МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Кваліфікаційна робота бакалавра

TELEGRAM-БОТ ДЛЯ НАДАННЯ ПЕРШОЇ МЕДИЧНОЇ ДОПОМОГИ

Здобувач освіти гр. ІН-82 Максим ОСТАПЕНКО

Науковий керівник,
кандидат фізико-математичних наук,
асистент кафедри комп'ютерних наук Олександр ВЛАСЕНКО

Завідувач кафедри
доктор технічних наук, професор Анатолій ДОВБИШ

СУМИ 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Затверджую_____
Зав. кафедрою Довбиш А.С.
"_____" _____2022 р.

ЗАВДАННЯ
до кваліфікаційної роботи

здобувача вищої освіти четвертого курсу, групи ІН-82 спеціальності «122 – Комп'ютерні науки» денної форми навчання Остапенка Максима Андрійовича.

Тема: «TELEGRAM-БОТ ДЛЯ НАДАННЯ ПЕРШОЇ МЕДИЧНОЇ ДОПОМОГИ»

Затверджена наказом по СумДУ
№_____ від _____2022 р.

Зміст пояснювальної записки: 1) аналіз проблеми та постановка задачі; 2) вибір метода розв'язання задачі; 3) розробка інформаційного і програмного забезпечення.

Дата видачі завдання «_____»_____ 2022 р.
Керівник роботи _____ Олександр ВЛАСЕНКО
Завдання прийняв до виконання _____ Максим ОСТАПЕНКО

РЕФЕРАТ

Записка: 45 стор., 33 рис., 2 табл., 1 додаток, 5 джерел.

Об’єкт дослідження – чат-боти у месенджері Telegram

Мета роботи – telegram-бот для надання першої домедичної допомоги

Методи дослідження – методології створення telegram-ботів, аналіз схожих рішень

Результати – розроблено telegram-бота з надання першої домедичної допомоги. Створений бот має структуровану інформацію, меню для швидкого доступу до неї, вибудувану логіку переходів між пунктами меню. Розробка проведена на базі мови програмування Python з використанням бібліотеки Aiogram. У ході тестування проблем не виявлено.

TELEGRAM, AIOGRAM, МЕДИЦИНА, PYTHON, API

ЗМІСТ

ВСТУП 5
1 ЛІТЕРАТУРНИЙ ОГЛЯД 6
1.1 Огляд аналогічних інформаційних систем 6
1.1.1 Telegram-бот для ПДД при передозуванні 6
1.1.2 Мобільний застосунок для навчання першій домедичній допомозі 9
1.2 Постановка задачі 13
2 МЕТОДИКА ВИРІШЕННЯ ПОСТАВЛЕНИХ ЗАДАЧ 14
2.1 Візуальне проектування боту 14
2.2 Реєстрація бота в системі 16
3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ 20
3.1 Вибір середовища розробки 20
3.2 Тестування Telegram боту 21
ВИСНОВКИ 31
СПИСОК ЛІТЕРАТУРИ 32
ДОДАТОК А 33

ВСТУП

Актуальність теми. Сучасний світ має шалений темп розвитку. Починаючи з весни 2020 року все кардинально змінилося через пандемію COVID-19. Соціальне дистанціювання, масковий режим, вакцинація, карантинні зони – важко було уявити, що таке можливо, проте, не дивлячись на такі виклики долі, світ зміг адаптуватися та продовжити хоч і не таке життя, як раніше, але все ж життя. Після пандемії в Україні трапився новий виклик – повномасштабне вторгнення з боку Російської Федерації.

Проте життя продовжується і в умовах війни для цивільного населення необхідно проводити інструктаж про способи надання першої домедичної допомоги. І дійсно, не дивлячись на велику кількість можливостей з отримання таких знань, попит значно йде на поступки пропозиції. Ймовірно це відбувається через такі фактори:

1. Для отримання таких знань потрібно виділити достатньо велику кількість часу. Це не завжди зручно для працюючого населення, студентів, школярів.
2. За період тренінгу люди отримують чимало інформації і є страх того, що можливо щось забути у стресових ситуаціях.

Завдяки сучасним технологіям, можливо вирішити як мінімум другий пункт. Це може бути застосунок для смартфону, веб-сайт, відеоматеріал або telegram-бот.

Метою роботи є розробка Telegram-боту, який може бути доступним широкій аудиторії, оскільки спостерігається стрімке збільшення користувачів і цільова аудиторія значно розширюється. По-друге, кросплатформеність. Додаток з інформацією про надання першої медичної допомоги може бути випущено не на всі операційні системи, веб-застосунок може некоректно працювати на різних ОС, але якщо смартфон підтримує Telegram – користувач без проблем зможе скористатися нашим telegram-ботом.

1 ЛІТЕРАТУРНИЙ ОГЛЯД

Першим етапом у створенні ефективного Telegram-бота буде аналіз відомих рішень. Даний етап важливий тим, що завдяки ньому ми можемо оглянути схожі за призначенням рішення, виявити в них переваги та недоліки і завдяки цьому спроектувати життєздатний додаток, яким зможуть користуватися люди.

1.1 Огляд аналогічних інформаційних систем

1.1.1 Telegram-бот для ПДД при передозуванні

Першим розглянемо Telegram-бота для першої домедичної допомоги при передозуванні психоактивними речовинами [1], що створений на базі Manybot – конструктора ботів, що працює за моделлю freemium (базовий функціонал безкоштовний, якщо треба більше – потрібно доплачувати).

Нас зустрічає назва боту та його опис, в якому вказано для чого призначений даний бот (рис.1.1). Як і зазвичай, для початку роботи з ботом нам потрібно натиснути на кнопку “Почати”.

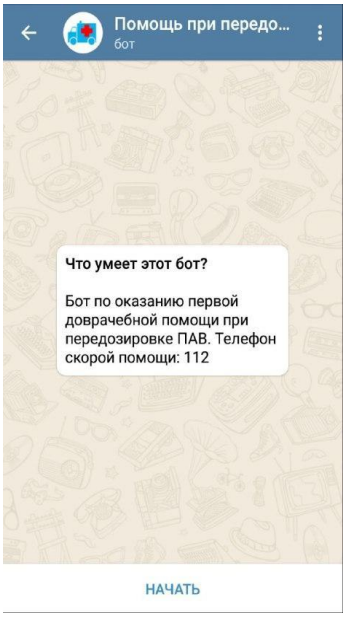

Рисунок 1.1 – Початок роботи з ботом

Відразу варто виділити перевагу бота – зручне меню. Воно містить усі потрібні функції, з якими зручно буде взаємодіяти навіть у стресових ситуаціях (рис.1.2).

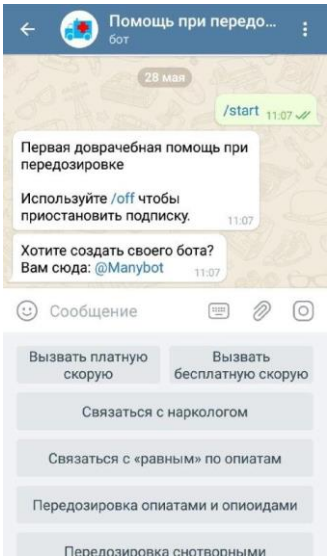

Рисунок 1.1 – Початкове меню Telegram-бота

Обираємо один з пунктів меню – “Передозування снодійним”. Після натискання на кнопку отримуємо всю потрібну інформацію, яка знадобиться для надання ПДД при передозуванні снодійним (рис.1.3).

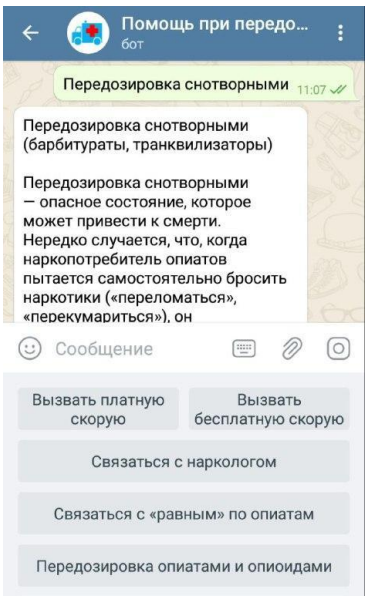

Рисунок 1.3 – Повідомлення після обрання пункту “Передозування снодійним”

Крім цього, повідомлення має додатковий контент (рис.1.4), яке допоможе користувачу при наданні першої домедичної допомоги.

