Міністерство освіти і науки України
Рівненський державний гуманітарний університет
Кафедра інформаційних технологій та моделювання

Кваліфікаційна робота

За освітнім ступенем «бакалавр»

На тему:

СУЧАСНІ ТЕХНОЛОГІЇ РОЗРОБКИ ІНФОРМАЦІЙНИХ ЧАТ-БОТІВ

Виконав:

здобувач IV курсу

групи ІПЗ-41

спеціальності 121 «Інженерія програмного забезпечення»

Якимчук Ілля Юрійович

Науковий керівник:

ст. викл. Кирик Т.А.

Рівне – 2024

ЗМІСТ

ВСТУП

У сучасному світі інформаційні технології стали невід'ємною частиною повсякденного життя. Однією з перспективних областей розвитку є створення чат-ботів, які можуть автоматично відповідати на запити користувачів у текстовій або голосовій формі. Чат-боти здатні забезпечувати високий рівень взаємодії та індивідуального підходу до кожного користувача, що робить їх незамінними у різних сферах: від обслуговування клієнтів до освітніх і медичних послуг.

У даній роботі буде представлений приклад розробки чат-боту про розклад занять факультету, який стане корисним інструментом для студентів та викладачів.

Актуальність теми зумовлена зростаючою популярністю чат-ботів та їх використанням у різних галузях, таких як обслуговування клієнтів, підтримка користувачів, освітні послуги. Чат боти стають все більш затребуваними завдяки своїй здатності надавати швидкі та ефективні відповіді на різні запити.

Зокрема, у сфері освіти інформаційні чат-боти можуть значно покращити доступ до розкладу занять, оперативно інформуючи студентів та викладачів про зміни в розкладі, відміни занять та інші важливі події. Це дозволяє зменшити навантаження на адміністративний персонал і забезпечити більш ефективне управління навчальним процесом. Дослідження сучасних технологій розробки інформаційних чат-ботів є необхідним для подальшого вдосконалення цих систем, їх інтеграції у навчальні процеси та забезпечення їхньої ефективної роботи в освітніх установах.

Метою даного дослідження є вивчення сучасних технологій розробки інформаційних чат-ботів та створення ефективного чат-боту для оповіщення про розклад факультету. Це включає аналіз існуючих технологій і методів, проектування архітектури системи, розробку та тестування чат-боту, який зможе оперативно інформувати студентів та викладачів про зміни в розкладі, відміни занять та інші важливі події. Дослідження також спрямоване на оцінку впливу використання чат-боту на підвищення ефективності управління навчальним процесом.

Об'єктом дослідження є інформаційні чат-боти, їх структура, принципи функціонування та методи розробки.

Предметом дослідження є сучасні технології та методи розробки інформаційних чат-ботів на прикладі чат-боту оповіщення про розклад занять.

Цілі дослідження:

- *Розгляд технологій створення чат-ботів.* Дослідити сучасні технології та інструменти для розробки інформаційних чат-ботів;
- *Покращення доступу до інформації.* Забезпечити студентам легкий та швидкий доступ до актуального розкладу занять через інтерфейс чат-боту;
- *Оптимізація взаємодії.* Забезпечити можливість отримання інформації безпосередньо в месенджері, що дозволяє студентам ефективно планувати свій час.

Завдання дослідження:

- Проаналізувати вимоги користувачів щодо функціональності та зручності чат-боту для оповіщення про розклад факультету.
- Вибрати оптимальні технології та архітектуру для розробки чат-боту.
- Розробити функціональний прототип чат-боту та провести його тестування.
- Визначити методи оптимізації та покращення функціональності чат-боту на основі зворотного зв'язку від користувачів.

При виконанні завдань кваліфікаційної роботи були використані наступні **методи** дослідження:

- *Метод аналізу.* Вивчення вимог користувачів. Збір вимог від потенційних користувачів чат-боту через опитування та інтерв'ювання. Визначення основних функціональних і нефункціональних вимог. Дослідження існуючих засобів розробки для програмування чат-ботів;
- *Метод проектування.* Вибір технології та архітектури. Оцінка різних технологій та архітектур для створення чат-боту. Вибір оптимальних засобів, які задовольняють вимоги проекту. Проектування інтерфейсу користувача. Створення прототипів та дизайну інтерфейсу чат-боту для забезпечення зручності взаємодії та легкого сприйняття інформації;

- *Експериментальний метод.* Розробка функціонального прототипу чат-боту, який включав основні функції, передбачені у проекті. Тестування та виправлення помилок, покращення UX чату.

Застосування вищеназваних методів допомогло отримати як результат функціональний прототип чат-боту, який відповідає основним вимогам проекту та задовольняє потреби користувачів.

Апробація і впровадження результатів дослідження. Результати роботи були представлені у доповіді на звітній науково-практичній конференції професорсько-викладацького складу, аспірантів та студентів Рівненського державного гуманітарного університету (16 травня 2024 року),

Структура роботи. Дипломна робота складається зі вступу, п’ятьох розділів, висновків та списку використаних джерел.

У першому розділі дипломної роботи детально розглядаються особливості створення чат-ботів. Починаючи з визначення поняття "чат-бот" і описування важливих функцій та можливостей цих інтелектуальних агентів, розділ також аналізує використання чат-ботів у сфері освіти. Основною метою цього розділу є дослідження ключових аспектів розробки та функціональних можливостей чат-ботів, зокрема їхнє використання в освітній галузі та роль у поліпшенні комунікації між студентами та навчальним закладом.

У другому розділі дипломної роботи детально розглядається архітектура системи створення чат-ботів. Описуються основні етапи вибору мови програмування, ролі Python у створенні чат-ботів, а також вибір соціальної мережі для їхнього розгортання. Метою цього розділу є вивчення технічних аспектів розробки чат-ботів та їхніх інтеграцій з різними платформами та сервісами.

Третій розділ дипломної роботи присвячено бібліотекам та процесу створення чат-ботів. Описується робота з Python в Telegram, встановлення IDE PyCharm та бібліотеки pyTelegramBotAPI. Також детально розглядаються підтримувані бібліотеки та середовища розробки. Метою цього розділу є ознайомлення з інструментами та технологіями, які використовуються для розробки чат-ботів.

У четвертому розділі дипломної роботи описується процес розробки конкретного чат-боту для оповіщення про розклад занять. Розглядаються основні кроки розробки, такі як створення команд чат-боту, клавіатури з розкладом, додаткових кнопок, логіки зчитування даних з гугл диску та обробки повідомлень від користувачів.

У п'ятому розділі дипломної роботи проводиться тестування та оцінка результатів роботи чат-боту. Розглядаються результати тестування, виявлені проблеми та можливі шляхи вдосконалення розробленого чат-боту.

РОЗДІЛ 1. ОСОБЛИВОСТІ СТВОРЕННЯ ЧАТ-БОТІВ

1.1. Поняття чат-боту

Чат-бот – це програма, яка призначена для автоматичної відповіді на запитання користувачів у чаті. Бот може взаємодіяти з користувачем природною мовою і виконувати різні завдання, відповідати на запитання, допомагати з вибором товарів або послуг, обробляти замовлення, проводити опитування тощо. Як правило, чат-бот пропонує користувачеві вибрати один з наданих варіантів. Це означає, що отримання інформації або виконання інших завдань здійснюється шляхом вибору з наданих відповідей або категорій, без необхідності вводити текст у вікно чату. Більш просунуті боти дозволяють вводити текстові запити та отримувати інформацію відповідно до запиту користувача [1].

Першими відомими чат-ботами в історії вважаються ELIZA (1966) та PARRY (1972). Після них з'явилися такі боти, як A.L.I.C.E., Jabberwacky, D.U.D.E. та інші. Чат-бот PARRY був розроблений американським психіатром Кеннетом Колбі в 1972 році. Програма моделювала поведінку пацієнта з параноїдальною шизофренією, що нагадувала мислення звичайної людини. PARRY використовував складну систему припущень, атрибуцій та «емоційних відповідей» і був перевірений за допомогою тесту Тюрінга на початку 1970-х років.

Чат-боти створюються з різними цілями та завданнями. Вони поділяються на кілька типів:

1. Інформаційні:
2. Комерційні;
3. Розважальні;
4. Персональні тощо.

Кожен із них має свої особливості, вимоги до розробки та нюанси у використанні [2].

1.2. Важливі функції та можливості

Створення та налаштування чат-бота може бути складним процесом. Якщо ви хочете створити бота самостійно, то для цього вам знадобляться знання програмування, зокрема, мов програмування Python, JavaScript або Java. Для створення бота вам також знадобиться розуміння роботи з базами даних, обробки природної мови, алгоритмів та інтерфейсу користувача.

При створенні чат-бота важливо враховувати ряд ключових функцій та можливостей для забезпечення його ефективності та корисності:

1. *Взаємодія з користувачем*: Здатність розпізнавати та відповідати на повідомлення користувачів, враховувати команди, запити та запити на допомогу.
2. *Підтримка команд*: Визначення команд, які сприймає бот, та відповідні дії, пов'язані з цими командами. Наприклад, отримання розкладу, інформації про факультет та інше.
3. *Оповіщення та сповіщення*: Здатність надсилати сповіщення студентам про зміни в розкладі або інші важливі оголошення.
4. *Персоналізованість*: Можливість адаптації відповідей на запитання або команди залежно від контексту чи індивідуальних налаштувань користувача.
5. *Зберігання даних та відстеження історії*: Можливість зберігання інформації про запити користувачів та їхню історію взаємодії з ботом.
6. *Мультимедійні можливості*: Здатність надсилати не лише текстові відповіді, а й фотографії, відео, аудіофайли або інші мультимедійні матеріали.
7. *Аналіз та вдосконалення*: Можливість аналізувати взаємодію бота з користувачами для постійного вдосконалення функціоналу та відповідей.
8. *Інтеграція з базою даних*: Здатність отримувати та обробляти дані з бази даних для надання користувачам актуальної інформації.
9. *Мультиплатформеність*: Забезпечення можливості використання бота на різних платформах чи месенджерах для максимальної доступності для користувачів.

10. *Захист і конфіденційність*: Забезпечення безпеки та конфіденційності інформації, обміну даними між користувачем та ботом.
11. *Адаптивність*: Можливість швидко реагувати на зміни в інформації або потреби користувачів, оновлення функціоналу з метою покращення взаємодії.
12. *Підтримка інтерфейсу користувача*: Надання зручного та зрозумілого інтерфейсу для спілкування з ботом, що сприяє зручності та ефективності взаємодії.

Ці аспекти є ключовими при створенні чат-бота, оскільки дозволяють створити ефективний та корисний інструмент для користувачів, а також забезпечити надійну та ефективну роботу бота в різних сферах застосування.

1.3. Чат-боти у освітній сфері

Чат-боти стають все більш популярними в освітній сфері завдяки своїм численним перевагам і можливостям. Вони здатні автоматизувати багато рутинних процесів, покращуючи ефективність та зручність як для викладачів, так і для студентів. Зокрема, чат-боти можуть використовуватися для оповіщення про розклад занять, відповіді на часті запитання, надання доступу до навчальних матеріалів, проведення опитувань та збору зворотного зв'язку.

Однією з основних переваг використання чат-ботів у навчальних закладах є автоматизація рутинних завдань. Наприклад, чат-боти можуть автоматично надсилати повідомлення студентам про зміни в розкладі занять, нагадування про важливі події або дедлайни. Це дозволяє звільнити викладачів та адміністративний персонал від виконання цих завдань, що сприяє ефективнішому використанню їхнього часу.

Чат-боти можуть значно підвищити доступність інформації для студентів. Вони можуть відповідати на часті запитання щодо розкладу занять, розташування аудиторій, правил відвідування та інших аспектів навчального процесу. Це дозволяє студентам швидко отримувати необхідну інформацію без потреби звертатися до викладачів або адміністративного персоналу.

Чат-боти також можуть сприяти інтерактивному навчанню. Вони можуть проводити опитування, тести та вікторини, допомагаючи студентам закріплювати знання та отримувати зворотний зв'язок у режимі реального часу. Це сприяє підвищенню зацікавленості студентів у навчальному процесі та покращенню якості освіти.

Завдяки аналізу даних про студентів, чат-боти можуть пропонувати персоналізовані рекомендації щодо навчальних матеріалів, додаткових ресурсів або заходів, що можуть бути корисними для конкретного студента. Це дозволяє враховувати індивідуальні потреби та інтереси кожного студента, що сприяє покращенню результатів навчання.

У контексті дистанційного навчання та використання віртуальних навчальних середовищ, чат-боти відіграють важливу роль у забезпеченні комунікації та координації між студентами та викладачами. Вони можуть надавати підтримку у вирішенні технічних проблем, допомагати з доступом до онлайн-курсів та інших ресурсів, а також сприяти організації віртуальних зустрічей та занять.

Незважаючи на численні переваги, впровадження чат-ботів у освітній сфері має також свої виклики. Це включає питання безпеки даних, забезпечення конфіденційності, а також необхідність адаптації чат-ботів до специфічних потреб навчальних закладів. Проте, з розвитком технологій та зростаючою інтеграцією цифрових рішень у навчальний процес, використання чат-ботів у освіті має великі перспективи і потенціал для подальшого розвитку.

Чат-боти стають важливим інструментом у сучасній освіті, сприяючи автоматизації процесів, підвищенню доступності інформації, інтерактивності та персоналізації навчального процесу. Вони надають нові можливості для покращення якості освіти та оптимізації роботи навчальних закладів, що робить їх незамінними у цифрову епоху.

Ось приклади чат-ботів, які активно використовуються в освітній сфері:

Blackboard Chatbot: Цей чат-бот інтегрований у платформу Blackboard Learn, що використовується в багатьох університетах для дистанційного навчання. Він допомагає студентам отримувати інформацію про курси, завдання, дедлайни та інші аспекти навчання.

CampusNexus Chatbot: Цей чат-бот спеціально розроблений для вищих навчальних закладів і допомагає студентам і абітурієнтам отримувати інформацію про вступні вимоги, фінансову допомогу, розклади занять та інші аспекти життя на кампусі.

AdmitHub: Цей чат-бот спрямований на підтримку абітурієнтів у процесі вступу до вищих навчальних закладів. Він надає відповіді на питання щодо вступних вимог, термінів подачі документів, процедур реєстрації та інших питань, що стосуються вступу.

РОЗДІЛ 2. АРХІТЕКТУРА СИСТЕМИ

2.1 Вибір мови програмування для створення чат-бота

Python — це потужна мова програмування, яка легко засвоюється. Вона оснащена ефективними структурами даних високого рівня і забезпечує простий та ефективний підхід до об’єктно-орієнтованого програмування. Завдяки елегантному синтаксису та динамічній типізації, а також інтерпретованому характеру, Python є ідеальною мовою для написання сценаріїв та швидкої розробки додатків у різних сферах на багатьох платформах [8].

Python був створений Гвідо ван Россумом в 1990 році. Це високорівнева мова програмування, розроблена з урахуванням читабельності коду та простоти використання.

Історія Python включає такі ключові моменти:

1. *Початки*: Робота над Python розпочалася у кінці 80-х років. Перша версія була випущена у лютому 1991 року.
2. *Версія 2 та 3*: Розвиток мови призвів до двох ключових версій: Python 2.x і Python 3.x. Python 2 був популярним упродовж багатьох років, але у 2020 році його підтримка офіційно завершилась, рекомендується використовувати версію 3.
3. *Філософія Python*: Python має "The Zen of Python", набір принципів, які підкреслюють його дизайн і філософію. Ці принципи описують якість Python, включаючи читабельність коду і простоту використання.
4. *Розвиток та спільнота*: Python став дуже популярним завдяки своїй простоті, гнучкості і багатій екосистемі бібліотек. Велика спільнота розробників продовжує внесок у розвиток мови, додаючи нові можливості та покращення.

2.2. Роль Python у створенні чат-ботів

Python відіграє ключову роль у створенні чат-ботів завдяки декільком аспектам:

1. *Простота використання*: Python відомий своєю легкістю та зрозумілістю синтаксису, що робить його ідеальним для швидкого розвитку проектів, таких як чат-боти.
2. *Багатий вибір бібліотек*: Мова Python має різноманітні бібліотеки, призначені спеціально для роботи з чат-ботами.
3. *Підтримка API*: Python пропонує багато API, що спрощують інтеграцію з популярними платформами чат-ботів, такими як Telegram, Facebook Messenger, Slack тощо.
4. *Широкий спектр фреймворків*: Наявність різноманітних фреймворків дозволяє розробникам вибрати той, який найбільше відповідає їхнім потребам і створити бота за короткий термін.
5. *Загальноприйнятість*: Python - одна з найпопулярніших мов програмування, і це означає, що вона має велику спільноту розробників, що допомагає вирішувати проблеми та отримувати підтримку у розвитку чат-ботів.
6. *AI та ML*: Завдяки бібліотекам для штучного інтелекту та машинного навчання, які є сильною стороною Python, можна створювати чат-ботів зі здатністю розуміти та відповідати на запитання, аналізувати текст тощо.

2.3. Вибір соціальної мережі для створення чат-бота

Бот-платформа — це програма, на API якої можна створити віртуального співрозмовника. Сьогодні чат-боти підтримують більшість популярних месенджерів. Серед них Facebook Messenger, Viber, Whatsapp, Skype, Telegram, Sender, Wechat та навіть Instagram. Для розробки свого чат-боту я обрав таку соціальну мережу, як Telegram.

Ботів у Телеграмі легко розпізнати по приставці “bot” у назві. Це одна з обов’язкових вимог платформи до власників автоматизованих чатів.

Логіка чат-бота в Telegram контролюється за допомогою HTTPS запитів до API платформи.

РОЗДІЛ 3. БІБЛІОТЕКИ ТА СТВОРЕННЯ ЧАТ-БОТА

3.1. Робота з Python в Telegram. Опис функціоналу Python для роботи з API Telegram, можливості інтеграції

Python має багатий функціонал для взаємодії з API Telegram та створення чат-ботів. Ось деякі аспекти цієї роботи:

1. *Бібліотеки для роботи з Telegram API*: У Python існують різноманітні бібліотеки, що дозволяють взаємодіяти з Telegram API. Наприклад, python-telegram-bot, pyTelegramBotAPI та інші, які пропонують гнучкі можливості для створення чат-ботів.
2. *Можливості отримання та обробки повідомлень*: Python дозволяє отримувати та обробляти різні типи повідомлень від користувачів, такі як текстові повідомлення, зображення, відео, документи тощо. Він також надає можливість реагувати на різні команди користувачів.
3. *Робота з клавіатурою бота*: Python дозволяє створювати різноманітні клавіатури для чат-ботів в Telegram. Це може бути зручно для спрощення навігації користувачів та виконання певних команд.
4. *Можливості взаємодії з користувачем*: Python надає засоби для відправки повідомлень, відповідей на запитання, обробки команд та взагалі інтеракції з користувачами через Telegram API.
5. *Інтеграція з іншими сервісами*: Python дозволяє створювати чат-ботів, які можуть взаємодіяти з іншими сервісами, такими як бази даних, зовнішні API, веб-сайти тощо.
6. *Асинхронність та паралельність*: Python має можливості для асинхронного програмування, що може бути корисним для обробки великої кількості повідомлень або взаємодії з користувачами одночасно.

3.2. Підтримувані бібліотеки та IDE.

Зазвичай для створення чат-ботів на Python для платформи Telegram використовують такі бібліотеки та фреймворки:

1. *python-telegram-bot*: Це одна з найпопулярніших бібліотек для створення чат-ботів в Telegram на Python. Вона надає простий та зручний інтерфейс для розробки, дозволяючи отримувати повідомлення, створювати реакції на команди та взагалі взаємодіяти з API Telegram.
2. *pyTelegramBotAPI*: Ця бібліотека також є досить популярною. Вона має простий синтаксис та добре документована, що дозволяє створювати ботів на основі Telegram API.
3. *Aiogram*: Це ще один варіант для створення чат-ботів, який пропонує асинхронні можливості та гнучкий інтерфейс для взаємодії з API Telegram.

Популярні інтегровані середовища розробки (IDE), які підтримують розробку на Python:

1. *PyCharm*: Розроблений компанією JetBrains, цей IDE має різні версії для різних типів розробки, включаючи професійну, освітню та спеціалізовані версії.
2. *Visual Studio Code*: Це легкий та потужний текстовий редактор від Microsoft, який може бути розширений для роботи з Python за допомогою різноманітних розширень.
3. *Jupyter Notebook / JupyterLab*: Це інтерактивне середовище для розробки, де можна створювати та виконувати код у вигляді окремих комірок.
4. *Spyder*: Це IDE, яке поєднує в собі властивості наукових обчислень з можливостями IDE, спеціально створене для роботи з Python.
5. *IDLE*: Офіційна середа розробки Python, яка надає просте та базове середовище для написання коду.

Для створення свого чат-бота було обрано бібліотеку *pyTelegramBotAPI* та середовище розробки *PyCharm*.

Бібліотека *pyTelegramBotAPI* - це інструмент для створення та розробки чат-ботів для платформи Telegram на мові програмування Python. Вона дозволяє зручно взаємодіяти з API Telegram, спрощуючи процес розробки чат-бота та надаючи доступ до необхідних функцій для обробки повідомлень, створення відповідей та управління ботом через Python [7].

PyCharm — це інтегроване середовище розробки (IDE) для Python, що забезпечує розробникам потужний набір інструментів. Воно підтримує автодоповнення коду, налагодження, рефакторинг, віртуальні середовища та багато інших функцій. Завдяки своїм можливостям PyCharm робить процес програмування ефективнішим і зручнішим [9].

