Nesnelerin Yapay Zekası (AIoT) Tabanlı Enerji Sistemlerinin Az-Kodlu Geliştirilmesi

Mehmet Emin Çağlayan [1,*] and Geylani Kardas [2]

[1] Ege University International Computer Institute, 35100, Bornova, Izmir, Türkiye
[2] Ege University International Computer Institute, 35100, Bornova, Izmir, Türkiye

* Corresponding author: mehmetemincaglayann@gmail.com

Özet

Yapay Zeka (AI) ve Nesnelerin İnterneti (IoT) teknolojilerinin birleşerek "Nesnelerin Yapay Zekası" (AIoT) altyapısını oluşturduğu gözlemlenmektedir. Bu entegrasyonun amacı, insan-makine etkileşimleri, IoT operasyonları, büyük veri analitiği gibi çeşitli alanlarda iyileştirmeler sağlamaktır. AIoT tabanlı enerji sistemleri kullanıcılara bu avantajları sunmasına rağmen, geliştiriciler özellikle veri toplama ile ilgili karmaşık yazılım bileşenlerinin hazırlanması ve birleştirilmesinde zorluklarla karşılaşabilmektedir. Ayrıca enerji sistemlerine yönelik artan güvenlik ve kalite gereksinimleri, geliştirme süreçlerinin daha etkin ve daha otomatik bir şekilde yönetilmesini zorunlu hale getirmiştir. Bu bağlamda, düşük kodlu geliştirme platformlarının (LCDP) sağlanması, bu tür sistemlerin tasarım ve gerçekleştirilmesini kolaylaştırabilir ve model güdümlü mühendisliği (MDE) destekleyebilir. Bu nedenle, bu çalışmada düşük kodlu geliştirme yaklaşımının yeni bir uygulama alanı olan AIoT destekli enerji sistemlerinde sunduğu avantajlardan yararlanmak amacıyla bir LCDP tanıtılmaktadır.
Bu çalışmada sunulan LCDP, mevcut IoT tabanlı enerji cihazlarında kullanılan çeşitli araçları entegre etmekte ve bu araçların tek bir sistem üzerinden yönetilmesini sağlamaktadır. Ayrıca, gereksinim tanımlamadan modelleme ve kod üretimine kadar olan tüm süreçlerin otomasyonunu desteklemektedir. Geliştirilen LCDP, cihaz tanımlamaları, gerçek zamanlı değer arayüzleri ve uygulanacak yapay zeka yaklaşımları gibi gereksinimlerin kolayca tanımlanmasına olanak tanımaktadır. Bu çalışmada önerilen LCDP'nin, farklı bileşen bakış açılarıyla AIoT destekli enerji sistemlerinin grafiksel sözdizimi ile kolay modellemesini nasıl sağladığı ve nihayetinde otomatik kod üretimini nasıl gerçekleştirebildiği ele alınmaktadır.

Anahtar kelimeler: Düşük kodlu geliştirme, Yapay Zeka, Nesnelerin İnterneti, Nesnelerin Yapay Zekası, Model güdümlü mühendislik

Low-code development of the Artificial Intelligence of Things (AIoT) enabled Energy Systems

Abstract

It is observed that Artificial Intelligence (AI) and Internet of Things (IoT) technologies are merging to form the infrastructure of "Artificial Intelligence of Things" (AIoT). The aim of this integration is to provide improvements in various areas such as human-machine interactions, IoT operations, big data analytics, and more. Although AIoT-based energy systems offer the aforementioned benefits to users, developers may encounter difficulties particularly in preparing and integrating the complex software components related to data collection in the development of such systems. Moreover, the increasing security and quality requirements of energy systems have made it necessary to manage development processes more effectively and in a more automated manner. Within this context, providing low-code development platforms (LCDP) may facilitate the design and implementation and support the model-driven engineering (MDE) of such systems. Hence, in this paper, we introduce an LCDP for AIoT-enabled energy systems to benefit from these promising features of the low-code development in this new application domain. The LCDP herein integrates the various tools used in current IoT-based energy devices and enables their management through a single tool. Furthermore, it supports the automation of all processes from requirement specification to modeling and code generation. The developed LCDP enables easy definition of requirements such as device descriptions, real-time value interfaces, and AI approaches to be applied. We discuss how the proposed LCDP may provide easy modeling of AIoT-enabled energy systems with its graphical syntax from various component viewpoints and hence is capable of automatic code generation in the end.

Key words: Low-code development, Artificial Intelligence, Internet of Things, Artificial Intelligence of Things, Model-driven engineering

1. Introduction

Artificial Intelligence (AI) applications have begun to be integrated into existing hardware including IoT devices. Today's modern world is experiencing significant growth in software technologies to digitize all human-related processes. With the combination of AI and IoT technologies (Madakam et a., 2015), the concept of "Artificial Intelligence of Things" (AIoT) is emerging (Matin et al., 2023). Integrating AI and IoT aims to achieve higher performance and increased system reliability. AIoT-enabled solutions are applied in various sectors such as industry and manufacturing, healthcare, agriculture, and energy infrastructure. These solutions enhance efficiency, reduce system downtimes, and improve safety measures (Matin et al., 2023). However, major challenges faced by AIoT products in this sector include data security, network integrity, energy efficiency, and the complexity of data analysis algorithms (Yang et al., 2021).

In the energy sector, AIoT enables the establishment of intelligent, efficient, and predictable systems for both energy production and consumption. While IoT devices collect data from the field, AI analyzes these data to provide prediction, optimization, and automation. AIoT-enabled production solutions reduce energy costs through resource and process planning. Additionally, AIoT contributes to efficient energy management in areas such as increasing grid efficiency, optimal use of renewable energy sources, and the creation of decentralized energy trading (Jnr, 2024). Nevertheless, AIoT technology heavily relies on sensors, IoT devices, network connections, and data centers. Sustaining the functionality of AIoT infrastructure requires energy to connect many devices. Therefore, research is being conducted to reduce the energy demands of AIoT solutions. Moreover, changes in consumer demands, behaviors, and technological developments have increased energy consumption. As a result, energy management has become a critical issue for sustainable production (Matin et al., 2023). In this context, daily electricity usage is analyzed by comparing it to previous days' usage to inform consumers. Power systems help prevent unexpected energy waste by cutting off undesired electrical loads (Arivukkody et al., 2022).

