DESIGN AND EFFICIENCY ASSESSMENT OF THE INFORMATION SYSTEM "DSEA-TELEGRAM-INTERFACE"

Yevhen Pelikh

Department of Infocommunication Engineering V.V. Popovsky,
Kharkiv National University of Radio Electronics,
Ukraine

E-mail: yevhen.pelikh@nure.ua

Abstract

This paper addresses the development of an information system, a Telegram bot, designed to automate and simplify access to educational information at the Donbas State Engineering Academy (DSEA). The system's primary goal is to solve the problem of inefficient data retrieval caused by information being scattered across various parts of the university website. The implementation is based on Python, using the TeleBot library for the user interface and BeautifulSoup4 for automated web scraping. To ensure portability and stability in a cloud environment, the system utilizes a modular architecture containerized with Docker. A key feature of the system is its enhanced reliability, achieved through a fallback mechanism: a PostgreSQL database stores critical data (eg, bell schedules), ensuring continuous operation even if the university website is unavailable. A quantitative analysis of the bot's performance was conducted, demonstrating a significant reduction in information access time—by up to 84.6%—compared to manual website navigation. The developed bot provides a fast, reliable, and unified access point to academic information for students and staff.

In the modern conditions of digitalization of education and the rapid development of mobile technologies, the speed and convenience of access to educational information is of particular importance. Students and teachers of higher education institutions regularly interact with large volumes of data: schedules of classes and sessions, rating lists, curricula and orders. One of the promising areas is the development of interactive services, in particular Telegram bots, which are becoming a powerful tool for automating the educational process [1].

The problem that this approach solves is the inefficiency of manual search for information, which is often scattered across different sections of official HEI websites. This forces users to spend a significant amount of time monitoring updates. Thus, the task of automating data collection and centralizing it in a single, easily accessible interface is a relevant scientific and practical task.
The technological basis for data collection in such systems is web scraping (web parsing) - the process of automated extraction of information from HTML pages of websites. Using the Python programming language and specialized libraries, such as BeautifulSoup4 (for analyzing the DOM structure of pages) and Requests (for executing HTTP requests), modules are implemented that extract the necessary data [2].

To deliver content to the user and implement an interactive interface, application programming interfaces (APIs) of messengers are used. In particular, the TeleBot library (pyTelegramBotAPI) provides tools for interacting with the Telegram Bot API, processing commands, displaying menus, and sending text and multimedia messages. To ensure reliability and storage of data that rarely changes, it is advisable to use database management systems, for example, PostgreSQL[3-4].

This paper presents the development of the DSEA-Telegram-Interface information system, created for students and employees of the Donbas State Engineering Academy (DSEA). The system is built on a multi-module architecture that clearly separates the logic of interaction with the user (bot/main.py), parsing modules (parsing/main.py) and database interaction functions (bot/db.py) [7]. The key feature of the development is to increase the reliability and fault tolerance of access to information. For this, the system uses the PostgreSQL database not only for storage, but also as a fallback mechanism.

If parsing critical data (for example, call schedule) from the official website of the DSEA ends with an error (the site is temporarily unavailable), the bot automatically accesses the database and provides the user with the last saved copy, ensuring service continuity [5-6]. To demonstrate the developed solution, Fig. 1 shows examples of user interaction with the DSEA-Telegram-Interface.