3.2.1. Встановлення PyCharm IDE для розробки

Перш ніж розпочати встановлення середовища розробки, було встановлено Python. Для цього потрібно перейти на офіційну сторінку Python, та завантажити саму нову версію. Завантажуючи інсталятор з офіційного сайту потрібно звернути увагу на сумісність з операційною системою, а також на стабільність версії (Рис. 3.1).

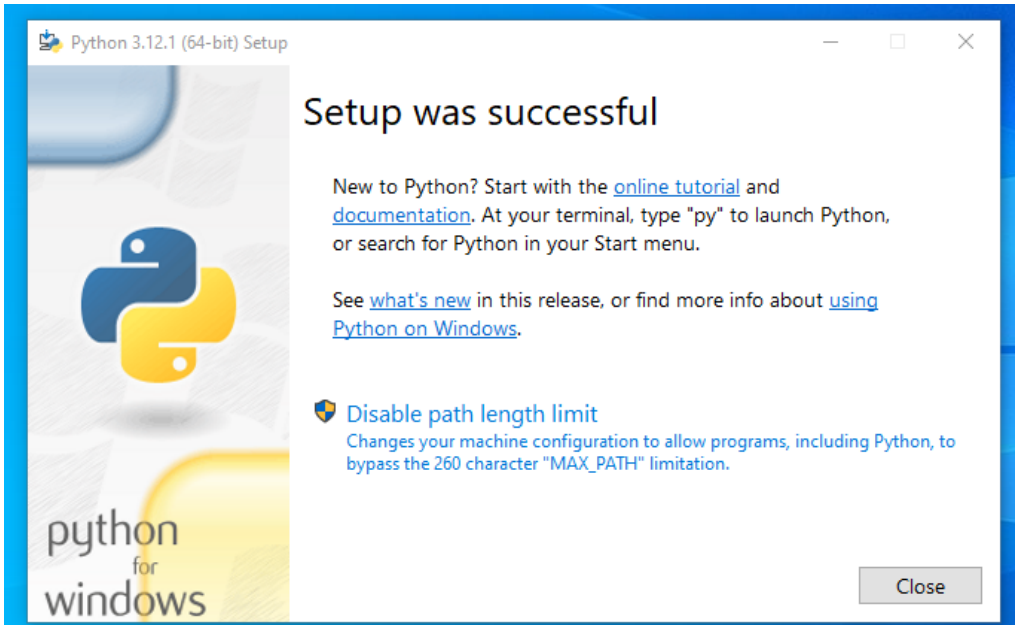

Рисунок 3.1. Успішне встановлення Python 3.12.1

Наступний крок, це завантаження IDE PyCharm. Для цього також потрібно перейти на офіційну сторінку JetBrains, обираємо PyCharm Community та завантажуємо інсталятор (Рис. 3.2, 3.3).

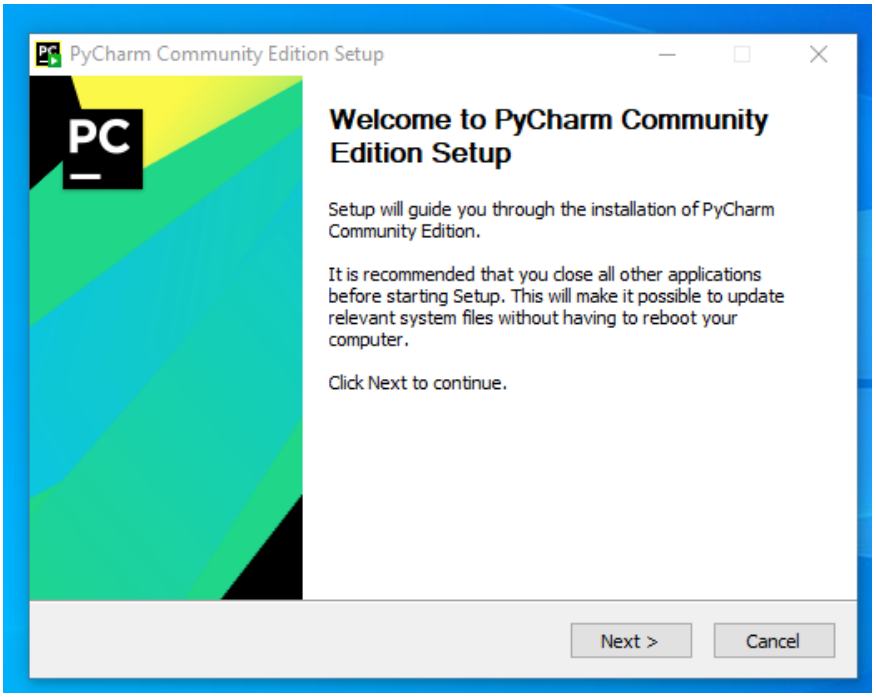

Рисунок 3.2. Процес встановлення PyCharm IDE

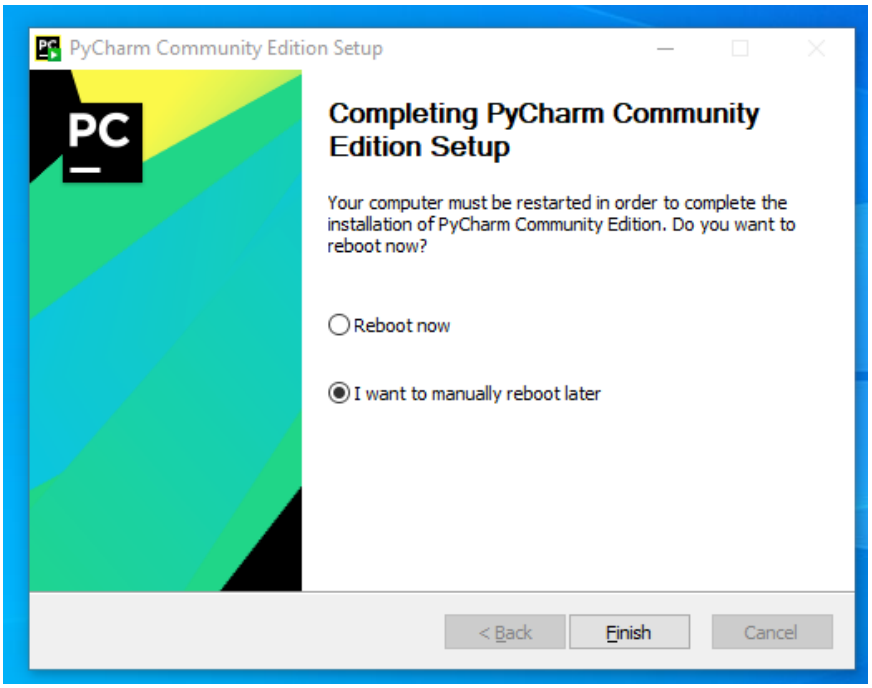

Рисунок 3.3. Успішне встановлення PyCharm IDE

Після успішного встановлення Python та PyCharm, запускаємо середовище розробки та створюємо новий проект. Назва мого проекту "main" (Рис. 3.4).

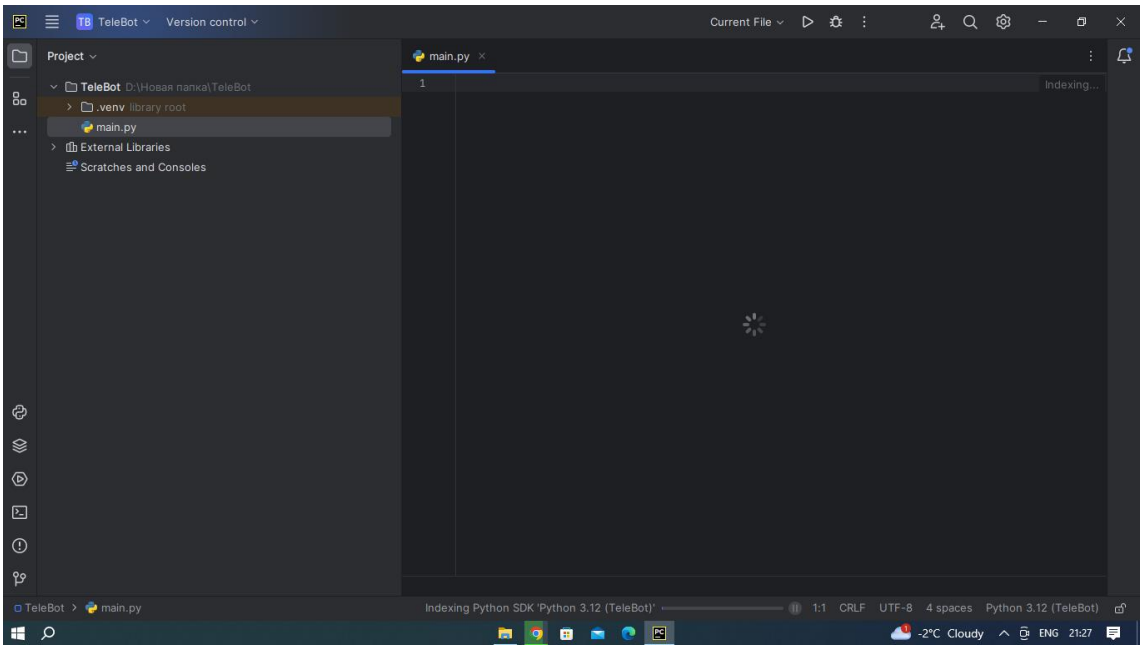

Рисунок 3.4. Запуск проєкту

3.2.2. Встановлення бібліотеки pyTelegramBotAPI

Щоб встановити бібліотеку pyTelegramBotAPI, потрібно перейти на сайт pypi.org/project/pyTelegramBotAPI, та скопіювати команду (Рис. 3.5) [7].

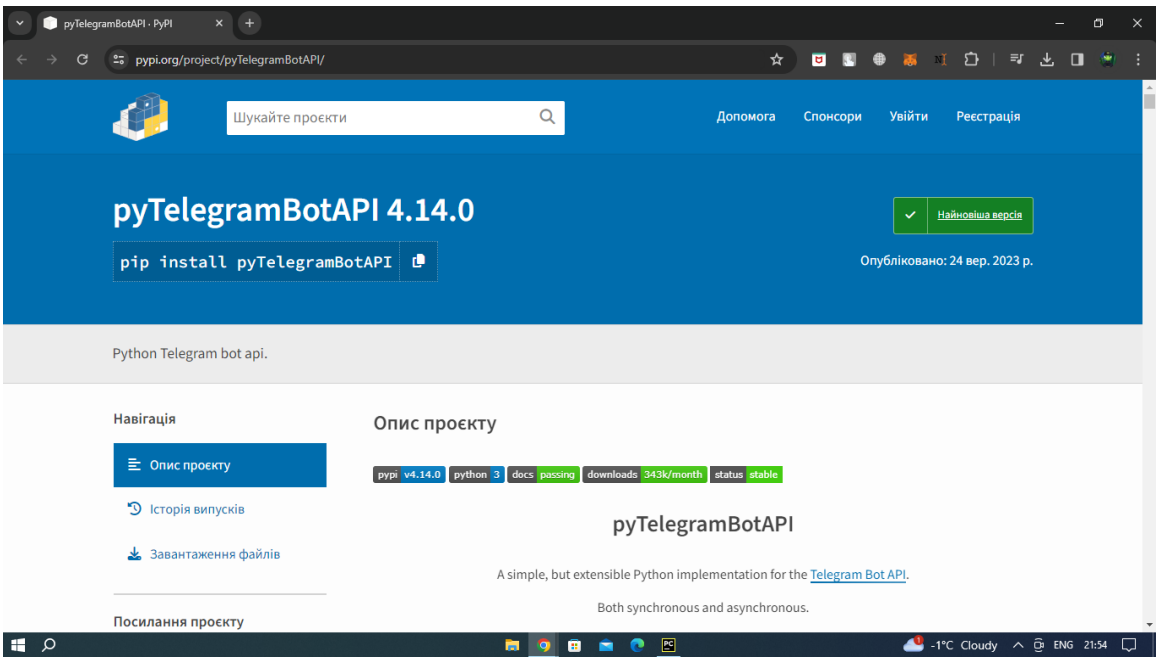

Рисунок 3.5. Розташування команди для встановлення бібліотеки

Потім слід відкрити термінал у своєму середовищі розробки та вставити скопійовану команду – pip install pyTelegramBotAPI (Рис. 3.6).

Pip — це інструмент для управління пакетами, призначений для інсталяції та адміністрування програмних пакетів, створених на мові Python [7].

Після натискання кнопки Enter бібліотека автоматично завантажилася та встановилася у проект (Рис. 3.7).

Цей метод дозволяє зручно інсталювати бібліотеку через командний рядок, надаючи необхідний функціонал для інтеграції з Telegram API у вашому Python-проекті [7].

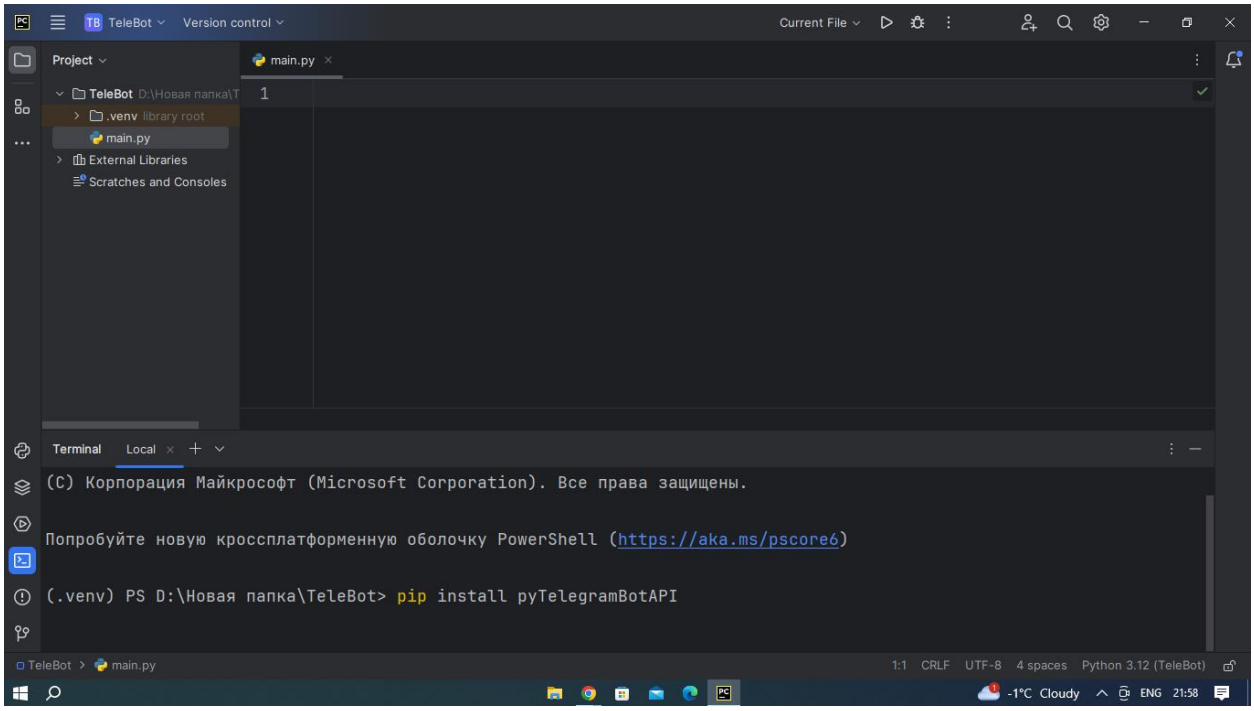

Рисунок 3.6. Введення команди в термінал

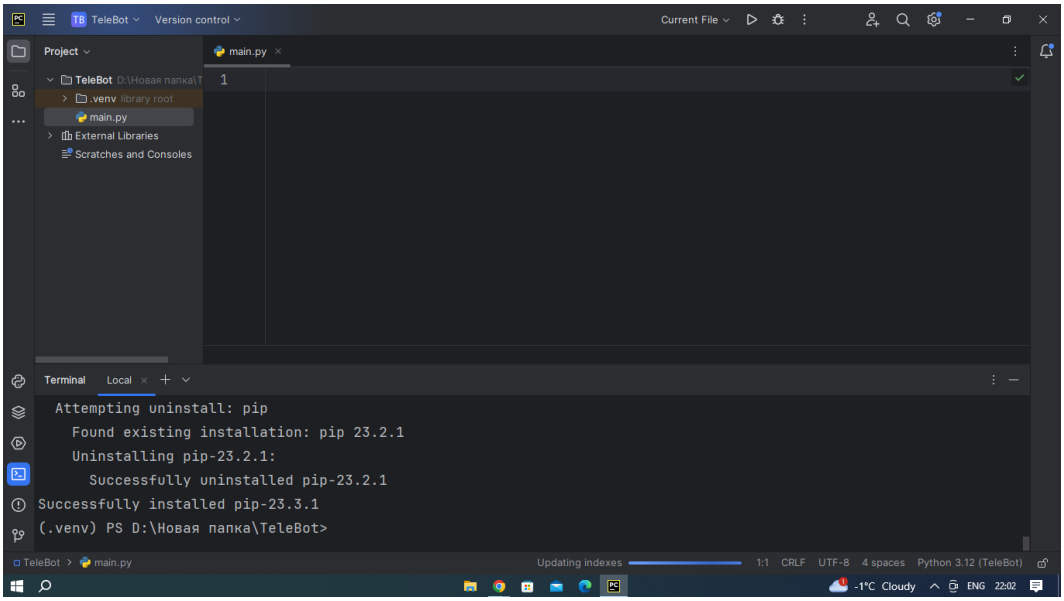

Рисунок 3.7. Успішне встановлення бібліотеки

3.3. Створення чат-боту та отримання ключа доступу

Після завершення вибору технологій та програмних засобів для реалізації проекту можна приступити до його виконання.

Спочатку слід відкрити Telegram та у поле пошуку ввести @botfather, після чого перейти до діалогу з цим ботом (Рис. 3.8) [10].

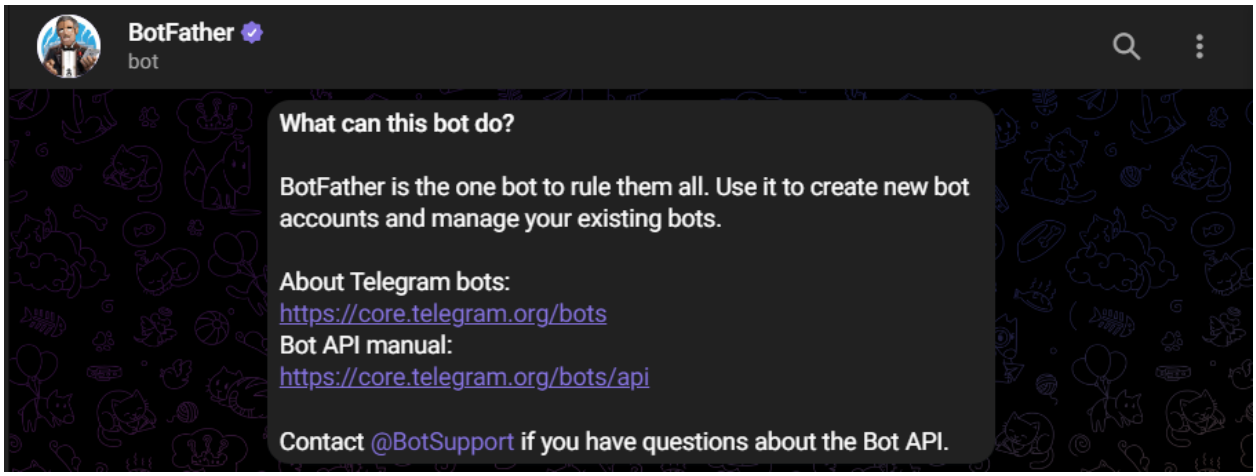

Рисунок 3.8. Вигляд боту @BotFather

Наступним кроком необхідно ввести команду “/start”. Після подання запиту, штучний інтелект відправляє меню доступних функцій, які можна виконати (Рис. 3.9).

