МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В.Н. Каразіна

Факультет: **ННІ Каразінський банківський інститут**
Кафедра: **Інформаційних технологій та математичного моделювання**
Спеціальність: **122 Комп'ютерні науки**
Освітня програма: **Комп'ютерні науки**
Група: АК-21М **денна форма навчання**

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

РОЗРОБКА ТА ДОСЛІДЖЕННЯ TELEGRAM-БОТА ДЛЯ ОРГАНІЗАЦІЇ СПОРТИВНИХ ЗАХОДІВ
ЗА НАКАЗОМ № 4601-5_3262 ВІД 15 вересня 2025 РОКУ

здобувача вищої освіти **Артемчук Артем Вадимович**

Робота допущена до захисту в ЕК
протокол кафедри ІТММ № 5 від 02.12.2025 р.

Завідувач кафедри ІТММ
к.п.н., доцент
_____ Н.І. Стяглик

Науковий керівник
к.ф.-м.н., доцент
_____Макарова Г. В.

м. Харків 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В. Н. Каразіна

Факультет навчально-науковий інститут "Каразінський банківський інститут"
Кафедра інформаційних технологій та математичного моделювання
Рівень вищої освіти другий (магістерський)
Спеціальність 122 Комп'ютерні науки
Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ **Н. І. Стяглик**
Підпис ініціали, прізвище

"15" вересня 2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ (ПРОЄКТ)

Артемчук Артем Вадимович
(прізвище, ім'я, по батькові студента)

1. Тема роботи: Розробка та дослідження Telegram-бота для організації спортивних заходів

керівник роботи к.ф.-м.н., доц Макарова Г.В.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від "15" вересня 2025 року № 4601-5 3262

2. Строк подання студентом роботи 24 листопада 2025 року

3. Перелік питань, які потрібно розробити:
У розділі 1: Провести аналіз особливостей організації спортивних заходів у цифровому середовищі;проаналізувати роль цифрових сервісів у спортивному менеджменті; вивчити можливості Telegram та специфіку створення Telegram-ботів; здійснити огляд інструментів, бібліотек та технологічних рішень для розробки ботів; сформувати концептуальну структуру Telegram-бота.
У розділі 2: Розробити архітектуру програмного забезпечення Telegram-бота; виконати проєктування модулів системи; побудувати структуру бази даних та спроєктувати її моделі; створити UML-діаграми (Use Case, Activity, Sequence, Class); визначити алгоритми роботи основних модулів; розглянути принципи обробки подій; оцінити безпекові аспекти та підходи до масштабування системи.
У розділі 3: еалізувати функціонал Telegram-бота для реєстрації користувачів,створення та адміністрування спортивних заходів; описати модулі програмної реалізації та надати фрагменти коду; виконати тестування роботи системи; проаналізувати ефективність програмного продукту; визначити можливості подальшого розвитку та шляхи вдосконалення функціоналу.

4. План роботи

№ з/п	Назви етапів роботи
1	Вибір здобувачем теми кваліфікаційної бакалаврської роботи
2	Затвердження плану і завдання кваліфікаційної бакалаврської роботи
3	Здача кваліфікаційної бакалаврської роботи керівнику
4	Підпис кваліфікаційної бакалаврської роботи керівника
5	Підпис кваліфікаційної бакалаврської роботи у нормоконтролера
6	Допуск завідувачем кафедри до захисту кваліфікаційної бакалаврської роботи
7	Захист кваліфікаційної бакалаврської роботи

5. Дата видачі завдання 15 вересня 2025 року

Студент _____ А.В.Артемчук
підпис ініціали, прізвище

Керівник роботи _____ Макарова Г. В.
підпис ініціали, прізвище

РЕФЕРАТ
НА КВАЛІФІКАЦІЙНУ МАГІСТЕРСЬКУ РОБОТУ
«Розробка та дослідження Telegram-бота для організації спортивних заходів»
Артемчук Артем Вадимович

Кваліфікаційна магістерська робота містить: 100 сторінки, 1 таблиць, 9 рисунків, список літератури із 30 найменувань.

Об′єктом дослідження є процес організації та управління спортивними заходами у цифровому середовищі.

Предметом дослідження є алгоритми, методи та програмні засоби автоматизації спортивних заходів за допомогою Telegram-бота.

Мета кваліфікаційної магістерської роботи полягає у розробці, дослідженні та впровадженні Telegram-бота, який забезпечує автоматизацію процесів реєстрації, адміністрування та комунікації під час організації спортивних заходів.

Завданнями кваліфікаційної магістерської роботи є:

– проаналізувати сучасний стан цифровізації спортивної сфери та інструменти автоматизації подієвої діяльності;

– дослідити архітектуру Telegram API та можливості створення інтелектуальних ботів;

– розробити структуру бази даних та програмну архітектуру Telegram-бота;

– реалізувати функціонал реєстрації користувачів, створення та адміністрування спортивних заходів;

– провести тестування програмного продукту та оцінити ефективність його використання;

– визначити перспективи подальшого розвитку системи.

Актуальність дослідження: Цифровізація спортивної сфери вимагає упровадження сучасних інструментів, здатних забезпечити ефективну організацію масових спортивних заходів, оптимізувати управління учасниками та покращити інформаційну взаємодію. Telegram є однією з найпопулярніших комунікаційних платформ, що робить використання Telegram-ботів перспективним напрямом автоматизації спортивного менеджменту. Дані інструменти дозволяють відмовитися від застарілих методів реєстрації, мінімізувати людські помилки та створити єдину інформаційну систему взаємодії між організаторами й учасниками.

За результатами дослідження: створено програмний Telegram-бот із модульною архітектурою, реалізованими алгоритмами реєстрації, управління подіями та комунікації з користувачами; проведено аналіз його ефективності, встановлено відповідність системи сучасним вимогам цифрового середовища та визначено напрями її подальшої модернізації.

Практична новизна: розроблено інтегровану систему автоматизації спортивних заходів на базі Telegram-бота, що поєднує гнучку архітектуру, асинхронний обробник подій та централізовану базу даних; удосконалено процеси реєстрації та комунікації між організаторами й учасниками;

запропоновано підходи до масштабування системи та адаптації її під різні види спортивних активностей.

Одержані результати можуть бути використані: у спортивних клубах, федераціях, навчальних закладах, під час організації масових змагань, турнірів та інших спортивних подій, що потребують автоматизованої системи управління учасниками та оперативної комунікації.

КЛЮЧОВІ СЛОВА: TELEGRAM-БОТ, СПОРТИВНІ ЗАХОДИ, АВТОМАТИЗАЦІЯ , PYTHON, AIOGRAM, БАЗА ДАНИХ, TELEGRAM API, ЦИФРОВІЗАЦІЯ СПОРТУ, ІНФОРМАЦІЙНІ СИСТЕМИ.

ABSTRACT
AT QUALIFICATION BACHELOR WORK

«Development and research of a Telegram bot for organizing sporting events»

Artemchuk Artem Vadymovych

The bachelor's thesis contains 100 pages, 1 tables, 9 drawings, a list of references of 30 titles.

The object of the is the process of organizing and managing sporting events in a digital environment.

The subject of the there are algorithms, methods, and software tools for automating sporting events using a Telegram bot.

The purpose of the bachelor thesis is to develop, research, and implement a Telegram bot that automates the processes of registration, administration, and communication during the organization of sporting events.

The tasks of a bachelor's degree are:

– to analyze the current state of digitalization in the sports sector and tools for automating event activities;

– to study the architecture of the Telegram API and the possibilities for creating intelligent bots;

– to develop the database structure and software architecture of the Telegram bot;

– to implement the functionality of user registration, creation, and administration of sporting events;

– to test the software product and evaluate the effectiveness of its use;

– to determine the prospects for further development of the system.

Relevance of the study: The digitalization of the sports sector requires the introduction of modern tools capable of ensuring the effective organization of mass sporting events, optimizing participant management, and improving information exchange. Telegram is one of the most popular communication platforms, making the use of Telegram bots a promising direction for the automation of sports management. These tools make it possible to abandon outdated registration methods, minimize human error, and create a unified information system for interaction between organizers and participants.

According to the results of the research: Based on the research results, a Telegram bot with modular architecture, implemented algorithms for registration, event management, and communication with users was created; its effectiveness was analyzed, the system's compliance with modern digital environment requirements was established, and directions for its further modernization were determined.

Practical novelty: an integrated system for automating sporting events based on a Telegram bot has been developed, combining flexible architecture, asynchronous event processing, and a centralized database; registration and communication processes between organizers and participants have been improved; approaches to scaling the system and adapting it to different types of sports activities have been proposed.

The results obtained can be used: in sports clubs, federations, educational institutions, during the organization of mass competitions, tournaments, and other sporting events that require an automated participant management system and operational communication.

KEYWORDS: TELEGRAM-BOT, SPORTS EVENTS, AUTOMATION, PYTHON, AIOGRAM, DATABASE, TELEGRAM API, DIGITALIZATION OF SPORTS, INFORMATION SYSTEMS.

ЗМІСТ

ВСТУП

У контексті поточної цифрової трансформації суспільства спостерігається зростання потреби в автоматизації процесів організації та адміністрування інформаційних потоків. Це питання особливо важливе для спортивних заходів, де швидкість, точність і зручність комунікації між організаторами та учасниками мають вирішальне значення. Традиційні методи, такі як паперові записи, телефонні дзвінки або окремі месенджери, не забезпечують необхідної структури та швидкості обміну інформацією. Тому є потреба в розробці спеціалізованих інформаційних систем, які могли б автоматизувати основні організаційні процеси. Одним із шляхів вирішення цих задач є використання Telegram-ботів – автоматизованих сервісів у популярному месенджері Telegram.

Актуальність теми Розробка та дослідження Telegram-бота для організації спортивних заходів визначається тим, що Telegram є однією з найзручніших платформ для автоматизації завдяки відкритому API, високому рівню захисту, широким можливостям інтеграції та популярності серед користувачів. Telegram-боти дають змогу створювати сервіси різної складності та забезпечують оперативну взаємодію між системою та користувачем без потреби у встановленні додаткового програмного забезпечення. Зі збільшенням кількості спортивних ініціатив, турнірів, тренувань та заходів різного масштабу, застосування Telegram-бота здатне суттєво підвищити ефективність організації. Отже, створення спеціалізованого Telegram-бота має як практичне, так і наукове значення.

Аналіз наявних досліджень показує, що проблематика спирається на дослідження в царині інформаційних систем, чат-ботів і цифрової взаємодії. Питання автоматизації спортивних заходів розглядалося у працях Хом’яка І. І. та Мельника Ю. В., які досліджували використання месенджерів у спорті та освіті; Весселя М. і Тіссе Ф., що аналізували вплив чат-ботів на цифрову взаємодію; Марченка О. О., який вивчав можливості використання Telegram-

ботів для бізнесу та організації. Дослідження з розробки ботів для Telegram, зокрема роботи Python Software Foundation, авторів бібліотеки Aiogram, а також офіційна документація Telegram Bot API, формують основу для розробки таких систем. Водночас, більшість існуючих рішень є загальними і не враховують унікальні потреби спортивної сфери, що підтверджує необхідність створення спеціалізованого інструменту для організації спортивних подій.

Метою цього аналізу є розробка та вивчення Telegram-бота, що автоматизує процеси організації спортивних заходів, включно з реєстрацією учасників, наданням інформації, формуванням розкладів і збором даних.

Для досягнення поставленої мети необхідно виконати наступні завдання:

– проаналізувати особливості організації спортивних заходів і визначити ключові процеси, які потребують автоматизації;

– вивчити сучасні інформаційні системи, месенджери та технології створення чат-ботів;

– вивчити структуру Telegram Bot API та можливості бібліотеки Aiogram;

– розробити структурну та інформаційну модель Telegram-бота;

– впровадити функціональні частини бота за допомогою Python та Aiogram;

– провести тестування створеного програмного забезпечення;

– провести аналіз продуктивності розробленої системи та визначити можливі шляхи її поліпшення.

Об’єктом аналізу є процес організації та інформаційного забезпечення спортивних заходів.

Предметом дослідження є методи та інструменти автоматизації цього процесу із застосуванням Telegram-бота.

У роботі використані наступні способи: аналіз і синтез даних, порівняння, системний підхід, структурне та об’єктно-орієнтоване

проєктування, способи моделювання, алгоритмічний аналіз, експериментальне тестування програмного продукту.

Матеріалом для цього аналізу слугували наукові праці, поточні публікації щодо використання чат-ботів, матеріали з розробки Telegram Bot API, офіційна документація Python та Aiogram, а також практичні приклади реалізації ботів, які наведені на професійних форумах.

Перевірка дієвості результатів аналізу проводилася в процесі демонстрації роботи Telegram-бота на практичних заняттях, консультаціях з науковим керівником і під час проходження переддипломної практики. Отримані дані можуть бути взяті для подальшого покращення системи або впровадження в діяльність спортивних організацій, клубів і навчальних закладів.

Таким чином, вступ окреслює основні умови, науковий контекст і план аналізу, визначає логіку подальшого викладу матеріалу і підкреслює важливість теми та її роль у теорії і практиці організації спортивних заходів у цифровому світі.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ ТА АНАЛІЗ СУЧАСНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ У СФЕРІ ОРГАНІЗАЦІЇ СПОРТИВНИХ ЗАХОДІВ

1.1Специфіка організації спортивних заходів у цифровому середовищі

Проведення спортивних заходів є складним процесом, що охоплює планування, координацію ресурсів, управління учасниками, інформаційний супровід та підготовку результатів. Традиційно, ці процеси здійснювалися вручну з використанням паперової документації, усних повідомлень або локальних електронних таблиць. Проте, прогрес в області цифрових технологій, інтернет-комунікацій і мобільних пристроїв значно змінив підхід до проведення спортивних подій, відкривши нові можливості для автоматизації та спрощення діяльності організаторів, тренерів і учасників.

Сьогодні цифрове середовище включає в себе широкий спектр технологій - від веб-сайтів і мобільних застосунків до інтерактивних сервісів у месенджерах, систем адміністрування подій і комплексних інформаційних рішень. У контексті спортивних подій перехід на цифрові технології дозволяє значно поліпшити ефективність організації, зменшити витрати часу, зменшити вплив людського фактора і забезпечити зручну взаємодію для всіх учасників. Перехід до цифрових інструментів став реакцією на потреби сучасної аудиторії - спортсменів, тренерів, суддів і організаторів, яким потрібен швидкий доступ до інформації, регулярні оновлення та можливість взаємодії.

