**Київський національний торговельно-економічний університет**

**Кафедра комп'ютерних наук та інформаційних систем**

# ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

## «Розробка інформаційної системи обліку зауважень та побажань користувачів»


Студента 4 курсу, 13 групи,

спеціальності
122 «Комп'ютерні науки» _____ Тимошенко Євгеній Олександрович

*підпис студента*

Науковий керівник
кандидат фізико-математичних наук, _____ Філімонова Тетяна Олегівна

*підпис керівника*

Гарант освітньої програми
кандидат технічних наук, доцент _____ Демідов Павло Георгійович

*підпис керівника*


**Київ 2021**

**Київський національний торговельно-економічний університет**

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»

**Затверджую**
Зав. кафедри _____Пурський О.І.
«18» грудня 2020р.

**Завдання**
**на випускну кваліфікаційну роботу студенту**

**Тимошенко Євгеній Олександрович**
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи (проекту)
*«Розробка інформаційної системи обліку зауважень та побажань користувачів»*
Затверджена наказом ректора від *«15» грудня 2020 р. № 3780*
2. Строк здачі студентом закінченої роботи *31 травня2021 року*
3. Цільова установка та вихідні дані до роботи
*Мета роботи: обгрунтування та розробка інформаційної системи обліку зауважень та побажань користувачів*
*Об'єкт дослідження: процес проектування системи обліку зауважень та побажань користувачів*
*Предмет дослідження: засоби створення інформаційної системи обліку зауважень та побажань користувачів*
4. Перелік графічного матеріалу_____

_____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

| Розділ | Консультант (прізвище, ініціали) | Підпис, дата | |
|---|---|---|---|
| | | Завдання видав | Завдання прийняв |
| 1 | Філімонова Т.О. | 22.12.2020 р. | 22.12.2020 р. |
| 2 | Філімонова Т.О. | 22.12.2020 р. | 22.12.2020 р. |
| 3 | Філімонова Т.О. | 22.12.2020 р. | 22.12.2020 р. |

6. Зміст випускної кваліфікаційної роботи (проекту) (перелік питань за кожним розділом)

*ВСТУП*

*РОЗДІЛ 1.Сучасне проектування інформаційних систем*

*1.1. Роль інформаційних систем в діяльності підприємства*

*1.2. Аналіз етапів проектування інформаційних систем*

*1.3. Огляд підходів щодо проектуання інформаційних систем*

*РОЗДІЛ 2. Організація розробки інформаційної системи обліку зауважень та побажань користувачів*

*Висновки по розділу*

*2.1. Розробка концепції інформаційної системи*

*2.2. Модель інформаційної системи обліку зауважень та побажань користувачів*

*2.3. Алгоритм створення інформаційної системи*

*Висновки по розділу*

*РОЗДІЛ 3. Реалізація проектування інформаційної системи обліку зауважень та побажань користувачів*

*3 .1.Програмна реалізація проекту*

*3.2. Апробація результатів дослідження*

*Висновки по розділу*

*ВИСНОВКИ*

*СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ*

*Додатки*

7. Календарний план виконання роботи

| № Пор. | Назва етапів випускної кваліфікаційної роботи | Строк виконання етапів роботи | |
|---|---|---|---|
| | | За планом | фактично |
| 1 | 2 | 3 | 4 |
| 1 | Вибір теми випускної кваліфікаційної роботи | 01.10.2020 | 01.10.2020 |
| 2 | Розробка та затвердження завдання на випускну кваліфікаційну роботу | 18.12.2020 | 15.12.2020 |
| 3 | Вступ | 03.02.2021 | 03.02.2021 |
| 4 | РОЗДІЛ 1. Сучасне проектування інформаційних систем | 26.02.2021 | 26.02.2021 |
| 5 | РОЗДІЛ 2. Організація розробки інформаційної системи обліку зауважень та побажань користувачів | 06.04.2021 | 06.04.2021 |
| 6 | РОЗДІЛ 3. Реалізація проектування інформаційної системи обліку зауважень та побажань користувачів | 12.05.2021 | 12.05.2021 |
| 7 | Висновки | 14.05.2021 | 14.05.2021 |

| | | | |
|---|---|---|---|
| 8 | Здача випускної кваліфікаційної роботи на кафедру науковому керівнику | 20.05.2021 | 20.05.2021 |
| 9 | Попередній захист випускної кваліфікаційної роботи | 26.05.2021 | |
| 11 | Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи | 27.05.2021 | |
| 12 | Представлення готової зшитої випускної кваліфікаційної роботи на кафедру | 31.05.2021 | |
| 13 | Публічний захист випускної кваліфікаційної роботи | За розкладом роботи ЕК | |

8. Дата видачі завдання «22» грудня 2020 р.

9. Керівник випускної кваліфікаційної роботи (проекту)
Філімонова Т.О _____
*(прізвище, ініціали, підпис)*

10. Гарант освітньої програми Демідов П.Г. _____
*(прізвище, ініціали, підпис)*

11. Завдання прийняв до виконання студент-дипломник
Тимошенко Є.О. _____
*(прізвище, ініціали, підпис)*

12. Відгук керівника випускної кваліфікаційної роботи (проекту)
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Керівник випускної кваліфікаційної роботи (проекту)
_____ 31.05.2021 р.
*(підпис, дата)*

**13. Висновок про випускну кваліфікаційну роботу (проект)**

Випускна кваліфікаційна робота (проект) студента _____
*(прізвище, ініціали)*
може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми _____Демідов П.Г.
*(підпис, прізвище, ініціали)*

Завідувач кафедри _____Пурський О.І.
*(підпис, прізвище, ініціали)*

«_____»_____2021 р.

# ЗМІСТ

## Анотація

У випускній кваліфікаційній роботі здійснено комплексну розробку моделей та автоматизованої інформаційної системи обліку побажань та зауважень користувачів з метою підвищення якості програмного забезпечення. Теоретично обґрунтовано основні положення формування та запропоновано концепцію створення інформаційної системи обліку відгуків користувачів. Розроблено метод автоматизованого обліку комплексної оцінки відгуків користувачів.


**Ключові слова:** автоматизована інформаційна система, чат-бот, моніторинг відгуків користувачів.



## Anotation

The graduation qualification work is devoted to the development of the model and the automated information system of the account of wishes and remarks of users for the purpose of increase of quality of the software is carried out. The main provisions of formation are theoretically substantiated and the concept of creation of information system of the account of responses of users is offered. A method of automated accounting of complex evaluation of user feedback has been developed.


**Keywords:** automated information system, chatbot, monitoring of user feedback.

# ВСТУП

Актуальність роботи: якщо взяти у приклад стартап або молодий бізнес, то для нього має значення кожен клієнт. І хоча можна докладати всіх зусиль для задоволення побажань клієнтів, незадоволені клієнти завжди будуть. І замість того щоб сприймати їх в багнети або ховати голову в пісок, заперечуючи існування проблеми, потрібно усвідомити, що скарга клієнта - це шанс стати краще.

Мета роботи: обґрунтування та розробка інформаційної системи обліку зауважень та побажань користувачів.

Досягнення мети обумовило необхідність вирішення таких завдань:

- аналітичне дослідження ролі інформаційних систем в діяльності підприємства
- розробки концепції інформаційної системи
- розробки моделі програмного забезпечення
- програмну реалізацію автоматизованої інформаційної системи

Предмет дослідження: засоби створеннння інформаційної системи обліку зауважень та побажань користувачів.

Об’єкт дослідження: процес моніторингу зауважень та побажань користувачів.

Методи досліджень: загальнонауковий аналітичний метод, системний метод, методи теорії бази даних, метод програмування.

Інформаційне забезпечення: дослідженню питань по роботі з відгуками користувачів присвячена значна кількість статтей і праць згідно з даними досліджень яких, в середньому кожна людина розповідає 15 людям про свій негативний досвід спілкування з компанією. Виходить, якщо було 10 скарг, то цілком ймовірно, що бінес втратив 150 потенційних клієнтів. Близько 70% покупців перед покупкою звертають увагу на відгуки та досвід покупок інших покупців, тому відгуки позитивно впливають на показник конверсії і є просто необхідною складовою для будь-якого сучасного інтернет-магазина [9-10].

Ще варто відзначити, що якщо скарги клієнтів вирішуються протягом 5 хвилин, то ті з більшою часткою ймовірності зроблять у вас покупку в майбутньому. Як бачимо, незадоволеного клієнта не так складно перетворити в щедрого покупця. Тому треба враховувати незадоволених клієнтів в бізнес-стратегії, продумувати заздалегідь, як потріно реагувати на їхні скарги, і вчитися на помилках, щоб в майбутньому пом'якшити потенційний збиток компанії.

З клієнтами, які незадоволені, але не скаржаться, набагато складніше працювати, оскільки бізнес ніколи не дізнається, як їх зробити задоволеними. Тому така висока цінність тих, хто відгукнувся і поділився думкою. Потрібно ставити їх в пріоритет.

Щасливі клієнти - це чудово, але це не дасть інформацію, яка стане поштовхом до розвитку. Незадоволені клієнти - це актив. Вплив незадоволених клієнтів величезний, тому варто зробити все можливе, щоб запросити цих клієнтів до діалогу. Ставитися до них потріно з усією увагою, швидко реагувати на їхні зауваження, прислухатися до їхніх порад і тримати їх в курсі того, які дії було зроблено для виправлення ситуації. Незважаючи на те, що в сучасному світі прийнято клієнтам в першу чергу писати на пошту або месенджери, дзвінок по телефону все ще може бути ефективним способом зв'язку. Особливо якщо бізнес пропонує специфічний продукт або послугу та має невелику клієнтську базу: тоді відносини з клієнтами більш особистісні, ніж у великих компаній, і дзвінок матиме позитивну реакцію від клієнта.

Не потрібно дзвонити всім підряд. Можна поговорити з постійними клієнтами і з тими, хто, залишився задоволений взаємодією. Такі клієнти з набагато більшою ймовірністю залишать детальний позитивний відгук, який підвищить авторитет вашого бренду.

# РОЗДІЛ 1. Сучасне проектування інформаційних систем

## 1.1 Роль інформаційних систем в діяльності підприємства

Вся діяльність підприємства стосовно ІС зводиться до таких базових операцій: доходи, видатки, залишок, баланс, аналіз і планування. Ці операції стосуються будь-яких об'єктів обліку, наприклад: товари, матеріали, основні засоби, безготівкові і готівкові грошові кошти. Практично всі бізнес-процеси підприємства чи будь-який вид економічної діяльності можна представити цими операціями.

Термін «Інформаційна система» належить до класу програмних продуктів, що автоматизують ведення бізнесу. Система називається інформаційною, тому що вона підтримує інформаційне забезпечення бізнесу.

ІС - сукупність інформації, різних методів і моделей, апаратних, програмних, організаційних та технологічних засобів. ІС - це організаційно впорядкована сукупність інформаційних ресурсів та інформаційних технологій, зокрема з використанням засобів обчислювальної техніки і зв'язку, що реалізовують такі інформаційні процеси, як отримання вхідних даних; обробка цих даних і/або зміна власного внутрішнього стану (внутрішніх зв'язків), видача результату або зміна свого зовнішнього стану (зовнішніх зв'язків). За допомогою ІС надається можливість встановлення зв'язку між усіма елементами бізнес-процесів підприємства, що покращує можливості планування, контролю й регулювання процесів. Інформаційна система, при формуванні якої використано принцип зворотного зв'язку на всіх рівнях управління і сучасні ІКТ (Інформаційно-комунікаційні технології), забезпечує зв'язок між елементами системи управління й елементами бізнес-процесів, тобто між усіма етапами прийняття рішень, а також надає можливість накопичення даних, аналізу і моделювання.