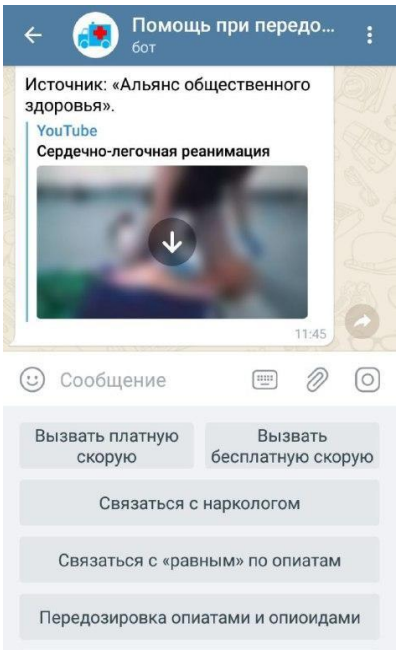

Рисунок 1.4 – Додатковий контент у повідомленні про ПДД при передозуванні снодійним

Також бот має інформацію про розробників боту. Це також корисно, адже така інформація формує довіру до продукту. Якщо його розробляла перевірена часом та досвідом організація – відповідно, таким продуктом можна користуватися як шпаргалкою для надання ПДД.

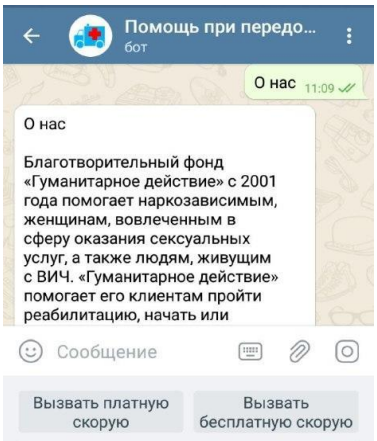

Рисунок 1.5 – Інформація про розробників боту

1.1.2 Мобільний застосунок для навчання першій домедичній допомозі

Перейдемо до розгляду мобільного застосунку [2]. Перше та важливе, що знаходимо у додатку – це розділ “Підготовка”, у якому знаходяться інструкції щодо надзвичайних ситуацій: землетрус, пандемія, ураган, хімічна аварія тощо.

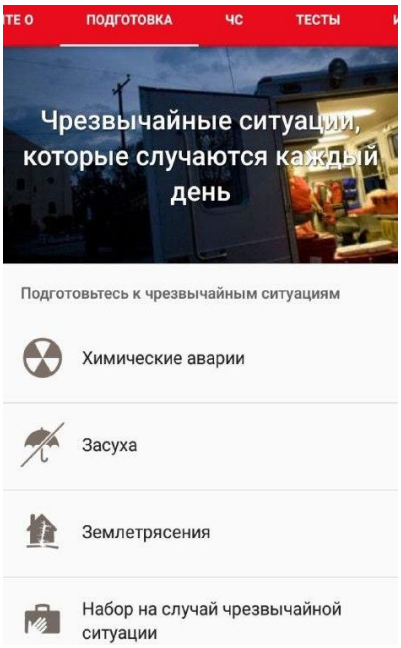

Рисунок 1.6 – Розділ підготовки до надзвичайних ситуацій

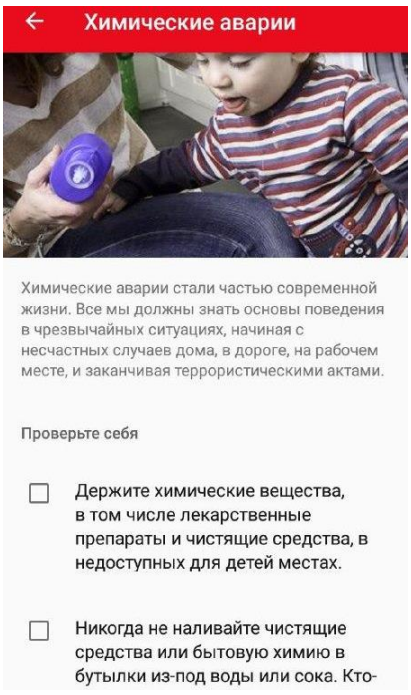

Рисунок 1.7 – Підготовка до ПДД під час хімічної аварії

Також зручним є розділ тестування, де користувач після отримання ним знань може перевірити себе на те, як він їх засвоїв хоча б в теорії (рис.1.8).

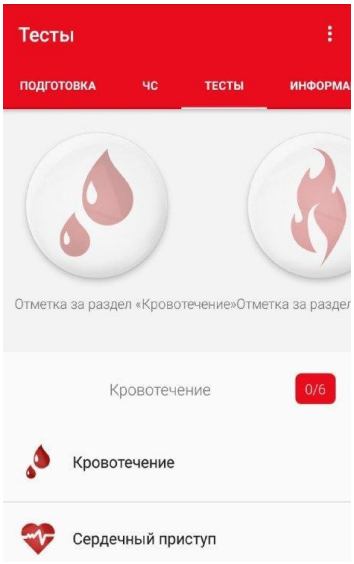

Рисунок 1.8 – Розділ «Тестування»

Якщо користувач ще не готовий до тестувань, він з легкістю може закріпити матеріал у розділі “НС” (Надзвичайні ситуації”) (рис. 1.9).

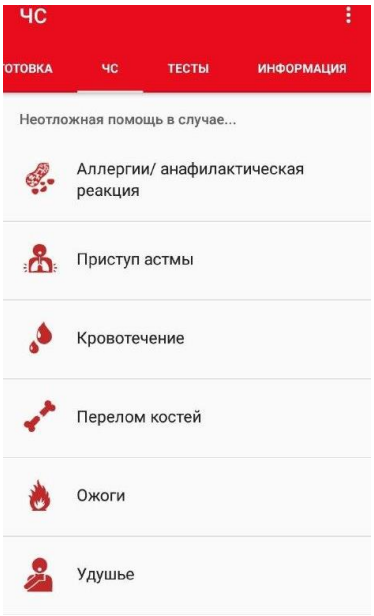

Рис.1.9 Розділ “НС”

У ньому можна знайти всю потрібну інформацію для невідкладної домедичної допомоги при різних можливих ситуаціях. Наприклад, відкриємо

розділ “Алергії” та бачимо першопочаткову інформацію про те, як саме можна допомогти людині з ними (рис.1.10).

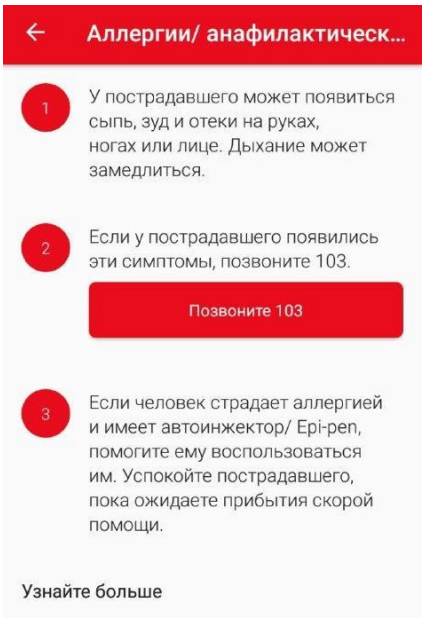

Рисунок 1.10 – Розділ “Алергії”

Варто зазначити, що інформація в даному додатку добре структурована: розміщений весь основний текст; потім, якщо залишається щось незрозумілим, існує розділ запитань та відповідей щодо потрібної теми (рис.1.11).

Рисунок 1.11 – Запитання та відповіді до теми “Алергія”

Додаток зручний та зрозумілий для будь кого, що досягається завдяки сучасному дизайну та правильному структуруванню інформації.

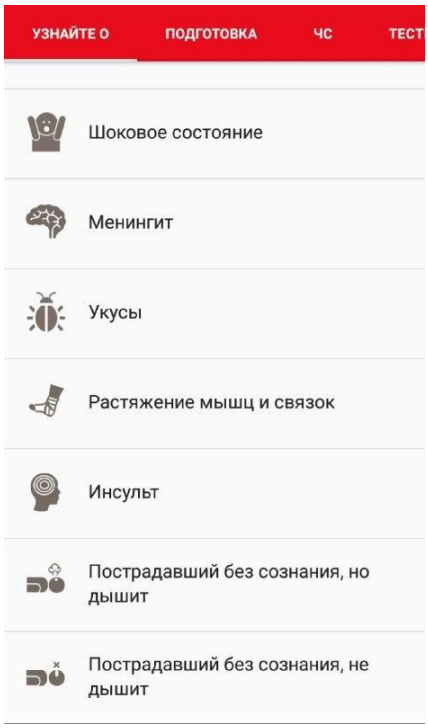

Рисунок 1.12 – Головна сторінка застосунку

Перевага даного додатку над Telegram-ботом в тому, що він доступний навіть офлайн, тобто при втраті мобільного інтернет-зв’язку все одно є можливість отримати всю потрібну інформацію.

Проте, основними недоліками є:

1. Пам’ять. Додаток все ж займає місце на телефоні, що може ускладнити користування іншими додатками або їх встановлення у майбутньому.

