ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

КВАЛІФІКАЦІЙНА РОБОТА

Тема: «Discord-бот «Princebot» з використанням розгалуженого алгоритму генерації»

Ступінь вищої освіти – бакалавр
Спеціальність – 122 «Комп’ютерні науки»
Освітня програма «Комп’ютерні науки»

ПОЯСНЮВАЛЬНА ЗАПИСКА

Виконав: здобувач 4 курсу
групи КН-21
Джума Кхаліфа

Керівник: викладач кафедр інформаційного менеджменту, математики та статистики
Олег МУШИНСЬКИЙ

Засвідчую, що кваліфікаційна робота оформлена відповідно до ДСТУ 3008:2015 та не містить запозичень з праць інших авторів без відповідних посилань.
Здобувач: _____
(підпис)

м. Київ – 2025 рік

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

«УНІВЕРСИТЕТ ЕКОНОМІКИ ТА ПРАВА «КРОК»

ЗАТВЕРДЖУЮ:
завідувач кафедри
комп'ютерних наук
_____Сергій МІЧКІВСЬКИЙ
«____»____20___р

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

Джума Кхаліфа Джума Білал Джума

Тема роботи	Discord-бот «PrinceBot» з використанням розгалуженого алгоритму генерації
Номер та дата наказу про затвердження теми	№121-7 від 24 грудня 2024 року
Коротка постановка завдання	Розробити функціонального Discord-бота з використанням розгалуженого алгоритму генерації для надання інтерактивних відповідей та автоматизації процесів у серверних каналах
Посилання на джерела інформації (не більше п'яти найменувань, які рекомендує науковий керівник)	1. David R. Morrison, Sheldon H. Jacobson, Jason J. Saupe, Edward C. Sewell. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning, Discrete Optimization, Volume 19. 2016. https://doi.org/10.1016/j.disopt.2016.01.005. 2. Ткаченко О., Шевченко А. Деякі аспекти розробки універсального серверного Discord-бота. *Цифрова платформа: інформаційні технології в соціокультурній сфері*, *4*(2), 173–186. 2021 https://doi.org/10.31866/2617-796X.4.2.2021.247476
Вимоги до кваліфікаційної роботи	Кваліфікаційна робота має передбачити теоретичне, системотехнічне або експериментальне дослідження складного спеціалізованого завдання або практичної проблеми в галузі комп'ютерних наук, яке характеризується комплексністю та невизначеністю умов і потребує застосування теорій і методів інформаційних технологій.

Дата видачі завдання 27 грудня 2024 р.

Керівник Олег МУШИНСЬКИЙ

Здобувач освітнього ступеня бакалавра Джума Кхаліфа

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання	Примітка
	Підготовчий етап		
1	Вибір напряму дослідження	02.12.2024 р.	*виконано*
2	Формування теми та призначення керівника	16.12.2024 р.	*виконано*
3	Затвердження теми кваліфікаційної роботи	23.12.2024 р.	*виконано*
4	Затвердження завдання на кваліфікаційну роботу	27.12.2024 р.	*виконано*
	Основний етап		
5	Розробка концепції кваліфікаційної роботи	13.01.2025 р.	*виконано*
6	Підбір та вивчення джерел інформації з напряму дослідження. Огляд існуючих аналогів	20.01.2025 р.	*виконано*
7	Затвердження розширеної постановки завдання. Підготовка та подання керівникові розділу 1 кваліфікаційної роботи	10.03.2025 р.	*виконано*
8	Проєктування. Підготовка та подання керівникові розділу 2 кваліфікаційної роботи	24.03.2025 р.	*виконано*
9	Підготовка доповіді для експертизи стану виконання кваліфікаційної роботи (проміжний контроль)	31.03-04.04.2025 р.	*виконано*
10	Реалізація. Підготовка та подання керівникові розділу 3 кваліфікаційної роботи	07.04.2025 р.	*виконано*
11	Підготовка та подання керівнику першого варіанту всієї кваліфікаційної роботи	14.04.2025 р.	*виконано*
12	Доопрацювання кваліфікаційної роботи з урахуванням зауважень керівника та представлення керівникові доопрацьованого варіанту кваліфікаційної роботи	21.04.2025 р.	*виконано*
	Завершальний етап		
13	Представлення рукопису для перевірки на плагіат	28.04-04.05.2025 р.	*виконано*
14	Підготовка презентації та доповіді на передзахист	05.05-11.05.2025 р.	*виконано*
15	Передзахист кваліфікаційної роботи	12.05-16.05.2025 р.	*виконано*
16	Доопрацювання роботи за результатами передзахисту	19.05-06.06.2025 р.	*виконано*
17	Експертиза роботи керівником та зовнішнім експертом	09.06-15.06.2025 р.	*виконано*
18	Доопрацювання доповіді та презентації для захисту	09.06-15.06.2025 р.	*виконано*
19	Захист кваліфікаційної роботи	16.06-22.06.2025 р.	*виконано*

Керівник Олег МУШИНСЬКИЙ

Здобувач освітнього ступеня бакалавра Джума Кхаліфа

Джума Кхаліфа Джума Білал Джума "Discord-бот «PrinceBot» з використанням розгалуженого алгоритму генерації "

Комп'ютерних наук Бакалавра. Спеціальність 122 Комп'ютерні науки «Університет економіки та права «КРОК» Київ 2025.

Цій роботі проводиться розробка Discord-Бота, пропонує широкий набір функцій для зручної взаємодії користувачів. Серед них: система рівнів та досвіду, відображення статистики, захист від спаму та розважальні команди. Бот розроблений з метою покращення комунікації між учасниками сервера, підвищення їхньої активності та надання корисного інструментарію як для адміністраторів, так і для звичайних користувачів.

Juma Khalifa Juma Bilal Juma "Discord bot 'PrinceBot' using a branched generation algorithm"

Bachelor's degree in Computer Science. Specialty 122 Computer Science

"University of Economics and Law "KROK" Kyiv 2025.

In this work, the development of a Discord Bot is carried out, which offers a wide range of features for convenient user interaction. Among them are: a system of levels and experience, statistics display, spam protection, and entertainment commands. The bot is designed to improve communication between server members, increase their activity, and provide useful tools for both administrators and ordinary users.

ЗМІСТ

ВСТУП

Актуальність теми. Автоматизація спілкування в соціальних мережах стає все більш важливою в сучасному цифровому світі. Discord є одним із найпопулярніших інструментів для спілкування, який можна використовувати для бізнесу, координації команд і розваг. Таким чином, розробка Discord-бота, який забезпечує корисні функції для управління сервером, підвищення активності користувачів і покращення досвіду взаємодії з користувачами, є важливою задачею.

Бот Discord дозволяє автоматизувати багато завдань, наприклад керування рівнями користувачів, відстеження активності, модерування контенту та створення інтерактивного спілкування. Це призводить до кращого адміністрування серверів, підвищення залученості учасників і кращого спілкування. Таким чином, створення бота Discord є актуальною темою, яка відповідає сучасним стандартам автоматизації цифрових платформ.

Мета дослідження. Мета проекту полягає в створенні функціонального та інтерактивного Discord-бота, який допоможе автоматизувати управління сервером, надавати користувачам корисну інформацію та надати додаткові можливості для взаємодії.

Завдання кваліфікаційної роботи містять:

- дослідження ринку discord-ботів;
- аналіз функціоналу кункурентів;
- вивчення потреб спільнот у автоматизації, модеруванні та розважальних функціях;
- планування та вибір технологій;
- визначення технічних вимог до бота;
- проєктування архітектури з розгалуженими алгоритмами команд і подій;
- розробка багатофункціонального бота;
- реалізація інтерактивних механіків;

- документування.

Об’єктом дослідження є автоматизація використання ботів Discord для керування спільнотами.

Предмет дослідження – технології створення Discord-бота для управління сервером та інтерактивної взаємодії.

Методи дослідження. Для виконання поставлених завдань використовуються такі методи дослідження: аналіз і порівняння існуючих варіантів розробки ботів; методи програмування для створення функціональних можливостей ботів; тестування та налагодження ботів для забезпечення стабільної роботи; і аналіз зворотного зв’язку від користувачів для вдосконалення функціоналу.

Структура та обсяг пояснювальної записки. Пояснювальна записка до проєкту складається зі вступу, трьох розділів, висновків, списку посилань (13 найменувань). Пояснювальна записка містить 23 рисунків, 2 таблиць. Загальний обсяг пояснювальної записки складає 45 сторінки.

РОЗДІЛ 1
ТЕОРЕТИЧНІ АСПЕКТИ РОЗРОБКИ DISCORD-БОТА ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Предметна область

Боти Discord - це автоматизовані програми, які працюють у межах серверів Discord і виконують різні функції, щоб допомогти користувачам спілкуватися краще та керувати спільнотами. Вони широко використовуються в багатьох областях, включаючи бізнес, освіту, розваги та підтримку клієнтів. Основною функцією Discord-бот є автоматизація управління сервером; це включає модерацію чату, розподіл ролей, привітання нових учасників, логування подій і нагадування про важливі події. Вони сприяють стабільності, регулюють вміст повідомлень, обмежують спам і накладають санкції на порушників.

Багато ботів використовують функції гейміфікації [1], такі як рівні, досягнення та нагороди, щоб зацікавити користувачів. Крім того, вони здатні виконувати розважальні завдання, такі як інтерактивні ігри, вікторини, створення мемів і відтворення музики в голосових каналах.