## 1.2 Аналіз етапів проектування інформаційних систем

При аналізі і проектуванні системи повинні бути побудовані її повні і несуперечливі моделі. При цьому під моделлю розуміється сукупність взаємопов'язаних абстрактних елементів з можливою вказівкою їх властивостей, поведінки та зв'язків між ними.

Класифікувати моделі можна за такими ознаками:

1. За строгістю опису:

1.1 Неформальні – представлені в неструктурованому вигляді і дають загальне уявлення про системи, які моделюються. Недостатньо наочні (особливо при складній взаємодії між об'єктами) і неприйнятні для будь-якого кількісного аналізу та обробки автоматичними засобами.

1.2 Формальні:

- Описові – моделі, де відомості представлені за допомогою спеціальних документів (бланки, форми, анкети, таблиці);

- Графічні – моделі являють собою схеми, креслення, графи, діаграми. Найбільш наочні і набули широкого поширення при проектуванні за допомогою CASE-засобів;

- Математичні – представляють модель мовою математичних відносин у вигляді функціональних залежностей, систем алгебраїчних або диференціальних рівнянь, логічних виразів.

2. За ступенем фізичної реалізації (логічної незалежності):

- Логічні – описують склад, структуру, стан або поведінку елементів системи без прив'язки до конкретних мов або середовищ програмування, СУБД, технічних засобів. При розробці системи це забезпечує гнучкість у виборі і швидкий перехід з однієї програмно-апаратної платформи на іншу;

- Фізичні – описують елементи системи відповідно до прийнятої фізичної реалізації цих елементів (мов програмування, СУБД, пристроїв і т. д.).

3. За ступенем відображення динаміки процесів:

- Статичні – описують склад і структуру системи.

- Динамічні – описують поведінку системи та/або її окремих елементів. Як правило, такі моделі описують порядок дій або стан системи і переходи між ними. Іншими словами, в цих моделях присутнє поняття часу.

4. За відображаємим аспектом:

- Функціональні – описують функції системи, можливі варіанти її використання; можуть містити відомості про циркулюючу в системі інформацію, об'єкти та суб'єкти, що взаємодіють з системою; можуть бути як динамічними, так і статичними моделями;

- Інформаційні – описують склад і структуру даних (реляційних БД, класів). Відносяться до статичних моделей;

- Поведінкові – описують стани системи та/або її окремих елементів і переходи між ними, взаємодію елементів, алгоритми обробки інформації. Відносяться до динамічних моделей;

- Компонентні – описують склад і структуру програмних і апаратних засобів. Відносяться до статичних моделей;

- Змішані – характеризують відразу кілька аспектів системи (діаграми потоків даних відображують роботи, накопичувачі даних, підсистеми).

На стадіях формування та аналізу вимоги спочатку починають з побудови неформальних моделей (змістовного опису предметної області), поступово переходячи до формальних. Аналогічно на стадії проектування починають зі

створення формальних логічних моделей і закінчують фізичними. Одним з найважливіших результатів проектування є набір логічних і фізичних моделей, що описують всі аспекти системи. Цей набір повинен бути достатнім для подальшої реалізації системи на стадії кодування.

Аналіз етапів проектування неможливий без детального розбору вимог до інформаційної системи, тому перш ніж почати розробляти концепції проектування інформаційних систем, потрібно детально оцінити вимоги.

Джерела відомостей про вимоги:

- Мета та завдання системи, як їх формулює замовник;
- Діюча система або колектив, що виконує її функції; загальні знання щодо проблемної галузі замовника;
- Відомчі стандарти замовника, що стосуються організаційних вимог, середовища функціонування майбутньої системи, її виконавських та ресурсних можливостей.

Методи збирання вимог:

- Інтерв'ю з носіями інтересів замовника та операторами;
- Спостереження за роботою діючої системи;
- Фіксація сценаріїв усіх можливих випадків використання системи, виконуваних при цьому системою функцій, ролей осіб, що запускають ці сценарії або взаємодіють з системою під час її функціонування.

Сучасна технологія проектування ПЗ ІС має забезпечувати:

- Відповідність стандартові ISO/IEC 12207;

- Гарантоване досягнення цілей розробки БС у межах бюджету з дотриманням якості й установленого часу;
- Можливість декомпозиції проекту на складові з наступною інтеграцією цих частин; мінімальний час одержання працездатного ПЗ ІС;
- Незалежність проектних рішень від засобів реалізації 1С (СУБД, операційних систем, мов і систем програмування);
- Підтримка CASE-засобів, що забезпечують автоматизацію процесів, виконуваних на всіх стадіях розробки.

Реальне застосування будь-якої технології проектування ПЗ 1С не можливе без розробки стандартів, яких мають дотримуватися всі учасники проекту (це особливо актуально при великій кількості розробників). До них належать стандарти проектування, оформлення проектної документації та інтерфейсу кінцевого користувача із системою

## 1.3. Огляд підходів щодо проектуання інформаційних систем

Серед основних методологій проектування ІС виділяють такі:

- Методологія функціонального моделювання робіт SADT (Structured Analysis and Design Technique – структурного аналізу і проектування);
- Методологія швидкої розробки застосунків RAD (Rapid Application Development);
- Методологія раціонального уніфікованого процесу RUP (Rational Unified Process).

### 1.3.1 Методологія функціонального моделювання робіт SADT.

Була розроблена Дугласом Т. Росом в 1969–1973 рр., базується на структурному аналізі систем і графічному зображенні організації у вигляді системи функцій, які мають три класи структурних моделей:

- функціональна модель,

- інформаційна модель,
- динамічна модель.

Процес моделювання за методологією SADT складається з таких етапів:

- Збирання інформації та її аналіз про предметну область.
- Документування отриманої інформації.
- Моделювання (IDEF0).
- Корегування моделі в процесі ітеративного рецензування.

Методологія, відома як нотація IDEF0, використовує формалізований процес моделювання ІС і має такі стадії: аналіз, проектування, реалізація, об'єднання та тестування, встановлення (інсталяція), функціонування. Проектування ІС за стандартом IDEF0 зводиться до декомпозиції основних функцій організації на окремі бізнес–процеси, роботи або дії. В результаті розробляється ієрархічна модель аналізованої організації, при цьому декомпозицію можна проводити багаторазово, прямуючи до чіткого та детального опису усіх процесів.

Діаграми IDEF0 верхнього рівня прийнято називати батьківськими, а нижнього рівня – дочірніми. Приклад діаграми IDEF0 верхнього рівня наведений на рис. 1.1. Аналізований процес подають у вигляді прямокутника: ліворуч зображають вхідні дані; справа – вихідні; дані, розташовані зверху – дії, які управляють або регламентуючі, а знизу – об'єкти управління. У діаграмі IDEF0 спочатку описуються усі зовнішні зв'язки досліджуваного процесу. Після цього здійснюється декомпозиція цього процесу, і відбувається опис внутрішніх підпроцесів із позначенням усіх зв'язків. При цьому раніше позначені стрілками зовнішні зв'язки не повинні загубитися. Вони переносяться на діаграму декомпозиції у відповідні підпроцеси. Приклад декомпозиції діаграми IDEF0 (дочірньої діаграми) наведено на рис. 1.2.

Основною перевагою цієї методології є простота і наочність. В якості недоліку можна вважати неможливість описати реакцію описуваного процесу на зовнішні чинники, які змінюються.

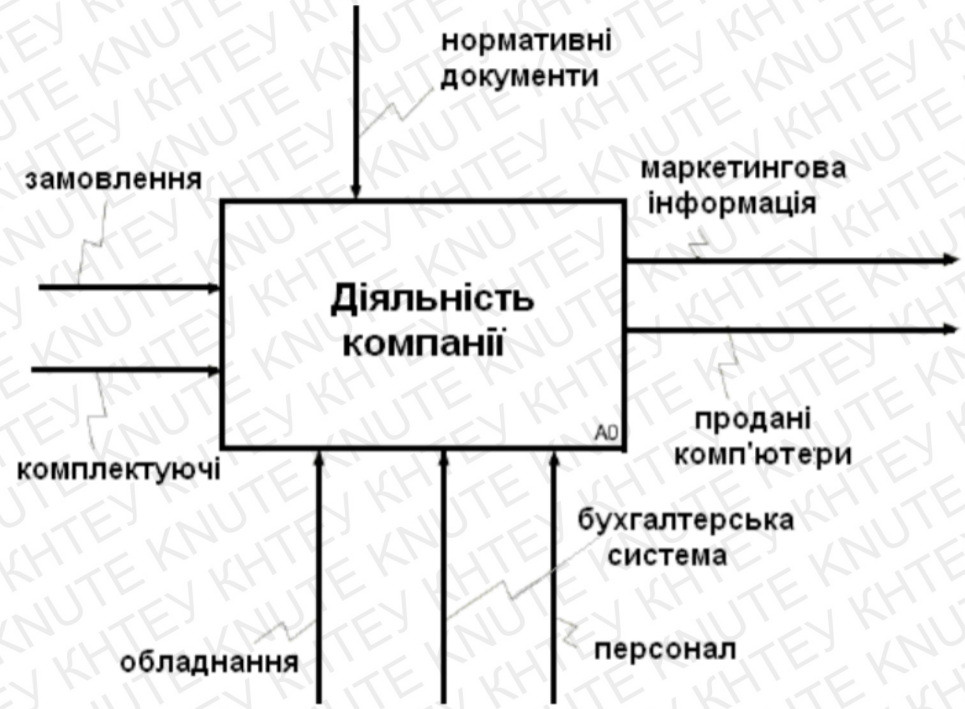


Рис. 1.1 - Діаграма IDEF0 верхнього рівня

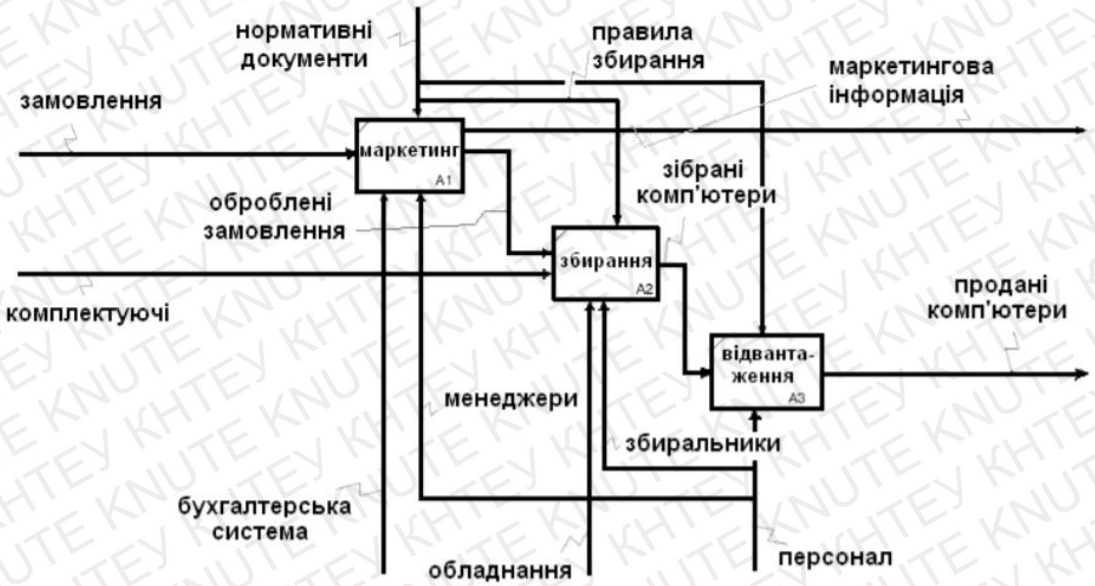


Рис. 1.2 - Дочірня діаграма IDEF0 (декомпозиція)

### 1.3.2 Методологія швидкої розробки застосунків RAD

Принципи RAD були сформульовані у 1980 р. співробітником компанії IBM Джеймсом Мартіном і базувалися на ідеях Скотта Шульца і Барри Бойма. Підхід RAD передбачає наявність трьох складових:

1. Невеликі групи розробників (від 3 до 7 осіб), які виконують роботи з проектування окремих підсистем ПЗ (це обумовлено вимогою максимальної керованості колективу). Команда розробників повинна бути групою професіоналів, які мають досвід в проектуванні, програмуванні та тестуванні ПЗ, здатних добре взаємодіяти з кінцевими користувачами і трансформувати їх пропозиції в робочі прототипи.

