**ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ**
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра інженерії програмного забезпечення

# Пояснювальна записка

до бакалаврської кваліфікаційної роботи
на ступінь вищої освіти бакалавр

на тему: **«РОЗРОБКА CHAT-BOT ДЛЯ ОПТИМІЗАЦІЇ НАВЧАЛЬНОГО ПРОЦЕСУ СИСТЕМОЮ MOODLE НА МОВІ PYTHON»**

Виконав: студент 4 курсу, групи ПД-44
спеціальності 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)
Чуб Є.М.
(прізвище та ініціали)
Керівник Негоденко О.В.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)
Нормоконтроль
(прізвище та ініціали)

Київ – 2021

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ**

**Навчально-науковий інститут інформаційних технологій**

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти - «Бакалавр»

Спеціальність -121 Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного
забезпечення
_____О.В. Негоденко
«____»_____2021 року

**З А В Д А Н Н Я**

**НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ**

**ЧУБУ ЄВГЕНІЮ МИХАЙЛОВИЧУ**

1.Тема роботи: «Розробка chat-bot для оптимізації навчального процесу системою Moodle на мові Python»

Керівник роботи Негоденко Олена Василівна, доцент, кандидат технічних наук затверджені наказом вищого навчального закладу від — «12» березня 2021 року №65.

2. Строк подання студентом роботи 01.06.2021

3. Вхідні дані до роботи:

3.1. Середовище розробки Sublime Text Editor 3

3.2. Алгоритм роботи бота

3.3. Aiogram

3.4. Науково-технічна література, пов’язана з розробкою ботів на основі

спеціальних API

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити).

4.1. Аналіз обов’язків розроблюваного бота

4.2. Аналіз та порівняння існуючих прототипів

4.3. Дослідження програмних засобів для розробки бота

4.4. Розробити функціонал створеного бота

6. Дата видачі завдання 19.04.2021

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів бакалаврської роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Підбір науково-технічної літератури | 19.04.21 – 22.04.21 | |
| 2 | Аналіз існуючих прототипів | 22.04.21 – 23.04.21 | |
| 3 | Дослідження програмних засобів | 23.04.21 – 01.05.21 | |
| 4 | Моделювання об’єкту проектування | 01.05.21 – 05.05.21 | |
| 5 | Розробка функціоналу бота | 05.05.21 – 07.05.21 | |
| 6 | Вступ, висновки, реферат | 07.05.21 – 08.05.21 | |
| 7 | Розробка презентації застосунку | 08.05.21 – 24.05.21 | |
| 8 | Попередній захист роботи | 25.05.21 | |

Студент

Керівник роботи

# РЕФЕРАТ

Текстова частина бакалаврської роботи 40с., 24 рис., 16 джерел.

Ключеві слова: Sublime Text, Python, Aiogram, Telegram, чат-бот, парсинг, Requests

*Об'єкт дослідження* – збір та обробка інформації в системі дистанційного навчання Moodle.

*Предмет дослідження* – інформаційна ситема для перегляду оцінок та отримання сповіщень пов'язаних з останніми термінами здачі робіт.

*Мета роботи* – підвищити ефективність дистанційного навчання за допомогою розробки чат-боту на мові програмування Python.

*Методи дослідження* – методи теорії інформації, методи структурного аналізу і проектування, методи розробки програмного забезпечення, методи тестування, валідації та верифікації програмного забезпечення.

В роботі виконано аналіз існуючих ботів аналогів. Встановлено переваги та недоліки існуючих ботів. В результаті аналізу було визначено основні потреби користувачів. Проаналізовано можливості середовища розробки Sublime Text. Розроблено логіку практичних завдань та загальну концепцію представлення інформації для користувачів.

Галузь використання – дистанційне навчання в Державному університеті телекомунікацій.

**ЗМІСТ**

# ВСТУП

Стрімкий розвиток інформаційних та телекомунікаційних технологій назавжди змінив уявлення людей про спілкування на відстані. Ще якихось 100 років тому люди й уявити не могли що їхні нащадки матимуть змогу спілкуватися між собою в режимі реальному часу знаходячись при цьому в різних куточках планети.

Загальна доступність таких пристроїв як комп'ютери та смартфони, а також зростаюча швидкість обміну даними в мережі інтернет сприяли новій хвилі популярності таких засобів зв'язку як месенджери, які дозволяють обмінюватися текстовими, аудіо, відео повідомленнями та файлами різних форматів.

Разом із поширенням месенджерів серед користувачів мережі інтернет, почали набувати розповсюдження додатки чат-боти розроблені на їх платформі. Чат-бот – це віртуальний помічник, який має певні функції та покликаний допомогти користувачеві в задоволенні його повсякденних потреб, використовуючи при цьому інтерфейс чату.

Актуальність дипломної роботи можна пояснити широкою областю застосування чат ботів. На сьогоднішній день вони активно використовуються в різних сферах людського життя, від розваг та пошуку інформації до здійснення електронних платежів та навчання.

Метою дипломної роботи є створення чат-боту на платформі месенджеру Telegram, котрий буде надавати інформацію студентам ДУТ стосовно оцінок з навчальних дисциплін та сповіщати їх щодо останніх термінів здачі лабораторних та практичних робіт.

# 1 АНАЛІЗ ВИМОГ

Завданням дипломного проекту було визначенно створення телеграмм-боту для оптимізації навчального процесу студентів ДУТ системою дистанційного навчання Moodle.

Перед початком створення телеграм-боту, було визначено та описано основні вимоги. Також було проведено аналіз предметної області, та розглянуто існуючі рішення. Виходячи з поставлених цілей було розроблено чат-бот, який має наступний функціонал:

a) Авторизація на сайті дистанційного навчання Moodle.
b) Збір та систематизація інформації про навчальні дисципліни, роботи та оцінки
c) Відправлення цієї інформації в структурованому вигляді користувачу за запитом
d) Збір та систематизація даних про останні терміни здачі робіт
e) Відправлення студентам сповіщень про останні терміни здачі робіт
f) Налаштування часу приходу сповіщень

# 2 АНАЛІЗ ОБЛАСТІ ДОСЛІДЖЕНЬ

## 2.1 Месенджери

Концепція обміну миттєвими повідомленнями стала широко розповсюдженою в 90-х роках 20 століття, дозволяючи друзям, родичам, знайомим, колегам та однодумцям з усього світу спілкуватися в режимі реального часу.

З тих пір обмін миттєвими повідомленнями спричинив революцію в образі людського спілкування і сьогодні близько 2.5 мільярдів осіб використовують хоча б один додаток для обміну повідомленнями. Теперішня система обміну миттєвими повідомленнями є бездоганною та інтегрує такі функції як, відео, фото, голос, електронна комерція та ігри зі звичайним обміном повідомленнями.

Однак, неймовірний успіх та широкий функціонал сучасних застосунків були б просто неможливими без ранніх досягнень їх більш обмежених попередників, тому розглянемо основні етапи еволюції засобів обміну миттєвими повідомленнями.

В 1961 році Сумісна система розподілу часу (CTSS) Масачусетського технологічного інституту, разом з іншими багатокористувацькими системами, стала піонером в області обміну миттєвими повідомленнями, дозволяючи до 30 користувачам спілкуватися в чаті в режимі реального часу.

В 1988 році було розроблено технологію Internet Relay Chat (IRC), яка дозволяла підключатись до мережі за допомогою клієнтського програмного забезпечення, для спілкування з групами людей в режимі реального часу. Пік популярності IRC припав на 1990 роки.

В 1996 році ізраїльска компанія Mirabilis запустила перший з відомих месенджерів для загального користування ICQ, який дозволяв користувачам спілкуватись як у приватних так і в групових чатах, обмінюватись файлами та здійснювати пошук по базі користувачів. В пік свого розвитку в 2001 році в ICQ було зареєстровано більш ніж 100 тисяч користувачів.

В 2009 році розробник українського походження Ян Кум створив сервіс, який мав змогу показувати статус усіх контактів в телефонній книзі користувача. Через функціональну обмеженість сервіс WhatsApp не був затребуваний серед користувацької аудиторії. Однак після ряду оновлень додатку, аудиторія WhatsApp за декілька тижнів виросла до 250 тисяч користувачів.

Почали набувати розвитку та поширення месенджери, які пропонували нові функціональні можливості – від здійснення дзвінків до обміну аудіо, фото та відео матеріалами. Кожен новий месенджер, який з'являвся на тих чи інших платформах, для того щоб бути конкуренто спроможним, повинен був представляти нові функціональні нововведення. Прикладом є Viber.

Viber, який був розроблений в 2010 році, став першим месенджером з функцією здійснення безкоштовних дзвінків через мобільний інтернет. Пізніше інші компанії також впровадили цю функцію в свої месенджери.

Функціональний розвиток месенджерів не стояв на місці. Наприклад в 2014 році китайський сервіс WeeChat запустив і відео-дзвінки.

Сьогодні месенджери – це соціальні платформи здатні передавати аудіо, відео, аудіовізуальні та текстові повідомлення за секунди не залежно від географічного розташування їх користувачів. Системи миттєвого обміну повідомленнями дозволяють взаємодіяти людям один з одним не тільки в форматі діалогу, а й за допомогою створення спільних чатів користувачів, об'єднаних спільними інтересами, цілями тощо.

## 2.2 Чат-боти

В останні роки такі програми як чат-боти стають все більш популярними серед користувачів сервісів обміну повідомленнями.

Перспективність їх використання обумовлена тим, що при відповідному підході їх можна інтегрувати практично в будь яку сферу людської діяльності – від розважальної до таких серйозних як бізнес, освіта та медицина.

Чат-боти – це програмне забезпечення розроблене на платформі месенджерів або соціальних мереж, яке має вигляд звичайного акаунту, але з деякими відмінностями та призначене для взаємодії з користувачем за допомогою текстових або аудіо повідомлень без участі людини-оператора.

Чат-бот працює по принципу розпізнання ключових слів, на основі яких він формує відповідь.

Існує два основних типи чат-ботів:

1. Чат боти на основі правил.

Даний тип – це чат-боти, розроблені на основі дерева рішень та запрограмовані за допомогою певного набору правил, призначених для вирішення конкретних запитів. Потенціальні діалоги визначені у вигляді блок-схеми з відповідями на всі можливі питання. Недоліком даних чат ботів є те, що вони не здатні відповідати на питання які не пов’язані між собою та є випадковими. Такі боти пропонують виключно відповіді на заздалегідь запропоновані питання.

2. Чат боти з використаннями штучного інтелекту

На відміну від попереднього типу, чат-боти зі штучним інтелектом можуть використовувати машинне навчання для розуміння намірів користувача та контексту повідомлень. З часом вони можуть вчитись на відгуках і помилках, з метою надання більш точних відповідей.

Розглянемо переваги використання чат-ботів:

1) Цілодобова доступність

Однією з головних переваг є те, що чат-боти доступні для клієнтів 24 години на добу. Крім того вони здатні дуже швидко та оперативно давати відповіді на задані питання. Це гарантує те, що клієнти завжди зможуть знайти рішення своїх проблем

в будь який день тижня та годину доби. З персональною підтримкою клієнтів досить важко забезпечити такий сервіс.