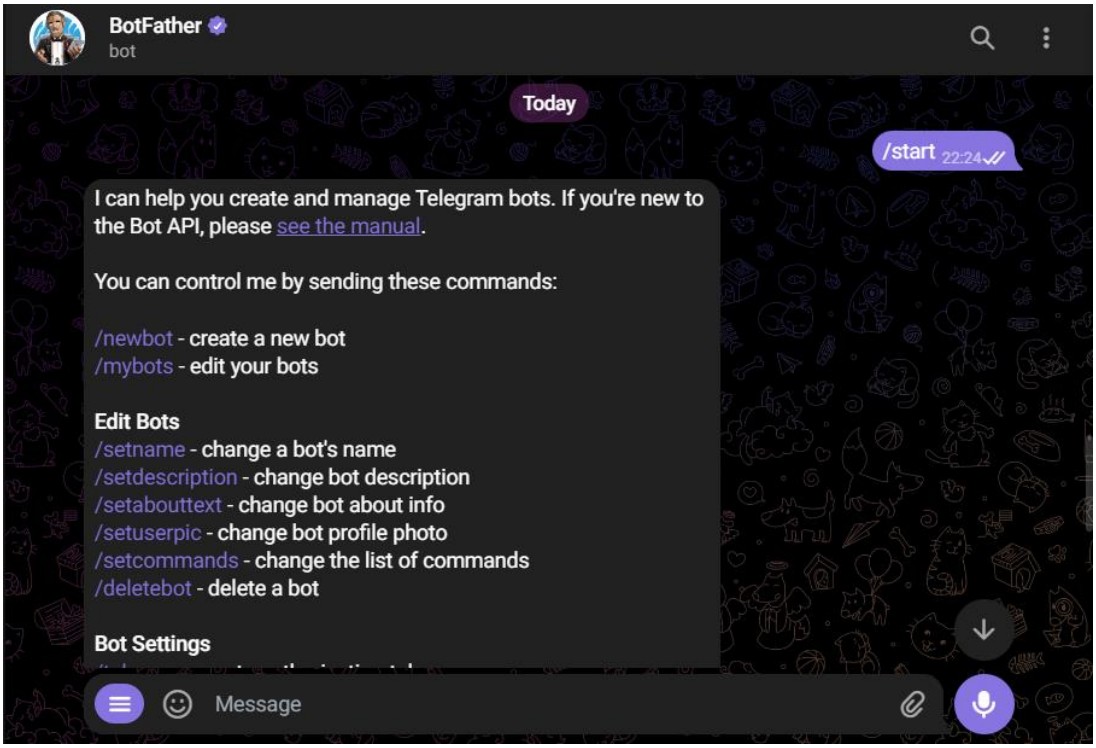

Рисунок 3.9. Функції для роботи з ботом

Для активації бота в додатку необхідно ввести команду "/newbot". На цьому етапі потрібно вказати назву бота. Далі BotFather запитає про ім'я користувача для бота, яке повинне бути унікальним, складатися з латинських літер, цифр та нижніх підкреслення та обов'язково закінчуватися на "bot". Після успішної реєстрації Bot Father повідомить про завершення процесу та надасть токен для доступу до HTTP API, а також посилання на бота в додатку Telegram (Рис. 3.10) [10].

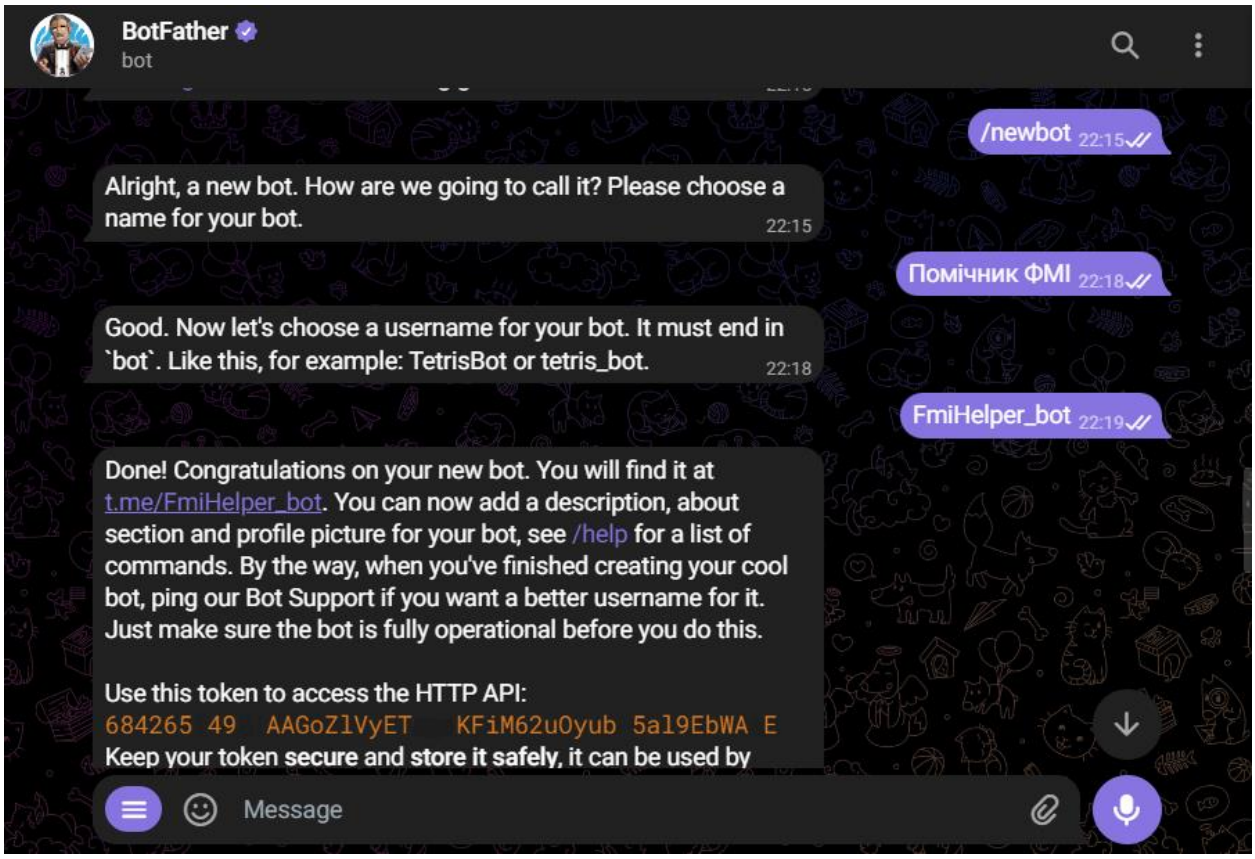

Рисунок 3.10. Отримання ключа доступу та посилання до створеного боту

Щоб запустити бота, необхідно в середовище розробки написати необхідну командую. Але перед тим, імпортуємо нашу бібліотеку TelegramBotAPI (Рис. 3.11).

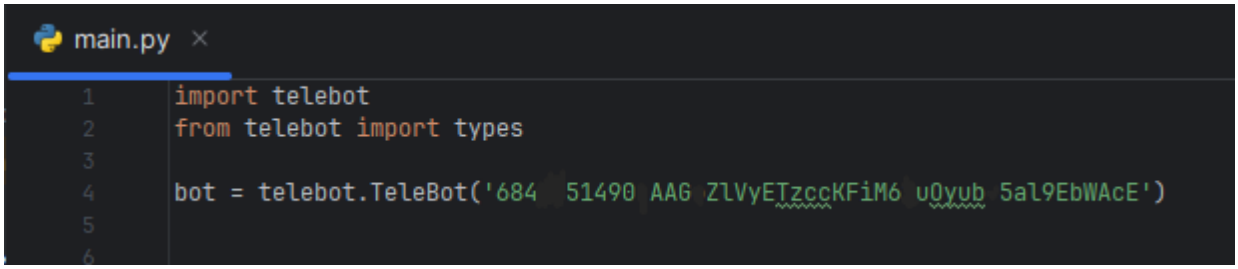

Рисунок 3.11. Імпортування бібліотеки та підключення чат-бота

РОЗДІЛ 4. РОЗРОБКА ЧАТ-БОТУ

4.1. Створення команд чат-боту

Команди у чат-ботах - це ключовий спосіб спілкування користувача з програмою. Вони визначають, які функції бот може виконати та яким чином відповісти на запитання. Ефективні команди дозволяють користувачам легко та швидко отримувати необхідну інформацію чи послуги.

Наступним кроком при створенні чат-боту були команди, а саме “/start”, “/help” та “/id”.

Команда “/start” використовується для початку роботи з ботом після першого запуску. Після введення цієї команди бот вмикається, та надає нам відповідь, яка була створена в середовищі розробки (Рис. 4.1).

```
@bot.message_handler(commands=['start'])
def main(message):
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    btn1 = types.KeyboardButton(text='Розклад 📅')

    markup.add(btn1)

    btn2 = types.KeyboardButton('Інформація ℹ️')

    markup.add(btn2)

    bot.send_message(message.chat.id, text: f'Привіт, <b>{message.from_user.first_name}</b>!🧡\nЯкщо ти навчаєшся в <b>РДГУ</b>, '
                                       f'а саме на факультеті <b>Математики та Інформатики</b>, '
                                       f'то я буду для тебе корисним 😉\n\n'
                                       f'Мої особливості:\n'
                                       f'📁 <b>Розклад</b>\n'
                                       f'📁 <b>Допомога вступникам</b>\n'
                                       f'📁 <b>Інформація для студентів</b>\n\n'
                                       f''
                                       f'📄Більше інформації - /info\n\n'
                                       f''
                                       f'Мерщій переходь до функціоналу ℹ️',
                     reply_markup=markup, parse_mode='html')
```

Рисунок 4.1. Створення команди start

Команда “/info” була створена для виведення додаткової інформації про чат-бот. Після введення цієї команди бот виводить додаткову інформацію, яка була створена в середовищі розробки (Рис. 4.2).

```
if message.text == '/info':
    bot.send_message(message.chat.id, text: 'ℹ️Більше інформації про бота:\n\n'
                                     f'📕 Дізнатись <b>ID</b> свого аккаунту - /id\n\n'
                                     f''
                                     f'🤖 Доречі, я трішки вмію спілкуватись, тому можеш привітатись, '
                                     f'запитати як в мене настрій, '
                                     f''
                                     f'та що я роблю, мені буде приємно 🙂\n\n'
                                     f'📹 А ще можу оцінити твою нову фотку чи відео,'
                                     f' яке ти плануєш запостити собі в <b>insta</b>, '
                                     f'сміло надсилай 🙂😁\n\n'
                                     f''
                                     f'👨‍💻Автор боту - студент групи ІП3-41 <b>Якимчук Ілля</b>, '
                                     f'надіюсь тобі сподобається😉'
                                     '', parse_mode='html')
```

Рисунок 4.2. Створення команди info

Команда “/id” була створена для виведення інформації про ID вашого акаунту в соц. мережі Telegram.

ID Telegram-акаунта - це унікальний ідентифікатор, який призначений для конкретного користувача чи чату у Telegram. Цей ID може бути використаний для відправки повідомлень, взаємодії з ботами чи для встановлення контакту з конкретним користувачем.

Кожен акаунт у Telegram має свій унікальний ID, який можна використовувати для різних цілей, таких як ідентифікація користувача в системі, створення чат-ботів чи надсилання повідомлень.

Після введення цієї команди бот виводить додаткову інформацію, в якій є ваш ID та заклик до продовження роботи з ботом по кнопкам знизу (Рис. 4.3).

```
if message.text == '/id':
    bot.send_message(message.chat.id,
                     text: f'ID твого аккаунту'  f' - <b>{message.from_user.id}</b>.\n'
                     f'Якщо бажаєте скоритатися іншими можливостями боту, '
                     f'використовуйте команди, або клавіші знизу 😁', parse_mode='html')
```

Рисунок 4.3. Створення команди id

Також для відображення команд в меню чат-боту (Рис. 4.4), необхідно перейти в BotFather, ввести команду /setcommands, обрати чат-бот та написати команди, які мають відображатись в меню (Рис. 4.5) [10].

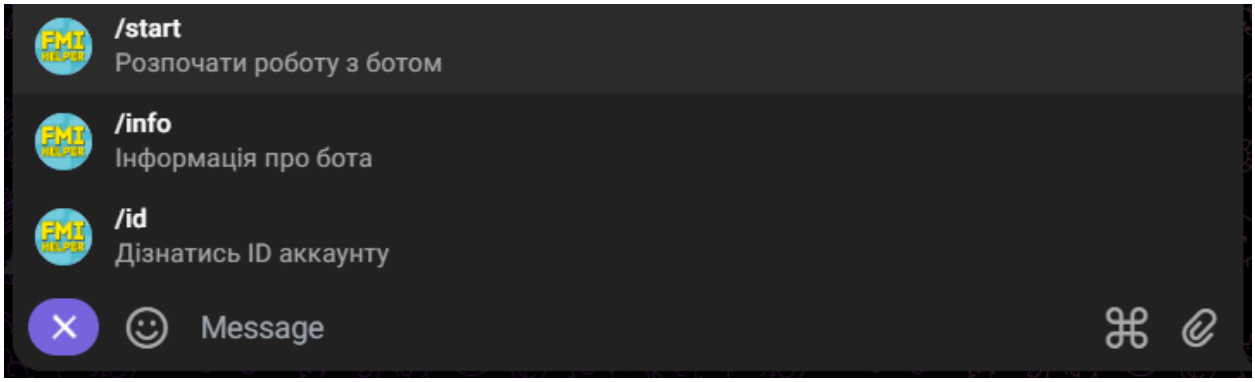

Рисунок 4.4. Демонстрація меню

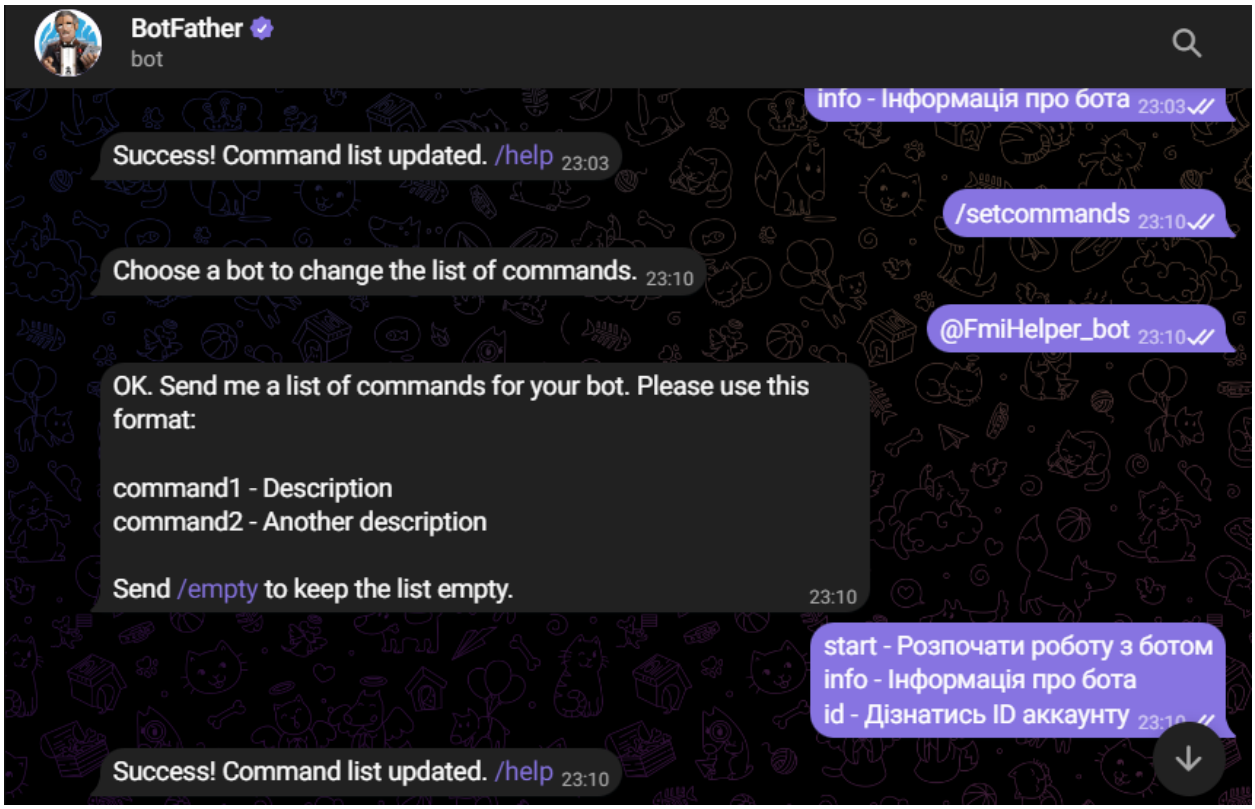

Рисунок 4.5. Додавання команд в меню

4.2. Створення клавіатури з розкладом

У чат-боті Telegram клавіатура - це інтерактивний елемент, який дозволяє користувачам взаємодіяти з ботом шляхом вибору кнопок заздалегідь визначених команд чи опцій. Клавіатура може бути представлена різними способами:

1. *Inline-клавіатура*: Вона відображається безпосередньо у чаті під повідомленням бота. Кнопки розташовані горизонтально та дозволяють користувачам вибирати опції шляхом натискання.

2. *Reply-клавіатура*: Ця клавіатура з'являється нижче повідомлення бота і містить кнопки вибору, на які можна натискати, щоб відправити конкретну команду чи відповідь.

Ці клавіатури можуть містити текстові команди, символи, іконки чи посилання. Вони допомагають спростити взаємодію з ботом, особливо коли користувачу потрібно вибрати з певного набору опцій чи виконати певні дії за допомогою кнопок, замість введення тексту вручну.

Створення клавіатур для чат-ботів в Telegram може виконуватися через спеціальні команди чи API, що забезпечують створення та відправку цих клавіатур користувачам.

Для початку була створена Reply клавіатура яка з’являється відразу після введення команди “/start” під полем для введення тексту. Перша клавіатура містить дві кнопки, а саме “Розклад” та “Інформація” (Рис. 4.6).

```
@bot.message_handler(commands=['start'])
def main(message):
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    btn1 = types.KeyboardButton(text='Розклад ')

    markup.add(btn1)

    btn2 = types.KeyboardButton('Інформація ')

    markup.add(btn2)
```

Рисунок 4.6. Створення Reply клавіатури після команди start

Наступник кроком була створена Reply клавіатура, яка відображає вибір курсу студента, наприклад “І курс”, “ІІ курс” та кнопку назад, яка повертає вас до головного меню (Рис. 4.7).

```python
@bot.message_handler(content_types=['text'])
def bot_message(message):
    if message.text == 'Розклад 🧾':
        markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
        item1 = types.KeyboardButton('I курс')
        item2 = types.KeyboardButton('II курс')
        item3 = types.KeyboardButton('III курс')
        markup.add( *args: item1, item2, item3)
        item4 = types.KeyboardButton('IV курс')
        item5 = types.KeyboardButton('V курс')
        item6 = types.KeyboardButton('VI курс')
        markup.add( *args: item4, item5, item6)
        item7 = types.KeyboardButton('Назад ↩️')
        markup.add(item7)
        bot.send_message(message.chat.id,  text: f'<b>{message.from_user.first_name}</b> , будь ласка, обери свій курс.',
                         reply_markup=markup, parse_mode='html')
```

Рисунок 4.7. Створення Reply клавіатури “Розклад”

Після створення Reply клавіатури “Розклад”, були створені Inline клавіатури, які відображаються безпосередньо під повідомлення в чаті чат-боту для кожного курсу. Наприклад, при натисканні кнопки “I курс” в чат виводиться текст “Будь ласка, обери свою групу”, а під нею Inline клавіатура, яка відображає всі групи першого курсу. Таким чином були створені Inline клавіатури для кожного курсу (Рис. 4.8, 4.9).

```python
elif message.text == 'I курс':
    markup = types.InlineKeyboardMarkup()
    btn_first_kurs_btn1 = types.InlineKeyboardButton( text: 'М-11', callback_data='М-11')
    btn_first_kurs_btn2 = types.InlineKeyboardButton( text: 'ІП3-11', callback_data='ІП3-11')
    btn_first_kurs_btn3 = types.InlineKeyboardButton( text: 'КН-11', callback_data='КН-11')
    markup.row( *args: btn_first_kurs_btn1, btn_first_kurs_btn2, btn_first_kurs_btn3)
    btn_first_kurs_btn4 = types.InlineKeyboardButton( text: 'І-11', callback_data='І-11')
    btn_first_kurs_btn5 = types.InlineKeyboardButton( text: 'ЦТ-11', callback_data='ЦТ-11')
    markup.row( *args: btn_first_kurs_btn4, btn_first_kurs_btn5)
    btn_first_kurs_btn6 = types.InlineKeyboardButton( text: 'Переглянути розклад на сайті 🌎',
                                                      url='https://docs.google.com/spreadsheets/d/1Cm7-Ky66wnGDprfmj
    markup.row(btn_first_kurs_btn6)
    bot.reply_to(message,  text: 'Будь ласка, обери свою групу 😀', reply_markup=markup)
```

Рисунок 4.8. Створення Inline клавіатури для першого курсу

```python
elif message.text == 'III курс':
    markup = types.InlineKeyboardMarkup()
    btn_third_kurs_btn1 = types.InlineKeyboardButton( text: 'М-31', callback_data='М-31')
    btn_third_kurs_btn2 = types.InlineKeyboardButton( text: 'ІП3-31', callback_data='ІП3-31')
    btn_third_kurs_btn3 = types.InlineKeyboardButton( text: 'КН-31', callback_data='КН-31')
    markup.row( *args: btn_third_kurs_btn1, btn_third_kurs_btn2, btn_third_kurs_btn3)
    btn_third_kurs_btn4 = types.InlineKeyboardButton( text: 'ПМ-31', callback_data='ПМ-31')
    btn_third_kurs_btn5 = types.InlineKeyboardButton( text: 'І-31', callback_data='І-31')
    btn_third_kurs_btn6 = types.InlineKeyboardButton( text: 'ЦТ-31', callback_data='ЦТ-31')
    markup.row( *args: btn_third_kurs_btn4, btn_third_kurs_btn5, btn_third_kurs_btn6)
    btn_third_kurs_btn7 = types.InlineKeyboardButton( text: 'Переглянути розклад на сайті 🌎',
                                                      url='https://docs.google.com/spreadsheets/d/140HAF_vzdgrY2HQq5D
    markup.row(btn_third_kurs_btn7)
    bot.reply_to(message,  text: 'Будь ласка, обери свою групу 😀', reply_markup=markup)
```

Рисунок 4.9. Створення Inline клавіатури для третього курсу

Умова “message.text ==” перевіряється, і якщо вона виконується (тобто, текст повідомлення співпадає з, наприклад, 'І курс'), виконуються певні дії чи відправляється відповідь, які були програмно визначені для цього випадку [3].

