КВАЛІФІКАЦІЙНА РОБОТА

Група ІПЗс-2019
Кишенюк А. М.

2023

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

на правах рукопису

Кишенюк Арсентій Мирославович

УДК 004.378

Розробка телеграм бота для знаходження столиць країн із використанням технології Python

Спеціальність 121 – «Інженерія програмного забезпечення»

Кваліфікаційна робота на здобуття кваліфікації бакалавра

Нормоконтроль

_____Стисло О.В.

(підпис, дата, розшифрування підпису)

Студент

_____Кишенюк А. М.

(підпис, дата, розшифрування підпису)

Допускається до захисту

Завідувач кафедри

_____к.т.н., доц. Пашкевич О.П.

(підпис, дата, розшифрування підпису)

Керівник роботи

_____к.т.н., доц. Ващищак С.П.

(підпис, дата, розшифрування підпису)

Івано-Франківськ – 2023

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

Освітній ступінь: «бакалавр»

Спеціальність: 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____

« ___ » _____ 2023 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Кишенюк Арсентій Мирославович

1. Тема кваліфікаційної роботи
Розробка телеграм бота для знаходження столиць країн із використанням технології Python
керівник роботи:
Ващищак Сергій Петрович, кандидат технічних наук
затверджена наказом вищого навчального закладу від «11» листопада 2022 року № 155/1НВ
2. Термін подання студентом роботи 14.06.2023
3. Вихідні дані роботи: Мова програмування Python
4. Зміст кваліфікаційної роботи
1. Аналіз та огляд існуючих програмних рішень
2. Огляд інструментів для реалізації проекту
3. Опис етапів розробки та реалізація телеграм бота
5. Дата видачі завдання 10.10.2022

КОНСУЛЬТАНТИ РОЗДІЛІВ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Розділ	Консультант (прізвище, ініціали та посада)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.	Огляд існуючих програмних рішень	24.02.2023	Виконано
2.	Огляд інструментів для реалізації проекту	02.03.2023	Виконано
3.	Опис етапів розробки та реалізація телеграм бота	30.04.2023	Виконано
4.	Оформлення пояснювальної записки	02.05.2023	Виконано
5.	Оформлення графічного матеріалу та підготовка до захисту роботи	02.05.2023	Виконано

Студент _____ Кишенюк А. М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Ващищак С.П.
(підпис) (прізвище та ініціали)

Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

АНОТАЦІЯ

Метою завдання було розробити телеграм бота, який буде виводити назву столиці за запитом.

Для досягнення мети було використано мову програмування Python, середовище для програмування PyCharm, а також засіб контроль версії Git.

Отриманим результатом є робочий бот який показує столицю країни за запитом, та повністю виконує поставлені завдання.

Покроково описано логіку розробки проекту.

Галузевою сферою застосування, як інтернет ресурсу.

ABSTRACT

The goal of the task was to develop a Telegram bot that would display the name of the capital upon request.

To achieve the goal, the Python programming language, the PyCharm programming environment, and the Git version control tool were used.

The resulting result is a working bot that shows the country's capital upon request and fully performs the assigned tasks.

The logic of project development is described step by step.

The field of application an Internet resource.

ЗМІСТ

ВСТУП

В сучасному світі зростає популярність месенджерів, які надають можливості для спілкування, отримання новин, розваг, освітньої складової тощо. Одним з таких месенджерів є Телеграм, який має понад 500 мільйонів активних користувачів по всьому світу та продовжує активно здобувати увагу нових користувачів.

У зв'язку з цим, можна виділити актуальність роботи, яка полягає в розробці різноманітних Телеграм-ботів, що допомагають користувачам в різних сферах життя. Одним із таких ботів є бот для знаходження столиці країни, який може бути корисним для людей, які подорожують або просто цікавляться географією.

Метою даної кваліфікаційної роботи є розробка Телеграм-бота з використанням мови програмування Python, який буде допомагати користувачам знаходити столицю країни, використовуючи систему інструментів і ресурсів в додатку API стороннього веб-сервісу.

Об'єктом дослідження є Телеграм-бот.

Предметом - розробка функціоналу для знаходження столиці країни засобами Python та Телеграм програмний інтерфейс API.

Завдання роботи полягають у наступному:

- дослідження можливостей використання Python та Телеграм API для розробки Телеграм-бота;
- дослідження теоретичної інформації, пов’язаною з розробкою бота в месенджері Телеграм;
- розробка функціоналу Телеграм-бота головна ціль якого знаходження столиці країни;
- тестування розробленого Телеграм-бота та виправлення всіх виявлених помилок.

Для досягнення поставленої мети та виконання завдань роботи були використані наукові методи дослідження.

Методи дослідження - аналіз літературних джерел з питань розробки Телеграм-ботів та використання API сторонніх веб-сервісів, експериментальний метод для тестування розробленого бота та метод моделювання процесів для розробки функціоналу.

Створений проект за рахунок можливостей додавання нового функціоналу потенційно може стати комерційним продуктом для вивчення географії, а також використовуватися освітніми закладами або туристичними компаніями.

Структура роботи: 3 розділи, загальний обсяг основної частини - 55 сторінок, список використаних джерел містить 23 посилання.

РОЗДІЛ 1. ОГЛЯД ІСНУЮЧИХ ПРОГРАМНИХ РІШЕНЬ

Пошук існуючих телеграм-ботів, які дозволяють знаходити назву столиці країни за її назвою, може бути часом досить складним завданням. Хоча в Інтернеті існує безліч сервісів та програм, які забезпечують такий функціонал, на перший погляд здається, що вони усі досить непродуктивні або просто на даний момент з причини відсутність хостингу.

Є декілька аргументів, які можуть пояснити, чому не вдалося знайти хороший існуючий аналог телеграм-бота для знаходження столиць за допомогою назви країни. Один з них - це відсутність подібних проектів, оскільки іншого функціоналу такі боти не мають і це доволі обмежений проект, хоч корисний, результативний та достатньо тренує навички розуміння програмування.

Крім того, можливо, що існуючі телеграм-боти для знаходження столиць за допомогою назви країни не такі популярні, якими здаються на перший погляд. Це може бути через те, що користувачі не знають про їх існування або не розуміють, яким чином вони можуть бути корисні. Невідомість і нерозуміння функціоналу телеграм-ботів може бути однією з головних причин, чому їх не використовують так активно.

Крім того, важливо пам'ятати, що розробка і підтримка телеграм-ботів для знаходження столиць за допомогою назви країни може вимагати певних фінансових витрат. Для того, щоб розробник міг підтримувати проект в робочому стані, необхідно мати ресурси для оплати хостингу та розробки нових функцій.

Також важливо зазначити, що деякі користувачі можуть вважати, що використання телеграм-ботів для знаходження столиць за допомогою назви країни не дуже зручне або не ефективне. Наприклад, для деяких користувачів може бути легше та швидше просто знайти відповідну інформацію в Інтернеті, або скористатися іншими програмами, які надають схожі послуги.

Отже, можна зробити висновок, що не вдалося знайти хороший існуючий аналог телеграм-бота для знаходження столиць за допомогою назви країни через декілька причин. Це може бути пов'язано з відсутністю проектів, які розміщені в Інтернеті, необізнаністю користувачів про функціонал таких програм, технічними проблемами, необхідністю фінансових витрат, а також з більш зручними альтернативами для користувачів. Також, існують схожі проекти, однак вони підтримуються лише мовою ворожої країни і описувати їх не має сенсу. Тому для роботи і для аналізу функціоналу будуть використовуватись не лише аналогічні за функціоналом телеграм - боти. Для ретельного дослідження будуть використовуватись усі можливі сервіси з ціллю перенести найкращі технічні рішення в єдиний телеграм бот, який буде розроблено в процесі виконання роботи.

1.1 Аналіз сервісу @world_capitals_quiz_bot

Телеграм бот "World Capitals Quiz" - це інтерактивна гра, яка дозволяє користувачам вивчати назви столиць різних країн та вдосконалювати свої знання з географії [16]. Скріншот інтерфейсу зображено на (рис. 1.1).

Рисунок 1.1 – Приклад роботи чат - боту «World Capitals Quiz»

Функціональність:

1. На початку гри бот випадковим чином питає користувача назву столиці певної країни.

2. Після того, як користувач відповідає на питання, бот повідомляє, чи є відповідь правильною, і надає користувачу можливість продовжити гру або завершити її.

3. Якщо користувач неправильно відповів на питання, бот повідомляє йому правильну відповідь.

4. У разі правильної відповіді, бот також повідомляє користувачу, скільки очок він отримав за відповідь, і загальний рахунок очок в грі.

Переваги:

1. "World Capitals Quiz" дуже простий та зрозумілий в користуванні. Користувач може грати в гру в будь-який зручний для нього час, що дозволяє використовувати цей бот як інструмент для вивчення географії у вільний час.

2. Бот надає користувачам можливість зануритися в світ географії та змагатися з іншими гравцями, які також користуються цим ботом.

3. Бот також мотивує користувачів грати ще більше, пропонуючи їм очки за правильні відповіді та ведення рахунку.

Недоліки:

1. Інтерфейс бота може бути трохи одноманітним, нудним та не цікавим для деяких користувачів.

2. Відсутність додаткових функцій, які б дозволяли користувачам отримувати додаткові знання про країни , що не пов'язані з назвами столиць, можуть знижувати інтерес користувачів до бота.

3. Немає можливості переглянути відповіді на запитання, які користувачі неправильно відповіли. Це може обмежити можливості користувачів в зануренні в процес вивчення географії.

Узагалі, "World Capitals Quiz" є досить ефективним та корисним інструментом для вивчення географії через Телеграм. Незважаючи на деякі

недоліки, цей бот може стати відмінним варіантом для тих, хто хоче покращити свої знання про столиці країн та відчути себе частиною онлайн-спільноти географів.

Цей бот найкраще візуалізує результати роботи по виконанню кваліфікаційної роботи. Практично, бот по знаходженню столиць країн за їх назвою має виглядати схожим чином та повертати дані за схожим принципом. Такий бот може використовувати певні кнопки, можливо опитування і буде корисним як для географів і студентів, так і для туристів і туристичних компаній.

1.2 Аналіз сервісу country.is

Сервіс country.is є онлайн-інструментом для вивчення географії та отримання інформації про країни світу [3]. У цьому аналізі буде розглянуто детально його функції та можливості.

1. Дизайн та інтерфейс

Перше, що сподобалося під час першого відвідування сервісу - це його дизайн та інтерфейс. Сайт має чистий та мінімалістичний дизайн, що робить його дуже легким у використанні. Крім того, інформація про кожну країну, яка включає назву, прапор, столицю, мову та валюту, відображається в компактному та організованому форматі.

2. Пошук

Один з головних функціональних елементів сервісу - це можливість швидкого та легкого пошуку країн за назвою. Цей функціонал дозволяє швидко знайти інформацію про будь-яку країну за кількома кліками. Крім того, сторінки з результатами пошуку містять детальну інформацію про країну, що дозволяє з легкістю знайти необхідну інформацію.

3. Карти

Сайт також має функціонал відображення країн на світовій карті. Це допомагає користувачам краще зрозуміти географічне розташування країн та їх взаємодію на глобальному рівні.

4. Список країн та їх столиць

Сервіс містить повний список країн світу та їх столиць. Це дозволяє користувачам швидко та безперешкодно отримати повну картину про країну, яку вони вивчають.

5. Статистика та факти

Сайт містить різні статистичні дані та цікаві факти про країни.

Продовжуючи аналіз сервісу https://country.is, можна сказати, що це є досить простий та зручний інструмент для отримання інформації про країни світу. Основна сторінка сервісу містить список всіх країн, в якому можна шукати необхідну користувачеві країну або відфільтрувати пошук країни за континентами.

