. Therefore, an anomaly flag is treated as an investigation cue rather than proof of energy theft. When connected to the application backend, an event can be stored with a timestamp, meter identifier, anomaly score, and severity, after which configured notification services can be invoked.

E. Multilingual Support

The dashboard and alerts support English, Marathi, Hindi, Gujarati, Tamil, Telugu, and Kannada through a translation microservice based on the Google Translate API:

TranslatedText = Translate (Text, SourceLang, TargetLang (7)

Frequent interface phrases are cached in a key–value dictionary to reduce repeated requests. Alerts and PDF/HTML reports are generated in the consumer's chosen language, translating both template text and dynamic content. Billing-and safety-related translations should be reviewed by fluent speakers before deployment.

F. AI-Assisted Consumer Chatbot

The chatbot is embedded in the dashboard and answers questions about bills, usage trends, and navigation. It retrieves authorized Realtime Database values, predictions, and summaries to answer queries such as "Why is my bill high this month?" or "What is my peak usage time?", gives step-by-step troubleshooting for missing node data or Wi-Fi disconnection, and explains anomaly alerts with probable causes and suggested actions. Users can converse in their preferred language while the backend logic operates in English.

The conversation flow is

Dashboard UI → chatbot interface → Firebase Cloud Function→ NLP/business logic and authorized data retrieval → response.

Cloud Functions keep session context using session identifiers and short-term history.

Table V Experimental Configuration from The Supplied Python Scripts.

Item	Value
Sensor node	ESP32 with PZEM-004T and ACS712
Edge target	ESP32-S3; on-device inference not demonstrated by supplied scripts
Transport	MQTT over Wi-Fi (WPA2-PSK); hardware telemetry interval reported as 5 s
Dataset	PJM AEP hourly electricity-consumption dataset (AEP_hourly.csv)
Forecast target	Next-hour demand (one-step-ahead)
Forecast features	Calendar variables, 24 hourly lags, six-hour rolling mean and standard deviation
Forecast split	Chronological 80:20 train/test split
Forecast model	LightGBM: 800 estimators, learning rate 0.05, max depth 10, 80 leaves
Forecast scaling	StandardScaler fitted on training features
Anomaly model	Isolation Forest: 200 estimators, contamination 0.001
Anomaly features	Calendar variables, lags 1/2/3/6/12/24, six-hour rolling statistics
Anomaly scaling	StandardScaler
Quantization	Not used in supplied Python training scripts
Raw dataset rows	121,273 hourly records
Samples after feature engineering	121,248
Chronological training / test samples	96,998 / 24,250
Test duration	Not specified in supplied output
On-device inference time	Not measured in supplied scripts

The chatbot distinguishes measured readings from estimates and model predictions, and accesses only records authorized for the authenticated user.

VII. EXPERIMENTAL RESULTS AND DISCUSSION

A. Experimental Setup

Table V summarizes the configuration specified by the supplied Python scripts. The training scripts do not provide all hardware-deployment or test-duration measurements.

B. Forecasting and System Performance

The forecasting script was executed on the PJM AEP dataset. It processed 121,273 raw hourly records and retained 121,248 samples after feature engineering. With the chrono-logical 80:20 split, the training and test sets contained 96,998 and 24,250 samples, respectively. The measured next-hour forecasting results are reported in Table VI. The anomaly script does not evaluate against a labelled independent test set; therefore, its anomaly count must not be interpreted as a validated false-positive or detection rate.

TABLE VI Forecasting Evaluation Outputs Available from The Supplied Script.

Metric	Value
Forecast target	Next-hour AEP demand (MW)
MAE	175.8 MW
RMSE	234.9 MW
R^2 score	0.9908 (99.08%)
MAPE	Not calculated by supplied script
Model file size	Not reported by supplied script

TABLE VII Additional Prototype Service Measurements Reported in The Project Source.

Parameter	Reported value
Dashboard propagation delay	80–150 ms
RTDB write throughput	300+ writes/s
Report generation time	2–3 s
Detection-to-notification latency	<1.2 s
Chatbot response time	100–250 ms

C. Discussion

The LightGBM model achieved an MAE of 175.8 MW and an RMSE of 234.9 MW on the final chronological 20% test set, with an R^2 score of 0.9908. The high R^2 indicates that the model explains a large proportion of the variance in the held-out AEP demand series; however, the absolute errors remain important when assessing operational usefulness. MAPE is not calculated by the supplied script. These results apply to the PJM AEP regional demand dataset and should not be interpreted as household-meter accuracy. The anomaly detector is fitted and scored on the same dataset in the supplied implementation, so its anomaly count is not a validated false-positive or detection rate. Further evaluation should use la-belled anomaly examples and a separate chronological test set. The exported LightGBM text file is not evidence of successful ESP32 inference.

