НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
Кафедра інженерії програмного забезпечення

ПОЯСНЮВАЛЬНА ЗАПИСКА

до бакалаврської роботи

на ступінь вищої освіти бакалавр
на тему: «**РОЗРОБКА ТЕЛЕГРАМ БОТУ ДЛЯ ЗНАЙОМСТВ МОВОЮ ПРОГРАМУВАННЯ PYTHON**»

Виконав: студент 4 курсу, групи ПД– 43
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності)
Швецов В.І.
(прізвище та ініціали)
Керівник Трінтіна Н.А.
(прізвище та ініціали)
Рецензент
(прізвище та ініціали)
Нормоконтроль
(прізвище та ініціали)

Київ – 2022

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

“_____”_____2022 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Швецову Владиславу Ігоровичу

(прізвище, ім‘я, по батькові)

1. Тема роботи: «Розробка Телеграм боту для знайомств мовою програмування Python»

Керівник роботи Трінтіна Н.А., к.т.н., доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від «18» лютого 2022 року № .

2. Строк подання студентом роботи «03» червня 2022 року
3. Вихідні дані до роботи:
   - 3.1. Документація мови програмування Python та додаткових бібліотек;
   - 3.2. Документація середовища розробки PyCharm;
   - 3.3. Документація Telegram Bot API;
   - 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 рішень

4.3. Розробка та тестування телеграм боту
4.4. Висновки

5. Перелік графічного матеріалу.
   5.1. Титульний слайд
   5.2. Мета, об’єкт та предмет дослідження
   5.3. Актуальність роботи
   5.4. Аналоги
   5.5. Порівняння з аналогами
   5.6. Технічне завдання
   5.7. Програмні засоби реалізації
   5.8. Інструменти використані для реалізації
   5.9. Архітектура гри
   5.10. Апробація результатів дослідження
   5.11. Висновки

6. Дата видачі завдання: 11.04.2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	19.04.2022	Виконано
2	Аналіз існуючих аналогів	20.04.2022	Виконано
3	Дослідження програмних засобів	21.04.2022 – 22.04.2022	Виконано
4	Розробка функціоналу бота	23.04.2022 – 27.04.2022	Виконано
5	Висновки, оформлення роботи	27.04.2022 – 30.04.2022	Виконано
6	Розробка демонстраційних матеріалів	30.04.2022	Виконано
7	Попередній захист роботи	24.05.2022	
8	Здача роботи	03.06.2022	

Студент _____ Швецов В.І.
(підпис) (прізвище та ініціали)

Керівник роботи_____ Трінтіна Н.А.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

СТВОРЕННЯ ГРИ НА ЮНІТІ, АНАЛІЗ ІГРОВОГО ЖАНРУ, ШУТЕР МЕХАНІКИ

Текстова частина бакалаврської роботи: 65с., 45 рис, 1 дод., 2 табл., 11 джерел.

Ключові слова: PyCharm, Python, Telegram, aiogram, чат-бот, Telegram-бот.

Об'єкт дослідження – Спрощення та процес онлайн знайомства людини.

Предмет дослідження – Телеграм бот для онлайн знайомств.

Мета роботи – Спрощення онлайн знайомств з людьми за допомогою телеграм боту.

В роботі розглянуто основні відомості про додатки для онлайн знайомств, їх характеристики. Проаналізовано дослідження бібліотек для створення чат ботів. Проаналізовано можливості середовища розробки PyCharm, мови програмування Python. Також був спроектований, розроблений та протестований чат-бот для онлайн знайомств на платформі Telegram.

Галузь використання – бота може використовувати будь-яка людина, яка бажає завести онлайн знайомства.

ЗМІСТ

ВСТУП

Оскільки сучасний світ не стоїть на місці, технології стрімко розвиваються та кожного дня з’являються нові тренди та ідеї, багато з них безпосередньо впливають на IT індустрію, яка в свою чергу впливає на інші галузі, що в подальшому відображатиметься на нашому повсякденному житті. Якщо до недавнього часу популярними були додатки, сайти або комп’ютерні програми, то на даний момент лідерство займають чат-боти, які мають великі перспективи в повсякденних сферах нашого життя. Для початку потрібно визначити, що представляє собою чат-бот.

Чатбот – це додаток, який дозволяє користувачам взаємодіяти з програмами або сервісами у середині іншого додатку, якщо існує така необхідність і все це виконано через всім відомий інтерфейс чату. Чат-бот – це деякий помічник, який спілкується з користувачами через повідомлення і має множину певних функцій. Тобто, можна отримати певну інформацію, написавши чат-боту спеціальну команду, яку в свою чергу останній інтерпретує певним чином. Так можна швидко переводити, коментувати, знаходити, тестувати, шукати, навчати, транслювати, вбудовуватися в інші сервіси і платформи, взаємодіяти з датчиками і речами, підключеними до інтернету.

Актуальність роботи зумовлено зростанням кількості користувачів месенджерів, що може дати змогу перехопити нову аудиторію для свого проекту. На платформах месенджерів є можливість розробки та розгортання власних чат-ботів, для вирішення задач або інтегрування сервісів, що в свою чергу може привести нову аудиторію або користувачів. Актуальністю цього чат-боту є можливість його інтегрування для різних закладів та компаній, наприклад для учбового закладу або університету.

Об’єктом дослідження є спрощення та процес онлайн знайомства людини

Предметом роботи є телеграм бот для онлайн знайомств

Метою роботи є спрощення онлайн знайомств з людьми за допомогою телеграм боту.

Виходячи з поставленої мети, були поставлені наступні завдання:

- аналіз обраної предметної області;
- порівняння наявних аналогів програм для онлайн знайомств;
- вибір технологій і середовища розробки;
- розробка та тестування чат-боту на платформі Telegram;

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Аналіз та опис платформи месенджеру Telegram

Технологія обміну миттєвими повідомленнями — це тип онлайн-чату, що дозволяє передавати текст у режимі реального часу через Інтернет або іншу комп'ютерну мережу. Повідомлення зазвичай передаються між двома або більше сторонами, коли кожен користувач вводить текст і ініціює передачу одержувачу, які всі підключені до спільної мережі. Вона відрізняється від електронної пошти тим, що розмови через обмін миттєвими повідомленнями відбуваються в режимі реального часу. Більшість сучасних програм миттєвого обміну повідомленнями (іноді їх називають «месенджерами») використовують технологію push, а також додають інші функції, такі як емодзі, передача файлів, чат-боти, голос через IP або відео можливості чату.

Сьогодні месенджери стали найпопулярнішим способом онлайн спілкування, що дозволяє за короткий час обмінюватися повідомленнями з людьми по всьому світі.

Жанр відеогри використовується для класифікації відеоігор відповідно до інтерактивних ігрових дій гравця.

Telegram — це безкоштовна, хмарна служба обміну миттєвими повідомленнями. Послуга також забезпечує наскрізне зашифроване відеодзвінки, VoIP, обмін файлами та багато інших функцій.

Він був запущений для iOS 14 серпня 2013 року та Android 20 жовтня 2013 року. Сервери Telegram розповсюджені по всьому світу з п'ятьма центрами обробки даних у різних регіонах, а операційний центр знаходиться в Дубаї в Об'єднаних Арабських Еміратах. Доступні різні клієнтські програми для настільних і мобільних платформ, включаючи офіційні програми для Android, iOS, Windows, macOS та Linux. Усі офіційні компоненти Telegram є відкритим вихідним кодом, за винятком сервера із закритим кодом і власністю.

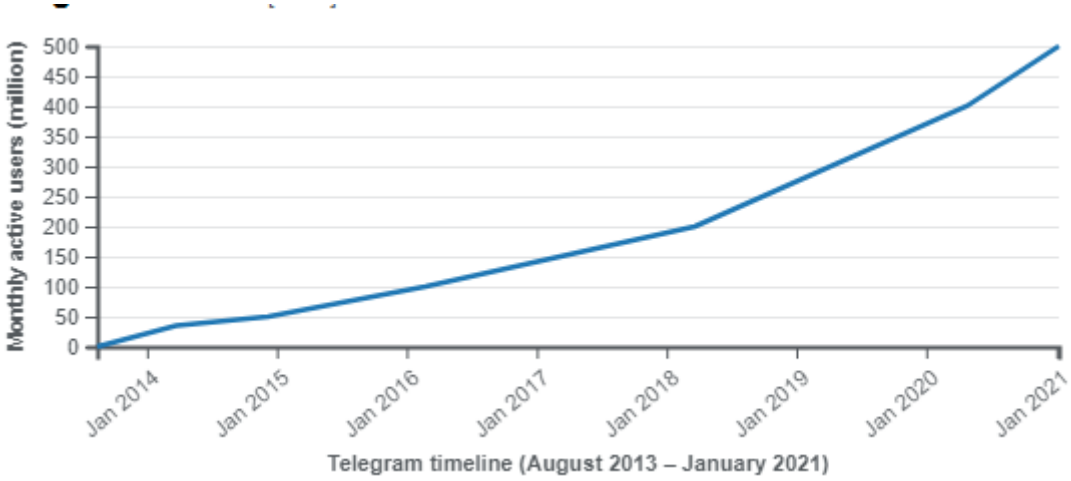


Рис. 1.1 – Кількість активних користувачів Telegram

Так за 2021 рік телеграм досяг близько 500 мільйонів активних користувачів за період одного місяця (рисунок 1.1). Що говорить про велику клієнтську базу телеграму.

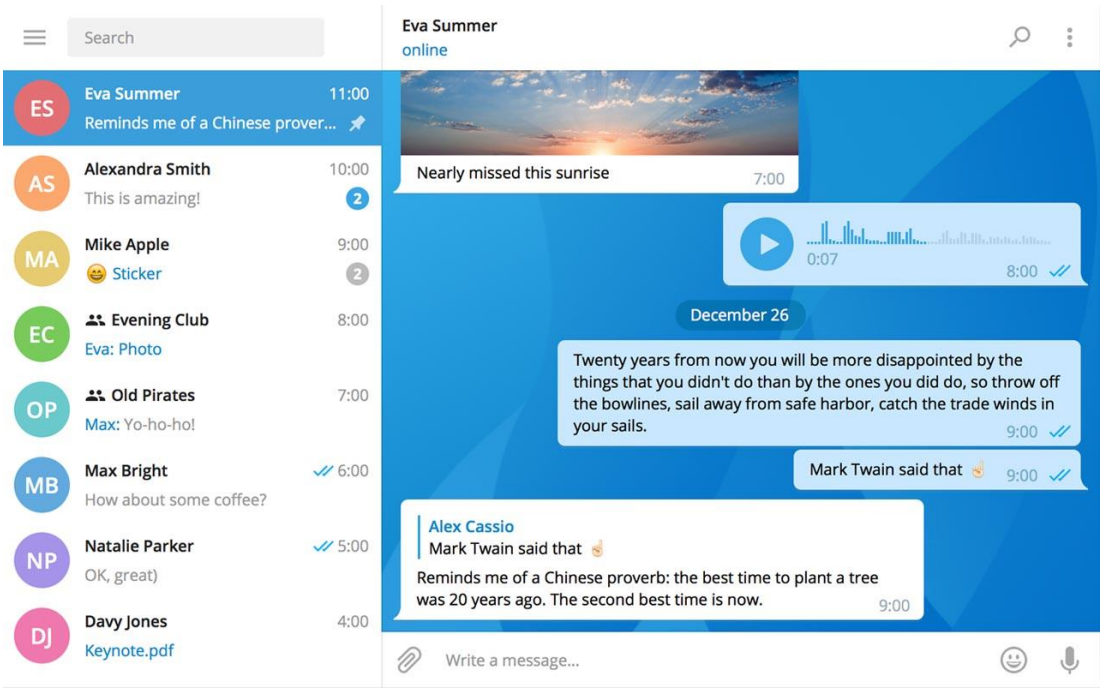


Рис. 1.2 – Інтерфейс додатку Telegram

Приклад інтерфейсу телеграм зображено на рисунку 1.2, також месенджер доступний на українській, англійській, польській, німецькій та інших мовах.

1.2. Аналіз та дослідження чат-ботів

Чатбот – це додаток, який дозволяє користувачам взаємодіяти з програмами або сервісами у середині іншого додатку, якщо існує така необхідність і все це виконано через всім відомий інтерфейс чату. Чат-бот – це деякий помічник, який спілкується з користувачами через повідомлення і має множину певних функцій. Тобто, можна отримати певну інформацію, написавши чат-боту спеціальну команду, яку в свою чергу останній інтерпретує певним чином. Так можна швидко переводити, коментувати, знаходити, тестувати, шукати, навчати, транслювати, вбудовуватися в інші сервіси і платформи, взаємодіяти з датчиками і речами, підключеними до інтернету.

Оскільки нові технології вкорінюються, очікування користувачів швидко зростають. Користувачі вимагають взаємодії в режимі реального часу — саме це і забезпечує чат-боти. Згідно з цим опитуванням, якщо вибирати між заповненням форми веб-сайту чи отриманням відповідей від чат-бота, лише 14% клієнтів вибрали б форму.

За результатами дослідження сайту salesforce отримуємо данні:

- 58% користувачів кажуть, що нові технології, такі як чат-боти та голосова допомога, змінили їхні очікування від компаній та послуг.
- 54% користувачів кажуть, що компаніям потрібно змінити спосіб взаємодії з ними.
- 77% користувачів стверджують, що чат-боти змінять їхні очікування щодо компаній або послуг протягом наступних п’яти років

Чат-боти мають одну велику перевагу: вони завжди ввімкнені. У звіті Survey Monkey споживачі оцінюють це як один з аспектів чат-ботів, що робить його кращим перед іншими методами спілкування бренду, такими як електронна пошта, чат або соціальні мережі.

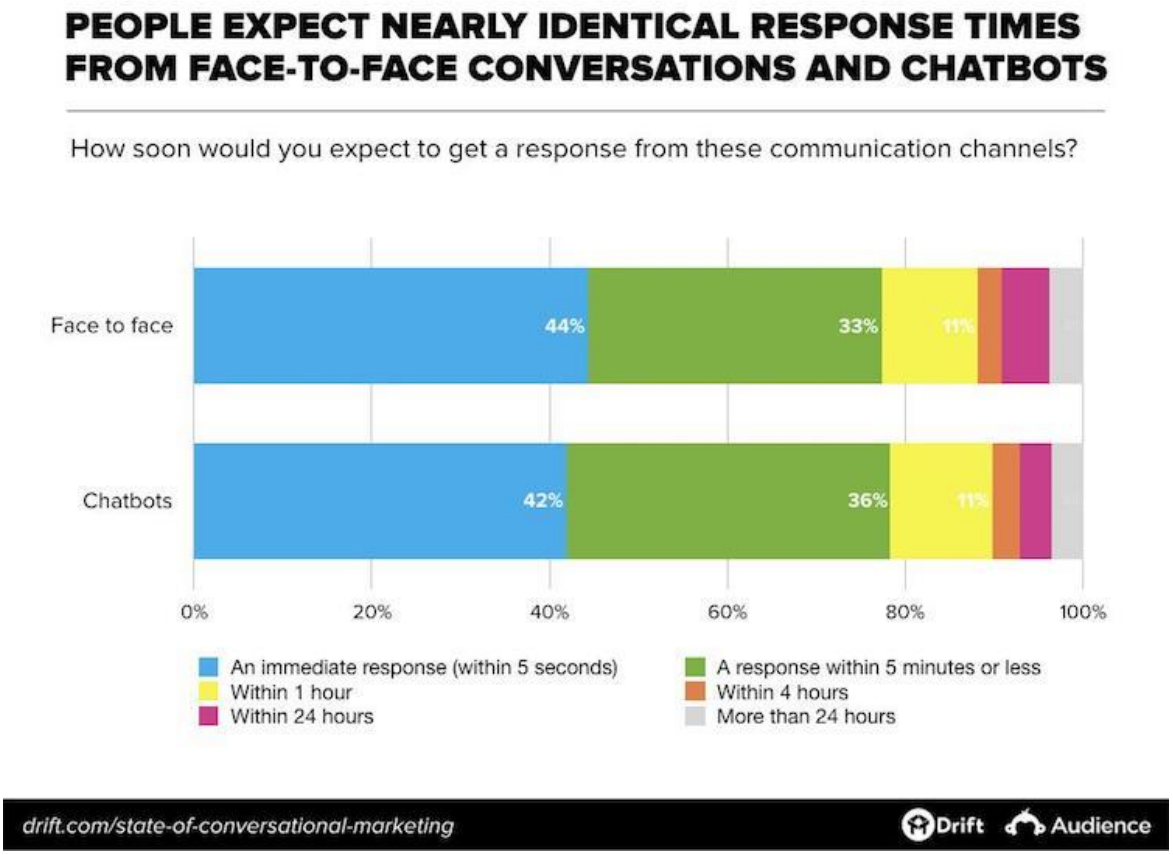


Рис. 1.3 – Діаграма дослідження очікування швидкості відповіді на питання чат-бота та людини у розмові віч-на-віч.