2. Короткий та ретельно пропрацьований виробничий графік.

3. Використання інкрементного прототипування – повторюваного циклу, при якому розробники в міру того, як додаток починає набувати форму, реалізують у продукті вимоги, отримані в результаті взаємодії із замовником.

Нині методологія RAD стала загальноприйнятою схемою для проектування і розробки ІС. Вона дозволяє на ранній стадії проектування ІС продемонструвати замовникові діючу інтерактивну модель системи–прототипу, уточнити проектні рішення та вимоги замовника, оцінити експлуатаційні характеристики.

Серед основних принципів підходу RAD можна виділити такі:

- Розробка додатків ітераціями;
- Необов'язковість повного завершення робіт на кожній стадії життєвого циклу ПЗ;
- Обов'язковість залучення користувачів у процес розробки;
- Застосування засобів управління конфігурацією, які полегшують внесення змін до проекту і супровід готової системи;

- Використання прототипування, що дозволяє повніше з'ясувати і задовольнити потреби користувачів;
- Тестування і розвиток проекту, здійснювані одночасно з розробкою;
- Ведення розробки нечисленною добре керованою командою професіоналів;
- Грамотне керівництво розробкою системи, чітке планування і контроль виконання робіт. Засоби розробки, грунтовані на RAD, є популярними за рахунок використання таких програмних середовищ розробки: IBM Lotus Domino Designer, Borland C++ Builder, Microsoft Visual Studio тощо.

В методології RAD швидка розробка додатків досягається за рахунок використання компонентно–орієнтованого конструювання і застосовується, якщо:

- Бюджет проектованої ІС обмежений,
- Нечітко визначені вимоги до інформаційної системи,
- Потрібно реалізувати проект ІС в мінімальні терміни,
- Інтерфейс користувача можна продемонструвати в прототипі,
- За функціональним призначенням проект можна розділити на складові елементи.

Методологія RAD має декілька недоліків вона: вимагає великий колектив розробників для великих ІС; використовується для ІС, які можна декомпозувати на окремі модулі і в яких продуктивність не є критичною величиною; не використовується у разі застосування нових технологій.

Життєвий цикл ПЗ відповідно до підходу RAD складається з чотирьох стадій: аналіз і планування вимог, проектування, реалізація та впровадження. RAD найкраще підходить для відносно невеликих проектів, що розробляються для конкретного замовника, його не застосовують для побудови складних

розрахункових програм, операційних систем або програм управління складними об'єктами в реальному масштабі часу, тобто програм, що містять великий обсяг унікального коду, а також систем, від яких залежить безпека людей (наприклад, керування літаком або атомною електростанцією).

### 1.3.3 Методологія раціонального уніфікованого процесу RUP.

Серед усіх фірм–виробників CASE–засобів компанія IBM Rational Software Corp. одна з перших усвідомила перспективність розвитку об'єктно–орієнтованих технологій аналізу і проектування програмних систем, що привело до появи перших версій мови UML. Ця ж компанія першою розробила інструментальний об'єктно–орієнтований CASE–засіб. Графічне подання методології RUP зображено на Рис. 1.3. RUP відповідає стандартам, які визначають проектні роботи в процесі життєвого циклу ІС.

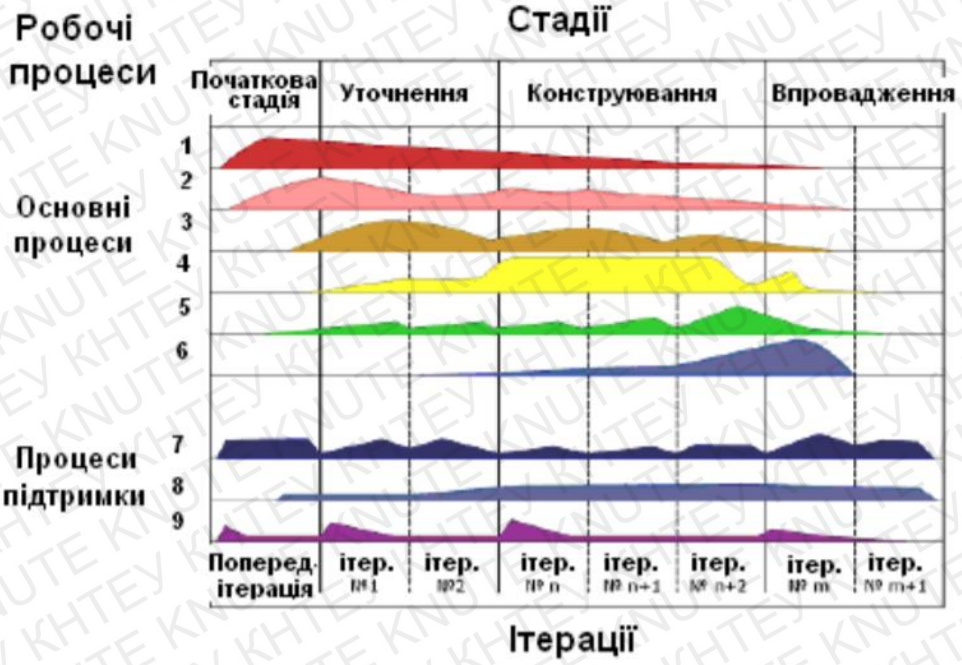


Рис. 1.3 - Подання методології RUP

У методології RUP реалізуються такі підходи:

1. Ітераційний та інкрементний.

2. Побудова системи на базі архітектури інформаційної системи.

3. Планування та управління проектом на основі функціональних вимог до ІС.

Розробка ІС виконується ітераціями: створюють окремі проекти, невеликі за об'ємом та змістом, які охоплюють свої власні етапи аналізу вимог, проектування, реалізації, тестування, інтеграції. Закінчуються ітерації створенням працюючої ІС. Ітераційний цикл характеризується періодичним зворотним зв'язком і може адаптуватися до ядра системи, яка розробляється. Створювана ІС поступово росте та вдосконалюється.

**Висновки по розділу:** Проектування ІС передбачає використання проектувальниками певної технології проектування, яка відповідає масштабу та особливостям розроблюваного проекту. Технологією проектування (ТП) ІС є сукупність методології та засобів проектування ІС, а також методів і засобів організації проектування (управління процесом створення та модернізації проекту ІС). В основі ТП ІС лежить технологічний процес, який визначає дії, їх послідовність, склад виконавців, кошти і ресурси, необхідні для виконання цих дій, задається регламентованою послідовністю технологічних операцій, що виконуються в процесі створення проекту на основі конкретного методу, в результаті чого стало б ясно, не тільки що повинно бути зроблено для створення проекту, але і як, кому і в якій послідовності це повинно бути зроблено.

## РОЗДІЛ 2. Організація розробки інформаційної системи обліку зауважень та побажань користувачів.

### 2.1. Розробка концепції інформаційної системи

Рівень розвитку сучасних технологій настільки високий, що дозволяє побудувати ІС будь–якої складності та функціональності. Проте, враховуючи вимоги бізнесу, грунтовані на показниках різних бізнес–оцінок, виникають задачі, розв'язання яких зводиться до забезпечення раціонального підходу до процесу проектування, реалізації та подальшої експлуатації ІС. Вибрану архітектуру можна вважати одним із основних показників ефективності створюваної ІС.

Основу системної побудови інформаційної системи становить її структура, яка має повністю описувати поведінку системи що розробляється. Засобами структуризації є процедури декомпозиції і композиції системи, тому важливим етапом проектування інформаційної системи є структуризація системи – локалізація її меж і виділення структурних складових частин.

1. Аналіз вимог - Специфікація програмного забезпечення
2. Проектування програмного забезпечення
3. Програмування
4. Тестування програмного забезпечення
5. Системна інтеграція
6. Впровадження програмного забезпечення (або Установка програмного забезпечення)
7. Супровід програмного забезпечення

На першому етапі проводиться обстеження об'єкта та обгрунтовується необхідність створення ІС, формулюються вимоги користувача до ІС, оформляється звіт про виконану роботу. Під час обстеження об'єкта з'ясовується документообіг, форми початкових та вихідних документів, методики розрахунку окремих показників. Обстеження має виявити проблеми, розв'язання яких можливе засобами обчислювальної техніки, та надати оцінку доцільності

створення ІС. Разом із замовником погоджуються вимоги до ІС. Серед вимог можуть бути суми максимальних витрат на розробку, термін виконання розробки, умови функціонування системи, перелік функцій, які система має забезпечити.

Проєктування програмного забезпечення - це процес у якому провадяться науково-дослідні роботи для пошуку шляхів та оцінки можливостей реалізації вимог користувача. На цьому етапі можна визначити методи, які будуть покладені в основу розрахунків, або принципові підходи до розв’язування конкретних задач. Він може містити оцінку необхідних для реалізації ресурсів розробки та самої ІС, давати порівняльну характеристику тих чи інших варіантів розробки інформаційної системи, визначати порядок оцінки якості системи.

Програмування - процес проектування, написання, тестування, зневадження і підтримки комп'ютерних програм. Програмування поєднує в собі елементи інженерії, фундаментальних наук (перш за все математики) і мистецтвар[8].

Тестування програмного забезпечення - це процес технічного дослідження, призначений для виявлення інформації про якість продукту відносно контексту, в якому він має використовуватись. Техніка тестування також включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою оцінки[14].

Системна інтеграція в інженерії - поєднання компонентів підсистем в єдину систему та забезпечення їх роботи у якості єдиної системи. В області інформаційних технологій системна інтеграція є процесом об'єднання різних обчислювальних систем і програмних застосунків фізично або функціонально. Системна інтеграція полягає у розробці комплексних рішень, призначених для досягнення максимальної ефективності функціонування системи шляхом налагодження ефективної взаємодії її підсистем [14].

Інсталяція — процес встановлення програмного забезпечення на комп'ютер кінцевого користувача. Виконується особливою програмою (пакетним менеджером), присутнім в операційній системі (наприклад, RPM і APT в Linux, Windows Installer в Microsoft Windows), або ж тим, що вже входить до складу самого програмного забезпечення засобом встановлення. В операційній системі GNU дуже поширене використання системи GNU toolchain і її аналогів для компіляції програмного забезпечення безпосередньо перед встановленням.