Although AIoT-enabled energy systems offer the aforementioned advantages to users, developers face challenges particularly in the preparation and integration of software components related to data collection. Creating an abstract model of such IoT systems before implementation and developing the necessary software with minimal coding through visual syntax can improve efficiency (Arslan et al., 2023). Accordingly, this paper introduces a low-code development platform (LCDP) (Cabot, 2020; Di Ruscio et al., 2022) that supports the model-driven engineering (MDE) (Brambilla et al., 2017) of AIoT-enabled energy systems. Increasing demands for security and quality in energy systems require more effective and automated management of development processes. This study aims to contribute to related novel application area by presenting an LCDP example and to integrate different tools used in existing AIoT- enabled energy devices for centralized management.

The remainder of the paper is structured as follows: Section 2 presents a detailed discussion of the metamodel that forms the foundation of the abstract syntax of the proposed LCDP. Section 3 introduces the graphical concrete syntax of the LCDP, which enables the modeling of AIoT-enabled energy systems. Section 4 discusses the implementation of the web-based integrated development environment (IDE) that supports the use of the LCDP. Section 5 presents a case study demonstrating the use of the LCDP and its IDE in the development process of such energy systems. Finally, Section 6 offers a general evaluation of the study and concludes the paper.

2. *Abstract Syntax*

Abstract syntax can be defined as a conceptual representation of a modeling language's fundamental constructs and their relationships, independent of concrete representations (Saritas & Kardas, 2014). This abstract structure focuses on identifying the main entities, core properties, and relationships within a language. It is typically defined through a metamodel and is used to explain the semantics and structure of a Domain-Specific Language (DSL) or a Domain-specific Modeling Language (DSML) (Kardas et al., 2023).

The metamodel of the AIoT-enabled energy systems that provides the underlying abstract syntax for our LCDP, includes the meta-entities and their relationships given in Figure 1. Energy devices, parameters, and users are defined with the entities that enable data collection, visualization, and modeling processes. The main entities of the metamdeol derived for the AIoT-enabled energy systems are described below:

- **EnergyDevices** represent the AIoT-enabled energy devices defined in the system. Each device is associated with one or more parameters, and these parameters are read from the energy devices. Energy Devices model the source of data within other environmental systems and form the foundation of the entire model.
- **Parameters** are defined for reading data from the sensors on energy devices or data produced internally. These parameters provide information about the data received from the energy device, normalized using attributes such as scale factor and offset.
- **Energy Device Data** are recorded for tracking real-time measurements collected from energy devices. Each entry corresponds to a specific parameter name associated with a device and includes a measured register value alongside a timestamp. This structure enables storing time-series data that reflect the dynamic behavior of the monitored system and supports visualization, analysis, and AI-enabled processing.
- **Graph Configuration** represents the user-defined configurations for visualizing data from energy devices via drag-and-drop features. It defines which parameters from which energy devices will be monitored, the type of graph, and update intervals—thus modeling the user's monitoring capabilities.
- **User Model Config** represents the energy devices and parameters selected by users when configuring AI models. It holds information such as the model type and whether the AI training has been completed.
- **User Tables** represents custom tables created by users visually, allowing them to monitor parameter values from specific energy devices in real time. This enables personalized real-time monitoring and reporting.

- **Users** represent the registered users of the system and their roles. Each user can monitor energy devices, create tables, visualize data, and define AI models. Users are the starting point for all personal configurations and interactions.

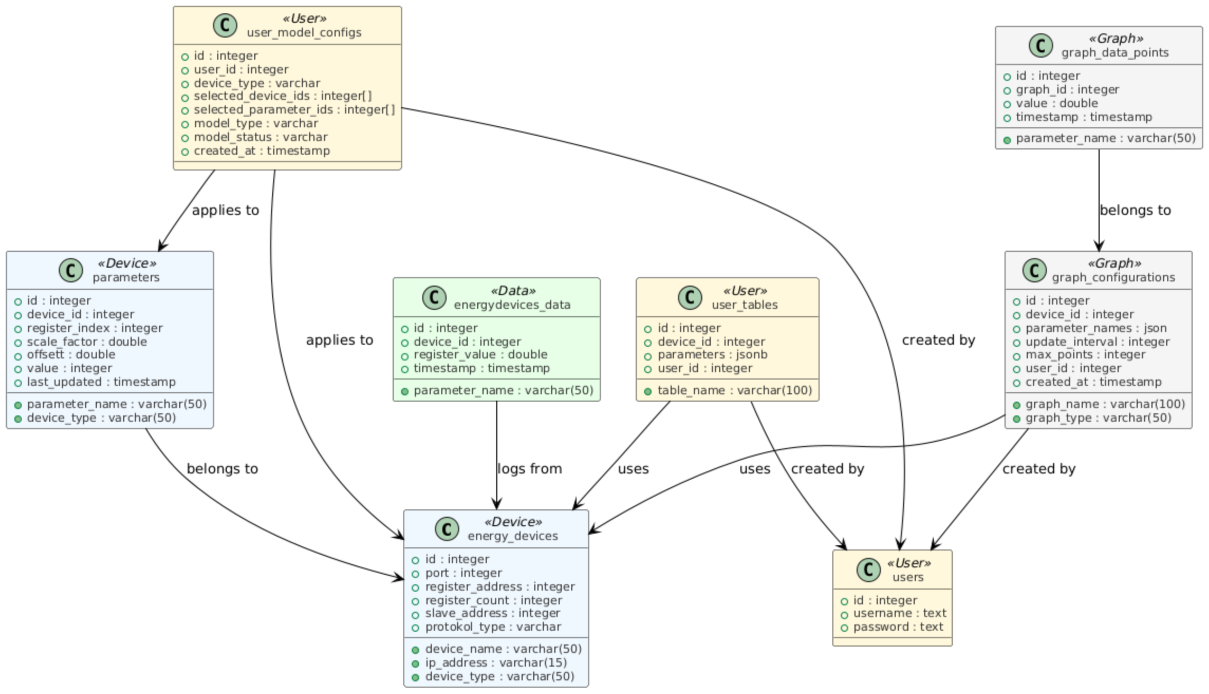


Figure 1. The platform metamodel for the AIoT-Enabled Energy Systems

3. *Concrete Syntax*

The metamodel presented in Section 2 defines the abstract syntax of the system, outlining AIoT core entities such as Devices and Users and their interrelationships. For the proper use of these abstract elements we need a concrete syntax reflecting the implementation-level counterparts of these abstract elements as is the case in various DSL, DSML and LCDP approaches (e.g. (Mernik et al., 2005); (Mohamed et al., 2020); (Celik et al., 2023)). This graphical concrete syntax includes visual and structural representations for the concepts and the relations of the proposed DSL for the MDE of AIoT-enabled energy systems.

