Elektron: An Electricity Monitor Application

Ganesh Thampi, Nishan Paudel, Weiyu Wang, Patrick Chau
Department of Computer Engineering
San Jose State University
May 7, 2024

***Abstract*—This paper presents the architectural design and implementation of Elektron, a full-stack application developed for to monitor electricity consumption and device health, alongside alerts and AI-generated insights. Elektron addresses the ever-increasing need for efficient energy management and maintenance in various environments. The system's architecture is derived from, integrating data acquisition from electrical devices using MQTT, real-time data processing with Kafka, storage in MongoDB, a Python-based backend API using FastAPI, and a React frontend. This approach allows Elektron to provide users with valuable insights into their energy usage, identify device errors, and receive real-time alerts to optimize energy consumption and prevent equipment failures. The paper details each component of the Elektron system, including its functionality, the technologies employed, and rationale behind the architectural choices. Furthermore, it discusses the system's results, performance considerations, scalability, and future potential.**

Keywords—Real-Time Data Monitoring, Electricity Monitoring, Kafka, MongoDB, FastAPI, React

I. Introduction

With the increasing demand for electricity, coupled with rising energy costs alongside the arrival of time-of-use rates, and growing concerns about environmental sustainability, has created a critical need for effective energy management solutions. Traditional methods of monitoring electricity consumption, often relying on infrequent manual readings or aggregated bills, provide limited visibility into real-time energy usage patterns and device health. In conjunction with the introduction of time-of-use rates, where pricing for electricity is greatly increased at certain times [1]. This only serves to further confuse unknowing consumers about their consumption and bills. For example, Kempton and Layne argue that this is similar to if you had received telephone bills, but without any of the information of importance such as the calls made [2]. This lack of data stops efforts to identify energy inefficiencies, discover potential equipment failures, and implement energy-saving measures. By following some of Fisher's characteristics of good feedback/response for energy, it should be noted that it consists of detailed information by which is a direct or indirect result of the user in order to bring attention to their activities, increases their perception of their ability to control their energy consumption, and may lead to certain outcomes such as framing the information to motivate energy efficiency [3]. With this in mind, a report on the effectiveness of feedback on energy consumption by Darby states savings of nearly 10-20% on electricity for systems that have some form of display [4].

To address these challenges/characteristics, this paper proposes Elektron, a full-stack application designed to provide a comprehensive and real-time solution for monitoring electricity usage, assessing device health, generating alerts, and receiving advice through artificial intelligence. Elektron allows users to gain a deeper look into their energy usage, to optimize consumption, and proactively manage electrical devices to prevent failures and reduce energy waste. The system uses a modern, distributed architecture, utilizing a combination of technologies to ensure scalability, reliability, and performance.

The Elektron system is designed to receive data from various electrical devices, which are also defined as producers. These producers can range from smart plugs and energy meters. The data collected includes real-time electricity consumption and other relevant parameters that can indicate device health. This data is then processed, stored, and presented to the user through a React web interface. The system's alerting capabilities notify users of abnormal energy consumption patterns or potential device malfunctions, letting them to respond quickly.

II. System Architecture Overview

Elektron's architecture is designed to be performant, scalable, and resilient, allowing it to handle a large amount of data from various sources. The system comprises of the following key components, as seen in Figure 1 below:

Fig 1. Architecture Diagram showing application workflow

A. *Data Acquisition:*

Electrical devices equipped with sensors and communication capabilities act as the data producers. These devices measure electricity consumption and other relevant parameters and transmit this data using the MQTT protocol. MQTT was chosen as it is lightweight and efficient, making it suitable for many conditions.

B. *Data Processing:*

To ingest data, we use Apache Kafka, it serves as the central message broker in the Elektron system. It receives data from the MQTT broker, buffers it, and makes it

available to aggregator consumers. Kafka's distributed architecture and design ensure high availability and scalability, enabling the system to handle a large amount of data. The use of Kafka allows for decoupled data producers and consumers, enhancing the system's flexibility.

C. Data Aggregator Consumer:

A consumer application processes the data from Kafka in real-time. This component aggregates the raw data into metrics, such as average power consumption over five-minute intervals. This aggregation reduces the amount of data stored in the database and provides more relevant information to the user.

D. Data Storage

MongoDB, a NoSQL database, is used to store the aggregated data. MongoDB's flexibility and scalability make it well-suited for handling the data structures generated by the various electrical devices. The database stores the processed energy consumption data, device health metrics, and alert history.

E. Backend Layer

A backend API, developed using Python and the FastAPI framework, is built between the MongoDB database and the frontend UI. FastAPI was chosen for its high performance, ease of development, and automatic API documentation generation. The API provides endpoints for retrieving various metrics such as energy consumption data, device health status, and alert information.

F. Frontend Layer

The user interface is built using React, a JavaScript library popular for building user interfaces. React provides a highly dynamic and responsive user experience, letting users visualize their energy consumption data, monitor device health, and configure alerts. The frontend communicates with the backend API to retrieve and display data.

III. Component Implementation and Details

This section gives a more detailed description of each component and its specific functionality in the system.

A. Data Producers (Electrical Devices):

Electrical devices, acting as producers, are instrumented with sensors, namely the Tavo P115 smart Wi-Fi socket, to measure electrical parameters such as voltage, current, and power [5]. These devices are configured to publish the collected data to an MQTT broker. The frequency of data transmission can be adjusted based on the specific requirements of the application and the capabilities of the device. The data format is standardized to ensure compatibility across different device types.