2) Економія ресурсів

Використання чат-ботів дозволяє бізнесу скоротити витрати, так як вони звільняють від необхідності у великому штаті спіробітників служби підтримки, яким щомісячно потрібно виплачувати заробітну плату.

3) Інтеграція в одному додатку

Ще однією безперечною перевагою є можливість, на платформі лише одного додатку, користуватись відразу декількома чат-ботами, які мають різне призначення, замість встановлення великої кількості окремих спеціалізованих застосунків на свій смартфон або комп'ютер.

Розглянемо чат-бот – для створення нагадувань в месенджері Telegram

Чат-бот Skeddy – це простий але ефективний інструмент, який дозволяє створювати нагадування та нотатки, та керувати ними.

Після налаштувань бот буде надсилати вам нагадування в заданий вами час.

Список основних функцій:

1. Створення простих нагадувань на природній мові. Наприклад: «Пройти медичний огляд 6 березня».
2. Бот має веб-інтерфейс, який дозволяє створювати нагадування з будь-яким розкладом. Наприклад: «Вносити оплату за комунальні платежі кожене 1 число день місяця».
3. Можливість у будь який момент переглянути список своїх нагадувань
4. Можливість вимкнути нагадування, якщо воно зараз не актуальне.
5. Можливість створювати прості нотатки без розкладу. Наприклад можна зберегти свій список покупок.

На рисунку 2 зображено приклад роботи чат-бота.

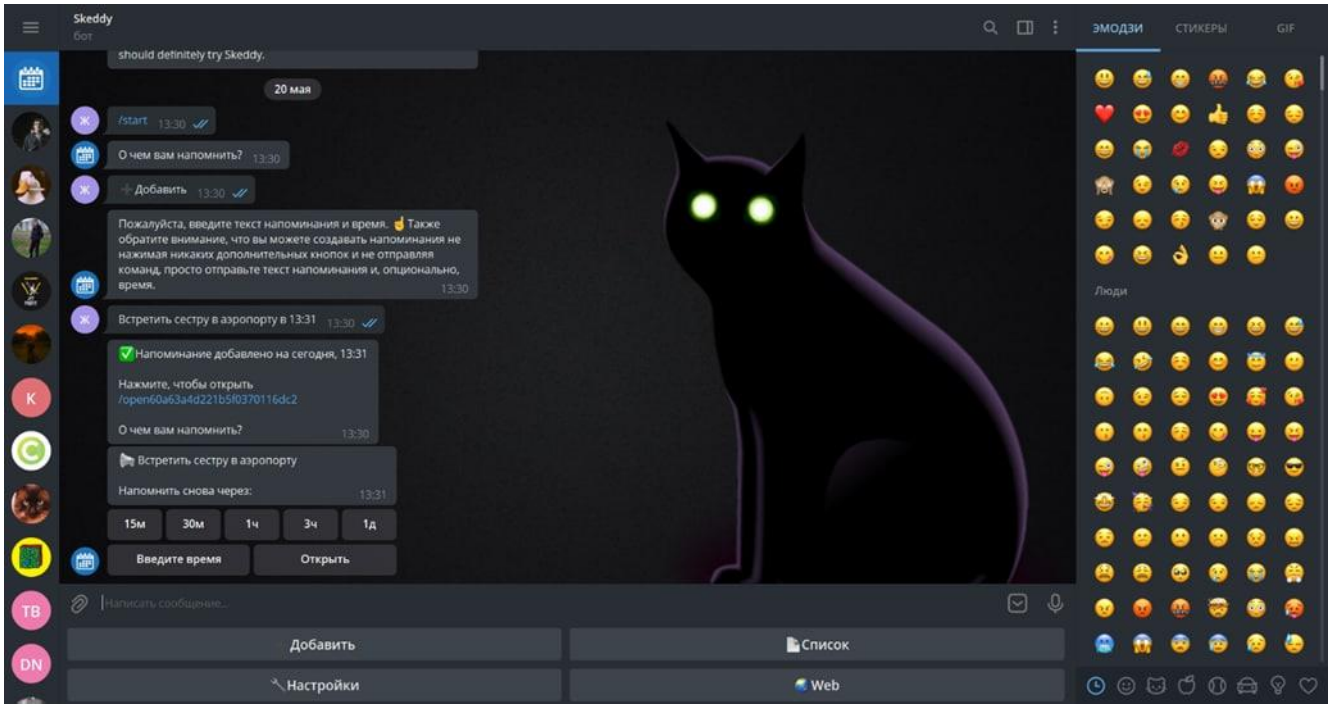


Рисунок 1 – Робота чат-боту Skeddy

### 2.3 Telegram

Telegram – це хмарний онлайн сервіс для обміну текстовими, аудіо, відео повідомленнями, а також файлами та документами різних форматів. Месенджер був розроблений братами Павлом та Миколою Дуровими в 2013 році на мові програмування C++. Telegram є кросплатформним і доступний на більшості сучасних операційних систем, таких як Windows, Mac, Linux, Android, Windows Phone, також є веб-версія месенджеру.

Розглянемо найбільш важливі переваги Telegram:

1. Секретні чати.

   Це функція Telegram за допомогою якої користувачі можуть прийняти участь в діалозі, повідомлення в якому мають наскрізне шифрування. Головна особливість даної функції в тому, що вона гарантує конфіденційне спілкування, не дозволяючи жодній із сторін діалогу переслати повідомлення із чату третім особам або зробити скріншот.

2. Таймери самознищення.

   Ця функція призначена для регулярного видалення повідомлень із чату, якщо є така потреба. Коли ви отримуєте повідомлення, воно залишається у вашому

чаті лише на деякий період часу, після чого видаляється. Користувач може налаштувати період зберігання, який може варіюватись від однієї секунди до одного тижня.

3. Можливість відправлення великих файлів.

   Telegram дозволяє користувачам ділитися файлами розміром до 1,5 ГБ, в той час як найпопулярніший на сьогодні месенджер WhatsApp має обмеження в 100 МБ.

4. Групи та канали.

   Telegram надає можливість об’єднувати людей у групи, в яких вони діляться знаннями, обмінюються інформацією тощо. Вони є двох типів – групи та супер групи, в яких можуть знаходитися до 200 та 5000 користувачів відповідно. Канали – це чати призначені для трансляції інформації на велику аудиторію, та на відміну від груп в канал

5. Синхронізація пристроїв.

   Всі повідомлення та діалоги синхронізуються між будь якими пристроями на яких встановлено додатки Telegram.

Отже Telegram, як засіб зв’язку пропонує швидкість, безпеку та різні методи комунікації. Завдяки різноманіттю функцій, які має дана платформа, користувачі не обмежені тільки текстовими повідомленнями як формою спілкування.

В таблиці 1 зображено порівняльну характеристику Telegram з найпопулярнішим месенджером WhatsApp

Таблиця 1 – Порівняльна характеристика Telegram з WhatsApp

| | Telegram | WhatsApp |
|---|---|---|
| Рік запуску | 2013 | 2009 |
| Кількість активних користувачів за місяць | 200 мільйонів | 1.6 трильйони |
| Типи чатів | Приватні<br>секретні | Приватні<br>групові |

| | | |
|---|---|---|
| | групові<br>канали | |
| Канали комунікації | Текстові повідомлення<br>зображення<br>відео,<br>відео-повідомлення,<br>аудіо-повідомлення,<br>стікери,<br>документи<br>GIF<br>голосові виклики<br>відео-виклики | Текстові повідомлення<br>зображення<br>відео,<br>відео-повідомлення,<br>аудіо-повідомлення,<br>стікери,<br>документи<br>GIF<br>голосові виклики<br>відео-виклики |
| Функції обміну повідомленнями | Редагування<br>Видалення<br>закріплення | Видалення |

На рисунку 1 зображено інтерфейс Telegram для Windows 10

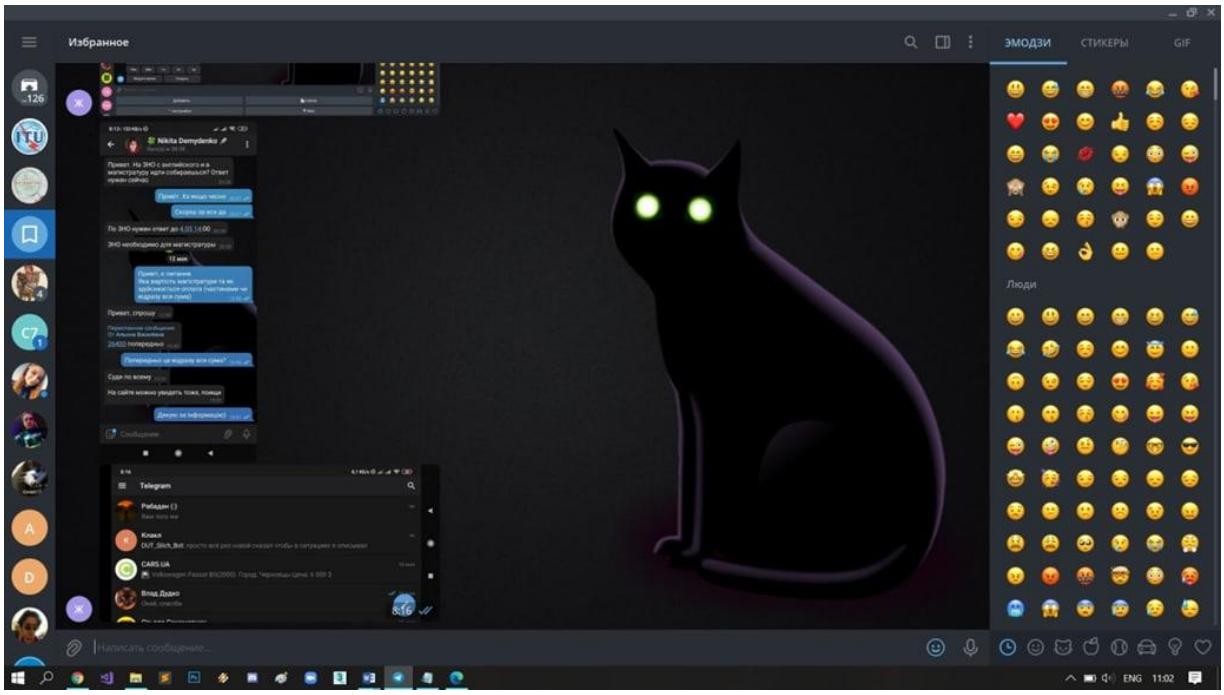


Рисунок 2 – Інтерфейс Telegram для ОС Windows 10

# 3 ВИБІР ТЕХНОЛОГІЙ І СЕРЕДОВИЩА РОЗРОБКИ

## 3.1 Обрані технології розробки

### 3.1.1 Python

Мова програмування Python розробленна Гвідо Ван Россумом в кінці 1980 початку 1990 років в Національному науково-дослідницькому інституті математики та комп'ютерних наук в Нідерландах, є похідною від багатьох інших мов програмування, включаючи C та C++ та командної оболонки Unix. Сьогодні Python підтримується командою розробників ядра в інституті.