2. Час. До надзвичайної ситуації людина могла не підготуватися та не встановити додаток, а при його встановленні вона може втратити цінний час, від якого залежить порятунок людини.

3. Кросплатформеність. Не всі пристрої можуть підтримувати даний додаток, аналоги можуть бути застарілими або може їх не бути взагалі.

Отже, нами було розглянуто додаток та Telegram-бот і тому можемо скласти порівняльну таблицю обох рішень за такими параметрами: початок роботи, поведінка під час НС, подання інформації, доступ до інформації, зв’язок з розробником

Таблиця 1.1 – Порівняння параметрів інформаційних систем

Параметри	Додаток	Бот
Початок роботи	Вимагається завантаження та встановлення	Якщо є месенджер, тоді шукаємо бота в ньому
Поведінка під час НС	Є можливість завчасно отримати інформацію про поведінку під час будь-яких НС	Є можливість завчасного завантаження інформації
Подання інформації	Інформація структурована зручно, є можливість переходу до кожного розділу окремо	Інформація та доступ до неї поділено за допомогою кнопок в чаті з Telegram-ботом
Доступ до інформації	Є можливість отримання інформації офлайн	Можливість відсутня без завчасного завантаження усієї інформації
Зв’язок із розробником	-	Є окремий пункт меню з інформацією про розробників та контактними даними

1.2 Постановка задачі

Після аналізу схожих рішень, визначено мету роботи - підготовка до розробки та безпосередньо розробка Telegram-боту зі структурованою інформацією для швидкого доступу користувача. Для вирішення цієї мети необхідно завершити наступні завдання:

1) Проектування логіки у Telegram-боті;
2) Реєстрація Telegram-бота в системі;
3) Визначення необхідного набору програмних засобів;
4) Проведення тестування сторонніми користувачами;

2 МЕТОДИКА ВИРІШЕННЯ ПОСТАВЛЕНИХ ЗАДАЧ

2.1 Візуальне проектування боту

Розглянувши аналогічні рішення та визначивши мету, кроки для її досягнення, потрібно провести візуальне проектування telegram-боту. Для цього ми використаємо MindMap - діаграму зв'язків, що допомагає у структуруванні ідей за допомогою графічного представлення. Дана мапа дозволить поступово спроектувати те, як саме працюватиме наш Telegram-бот.

Фундаментом нашого MindMap буде набір команд, за допомогою яких користувач взаємодіятиме з ботом (рис.2.1).

Рисунок 2.1 – Фундамент майбутнього продукту

На рис.2.2 зображено те, що відбуватиметься під час виконання команди /start, а саме: користувач отримає вітальне повідомлення зі списком доступних команд.

Рисунок 2.2 – Деталізація команди "/start"

На рисунку 2.3 представлено деталізацію пунктів меню, яке викликається під час команди /danger.

Рисунок 2.3 – Деталізація меню загроз

Крім цього, після натискання на кнопку «Загроз не виявлено», користувач отримає повідомлення із командою /hurt, яка надасть доступ до списку можливих пошкоджень. Для даної команди було розроблено меню можливих пошкоджень. Деталізація цього меню зображена на рисунку 2.4.

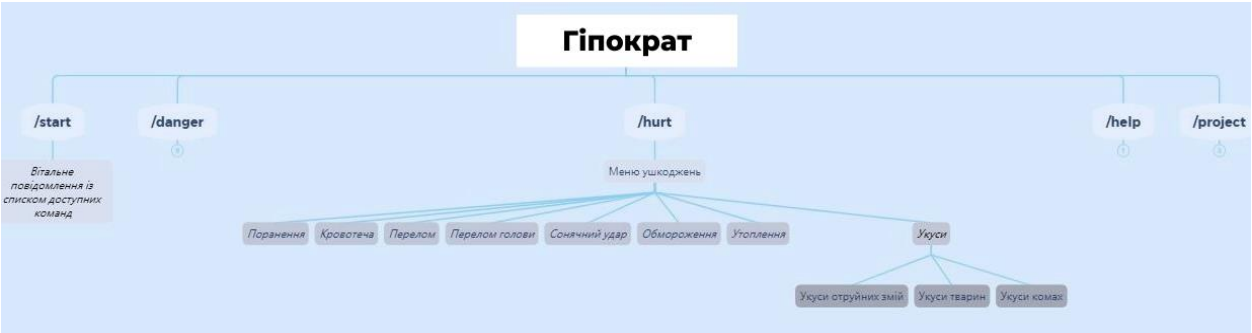

Рисунок 2.4 – Деталізація меню ушкоджень

Таким чином було проаналізовано та систематизовано інформацію, необхідну для надання першої домедичної допомоги. На рисунку 2.5 зображено деталізацію команди /help, яка викликає повідомлення зі списком команд та деталізацію команди /project, яка надає посилання на чат із розробником та на джерело інформації про надання ПМД.

Рисунок 2.5 – Деталізоване меню проекту

2.2 Реєстрація бота в системі

На першому етапі розробки Telegram-бота, потрібно його зареєструвати безпосередньо у самому месенджері для того, щоб отримати унікальний токен. Даний токен забезпечить взаємодію між ботом та Telegram API за допомогою HTTP-протоколу. Зареєструвати Telegram-бот можна за допомогою BotFather – реєстратор ботів у системі, який має зручний інтерфейс та багато потрібних функцій. На рисунку 2.6 представлено початкову сторінку даного реєстратора. [3]

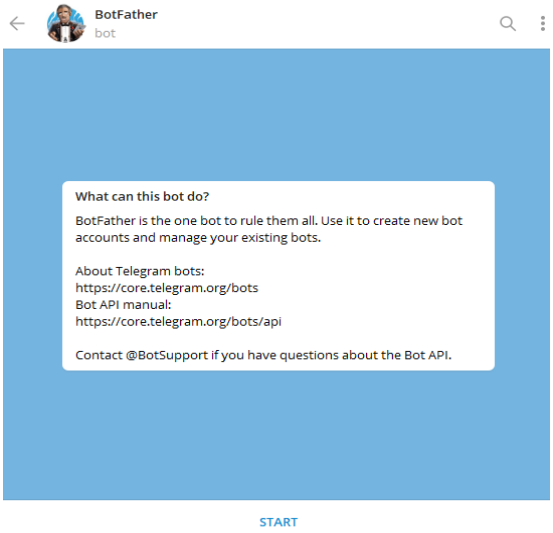

Рисунок 2.6 – Початок роботи з реєстратором ботів

Починаємо роботу з ним, натиснувши на кнопку “START” та вводимо команду /newbot, яка почне реєстрацію telegram-бота (рис.2.7).

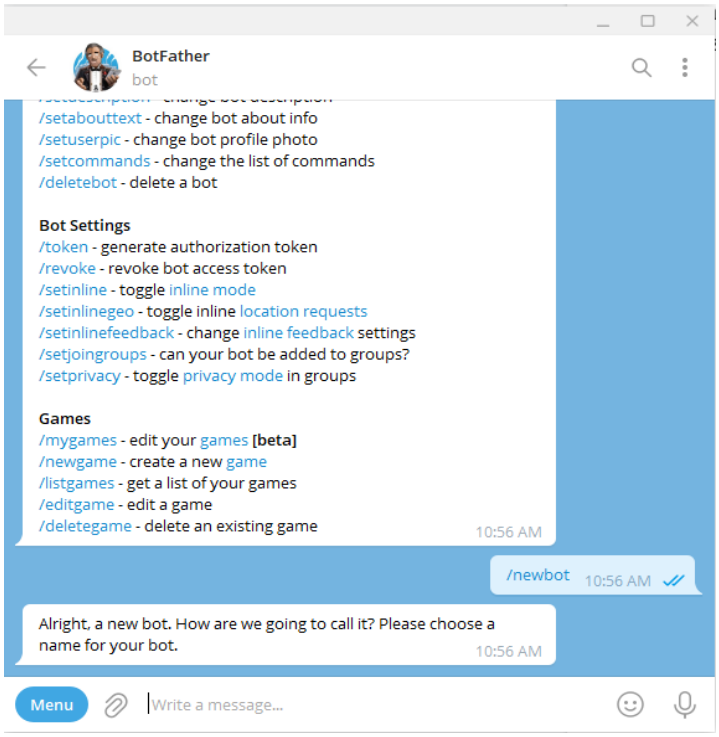

Рисунок 2.7 – Перелік команд бота-реєстратора та початок реєстрації

Необхідно ввести ім’я майбутнього бота, тому вводимо його і потім отримуємо прохання надати нікнейм, по якому користувачі зможуть його знаходити. На рисунку 2.8 представлено дані процеси. [6]