VIII. LIMITATIONS AND FUTURE WORK

The system is a prototype for energy monitoring and consumer assistance, not a certified revenue meter or a utility-grade protection and control system. Performance depends on sensor calibration, installation quality, network reliability, and the representativeness of the forecasting dataset. Wi-Fi or cloud outages can limit remote visibility even while local acquisition and inference continue.

Future work includes documenting the full training con-figuration, evaluating forecasts across multiple load profiles and seasons, quantifying false-positive and false-negative rates on labelled anomaly events, and running longer reliability tests. Device provisioning, encryption, key management, OTA update integrity, electrical isolation, and multi-node scalability require further validation. Renewable-energy forecasting and larger feeder or substation deployments can follow, along with Transformer-based models and federated learning for privacy-preserving analytics.

IX. CONCLUSION

This paper presented an IoT-enabled smart energy monitoring architecture combining ESP32-based sensing, cloud-assisted services, multilingual consumer support, LightGBM next-hour demand forecasting, and Isolation Forest anomaly scoring. The supplied training code defines a chronological 80:20 forecasting split, engineered lag and calendar features, and the model hyperparameters, and exports the fitted model and preprocessing artifacts. On the PJM AEP dataset, the LightGBM forecaster achieved an MAE of 175.8 MW, an RMSE of 234.9 MW, and an R^2 score of 0.9908 on the chronological test set. These results establish forecasting performance for the evaluated

regional demand series, but do not establish household-meter accuracy or successful on-device inference. ESP32 inference measurements and independently labelled anomaly-detection results remain necessary for vali-dating embedded deployment and anomaly performance. The work provides a foundation for further validation, integration, and larger-scale evaluation.

ACKNOWLEDGMENT

The authors sincerely thank their project guides, Dr. Narendra Bhagat and Dr. Rajendra Sawant, for their mentorship, technical guidance, and support throughout this research.

REFERENCES

[1] Al-Turjman and M. Abujubbeh, "IoT-enabled smart grid via SM: An overview," *Future Generation Computer Systems*, vol. 96, pp. 579–590, 2019, doi: 10.1016/j.future.2019.02.012.

[2] M. Pau, E. Patti, L. Barbierato, A. Estebsari, E. Pons, F. Ponci, and A. Monti, "A cloud-based smart metering infrastructure for distribution grid services and automation," Sustainable Energy, Grids and Networks, vol. 15, pp. 14–25, 2018, doi: 10.1016/j.segan.2017.08.001.

[3] N. Hossein Motlagh, M. Mohammadrezaei, J. Hunt, and B. Zakeri, "Internet of Things (IoT) and the energy sector," Energies, vol. 13, no. 2, Art. no. 494, 2020.

[4] S. E. Bibri, "The IoT for smart sustainable cities of the future: An analytical framework for sensor-based big data applications for environmental sustainability," Sustainable Cities and Society, vol. 38, pp. 230–253, 2018.

[5] A. Ullah et al., "Secure healthcare data aggregation and transmission in IoT—A survey," IEEE Access, vol. 9, pp. 16849–16865, 2021.

[6] D. Mocrii, Y. Chen, and P. Musilek, "IoT-based smart homes: A review of system architecture, software, communications, privacy and security," Internet of Things, vols. 1–2, pp. 81–98, 2018.

[7] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in Advances in Neural Information Processing Systems, vol. 30, 2017.

[8] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in Proc. IEEE Int. Conf. Data Mining, 2008, pp. 413–422.

[9] OASIS, "MQTT Version 3.1.1," OASIS Standard, Oct. 2014. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html

[10] Google, "Firebase Realtime Database documentation." [Online]. Available: https://firebase.google.com/docs/database. Accessed: Nov. 2, 2026.

[11] Google, "Understand Firebase Realtime Database Security Rules." [Online]. Available: https://firebase.google.com/docs/database/security. Accessed: Nov. 2, 2026.