Python – це високорівнева інтерпретована мова програмування загального призначення та підтримує такі парадигми програмування як структурне, об'єктно-орієнтоване, функціональне імперативне та аспектно-орієнтоване.

Python націлений на підвищення продуктивності розробника та читабельності коду, має простий та лаконічний синтаксис та є однією з найпростіших у вивченні мов програмування. Читання програми на даній мові нагадує читання тексту на англійській мові. Така псевдо-кодова природа Python є однією з найсильніших його сторін та дозволяє зосередитися саме на вирішенні поставлених завдань, а не на самій мові. Python дає розробникам можливість виконати більший об'єм роботи, затрачуючи при цьому менше зусиль та часу. Код програмного забезпечення написаного на Python є компактним та дозволяє вмістити рішення одних і тих самих завдань в меншу кількість рядків у порівнянні з такими мовами як, наприклад, C++, C# або Java. Розглянемо приклад програми "Hello World" написаної на мові Python (рисунок 3) та C++ (рисунок 4)

```
# This program prints Hello, world!

print('Hello, world!')
```

Рисунок 3 – Код програми «Hello world» на мові Python

```cpp
// Your First C++ Program

#include <iostream>

int main() {
    std::cout << "Hello World!";
    return 0;
}
```

Рисунок 4 – Код програми «Hello world» на мові C++

Python має об’ємну стандартну бібліотеку модулів протестованого коду, які можуть включатися в програми для вирішення різних завдань пов’язаних з регулярними виразами, генерацією документації, тестуванням блоків коду, базами даних, веб-браузерами, CGI та багато іншого. Крім того існує велика кількість бібліотек від сторонніх розробників для вирішення більш складних та специфічних завдань. Також при бажанні мову можна розширити власними бібліотеками.

Python є мультиплатформною мовою. Більшість програм на Python можна запустити без істотних змін на всіх основних операційних системах. Наприклад, перенесення коду Python між платформами Windows та Linux, в більшості випадках, лише питання копіювання коду з одної операційної системи на іншу.

Python активно використовують для реалізації своїх проектів такі гіганти сфери IT як Google, Microsoft, Yandex, Intel та інші. Такі успішні сервіси як пошукова система Google, відео хостинг YouTube та соціальні мережі Facebook та Instagram, відомі кожному користувачу мережі інтернет, були розроблені з використанням Python.

### 3.1.2 Asyncio та Aiogram

Asyncio – це стандартна бібліотека Python, призначена для написання паралельно виконуваного мережевого вводу-виводу з використанням синтаксису async / await. Бібліотека дозволяє розробнику писати код, який виглядає як

синхронний, але працює асинхронно. Використовується як основа для деяких асинхронних фреймворків, в тому числі aiogram та asyncpg.

Асинхронне програмування дає можливість додаткам виконувати відразу декілька операцій в конкретний проміжок часу, тоді як використання синхронного підходу передбачає послідовний перехід від однієї операції до іншої після завершення роботи пепередньої. Таким чином асинхронний підхід дозволяє значно підвищити швидкість виконання коду.

В розробці застосунку було використано фреймворк Aiogram для роботи з Telegram Bot API, написаний на мові програмування Python 3.7 на основі asyncio та aiohttp.

### 3.1.3 PostgreSQL

PostgreSQL – це потужна об'єктно-реляційна система керування базами даних з відкритим кодом, яка використовує та розширює мову SQL в поєднанні з великою кількістю функцій, які безпечно зберігають та масштабують самі складні робочі навантаження з даними.

PostgreSQL має більш ніж 30-річну історію розробки. Система керування об’єктно реляційними базами даних виникла в рамках проекту POSTGRES Каліфорнійського університету в Берклі. Він був розпочатий ще в 1986 році під керівництвом Майкла Стоунбрейкера та фінансувався Агентством перспективних оборонних досліджень та Національним науковим фондом. В 1994 році студенти Ендрю Ю та Джолі Чен додали інтерпретатор SQL до базового коду, після чого ця нова, приблизно на 30-50% швидша, модифікація отримала назву Postgres95. Через два роки, з виходом версії 6.0 бази даних, вона отримала свою сучасну назву PostgreSQL.

Гнучкість PostgreSQL очевидна не тільки в області функціональності, розширюваності та адаптованості. База даних також забезпечує велику свободу дій в налаштуванні програмного та апаратного забезпечення. PostgreSQL включений до більшості дистрибутивів UNIX/Linux та поставляється Apple в якості стандартної бази даних, починаючи з Mac OS X Lion. Операційні системи Windows

також можуть бути обрані в якості платформи для системи завдяки відповідним установочним пакетам. Вимоги до обчислювальної потужності та ємкості сховища залежать тільки від розміру запланованої системи баз даних. Для самого програмного забезпечення потрібно не більше 20 МБ. В таблиці 2 продемонстровано ключові характеристики бази даних.

Таблиця 2 – Ключові характеристики PostgreSQL

| | |
|---|---|
| Максимальний розмір бази даних | Необмежений |
| Максимальний розмір таблиці | 32 терабайта |
| Максимальний розмір запису даних | 1.6 терабайт |
| Максимальний розмір поля | 1 гігабайт |
| Максимальна кількість стовпців | Від 250 до 1600 в залежності від типу полів |
| Максимальна кількість рядків | Необмежена |
| Максимальна кількість індексів | Необмежена |

PostgreSQL засновано на типовій моделі клієнт-сервер – компонент центрального серверу під назвою postmaster керує всіма файлами бази даних та усіма зв’язками, які встановлюються для зв’язку з сервером бази даних. Користувачам потрібна тільки підходяща клієнтська програма для встановлення з’єднання. При цьому програмний пакет PostgreSQL з psql уже має власні рішення для роботи через командний рядок або інтегрований термінал.

Asyncpg – це повнофункціональна клієнтська бібліотека Python з відкритим кодом для роботи з PostgreSQL. Використовується разом з фреймворком asyncio.

В порівнянні з іншими бібліотеками для інтеграції з PostgreSQL, asyncpg показує найвищі результати швидкодії (рисунок 5)

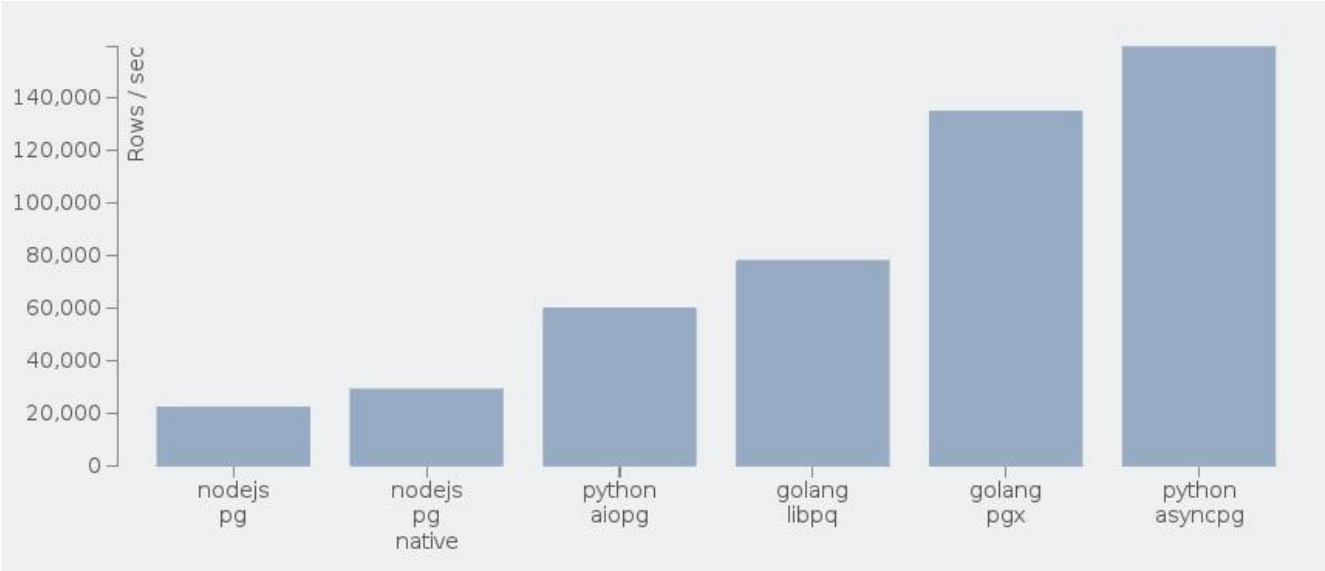


Рисунок 5 – Порівняння швидкодії бібліотек для роботи з PostgreSQL

### 3.1.4 Бібліотека Requests та HTTP запити

HTTP – це протокол передачі даних, призначений для передачі гіпертекстових документів в мережі. HTTP працює по принципу клієнт-сервер. Основним його застосуванням є передача веб-сторінок та даних між веб-сервісом та браузером. Для отримання доступу до сайту клієнт відправляє запит на сервер. За допомогою URL-адреси клієнт звертається до файлу на сервері який повинен бути відправлений йому. Потім запит обробляється і, у відповідь, сервер відправляє дані та код стану, який показує результат запиту. Існує 5 класів коду стану:

1. 100-199 (інформаційні коди стану) – інформують про те що запит ще знаходиться в обробці
2. 200-299 (успішно) – інформують про те що запит успішно прийнято та оброблено
3. 300-399 (перенаправлення) – інформують про те що потрібні додаткові дії для завершення обробки запиту
4. 400-499 (помилки з боку клієнта) – інформують про те що сталася помилка з боку клієнта
5. 500-599 (помилки з боку сервера) – інформують про те що сталася помилка з боку сервера

Бібліотека requests є стандартом для створення HTTP запитів в Python. Вона

абстрагується від складності виконання запитів за красивим та простим API. Це дозволяє зосередити свою увагу на взаємодії зі службами та використанні даних в розроблюваних додатках.

В таблиці 3 зображено методи бібліотеки requests.

Таблиця 3 – Методи бібліотеки requests

| | |
|---|---|
| get(url, params, args) | Надсилає GET запит на визначений url |
| post(url, data, json, args) | Надсилає POST запит на визначений url |
| put(method, url, args) | Надсилає PUT запит на визначений url |
| patch(url, data, args) | Надсилає PATCH запит на визначений url |
| head(url, args) | Надсилає HEAD запит на визначений url |
| delete(url, args) | Надсилає DELETE запит на визначений url |

В процесі розробки дипломного проекту було застосовано 2 типи запитів.

Один з найбільш поширених методів HTTP – це метод GET запиту. Він використовується для отримання даних ресурсу, наприклад інформації з веб-сайту з HTTP серверу. На рисунку 6 зображено схему роботи GET запиту.