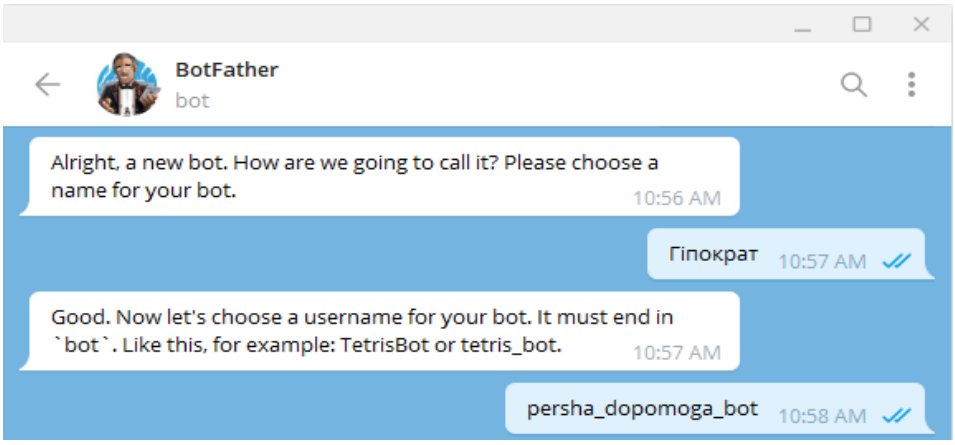

Рисунок 2.8 – Призначення боту імені та нікнейму

Після надання потрібних даних, надходить унікальний токен (прихований для забезпечення безпеки), завдяки якому бот взаємодіятиме з Telegram API (рис.2.9).

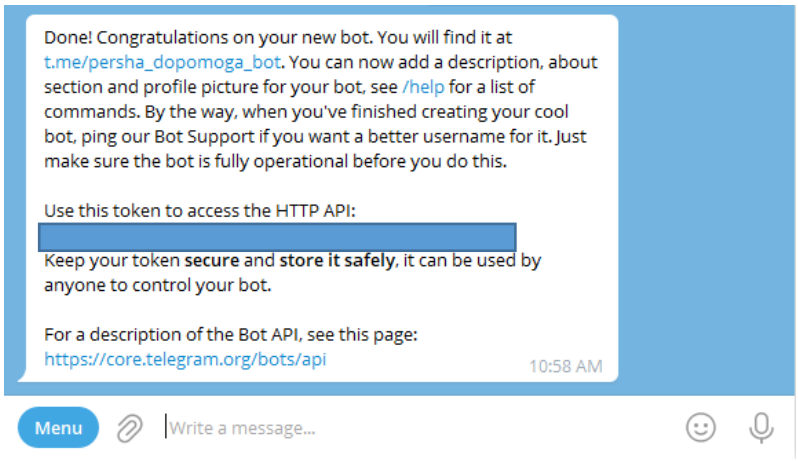

Рисунок 2.9 – Повідомлення з токеном для взаємодії бота і Telegram API

Також є можливість додати опис, зображення та список команд бота, але за необхідності даний крок можна виконати пізніше.

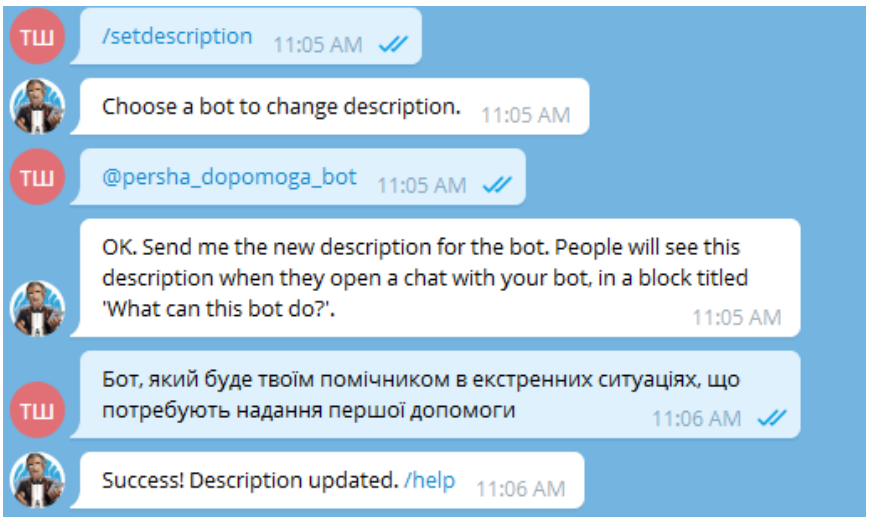

Рисунок 2.10 – Додавання опису бота

Рисунок 2.11 – Встановлення аватарки бота

Після проектування бота можна переходити до його реалізації.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ

3.1 Вибір середовища розробки

Після проєктування бота переходимо безпосередньо до його розробки. Проте, перед цим необхідно обрати мову програмування. Для цього порівняємо 4 основні мови, які широко використовуються у веб-розробці – Javascript, C#, Python та PHP.

- Javascript – раніше широко використовувалася лише для створення front-end складової веб-застосунків. На теперішній стан вона дозволяє реалізувати власний штучний інтелект, серверну частину веб-застосунку або повноцінний desktop додаток. Також варто зазначити, що Javascript та безпосередньо фреймворк Node.js підходять і для розробки Telegram-ботів завдяки великій кількості бібліотек та інструкцій з розробки.
- C# має швидший та стрімкий розвиток, більшу читабельність, наявність sugar syntax, а також вищу швидкодію і меншу ресурсозатратність, навіть у порівнянні з Java. C# має не дуже велику кількість бібліотек для досягнення нашої мети, проте всі вони перевірені часом та досі розвиваються.
- Python – це мова програмування, яка популярна в усіх сверах, наприклад, роботизація, аналіз даних, бізнес-аналітика, веб-розробка, наукова діяльність тощо. Якщо в C# лише присутні деякі sugar syntax елементи, то Python сам по собі “цукровий” та до його вивчення прибігають навіть у школах, на заміну Pascal або Delphi. Для досягнення поставленої задачі вона ідеально підходить, адже має такі популярні бібліотеки, як Telebot, Aiogram, Pyrogram, PyTelegramBotAPI [5]. Всі вони містять в собі абсолютно різні переваги та недоліки і підійде для будь-яких задач, які повинен вирішити Telegram-бот.
- PHP – дещо застаріла мова програмування, але яка досі розвивається з року в рік. Вона, як і Python, підходить для створення MVP “на швидку руку”,

але на відміну від нього має складнощі у масштабуванні. Це пов’язано з тим, що більша частина цільової аудиторії PHP нехтує правилами чистого коду, а його чистота впливає на подальшу підтримку та масштабування. Проте, поставлену задачу також може допомогти вирішити, для цього є всі можливості та бібліотеки.

Таблиця 3.1 – Порівняння мов програмування для створення Telegram-ботів

Параметри	JavaScript	C#	Python	PHP
Шаблони	+	+	+	+
Бібліотеки	+/-	+	+	+/-
Розповсюдженість мови	+	+	+	+
Читабельність	+	+/-	+	+/-
Пристосування мови для нашої задачі	+/-	-	+	+

Даний telegram-бот має перспективу до масштабування, причому масштабування може відбуватися у короткі терміни. Для забезпечення коротких термінів, потрібен максимально зрозумілий код. Мова програмування повинна забезпечити ефективність та швидкодію, адже мова йде про першу домедичну допомогу, тому бот повинен мати нульову терпимість до можливих перебоїв. Проаналізувавши мови програмування, було прийнято рішення про обрання Python та фреймворку Aiogram [8].

3.2 Тестування Telegram боту

Після завершення розробки Telegram-бота з надання першої домедичної допомоги, вихідний код якого знаходиться у додатку А, переходимо безпосередньо до його тестування.

На рисунку 3.1 зображено стартову сторінку бота з його назвою, аватаркою та описом.

