https://doi.org/10.51707/2618-0529-2026-35-09

УДК 004.89:004.738.5:681.5

S. V. Sulima,
L. S. Globa,
M. S. Tkachenko,
Ya. V. Savchenko

AI-POWERED SMART HOME SYSTEMS AS AN EDUCATIONAL PLATFORM BASED ON NLP AND IOT TECHNOLOGIES

Abstract. *This paper presents an approach to designing a locally deployed smart home control system as an educational platform based on the integration of Artificial Intelligence (AI), Natural Language Processing (NLP), and Internet of Things (IoT) technologies. The relevance of the study is driven by the widespread reliance of modern smart home solutions on cloud-based platforms, which results in increased response latency, limited offline functionality, higher security risks, and insufficient support for less common natural languages, including Ukrainian, as well as the growing need for practical educational platforms that enable students to gain hands-on experience with AI-driven automation systems and intelligent interfaces. The primary objective is to improve the reliability, responsiveness, and privacy of smart home systems while developing a versatile educational tool for teaching modern technologies in intelligent systems design and natural language processing. The proposed system is built on the open-source Home Assistant platform and the MQTT messaging protocol, enabling low-latency message delivery, reduced network overhead, and stable operation in environments with limited internet connectivity. Special attention is given to system security through MQTT broker configuration with authentication mechanisms and encrypted credentials. User interaction is enhanced through an AI-driven NLP module implemented in Python using the spaCy library and the Pymorphy3 morphological analyzer. The module performs text normalization, tokenization, lemmatization, and morphological analysis of Ukrainian-language commands, allowing accurate extraction of user intent and target devices. A Telegram bot provides a cross-platform user interface without additional client-side software. The modular architecture allows students to modify individual components, experiment with NLP models, and develop custom extensions, making the platform suitable for laboratory work and research activities. Experimental validation demonstrates reduced command execution latency of 150–200 milliseconds compared to cloud-dependent solutions (500–800 milliseconds). The NLP module achieved accuracy rates of 94% i for single-device commands, 91% i for multi-device commands, and 87% i for complex conditional commands. The results indicate that the system represents both a viable alternative to proprietary cloud-based platforms and an effective educational platform for developing professional competencies in AI-driven automation technologies.*

Keywords: *education technologies, educational platform, smart home, artificial intelligence, IoT.*

Introduction. Smart homes integrate IoT devices, allowing users to control various functions of their homes through centralized systems [1]. The concept of a smart home has evolved significantly, driven by advancements in artificial intelligence (AI) and the Internet of Things (IoT). Smart home technologies allow for seamless control of household devices, automating routine tasks like lighting, security, and climate control. Technologies like Google Home [2], Apple HomeKit [3], and Amazon Alexa [4] have popularized this field, offering cloud-based services for device

control. However, these systems face challenges, particularly when relying on cloud-based services. Recent developments in Natural Language Processing (NLP) provide new possibilities for enhancing user interaction, increasing system reliability by allowing local control, and reducing dependence on external networks.

In addition to practical smart home applications, modern AI- and IoT-based systems are increasingly considered as educational tools for training specialists in information technologies. Higher education institutions face the challenge of providing students with practice-oriented environments that integrate artificial intelligence, Internet of Things technologies, distributed systems, and cybersecurity. Traditional laboratory setups are often expensive, inflexible, and poorly scalable, which limits students' hands-on experience with modern intelligent systems. Therefore, there is a need for educational platforms that combine real-world IoT infrastructure with AI-driven interfaces and can be deployed locally within university laboratories.

The proposed system addresses this gap by serving dual purposes: (1) as a functional smart home control solution with enhanced privacy and responsiveness, and (2) as a comprehensive educational platform enabling hands-on learning in AI, NLP, and IoT integration. The modular architecture allows students to progressively develop competencies from basic system interaction to advanced custom integration development, while the use of Ukrainian language processing demonstrates NLP adaptation methodologies applicable to other low-resource languages in educational contexts.

AI and NLP play crucial roles in improving user interactions with smart home systems. Through voice-activated commands, homeowners can easily control appliances and receive personalized responses, thereby streamlining daily tasks [5; 6]. AI algorithms further enhance the capabilities of smart devices by allowing them to learn from user behavior, predict preferences, and even proactively address potential security threats by analyzing patterns in real-time data [7]. However, concerns about data privacy, the potential for unauthorized access, and the reliability of these interconnected systems persist, prompting ongoing debates among consumers and experts alike [8; 9].

The IoT serves as the backbone of smart home technology, interconnecting various devices to create an integrated ecosystem that continuously collects and shares data [10]. This interconnectedness not only leads to increased efficiency in home management but also raises challenges regarding device compatibility and interoperability, as different manufacturers often utilize proprietary procols [11]. As the number of connected devices is projected to reach upwards of 40 billion by 2030, establishing robust security measures and ensuring seamless communication between devices will be crucial for protecting user privacy and enhancing the overall smart home experience [12].

Nevertheless, it also necessitates careful consideration of privacy concerns and interoperability challenges, as the smart home ecosystem continues to evolve in response to technological advancements and consumer needs [13; 14]. Addressing these weaknesses through localized systems is crucial. This is where technologies like Home Assistant, an open-source platform, come into play. Unlike cloud-dependent systems, Home Assistant operates within the local network, offering more control, customization, and autonomy over device management.

At the same time, most existing studies and commercial platforms focus on end-user applications, while the educational potential of smart home systems as laboratory and project-based learning environments for students in information technologies remains insufficiently explored. In addition to practical smart home applications, the proposed system can be considered as an educational and research platform for training students in information technologies, telecommunications, and computer engineering. The integration of AI, Natural Language Processing, and IoT within a locally deployed architecture enables hands-on learning of modern educational technologies, including distributed systems, intelligent interfaces, and secure data communication. Such an approach supports practice-oriented education and the development of professional competencies in AI-driven automation systems.