The concrete syntax of our LCDP is composed of various viewpoints / diagram types each representing an essential concept family in a data-oriented architecture. The **Device diagram** is central to the system and focuses on real-time data collection, configuration, and modeling. It works alongside the **Data table** to store parameter information in detail (e.g., scale, offset, value, timestamp).

The **GraphConfiguration** and **GraphDataPoint** diagrams form the concrete syntax structures for visual monitoring. While GraphConfiguration defines how data is grouped and visualized, GraphDataPoint includes the actual values to be displayed—offering a practical representation of data-based monitoring.

UserModelConfig supports user-oriented modeling by defining the devices, parameters, and model types selected by each user. **UserTable** manages user-specific diagrams and related parameters.

To ensure access control and security, a **Users diagram** is also integrated into the model, supporting a secure, multi-user architecture.

This structure systematically supports the transition from abstract to concrete in accordance with MDE principles and provides a clear, practical, and extensible foundation for both developers and end-users. Table 1 lists some of the concrete syntax elements with their correponding visual notations used inside the proposed LCDP.

Table 1: A fragment from the concrete syntax of the AIoT LCDP

Concept	Notation	Concept	Notation
Device	🖥	User	👤
Parameter Definition	⚙	User Model Configuration	🤖
Parameter Value (Live)	🖵	User Table	▦
Graph Configuration	📈	Graph Data Point	📄

4. ***Online Development Environment for Modeling and Implementation of AIoT- Enabled Energy Systems***

In this study, a web-based LCDP has been designed as an integrated tool that enables the transition from abstract modeling to application-level implementation of AIoT- enabled energy systems. Based on the language specification given in Section 2 and 3 the platform facilitates MDE of AIoT-enabled energy systems and enables unified modeling of AIoT-enabled energy devices, parameters, users, visualization configurations, and AI models. Users can define system components, visually model them, and transform these models into executable software components using a drag-and-drop method, without the need for technical programming knowledge.

The integrated development environment (IDE) is entirely web-based and includes user authentication. This ensures each user's project files, models, and AI outputs are securely stored online and can be accessed from multiple devices. The interface consists of two main sections: a project navigator on the left panel, and a modeling area with drag-and-drop functionality in the center. Once the modeling process is completed, the corresponding output is automatically displayed in the same area (see Figure 2).

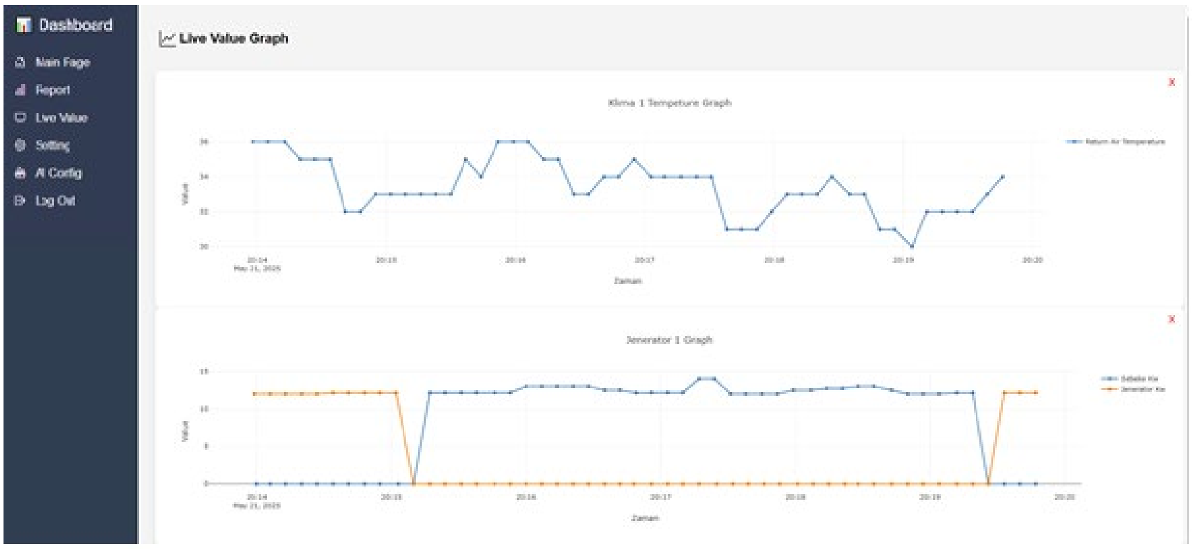


Figure 2. An overview of the IDE for the proposed LCDP

Each project is structured in a layered architecture that includes AIoT-enabled Energy devices, sensor parameters, user-specific tables, and AI configurations. For example, energy devices such as inverters, generators, and climate systems within a power plant can be defined as “Device” objects in the system, and parameters such as temperature, voltage, and current can be linked as “Parameter” objects. These parameters are normalized and monitored in real-time through the database and presented to the user via live tables.

Graph and table components can be dynamically defined in the user interface. The user can drag and drop the desired parameters into table or chart components. The chart component allows for historical data analysis, while the table component is used to report real-time values. Each user can create a personalized monitoring interface, which is saved by the system and made accessible in future sessions.

In the AI configuration screen, users can select a model type and define training data accordingly. Training data can be generated in two ways: (i) by automatically extracting previously recorded parameters within the system, or (ii) by uploading external CSV or Excel files.This enables users to handle the necessary preprocessing tasks and store the resulting model as a .pkl file on the system. The results of the selected model can also be monitored in the user interface via charts or tables.

This web-based LCDP enables the execution of data collection, visualization, modeling, and AI-driven analysis functions required for AIoT-enabled energy systems, with minimal need for software development expertise. Thanks to the model control mechanisms integrated into the platform, the establishment of inconsistent or invalid relationships between defined model components is prevented. System integrity is maintained through constraint checks, ensuring that users construct valid model structures. Moreover, by performing both data collection and AI model processing within the same environment, potential data security risks are eliminated, and the need for third-party applications is removed.