Користувачі можуть заробляти віртуальну валюту та витрачати її на внутрішньо серверні покупки завдяки економічним механікам, які пропонують деякі боти. Інформаційні боти надають аналітику, курси валют, погоду та новини.

Крім того, Discord-боти можуть інтегруватися з різними сервісами, такими як Twitch, YouTube, Google Calendar, Trello та GitHub, щоб спростити роботу та автоматизувати процеси. Вони є незамінним інструментом для адміністраторів серверів завдяки своїй функціональності та гнучкості. Вони також значно покращують користувацький досвід Discord.

Discord є однією з найпопулярніших платформ для спілкування в Інтернеті, яка об’єднує користувачів за інтересами. Він був спочатку розроблений для геймерів, але зараз використовується в багатьох сферах,

таких як освіта, бізнес, творчість і розважальні сервери. Discord-боти використовуються для покращення взаємодії користувачів і спрощення управління серверами. Вони допомагають модерувати, надавати інформацію, підтримувати інтерактивну взаємодію та створювати комфортний користувацький досвід [1].

Розробка бота для Discord охоплює кілька основних елементів: автоматизація рутинних завдань, таких як модерація, надання статистики; впровадження механізмів аналізу даних, які покращують взаємодію користувачів із ботами; забезпечення безпеки та стабільності роботи бота.

На сьогоднішній день існує велика кількість готових ботів, які можуть виконувати різні завдання, наприклад модерацію, передачу статистики та розважальні команди. Проте не всі з них пропонують повний набір необхідних функцій у межах одного продукту. Далі ми розглянемо головних конкурентів бота, який розробляється.

MEE6 – це один із найпопулярніших ботів для Discord, який допомагає автоматизувати управління сервером. Він пропонує такі функції, як налаштування привітань, система рівнів, авто-модерація та інтеграція з іншими платформами.

Основні функції:

- накопичення досвіду та систему рівнів;
- автоматичне вітання тих, хто приймає участь у сервера;
- модерація чату за допомогою автобану, муту та фільтру нецензурної лексики;
- використання YouTube, Twitch і Twitter для автоматичних сповіщень;
- створення власних команд (платний доступ).

Недоліки:

- багато функцій обмежені платною версією;
- обмежені можливості кастомізації команд без преміум-доступу;
- відсутність детальної аналітики активності користувачів [2];

Dyno – це універсальний бот, який дозволяє модерувати сервери, автоматизувати різні процеси та аналізувати активність користувачів.

Основні функції:

- автоматична модерація (автобан, автоматичні попередження);
- фільтр для боротьби зі спамом;
- автоматичне вітання нових учасників;
- налаштування власних команд та реакцій;
- система ведення логів подій на сервері.

Недоліки:

- відсутність системи рівнів та накопичення досвіду;
- безкоштовна версія має обмежені функції, а розширені можливості доступні лише за платною підпискою;
- обмежена інтеграція зі сторонніми сервісами [3].

Dank Memer – це розважальний Discord-бот, який фокусується на гумористичному контенті, мемах та інтерактивних іграх. Він популярний серед молодіжної аудиторії та має власну економічну систему, яка стимулює активність користувачів.

Основні функції:

- генерація мемів за допомогою команд;
- система внутрішньої валюти (монет), яку можна заробляти або втрачати;
- азартні міні-ігри (лотереї, ставки, пограбування);
- система інвентарю, банку та крадіжок;
- багата колекція розважальних команд.

Недоліки:

- відсутність системи рівнів та прокачування досвіду;
- не підтримує модераційний функціонал;
- високий ризик зловживань через ігрові механіки;
- частина контенту доступна лише з преміум-доступом [4].

Підсумкові результати порівняння представлені у таблиці 1.1, де відображено наявність ключових функцій у кожному з розглянутих ботів.

Таблиця 1.1 – Порівняльна конкурентов

Функція	MEE6	Dyno	Dank Memer	PrinceBot(Розроблюваний бот)
Автоматичне привітання	ТАК	ТАК	ТАК	ТАК
Система рівнів	ТАК	НІ	НІ	ТАК
Антиспам	НІ	ТАК	НІ	ТАК
Інтерактивні команди	НІ	НІ	ТАК	ТАК
Гнучкі налаштування	НІ	НІ	НІ	ТАК
Логування подій	НІ	ТАК	НІ	ТАК
Система економіки	ТАК	НІ	ТАК	ТАК
Міні-ігри	ТАК	НІ	ТАК	ТАК
Візуалізація рівнів	НІ	НІ	НІ	ТАК

На (*рис*. 1.1) видно привітання від бота MEE6 у вигляді стильної картки. Відображається аватарка, нікнейм нового учасника, його номер на сервері (наприклад, "Ти 1,234-й учасник!") та дата реєстрації в Discord. Додано кастомний текст вітання (типу "Ласкаво просимо! Заглянь у #правила") та інформацію про початковий рівень. Вся картка оформлена у кольорах сервера з емодзі та роздільниками.

Магазин працює на основі внутрішньої валюти, яку користувачі заробляють за свою активність. Асортимент включає різноманітні товари та послуги: від декоративних елементів (спеціальні ролі, кастомні емодзі, унікальні нікнейми) до функціональних покращень (тимчасові підвищення прав, додаткові можливості для спілкування). Наприклад, користувач може придбати тимчасову роль з особливим кольором або отримати право створювати власні голосування.

Технічна реалізація магазину передбачає інтерактивне меню з категоріями товарів, де кожен предмет має свою ціну та опис. При покупці бот

автоматично перевіряє наявність коштів у користувача, проводить транзакцію та негайно надає придбаний товар.

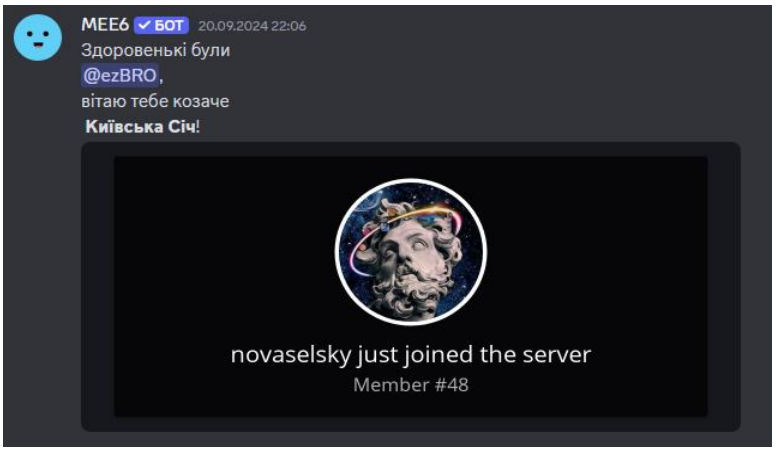

Рисунок 1.1 - Discord «MEE6»

Джерело: розроблено автором

1.2 Потенційні переваги програмного продукту у порівнянні з існуючими аналогами

Розроблений Discord-бот вирізняється серед існуючих аналогів завдяки поєднанню широкої функціональності, високого рівня кастомізації, відкритості у використанні. На відміну від переважної більшості сучасних рішень, які переважно зосереджені на виконанні однієї або кількох окремих задач (наприклад, виключно модерації або розваг), цей бот є багатофункціональним інструментом з інтеграцією кількох важливих компонентів – автоматизації управління сервером, інструментів модерації, аналітики активності та інтерактивних функцій для користувачів.

Обґрунтування очікуваних переваг:

– Комплексний підхід до реалізації функціоналу

–Бот поєднує у собі декілька ролей – від автоматичного модерування та фільтрації контенту до генерації звітів і взаємодії з користувачами через команди, реакції та інтеграції з іншими сервісами. Це зменшує необхідність встановлення кількох ботів з обмеженою функціональністю, що покращує продуктивність сервера і спрощує адміністрування.

– Гнучкість у налаштуванні

– Система налаштувань побудована таким чином, щоб користувач міг адаптувати бота без спеціальних технічних знань. Зручний інтерфейс та можливість конфігурації через прості команди або інтерактивні меню роблять бота доступним навіть для новачків.

– Відкритість функціоналу та відсутність прихованих обмежень

– На відміну від конкурентів, які часто блокують ключові функції за платною підпискою або потребують встановлення додаткових розширень, цей бот надає більшість основних можливостей у безкоштовному доступі. Такий підхід робить його доступним для широкого кола користувачів – зокрема, невеликих спільнот або навчальних груп.

– Оптимізована продуктивність і стабільність

– Архітектура бота побудована з урахуванням високих навантажень. Завдяки оптимізованим алгоритмам обробки подій та асинхронному виконанню команд, бот забезпечує стабільну роботу без затримок навіть на великих серверах із великою кількістю користувачів.

– Адаптивність до потреб спільноти

– Завдяки відкритій архітектурі та можливості додавання нових модулів, бот легко розширюється та адаптується до змінних вимог серверу. Це дозволяє розробляти й впроваджувати індивідуальні рішення під конкретні потреби проєкту.

Очікувані переваги програмного продукту полягають у зменшенні витрат на адміністрування серверу, підвищенні ефективності комунікації в межах спільноти, забезпеченні стабільної та безпечної взаємодії, а також в унікальній можливості використання сучасной програмного забезпечення без втрати якості або функціоналу. Це робить розроблений Discord-бот конкурентоспроможним, сучасним та корисним рішенням для широкого кола користувачів.