Централізоване адміністрування процесами є однією з найважливіших особливостей цифрового середовища. У спортивних заходах це відображається в можливості ведення обліку учасників, управління розкладом, автоматичного інформування про зміни та збору підсумків через єдину платформу. Такі системи дозволяють уникнути дублювання даних, зменшити кількість помилок, які трапляються при ручному введенні, і створити чітку структуру управління. Цифрові технології також гарантують

мобільність і доступність - організатори й учасники можуть отримати необхідну інформацію, де б вони не були.

Швидкість є особливо важливою в цифровому просторі. Спортивні заходи часто вимагають швидких рішень, миттєвої реакції на зміни, координації дій між суддями, тренерами, волонтерами та учасниками. Традиційні інструменти, як-от електронна пошта або роздруковані розклади, не завжди здатні забезпечити необхідну швидкість комунікації. На відміну від них, цифрові канали, особливо ті, що працюють у форматі миттєвих повідомлень, забезпечують майже миттєву передачу інформації. Саме тому месенджери є одним з найефективніших каналів для спілкування всередині організації.

Під час організації спортивних заходів у цифровому середовищі важливо зберігати і швидко обробляти великі обсяги даних. Це можуть бути списки учасників, суддівські протоколи, результати змагань, розклади, характеристики команд або окремих спортсменів. З появою цифрових технологій упорядкування та обробка цих даних стали набагато ефективнішими. Інформаційні системи дозволяють автоматично формувати статистику, створювати звіти, аналізувати дані та контролювати діяльність у режимі реального часу.

Інтерактивність також є важливою частиною цифрового середовища, оскільки вона дає змогу користувачам не тільки отримувати інформацію, а й взаємодіяти з системою. У випадку спортивних заходів, учасники можуть реєструватися на змагання, отримувати сповіщення, переглядати результати, подавати заявки на участь у нових подіях або змінювати дані в профілі. Організатори можуть адмініструвати події, відстежувати активність користувачів і реагувати на непередбачені обставини в режимі реального часу.

Доступність і універсальність цифрових інструментів відіграють важливу роль. Мобільні пристрої стали основним засобом отримання інформації для більшості людей, що робить мобільні сервіси ідеальним

рішенням для організації спортивних заходів. Застосунки та сервіси, інтегровані в мобільні платформи, дозволяють користувачам взаємодіяти будь-коли, без потреби бути на робочому місці біля комп'ютера. Це особливо важливо для спорту, де більшість активностей відбуваються поза офісом.

Крім того, перехід на цифрові технології поліпшує взаємодію між усіма учасниками процесу. Можливість створення групових чатів, каналів, інтерактивних меню та автоматичних сповіщень забезпечує високий рівень обізнаності, швидкість реагування та зменшує ризик втрати важливих даних. Зокрема, Telegram став однією з найпопулярніших платформ для цих цілей завдяки своїй структурі, відкритості API, захищеності даних і легкості інтеграції зі сторонніми сервісами.

Важливо зазначити, що впровадження цифрових технологій відкриває широкі можливості для зворотного зв'язку. Організатори можуть швидко отримувати відгуки учасників, проводити опитування, збирати пропозиції та покращувати майбутні заходи на основі отриманих даних. Цифрові платформи дозволяють аналізувати статистику поведінки користувачів, визначати найпопулярніші заходи, вивчати зацікавленість за віком або видами спорту.

При організації спортивних заходів у цифровому просторі важливо забезпечити належний рівень безпеки. Оскільки такі платформи працюють з особистими даними, їх потрібно захищати відповідно до міжнародних стандартів кібербезпеки. У цьому випадку Telegram забезпечує багаторівневий захист, включно з шифруванням,автентифікацією за допомогою токенів і контролем доступу до API.

Отже, організація спортивних заходів у цифровому середовищі характеризується швидким обміном інформацією, багаторівневою інтерактивністю, централізованим управлінням, автоматизацією рутинних задач, захистом даних і широкою доступністю. Все це створює сприятливі умови для активного використання Telegram-ботів як ефективного способу автоматизації організаційних процесів у спорті.

1.2 Інформаційні системи управління спортивними подіями: види та їхнє призначення

Інформаційні системи управління спортивними заходами відіграють важливу роль у сучасному спорті, надаючи організаторам, учасникам і глядачам засоби для планування, координації та контролю. Поява та розвиток таких систем стали відповіддю на збільшення масштабів спортивних заходів, підвищення вимог до організації та необхідність оперативної обробки великих обсягів даних. Таким чином, інформаційні технології стали необхідним інструментом для організаторів спортивних змагань різного рівня, від місцевих турнірів до міжнародних чемпіонатів.

Інформаційні системи в цій галузі є комплексними, оскільки вони охоплюють різні етапи проведення заходів: підготовку, реєстрацію учасників, управління інфраструктурою, контроль під час змагань, фіксацію та аналіз результатів. За допомогою цифрових технологій забезпечується відкритість процесів, зменшуються організаційні ризики, збільшується точність обробки інформації та зменшується обсяг помилок, пов'язаних із людським фактором.

Залежно від функціонального призначення та специфіки, інформаційні системи управління спортивними заходами можна поділити на декілька основних видів.

Перший вид - це системи для реєстрації учасників і адміністрування їхніх даних. Такі платформи дозволяють спортсменам подавати заявки онлайн, створювати особисті профілі, завантажувати документи та вибирати види спорту для участі. Організатори можуть контролювати кількість зареєстрованих учасників, формувати списки та вести базу даних у структурованому вигляді. Подібні сервіси спрощують підготовку до змагань, зменшують час обробки заявок і мінімізують ризик дублювання інформації.

До другого виду належать системи управління розкладом і логістикою заходу. Вони дають змогу створювати детальні графіки змагань, розподіляти ресурси, формувати турнірні таблиці та визначати час для кожної активності. Такі системи особливо важливі для масштабних подій, де велика кількість

дисциплін і категорій вимагає точного планування. Крім того, цифрові розклади можуть оновлюватися в реальному часі, що дозволяє швидко вносити зміни та повідомляти про них учасників.

Третій вид охоплює системи збору та обробки результатів змагань. Вони використовуються для фіксації спортивних показників, ведення протоколів, розрахунку турнірних таблиць і формування підсумкових даних. У таких системах передбачена автоматизація збору даних, що забезпечує високу точність і пришвидшує процес підбиття підсумків. Ці сервіси можуть інтегруватися з електронним хронометражем і спеціальними пристроями.

Четвертий вид - інформаційно-комунікаційні системи, що забезпечують взаємодію між організаторами, учасниками та глядачами. Це можуть бути веб-сайти, мобільні застосунки або сервіси на основі месенджерів, які надають доступ до розкладів, новин, сповіщень і результатів. З розвитком комунікаційних технологій з'явилися інтерактивні рішення, як-от чат-боти, які автоматизують надання інформації та дозволяють користувачам взаємодіяти з системою у зручний спосіб.

П'ятий вид - аналітичні системи, які обробляють дані, контролюють, оцінюють результативність спортивних заходів і створюють статистичні звіти. Вони дозволяють організаторам приймати обґрунтовані управлінські рішення, виявляти закономірності та прогнозувати розвиток подій. Вони можуть об'єднуватися з іншими платформами для отримання повної інформації про спортивний процес і створення аналітичних моделей.

Крім класифікації за функціями, інформаційні системи можна розділити за способом впровадження. До них належать локальні системи, що використовуються в межах певного спортивного об'єкта чи організації; веб-системи, доступ до яких здійснюється через інтернет; і мобільні застосунки, призначені для використання на смартфонах і планшетах. Сучасні тенденції вказують на зростання популярності хмарних сервісів, які забезпечують масштабованість, надійність і зручний доступ для учасників процесу.

Зараз інформаційні системи інтегруються з іншими цифровими платформами, як-от соціальні мережі, платформи для онлайн-трансляцій, системи продажу квитків і CRM. Завдяки цьому спортивні події стають більш відкритими та інтерактивними, організатори мають змогу розширити аудиторію та поліпшити досвід відвідувачів.

У цифровій екосистемі особливе положення займають чат-боти, зокрема Telegram-боти. Вони поєднують у собі функціональність інформаційної системи та комунікативні можливості месенджера. Завдяки легкості використання, швидкості обміну повідомленнями та доступності широкій аудиторії, Telegram-боти стали ефективним інструментом організації спортивних подій. Вони допомагають реєструватися, повідомляти, нагадувати, збирати дані та проводити аналіз, що робить їх багатофункціональним інструментом для спортивних організацій різного рівня.

Отже, інформаційні системи управління спортивними заходами охоплюють широкий спектр технологій, які автоматизують організаційні процеси, роблять їх більш ефективними та забезпечують взаємодію між усіма учасниками спортивного процесу. Їхня класифікація дозволяє глибше зрозуміти особливості використання різних видів систем і визначити оптимальні інструменти для виконання певних завдань. У контексті поточної цифрової переорієнтації особливого значення набувають мобільні та інтерактивні рішення, в яких Telegram-боти займають чільне місце завдяки своїй доступності, універсальності та функціональності.

1.3 Месенджери як платформа для автоматизації процесів: огляд сучасних сервісів

У сучасному інформаційному суспільстві месенджери стали одним із ключових інструментів комунікації, які поступово перетворилися на універсальні платформи для вирішення широкого спектра питань - від повсякденного спілкування до реалізації складних бізнес-процесів. Їхня популярність зумовлена доступністю, простотою використання, високою швидкістю передачі повідомлень і можливістю об'єднання з іншими цифровими сервісами. У спорті месенджери відіграють особливу роль, оскільки забезпечують оперативну комунікацію, автоматизують взаємодію з учасниками та створюють канали для розповсюдження інформації.

Мобільність є однією з основних причин зростання інтересу до месенджерів. Більшість власників смартфонів мають змогу отримувати повідомлення та користуватися інтерактивними сервісами практично будь-де. Це перетворює месенджери на платформу для організації заходів, де важливо оперативно повідомляти учасникам про зміни в розкладі, місці чи інших умовах. Крім того, оскільки основні канали комунікації знаходяться в одному застосунку, користувачам не потрібно встановлювати додаткові програми, що спрощує взаємодію.

Месенджери виконують роль універсального інтерфейсу для автоматизації. Багато з них підтримують створення ботів - спеціальних програмних модулів, які можуть виконувати команди користувачів, надавати інформацію, обробляти дані та керувати сторонніми сервісами. У такому випадку месенджери розширюють організаційні спроможності, оскільки дозволяють значною мірою автоматизувати реєстрацію, інформування та збір даних.

Найбільш відомі месенджери сьогодення - Telegram, Viber, WhatsApp, Facebook Messenger і Discord. Кожен із них має певні особливості й інструменти для взаємодії, проте Telegram визнано найзручнішою та

найпопулярнішою платформою для ботів. Її перевагами є: відкритий API, розвинена інфраструктура та широкі можливості для персоналізації.

Telegram відрізняється високим рівнем захисту даних, можливістю створення ботів будь-якої складності, гнучкою системою сповіщень і підтримкою інструментів - від інлайн-кнопок до webhook-з'єднань. Telegram став платформою вибору для багатьох розробників інтерактивних рішень. Його екосистема дає змогу створювати сервіси для реєстрації учасників, управління розкладом, проведення опитування та формування сповіщень у рамках спортивних подій будь-якого масштабу.

Viber є популярним у деяких регіонах, проте має обмежені можливості для автоматизації. Боти Viber підтримують базовий функціонал, проте значно поступаються Telegram у гнучкості, швидкості інтеграції та масштабованості. Проте для певних локальних заходів він може бути ефективним, особливо якщо цільова аудиторія активно використовує цю платформу. Viber-боти здебільшого застосовуються для розсилок та інформування.

WhatsApp залишається одним із наймасовіших месенджерів у світі, але можливості ботів є суттєво обмеженими. WhatsApp Business API передбачає складну процедуру реєстрації й налаштування, а також обмеження на типи повідомлень. Через це WhatsApp рідше використовується для автоматизації спортивних заходів, хоча для базової комунікації та розсилок він є зручним інструментом.

Facebook Messenger має розвинену систему створення ботів, проте його популярність падає через зміни політики компанії, високі вимоги до модерації та зменшення активності користувачів в українському регіоні. Незважаючи на це, Messenger підтримує ботів і надає інструменти для об'єднання з іншими системами.

Discord став популярним в ігрових спільнотах, але активно використовується для організації подій, спільнот і турнірів. Його боти можуть виконувати багато функцій - від модерації до збору статистики. Проте для

традиційних спортивних заходів Discord використовується не так часто, оскільки значна частина аудиторії залишається поза межами цієї платформи.

Порівнюючи функціональність месенджерів, можна стверджувати, що Telegram є найоптимальнішою платформою для автоматизації процесів у спорті. Головні переваги - відкритість API, детальна документація, підтримка бібліотек для різних мов програмування, підтримка інлайн-елементів, високий рівень безпеки та можливість створення ботів без додаткових витрат. Важливим аспектом є підтримка великої кількості інтерактивних елементів - кнопок, меню, реакцій, каруселей, що дозволяє створювати зручний інтерфейс для користувача.

Тому месенджери стали ефективним інструментом у спорті. Вони автоматизують основні процеси, зменшують навантаження на організаторів і забезпечують учасникам зручний спосіб отримання інформації. Месенджери дають змогу впроваджувати гібридні рішення - об'єднання веб-сайтів із ботами, що дозволяє користувачам взаємодіяти зі складними системами у форматі чату.

Отже, месенджери стали середовищем для автоматизації в спорті. Їхня популярність, мобільність, можливості та розвиток ботів дозволяють створювати рішення, які спрощують зв'язок, роблять адміністрування простішим і підвищують ефективність організаційних процесів. Telegram є гнучкою платформою для створення сервісів.

1.4 Telegram як інструмент створення інтерактивних сервісів: структура, можливості, API

Telegram є одним із найсучасніших і функціональних месенджерів, який трансформувався в платформу для створення цифрових сервісів. Його популярність визначається високою швидкістю роботи, надійністю, можливістю використання на різних платформах і можливостями для розробників. На відміну від багатьох месенджерів, Telegram має структуру та офіційно підтримує створення програмних модулів - ботів, що робить його однією з найкращих платформ для автоматизації процесів, у тому числі в спорті.

Структура Telegram розроблена для масштабованості, високої передачі даних та захисту. Telegram використовує систему серверів у різних країнах. Це дозволяє системі обробляти великі обсяги трафіку та надавати доступ до сервісу. Крім того, Telegram використовує шифрування, що гарантує безпеку даних користувачів, у тому числі під час взаємодії з ботами.

Головним елементом у роботі ботів є Telegram Bot API - інтерфейс взаємодії між ботом і Telegram. Він працює за принципом REST-API, що дозволяє відправляти запити та отримувати відповіді у форматі JSON. Bot API підтримує методи, які забезпечують можливість надсилання текстових повідомлень, файлів, даних, документів, геолокації, кнопок. Розробники можуть створювати з усім сервіси - від інформаційних ботів до систем управління заходами.