Після вибору країни користувач отримує інформацію про неї: прапор, герб, назву столиці, населення, площу, мови та валюту. Додатково також дає змогу користувачам дізнатися про географічне положення країни та найвищу точку на її території.

Особливу увагу можна приділити дизайну та функціоналу сервісу. Сайт виглядає дуже сучасно та мінімалістично, що робить його зручним та легким у використанні. Крім того, можна зазначити, що сайт є адаптивним до різних розмірів екранів, що робить його доступним для користувачів на різноманітних пристроях.

Однак, слід зазначити, що сервіс не містить більш глибокої та детальної інформації про країни світу, яку можна знайти на інших ресурсах. Наприклад, не надається детальна інформація про історію, культуру, економіку та інші важливі аспекти країни. Також можна зазначити, що на сайті немає можливості шукати країни за ключовими словами або за функціями, такими як, наприклад, пошук країни за кліматом або за наявністю гірських країв.

Крім основної інформації про країну, сервіс також надає корисну статистику, наприклад, кількість населення, площу території, статус економіки та інші цікаві дані. Також є можливість подивитися на мапі, де знаходиться відповідна країна, або подивитися на знамена країн.

У загальному, сервіс country.is є корисним та зручним джерелом інформації про країни світу, зокрема для тих, хто шукає інформацію про бізнес-партнерів, туристичні маршрути та інше. Однак, він має обмеження та може не задовольнити потреби користувачів.

1.3 Аналіз сервісу countryreports

Сервіс countryreports.org є джерелом інформації про країни світу та їх культуру, історію, географію, економіку та багато іншого. Сервіс надає багато корисної інформації, такої як фотографії та відео, а також розповіді про історію та культуру країн [4].

На головній сторінці сервісу є пошукова функція, що дозволяє знайти інформацію про конкретну країну. Крім того, сервіс має список країн світу, що дозволяє легко знайти інформацію про будь-яку країну.

Один з найцікавіших розділів сервісу - це розділ про культуру країн. Він містить багато корисної інформації про культурні особливості та традиції різних країн. Зокрема, тут можна дізнатися про музику, танці, національні свята та інші культурні події країн.

Окрім того, на сайті є розділи про географію, економіку, історію та багато іншого. Кожен розділ містить багато корисної інформації про країни світу та їх особливості.

Сайт виглядає досить професійно та зручно для використання. Дизайн сторінок є чистим та простим, що дозволяє зосередитися на основній інформації. Також сайт має зручне меню, що дозволяє швидко знайти необхідну інформацію.

Однак, сервіс не має вбудованого тесту або бота, які допомагали б користувачам навчитися столиці країн. Це може бути недоліком для тих, хто хоче швидко хоче вивчити столиці. Вигляд інтерфейсу сайту представлено на (рис. 1.2).

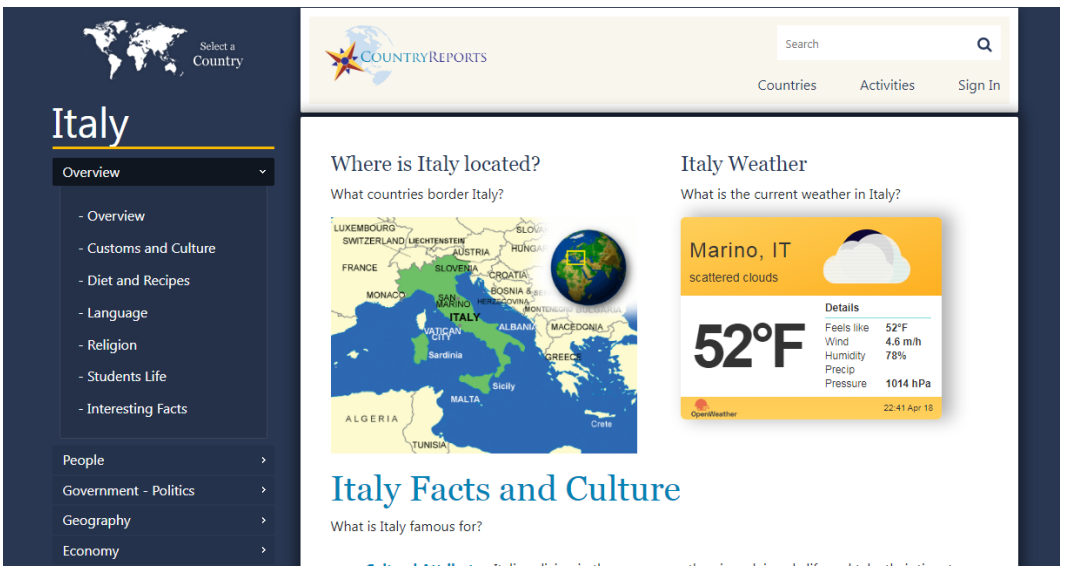


Рисунок 1.2 – Інтерфейс сайту countryreports.org

Огляд функцій та можливостей:

- сайт містить статистику, дані та факти про країни світу;
- користувачі можуть знайти інформацію про кожну країну світу, включаючи її історію, географію, населення, економіку, культуру та інше;
- на сайті є можливість використовувати інтерактивні карти, щоб дізнатися більше про різні регіони та країни;
- сервіс надає можливість знайти інформацію про візові вимоги, які необхідні для подорожей в різні країни світу;
- користувачі можуть використовувати функцію порівняння країн, щоб зрозуміти, як вони відрізняються за різними параметрами, такими як населення, економіка та інше.

Переваги сервісу:

- доступність безкоштовної та актуальної інформації про різноманітні країни світу;
- легкий інтерфейс сайту та проста навігація;
- інтерактивні карти та можливість порівняти різні країни за різними параметрами.

Недоліки сервісу:

- відсутність інформації про деякі країни;
- не завжди актуальна інформація;
- відсутність змоги отримати персоналізовану інформацію.

Враховуючи користь даної інформації, телеграм бот, який буде видавати столицю країни після введення її назви буде мати ще кнопку для отримання інформації та охоплення більшої аудиторії користувачів.

1.4 Порівняння отриманої інформації

Проект кваліфікаційної роботи повинен відповідати деяким стандартам. Серед них, достовірність отриманої інформації, можливість використовувати бот автономно та зрозумілість алгоритму роботи для пересічного користувача. На базі цих вимог буде проектуватись повнофункціональна система. Також, варто зазначити, що в даному проекті не потрібно враховувати вимоги до апаратного забезпечення. Усі операції з Телеграм ботом будуть виконуватись у месенджері Телеграм, а він є достатньо оптимізованим під будь-які пристрої. В тому числі на комп’ютери та смартфони.

Відносно самого проекту, прогнозується, що бот буде приймати від користувача команду з певним параметром, в даному випадку це назва країни та видаватиме її столицю автоматично. Також, проаналізувавши знайдені схожі застосунки, було прийняте рішення реалізувати функціонал кнопки, яка дозволить отримувати більше інформації про країну та столицю. У разі успішного виконання скріпта, користувач зможе

продовжити свою роботу з ботом. Також, будуть відповідні повідомлення про помилки, якщо користувач помилиться та введе не той параметр. Крім того, бот працюватиме автономно та з ним зможуть паралельно взаємодіяти декілька користувачів, а вся історія роботи з ботом буде доступна користувачу в окремому діалозі.

В проекті не передбачено надмірного дизайну, оскільки платформа Телеграм підтримує лише обмежену кількість елементів. Керування ботом здійснюватиметься за допомогою спеціальних команд, які будуть описані в ході виконання проекту.

1.5 Етапи розробки проекту та постановка завдання на роботу

Етапи розробку проекту повинні плануватися заздалегідь для того, щоб мати можливість оптимізувати безпосередньо сам процес зробити його більш визначеним та точним. Завдяки прогнозуванню подій відбувається значна економія ресурсів, а ефективність роботи отриманого програмного забезпечення залишається на високому рівні.

Етапи розробки проекту також по-іншому можуть називатися життєвим циклом програмного забезпечення, який включає в себе стадії, які проходить проект від імплементації до випуску (продакшн).

Зазвичай розрізняють такі основні та головні етапи життєвого циклу програмного забезпечення:

1. Аналіз.
2. Проектування.
3. Програмування.
4. Тестування.
5. Виправлення проблем (внаслідок тестування).
6. Реалізація.
7. Супровід та підтримка [19].

Перелік етапів може відрізнятися від індивідуальних особливостей діяльності окремої компанії чи розробника програмного забезпечення.

При визначенні етапів розробки проекту обов’язково потрібно згадати про модель життєвого циклу проекту, яка включає в себе варіативність тих чи інших етапів. Тобто можна сказати, що модель представляє собою досить таки узагальнене планування процесу розробки, завдяки якому з’являється можливість формувати уявлення про подальші дії з проектом.

Існують такі моделі життєвого циклу програмного забезпечення:

1. Каскадна або водоспадна модель.
2. Ітераційна модель.
3. Інкрементна модель.
4. Спіральна модель.

Важливо зазначити, що всі моделі значно відрізняються одне від одного, мають вагомі переваги та недоліки. Найбільш стандартною є каскадна модель, яка працює за наступним принципом: попередній етап закінчується і запускає наступний етап [20]. Тобто, розробка за каскадною моделлю відбувається таким чином, що увесь процес розробки проходить поступовим каскадним способом вперед без можливості повернутись на попередній етап. Ілюстрація каскадної моделі зображена на (рис. 1.3).

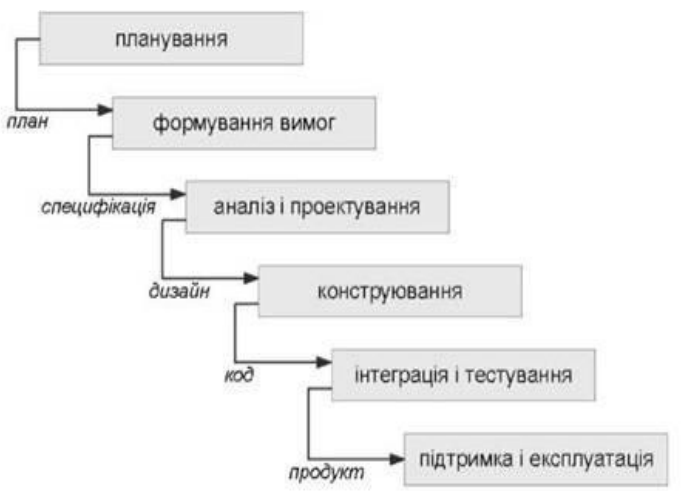


Рисунок 1.3 – Каскадна модель життєвого циклу програмного забезпечення

Деякі процеси створення продуктів вимагають виконання деяких функцій паралельно одна до одної. В каскадній моделі така можливість зберігається, але має обмежений характер, який виникає, коли вся система при переході до наступного етапу не може додатково дороблятись, а інтеграції паралельних частин проходить на попередньо обраному етапі.

Основним недоліком цієї системи є те, що помилки складно виявляти та в подальшому усувати, адже команда розробників займається власним етапом. Через це виникають проблеми з тим, що при реалізації застосунку неможливо чітко визначити, на якому етапі виникла помилка.

Складність цього процесу полягає в тому, що повернення назад у розробці тягне за собою витрату багатьох ресурсів. Також відсутня можливість внесення правок в ході виконання завдання чи надання додаткових ідей, адже повернення можливе лише по завершенню циклу.

При аналізі етапів розробки проекту також варто згадати про ітераційну модель, яка полягає в тому, щоб розбивати проект на декілька частини і проходити кожен етап життєвого циклу на кожній з них. Кожен етап починає бути окремим закінченим проектом, який в результаті стає готовим програмним продуктом [18].