Despite significant progress in smart home automation and AI-driven interfaces, several unresolved problems persist in the educational domain. These include the lack of accessible local platforms for teaching AI, NLP, and IoT integration; limited opportunities for students to experiment with real-time device control and natural language interfaces; and insufficient support

for native-language interaction in educational laboratories. Addressing these gaps is essential for developing effective educational technologies that support hands-on learning and competency-based training.

The aim of this article is to develop and evaluate a locally deployed AI-powered smart home system that serves both as a functional control solution and as an educational platform for teaching information technologies, artificial intelligence, natural language processing, and Internet of Things concepts.

To achieve this aim, the following objectives were defined:

- to design a local smart home architecture based on Home Assistant and MQTT suitable for educational laboratories;
- to implement an NLP module for processing Ukrainian-language commands as a case study for AI-based user interaction;
- to integrate a Telegram-based interface for demonstrating client-server communication and distributed system principles;
- to evaluate the system's performance and demonstrate its applicability in educational and training scenarios.

Enhancing interaction through NLP and AI. A key element in ensuring a robust and efficient smart home is a reliable communication protocol. MQTT, a lightweight protocol designed for resource-constrained devices and low-bandwidth networks, serves this purpose well. In the modified smart home system proposed in this work, MQTT facilitates communication between devices and the server, reducing the latency typically encountered in cloud-based solutions.

Natural Language Processing (NLP) has transformed how users interact with technology. In smart homes, NLP enables users to control devices through voice commands, enhancing usability and accessibility. Integrating AI-driven NLP into smart home management systems allows commands to be processed efficiently in multiple languages, including Ukrainian.

Local processing of voice commands offers several advantages over cloud-based solutions:

- Reduced Latency: Since the system does not need to send commands to remote servers, the response time is significantly faster.
- Enhanced Security: Localized systems do not require internet connectivity for basic operations, reducing the risk of external hacking or data breaches.
- Privacy: Data is processed locally, ensuring that sensitive voice data does not leave the home network, a concern with many cloud-based systems.

However, challenges remain, particularly in ensuring interoperability between different brands and devices. The shift towards open-source platforms like Home Assistant, combined with the flexibility of MQTT, points towards a future where users have complete control over their smart environments, free from the constraints of proprietary cloud services.

The proposed smart home management system is built upon a three-tier architecture designed to ensure low-latency communication, enhanced security, and independence from cloud-based services.

The system integrates three primary components: 1) Home Assistant as the central device management platform, 2) Eclipse Mosquitto as the MQTT message broker, and 3) a custom NLP processing module for command interpretation. This architecture was selected based on the following criteria:

- Open-source availability: All components utilize open-source software, enabling students to examine source code, understand implementation details, and develop custom modifications without licensing restrictions.
- Local deployment capability: The entire system operates within the local network, eliminating dependency on external cloud services and reducing communication latency.
- Educational accessibility: The modular design allows instructors to demonstrate individual components separately before integrating them into the complete system.

The Message Queuing Telemetry Transport (MQTT) protocol serves as the communication backbone between system components. MQTT was selected over alternative protocols (HTTP, CoAP, WebSocket) due to its lightweight nature, publish-subscribe model, and suitability for resource-constrained IoT devices.

The MQTT broker configuration implements several security mechanisms:

1. Authentication enforcement: Anonymous connections are disabled; all clients must authenticate using encrypted credentials.

ISSN 2786-4510 (Online)

2. Password encryption: User credentials are stored using industry-standard hashing algorithms, preventing plaintext password exposure.

3. Topic-based access control: Different system components operate on designated topics, enabling fine-grained permission management.

4. Port isolation: The broker listens on a dedicated port (default: 1883) isolated from other network services.

The publish-subscribe model *(Fig. 1)* facilitates loose coupling between system components, allowing students to understand distributed system architecture principles. Publishers (Telegram bot, automation triggers) send messages to specific topics without direct knowledge of subscribers (device controllers), while the broker handles message routing and delivery guarantees.

Home Assistant acts as the main platform for managing devices, offering unified interfaces for different IoT devices. The platform runs in a Docker container. This setup keeps the system separate from the host operating system, making it easier to deploy in various lab environments. Container images allow for identical deployments, giving students the same experience across different lab sessions. Containerization also enables precise control of CPU and memory resources, letting multiple students work on shared hardware simultaneously.

The Home Assistant configuration enables MQTT integration through native protocol support, allowing device state changes to be triggered by incoming MQTT messages. This integration demonstrates the concept of protocol bridging, where the platform translates between device-specific protocols (Zigbee, Z-Wave, Wi-Fi) and the unified MQTT interface.

The system implements multiple security layers appropriate for educational environments while maintaining real-world security standards:

- Network isolation: All communication occurs within the local network, preventing external access to IoT devices and reducing attack surface;
- Credential management: Authentication credentials are stored in encrypted form and loaded at runtime, demonstrating secure configuration management practices;
- Message encryption: While MQTT operates over TCP by default, the architecture supports TLS/SSL encryption for sensitive deployments;
- Access logging: All connection attempts and message transactions are logged, enabling students to analyze system behavior and detect anomalous activity.

This security-conscious design allows instructors to teach cybersecurity concepts within the context of IoT systems, including authentication mechanisms, encryption protocols, and intrusion detection principles.