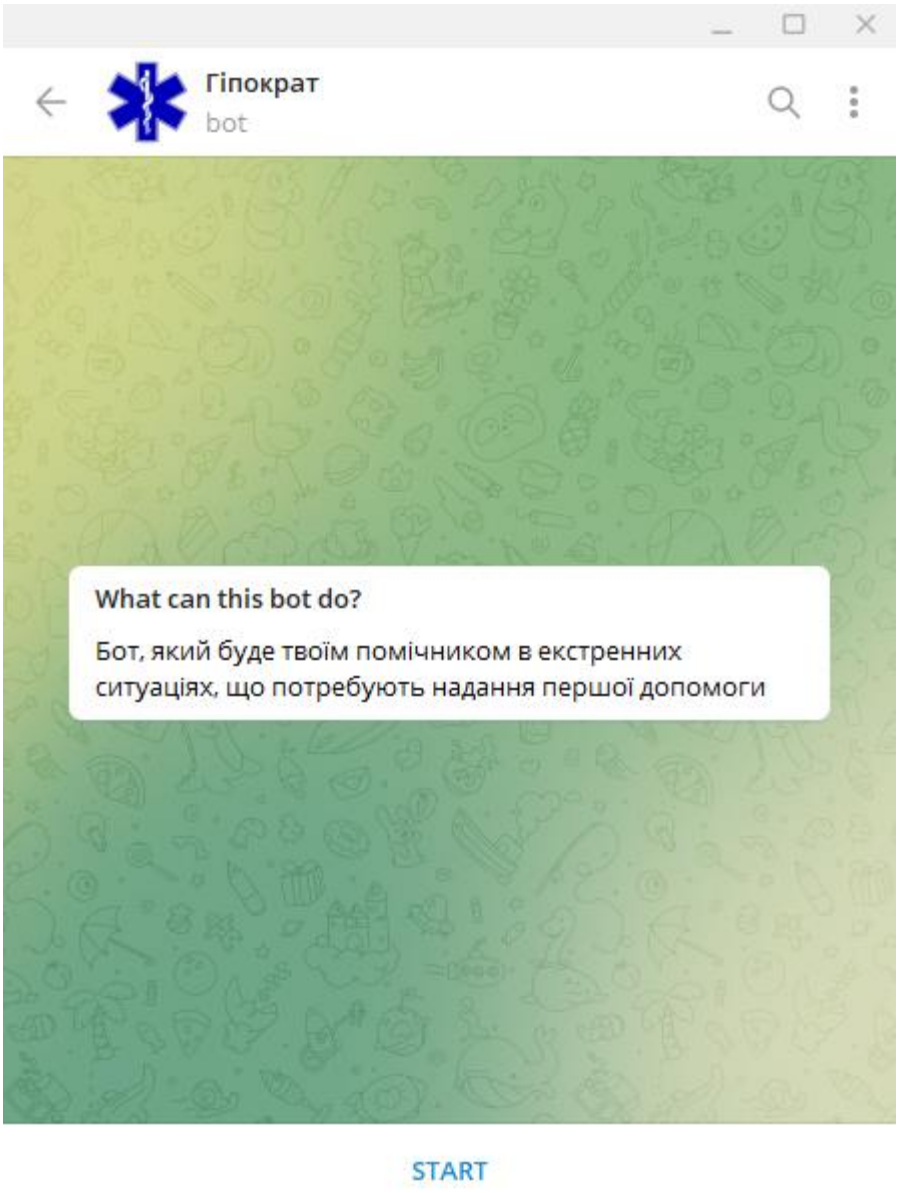

Рисунок 3.1 – Стартова сторінка боту з назвою та описом

Натискаємо на кнопку “Start” і нас зустрічає вітальне повідомлення та клавіатура з командами (рис.3.2).

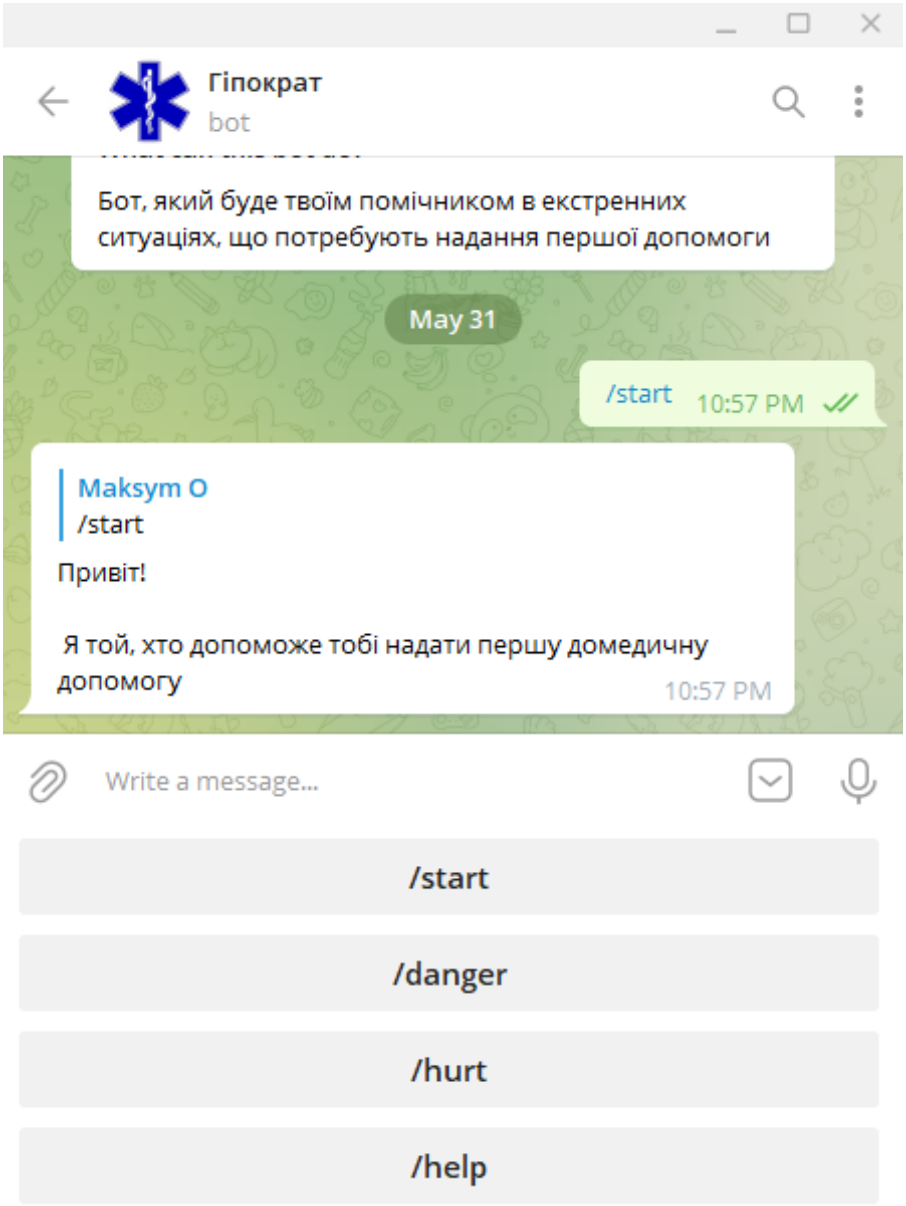

Рисунок – 3.2 Вітальне повідомлення та клавіатура з командами

Протестуємо логіку роботи команди /danger. На рисунку 3.3 зображено клавіатуру з переліком ситуацій, з яких ми обираємо потрібну.