Супроводження програмного забезпечення — процес покращення, оптимізації та виправлення дефектів у програмному забезпеченні після його вводу до експлуатації.


Рис 2.1 - Концепція інформаційної системи

## 2.2 Модель інформаційної системи обліку зауважень та побажань користувачів.

Щоб розробити ІС для обліку зауважень та побажань користувачів потрібно пройти такі кроки:

1. Проаналізувати методи отримання зворотнього зв’язку від користувачів.
2. Вибір системи що дозволяє отримувати відгуки.
3. Розробка автоматизованої системи ведення обліку зауважень та побажань користувачів
4. Впровадження системи в експлуатацію

На даний момент майже кожна людина у світі взаємодіє з тим чи іншим програмним забезпеченням та зареєстрована у соціальних мережах, проте зараз присутня тенденція міграції користувачів у месенджери, тому що месенджери зазвичай швидші та простіші за додатки соціальних мереж. Головною перевагою можна назвати те, що месенджери захищають конфіденційні дані користувачів(Telegram, ICQ).

Більш прогресивний і функціональний вид автоматизованої системи - це чатботи, які навіть певною мірою можуть розуміти живу мову. Спілкування з таким ботом наближене до реального людського, але має додаткові функціональні особливості. Можливості ботів такого типу ширше кнопкових. Вони можуть розпізнати людську мову, проаналізувати запит і видати на нього реакцію - відповідь в форматі діалогу. Такі чатбот спираються на системи розпізнавання мови, щоб спілкуватися з користувачами.

Крім відповідей на заздалегідь підготовлені питання, робот може зрозуміти про що мова по набору слів, і діалог відбувається в форматі спілкування двох людей. бот на кнопках і командах; він також має інтерфейс месенджера, але замість можливості написати йому текстом, людині пропонуються на вибір категорії (або питання / пропозиції), які можуть його цікавити.

В даній системі реалізований кнопковий чат-бот, у якому «спілкування» відбувається шляхом натискання кнопок, бот реагує на них, як на команди. Такий чат-бот найбільш схожий на звичний мобільний додаток, з тією лише різницею, що у нього немає свого інтерфейсу, а для роботи він використовує месенджер Telegram. При взаємодії з командами бота, дані відправляються та оброблюються на backend-і та записуються у базу данних SQLite.

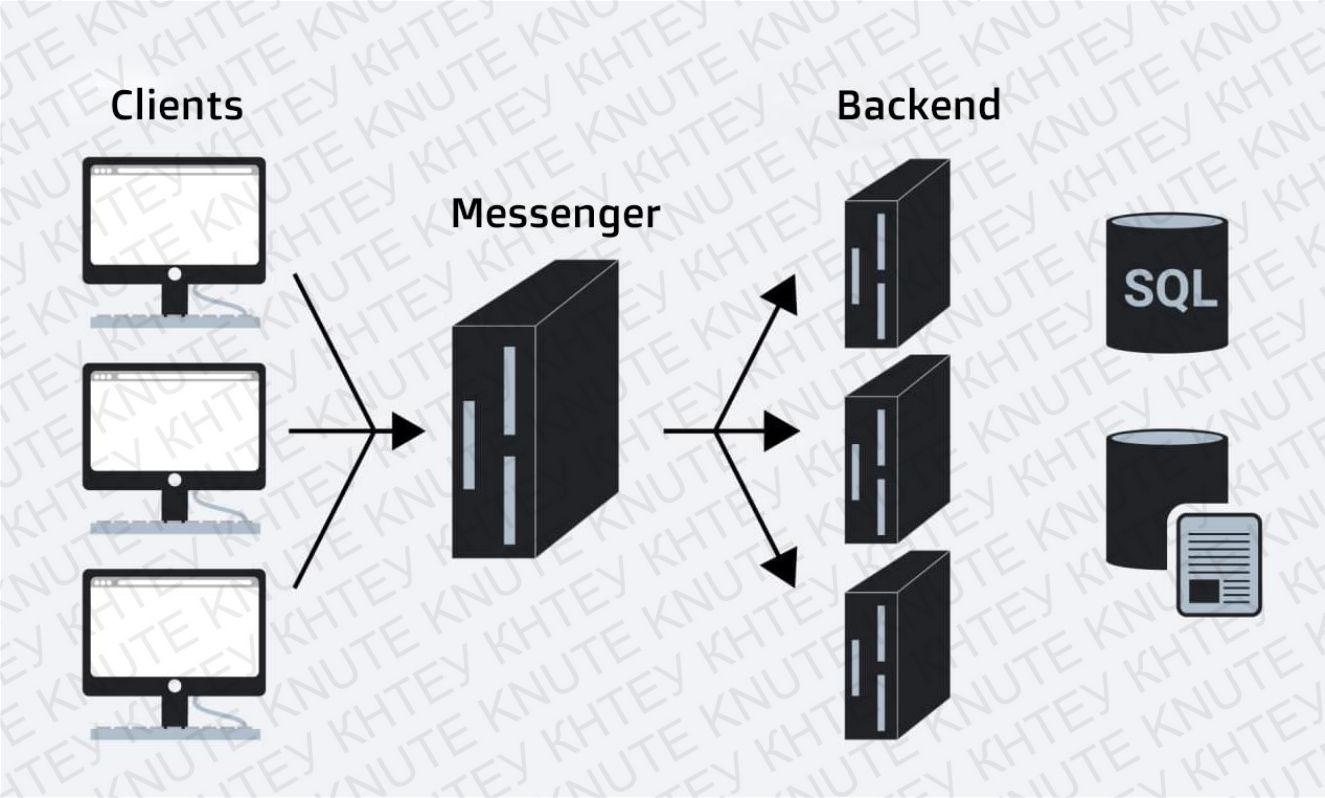


Рис. 2.2 Модель програмного забезпечення

## 2.3. Алгоритм створення інформаційної системи

Для реалізації інформаційної системи потрібно:

Завантажити:

1. SQLite – База даних
2. PyCharm – Python IDE

Створити:

1. Чат-бот – Можна створити за допомогою BotFather
2. Папка з файлами для розробки ІС (Рис 2.3):
   - Main.py – файл основного коду програми, до якого мають підключають інші файли с розширенням .py
   - sqlGets та sqlSets – файли з кодом для роботи БД з основним кодом програми.
   - userClass.py – класс користувача та його конструктори
   - users.db – база даних, зберігає відгуки користувачів.

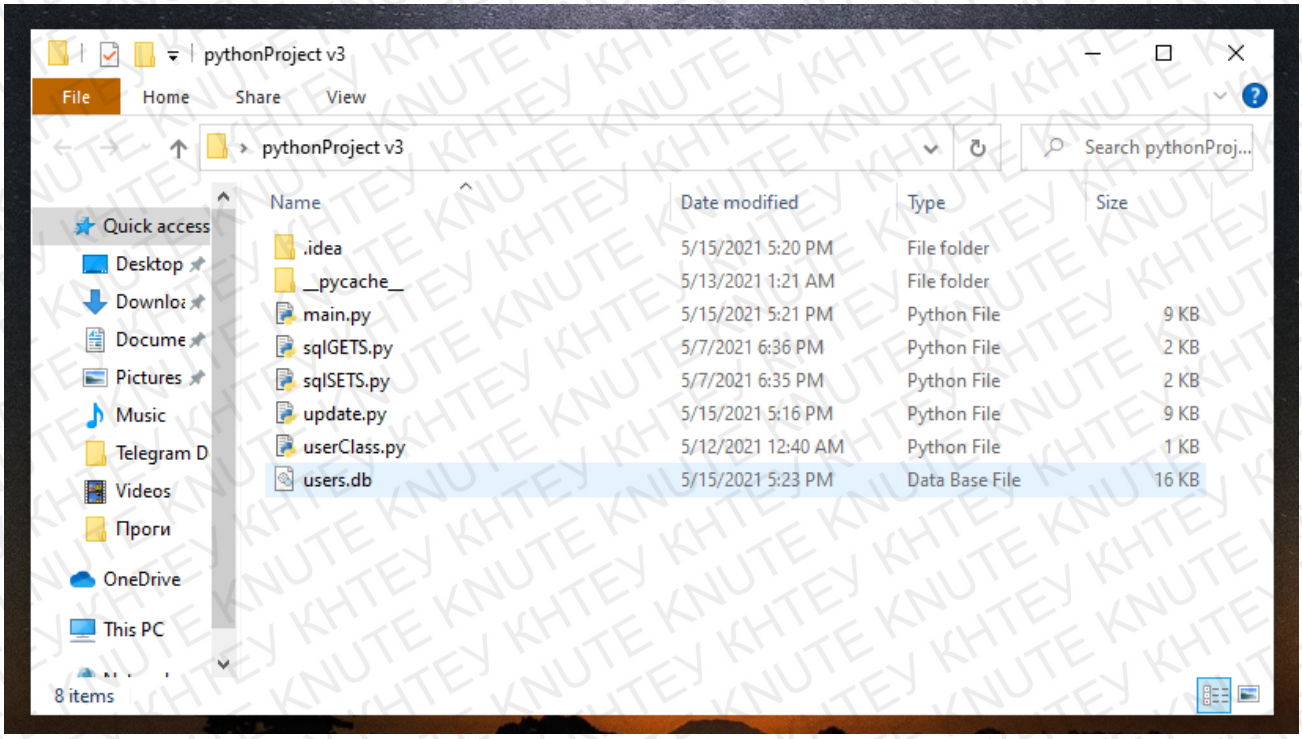


Рис. 2.3 – Необхідні файли для створення проекту

Короткий опис роботи: Спочатку завантажуємо та налаштовуємо середовище розробки PyCharm IDE. Створюємо необхідні файли для написання коду. Завантажуємо SQLite – система управління базами даних. Для того щоб

оцінити коректність роботи інформаційною системи (пройти очікуваний сценарій роботи ІС), потрібно написати програмний код, імпортувати всі файли у головний файл, запустити скрипт та пройти послідовні дії у чат-боті. Після чого можна буде відкрити файл  бази данних у SQLite та побачити результат (записаний відгук користувача).

**Висновки по розділу:** Отже, інформаційна система являє собою сервіс для роботи с відгуками користувачів, який збирає та записує в базу даних побажання та зауваження користувачів з можливістю поставити оцінку програмному забезпеченню. У цьому розділі була описана загальна концепція програмного забезпечення. В ній розписано, що саме знадобиться у роботі та які додаткові програми було встановлено. Основна робота буде зроблена в програмі PyCharm IDE, також встановлена база даних SQLite та створений чат-бот.

## РОЗДІЛ 3. Реалізація проектування інформаційної системи обліку зауважень та побажань користувачів

### 3.1 Створення чат-бота.

Для реалізації проекту спершу потрібно створити чат-бот. Створити можна за допомогою майстра по створенню ботів – Bot Father у месенджері «Telegram» та за допомогою деяких команд задаємо певні особливості новоствореному боту[3].