Telegram використовує механізми взаємодії ботів із серверами: polling і webhook.

Polling передбачає надсилання ботом запитів на отримання нових повідомлень. Цей спосіб простий у впровадженні та підходить для тестування.

Webhook працює за принципом передачі даних: Telegram надсилає боту інформацію про події. Цей підхід оптимальний для великих систем і гарантує швидку реакцію бота.

Telegram підтримує інструменти, що створюють зручний інтерфейс. Серед них - InlineKeyboard, ReplyKeyboard, CallbackQuery, меню, каруселі. Використання інструментів поліпшує взаємодію з користувачем і дозволяє структурувати логіку. Наприклад, організатор заходів може створити зручне меню для реєстрації, зміни розкладу.

Перевагою Telegram є інтеграція ботів із сервісами. За допомогою API розробник може підключати бази даних, календарі та системи. Це створює цифрові рішення, які автоматизують більшість процесів заходів. Бот може формувати списки, надсилати сповіщення та копіювати дані.

Telegram підтримує асинхронну взаємодію для ботів із користувачами. Aiogram є однією з бібліотек, яка забезпечує зручну логіку ботів і управління проєктами.

Telegram не має обмежень для розробників. Telegram дає змогу запускати бота будь-якої складності, що робить його зручним для автоматизації заходів зі швидкою розробкою.

Властивістю Telegram є багатоплатформність. Користувачі можуть взаємодіяти з ботом на будь-якому пристрої, що забезпечує доступність для учасників.Telegram є платформою для інтерактивних сервісів завдяки структурі, гнучкості й інтерфейсним елементам. Використання Telegram у спорті надає автоматизацію, та взаємодію зі сторонніми сервісами. Тому Telegram є платформою для розробки ботів, які відповідають вимогам спортивної сфери.

1.5 Telegram-боти: принципи роботи, класифікація, інтерфейси взаємодії

Telegram-боти є складником екосистеми Telegram, який об'єднує функції автоматизації, інтерактивної взаємодії та доступу до зовнішніх сервісів. Завдяки широкому набору інструментів і можливостям інтеграції, Telegram-боти є засобом створення рішень у різних сферах - від електронної комерції до освіти та спорту. Використання дозволяє покращити взаємодію та підвищити ефективність систем.

Основним принципом роботи Telegram-ботів є взаємодія між користувачем і програмним модулем через Telegram Bot API. Бот не є окремою програмою, а отримує повідомлення від користувачів через Telegram, обробляє їх, а потім надсилає відповідь у вигляді тексту, кнопок. Так, Telegram-бот є сполучною ланкою між користувачем і розробником.

Telegram-боти працюють за моделями: long polling і webhook.

Перший метод полягає у надсиланні запитів до Telegram для отримання повідомлень. У такий спосіб бот перевіряє наявність оновлень. Ця модель проста у виконанні та підходить для тестування.

Webhook забезпечує передачу даних у часі - Telegram надсилає інформацію для обробки на вказану URL-адресу. Цей метод є ефективнішим для великої аудиторії, оскільки забезпечує низьку затримку та зменшує навантаження на сервер.

Telegram-боти відрізняються за функціональністю та способом взаємодії. Залежно від використання їх поділяють на декілька основних груп.

Першу групу становлять інформаційні боти, які надають користувачам інформацію, відповідають на запити та надсилають новини. Такі боти корисні для спортивних організацій, оскільки автоматично надсилають інформацію про розклад подій.

Другу групу утворюють сервісні або функціональні боти, які виконують задачі: реєстрацію, обробку форм та бронювання. Вони є

найпопулярнішим типом ботів у спорті, бо дозволяють автоматизувати подання заявок і отримання статистики.

До третьої групи належать інтерактивні боти, що взаємодіють з користувачами за допомогою кнопок і меню. Telegram підтримує типи кнопок inline та reply, які дозволяють реалізувати логіку діалогів. Користувачі можуть обирати опції та переходити по меню.

Четверту групу становлять інтеграційні боти, призначені для взаємодії з сервісами та платформами. Вони можуть підключатися до баз даних і зовнішніх платформ. Такі боти дозволяють реалізувати системи аналізу даних, що є актуальним у спорті.

Окремо адміністративні боти виконують функції модератора та займаються управлінням групами. У контексті спортивних заходів такі боти підтримують порядок у чатах.

Інтерфейси взаємодії відіграють важливу роль у забезпеченні зручності роботи сервісу. Telegram пропонує інструменти управління діалогами. Одним із них є текстовий інтерфейс, що передбачає взаємодію через введення команд. Він підходить для звичайних сценаріїв, але може бути незручним при великій кількості функцій.

Іншим інтерфейсом є кнопковий, який включає ReplyKeyboardMarkup та InlineKeyboardMarkup. Перший дозволяє створити кнопки, що відображаються замість клавіатури користувача, тоді як другий формує кнопки під повідомленням. Inline-кнопки можуть передавати дані, забезпечуючи швидку реакцію без надсилання окремих повідомлень.

Окремим механізмом є меню команд, яке дозволяє користувачам орієнтуватися у функціоналі бота.

Telegram підтримує режими (inline mode), що дозволяють здійснювати пошук інформації або виконувати дії з поля введення в чаті. Це дозволяє створювати каталоги, які застосовують для пошуку результатів спортсменів.

Завдяки гнучкості Telegram API та підтримці бібліотек боти мають широке поле застосування в спорті. Вони дозволяють автоматизувати

взаємодію з учасниками, створювати інтерфейси, управляти даними та аналізувати статистику. У поєднанні з простотою використання, Telegram-боти є рішенням для спортивних систем.

1.6 Технічні та функціональні вимоги до системи організації спортивних заходів

Створення онлайн-системи для організації спортивних заходів потребує чіткого визначення технічних і функціональних вимог, що гарантують її надійність і зручність. З огляду на те, що спортивні змагання характеризуються великою кількістю людей, система повинна мати широкий функціонал, можливість розширення та забезпечувати стабільну роботу. Розробка Telegram-бота є головним інструментом, а вимоги стають важливими, оскільки бот є інтерфейсом між організаторами та учасниками.

Технічні вимоги визначають технічні особливості і архітектуру. Однією із вимог є технічний стек, що забезпечує стабільність, легкість інтеграції та масштабованість. Використання Telegram Bot API передбачає мову програмування, яка гарантує взаємодію з API та роботу у мережі. Розв'язанням є мова Python з фреймворком Aiogram, який гарантує стабільну обробку і швидкість розробки проєктів.

Важливим елементом є серверне середовище, де відбуватиметься обробка запитів бота. Зараз популярні рішення є хмарні платформи, такі як Heroku, AWS, Render, чи інші сервіси, які забезпечують доступність. Оскільки спортивні змагання мають навантаження, то сервер, повинен підтримувати масштабування, асинхронне виконання запитів.

Важливою є вимога управління базами даних. Бази даних потрібні для збереження інформації і швидкого доступу. Найчастіше у зв’язці з Telegram-ботами застосовують реляційні бази даних, такі як PostgreSQL або MySQL. PostgreSQL є популярною через стабільність. Розроблені системи повинні відповідати вимогам кібербезпеки та мати шифрування.

Функціональні вимоги визначають інструменти, які повинен мати Telegram-бот для управління спортивними змаганнями. Передусім, має бути механізм реєстрації користувачів зі створенням профілю і ролі. Функціонал реєстрації має бути простим, оскільки він є елементом взаємодії з користувачем.

Важливим функціоналом є управління спортивними подіями, що передбачає створення нових подій, додавання опису. Організатор, повинен мати змогу змінювати інформацію, вносити зміни до розкладу подій. Має бути автоматизоване сповіщення учасників про зміни, що гарантує оперативність.

Функцією, є й формування графіку. Telegram-бот генерує структуру сіток і графіків. Система має уможливлювати введення результатів та оновлювати турнірні таблиці. Система має підтримувати генерацію статистики.

Окремою вимогою є система сповіщень, зокрема Telegram. Сповіщення стосуються початку змагань, зміни розкладу, результатів.

Інтерфейс користувача є частиною функціональних вимог. Боти, мають забезпечити чітку структуру меню і зворотний зв’язок.

Отже, вимоги до системи організації спортивних заходів формують основу для Telegram-бота. Вони визначають рішення розробки, логіку з користувачами і безпеку. Дотримання, цих вимог є необхідною умовою створення надійної системи, здатної автоматизувати ключові процеси і гарантувати взаємозв'язок між учасниками.

1.7 Аналіз наявних розв'язань для автоматизації

Нині спортивна індустрія активно впроваджує цифрові технології для збільшення ефективності організаційних процесів, поліпшення зв'язку та гарантування високої якості управління подіями. За останнє десятиліття з'явилася велика кількість інструментів, розроблених для автоматизації спортивних заходів різних масштабів - від місцевих змагань до міжнародних турнірів. Попри велику різноманітність наявних рішень, більшість із них мають обмеження, які ускладнюють пристосування до специфічних потреб організаторів, що створює причини для пошуку гнучкіших та інтерактивних платформ, як-от Telegram-боти.

З-поміж поширених рішень для управління спортивними подіями виділяють онлайн-платформи, мобільні програми, спеціалізовані системи управління подіями та сервіси обміну повідомленнями. Однією з найвідоміших міжнародних платформ є Sportlyzer, яка пропонує повний набір функцій для клубів, тренерів і спортсменів, зокрема календар подій, планування тренувань, ведення журналів і статистики. Проте Sportlyzer зорієнтований насамперед на командні види спорту та вимагає платної підписки, що обмежує його застосування для місцевих чи аматорських заходів.

Інший приклад - Eventbrite, популярна платформа для організації різних типів подій, зокрема спортивних. Вона дає можливість створювати сторінки заходів, продавати квитки, реєструвати учасників і поширювати відомості через соціальні мережі. Однак Eventbrite не має спеціальних інструментів саме для спортивних змагань, як-от формування сіток турнірів, ведення протоколів або облік спортивних результатів. Крім того, платформа орієнтована на комерційні події, що не завжди доречно для спортивних організацій.

Широко використовується також SportEventPlanner, яка забезпечує створення календаря змагань, облік учасників і планування інфраструктури. Хоча ця система має великі можливості, її інтерфейс важкий для нових

користувачів, а функції - надто великі для малих заходів. До того ж більшість таких платформ вимагають реєстрації на веб-сайті, що може знизити залучення спортсменів, які здебільшого користуються мобільними месенджерами.

Мобільні застосунки, розроблені для управління змаганнями, як-от TeamSnap, Heja, Spond та інші, теж досить популярні. Вони забезпечують зручний мобільний інтерфейс, дають змогу формувати команди, планувати матчі та надсилати сповіщення. Утім, багато з них зорієнтовані лише на аматорські спортивні клуби або мають обмежені можливості без оплати преміум-функцій. Деякі застосунки не підтримують гнучке адміністрування подій або не дають змоги приєднувати зовнішні сервіси, що обмежує їхні функції у великих змаганнях.

В Україні можна відзначити сервіси PlayScore, Turnir.ua, Sport.ua Tournament Tools. Вони пропонують базові засоби для формування турнірних таблиць, створення сіток і обліку результатів. Проте більшість із них мають певні недоліки: потрібно постійно переходити між веб-сторінками, немає зручних мобільних інтерфейсів, слабка інтерактивність, важкий процес реєстрації учасників і недостатня автоматизація. Ці системи часто вимагають втручання оператора, який власноруч вводить дані, що сповільнює роботу та збільшує ризик помилок.

Окрему категорію складають рішення, що ґрунтуються на месенджерах. Наприклад, деякі спортивні клуби використовують групи у Viber чи WhatsApp для комунікації, але ці інструменти мають значні обмеження: немає структурованих меню, незручний пошук відомостей, неможливо автоматизувати реєстрацію, слабка підтримка інтерактивних інструментів. WhatsApp Business API також суворо обмежує автоматичні розсилання, що ускладнює його застосування для великих подій. У такому вигляді месенджери виконують допоміжну роль, але не є повноцінними керуючими системами.

На цьому тлі Telegram-боти показують суттєві переваги. Telegram має відкриту платформу для створення ботів, підтримує програмовані інтерфейси, інтерактивні кнопки, швидкі повідомлення та великий спектр форматів вмісту. Крім того, Telegram-бот може працювати без встановлення зовнішніх застосунків, забезпечуючи доступність для всіх учасників. Але навіть у Telegram вже є певні рішення для спортивних заходів, хоча їх значно менше, ніж веб- чи мобільних платформ.

Прикладами Telegram-рішень є різні боти для реєстрації на тренування, формування команд або проведення голосувань. Проте більшість із них є простими, виконують лише одну функцію та не забезпечують повного спектра можливостей, потрібних для організації спортивного заходу: формування календаря, управління сіткою змагань, сповіщення, аналіз результатів та приєднання до бази даних. Ці боти не є універсальними та часто створюються для конкретного клубу чи заходу, без можливості збільшення масштабу чи зміни.

Основні недоліки наявних рішень можна поділити за кількома напрямами:

Обмежені функції. Багато платформ виконують лише окремі завдання (реєстрація, сповіщення, результати), але не забезпечують комплексного підходу.

Недостатня інтерактивність. Більшість веб-сервісів і мобільних застосунків вимагають додаткових дій від користувача, тоді як інтерактивні боти дають змогу виконувати дії у два-три кліки.

Високі фінансові витрати. Деякі платформи мають платні тарифи, що робить їх недоступними для аматорських чи місцевих спортивних подій.

Відсутність автоматизації. У багатьох системах дані вводяться власноруч, а сповіщення не утворюються автоматично, що сповільнює роботу та збільшує ризик помилок.

Складність у користуванні. Деякі сервіси мають важкий інтерфейс, що знижує зручність взаємодії для користувачів, особливо для спортсменів молодшого або старшого віку.

Проблеми з доступом. Мобільні застосунки потребують установлення, а веб-сайти - авторизації, що може негативно впливати на залученні.

Отже, аналіз наявних рішень вказує на те, що є широкий спектр інструментів для автоматизації спортивних подій, однак жодне з них не пропонує універсальний, доступний, мобільний та інтерактивний механізм, який легко пристосовується до різних форматів спортивних заходів. Це створює причини для розробки комплексного Telegram-бота, який поєднуватиме функції спеціалізованих систем із простотою та доступністю месенджера.