5. Case Study

Using a low-code development approach, this case study illustrates the design and implementation of an AIoT enabled energy system. It exemplifies the definition of AIoT-enabled energy devices in a mobile base station; the identification and real-time monitoring of parameters; visualization through graphs; and the creation of AI models.

In a base station, a rectifier converts AC power to DC power, a generator activates when the grid fails to ensure system continuity, and an air conditioning unit maintains operational temperature. These devices are typically found in a base station, and their number may increase based on the capacity.

Users log in to the LCDP system with a username and password, thereby preventing unauthorized access. After the user logs in, the dashboard screen is displayed, and the user can use the navigation panel on the left to perform the desired operations (Figure 3).

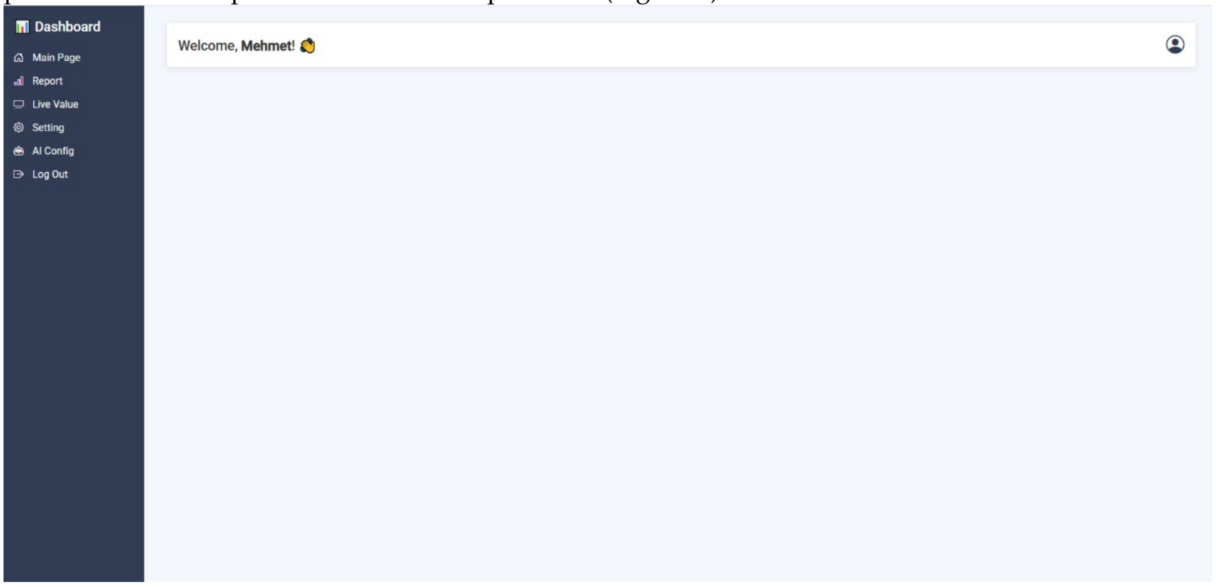
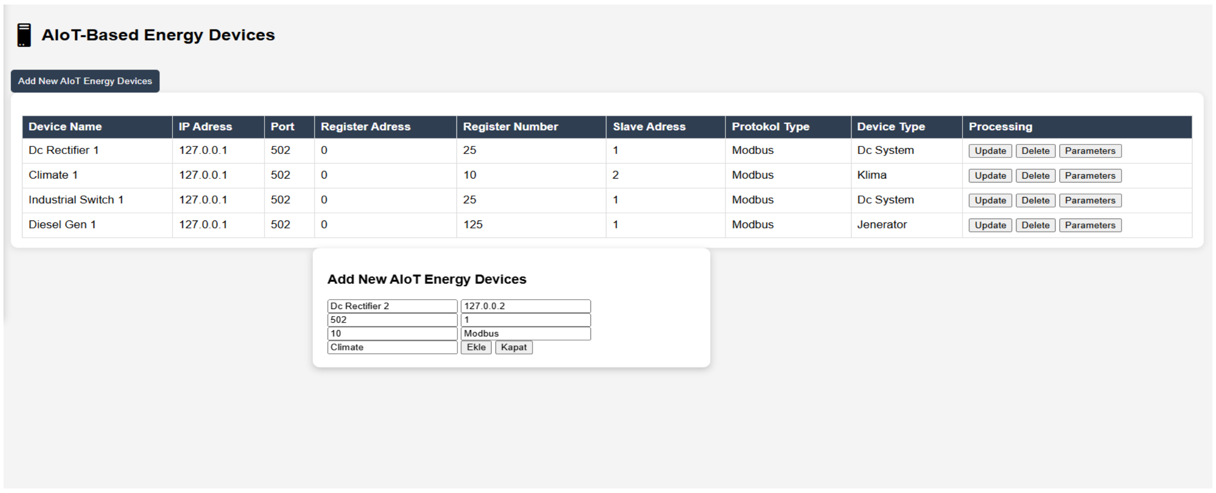


Figure 3. Main page screen for modeling the AIoT-enabled energy system

On settings page, the system responsible for making AIoT energy devices "readable" implements a device-modeling workflow. When the user clicks the "Add New AIoT Energy Device" button, a popup form appears in which they specify details such as the device name, IP address, protocol type, and other relevant parameters. Upon clicking "Save," the system automatically generates the appropriate code for that device, thereby rendering it accessible and "readable" by the platform (Figure 4).


