Live Residential Energy Consumption Monitoring via Internet of Things: A Cost-Effective Prototype with Renewable-Centric Analysis

Shubham Kumar
Department of Computer Science and Engineering
RV Institute of Technology and Management
Bengaluru, Karnataka, India
suku3345@gmail.com

How to Cite this Article:
Kumar, S. (2026). Live Residential Energy Consumption Monitoring via Internet of Things: A Cost-Effective Prototype with Renewable-Centric Analysis. International Journal of Creative and Open Research in Engineering and Management, 02(05). https://doi.org/10.55041/ijcope.v2i5.488

https://doi.org/10.55041/ijcope.v2i5.488

Abstract

The forward-looking feedback systems in the hands of consumers become non-starters in certain economies; and over-use of electricity is an emerging issue with regard to household consumption. In this paper a prototype of the low cost, IoT based household energy consumption measuring system is designed and programmed with ESP32 microcontroller, ACS712 hall-effect current sensor and ZMPT101B voltage sensor, and software for the system is developed and tested. The node captures samples from both mediums, providing RMS voltage, RMS current and active power at a pre-specified power factor, progressive energy and then pushing all of these samples via WiFi to a locally hosted python-flask web server as json payload. All records are stored in the backend in a flat CSV file as well as a relational table in a sqlite, no need of external packages to create trend charts in SVG format, alerts when thresholds are met and trigger a self-refreshing browser based dashboard. The rooftop solar contribution is a damping of a measured power use by a subordinate model, which reasons sustainability without installing any additional equipment. The validation data consisted of a synthetic 24 hour set of 144 ten minute readings yielding an average power of 230.66 W, peak power of 493.55 W, total energy produced of 5.536 kWh, energy consumed from the electric grid of 4.195 kWh (given that the renewable offset modelled 1.341 kWh) and eight alerts for threshold breaches, most of which occurred between 16:00 and 20:00 (even evening). The prototype is deliberately reproducible, and is appropriate for submission to students' mini-projects and for entry-level academic publications. This is an AC power consumption measurement system for consumers with the integration of ESP32 microcontroller, Flask dashboard and two household energy sensors: the ACS712 and ZMPT101B.The project is a power usage measurement system for home appliances that incorporates two home energy monitors: ACS712 and ZMPT101B, a Flask dashboard and an ESP32 microcontroller.

I. INTRODUCTION

Demand for residential energy, including a growing number of affordable consumer electronics, air-conditioning systems and EV charging stations, has been on a rise worldwide. Across all member households with a developing-region household member, the universal consumption indicator is a monthly bill for the utilities consumed which, in most cases, cannot be attributed to a particular appliance or does not have temporal granularity. One limitation on voluntary demand reduction that is commonly cited is a lack of actionable feedback [1, 2]. The Internet of Things (IoT) platforms have ushered in the era of technically and economically feasible sub-daily energy telemetry. This sensing node will fetch the data of current and voltage at the main distribution board and upload the obtained data to the server using existing home WiFi network and data will be generated and filled in the dashboard in near real-time with components cost less than 1000 INR. Yet, a survey of published systems that students can access reveals a common issue: technically and sophisticated enough to be published in, say, a journal they tend to be dependent on a cloud-level middleware that is not freely available or paid API tiers/hardware that likely are hard to obtain in Tier-2 cities where a VTU lab may be located, even though that system is truly available within a VTU lab, the level of storage, analytics, and communication that one would expect if a system is available for journal submission is lacking. The prototype presented here aims at meeting that need. It features a small Bill of Material and conveys everything via a locally hosted server and is explicitly concerned with the whole arc from analogue to browser visualization, dual format logging, alarm generation and estimation of the net renewable demands; an end-to-end space worthy of scholarly publication and achievable in a student bench. The rest of this paper is organized as follows: Section II reviews related work in Section III presents problem and goals, described in Section IV is the proposed architecture, in Section V elaborates algorithms, in Section VI explains the experimental setup, presents experimental results in Section VII, and in Section VIII provides the graph generation to be presented in Section IX for critical discussion and in Section X concludes and proposes future extensions.

