Список використаних джерел:

1. DMOJ: Modern Online Judge. URL: https://dmoj.ca/ (дата звернення: 20.03.2026).
2. DOMjudge. Programming Contest Jury System. URL: https://www.domjudge.org/ (дата звернення: 20.03.2026).
3. Judge0. Open-source online code execution system. URL: https://judge0.com/ (дата звернення: 20.03.2026).
4. LeetCode. The world’s leading online programming learning platform. URL: https://leetcode.com/ (дата звернення: 20.03.2026).
5. HackerRank. Online coding tests and technical interviews. URL: https://www.hackerrank.com/ (дата звернення: 20.03.2026).
6. EOlymp. Website dedicated to competitive programming, algorithms and problem solving. URL: https://eolymp.com/ (дата звернення: 20.03.2026).
7. PC² Contest Control System. URL: https://pc2ccs.github.io/ (дата звернення: 20.03.2026).

УКД 004.415.2

АРХІТЕКТУРНІ ПІДХОДИ ДО ПРОЄКТУВАННЯ ЧАТ-БОТІВ У СУЧАСНИХ ПРОГРАМНИХ СИСТЕМАХ

Антоненко А.Я
annantonenko4567@gmail.com
Черкаський державний фаховий бізнес-коледж
Немченко В.Ю.
м. Черкаси, Україна

Активний розвиток технологій штучного інтелекту, обробки природної мови та хмарних обчислень сприяв впровадженню чат-ботів у сучасні програмні системи. Сьогодні такі рішення використовуються для автоматизації взаємодії з користувачами, надання довідкової інформації, підтримки клієнтів та виконання типових операцій у веб-сервісах, мобільних застосунках і корпоративних платформах. У зв’язку з цим особливого значення набуває архітектурне проєктування чат-ботів, оскільки саме структура програмної системи визначає її

масштабованість, здатність до інтеграції з іншими сервісами та можливість подальшого розвитку функціоналу. Метою дослідження є аналіз сучасних архітектурних підходів до розробки чат-ботів і визначення ключових принципів побудови їх програмної структури [1].

Чат-боти можна розглядати як багаторівневі програмні системи, що забезпечують автоматизований діалог між користувачем і інформаційною платформою. У більшості сучасних рішень архітектура формується на основі модульного підходу, де кожен компонент виконує окрему функцію у процесі обробки запиту. Типова структура системи включає інтерфейс взаємодії з користувачем, модуль обробки природної мови, механізм керування діалогом, підсистему доступу до даних та модулі інтеграції із зовнішніми сервісами. Такий підхід дозволяє відокремити логіку обробки запитів від інтерфейсної частини та спрощує масштабування системи при зростанні навантаження [1; 2].

Інтерфейс взаємодії забезпечує приймання повідомлень користувача та передавання результату назад у вибраний канал комунікації. У сучасних чат-ботах цей рівень часто підтримує кілька платформ одночасно: веб-чат, мобільний застосунок або популярні месенджери. Архітектурно інтерфейс виконує роль адаптера, що перетворює повідомлення у стандартизований формат для подальшої обробки системою. Це дозволяє інтегрувати додаткові канали взаємодії, не змінюючи внутрішню архітектуру чат-бота.[2].

Важливою складовою архітектури є модуль обробки природної мови. Він аналізує текст повідомлення, визначає намір користувача та виділяє ключові елементи запиту. Отримані результати передаються до модуля керування діалогом, який визначає подальшу логіку взаємодії. У межах архітектури цей компонент зазвичай реалізується як окремий сервіс, що дозволяє змінювати або вдосконалювати мовні моделі без впливу на інші частини системи. Подібна ізоляція функцій є важливою умовою гнучкості програмної архітектури.

Керування діалогом забезпечує визначення сценарію відповіді та координацію взаємодії між різними модулями системи. Саме тут формується рішення про те, чи потрібно звернутися до бази знань, виконати запит до

зовнішнього сервісу або згенерувати відповідь на основі попередньо визначених шаблонів. Такий механізм дозволяє поєднувати різні джерела інформації та формувати релевантні відповіді відповідно до контексту запиту користувача.