AIoT-Based Energy Devices

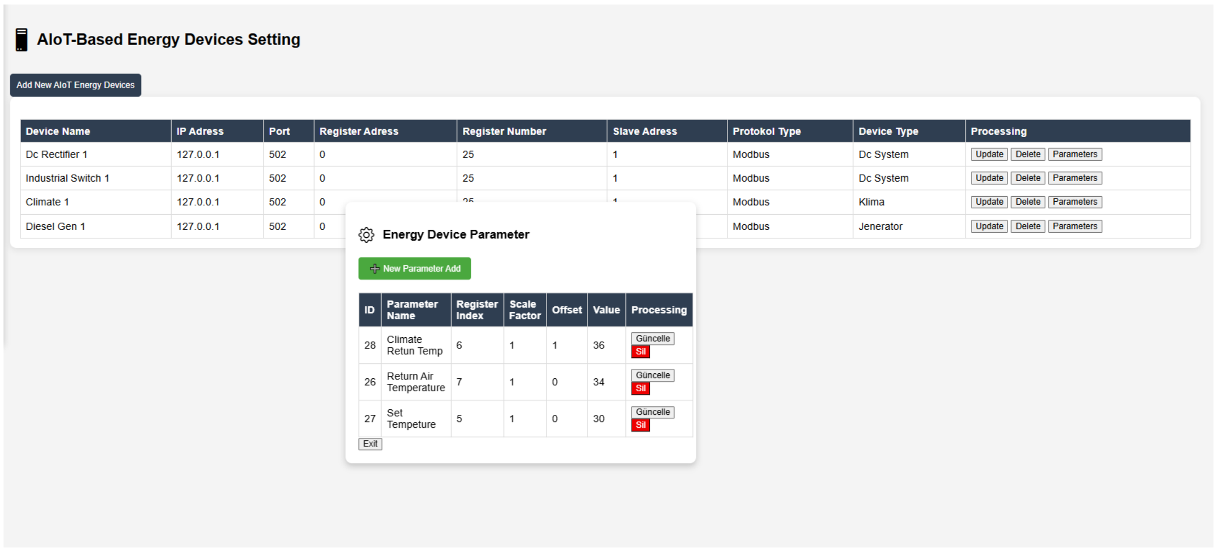
Device Name	IP Adress	Port	Register Adress	Register Number	Slave Adress	Protokol Type	Device Type	Processing
Dc Rectifier 1	127.0.0.1	502	0	25	1	Modbus	Dc System	Update Delete Parameters
Climate 1	127.0.0.1	502	0	10	2	Modbus	Klima	Update Delete Parameters
Industrial Switch 1	127.0.0.1	502	0	25	1	Modbus	Dc System	Update Delete Parameters
Diesel Gen 1	127.0.0.1	502	0	125	1	Modbus	Jenerator	Update Delete Parameters

Add New AIoT Energy Devices

Dc Rectifier 2	127.0.0.2
502	1
10	Modbus
Climate	Ekle Kapat

Figure 4. Device modeling

After the relevant energy devices have been registered, the user can click the Parameters button on the same interface to select the parameters to be accessed according to the communication protocol of the device in the popup window, or manually enter the information (e.g., parameter identifier, register index, offset value). Upon clicking the Save button, the system automatically initiates the corresponding protocol handlers and data acquisition routines, thus enabling the automatic retrieval of the specified parameters from the respective devices (Figure 5).

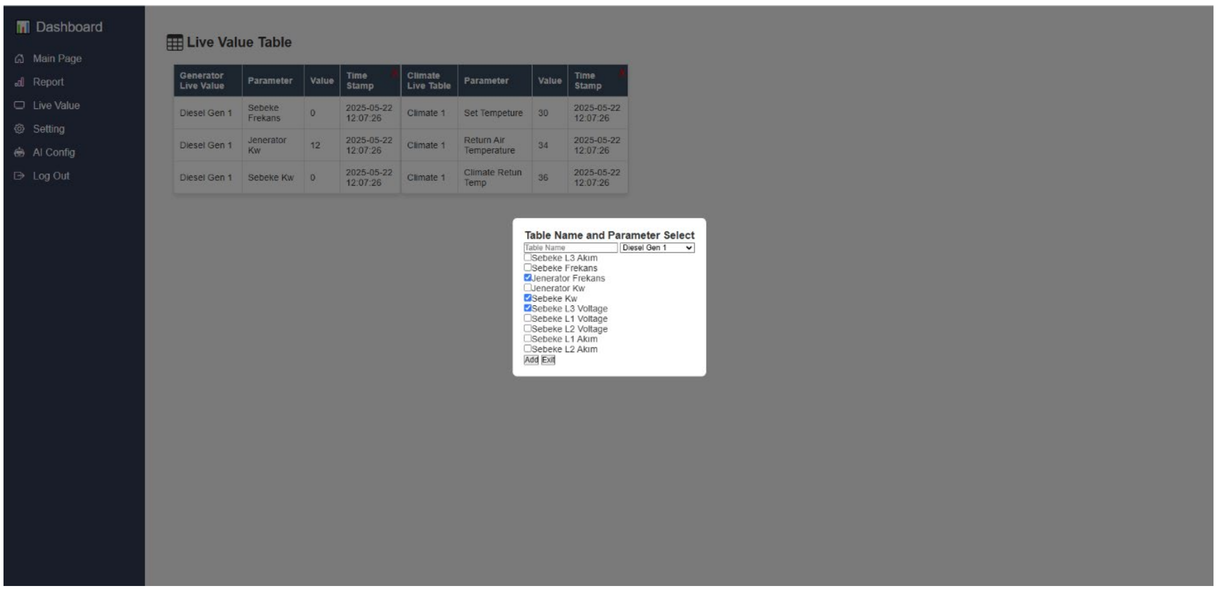

AIoT-Based Energy Devices Setting
Add New AIoT Energy Devices

Device Name	IP Adress	Port	Register Adress	Register Number	Slave Adress	Protokol Type	Device Type	Processing
Dc Rectifier 1	127.0.0.1	502	0	25	1	Modbus	Dc System	Update Delete Parameters
Industrial Switch 1	127.0.0.1	502	0	25	1	Modbus	Dc System	Update Delete Parameters
Climate 1	127.0.0.1	502	0	25	1	Modbus	Klima	Update Delete Parameters
Diesel Gen 1	127.0.0.1	502	0			Modbus	Jenerator	Update Delete Parameters

ID	Parameter Name	Register Index	Scale Factor	Offset	Value	Processing
28	Climate Retun Temp	6	1	1	36	Güncelle Sil
26	Return Air Temperature	7	1	0	34	Güncelle Sil
27	Set Tempeture	5	1	0	30	Güncelle Sil

Figure 5. Device parameter modeling

After device registration, real-time parameter monitoring is initiated by selecting Live Value Table under the Live Value section in the main menu. In this interface, the operator can model a monitoring table using a drag-and-drop method. When the table is placed in the workspace, a popup window prompts the operator to enter a unique table identifier. The operator then selects the target devices and specifies the parameters for continuous monitoring. Finally, when the Save command is activated, the configuration is completed, enabling uninterrupted real-time visualization of the selected device measurements (Figure 6).