1.3 **Завдання на кваліфікаційну роботу**

У сучасних умовах Discord став однією з найпопулярніших платформ для комунікації. Його активно використовують не лише геймери, а й освітні установи, професійні спільноти, IT-компанії та інші організації, які цінують зручність, швидкість і можливості персоналізації. Завдяки відкритому API, платформа надає широкі можливості для розширення функціоналу за допомогою ботів, які здатні автоматизувати багато аспектів управління серверами та взаємодії з учасниками.

Ця робота присвячена розробці багатофункціонального Discord-бота, який поєднуватиме адміністративні інструменти, засоби автоматизації, інтерактивні функції та розважальний контент. Основна ідея полягає в створенні універсального інструменту, який може адаптуватися до потреб будь-якої спільноти: від навчального серверу до великої онлайн-групи за інтересами.

Мета та задачі роботи

Мета:

Розробити функціонального Discord-бота, здатного автоматизувати певні процеси на сервері, забезпечуючи інтерактивність, адміністративний контроль та додаткові корисні функції для покращення взаємодії учасників.

Основні задачі:

- для досягнення поставленої мети, необхідно вирішити такі ключові завдання;
- провести дослідження існуючих аналогів discord-ботів, проаналізувати їхні можливості та визначити найкращі практики реалізації;
- вивчити особливості discord api, обрати відповідні інструменти та бібліотеки, які забезпечують ефективну взаємодію з платформою;
- сформувати архітектуру майбутнього бота, розробити систему команд, передбачити обробку різних подій, реалізувати збереження даних у базі;

- реалізувати основний функціонал, який включає команди адміністрування, інтерактивні елементи, відповідь на події, а також механізми логування;
- забезпечити простоту використання, безпеку доступу до конфіденційної інформації;
- провести тестування системи в умовах реального середовища, зібрати відгуки та усунути можливі недоліки;
- підготувати повну документацію, яка включає інструкції з розгортання, опис функціоналу, а також технічну інформацію для розробників.

Технічні вимоги до розробки

Для реалізації проєкту обрано технології, які поєднують простоту, гнучкість та доступність. Це дозволяє забезпечити стабільну роботу бота навіть при обмежених ресурсах, а також створити рішення, яке легко адаптувати під нові вимоги.

Програмне забезпечення:

Операційна система: Windows

Мова програмування: Python (завдяки бібліотеці discord.py, яка добре задокументована та підтримує останні оновлення Discord API)

База даних: SQLite – проста, зручна та легка у використанні система для локального зберігання статистики та налаштувань.

Функціональні вимоги:

Розроблений бот повинен:

- обробляти текстові команди, що надходять від користувачів;
- реагувати на взаємодії (реакції, ембеди, повідомлення);
- мати адміністративні можливості: видалення повідомлень, видача ролей, бан/мут користувачів;
- зберігати статистику в базі даних: кількість повідомлень, активність користувачів тощо;
- логувати ключові дії: виконання команд, зміну налаштувань;

- пропонувати додаткові інтерактивні функції: міні-ігри, генерація повідомлень, інтеграції з арі зовнішніх сервісів.

Нефункціональні вимоги:

- забезпечення стабільної та безперебійної роботи;
- оптимізація швидкості обробки запитів;
- просте розгортання на сервері з можливістю швидкого оновлення без перезапуску всього сервісу.

Етапи виконання роботи

Проєкт реалізовуватиметься поетапно, з поступовим нарощуванням функціоналу:

Дослідження та аналіз. На першому етапі буде проаналізовано структуру Discord API, бібліотеки (discord.py, random, SQLite) та можливості інтеграції з іншими сервісами. Також буде проведено порівняльний аналіз існуючих популярних ботів.

Проєктування архітектури. Обрані технології буде адаптовано під специфіку завдань. Буде розроблено модульну архітектуру, що дозволить масштабувати функціонал у майбутньому.

Реалізація основного функціоналу. Включає підключення до Discord, створення базових команд, налаштування обробки подій (приєднання нового учасника) та збереження даних у базі.

Розробка додаткових функцій. Реалізація інтерактивного контенту, міні-ігор (слоти,вгадати число).

Тестування та оптимізація. Функціонал буде протестований у реальних умовах на тестовому сервері. Після тестування планується усунення багів, оптимізація швидкодії та перевірка стабільності роботи при високому навантаженні.

Підготовка документації. У завершальному етапі буде створено документацію, що охоплює інструкції з налаштування, опис команд, а також технічну частину для майбутніх розробників або адміністраторів.

Очікувані результати

Результатом роботи стане повнофункціональний Discord-бот, готовий до використання на будь-якому сервері. Він:

- забезпечить автоматизацію основних адміністративних процесів
- запропонує інтерактивні функції, які зроблять спілкування більш динамічним та цікавим
- працюватиме стабільно та швидко, з можливістю простого розгортання
- супроводжуватиметься документацією, яка спростить обслуговування, підтримку та розвиток бота в майбутньому

Таким чином, проєкт має на меті не лише створити корисний інструмент, а й закласти фундамент для подальшого розвитку у сфері розробки ботів, автоматизації процесів та розумного управління онлайн-спільнотами.

Висновок до розділу 1

У першому розділі було розглянули теоретичні основи створення Discord-ботів, визначив їхню роль у сучасному цифровому середовищі та можливості автоматизації серверних процесів. Провів порівняльний аналіз популярних конкурентів, таких як MEE6, Dyno та Dank Memer, що допомогло виявити їхні сильні та слабкі сторони. В результаті обґрунтував необхідність створення нового багатофункціонального бота з розширеним функціоналом, який буде орієнтований на інтеграцію адміністративних, аналітичних та розважальних можливостей в одному програмному продукті.

Також визначив предметну область, сформулював мету та завдання дослідження, а також описав технічні вимоги до реалізації програмного забезпечення. Окрему увагу приділили потенційним перевагам : гнучкість налаштувань, відсутність платних обмежень, оптимізована продуктивність і адаптивність до потреб різних спільнот.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА АРХІТЕКТУРА DISCORD-БОТА «PRINCEBOT»

2.1. Моделювання поведінки продукту

Проектування та архітектура Discord-бота охоплює всі аспекти створення ефективної, масштабованої та безпечної системи, що включає такі основні компоненти, як економіка, магазин, ігри, модерація, автоматизація та інші функції.

На (*рис 2.1*) представлено структурну діаграму Discord-бота, який складається з головного класу PrinceBot та кількох функціональних модулів. Така структура реалізує принцип модульності, що дозволяє легко додавати, змінювати або оновлювати функціонал без зміни всієї системи. Кожен модуль відповідає за певну частину функціональності бота, що покращує масштабованість, підтримку та розробку.

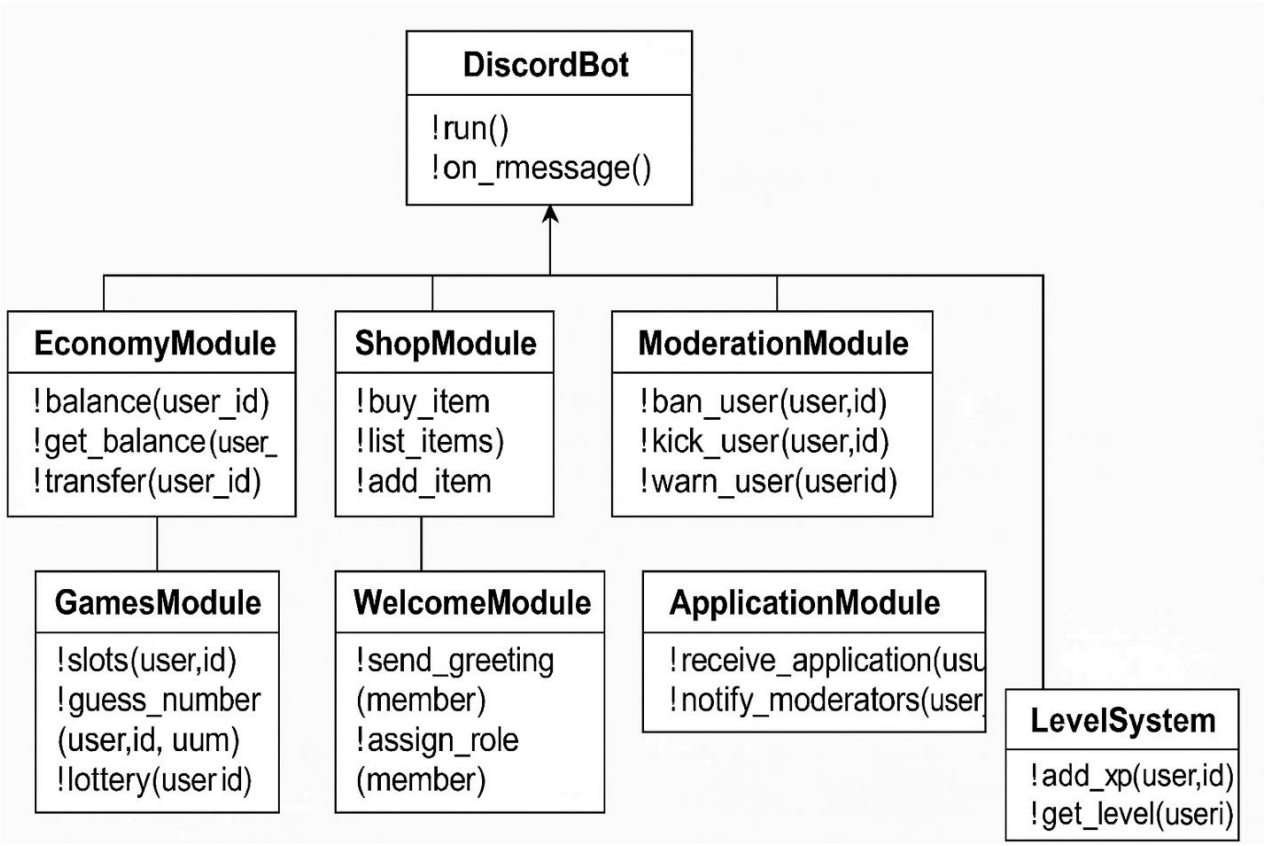

Рисунок 2.1 – Diagrama «Object»

Джерело: розроблено автором

1.**Основний клас** DiscordBot

Цей клас є ядром бота та містить основні методи для його роботи:

- run() – запускає бота, підключаючи його до Discord API.
- on_message() – обробляє всі повідомлення, надіслані на сервері, та визначає, які дії потрібно виконати у відповідь.