Наступник кроком було відображення днів тижня, після обраної групи користувачем чат-боту. Для цього використовувався параметр “callback_data”, який зчитував інформацію після натискання кнопки, так як Inline клавіатура не виводить інформацію після натискання кнопки в чат, на відміну від Reply клавіатури. Цей параметр зчитував дані, та оновляв Inline клавіатуру відштовхуючись від зчитаної інформації. Клавіатура оновлювалась з відображення груп на дні тижня (Рис. 4.10, 4.11).

Також була створена кнопка “Назад”, яка повертала попереднє меню (Рис. 4.12).

```
if callback.data == "М-11":
    markup = types.InlineKeyboardMarkup()
    m1_1 = types.InlineKeyboardButton(text="Понеділок", callback_data="m1_1")
    t1_1 = types.InlineKeyboardButton(text="Вівторок", callback_data="t1_1")
    w1_1 = types.InlineKeyboardButton(text="Середа", callback_data="w1_1")
    markup.add( *args: m1_1, t1_1, w1_1)
    th1_1 = types.InlineKeyboardButton(text="Четвер", callback_data="th1_1")
    f1_1 = types.InlineKeyboardButton(text="П'ятниця", callback_data="f1_1")
    markup.add( *args: th1_1, f1_1)
    main_menu = types.InlineKeyboardButton(text="Назад ↩", callback_data="mainmenu1")
    markup.add(main_menu)
    bot.edit_message_text(chat_id=callback.message.chat.id, message_id=callback.message.message_id, text="Обери день тижня ⬇",
                          reply_markup=markup)
```

Рисунок 4.10. Створення Inline клавіатури з відображенням днів тижня для групи М-11

```
if callback.data == "ЦТ-11":
    markup = types.InlineKeyboardMarkup()
    m1_5 = types.InlineKeyboardButton(text="Понеділок", callback_data="m1_5")
    t1_5 = types.InlineKeyboardButton(text="Вівторок", callback_data="t1_5")
    w1_5 = types.InlineKeyboardButton(text="Середа", callback_data="w1_5")
    markup.add( *args: m1_5, t1_5, w1_5)
    th1_5 = types.InlineKeyboardButton(text="Четвер", callback_data="th1_5")
    f1_5 = types.InlineKeyboardButton(text="П'ятниця", callback_data="f1_5")
    markup.add( *args: th1_5, f1_5)
    main_menu = types.InlineKeyboardButton(text="Назад ↩", callback_data="mainmenu1")
    markup.add(main_menu)
    bot.edit_message_text(chat_id=callback.message.chat.id, message_id=callback.message.message_id, text="Обери день тижня ⬇",
                          reply_markup=markup)
```

Рисунок 4.11. Створення Inline клавіатури з відображенням днів тижня для групи ЦТ-11

```python
@bot.callback_query_handler(func=lambda callback: True)
def callback_message(callback):

    if callback.data == "mainmenu1":
        markup = types.InlineKeyboardMarkup()
        btn_first_kurs_btn1 = types.InlineKeyboardButton( text: 'М-11', callback_data='М-11')
        btn_first_kurs_btn2 = types.InlineKeyboardButton( text: 'ІПЗ-11', callback_data='ІПЗ-11')
        btn_first_kurs_btn3 = types.InlineKeyboardButton( text: 'КН-11', callback_data='КН-11')
        markup.row( *args: btn_first_kurs_btn1, btn_first_kurs_btn2, btn_first_kurs_btn3)
        btn_first_kurs_btn4 = types.InlineKeyboardButton( text: 'І-11', callback_data='І-11')
        btn_first_kurs_btn5 = types.InlineKeyboardButton( text: 'ЦТ-11', callback_data='ЦТ-11')
        markup.row( *args: btn_first_kurs_btn4, btn_first_kurs_btn5)
        btn_first_kurs_btn6 = types.InlineKeyboardButton( text: 'Переглянути розклад на сайті 🌐',
                                                         url='https://www.rshu.edu.ua/vstupnyku/vstupna-kampaniia')
        markup.row(btn_first_kurs_btn6)
        bot.edit_message_text(chat_id=callback.message.chat.id, message_id=callback.message.message_id,
                              text="Будь ласка, обери свою групу 🙂", reply_markup=markup)
```

Рисунок 4.12. Створення Inline клавіатури з поверненням відображення вибору груп першого курсу

Після чого був процес створення самого розкладу. Для цього як і в відображенні днів тижня використовувався параметр “callback_data”. Після зчитування даних зворотного виклику (callback_data) Inline клавіатура оновлювалась і відображала розклад, відштовхуючись від натиснутої кнопки, наприклад “m1_1” це перший курс, понеділок (m – Monday) , перша група в таблиці розкладу від деканату (М-11), “w3_5” це третій курс, середа (w - Wednesday), п’ята група в таблиці розкладу від деканату (І-31). Також була створена кнопка назад, яка оновлювала Inline клавіатуру до попереднього кроку, а саме вибору днів тижня. (Рис. 4.13, 4.14).

```python
if callback.data == "t1_1":
    markup = types.InlineKeyboardMarkup()
    main_menu = types.InlineKeyboardButton(text="Назад ↩️", callback_data="М-11")
    markup.add(main_menu)
    bot.edit_message_text(chat_id=callback.message.chat.id, message_id=callback.message.message_id,
                          text="📃 <b>Вівторок</b>\n\n"
                               "| 8:00 | - | 9:20 |\n"
                               "• Аналогічна геометрія. доц.\n Присяжнюк І. М.  ауд. 405\n\n"

                               "| 9:35 | - | 10:55 |\n"
                               "• Аналогічна геометрія. доц.\n Присяжнюк І. М.  ауд. 405\n\n"

                               "| 11:10 | - | 12:30 |\n"
                               "• Немає\n\n"

                               "| 12:45 | - | 14:05 |\n"
                               "• Немає",
                          reply_markup=markup, parse_mode='html')
```

Рисунок 4.13. Створення Inline клавіатури з відображенням розкладу для групи М-11 (вівторок), та кнопки “Назад”

```
if callback.data == "w2_1":
    markup = types.InlineKeyboardMarkup()
    main_menu = types.InlineKeyboardButton(text="Назад ", callback_data="M-21")
    markup.add(main_menu)
    bot.edit_message_text(chat_id=callback.message.chat.id, message_id=callback.message.message_id,
                          text=" <b>Середа</b>\n\n"
                               "| 8:00 | - | 9:20 |\n"
                               "• Проективна геометрія і методи зображень\n ст. в. Тимчук М.В.\n\n"

                               "| 9:35 | - | 10:55 |\n"
                               "• Немає\n\n"

                               "| 11:10 | - | 12:30 |\n"
                               "• Немає\n\n"

                               "| 12:45 | - | 14:05 |\n"
                               "• Немає\n\n"

                               "| 14:20 | - | 15:40 |\n"
                               "• Немає",
                          reply_markup=markup, parse_mode='html')
```

Рисунок 4.14. Створення Inline клавіатури з відображенням розкладу для групи М-21 (середа), та кнопки “Назад”

4.3. Створення додаткових кнопок – Інформація, Сайт, Контакти і тд.

В чат-боті була вбудована додаткова інформація (Рис. 4.15), а саме сайт університету, де окрім інформації була створення кнопка-посилання, після натискання якої відбувається перехід на сайт університету (Рис. 4.16), контакти університету, такі як номера телефонів та електрона адреса (Рис. 4.17), інформація для вступників та кнопка-посилання, яка містить в собі посилання на вступну кампанію (Рис. 4.18), соціальні мережі факультету, де є декілька кнопок-посилань, за допомогою яких можна перейти на соц.мережі факультету ФМІ (Рис. 4.19).

Це було реалізовано через оновлення Reply клавіатури. Бот зчитував дані введені в чат користувачем, та оновлював клавіатуру в залежності від зчитаних даних.

До цієї клавіатури була додана також кнопка “Назад”, яка повертала користувача в головне меню (Рис. 4.20).

```
if message.text == 'Інформація ℹ️':
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    item1 = types.KeyboardButton('Для вступників 👨‍🎓')
    markup.add(item1)
    item2 = types.KeyboardButton('Соціальні мережі 📱')
    item3 = types.KeyboardButton('Контакти ☎️')
    item4 = types.KeyboardButton('Сайт 🌐')
    markup.add( *args: item2, item3, item4)
    item5 = types.KeyboardButton('Назад ↩️')
    markup.add(item5)
    bot.send_message(message.chat.id, text: f'Меню "<b>Інформація</b>" відкрито. Скористайтесь клавішами знизу ℹ️',
                     reply_markup=markup, parse_mode='html')
```

Рисунок 4.15. Reply клавіатура “Інформація”

```
if message.text == 'Сайт 🌐':
    markup = types.InlineKeyboardMarkup()

    soc3 = types.InlineKeyboardButton( text: 'Сайт 🌐', url='https://www.rshu.edu.ua')
    markup.row(soc3)
    bot.reply_to(message, text: '💬 Бажаєш переглянути сайт нашого університету?\n\n<b>Натискай на кнопку</b> ℹ️',reply_markup=markup,
```

Рисунок 4.16. Інформація про сайт університету, та кнопка-посилання

```
if message.text == 'Контакти ☎️':
    markup = types.InlineKeyboardMarkup()

    bot.reply_to(message, text: '<b>📱 Контакти</b> \n\n'
                               '<b>Номера телефонів ☎️</b>\n'
                               '▫️+0362620356\n'
                               '▫️+0362634224\n\n'
                               '<b>E-mail </b>✉️\n'
                               '▫️rectorat@rshu.edu.ua', reply_markup=markup, parse_mode='html')
```

Рисунок 4.17. Контакти університету

```
if message.text == 'Для вступників 👨‍🎓':
    markup = types.InlineKeyboardMarkup()

    vstup = types.InlineKeyboardButton( text: 'Вступна кампанія 📱', url='https://www.rshu.edu.ua/vstupnyku/vstupna-kampaniia')
    markup.row(vstup)
    bot.reply_to(message, text: '👨‍🎓 Бажаєш переглянути інформацію для вступників?\n\n'
                               '<b>Натискай на кнопку</b> ℹ️', reply_markup=markup, parse_mode='html')
```

Рисунок 4.18. Інформація для вступників

```
if message.text == 'Соціальні мережі 📱':
    markup = types.InlineKeyboardMarkup()
    soc1 = types.InlineKeyboardButton( text: 'Telegram 📩', url='https://t.me/FmiHelper_bot')
    markup.row(soc1)
    soc2 = types.InlineKeyboardButton( text: 'Instagram 📸 ', url='https://instagram.com/fmi_rshu')
    markup.row(soc2)
    soc3 = types.InlineKeyboardButton( text: 'Tik Tok 📱', url='https://www.tiktok.com/@fmi_rshu')
    markup.row(soc3)
    bot.reply_to(message, text: '💬 Соціальні мережі ФМІ:', reply_markup=markup)
```

Рисунок 4.19. Соціальні мережі факультету

```
if message.text == 'Назад ':
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    btn1 = types.KeyboardButton(text='Розклад ')
    markup.add(btn1)
    btn2 = types.KeyboardButton('Інформація ')
    markup.add(btn2)
    bot.send_message(message.chat.id, text: f'Ви в головному меню ',
                     reply_markup=markup, parse_mode='html')
```

Рисунок 4.20. Реалізація повернення в головне меню

4.4. Створення логіки зчитування даних з гугл диску

Для більш зручного використання чат боту було реалізовано автоматичне зчитування даних з гугл таблиць, та виведення їх в чат-бот.

Google Sheets — це застосунок для роботи з електронними таблицями, що входить до безкоштовного веб-пакету Google Диск. Сервіс доступний в режимі онлайн, а також має мобільні версії для Android, iOS, Windows, BlackBerry та настільну версію для Google ChromeOS. Інтерфейс Google Sheets подібний до Microsoft Excel, що є частиною Microsoft Office, і підтримує формати файлів Excel. Користувачі можуть редагувати і форматувати таблиці в режимі реального часу та надавати доступ іншим користувачам для спільної роботи. [11].

Для роботи чат - бота з таблицею, необхідно з’єднати його через сервіс Google Cloud.

Google Cloud – це хмарна платформа, яка надає широкий спектр послуг і рішень для зберігання, обробки, аналізу та розгортання програм. Він розроблений, щоб допомогти компаніям і розробникам створювати, запускати та масштабувати свої програми, веб-сайти та інші служби, використовуючи потужність хмарних обчислень.

Першим кроком було створення аккаунту в сервісі Google Cloud, та проекту під назвою chatbot (Рис. 4.21, 4.22, 4.23).

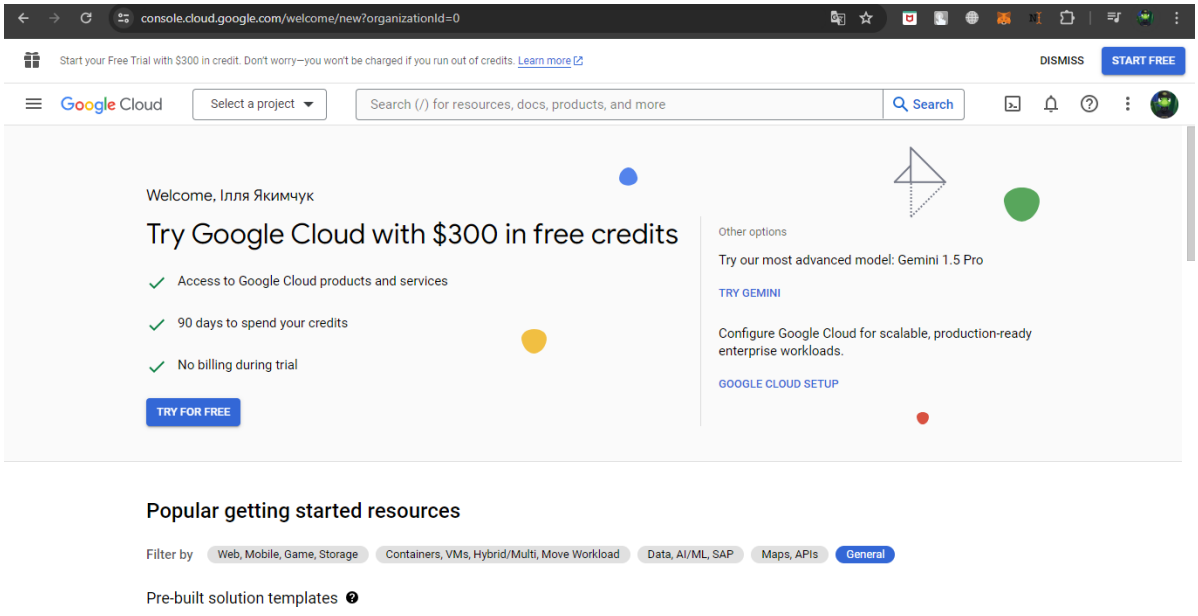

Рисунок 4.21. Створення аккаунту в сервісі Google Cloud

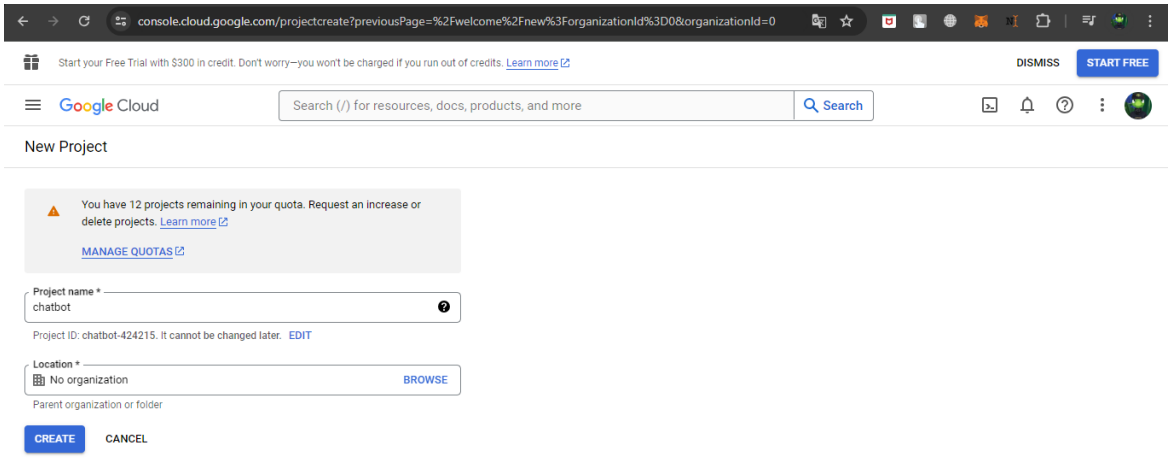

Рисунок 4.22. Процес створення проекту

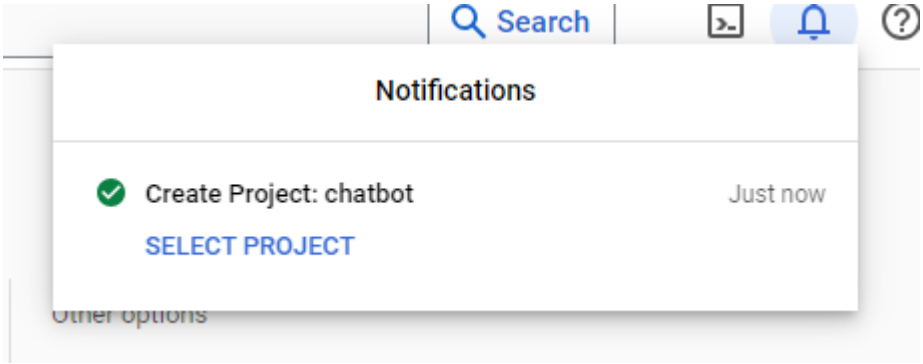

Рисунок 4.23. Успішне створення проекту

Наступним кроком було завантаження Google Drive API та Google Sheets API (Рис. 4.24, 4.25, 4.26, 4.27, 4.28, 4.29). Якщо говорити простими слова, то API це

посередник між програмами. В назві API міститься слово “Інтерфейс”. Розробники його використовують, надсилаючи запити та отримують відповідь.