Такий підхід застосовується тоді, коли потенційний результат проекту визначити неможливо, адже ідеї створюються в процесі розробки, як і можливість внесення різноманітних правок.

Особливість цієї моделі полягає в тому, що вона швидко адаптується та є більш стійкою до можливості глобального провалу в порівнянні з каскадною моделлю. Водночас основна проблема полягає в тому, що вона все ще не є достатньо оптимізованою, коли мова йде про витрати ресурсів. На (рис. 1.4) зображено візуальна імплементація ітераційної моделі життєвого циклу програмного забезпечення.

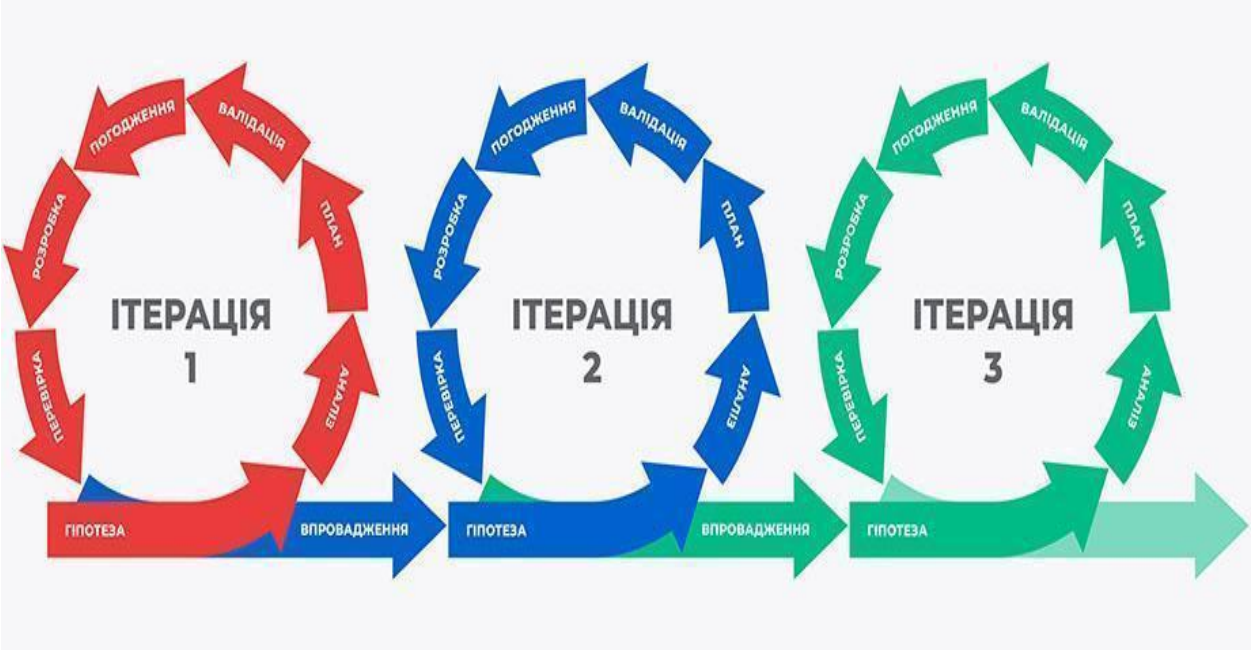


Рисунок 1.4 – Ітераційна модель життєвого циклу програмного забезпечення

Також варто розглянути спіральну модель життєвого циклу проекту, адже вона зображається витками, які характеризуються процес розробки [17]. Процес розробки повільно підіймається і на кожному витку(циклі) проходить один етап розробки. Результати кожного витка є кінцевою реалізацію готового продукту. Візуальне зображення спіральної моделі зображено на (рис 1.5).

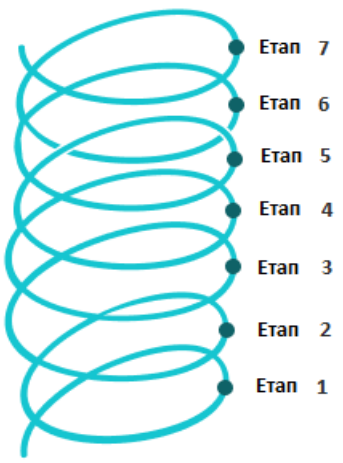


Рисунок 1.5 – Спіральна модель життєвого циклу програмного забезпечення

Ще однією моделлю, на яку варто звернути увагу, є інкрементна модель життєвого циклу проекту. Вона зосереджується на тому, щоб розширювати можливості готового проекту, добудовуючи до основної частини додаткові модулі чи функції.

Основна відмінність цієї моделі від каскадної полягає в тому, що там присутні лише два стани готового продукту: готовий проект чи не готовий проект, натомість в інтераційній моделі є можливість створення кілька прототипів програм.

В результаті, коли попередня версія починає використовуватися кінцевим споживачем, наступна версія починає плануватися чи розроблятися, за рахунок чого вона матиме більш розширений функціонал, ніж попередня.

На (рис. 1.6) зображено візуальне відображення інкрементної моделі життєвого циклу програмного забезпечення.

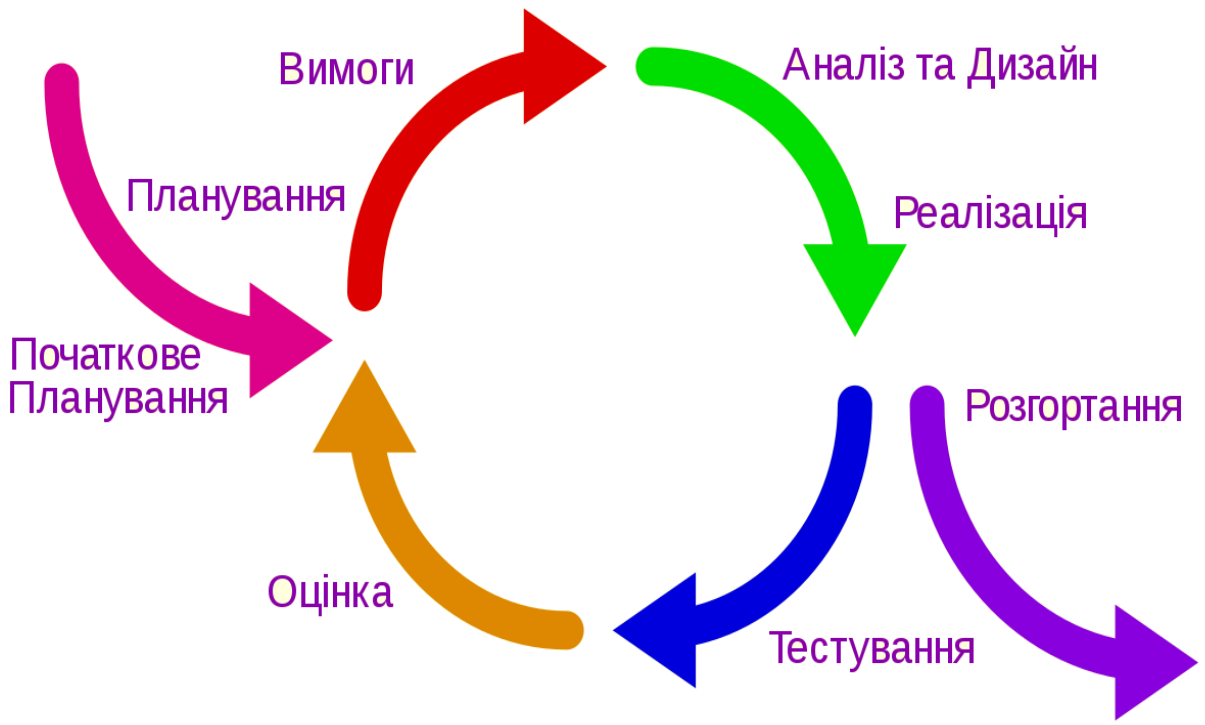


Рисунок 1.6 – Інкрементна модель життєвого циклу програмного забезпечення

Для Телеграм бота буде використовуватись інкрементна модель життєвого циклу програмного забезпечення, оскільки вона ідеально підходить під досягнення цілей даного проекту.

Підсумовуючи на проект кваліфікаційної роботи було поставлено такі цілі та завдання:

- розробити активний чат-бот, який буде виконувати свої функції на постійній основі ;
- розробити перелік команд для роботи з ботом, щоб користувачі без досвіду в програмуванні могли легко з ним взаємодіяти;
- можливість отримання додаткової інформації про країну;
- можливість використання бота не тільки одним, а декількома користувачами одночасно.

В ході виконання кваліфікаційної роботи прогнозується розробити повністю функціональне програмне забезпечення, яке виконуватиме поставлені завдання.

Висновки до розділу 1

У першому розділі було проведено огляд існуючих програмних рішень, які використовуються для подібного функціоналу, що планувався реалізувати в Телеграм-боті.

Аналіз сервісу @world_capitals_quiz_bot показав, що цей бот виконує функцію гри, де користувачі можуть перевірити свої знання столиць країн. Бот запитує питання та приймає відповіді від користувачів. Цей сервіс надавав інтерактивний та цікавий спосіб отримати інформацію та перевірити свої знання.

Сервіс country.is, який був проаналізований, надавав інформацію про різні країни, включаючи столиці, населення, географічне розташування та інше. Цей сервіс дозволяв отримати детальну інформацію про країни шляхом запитів користувача.

Сервіс countryreports також був проаналізований і надавав широкий спектр інформації про різні країни, включаючи географічні, економічні, соціальні та політичні дані. Цей сервіс був використаний для отримання детальних даних та статистики про країни.

Порівняння отриманої інформації з аналізу цих сервісів дало змогу поставити конкретні завдання на розробку Телеграм-бота. Враховуючи переваги та недоліки кожного з сервісів, були визначені ключові функції та особливості, які мали бути реалізовані у боті.

Таким чином, огляд існуючих програмних рішень дав змогу отримати інформацію про функціонал, доступний у подібних сервісах, та встановити вимоги до функціоналу розроблюваного Telegram-бота. Це забезпечило підґрунтя для подальшої розробки та реалізації проекту.

РОЗДІЛ 2. ОГЛЯД ІНСТРУМЕНТІВ ДЛЯ РЕАЛІЗАЦІЇ ПРОЕКТУ

2.1 Мова програмування Python

Python (укр. Пайтон) — високорівнева мова програмування, яку називають другою за популярністю в світі [22]. Застосовується в розробці вебзастосунків, програмного забезпечення, машинного навчання тощо. Мову програмування застосовуються в таких компаніях як Google, Instagram, Facebook, IBM, NASA,Netflix та інших [7].

Мова програмування Python використовується у таких сферах:

1. Створення програмних застосунків для різних галузей.
2. Робота над серверними частинами мобільних застосунків.
3. Створення комп'ютерних ігор для сучасних комп'ютерів (може бути повноцінним рушієм чи частково написаними на python).
4. Є частиною вбудовані системи для різних пристроїв.
5. Написання скриптів та розробка плагінів до уже реалізованих програм, які займаються автоматизацією процесів чи створення інших рішень.
6. Автоматизація процесу тестування готових застосунків.
7. Написання алгоритмів у сфері machine learning.

Python є мовою програмування загального призначення з відкритим кодом. Автором цієї мови програмування є Гвідо вам Россум, який розробив її на основі мови програмування ABC (яка на разі не існує). Основним завданням, яке виділяв для себе автор, є створення мови програмування, яка буде читабельною, матиме здатність розширюватися за рахунок модулів [7].