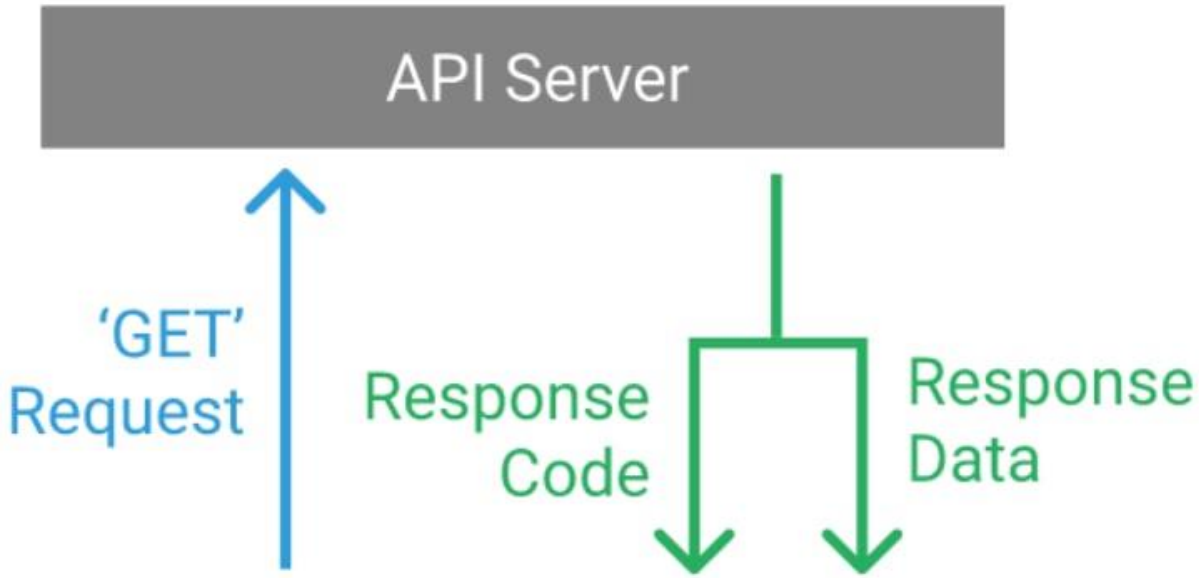


Рисунок 6 – Схема роботи запиту GET

Метод POST використовується для оновлення або відправлення даних у визначеній формі на сервер. Наприклад заповнення форми авторизації на сайті. На рисунку 7 зображено схему роботи POST запиту.

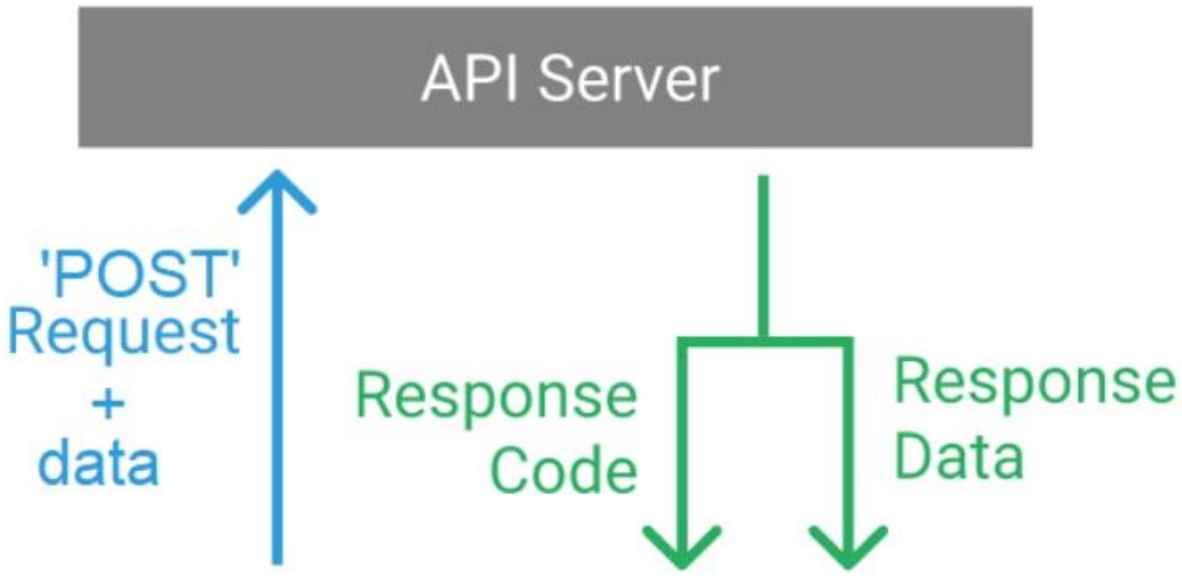


Рисунок 7 – Схема роботи запиту POST

### 3.1.5 Модулі Datetime та Calendar

Datetime – це модуль в Python який надає класи для роботи з датами та часом. В таблиці 4 наведено класи бібліотеки datetime.

Таблиця 4 – Класи бібліотеки datetime

| | |
|---|---|
| date | Данні про дату на основі григоріанського календаря |
| time | Данні про час |
| datetime | Данні про час та дату на основі григоріанського календаря |
| timedelta | Описує різницю між двома різними періодами часу |
| tzinfo | Надає інформацію про часові пояси |
| timezone | Описує час на основі стандарту UTC |

Модуль Calendar в Python дозволяє виконувати обчислення для різних завдань пов’язаних з датою та часом, використовуючи дату місяць та рік. Також

класи TextCalendar та HTMLCalendar дозволяють створювати і редагувати календар та використовувати його у відповідності з поставленими вимогами

Функції та атрибути модуля Calendar:

- calendar.setfirstweekday() – встановлює початок тижня
- calendar.firstweekday() – повертає перший день тижня
- calendar.isleap() – повертає True, якщо рік є високосним
- calendar.leapdays() – повертає кількість високосних років в певному діапазоні
- calendar.weekday() – повертає день тижня по даті
- calendar.monthrange() – повертає перший робочий день указаного місяця
- calendar.monthcalendar() – повертає календар на указаний місяць у вигляді матриці
- calendar.prmonth() – друкує в інтерпретатор календар на вказаний місяць
- calendar.month() – повертає календар на місяць для збереження в файл
- calendar.prcal() – друкує в інтерпретатор календар на рік
- calendar.calendar() – повертає календар на рік для збереження в файл
- calendar.timegm() – повертає секунди із структури часу
- calendar.day_name() – ітератор назв днів тижня
- calendar.day_abbr() – ітератор скорочених назв днів
- calendar.month_name() – ітератор місяців року
- calendar.month_abbr() – ітератор скорочених назв місяців року

### 3.1.7 Бібліотека Beautiful Soup 4

Beautiful Soup – це бібліотека мови програмування Python, яка призначена для зчитування, аналізу та подальшої обробки HTML та XML документів. Вона застосовується для парсингу, забезпечуючи методи навігації пошуку та зміни дерева синтаксичного аналізу.

## 3.2 Середовище розробки

### 3.2.1 Sublime Text Editor

Sublime Text Editor – це повнофункціональний текстовий редактор для редагування локальних файлів та бази коду, розроблений в 2008 році з використанням C++ та Python. Він включає в себе різні функції для редагування бази коду. Sublime має такі функції:

- Підсвічування синтаксису
- Міні карта коду
- Мультипанелі
- Автоматичні відступи
- Розпізнання типів файлів
- Бокова панель з файлами указаного каталогу
- Макроси
- Плагіни та пакети
- Зв’язок з системами контролю версій Git та Mercurial

Редактор використовується як інтегроване середовище розробки. Починаючи з версії 2 доступний на платформах Windows, MacOS та Linux. На рисунку 8 зображено інтерфейс редактору.

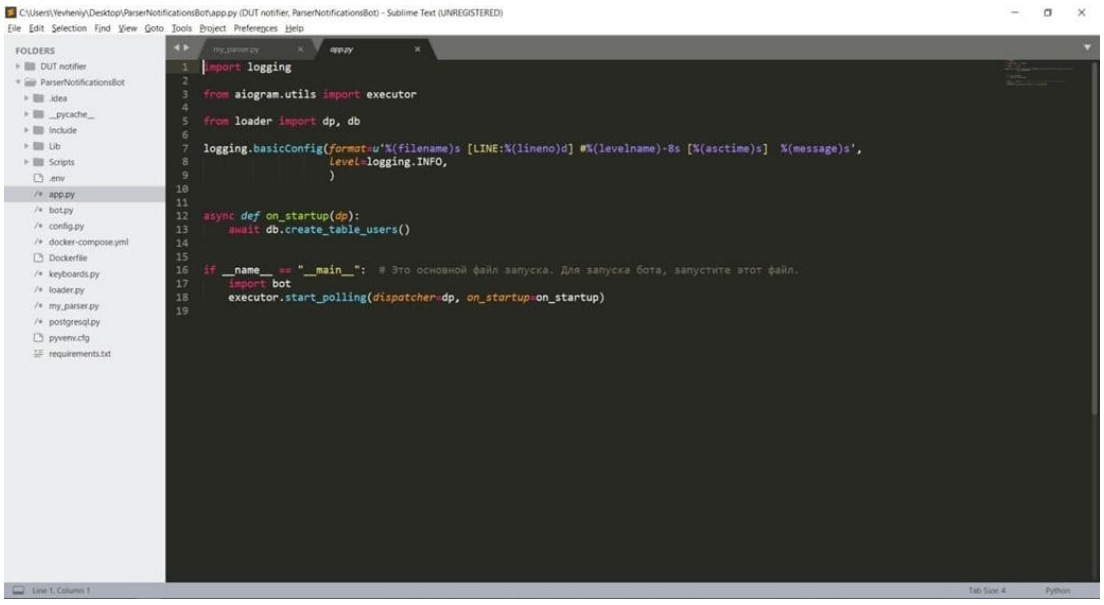


Рисунок 8 – Інтерфейс Sublime Text Editor

# 4 РОЗРОБКА ЧАТ-БОТА

## 4.1 Реєстрація чат-бота в Telegram

Щоб створити нового чат-бота в Telegram, спочатку необхідно знайти в месенджері спеціального бота під назвою @BotFather, який призначений для реєстрації нових чат-ботів та керування вже створеними.

Після запуску BotFather, потрібно ввести команду /newbot, після чого буде запропоновано ввести назву та username нового бота. Назва бота може бути будь якою, але username повинен бути унікальним, та обов'язково закінчуватись на слово «bot». Після завершення цієї операції бот згенерує та відправить особистий token, який є ключем для з'єднання з HTTP API Telegram. На рисунку 9 зображений процес реєстрації та отримання токену.