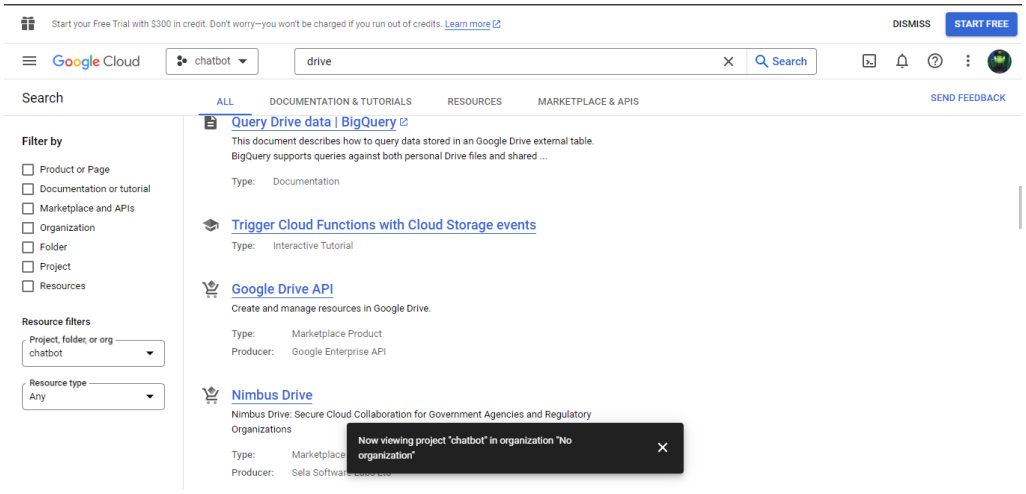

Рисунок 4.24. Пошук Google Drive API

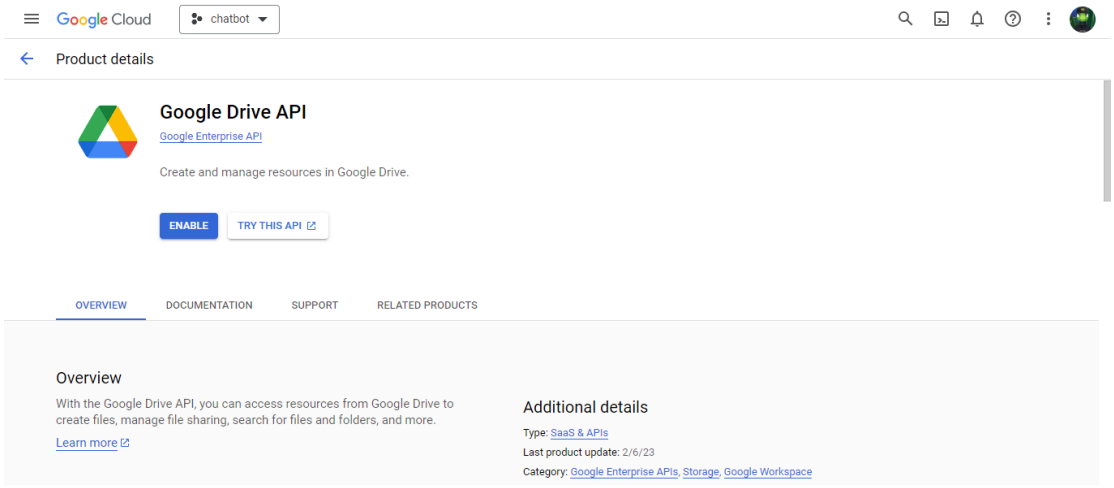

Рисунок 4.25. Процес встановлення Google Drive API

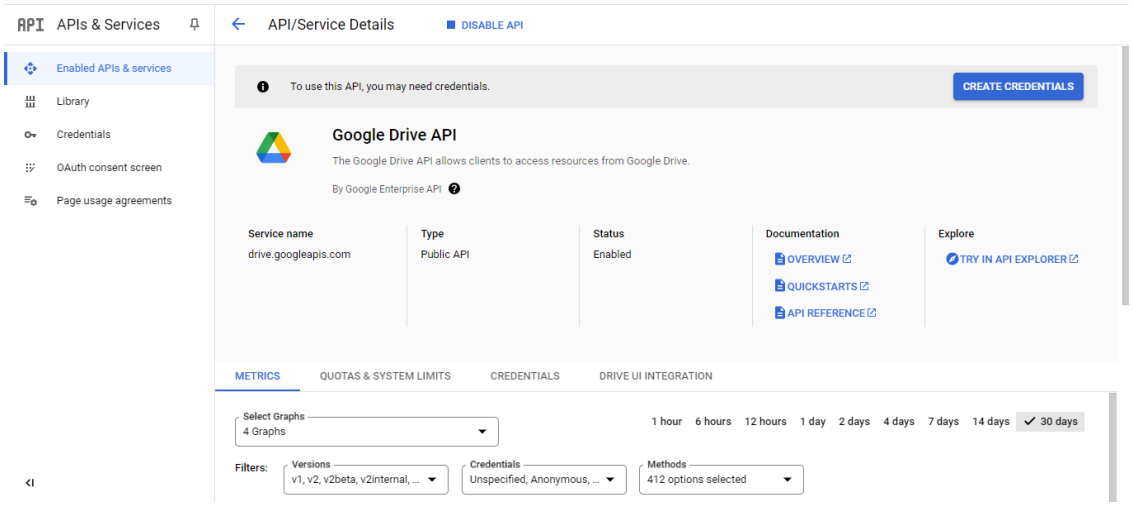

Рисунок 4.26. Успішне встановлення Google Drive API

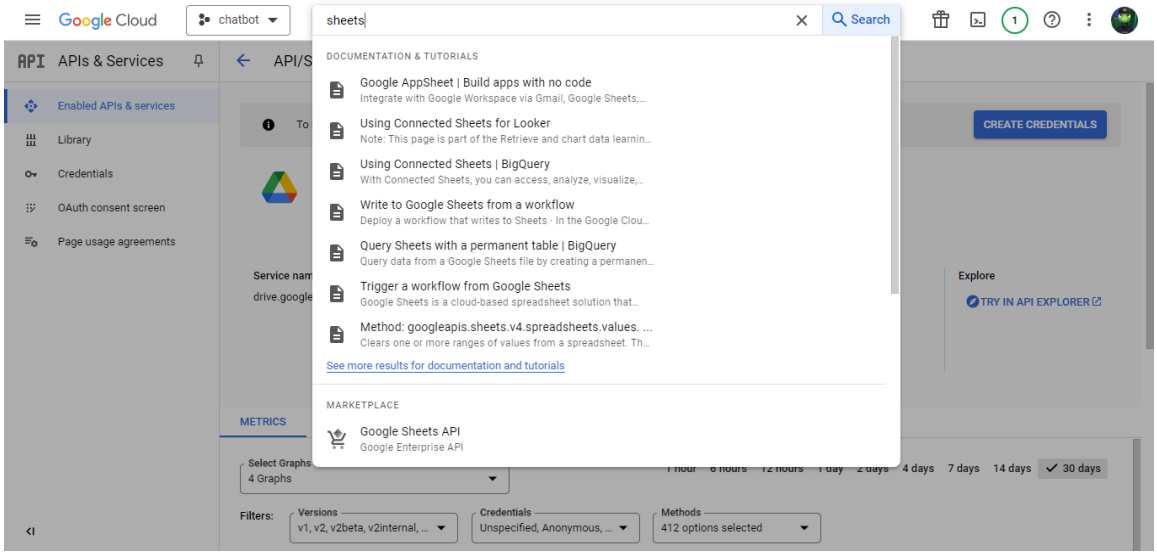

Рисунок 4.27. Пошук Google Sheets API

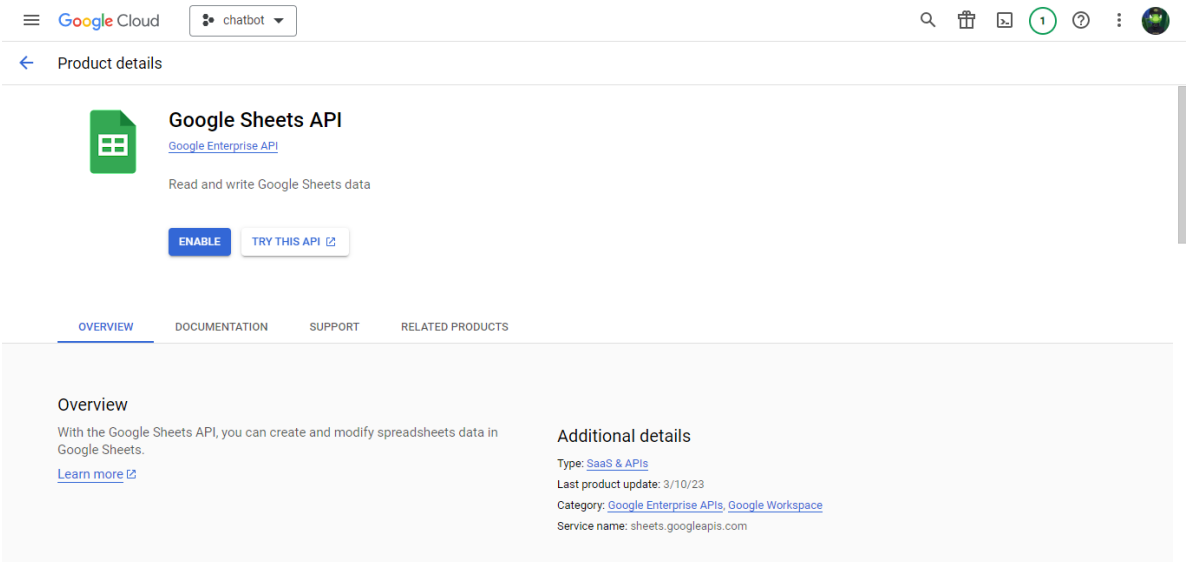

Рисунок 4.28. Процес встановлення Google Sheets API

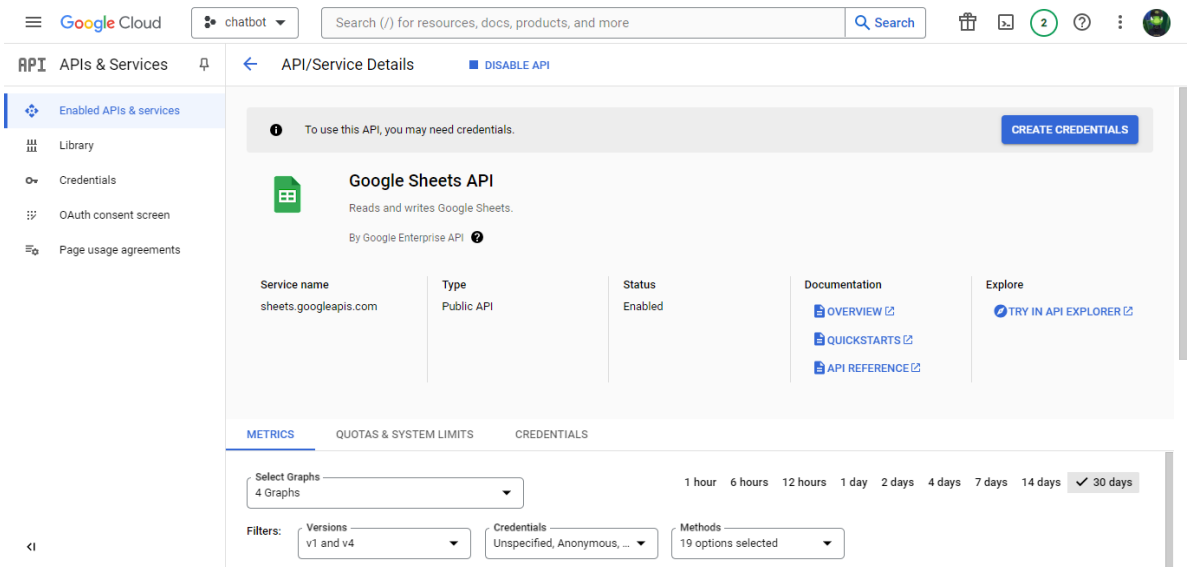

Рисунок 4.29. Успішне встановлення Google Sheets API

Далі процес створення сервіс акаунту. Він грає важливу роль в роботі чат-бота з гугл таблицею, так як після створення сервіс акаунту було отримано сервіс пошту, яка потрібна для подальшого з’єднання чат-боту з Google Sheets (Рис. 4.30, 4.31, 4.32, 4.33).

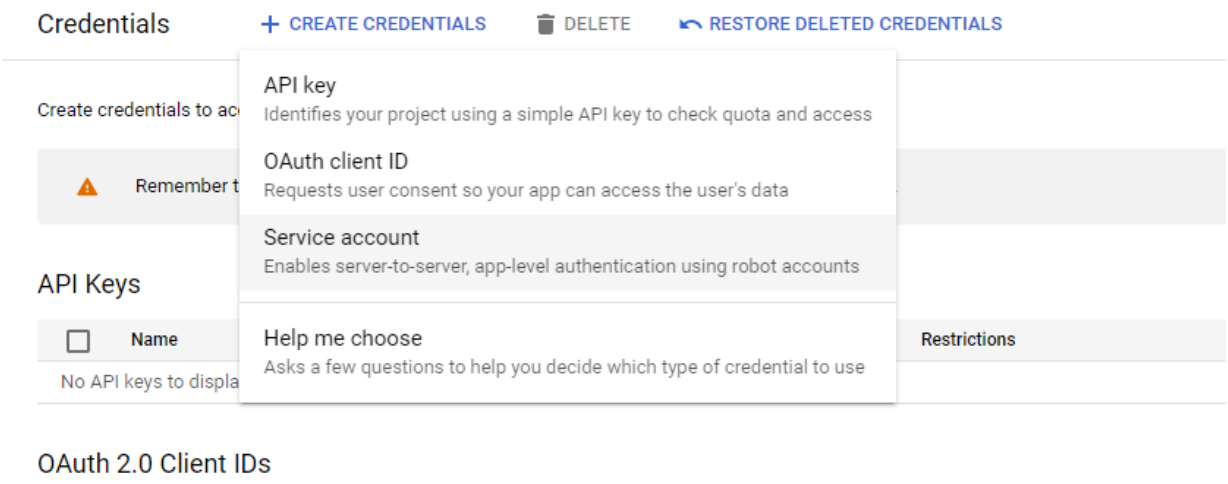

Рисунок 4.30. Початок створення сервіс акаунту

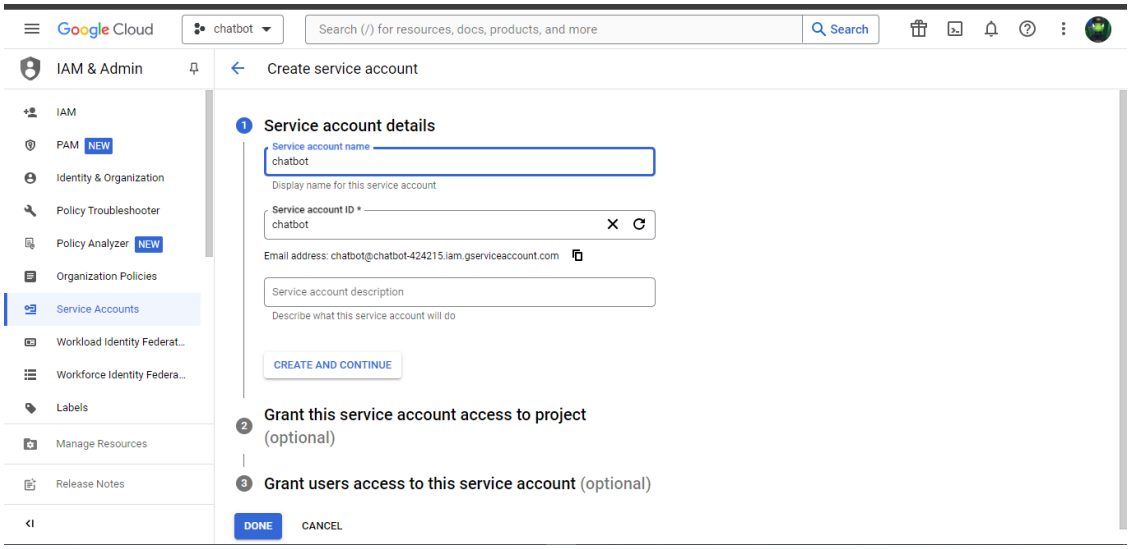

Рисунок 4.31. Процес налаштування сервіс акаунту

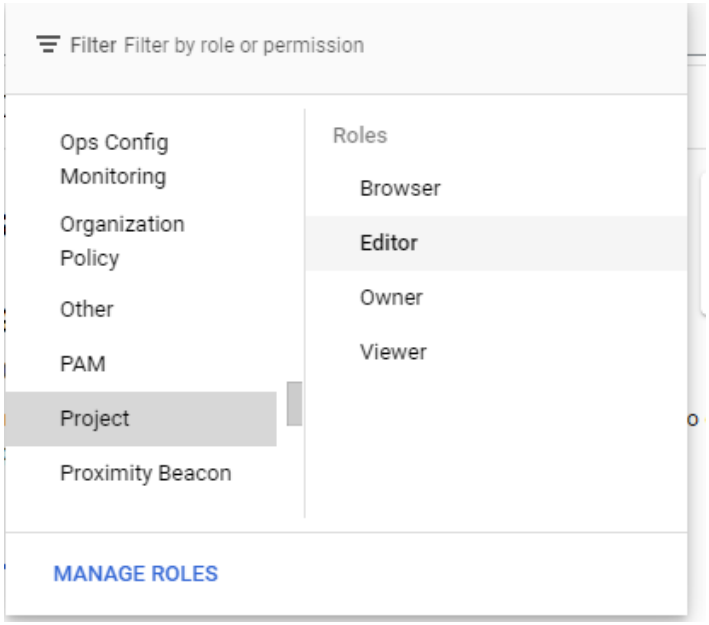

Рисунок 4.32. Встановлення ролі цього акаунту в проекті

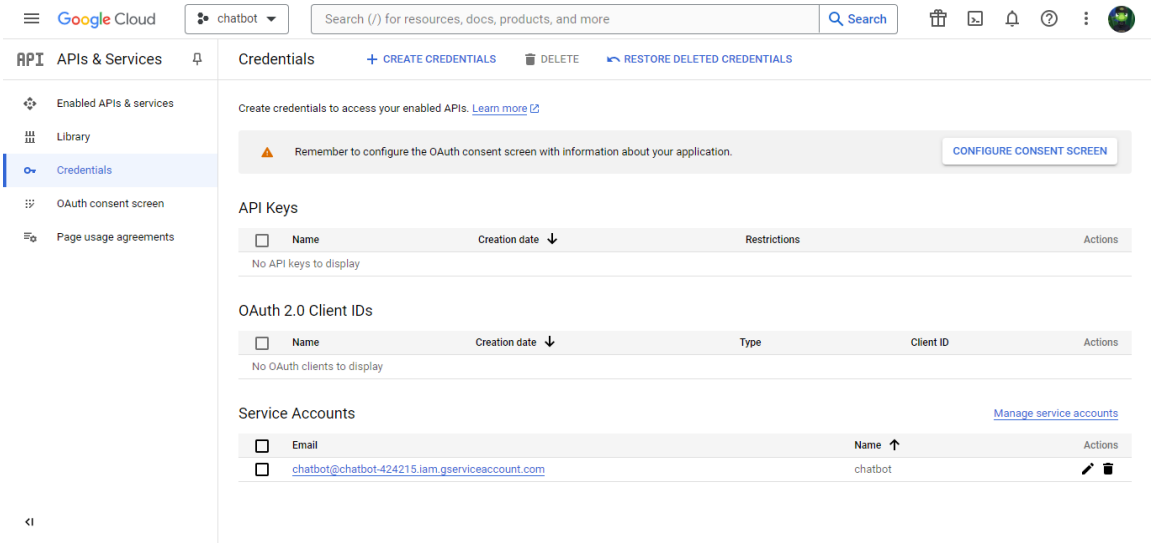

Рисунок 4.33. Успішне створення сервіс акаунту

Наступним кроком було створення consent screen (Рис. 4.34 – 4.41). Consent screen (екран згоди) в Google Cloud необхідний для інформування користувачів про те, які дані запитує ваша програма та як ці дані використовуватимуться. Цей екран є обов’язковим компонентом процесу автентифікації та авторизації користувача через OAuth 2.0 і Google API. Також він необхідний для створення тестового клієнта.