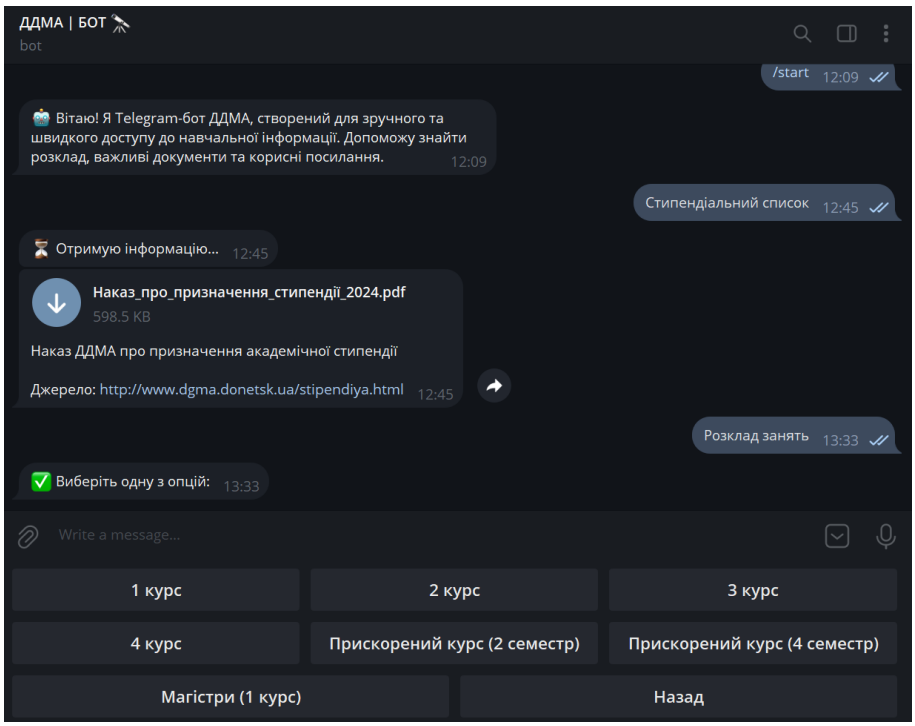

Fig. 1. Examples of interaction with the DSEA-Telegram-Interface information system (example of receiving PDF documents)

The software implementation of the system is carried out in compliance with the principles of modularity and separation of responsibilities (Separation of Concerns). The project structure clearly distinguishes the logic of interaction with the user and the logic of data acquisition. The bot directory contains modules responsible for processing Telegram API commands, working with menus, and managing dialog states. In contrast, the parsing module is isolated from the bot interface and contains only web page scraping algorithms. This approach allows you to easily scale the system by adding new data sources without the need to interfere with the main bot core.

To ensure portability, reliability, and rapid deployment of the system in a cloud environment, containerization technology was used. The presence of Dockerfile and docker-compose.yml configuration files in the project allows you to create an isolated execution environment that contains all the necessary dependencies (specified in requirements.txt), including the Python interpreter and libraries. This ensures that the software will run identically on both the developer's local machine and a remote server, eliminating version compatibility issues.

Special attention has been paid to security and configuration management. All sensitive data, such as the Telegram bot token and PostgreSQL database connection parameters, have been moved from the main code to the .env environment variable file. Using the python-decouple library allows these settings to be safely loaded when the container is launched, which meets modern standards for developing secure web applications and prevents the leakage of sensitive information into the version control system.

Also, an engineering feature of the project is the implementation of a keep-alive mechanism, which ensures stable operation of the bot on cloud PaaS platforms (for example, Render or Railway). Since such platforms often put applications into sleep mode when there is no activity, the system implements the keep_alive.py module. It launches a lightweight Flask-based web server in a separate thread (threading) that runs in parallel with the main bot.polling() cycle. This server processes periodic HTTP requests from external monitoring services, preventing the container from stopping and ensuring the bot is available to users around the clock.

As part of the development of the DSEA Telegram bot, a mathematical calculation of the effectiveness of its use for accessing educational information from the academy's website was also carried out. The main goal is to compare two alternative methods of obtaining data: traditional (via the academy's website) and automated-simplified (via the Telegram bot), with a focus on speed, number of actions, as well as the impact of unstable operation of the academy's website on the result.

The algorithm for calculating the effectiveness of the DSEA bot involves the sequential execution of a number of steps aimed at an objective assessment of the quality of its work. The analysis process takes into account key aspects of the functionality of the academy bot, stability, user friendliness and system speed. Each stage allows for a detailed analysis of individual characteristics of the DSEA bot, which, in turn, allows for detailed conclusions about the level of its effectiveness. The sequence and content of the evaluation steps will be presented below [8-9].

1. Input of variables. The following variables were input for the evaluation:

– d_s– number of actions when using the site;
– d_b– number of actions when using the bot;
– t_s– average time to obtain information through the site (under normal conditions);
– t_s^{avg}– average time to obtain information through the site (taking into account instability);
– t_b– average time to receive information via Telegram bot.

These variables are used in further formal calculations to determine the level of effectiveness of both approaches (Telegram bot and academy website).

2. Initial data: Two scenarios were modeled based on observations.

Scenario 1: Obtaining information through the academy website. Table 1 shows the approximate time required to perform each action when interacting with the DSEA website through a browser.

Table 1. Approximate time for performing actions when working with the DSEA website

Action	Description	Approximate time
1	Open browser	~10 seconds
2	Go to the academy website	~20 seconds
3	Find the section you need	~30 seconds
4	Go to section	~15 seconds
5	Download document	~10 seconds

Total: the user performs 5 consecutive actions, which in total take approximately 85 seconds. This is the total time required to obtain the necessary information through the academy website within the first scenario. That is: d_s= 5, t_s= 85.