1.8 Висновки до розділу 1

У першому розділі роботи було проведено ретельний аналіз теоретичних аспектів, що стосуються переходу до цифрових технологій в організації спортивних заходів, також досліджено особливості застосування сучасних інформаційних технологій для забезпечення їхньої продуктивної роботи. Розгляд головних положень цифрового середовища дав змогу зробити висновок, що традиційні підходи до управління спортивними подіями вже не відповідають нинішнім вимогам, необхідності оперативності, інтерактивності та доступності інформації.

Встановлено, що цифрове середовище створює нові можливості для автоматизації основних процесів - від реєстрації до підбиття підсумків змагань. Мобільність, інтерактивність та можливість обробляти великі обсяги даних роблять цифрові платформи незамінним інструментом для сучасних спортивних організаторів. Особливу увагу було приділено ролі месенджерів, які завдяки своїй популярності та простоті використання стають універсальними каналами зв'язку між учасниками й організаторами спортивних подій.

Аналіз поширених месенджерів показав, що Telegram вирізняється серед інших платформ завдяки відкритому API, розвиненій структурі, гнучким засобам приєднання, великій кількості інструментів для взаємодії з користувачами та високим стандартам безпеки. Telegram-боти мають широкий спектр можливостей - від обробки команд до автоматизації складних сценаріїв, що робить їх дієвим засобом для створення інтерактивних інформаційних систем.

Під час розгляду технологічних і функціональних вимог було визначено основні параметри, яким має відповідати система організації спортивних заходів: надійність, можливість збільшення, безпека, зрозумілий інтерфейс, швидкість і підтримка приєднання до баз даних і зовнішніх сервісів. Саме ці вимоги є основою для проєктування Telegram-бота, здатного забезпечити

повний цикл управління спортивною подією - від реєстрації спортсменів до формування результатів.

Проведений аналіз наявних рішень показав, що попри велику кількість платформ і застосунків, жодне з них не дає комплексного підходу, який би поєднував інтерактивність, доступність, мобільність і гнучкість із можливістю пристосування до потреб конкретного заходу. Значна частина таких систем має обмежені функції, важкий інтерфейс або залежить від платних підписок, що робить їх непридатними для широкого застосування, особливо в місцевих чи аматорських спортивних організаціях.

Отже, підсумовуючи результати першого розділу, можна дійти висновку, що розробка Telegram-бота для організації спортивних заходів є актуальною та виправданою науково-практичною задачею. Бот здатен об’єднати в собі всі потрібні елементи дієвої інформаційної системи: інтуїтивний інтерфейс, гнучку логіку, швидку взаємодію з користувачами, автоматизацію головних процесів та доступність із будь-яких пристроїв. Отримані теоретичні результати створюють основу для подальшої розробки структури програмного забезпечення, яку буде розглянуто у другому розділі.

РОЗДІЛ 2.
АНАЛІЗ ПРОТОКОЛІВ ТА АЛГОРИТМІВ У СУЧАСНИХ МЕСЕНДЖЕРАХ

2.1 Вибір технологічного набору: Python, Aiogram, бази даних

Вибір технологічного набору є одним із найважливіших етапів розробки Telegram-бота, оскільки визначає можливості системи, її продуктивність, гнучкість і масштабованість. Для створення інструмента, здатного забезпечити продуктивну організацію спортивних заходів, потрібно обрати такі технології, які забезпечують надійну роботу з даними, асинхронну обробку великої кількості запитів, швидку взаємодію з Telegram Bot API та можливість зручного розміщення в хмарному середовищі. У цьому розділі обґрунтовується вибір Python як основної мови програмування, фреймворку Aiogram, бази даних та хмарних технологій.

Python є однією з найпопулярніших мов програмування для створення ботів завдяки своїй простоті, зручності читання й великій екосистемі бібліотек. Його синтаксис дає можливість швидко розробляти та підтримувати програмні рішення будь-якої складності. До того ж Python підтримує асинхронне програмування, яке є дуже важливим для роботи Telegram-ботів. Завдяки бібліотекам asyncio, aiohttp та іншим фреймворкам розробники можуть обробляти велику кількість паралельних запитів без втрати продуктивності. Це забезпечує швидку реакцію бота на повідомлення користувачів, що особливо важливо під час організації спортивних заходів, де інформація має надходити вчасно та без затримок.

Фреймворк Aiogram є однією з найкращих бібліотек для розробки ботів на Python. Він працює на основі асинхронної моделі та дає великі можливості для побудови складних логічних сценаріїв. Aiogram має зручну структуру обробників, підтримує FSM (Finite State Machine) для побудови діалогових станів, дає змогу створювати інлайн-кнопки, меню, callback-запити та втілювати багаторівневі інтерфейси. Порівнюючи з іншими фреймворками,

як-от Telebot або PyTelegramBotAPI, Aiogram забезпечує значно вищу продуктивність, зручність структурування коду та можливість збільшення масштабу проєкту.

Важливим елементом технологічного набору є система управління базами даних. Оскільки Telegram-бот для організації спортивних заходів має зберігати великий обсяг структурованої інформації - дані користувачів, події, розклади, результати - потрібне застосування надійної та продуктивної бази даних. Найкраще рішення - PostgreSQL, яка забезпечує стабільність, підтримку транзакцій, великі можливості автоматизації та високий рівень безпеки. PostgreSQL має високу продуктивність під час роботи зі складними запитами, що важливо у випадку аналітики або створення розкладів спортивних подій. Крім того, ця база даних добре приєднується до Python через бібліотеку asyncpg, яка підтримує асинхронну роботу.

Для невеликих або навчальних проєктів можна застосовувати й SQLite, однак її можливостей недостатньо для систем, що працюють у реальному часі, мають велику кількість користувачів або потребують збільшення масштабу. Тому вибір PostgreSQL є більш обґрунтованим для розробки Telegram-бота, здатного ефективно працювати у спортивному середовищі.

Окрему роль у виборі технологічного набору відіграють хмарні сервіси, які забезпечують можливість безперебійної роботи Telegram-бота в режимі 24/7. З-поміж популярних рішень - платформи Heroku, Render, Railway, AWS, також VPS-сервіси. Хмарна інфраструктура забезпечує автоматичний перезапуск бота у випадку помилок, підтримку збільшення масштабу, можливість приєднання webhook та стабільну роботу навіть за великих навантажень.

Heroku тривалий час був популярним серед розробників, однак після введення обмежень для безкоштовних тарифів менш придатний для постійних проєктів. Натомість Render і Railway надають гнучкі плани та простоту розміщення, що робить їх актуальними для Telegram-ботів середнього рівня складності. Для більш професійних рішень можна

використовувати Amazon Web Services, де EC2 та RDS забезпечують максимально високу надійність роботи системи.

Хмарні сервіси також дають змогу підключати інструменти моніторингу, як-от Grafana, Prometheus чи системи ведення журналів, що важливо для підтримки стабільної роботи Telegram-бота. Застосування webhook, яке можливе лише на сервері з відкритим HTTPS-доступом, є більш дієвим рішенням порівнюючи з polling, особливо за великого навантаження. Отже, хмарне середовище є обов’язковим для втілення сучасної системи організації спортивних заходів.

Вибір технологічного набору має враховувати вимоги до безпеки. Telegram-боти можуть обробляти персональні дані спортсменів — імена, контактні дані, приналежність до команди, що вимагає дотримання сучасних стандартів захисту інформації. Python та PostgreSQL дають можливість застосовувати токенізацію, шифрування, контрольований доступ і застосування захищених каналів зв’язку. Хмарні сервіси підтримують багаторівневу перевірку особи та контроль доступу до ресурсів.

Таким чином, вибір технологічного набору для Telegram-бота організації спортивних заходів має забезпечувати:

– стабільність роботи;

– асинхронну обробку запитів;

– можливість збільшення масштабу;

– підтримку приєднання до баз даних;

– простоту розміщення в хмарному середовищі;

– високий рівень безпеки;

– зручність розробки та супроводу.

Python, Aiogram, PostgreSQL та сучасні хмарні платформи відповідають цим вимогам повністю, що робить їх оптимальним вибором для створення Telegram-бота, здатного ефективно автоматизувати процеси організації спортивних заходів. Це формує базу для подальшого проєктування структури системи, що буде розглянуто в наступних розділах.

2.2 Порівняльний аналіз фреймворків Telegram Bot API для Python

Створення Telegram-ботів у Python може відбуватися за посередництвом різних бібліотек, які реалізують взаємодію з Telegram Bot API. Вибір конкретного фреймворку впливає на швидкість розробки, гнучкість структури, простоту підтримки програмного коду й можливість збільшення масштабу системи. У цьому розділі наведено порівняльний аналіз найпоширеніших Python-фреймворків: Aiogram, PyTelegramBotAPI (Telebot), python-telegram-bot, також менш поширених рішень, як-от AIOGramX та Telethon (частково застосовується для ботів, але частіше - для клієнтських програм).

Мета аналізу - визначити фреймворк, який найкраще відповідає вимогам системи організації спортивних заходів, що передбачає асинхронну обробку великої кількості запитів, побудову складних сценаріїв взаємодії з користувачами, приєднання до бази даних і можливість подальшого збільшення масштабу.

Aiogram - сучасний асинхронний фреймворк, написаний з використанням бібліотеки asyncio. Він надає розробникам високопродуктивні інструменти для побудови ботів будь-якої складності.

Основні переваги Aiogram:

- асинхронність, яка дає змогу обробляти сотні й тисячі запитів без блокування;
- структуровані обробники подій, що забезпечують чисту структуру й зручність читання коду;
- підтримка Finite State Machine (FSM) для реалізації складних діалогів;
- широкий набір інструментів для створення клавіатур, callback-запитів, інлайн-меню;
- підтримка middleware, що дає змогу створювати фільтри, логери та обробники помилок на глобальному рівні;

- гнучке приєднання до PostgreSQL та асинхронних бібліотек (asyncpg, aiohttp).

Aiogram є одним із найкращих рішень для створення систем, де потрібно збільшувати масштаб, оскільки він побудований за принципами асинхронного програмування. Це особливо важливо для Telegram-бота спортивних заходів, де події збирають велику кількість користувачів, які можуть одночасно взаємодіяти з ботом.

З-поміж недоліків можна виділити лише відносно вищий поріг входу через необхідність розуміння асинхронності, проте у професійних проєктах це швидше перевага, ніж обмеження.

PyTelegramBotAPI - простий і популярний синхронний фреймворк, що згодиться для невеликих або навчальних проєктів. Його головна перевага — простота застосування. Досить кількох рядків коду, щоб отримати працюючого бота.

Переваги Telebot:

- низький поріг входу;
- проста структура обробників;
- велика кількість прикладів у мережі;
- зручність швидкого створення прототипів.

Утім, для системи середньої чи високої складності Telebot має суттєві обмеження:

синхронність, яка не дає змоги ефективно обробляти велику кількість паралельних запитів;

- складність збільшення масштабу;
- слабша гнучкість побудови великих застосунків;
- відсутність повноцінної підтримки FSM;
- значно нижча швидкість роботи порівнюючи з Aiogram.

Отже, Telebot - не найкращий вибір для дипломного проєкту, оскільки організація спортивних заходів передбачає напружену роботу з користувачами.

python-telegram-bot

Один із найстаріших і найстабільніших фреймворків для Telegram-ботів у Python. Популярність бібліотеки пояснюється широкими функціями та повною підтримкою Telegram Bot API. У нових версіях python-telegram-bot також отримав асинхронний режим, що значно розширило його можливості.

Переваги python-telegram-bot:

- висока стабільність і якість документації;
- підтримка як синхронного, так і асинхронного режимів;
- великий функціонал для створення клавіатур, меню, webhook;
- зручність об'єднання з іншими Python-бібліотеками.

Недоліки:

- менш гнучкий FSM порівнюючи з Aiogram;
- складніша структура побудови логіки;
- менша продуктивність в асинхронних сценаріях.

Порівнюючи цей фреймворк з Aiogram, останній виявляється більш пристосованим для високих навантажень і збільшення масштабу систем.

Telethon більше повноцінний клієнт Telegram API, ніж бібліотека для ботів. Він дає можливість автоматизувати дії, доступні звичайному обліковому запису користувача. Telethon може взаємодіяти з групами, каналами, чатами, медіафайлами на низькому рівні.

Переваги Telethon:

доступ до повного Telegram API, не лише Bot API;

можливість створювати клієнтів, а не лише ботів;

асинхронність;

потужні можливості під час складних приєднань.

Недоліки:

складність застосування для ботів;

зайві функції для стандартних задач;

відсутність високорівневих інструментів для обробки команд.

Telethon - не найкращий вибір для дипломного проєкту, оскільки він надміру складний і не призначений саме для роботи з Bot API.

AIOGramX та інші нові фреймворки

AIOGramX - розширення Aiogram, яке додає кілька додаткових можливостей. Проте воно менш стабільне та має значно менше документації. У дипломних і реальних проєктах зазвичай радять використовувати класичну версію Aiogram, яку добре підтримують і регулярно оновлюють.

Фреймворк	Асинхронність	Швидкодія	Підтримка FSM	Придатність для масштабування	Зручність	Підходить для складних проєктів
Aiogram	Так	Висока	Дуже хороша	Висока	Висока	✓
python-telegram-bot	Частково	Середня	Середня	Середня	Висока	✓
Telebot	Ні	Низька	Низька	Низька	Дуже висока	✕
Telethon	Так	Висока	Низька	Висока	Середня	✕
AIOGramX	Так	Висока	Хороша	Середня	Середня	✓ (обмежено)

Таблиця 1.1- Порівняльна таблиця фреймворків

Висновок щодо вибору фреймворку

На основі проведеного аналізу можна зробити висновок, що Aiogram найкраще підходить для втілення Telegram-бота, націленого на організацію спортивних заходів. Його переваги повністю відповідають вимогам цього дипломного проєкту:

- асинхронність;
- можливість збільшення масштабу;
- велика кількість інструментів для побудови складної логіки;
- зручність приєднання до баз даних;
- можливість роботи в середовищах із високим навантаженням;

- чиста, модульна структура.

Застосування Aiogram дасть змогу створити систему, здатну забезпечити швидку, стабільну та зручну взаємодію з користувачами, що дуже важливо для цифрового середовища спортивних заходів.

2. 3. Структура програмного забезпечення Telegram-бота

Структура програмного забезпечення Telegram-бота визначає структуру системи, взаємозв’язки між її компонентами, принципи обробки даних і логіку взаємодії з користувачами. Продуктивність, можливість збільшення масштабу, надійність роботи та можливість подальшого розширення функціоналу залежать від того, наскільки правильно побудована структура. Telegram-бот, націлений на організацію спортивних заходів, має обробляти велику кількість запитів, забезпечувати швидкий доступ до інформації, підтримувати складні сценарії взаємодії та зберігати структуровані дані про користувачів і події. Тому структура має бути модульною, асинхронною та зрозумілою.