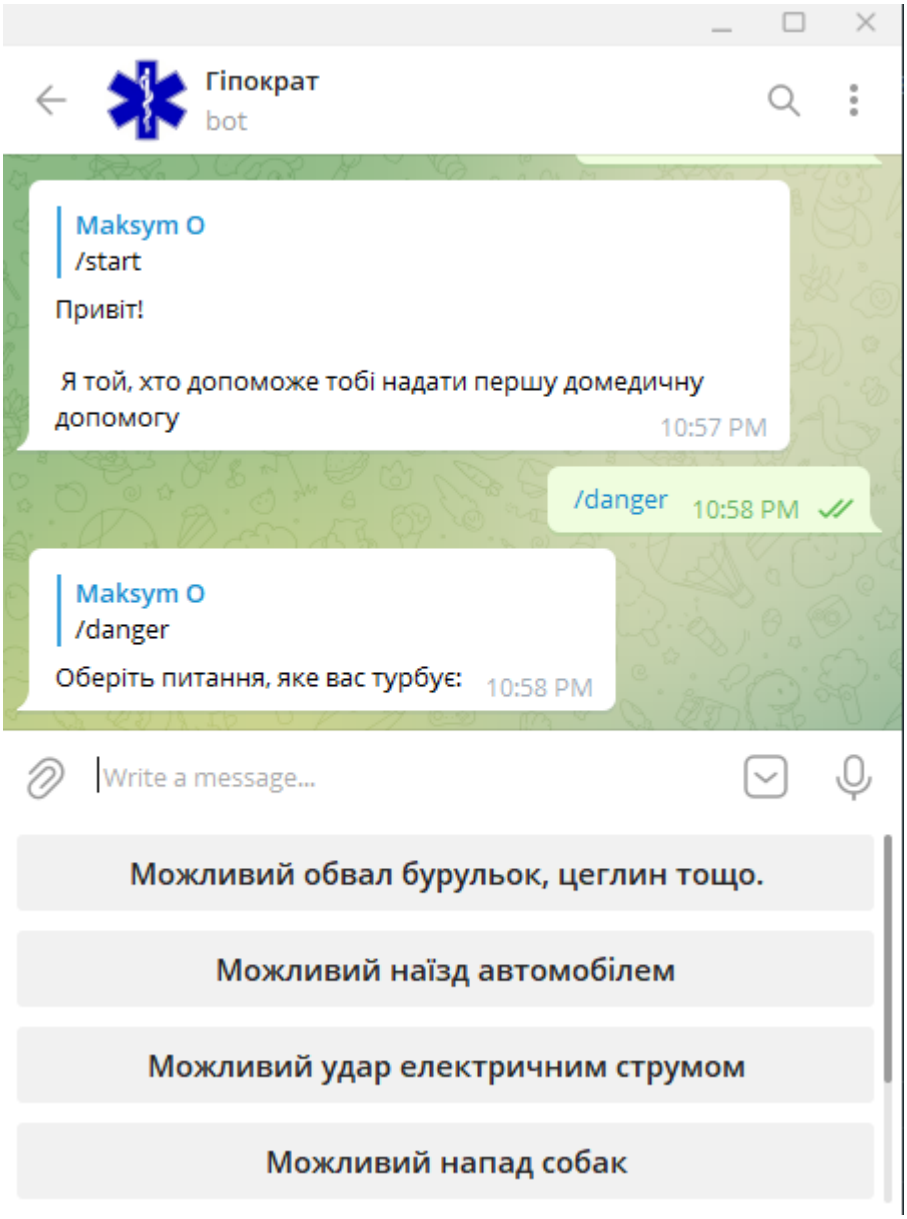

Рисунок 3.3 – Результат виконання команди /danger

Оберемо, наприклад, “Можливий удар електричним струмом”. Наразі ми отримаємо повідомлення-заглушку (рис.3.4), адже нам треба продемонструвати лише логіку роботи бота. Пізніше у майбутньому планується додати корисний та змістовний текст.

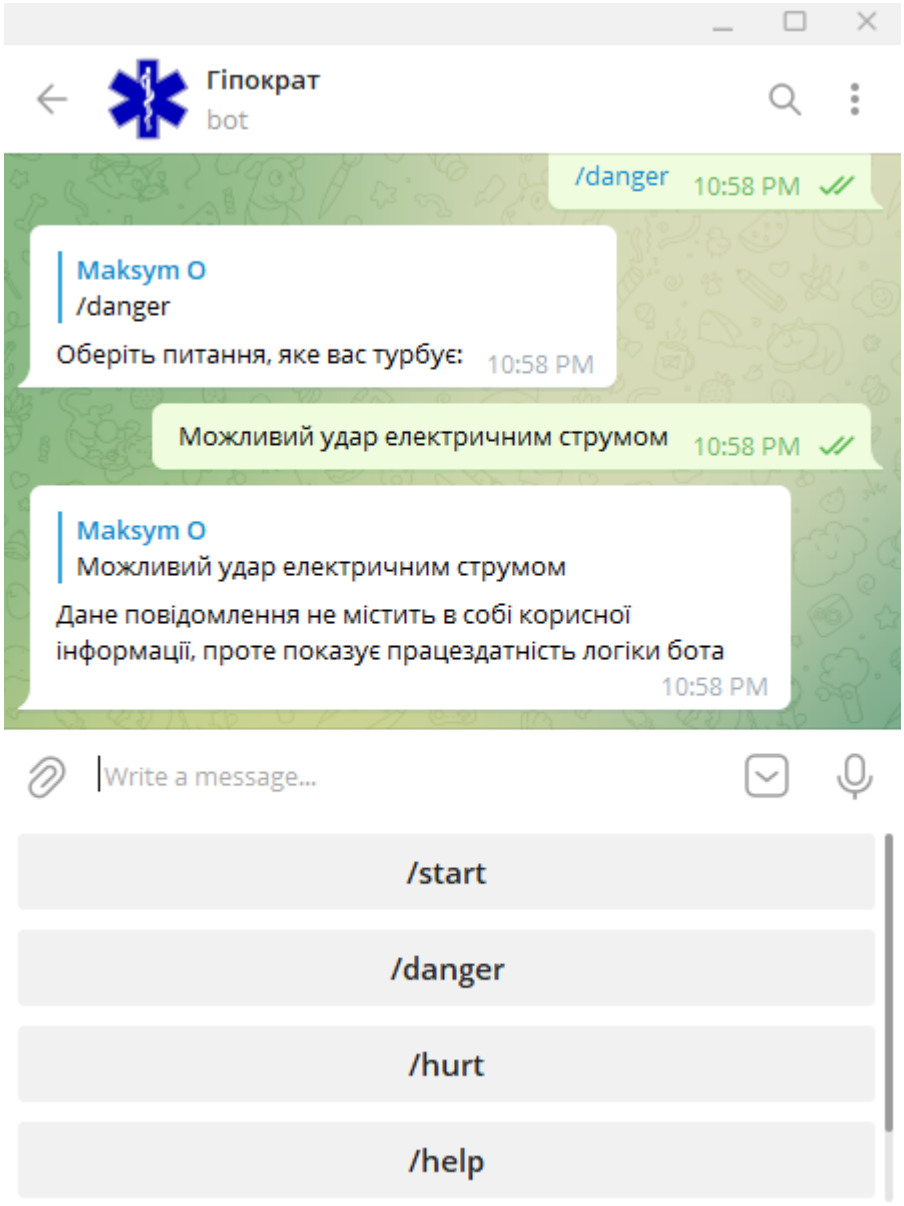

Рисунок 3.4 – Відповідь бота на кнопку “Можливий удар електричним струмом”

Знову оберемо команду /danger та підпункт “Загроз не виявлено”, щоб перевірити чи отримаємо ми повідомлення про можливість виконати команду /hurt. На рис.3.5 зображено, що команда виконана успішно та нам пропонують написати команду /hurt, щоб обрати вид ушкодження із меню ушкоджень.

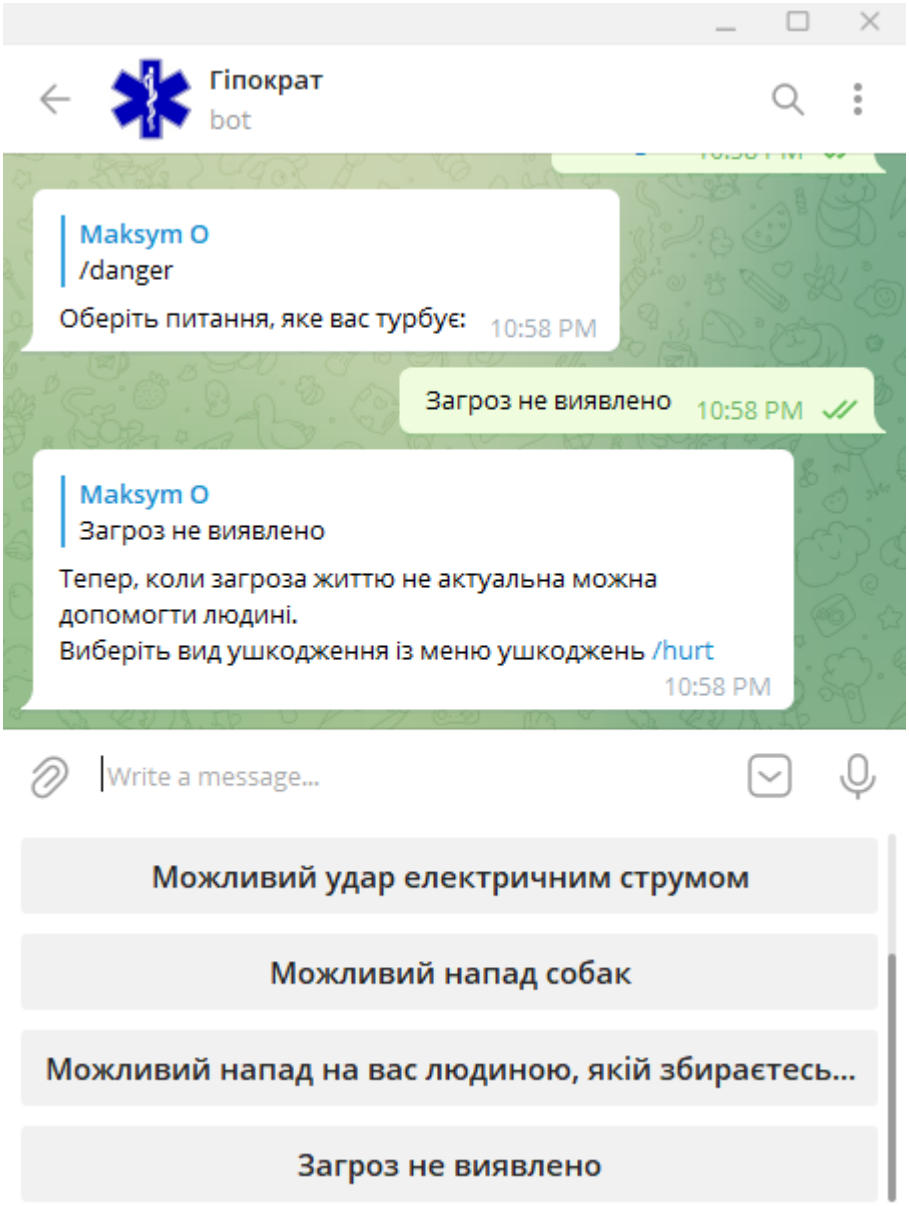

Рисунок 3.5 – Підпункт “Загроз не виявлено”

Надсилаємо боту команду /hurt та отримуємо типи поранень, після чого обираємо тип поранення “Укуси”, адже нам потрібно перевірити чи отримаємо підпункти для цього типу поранення. На рис.3.6 та рис.3.7 бачимо, що дана логіка працює.