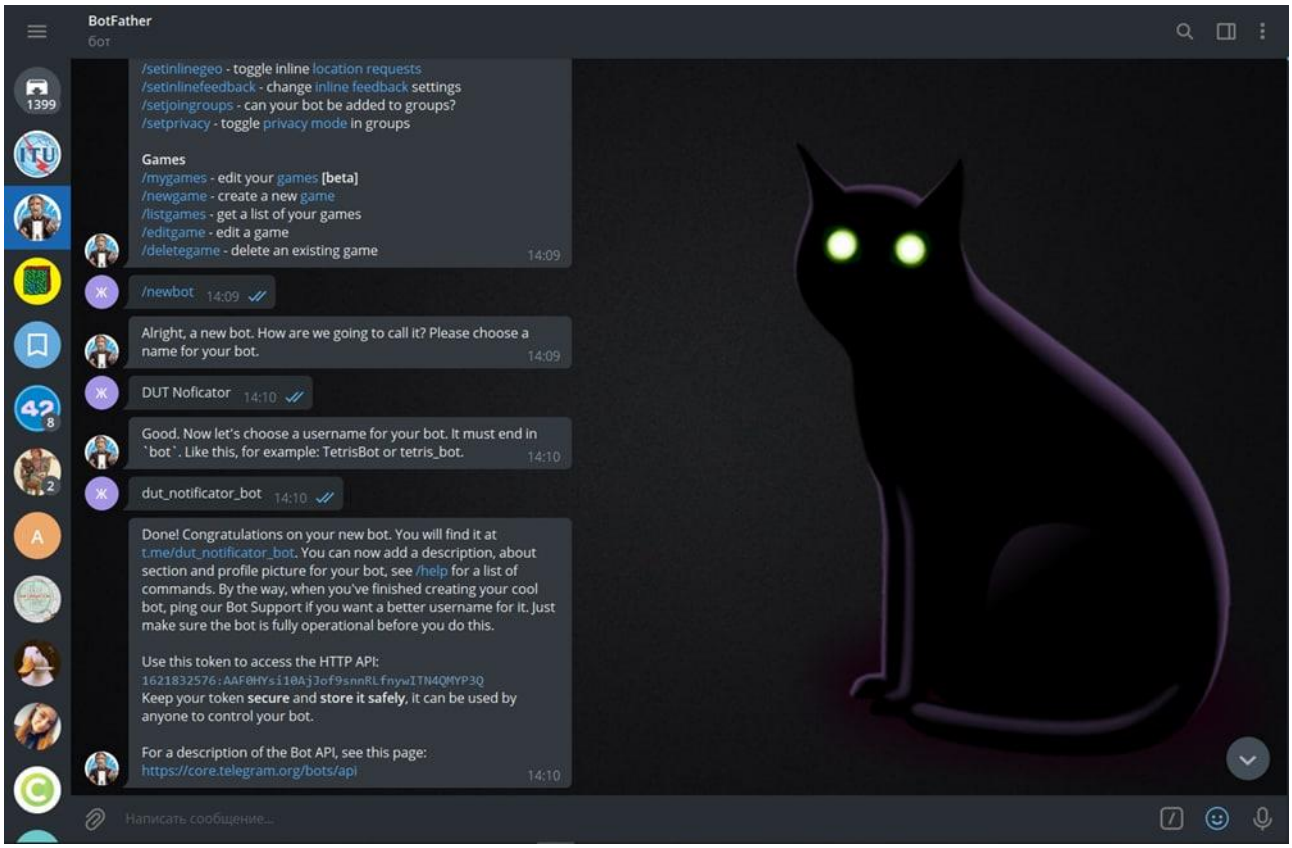


Рисунок 9 – Реєстрація чат-бота та отримання токену

BotFather має дві основні команди - /newbot та /mybots. Команда /newbot

дозволяє створювати нових ботів, а /mybots дає можливість встановлювати ти змінювати налаштування зареєстрованих ботів.

В таблиці 5 наведено команди для редагування основних параметрів чат-ботів.

Таблиця 5 – Команди для редагування основних параметрів чат-бота

| | |
|---|---|
| /setname | Команда для редагування назви чат-бота |
| /setdescription | Команда для редагування опису чат-бота |
| /setabouttext | Команда для редагування інформації в профілі чат-бота |
| /setuserpic | Команда для встановлення фотографії профілю чат-бота |
| /setcommands | Команда відповідає за створення команд чат-бота |
| /deletebot | Команда для видалення чат-бота |

В таблиці 6 наведено команди для встановлення додаткових налаштувань для чат-ботів.

Таблиця 6 – Команди для встановлення додаткових налаштувань чат-бота

| | |
|---|---|
| /token | Повертає токен обраного чат-бота |
| /revoke | Анулює токен обраного чат-бота |
| /setinline | Вмикає або вимикає виклик ботів з інших чатів |
| /setinlinegeo | Вмикає або вимикає передачу місцезнаходження бота |
| /setinlinefeedback | Дозволяє отримувати інформацію про кількість обраних користувачами команд |
| /setjoingroups | Вмикає або вимикає можливість додавання ботів в інші чати |
| /setprivacy | Вмикає або вимикає режим конфіденційності |

Після встановлення всіх необхідних налаштувань чат-бота в BotFather, переходимо до програмної реалізації.

### 4.2 Реалізація основних функцій чат-бота

Запуск чат-бота починається з відправлення користувачем повідомлення з командою /start.

Після цього повідомлення повинне пройти обробку. В Aiogram обробка повідомлень реалізована за допомогою message handlers, які представляють собою набір фільтрів для вхідних повідомлень. Якщо повідомлення проходить фільтр, то викликається функція для виконання певного завдання. На рисунку 10 зображено message handler, який оброблює команду /start

```
13  @dp.message_handler(CommandStart())
14  async def start_handler(message: types.Message):
15      """Ответ на команду start"""
16      await message.answer(
17          """DUT notification - це бот для інформування с
18          1. Авторизуйтесь в особистому кабінеті системи
19          2. Оберіть один з пунктів меню: ''Звіт оцінок з
20          3. В меню ''Звіт оцінок з дисциплін'' ви можете
21          4. В меню ''Налаштування'' ви можете ввімкнути
22          """,
23          reply_markup=auth_keyboard)
24
```

Рисунок 10 – Message handler команди /start

Після обробки, запускається функція start_handler, яка надсилає користувачу повідомлення з інформацією про доступні функції та інструкцію використання (рисунок 11).

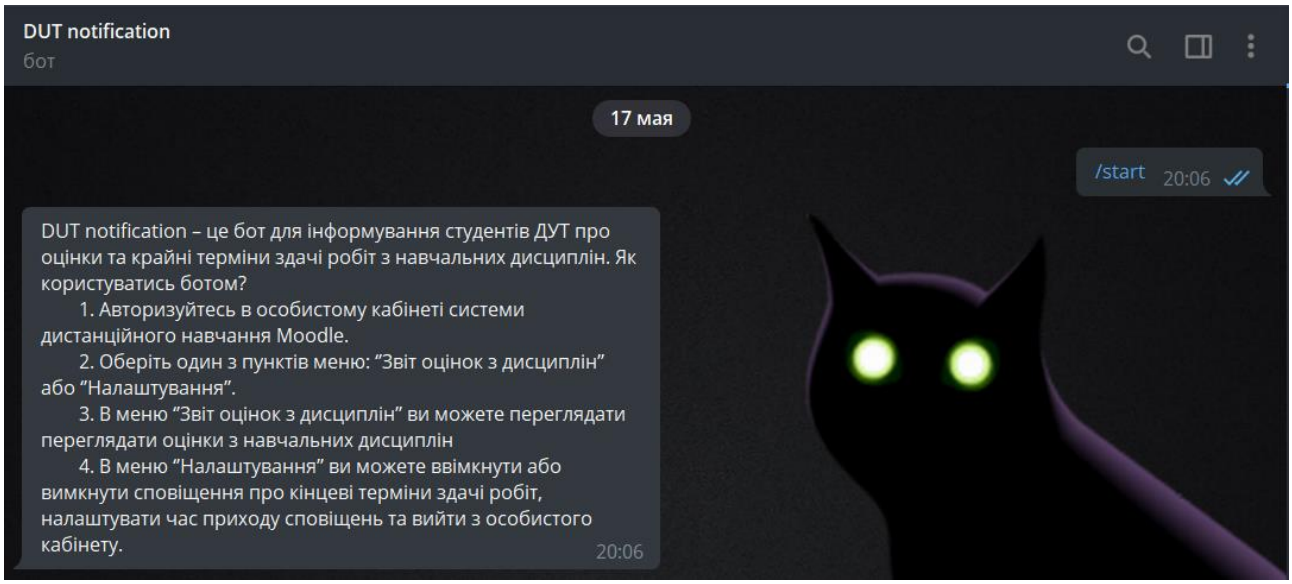


Рисунок 11 – Відповідь бота на команду /start

Після того як користувач отримує відповідь на команду /start, він потрапляє в меню авторизації. Для реалізації авторизації на сайті дистанційного навчання Moodle через інтерфейс чат-боту було використано бібліотеку для створення HTTP запитів requests.

В меню авторизації є клавіша «Авторизація», реалізована за допомогою клавіатури Telegram. Після натискання на неї запуститься функція «get_login», яка відповідає за введення користувачем логіну від особистого кабінету. Вслід за нею йде функція «get_password», яка приймає пароль від особистого кабінету користувача. Після введення даних запускається функція «authorization», яка авторизує користувача в особистому кабінеті (рисунок 12).

```
41 @dp.message_handler(state="get_password")
42 async def authorization(message: types.Message, state: FSMContext):
43     """Результат авторизации"""
44     data = await state.get_data()  # Достаем сохраненные данные в машине состояний
45     login = data["login"]  # Достаем логин из данных
46     id = await db.count_users() + 1  # Получаем количество пользователей в базе данных (позже передадим его в id)
47     password = message.text
48     pib = await auth_user(login,
49                           password)  # Здесь мы вызываем нашу фукнцию авторизации на сайте. Так как я сделал функцию асинхронно
50     if not pib == "ВХІД":  # Если пользователь ввел верные данные, фукнция auth_user вернет ФИО пользователя, и бот входит в ли
51         check_user = await db.select_user(user_id=message.from_user.id,
52                                           name=pib)  # Проверяем, есть ли пользователь в базе данных, если нет, регистрируем ег
53         if check_user == None:
54             await db.add_user(id=id, user_id=message.from_user.id, name=pib)
55             await state.update_data(name=pib)
56             await message.answer(f"Вітаю, {pib}, ви супішно авторизовані до вашого особистого кабінету",
57                                  reply_markup=main_keyboard)
58             await state.update_data(password=password)  # Сохраняем пароль в машине состояний (пригодится в дальнейшем)
59             await state.set_state("authorization")  # Задаем состояние авторизированного пользователя
60         else:
61             await state.update_data(name=pib)
62             await message.answer(f"Вітаю, {pib}, ви успішно авторизовані", reply_markup=main_keyboard)
63             await state.update_data(password=password)  # Сохраняем пароль в машине состояний (пригодится в дальнейшем)
64             await state.set_state("authorization")  # Задаем состояние авторизированного пользователя
65     else:  # Если пользователь ввел неверные данные, функция вернет "ВХІД", и бот отвечает пользователю, что данные не верны
66         await message.answer("Невірний логін або пароль, спробуйте ще раз", reply_markup=auth_keyboard)
67         await state.reset_state()  # Сбрасываем состояние
```

Рисунок 12 – Функція authorization

В тілі даної функції в рядку 48 викликається функція auth_user (рисунок 13), яка відповідає за надсилання POST запиту з отриманими від користувача логіном та паролем на сервер та отримання адреси головної сторінки сайту та ПІБ авторизованого користувача з особистого кабінету, якщо користувач ввів вірні дані. Далі функція перевіряє чи зареєстровано користувача в базі даних. Якщо ні вона реєструє його. У випадку, якщо користувач ввів невірні дані, функція поверне «ВХІД» та запропонує повторно ввести логін та пароль.

```
15 async def auth_user(username,
16                       password):  # вместо def auth_user я прописал asy
17     session = requests.Session()
18
19     data = {
20         'username': username,
21         'password': password
22     }
23
24     session.post(LOGIN_URL, data=data, headers=HEADERS)
25     response = session.get(LOGIN_URL, headers=HEADERS)
26     soup = BeautifulSoup(response.content, 'html.parser')
27     pib = soup.html.body.div.div.div.div.a.get_text()
28     return pib
29
```

Рисунок 13 – Функція auth_user

На рисунку 14 зображено процес авторизації користувача в чат-боті.