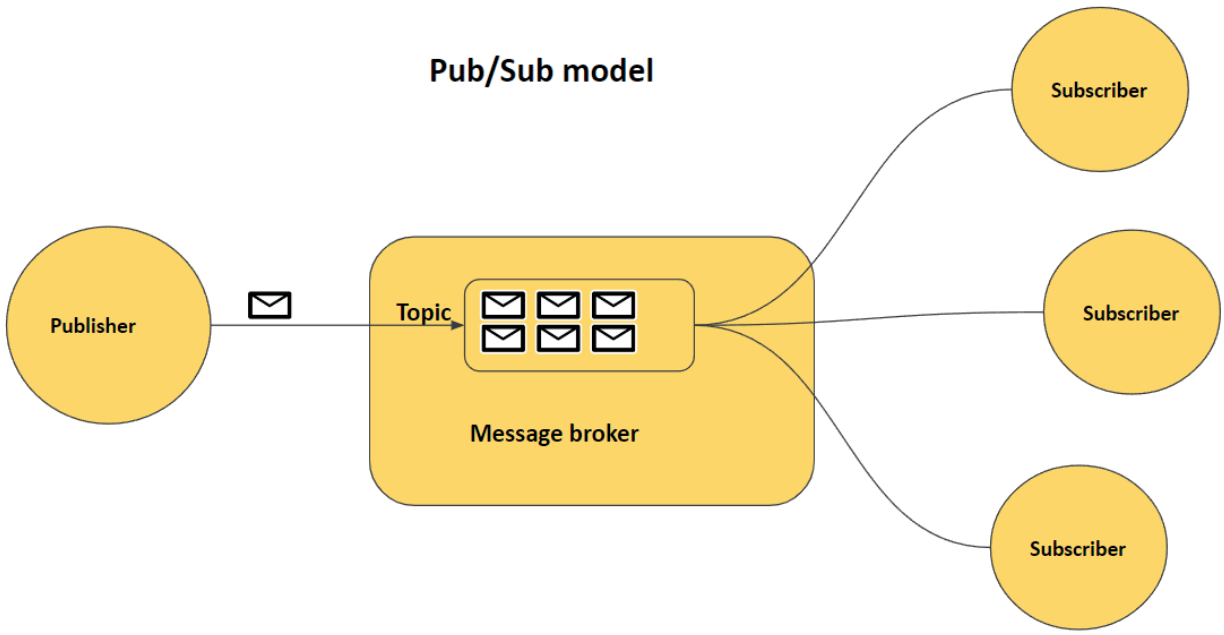

Fig. 1. Publisher-subscriber model

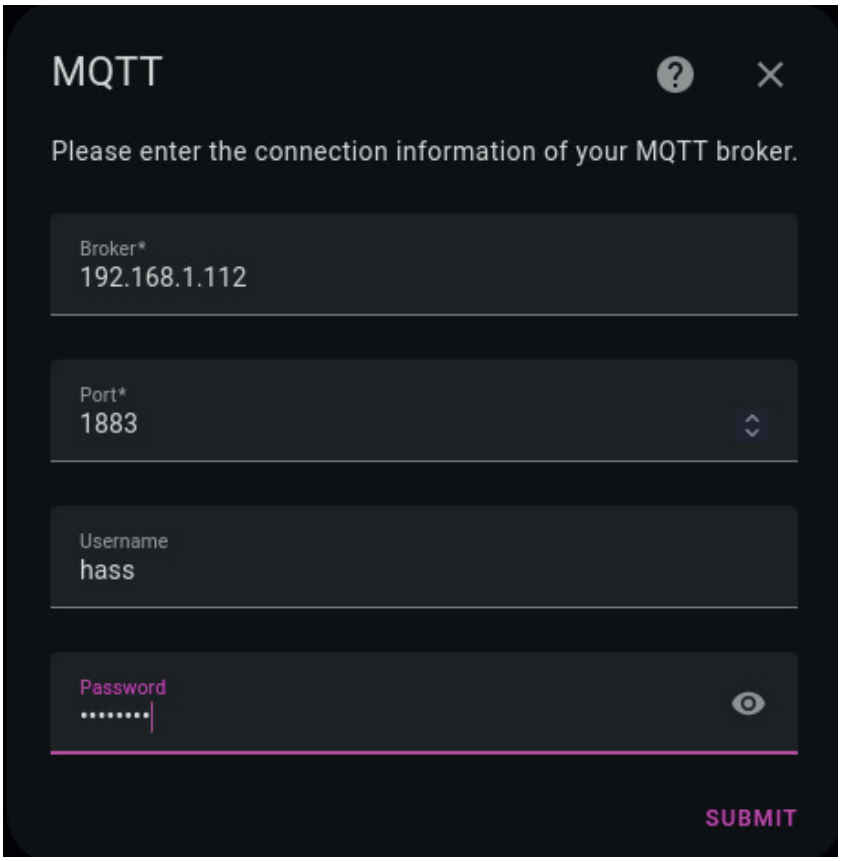

Fig. 2. Connecting to the MQTT broker

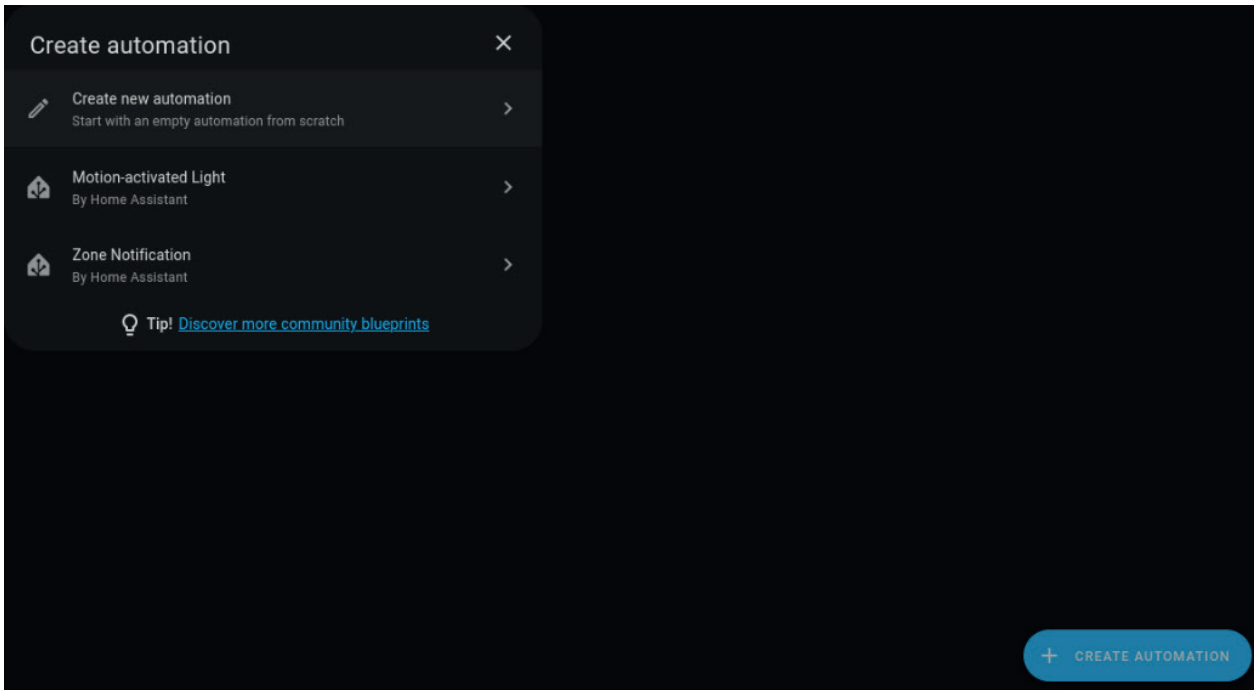

Fig. 3. Creating a new automation using the “Create Automation” button

ISSN 2786-4510 (Online)

To successfully connect to the broker, you need to enter the broker's IP address, port, login and password *(Fig. 2).*

In order to change the state of these devices, you should create and use automation triggers. Automation triggers in HASS can be created both through the web interface *(Fig. 3)* and by entering information about the desired trigger conditions in the configuration file configuration.yaml.

Since at the moment there is already a process of interacting with the smart home device management system using MQTT protocol messages, it makes sense to create a software module that could send MQTT messages. Eclipse Paho MQTT is an open source library that provides a client-side implementation of the MQTT protocol. This library provides an API for using a Python program as an MQTT client.

Since the devices in the system are managed using MQTT messages, our program code should send messages to a given topic with a given load *(Fig. 4).*

In the created class, we call the connect method to connect to the broker. In the case of a successful connection, the Boolean variable connected will be True and it will be possible to send a message using the publish method to the specified topic and with the specified load. The start method will start the message processing process, this should be done once before sending the message. If it is impossible to connect to the broker, an exception will be

```python
import paho.mqtt.client as mqtt


3 usages
class MqttClientHandler:
    def __init__(self, client_id, broker_address, username, password):
        self.client = mqtt.Client(client_id=client_id)
        self.broker_address = broker_address
        self.client.username_pw_set(username, password)
        self.connected = False
        self.client.on_disconnect = self.on_disconnect

    1 usage
    def on_disconnect(self, client, userdata, flags, rc):
        self.connected = False

    2 usages
    def connect(self):
        try:
            self.client.connect(self.broker_address)
            self.connected = True
        except:
            self.connected = False

    2 usages
    def start(self):
        self.client.loop_start()

    1 usage
    def publish(self, topic, payload):
        self.client.publish(topic, payload)