Перша версія Python була випущена в 1991 році, а більш повнофункціональна версія Python 2.0 була випущена в 2000 році. Обидва випуски з тих пір були припинені [7].

Python 3.0 був представлений у 2008 році, але він несумісний із попередніми версіями. Незважаючи на те, що Python включив утиліту

оновлення 2to3, це рішення було дуже суперечливим і створило значні проблеми для бази користувачів. Наразі Python Software Foundation адмініструє Python, і вони продовжують працювати над новими функціями та постійними покращеннями продуктивності [22].

Популярність цієї мови програмування активно зростає, тому вона входить до п’ятірки найбільш комфортних та оптимізованих мов програмування. Також за рахунок цих якостей вона використовується у машинному навчанні, штучному інтелекті, створенні веб-додатків тощо. В багатьох випадках Python використовується з зовнішніми фреймворками чи бібліотеками для пришвидшення процесу розробки.

Мова програмування Python використовує безліч концепцій, команд і структур, характерних також для інших популярних мов програмування. Тим не менш, є й відмінні риси, які стимулюють розробників до того, щоб надавати перевагу саме Python.

Важливими рисами цієї мови програмування є те, що вона дозволяє створювати чіткий дизайн коду і водночас робити його досить гнучким для внесення змін, доповнень чи оновлень.

Для того, щоб зрозуміти, як функціонує Python, важливо зрозуміти, якими є характеристики цієї мови програмування. В першу чергу вона вирізняється високим рівнем читабельності. У цьому відношенні Python схожий на інші мови програмування, включаючи JavaScript, Rust і C++, але він навіть більш зрозумілий і розбірливий.

Також важливою рисою Python є те, що він підтримує об’єктно-орієнтоване програмування (ООП): Python — це ООП-мова з підтримкою класів, методів, успадкування та інкапсуляції. На відміну від Java, Python не нав’язує ООП-модель, а принципи об’єктно-орієнтованого проектування суто необов’язкові. Тому можна використовувати Python виключно в Імперативному або процедурному режимі для не великих, коротких програм і простих утиліт.

Ще однією характеристикою Python є те, що це інтерпретована мова: розробникам не потрібно компілювати Python у асемблер або машинний код, як це нерідко відбувається у інших мовах програмування. Завдяки цьому, вище перелічиному, виникає значно менше проблем та витрачається менше часу на тестування програм.

Зокрема, при завершенні програми можна швидко запустити її, не використовуючи проміжні кроки, адже інтерпретатор, який є частиною Python, одразу має здатність розшифровувати кожен рядок. Це кардинально відрізняється від таких мов, як C/C++, що вимагають попередньої ручної компіляції від розробника (користувача). Python компілює програму до низькорівневого байт-коду для віртуальної машини Python (PVM) для інтерпретації та виконання [1].

Ще однією притаманною характерною рисою, яка вирізняє мову програмування Python серед інших, є динамічна типізація, яка не вимагає призначання типу, коли він використовується перший раз. Тип змінної визначається протягом виконання, залежно від її значення та способу її використання. Завдяки Python є можливість динамічно змінювати тип змінної протягом тривалості програми.

Ще одна особливість Python полягає в тому, що вона не залежить від платформи, завдяки чому її можна використовувати абсолютно на будь-якому застосунку.

Після окреслення характеристик мови програмування Python, варто окреслити переваги та недоліки загалом.

Основною перевагою Python є можливість надзвичайно просте використання за рахунок стислого та зрозумілого синтаксису. Програму легко читати, оскільки вона заснована на англійській мові. Також в мові програмування досить прості структури керування, які можна легко зрозуміти інтуїтивно.

Python динамічно типізується, тому не потребує оголошення кожної змінної, що робить цю мову програмування однією з найбільш ефективних та продуктивних мов.

Цю мову програмування легко вивчати, тому вона є вдалим та хорошим варіантом для користувачів-новачків. Максимальної ефективності може досягнути як користувач, який тільки почав вивчати програмування, так і розробник, який звик працювати з іншими мовами програмування, наприклад, C або Java. Пакет Python містить інтегроване середовище розробки та навчання (IDLE).

Python є універсальним засобом програмування, який підтримує процедурне і об'єктивно-орієнтоване програмування, а також може виконувати широке коло завдань, завдяки вбудованим та стороннім пакетам. Ця мова програмування домінує в сфері науки про дані та машинне навчання, а також має широке застосування для серверної веб-розробки та Інтернету речей. Перевагою також є те, що код Python можна вбудовувати в проекти, написані іншими мовами, наприклад C++, а код з інших мов можна вбудовувати в Python.

Завдяки простому синтаксису та універсальності, Python застосовується для швидкої і ефективної розробки, оскільки безпосередньо сама розробка займає значно меншу кількість часу.

Значна перевага мови програмування Python полягає в тому, що вона є надзвичайно портативною. Як уже було зазначено раніше, він не потребує компіляції, а тому користувачі можуть запускати одразу справжню програму, а не виконуваний файл. Це означає те, що будь-яка програма на мові Python може працювати на будь-якій системі, яка підтримує Python.

Ще однією, безумовно, вагомою перевагою є те, що Python не має вказівників і тому не потребує призначення вільного простору у пам'яті. Автоматичний розподіл пам'яті дозволяє автоматично виділяти пам'ять, а також переробляти її із відкинутих об'єктів. Це означає, що розробникам не

потрібно хвилюватися про так звані «скроблери», себто витоки пам’яті, недійсні посилання на покажчики або розмір кожного об’єкта окремо.

Значна кількість вбудованих об’єктів і бібліотек, які значно полегшують роботу з цією мовою програмування, є ще однією вагомою перевагою мови програмування Python. Вони включають в себе:

1. Велику кількість вбудованих складених об’єктів.
2. Списки.
3. Набори та словники тощо.

Ці об’єкти мають набір методів, які можна легко опрацьовувати, що застосовується для мережевих комунікацій, веб-інтеграцій, обробки даних та взаємодії з обладнанням. Таким чином процес написання програм стає значно швидшим та більш оптимізованим.

Додаткові сторонні бібліотеки є ще однією особливістю та перевагою мови програмування Python. Завдяки їм користувач може отримати доступ до багатьох безкоштовних зовнішніх бібліотек, які легко імпортувати та інсталювати за допомогою менеджера пакетів pip Python. Пакети можна завантажити зі сховища Python Package Index (PyPI). PyPI також дозволяє розробникам публікувати власні пакети [8].

Ще однією вагомою перевагою є відкритий код та можливість безкоштовного використання продукту. Також автори допустили можливість модифікування та безкоштовного поширення Python, що значно знизило витрати на розробку.

Проте, незважаючи на значні переваги, Python має також недоліки, на які варто звернути увагу та врахувати при створенні застосунків.

Зокрема, вагомим недоліком є те, що Python є значно повільнішим, ніж інші мови, такі як C чи Java. Python має здатність інтерпретуватися та динамічно типізуватися, за рахунок чого компілятор часу виконання повинен перевіряти тип кожної змінної.

Проблеми з використанням пам’яті також є викликом для розробників, які звикли користуватися Python. Вона використовує

надзвичайно багато оперативної пам’яті, тому її в деяких випадках доцільніше замінювати більш економною мовою програмування. Тим не менш, цей недолік компенсується простотою використання Python.

Важче уникнути помилок під час виконання: Python не компілюється до моменту виконання та динамічно типізується. Таким чином, багато проблем, які інакше були б виявлені компілятором, не з'являються, доки програма не запуститься. Це може включати щось таке просте, як синтаксична помилка, але це може включати проблеми, як спроба додати ціле число та рядок разом.

Python рідше застосовується у розробці мобільних програм, за рахунок того, що деякі засоби розробки є більш обмеженими, ніж фреймворки для інших мов.

Ще одним вагомим недоліком є відсутність оптимізованого доступу до бази даних. Частково це зумовлено тим, що Python не має потужного, високоякісного, легкого у використанні інтерфейсу, такого як Java Database Connectivity (JDBC). Його зручно використовувати, якщо бази даних є відносно простими, однак, не варто застосовувати для програм, які мають складні зв’язки в корпоративних базах даних.

Відсутність багатопотоковості є значним недоліком мови програмування Python, на який варто зважати при плануванні розробки програми. Замість цього використовується так звана багатопроцесорність, де кожен «потік» виконується в окремому процесі Python.

При розробці Телеграм бота буде використовуватись мова програмування Python. На вибір вплинули її переваги, а також недоліки, які можна було компенсувати, використанням правильних інструментів та бібліотек, описаних далі.

2.2 Інтегроване середовище розробки PyCharm.

В створенні Телеграм бота використовувалося інтегроване середовище розробки PyCharm, яке є одним з найбільш популярних IDE Python. Для цього є багато причин, у тому числі той факт, що його розроблено JetBrains, розробником популярної IntelliJ IDEA IDE, яка є однією з трьох найбільших Java IDE, і «найрозумнішою JavaScript IDE» WebStorm. Підтримка веб-розробки за допомогою Django є ще однією серйозною причиною [12].

Вважається, що PyCharm є однією з найбільш повних та всебічно інтегрованих середовищ розробки для роботи.

PyCharm, доступний як кросплатформенний додаток, сумісний із платформами Linux, macOS і Windows, також забезпечує підтримку версій Python 2 (2.7) і Python 3 (3.5 і вище) [5].

Головна особливість PyCharm в тому, що він має безліч модулів, пакетів та інструментів, а також легко підлаштовується під вимоги розробки та особисті вподобання.

Інтегроване середовище програмування, окрім аналізу коду виконує такі функції:

1. Є графічним налагоджувачем.
2. Є інтегрованим блок-тестером.
3. Підтримує інтеграцію систем контролю версій (vcs).
4. Підтримує науку про дані за допомогою anaconda.

Основною причиною, по якій PyCharm створив цю IDE, було програмування на Python і робота на кількох платформах, таких як Windows, Linux і macOS. Це середовище застосовується для того, щоб розробники могли створювати плагіни Python за допомогою різних доступних API [6].

PyCharm є досить поширеним середовищем розробки Python, яке містить в собі редактор коду та компілятор.

Також PyCharm має функцію автозаповнення, завдяки якій відбувається створення пропозицій під час написання коду. Це дозволяє зробити процес програмування більш ефективним, швидшим, а також застерегти від численних помилок.

Іншими стандартними функціями, які пропонує PyCharm, є:

1. Наявність вікна редактора проекту, яке дозволяє більш оптимізовано керувати файлами, а також організовувати їх.
2. Перевірка виводу коду.
3. Автоматичні пропозиції, які стосуються усунення та ліквідації помилок і попереджень.
4. Наявність низки модулів і пакетів, доступних в одному місці.

Для застосування такого середовища для програмування, необхідно заздалегідь визначити його переваги та недоліки, а також можливість нівелювання останніх.

До переваг PyCharm належать наступні:

- простота використання;
- інтеграція бібліотеки та автозаповнення;
- можливість переглядати вихідний код одним клацанням миші;
- швидкий розвиток;
- можливість виділяти помилку прямо в коді.

Однак PyCharm має також й мінуси, які можуть бути досить вагомими при веденні значних за обсягами проєктів:

- інтегроване середовище для програмування не підходить для початківців, оскільки автоматично пропонує велику кількість речей, а отже важко прослідкувати роботу поетапно;
- можуть виникати проблеми з налаштуванням інструментів.

Оскільки переваг цієї системи є значно більше, ніж недоліків, вона застосовувалася у розробці Телеграм бота.

2.3 Система контролю версії Git

Також для виконання роботи використовувалася система контролю версії Git, що надає командам розробників програмного забезпечення можливість отримувати декілька локальних копій кодової бази проекту, які розміщуються незалежно одна від одної [11].