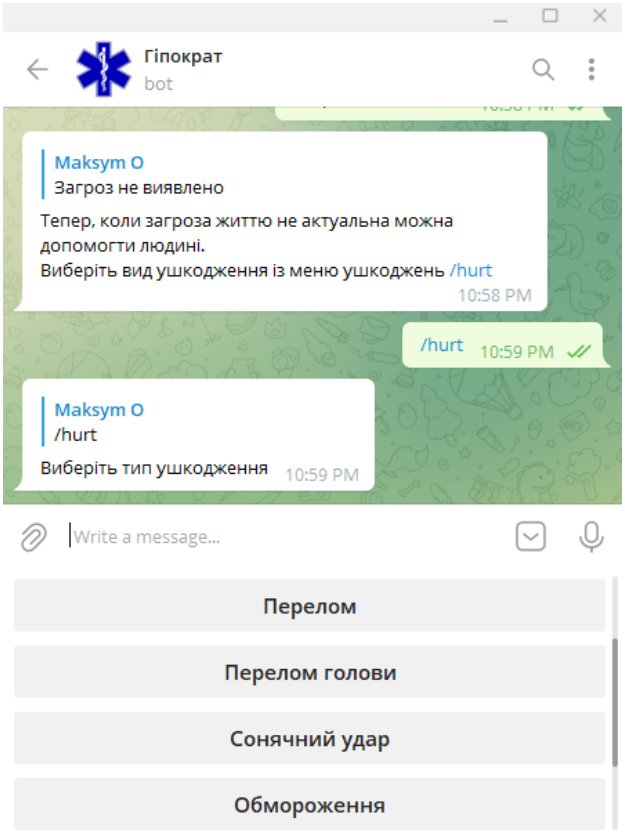

Рисунок 3.6 – Меню вибору типу ушкодження

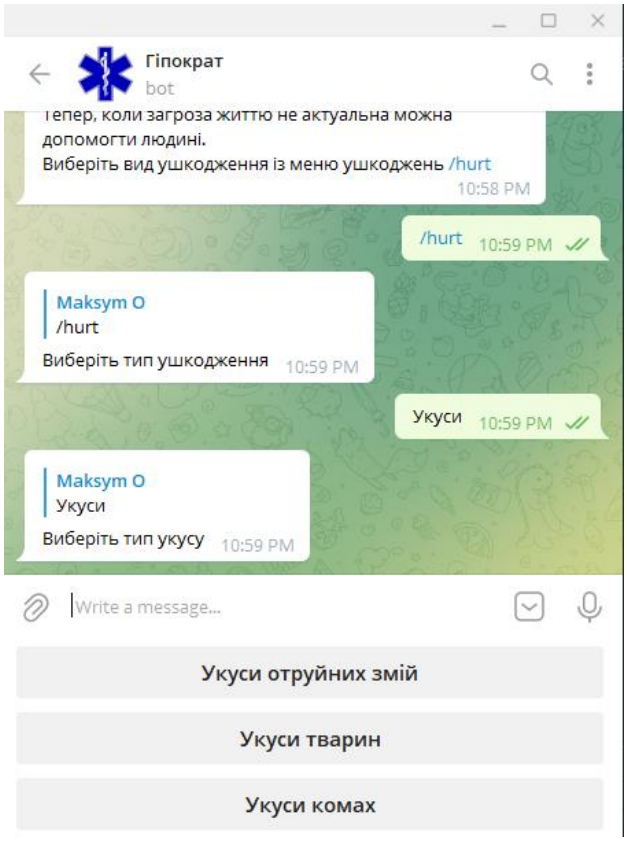

Рисунок 3.7 – Меню вибору типу укусу

Обираємо, наприклад, тип “Укуси комах” та на рис.3.8 бачимо, що команда виконана вдало – отримуємо повідомлення заглушку, яке знову таки може бути заміненим іншим разом.

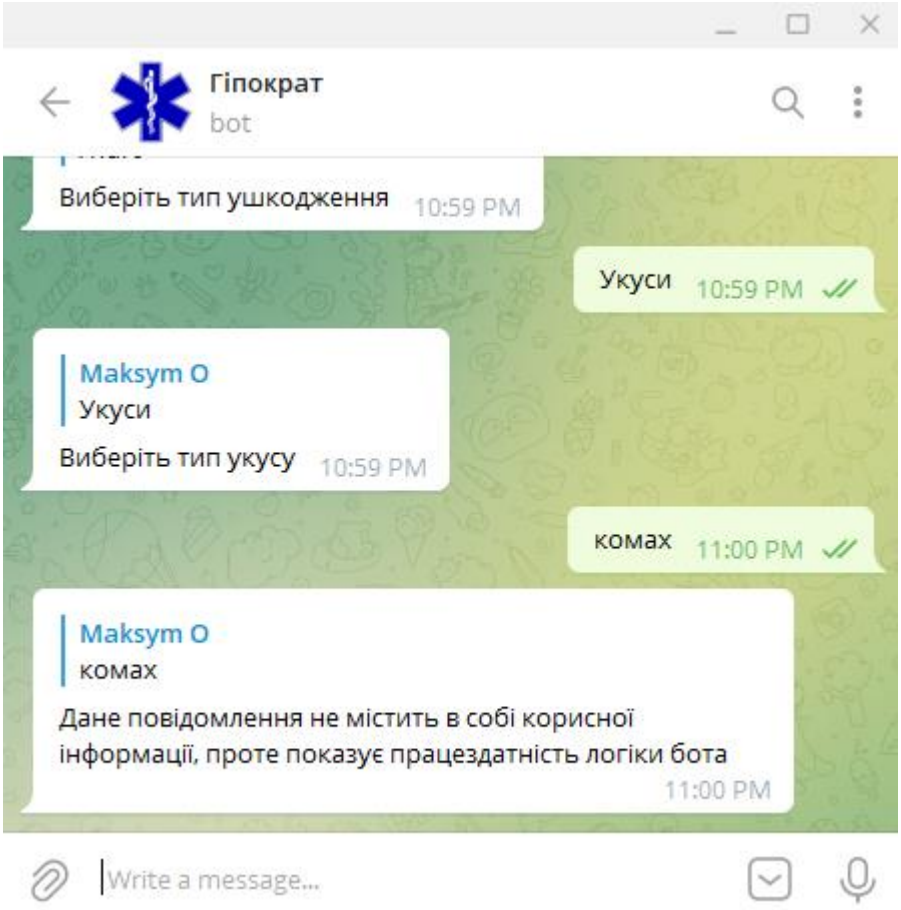

Рисунок 3.8 – Відповідь на підпункт “Укуси комах”

Перевіримо, чи вірно працюють команди /help (отримання інформації про наявні команди) та /project (отримання інформації про розробника). На рис.3.9 та рис.3.10 бачимо, що команди відпрацьовують вдало.