Іншим основним показником чудового досвіду роботи з чат-ботом є швидкість. «Обслуговування користувачів у режимі реального часу» — це фраза, яка досить часто лунає, але чат-ботам справді потрібно виконувати її. Наведена вище діаграма дослідження сайту drift показує, що споживачі очікують, що чат-боти відповідатимуть майже так само швидко, як і представник служби в розмові віч-на-віч.

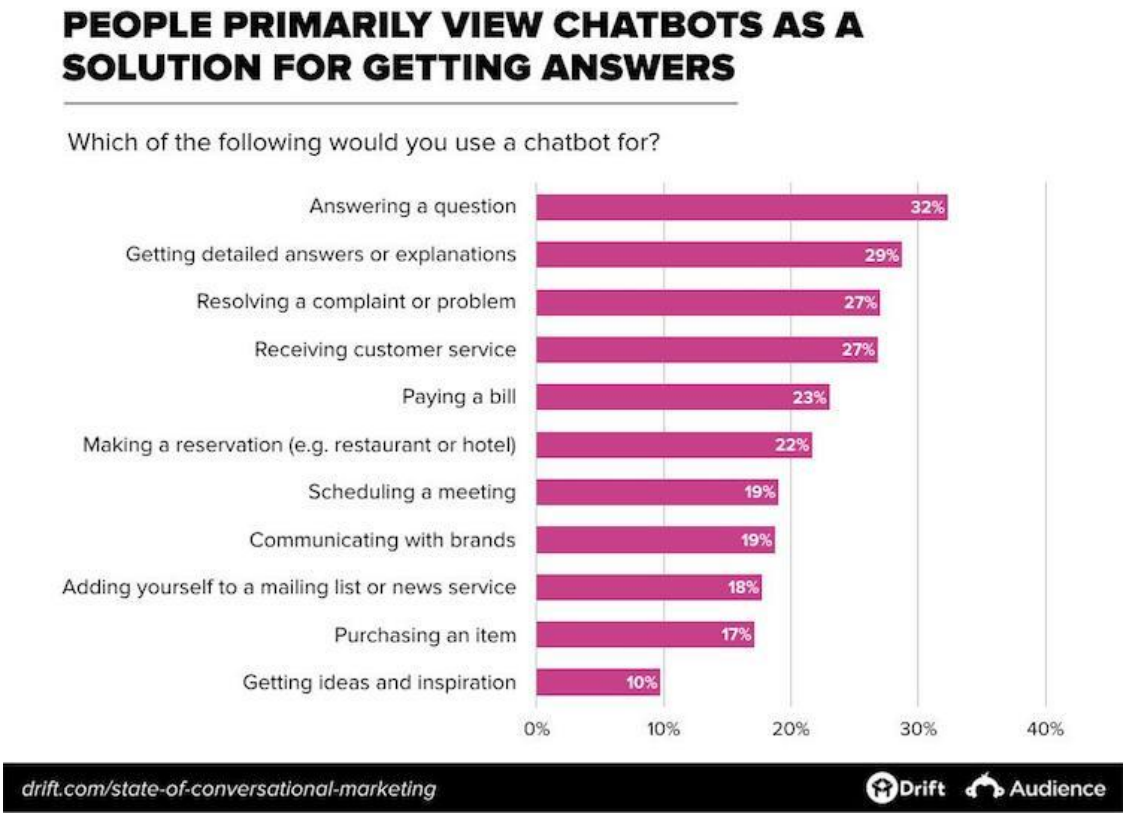


Рис. 1.4 – Діаграма дослідження використання чат-ботів за думкою користувачів

У діаграмі вище показано, як споживачі кажуть, що вони будуть використовувати чат-ботів. Існує певна застереження щодо використання чат-ботів для таких завдань, як оплата рахунків і покупка товарів, але коли йдеться про гроші, люди часто вагаються з новими технологіями. Це закономірна частина впровадження нових технологій, і якщо ми поглянемо на зорі електронної комерції, то розгорнулася та ж тенденція.

Тож ми розібралися, що чат-бот дуже добре підходить під нашу ідею і зможе дозвалити максимально комфортно знайомитися нашим користувачам, використовуючи вже знайому для них програму.

Одним із найпопулярніших зараз месенджерів є Telegram. Одже використовуючи уже популярний додаток, у нас не виникатиме проблеми з притоком нової аудиторії, а також користувачам не потрібно буде вчитися використовувати новий додаток.

1.3. Аналіз програмного забезпечення для онлайн знайомств

Офлайн-знайомства – це самий простий та доступний варіант познайомитися з людиною. Можна познайомитися з новими людьми на навчанні, роботі, займаючись улюбленим хобі або ж просто з незнайомцем на вулиці.

Але офлайн-знайомство не завжди може відбуватися у зручний час, іноді ви або інші люди можуть бути не в настрої, або ж, банально, не мати на це вільний час. До того ж такі знайомства найчастіше відбуваються випадково і не завжди з тими людьми які на подобаються. Тож це означає, що люди знаходять своїх майбутніх партнерів у барах, університетах, на роботі через друзів батьків і навіть у церкві. Тобто, коло потенційних знайомств так чи інакше буде обмежено соціальним статусом та місцем проживання.

Онлайн побачення повністю змінили цю систему. В інтернеті перетинаються люди з усіх соціальних груп та верств. Ті, хто раніше навіть не могли опинитися на одній вулиці, тепер можуть познайомитись та одружитися.

Крім цього, вчені виявили ще один цікавий феномен: шлюби для людей, які познайомилися в інтернеті, розпадаються рідше, ніж шлюби для людей, які познайомилися традиційним чином. На думку дослідників, це пов'язано з тим, що інтернет-знайомства ламають усі соціальні бар'єри, дають людям можливість знайомитися і впізнавати один одного без прив'язки до оточення, стереотипів та звичного способу проведення часу.

«Знайомства в інтернеті зменшують шанси на швидке розлучення, оскільки партнери керуються не лише зовнішніми даними, а й тим, наскільки їм цікаво спілкуватися одне з одним, — додав один із авторів дослідження Філіп Гергович із Університету Відня. — Більше того, пари, які почали фліртувати в додатках для побачень, переносять тривалу розлуку легше за пари, які познайомилися офлайн. Можливо, саме онлайн-знайомства, зрештою, допоможуть нам побудувати більш гармонійне суспільство».

В якості онлайн знайомств розглянемо декілька прикладів, а саме: Сайти для

знайомств, Додатки для знайомств та Боти для знайомств

1.3.1. Сайти для знайомств

Одним із найстарших інструментів для онлайн знайомств є сайти для знайомств. Щоб скористатися цим варіантом, користувачу потрібен лише комп’ютер або телефон з доступом у мережу інтернет. Зазвичай користувачу потрібно обов’язково зареєструватися на такому сайті за допомогою пошти, а вже потім після підтвердження буде можливість створювати анкету та знайомитися з людьми.

Після реєстрації користувач створює анкету, заповнюючи форму, додає фотографію та має змогу шукати нові знайомства. Із функціоналу сайти мають найдетальніші фільтри для пошуку, оскільки це може дозволити декстоп адаптований інтерфейс. Користувач може вказати країну, місто, вік та інтереси для подальшого пошуку людей. Але такі сайти у котрих деякі функції вимагають платної підписки.

Прикладами таких сайтів є сайт obyava.ua

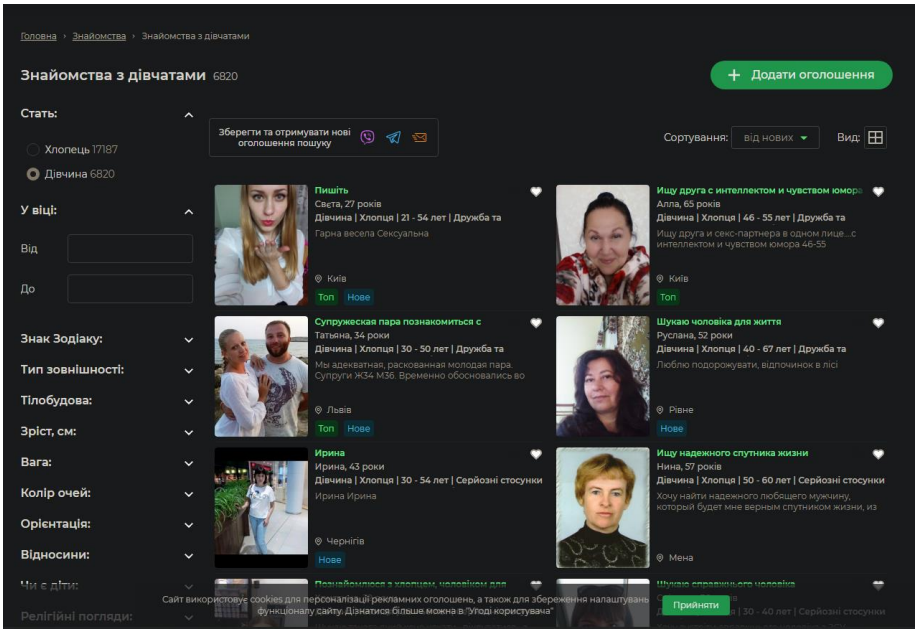


Рис. 1.5 – Вигляд сайту для знайомств obyava.ua

obyava - Це український сервіс для знайомства і спілкування. Можна написати повідомлення вподобаному користувачеві, посміхнутися або оцінити фотографії. Також можна поставити в статус свій настрій і заявити про бажання. Наприклад, поставити "хочу в кіно" або "без настрою".

Переваги:

- Доступність, потребує лише комп’ютера або телефона, доступу до інтернету та аккаунт електронної пошти.
- Високий функціонал, має можливість написання великої кількості функціоналу доступному для архітектури сайту

Недоліки:

- Неактуальніть, більшість користувачів 30+ років
- Велика кількість шахраїв
- Найбільша кількість реклами та платних функцій

1.3.2. Додатки для знайомств

Після розповсюдження смартфонів почали набирати популярності мобільні додатки, за своєю концепцією, вони спочатку повністю адаптували функціонал сайтів для мобільних пристроїв, пізніше вже стали розроблятися додатки винятково для смартфонів, не маючи інтернет аналогів.

Прикладами таких додатків можна назвати Tinder та Badoo

Tinder – цей додаток популярний не тільки на території України, а й у всьому світі. Тому тут можна знайти пару практично в будь-якому куточку планети. Плюсом є також те, що геолокація підбирає потенційних партнерів, які є поблизу. Елементарний інтерфейс, можливість приєднати профіль в інстаграм (щоб не довантажувати купу фото) й убезпечити себе від настирливих користувачів роблять цей додаток одним із кращих для знайомств у мережі. Так, тільки взаємно підтверджені користувачі можуть почати спілкування.

Рис. 1.6 – Вигляд додатку для знайомств Tinder

Badoo – Додаток для знайомств, у якого практично стільки ж переваг, скільки й недоліків. Брак інформації в анкеті компенсується великою кількістю користувачів із різних країн, що буде корисно тим, хто хоче підтягнути іноземну мову з носієм. Цікавою функцією програми є пошук свого двійника серед зовні схожих людей. Також можна ввімкнути місце розташування і подивитися, хто поруч.

Рис. 1.7 – Вигляд додатку для знайомств Badoo

Такі додатки потрібно попередньо завантажити його через один із магазинів. Відкривши його, користувачу потрібно буде також пройти реєстрацію за допомогою електронної пошти, після чого він зможе ним користуватися.

Спочатку користувач створює анкету, після чого може користуватися додатком. Після реєстрації користувач створює анкету і отримує доступ до знайомств.

В додатках основна система для пошуку – «свайпи», коли користувачу у майже нескінченний спосіб один за одним пропонують кандидатів для знайомства, і користувач обирає подобається йому людина чи ні, якщо так, то він у такий же спосіб потрапляє до списку того хто йому сподобався, і якщо це взаємно то додаток пропонує почати спілкування. Також популярна функція у додатках «люди поруч» яка за допомогою геолокації знаходить аккаунти людей які знаходяться поруч з користувачем, який відразу може почати з ними спілкування.

Переваги:

- Актуальність, мають більш продвинуті та цікаві функції, мають більш молодшу аудиторію та найпопулярніші на ринку.
- Функціонал, мають кращі фільтри та алгоритми пошуку, можуть використовувати штучній інтелект.

Недоліки:

- Наявність в деяких додатках платних функцій
- Із-за популярності мають велику кількість ботів та шахраїв
- Іноді можуть містити рекламу

1.3.3. Боти для знайомств

Після розповсюдження месенджерів стали набирати популярності і боти, які дозволяють розширити функціонал додатку. Одними з таких стали боти для знайомств.

Прикладом можуть бути телеграм боти: Леонардо Дайвинчик Бот, ЛавСтори та StrangerBot

«LeonardoDavinchik» — Бот від популярного каналу з новинами та мемами, націлений більше на молоду аудіторію. Простий у використанні.

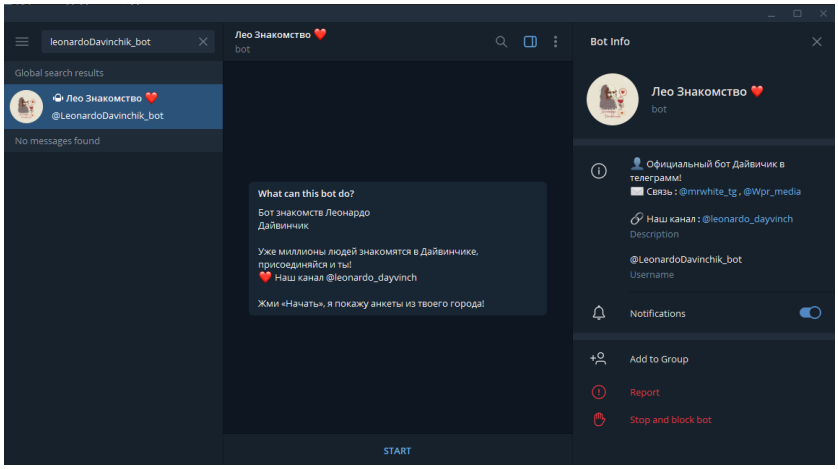


Рис. 1.8 – Вигляд боту для знайомств LeonardoDavinchik

«LoveStory» — Бот знайомств, який має функцію автоматичного перегляду анкет та якісний захист від спаму.

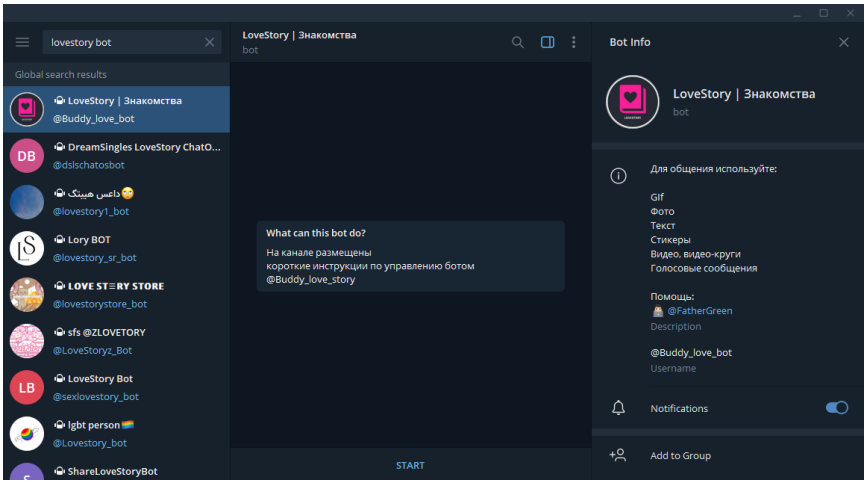


Рис. 1.9 – Вигляд боту для знайомств LoveStory

«StrangerBot» — Бот, створений насамперед для спілкування з випадковим співрозмовником. Мінусом і те, що не можна виставити критерії пошуку співрозмовника.