Завдяки системі контролю версії Git з'являється можливість швидко створювати, об'єднувати та видаляти копії або гілки, завдяки чому команди мають змогу витрачати небагато ресурсів на обчислення перед об'єднанням з основною гілкою. Система контролю версії Git вирізняється своєю швидкістю, сумісністю робочих процесів і відкритим кодом.

Git має три стани файлів: modified, staged, і committed:

1. Файл «modified» змінюється, але не закріплюється в базі даних.
2. Файл «staged» налаштовується на перехід до «committed», яка являється функцією.
3. Коли файл «committed», дані зберігаються в базі даних [15].

Система контролю версії (яку по-іншому також іноді називають контролем джерел) застосовують для того, щоб відстежувати різні версії, а також знаходити й вирішувати конфлікти інтеграції в коді. Також цю систему можуть застосовувати для того, щоб безперешкодно спілкуватися, вносити зміни та відтворювати між розробниками та іншими членами команди. Ще вона допомагає в керуванні артефактами, зокрема, особливостями дизайну, даних, зображень тощо.

Для того, щоб мати змогу застосовувати Git на практиці, важливо визначити її переваги й недоліки. Зокрема, перевагами є:

- гнучкість робочого процесу;
- швидкість роботи;
- надійність;
- полегшення спільної роботи.

Git відкриває можливість спільної роботи, за рахунок наявності різних стратегій розгалуження, які є недоступними в інших системах контролю версії. В межах цієї особливості користувачі мають можливість самостійно обирати чи оптимізовувати робочий процес, в залежності від проєкту, команди чи окремих процесів.

Git надає локальний контроль версій. Завдяки цьому немає змоги перевіряти сервер для перевірки історії проєкту і визначення зроблених змін між версіями. Git має змогу швидко і миттєво виконати обчислення локальної різниці [15].

Система контролю версії має декілька резервних копій, оскільки кожен користувач має локальне сховище. Це надзвичайно вагома перевага, оскільки при збоях в системі з’являється можливість заміни основного серверу копією.

У разі відсутності мережі є можливість продовження виконання зобов’язань офлайн, що також є вагомогою перевагою системи контролю версії Git.

Наявність моделі розгалуження має вплив на спільну розробку, а саме значно полегшує її. Вона надає можливість створення гілок, проведення об’єднання коду з основними гілками, а також відстежувати зміни під час перегляду коду таким чином, щоб усі члени команди могли співпрацювати над запитами на злиття чи витягування.

Важливо визначити такою мінуси системи контролю версії Git, оскільки це допоможе прорахувати ресурси, які будуть витрачені на розробку боту для Телеграм.

По-перше, це середовище не підходить для користувачів-початківців, оскільки вимагає внесення локальних змін, організовування цих змін та об’єднання їх назад в основну гілку, що є досить складним процесом для вивчення та застосування.

По-друге, відсутність контролю доступу окремих елементів може бути проблемою при використанні системи контролю версії Git. Користувач

може застосувати обмеження на створення гілок та об’єднання змін в основному сховищі, проте, кожен, хто має доступ до цього сховища, буде мати доступ до всього всередині нього.

Обробка та зберігання великих бінарних файлів часто є не ефективною. Система контролю версії Git немає можливості стискати такі файли, а отже розмір сховища може експоненціально зростати з кожною зміною великого бінарного файлу [9].

2.4 Бібліотеки, фреймворки та API

При створенні застосунків з використанням мови програмування Python часто використовуються бібліотеки, фреймворки та API, які є важливими елементами оптимізації та підвищення ефективності системи. Спочатку варто розглянути основні бібліотеки, які можуть застосовуватися для створення Телеграм бота [2].

Спершу варто виділити Python телеграм-bot — це обгортка для API Телеграм Bot на Python з великою кількістю функціональності, такою як розсилки повідомлень, обробка вхідних повідомлень, інтеграція з клавіатурами, меню і багато іншого.

Також цікавим застосунком є pyTelegramBotAPI. Це проста, але розширювана бібліотека для API Telegram Bot на Python. Відрізняється легким синтаксисом та можливістю легко та швидко створювати ботів з різною функціональністю.

AIOGram — це досить проста та повністю асинхронна бібліотека для API Telegram Bot на Python, яка написана з використанням asyncio та aiohttp. Вона дозволяє розробникам створювати ефективні та швидкі боти з підтримкою асинхронних запитів.

Бібліотека TGramBot — це частково автоматично створена асинхронна платформа для розробки Телеграм-ботів на Python з підтримкою Minimal Telegram Bot API. Вона має досить легки для

розуміння синтаксис та дозволяє розробникам швидко створювати ботів з базовою функціональністю.

OrigamiBot — це бібліотека для розробки Телеграм-ботів на Python з підтримкою API Telegram Bot. Використовується для того, щоб розробники мали змогу створювати ботів з різноманітною функціональністю, такою як створення групових чатів, надсилання повідомлень, додавання користувачів і багато іншого.

Також для створення ботів в Телеграм можна застосовувати pytgbot. Це модуль для доступу до API Telegram Bot на Python, що дозволяє розробникам швидко створювати ботів з базовою функціональністю.

Вдалим рішенням є використання teleflask — це фреймворк на основі Flask та pytgbot для розробки Телеграм-ботів на Python, який використовується розробниками для того, щоб створювати ботів з різноманітною функціональністю, зокрема, створення клавіатур, обробка повідомлень тощо.

Останнім модулем є telegram-text, який застосовується для того, щоб мати змогу з легкістю створювати текст з форматуванням в Телеграм. Він може бути використаний в будь-якому фреймворку Python, який підтримує роботу з текстом.

У роботі використовувалася асинхронна бібліотека AIOGram для того, щоб надати інтерфейс для створення та розробки ботів для месенджерів, зокрема Телеграм, використовуючи API ботів Телеграм.

Основна перевага AIOGram полягає в тому, що вона підтримує асинхронну обробку запитів, що дозволяє боту працювати зі швидкістю, не блокуючи інші процеси. Крім того, AIOGram має досить простий та зрозумілий інтерфейс, що дозволяє дуже швидко створювати та налаштовувати ботів.

Основні функції, які пропонує AIOGram виглядають так:

1. Робота з повідомленнями — надсилання, отримання та обробка повідомлень в реальному часі.

2. Робота з клавіатурами — створення та налаштування клавіатур для взаємодії з користувачами.

3. Робота з файлами — отримання та відправка файлів.

4. Робота з базою даних — зберігання та отримання даних про користувачів, повідомлення та інше.

Крім того, AIOGram підтримує використання плагінів, які розширюють функціональність бота та дозволяють найпростішим способом налаштовувати його під конкретні потреби.

AIOGram — це бібліотека, яка дозволяє створювати різноманітних ботів для Телеграм зі значними можливостями та простим інтерфейсом.

2.5 Меседжер Телеграм

Телеграм — це месенджер, який використовується як інструмент для миттєвого обміну повідомлення навіть в тому випадку, коли користувач не знає номеру телефона свого співрозмовника. Така функція є можливою завдяки протоколу зв’язку, відомого як MTProto, який дає можливість відкривати різні сеанси на кількох пристроях без одночасного підключення. Програма була створена в 2013 році братами Павлом і Миколою Дуровими як альтернативний спосіб спілкування поза WhatsApp з можливістю обмінюватися файлами, що зберігаються у власній хмарі, а також наявністю наскрізного шифрування [13].

Телеграм є багатофункціональною програмою, яка дозволяє не лише надсилати повідомлення, а й обмінюватися файлами, зображеннями, або ж створювати інтерактивні боти.

В першу чергу Телеграм є платформою миттєвого обміну повідомленнями між співрозмовниками, які є контактами одне одного або мають бажання залишити свій номер телефону приватним. Велику роль грає конфіденційність, тому акаунти можна приховати. При цьому спілкування надалі буде можливим.

Телеграм надає можливість створити ім’я користувача (яке не обов’язково збігатися з реальними даними), за допомогою якого відбуватиметься пошук через пошукову систему.

Також Телеграм є платформою для надсилання повідомлень різного характеру, зокрема, відео, аудіо, для відеоконференцій на комп’ютері чи інших пристроях.

Однією з переваг Телеграм, яка робить цю платформу цікавою користувачам та розробникам, є можливість зберігання великих об’ємів даних на власній хмарі, допоки користувач не видалить їх самостійно.

При цьому, обсяг місця для зберігання є нескінченним, а отже є можливість зберігати будь-які відео, аудіо записи.

Ще однією перевагою Телеграм серед інших месенджерів є можливість створення окремих чатів, а також поділ їх на різну тематику для оптимізації процесів, зокрема, у робочому спілкуванні.

При відсутності Premium версії розмір при надсиланні файлу буде обмеженим та не може перевищувати 15 Гб.

Обмін повідомленнями в Телеграм можливий не лише між окремими користувачами, а й в групах і супергрупах, кількість учасників яких може перевищувати 20 000 [14]. Це корисно, наприклад, для людей, які працюють у великих компаніях і мають виконувати багато процесів.

Також Телеграм дозволяє створювати канали, які можуть бути не лише розважальними чи представляти собою платформу для зберігання контенту в хмарі, а й інструментом для підприємств, де адміністратор може надсилати повідомлення в односторонньому порядку. Такі канали мають змогу бути публічними чи приватними, а також є можливість виступати каналом зв’язку з спільнотою.

Зважаючи на високе різноманіття функцій Телеграм, було прийняте рішення створювати бот саме з застосуванням цієї платформи, а також мови програмування Python.

Висновки до другого розділу

У даному розділі було проведено огляд інструментів, які були використані для реалізації проекту Телеграм-бота.

Мова програмування Python була вибрана як основна мова для розробки бота. Python має багатий набір бібліотек та фреймворків, що сприяють швидкій та зручній розробці. Вона є популярним вибором для розробки ботів, оскільки має простий синтаксис, розширені можливості обробки даних та машинного навчання.

Інтегроване середовище розробки PyCharm було використано для зручного написання, налагодження та керування проектом. Воно забезпечує багатий набір функціональностей, таких як автодоповнення, відлагодження, керування залежностями та багато інші, що роблять роботу розробників в рази легшою.

Git дозволяє відстежувати та керувати змінами в коді проекту. Git надає можливість працювати з різними версіями коду, спільно працювати над проектом, а також забезпечує збереження історії змін.

Було розглянуто бібліотеки, фреймворки та API, які були використані під час розробки бота. Ці інструменти допомагали виконувати певні функції та розв'язувати завдання проекту. Наприклад, python-telegram-bot надавала зручні засоби для взаємодії з API Telegram, а бібліотека requests дозволяла здійснювати HTTP-запити до інших зовнішніх сервісів.

Месенджер Телеграм був обраний як основна платформа для функціонування бота. Він має широкий функціонал та добре задокументоване API.

В результаті огляду цих інструментів було визначено набір технологій та засобів, що відповідають вимогам проекту розробки Telegram-бота. Вони дозволили забезпечити ефективну розробку, гнучкість та функціональність бота, а також зручне управління проектом та Git контроль версій.

РОЗДІЛ 3. ОПИС ЕТАПІВ РОЗРОБКИ ТА РЕАЛІЗАЦІЇ ТЕЛЕГРАМ БОТА

3.1 Особливості логіки та використаних алгоритмів в проекті

Розробка будь – якого Телеграм бота розпочинається з ініціалізації його в спеціальному боті, який є «батьком» всіх ботів цієї платформи. Кожному боту присвоюється унікальний токен. Токен – це спеціальний API ключ, який слугує ідентифікатором ботів на цій платформі.