    4 usages
    def is_connected(self):
        return self.connected
```

Fig. 4. Program class for sending MQTT messages

thrown, which will be processed and the boolean variable connected will be False. The on_disconnect method will be called automatically if the connection between the broker and the client is lost, it will also reset the boolean variable connected to False.

Natural Language Processing (NLP) is a subfield of artificial intelligence that uses machine learning to interact between computers and human language [15].

The NLP module represents the intelligent interface layer, translating user commands in Ukrainian language into structured device control messages. The module employs a multi-stage processing pipeline that demonstrates fundamental NLP concepts while maintaining practical applicability.

The command interpretation process follows six sequential stages *(Fig. 5):*

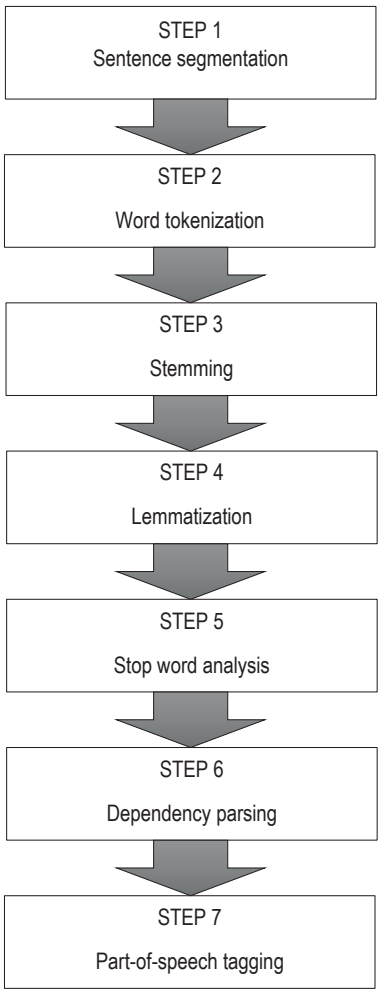

Fig. 5. Stages of natural language processing in NLP

1. Text segmentation: Dividing input text into sentence boundaries, handling common Ukrainian punctuation patterns and command structures.

2. Tokenization: Breaking sentences into individual tokens (words, punctuation marks), accounting for Ukrainian morphological complexity.

3. Lemmatization: Reducing inflected word forms to their dictionary forms (lemmas), essential for Ukrainian due to its rich inflectional morphology. For example, “світло” (light-nominative), “світла” (light-genitive), and “світлом” (light-instrumental) all reduce to the base form “світло”.

4. Morphological analysis: Identifying grammatical characteristics including part of speech, case, gender, number, and tense. This stage utilizes the Pymorphy3 library, which provides comprehensive morphological dictionaries for Ukrainian.

5. Syntactic parsing: Analyzing grammatical structure to identify subject- predicate relationships and dependency trees, enabling understanding of command structure.

6. Semantic analysis: Extracting user intent (action to perform) and target entities (devices to control) from the parsed structure.

The NLP module integrates two primary libraries:

1. SpaCy Framework: A production-grade NLP library providing pre-trained models for Ukrainian language processing. SpaCy offers efficient processing pipelines optimized for performance while maintaining accuracy. The framework’s modular architecture allows students to examine individual pipeline components and understand their contributions to overall accuracy.