2. Модуль економіки (Economy Module**)**

Цей модуль відповідає за валютну систему бота:

- balance(user_id) – повертає поточний баланс користувача.
- transfer(user_id) – переказує валюту між користувачами.

Користувачі отримують віртуальну валюту за різні види активності: написання повідомлень, участь у серверних подіях або виконання спеціальних завдань. Наприклад, система може автоматично нараховувати невелику кількість валюту за кожне повідомлення або видавати більші винагороди за участь у конкурсах та вікторинах.

Накопичені кошти користувачі можуть використовувати для різних цілей. Вони можуть купувати унікальні ролі, спеціальні права або інші бонуси на сервері. Також передбачена можливість міжкористувацьких переказів, що створює додаткові можливості для соціальної взаємодії. Для любителів ризику є функції ставок та інших азартних механік.

Було визначено наступний принцип роботи системи балансу:

- Нарахування коштів
- Користувачі отримують віртуальну валюту за різні дії:
- Активність у чаті (наприклад, за кожне повідомлення).
- Участь у івентах (розіграші, вікторини, міні-ігри).
- Виконання завдань (щоденні нагороди, квести).
- Використання балансу
- Накопичені кошти можна витрачати на:
- Купівлю внутрішньо ігрових предметів (ролі, кастомні емодзі, унікальні статуси).
- Участь у азартних механіках (рулетка, ставки).

- Перекази іншим користувачам (соціальна взаємодія).

Баланс зберігається в базі даних SQLite, де кожному користувачу присвоюється унікальний ID та поточна кількість валюти (рис 2.2). Бот обробляє транзакції, перевіряє легітимність операцій і запобігає зловживанням (спам-повідомлення для фарму [5] валюти).

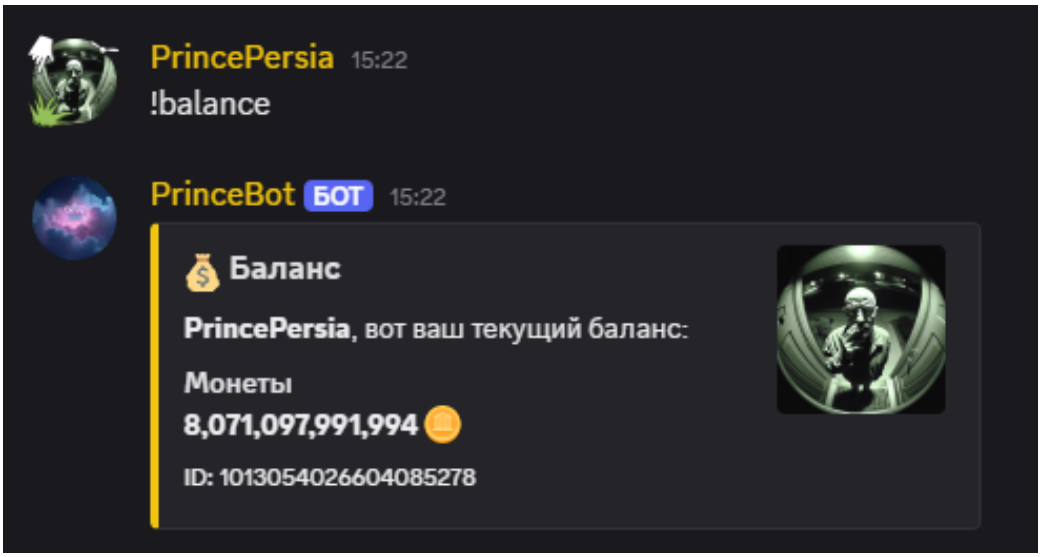

Рисунок 2.2 – Баланс у Discord «PrinceBot»

Джерело: розроблено автором

3. Модуль магазину (ShopModule**)**

Забезпечує функціонал для торгівлі та управління предметами:

- buy_item () – Цей процес відповідає за те, як користувач купує товар. Алгоритм перевіряє, чи є у користувача достатньо коштів, чи є товар в наявності, і якщо все гаразд – зменшує баланс та надає відповідну роль, емодзі або інший бонус. Якщо ж покупка не може бути здійснена, бот надсилає повідомлення з причинами відмови.

- list_items () – відображає список доступних предметів.

- add_item () – Ця функція доступна тільки адміністраторам і дозволяє їм динамічно додавати нові товари до магазину. Вона додає запис до бази даних і оновлює кеш.

Магазин працює на основі внутрішньої валюти, яку користувачі заробляють за свою активність. Асортимент включає різноманітні товари та послуги: від декоративних елементів (спеціальні ролі, кастомні емодзі, унікальні нікнейми) до функціональних покращень (тимчасові підвищення прав, додаткові можливості для спілкування). Наприклад, користувач може придбати тимчасову роль з особливим кольором або отримати право створювати власні голосування.

Технічна реалізація магазину передбачає інтерактивне меню з категоріями товарів, де кожен предмет має свою ціну та опис. При покупці бот автоматично перевіряє наявність коштів у користувача, проводить транзакцію та негайно надає придбаний товар.

В цьому модулі принцип роботи магазину наступний (рис 2.3):

1. Відкриття інтерфейсу магазину

- Користувач використовує команду (!shop) (рис 2.4).
- Бот відправляє інтерактивне меню з категоріями товарів (ролі, емодзі, послуги тощо).

2. Перегляд товарів

Кожен товар має:

- Назву наявність та опис
- Ціну у віртуальній валюті
- Обмеження.

3. Процес покупки

a. Користувач обирає товар (вводить команду).

b. Бот перевіряє:

Достатність коштів на балансі

Наявність обмежень (наприклад, чи має користувач потрібну роль)

c. Якщо умови виконані:

Валюта списується з балансу

Товар негайно надається (наприклад, роль або доступ)

4. Доставка товару

- Цифрові товари (ролі, права):
- Автоматична видача через Discord API

5. Обробка помилок

- Якщо коштів недостатньо повідомлення про відмову
- Якщо товар недоступний інформація про причину
- При технічних збоїв запис у лог та сповіщення адмінів

6. Оновлення асортименту

- Адміни можуть через окремі команди:
- Додавати нові товари
- Змінювати ціни
- Запускати акції (знижки, набори)

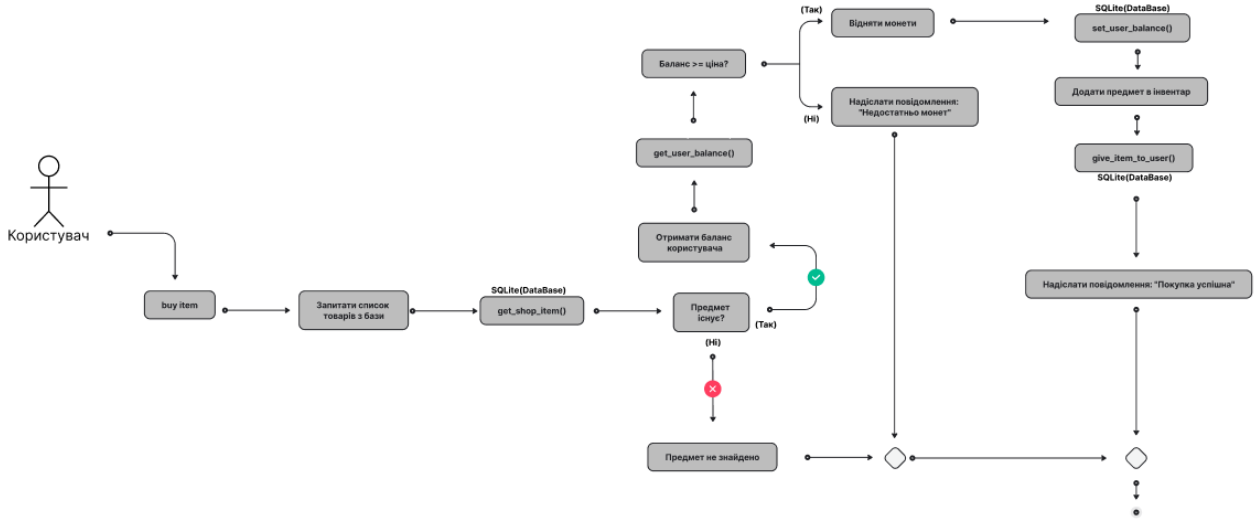

Рисунок 2.3 – Diagram Activity команди !shop«PrinceBot»

Джерело: розроблено автором

Технічні особливості модулю «магазин» наступні:

- база даних зберігає: історію покупок, залишки валюти, активні товари;
- для тимчасових товарів фонова перевірка термінів дії;
- захист від спаму (обмеження кількості покупок за час).