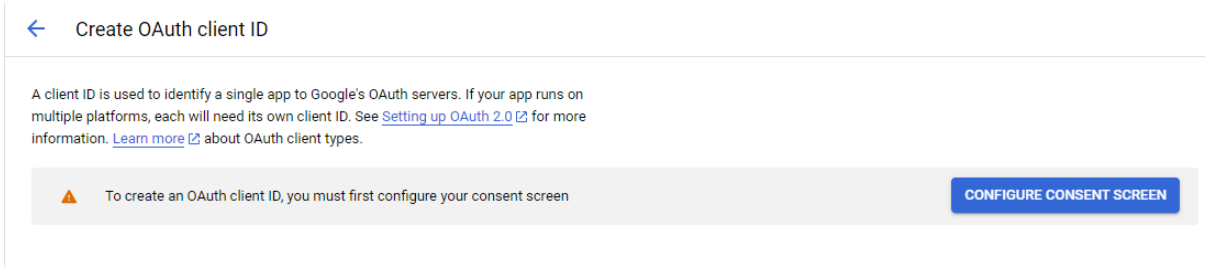

Рисунок 4.34. Вимога створити consent screen під час створення клієнта

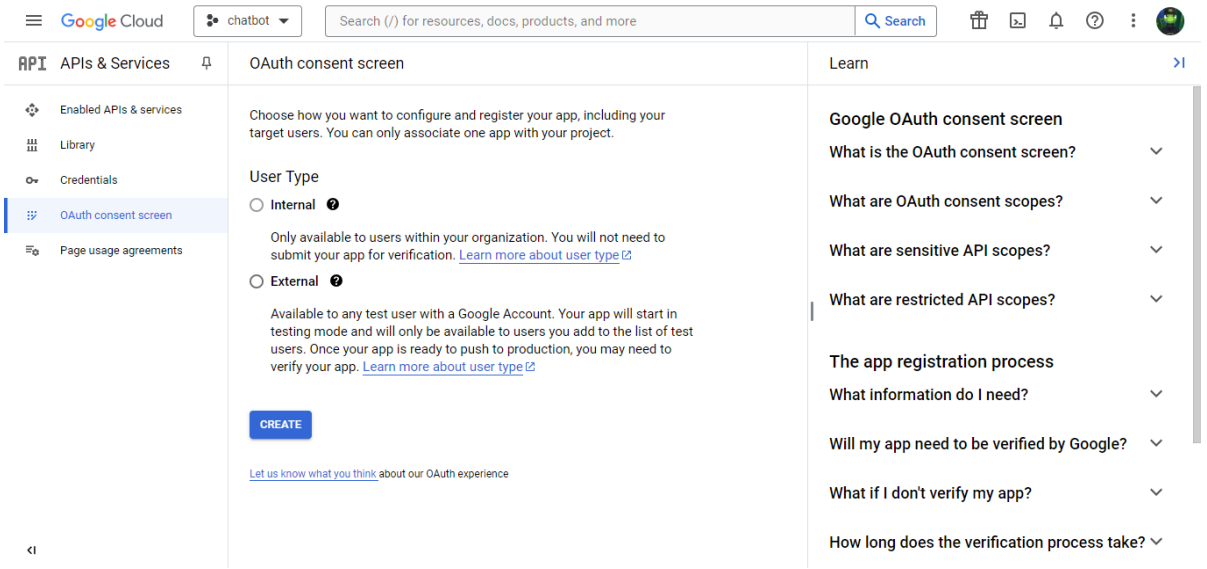

Рисунок 4.35. Початок створення екрану згоди

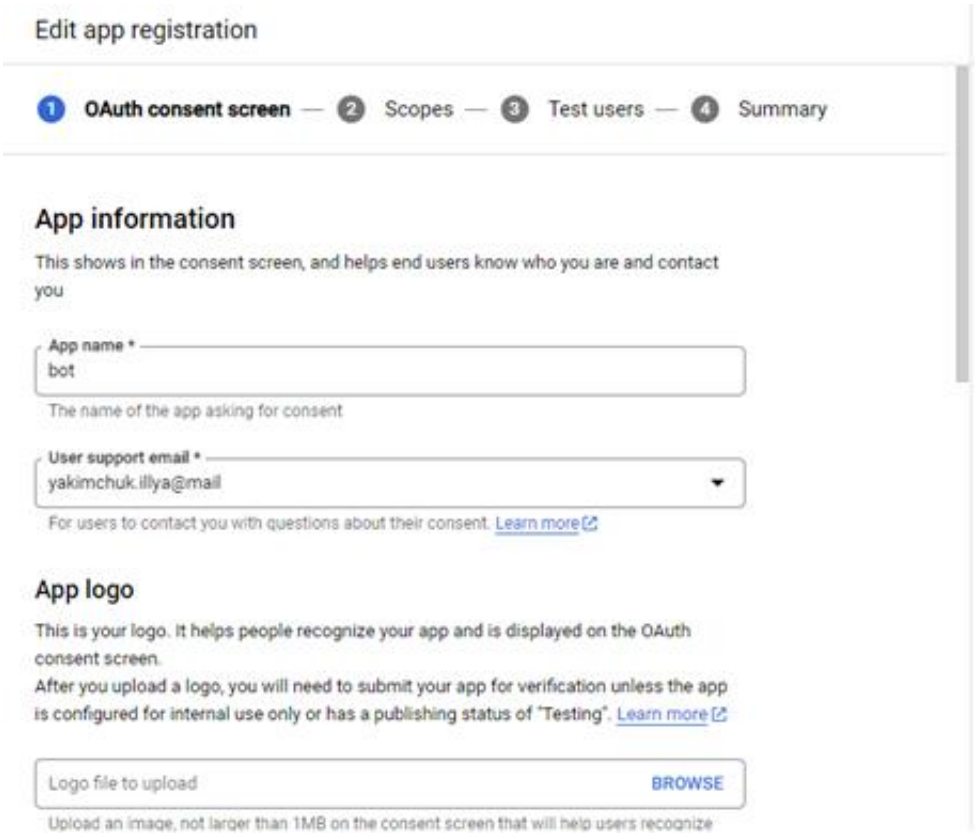

Рисунок 4.36. Налаштування consent screen

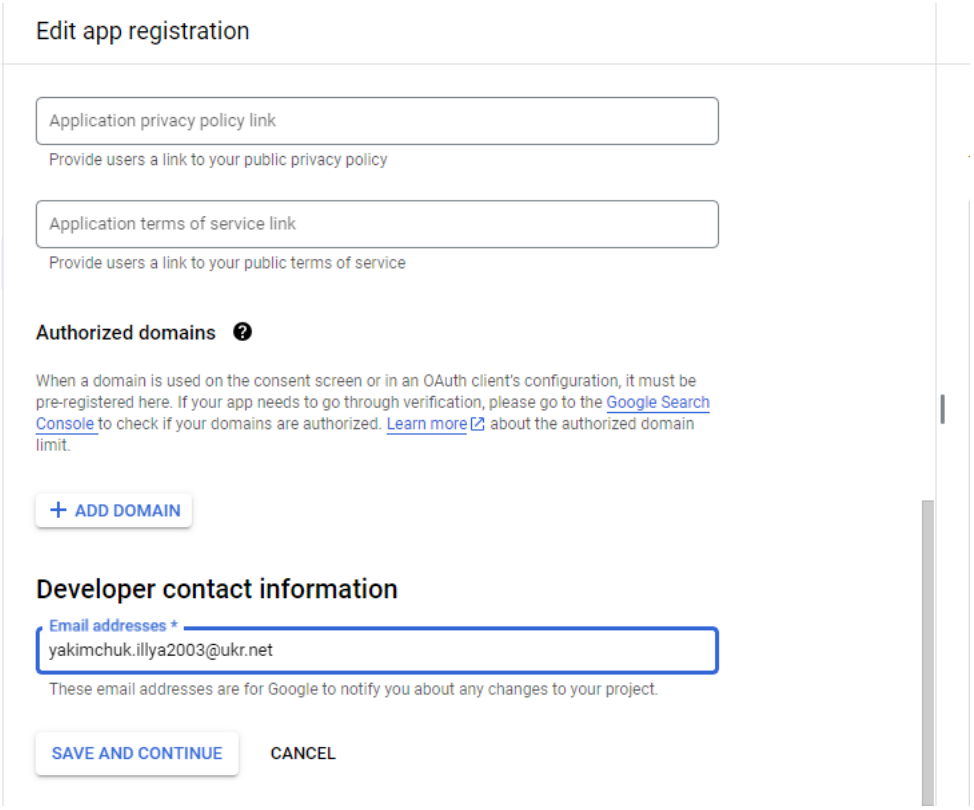

Рисунок 4.37. Налаштування consent screen. Внесення електроної пошти розробника

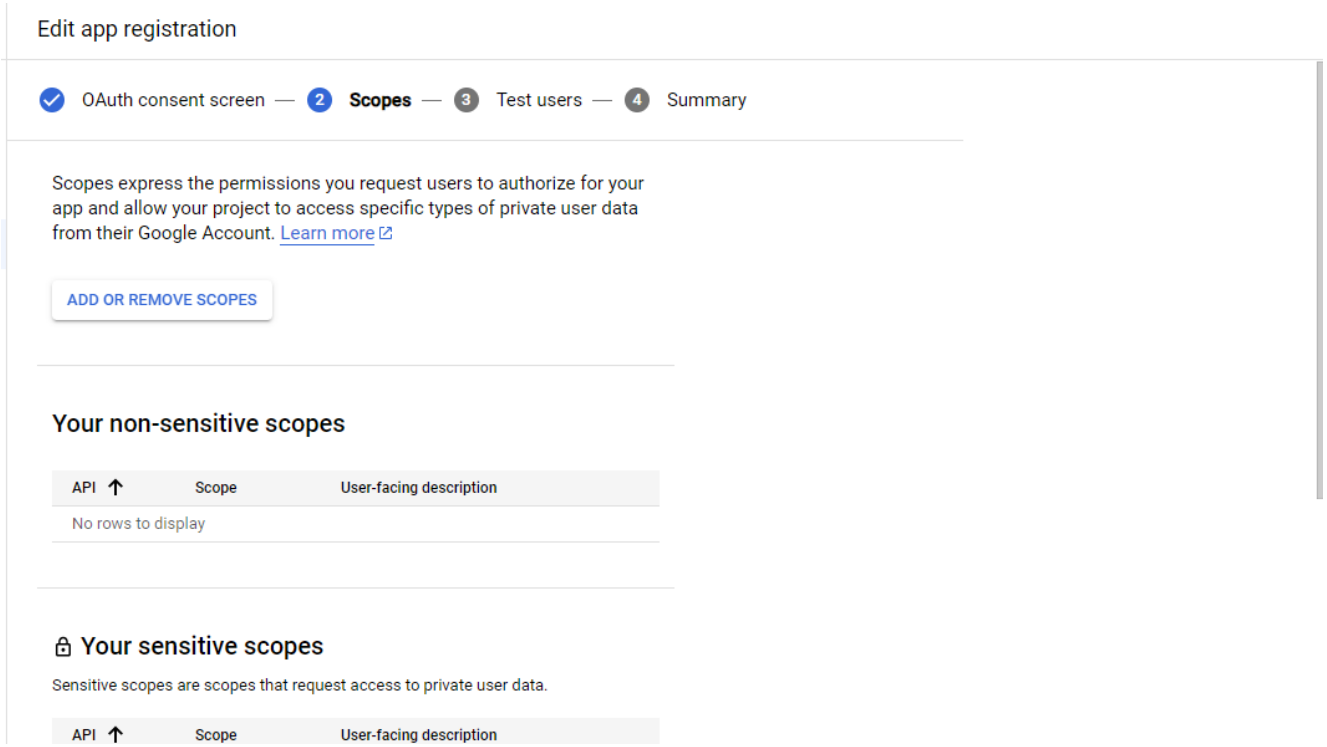

Рисунок 4.38. Налаштування consent screen. Надання прав

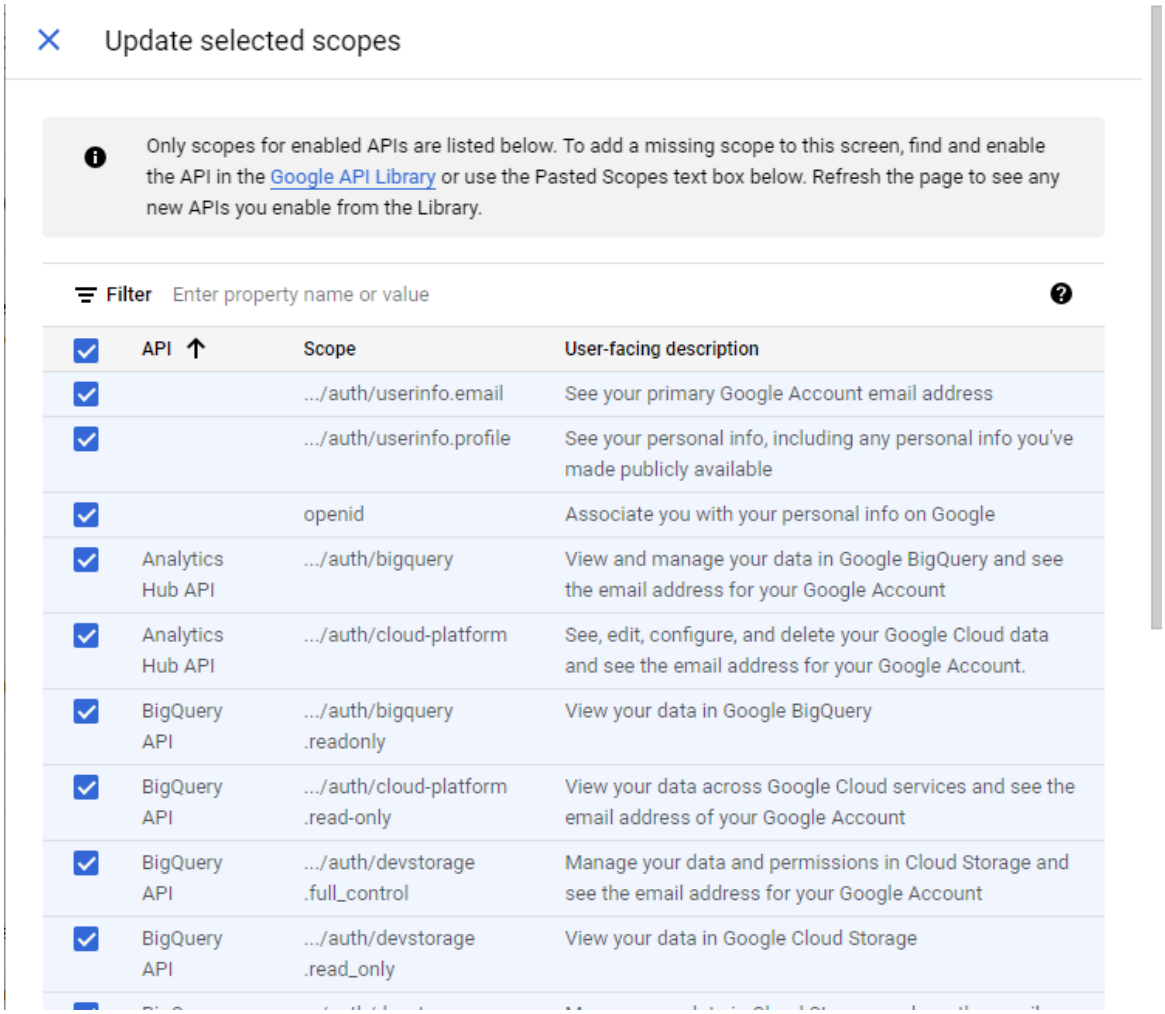

Рисунок 4.39. Налаштування consent screen. Ввімкнення всіх прав проекту

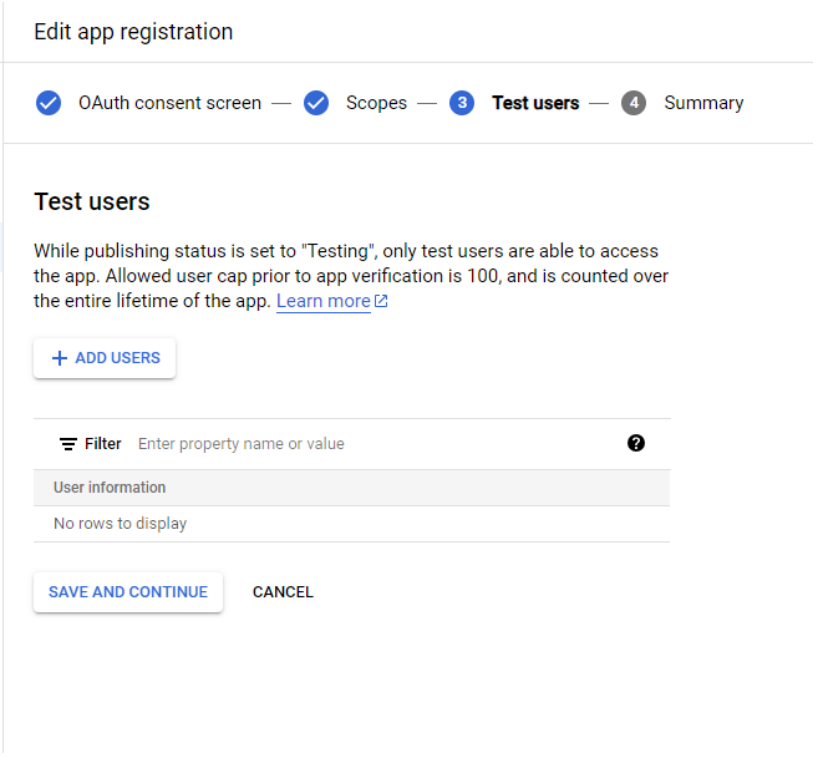

Рисунок 4.40. Налаштування consent screen. Меню Test users

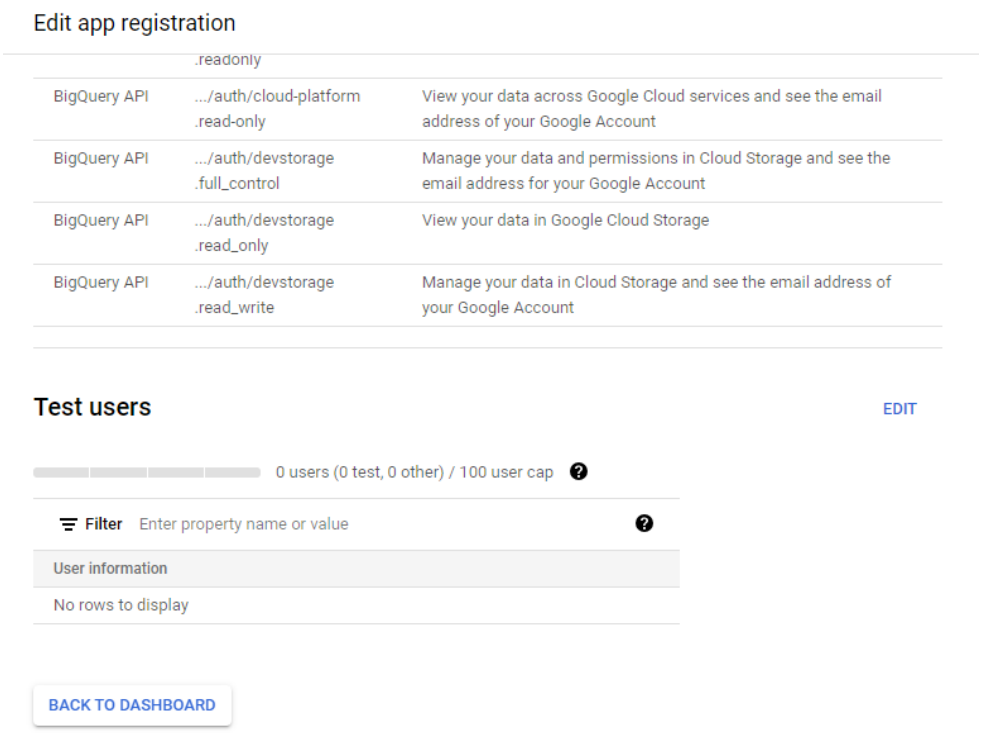

Рисунок 4.41. Фінальне налаштування consent screen

Далі було створено ідентифікатор клієнта (Рис. 4.42 – 4.44). Ідентифікатор клієнта OAuth у Google Cloud – це спеціальний ідентифікатор, який надається вашій програмі для автентифікації та авторизації користувачів за допомогою протоколу OAuth 2.0. Іншими словами, це «паспорт» вашого додатка, який дозволяє йому запитувати доступ до даних користувача в сервісах Google. Це головний етап, так як після створення ідентифікатору клієнта було отримано json файл, який необхідний для взаємодії чат-боту та гугл диску.

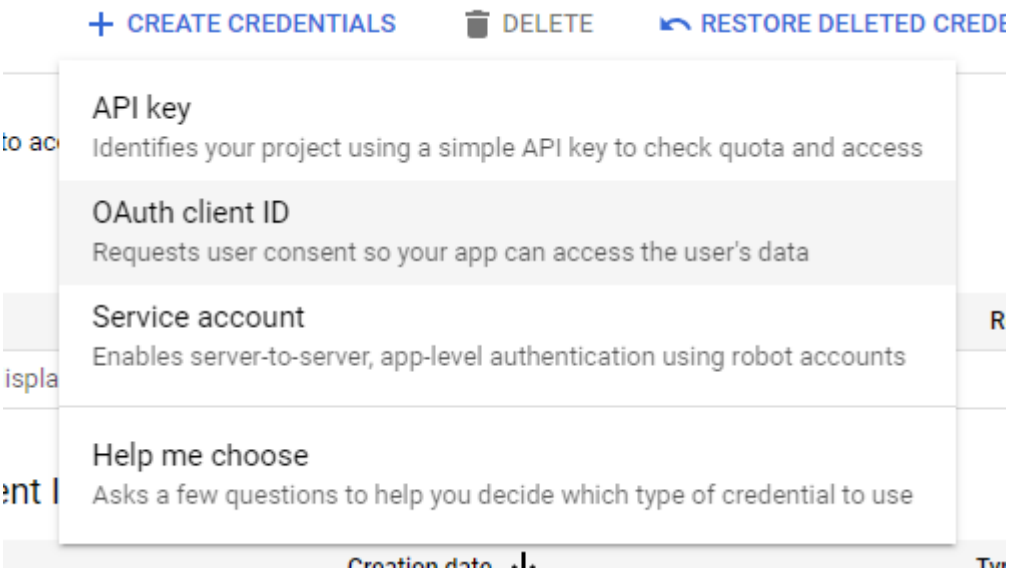

Рисунок 4.42. Створення ідентифікатора клієнта

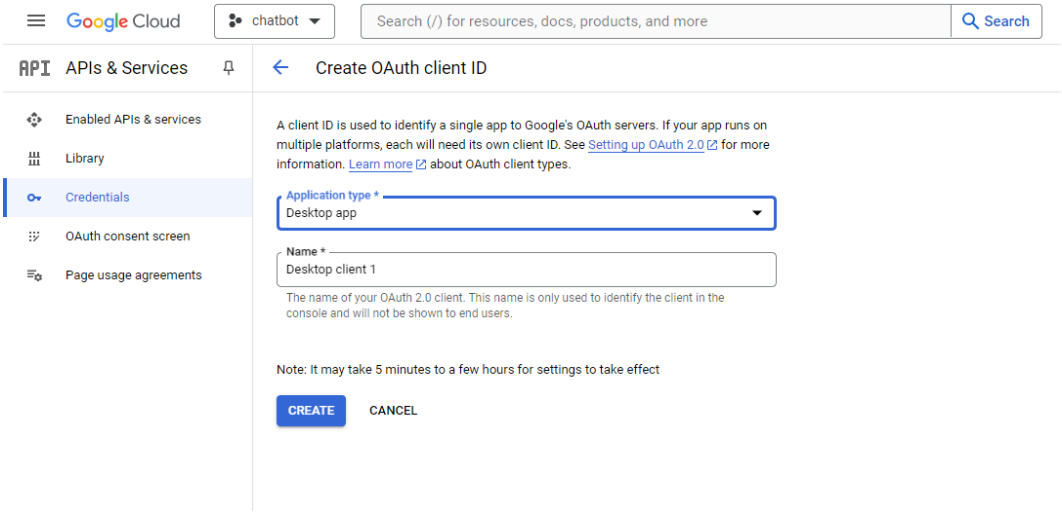

Рисунок 4.43. Налаштування ідентифікатора клієнта

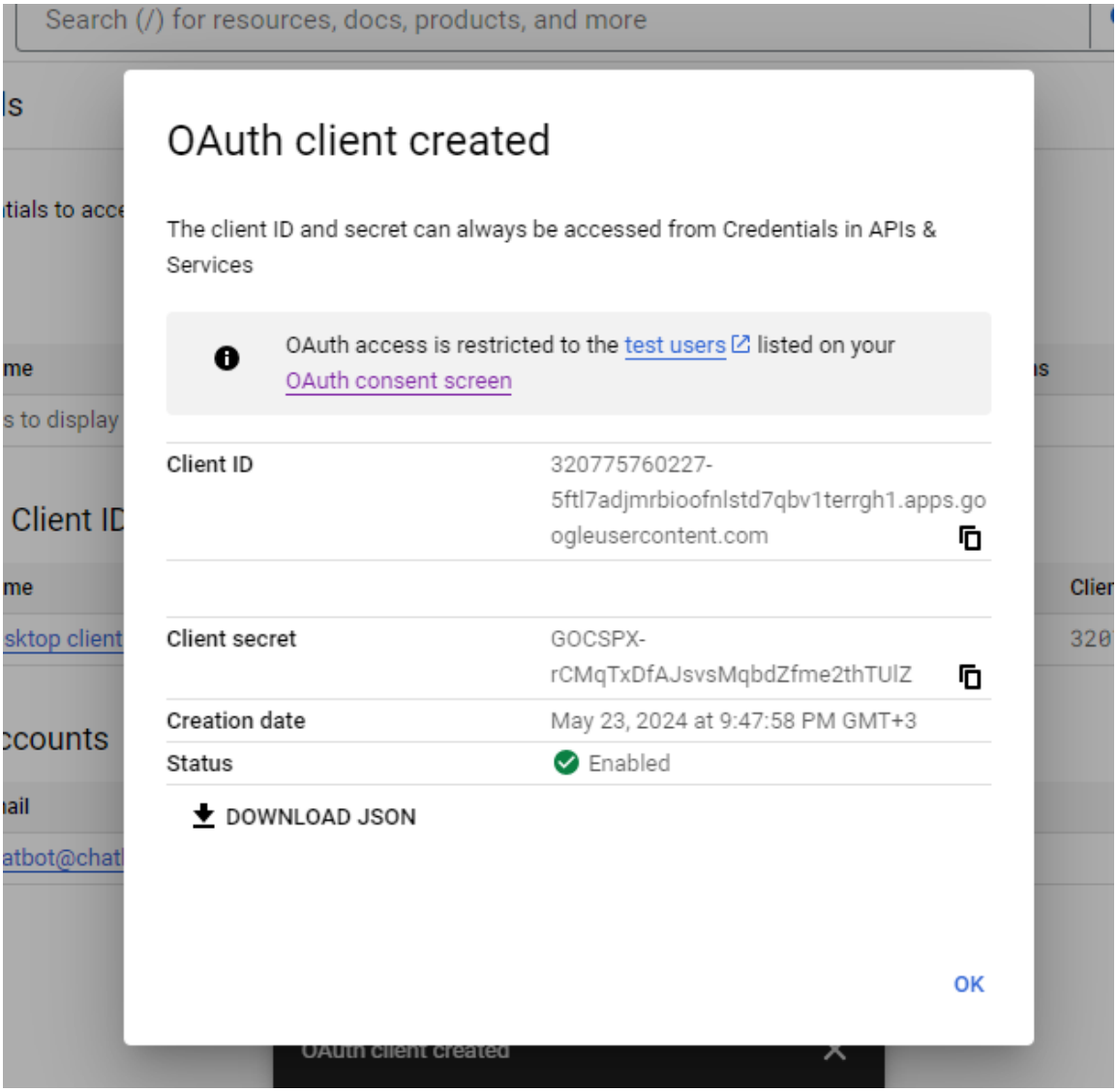

Рисунок 4.44. Успішне завершення створення ідентифікатора клієнту та завантаження json файлу

Наступним кроком було завантаження json файлу в папку з проектом чат-боту та перейменування його на credentials.json (Рис. 4.45).