Підсистема доступу до даних відповідає за зберігання та отримання інформації, необхідної для роботи чат-бота. Вона може включати базу знань, історію діалогів, профілі користувачів або інші інформаційні ресурси. Розділення логіки обробки запитів і рівня зберігання даних дозволяє підвищити продуктивність системи та забезпечити більш ефективне управління інформаційними ресурсами. Крім того, архітектура чат-бота часто передбачає інтеграцію з зовнішніми інформаційними системами через API, що дає змогу отримувати дані з корпоративних сервісів або інших програмних платформ [3].

У сучасних дослідженнях також розглядаються архітектурні моделі, у яких функції чат-бота розподіляються між кількома взаємодіючими агентами. У такій структурі окремі компоненти відповідають за аналіз запиту, пошук інформації, перевірку результатів або формування відповіді. Мультиагентний підхід підвищує гнучкість системи та дозволяє ефективніше організувати взаємодію між різними функціональними модулями.

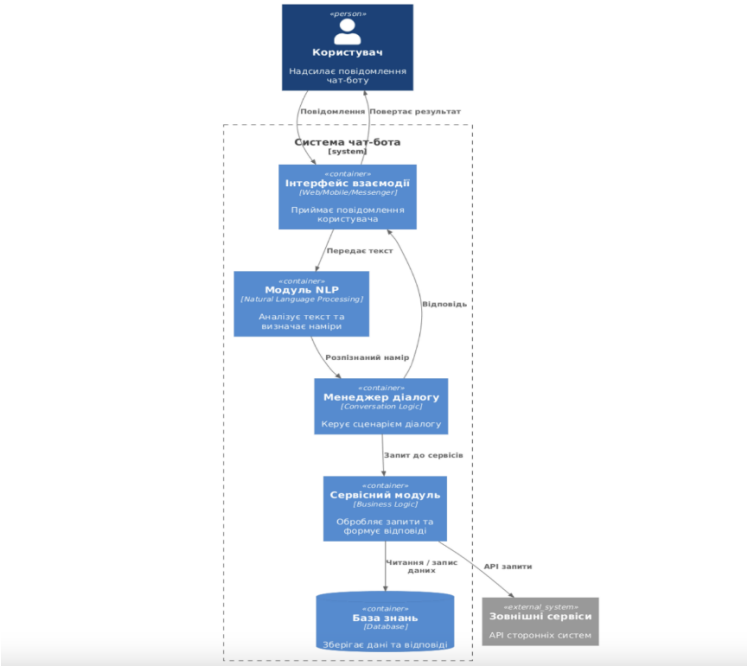

Рисунок 1 – Архітектура програмної системи чат-бота

Таким чином, архітектура сучасного чат-бота формується як модульна програмна структура, що поєднує інтерфейсні компоненти, механізми обробки природної мови, систему керування діалогом та підсистеми доступу до даних і зовнішніх сервісів. Використання таких архітектурних принципів забезпечує масштабованість, спрощує розвиток функціоналу та дозволяє створювати ефективні діалогові системи для веб-сервісів, мобільних застосунків і корпоративних інформаційних платформ [1,2].

Список використаних джерел

1. Ciesla R. The Book of Chatbots: From ELIZA to ChatGPT. Springer Nature, 2024. 159 c.
2. Skuridin A., Wynn M. Chatbot Design and Implementation: Towards an Operational Model for Chatbots. Mdpi. 17.04.2024. URL:https://www.mdpi.com/2078-2489/15/4/226 (дата звернення: 15.03.2026).
3. Aleedy M., Atwell E., Meshoul S. A Multi-Agent Chatbot Architecture for AI-Driven Language Learning. Mdpi. 01.10.2025. URL: https://www.mdpi.com/2076-3417/15/19/10634 (дата звернення: 15.03.2026).

УДК 004.738.5

РОЗРОБКА ФРОНТЕНДУ ЗА ДОПОМОГОЮ РІЗНИХ ФРЕЙМВОРКІВ: ПЕРЕВАГИ, НЕДОЛІКИ ТА ЗАВДАННЯ

Стеценко Я. І.
yanastetforcomp@gmail.com
Черкаський державний фаховий бізнес-коледж
Подорошко Д. І.
м. Черкаси, Україна

Сучасна розробка фронтенду є фундаментальною складовою створення прикладних рішень, що відповідають високим вимогам користувачів щодо швидкодії, масштабованості та безпеки. Еволюція вебтехнологій призвела до домінування односторінкових додатків (SPA), які забезпечують миттєву реакцію