- *Рисунок 2.4 – Магазин товарів Discord «PrinceBot»*

Джерело: розроблено автором

4. **Модуль модерації (**ModerationModule**)**

Призначений для управління користувачами та підтримки порядку (рис 2.5):

- ban_user(user_id) – Функція відповідає за бан користувача на сервері. Перед тим, як видалити користувача, перевіряється, чи має бот необхідні права, а також чи не намагається він забанити адміністратора або самого себе. Усі випадки банів фіксуються в логах для подальшої звітност.

- kick_user(user_id) – Видаляє користувача з сервера без блокування. Зазвичай це менш суворий захід. Перед виконанням дій бот надсилає попередження, а модератор отримує підтвердження про виконання дії.

- warn_user(user_id) – Система записує порушення користувача в базу даних, вказуючи причину та дату. Коли кількість попереджень досягає певного рівня, вона може автоматично вжити заходів, таких як тимчасовий блокування або заборона писати в чаті.

Рисунок 2.5 – Diagram Use Case модерації«PrinceBot»

Джерело: розроблено автором

5. **Ігровий модуль (**GamesModule**)**

Містить розважальні функції для користувачів:

- slots(user_id) – Симулює гру в слоти: бот створює випадкові емодзі та визначає виграш на основі комбінацій. Якщо пощастить, то користувач отримає віртуальну валюту, яка зарахується на баланс. Є також внутрішній механізм контролю ймовірностей, щоб запобігти зловживанням.
- guess_number(user_id, num) – Користувачеві потрібно вгадати число, яке випадково обрав бот. За правильну відповідь нараховується бонус. Якщо не вдається вгадати, він пропонує спробувати ще раз.
- lottery(user_id) – Функція дозволяє купити лотерейний білет. Переможці обираються випадковим чином серед усіх учасників один раз на добу. Призовий фонд формується з комісії, що стягується з кожної покупки білета.

6. **Модуль привітання (**WelcomeModule**)**

Автоматизує вітання нових учасників за допомогою send_greeting(member), що надсилає привітальне повідомлення новому користувачу.

7. **Модуль заявок (**ApplicationModule**)**

Використовується для обробки заявок, на отримання ролі. Команда receive_application(user_id) обробляє запит на модераторську. Користувач вводить команду !apply, після чого бот надсилає відповідь у канал модераторів для розгляду.

8. **Система рівнів (**LevelSystem**)**

Відповідає за мотивацію користувачів через систему досвіду:

- add_xp(user_id) – Кожна дія, яку виконує користувач на сервері, може приносити певну кількість досвіду (XP). Алгоритм має вбудовані обмеження, щоб запобігти "фарму" – наприклад, XP нараховується лише раз на кілька хвилин. Коли користувач досягає нового рівня, бот автоматично надсилає вітання.

- get_level(user_id) – Ця команда витягує інформацію про поточний рівень, прогрес до наступного рівня та загальну кількість XP. Дані беруться з бази даних і можуть бути представлені у вигляді візуального рівня з прогрес-баром.

Discord-бот має модульну архітектуру, що робить його дуже зручним для розширення функцій. Кожен модуль виконує конкретні завдання, завдяки чому бот стає універсальним інструментом для управління сервером, розваг та взаємодії з користувачами. Така структура забезпечує гнучкість і зручність у використанні.

2.2Моделювання структури продукту

На (*рис* 2.6) представлено інтерфейс, пов’язаний з управлінням сервером у Discord платформі. Елементи інтерфейсу поділені на категорії та функціональні блоки, які можуть стосуватися налаштувань ботів, модерації чи економіки сервера.

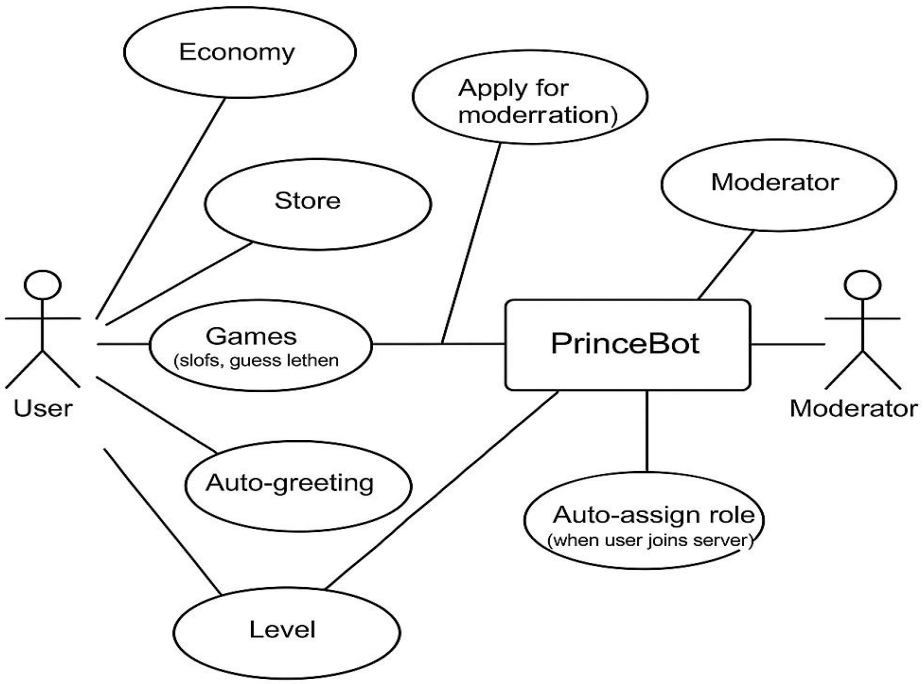

Рисунок 2.6 – Discord «PrinceBot»

Джерело: розроблено автором

Детально опишемо модулі даного інтерфейсу:

- "Economy". Розташований у верхній частині, що вказує на розділ, пов’язаний з економічними функціями сервера (наприклад, віртуальна валюта, магазин тощо).
- "Apply for". Містить підпункт "moderation". Ця функція подачі заявок на роль модератора.
- "Store". Відповідає за віртуальний магазин сервера, де користувачі можуть купувати товари чи послуги за внутрішню валюту.
- "Moderator". Містить інструменти для модераторів, такі як видалення повідомлень, бан користувачів тощо.
- "Games". Вказує на наявність ігрових функцій міні-ігор на сервері.
- "User". Це акаунт користувача з унікальним який і буде використовувати команди на сервери
- "Moderator" (повторно). Дублюється додаткові підпункти, такі як "Auto-greeting" (автоматичне вітання нових користувачів) та "Auto-assign role" (автоматичне призначення ролей при вході на сервер).
- "Level". Рівнів користувачів, де активність на сервері винагороджується підвищенням рівня. Останній елемент "PrinceBot"

2.3 Опис архітектури продукту

Програмний продукт побудований з урахуванням сучасних принципів розробки, таких як модульність, розділення відповідальностей (SoC) та легкість підтримки. Використання архітектурного підходу на основі подієво-орієнтованої моделі (за допомогою бібліотеки discord.py) дозволяє ефективно керувати взаємодією з Discord API, забезпечує гнучкість у розширенні функціоналу та спрощує процес тестування.

Основними компонентами системи є модульна структура (Cogs). Функціонал бота розділений на незалежні модулі (Cogs), кожен з яких відповідає за певну групу команд систем. Це дозволяє:

- Легко додавати нові функції без змін у головному коді.

- Виконувати оновлення окремих частин без впливу на інші.
- Уникати конфліктів коду за рахунок чіткої ізоляції логіки.
- Оптимізувати завантаження бота, підключаючи лише необхідні модулі.

Бібліотеки та простори імен Система використовує стандартні та спеціалізовані бібліотеки Python:

- discord.py – основа для взаємодії з Discord API (робота з повідомленнями, користувачами, серверами).
- discord.ext.commands – розширення для створення команд, обробки подій та роботи з Cog-модулями.
- sqlite3 – система зберігання даних (користувацькі профілі, економіка, налаштування).
- Asyncio – асинхронне виконання операцій для підвищення продуктивності.
- Допоміжні модулі (random,os,json,datetime) – обробка випадкових подій, робота з файлами, часові обмеження.

Для реалізації продукту *Discord «PrinceBot* нами було визначено ключові класи та їх призначення:

- BotClient (наслідує commands.Bot) – головний клас, що ініціалізує з'єднання з Discord, завантажує модулі та обробляє базові події (наприклад, on_ready).
- LevelSystem – система досвіду та рівнів з механікою нарахування балів за активність.
- EconomyModule – управління віртуальною валютою (перекази, баланс, транзакції).
- ShopModule – інтерактивний магазин з можливістю купівлі предметів.
- ModerationModule – інструменти модерації (бан, мут, попередження) з логуванням дій.