Generator Live Value	Parameter	Value	Time Stamp	Climate Live Table	Parameter	Value	Time Stamp
Diesel Gen 1	Sebeke Frekans	0	2025-05-22 12:07:26	Climate 1	Set Tempeture	30	2025-05-22 12:07:26
Diesel Gen 1	Jenerator Kw	12	2025-05-22 12:07:26	Climate 1	Return Air Temperature	34	2025-05-22 12:07:26
Diesel Gen 1	Sebeke Kw	0	2025-05-22 12:07:26	Climate 1	Climate Retun Temp	36	2025-05-22 12:07:26

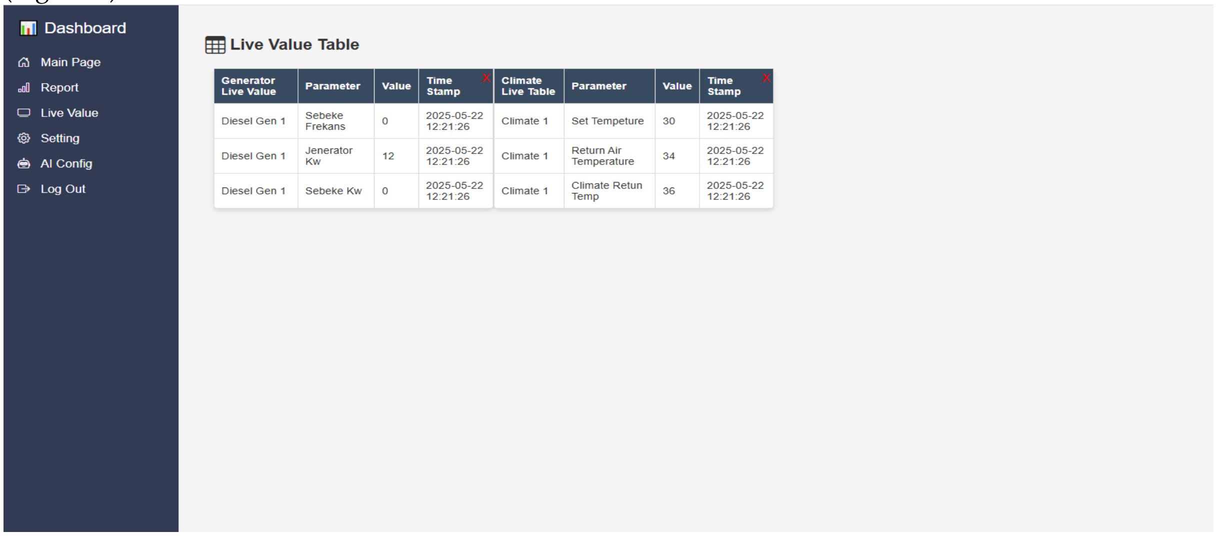
Table Name and Parameter Select

Table Name | Diesel Gen 1

- ☐ Sebeke L3 Akım
- ☐ Sebeke Frekans
- ☑ Jenerator Frekans
- ☐ Jenerator Kw
- ☑ Sebeke Kw
- ☑ Sebeke L3 Voltage
- ☐ Sebeke L1 Voltage
- ☐ Sebeke L2 Voltage
- ☐ Sebeke L1 Akım
- ☐ Sebeke L2 Akım

Figure 6. Modeling of Live Value Table

When you finalize the Live Value table configuration or access the Live Value Table interface later, the platform automatically visualizes real-time data for all pre-configured tables. In this way, the developed LCDP enables continuous, live visualization of AIoT-enabled energy device measurements (Figure 7).

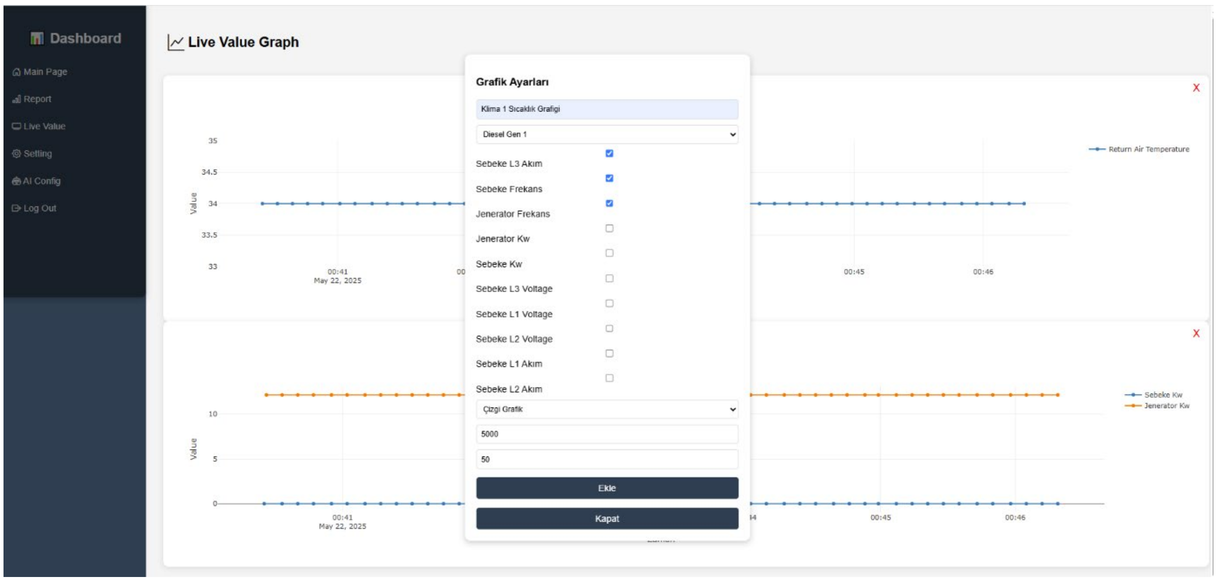


Live Value Table

Generator Live Value	Parameter	Value	Time Stamp	Climate Live Table	Parameter	Value	Time Stamp
Diesel Gen 1	Sebeke Frekans	0	2025-05-22 12:21:26	Climate 1	Set Tempeture	30	2025-05-22 12:21:26
Diesel Gen 1	Jenerator Kw	12	2025-05-22 12:21:26	Climate 1	Return Air Temperature	34	2025-05-22 12:21:26
Diesel Gen 1	Sebeke Kw	0	2025-05-22 12:21:26	Climate 1	Climate Retun Temp	36	2025-05-22 12:21:26

Figure 7. Data visualization fori Live Value Table

By navigating to the Graph page under the Live Value section in the left-hand menu, real-time parameter graphs can be created. In this interface, the operator models a graph using a drag-and-drop manner. When the graph is dragged into the workspace, a popup window prompts the operator to specify the graph name, target devices to be monitored, parameters to be observed, and the type of graph. Finally, when the Save command is activated, the configuration is saved, enabling uninterrupted, real-time visualization of the selected device measurements in a graphical form (Figure 8).