Загальна структура Telegram-бота може бути представлена у вигляді багаторівневої моделі, яка охоплює основні компоненти: шар представлення, шар бізнес-логіки, шар даних, інфраструктурний шар та інтеграційні компоненти. Кожен із цих елементів виконує свою роль, забезпечуючи структуровану роботу всього застосунку.

На верхньому рівні - шар представлення, який відповідає за взаємодію з користувачами через Telegram. До цього шару зараховують обробники команд, меню, клавіатури, інлайн-кнопки, повідомлення та інші інтерфейсні елементи. Мета цього шару — забезпечити зручний та інтуїтивний спосіб комунікації між користувачем і системою. У системі організації спортивних заходів через шар представлення реалізується відображення списку подій, реєстрація на турніри, отримання сповіщень, перегляд деталей змагань тощо.

Наступний елемент - шар бізнес-логіки, який охоплює головні алгоритми поведінки бота: обробку сценаріїв, перевірку правильності даних, визначення послідовності дій, управління станами (FSM) та виконання правил доступу. Цей шар формує логіку взаємодії між користувачами та системою, а також забезпечує виконання функцій. Наприклад, під час реєстрації користувача бізнес-логіка перевіряє, чи міститься такий

користувач у базі даних, чи дозволена реєстрація на конкретний захід, чи дотримано часових обмежень, і визначає подальший крок взаємодії.

Важливий елемент структури — шар даних, який охоплює базу даних, запити до неї, моделі сутностей і засоби доступу до інформації. Для Telegram-бота, який опрацьовує спортивні заходи, база даних містить такі сутності: користувачі, події, категорії, розклади, реєстрації, результати. На цьому рівні формується структура даних і забезпечується їхня цілісність, узгодженість і безпека. У випадку застосування PostgreSQL вся взаємодія з базою відбувається через асинхронний драйвер asyncpg, що підвищує швидкодію бота та дає можливість ефективно обробляти паралельні запити.

Інфраструктурний шар забезпечує технічний фундамент системи, наприклад, конфігурації приєднання, робота з API Telegram, налаштування webhook, ведення журналів, обробка помилок, кешування, захист від небажаних повідомлень і перевантажень. Сюди також входять компоненти, що відповідають за розміщення бота в хмарному середовищі та взаємодію з операційною системою сервера. Правильна побудова інфраструктурного шару гарантує стабільну роботу Telegram-бота.

Окрему роль відіграють інтеграційні компоненти, призначені для розширення функціоналу системи. До них можуть належати сервіси для розсилок, аналітики, формування PDF-звітів, приєднання до календарів, спортивних платформ або зовнішніх API. У спортивній сфері такі приєднання забезпечують автоматичне надсилання нагадувань, синхронізацію розкладів або обробку статистики.

Структуру Telegram-бота можна також описати у вигляді модульної структури, що охоплює:

Модуль обробки команд і повідомлень, який відповідає за взаємодію з Telegram API й обробку дій користувачів.

Модуль управління станами (FSM), який реалізує логіку діалогів, багатокрокові сценарії та збереження проміжних даних.

Модуль бізнес-логіки, який охоплює алгоритми реєстрації, створення заходів, формування розкладу, підтвердження участі, повідомлень тощо.

Модуль доступу до бази даних, який реалізує операції CRUD, транзакції та роботу з асинхронними запитами.

Модуль сповіщень, який здійснює планування нагадувань, розсилок, сповіщень про зміни у розкладі.

Модуль адміністрування, який відповідає за створення подій, модерацію, перевірку та керування користувачами.

Модуль інфраструктури, який охоплює конфігураційні файли, логери, middleware, обробники помилок.

Модуль приєднань, який дає змогу приєднувати зовнішні сервіси, хмарні розширення, платіжні системи (за потреби).

Кожен модуль виконує свою функцію, а завдяки асинхронності Aiogram вони працюють незалежно один від одного, і все працює без перерв.

Структура є масштабованою та дає змогу розширювати функціонал. У систему можна додати автоматичне формування турнірних сіток, приєднання до таблиць Google Sheets, можливість онлайн-реєстрації команд, персональні кабінети тощо — без значної зміни базових модулів.

Отже, побудована структура Telegram-бота забезпечує чіткий розподіл функцій між компонентами, високу продуктивність, модульність, простоту супроводу та стабільність роботи. Це створює основу для подальшої детальної розробки бази даних і логіки взаємодії, що розглядатиметься в наступних розділах.

2.4 Структура бази даних: об'єкти, зв'язки, ER-діаграми

При створенні Telegram-бота важливим етапом є розробка бази даних. Саме вона відповідає за збереження, цілісність і доступність інформації про спортивні заходи. Серверна частина Telegram-бота повинна швидко зчитувати та змінювати дані, обробляти запити від користувачів одночасно, а також гарантувати правильність інформації. Тому, при розробці структури даних потрібно враховувати особливості спортивної сфери, те, як користувачі взаємодіють з ботом, і можливість розширення системи.

Аналіз вимог показав, що база даних повинна містити інформацію про: користувачів, спортивні заходи, види спорту, розклад, реєстрацію учасників, ролі користувачів, а також результати змагань. Отже, база даних повинна бути логічно структурованою, щоб відображати ці взаємозв'язки та ефективно працювати з запитами.

Проєктування бази даних включає кілька етапів:

- визначення об'єктів;
- опис властивостей об'єктів;
- встановлення зв'язків між об'єктами;
- створення ER-моделі;
- приведення структури таблиць до нормальної форми.

Основні об'єкти бази даних

Виходячи з аналізу, визначено такі основні об'єкти:

1. Користувачі

Кожен користувач, який використовує бот, має унікальний ідентифікатор. Тут зберігається інформація про учасників, організаторів та адміністраторів.

Основні властивості:

- user_id (ідентифікатор користувача в Telegram);
- повне ім'я;
- ім'я користувача;
- номер телефону;

- роль (учасник, тренер, адміністратор);
- дата реєстрації.

2. Спортивні заходи

Тут описуються всі спортивні заходи, які створюють організатори.

Властивості:

- event_id;
- назва;
- опис;
- дата проведення;
- місце проведення;
- статус (активний, завершений, скасований).

3.Реєстрації

Тут зберігається інформація про зв'язок між користувачами та подіями. Це таблиця, яка показує, хто на який захід зареєструвався.

Властивості:

- registration_id;
- event_id;
- user_id;
- category_id;
- дата подачі заявки;
- статус (підтверджено/очікує/відхилено).

4. Результати

Тут міститься інформація про результати виступів учасників.

Властивості:

- result_id;
- event_id;
- user_id;
- category_id;
- місце;

- оцінка.

5.Адміністратори

Тут зберігається інформація про адміністраторів та їхні права.

Властивості:

- admin_id;
- user_id;
- рівень доступу.

Зв'язки між об'єктами:

Для правильної роботи бази даних, визначено такі типи зв'язків:

- Користувачі - Реєстрації: один користувач може бути зареєстрований на багато заходів (1→N).
- Події - Реєстрації: одна подія має багато реєстрацій (1→N).
- Категорії - Події: одна подія містить багато категорій (1→N).
- Реєстрації - Категорії: користувач може зареєструватися тільки в одну категорію на певну подію.
- Розклад - Категорії: кожна категорія може мати декілька блоків розкладу (1→N).
- Результати - Реєстрації: кожен результат відноситься до конкретної реєстрації.

Зв'язки встановлюються за допомогою первинних і зовнішніх ключів, що гарантує цілісність даних та правильність посилань.

ER-діаграма бази даних

Для створення моделі бази даних використовується ER-діаграма. Вона показує об'єкти, їх властивості та зв'язки між ними. Для Telegram-бота ER-модель виглядає так:

Users ← Registrations → Events

Events → Categories

Categories → Schedule

Registrations → Results

Ця структура показує, як елементи системи взаємодіють між собою: користувач взаємодіє з подією через реєстрацію; подія може містити різні категорії; результати базуються на реєстраціях; розклад показує деталі проведення заходу.

База даних відповідає третій нормальній формі (3NF). Це означає, що:

- немає повторень даних;
- структура логічна;
- можна додавати нові функції;
- легко створювати SQL-запити;
- менший ризик помилок при оновленні даних.

Розподіл інформації по окремим об'єктам зменшує дублювання та спрощує підтримку системи.

Переваги обраної моделі бази даних

Ця структура має такі переваги:

- можна розширювати - додавати нові об'єкти (наприклад, команди, судді, документи);
- гнучкість - підтримує будь-яку кількість спортивних подій;
- кожен модуль бота працює зі своїми об'єктами;
- швидкий доступ до PostgreSQL покращує роботу з великими обсягами даних;
- зручно аналізувати дані - структура дозволяє будувати статистику участі та результатів.

Розробка бази даних є важливим етапом у створенні Telegram-бота. Структура, що включає об'єкти «Користувачі», «Події», «Категорії», «Реєстрації», «Розклад» та «Результати», є надійною основою для реалізації всіх функцій системи. ER-діаграма та встановлені зв'язки гарантують цілісність даних, а вибір PostgreSQL забезпечує швидку роботу та можливість розширення. Це робить модель оптимальною для проєкту організації спортивних заходів.

2.5 Модуль взаємодії з користувачем: клавіатури, команди, логіка діалогів

Модуль взаємодії з користувачем є важливим компонентом Telegram-бота. Він визначає, як система спілкується з учасниками спортивних заходів. Його основна задача - створити простий та зрозумілий інтерфейс, за допомогою якого користувачі можуть легко знаходити інформацію, реєструватися на події, переглядати розклад або результати, а адміністратори - управляти заходами.

В основі взаємодії лежить Telegram Bot API, який дозволяє використовувати:

- команди;
- текстові та інлайн-клавіатури;
- callback-запити;
- меню та вкладені інтерфейси;
- стани (FSM) для створення діалогів.

Модуль взаємодії є однією з найважливіших частин Telegram-бота, оскільки він впливає на зручність використання, швидкість роботи та зрозумілість інтерфейсу.

Структура команд Telegram-бота

Команди - це основний спосіб доступу до функцій. Їх перелік залежить від ролі користувача та задач системи.

Команди для учасників

- /start - початок роботи, привітання, реєстрація в системі.
- /events - перегляд списку доступних спортивних заходів.
- /register - подача заявки на участь.
- /schedule - перегляд розкладу заходу.
- /my_events - список заходів, на які користувач зареєстрований.

Команди для адміністраторів

- /create_event - створення спортивного заходу.

- /manage_event - редагування параметрів події.
- /participants - перегляд списку учасників.

Команди допомагають швидко орієнтуватися в боті. Однак, зараз основний акцент робиться на інлайн-кнопки та меню, що робить інтерфейс більш зручним.

Механізм клавіатур у Telegram

Існує два типи клавіатур:

- Reply-клавіатури (кнопки внизу екрана).
- Inline-клавіатури (кнопки в повідомленнях з callback-даними).

Reply-клавіатури

Reply-клавіатури використовуються для простих дій, наприклад:

- повернення в головне меню;
- вибір категорії;
- підтвердження або відхилення дії.

Вони прості та інтуїтивні. Але, займають багато місця на екрані та не підходять для складних меню.

Inline-клавіатури

Інлайн-кнопки дозволяють створювати багаторівневі меню, які не займають багато місця. Вони є основним інструментом для організації спортивних заходів.

Приклади використання:

- вибір події зі списку;
- підтвердження реєстрації;
- перегляд інформації про захід;
- навігація по категоріях;
- адміністрування (редагування, видалення, створення).

Inline-клавіатури працюють через callback-запити. Це дозволяє надсилати дані у відповідь на натискання кнопки та викликати потрібні функції.

Callback-логіка

Callback-запити важливі для створення інтерактивного інтерфейсу. Коли користувач натискає кнопку, бот отримує callback_data та виконує певні дії.

Приклад:

1. Користувач натискає кнопку Переглянути подію.
2. Бот отримує callback_data = event_12.
3. Модуль визначає, що це подія з ID = 12, та викликає потрібну функцію.
4. Бот надсилає відповідь.

У Telegram-боті callback-запити використовуються для:

- навігації по меню;
- вибору категорій змагань;
- підтвердження реєстрації;
- переходів між розділами;
- відображення результатів або розкладу.

Логіка діалогів і FSM (машина станів)

Реєстрація на захід може включати декілька етапів: вибір події ; категорії ; підтвердження ; введення даних. Для цього використовується FSM (Finite State Machine).

FSM дозволяє:

- вести користувача через чітко визначені етапи;
- зберігати дані;
- перевіряти введену інформацію;
- обробляти помилки;
- створювати складні сценарії.

Приклади станів FSM:

RegistrationState

- вибір події
- вибір категорії

- підтвердження

EventCreationState

- введення назви
- введення опису
- вибір дати
- вибір місця проведення

NotificationState

- введення тексту
- підтвердження розсилки

FSM особливо важлива для систем спортивних заходів, де багато сценаріїв, пов'язаних з реєстрацією, адмініструванням та обробкою інформації про змагання.

Модуль меню та навігації

Для простоти використання, бот використовує багаторівневе меню:

Головне меню

- Переглянути події
- Мої реєстрації
- Розклад
- Профіль

Адмін-меню

- Створити подію
- Керувати подією
- Переглянути учасників
- Надіслати повідомлення

Меню змінюється в залежності від ролі користувача та доступних функцій.

Принципи UX-дизайну у Telegram-боті

При створенні взаємодії використовуються такі принципи:

- Мінімум дій: користувач повинен досягати цілі за 2–3 натискання.
- Логічність: перехід між етапами завжди зрозумілий.
- Інформативність: бот повинен пояснювати, що відбувається на кожному кроці.
- Контекстність: наприклад, якщо подія завершена — кнопки реєстрації приховані.
- Єдиний стиль: форматування, емоджі, розділи — все повинно бути однаковим.

Модуль взаємодії з користувачем є дуже важливим елементом Telegram-бота. Він забезпечує:

- швидкий доступ до функцій;
- зменшення кількості помилок;
- зручну навігацію;
- ефективну роботу через callback-логіку та інлайн-кнопки;
- підтримку складних діалогів завдяки FSM;
- інтерфейс, який змінюється в залежності від ролі користувача.

Отже, добре продуманий інтерфейс дозволяє створити Telegram-бота, який стане зручним інструментом для організації спортивних заходів, забезпечуючи позитивний досвід користувача та автоматизацію процесів.

2.6 FSM (Finite State Machine) у проєктуванні сценаріїв взаємодії

Використання машини станів (FSM) у розробці Telegram-бота дозволяє створювати структуровані та логічні діалоги з користувачами. Це особливо важливо для систем, які потребують збору даних, наприклад, у системах організації спортивних заходів, де користувач може проходити реєстрацію, обирати категорії, підтверджувати участь або вводити додаткову інформацію.