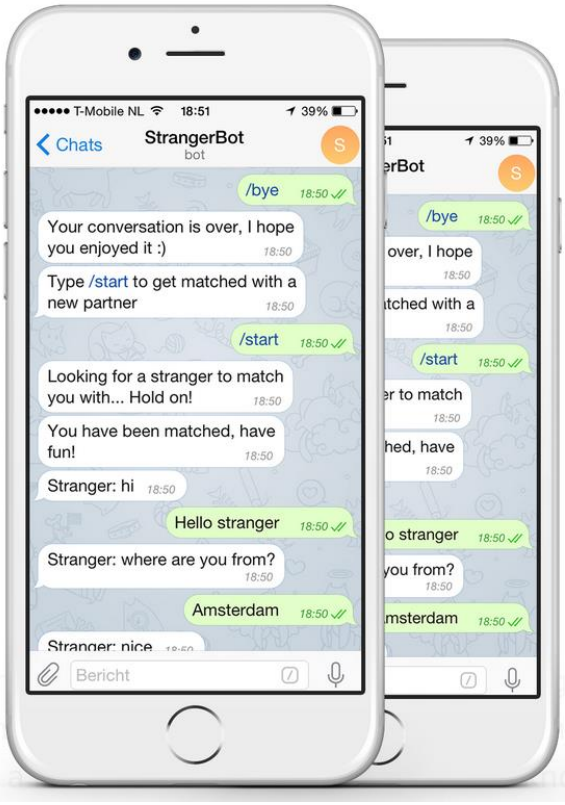


Рис. 1.10 – Вигляд боту для знайомств StrangerBot

Боти для знайомств практичні тим, що користувачу потрібно лише знайти бота для свого месенджеру, яким він користується кожен день. Для реєстрації не потрібно нічого вказувати, за все буде відповідати аккаунт у месенджері або соц. мережі. Після знаходження бота та початку з ним розмови, користувачу одразу пропонується створити анкету, після чого йому буде доступна нескінченна лента анкет, майже як у додатку. Якщо користувачу подобається людина, він зможе вподобати її. Після чого цій людині прийде оповіщення, після чого можна буде почати спілкування та обмінятися контактами месенджера.

Переваги:

- Інтегрованість у месенджер, одразу доступний зручний чат
- Націленість на молодшу аудиторію
- Простота використання
- Відсутність реєстрації

Недоліки:

- Малий функціонал порівняно з аналогами
- Багато фейкових або анонімних аккаунтів

Таблиця 1.1 – Зведені результати порівняння програмного забезпечення для онлайн знайомств

Показник	**Сайти**	**Додатки**	**Боти**
Платформи	Всі платформи з браузером до виходом у мережу.	Android, IOS	Інтеграція в додаток месенджеру(залежить від платформ месенджеру)
Нескінченна лента запропонованих знайомств	-	+	+
Функція «люди поруч»	-	+	-
Фільтри для пошуку	+	+	-
Можливість користування без реєстрації	-	-	+(окрім, реєстрації месенджеру)
Зручний та актуальний чат	-	-	+
Зручність повсякденного використання	-	+	+

1.4. Постановка завдань дослідження

Під час розробки концепції телеграм боту для знайомств було визначено, що перш ніж розробляти бота потрібно дослідити бібліотеки для для створення телеграм ботів та визначити які технології потрібно використати для розробки.

Тому для ефективності побудови бота, були поставлені завдання на дослідження певних технологій та інструментів, що дозволять реалізувати розроблену концепцію гри, а саме:

- Дослідити мову програмування Pyhon
- Дослідити інструменти та бібліотеки для розробки телеграм ботів
- Дослідити інтегровану середу розробки PyCharm

2. ОГЛЯД ТА АНАЛІЗ ІСНУЮЧИХ ІСТРУМЕНТІВ ДЛЯ СТВОРЕННЯ ТЕЛЕГРАМ БОТУ

2.1. Мова програмування Python

Для того щоб створити телеграм бота, для початку потрібно обрати мову програмування, зазвичай розробку телеграм ботів підтримують усі мови програмування на яких розробляється backend, тобто, програмно-апаратна частину сервісу або сайту, що працює на сервері. Із популярних мов можна обирати з PHP, Python, Node js, C# або Go. Мною була обрана мова програмування Python.

Python — це високорівнева, інтерпретована мова програмування загального призначення. Його філософія дизайну підкреслює читабельність коду з використанням значних відступів. Python має сувору динамічну типізацію.

Об’єктно-орієнтоване програмування та структурне програмування повністю підтримуються, і багато його функцій підтримують функціональне програмування та аспектно-орієнтоване програмування (включаючи метапрограмування та метаоб’єкти). Багато інших парадигм підтримуються за допомогою розширень, включаючи проектування за контрактом і логічне програмування.

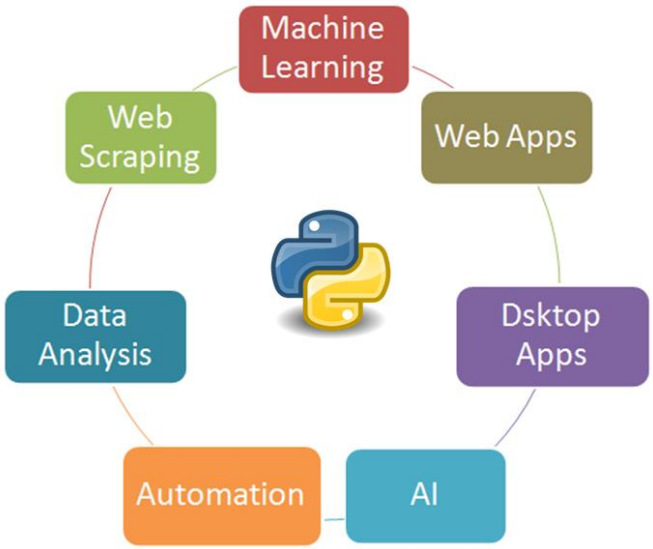


Рисунок 2.1 – Діаграма використання мови програмування Python

Серед основних її переваг можна назвати такі:

- чистий синтаксис (для виділення блоків слід використовувати відступи);
- переносність програм (що властиве більшості інтерпретованих мов);
- стандартний дистрибутив має велику кількість корисних модулів (включно з модулем для розробки графічного інтерфейсу);
- можливість використання Python в діалоговому режимі (дуже корисне для експериментування та розв'язання простих задач);
- стандартний дистрибутив має просте, але разом із тим досить потужне середовище розробки, яке зветься IDLE і яке написане мовою Python;
- зручний для розв'язання математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної величини, у діалоговому режимі може використовуватися як потужний калькулятор);
- відкритий код (можливість редагувати його іншими користувачами).

Також для того щоб швидко та просто використовувати сторонні бібліотеки, був розроблений менеджер пакетів PIP.

PIP — це інсталятор пакетів для Python. Ви можете використовувати pip для встановлення пакетів із індексу пакетів Python та інших індексів.

```
[root@localhost distribute-0.7.3]# pip --help

Usage:
  pip <command> [options]

Commands:
  install                     Install packages.
  uninstall                   Uninstall packages.
  freeze                      Output installed packages in requirements format.
  list                        List installed packages.
  show                        Show information about installed packages.
  search                      Search PyPI for packages.
  wheel                       Build wheels from your requirements.
  help                        Show help for commands.

General Options:
  -h, --help                  Show help.
  --isolated                  Run pip in an isolated mode, ignoring environment variables and user configuration.
  -v, --verbose               Give more output. Option is additive, and can be used up to 3 times.
  -V, --version               Show version and exit.
  -q, --quiet                 Give less output.
  --log <path>                Path to a verbose appending log.
  --proxy <proxy>             Specify a proxy in the form [user:passwd@]proxy.server:port.
  --retries <retries>         Maximum number of retries each connection should attempt (default 5 times).
  --timeout <sec>             Set the socket timeout (default 15 seconds).
  --exists-action <action>    Default action when a path already exists: (s)witch, (i)gnore, (w)ipe, (b)ackup.
  --trusted-host <hostname>   Mark this host as trusted, even though it does not have valid or any HTTPS.
  --cert <path>               Path to alternate CA bundle.
  --client-cert <path>        Path to SSL client certificate, a single file containing the private key and the certificate in PEM
                              format.
  --cache-dir <dir>           Store the cache data in <dir>.
  --no-cache-dir              Disable the cache.
  --disable-pip-version-check
                              Don't periodically check PyPI to determine whether a new version of pip is available for download.
                              Implied with --no-index.
```

Рисунок 2.2 – Взаємодія з менеджером пакетів PIP за допомогою консолі

2.2. Огляд існуючих бібліотек для написання телеграм боту

Бібліотека Python — збірка модулів, об'єктів, підпрограм для вирішення близьких за тематикою задач засобами мови Python.

Для Python існує декілька найпопулярніших бібліотек:

- pyTelegramBotAPI
- AIOGram
- python-telegram-bot
- Telethon

2.2.1. Бібліотека pyTelegramBotAPI

pyTelegramBotAPI - одна з простих бібліотек яка має зрозумілий синтаксис та проста у використанні. Підходить для нескладних телеграм ботів, має зрозумілу та обширну документацію. Має версію для асинхронного програмування.

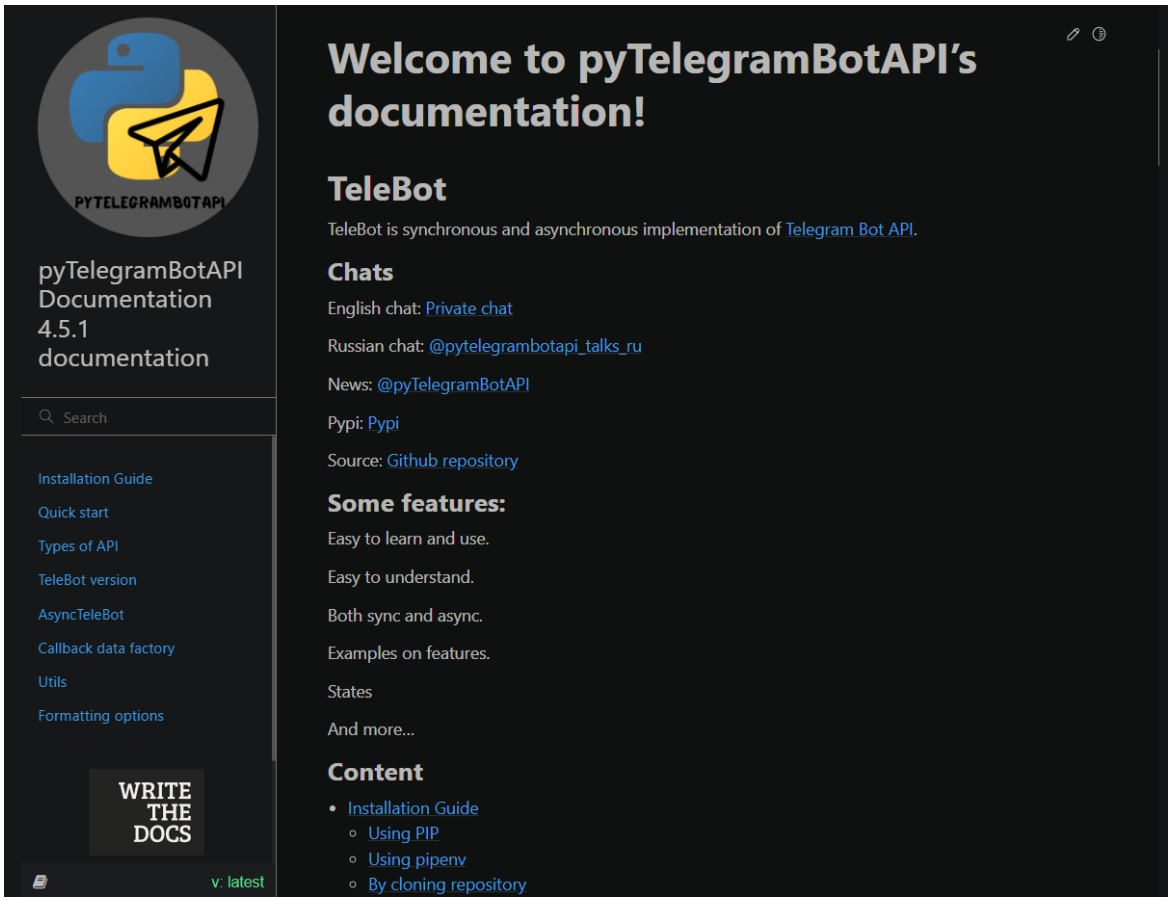


Рисунок 2.3 – Документація бібліотеки pyTelegramBotAPI

Переваги:

- Простий синтаксис та код
- Зручність використання
- Підтримка, документація та спільнота

Недоліки:

- Потребує окрему версію під асинхронне програмування.

2.2.2. Бібліотека AIOGram

AIOGram – підходить для важких та високо-об’ємних проектів, має велику підтримку від користувачів та спільноти, але має більш складніший інтерфейс та займає більше місця для коду. Повністю підтримує асинхронне програмування.

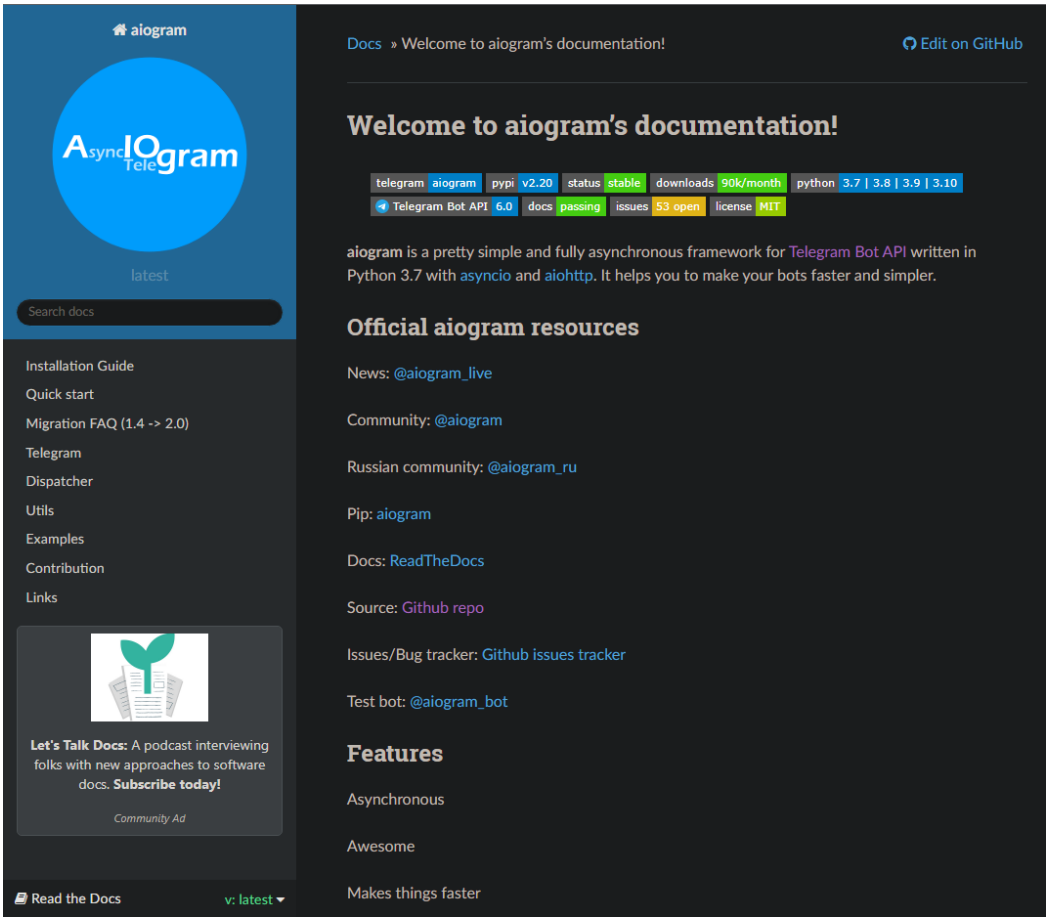


Рисунок 2.4 – Документація бібліотеки AIOGram

Переваги:

- Проста робота з великим обсягом запитів
- Підтримка асинхронного програмування
- Підтримка, документація та спільнота

Недоліки:

- Складність у використанні

2.2.3. Бібліотека python-telegram-bot

python-telegram-bot – асинхронний інтерфейс для Telegram Bot API. Має найбільш «чисту» реалізацію функцій Telegram Bot API, працює у додатку з саб-модулем telegram.ext для більш зручної реалізації деяких функцій.

Рисунок 2.5 – Документація бібліотеки python-telegram-bot

Переваги:

- Підтримка асинхронного програмування
- Майже «чиста» реалізація інтерфейсу для Telegram Bot API

Недоліки:

- Складність у використанні
- Потребує розуміння функцій Telegram Bot API

2.2.4. Бібліотека Telethon

Telethon – альтернативна бібліотека для взаємодії з API Telegram через обліковий запис бота та протокол Python 3 MTProto. Може обходити стандартні обмеження API Telegram для клієнта(Наприклад, завантажувати файли об’ємом більше 40мб).