Використані команди:

- /start – починає роботу з ботом (Рис 3.1)
- /newbot – створює нового бота (Рис 3.2)
- /mybots – показує доступних ботів (Рис 3.3)

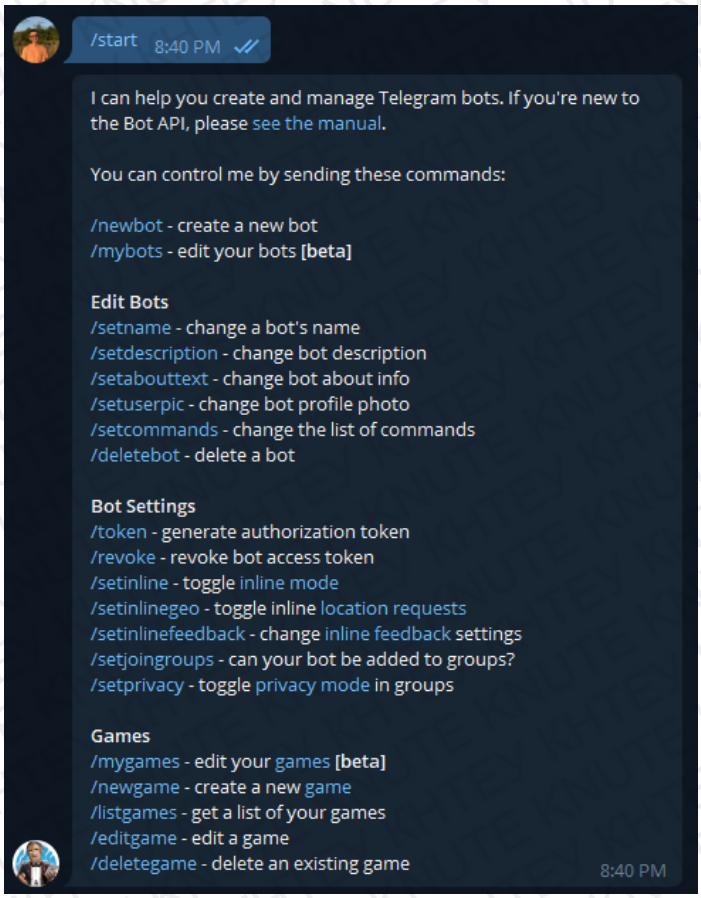


Рис 3.1 – Команда для початку роботи «/start»

Команда /newbot дає можливість дати ім’я (TymoshenkoBot) та назву чат-бота @customerReviewbot.


Рис 3.2 – Команда створення нового боту «/newbot»

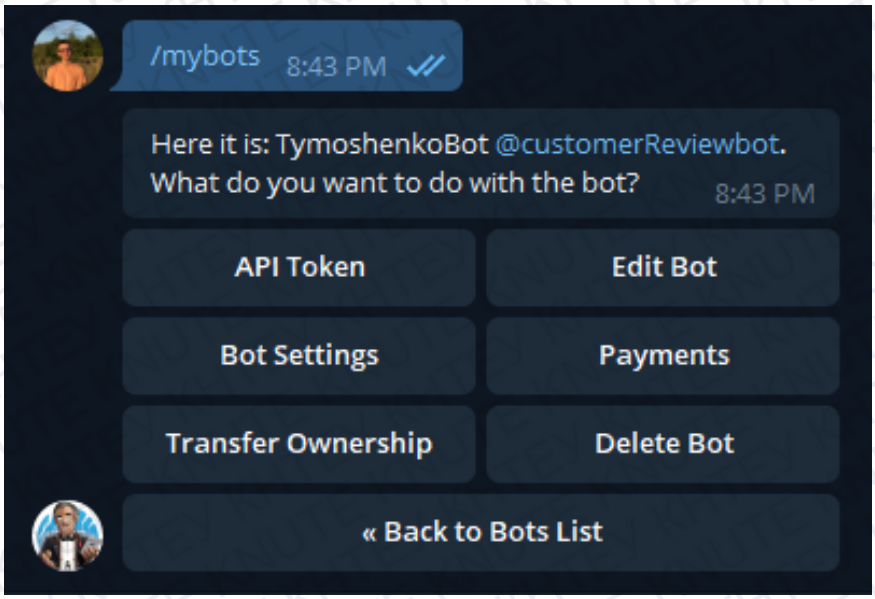


Рис 3.3 – Перегляд існуючих ботів «/mybots»

Після вводу команди /mybots, потрібно натиснути на кнопку «API Token» щоб отримати токен бота для його підключення у коді програми (Рис 3.4)

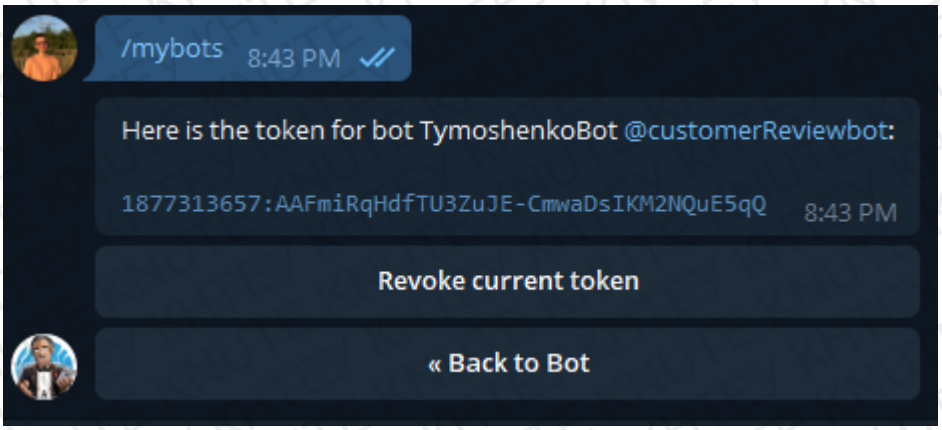


Рис 3.4 – API Token

### 3.2 Програмна реалізація проекту

Наступний крок – створення скрипту, який буде працювати з месенджером Telegram. На (Рис 3.5) представлена схема папок проекту на мові програмування Python.

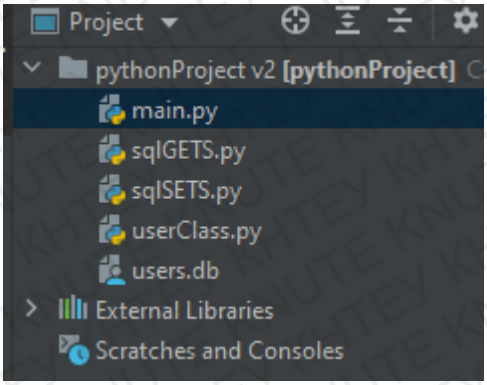


Рис 3.5 – Структура проекту

Після того як був творений чат-бот, потрібно створити проект у PyCharm IDE, створити файли імпортувати бібліотеку «telebot», для створення команд бота та підключення токену щойно створеного бота [1,2,4-6]. Фрагмент коду:

```
import = telebot

bot = telebot.TeleBot("1877313657:AAFmiRqHdfTU3ZuJE-
CmwaDsIKM2NQuE5qQ")
```

Наступним кроком буде створення класу та конструктору класу «user» у окремому файлі userClass. Фрагмент коду:

```
class User:
   name: str
   description: str
   rating: str

   def __init__(self, name: str = "", rating: str = "",
                description: str = ""):
      self.name = name
      self.description = description
      self.rating = rating


usertable = default_write_table("users",
                        id="id INTEGER PRIMARY KEY",
                        name="name TEXT NOT NULL",
                        rating="rating TEXT",
                        description="description TEXT NOT NULL")


is_description = False
```

Файл main.py описує весь функціонал для взаємодії системи з користувачем. Команда /start використовується для привітання при старті розмови та направляє користувача до подальшої взаємодії за допомогою кнопок write_feedback та help. Фрагмент коду для привітання користувача:

```
@bot.message_handler(commands=['start'])
def start(message):
    database = create_connection("users.db")
    new_users = default_add_info_table("users",
                                       "id, name,rating,description",
                                       user="('{id}', '{name}','{rating}','{description}')"
                                       .format(id=message.chat.id,
                                               name=message.from_user.username,
                                               rating="None",
                                               description=message.text))

    write_query(database, usertable)
    write_query(database, new_users)

    markup_inline = types.InlineKeyboardMarkup()
    item_yes = types.InlineKeyboardButton(text='write_feedback',
callback_data='yes')
    item_no = types.InlineKeyboardButton(text='help', callback_data='no')

    markup_inline.add(item_yes, item_no)
    bot.send_message(message.chat.id, 'I am glad to see you :) \n Select action, please',
                     reply_markup=markup_inline)
```

При выборі однієї з кнопки запускається callback_query_handler та зчитує параметр «callback_data».

```
@bot.callback_query_handler(func=lambda call: True)
def answer(call):
    connect = sqlite3.connect('users.db')
```

```
    read_id = "SELECT users.id FROM users"
    id = read_query(connect, read_id)
    for i in range(len(id)):
        try:
            if id[i][0] == call.message.chat.id:
                id = id[i][0]
        except:
            pass
    rating = None
    if call.data == 'yes':
        set_rating(call.message)
    if call.data == 'no':
        help_command(call.message)
    rating_call("1", call, connect, id)
    rating_call("2", call, connect, id)
    rating_call("3", call, connect, id)
    rating_call("4", call, connect, id)
    rating_call("5", call, connect, id)
```

Якщо користувач натискає на кнопку help, то бот виконує команду /help, відправляє інформаційне повідомлення та допомагає розібратися у функціях системи. Фрагмент коду:

```
@bot.message_handler(commands=['help'])
def help_command(message):
    bot.send_message(
        message.chat.id,
       'We use this bot to collect a feedback and wishes about our software. To leave a
```

response, click on the command /rating',
)

Якщо користувач натискає на кнопку write_feedback, то бот виконує команду set_rating та направляє користувача на виставлення оцінки від одного до п’яти за допомогою кнопок. Фрагмент коду:

```
@bot.message_handler(commands=['rating'])
def set_rating(message):
  markup_inline = types.InlineKeyboardMarkup()
  item_1 = types.InlineKeyboardButton(text='1', callback_data='1')
  item_2 = types.InlineKeyboardButton(text='2', callback_data='2')
  item_3 = types.InlineKeyboardButton(text='3', callback_data='3')
  item_4 = types.InlineKeyboardButton(text='4', callback_data='4')
  item_5 = types.InlineKeyboardButton(text='5', callback_data='5')
  markup_inline.add(item_1, item_2, item_3, item_4, item_5)
  bot.send_message(
    message.chat.id,
    'Rate us', reply_markup=markup_inline
  )
```

Після вводу оцінки виконується функція rating_call, яка записує у базу даних оцінку, яку виставив користувач. Та запускає наступну функцію write_feedback. Фрагмент коду:

```
@bot.message_handler(content_types=["text"])
def write_feedback(message):
```

```
    global is_description
    if is_description:
        database = create_connection("users.db")
        description = "UPDATE users SET description = '{text}' WHERE id =
{id};".format(id=message.chat.id, text=message.text)
        write_query(database, description)
        is_description = False
        bot.send_message(
            message.chat.id,
            'Thanks for your feedback, we are getting better with your help',
        )
```

Закінчується головний блок функцією «bot.polling» з параметром “none_stop= True” для того щоб робота бота не завершувалась. Фрагмент коду:

```
    bot.polling(none_stop= True)
```

Файл sqlGets.py має одну функцію, яка використовується для зчитування id користувача з таблиці та створює зв'язок с базою даних. Фрагмент коду:

```
def read_query(connection, query):
    cursor = connection.cursor()
    result = None
    try:
        cursor.execute(query)
        result = cursor.fetchall()
        return result
    except Error as e:
        print(f"The error '{e}' occurred")
```