Grafik Ayarları

Klima 1 Sıcaklık Grafigi

Diesel Gen 1

- [✓] Sebeke L3 Akım
- [✓] Sebeke Frekans
- [✓] Jenerator Frekans
- [ ] Jenerator Kw
- [ ] Sebeke Kw
- [ ] Sebeke L3 Voltage
- [ ] Sebeke L1 Voltage
- [ ] Sebeke L2 Voltage
- [ ] Sebeke L1 Akım
- [ ] Sebeke L2 Akım

Çizgi Grafik

5000

50

Figure 8. Graph modeling for Live Value

Once the Live Value Graph configuration is completed or the Graph interface under the Live Value section is accessed later, the platform automatically plots real-time data for all previously defined graphs. In this way, the LCDP enables uninterrupted and real-time graphical visualization of measurements from AIoT-enabled energy devices (Figure 9).

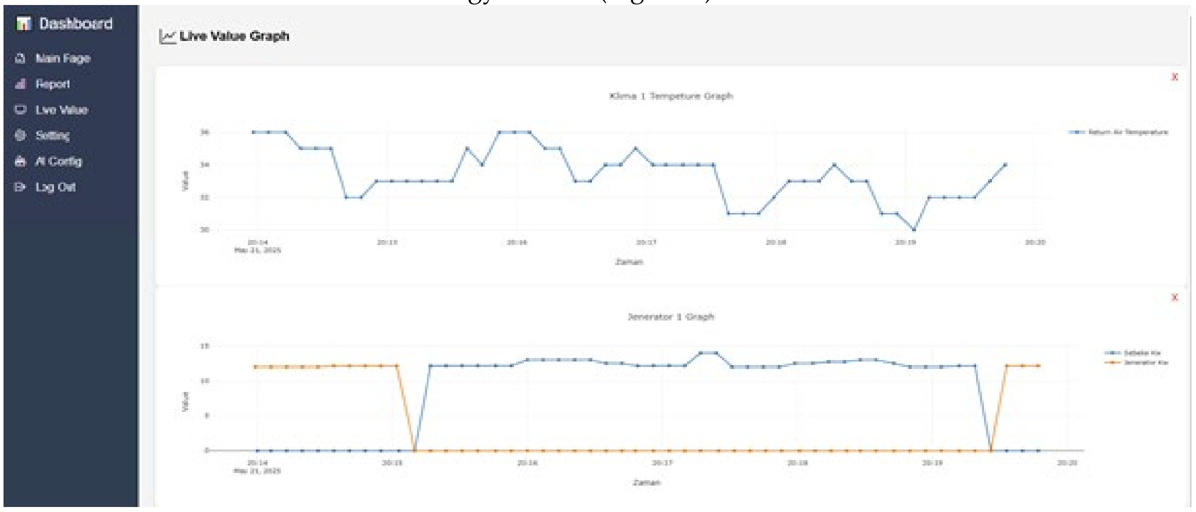
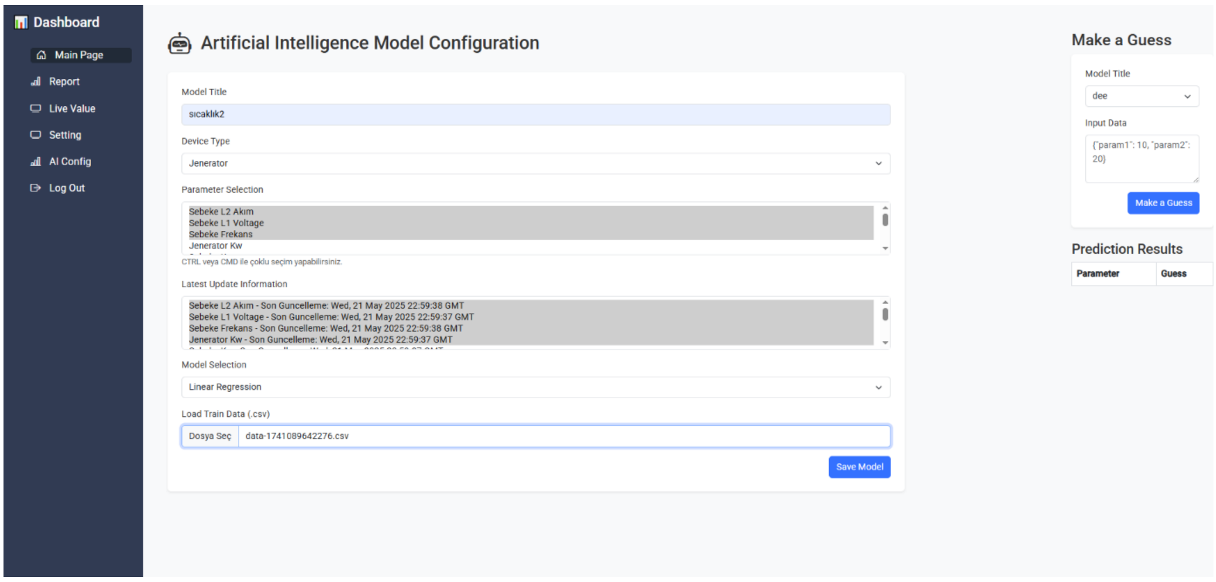


Figure 9. Live Value Graph Visualization

Artificial intelligence (AI) model configurations are defined via a web-based interface in order to minimize dependency on developers. Users specify the model name, device type, and the parameters to be used. Training data can be retrieved directly from the database or imported via an Excel file. This enables data cleaning and correction when necessary. The trained model is saved on the server as a "**.pkl"** file without requiring any coding. The selected model, corresponding parameters, and AI outputs are visualized and presented to the user through the web interface (Figure 10).



Artificial Intelligence Model Configuration

Model Title: sıcaklık2

Device Type: Jenerator

Parameter Selection:
Sebeke L2 Akım
Sebeke L1 Voltage
Sebeke Frekans
Jenerator Kw

CTRL veya CMD ile çoklu seçim yapabilirsiniz.