II. LITERATURE SURVEY

At least for the last decade, monitoring residential energy use has been a research topic. Ahmed et al. [1] did mention two key things, which deter the economical distribution of a system: the need to deploy the system and the cost of the system itself. While Machorro-Cano et al. [2] present in detail how to implement HECS with a pipeline of big-data and with classifiers based on machine learning, the demands on the compute and productivity connectivity, especially involving an undergraduate-level student project, is a challenge to fit on limited resources. Next, after calling the user to work with local network protocols rather than by relying on a brokering in the cloud (as he does in this work), Ben Dhaou [3] designed his own smart-meter and smart-plug which could be connected using local network protocols to communicate between each other, publishing an REST API. There's the last example, extending HEMS to the optimisation of the optimisation of the appliances themselves (OOFORA), Jo et al. [4] give convincing and concrete evidence of the feasibility of automated demand response and some sign of the extra complexity of the algorithms and hardware that will trip up the novice. Future-scope, appliance classification: Classification of appliance signatures using a combination of the aggregate current waveform and future-scope features was inspired by the idea of Lin et al. [5] who employed a neuro-fuzzy classifier and non-intrusive load monitoring (NILM). Thus, Biswal and Suryadevara [6] claimed that usage of this smart-plug paradigm in smart city grid environment could hardly influence the grid infrastructure and could be a promising beginning towards the smart building-wide grid system. Even in the case of an energy node with a light-weight energy application (HTTP), the security aspect was mentioned in Martinez-Martínez et al. [7] indicating it as a future improvement to be done, including token authentication. Of this work, the two characteristics are: For one, the sensing-to-dashboard sub-system is familiar and just up to date and secondly present voltage meter apparatus are turning into normal today. In addition, the research has continued towards intelligent scheduling, hardening, multi-energy-vector optimisation, which is more complicated and needs to be deployed in a prototype in the future. One on hand there’s the traditional “display-only” meter and, on the other, a totally new “HEMS” (home energy management system); this work can be summarised as technically sound enough to enable it to illustrate the concepts behind IoT, dual-format persistence, SVG format visualisation and reasoning about the need to refill an offset and alert when thresholds are breached, and be covered in one and done.

TABLE I

Comparison of Related Work Against Present Prototype

Ref.	Key Contribution	Notable Strength	Limitation Addressed Here
[1]	Low-cost IoT household distribution system	Practical and affordable hardware	No emphasis on beginner-reproducible packaging or renewable estimation
[2]	Big-data HEMS with ML classification	Strong conceptual and algorithmic depth	Compute and cloud requirements exceed student lab capacity
[3]	MQTT smart meter and smart-plug pair	Rich feature set with REST API	Higher implementation complexity; no local-only offline mode
[4]	Appliance-scheduling HEMS with IoT control	Optimisation-oriented, strong demand-response view	Requires smart appliances; unsuitable for introductory prototyping

III. PROBLEM STATEMENT AND OBJECTIVES

A. Typical consumer in India for residential use gets electricity bills fortnightly or monthly with no indication of the peak demand hours of the day, the proportionate contribution of different loads that make up the bill and whether there are any savings to gain under demand side measures. Commercial smart meters have been unable to fill in the gap as these have been being deployed by the utility provider or are too expensive to pay up front by consumers. In practice it would be desirable to have an available node that is local, self-installing, and preserves privacy and is assembled and operated by the student at any electronics store. B. Objectives – Measure one-phase, residential voltage and current to engineering tolerances with commercially available voltage and current sensors. Use an existing home wi-fi network to connect the readings from the ESP32 to a Python - Flask backend via an HTTP-JSON protocol and calculate the RMS voltage, the RMS current, average power dissipated at each sampling time, and the total energy dissipated. Enable live SVG trend graphs, and provide a self-refreshing browser-dashboard to transmit all readings (useful for human interaction); provide an alert and log of threshold breaches during peak load, without requiring additional hardware to perform load-management decisions based on quantification of sustainability in the net grid demand sub-model; provide a renewable-offset sub-model – useful without any additional hardware to estimate net grid demand. IV. The PROPOSED METHODOLOGY is divided into 2 layers such as 1) Hardware acquisition layer, 2) Software processing layer. The ESP32 development board is adopted for the hardware level to realize the central processing unit. That is connected to the ACS712 current sensor (for the ±5 A variant) via one analogue input pin and is connected to the ZMPT101B voltage sensing module via another pin. All sensors will give low voltage analog output signals proportional to the instantaneous AC output signal. The software layer is installed on all machines in the local network and hosts a Flask application, where one can send an HTTP POST request, this will be processed, stored and alerted, and generate SVG visualization charts, then display the dashboard. The switching power supply noise from the board itself and capacitive coupling from the nearby AC conductors results in noise on the ADC readings of the ESP32s. To overcome this, the firmware makes N samples (N = 500 in the reference code) of each channel, and obtains the RMS value of the channel from the N samples. The burst-RMS method is used for minimizing high frequency noise level and recording audio sample window with statistical significance as a replica of the AC half-cycle. ADC counts are calibrated to physical units by linearly fitting the readings from the reference clamp meter (and the voltage standard). Data Persistence Strategy – C. Dual format persistence is used in two different ways. The CSV log is not specific to the machine and so can be read by any spreadsheet program, either to view each reading in the log or to conduct any summary calculations. In addition to being able to be used in the SQLite database, parameterized queries, aggregate functions, and where clauses to filter rows are present, which are not present in a flatfile. Records are identical in both stores and either store can be a source of authoritative records (depending on the downstream task). This design decision also acts to be fault tolerant if the SQLite write fails for any reason as it is a result of the SQLite database being locked, the CSV record will not fail. D. Renewable-Offset Sub-Model: The actual PV hardware would have been about triple the