Для початку необхідно зареєструватись в Телеграм та отримати API ключ. Це можна зробити, створивши новий акаунт в Telegram та звернувшись до бота @BotFather, щоб створити нового бота. Після створення бота @BotFather поверне API ключ, який потрібно буде використовувати для зв'язку з сервером Телеграм. Для цього вводимо відповідну команду (рис. 3.1), копіюємо токен і тег бота та імпортуємо в середовище PyCharm.

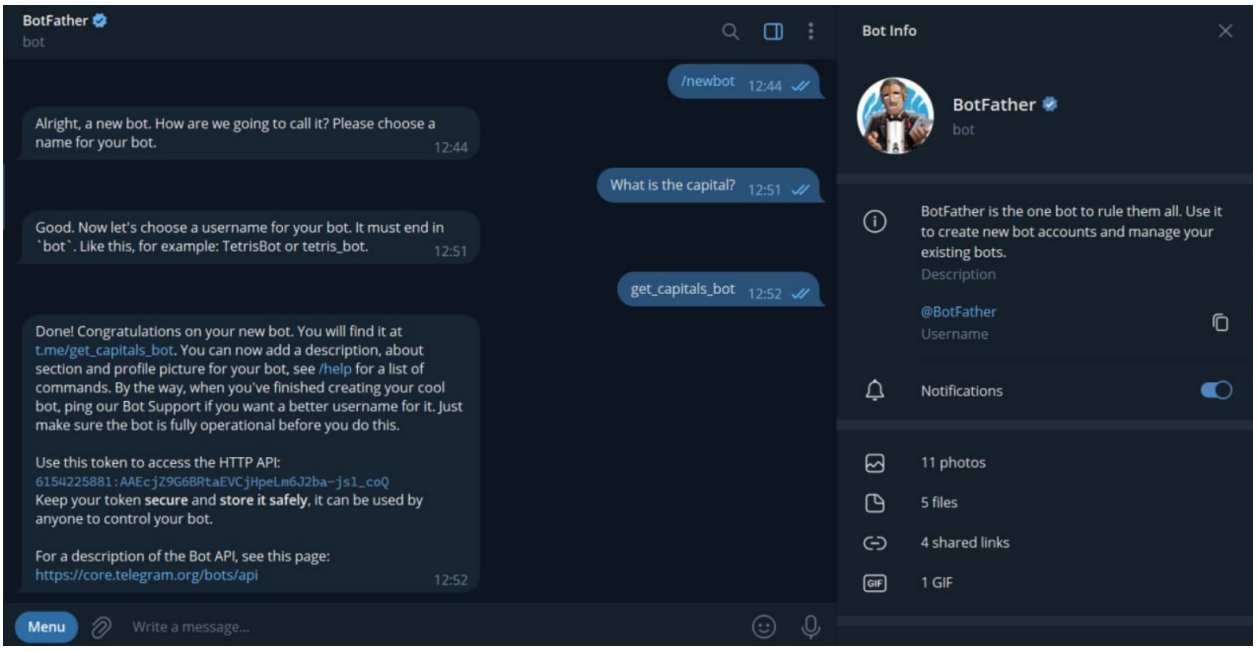


Рисунок 3.1 – Створення бота у @BotFather

Першою і головною командою Телеграм бота буде /capital. Вона відповідатиме за парсинг повідомлення користувача, аби задіяти основну функцію бота відповідно до вказаних користувачем параметрів. Після запуску цієї команди, прогнозується, що буде отримано лише країну та більш детальну інформацію про неї. Для більш правильної роботи боту використовуємо спеціальний хенделер, код якого зображено на (рис. 3.2).

"Хендлер" (англ. handler) - це функція, яка виконує певні дії при отриманні певного повідомлення від користувача. У контексті створення телеграм бота, хендлер на команду /capital означає функцію, яка буде виконуватись під час отримання команди "/capital" від користувача бота.

Рисунок 3.2 – Handler на команду /capital

Парсинг повідомлення користувача відбувається з урахуванням можливості користувачем надавати інші команди. Однак, бот прийматиме лише назву країни для того, щоб видати результат. Такий функціонал доступний за рахунок використання спеціальних бібліотек, котрі були згадані та описані вище.

Відповідь користувача програмується за допомогою спеціальної функції в бібліотеці для роботи з Телеграм.

Функція message.reply_text() є одним з методів об'єкта повідомлення (message) в бібліотеці pyTelegramBotAPI, який дозволяє нам відправити відповідь на повідомлення, яке отримав бот. Ця функція відправляє у відповідь текстове повідомлення відповідно на оригінальне повідомлення, до якого ви долучаєте цей метод. Результат відправки буде показано в розділі тестування.

Для того, щоб бот працював стабільно та швидко, потрібно імпортувати спеціальний модуль, який буде оновлювати сервери з певним не великим інтервалом.

У контексті програмування телеграм-ботів, таким сервісом є «екзек'ютер». Він означає частину програмного коду, яка відповідає за виконання різних функцій бота. В бібліотеці pyTelegramBotAPI для цього є призначений об'єкт Updater, який відповідає за сприйняття повідомлень від серверів Телеграм та відповідає за передачу їх на обробку до наших функцій (handlers).

Для того, щоб бот міг працювати, ми маємо запустити цей Updater і встановити зв'язок з серверами Телеграм, щоб постійно отримувати оновлення з повідомленнями від користувачів. Для цього ми використовуємо метод start_polling() у об'єкта Updater.

Отже, коли ми запускаємо цю функцію, бот починає постійно опитувати сервери Телеграм на наявність нових повідомлень. Коли бот отримує нове повідомлення від користувача, Updater передає його до відповідної функції (handler), яка відповідає за обробку цього повідомлення та відправку відповіді назад користувачу.

Таким чином, додатково до імпортування бібліотек та створення функцій, ми також імпортуємо Updater та запускаємо метод start_polling() для початку нескінченного опитування серверів Телеграм на наявність нових повідомлень. Це дозволяє боту постійно працювати та готовому до обробки нових повідомлень від користувачів.

Для більш розширеного функціоналу створюємо нову функцію handler не на повідомлення, а на команди боту. Коли ми створюємо хендлер на команду, то функція, яку ми створюємо, буде викликатися тільки при введенні користувачем цієї команди в чаті з ботом. Інші повідомлення від користувача, наприклад, текстові повідомлення, фотографії чи відео, не будуть сприйматися цим хендлером, і нам потрібно буде створити окремий хендлер для обробки повідомлень.

Цей handler відповідатиме за статистику, яку можна збирати від взаємодії користувачів з ботом. Фрагмент коду наведено на (рис. 3.3).

```
import countryinfo
from aiogram import Bot, types
from aiogram.dispatcher import Dispatcher
from aiogram.contrib.fsm_storage.memory import MemoryStorage
from aiogram.utils import executor

import sqlite3

conn = sqlite3.connect('./users.db', check_same_thread=False)
conn.row_factory = lambda k, l: {c[0]: l[i] for i, c in enumerate(k.description)}
cur = conn.cursor()

cur.execute(
    """
        CREATE TABLE IF NOT EXISTS users_requests(
            id INT,
            username TEXT,
            request TEXT
        )
    """
)


bot = Bot(token="6154225881:AAEcjZ9G6BRtaEVCjHpeLm6J2ba-js1_coQ")
dp = Dispatcher(bot, storage=MemoryStorage())
```

Рисунок 3.3 – Фрагмент коду handler статистики

На цьому фрагменті ми імпортуємо базу даних, створюємо локальне сховище, налаштовуємо відповіді сховища у вигляді словника, створюємо курсор та даємо запит на створення бази даних.

Коли користувач взаємодіє з телеграм-ботом, він може відправляти різні запити, наприклад, запити на отримання інформації або на виконання певних дій. Для того, щоб зберігати ці запити та додаткову інформацію про користувача (наприклад, ідентифікатор чату, з якого був відправлений запит), можна створити функцію для записування запитів в базу даних або в файл. Вона буде створена для запису статистики чатів з ботом.

Ця функція може допомогти зберігати історію запитів користувачів, а також зробити аналіз запитів для покращення функціональності бота і зручності взаємодії з користувачами. Наприклад, можна збирати статистику

про популярні запити, що може допомогти зрозуміти потреби користувачів і покращити функціональність бота.

Також для більш широкого зацікавлення користувачів, було прийняте рішення створити кнопку після виводу столиці для визначення додаткової інформації про столицю чи країну. Робимо окрему функцію та додаємо кнопку з додатковою інформацією. За нею також закріплюємо хендлер, структуруємо вивід інформації та оптимізовуємо функції. Результат зображено на (рис. 3.4).

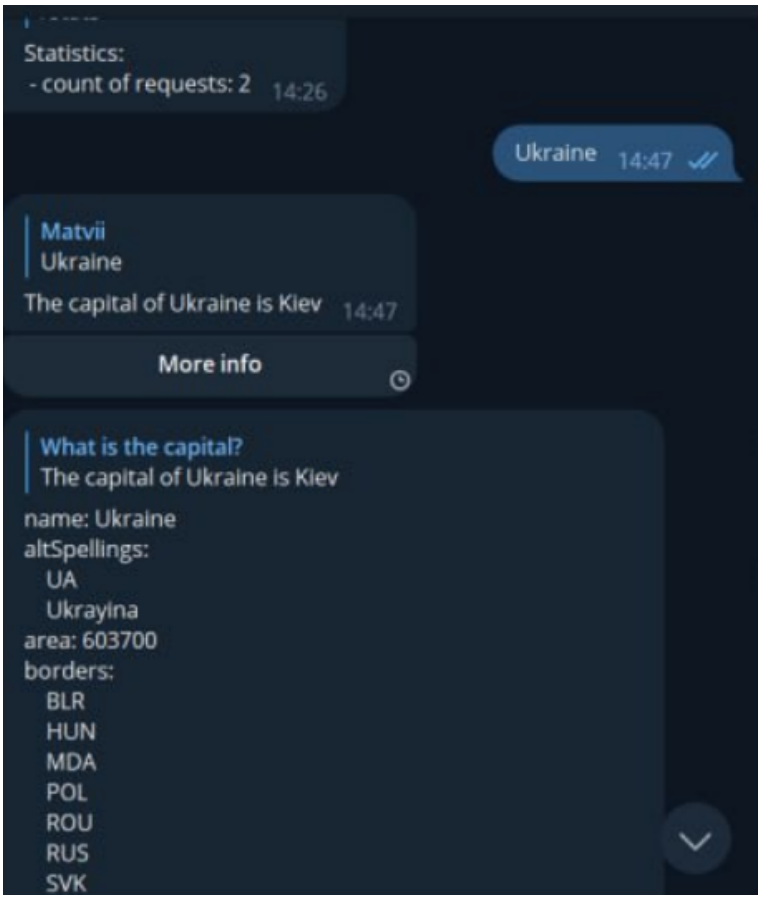


Рисунок 3.4 – Результат додавання та використання кнопки.

Також, важливо забезпечити можливість аналізу помилок боту, якщо такі будуть. Такі помилки можуть статися від неправильної взаємодії користувача з ботом, а також через інші технічні причини, такі як різні дані або не справність програмного коду. Для того, щоб таких моментів знизити до мінімума для цього додаємо обробку помилок і виправляємо всі

неточності. Готове програмне забезпечення запускатиме створений бот в вигляді, який зображено на (рис. 3.5).