FSM дозволяє розділити взаємодію між користувачем та ботом на окремі стани, кожен з яких відповідає певному етапу процесу. Кожен стан обробляється окремо, що дозволяє контролювати дії користувача, обробляти лише потрібні дані та уникати помилок.

Переваги використання FSM у Telegram-ботах

FSM ефективна завдяки:

1. Простоті сценаріїв

FSM дозволяє розбити складний процес на прості етапи. Це робить код більш зрозумілим.

2. Контролю вводу даних

У кожному стані дозволяється тільки певний тип введення, тому бот уникає введення неправильної інформації.

3. Спрощенню діалогів

У Telegram-ботах часто потрібна багатокрокова взаємодія, наприклад, створення подій, заповнення заявок. FSM дозволяє впорядкувати такі сценарії.

4. Підтримці асинхронності

Aiogram підтримує FSM і дозволяє створювати асинхронні діалоги, які можуть виконуватися одночасно для багатьох користувачів.

5. Масштабуванню

FSM можна легко розширити, додавати нові стани або перевірки даних.

Роль FSM у Telegram-боті для спортивних заходів

Система організації спортивних заходів містить різні групи користувачів:

- учасників;
- тренерів;
- організаторів;
- адміністраторів.

Для кожної групи передбачено різні сценарії, що включають декілька станів. FSM забезпечує логічну послідовність цих процесів.

Приклади використання FSM:

1. Реєстрація користувача на спортивний захід

Процес складається з таких станів:

- RegistrationState.choose_event - вибір події зі списку;
- RegistrationState.choose_category - вибір категорії;
- RegistrationState.confirm - підтвердження заявки;
- RegistrationState.finished - завершення та запис даних у базу.

2. Створення спортивного заходу (адмін-сценарій)

- EventCreationState.title - введення назви;
- EventCreationState.description - додавання опису;
- EventCreationState.date - вибір дати;
- EventCreationState.location - введення місця проведення;
- EventCreationState.categories - додавання категорій;
- EventCreationState.confirmation - підтвердження створення.

3. Розсилка повідомлень

- NotificationState.input_text - введення тексту;
- NotificationState.select_target - вибір групи отримувачів;
- NotificationState.confirm - підтвердження розсилки.

Структура FSM у Aiogram

Aiogram підтримує FSM завдяки класам StatesGroup та State. Це дозволяє:

- створити групу станів;

- використовувати їх у обробниках;
- зберігати дані у FSMContext.

FSM у Aiogram працює так:

1. Увійти в стан за допомогою await state.set_state(...).
2. Обробити дані.
3. Перейти до наступного стану.
4. Завершити FSM через await state.clear() після завершення процесу.

Кожен користувач може мати свій набір станів, що не впливають один на одного.

Підхід до створення сценаріїв

Проєктування FSM відбувається у декілька етапів:

1. Аналіз вимог: визначаються всі процеси, що виконуються користувачами.
2. Розбиття процесів на етапи: кожний етап стає окремим станом FSM.
3. Визначення переходів між станами: встановлюється логіка переходів.
4. Обробка помилок: продумуються сценарії неправильного введення.
5. Створення фінального стану: всі дані об’єднуються й записуються в базу.
6. Узгодження FSM з бізнес-логікою: FSM викликає функції відповідно до стану.

Взаємодія FSM із базою даних

У кожному стані бот виконує лише ті операції, які потрібні на поточному етапі. Наприклад:

- у стані choose_event бот зчитує перелік подій;
- у choose_category - передає ID події та читає список категорій;

- у стані confirm - перевіряє, чи користувач не подав заявку раніше.

Це зменшує навантаження на базу даних та кількість помилок.

Взаємодія FSM із модулем клавіатур

FSM пов’язана з інтерфейсом:

- У кожному стані користувачеві може подаватися нове меню.
- Інлайн-кнопки змінюються відповідно до етапу.
- Доступні кнопки залежать від попередньо вибраних значень.

Переваги FSM у спортивному боті

Для системи спортивної організації FSM забезпечує:

- Контроль бізнес-процесів: неможливо перескочити стан або виконати неправильну команду.
- Стабільність: сценарії легко тестувати.
- Гнучкість: легко додавати нові стани.
- Масштабованість: кожен користувач працює у власному наборі станів.

Використання FSM у Telegram-боті необхідне для створення складних сценаріїв взаємодії, роблячи їх передбачуваними та керованими. Для системи організації спортивних заходів FSM забезпечує зручність та стабільність роботи.

2.7 UML-діаграми Telegram-бота (Use Case, Activity, Sequence)

UML-діаграми допомагають моделювати структуру та поведінку програмних систем. Використання UML дозволяє описати процеси, що відбуваються у Telegram-боті, та зробити їх зрозумілими на етапі створення. Telegram-бот для організації спортивних заходів включає різні сценарії взаємодії, тому UML дозволяє наочно показати ці процеси.

У проєкті розроблено три діаграми:

- діаграма варіантів використання (Use Case): описує ролі користувачів і функції, доступні їм;
- діаграма діяльності (Activity): показує логіку основних процесів у Telegram-боті;
- діаграма взаємодії (Sequence): відображає послідовність повідомлень між компонентами системи.

UML Діаграма варіантів використання (Use Case)

Діаграма варіантів використання показує взаємодію між користувачами та системою. У проєкті є три типи користувачів:

- звичайний користувач (учасник);
- адміністратор / організатор заходу;
- Telegram API (зовнішній сервіс).

Кожен користувач має набір доступних функцій.

Функції для учасника:

- перегляд доступних спортивних подій;
- перегляд інформації про подію;
- реєстрація на захід;
- вибір категорії;
- перегляд власних реєстрацій;
- перегляд розкладу події;
- отримання сповіщень.

Функції для адміністратора:

- створення заходу;
- редагування параметрів;
- додавання категорій;
- перегляд списку учасників;
- підтвердження заявок;
- надсилання сповіщень;

Функції Telegram API:

- передача повідомлень від користувачів;
- доставка відповідей від бота;
- обробка callback-запитів.

Діаграма варіантів використання допомагає визначити вимоги до Telegram-бота та описує всі сценарії взаємодії з системою.

UML Діаграма діяльності (Activity)

Діаграма діяльності описує логіку виконання процесів у системі.

У проєкті розроблено діаграму, яка показує процес реєстрації на спортивний захід.

Ключові етапи діаграми:

- Початок: користувач вибирає опцію Реєстрація.
- Вибір спортивного заходу.
- Система надсилає список доступних подій.
- Якщо подій немає - сценарій завершується.
- Вибір категорії.
- після вибору події система надсилає перелік доступних категорій.
- Підтвердження участі.
- Користувач підтверджує реєстрацію, після чого система перевіряє, чи не зареєстрований він раніше.
- Запис у базу даних: система створює запис у таблиці Registrations.

- Повідомлення користувача: бот надсилає підтвердження реєстрації.
- Завершення процесу.

Діаграма показує логіку та послідовність дій, що гарантує правильну реалізацію функцій.

UML Діаграма послідовності (Sequence)

Діаграма послідовності показує взаємодію між компонентами системи в часі. Вона описує процес обробки callback-запиту Зареєструватися на подію.

У сценарії беруть участь такі об'єкти:

- Користувач
- Telegram API
- Бот (обробник Aiogram)
- Модуль FSM
- Модуль доступу до бази даних
- База даних (PostgreSQL)

Основна послідовність дій:

1. Користувач натискає кнопку Реєстрація.
2. Telegram API надсилає callback-запит боту.
3. Обробник Aiogram визначає команду та передає управління FSM.
4. FSM переходить у відповідний стан (наприклад, choose_category).
5. FSM надсилає користувачеві наступне меню.
6. Після вибору категорії FSM викликає модуль бази даних.
7. Модуль БД виконує SQL-запит на додавання запису.
8. PostgreSQL повертає підтвердження виконання запиту.
9. Бот надсилає користувачеві повідомлення про успішну реєстрацію.
10. Діаграма послідовності показує логіку системи та порядок виклику методів.

Значення UML-діаграм у проєкті

UML забезпечує:

- розуміння архітектури системи;
- зменшення помилок при створенні логіки;
- спрощення обговорення проєкту між розробниками;
- спрощення документування роботи.

Діаграми допомагають формалізувати вимоги, структурувати логіку та створити якісні сценарії.

Створення UML-діаграм є важливим етапом проєктування Telegram-бота. Діаграма варіантів використання допомагає визначити ролі користувачів та функціонал для кожної ролі. Діаграма діяльності описує основні бізнес-процеси, такі як реєстрація на спортивний захід. Діаграма послідовності розкриває внутрішню логіку взаємодії між компонентами системи.

2.8 Безпека Telegram-ботів та захист персональних даних

Безпека Telegram-бота є важливим аспектом, оскільки система обробляє персональні дані користувачів. Бот працює з іменами, контактами, інформацією про участь у змаганнях, що вважається персональними даними згідно із законодавством України та міжнародними нормами (GDPR, ISO/IEC 27001). Тому, при розробці Telegram-бота важливо забезпечити захист даних і запобігти можливим загрозам.

Telegram має вбудовані механізми шифрування даних, які забезпечують безпечний зв’язок між користувачем і сервером Telegram. Але, розробник повинен реалізовувати додаткові заходи безпеки на стороні бота та серверної частини. Вразливості в обробці даних, зберіганні інформації та доступі до інтерфейсів керування, можуть стати мішенню для кібератак.

Основні загрози безпеці Telegram-ботів

Основними загрозами є:

1. Компрометація токена бота: токен дозволяє боту працювати з Telegram API. Якщо зловмисник отримає доступ до токена, він може керувати ботом.

Приклади загроз:

- підміна відповідей;
- надсилання фішингових повідомлень;
- блокування роботи бота;
- витік даних користувачів.

2. Несанкціонований доступ до бази даних.

Якщо дані для доступу до бази даних не захищені:

- витік персональних даних;
- зміна або видалення інформації;
- фальсифікація результатів змагань.

3. Атаки типу SQL Injection

Зловмисник може виконати небажані SQL-запити, якщо система не перевіряє вхідні дані.

4. Атаки типу Flood та DDoS

Бот може отримувати багато запитів, що призведе до перевантаження системи.

5. Фішингові атаки

Зловмисники можуть імітувати меню бота або надсилати фальшиві повідомлення.

6. Неправильне зберігання даних

Дані користувачів можуть зберігатися у відкритому вигляді в логах сервера.

Принципи забезпечення безпеки при розробці Telegram-бота

Застосовуються такі підходи до безпеки:

1. Захист токена:

- не зберігати токен у коді;
- використовувати .env-файли та системні змінні;
- обмежувати доступ до файлів;
- змінювати токен при підозрі компрометації.

2. Використання безпечних протоколів:

- використання SSL-сертифікатів;
- використання TLS 1.2;
- заборона HTTP-з’єднань.

3. Захист бази даних:

- використання складного пароля;
- доступ до бази даних лише з локального сервера або визначених IP-адрес;
- шифрування з’єднання (sslmode=require);
- резервне копіювання;
- розподіл прав між ролями (адміністратор, бот, модератор);

- обмеження відкритих портів.

4. Перевірка введених даних:

Для запобігання SQL Injection та XSS необхідно використовувати:

- параметризовані SQL-запити;
- фільтри вводу у Aiogram;
- перевірку форматів даних.

5. Обмеження прав доступу у Telegram:

- перевірка user_id у білому списку;
- прив’язка ролей до таблиці Users;
- приховання окремих команд для звичайних користувачів.

6. Захист від флуду та атак:

- обмеження частоти запитів;
- блокування підозрілих акаунтів;
- логування активності;
- кешування відповідей.

7. Логування:

Система повинна записувати:

- помилки в роботі бота;
- спроби доступу;
- дії адміністраторів;
- швидкість відповіді системи.

8. Анонімізація даних:

Для зберігання історичних даних про статистику можна використовувати методи анонімізації.

Дотримання законодавства щодо персональних даних

Telegram-боти повинні відповідати:

- Закону України «Про захист персональних даних»;
- принципам GDPR;
- міжнародним рекомендаціям NIST та ISO.

Основні вимоги:

- зберігати лише необхідні дані;
- обмежувати доступ до персональної інформації;
- забезпечувати можливість видалення даних на запит;
- вести облік операцій.

Забезпечення безпеки Telegram-бота є важливим процесом, що охоплює:

- захист токенів;
- бази даних;
- механізмів введення;
- ведення логів;
- обмеження прав доступу.

2.9 Продуктивність і розширюваність системи

У наш час продуктивність і здатність до розширення є найважливішими якостями Telegram-ботів, особливо коли йдеться про обслуговування великої кількості користувачів, організацію масштабних спортивних заходів та обробку великих обсягів інформації. Тому Telegram-бот для спортивних подій має швидко реагувати на запити, працювати стабільно за умов підвищених навантажень, і його можна легко розширити без змін в основній структурі. Все це впливає на те, як буде реалізовано програмне забезпечення, структуру бази даних, серверне середовище та інтеграцію з Telegram API.

Швидкість роботи системи

Під продуктивністю Telegram-бота мається на увазі його здатність швидко обробляти запити клієнтів, вчасно реагувати на них і стабільно працювати при великих обсягах інформації. Для нашої системи важливо забезпечити:

Час відповіді бота

Telegram радить, щоб бот відповідав не довше ніж за 1 секунду.

Це особливо важливо для:

- обробки callback-запитів (інлайн-кнопок);
- навігації по меню;
- реєстрації на спортивні змагання;
- перевірки статусу учасника;
- роботи FSM (кінцевого автомата).

Для цього ми передбачили:

- використання асинхронного фреймворку Aiogram 3.x, який дозволяє обробляти багато з'єднань одночасно;
- оптимізацію запитів до бази даних;
- збереження часто використовуваних даних у кеші (наприклад, переліку доступних подій).

Покращення бази даних

Найчастіше боти працюють повільно не через код Python, а через проблеми з базою даних. Тому в проєкті ми:

- створюємо індекси для полів, які часто використовуються в SELECT-запитах;
- розділяємо таблиці за призначенням (події, категорії, реєстрації, користувачі);
- використовуємо пул з'єднань для asyncpg;
- намагаємося уникати довгих операцій в процесі FSM.

Це дає змогу обробляти сотні запитів за секунду навіть на недорогому сервері.

Асинхронна структура

Aiogram працює на базі asyncio, що забезпечує:

- миттєву обробку подій;
- одночасну роботу з великою кількістю користувачів;
- швидкий обмін даними з Telegram API.

Завдяки цьому система може одночасно взаємодіяти з сотнями користувачів, наприклад, під час реєстрації на спортивні події.

