UDC 004.94

INTELLIGENT MULTILINGUAL ASSISTANT BOT FOR GLOBAL USER SUPPORT SYSTEMS

Denys Mitin[1], *Soloschuk Mykhailo*[2]

[1] *Master's student of the Department of Systems Analysis and Information-Analytical Technologies, NTU "KhPI", Kharkiv, Ukraine*

[2] *Associate Professor of Systems Analysis and Information and Analytical Technologies, Ph.D. tech. Sciences, NTU "KhPI", Kharkiv, Ukraine*

Denys.Mitin@cs.khpi.edu.ua

The modern world is characterized by the rapid development of information technologies and globalization, which leads to an increasing number of users from different countries and cultures. Every day, people need to search for information, often using the Internet for this purpose. Students, professionals, and ordinary people interested in new topics encounter certain difficulties.

When reading new material, it is often difficult to understand certain terms, and finding their definitions using a standard search can be even more challenging. This problem can be addressed by artificial intelligence capable of recognizing the meaning of terms, consolidating information from multiple sources, and explaining complex text in a simplified manner. A browser extension-assistant can greatly simplify the process of understanding complex or new terms for users.

Globalization and language diversity make a multilingual assistant-bot an important tool that can eliminate language barriers and provide access to information for users from various countries. Today, users expect instant answers to their questions, and an intelligent assistant-bot can provide 24/7 support, increasing customer satisfaction and reducing the workload on support teams.

An assistant-bot can be useful in many situations, including:

– Education: Students can use the assistant to understand complex terms while studying new subjects, providing simple definitions and usage examples.

– Professional activities: Professionals can refer to the assistant for quick explanations of specialized terms.

– Research: Researchers can use the assistant to get synthesized answers to their queries, saving time.

– Language learning: Users can use the assistant for translating terms and phrases.

Thus, the development of an “Intelligent Multilingual Assistant-Bot for Global User Support Systems” is highly relevant in the modern business environment, where it is crucial to provide efficient, fast, and high-quality support tailored to individual user needs. Technological advances in artificial intelligence, natural language processing, and machine learning open new opportunities for creating global intelligent support systems that can learn from the history of interactions with users.

The aim of this project is to create an assistant bot designed to provide users with clear explanations of highlighted text on web pages, answer questions through a simple chat interface, and support multilingual queries. The bot will be capable of addressing user questions, helping clarify complex terms, and finding relevant information efficiently. It will also interpret selected text in the user's language, offering explanations that are easy to understand.

One of the assistant bot's strengths is its ability to engage in active dialogue, keeping track of previous interactions to provide contextually relevant responses. This makes communication smoother and more intuitive. Additionally, the bot will be equipped with multilingual processing, allowing it to interact seamlessly with users in various languages by automatically detecting and responding in the appropriate one.

The development of the assistant bot involves several advanced technologies. JavaScript [1] is used to build the user interface and integrate the bot into web browsers. Python [2] powers the backend, managing request handling, user interaction logic, and connections with external services. For natural language processing, the OpenAI API [3] is employed to generate accurate and meaningful responses, as well as to support multiple languages. MongoDB [4] serves as the database for user data, query histories, and settings, offering flexibility and scalability. NGINX [5] web server was used as a powerful gateway solution for request routing. Lastly, a Google Chrome extension [6] is developed,

enabling users to easily access the assistant from any webpage and interact with selected text using browser APIs.

The UML sequence diagram in Figure 1 illustrates how the Intelligent Assistant Bot processes different types of user requests, specifically “Explain” requests and “Chat message” requests.

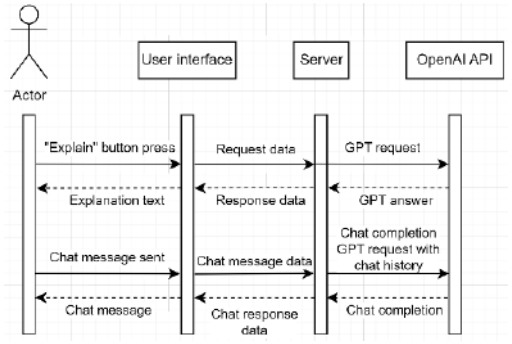

Figure 1 – assistant bot sequence diagram

These requests trigger distinct interactions between the user interface, backend server and OpenAI API. When a user initiates an “Explain” request, the process begins with the user selecting a piece of text on a web page and asking the bot to explain it. On the other hand, a “Chat message” request occurs when the user engages in an active conversation with the bot by asking a question or making a statement in the chat interface.

In both scenarios, the Intelligent Assistant Bot relies on advanced language processing and contextual understanding to deliver accurate and relevant responses, creating an efficient and user-friendly experience. These interactions highlight the bot's ability to simplify complex information and maintain meaningful dialogue, enhancing its role as a valuable tool for users navigating diverse content on the web.

As a result of the completed work, an application was developed to assist in searching and interpreting information, and the following tasks were accomplished:

Existing analogs were compared, and a list of functional requirements for the project was created.

1. The client-side of the application was developed using JavaScript and HTML.

2. The server-side of the application was implemented using Python, with Gunicorn as the HTTP server, OpenAI-Python as the NLP request processor, and MongoDB as the database.

3. The server was deployed using NGINX as a gateway for aggregating HTTP and HTTPS traffic, along with the Python server component.

4. Functional testing of the application was conducted.

List of references:

1. JavaScript | MDN [Electronic resource] // developer.mozilla.org. – 2024. – Resource access mode: https://developer.mozilla.org/en-US/docs/Web/JavaScript.

2. Python for Beginners | Python.org [Electronic resource] // python.org. – 2024. – Resource access mode: https://www.python.org/about/gettingstarted.

3. Developer Quickstart – OpenAI API // platform.openai.com. – 2024. – Resource access mode: https://platform.openai.com/docs/quickstart.

4. Introduction to MongoDB – MongoDB Manual [Electronic resource] // mongodb.com. – 2024. – Resource access mode: https://www.mongodb.com/docs/manual/introduction.

5. Beginner’s Guide // nginx.org. – 2024. – Resource access mode: https://nginx.org/en/docs/beginners_guide.html.

6. Extensions / Get started | Chrome for Developers // developer.chrome.com. – 2024. – Resource access mode: https://developer.chrome.com/docs/extensions/get-started.