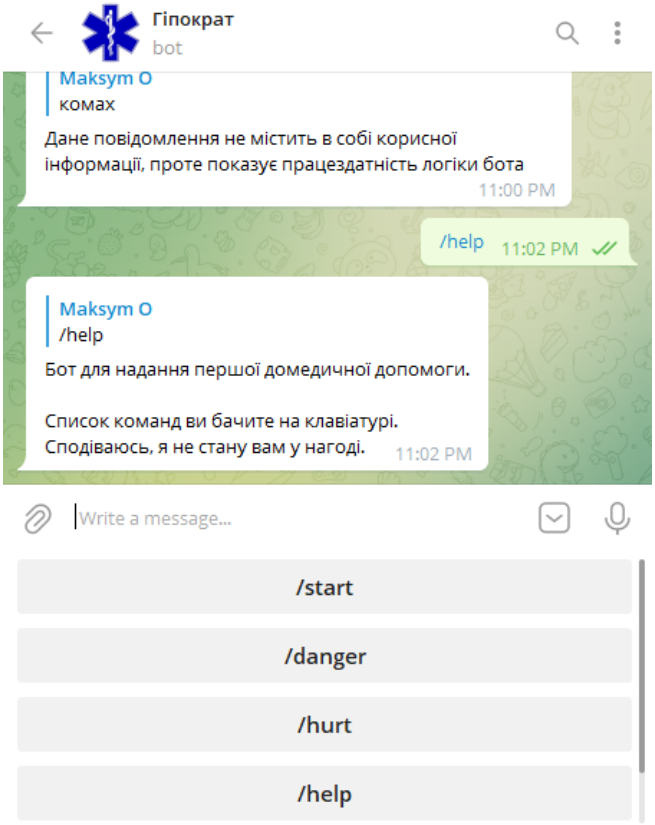

Рисунок 3.9 – Відповідь на команду /help

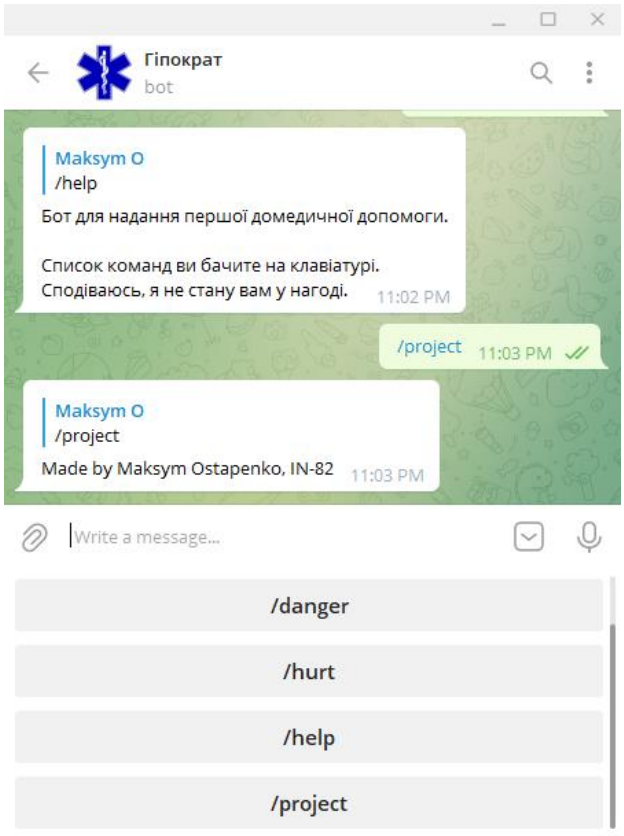

Рисунок – 3.10 Відповідь на команду /project

Таким чином ми перевірили усі можливі команди бота та можемо зробити висновок, що бот працює справно та він готовий до масштабування і наповнення більш корисним контентом, щоб ним могли користуватися звичайні люди.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було виконано всі етапи для досягнення поставлених цілей, а саме:

- аналіз програмних засобів для розробки Telegram-боту;
- аналіз схожих рішень;
- проектування майбутнього Telegram-бота;
- розробка та тестування працездатності Telegram-бота.

Результат виконання роботи – Telegram-бот, який допоможе надати першу домедичну допомогу.

Розробка має ряд переваг:

- зручний інтерфейс
- логічну ієрархію команд
- структуровану інформацію
- інформацію про розробника
- швидкодію

СПИСОК ЛІТЕРАТУРИ

1. Telegram-bot при передозуванні ПАР [Електронний ресурс] – Режим доступу: https://shorturl.at/abntM
2. Додаток для надання першої медичної допомоги [Електронний ресурс] – Режим доступу: https://shorturl.at/pxHX7
3. Реєстрація бота в Telegram [Електронний ресурс] – Режим доступу: https://shorturl.at/jxST0
4. Створення бота в Telegram. Бібліотека Aiogram [Електронний ресурс] – Режим доступу : https://shorturl.at/quwA9
5. Документація Telegram API [Електронний ресурс] – Режим доступу: https://shorturl.at/txKQ5
6. Як створити Telegram-бота. Загальна інструкція [Електронний ресурс] – Режим доступу: https://shorturl.at/rtDKU
7. Telegram Bot API [Електронний ресурс] - Режим доступу: https://shorturl.at/hituW
8. How to use Redis with Python [Електронний ресурс] – Режим доступу: https://shorturl.at/tuxZ6
9. Перша долікарська допомога. Фармацевтична енциклопедія [Електронний ресурс] – Режим доступу: https://shorturl.at/govIY
10. How to deploy a Telegram bot [Електронний ресурс] – Режим доступу: https://shorturl.at/akFHT

ДОДАТОК А

Програмна реалізація

```
from aiogram import Bot, types
from aiogram.dispatcher import Dispatcher
from aiogram.types import KeyboardButton,
ReplyKeyboardMarkup
from aiogram.utils import executor
from emoji import emojize


bot = Bot(token="")
dp = Dispatcher(bot)


start_btn = KeyboardButton('/start')
danger_btn = KeyboardButton('/danger')
hurt_btn = KeyboardButton('/hurt')
help_btn = KeyboardButton('/help')
project_btn = KeyboardButton('/project')
```

```
    button6 = KeyboardButton('Можливий обвал бурульок, цеглин
тощо.')
    button7 = KeyboardButton('Можливий наїзд автомобілем')
    button8 = KeyboardButton('Можливий удар електричним
струмом')
    button9 = KeyboardButton('Можливий напад собак')
    button10 = KeyboardButton('Можливий напад на вас людиною,
якій збираєтесь допомогти')
    button11 = KeyboardButton('Загроз не виявлено')
    button12 = KeyboardButton('Поранення')
    button13 = KeyboardButton('Кровотеча')
    button14 = KeyboardButton('Перелом')
    button15 = KeyboardButton('Перелом голови')
    button16 = KeyboardButton('Сонячний удар')
    button17 = KeyboardButton('Обмороження')
    button18 = KeyboardButton('Утоплення')
    bite_btn = KeyboardButton('Укуси')
    button19 = KeyboardButton('Укуси отруйних змій')
    button20 = KeyboardButton('Укуси тварин')
    button21 = KeyboardButton('Укуси комах')

    start_keyboard =
ReplyKeyboardMarkup(resize_keyboard=True).add(start_btn).add(dan
ger_btn).add(hurt_btn) \
        .add(help_btn).add(project_btn)
```

```
    danger_keyboard =
ReplyKeyboardMarkup(resize_keyboard=True).add(button6).add(butto
n7).add(button8).add(button9) \
        .add(button10).add(button11)
    hurt_keyboard =
ReplyKeyboardMarkup(resize_keyboard=True).add(button12).add(butt
on13).add(button14) \

.add(button15).add(button16).add(button17).add(button18).add(bit
e_btn)
    bite_keyboard =
ReplyKeyboardMarkup(resize_keyboard=True).add(button19).add(butt
on20).add(button21)


    @dp.message_handler(commands=['start'])
    async def start_command(message: types.Message):
        await message.reply(emojize("Привіт!\n\n Я той, хто
допоможе тобі надати першу домедичну допомогу"),
                            reply_markup=start_keyboard)


    @dp.message_handler(regexp=r'/danger')
    async def danger_command(message: types.Message):
        await message.reply("Оберіть питання, яке вас турбує:",
                            reply_markup=danger_keyboard)
```

```
    @dp.message_handler(regexp=r'Можливий удар електричним
струмом')
    async def danger_command(message: types.Message):
        await message.reply("Дане повідомлення не містить в собі
корисної інформації, "
                            "проте показує працездатність логіки
бота",
                            reply_markup=start_keyboard)


    @dp.message_handler(regexp=r'Загроз не виявлено')
    async def no_warning_command(message: types.Message):
        await message.reply("Тепер, коли загроза життю не
актуальна можна допомогти людині. \n"
                            "Виберіть вид ушкодження із меню
ушкоджень /hurt",
                            reply_markup=danger_keyboard)


    @dp.message_handler(regexp=r'/hurt')
    async def no_warning_command(message: types.Message):
        await message.reply("Виберіть тип ушкодження",
                            reply_markup=hurt_keyboard)
```

```
@dp.message_handler(regexp=r'Укуси')
async def no_warning_command(message: types.Message):
    await message.reply("Виберіть тип укусу",
                        reply_markup=bite_keyboard)


@dp.message_handler(regexp=r'тварин')
async def no_warning_command(message: types.Message):
    await message.reply("Дане повідомлення не містить в собі корисної інформації, "
                        "проте показує працездатність логіки бота",
                        reply_markup=start_keyboard)


@dp.message_handler(regexp=r'/help')
async def no_warning_command(message: types.Message):
    await message.reply("Бот для надання першої домедичної допомоги. \n\n"
                        "Список команд ви бачите на клавіатурі. \n"
                        "Сподіваюсь, я не стану вам у нагоді.",
                        reply_markup=start_keyboard)
```

```
@dp.message_handler(regexp=r'/project')
async def no_warning_command(message: types.Message):
    await message.reply("Made by Maksym Ostapenko, IN-82",
                        reply_markup=start_keyboard)


if __name__ == '__main__':
    executor.start_polling(dp)
```