2. Pymorphy3 Morphological Analyzer: A specialized library for morphological analysis of Slavic languages. Pymorphy3 provides detailed morphological parsing capabilities exceeding SpaCy’s base functionality, particularly for handling Ukrainian case system and verb conjugations. The library’s dictionary-based approach enables deterministic analysis, making it suitable for educational demonstrations where students can trace exact processing steps.

The core functionality of the NLP module involves mapping natural language commands to device control operations. This mapping employs a rule-based approach enhanced with linguistic normalization:

1. Action Recognition: The system maintains a dictionary of action verbs and their synonyms in Ukrainian. For example, the action

ISSN 2786-4510 (Online)

"activate" can be expressed as "увімкни" (turn on), "включи" (switch on), "запусти" (start), or "активуй" (activate). All variants are lemmatized to a canonical form, enabling robust recognition despite morphological variations.

2. Device Identification: Device names are matched against a configurable list through fuzzy string matching with lemmatization. This approach handles diminutive forms (common in Ukrainian), spelling variations, and partial matches. For instance, "світло" (light), "світлик" (little light), and "освітлення" (lighting) can all map to the same device entity.

3. Multi-Device Commands: The module supports commands targeting multiple devices simultaneously through conjunction analysis. Commands like "увімкни світло і телевізор" (turn on the light and TV) are parsed to identify coordinated noun phrases, extracting multiple device targets from a single command.

4. Conditional Logic: Advanced command structures incorporating conditional clauses are handled through dependency parsing. For example, "якщо температура понад 25 градусів, увімкни кондиціонер" (if temperature exceeds 25 degrees, turn on air conditioner) requires identifying conditional markers, extracting threshold values, and mapping them to automation triggers.

The NLP module is implemented as a Python class with the following key methods:

- normalize_text (): Performs case normalization, punctuation handling, and whitespace standardization to prepare input for processing;
- lemmatize_text (): Applies lemmatization to reduce words to base forms, utilizing both SpaCy and Pymorphy3 for comprehensive coverage;
- extract_intent (): Identifies the primary action verb and maps it to a system operation (activate, deactivate, toggle, query);
- identify_devices (): Extracts device entities through named entity recognition and fuzzy matching against the device registry;
- parse_conditions (): Analyzes conditional structures and extracts parameters for trigger-based automation.

This modular design lets students change individual processing stages, test different algorithms, and measure their effect on accuracy. Students can modify the lemmatization strategy to compare dictionary-based and statistical approaches. They can expand the action synonym dictionary to recognize more commands. Students can create custom entity recognition rules for new device types. They can also develop context-aware processing to handle unclear references.

The NLP module serves multiple pedagogical purposes:

1. Algorithm Understanding: Students can trace command processing step-by-step, observing how raw text transforms into structured data.

2. Performance Optimization: By measuring processing time for each pipeline stage, students learn to identify performance bottlenecks and apply optimization techniques.

3. Accuracy Analysis: Students can generate confusion matrices, analyze misclassification patterns, and develop targeted improvements.

4. Language Adaptation: The Ukrainian-focused implementation demonstrates NLP adaptation methodologies applicable to other languages, particularly those with limited pre-trained model availability.

The Telegram messaging platform serves as the primary user interface, providing cross-platform accessibility without requiring custom mobile application development.

The bot operates as a bridge between users and the smart home system, implementing the following workflow *(Fig. 6):*

1. User submits natural language command via Telegram client (mobile, desktop, or web interface).

2. Telegram Bot receives the message through Telegram Bot API, validates user authorization, and forwards the command text to the NLP processing module.

3. NLP Module analyzes the command, extracts intent and target devices, and returns structured action data to the bot.

4. MQTT Publisher component constructs appropriate MQTT messages with correct topic paths and payloads, then publishes them to the broker.

5. MQTT Broker receives published messages and distributes them to subscribed components (Home Assistant) based on topic matching.

6. Home Assistant processes received MQTT messages, translates them into device-specific protocols, and executes state changes on target IoT devices.