Файл sqlSets.py має декілька функцій які виконують основні дії з записом відгуків. Перша функція «default_write_table» створює таблицю у базі даних в залежності від аргументів[12]. Фрагмент коду:

```
def default_write_table(name_table: str, **kwargs: str):
    fields = ""
    for args in kwargs.values():
        fields += args + ","
    fields = fields[:-1]
    create_users_table = """
    CREATE TABLE IF NOT EXISTS {name_table} (
        {args}
    );
    """.format(name_table=name_table, args=fields)
    return create_users_table
```

Функція «write_query» створює зв'язок з базою даних та робить запис у таблицю.

```
def write_query(connection, query):
    cursor = connection.cursor()# object of db
    try:
        cursor.execute(query) # integrate values in db
        connection.commit()
        print("Query executed successfully")
    except Error as e:
        print(f"The error '{e}' occurred")
```

Функція «create_connection» тільки створює зв'язок з базою даних. Фрагмент коду:

```
def create_connection(path=DB):
    connection = None
    try:
        connection = sqlite3.connect(path)
        print("Connection to SQLite DB successful")
    except Error as e:
        print(f"The error '{e}' occurred")
    return connection
```

Функція «default_add_info_table» записує відгук користувача (description) у базу даних. Фрагмент коду:

```
def default_add_info_table(name_table: str, fields: str, **kwargs: str):
    info = ""
    for args in kwargs.values():
        info += args + ","
    values = info[:-1]
    add_users = """
    INSERT INTO
     {name_table}({fields})
    VALUES
     {values};
    """.format(name_table=name_table, fields=fields, values=values)
    return add_users
```

### 3.3 Апробація результатів дослідження

Щоб побачити роботу бота, першим чином потрібно запустити програмний код, відкрити чат з ботом та ввести команду /start, на що бот відправить повідомлення-привітання та запропонує натиснути на одну з двох кнопок (Рис 3.5). Команда /help допомагає користувачу розібратися з функціоналом бота (Рис 3.6).

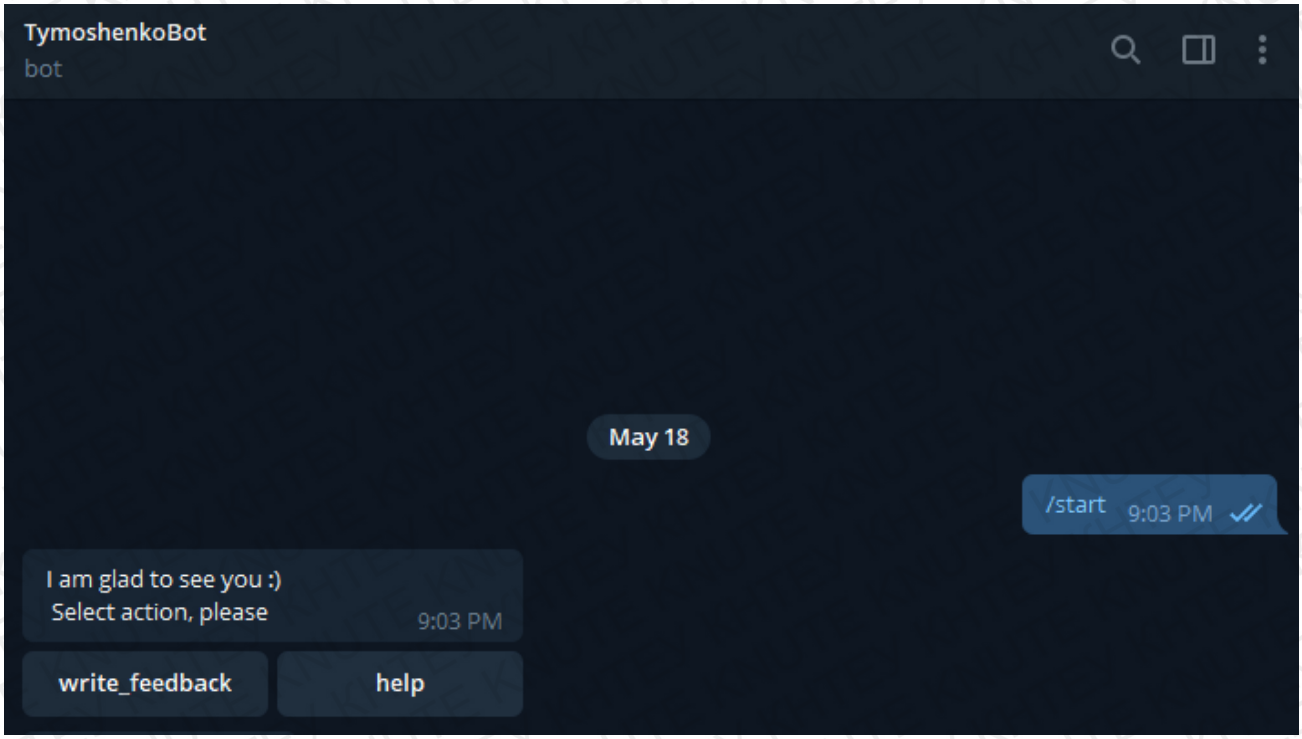


Рис 3.5 – Відповідь бота на команду /start

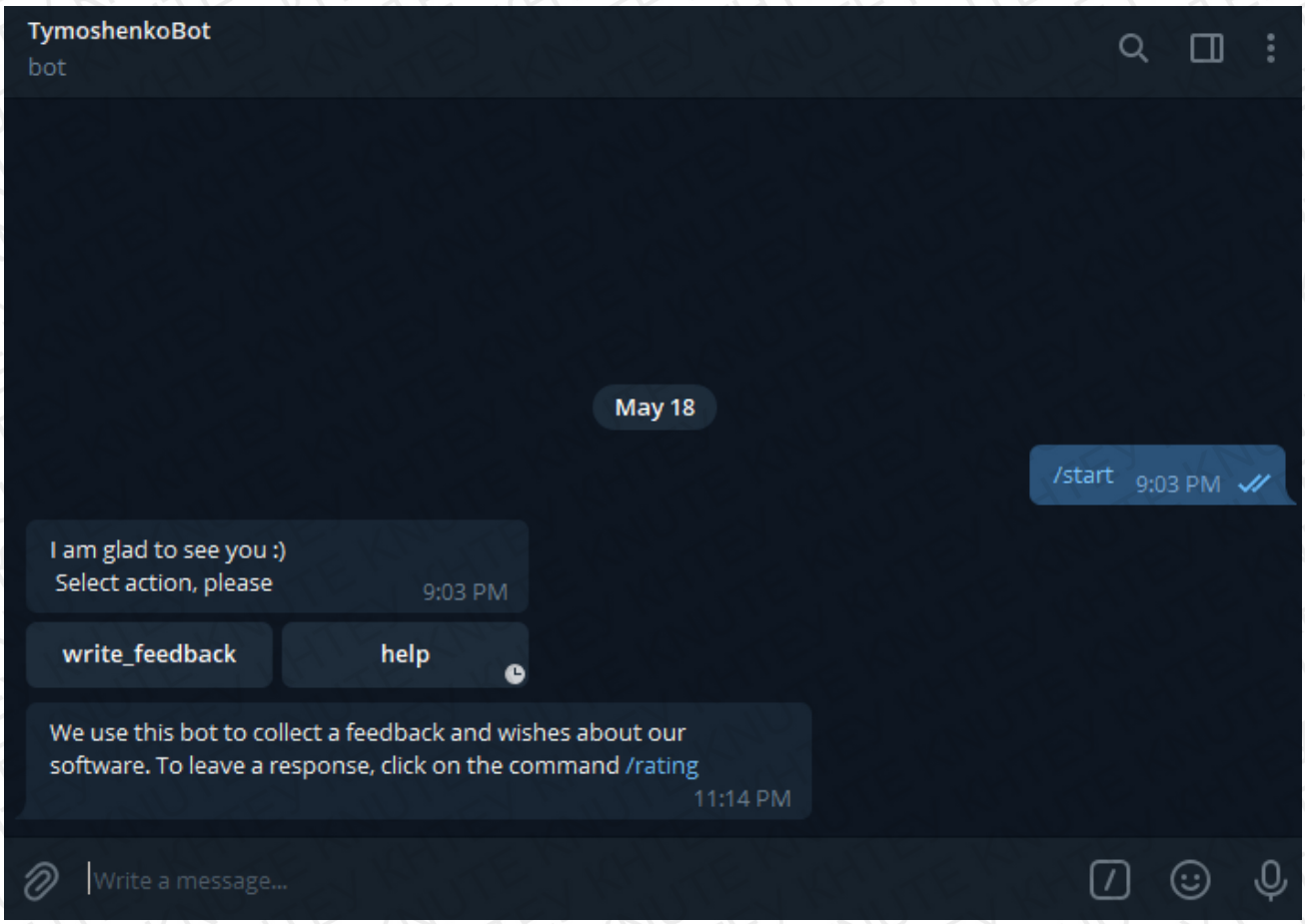


Рис 3.6 – Відповідь бота на команду /help

Команда write_feedback запускає деяку послідовність подій для отримання відгуку від користувача. Перша дія – це оцінювання роботи сервісу (Рис 3.7), друга дія – написання відгуку (Рис 3.8).