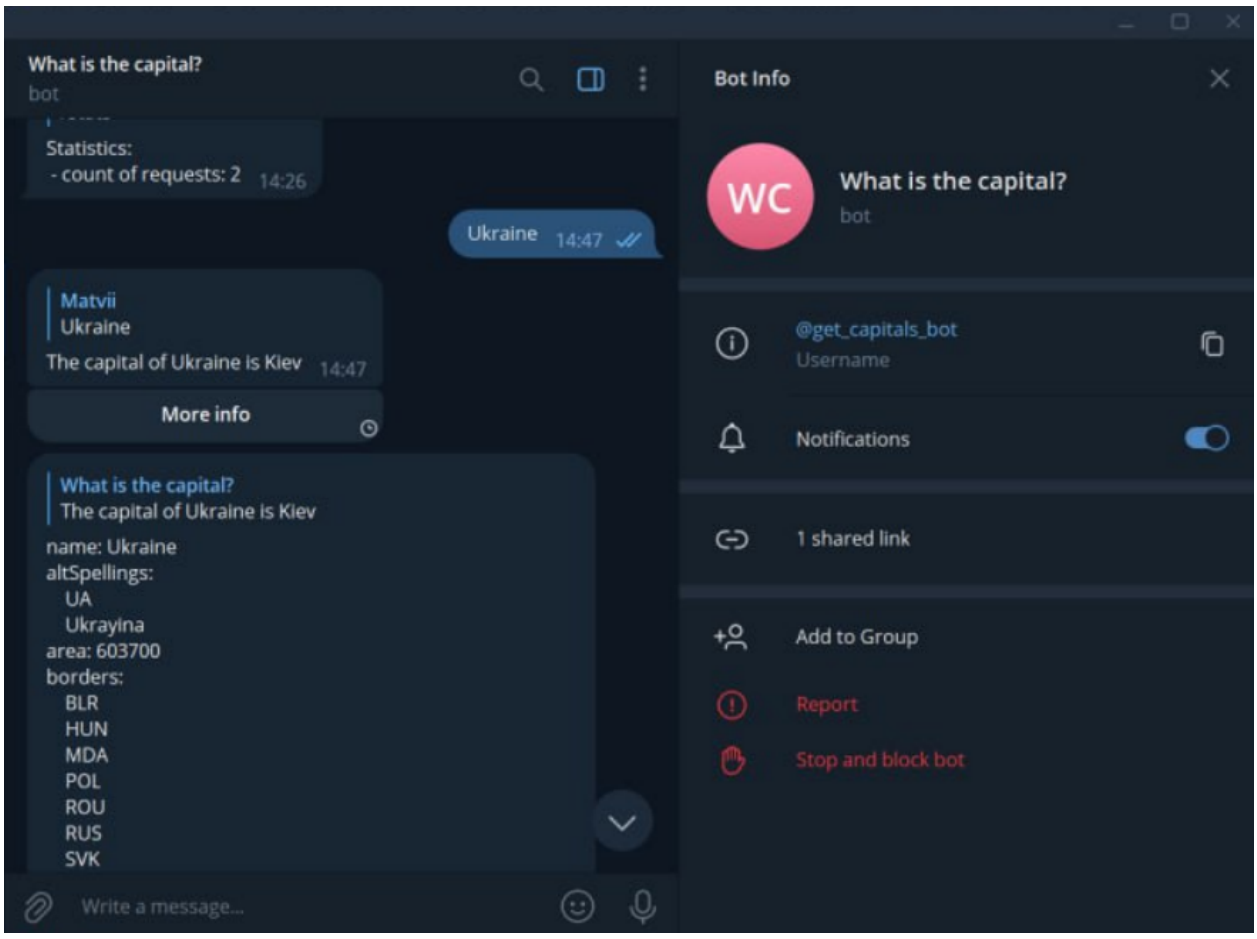


Рисунок 3.5 – Вигляд готового програмного забезпечення

Як результат отримуємо готовий чат – бот на платформі. Для його роботи потрібно лише завантажити файли проекту на хостинг.

3.2 Огляд розробленого інтерфейсу

Телеграм – боти мають доволі обмежений візуальний інтерфейс. Всі вони стандартні та пропонують звичайний набір функцій. Однак, в проекті було використано певні особливості. Такий інтерфейс дозволяє боту бути більш інтерактивним та забезпечує більшу взаємодію з користувачем, що може покращити користувацький досвід. Крім того, це може допомогти

збільшити ефективність роботи бота, оскільки кнопки дозволяють користувачам взаємодіяти та здійснювати дії швидше та з меншою кількістю введення тексту. Алгоритм роботи користувача з ботом буде мати вигляд наступним чином:

- Користувачу дається змога надсилати запити боту за допомогою текстових повідомлень.
- Бот оброблює запит та генерує відповідь у вигляді тексту, який містить необхідну інформацію, а також кнопку.
- Кнопка має текстову назву або символьний значок, і вона призначена для того, щоб користувач міг здійснити певну дію, у нашому випадку отримати більш поглиблену інформацію.
- Користувач натискає на кнопку і бот відповідно оброблює дію, що пов'язана з кнопкою.

Таким чином, вдалось модифікувати функціонал чат – боту.

3.3 Джерела даних

Даний код є фрагментом програми на мові Python, яка використовує бібліотеку sqlite3 для зберігання та обробки даних в базі даних SQLite.

SQLite - це вбудовувана, сервер-менеджерна, реляційна база даних, яка не вимагає окремого сервера та може бути використана в якості локальної бази даних для різних додатків та програм. Бібліотека sqlite3 є інтерфейсом мови програмування Python до цієї бази даних.

sqlite3 дозволяє створювати, зберігати та обробляти дані у форматі таблиць з декількома стовпцями та рядками. Кожен стовпець має свій тип даних (наприклад, ціле число, десяткове число, рядок, дата тощо), який визначає допустимі значення у відповідному стовпці.

Для роботи з базою даних у sqlite3 необхідно виконати кілька кроків: встановити з'єднання з базою даних, створити таблиці та схему бази даних,

вставляти, змінювати та видаляти дані у таблицях та виконувати запити для отримання необхідної інформації.

```
import sqlite3

conn = sqlite3.connect('./users.db', check_same_thread=False)
conn.row_factory = lambda k, l: {c[0]: l[i] for i, c in enumerate(k.description)}
cur = conn.cursor()

@dp.message_handler(commands=['stats'])
async def stats(message: types.Message):
cur.execute(f'SELECT * FROM users_requests;')
users = cur.fetchall()

Copy
await message.reply(f'Statistics:\n - count of requests: {len(users)}')
def reg_user(user_id, user_name, request):
cur.execute(
'INSERT INTO users_requests (id, username, request) VALUES (?, ?, ?);',
(user_id, user_name, request,)
)
conn.commit()
```

У даному коді показано, як встановити з'єднання з базою даних у файлі ./users.db, створити таблицю users_requests, яка містить три стовпці id, username та request, та як вставити дані у цю таблицю за допомогою функції reg_user(). Також, показано як виконати запит до таблиці за допомогою методу execute() та отримати результати у вигляді списку словників, де кожен словник містить значення для кожного свого квідповідного стовпця таблиці.

У загальному, sqlite3 є зручним та легким у використанні інструментом для роботи з базами даних у Python, який може бути використаний для зберігання та обробки даних у різних програмах.

Код створює з'єднання з базою даних, вказаною у файлі ./users.db, та встановлює параметр check_same_thread=False, щоб дозволити з'єднанням працювати в різних потоках. Після цього встановлюється функція для отримання результатів запитів до бази даних у вигляді словника, де ключами є імена стовпців, а їх значеннями є значення з рядка таблиці.

Далі, за допомогою декоратора, функція stats() відповідає на команду /stats в месенджері та виконує запит до бази даних, щоб отримати кількість записів у таблиці users_requests. Отримана кількість записів повертається у відповідь користувачеві.

Нарешті, функція reg_user() призначена для додавання нового запису у таблицю users_requests. Функція отримує параметри user_id, user_name та request, які вставляються у відповідні відведенні для цього стовпці таблиці. Після цього зміни зберігаються у базі даних за допомогою методу commit().

Цей фрагмент коду може бути використаний у програмах для зберігання та обробки даних користувачів, які взаємодіють з ботом у месенджері. Наприклад, він може бути використаний для зберігання запитів користувачів, щоб через деякий проміжок часу аналізувати їх та покращувати функціональність бота.

3.4 Тестування проекту

Процес тестування повинен відповідати загальним стандартам та нормам ефективності, за рахунок чого ефективно відображати помилки роботи програми. Для цього потрібно дати характеристику тестування та зрозуміти, які основі етапи може включати цей процес.

Отже, тестуванням називають такий процес, який дозволяє визначити якість розробленого програмного забезпечення на основі перевірки

функціональності, швидкодії та правильності готового продукту [21]. В основному, тестування відбувається на основі тих аспектів, з якими може стикнутися користувач безпосередньо при роботі з проектом. Однак нерідко за наявності ресурсів розробники запрошують для тестування спеціалістів, які займаються тестуванням технічних частин проекту, зосереджуючись на міжнародних стандартах. Знайдені помилки в процесі тестування обов'язково виправляються (з урахуванням особливостей оптимізованого витрачання ресурсів), а тестування проводиться повторно.

Важливо зазначити, що якість кінцевого продукту неможливо виміряти в абсолютному показнику, оскільки кожен фахівець визначає цей показник для себе самостійно, отже величина є радше суб'єктивною. Можна сказати, що в цьому полягає основна проблема проведення процесу тестування, оскільки не існує ідеального програми та коректності її роботи.

Існує надзвичайно багато методів та підходів до тестування, які перетворюють цей процес на цікаве та творче завдання. З урахуванням цього варто розуміти основні аспекти тестування, які є невід'ємними для кожного циклу розробки проекту:

- тестування має великий та знвчущий вплив на весь життєвий цикл програмного забезпечення;
- тестування - ітераційний процес, який залежить від кількості знайдених та виправлених помилок;
- успішне тестування не доводить повну відсутність помилок, а лише їхню наявність;
- правильне тестування забезпечується документацією [21].

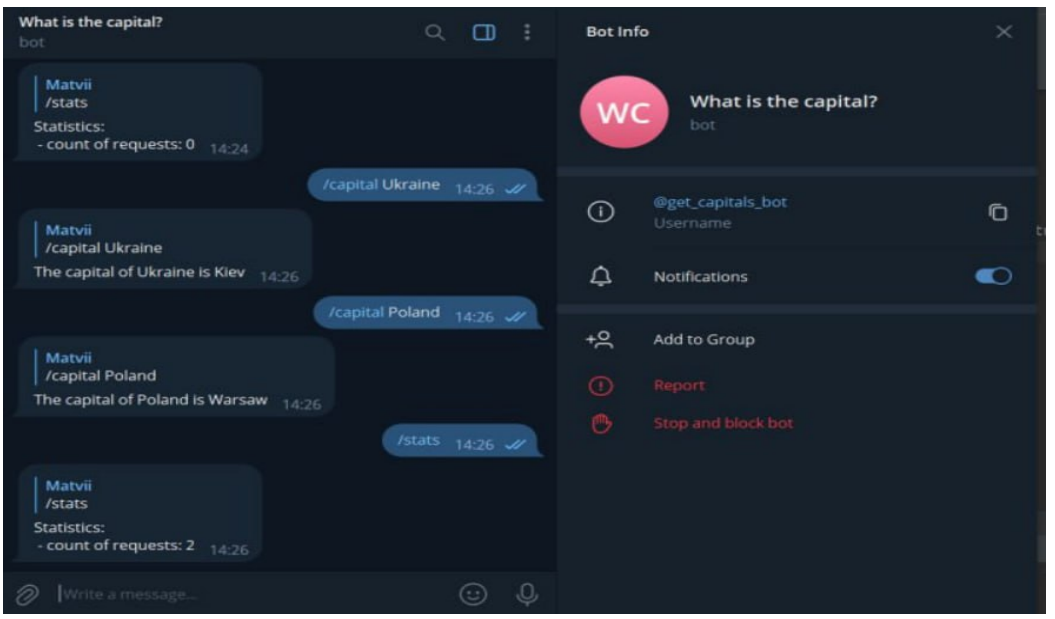
Вважати тестування успішним можна буде після умовного поділу його на декілька етапів. Перший етап має використовуватися для того, щоб провести та виконати функціональне тестування. Другий етап називається «тестуванням продуктивності».