Здатність системи до розширення

Розширюваність - це здатність системи збільшувати свою продуктивність при зростанні навантаження без значних змін в її структурі. Telegram-бот, який працює зі спортивними заходами, може зіткнутися з великим навантаженням у таких випадках:

- під час масової реєстрації перед змаганнями;
- коли потрібно одночасно розіслати повідомлення великій кількості учасників;
- під час обробки статистики турнірів;
- коли часто звертаються до списків подій.

Тому структуру потрібно будувати з можливістю як вертикального, так і горизонтального розширення.

Вертикальне розширення

Вертикальне розширення передбачає збільшення обчислювальних ресурсів сервера:

- CPU (кількість ядер);
- RAM;
- SSD;
- швидкість мережі.

Горизонтальне розширення

Telegram дозволяє використовувати лише один webhook на бота, тому кластеризація працює інакше:

- створюються окремі обробники для черг повідомлень;
- складні операції переносяться у фонові задачі (celery, aiogram background tasks);
- використовується Redis для кешування і як брокер повідомлень;
- логіка розділяється на декілька мікросервісів (наприклад, основний бот + модуль аналітики).

Завдяки такому підходу можна обробляти значно більші потоки даних, ніж у звичайному Telegram-боті.

Розширення бази даних

- розподіл даних по різних базах даних (шардінг) – актуально для 20–50 тис. записів;
- реплікація – створення копій бази даних для читання;
- кешування популярних даних (наприклад, списків подій, майбутніх матчів);
- архівування застарілих записів(статистика турнірів, результати).

Для поточного проєкту достатньо реплікації для читання та індексації основних полів.

Обробка великої кількості повідомлень

Під час масових заходів система може надсилати:

- тисячі нагадувань;
- списки учасників;
- результати ігор.

Telegram має обмеження:

- не більше 30 повідомлень за секунду для одного бота.

Щоб це обійти:

- використовується черга повідомлень (Redis + Celery/Arq);
- відправлення повідомлень відкладається;
- сповіщення групуються (одне велике повідомлення замість багатьох маленьких).

Це забезпечує стабільну роботу навіть при великих навантаженнях, як у великих спортивних організаціях.

Надійність і стійкість до відмов

Telegram-бот повинен працювати без зупинок, особливо під час спортивних подій, коли важлива кожна хвилина.

Основні вимоги:

- автоматичний перезапуск сервісу (systemd, Docker restart policy);
- регулярне створення резервних копій бази даних;
- постійний контроль за роботою (Grafana, Prometheus);
- обмеження використання ресурсів через Docker-контейнери;
- запуск бота в кластері (якщо можливо).

Перевірка продуктивності

Перед запуском системи необхідно провести:

- Стрес-тестування: перевірка роботи при максимальному навантаженні.
- Тестування під час реальних подій: аналіз поведінки бота у пікові періоди.
- Перевірка роботи FSM під час великої кількості одночасних діалогів.

- Перевірка SQL-запитів на швидкість.

Результати тестування допомагають виявити слабкі місця та покращити роботу системи.

Отже, Telegram-бот для організації спортивних заходів повинен обслуговувати велику кількість користувачів, швидко реагувати на запити та стабільно працювати під час великих навантажень. Для цього ми використовуємо асинхронну архітектуру на базі Aiogram, оптимізуємо SQL-запити, використовуємо кешування, проводимо тестування продуктивності та забезпечуємо можливість горизонтального та вертикального розширення. Це дозволить створити гнучку, надійну та продуктивну систему, готову до зростання.

2.10 Утотожнення до розділу 2

У другому розділі ми детально розглянули структуру, технології та засоби, необхідні для створення ефективного Telegram-бота для організації спортивних заходів. Дослідження допомогло нам отримати чітке уявлення про основні технічні, функціональні аспекти та аспекти безпеки, які впливають на якість і надійність програмного продукту.

Перш за все, ми вивчили особливості архітектури Telegram API, яка забезпечує обмін даними між користувачем і сервером. З'ясувалося, що основні способи взаємодії - webhook і long polling - мають різні показники продуктивності. Для створення сервісу, який можна масштабувати, краще використовувати архітектуру з webhook, яка забезпечує швидкий час реакції, стабільність і можливість інтеграції з сучасними серверними технологіями.

У цьому розділі ми дослідили можливості фреймворку Aiogram, одного з кращих інструментів для розробки Telegram-ботів. Асинхронна модель обробки подій, система FSM (кінцевий автомат), підтримка callback-запитів і модульна структура коду дозволяють створювати гнучкі, масштабовані та швидкі рішення. Застосування Aiogram допомагає обробляти запити користувачів швидше та покращує взаємодію з базою даних.

Особливу увагу ми приділили розробці бази даних для системи реєстрації на спортивні заходи. Створена схема на основі PostgreSQL забезпечує цілісність даних, швидкий доступ до великих обсягів інформації та оптимізує операції читання і запису. Ми визначили основні сутності – події, категорії, користувачі, реєстрації, а також розглянули принципи нормалізації, індексування та оптимізації SQL-запитів.

Важливим етапом стало моделювання роботи Telegram-бота за допомогою UML-діаграм. Діаграма варіантів використання допомогла нам описати взаємодію між користувачами, адміністраторами та системою. Діаграма діяльності показала послідовність дій при реєстрації на подію, а діаграма послідовностей наочно продемонструвала взаємодію між ботом, FSM і базою даних.

У розділі ми також звернули увагу на питання безпеки. Ми проаналізували основні загрози для Telegram-ботів, такі як крадіжка токена, SQL-ін'єкції, DDoS-атаки, несанкціонований доступ до бази даних та фішингові атаки. На основі цього ми визначили перелік заходів безпеки, які включають захист токенів, використання HTTPS, перевірку даних, розмежування прав доступу, кешування, логування та моніторинг роботи системи. Дотримання цих рекомендацій забезпечує захист особистих даних і стабільність роботи системи.

Окремо ми проаналізували вимоги до продуктивності та можливості розширення. З'ясувалося, що використання асинхронної архітектури, оптимізація бази даних, кешування часто використовуваних даних і можливість горизонтального та вертикального розширення дозволяють системі обслуговувати багато користувачів і витримувати пікові навантаження під час масових спортивних подій.

В результаті проведеного аналізу ми отримали повне технічне обґрунтування для реалізації Telegram-бота. Розділ 2 створює надійну основу для переходу до практичної частини проєкту, яка передбачатиме розробку функціоналу, написання коду, тестування системи та аналіз результатів. Отримані результати забезпечують теоретичну і технологічну базу для успішної реалізації програмного продукту.

РОЗДІЛ 3.
ПРАКТИЧНА РЕАЛІЗАЦІЯ TELEGRAM-БОТА ДЛЯ ОРГАНІЗАЦІЇ СПОРТИВНИХ ЗАХОДІВ

3.1 Постановка завдання для програмної реалізації

У цій дипломній роботі потрібно зробити програму, яка допомагатиме організовувати, планувати та проводити спортивні заходи через телеграм-бота. Це важливо, тому що спортивні події для аматорів і напівпрофесіоналів стають дедалі популярнішими. З ростом таких заходів потрібен інструмент, який дозволить організаторам добре керувати інформацією, стежити за реєстрацією учасників, створювати розклад, повідомляти про зміни та спілкуватися з учасниками. Зазвичай це роблять вручну через месенджери, групові чати, таблиці та особисті повідомлення, що займає багато часу і може призвести до помилок. Телеграм-бот може автоматизувати ці процеси, зменшити вплив людського фактору, зробити обмін інформацією точнішим і забезпечити зрозуміле управління спортивними заходами.

Завдання передбачає визначення, що саме має робити програма, які в неї будуть технічні вимоги та як вона буде працювати. Також потрібно проаналізувати, як її можна буде використовувати в реальній роботі спортивних секцій і груп, і визначити основні принципи, на яких базуватиметься розробка. Головна ідея проєкту – створити систему, де різні користувачі (адміністратори, організатори та учасники) можуть взаємодіяти між собою. Кожен рівень користувачів повинен мати різні права, команди та можливості, які визначатимуть, що вони можуть робити в боті. Організатори зможуть швидко створювати та змінювати події, закривати реєстрацію, переглядати списки учасників і спілкуватися з ними через автоматичні повідомлення. Адміністратор матиме більше можливостей для контролю над системою: призначати ролі, керувати користувачами, створювати резервні копії даних, переглядати статистику та стежити за роботою системи.

Учасники зможуть легко переглядати актуальні заходи, реєструватися на них, отримувати інформацію, переглядати свій профіль та історію участі.

Програма повинна дозволяти створювати спортивні події з основними параметрами: назва, опис, місце, дата та час, ліміт учасників, організатор і, за потреби, додаткові параметри. Важливо, щоб було зручно керувати участю: бот повинен автоматично вести облік вільних місць, не дозволяти перевищувати ліміт і повідомляти користувачам, якщо реєстрація неможлива. Також потрібен механізм скасування участі, який дозволить організатору оновлювати списки та відкривати місця для інших користувачів. Реєстрація та скасування повинні супроводжуватися повідомленнями: бот автоматично надсилатиме повідомлення про зміни статусу або важливі оновлення.

Потрібно розробити зручний інтерфейс для спілкування з ботом. Для цього потрібно використовувати інлайн-кнопки, клавіатури, систему callback-запитів і відповідні меню. Користувач повинен мати можливість легко переходити між розділами, вибирати події, переглядати інформацію та швидко реєструватися. Інтерфейс повинен бути компактним, зрозумілим і пристосованим для мобільних пристроїв, оскільки Telegram найчастіше використовують саме на них. Також потрібно забезпечити обробку помилкових дій користувача, надавати підказки та можливість повернутися до попередніх меню.

Для правильної роботи системи потрібно створити серверну частину бота, яка оброблятиме команди, керуватиме логікою, взаємодіятиме з базою даних і зберігатиме інформацію. У проєкті планується використовувати мову програмування Python і фреймворк python-telegram-bot версії 20+, який забезпечує швидку обробку повідомлень, гнучку структуру логіки та широкі можливості для розширення. Швидка обробка важлива, оскільки бот повинен швидко реагувати та обробляти багато запитів одночасно. База даних повинна мати таблиці користувачів, подій, ролей, реєстрацій та логів. З огляду на особливості телеграм-бота, краще використовувати SQLite як просту та

надійну локальну базу даних. У майбутньому можна буде перейти на PostgreSQL або іншу повноцінну систему керування базами даних.

Важливо визначити вимоги до безпеки та захисту даних. Оскільки бот працює з особистими даними користувачів (Telegram-ID, ім’я, контакти), потрібно їх захистити та обмежити доступ до адміністративних функцій. Система ролей повинна бути розроблена так, щоб користувачі без дозволу не могли отримати доступ до функцій організатора або адміністратора. Також потрібно передбачити авторизацію через пароль або підтвердження від адміністратора. Захист від зловживань включає обмеження на часті запити, контроль повідомлень і обробку недійсних даних.

З точки зору організації, потрібно розробити систему, яку можна використовувати у спортивних клубах, школах, секціях та інших організаціях, що проводять спортивні заходи. Бот повинен підтримувати декілька організаторів одночасно, щоб вони могли незалежно керувати своїми подіями. Це дозволить створити єдину платформу для спортивної активності, де кожен організатор може вести свій календар подій, а користувачі можуть зручно стежити за оновленнями та брати участь у заходах. Такий підхід сприяє залученню аудиторії та покращує організацію. Отже, завданням є створення багатофункціонального телеграм-бота, який автоматизує ключові процеси організації спор тивних заходів. Бот повинен мати модулі для управління подіями, спілкування з користувачами, систему ролей, внутрішню статистику, базу даних та інтерфейс. Система повинна працювати стабільно, забезпечувати зручний доступ до інформації, швидку взаємодію та надійність. Для цього потрібно використовувати сучасні технології, гнучкий підхід до структури та враховувати потреби організаторів і учасників спортивних заходів.

3.2 Обгрунтування вибору інструментів та бібліотек

Для створення телеграм-бота, який допомагатиме організовувати спортивні заходи, потрібно ретельно вибрати інструменти, які забезпечать стабільну роботу системи, можливість її розширення, зручність у використанні та подальшому вдосконаленні. Вибір технологій залежить від того, що має робити проєкт, і від того, де він буде працювати. Основними критеріями вибору інструментів є надійність, продуктивність, простота використання, популярність серед розробників, наявність документації та сумісність між компонентами системи. Тому потрібно детально пояснити, чому було обрано кожен інструмент.

Для серверної частини телеграм-бота обрано мову програмування Python, оскільки вона є однією з найпопулярніших і найзручніших для розробки автоматизованих систем, зокрема ботів для месенджерів. Python має багато переваг, які роблять її придатною для цих завдань. По-перше, мова дозволяє використовувати асинхронне програмування, що дає змогу обробляти багато запитів одночасно без значного збільшення навантаження. Це важливо для телеграм-бота, оскільки користувачі можуть взаємодіяти з системою одночасно, і відповіді мають оброблятися швидко. По-друге, Python має простий синтаксис, що прискорює розробку та зменшує ймовірність помилок. По-третє, є багато якісних бібліотек для інтеграції, роботи з базами даних, обробки API та різних алгоритмів. Крім того, велика спільнота розробників забезпечує доступ до оновлень, прикладів і технічної підтримки.

Для роботи з Telegram API обрано бібліотеку python-telegram-bot версії 20+, оскільки вона є однією з найбільш стабільних і функціональних для створення ботів. Цей фреймворк підтримує асинхронний режим роботи, має модульну структуру та надає багато готових інструментів, які спрощують роботу розробника. Бібліотека відповідає офіційній специфікації Telegram Bot API, що гарантує правильну обробку подій, оновлень і callback-запитів. Вона містить класи для роботи з інлайн-кнопками, меню, повідомленнями, медіа,

webhooks та іншими елементами, що дозволяє створювати складні інтерфейси без зайвого коду. Python-telegram-bot активно підтримується, регулярно оновлюється та має детальну документацію, що важливо для систем із тривалим терміном служби.

Особливу увагу приділено вибору системи керування базами даних. Вирішено використовувати SQLite як основну СКБД. Бот не потребує великих обсягів транзакцій і розподіленої мережевої структури, натомість йому потрібна легка, стабільна та невибаглива база даних, яка працюватиме локально та забезпечуватиме швидкий доступ. SQLite зберігає базу у вигляді одного файлу, що спрощує її розгортання та не потребує налаштування серверної інфраструктури. Це оптимальне рішення, оскільки дозволяє зосередитися на логіці бота без потреби у додаткових сервісах. SQLite підтримує транзакції, індекси, зв'язки та цілісність даних, що робить її повноцінною СКБД для невеликих і середніх проєктів. У майбутньому базу можна буде легко перенести на PostgreSQL або MySQL, зберігаючи структуру таблиць завдяки використанню ORM.