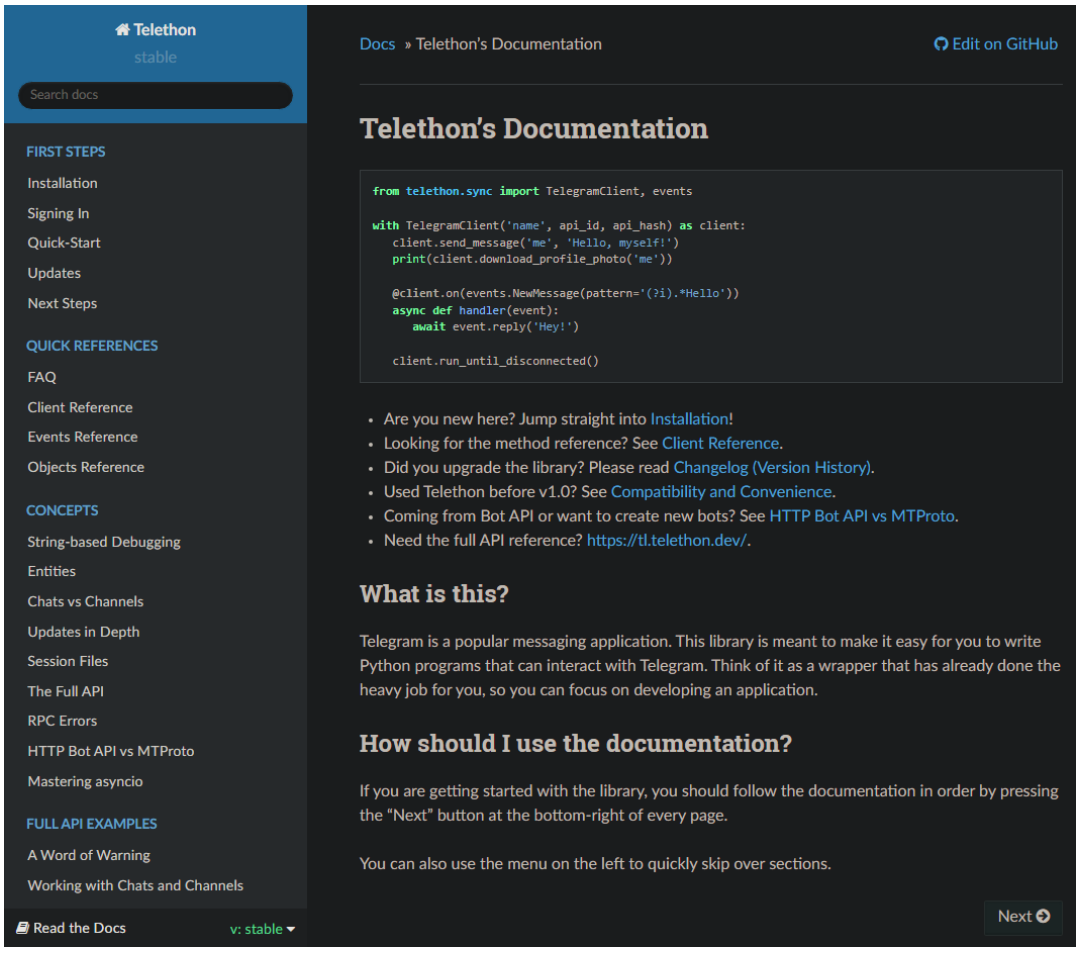


Рисунок 2.6 – Документація бібліотеки Telethon

Переваги:

- Підтримка асинхронного програмування
- Обхід більшості обмежень стандартного Telegram Bot API

Недоліки:

- Складність у використанні
- Потребує знання роботи проколу MTProto
- Працює лише у режимі «клієнта»

2.3. Огляд та переваги СУБД MariaDB

MariaDB Server є однією з найпопулярніших реляційних баз даних з відкритим кодом. Він створений оригінальними розробниками MySQL і гарантовано залишається відкритим вихідним кодом. Він є частиною більшості хмарних пропозицій і стандартним у більшості дистрибутивів Linux.

2.3.1. Уявлення

У частині продуктивності уявлень MariaDB проведена істотна оптимізація. "Уявлення" - це, по суті, віртуальні таблиці бази даних, до яких можна звертатися, як до звичайних таблиць бази даних. У MySQL при запиті до подання запитуються всі таблиці, пов'язані з цим поданням, незалежно від того, що для запиту можуть не знадобитися деякі уявлення. На відміну від MySQL, MariaDB, запитуються тільки ті таблиці, які необхідні для запиту.

У MariaDB додані оптимізації, які підвищують продуктивність СУБД відповідно до оригінального MySQL.

2.3.2. Колонкове сховище

MariaDB надає ще одне потужне покращення продуктивності, що досягається за допомогою нового типу таблиць, представлених не у формі

рядкового сховища, а у формі колонкового сховища. Стовпчики часто використовуються в аналітиці великих даних. MariaDB дозволяє масштабувати сховище даних до петабайтного розміру, забезпечуючи лінійне підвищення продуктивності запитів до даних, що зберігаються при додаванні нових серверів.

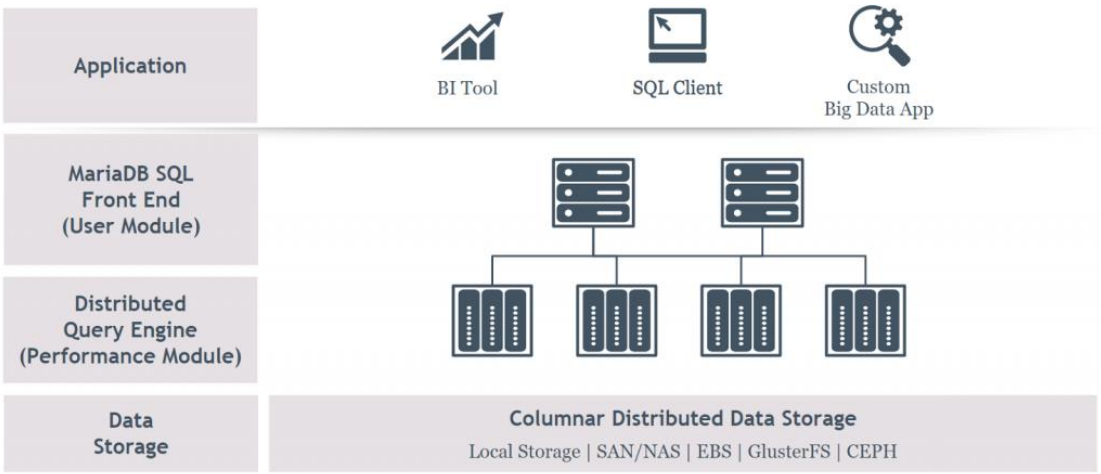


Рисунок 2.7 – Діаграма представлення колонкового сховища даних

2.3.3. Вища продуктивність на SSD

MariaDB надає механізм зберігання MyRocks, який дозволяє зберігати дані у RocksDB. RocksDB — це база даних, що вбудовується, яка була розроблена для підвищення продуктивності обробки даних, що зберігаються на SSD-накопичувачах.

2.3.4. Паралельне виконання запитів

Одна з останніх версій MariaDB - 10.0 припускає паралельне виконання кількох запитів. Ідея у тому, деякі запити від Master може бути передані виконання на ведені сервери (slave). Цей паралелізм у виконанні запитів, безумовно, забезпечує MariaDB перевагу над MySQL.

2.3.5. Пул потоків

MariaDB також представляє нову концепцію під назвою "Thread Pooling". Раніше, коли потрібно кілька з'єднань з базою даних, для кожного з'єднання відкривався потік, що призводило до архітектури «один потік на з'єднання». З використанням "Thread Pooling" використовується пул потоків, які можуть повторно використовуватись. Таким чином, новий потік не потрібно відкривати для кожного нового запиту підключення, що призводить до більш швидких результатів запиту. Ця функція доступна у комерційній версії MySQL, але, на жаль, недоступна у версії спільноти.

2.4. Огляд та переваги інтегрованої середи розробки PyCharm

Після визначення з мовою програмування та бібліотекою для створення телеграм бота, час визначитися з середою розробки.

PyCharm – це інтегроване середовище розробки для Python, яке має повний комплект засобів, необхідних для ефективного програмування на Python.

Зараз PyCharm поширюється у двох варіантах: платному (PyCharm Professional Edition) та безкоштовному (PyCharm Community Edition). Безкоштовна версія має відкритий вихідний код і розповсюджується під ліцензією Apache 2. Це полегшене середовище, яке підходить для розробки лише на Python. Платний варіант є більш розширеною та функціональною версією з можливістю розробки в тому числі багатомовних веб-додатків. Professional Edition підтримує фреймворки:

- Django
- Flask
- Google App Engine
- Pyramid
- web2py

І дає можливість дистанційної розробки, а також роботи з базами даних.

PyCharm робить розробку максимально продуктивною завдяки функціям

автодоповнення та аналізу коду, миттєвому підсвічуванню помилок та швидким виправленням. Автоматичні рефакторинги допомагають ефективно редагувати код, а зручна навігація дозволяє миттєво переміщатися за проектом.

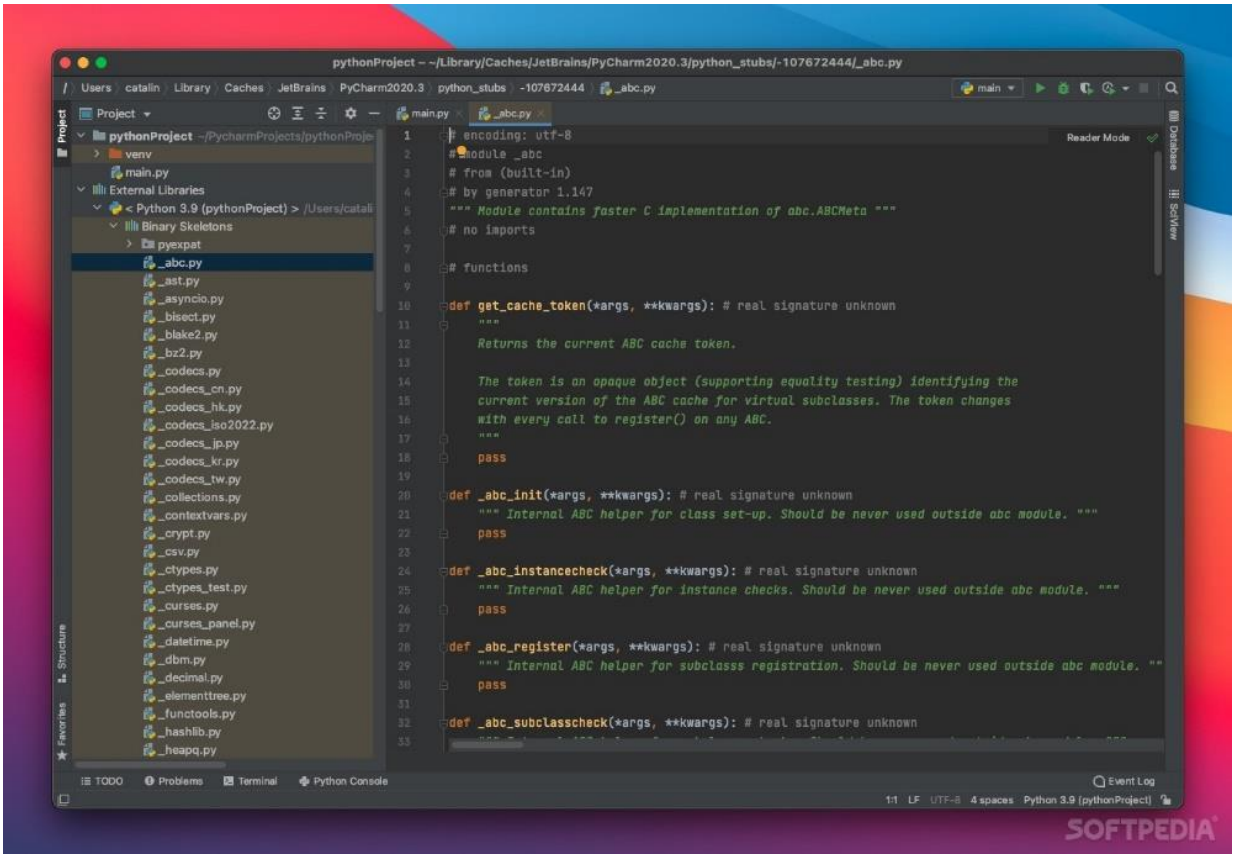


Рисунок 2.8 – Вигляд середовища розробки PyCharm

PyCharm пропонує великий набір інструментів з коробки: вбудований відладчик та інструмент запуску тестів, профільник Python, повнофункціональний вбудований термінал, інструменти для роботи з базами даних. IDE інтегрована з популярними системами контролю версій, містить вбудований SSH-термінал, підтримує можливості віддаленої розробки та віддалені інтерпретатори, а також інтеграцію з Docker та Vagrant.

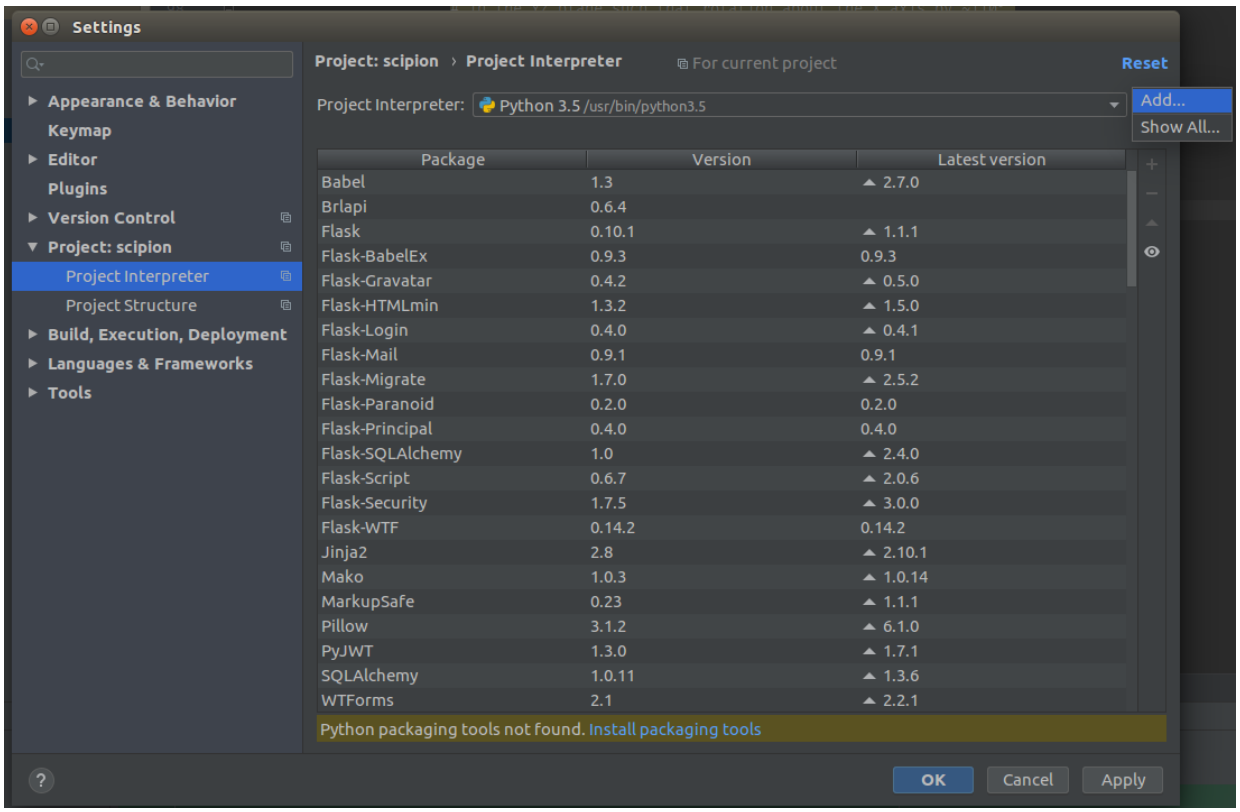


Рисунок 2.9 – Перегляд пакетів для інтерпритатора Python в PyCharm

Переваги:

- Проста та зрозуміла інтеграція з Git
- Просте створення та організація проектів
- Розумний автокомпліт
- Інтеграція з Python фрейморком Django
- Можливість вивантажувати проект одразу на FTP сервер
- Із-за поширеності продукції компанії, також має зручну можливість одразу редагувати HTML та JS код

Недоліки:

- Відсутність пакетного менеджера
- Відсутність Live unit-тестування
- Погана крос-платформа для Linux

3. СТВОРЕННЯ ТА РЕАЛІЗАЦІЯ ЧАТ БОТУ ДЛЯ ЗНАЙОМСТВ

3.1. Реєстрація домену чат бота

Для того, щоб зареєструвати свого чат бота, потрібно через телеграм знайти чат-бота з іменем «BotFather».