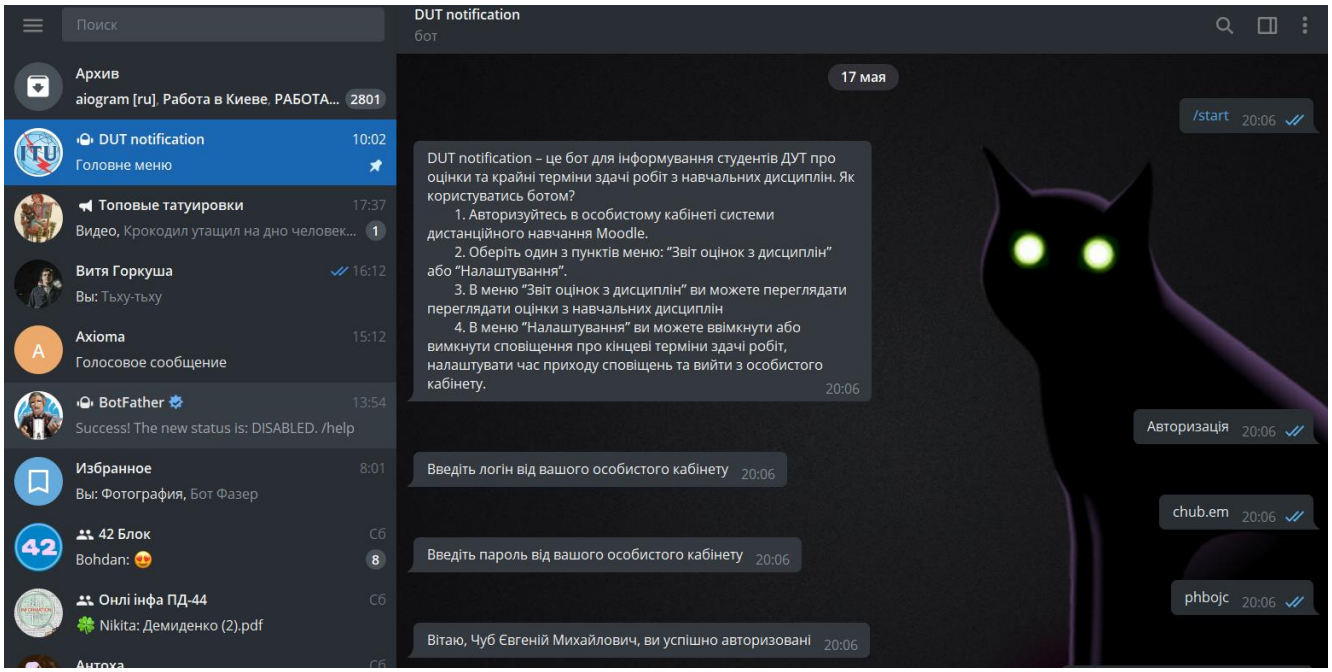


Рисунок 14 – Процес авторизації в чат-боті

Для зберігання даних про користувачів було прийнято рішення використати базу даних PostgreSQL. База має одну таблицю Users, яка зберігає інформацію про всіх користувачів, які зареєстровані в чат-боті та дані про налаштування сповіщень. Підключення до бази даних та створення таблиці Users зображено на рисунку 15.

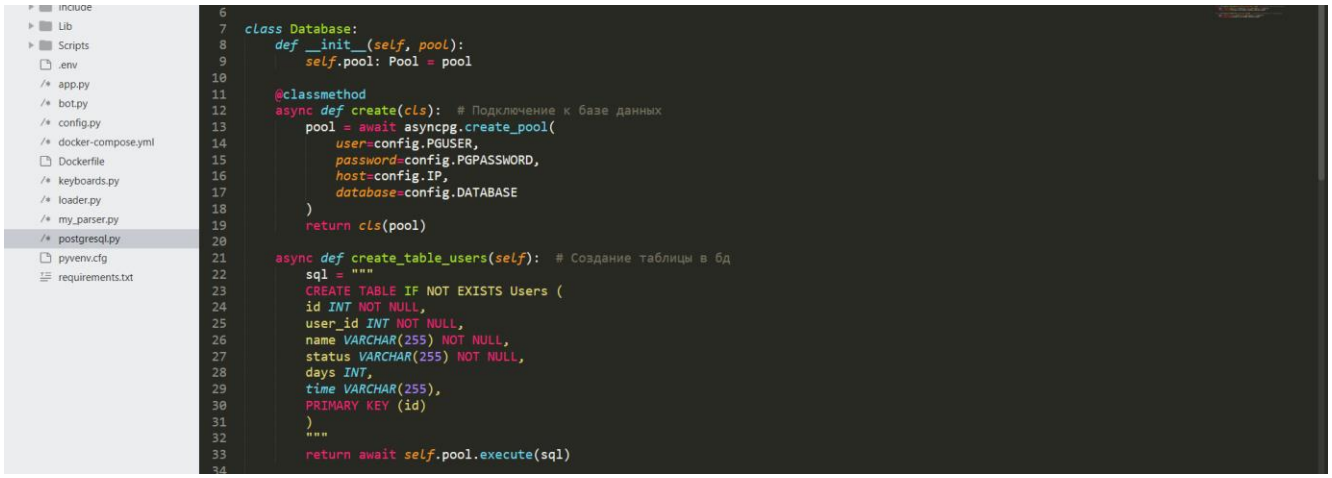


Рисунок 15 – Підключення та створення таблиці Users в базі даних

На рисунку 16 зображено структуру бази даних чат-бота.

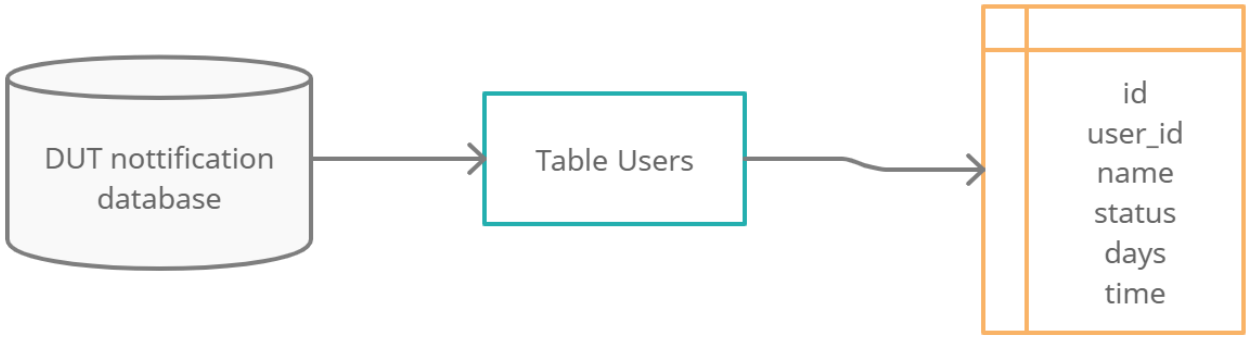


Рисунок 16 – Діаграма структури бази даних

Поля id, user_id, name – зберігають інформацію про користувачів, а status, days та time – інформацію про налаштування сповіщень.

На наступному розробки етапі потрібно здійснити реалізацію функцій для збору даних про навчальні дисципліни, роботи та оцінки з HTML сторінок сайту дистанційного навчання Moodle. Для цієї мети було використано технологію парсингу, яка реалізована в додатку за допомогою бібліотеки Python – Beautiful Soup, та бібліотеки для створення HTTP запитів – requests.

Було створено 3 функції для парсингу потрібних даних, які працюють по одному і тому ж принципу. Спочатку вони надсилають GET запит на HTML адресу, з якої потрібно дістати інформацію. Після цього HTML документ перетворюється в об'єкт Beautiful Soup. Потім за допомогою методів бібліотеки Beautiful Soup здійснюється пошук тегів по вкладеній структурі HTML сторінок та зберігається цільова інформація, яка знаходиться всередині них.

Список функцій та їх призначення:

1. disciplines - повертає список дисциплін та посилань на них
2. get_disc_info - повертає посилання на сторінку з роботами та оцінками
3. get_marks - повертає список відсортованих завдань та оцінок

Перша функція надалі буде використовуватись в функції disc_keyboard, призначеної для створення клавіатури Telegram, в клавішах якої будуть

відображені назви дисциплін. В свою чергу функція disc_keyboard буде викликатись в тілі функції choose_disc (рисунок 17), яка відповідає за вибір потрібної дисципліни.

```python
@dp.message_handler(state="authorization", text="Звіт оцінок з дисциплін")
async def choose_disc(message: types.Message, state: FSMContext):
    """Ответ на нажатие кнопки "Звіт оцінок з дисциплін"."""
    data = await state.get_data()  # Достаем данные из машины состояний
    login = data.get("login")  # Достаем логин из данных
    password = data.get("password")  # Достаем пароль из данных
    d_keyboard, discs, urls = await disc_keyboard(login, password)  # Вызываем функцию создания клавиатуры дисциплин
    await message.answer("Оберіть навчальну дисципліну", reply_markup=d_keyboard)  # Передаем сюда созданную клавиатуру
    await state.update_data(discs=discs, urls=urls)  # Сохраняем название дисциплины и список ссылок на все дисциплины
    await state.set_state("choose_disc")
```

Рисунок 17 – функція choose_disc

Наступні дві функції будуть викликані в тілі функції show_disc_list (рисунок 18), яка призначена для відправлення користувачу оцінок за роботи з навчальних дисциплін

```python
@dp.message_handler(state="choose_disc")
async def show_disc_list(message: types.Message, state: FSMContext):
    """Ответ на выбор дисциплины"."""
    disc = message.text
    data = await state.get_data()  # Достаем данные из машины состояний
    login = data.get("login")  # Достаем логин
    password = data.get("password")  # Достаем пароль
    discs = data.get("discs")  # Достаем название дисциплины
    urls = data.get("urls")  # Достаем список ссылок на дисциплины
    disc_index = discs.index(disc)  # Находим индекс нужной нам ссылки в списке ссылок
    url = urls[disc_index]  # Берем нужную нам ссылку
    marks_url = await get_disc_info(login, password, url)  # Получаем ссылку на страницу дисциплины
    if marks_url == False:  # Если, вдруг, на странице дисциплины нет пункта "Оценки", выводим "Немає робіт"
        await message.answer("Немає робіт")
    else:
        tasks = ""
        tasks_list = await get_marks(login, password, marks_url)  # Парсим оценки
        for num, task in enumerate(tasks_list):
            tasks += f"{num}. {task}\n"
        await message.answer(tasks)  # Отправляем пользователю оценки
```

Рисунок 18 – функція show_disc_list

В Telegram процес взаємодії користувача з меню «Звіт оцінок з дисциплін» чат-бота зображено на рисунку 19.