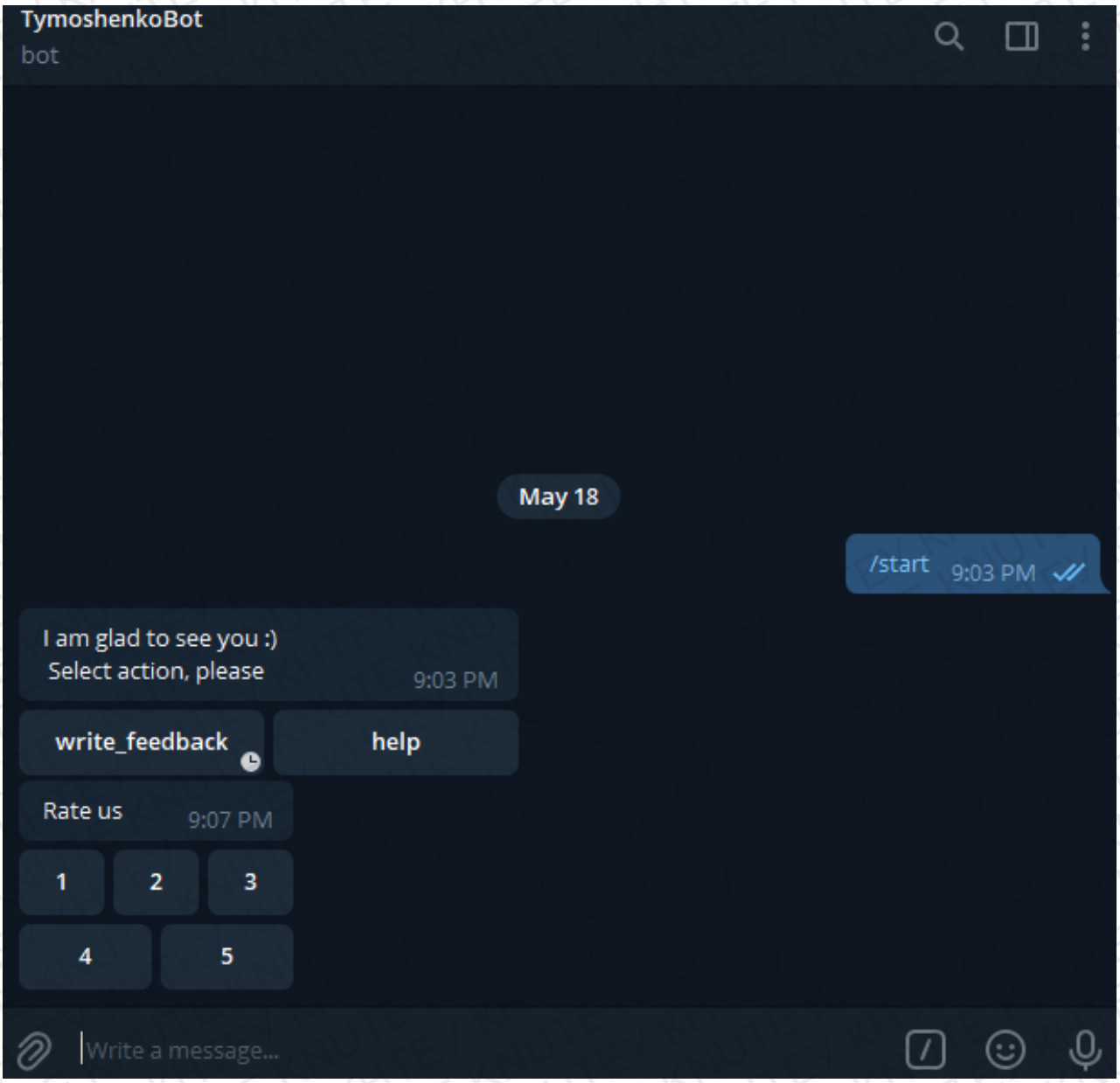


Рис 3.7 – Оцінювання роботи сервісу

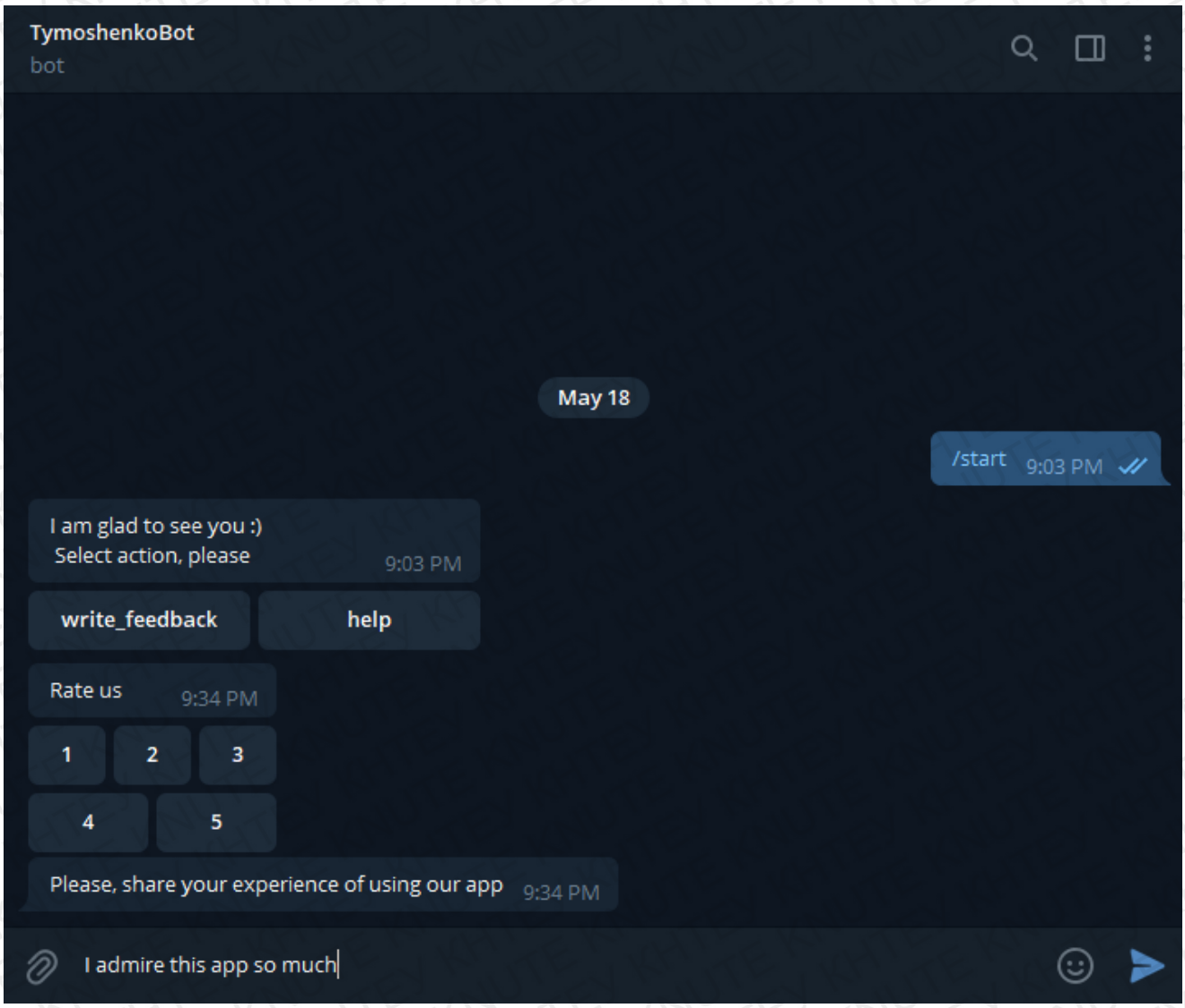


Рис 3.8 – Написання відгуку

Після відправлення відгуку, користувач отримує повідомленння-подяку за допомогу по покращенню роботи сервісу (Рис 3.9).

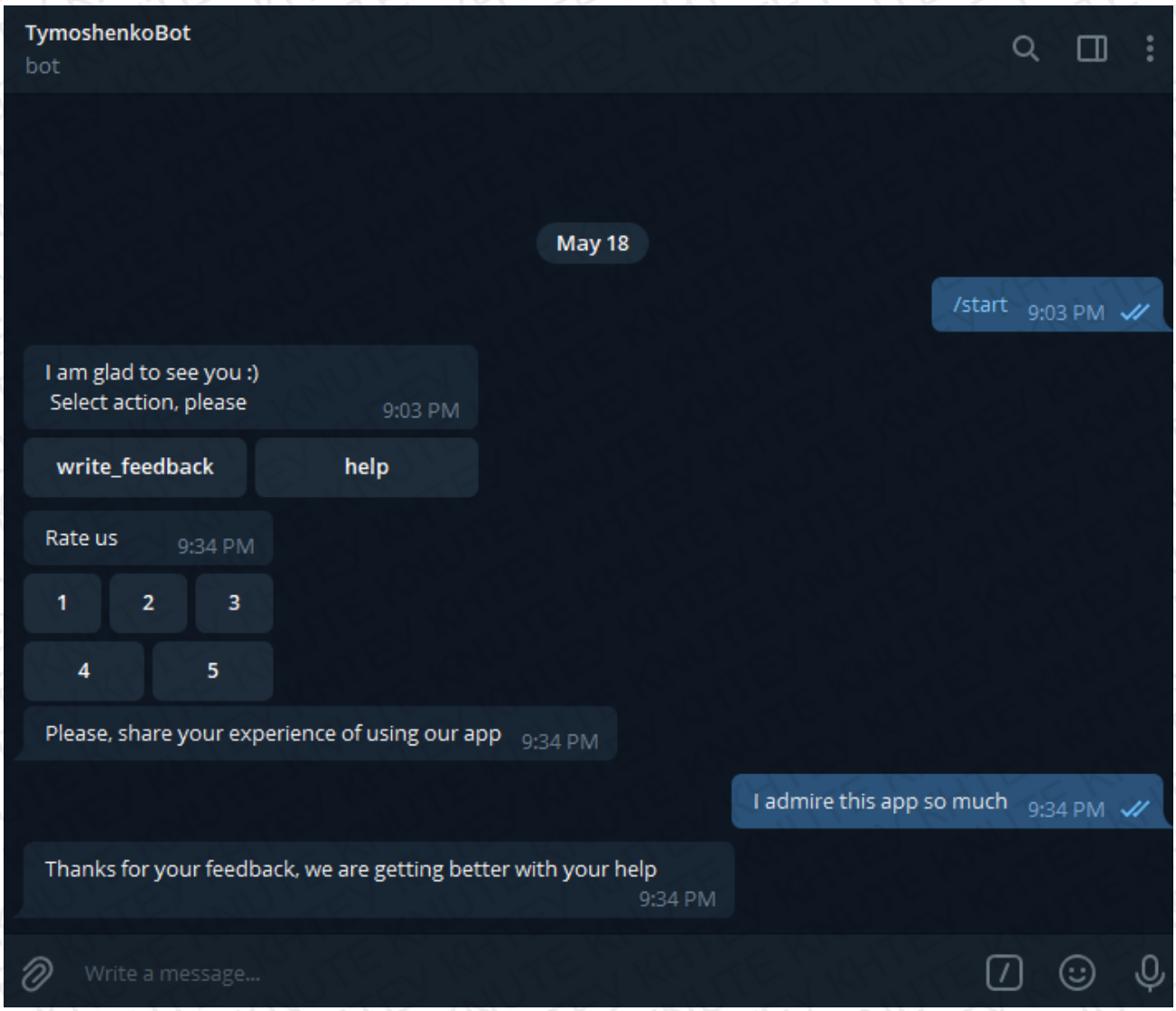


Рис 3.9 – Заключне повідомлення після відправки відгуку

Після вдало написаного відгуку у базі даних з’являються записи (Рис 3.10). Під час даного експерименту було написано відгук, який має такі поля і записаний у базі даних під номером 2 (Рис 3.10):

- Telegram Id – 343113858
- Telegram name – EvheniiT
- Rating – 4
- Description - I admire this app so much