Scenario 2: Receiving information via the DSEA Telegram bot. Table 2 shows the approximate time required to perform each action when receiving information via the Academy's Telegram bot.

Table 2. Approximate time for performing actions when working with the academy's Telegram bot

Action	Description	Approximate time
1	Open Telegram	~5 seconds
2	Open a chat with the bot	~5 seconds
3	Press the menu button	~5 seconds

Total: the user performs 3 actions, which in total take about 15 seconds. This is the total time required to obtain the same information via the Telegram bot, which is much faster compared to working via the DSEA website. That is: d_b= 3, t_b= 15.

3. Calculations without taking into account instability. At this stage, the winnings of the academy's Telegram bot are calculated in the absence of site failures:

– reducing the number of actions:

$$d = d_s - d_b = 5 - 3 = 2, \tag{1}$$

– reducing information access time:

$$t = t_s - t_b = 85 - 15 = 70, \tag{2}$$

– percentage reduction in the number of actions:

$$P_d = (2/5) * 100\% = 40\%, \tag{3}$$

– percentage reduction in access time:

$$Pt = \left(\frac{70}{85}\right) \cdot 100\% = 82.35\%, \tag{4}$$

4. Taking into account site instability. Since the official website of the academy periodically operates unstable, the probability of technical failures is taken into account p_s, which is approximately 1 time in 7 days or:

$$p_s = \frac{1}{7} \approx 0.143, \tag{5}$$

– in case of failure, the access time is doubled. The average time taking into account the probability is determined by the formula:

$$t_s^{avg} = (1 - p_s) * t_s + p_s * 2 * t_s, \tag{6}$$

– substituting the values:

$$t_s^{avg} = 0.857 * 85 + 0.143 * 170 = 72.845 + 24.31 = 97.155 = 97.2, \tag{7}$$

therefore, taking into account instability: $t_s^{avg} = 97.2$seconds.

– time saving taking into account instability:

$$t^{avg} = 97.2 - 15 = 82.2, \tag{8}$$

– percentage savings taking into account failures:

$$P_t^{avg} = (82.2/97.2) * 100\% = 84.6\%. \tag{9}$$

5. Summary. The conducted comparative evaluation of performance indicators confirms the significant advantage of the DSEA Telegram bot over traditional access to information through the academy's website:

– the number of actions is reduced by 40%;
– average access time without failures is reduced by 82.35%;
– average access time including failures – by 84.6%.

Table 3 shows comparative results of the effectiveness of access to information through the website and the Telegram bot.

Table 3. Comparison of the effectiveness of access to information via the website and Telegram bot

Access method	Number of actions	Time (without failures)	Time (with failures)
Through the website	5	85 seconds	97 seconds
Via Telegram bot	3	15 seconds	15 seconds

Thus, the DSEA Telegram-bot provides stable, reliable, fast and significantly more effective interaction with educational resources than using the academy's website. Its use allows not only to save time, but also to avoid technical difficulties associated with slow page loading, possible failures or structural complexity of site

navigation. This confirms the feasibility of implementing the DSEA Telegram-bot as a full-fledged information tool for students and teachers of the academy.

To confirm the comparison results, a software solution for calculating the bot's efficiency using Python was also implemented. All calculations were automated, and the results were displayed in the form of a convenient text report [8-9]. Fig. 2 shows an example of the bot's efficiency calculation results obtained during the execution of the corresponding Python code in the efficiency.py file.