bill of material just and included AC side safety considerations, which would not make for a mini-project. Instead, a computer-generated, parameterised solar-generation curve is determined in software, with a sinusoidal model that rises to a maximum at solar noon and is zero outside a user-defined daylight window. At every time step the model gives an estimation of the Renewable energy P_r, while the common demand on grid is the max between P and − P_r, where P is the measured power. This new term is not measured but modelled, making the project relevant to the new residential rooftop-solar market segment, as well as a new term, the self-consumption ratio, that has became an increasing requirement for building-energy audits. V. ALGORITHMS A. RMS Estimation Then the RMS value of a burst of N ADC-derived samples x_1, x_2, …, x_N (after calibration in physical units) is: $X_{sms} = \sqrt{[\ (1/N) \times \Sigma\ x_i^2\ ]}$ This expression is used separately on both the voltage and current channels to obtain V_{sms} and I_{sms}, respectively. C. Phase angle for B. Active Power and Energy - Phase angle cos(φ), is assumed as 0.9, which is a typical power factor used by households. At each time step, the energy E (in kWh) is updated using the trapezoidal approximation to an integral: $E \leftarrow E + P \times \Delta t / 3600000$, where P is the power (in watts) and Δt is the time interval between measurements (in milliseconds).C. Threshold Alert An alert is generated if P is above a given threshold τ_P when a reading occurs. The reference configuration is τ_P = 450 W where a log record with a timestamp is added to a dedicated log table in SQLite, and is displayed on the dashboard, where the user will see an alert record. The threshold is intentionally exposed as an environment variable for future deployment to make it flexible to the baseline consumption of each house. The following are methods for estimating net demand for renewables: The estimate of the renewable power available at time t is given by $P\_r(t) = P\_r(t) = P_{peak} \times \sin^2[\pi(t - t_{sa}) / (t_{set} - t_{sa})]$ for $t_{sa} \le t \le t_{set}$ $P\_r(t) = 0$ otherwise Peak Renewable power is P_{peak} which is configured (default value 200 W), solar window is t_{sa} = 06:00 and t_{seh} = 18:00 respectively. This means that the net grid demand is P_net = max(P − P_r, 0). R. Firmware Control Flow The ESP32 firmware then performs the following operation: (1) takes N voltage readings and N current readings; (2) calculates the RMS values of voltage and current; (3) calculates the voltage times current (power); (4) serialises the JSON object {timestamp, V_rms, I_rms, P, E} into a string; (5) POSTs the string to the Flask server via an HTTP connection; (6) waits for the Flask server to return an HTTP 200 response; (7) sleeps for the desired sleep time (default 600 s). On failure the firmware will repeat the POST up to 3 times before recording a connection error to UART and falling back to sleep.VI. EXPERIMENTAL SETUP

All the components are easily available from the component distributors in Bengaluru and the cost of purchase at the time of writing is less than 700 INR.The components used here are: ESP32-WROOM-32, one current-sensor breakout board (ACS712-05B), one voltage-transformer module (ZMPT101B) for measuring single-phase voltage, a half-size solderless breadboard and a handful of jumper wires whose total cost at the time of this writing is less than 700 INR. The software stack is installed on any environment with python 3.9+ support, and only requires three packages (Flask, sqlite3 [which has been installed in the standard python library] and the csv library - no package requires a paid licence or connection to a network after installation). Because of the safety concerns of using live mains as opposed to a mini-project lab, software validation was done using an Indian-dwelling environment load profile in the form of a generated dataset. The generated load comprises 144 records with interval of 10 minuters, starting from the midnight class, distributed between two different power levels, one during the night (-80W) and another during the day (moderate power -150W to 250W) with a strong peak during the evening (-350W to 500W) that resemble those presented in published Indian residential load curves. Each sample has noise added to it to account for sensor variation: (the nominal value ±5 % of nominal value.) This dataset is then sent via the same ESP32 communicating HTTP endpoint of the server to test all lines of server code.