Latest Update Information:
Sebeke L2 Akım - Son Guncelleme: Wed, 21 May 2025 22:59:38 GMT
Sebeke L1 Voltage - Son Guncelleme: Wed, 21 May 2025 22:59:37 GMT
Sebeke Frekans - Son Guncelleme: Wed, 21 May 2025 22:59:38 GMT
Jenerator Kw - Son Guncelleme: Wed, 21 May 2025 22:59:37 GMT

Model Selection: Linear Regression

Load Train Data (.csv): Dosya Seç data-1741089642276.csv

Make a Guess

Model Title: dee

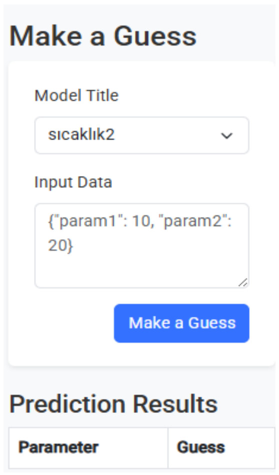
Input Data: {"param1": 10, "param2": 20}

Prediction Results

Parameter	Guess

Figure 10. Creating AI Models

The trained model, using the parameters selected during the training phase as inputs, enables the display of prediction outputs within the Live Value Table (Figure 6) and Live Value Graph (Figure 8) interfaces. Additionally, the platform provides a Make a Guess section, allowing users to test the trained model by selecting it and entering appropriate input parameters, thus facilitating model validation (Figure 11).


Make a Guess

Model Title: sıcaklık2

Input Data: {"param1": 10, "param2": 20}

Prediction Results

Parameter	Guess

Figure 11. AI Model "Make a Guess"

In this way, all AIoT-enabled energy devices in a base station have been monitored in real time with minimal coding; their data has been visualized and reported through tables and graphs, and modeled using AI models.

6. Conclusion

In this study, an MDE approach is proposed by integrating DSML and LCDP technologies for the development of AIoT-enabled energy systems. This integrated approach enables system designers to graphically model the structural and behavioral components of smart energy systems, while also supporting the integration of AI and IoT infrastructure.

The proposed DSML with the supporting LCDP allows for the platform-independent representation of physical components, data collection processes, and AI-enabled decision-making mechanisms through a metamodel and graphical syntax. Moreover, with the developed LCDP prototype, it is possible to generate executable code directly from the models, providing significant time savings during rapid prototyping and development phases.

To the best of our knowledge, this study is among the first to address low-code and model-driven development approaches in the context of AIoT-enabled energy systems. Although the syntax and semantics definitions for the proposed DSML and its LCDP are discussed in this paper, the systematic assessment of the usability of these language elements and the provided IDE needs to be performed. Within this context, our future work will evaluate the promising features of this LCDP by means of artifact generation and time savings according to a well-defined set of DSL criteria (Kahraman & Bilgen, 2015) and language evaluation methodology previously we applied in similar systematic DSL, DSML and MDE evaluation studies (e.g. Marah et al., 2021; Leblebici et al., 2022; Arslan et al., 2024).

References

Arivukkody, V., Gokulakannan, T., & Kalpana, S. (2022). AIoT based residential smart energy meter with power saving methodology. 2022 1st International Conference on Computational Science and Technology (ICCST), 80–85.

Arslan, S., Kardas, G., & Alfraihi, H. (2024). On the Usability of a Modeling Language for IoT-Based Public Transportation Systems. Applied Sciences, 14(13), 5619.

Arslan, S., Ozkaya, M., & Kardas, G. (2023). Modeling languages for internet of things (IoT) applications: A comparative analysis study. Mathematics, 11(5), 1263.

Brambilla, M., Cabot, J., & Wimmer, M. (2017). Model-driven software engineering in practice. Morgan & Claypool Publishers.

Cabot, J. (2020). Positioning of the low-code movement within the field of model-driven engineering. Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings, 1–3.

Celik, B., Kardas, G. and Tezel, B. T. (2023) "An Online Modeling Language for the Low-code Development of Belief-Desire-Intention (BDI) Agents", In proceedings of the 2nd International Symposium Series on Graduate Researches (ISGR 2023), November 30, 2023, Izmir, Turkey, pp. 186-196

Di Ruscio, D., Kolovos, D., de Lara, J., Pierantonio, A., Tisi, M., & Wimmer, M. (2022). Low-code development and model-driven engineering: Two sides of the same coin? Software and Systems Modeling, 21(2), 437–446.

Jnr, B. A. (2024). Decentralized AIoT based intelligence for sustainable energy prosumption in local energy communities: A citizen-centric prosumer approach. Cities, 152, 105198.

Kahraman, G., & Bilgen, S. (2015). A framework for qualitative assessment of domain-specific languages. Software & Systems Modeling, 14, 1505–1526.

Kardas, G., Ciccozzi, F., & Iovino, L. (2023). Introduction to the special issue on methods, tools and languages for model-driven engineering and low-code development. Journal of Computer Languages, 74.

Leblebici, O., Kardas, G., & Tuglular, T. (2022). A domain-specific language for the document-based model-driven engineering of business applications. IEEE Access, 10, 104093–104110.

Madakam, S., Ramaswamy, R., & Tripathi, S. (2015). Internet of Things (IoT): A literature review. Journal of Computer and Communications, 3(5), 164–173.

Marah, H., Kardas, G., & Challenger, M. (2021). Model-driven round-trip engineering for TinyOS-based WSN applications. Journal of Computer Languages, 65, 101051.

Matin, A., Islam, M. R., Wang, X., Huo, H., & Xu, G. (2023). AIoT for sustainable manufacturing: Overview, challenges, and opportunities. Internet of Things, 24, 100901.

Mernik, M., Heering, J., & Sloane, A. M. (2005). When and how to develop domain-specific languages. ACM Computing Surveys (CSUR), 37(4), 316–344.

Mohamed, M. A., Challenger, M., & Kardas, G. (2020). Applications of model-driven engineering in cyber-physical systems: A systematic mapping study. Journal of Computer Languages, 59, 100972.

Saritas, H. B., & Kardas, G. (2014). A model driven architecture for the development of smart card software. Computer Languages, Systems & Structures, 40(2), 53–72.

Yang, C.-T., Chen, H.-W., Chang, E.-J., Kristiani, E., Nguyen, K. L. P., & Chang, J.-S. (2021). Current advances and future challenges of AIoT applications in particulate matters (PM) monitoring and control. Journal of Hazardous Materials, 419, 126442.