- GamesModule – розважальні міні-ігри ("Слоти", "Лотерея", "Вгадай число").
- ApplicationModule – система подачі заявок через реакції.
- WelcomeModule – автоматичне привітання нових учасників сервера.
- Приклади методів та їх логіка
- on_ready()– викликається при успішному підключенні бота, виводить статус у консоль.
- add_xp(user_id)– додає досвід користувачу з перевіркою на досягнення нового рівня.
- buy_item(ctx, item_name)– перевіряє наявність грошей, видає предмет та оновлює баланс.
- warn_user(ctx, member)– записує попередження в базу даних і повідомляє адміністрацію.
- play_slots(ctx) – імітує гру в слоти з випадковими результатами та виграшами.

Така побудова архітектури програмного продукту *Discord «PrinceBot* надає наступні переваги:

- Масштабованість. Додавання нових модулів не вимагає змін у існуючому коді.
- Надійність. Ізоляція помилок (збій у одному Cog не впливає на інші).
- Гнучкість. Можливість відключати окремі функції без зупинки бота.
- Зручність розробки. Чітка структура спрощує колаборацію в команді.

Висновок до розділу 2

Архітектура Discord-бота "PrinceBot" побудована на принципах модульності, гнучкості та масштабованості, що дозволяє ефективно

інтегрувати різноманітні функції – від економіки та модерації до розваг і автоматизації. Використання сучасних технологій, таких як Python, discord.py та SQLite, гарантує стабільну роботу, легкість у підтримці та можливість подальшого розширення. Модульний підхід із чітким розділенням відповідальності спрощує розробку, тестування та впровадження нових можливостей, роблячи бота ідеальним рішенням для управління серверами Discord, залучення користувачів та створення інтерактивного середовища.

РОЗДІЛ 3
ОСОБЛИВОСТІ РЕАЛІЗАЦІЇ ПРОГРАМНОГО ПРОДУКТУ DISCORD «PRINCEBOT

3.1 Загальний опис програмного продукту

Завдяки поєднанню різних функцій у єдиному продукті розроблюваний бот має значні переваги над відомими аналогами, як показано в таблиці 1.1.

Переваги над конкурентами:

- гнучка система рівнів
- розширена аналітика
- інтерактивність
- модерація
- орієнтація на спільноту

Недоліки з конкурентами є обмежена функціональність та залежність від серверних ресурсів

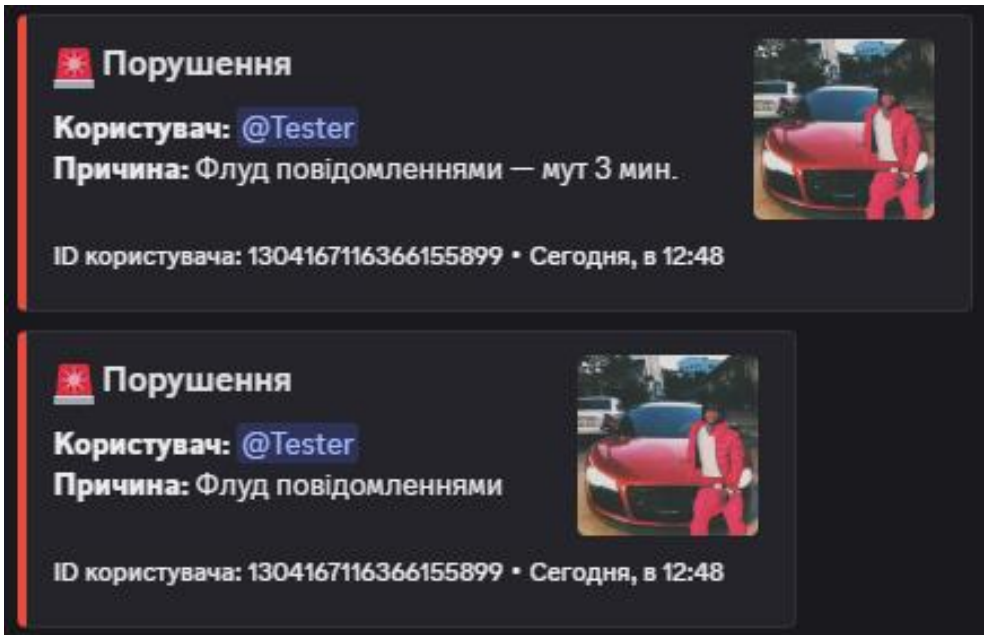

Рисунок 3.1 – Робота модераційного «PrinceBot»

Джерело: розроблено автором

На рис 3.1 зображено принцип роботи модераційного бота. Якщо користувач відправляє надмірну кількість повідомлень за короткий проміжок часу, бот автоматично виявляє таку активність і надсилає попередження про можливий спам.

У разі, якщо користувач продовжує надмірно відправляти повідомлення, бот застосовує покарання у вигляді мутатування на певний час. Час мута залежить від інтенсивності порушення та кількості попереджень, виданих користувачеві. Такий підхід дозволяє підтримувати порядок у чаті та захищати учасників від спаму, зберігаючи комфортне спілкування для всіх.

Таблиця 3.1 – Технологічний стек Discord «PrinceBot

Компонент	**Технологія**	**Опис**
Мова програмування	Python 3.11	Простий синтаксис,багата екосистема,асинхронна підтримка
Бібліотека Discord	discord.py	Повна підтримка слш-команд, кнопок, модальних вікон
База Даних	SQLite	Легкість використання для невеликих проектів, не потребує окремого сервера

Розроблений бот є спеціалізованим програмним рішенням, призначеним для автоматизації та покращення взаємодії користувачів у месенджері Discord. Він був створений з використанням сучасних інструментів розробки, зокрема мови програмування Python, яка забезпечує високу продуктивність, гнучкість та простоту підтримки коду. Для інтеграції з платформою Discord застосовано бібліотеку discord.py, що надає зручний та ефективний інтерфейс для роботи з Discord API.

Ця бібліотека (рис 3.2) дозволяє реалізовувати широкий функціонал, включаючи обробку повідомлень, керування користувачами, створення

автоматичних відповідей та інтеграцію з іншими сервісами. Використання discord.py забезпечує стабільність роботи бота, а також спрощує процес його налаштування та розширення можливостей у майбутньому.

Таким чином, програмний продукт відповідає сучасним вимогам до швидкодії, надійності та зручності використання, що робить його ефективним інструментом для вирішення поставлених завдань.

Головний вхідний файл програми, який відповідає за ініціалізацію та запуск Discord-бота. У цьому модулі здійснюється конфігурація клієнта бота, підключення основних обробників подій та завантаження команд. Також тут можуть знаходитися глобальні налаштування, такі як префікс команд, статус бота тощо.

```python
import discord
from discord.ext import commands, tasks
import random
import json
from antispam import handle_antispam
import economy
from shop import Shop
from economy import Economy
import aiosqlite
import os
from discord import Embed
import sqlite3
import time
from dotenv import load_dotenv

load_dotenv()
TOKEN = os.getenv("DISCORD_TOKEN")

intents = discord.Intents.default()
intents.message_content = True
intents.members = True
intents.messages = True
intents.guilds = True
intents.voice_states = True
intents.reactions = True
bot = commands.Bot(command_prefix='!', intents=intents)



@bot.event
async def on_ready():
    print(f'Бот {bot.user} підключений до Discord!')
```

Рисунок 3.2 – Discord «PrinceBot»

Джерело: розроблено автором

commands/

Каталог, що містить модулі окремих команд бота (рис 3.3). Кожна команда реалізована у вигляді окремого файлу або класу, що дозволяє легко додавати нові функції або модифікувати існуючі. Така організація спрощує розробку та тестування, оскільки зміни в одній команді не впливають на інші.

```
@bot.command(name='commands')
async def show_commands(ctx):
    embed = discord.Embed(
        title="📜 Список команд",
        description="Ось доступні команди бота:",
        color=discord.Color.blue()
    )

    embed.add_field(name="🕯️ Общие", value="""
`!stats [користувач]` - Показати статистику користувача
`!Hello` - Привітання від бота
``!ping` - Показати пінг бота
`!8ball [питання?]` - Магічна куля відповість на твоє запитання
""", inline=False)

    embed.add_field(name="🪙 Економіка", value="""
`!balance` - Перевірити баланс
`!work` - Отримати щоденну нагороду
`!give @користувач <сума>` - Передати монети
`!shop` - Відкрити магазин
`!buy <предмет>` - Купити предмет
`!removeitem <назва>` - Видалити предмет із магазину
`!additem <назва> <ціна> <кількість>` - Додати предмет у магазин
""", inline=False)

    embed.add_field(name="🎮 Игры", value="""
`!slots <ставка>` - Ігровий автомат
`!guess <ставка> <число від 1 до 10>` - Вгадай число
""", inline=False)

    embed.add_field(name="📈 Уровни", value="""
`!lvl [користувач]` - Показати рівень і досвід
`!givelvl @користувач <рівень>` - Видати рівень (Можуть використовувати адміни)
""", inline=False)

    embed.add_field(name="👮 Модерація", value="""
`!apply` — Подати заявку на модератора
""", inline=False)

    embed.set_footer(text="Для подробиць по кожній команді - звернися до розробника 😊")
    await ctx.send(embed=embed)
```

Рисунок 3.3 – Discord «PrinceBot»

Джерело: розроблено автором

utils/

Містить допоміжні функції, утиліти та сервісні класи, які використовуються в різних частинах програми. Наприклад, тут можуть бути реалізовані:

- функції для обробки даних,
- класи для роботи з API сторонніх сервісів,
- декоратори для обробки помилок,

- інші допоміжні інструменти, що спрощують розробку.

config/

Включає конфігураційні файли, такі як:

- токени доступу до Discord API та інших сервісів,
- налаштування роботи бота,
- параметри підключення до бази даних
- інші змінні середовища.