7. Smart Home Devices respond to commands by changing operational states (turning on/off, adjusting settings).

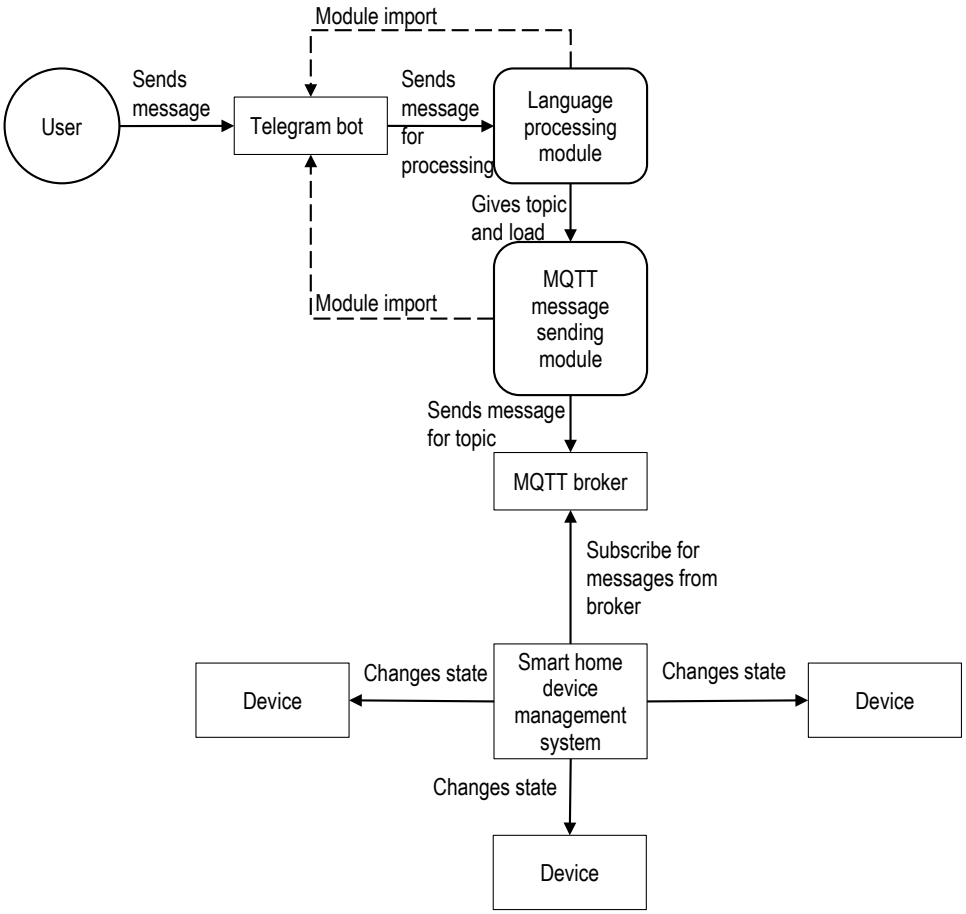

Fig. 6. Structural diagram of a modified smart home device control system

8. Home Assistant monitors device state changes and publishes confirmation messages back through MQTT, enabling status feedback to users.

This architecture demonstrates several key concepts in distributed systems through its operational characteristics — Fig. 7.

From a pedagogical perspective, the Telegram bot interface provides several learning opportunities:

1. API Integration: Students learn to interact with third-party APIs (Telegram Bot API), handling authentication, webhook configuration, and message parsing.

2. Error Handling: The bot implements comprehensive error handling for network failures, invalid commands, and device unavailability, demonstrating robust software design principles.

3. User Experience Design: Students can modify bot responses, implement command suggestions, and add interactive buttons, exploring human-computer interaction concepts.

4. Security Considerations: The bot implements user authorization checks, rate limiting, and input validation, teaching security-conscious development practices.

The modular architecture allows students to develop alternative interfaces as project assignments: voice interface using speech-to-text services (Google Speech API, Mozilla DeepSpeech); web dashboard using Flask or Django frameworks; mobile applications using React Native or Flutter; smart speaker integration (simulated Alexa or Google Home).

These extensions serve as capstone projects where students integrate multiple technologies while maintaining compatibility with the existing MQTT-based backend.

Discussions and results. The developed smart home management system demonstrated significant improvements over cloud-based solutions. Local MQTT processing reduced average response times to 150–200 milliseconds, represent-

ISSN 2786-4510 (Online)

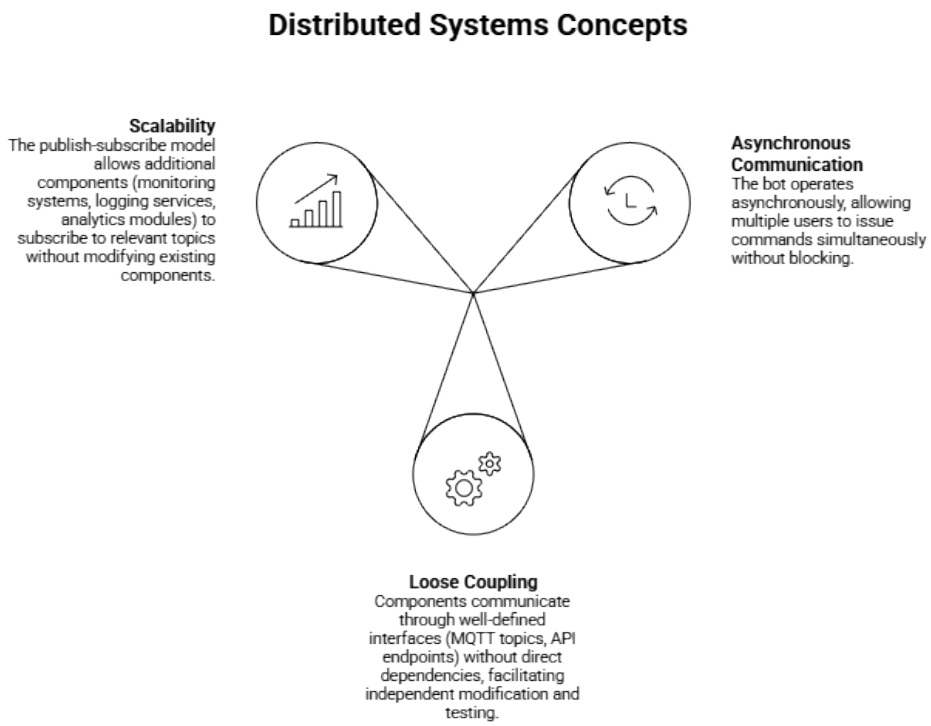

Fig. 7. Distributed systems concepts

ing a 60–70% i improvement compared to typical cloud delays of 500–800 milliseconds.

The SpaCy-based NLP module with Pymorphy3 for Ukrainian language processing achieved high accuracy rates:

- Single-device commands: 94% i accuracy;
- Multi-device commands: 91% i accuracy;
- Complex conditional commands: 87% i accuracy.

The system effectively handled synonyms and grammatical variations through lemmatization and tokenization. Challenges emerged with ambiguous commands and unclear device references.

Local network architecture provided measurable advantages. The system maintained full functionality during internet outages, confirming independence from external services. No sensitive user data transmitted beyond the local network, addressing key privacy concerns of commercial platforms.

Several limitations were identified:

- Proprietary devices required custom integration scripts;
- Limited context awareness for sequential commands;
- Performance degradation with networks exceeding 50 devices;
- Text-only input requiring additional speech recognition for voice control.

Compared to commercial solutions, the system offers superior privacy and customization while commercial platforms maintain advantages in ecosystem breadth and setup simplicity.

From an educational perspective, the developed system demonstrated high potential as a laboratory and project-based learning environment. Students can explore practical aspects of IoT device integration, MQTT-based communication, and natural language command processing. The modular architecture allows learners to modify individual components, experiment with NLP models, and analyze system performance, making the platform suitable for use in courses related to artificial intelligence, Internet of Things, and intelligent information systems.

Conclusions. The system addresses growing concerns about data privacy and service dependency, functioning reliably regardless of internet connectivity.

The proposed architecture can also be applied in the educational process as a training and experimental platform for teaching modern information technologies. Its use in laboratory classes and student research projects contributes to the formation of practical skills in AI, NLP, and IoT system design.