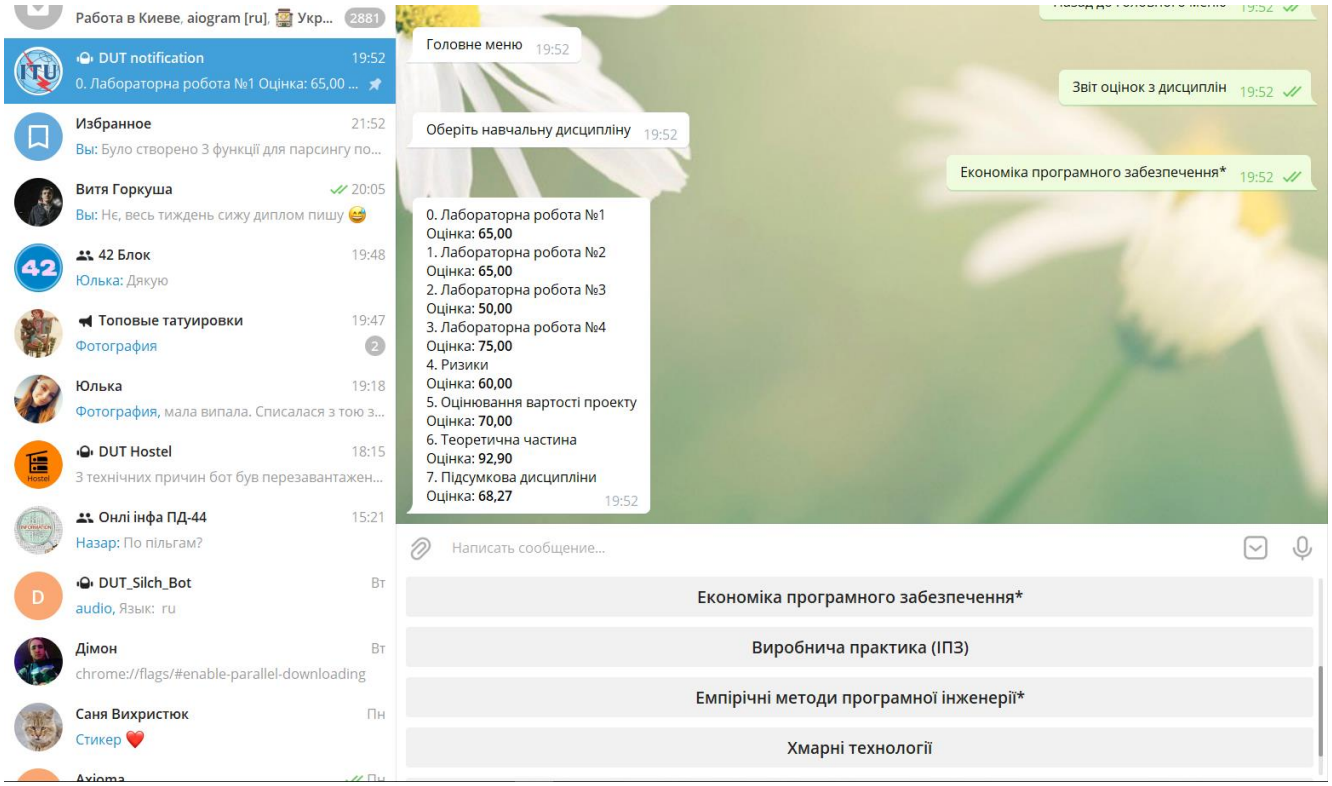


Рисунок 19 – Взаємодія користувача з меню «Звіт оцінок з дисциплін»

На даному етапі розробки чат-боту потрібно реалізувати сповіщення про крайні терміни здачі робіт з навчальних дисциплін. Для цієї мети були використано бібліотеки для роботи з датою та часом datetime та calendar, бібліотеку для створення HTTP запитів – requests, а також бібліотеку для парсингу даних з HTML документів Beautiful Soup.

Коли користувач переходить до меню налаштувань сповіщень, запускається функція get_days, яка приймає від користувача кількість днів, за збігом яких будуть надсилатись сповіщення. Слідом за нею починає роботу функція get_time, яка приймає годину доби, в яку приходять сповіщення. Після встановлення кількості днів та часу, спрацьовує функція set_not_time (рисунок 20), яка приймає налаштування та записує їх до бази даних.

```python
@dp.message_handler(state="get_time")
async def set_not_time(message: types.Message, state: FSMContext):
    """Устанавливает время оповещаний"""
    await message.answer("Дякую, налаштування прийняті", reply_markup=not_options_keyboard)
    data = await state.get_data()
    days = int(data.get("days"))
    time = message.text
    user_id = message.from_user.id
    name = data.get("name")
    await db.update_time(days, time, user_id, name)
    await state.set_state("notifications_option")
```

Рисунок 20 – Функція set_not_time

Функція get_url використовує методи бібліотеки Beautiful Soup для парсингу завдань та кінцевих термінів здачі робіт. Далі функція checker приймає дані про роботи та шукає дату дедлайну. Після цього йде порівняння сьогоднішньої дати з датою останнього терміну та у відповідності з встановленими налаштуваннями визначається, коли повинне бути надіслане сповіщення.

Функція notifications призначена для перевірки статусу сповіщень. Якщо повідомлення ввімкнені, то спрацьовують функції get_url та checker.

Для встановлення статусу сповіщень використовуються функції notifications_on та notifications_off (рисунок 21), в тілі яких викликається функція notifications.

```python
@dp.message_handler(state="notifications_option", text="Ввімкнути сповіщення")
async def notification_on(message: types.Message, state: FSMContext):
    data = await state.get_data()
    user_id = message.from_user.id
    name = data.get("name")
    status = "on"
    login = data["login"]
    password = data["password"]
    await notifications(user_id=message.from_user.id, name=name, status=status, username=login, password=password)


@dp.message_handler(state="notifications_option", text="Вимкнути сповіщення")
async def notification_off(message: types.Message, state: FSMContext):
    data = await state.get_data("name")
    user_id = message.from_user.id
    name = data.get("name")
    status = "off"
    login = data["login"]
    password = data["password"]
    await notifications(user_id=message.from_user.id, name=name, status=status, username=login, password=password)

```

Рисунок 21 – Функції notifications_on та notifications_off

На рисунку 22 зображений процес встановлення налаштувань, та отримання сповіщень в чат-боті

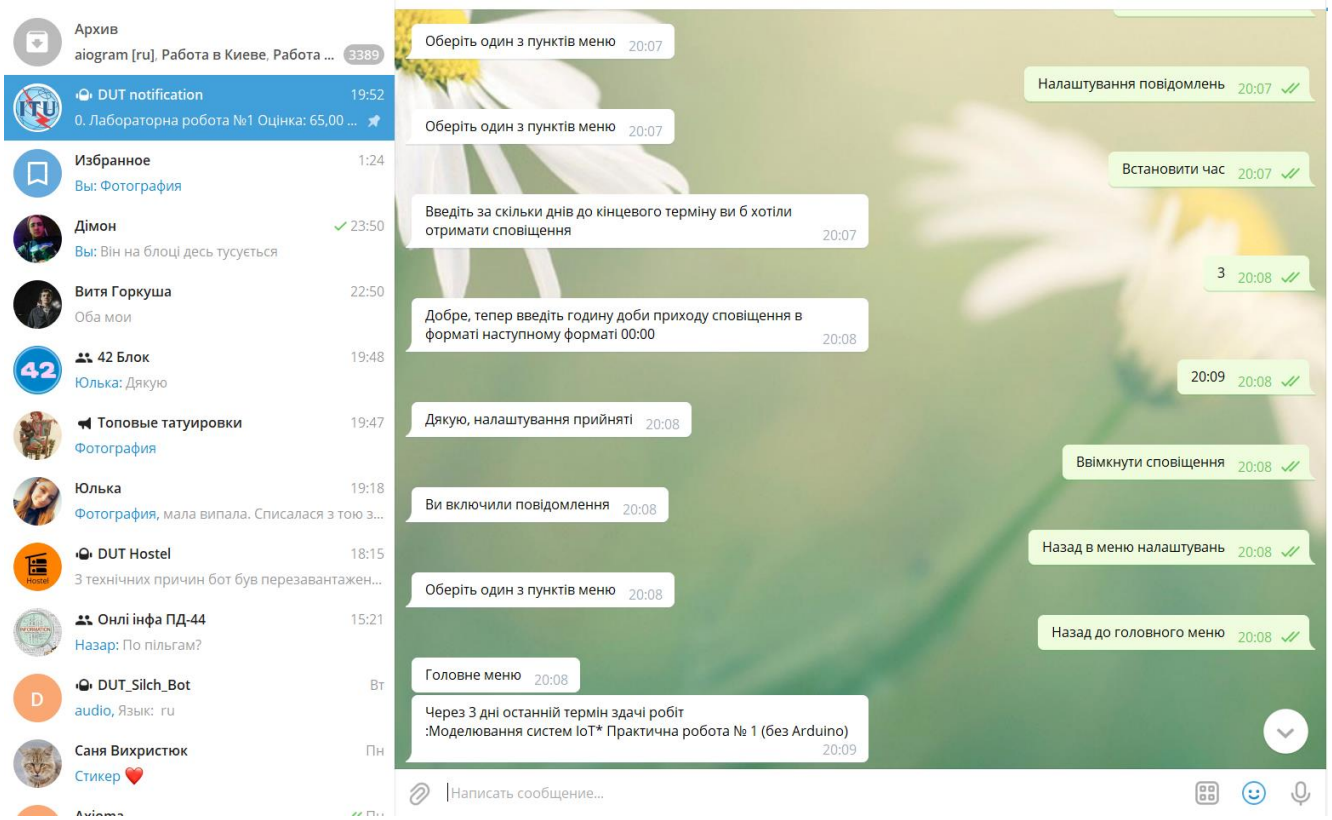


Рисунок 22 – Процес встановлення налаштувань та отримання сповіщень

Для постійної та стабільної роботи, було прийнято рішення розмістити чат-бота на сервері «Макхост».

VPS (Virtual Private Server) – це віртуальний виділений сервер, який розміщено на окремому фізичному пристрої. Має root доступ до системи, який забезпечує можливість встановлення будь-яких програм та зміну налаштувань серверу.

На рисунку 23 зображено використання ресурсів серверу чат-ботом.