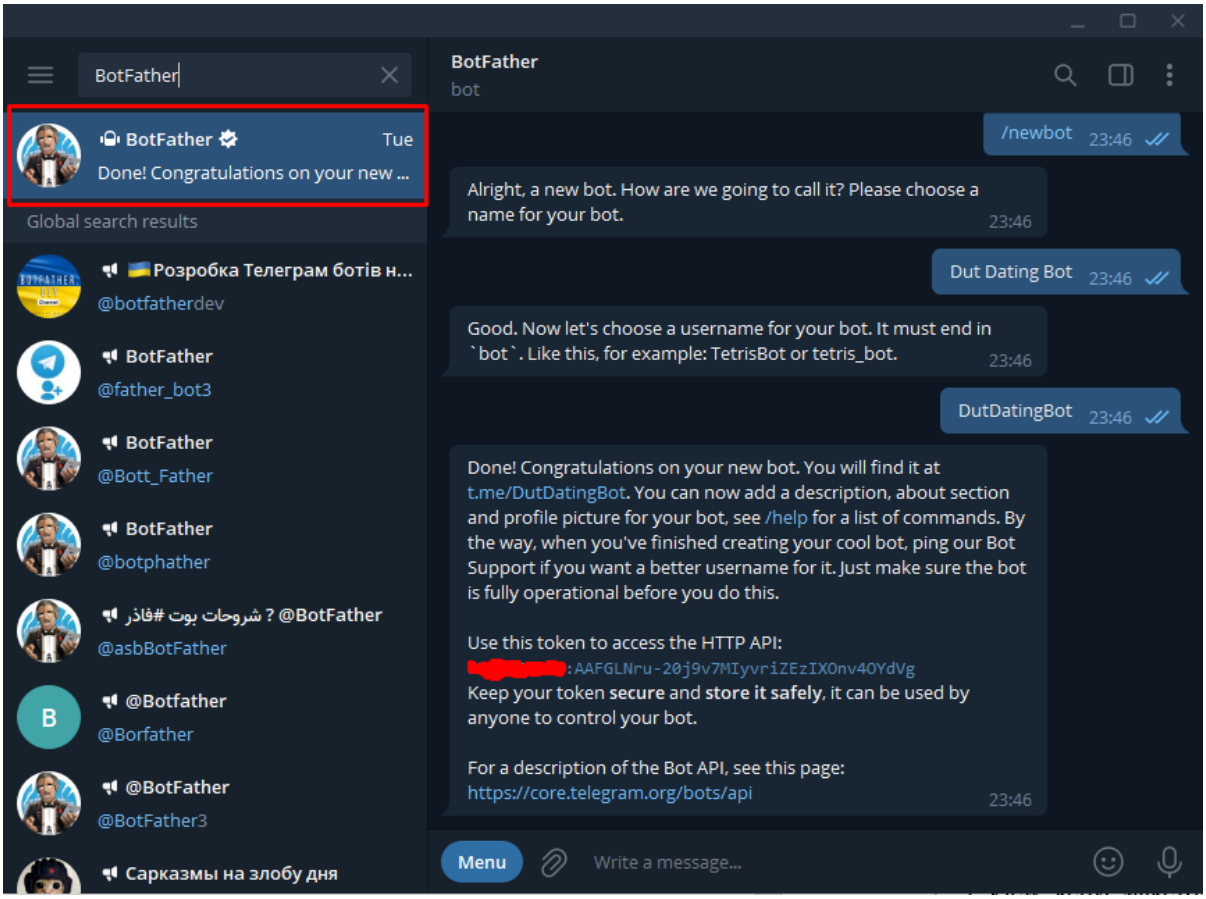


Рисунок 3.1 – Вигляд середовища розробки PyCharm

Після чого, бот запросить ввести у наступному повідомленні ім’я яке буде відображатися як назва нашого бота, після чого потрібно буде ввести ідентифікатор бота який обов’язково має містити закінчення «Bot» або «_bot».

Після того як ви ввели всі необхідні данні бот відправить вам url посилання на вашого бота та API-токен (який надає доступ до HTTP API Telegram). Приклад діалогу для реєстрації також наведено на рисунку 3.1.

Токен - це спеціальний ключ від бота, за допомогою якого його можна підключати до сторонніх сервісів. Токен потрібно зберегти і нікому не показувати, він такий же важливий, як і пароль від пошти.

Тепер боту можна налаштувати додаткові параметри, а саме, встановити аватар, додати опис а також видяляти або редагувати його. Повиний список команд доступних до взаємодії з BotFather наведено у таблиці 3.1

Таблиця 3.1 – Команди чат-боту «BotFather»

Команда	**Опис взаємодії**
/newbot	створити нового бота
/mybots	редагувати своїх ботів
/setname	змінити ім'я бота
/setdescription	змінити опис бота
/setabouttext	змінити бота про інформацію
/setuserpic	змінити фотографію профілю бота
/setcommands	змінити список команд
/deletebot	видалити бота
/token	створити нового бота
/revoke	редагувати своїх ботів
/setinline	змінити ім'я бота
/setinlinegeo	змінити опис бота
/setinlinefeedback	змінити бота про інформацію
/setjoingroups	змінити фотографію профілю бота
/setprivacy	змінити список команд

Після того як ми отримали API токен, через який будуть йти HTTP запити до API Telegram, ми можемо починати розробку телеграм боту.

3.2. Опис програми та її алгоритми

Для того щоб описати дії які зможе зробити користувач користуючись чат-ботом, було розроблено UML діаграму прецедентів.

Функції системи представлені UML діаграмі прецедентів(Див. рис. 3.3)

Рисунок 3.2 - UML діаграма прецедентів

Діаграма прецедентів зображує, що користувач може:

– Створити власну анкету, яка буде відображатися іншим користувачам

– Переглядати анкети інших користувачів

– Вподобати анкету іншого користувача

– Користувач буде отримувати повідомлення коли хтось вподобає його анкету

Алгоритм роботи чат-боту є послідовністю того, як використовує бота користувач. При старті боту перевіряється наявність анкети у користувача. Якщо у користувача відсютня анкета, то йому запропонує створити анкету. Для створення анкети користувачу потрібно буде ввести свою стать, яку стать він шукає, вік, ім’я, міста, опис та фото. Після створення анети користувачу видасть головне меню, в якому він зможе або передивлюватися анкети інших користувачів, або редагувати свою анкету. При редагуванні доступна функція створити анкету спочатку, або відредагувати якийсь конкретний елемент. При перегляді анкети, користувач може або пропустити її і перейти до наступної або вподобати, після чого тій людині прийде сповіщення(Див. рис. 3.3)

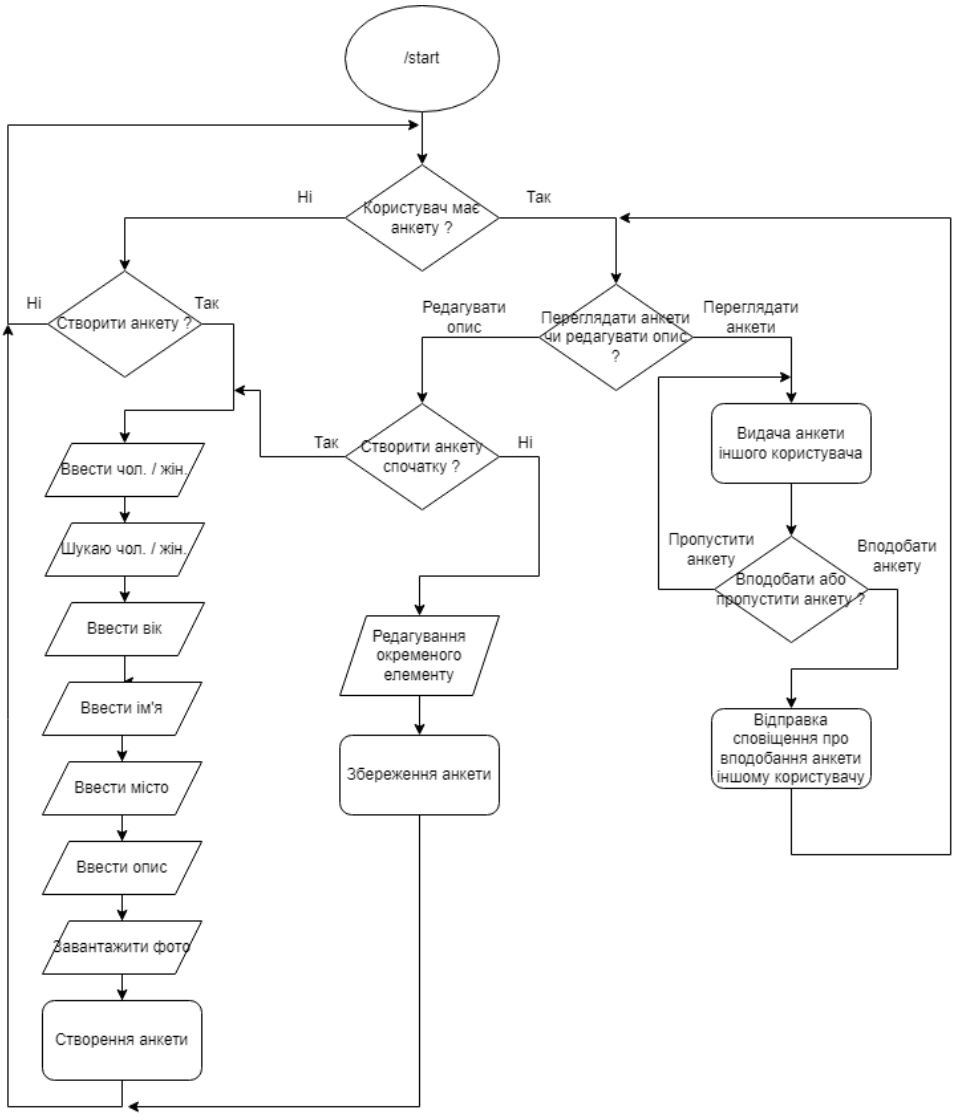


Рисунок 3.3 - Алгоритм роботи чат-боту

Для розробки телеграм бота потрібно ще розуміти як наш бот буде взаємодіяти з Telegram API та додатком Telegram (Див. рис. 3.4)

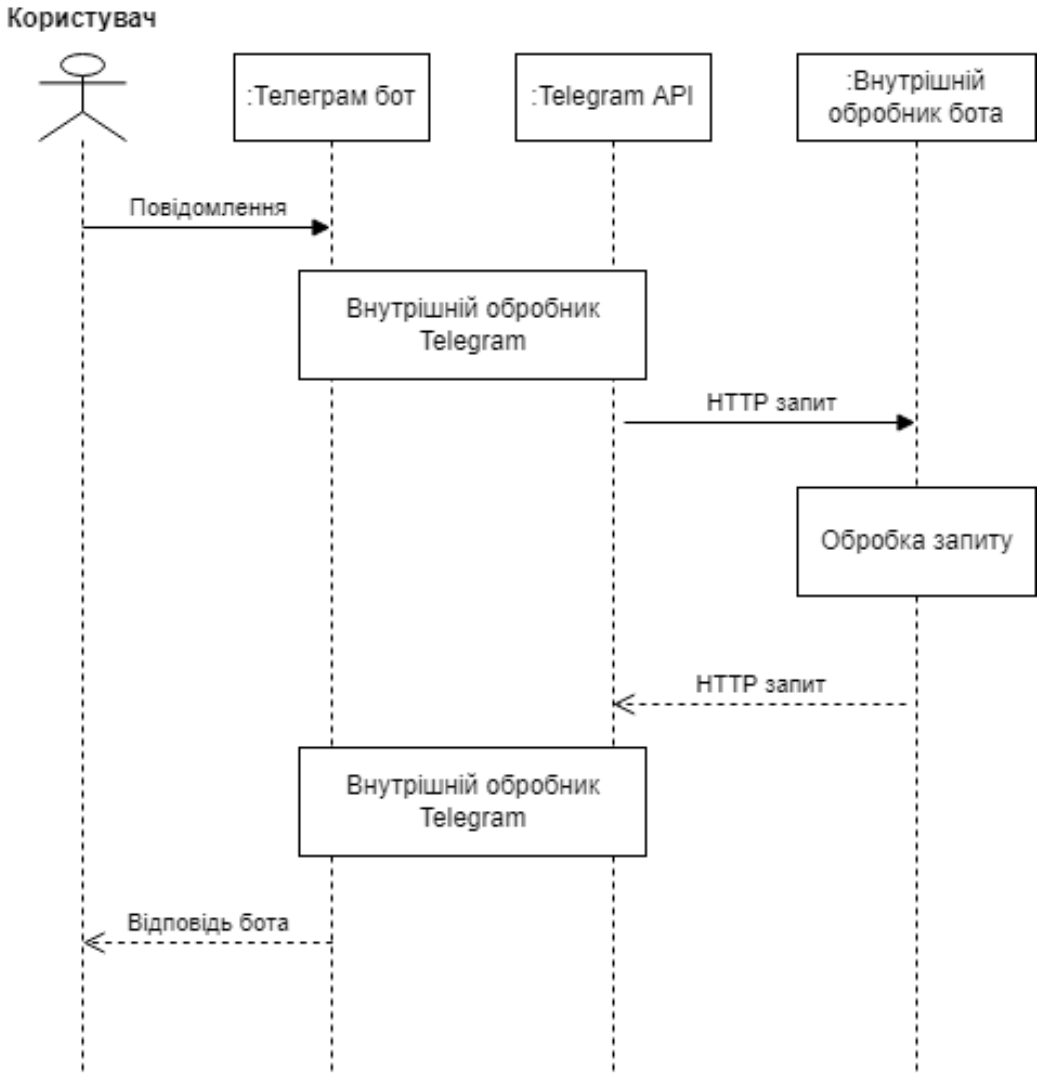


Рисунок 3.4 – Діаграма послідовності виконання запитів користувача

Коли користувач відправляє повідомлення телеграм боту, телеграм бот перенаправляє його як код до Telegram API. Після цього до нашого бота за допомогою API-токена надходить HTTP запит, який містить у собі інформацію про користувача, час, додаткові уточнення та його повідомлення. Після обробки запиту користувача наш код боту відправляє у відповідь HTTP запит, також за допомогою API-токена, де вже Telegram API віддає отриману відповідь від імені нашого телеграм бота.

Для зберігання анкет користувачів та інших даних для роботи бота використовується база даних. Для її проектування була розроблена діаграма структури бази даних(Див. рис. 3.5)

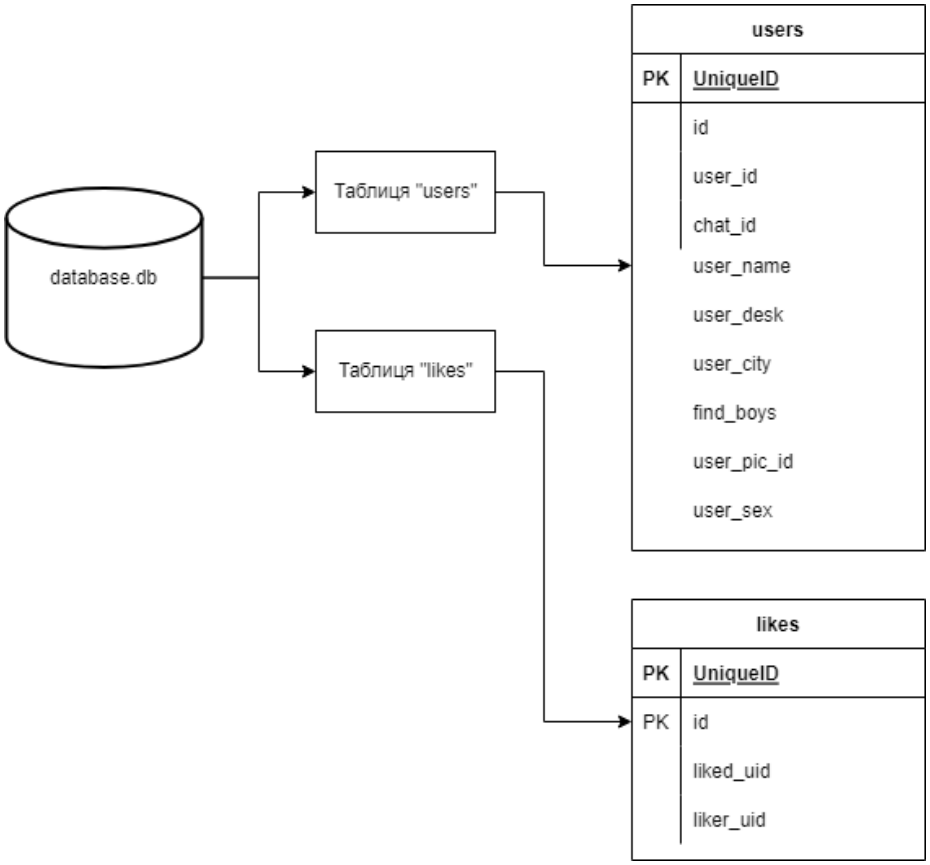


Рисунок 3.5 – Діаграма структури бази даних

Таблиця users зберігає анкети користувачів та інформацію про них. Поля user_id та chat_id зберігають інформацію про телеграм аккаунти користувачів, для розуміння кому та куди надсилати повідомлення. User_name, user_desk, user_city та user_pic_id зберігаються інформацію про анкету користувача, його ім’я, опис, місто та телеграм адресу його зображення. User_sex та find_boys означають стать користувача та кого він шукає.