Ці дані зазвичай зберігаються у форматі .env для забезпечення безпеки та гнучкості налаштувань (рис 3.4).

Рисунок 3.4 – Discord «PrinceBot»

Джерело: розроблено автором

data/

Локальні бази даних (SQLite),

кешовані файли (наприклад, тимчасові дані користувачів),

логи роботи бота,

інші структуровані або неструктуровані дані.

Рисунок 3.5 – Discord «PrinceBot»

Джерело: розроблено автором

Для організації зберігання даних у програмному продукті було обрано SQLite – легку, вбудовану реляційну систему управління базами даних. Головною перевагою цього рішення є те, що SQLite не вимагає окремого серверного середовища для функціонування, що значно спрощує процес розгортання бота та знижує вимоги до системних ресурсів.

Обґрунтування вибору SQLite (рис 3.6-3.8). Локальна робота без необхідності підключення до сервера – усі дані зберігаються у локальному файлі, що робить систему автономною та незалежною від зовнішніх джерел.

Простота інтеграції з Python – завдяки вбудованій бібліотеці sqlite3 мова Python надає зручний інструментарій для роботи з базою даних без необхідності використання додаткових модулів.

Оптимальна продуктивність для невеликих обсягів даних – SQLite ефективно справляється із завданнями, пов’язаними із зберіганням та обробкою даних у малих та середніх проектах, що цілком відповідає вимогам даного рішення.

Таким чином, використання SQLite дозволяє забезпечити стабільну, ефективну та простоту в обслуговуванні систему зберігання даних, що є оптимальним варіантом для реалізації поставлених завдань.

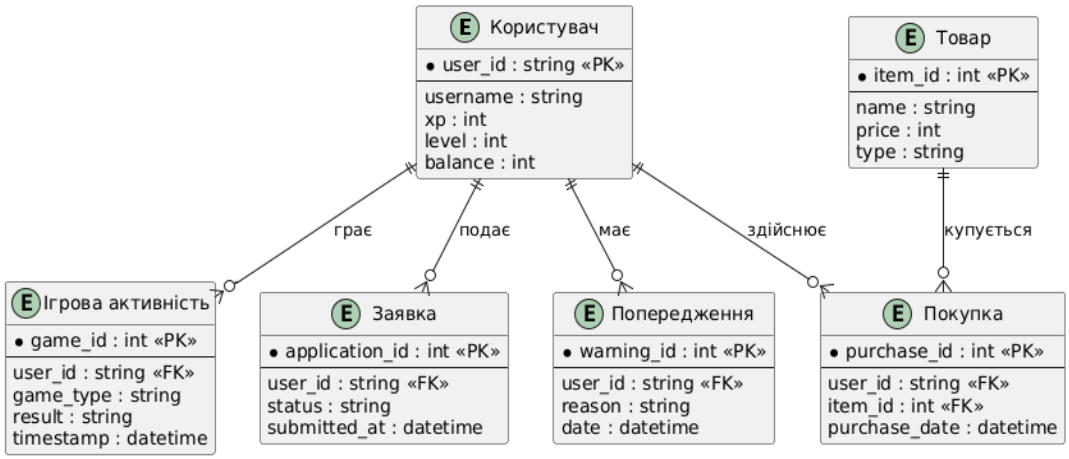

Рисунок 3.6 – Diagram ER «PrinceBot»

Джерело: розроблено автором

	user_id	balance
	Filter	Filter
1	275637863331397633	100000
2	1013054026604085278	8071099524000
3	1031910605189300275	300
4	1285636094628597821	100000
5	1304167116366155899	100000

Рисунок 3.7 – Discord «PrinceBot»

Джерело: розроблено автором

```python
async def leaderboard(self):
        async with aiosqlite.connect(self.db_path) as db:
            async with db.execute("SELECT user_id, balance FROM users ORDER BY balance DESC LIMIT 10") as cursor:
                return await cursor.fetchall()
```

Рисунок 3.8 – Discord «PrinceBot»

Джерело: розроблено автором

```python
def create_db():
    conn = sqlite3.connect("leveling.db")
    c = conn.cursor()
    c.execute("""
    CREATE TABLE IF NOT EXISTS users (
        user_id INTEGER PRIMARY KEY,
        xp INTEGER DEFAULT 0,
        level INTEGER DEFAULT 1
    )
    """)
```

Рисунок 3.9 – Discord «PrinceBot»

Джерело: розроблено автором

На (*рис* 3.9) зображенна струкура таблиці Level:

- user_id: Discord ID користувача.
- xp: кількість набраного досвіду.
- level: поточний рівень.

3.2 Тестування програмного продукту Discord «Princebot»

Для забезпечення надійності та стабільності роботи програмного продукту було проведено комплексне функціональне тестування (рис 3.10). Тестування охопило всі ключові аспекти роботи системи та дозволило перевірити її відповідність вимогам.

Функціональне тестування включало наступні етапи:

- тестування обробки команд користувача;
- детально перевірено коректність розпізнавання вхідних запитів;
- протестовано логіку обробки як стандартних, так і нестандартних команд;
- перевірено реакцію системи на помилкові введення;
- оцінено швидкість відгуку на запити;
- тестування роботи з даними;
- перевірено коректність запису даних у сховище;
- протестовано точність читання та відновлення інформації;
- оцінено цілісність даних при тривалій роботі;
- перевірено обробку крайових випадків при роботі з даними;
- тестування стабільності системи;
- проведено тривалі навантажувальні тести (до 72 годин безперервної роботи);
- оцінено споживання ресурсів під час роботи;
- проаналізовано поведінку системи при пікових навантаженнях;
- перевірено відновлюваність після аварійного завершення.

Додатково було проведено перехресне тестування на різних платформах і тестування сумісності з різними версіями ОС.

Результати тестування показали високу стабільність системи та повну відповідність заявленим функціональним вимогам. Усі виявлені під час тестування незначні помилки були успішно виправлені.

Рисунок 3.10 – етап тестування Discord «PrinceBot»

Джерело: розроблено автором

3.3 Інструкція користувачеві

Для початку роботи з ботом необхідно виконати наступні кроки:

Додавання бота на сервер Discord

1. Перейти за спеціальним OAuth2 посиланням, яке надає розробник
2. Обрати потрібний сервер зі списку доступних
3. Підтвердити додавання бота
4. Налаштування прав доступу

Обов'язково надати боту такі права:

- читання повідомлень
- надсилання повідомлень
- керування повідомленнями
- доступ до історії чату

Рекомендовано додавати надати права на:

- використання вбудованих посилань
- додавання реакцій
- система команд

Усі команди викликаються з префіксом "!"

Розглянемо основні команди боту (рис 3.11).

!helpme - виводить повний список доступних команд з описом

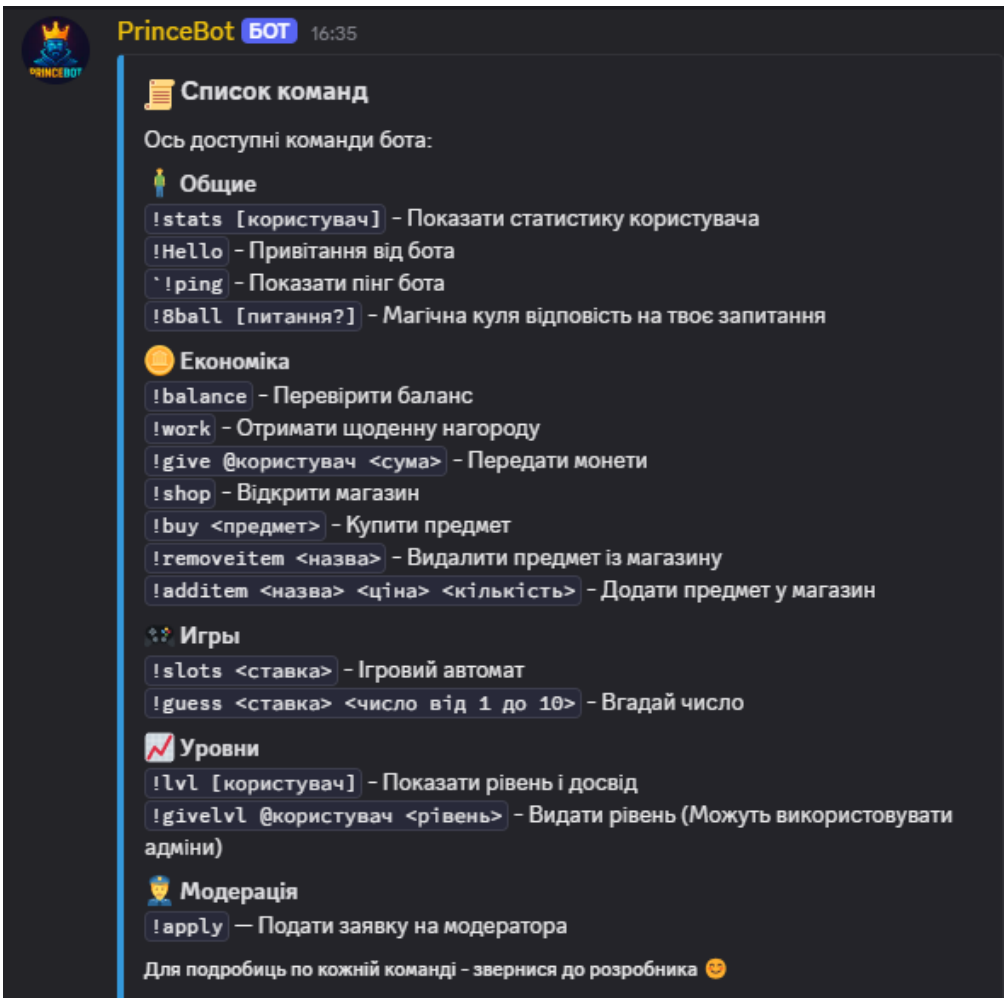

Рисунок 3.11 – Список команд Discord «PrinceBot»

Джерело: розроблено автором

!hello – привітальне повідомлення. Команда щоб зрозуміти чи відгукується бот на повідомлення (рис 3.12).

