Автор:
Шиловець Віталій Васильович,
студент 41 ІПЗ групи
Науковий керівник:
Майданюк Іван Вікторович
кандидат технічних наук, доцент

РОЗРОБЛЕННЯ ІНТЕЛЕКТУАЛЬНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ НА ОСНОВІ МЕТОДІВ МАШИННОГО НАВЧАННЯ

Анотація. У даному дослідженні розроблено текстовий бот, інтелектуальної інформаційної системи підтримки прийняття рішень у вигляді Telegram-бота, інтегрованого з API Google Classroom. Система реалізована з використанням Python, бібліотеки aiogram 3.x, протоколу OAuth 2.0 та Gemini API. Основну увагу приділено архітектурі системи, методам машинного навчання для освітньої аналітики, а також практичній реалізації рекомендацій для студентів на базі штучного інтелекту. Розглянуто особливості контролю якості програмного забезпечення шляхом тестування, а також обмеження прототипу та напрями його подальшого масштабування.

Ключові слова: Telegram-бот, машинне навчання, Google Classroom API, штучний інтелект, Gemini API, Python, aiogram, OAuth 2.0, освітня аналітика.

Вступ. З кожним роком цифровізація освіти набуває дедалі більшого масштабу. Хмарні LMS-платформи, зокрема Google Classroom, стали основою дистанційного навчання в університетах України та світу – її аудиторія перевищує 170 мільйонів користувачів у понад 190 країнах. Водночас стандартний функціонал платформи не надає студентам зведеної аналітики їхнього навчального прогресу. Паралельно Telegram із понад 900 мільйонами користувачів є найзручнішим середовищем для молоді, а сучасні великі мовні моделі (LLM) [1] – такі як Gemini – відкривають якісно новий рівень персоналізованої підтримки. Поєднання цих трьох складових формує основу для системи підтримки прийняття рішень у навчанні.

Постановка задачі. У процесі розроблення системи необхідно врахувати вимоги до її функціональності [2]: реалізувати механізм автентифікації через OAuth 2.0 з автоматичним оновленням токенів; забезпечити збір та аналіз навчальних даних студента з Google Classroom API; реалізувати модулі відображення дедлайнів, успішності та курсів; розробити ШІ-асистента на базі Gemini API [3] у двох режимах – автоматичних порад та вільного діалогу. Крім того, важливими є: тестування системи (модульне й інтеграційне), визначення обмежень прототипу та формулювання рекомендацій щодо масштабування. Задачею дослідження є проектування та реалізація архітектури системи, що поєднує зручність месенджера з практичністю освітньої аналітики [5].

Мета роботи. Метою даного дослідження є створення програмного прототипу інтелектуальної інформаційної системи підтримки рішень у вигляді Telegram-бота, інтегрованого з API Google Classroom [4], що забезпечує студентів інструментами для індивідуального аналізу навчальної діяльності, моніторингу дедлайнів та отримання персоналізованих рекомендацій на основі штучного інтелекту.

Основна частина. Одним із ключових аспектів при розробці системи є вибір відповідної архітектури [5]. У дослідженні реалізовано модульну архітектуру, де кожен компонент виконує чітко визначену функцію: google_auth.py – авторизація та управління токенами OAuth 2.0; classroom_service.py – взаємодія з Google Classroom API; ai_service.py – генерація персоналізованих рекомендацій через Gemini API; database.py – локальне зберігання даних у SQLite. Основним технологічним стеком є Python 3.14 у поєднанні з aiogram 3.x [6], що забезпечує нативну асинхронність (asyncio), вбудовану підтримку кінцевих автоматів (FSM) для управління діалогом та безкоштовне використання.

Ключовою особливістю системи є гібридний підхід до аналітики: класичний статистичний аналіз для розрахунку освітніх метрик (кількість виконаних/невиконаних/прострочених завдань, середній бал, коефіцієнт виконання) та генеративний ШІ для формулювання персоналізованих рекомендацій. При натисканні кнопки «Порада ШІ» система збирає повний освітній контекст студента за допомогою функції build_student_context(), конвертує його у текстовий формат та передає до Gemini API. Відповідь формується українською мовою і прив'язана виключно до реальних даних конкретного студента. Реалізовано також режим вільного діалогу зі ШІ через FSM-стан AIChatState.waiting_for_question.

Тестування системи проводилось поетапно: ручні модульні тести для перевірки коректності OAuth-авторизації, формування навчального контексту та обробки помилок Gemini API; інтеграційне тестування всього ланцюжка дій від аутентифікації до отримання ШІ-відповіді. Всі дев'ять варіантів використання успішно протестовані. Фінальний етап – тестування на реальному акаунті з 8 активними курсами та понад 50 завданнями – підтвердив стабільну роботу системи та коректність відображення даних.

Висновки. Розроблений прототип Telegram-бота, підключеного до Google Classroom API, підтвердив свою практичну ефективність. Система надає студентам аналітику прогресу, нагадування про дедлайни та персоналізовані поради від ШІ, що прив'язані до реальних навчальних даних. Порівняльний аналіз шести існуючих аналогів підтвердив унікальність рішення: жодна з розглянутих систем не поєднує одночасно реальну інтеграцію з Google Classroom API, генерацію персоналізованих рекомендацій через LLM та зручний Telegram-інтерфейс. У майбутньому можливе розширення функціональності системи: впровадження Docker-контейнеризації, ML-моделей для прогнозування академічного ризику, інтеграції з Google Calendar та хмарного розгортання.

Список використаних джерел

1. Малежик М.П., Малежик П.М., Малежик П.М. Особливості використання великих мовних моделей в освітньому просторі. Розділ в колективній монографії Цифрова трансформація освіти: теоретико-методичні засади: монографія / за заг. ред. В. П. Сергієнка; за наук. ред. Н. П. Франчук – Київ :Вид-во УДУ імені Михайла Драгоманова, 2024. – 382 с.
2. Google Classroom. URL: https://classroom.google.com (дата звернення: 10.01.2026).
3. Google Gemini for Education. URL: https://gemini.google.com (дата звернення: 19.01.2026).
4. Myronova O. Використання сервісу Google Classroom у навчальному процесі. URL: https://kerivnyk.info/2020/05/myronova-myronov-google-classroom (дата звернення: 12.01.2026).
5. Архітектура програмної системи і вплив на масштабування та підтримуваність. URL: https://visuresolutions.com/uk/alm-guide/systems-achitecture/ (дата звернення: 28.03.2026).
6. aiogram 3.x documentation. URL: https://docs.aiogram.dev/en/v3.17.0/ (дата звернення: 20.02.2026).
7. Oleksiienko, I.V., Franchuk, V.M.: Web-oriented electronic schedule. CEUR Workshop Proceedings 2292, 128–131 (2018).