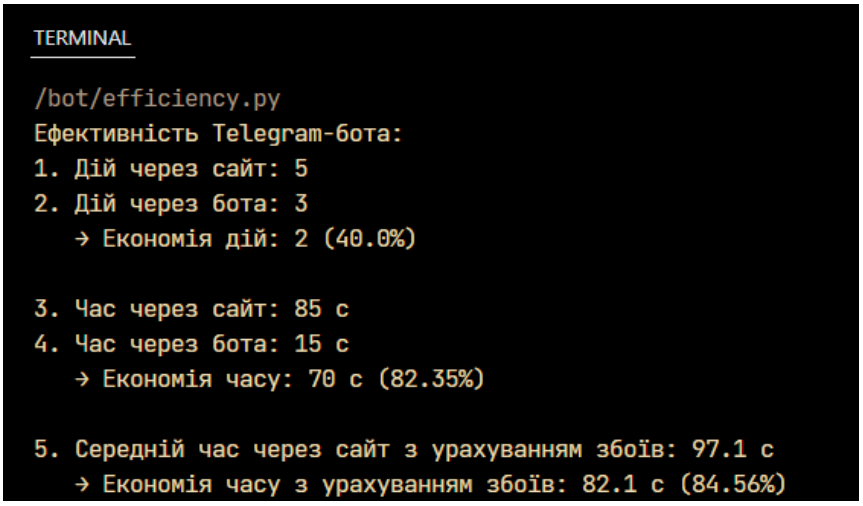

Fig. 2. Results of calculating the efficiency of the bot – efficiency.py

Literature

1. Освітній чат-бот @EducationUaBot [Електронний ресурс] // Міністерство освіти і науки України. Режим доступу: https://mon.gov.ua/osvita-2/tsifrova-transformatsiya-osviti-i-nauki/osvitniy-chat-bot (дата звернення: 14.03.2025).

2. Chat-боти в освіті: переваги, приклади, інтеграція [Електронний ресурс] // SendPulse. Режим доступу: https://sendpulse.ua/blog/chatbot-for-education-sphere (дата звернення: 15.03.2024).

3. Мельников О. Ю., Пеліх Є. П. Постановка задачі створення «Телеграм-бота» для підтримки навчального процесу у закладі вищої освіти // Автоматизація та комп'ютерно-інтегровані технології у виробництві та освіті: стан, досягнення, перспективи розвитку: матеріали Всеукраїнської науково-практичної Internet-конференції. – Черкаси, 2024. – С. 263–265.

4. Мельников О. Ю., Пеліх Є. П. Використання телеграм-ботів для спрощення доступу до інформації закладу вищої освіти // Сучасна освіта – доступність, якість, визнання: збірник наукових праць XVI Міжнародної науково-методичної конференції, 13–14 листопада 2024 року, мм. Краматорськ-Вінниця-Тернопіль / [за заг. ред. д-ра техн. наук., проф. С. В. Ковалевського і Hon.D.Sc., prof. Predrag Dasic]. – Краматорськ : ДДМА, 2024. – С. 207–210. – ISBN.

5. Пеліх Є. П., Мельников О. Ю. Задача створення «Телеграм-бота» для спрощення доступу до інформації закладу вищої освіти // Молодіжна наука: інновації та глобальні виклики: збірник тез за матеріалами Міжнародної науково-практичної конференції студентів, аспірантів та молодих вчених. – Полтава: НУПП імені Юрія Кондратюка, 2024. – С. 539–540.

6. Мельников О. Ю., Пеліх Є. П. Спрощення доступу до навчальної інформації за допомогою Telegram-бота // Автоматизація та комп'ютерно-інтегровані технології у виробництві та освіті: стан, досягнення, перспективи розвитку: матеріали Всеукраїнської науково-практичної Internet-конференції. – Черкаси, 2025. – С. 125–128.

7. Мельников О. Ю., Пеліх Є. П. Об'єктно-орієнтоване проєктування програмного забезпечення для спрощення доступу до інформації закладу вищої освіти за допомогою Телеграм-бота» // Сучасні інформаційні технології, засоби автоматизації та електропривод: матеріали IX Всеукраїнської науково-практичної конференції, 17–19 квітня 2025 р. / За заг. ред. О. Ф. Тарасова. – Краматорськ – Тернопіль: ДДМА, 2025. – С. 190–196. – ISBN 978-617-7893-00-3.

8. Пеліх Є. П., Мельников О. Ю. Аналіз ефективності використання Telegram-бота для спрощення доступу до інформації закладу освіти // Матеріали П'ятнадцятої Міжнародної наукової конференції студентів та молодих вчених «Сучасні інформаційні технології – 2025» «Modern

Information Technology – 2025» (15-16 травня 2025 р., м. Одеса) / МОН України; Національний університет «Одеська політехніка»; Ін-т комп'ют. систем. – Одеса : Наука і техніка, 2025. – С.80–82. – ISSN 2708-311X. – https://doi.org/10.5281/zenodo.15521809.

9. Пеліх Є. П. Математична модель для спрощення доступу до інформації закладу вищої освіти за допомогою "Телеграм-бота" // Сучасні інформаційні технології та системи в управлінні [Електронний ресурс] : Зб. матеріалів VI Міжнар. наук.-практ. конф. молодих вчених, аспірантів і студентів ; 10–11 квітня 2025 р. – Київ: КНЕУ, 2025. – С. 148–150. – ISBN 978-966-926-542-5.