3.3. Реалізація чат-боту

Початок взаємодії користувача та бота починається з команди «/start». Для того щоб обробити цю команду використовується обробник повідомлень «message_handler»(Див. рис. 3.5).

```
@dp.message_handler(commands=['start'])
async def send_welcome(message: types.Message):
    if(not db.users_exists(message.from_user.id)):
        go_reg = types.ReplyKeyboardMarkup(resize_keyboard=True)
        item1 = types.KeyboardButton("🔰 Реєстрація 🔰")
        go_reg.add(item1)

        await message.reply("Привіт, {0}!\nСхоже, ти мій новий користувач. Давай пройдемо
            невеличку реєстрацію.\n\nНатисність кнопку щоб продовжити" .format(message.from_user.
            first_name), reply_markup=go_reg)
        #db.add_user(message.from_user.id, u_lang, "yes", message.chat.id,
        message.from_user.first_name)
    else:
        reset_data(message.from_user.id)
        main_menu = types.ReplyKeyboardMarkup(resize_keyboard=True)
        search = types.KeyboardButton("🚀")
        my_anketa = types.KeyboardButton("📖")
        main_menu.add(search, my_anketa)

        return await message.answer("Головне меню:\n🚀 - Перегляд анкет\n📖 - Перегляд/редагувати
            свою анкету", reply_markup=main_menu)
```

Рисунок 3.6 – Код обробки запиту «/start»

Після команди «/start» ми перевіряємо наявність анкети користувача у базі даних та якщо анкети користувача у нас не має, ми привітаємо його як нового користувача та запропонуємо йому заповнити анкету. Якщо користувач існує то просто виведемо йому клавіатуру з головним меню. (Див. рис. 3.6)

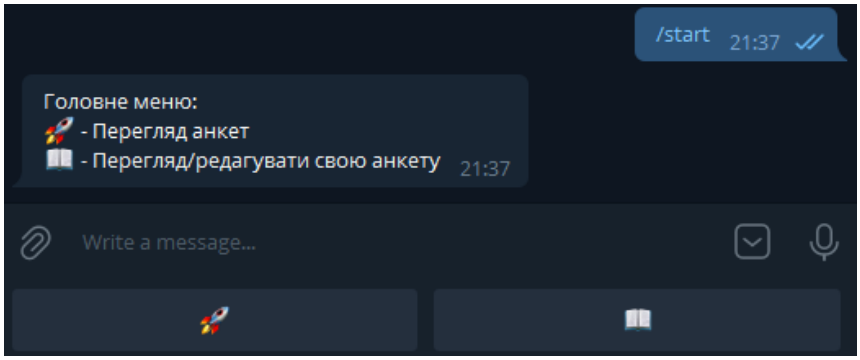


Рисунок 3.7 – Вигляд клавіатури головного меню

Для того щоб користувач міг надалі зручніше користуватися функціями бота, є можливість додавати різні клавіатури з кнопками дані з яких будуть обробляти обробники повідомлень «message_handler» де можна буде на окрему кнопку запрограмувати якусь дію. Клавіатура у месенджері телеграм представлена як об’єкт, який має певний набір характеристик, а текст кнопки представляє собою JSON-рядок. Серед параметрів кнопки є обов’язкове значення – текст кнопки, за допомогою якого ми можемо обробляти її натискання, можливість додавати вбудовані запити, можливість введення з клавіатури та можливість додання оплати певних функцій.

Клавіатура створюється за допомогою функції «ReplyKeyboardMarkup». Після чого створюються кнопки за допомгою функції «KeyboardButton». (Див. рис. 3.7)

```
class cMarkup:
    def mm(self):
        global markup_mm

        markup_mm = types.ReplyKeyboardMarkup(resize_keyboard=True)
        search = types.KeyboardButton("🚀")
        my_anketa = types.KeyboardButton("⬜")
        markup_mm.add(search, my_anketa)

    def anketa(self):
        global markup_anketa

        markup_anketa = types.ReplyKeyboardMarkup(resize_keyboard=True)
        new_ank = types.KeyboardButton("🔄")
        change_img = types.KeyboardButton("🖼")
        change_desk = types.KeyboardButton("🔲")
        change_insta = types.KeyboardButton("🌐")
        home = types.KeyboardButton("⬅")
        wath_anketas = types.KeyboardButton("🚀")
        markup_anketa.add(new_ank, change_img, change_desk, change_insta, home, wath_anketas)

    def use_anketa(self):
        global markup_anketa_use

        markup_anketa_use = types.ReplyKeyboardMarkup(resize_keyboard=True)
        like = types.KeyboardButton("❤")
        markup_anketa_use.add(like)
```

Рисунок 3.8 – Код створення клавіатури з кнопками

Після створення обробки початкового обробника користувачу потрібно буде створити власну анкету.

Обробляючи повідомлення за допомогою «message_handler», використовуючи послідовність запитань, ми зчитуємо введені повідомлення користувача за допомогою аргументу, що приходить у запиті, «Message». Після чого записуємо відповіді у тимчасову змінну.(Див. рис.3.8)

```python
@dp.message_handler()
async def messages_handler(message: types.Message):
    global go_reg_age_tf, go_reg_name_tf, go_reg_city_tf, go_reg_desk_tf, go_reg_img_tf,
        go_reg_findis_tf, go_reg_sex_tf, \
    go_reg_age, go_reg_name, go_reg_city, go_reg_desk, go_reg_findis, go_reg_sex, go_reg_imgpath,
    change_insta_tf, update_user_pic, update_user_desk


    user_id = message.from_user.id

    if message.text == "🚹 Реєстрація 🚺":
        go_reg_sex_tf[message.from_user.id] = True

        reg_sex = types.ReplyKeyboardMarkup(resize_keyboard=True)
        male = types.KeyboardButton("👦 Чоловік")
        female = types.KeyboardButton("👧 Жінка")
        reg_sex.add(male, female)

        return await message.answer("Ви: Чоловік / Жінка ?", reply_markup=reg_sex)



    if message.text == "👦 Чоловік":
        go_reg_sex_tf[message.from_user.id] = False
        go_reg_sex[message.from_user.id] = "male"
        reg_find_is = types.ReplyKeyboardMarkup(resize_keyboard=True)
        male = types.KeyboardButton("👦 Чоловіка")
        female = types.KeyboardButton("👧 Жінку")
        reg_find_is.add(male, female)
        return await message.answer("Кого шукаэмо ?", reply_markup=reg_find_is)

    if message.text == "👧 Жінка":
        go_reg_sex_tf[message.from_user.id] = False
        go_reg_sex[message.from_user.id] = "female"
        reg_find_is = types.ReplyKeyboardMarkup(resize_keyboard=True)
        male = types.KeyboardButton("👦 Чоловіка")
        female = types.KeyboardButton("👧 Жінку")
        reg_find_is.add(male, female)
        return await message.answer("Кого шукаэмо ?", reply_markup=reg_find_is)
```

Рисунок 3.9 – Код обробки заповнення анкети

Після того як користувач відповість на всі запитання, можна буде створити його анкету, але перед цим потрібно перевіряти данні на відповідність, наприклад що користувачу має бути більше 13 років та менше 100. (Див. рис. 3.9)

```
try:
    if go_reg_age_tf[message.from_user.id] == True:
        go_reg_age[message.from_user.id] = int(message.text)
        if go_reg_age[message.from_user.id] < 13:
            go_reg_age_tf[message.from_user.id] = True
            return await message.reply("Пробач, але тобі має бути більше 13-ти років..")

        elif go_reg_age[message.from_user.id] > 100:
            go_reg_age_tf[message.from_user.id] = True
            return await message.reply("Пробач, але тобі має бути меньше 100-а років..")

        go_reg_age_tf[message.from_user.id] = False
        go_reg_name_tf[message.from_user.id] = True

        #print("[id: {0}] age: {1}" .format(message.from_user.id, go_reg_age))
        return await message.reply("Добре. Тепер, як тебе звати?")
```

Рисунок 3.10 – Код перевірки валідності віку користувача

Після перевірки введених даних, обнуляємо тимчасові дані та можемо додавати анкету до бази даних. Після відправки даних до іншого класу, виводимо користувачу що реєстрація звершена та показуємо головне меню.

```
if message.text == "👌 Вірно":
    go_reg_age_tf[message.from_user.id] = False
    go_reg_name_tf[message.from_user.id] = False
    go_reg_sex_tf[message.from_user.id] = False
    go_reg_city_tf[message.from_user.id] = False
    go_reg_findis_tf[message.from_user.id] = False
    go_reg_desk_tf[message.from_user.id] = False
    go_reg_img_tf[message.from_user.id] = False

    db.add_user(message.from_user.id, message.chat.id, go_reg_age[message.from_user.id],
        go_reg_name[message.from_user.id], go_reg_desk[message.from_user.id], go_reg_city[
        message.from_user.id], go_reg_findis[message.from_user.id], sex=go_reg_sex[message.
        from_user.id])


    marukps.mm()
    await message.answer("Реєстрація завершена.")
    return await message.answer("Головне меню:\n🚀 - Перегляд анкет\n📖 - Перегляд/редагувати
        свою анкету", reply_markup=markup_mm)
```

Рисунок 3.11 – Код відправка даних анкети до іншого класу

Після прийняття даних, їх потрібно записати до таблиці, для цього підключаємось до бази даних, та за допомогою курсору, ініціалізуємо додавання даних до таблиці командою «INSERT INTO», вказуємо таблицю, рядки у які ми хочемо додатки та у наступному параметрі передаємо їх значення.

```
def add_user(self, user_id, chat_id=0, age=17, name="lovebot", desk=" ", city="Kyiv", find=
    False, user_pic="", sex="male"):
    with self.connection:
        return self.cursor.execute("INSERT INTO `users` (`user_id`, `chat_id`, `user_age`, `
            user_name`, `user_desk`, `user_city`, `find_boys`, `user_pic_id`, `user_sex`)
            VALUES\
            (?, ?, ?, ?, ?, ?, ?, ?, ?)", (\
            user_id, chat_id, age, name, desk, city, bool(find), user_pic, sex))
```

Рисунок 3.12 – Код запис анкети до бази даних

Після того як користувач створив анкету, він матиме змогу переглядати анкети інших користувачів, які так само раніше заповнювали анкети.

```
async def show_ankets(message):
    try:
        global liked_user_id, i_see_this_person


        result = list(db.get_sort_users(message.from_user.id, db.get_u_age(message.from_user.id), i_find="girl"))
        choice = random.choice(result)
        city = db.get_u_city(message.from_user.id)

        data = i_see_this_person

        if liked_user_id[message.from_user.id] != 0:
            for x in range(len(data[message.from_user.id])):
                #print("data[message.from_user.id][x] = {0}" .format(data[message.from_user.id][x]))
                if choice[1] == data[message.from_user.id][x]:
                    print("Skip: {0}" .format(choice[1]))
                    return await show_ankets(message)

        succ_chance = fuzz.token_set_ratio(choice[6], city)
        #print("{0} {1} || {2} = {3} / {4} %" .format(choice[4], choice[3], city, choice[6], succ_chance))
        if succ_chance > 75:
            try:
                image = open("./files/images/user_pic/"+str(choice[1])+".jpg", 'rb')

            except FileNotFoundError:
                image = open("./files/images/user_pic/"+str(choice[1])+".png", 'rb')

            liked_user_id[message.from_user.id] = choice[1]
            marukps.use_anketa()
            return await message.reply_photo(image, "{0}, {1} років - {2}\n{3}" .format(\
                choice[4], choice[3], choice[6], choice[5]), reply_markup=markup_anketa_use)

        elif fuzz.WRatio(choice[6], city) > 80:
            try:
                image = open("./files/images/user_pic/"+str(choice[1])+".jpg", 'rb')

            except FileNotFoundError:
                image = open("./files/images/user_pic/"+str(choice[1])+".png", 'rb')

            return await message.reply_photo(image, "{0}, {1} років - {2}\n{3}" .format(\
                choice[4], choice[3], choice[6], choice[5]))

        elif succ_chance > 50 and succ_chance < 75:
            for data in datas:
                if fuzz.token_set_ratio(data[1], choice[6]) > 80:
                    print("(token_set_ratio) ?? > {0} = {1} | {2} %" .format(data[1], choice[6], fuzz.token_set_ratio(data[1], city)))
                    try:
                        image = open("./files/images/user_pic/"+str(choice[1])+".jpg", 'rb')

                    except FileNotFoundError:
                        image = open("./files/images/user_pic/"+str(choice[1])+".png", 'rb')

                    return await message.reply_photo(image, "{0}, {1} років - {2}\n{3}" .format(\
                        choice[4], choice[3], choice[6], choice[5]))

                if fuzz.WRatio(data[1], choice[6]) > 80:
                    print("(WRatio) ?? > {0} = {1} | {2} %" .format(data[1], choice[6], fuzz.token_set_ratio(data[1], city)))
                    try:
                        image = open("./files/images/user_pic/"+str(choice[1])+".jpg", 'rb')

                    except FileNotFoundError:
                        image = open("./files/images/user_pic/"+str(choice[1])+".png", 'rb')

                    return await message.reply_photo(image, "{0}, {1} років - {2}\n{3}" .format(\
                        choice[4], choice[3], choice[6], choice[5]))

        else:
            await show_ankets(message)
```

Рисунок 3.13 – Код видачі анкети іншого користувача

Користувачів будемо видавати з того самого міста який було вказано при реєстрації користувачам, але щоб уникнути помилок користувачів, ми будемо використовувати модуль «fuzzywuzzy» для того щоб знаходити схожість між словами, як наприклад: «Київ» та «Киев». Після показу анкета буде вибір або показати наступну або вподобати її.

```
if message.text == "❤️":
    global liked_user_id, i_see_this_person
    if liked_user_id[message.from_user.id] != 0:

        db.add_like(liked_user_id[message.from_user.id], message.from_user.id)
        likes = list(db.get_user_likes(user_id=liked_user_id[message.from_user.id]))
        if len(likes) == 1:
            await bot.send_message(db.get_u_chatid(liked_user_id[message.from_user.id]), "Ваша
                анкета сподобалась {0} користувачу(ам)" .format(len(likes)))
        elif len(likes) > 1:
            await bot.send_message(db.get_u_chatid(liked_user_id[message.from_user.id]), "Ваша
                анкета сподобалась {0} користувачу(ам)" .format(len(likes)))


        temp.append(int(liked_user_id[message.from_user.id]))
        i_see_this_persons[message.from_user.id] = {}
        i_see_this_person = {fruit : temp for fruit in i_see_this_persons}
        return await show_ankets(message)
```

Рисунок 3.14 – Код вподобання анкети

Після того як користувач знайшов анкету людини яка йому подобається, він може її вподобати, після чого йому користувачу прийде сповіщення про вподобання. Для «лайків» була створена окрема таблиця у яку записується людина яка поставила «лайк» та той хто його отримав.

```
def add_like(self, liked_id, liker_id):
    with self.connection:
        return self.cursor.execute("INSERT INTO `likes` (`liked_uid`, `liker_uid`) VALUES(?, ?)
            ", (liked_id, liker_id))
```

Рисунок 3.15 – Код додання вподобання до бази даних

3.4. Тестування чат-бота

Для того щоб знайти бота потрібно у пошуку вбити його назву «Dut Dating Bot» або телеграм-адресу «@DutDatingBot».