Future enhancements should focus on context-aware dialogue capabilities to enable natural sequential commands without repeating device names. Extending language support beyond Ukrainian and incorporating direct speech-to-text capabilities using systems like Mozilla DeepSpeech would broaden accessibility while preserving local processing advantages. Developing behavior-learning models with federated learning could enable predictive automation based on user patterns. Improving interoperability through standardized protocols and automated device discovery would reduce setup complexity. Finally, investigating distributed processing for scalability and implementing enhanced security measures including end-to-end encryption and anomaly detection would strengthen the system for large-scale deployments.

The convergence of AI, IoT, and privacy-conscious design demonstrated in this work establishes a foundation for next-generation smart home technologies prioritizing user autonomy and accessibility.

References

1. Gopinath, V., Srija, A., Dr. S. Krishna, Rao, & Madhuri, A. (2018). Smart homes: Steps, Components, Utilities, and Challenges. *International Journal of Engineering & Technology*, 7, 2.7, 436–441. DOI: https://doi.org/10.14419/ijet.v7i2.7.10858.
2. Google Inc. Google Home documentation. 2025. Retrieved from https://support.google.com/googlenest/.
3. Apple Inc. Apple HomeKit documentation. 2025. Retrieved from https://developer.apple.com/design/human-interface-guidelines/homekit.
4. Amazon.com, Inc. Alexa documentation. Retrieved from https://developer.amazon.com/en-US/docs/alexa/documentation-home.html.
5. Sheehan, B. Top trends in smart home technology. *Claritas*. Retrieved from https://claritas.com/trends-in-smart-home-tech/.
6. Promwad. The future of smart homes: Key trends shaping 2025 and beyond. Retrieved from https://promwad.com/news/smart-home-trends-2025.
7. CNET. I thought I'd hate AI in home security. I couldn't have been more wrong. 2024. Retrieved from https://www.cnet.com/home/security/features/i-thought-id-hate-ai-in-home-security-i-couldnt-have-been-more-wrong/.
8. Praos Solutions. The future of home security: Integrating AI and smart technology in alarm systems. Retrieved from https://www.praos-solutions.com/blog/the-future-of-home-security-integrating-ai-and-smart-technology-in-alarm-systems/.
9. Crank Software. AI in IoT projects. *Crank Software Blog*. Retrieved from https://blog.cranksoftware.com/ai-in-iot-projects.
10. WebbyLab. Natural language processing (NLP): Can AI understand human language? Retrieved from https://webbylab.com/blog/nlp-how-ai-understands-human-language/.
11. Meritech Solutions. Understanding smart homes: IoT, AI/ML & security considerations. Retrieved from https://meritechsolutions.com/blog-post/understanding-smart-homes-iot-ai-ml-security-considerations/.
12. XenonStack. AI and IoT in smart homes. Retrieved from https://www.xenonstack.com/blog/ai-iot-in-smart-homes.
13. Intuz. AI smart home: Transforming smart home automation. Retrieved from https://www.intuz.com/blog/smart-homes-with-ai.
14. Needhi, J. (2025). Smart homes and AI-IoT integration. *Medium*. Retrieved from https://medium.com/@jeyadev_needhi/smart-homes-and-ai-iot-integration-3bc7365b4d20.
15. Mishra, P. (2021). *Explainability for NLP.* Practical Explainable AI Using Python: Artificial Intelligence Model Explanations Using Python-based Libraries, Extensions, and Frameworks. (pp. 193–227). New York : Apress. DOI: https://doi.org/10.1007/978-1-4842-7158-2_7.

Список використаних джерел

1. Smart homes: Steps, components, utilities, and challenges / Gopinath V. et al. *International Journal of Engineering & Technology*. 2018. Vol. 7. № 2.7. Pp. 436–441. DOI: https://doi.org/10.14419/ijet.v7i2.7.10858.
2. Google Inc. Google Home documentation. 2025. URL: https://support.google.com/googlenest/ (дата звернення: 19.12.2024).
3. Apple Inc. Apple HomeKit documentation. 2025. URL: https://developer.apple.com/design/human-interface-guidelines/homekit (дата звернення: 19.12.2025).
4. Amazon.com, Inc. Alexa documentation. URL: https://developer.amazon.com/en-US/docs/alexa/documentation-home.html (дата звернення: 19.12.2025).
5. Sheehan B. Top trends in smart home technology. *Claritas*. URL: https://claritas.com/trends-in-smart-home-tech/ (дата звернення: 19.12.2025).
6. Promwad. The future of smart homes: Key trends shaping 2025 and beyond. URL: https://promwad.

com/news/smart-home-trends-2025 (дата звернення: 19.12.2025).
7. CNET. I thought I'd hate AI in home security. I couldn't have been more wrong. 2024. URL: https://www.cnet.com/home/security/features/i-thought-id-hate-ai-in-home-security-i-couldnt-have-been-more-wrong/ (дата звернення: 19.12.2025).
8. Praos Solutions. The future of home security: Integrating AI and smart technology in alarm systems. URL: https://www.praossolutions.com/blog/the-future-of-home-security-integrating-ai-and-smart-technology-in-alarm-systems/ (дата звернення: 19.12.2025).
9. Crank Software. AI in IoT projects. *Crank Software Blog*. URL: https://blog.cranksoftware.com/ai-in-iot-projects (дата звернення: 19.12.2025).
10. WebbyLab. Natural language processing (NLP): Can AI understand human language? URL: https://webbylab.com/blog/nlp-how-ai-understands-human-language/ (дата звернення: 19.12.2025).
11. Meritech Solutions. Understanding smart homes: IoT, AI/ML & security considerations. URL: https://meritechsolutions.com/blog-post/understanding-smart-homes-iot-ai-ml-security-considerations/ (дата звернення: 19.12.2025).
12. XenonStack. AI and IoT in smart homes. URL: https://www.xenonstack.com/blog/ai-iot-in-smart-homes (дата звернення: 19.12.2025).
13. Intuz. AI smart home: Transforming smart home automation. URL: https://www.intuz.com/blog/smart-homes-with-ai (дата звернення: 19.12.2025).
14. Needhi J. Smart homes and AI-IoT integration. *Medium*. 2025. URL: https://medium.com/@jeyadev_needhi/smart-homes-and-ai-iot-integration-3bc7365b4d20 (дата звернення: 19.12.2025).
15. Mishra P. Explainability for NLP. *Practical Explainable AI Using Python: Artificial Intelligence Model Explanations Using Python-based Libraries, Extensions, and Frameworks.* New York : Apress, 2021. P. 193–227. DOI: https://doi.org/10.1007/978-1-4842-7158-2_7.