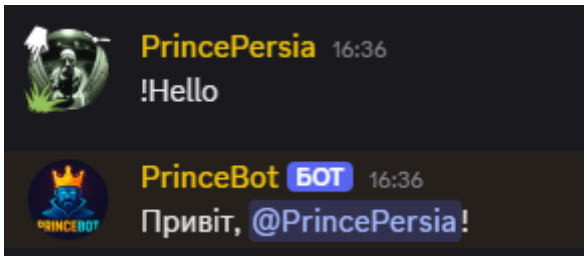

Рисунок 3.12 – Команда «!hello» Discord «PrinceBot»

Джерело: розроблено автором

Додаткові команди:

!stats – виводить статиску користувача на сервері (рис 3.13)

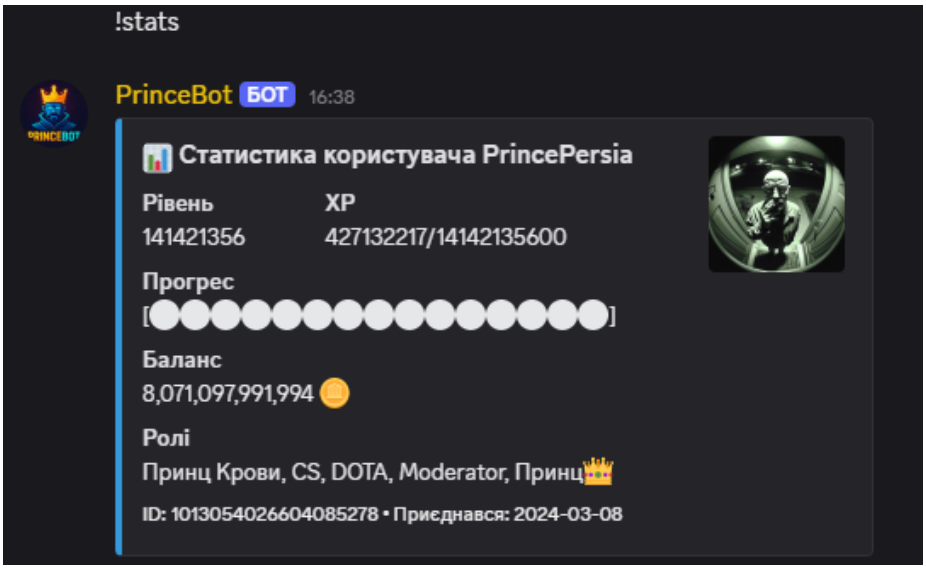

Рисунок 3.13 – Команда «!stats» Discord «PrinceBot»

Джерело: розроблено автором

!apply – запит на ролі модерації (рис 3.14)

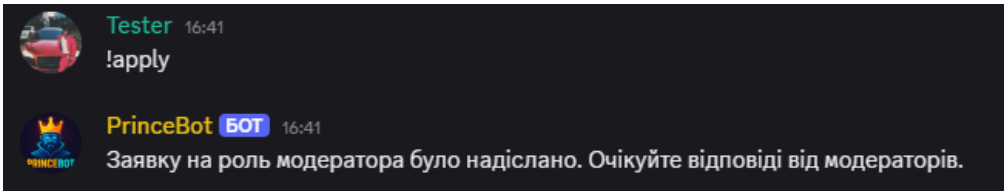

Рисунок 3.14 – Команда «!apply» Discord «PrinceBot»

Джерело: розроблено автором

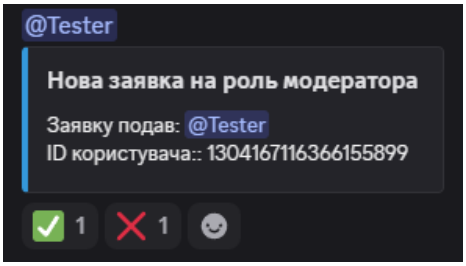

Рисунок 3.15 – Команда «!apply» Discord «PrinceBot» - 2

Джерело: розроблено автором

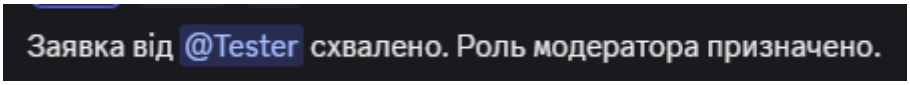

Рисунок 3.16 – Команда «!apply» Discord «PrinceBot» - 3

Джерело: розроблено автором

Висновки до розділу 3

У цьому розділі було розглянуто реалізацію програмного продукту — Discord-бота, створеного відповідно до проєктних рішень, викладених у попередніх розділах. Було описано архітектуру програмного забезпечення, основні складові системи, вибрані технології та методи комп’ютерних наук, що застосовувались під час розробки.

Для зберігання даних було обґрунтовано використання вбудованої бази даних SQLite, яка забезпечує зручність та ефективність для невеликих проєктів, що не потребують високого навантаження.

Проведено функціональне та модульне тестування програмного продукту, що підтвердило його працездатність та відповідність поставленим вимогам. Також наведено інструкцію користувача для ефективного використання можливостей бота.

ВИСНОВОК

У межах кваліфікаційної роботи було успішно реалізовано повноцінного Discord-бота «PrinceBot», призначеного для автоматизації керування сервером, підвищення активності користувачів і створення інтерактивного середовища.

Проєкт об’єднує низку ключових функцій, серед яких – система рівнів і досвіду, економічний модуль, модуль модерації, ігрові механіки, вітання нових користувачів, система заявок та магазин із внутрішньою валютою. Завдяки модульній структурі архітектури, кожен елемент системи функціонує незалежно, що значно полегшує підтримку, масштабування та впровадження нових можливостей.

Для реалізації проєкту обрано сучасний стек технологій, зокрема мову програмування Python, бібліотеку discord.py, яка забезпечує ефективну взаємодію з Discord API, та базу даних SQLite, що дозволяє зручно зберігати всю необхідну інформацію про користувачів, транзакції та активність. Усі компоненти системи були протестовані, що гарантує стабільну та безпечну роботу продукту навіть у разі тривалого навантаження.

Розроблений бот демонструє переваги над існуючими аналогами завдяки поєднанню широкого функціоналу, інтуїтивно зрозумілого інтерфейсу, підтримки української мови та відсутності обмежень, пов’язаних із преміум-доступом. Він може бути корисним для будь-яких Discord-серверів, незалежно від тематики чи розміру спільноти, і є ефективним інструментом для створення активного та дружнього середовища.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Основне джерело інформації про можливості Discord API. URL:https://discord.com/developers/docs/intro (дата звернення: 15.04.2025.)

2. MEE6 – Discord Bot for Moderation, Leveling, Twitch, youtube. Офіційний сайт.URL: https://mee6.xyz/en/ (дата звернення: 15.04.2025.)

3. Dyno – The Discord Bot for Moderation and Custom Commands. Офіційний сайт. url: https://dyno.gg (дата звернення: 15.04.2025.)

4. Dank Memer – A Discord Bot with Memes, Economy and More. Офіційний сайт.URL: https://dankmemer.lol (дата звернення: 15.04.2025.)

5. Фармінг – це процес систематичного накопичення ресурсів, досвіду, грошей або інших ігрових благ у відеоіграх, зазвичай через багаторазове виконання одних і тих самих дій. Гравці займаються фармінгом, щоб швидше розвивати свого персонажа, отримувати цінні предмети або підвищувати свій рівень у грі.URL: https://chatgpt.com (дата звернення: 15.04.2025.)

6. загальна інформація про web/API. url:https://developer.mozilla.org (дата звернення: 15.04.2025.)

7. Discord developer portal. URL: https://discord.com/developers/docs (дата звернення: 15.04.2025.)

8. Lutz, M. Learning Python. Sebastopol: O'Reilly Media, 2013. 1648 c.

9. Nguyen, T. Gamification in Chatbot Systems: XP, Levels, and Engagement // ACM Conference, 2021.

10. Smith, J. Chatbot Development in Python for Discord Platform // International Journal of AI Research, 2022.

11. Гіди та туторіали з Python URL https://realpython.com (дата звернення: 15.04.2025.)

12. Загальна інформація про web/API URL https://developer.mozilla.org (дата звернення: 15.04.2025.)

13. Стаття про Python URL https://uk.wikipedia.org/wiki/python (дата звернення: 15.04.2025.)