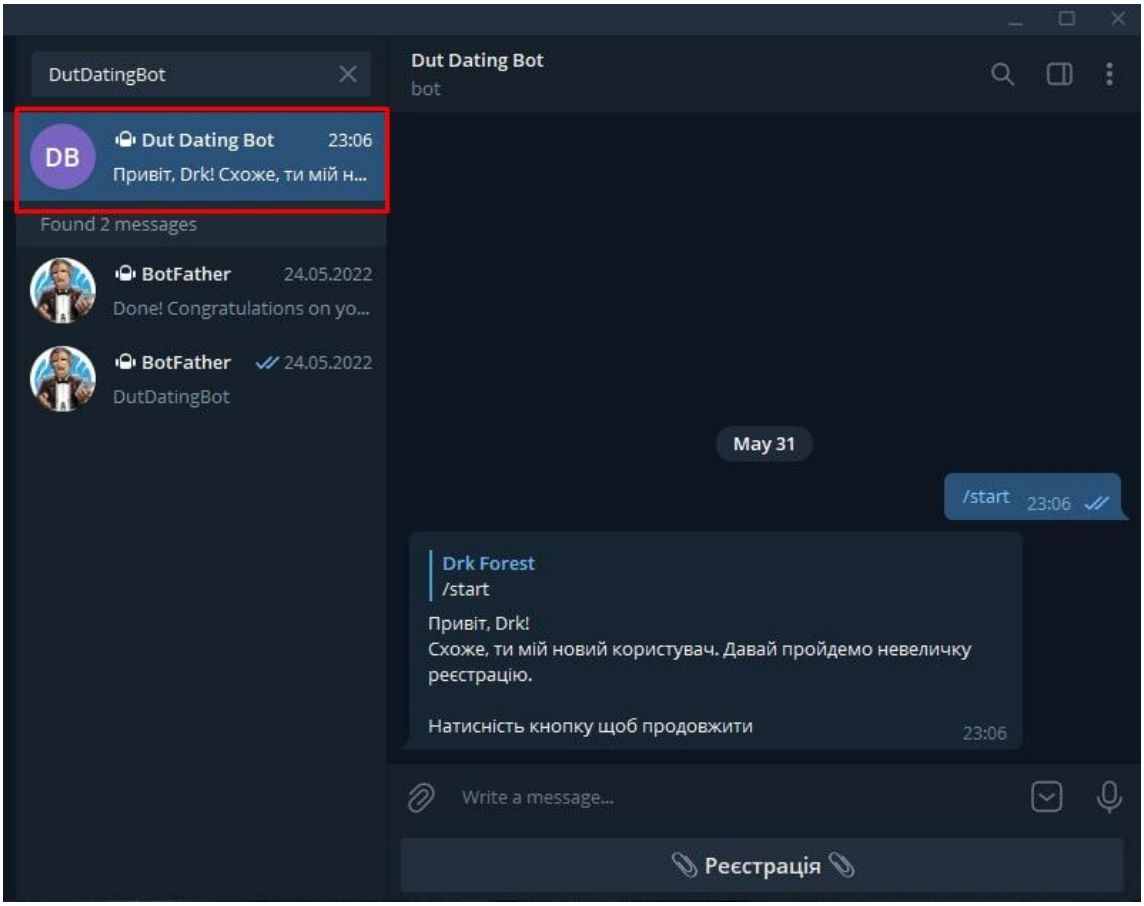


Рисунок 3.16 – Пошук бота

Після знаходження та запуску бота автоматично виконається команда «/start», бот привітає нового користувача та запропонує пройти реєстрацію.

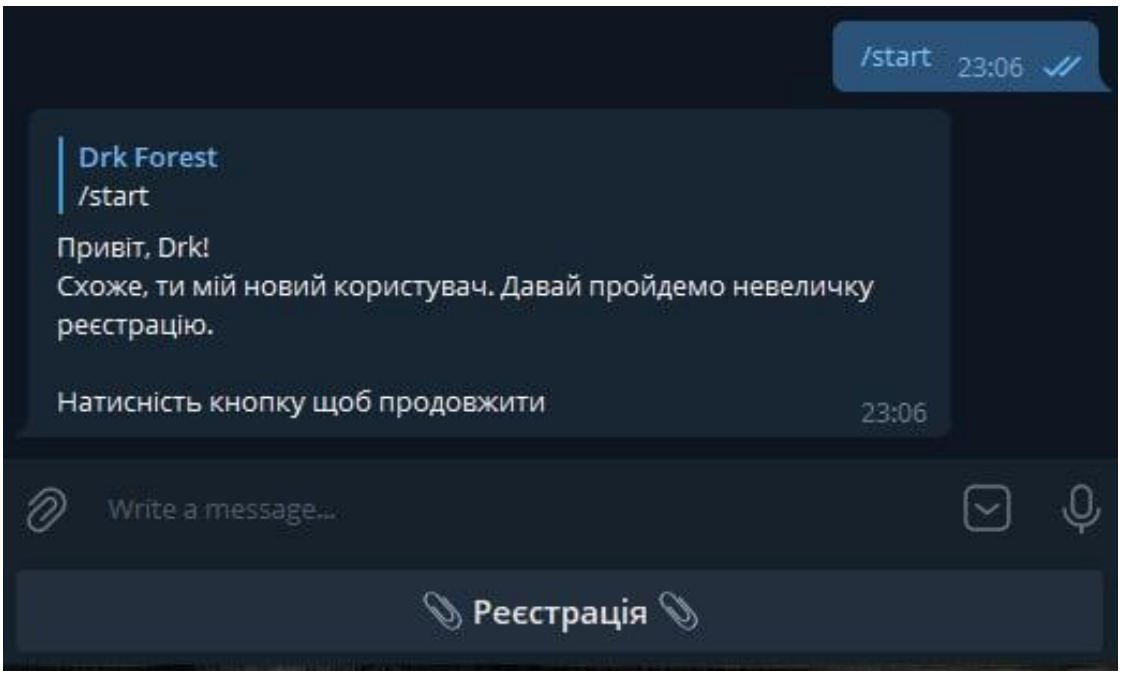


Рисунок 3.17 – Початок реєстрації

Для реєстрації користувачу потрібно буде відповісти на запитання бота щоб заповнити анкету.

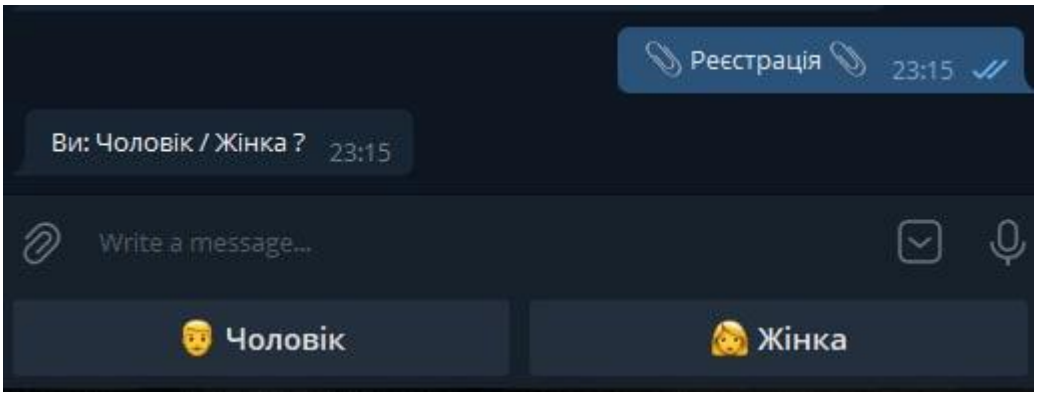


Рисунок 3.18 – Стать та кого шукаємо

Після вводу статі користувача та статі для пошуку, потрібно буде ввести кількість років користувача, йому має бути більше тринадцяти та менше ста років.

Рисунок 3.19 – Кількість років користувача

Після цього потрібно буде вказати ім’я користувача, місто проживання та описати себе в декількох словах, або можна залишити опис пустим

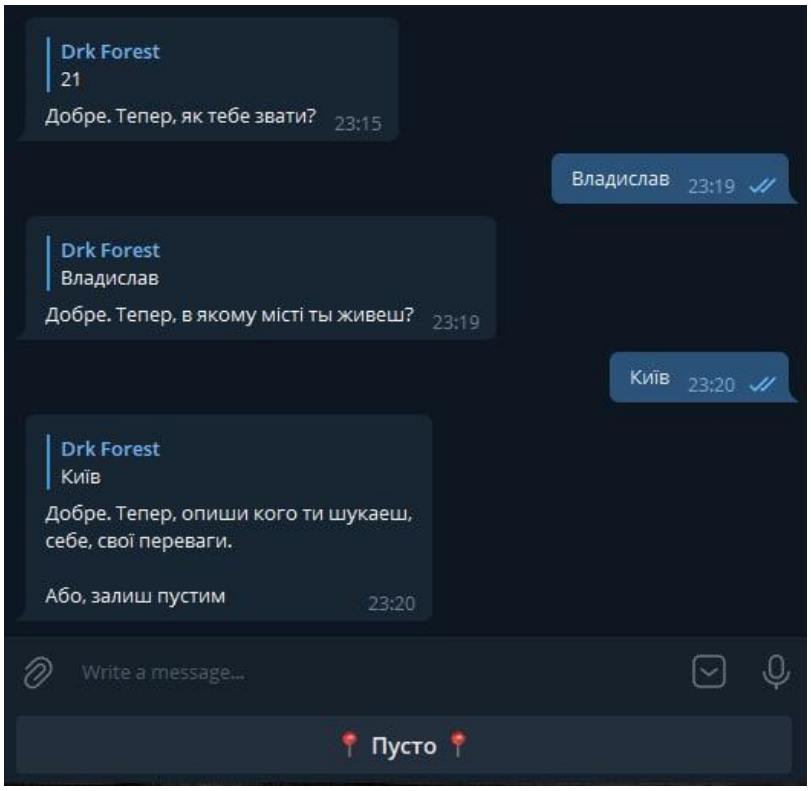


Рисунок 3.20 – Стать та кого шукаємо

Потім, після заповнення текстових даних, потрібно буде завантажити свій аватар або фотографію, яку будуть бачити інші користувачі.

Рисунок 3.21 – Завантаження зображення

Після чого бот покаже майбутню анкету користувачу та запитає чи все вірно або потрібно переробити.

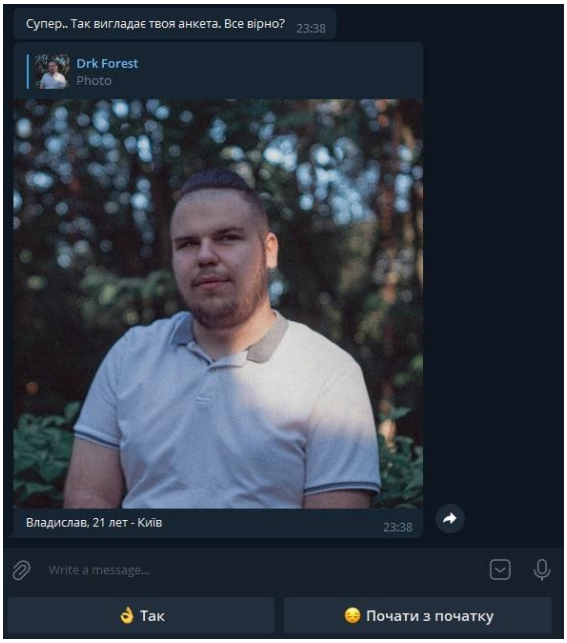


Рисунок 3.22 – Перевірка правильності анкети

Після закінчення реєстрації, користувач потрапляє у головне меню, у якому користувач може вибрати чи переглядати анкети інших користувачів або, за потреби, переглянути або редагувати свою анкету.

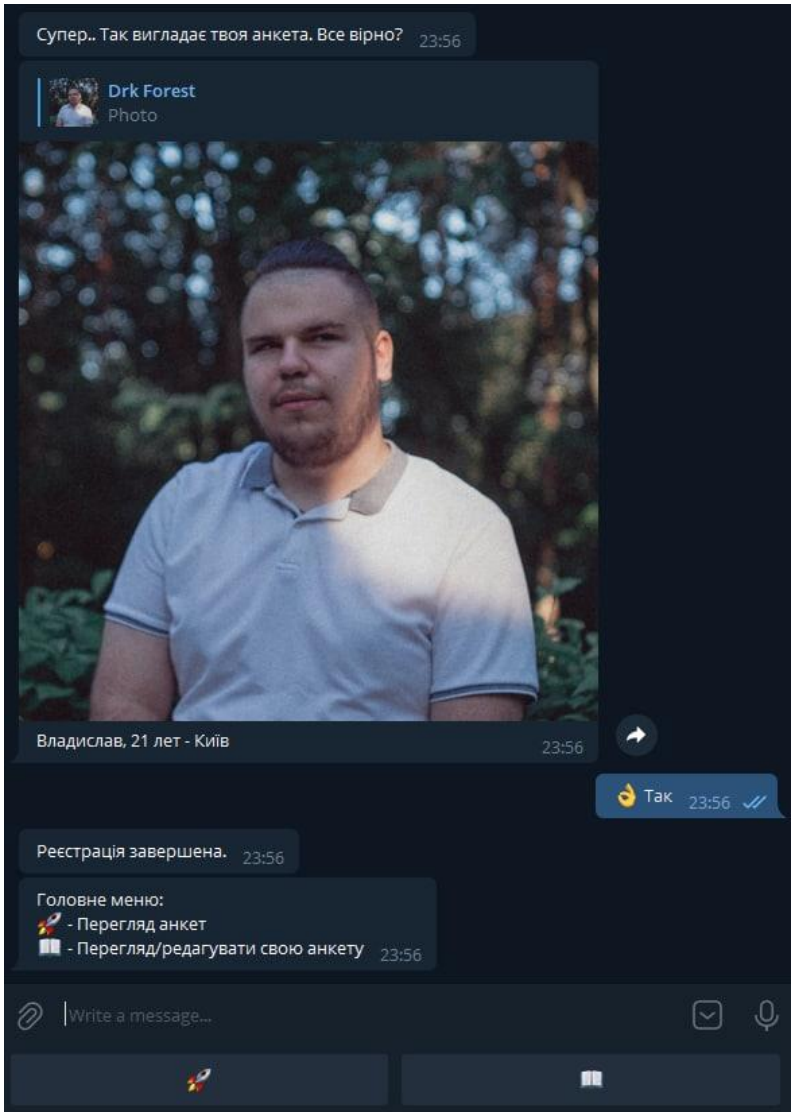


Рисунок 3.23 – Головне меню

Якщо користувачу потрібно переглянути або відредагувати свою анкету, у наступному меню у нього буде вибір: заповнити анкету спочатку, редагувати зображення анкети, редагувати опис або вийти до головного меню.

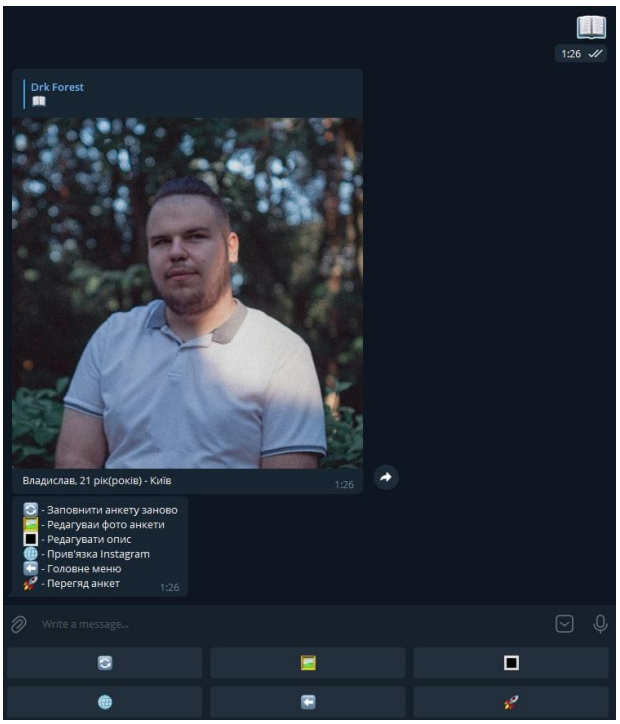


Рисунок 3.24 – Меню редагування анкети

Якщо користувач вибирає перегляд анкет інших користувачів, то йому будуть видаватися анкети які підходять за його вказаним місцем проживання.