VII. RESULT ANALYSIS

Table II summarises the key performance metrics extracted from the 144-reading validation run.

TABLE II

Summary of Validation Run Metrics

Metric	Value
Total readings	144
Average power	230.66 W
Peak power	493.55 W
Total energy consumed	5.536 kWh
Modelled renewable offset	1.341 kWh
Net grid energy	4.195 kWh
Threshold-breach alerts	8

The power profile has three behavioural stages, as predicted by the generation model. Between midnight and about 06:00, the consumption level remains reasonably stable at the 80 W standby level, which is thought to be caused by always-on consumers like routers, STBs and refrigerator cycling. When the power is on, between 130 W and 260 W is generated during the day, when kitchen appliances and charging loads are switched on or off. The most active time of the day is the evening window, from 16:00 to 21:00, as air-conditioner startup times, times of use of cooking ranges and times of television and driving with more than 450W are the most frequent at eight different times. The maximum in the dataset was 493.55 W, at 18:20, and the longest sustained alert sequence. The modesty of any 200 W-peak rooftop installation would, notionally, offset 24.2 % (1.341 / 5.536) of the daily domestic demand for energy and reduce the energy use supplied by the grid from 5.536 kWh to 4.195 kWh. However, the offset does not help mitigate the alert events as the evening peak falls outside of the peak period of solar generation (14:00–18:00 in the model), reinforcing the need for battery storage in any real hybrid energy home design, and justifying the first future-scope item in Section XI.

VIII. GRAPH GENERATION

The chart builder generates SVG markup directly in Python without binding to the heavy weight plotting library Matplotlib. This decision removes a major installation requirement in resource limited environments, and makes generated files portable; they can be viewed without a Python runtime in any browser or document presentation program that supports SVG. These steps are (i) query SQLite for N most recent readings ordered by timestamp, (ii) extract the target series (power_w or net_grid_w), (iii) transform data coordinates to pixel coordinates by applying a linear transformation with user-specified margins, (iv) render axes, label/attributed tick marks, a shaded area polygon, and a polyline connecting the data points, and (v) send the resulting SVG string to a Jinja2 template served by Flask. The dashboard polls the /chart endpoint every 30 seconds using a meta-refresh tag - no WebSocket infrastructure is required for a near real-time update.

IX. DISCUSSION

The real interest of this work is not any one new novelty in the algorithms, such as power-factor-corrected RMS measurement and threshold alerting have both already been published in other places, but that a thorough and tested end-to-end pipeline has been set loose from these algorithms, which can be reproduced. When analysing the design, there are several factors that should be considered. The major restriction in the accuracy is the assumed value of 0.9 for the power factor. The residential load ranges from slightly less than unity (for a resistive water heater) to under 0.7 (older, un-capacitor corrected induction motors). Thus, a fixed value would thus be a systematic error which is dependent on the load mix. This error is, however, limited: At the extremes of the range ($\cos(\varphi) = 0.7–1.0$) the fixed-0.9 assumption will lead to an over or under estimation of active power by at most 22 % and 11 % respectively. The accuracy class is considered sufficient for the monitoring application which is not used for billing purposes and is used to detect trends and create alerts. The modelled renewable

offset is a notional, rather than an actual, volume – like the two other renewable resources. From a pedagogical view, it's an acceptable addition as they will have to consider the "production-consumption balance", "self-consumption ratio" and "peak-demand mismatch", which are terms in energy-engineering courses, even if they have not needed to add an inverter interface, or added it with the associated cost and safety measures. The model is also architecturally clean: it's only a single function call in the server; make it a live inverter's API is just one line substitution. The current implementation doesn't contain (purposefully) security. Flask server can be used by any client on the local network to make a POST request without authentication. This is OK for a prototype lab network on a single subnet, but wouldn't work for a home appliance deployment. The smallest set of additional measures that needs to be employed in a production deployment is token-based authentication and the usage of HTTPS using self-signed certificates, according to Martínez-Martínez et al. [7].