Функціональним називають таке тестування, яке буде займатися безпосередньо тестуванням проекту [23]. Основна увага акцентується на:

- функціях;
- компонентах;
- методах формування фінального вигляду проекту;
- аналізі реакції системи на ті чи інші «подразники».

Тестуванням продуктивності називаються такий процес, коли необхідно визначити рівень спроможності системи працювати на різних програмних забезпеченнях, враховуючи особливості кожної системи окремо. Для цього проводяться оцінки адаптованості системи стосовно різних ситуацій, зокрема, при навантаженні.

Оскільки за своєю структурою функціонал програми є не складним, а простим, достатньо перевірити дві категорії — чи є коректне відображення столиці країни. Процес проведення тестування зображений на (рис. 3.6)



Рисунку 3.6 – Процес тестування Телеграм бота

Спочатку вводимо команду /capital і пишемо назву країни. В нашому випадку це Україна, отже команда виглядатиме /capital Ukraine. Як ми бачимо на (рис 3.6), результат відображається коректно.

Далі вводимо команду /capital Poland і бачимо, що результат знову відображається коректно. При введенні команди /start відбувається відображення статистики отриманих результатів.

При введені столиці чи назви країни з помилками, результат не відображається. Отже, можемо стверджувати, що тестування функціоналу показало успішний результат.

Стосовно тестування продуктивності, то платформа Телеграм має мобільну, десктопну та браузерну версію. В усіх трьох варіантах бот відображає коректні результати, однак, варто зауважити, що проблема може виникнути при встановленні старішої версії програми.

Висновки до третього розділу

У цьому розділі були описані етапи розробки Телеграм-бота.

Особливості логіки та використаних алгоритмів в проекті були також описані. Залежно від поставлених завдань, у бота були реалізовані різні алгоритми, такі як обробка текстових повідомлень, взаємодія з базою даних. Ці алгоритми забезпечували коректну та ефективну роботу бота.

Огляд розробленого інтерфейсу був проведений з метою оцінки зручності та доступності бота для користувачів. Було розглянуто різні скріншоти інтерфейсу, пояснені його особливості та функціонал. Чіткість та зрозумілість інтерфейсу гарантували зручне взаємодію з ботом.

Джерела даних, використані під час розробки бота, були відображені у розділі. Це були вхідні дані від користувачів, дані з інших сервісів, бази даних. Важливою складовою розробки було належне управління цими даними та їхнє використання у процесі роботи бота.

Наприкінці було проведено тестування проекту, щоб переконатися у правильній роботі всіх функціональних блоків та відповідності розробленого продукту вимогам. Тестування допомогло виявити та

виправити можливі помилки, а також перевірити швидкість та стабільність роботи бота.

У результаті виконання розділу було розкрито процес розробки та реалізації Телеграм-бота, описано особливості логіки та використаних алгоритмів, представлено огляд розробленого інтерфейсу, зазначено джерела даних та проведено тестування проекту. Ці кроки забезпечили успішну реалізацію функціоналу бота та гарантували його правильну та надійну роботу.

ВИСНОВКИ

У даній кваліфікаційній роботі було розглянуто розробку та реалізацію Telegram-бота, який надає інформацію про країни та їх характеристики. Робота складається з трьох розділів, кожен з яких містить певну інформацію та аналіз.

У першому розділі було проведено огляд існуючих програмних рішень, а також здійснено аналіз трьох конкретних сервісів, пов'язаних з країнами та їх характеристиками. Процес порівняння цих сервісів дало змогу визначити всі функції та постановити завдання для розробки та реалізація Telegram-бота.

Другий розділ присвячений огляду інструментів, використаних для реалізації проекту. Розглянуто мову програмування Python, інтегроване середовище розробки PyCharm, систему контролю версій Git, а також бібліотеки, фреймворки та API. Ці інструменти були обрані з урахуванням потреб проекту та сприяли ефективній розробці бота.

Третій розділ містить опис етапів розробки та реалізації Telegram-бота. Розглянута логіка та використані алгоритми, огляд розробленого інтерфейсу, джерела даних та процес тестування проекту.

Розробка та реалізація Telegram-бота, який надає інформацію про країни та їх столиці, є актуальним проектом. Використання відповідних інструментів та аналіз існуючих програмних рішень дозволили розробити функціональний та ефективний бот. Робота дозволила поглибити знання у сфері програмування, розробки інтерфейсів та роботи з API.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Advantages and Disadvantages of Python | Python Language Advantages, Disadvantages and Its Applications вебсайт. URL: https://www.aplustopper.com/advantages-and-disadvantages-of-python/ (дата звернення: 21.02.2023)
2. Bot API Library Examples вебсайт. URL: https://core.telegram.org/bots/samples (дата звернення: 02.03.2023)
3. Country.is вебсайт. URL: https://country.is/ (дата звернення: 21.02.2023)
4. countryreports.org вебсайт. URL: https://www.countryreports.org/ (дата звернення: 23.02.2023)
5. PyCharm вебсайт. URL: https://www.trustradius.com/products/pycharm/reviews?qs=pros-and-cons#reviews (дата звернення: 01.03.2023)
6. PyCharm Features вебсайт. URL : https://www.jetbrains.com/pycharm/features/ (дата звернення: 01.03.2023)
7. Python вебсайт. URL: https://www.python.org/ (дата звернення: 28.02.2023)
8. Python Advantages and Disadvantages – Step in the right direction вебсайт. URL: https://techvidvan.com/tutorials/python-advantages-and-disadvantages/ (дата звернення: 27.02.2023)
9. SVN vs Git: Which Version Control System Should You Use? вебсайт. URL: https://www.linode.com/docs/guides/svn-vs-git/ (дата звернення: 02.03.2023)

10. The Pros and Cons of Python Programming вебсайт. URL: https://www.linode.com/docs/guides/pros-and-cons-of-python/ (дата звернення: 27.02.2023)
11. What is Git version control? вебсайт. URL: https://about.gitlab.com/topics/version-control/what-is-git-version-control/ (дата звернення: 02.03.2023)
12. What is PyCharm? Features, Advantages & Disadvantages вебсайт. URL: https://hackr.io/blog/what-is-pycharm (дата звернення: 01.03.2023)
13. What is Telegram and why is it so special? вебсайт. URL: https://www.thepowermba.com/en/blog/what-is-telegram-and-why-is-it-so-special (дата звернення: 15.04.2023)
14. What is Telegram and why should I use it? вебсайт. URL: https://www.androidauthority.com/what-is-telegram-messenger-979357/ (дата звернення: 16.04.2023)
15. What is version control? вебсайт. URL: https://www.atlassian.com/git/tutorials/what-is-version-control (дата звернення: 02.03.2023)
16. World Capitals Quiz вебсайт. URL: https://telegramic.org/bot/world_capitals_quiz_bot/ (дата звернення: 22.02.2023)
17. Алексєєв, В. А., В. С. Терещенко. «Розвиток спіральної моделі життєвого циклу програмних систем.» (2003) (дата звернення: 19.03.2023)
18. Голян, В. В., О. К. Кравченко. «Порівняння моделей життєвих циклів програмного забезпечення з метою виявлення найефективнішого». Системи обробки інформації 2 (157) (2019): 63-70. (дата звернення: 16.03.2023)
19. Етапи життєвого циклу розробки ПЗ вебсайт. URL: https://icstudio.online/post/etapi-zhittyevogo-ciklu-rozrobki-pz (дата звернення: 02.03.2023)

20. КАСКАДНА МОДЕЛЬ ЖИТТЄВОГО ЦИКЛУ: ПЕРЕВАГИ І НЕДОЛІКИ вебсайт. URL: https://government.com.ua/nashi-hroshi/kaskadna-model-zhittevogo-tsiklu-perevagi-i-nedoliki.html (дата звернення: 18.03.2023)
21. Коротун, Т. М. «Моделі і методи тестування програмних систем.» (2007). (дата звернення: 29.04.2023)
22. Путівник мовою програмування Python вебсайт. URL: https://pythonguide.rozh2sch.org.ua/ (дата звернення: 05.03.2023)
23. Функціональне та нефункціональне тестування: що потрібно знати вебсайт. URL: https://codeguida.com/post/2379 (дата звернення: 30.04.2023)

ДОДАТКИ

Додаток А Лістинг програмного коду

```
import countryinfo

from aiogram import Bot, types

from aiogram.dispatcher import Dispatcher

from aiogram.contrib.fsm_storage.memory import MemoryStorage

from aiogram.types import InlineKeyboardMarkup, InlineKeyboardButton

from aiogram.utils import executor


import sqlite3


conn = sqlite3.connect('./users.db', check_same_thread=False)

conn.row_factory = lambda k, l: {c[0]: l[i] for i, c in enumerate(k.description)}

cur = conn.cursor()


cur.execute(

        """

        CREATE TABLE IF NOT EXISTS users_requests(

        id INT,

        username TEXT,

        request TEXT
```

```
        )
        """
)



bot = Bot(token="6154225881:AAEcjZ9G6BRtaEVCjHpeLm6J2ba-js1_coQ")
dp = Dispatcher(bot, storage=MemoryStorage())



@dp.message_handler(commands=['stats'])
async def stats(message: types.Message):
        cur.execute(f'SELECT * FROM users_requests;')
        users = cur.fetchall()

        await message.reply(f'Statistics:\n - count of requests: {len(users)}')



def reg_user(user_id, user_name, request):
        cur.execute(
        'INSERT INTO users_requests (id, username, request) VALUES (?, ?, ?);',
        (user_id, user_name, request,)
        )
        conn.commit()
```

```
def return_country_with_button(user, country):
        reg_user(user.id, user.username, country)

        try:
         country_info = countryinfo.CountryInfo(country)
        capital = country_info.capital()

        answer = f'The capital of {country} is {capital}'
        markup = InlineKeyboardMarkup()
    markup.add(InlineKeyboardButton('More info', callback_data=f'info'))

        except:
        answer = 'Wrong input!'
        markup = None

        return answer, markup


@dp.message_handler(commands=['capital'])
async def get_capital_command(message: types.Message):
        country = message.text.split(' ')[1]
        answer, markup = return_country_with_button(message.from_user, country)
```

```
        await message.reply(text=answer, reply_markup=markup)



@dp.message_handler()
async def get_capital_message(message: types.Message):
        country = message.text
        answer, markup = return_country_with_button(message.from_user, country)

        await message.reply(text=answer, reply_markup=markup)



def pretty_print_json(data):
        result = ""

        for key in data:
        if type(data[key]) is list:
        result += f"{key}: \n"
        for item in data[key]:
        result += f"\t\t\t\t{item}\n"
        elif type(data[key]) is dict:
        result += f"{key}: \n"
        for item in data[key]:
        result += f"\t\t\t\t{item}: {data[key][item]}\n"
        else:
```

```
        result += f"{key}: {data[key]}\n"


        return result
@dp.callback_query_handler(lambda call: call.data == 'info')
async def callback_cur_button(call: types.CallbackQuery):
        country = call.message.text.split(' ')[-3]
        country_info = countryinfo.CountryInfo(country)
        await call.message.reply(pretty_print_json(country_info.info()))
if __name__ == "__main__":
        executor.start_polling(dp, skip_updates=True)
```