B. Kafka:

Kafka acts as a distributed, fault-tolerant message broker. Data published by the MQTT broker is ingested into Kafka topics. Kafka's partitioning and replication mechanisms ensure data durability and high availability. The use of Kafka allows for asynchronous processing of data, decoupling the producers from the consumers and enabling the system to handle bursts of data.

C. Data Aggregator Consumer

The data aggregator consumer subscribes to the relevant Kafka topics and processes the incoming data stream. This component performs data aggregation, generally calculating average values over a defined time window (for our application, five minutes). The aggregated data is then written to the MongoDB database. This aggregation process reduces the storage requirements and provides a better representation of the energy consumption patterns.

D. MongoDB

MongoDB stores the aggregated data in collections, with each document representing a specific time interval and device. The database schema is designed to accommodate the various data fields, including timestamps, device identifiers, energy consumption values, and other metrics. MongoDB's indexing capabilities are utilized to optimize query performance, ensuring fast retrieval and exchange of data by the backend API.

E. Backend API (FastAPI)

The FastAPI-based backend API provides a way to access the data stored in MongoDB. The API defines endpoints for retrieving energy consumption data, device health status, and alert information. The backend can also send email alerts using an email the user has specified. Additionally, integration with Google’s Gemini artificial intelligence has been made in order to provide helpful insights on-demand. The API is designed to be performant and scalable.

F. Frontend(React)

The React-based frontend provides a user-friendly interface for visualizing and interacting with the data. The frontend displays real-time energy consumption data in various formats, such as charts and graphs, allowing users to easily identify trends and patterns through a dashboard for monitoring device health. Furthermore, additional settings such as dark mode, email notifications, and threshold for alerts can be specified, and also allows users to interact with Google’s Gemini. The frontend communicates with the backend API using asynchronous HTTP requests.

IV. Results and Performance Considerations

Our application, Elektron provides a fairly encompassing solution for monitoring electricity consumption, tracking device health, and generating alerts. Our system's architecture ensures scalability, reliability, and performance. The use of Kafka enables real-time data processing, while MongoDB provides a flexible and efficient data storage solution. The Python-based FastAPI backend provides a high-performance API, and the React frontend delivers a user-friendly and interactive experience.

The system offers several benefits:

- Energy Efficiency: By providing detailed insights into energy consumption patterns, Elektron allows users to see areas where they can reduce energy usage and lower their electricity bills.
- Device Health Monitoring: The system's ability to track device health metrics allows proactive

maintenance and preventing failure.

- Alerting and Notifications: Elektron provides real-time alerts alongside with email alert notifications for critical events, such as excessive power consumption or device malfunctions, allowing users to take immediate action.

- Data-Driven Decision Making: The system's comprehensive data analysis capabilities alongside AI-driven insights allow users to make informed decisions about energy management and device maintenance.

V. Conclusion and Future Work

In conclusion, our application proves to set out its goal of providing actionable insights into a user's electricity consumption pattern by allowing them to view and monitor their devices alongside various metrics and data interpretations regarding electricity consumption. Furthermore, we have also implemented AI-generated insights and real-time alerts and notifications to inform the user on certain critical events. However, there is still some work we can do regarding our system.

Future work will focus on several areas to further enhance the capabilities of Elektron:

Integration with Smart Grids: Integrating Elektron with smart grid technologies to enable more efficient management of electricity distribution and consumption. This could involve providing real-time data to grid operators, enabling demand-side management, and supporting the integration of renewable energy sources.

Support for Additional Protocols: Expanding support for additional data acquisition protocols beyond MQTT to accommodate a wider range of electrical devices and sensors. This could include protocols such as Modbus, Zigbee, and Wi-Fi.

Enhanced Alerting: Implementing more sophisticated alerting mechanisms, such as multi-level alerts and integration with various notification channels (e.g., SMS or push notifications).

Elektron represents a significant step towards enabling more efficient energy management and proactive device maintenance. By providing users with detailed, real-time insights into their energy consumption and device health, Elektron allows them to make informed decisions, optimize energy usage, and prevent costly equipment failures.

References

[1] "Time of Use," SVCE. Available: https://svcleanenergy.org/time-of-use/

[2] W. Kempton and L. L. Layne, "The consumer's energy analysis environment," Energy Policy, vol. 22, no. 10, pp. 857–866, Oct. 1994, doi: 10.1016/0301-4215(94)90145-7. Available: https://www.sciencedirect.com/science/article/pii/0301421594901457.

[3] C. Fischer, "Feedback on household electricity consumption: a tool for saving energy?," Energy Efficiency, vol. 1, no. 1, pp. 79–104, Feb. 2008, doi: 10.1007/s12053-008-9009-7. Available: https://doi.org/10.1007/s12053-008-9009-7

[4] S. Darby, "THE EFFECTIVENESS OF FEEDBACK ON ENERGY CONSUMPTION A REVIEW FOR DEFRA OF THE LITERATURE ON METERING, BILLING AND DIRECT DISPLAYS," Apr. 2006. Available: https://smartgridawareness.org/wp-content/uploads/2016/05/effectiveness-of-feedback-on-energy-consumption-darby-2006.pdf

[5] "Mini Smart Wi-Fi Plug, Energy Monitoring." Available: https://www.tp-link.com/us/home-networking/smart-plug/tapo-p1