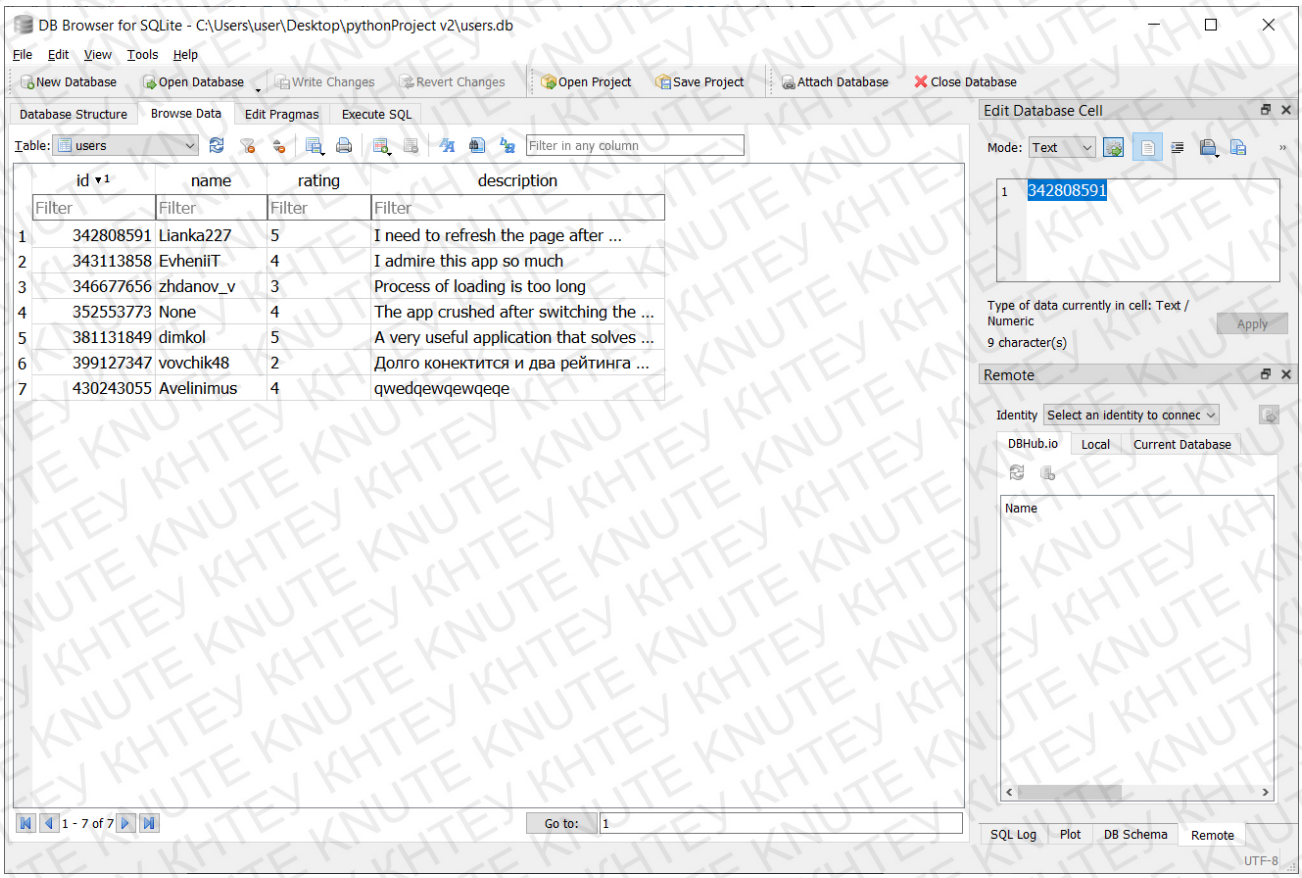


Рис 3.10 – Результати досліджень

**Висновки по розділу:** Була розроблена автоматизована інформаційна система для обробки зауважень та побажань користувачів за допомогою чат-боту Telegram та бази даних SQLite на базі мови програмування Python. Представлені основні елементи коду та описані властивості функції. Також була представлена взаємодія користувача з чат ботом.

# ВИСНОВКИ

1. Інформаційні технології - це клас областей діяльності, що відносяться до технологій управління і обробки величезного потоку інформації із застосуванням обчислювальної техніки. У сучасному світі просто неможливо уявити життя без інформаційних технологій, незважаючи на те, що в недалекому минулому людина і гадки не мала про них. Наразі інформаційні технології застосовуються в усіх сферах життя людства, виконуючи особливо значиму роль.

2. У випускній кваліфікаційній роботі представлено результати теоретичних і прикладних досліджень, що полягають у розробці концепції та автоматизованої інформаційної системи обліку зауважень та побажань користувачів з метою підвищення якості програмного забезпечення за допомогою скриптової мови Python, месенджеру Telegram та бази даних SQLite.

3. Розроблено метод автоматизованого обліку комплексної оцінки відгуків користувачів, обґрунтовано основні положення формування та запропоновано концепцію створення автоматизованої інформаційної системи.

4. Виходячи з аналізу актуальності та необхідності розробки інформаційних систем, було вирішено розробити автоматизовану інформаційну систему, тому що частина функції (підсистем) керування або опрацювання даних здійснюється автоматично, а частина — людиною, яка буде використовувати чат-бота у месенджері Telegram, який зможе обробляти та направляти до бази даних побажання та зауваження користувачів з метою покращення сервісу чи програмного продукту, а та в свою чергу зберігати та працювати з відгуками.

5. Аналіз аналогів дав можливість зрозуміти, які типи та які категорії сучасних чат – ботів можна знайти у мережі. Дана розробка безкоштовна і буде знаходитись у відкритому доступі для всіх верств населення, у яких є смартфон, підключення до мережі Інтернет, а також месенджер Telegram. Дослідна експлуатація чат-боту довела конструктивність підходу та ефективність роботи.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Welcome to Python Telegram's bot documentation! Режим доступу: Welcome to Python Telegram Bot's documentation! — python-telegram-bot 13.5 documentation

2. Telegram Bot API [Електронний ресурс] - Режим доступу: https://core.telegram.org/bots/ap

3. Инструкция: Как создавать ботов в Telegram [Електронний ресурс] - Режим доступу: https://habr.com/ru/post/262247/

4. Пишем telegram-бота на python с помощью библиотеки telebot [Електронний ресурс] - Режим доступу: https://habr.com/ru/post/448310/

5. PyTelegramBotAPI GitHub - eternnoir/pyTelegramBotAPI: Python Telegram bot api.

6. Чат-боты: обзор и состояние технологий в отрасли [Електронний ресурс] – Режим доступу: http://nlpx.net/archives/425

7. Системна інтеграція https://tinyurl.com/2p64fvjy

8. Програмування Вікіпедія https://tinyurl.com/3bpjjuhx

9. https://www.invespcro.com/blog/the-importance-of-online-customer-reviews-infographic/

10. https://www.getresponse.ru/blog/nedovolnye-klienty

11. Проектування інформаційних систем Марченко Анна Вікторівна - https://elearning.sumdu.edu.ua/free_content/lectured:de1c9452f2a161439391120eef364dd8ce4d8e5e/20160217112601/content-20160217112601.pdf

12. Basic SQLite tutorial https://www.sqlitetutorial.net/

13. Системна інтеграція в інженерії Вікіпедія https://tinyurl.com/2hj752fz

14. Тестування Вікіпедія https://tinyurl.com/puy7y6fh

## Додаток

Програмний код реалізації автоматизованої інформаційної системи

Файл main.py

```
import telebot
from sqlGETS import *
from sqlSETS import *
from telebot import  types

bot = telebot.TeleBot("1877313657:AAFmiRqHdfTU3ZuJE-
CmwaDsIKM2NQuE5qQ")

usertable = default_write_table("users",
                        id="id INTEGER PRIMARY KEY",
                        name="name TEXT NOT NULL",
                        rating="rating TEXT",
                        description="description TEXT NOT NULL")


is_description = False


def rating_call(rating, call, connect, id):
    global is_description
    if call.data == rating:
        rating = "UPDATE users SET rating = {text} WHERE id = {id};".format(id=id,
text=rating)
        bot.send_message(
            call.message.chat.id, "Please, share your experience of using our app"
        )
        write_query(connect, rating)
        is_description = True


@bot.message_handler(commands=['start'])
def start(message):
    database = create_connection("users.db")
    new_users = default_add_info_table("users",
                            "id, name,rating,description",
                            user="('{id}', '{name}','{rating}','{description}')"
                            .format(id=message.chat.id,
                                    name=message.from_user.username,
                                    rating="None",
```

```
                                description=message.text))

    write_query(database, usertable)
    write_query(database, new_users)

    markup_inline = types.InlineKeyboardMarkup()
    item_yes = types.InlineKeyboardButton(text='write_feedback',
callback_data='yes')
    item_no = types.InlineKeyboardButton(text='help', callback_data='no')

    markup_inline.add(item_yes, item_no)
    bot.send_message(message.chat.id, 'I am glad to see you :) \n Select action, please',
                     reply_markup=markup_inline)


@bot.callback_query_handler(func=lambda call: True)
def answer(call):
    connect = sqlite3.connect('users.db')
    read_id = "SELECT users.id FROM users"
    id = read_query(connect, read_id)
    for i in range(len(id)):
        try:
            if id[i][0] == call.message.chat.id:
                id = id[i][0]
        except:
            pass
    rating = None
    if call.data == 'yes':
        set_rating(call.message)
    if call.data == 'no':
        help_command(call.message)
    rating_call("1", call, connect, id)
    rating_call("2", call, connect, id)
    rating_call("3", call, connect, id)
    rating_call("4", call, connect, id)
    rating_call("5", call, connect, id)


@bot.message_handler(commands=['rating'])
def set_rating(message):
    markup_inline = types.InlineKeyboardMarkup()
    item_1 = types.InlineKeyboardButton(text='1', callback_data='1')
    item_2 = types.InlineKeyboardButton(text='2', callback_data='2')
    item_3 = types.InlineKeyboardButton(text='3', callback_data='3')
```

```
    item_4 = types.InlineKeyboardButton(text='4', callback_data='4')
    item_5 = types.InlineKeyboardButton(text='5', callback_data='5')
    markup_inline.add(item_1, item_2, item_3, item_4, item_5)
    bot.send_message(
        message.chat.id,
        'Rate us', reply_markup=markup_inline
    )


@bot.message_handler(commands=['help'])
def help_command(message):
    bot.send_message(
        message.chat.id,
       'We use this bot to collect a feedback and wishes about our software. To leave a
response, click on the command /rating',
    )


@bot.message_handler(content_types=["text"])
def write_feedback(message):
    global is_description
    if is_description:
        database = create_connection("users.db")
        description = "UPDATE users SET description = '{text}' WHERE id =
{id};".format(id=message.chat.id, text=message.text)
        write_query(database, description)
        is_description = False
        bot.send_message(
            message.chat.id,
            'Thanks for your feedback, we are getting better with your help',
        )


bot.polling(none_stop= True)
```

Файл userClass.py

```
class User:
    name: str
    description: str
    rating: str

    def __init__(self, name: str = "", rating: str = "",
                 description: str = ""):
        self.name = name
```

```
        self.description = description
        self.rating = rating
```

Файл sqlGets.py

```
import sqlite3
from sqlite3 import Error


def read_query(connection, query):
    cursor = connection.cursor()
    result = None
    try:
        cursor.execute(query)
        result = cursor.fetchall()
        return result
    except Error as e:
        print(f"The error '{e}' occurred")
```

Файл sqlSets.py

```
import sqlite3
from sqlite3 import Error
DB = "BookDB.sqlLite"


def create_connection(path=DB):
    connection = None
    try:
        connection = sqlite3.connect(path)
        print("Connection to SQLite DB successful")
    except Error as e:
        print(f"The error '{e}' occurred")
    return connection

def write_query(connection, query):
    """Делает подключение к бд и передает в нее запись оставляю """
    cursor = connection.cursor()# object of db
    try:
        cursor.execute(query) # integrate values in db
        connection.commit()
        print("Query executed successfully")
    except Error as e:
        print(f"The error '{e}' occurred")
```

```
def default_write_table(name_table: str, **kwargs: str):
    fields = ""
    for args in kwargs.values():
        fields += args + ","
    fields = fields[:-1]
    create_users_table = """
    CREATE TABLE IF NOT EXISTS {name_table} (
        {args}
    );
    """.format(name_table=name_table, args=fields)
    return create_users_table


def default_add_info_table(name_table: str, fields: str, **kwargs: str):
    info = ""
    for args in kwargs.values():
        info += args + ","
    values = info[:-1]
    add_users = """
    INSERT INTO
     {name_table}({fields})
    VALUES
     {values};
    """.format(name_table=name_table, fields=fields, values=values)
    return add_users
```