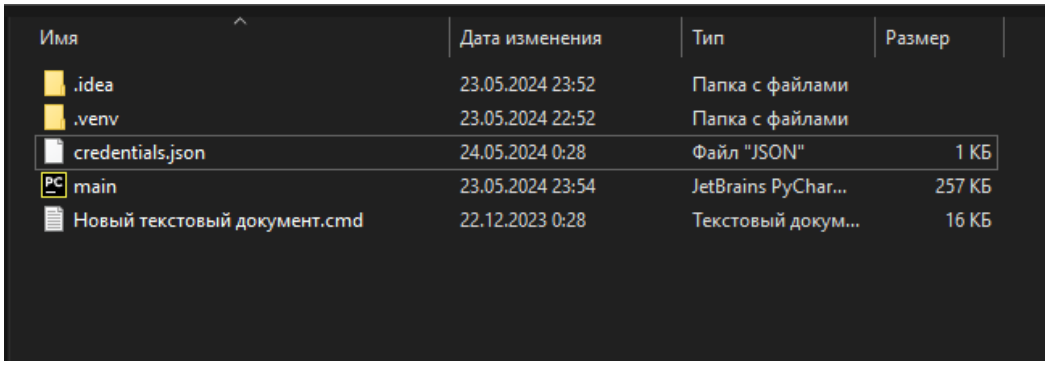

Рисунок 4.45. Завантаження json файлу в папку з проектом

Після встановлення файлу в папку з проектом, було встановлено бібліотеки, для роботи з гугл таблицями, а саме gspread та oauth2client (Рис. 4.46).

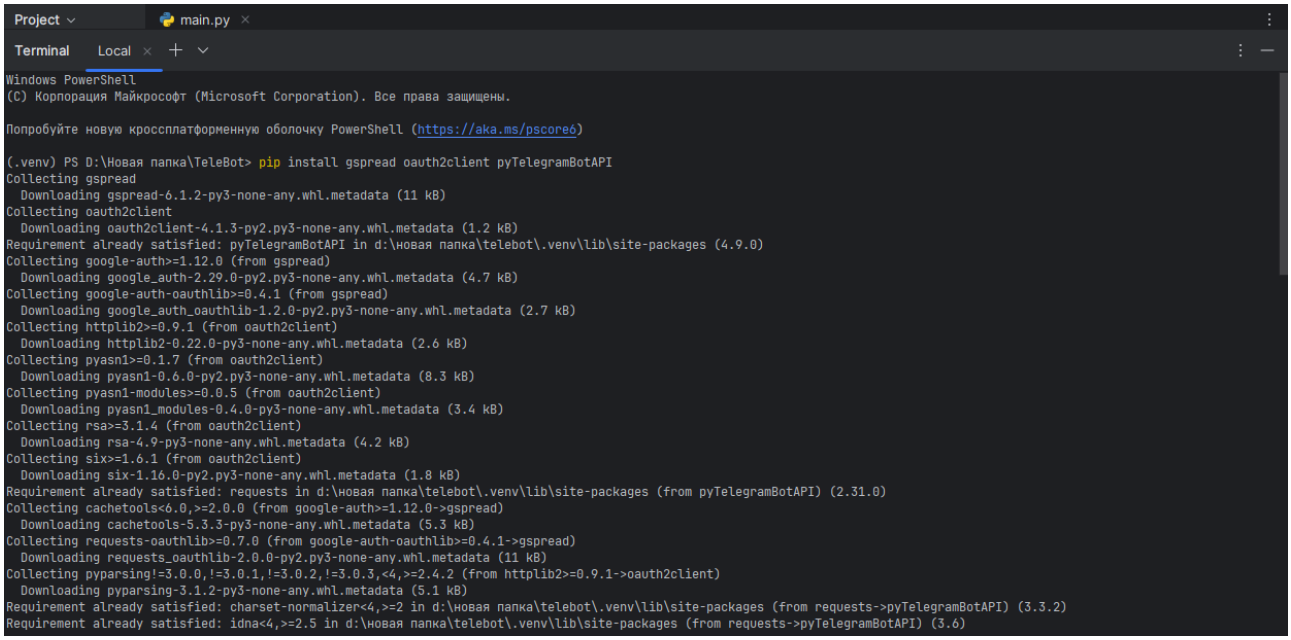

Рисунок 4.46. Встановлення бібліотек для роботи з таблицями

Фінальним кроком у підключенні Google sheets до чат-боту в телеграмі була інтеграція таблиці в бот. По-перше, була підключена бібліотека після встановлення в код, а саме gspread. Також, з цієї бібліотеки був імпортований файл ServiveAccountCredentials (Рис. 4.47). По-друге, виконалась інтеграція json файлу в бот, який був попередньо встановлений з Google Cloud. Для цього було вказано шлях до файлу (Рис. 4.48). Також разом з шляхом до json файлу була вказа адреса

таблиці, в яку завантажений розклад занять (Рис. 4.49). По-третє, була створена перевірка на правильне існування json файлу, та чи правильно вказаних шлях до нього (Рис. 4.50). Останнім кроком було відкриття таблиці та створення функції для отримання інформації про розклад (Рис. 4.51).

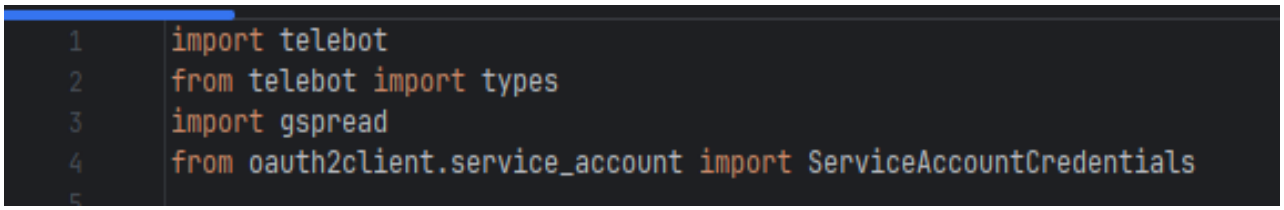

Рисунок 4.47. Встановлення бібліотеки gspread та імпорт файлу з неї

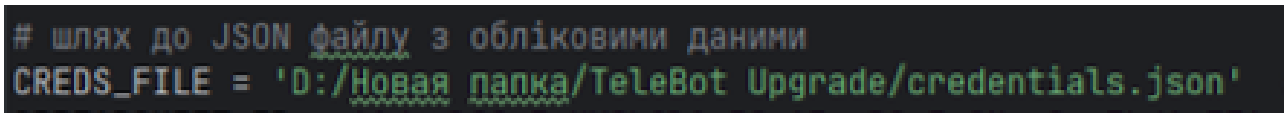

Рисунок 4.48. Шлях до json файлу

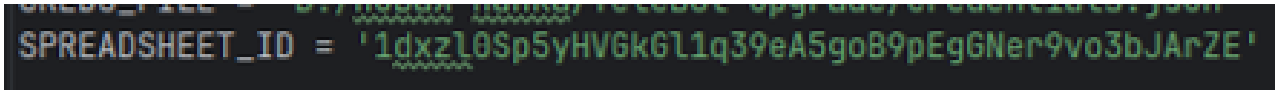

Рисунок 4.49. Ідектифікатр гугл таблиці

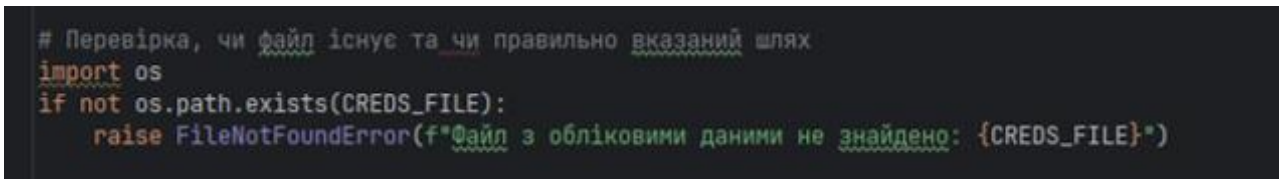

Рисунок 4.50. Перевірка на існування та правильність вказання шляху до json файлу

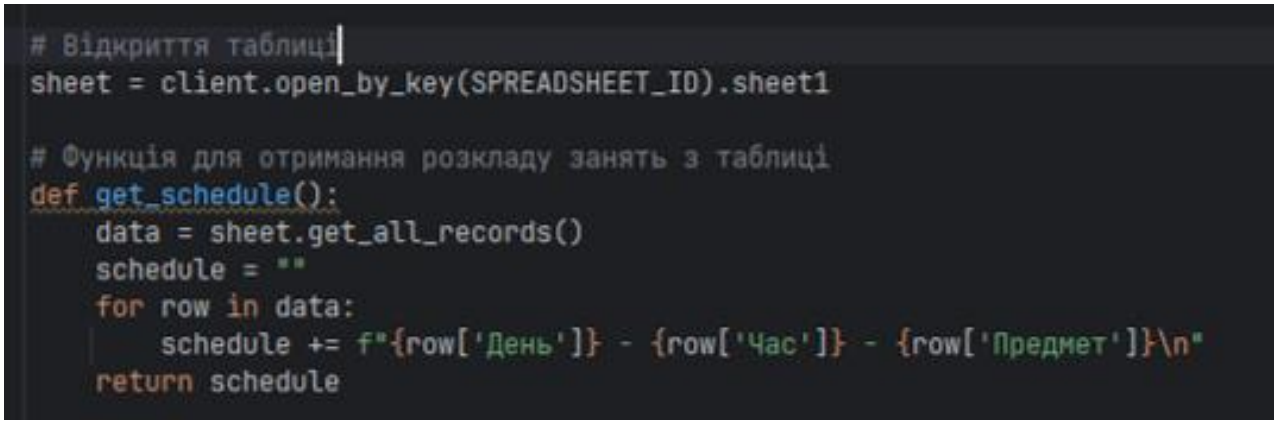

Рисунок 4.51. Написання коду для відкриття таблиці та створення функції для отримання розкладу занять з таблиці

4.5. Розробка логіки обробки повідомлень, враховуючи команди від користувачів та їхні запити

В чат-бот була додана логіка оброблення базових повідомлень від користувача. Якщо користувач надсилає базові фрази в чат-бот то він на це реагує та відповідає на дане повідомлення. Наприклад, якщо користувач надсилає “Привіт”, то бот зчитує цю інформацію та відповідає користувачу – “Привіт, {Ваше Ім’я}, радий тебе бачити!”. Так само чат-бот реагує на “Як справи”, “Бувай”, та “Що робиш?”. Були додані такі базові фрази, щоб бот міг також взаємодіяти з користувачем не тільки через команди та кнопки. Це було реалізовано через зчитування тексту, який попадає в чат. Якщо він співпадає з фразами, заданими в коді, бот реагує на це та відповідає користувачу (Рис. 4.52).

```
if message.text.lower() == 'привіт':
    bot.send_message(message.chat.id,
                     text: f'Привіт, <b>{message.from_user.first_name}</b> , радий тебе бачити!', parse_mode='html')
elif message.text.lower() == 'бувай':
    bot.send_message(message.chat.id, text: f'Бувайте, друже! Буду радий тебе бачити знову!', parse_mode='html')
elif message.text.lower() == 'як справи':
    bot.send_message(message.chat.id,
                     text: f'Чудово!', parse_mode='html')
elif message.text.lower() == 'як справи?':
    bot.send_message(message.chat.id, text: f'Чудово!', parse_mode='html')
elif message.text.lower() == 'що робиш':
    bot.send_message(message.chat.id, text: f'Чекаю, щоб допомогти тобі, <b>{message.from_user.first_name}</b> =)', parse_mode='html')
elif message.text.lower() == 'що робиш?':
    bot.send_message(message.chat.id, text: f'Чекаю, щоб допомогти тобі, <b>{message.from_user.first_name}</b> =)', parse_mode='html')
```

Рисунок 4.52. Реалізація логіки обробки повідомлень

Також, було додано реакцію боту на фото та відео. Якщо користувач надсилає чат-боту фото чи відео, бот на це реагує та відповідає (Рис. 4.53, 4.54).

Крім цього, ще була додана можливість видалити надіслану вами фотографію чи відео. Для цього були додані Inline кнопки, після натискання яких ваші файли видаляються (Рис. 4.55).

```
@bot.message_handler(content_types=['video'])
def del_video(message):
    markup = types.InlineKeyboardMarkup()
    video_btn1 = types.InlineKeyboardButton(
        text: 'Видалити відео', callback_data='delete_video')
    markup.row(video_btn1)
    bot.reply_to(message,
                 text: 'Класне відео! Мені сподобалось.', reply_markup=markup)
```

Рисунок 4.53. Реалізація реакції на надіслане відео

```
@bot.message_handler(content_types=['photo'])
def del_photo(message):
    markup = types.InlineKeyboardMarkup()
    photo_btn1 = types.InlineKeyboardButton(
        text: 'Видалити фотографію', callback_data='delete_photo')
    markup.row(photo_btn1)
    bot.reply_to(message,
                 text: 'Дуже красива фотографія!', reply_markup=markup)
```

Рисунок 4.54. Реалізація реакції на надіслане фото

```
if callback.data == 'delete_photo':
    bot.delete_message(callback.message.chat.id, callback.message.message_id - 1)
elif callback.data == 'delete_video':
    bot.delete_message(callback.message.chat.id, callback.message.message_id - 1)
```

Рисунок 4.55. Реалізація видалення фото та відео

4.6. Створення аватарки та хостинг чат-боту

"Аватарка" (або "аватар") - це зображення, яке представляє користувача в інтернеті. Це може бути фотографія особи, ілюстрація, символ чи будь-яке інше зображення, яке використовується для ідентифікації користувача на платформах соціальних мереж, форумів, чат-ботів тощо. Зазвичай аватарка відображається поруч із ім'ям користувача чи профілем у цих мережах та служить для візуальної ідентифікації користувача в онлайн-середовищі.

В моєму випадку була створена картинка яка представляє чат-бот. Створення відбувалось в програмі Photoshop.

Photoshop - це програма для обробки зображень, розроблена компанією Adobe. Це потужний інструмент для редагування фотографій, створення графіки, малювання, ретуші, монтажування зображень та багато іншого.

Процес створення аватарки та кінцевий результат (Рис. 4.56).

Рисунок 4.56. Процес створення аватарки чат-бота

Для встановлення картинки в чат-бот, необхідно перейти в управління нашим ботом, через @BotFather. Після цього було введено команду /setuserpic, та встановлено аватарку чат-боту (Рис. 4.57).

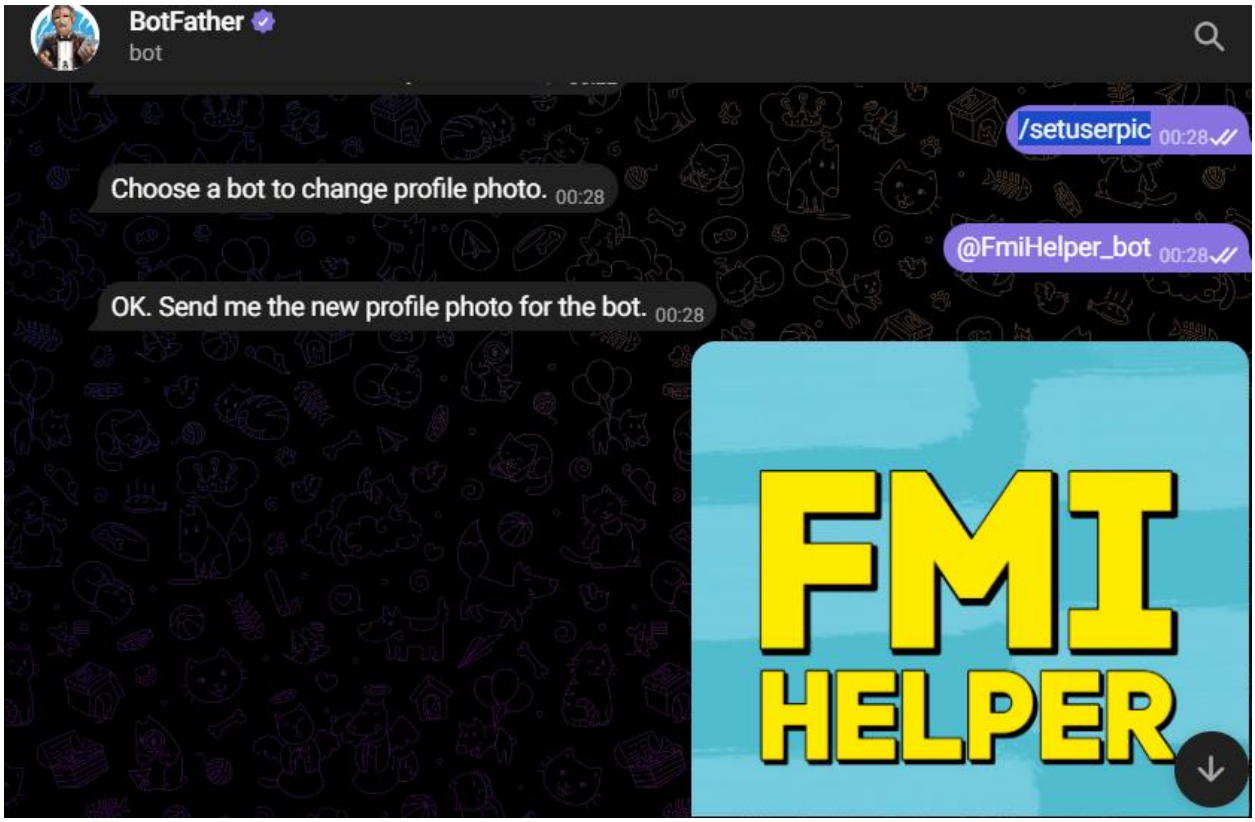

Рисунок 4.57. Встановлення аватарки

Наступним кроком було завантаження чат-боту на хостинг, так як бот працював тільки при запущеному проекті в середовищі розробки PyCharm.

Хостинг для чат-бота - це місце, де програмний код бота розміщується, щоб бот був доступний онлайн і міг взаємодіяти з користувачами у реальному часі.

Існує кілька способів хостити чат-бота:

1. *Хостинг на сервері*: ви можете встановити програмний код чат-бота на власному сервері або орендованому хостингу.
2. *Хостинг через платформи для ботів*: деякі платформи, такі як Heroku, PythonAnywhere, DigitalOcean, пропонують послуги хостингу для різноманітних програм, включаючи чат-ботів.
3. *Хостинг на платформах месенджерів*: деякі платформи месенджерів, наприклад, Telegram, можуть надавати власні можливості хостингу чат-ботів на своїх серверах.

Мною було обрано варіант хостингу через платформу для ботів, а саме на платформі PythonAnywhere.