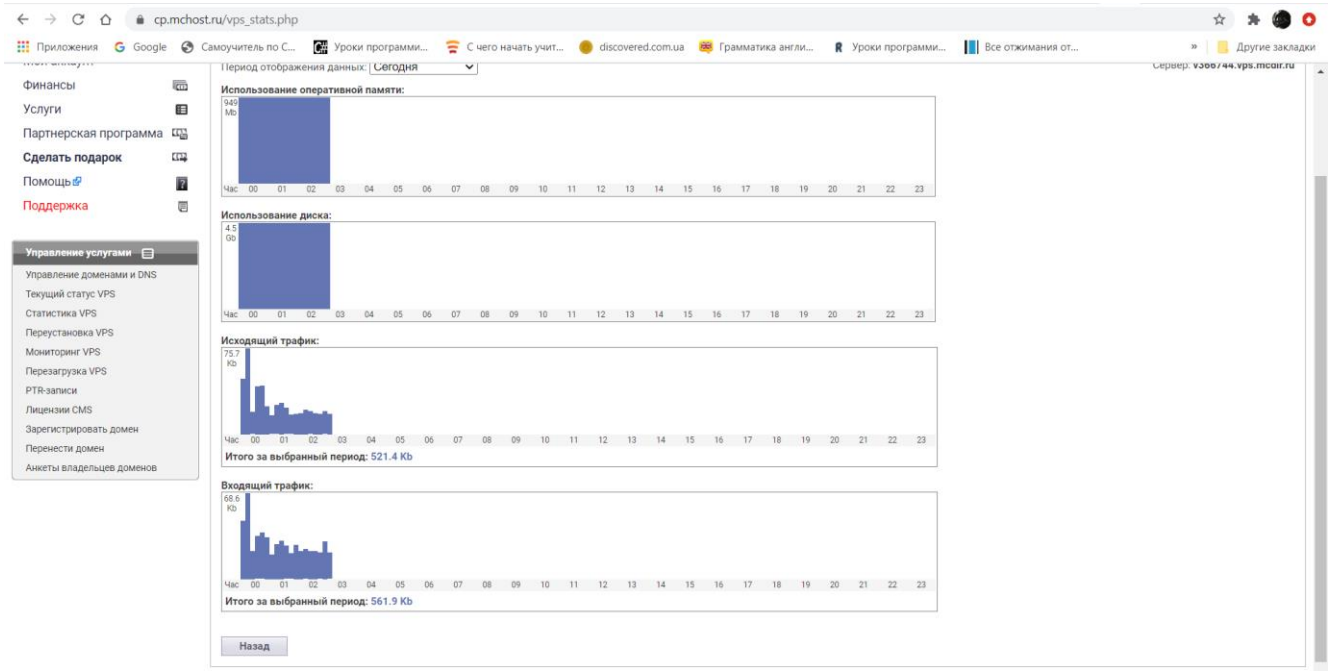


Рисунок 23 – Використання чат-ботом ресурсів серверу

### 4.3 Тестування та алгоритм роботи додатку

Під час тестування було перевірено на працездатність усі основні функції розроблюваного чат-боту.

Реалізовано меню авторизації користувача на сайті дистанційного навчання через інтерфейс чат-боту. Після запуску бота командою /start, користувач потрапляє в меню авторизації, в якому є клавіатура з клавішею «Авторизація». Після натискання на неї, користувачу буде запропоновано ввести логін та пароль від особистого кабінету системи дистанційного навчання. Якщо введені дані вірні, то користувач авторизується та потрапляє до головного меню чат-бота, якщо ж ні то бот запропонує повторно ввести дані та поверне користувача до першого етапу авторизації. Під час тестування меню було виконано вхід до особистого кабінету з 20 різних акаунтів. В усіх випадках помилок виявлено не було та всі необхідні функції спрацювали коректно. Було виключено всі можливі варіанти поведінки користувача при яких могла б статися помилка.

Головне меню представлене у вигляді клавіатури Telegram, з пунктами «Звіт оцінок з дисциплін» та «Налаштування».

Меню «Звіт оцінок з дисциплін» представлено у вигляді клавіатури телеграм

з клавішами дисциплін авторизованого студента та клавіші «Назад до головного меню». Після вибору потрібної дисципліни виводиться повідомлення з назвами робіт та оцінками за них. Натиснувши на клавішу «Назад до головного меню», бот повертає користувача до головного меню чат-бота. В процесі тестування головного меню було виявлено та усунуто одну помилку, а саме – бот не повертав користувача до головного меню при натисканні відповідної клавіші.

Меню «Налаштування» представлене у вигляді клавіатури з пунктами «Налаштування сповіщень» та «Вийти з особистого кабінету».

При виборі другого пункту меню «Вийти з особистого кабінету» користувач повертається на етап авторизації.

Обравши пункт «Налаштування сповіщень» користувач потрапляє до меню налаштувань сповіщень, яке представлене п’ятьма підпунктами, реалізованими за допомогою клавіатури Telegram: «Назад до головного меню», «Назад до меню налаштувань», «Встановити час», «Ввімкнути сповіщення» та «Вимкнути сповіщення».

Перші два пункти призначені для повернення до головного меню та меню налаштувань відповідно.

Два останні встановлюють статус сповіщень, тобто ввімкнені вони або вимкнені.

При виборі пункту «Встановити час» користувачу спочатку пропонується ввести за скільки днів до останнього терміну здачі робіт повинне прийти сповіщення, а після цього годину доби о котрій користувач бажає його отримати. В результаті налаштування приймаються і, в залежності від встановленого статусу, приходять сповіщення у відповідності з заданими налаштуваннями.

Провівши тестування меню налаштувань, було виявлено що всі розроблені функції коректно виконують свої завдання та працюють без помилок.

На рисунку 24 зображено діаграму, яка показує всі аспекти взаємодії користувача з чат-ботом.

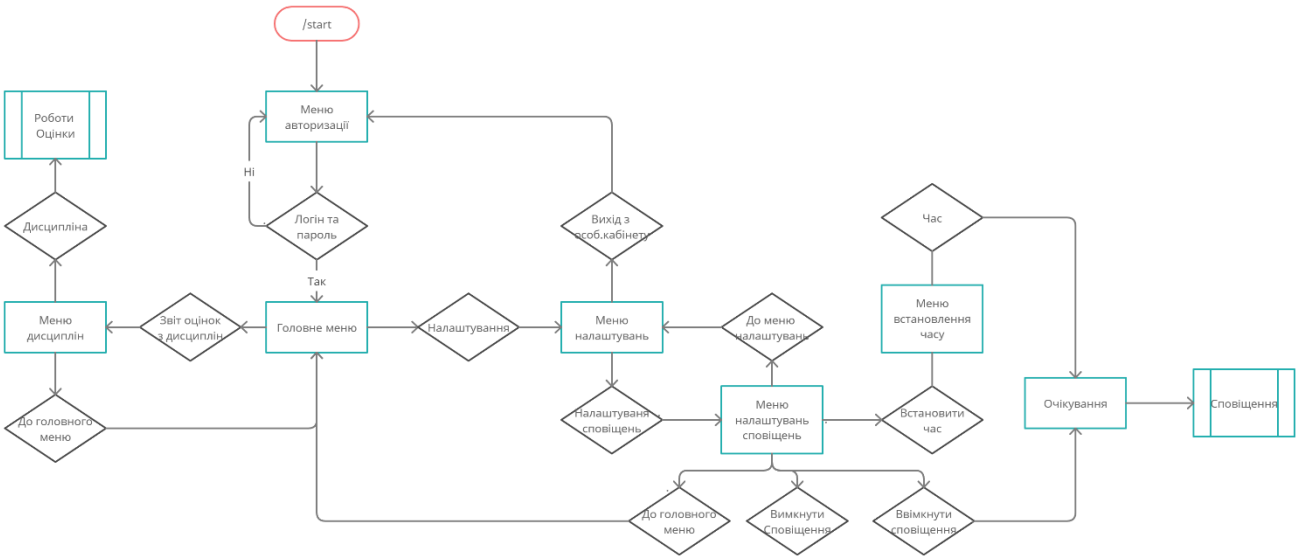


Рисунок 24 – Діаграма взаємодії

# ВИСНОВОК

В процесі розробки дипломного проекту визначено основні вимоги до розроблюваного додатку, проведено аналіз предметної області а саме сервісу для обміну миттєвими повідомленнями Telegram та чат-ботів. Проаналізовано чат-боти аналоги та встановлено їх переваги та недоліки.

На основі аналізу для розробки застосунку обрано редактор для написання програмного коду Sublime Text Editor, базу даних PostgreSQL, асинхронний фреймворк для роботи з Telegram Bot API – Aiogram, мову програмування Python, бібліотеки requests, Beautiful Soup, datetime та calendar.

Реалізовано наступні функції: авторизація в особистому кабінеті сайту дистанційного навчання, відправлення повідомлень з оцінками з навчальних дисциплін за запитом користувача, надсилання та налаштування сповіщень про останні терміни здачі робіт.

В подальшому планується вдосконалити додаток введенням функцій завантаження виконаних робіт на сайт дистанційного навчання через чат-бот та надсилання сповіщень з новими оцінками.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ


1) Intro to Chatbots [Електронний ресурс] - Режим доступу: https://capacity.com/chatbots/intro-to-chatbots/
2) ЩО ТАКЕ ЧАТ-БОТИ І 7 СПОСОБІВ, ЯК ВИКОРИСТОВУВАТИ ЇХ ДЛЯ БІЗНЕСУ [Електронний ресурс]: - Режим доступу: https://ag.marketing/chat-boti-dlya-biznesu-2/
3) What is Telegram and how different it is from other messaging apps [Електронний ресурс]: - Режим доступу: https://justaskthales.com/en/telegram-different-messaging-apps/
4) PostgreSQL: About [Електронний ресурс]: - Режим доступу: https://www.postgresql.org/about/
5) 1M rows/s from Postgres to Python [Електронний ресурс]: - Режим доступу: http://magic.io/blog/asyncpg-1m-rows-from-postgres-to-python/
6) asyncio — Asynchronous I/O [Електронний ресурс]: Документація – Режим доступу: https://docs.python.org/3/library/asyncio.html
7) HTTP/1.1: Status Code Definitions [Електронний ресурс]: – Режим доступу: https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html
8) Guide to Parsing HTML with BeautifulSoup in Python [Електронний ресурс]: – Режим доступу: https://stackabuse.com/guide-to-parsing-html-with-beautifulsoup-in-python/
9) Python's Requests Library (Guide) – Real Python [Електронний ресурс]: – Режим доступу: https://realpython.com/python-requests/#getting-started-with-requests
10) Beautiful Soup Documentation – Beautiful Soup 4.9.0 documentation [Електронний ресурс]: – Режим доступу: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
11) General Python FAQ – Python 3.9.5 documentation [Електронний ресурс]: – Режим доступу: https://docs.python.org/3/faq/general.html

12) Sublime Text Tutorial – Tutorialspoint [Електронний ресурс]: – Режим доступу: https://www.tutorialspoint.com/sublime_text/index.htm
13) aiogram PyPI [Електронний ресурс]: https://pypi.org/project/aiogram/
14) datetime – Basic date and time types – Python 3.9.5 documentation [Електронний ресурс]: – Режим доступу: https://docs.python.org/3/library/datetime.html#module-datetime
15) calendar — General calendar-related functions [Електронний ресурс]: – Режим доступу: https://docs.python.org/3/library/calendar.html#module-calendar
16) Bots: An introduction for developers [Електронний ресурс]: – Режим доступу: https://core.telegram.org/bots