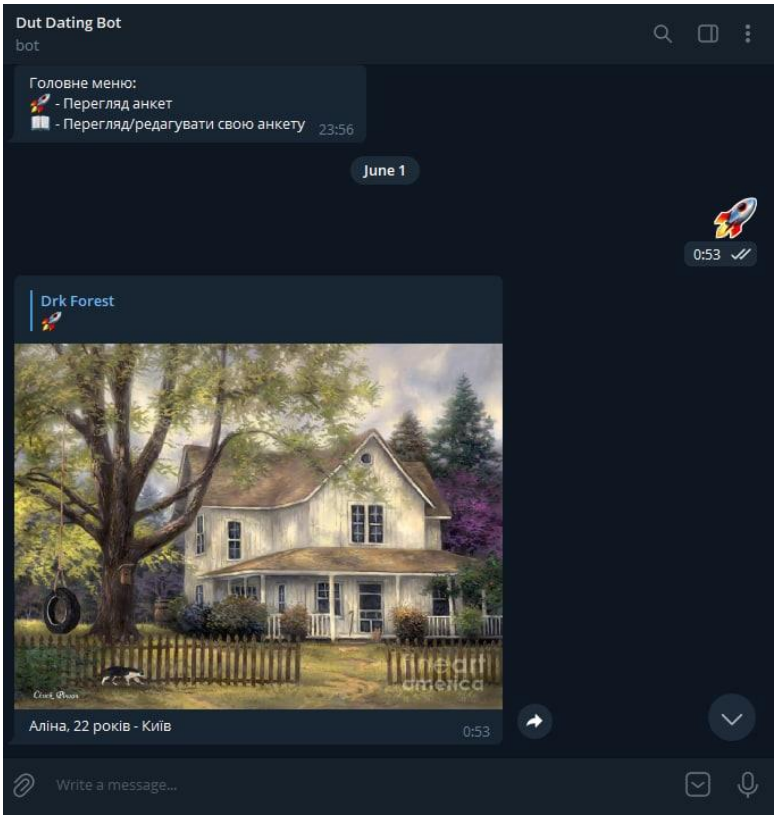


Рисунок 3.25 – Перегляд анкет

Якщо користувачу сподобається якась анкета, він може її вподобати та цій людині прийде сповіщення від бота, що її анкету вподобали.

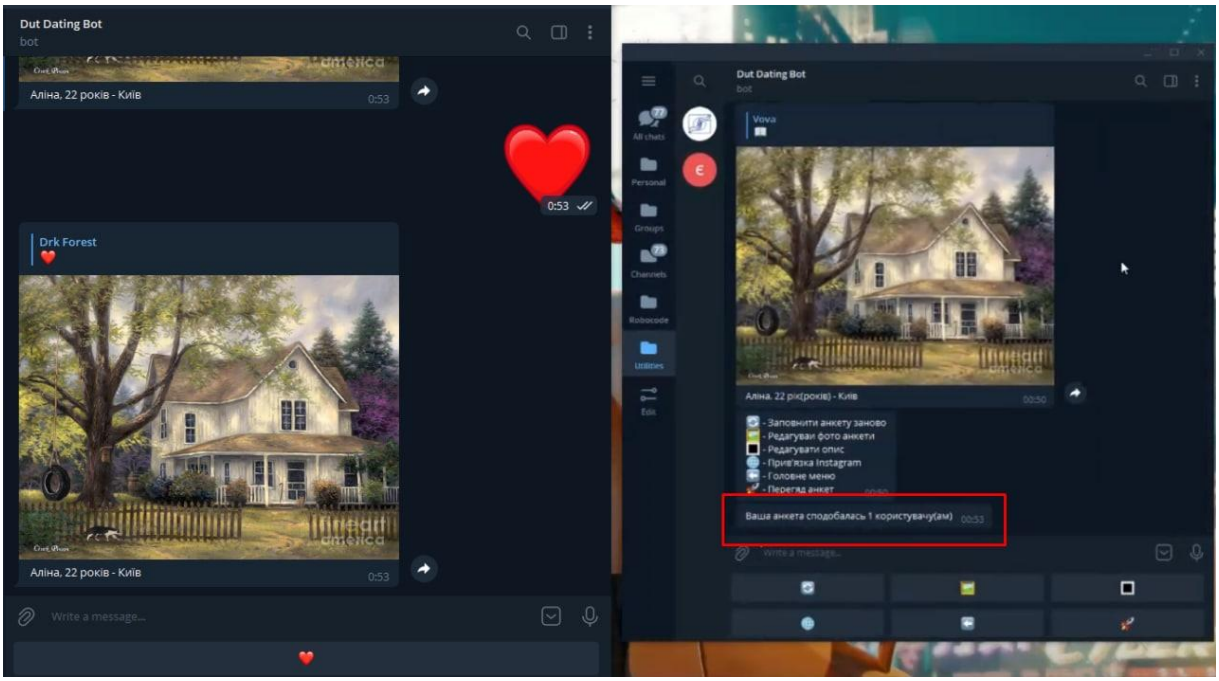


Рисунок 3.26 – Оповіщення про впродобання

ВИСНОВКИ

Цей проект був спрямований на аналіз та розробку чат-боту «Dut Dating Bot». Було проаналізовано та визначено актуальність теми проекту та її наукову новизну шляхом порівняння можливостей та варіантів онлайн знайомства. Було досліджено та порівняно між собою популярні бібліотеки для розробки телеграм ботів, де виявили яка з них більше підходить під цей проект.

1. Було розроблено чат-бот, де було реалізовано основні потреби для онлайн знайомства, де користувач може створити власну анкету, переглядати анкети інших користувачів та вподобати їх, після чого їм прийде сповіщення.

Було обрано інструменти для розробки чат-боту, такі як PyCharm та бібліотеку aiogram, описано їх переваги.

Для того щоб спроектувати чат-бота, було проаналізовано варіанти онлайн знайомства, та створено набір вимог у вигляді UML-діаграм.

В ході проектування чат-боту було розроблено алгоритм роботи, де було читко виділено основі функції роботи чат-боту.

2. Описано програмні засоби та інструменти, які було застосовано для розробки чат-боту. Проаналізовано яка бібліотека є найбільш вдалою для поставленої задачі

3. Високу ефективність роботи було досягнено за допомогою використання бібліотеки aiogram, що дозволило проектувати чат-бота за допомогою асинхронного програмування.

4. За допомогою месенджеру телеграм, відпадає потреба окремого проектування інтерфейсу, що зменшує витрати часу на розробку.

5. Роботу було апробовано серед висококваліфікованих спеціалістів в сфері Python development та Data scientist, у апробації взяли участь 2 осіб з рівнем знань на посадах Middle та Project manager. Було проаналізовано переваги та недоліки системи. Також було спроектовано план покращень для наступних версій.

6. Виявлено, що основнимиперевагами гри є:

- Простота використання та не потрібність додаткового встановлення;
- Можливість інтегрування та переробки під різні потреби;
- Розробка під потреби використання під великими навантаженнями та великим напливом користувачів;

Результати дослідження бакалаврської роботи апробовані на всеукраїнських науково-технічних конференціях: «Застосування програмного забезпечення в ІКТ» та «Сучасні інтелектуальні інформаційні технології в науці та освіті», на які подавались тези «Використання чат-ботів для бізнесу та маркетингу» та «Порівняння бібліотек для веб-скрапінгу даних мовою програмування Python».

ПЕРЕЛІК ПОСИЛАНЬ

1. Python (programming language) – [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Python_(programming_language)
2. Сайти для онлайн знайомств в Україні– [Електронний ресурс] – Режим доступу: https://eduzorro.com/5-najkrashhih-sajtiv-dlya-serjoznih-stosunkiv-v-ukraini/
3. pyTelegramBotAPI документація – [Електронний ресурс] – Режим доступу: https://github.com/eternnoir/pyTelegramBotAPI#asynctelebot
4. AIOGram документація – [Електронний ресурс] – Режим доступу: https://github.com/aiogram/aiogram
5. python-telegram-bot документація – [Електронний ресурс] – Режим доступу: https://github.com/python-telegram-bot/python-telegram-bot
6. Telethon документація – [Електронний ресурс] – Режим доступу https://github.com/LonamiWebs/Telethon
7. Порівняння MariaDB та MySQL 2019 [Електронний ресурс]:– Режим доступу: https://netpoint-dc.com/blog/mariadb-mysql-2019/.
8. Telegram-bot-api VS aiogram [Електронний ресурс]:– Режим доступу: https://www.libhunt.com/compare-tdlib--telegram-bot-api-vs-aiogram
9. Посібник з мови програмування Python [Електронний ресурс]:– Режим доступу: https://metanit.com/python/tutorial/
10. Telegram Bot API документація [Електронний ресурс]:– Режим доступу: https://core.telegram.org/bots/api
11. Telegram Bot на Python3 та aiogram [Електронний ресурс]:– Режим доступу: https://surik00.gitbooks.io/aiogram-lessons/content/chapter1.html

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО- НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

РОЗРОБКА ТЕЛЕГРАМ БОТУ ДЛЯ ЗНАЙОМСТВ МОВОЮ ПРОГРАМУВАННЯ PYTHON

Виконав студент 4 курсу

групи ПД-43
Швецов Владислав Ігорович

Керівникроботи

к.т.н., доц. кафедри ІПЗ, Трінтіна Наталія Альбертівна

Київ – 2022

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи – Спрощення онлайн знайомств з людьми за допомогою телеграм боту

Об’єкт дослідження– Спрощення та процес онлайн знайомства людини.

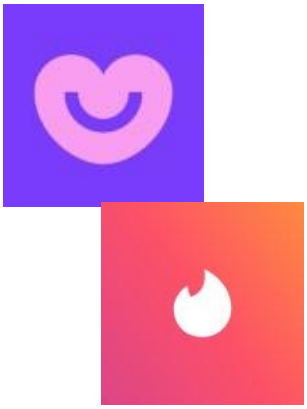
Предмет дослідження– Телеграм бот для онлайн знайомств.

Аналоги

Сайти

OBYAVA.UA

Додатки



Боти

3

Порівняння з аналогами

Показник	**Сайти**	**Додатки**	**Боти**
Платформи	Всі платформи з браузером до виходом у мережу.	Android, IOS	Інтеграція в додаток месенджеру(залежить від платформмесенджеру)
Нескінченналента запропонованих знайомств	-	+	+
Функція«люди поруч»	-	+	-
Фільтридля пошуку	+	+	-
Можливість користування без реєстрації	-	-	+(окрім, реєстрації месенджеру)
Зручний та актуальнийчат	-	-	+
Зручність повсякденного використання	-	+	+

4

ТЕХНІЧНЕ ЗАВДАННЯ

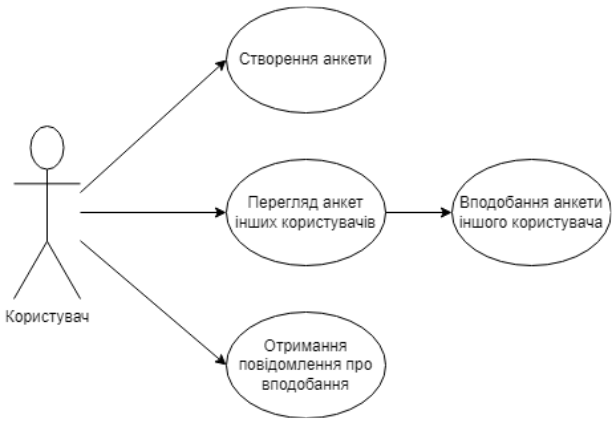
1. Дослідити та обрати бібліотеку для розробки телеграм ботів
2. Спроектувати алгоритм роботи телеграм боту
3. Розробити систему створення анкет користувачів
4. Розробити можливість перегляду та вподобання анкет
5. Розробити чат-бота та його основні функції

5

ЗАСОБИ РЕАЛІЗАЦІЇ

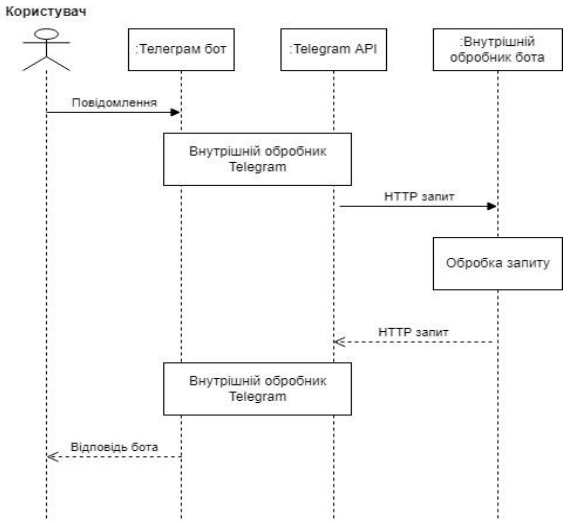
6

UML Діаграма прецедентів



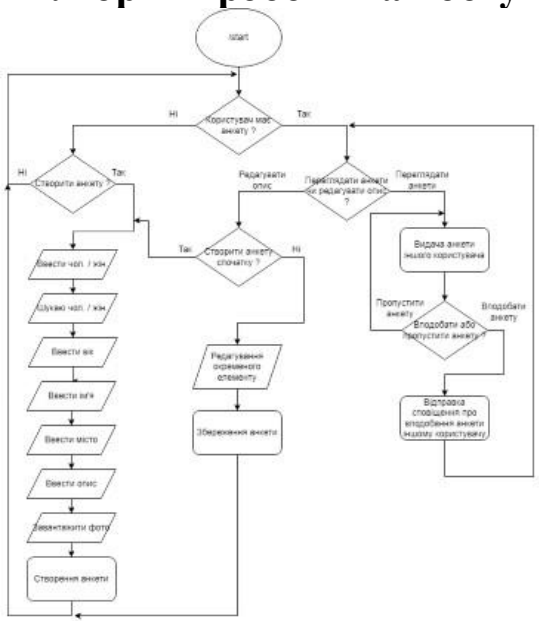
7

Архітектура телеграм боту



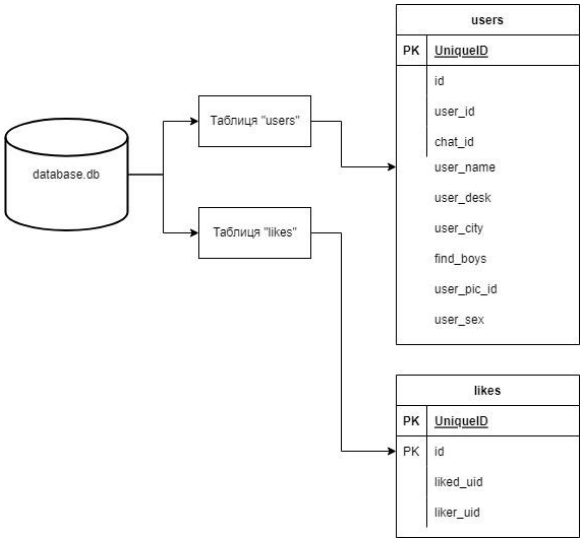
8

Алгоритм роботи чат-боту



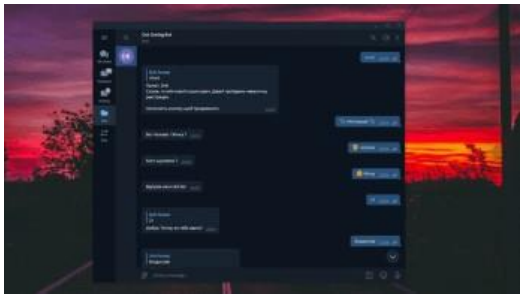
9

Діаграма структури бази даних

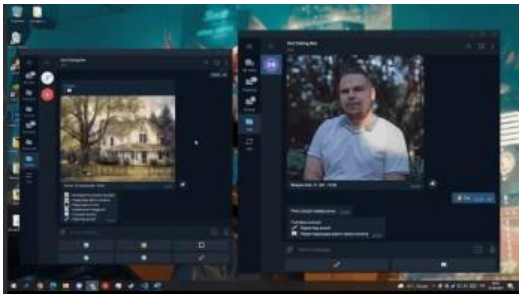


10

Приклад роботи чат боту



Створення анкети



Пошук та вподобання анкети

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Швецов В.І. Використання чат-ботів для бізнесу та маркетингу: Матеріали науково-технічної конференції «Застосування програмного забезпечення в ІКТ». Збірник тез.\- 20.04.2022, ДУТ, м.Київ, ст 31-33

Швецов В.І. Порівняння бібліотек для веб-скрапінгу даних мовогою програмування Python: Матеріали II науково-технічної конференції «Сучасні інтелектуальні інформаційні технології в науці та освіті». Збірник тез.\- 05.04.2022, ДУТ, м.Київ, ст 84-86

ВИСНОВКИ

1. Досліджено та обрано бібліотеку для розробки телеграм ботів
2. Спроектувано алгоритм роботи телеграм боту
3. Розроблено систему створення анкет користувачів
4. Розроблено можливість перегляду та вподобання анкет
5. Розроблено телеграм чат-бота та його основні функції

ДЯКУЮ ЗА УВАГУ!