PythonAnywhere - це хмарна платформа для розгортання веб-додатків на мові програмування Python. Вона надає можливість розміщувати, редагувати та запускати Python-код безпосередньо в хмарі, що дозволяє створювати й виконувати веб-додатки, включаючи чат-боти.

Для хостингу спочатку необхідно завантажити наш проект на сайт хмарної платформи (Рис. 4.58).

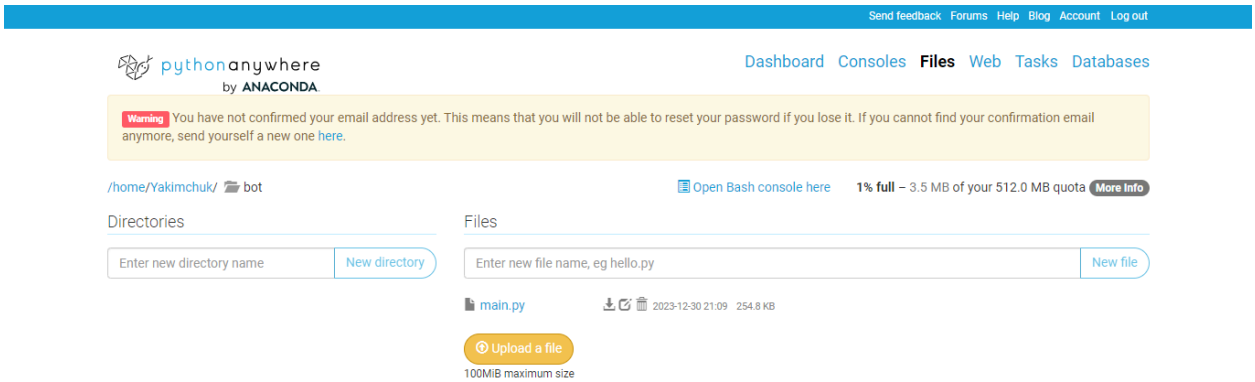

Рисунок 4.58. Завантаження файлу проекту на платформу

Після чого, була відкрита консоль, завантажена бібліотека pyTelegramBotAPI, за допомогою команди pip install (Рис. 4.59), та відкриття нашого файлу в консолі через команду python (назва проекту.py) (Рис. 4.60).

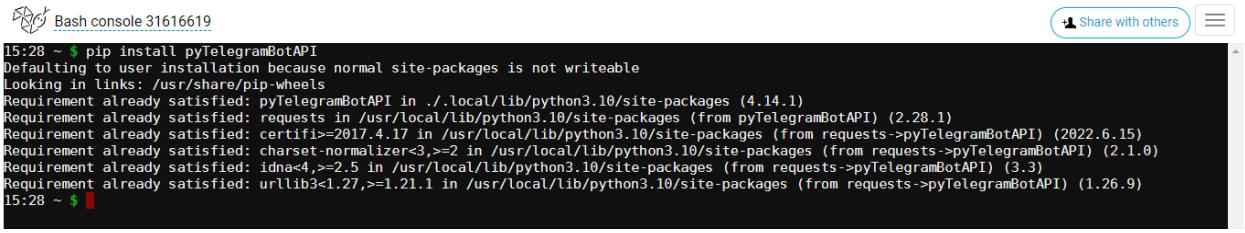

Рисунок 4.59. Завантаження бібліотеки в консоль на платформі

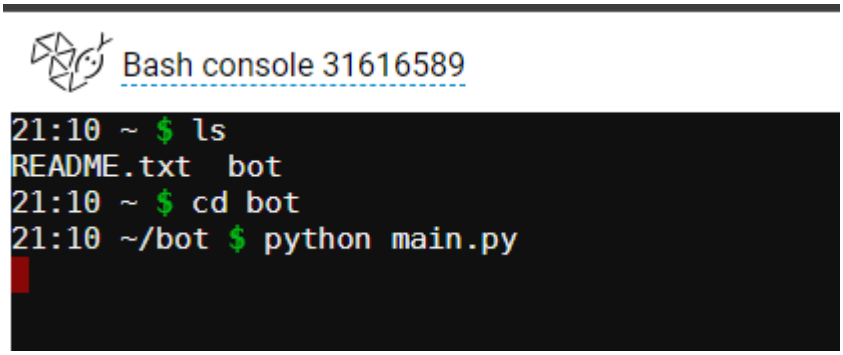

Рисунок 4.60. Запуск боту на платформі

РОЗДІЛ 5. ТЕСТУВАННЯ ТА ОЦІНКА РЕЗУЛЬТАТІВ

Наступним кроком було тестування та оцінка результатів чат-боту. *Ручне тестування* - це метод тестування, при якому тести виконуються вручну без використання автоматизованих скриптів або інструментів. У цьому методі тестер вручну виконує тести, надаючи вхідні дані та перевіряючи результати, щоб переконатися, що програмний продукт працює правильно та відповідає вимогам.

Мною було проведення тестування в декількох етапах:

1. *Тестування функціональності*:

   Тестування можливості бота реагувати на різні запити щодо розкладу, наприклад, запити на конкретні предмети, групи тощо.

   Перевірка точності інформації, яку надає бот.

2. *Тестування взаємодії*:

   Перевірка способів взаємодії з ботом (натискання кнопок, введення тексту).

3. *Тестування на різних пристроях*:

   Перевірка відображення та функціональності бота на різних пристроях та платформах (ПК, смартфони, планшети тощо).

Першочергово було проведено *тестування функціональності*. Перший пункт - Тестування можливості бота реагувати на різні запити щодо розкладу, наприклад: запити на конкретні предмети, групи тощо. На цьому етапі тестування ніяких проблем виявлено не було, чат-бот реагує на запити коректно (Рис. 5.1, 5.2, 5.3).

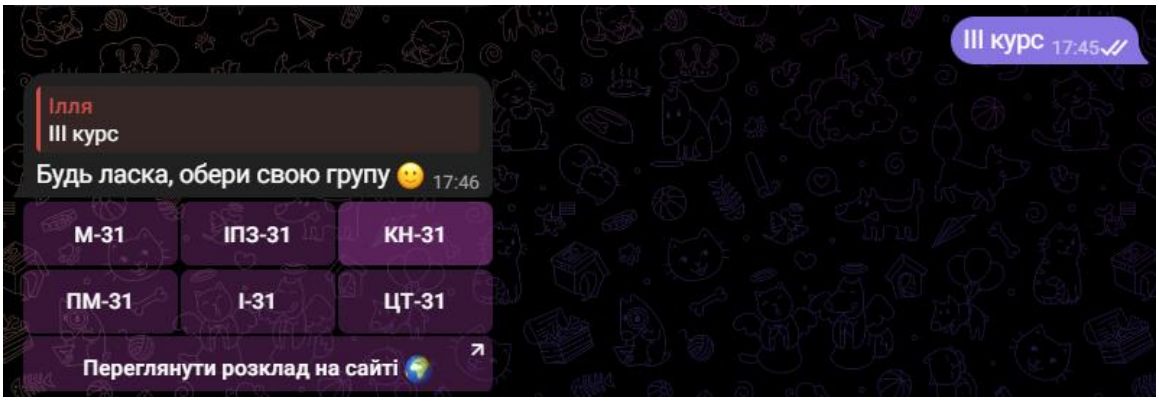

Рисунок 5.1. Коректне відображення груп третього курсу

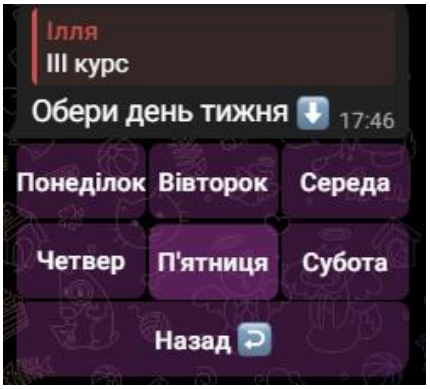

Рисунок 5.2. Коректне відображення днів тижня для групи КН-31

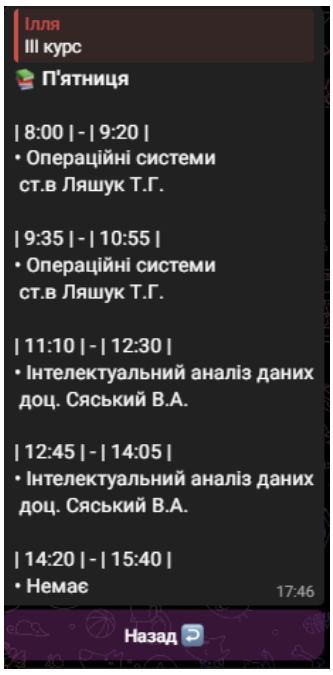

Рисунок 5.3. Коректне відображення розкладу на П’ятницю для групи КН-31

Далі була перевірка точності інформації, яку надає бот. Так як вся інформація про розклад для чат-боту бралась з офіційного розкладу від деканату, вся інформація співпадає (Рис. 5.4).

Рисунок 5.4. Відображення розкладу на П’ятницю для групи КН-31

Другим кроком було проведено *тестування взаємодії*, а саме перевірка способів взаємодії з ботом (натискання кнопок, введення тексту). На цьому етапі проблем не було виявлено, всі кнопки, команди та введеня тексту працюють коректно. Для прикладу на рисунках зображені: реакція боту на команду /start, коректне відображення вікна “Контакти”, реакція боту на написання в чат “Привіт” та реакція боту на надсилання фото в чат (Рис. 5.5, 5.6, 5.7, 5.8).

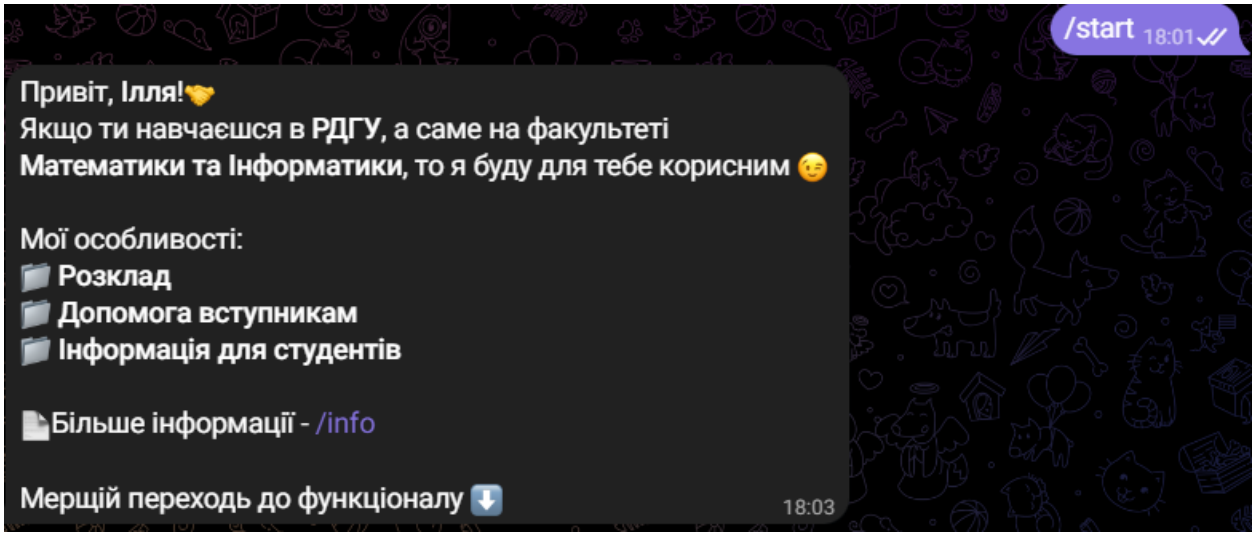

Рисунок 5.5. Реакція боту на команду /start

Рисунок 5.6. Відображення вікна “Контакти”

Рисунок 5.7. Реакція боту на написання в чат “Привіт”

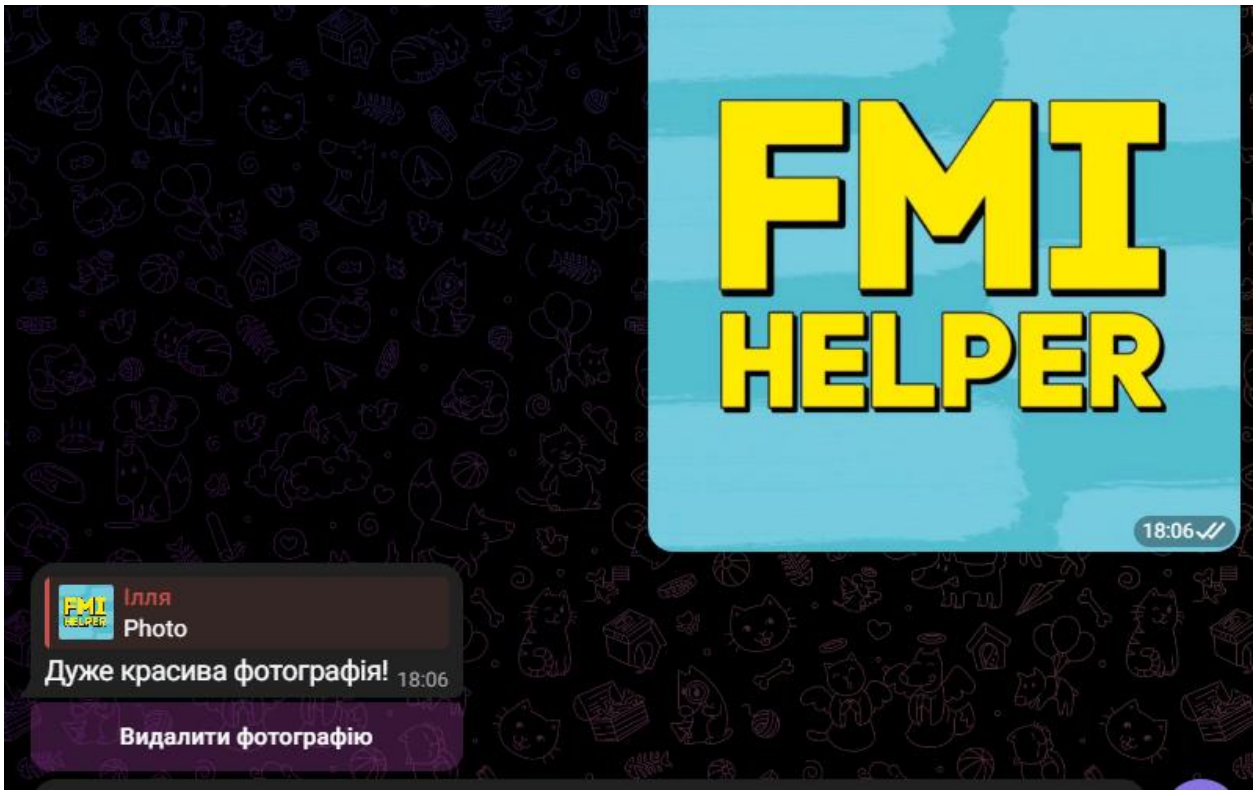

Рисунок 5.8. Реакція боту на надсилання фото в чат

Третім кроком було проведено *тестування на різних пристроях*, а саме перевірка відображення та функціональності бота на різних пристроях та платформах (ПК, смартфони, планшети тощо).

На цьому етапі тестування також ніяких проблем виявлено не було, чат-бот коректно відображає інформацію як на ПК, так і на телефонах та планшетах. Приклад роботи бота на телефоні (Рис. 5.9).

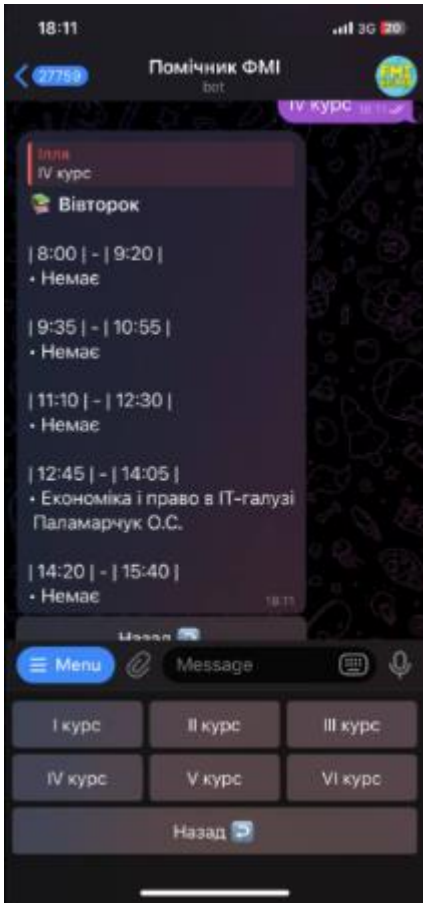

Рисунок 5.9. Робота чат боту на телефоні

ВИСНОВКИ

Результати дослідження підтверджують важливість впровадження сучасних технологій розробки інформаційних чат-ботів у різних сферах. На прикладі чат-бота для оповіщення про розклад занять факультету, створеного з використанням Python у поєднанні з бібліотекою pyTelegramBotAPI та платформою Telegram, було вирішено низку проблем, пов'язаних із забезпеченням студентів актуальною інформацією про розклад занять. Розроблений інструмент надає студентам зручний та швидкий доступ до розкладу, що значно покращує освітній процес.

В процесі розробки було розглянуто основні аспекти інтеграції чат-ботів у сучасну освітню сферу. Зокрема, було вирішено питання оперативного оновлення інформації та забезпечення її доступності з будь-якого пристрою, де є месенджер. Інтерактивний інтерфейс чат-бота підвищує зручність і доступність інформації для користувачів, надаючи важливий ресурс для ефективного навчання.

Дослідження підкреслює важливість інноваційних технологій у сучасній освіті та їх здатність полегшувати доступ до необхідних даних. Результати демонструють, як технологічні інновації можуть змінити підхід до освіти, забезпечуючи більш ефективний та доступний процес навчання.

Подальший розвиток застосунку включатиме додаткові можливості, такі як розширена інтеграція з іншими платформами для підтримки актуальності інформації та забезпечення максимальної зручності користувачам. Такі ініціативи сприятимуть покращенню процесів навчання та співпраці в освітній галузі. Окрім оповіщення про розклад, чат-бот в майбутньому може розширити свій функціонал, надаючи: нагадування про важливі події та дедлайни, запровадження системи зворотного зв'язку для отримання відгуків від користувачів з метою постійного вдосконалення сервісу, тощо .

Таким чином, результати дослідження свідчать, що використання інформаційних чат-ботів, таких як чат-бот для розкладу занять, є важливим для вдосконалення освітнього процесу та покращення комунікації між студентами і навчальним закладом. Технологічні інновації, реалізовані в цій роботі, відкривають перспективи для подальших розробок у сфері інформаційних чат-ботів, що сприятиме покращенню процесів навчання та співпраці в освіті.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Чат-бот. Поняття, історія розвитку. URL: https://uk.wikipedia.org/wiki/чат-бот (дата звернення: 14.05.2024).
2. Самко М. Як створюються чат-боти. URL: https://lemon.school/blog/yak-stvoryuyutsya-chat-boty (дата звернення: 14.05.2024).
3. Python Software Foundation. Підручник з Python. URL: https://docs.python.org/uk/3/tutorial (дата звернення: 16.05.2024).
4. How to Create a Telegram Bot using Python. URL: https://www.freecodecamp.org/news/how-to-create-a-telegram-bot-using-python/ (дата звернення: 13.05.2024).
5. Nicolas Modrzyk. Building Telegram Bots: Develop Bots in 12 Programming Languages using the Telegram Bot API. NY: Apress. 2018. 319 p.
6. Bots: An introduction for developers. URL: https://core.telegram.org/bots (дата звернення: 10.05.2024).
7. pyTelegramBotAPI 4.0.1. URL: https://pypi.org/project/pyTelegramBotAPI/ (дата звернення: 18.05.2024).
8. Підручник з Python. URL: https://docs.python.org/uk/3.10/tutorial/index.html (дата звернення: 7.05.2024).
9. PyCharm. URL: https://uk.wikipedia.org/wiki/PyCharm (дата звернення: 31.05.2024).
10. Як створити чат-бота у Telegram. URL: https://sendpulse.ua/knowledge-base/chatbot/telegram/create-telegram-chatbot (дата звернення: 21.05.2024).
11. Google Таблиці. URL: https://uk.wikipedia.org/wiki/Google_Таблиці (дата звернення: 20.05.2024).
12. Getting started. PyCharm Documentation. URL: https://www.jetbrains.com/help/pycharm/getting-started.html (дата звернення: 13.05.2024).
13. Sumit Raj. Building Chatbots with Python: Using Natural Language Processing and Machine Learning. CA: Apress. 2019. 217 p.

14. Yuli Vasiliev. Python for the Real World: Build Skills in Python Programming and Understand How to Work with APIs. NY: No Starch Press. 2021. 300 p.
15. Paul Barry. Head First Python: A Brain-Friendly Guide. CA: O'Reilly Media. 2016. 624 p.
16. Alexander Stepanov. Mastering PyCharm: A Comprehensive Guide to Using PyCharm Effectively for Python Development. CA: Packt Publishing. 2016. 265 p.
17. Google Sheets API Overview. URL: https://developers.google.com/sheets/api/guides/concepts?hl=en (дата звернення: 4.05.2024).
18. Python quickstart. Google Sheets. Google for Developers. URL: https://developers.google.com/sheets/api/quickstart/python?hl=en (дата звернення: 8.05.2024).