X. CONCLUSION

This paper introduces a simple prototype that is easily replicable, economical, and includes the whole system from analogue sensors to visualization in the browser, which is played out in a local, self-hosted architecture. The device implementing the adaptation is comprised of an ESP32 microcontroller along with an ACS712 and a ZMPT101B sensor, which capture the current and voltage information present in the AC mains line, compute RMS value of power and accumulated energy, and communicates with the back-end using standard WiFi through sending JSON records to a simple Flask-based back-end. It keeps all the records in the two formats CSV and SQLite; produces dependency-free SVG charts; provides notification alerts if the thresholds, defined by the configurations, are exceeded; and calculates the net demand on the power grid when adjusted for renewable energy without the use of additional hardware. The consistency of all the pipeline stages in performing their operations was checked and the results (230.66 W Average power, 493.55 W Peak power, 5.536 kWh Total energy, 8 Alerts and 24.2 % Estimated renewable coverage) were validated with a 144-point synthetic daily load profile and are consistent with published Indian residential load characteristics. The prototype has been designed as a stepping stone to more advanced designs which will include the ability to accurately measure the power factor of the appliance, appliance-level NILM classification and secure cloud connected operation.

XI. FUTURE SCOPE

Where the modelled renewable offset is substituted with this, a real data feed from an intelligent Solomon Grid solar inverter or a battery management system with Modbus or SunSpec REST communication will be used. Remove the assumption of the constant PF = 0.9, and go to actual power factor estimation, using simultaneous sampling of the voltage/current signal from the bridge and digital cross-correlation. Refer to Lin et al. [5] for an application of non-intrusive load monitoring to disaggregating the current signal in the aggregate to individual current loads of every appliance using a lightweight edg-ML classifier. Apply security measures (HTS, JWT token authentication and TLS) at the HTTP endpoint in accordance the security guidelines presented by Martínez-Martínez et al. [7]. Extend sensing node to 3phase supply to accommodate commercial and institutional and small industry applications and provide automatic load scheduling, similar to that proposed by Jo et al. [4] where loads are scheduled to off-peak periods (deferrable loads – water heaters, washing machines etc.).

XII. REFERENCES

[1] M. M. Ahmed, S. S. M. Ghoneim, M. I. Alghamdi, and S. M. A. Alelyani, “Cost-effective design of IoT-based smart household distribution system,” Designs, vol. 5, no. 3, Art. no. 55, 2021, doi: 10.3390/designs5030055.

[2] I. Machorro-Cano, G. Alor-Hernández, M. A. Paredes-Valverde, L. Rodríguez-Mazahua, J. L. Sánchez-Cervantes, and J. O. Olmedo-Aguirre, “HEMS-IoT: A big data and machine learning-based smart home system for energy saving,” Energies, vol. 13, no. 5, Art. no. 1097, 2020, doi: 10.3390/en13051097.

[3] I. Ben Dhaou, “Design and implementation of an internet-of-things-enabled smart meter and smart plug for home-energy-management system,” Electronics, vol. 12, no. 19, Art. no. 4041, 2023, doi: 10.3390/electronics12194041.

[4] H.-C. Jo, H.-A. Park, S.-Y. Kwon, and K.-H. Cho, "Home energy management systems (HEMSs) with optimal energy management of home appliances using IoT," Energies, vol. 17, no. 12, Art. no. 3009, 2024, doi: 10.3390/en17123009.

[5] Y.-H. Lin, "Design and implementation of an IoT-oriented energy management system based on non-intrusive and self-organizing neuro-fuzzy classification as an electrical energy audit in smart homes," Applied Sciences, vol. 8, no. 12, Art. no. 2337, 2018, doi: 10.3390/app8122337.

[6] N. K. Suryadevara and G. R. Biswal, "Smart plugs: Paradigms and applications in the smart city-and-smart grid," Energies, vol. 12, no. 10, Art. no. 1957, 2019, doi: 10.3390/en12101957.

[7] J. Martínez-Martínez, D. Aponte-Roa, I. Vergara-Laurens, and W. W. Weaver, "A low-cost secure IoT mechanism for monitoring and controlling polygeneration microgrids," Applied Sciences, vol. 10, no. 23, Art. no. 8354, 2020, doi: 10.3390/app10238354.

[8] National Institute of Standards and Technology, NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 4.0, NIST Special Publication 1108r4, Gaithersburg, MD, USA, 2021, doi: 10.6028/NIST.SP.1108r4.

[9] Allegro MicroSystems, "ACS712 fully integrated, Hall-effect-based linear current sensor IC," Datasheet, 2023. [Online]. Available: https://www.allegromicro.com/en/products/sense/current-sensor-ics/acs712. Accessed: May 2026.