С. В. Суліма,
Л. С. Глоба,
М. С. Ткаченко,
Я. В. Савченко

СИСТЕМИ РОЗУМНОГО БУДИНКУ НА БАЗІ ШТУЧНОГО ІНТЕЛЕКТУ ЯК ОСВІТНЯ ПЛАТФОРМА НА ОСНОВІ ТЕХНОЛОГІЙ NLP ТА IOT

***Анотація**. У цій статті представлено підхід до проєктування локально розгорнутої системи управління розумним будинком як освітньої платформи на основі інтеграції технологій штучного інтелекту (ШІ), обробки природної мови (NLP) та Інтернету речей (IoT). Актуальність дослідження зумовлена широким застосуванням сучасних рішень для розумного будинку на хмарних платформах, що призводить до збільшення затримки відгуку, обмеженої офлайн-функціональності, підвищених ризиків безпеки та недостатньої підтримки менш поширених природних мов, зокрема української, а також зростаючої потреби в практичних освітніх платформах, які дають змогу студентам отримати практичний досвід роботи з системами автоматизації на основі ШІ та інтелектуальними інтерфейсами. Основна мета полягає у підвищенні надійності, швидкості реагування та конфіденційності систем розумного будинку, а також у розробці універсального навчального інструменту для викладання сучасних технологій у галузі проєктування інтелектуальних систем та обробки природної мови. Запропонована система побудована на платформі Home Assistant з відкритим кодом та протоколі обміну повідомленнями MQTT, що зменшує навантаження на мережу, забезпечує доставку повідомлень з мінімальною затримкою й стабільну роботу в середовищах з обмеженим доступом до інтернету. Особлива увага приділяється безпеці системи внаслідок конфігурації брокера MQTT із механізмами автентифікації та шифрованими обліковими даними. Взаємодія з користувачем покращена завдяки модулю NLP на базі штучного інтелекту, реалізованому в Python із використанням бібліотеки spaCy та морфологічного аналізатора Pymorphy3. Модуль виконує нормалізацію тексту, токенізацію, лематизацію та морфологічний аналіз команд українською мовою, що дає можливість точно виявляти наміри користувача та цільові пристрої. Бот Telegram надає міжплатформний користувацький інтерфейс без додаткового програмного забезпечення на стороні клієнта. За допомогою модульної архітектури студенти мають змогу модифікувати окремі компоненти, експериментувати з моделями NLP та розробляти власні розширення, що робить платформу придатною для лабораторних робіт і дослідницької діяльності. Експериментальна валідація демонструє зменшення затримки виконання команд на 150–200 мілісекунд порівняно з хмарними рішеннями (500–800 мілісекунд). Модуль NLP досяг точності 94% для команд одного пристрою, 91% і для команд декількох пристроїв і 87% і для складних умовних команд. Результати показують, що система є як життєздатною альтернативою пропрієтар-*

ним хмарним платформам, так і ефективною освітньою платформою для розвитку професійних компетенцій у сфері технологій автоматизації на основі штучного інтелекту.

***Ключові слова**: освітні технології, освітня платформа, розумний будинок, штучний інтелект, IoT.*

INFORMATION ABOUT THE AUTHORS

Sulima S. V. — PhD in Engineering, Associate Professor, Associate Professor of the Department of Information Technologies in Telecommunications, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine, itssulima@gmail.com; ORCID ID: https://orcid.org/0000-0002-6333-7693

Globa L. S. — D. Sc. in Engineering, Professor, Professor of the Department of Information Technologies in Telecommunications, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine, lgloba@its.kpi.ua; ORCID ID: https://orcid.org/0000-0003-3231-3012

Tkachenko M. S. — Bachelor of Science, Department of Information Technologies in Telecommunications, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine, its30316@gmail.com; ORCID ID: https://orcid.org/0009-0001-8740-6382

Savchenko Ya. V. — D. of Philosophy (PhD), Senior Researcher of the Department of Information and Didactic Modeling, NC "Junior Academy of Sciences of Ukraine", Kyiv, Ukraine, savchyarik@gmail.com; ORCID ID: https://orcid.org/0000-0001-5790-6629

ІНФОРМАЦІЯ ПРО АВТОРІВ

Суліма Світлана Валеріївна — канд. техн. наук, доцент, доцент кафедри інформаційних технологій у телекомунікаціях, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», м. Київ, Україна, itssulima@gmail.com; ORCID ID: https://orcid.org/0000-0002-6333-7693

Глоба Лариса Сергіївна — д. техн. наук, професор, професор кафедри інформаційних технологій у телекомунікаціях, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», м. Київ, Україна, lgloba@its.kpi.ua; ORCID ID: https://orcid.org/0000-0003-3231-3012

Ткаченко Микита Сергійович — бакалавр, кафедра інформаційних технологій у телекомунікаціях, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», м. Київ, Україна, its30316@gmail.com; ORCID ID: https://orcid.org/0009-0001-8740-6382

Савченко Ярослав Володимирович — д. філос., старший науковий співробітник відділу інформаційно-дидактичного моделювання, НЦ «Мала академія наук України», м. Київ, Україна, savchyarik@gmail.com; ORCID ID: https://orcid.org/0000-0001-5790-6629

Рукопис надійшов до редакції / Received 05.01.2026
Рукопис прийнято до друку / Accepted 03.03.2026
Опубліковано / Available online 30.04.2026