Для роботи з базою даних використано ORM-бібліотеку SQLAlchemy, яка дозволяє працювати з даними на рівні об’єктів, а не через SQL-запити. Це робить код зрозумілішим і структурованішим, спрощує підтримку системи, підвищує її безпеку та дозволяє уникнути помилок, пов’язаних з некоректними SQL-запитами. SQLAlchemy підтримує перевірку даних, автоматичне створення таблиць, роботу з сесіями, зв’язками та забезпечує переносимість коду між різними СКБД. У проєкті ORM є проміжним шаром між логікою бота та зберіганням інформації, забезпечуючи зручну взаємодію з таблицями користувачів, подій, реєстрацій і ролей.

Важливим елементом є система ролей і доступу. Інформація про ролі зберігається в базі даних, а перевірка прав доступу виконується за допомогою функцій, інтегрованих у логіку бота. Це дозволяє легко керувати повноваженнями, забезпечувати гнучкість і гарантувати, що організатори та адміністратори матимуть доступ лише до відповідного функціоналу. Нема

потреби використовувати складні бібліотеки, оскільки система не потребує багатофакторної автентифікації чи інтеграції з сервісами. Власна система ролей дає змогу повністю контролювати логіку доступу та адаптувати її до спортивних заходів.

Для структурування проєкту використано модульну організацію коду. Кожен логічний блок системи (обробка подій, робота з меню, управління профілем, обробка ролей, взаємодія з базою) реалізований у окремому модулі. Це спрощує підтримку коду, забезпечує масштабування та покращує читабельність. Python добре підтримує модульність, працює з пакетами та має розвинену систему імпортів, що робить цей підхід логічним.

На етапі розробки бот запускався у локальному середовищі, що дозволяє швидко перевіряти роботу скриптів, тестувати компоненти та контролювати зміни в реальному часі. Python надає інструменти для тестування (pytest, unittest), які можна інтегрувати в проєкт для забезпечення надійності коду. Для тестування інтерактивних команд і callback-запитів Telegram API використовувалися механізми режиму sandbox, що дозволяють перевірити роботу бота без впливу на реальних користувачів.

Важливо вибрати інструменти для розгортання та подальшого використання бота. Telegram-боти можуть бути розміщені на локальних серверах або у хмарних платформах. Можна використовувати хмарні сервіси (Render, Amazon Web Services або VPS-провайдери), що надають інструменти для розгортання Python-додатків. Python-telegram-bot підтримує роботу в режимі long polling і в режимі webhook, що робить систему гнучкою та адаптованою до різних умов.

Вибір інструментів і бібліотек для створення телеграм-бота ґрунтується на вимогах проєкту та можливостях технологій. Python як мова програмування, бібліотека python-telegram-bot, СКБД SQLite, ORM SQLAlchemy та модульна структура коду є основою ефективної, стабільної та зручної системи. Вони забезпечують достатній рівень гнучкості для розвитку

бота, дозволяють інтегрувати нові функції та адаптувати систему під потреби спортивних клубів, що робить її перспективною.

3.3 Структура програмного забезпечення та опис модулів

Структура програми Telegram-бота для організації спортивних заходів базується на модульності, можливості розширення та чіткому розподілі відповідальності між компонентами. Це забезпечує керованість коду, його масштабованість і спрощення підтримки. Бот виконує багато функцій, тому важливо створити логічну структуру, що об’єднує компоненти в єдину систему.

Архітектура програми побудована за схемою «клієнт - сервер», де Telegram API є клієнтським рівнем, а серверна частина, написана на Python з використанням фреймворку Aiogram, реалізує бізнес-логіку, обробку станів, команд і бази даних PostgreSQL. Уся логіка розділена на модулі за функціональним призначенням, що дозволяє розвивати підсистеми, тестувати їх і забезпечувати стійкість програми до збоїв.

Загальна структура програмного забезпечення

Структура програмного проєкту складається з таких основних блоків:

1. Головний модуль запуску (bot.py)
2. Модуль обробки команд і callback-запитів
3. Модуль станів (FSM)
4. Модуль роботи з базою даних
5. Модуль моделей (схема БД та ORM-представлення, якщо використовується)
6. Модуль адміністрування подій
7. Модуль користувацької взаємодії
8. Модуль генерації повідомлень та клавіатур
9. Модуль конфігурацій та змінних середовища
10. Модуль логування та обробки помилок

Головний модуль запуску (**bot.py**)

Це ядро Telegram-бота, яке відповідає за ініціалізацію основних компонентів:

- створення об’єкта Bot;

- підключення диспетчера Dispatcher;
- реєстрацію хендлерів;
- підключення системи логування;
- запуск механізму webhook або long polling;
- ініціалізацію підключення до бази даних.

У цьому модулі виконується імпорт усіх допоміжних компонентів і модулів, що забезпечує централізоване керування роботою бота.

Модуль обробки команд та callback-запитів

Цей модуль містить набір хендлерів, які реагують на дії користувача:

- текстові команди (/start, /help, /events);
- callback-дії (натискання кнопок меню та списків);
- контекстні відповіді у межах FSM.

Модуль реалізує взаємодію з користувачем через Telegram API та викликає відповідну бізнес-логіку залежно від запиту.

Структурно він поділяється на:

- хендлери для користувачів;
- хендлери для адміністраторів;
- хендлери для реєстрації та перегляду подій;
- помічники для навігації між меню.

Завдяки цьому розділенню код залишається чистим, а масштабування - передбачуваним.

Модуль станів (FSM)

FSM - основа логічної моделі взаємодії користувача з ботом.

У модулі описано:

- оголошення класів станів;
- переходи між станами;
- валідацію введених даних;
- обробку частково завершених дій.

Використання Finite State Machine дозволяє:

- уникати помилок під час реєстрації;

- чітко контролювати етапи взаємодії (вибір події → вибір категорії → підтвердження);
- запобігати некоректним повідомленням;
- підтримувати складні діалогові сценарії.

FSM є обов’язковою частиною ботів цього типу, оскільки забезпечує контрольований процес взаємодії.

Модуль роботи з базою даних

Модуль відповідає за:

- встановлення підключення до PostgreSQL;
- виконання SQL-запитів;
- асинхронну роботу з даними через asyncpg;
- структуру таблиць та їх взаємозв’язки;
- операції CRUD (create, read, update, delete).

Тут визначено методи для:

- створення події;
- отримання списку подій;
- створення категорій;
- реєстрації учасників;
- перевірки повторної реєстрації;
- отримання даних для адміністратора.

Модуль є критично важливим, оскільки забезпечує збереження та цілісність інформації.

Модуль адміністрування

У цьому компоненті реалізовано функціонал для адміністраторів:

- створення нових спортивних заходів;
- редагування параметрів існуючих;
- керування категоріями;
- перегляд списків учасників;
- надсилання інформаційних повідомлень;
- обробка організаційних запросів.

Усі адміністративні операції перевіряються на наявність прав доступу.

Модуль користувацької взаємодії

Цей модуль відповідає за логіку взаємодії звичайного користувача із системою:

- перегляд доступних подій;
- реєстрація на спортивний захід;
- отримання інформації;
- скасування реєстрації (якщо передбачено);
- отримання сповіщень.

Модуль взаємодіє з FSM, модулем бази даних та системою повідомлень.

Модуль генерації повідомлень та клавіатур

Telegram-бот активно використовує:

- інлайн-кнопки;
- меню навігації;
- динамічно сформовані списки;
- повідомлення з форматуванням Markdown або HTML.

У цьому модулі створюються клавіатури, які відображають:

- список подій;
- категорії заходів;
- адміністративні меню;
- кнопки підтвердження;
- навігацію назад/вперед.

Це забезпечує зручний інтерфейс користувача, що є важливою частиною UX Telegram-систем.

Модуль конфігурацій

У модулі конфігурацій зберігаються:

- токен Telegram-бота (через .env);
- параметри підключення до БД;
- налаштування webhook;

- глобальні константи;
- налаштування логування.

Використання .env дозволяє забезпечити безпеку ключів і гнучкість конфігурації.

Модуль логування та обробки помилок

Система логування фіксує:

- помилки під час виконання коду;
- дії адміністраторів;
- перевантаження;
- SQL-помилки;
- доступ сторонніх користувачів до закритого функціоналу.

Помилки обробляються централізовано, що дозволяє уникати критичних збоїв і відстежувати проблеми.

3.4 Реалізація функціоналу Telegram-бота

Реалізація програмного функціоналу Telegram-бота для організації спортивних заходів здійснювалася на основі асинхронного Python-фреймворку Aiogram, який забезпечує ефективну взаємодію з Telegram API та дозволяє формувати модульну архітектуру застосунку. Процес розробки вимагав поетапної побудови логічної структури коду, визначення ключових обробників команд, створення системи станів для керування діалогами, а також імплементації інтеграції з базою даних PostgreSQL. У цьому підрозділі наведено детальний опис реалізації основних функціональних компонентів, що становлять основу роботи Telegram-бота.

Початковим етапом реалізації є створення головного файлу програми, який відповідає за ініціалізацію об’єкта Bot, диспетчера подій та підключення до конфігураційних змінних середовища. У коді визначено структуру імпорту модулів, після чого виконується реєстрація всіх хендлерів. Основна функція запуску застосунку містить цикл обробки подій, що активується після встановлення webhook або через механізм long polling. Хоча Telegram підтримує обидва методи, у даній розробці використано long polling через його простоту та відсутність необхідності додаткових серверних налаштувань.

Обробка команд реалізується через декоратори Aiogram, які дають змогу визначати функції, що виконуються у відповідь на певні події. Наприклад, команда /start ініціює вітальне повідомлення та формує меню, доступне користувачу.

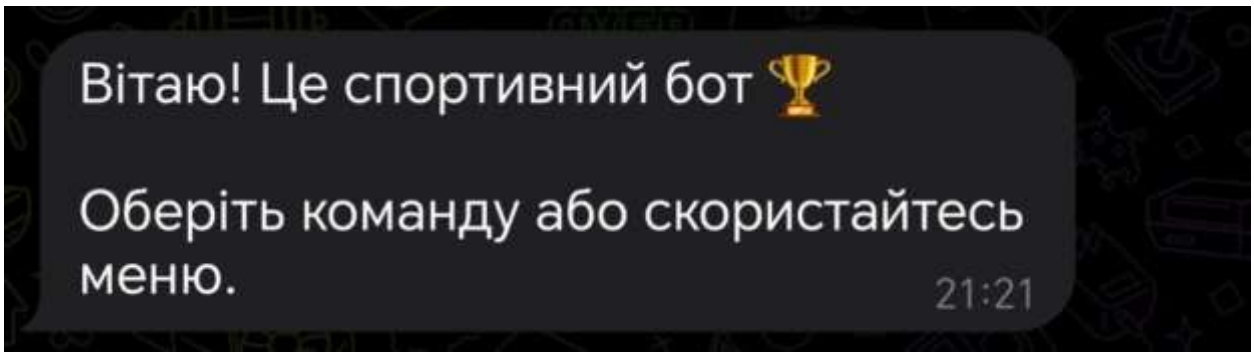

Рис. 3.1 – вітальне повідомлення

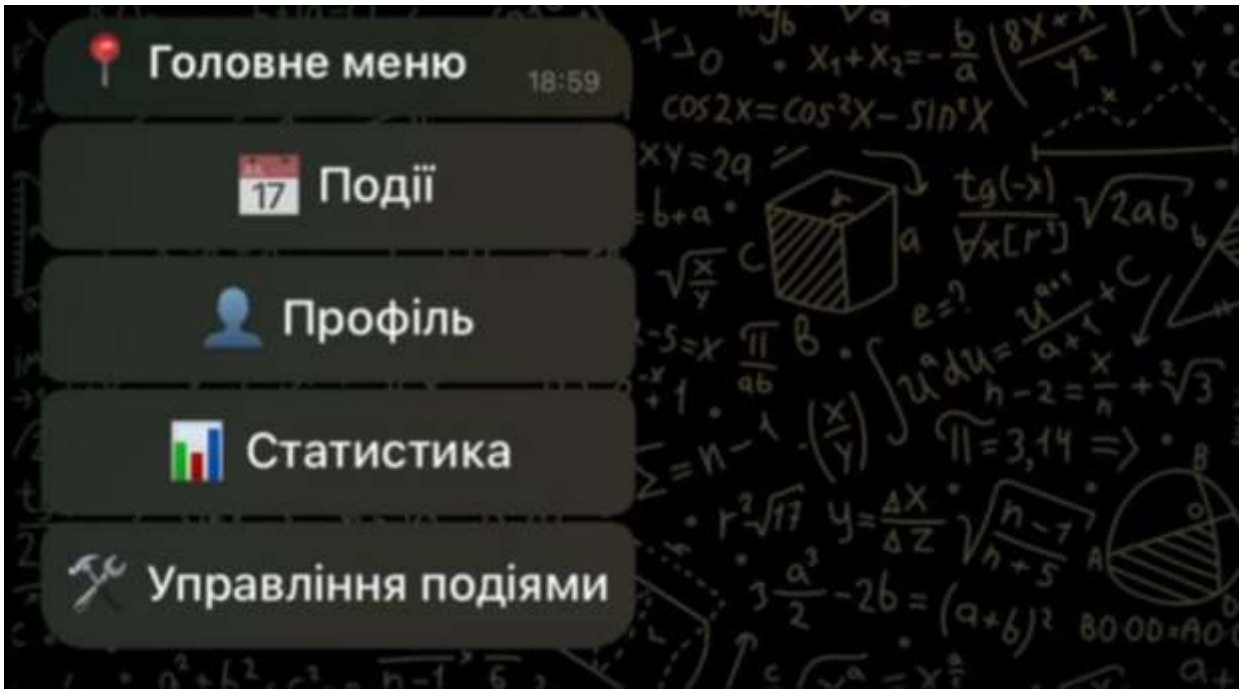

Рис. 3.2 – формевання меню

У коді ця логіка реалізується шляхом виклику об’єкта повідомлення (Message) та надсилання тексту у супроводі інлайн-клавіатури. Структура клавіатур формується окремими функціями, що повертають об’єкти InlineKeyboardMarkup. Такий підхід забезпечує відокремленість користувацького інтерфейсу від бізнес-логіки.

Ключовим елементом роботи бота є можливість перегляду спортивних подій та реєстрації на них. Для цього використовується система callback-хендлерів, які реагують на натискання кнопок у меню. У коді кнопка формується разом із унікальним callback-даними, що дозволяє визначити, яку

саме подію обрав користувач. Після натискання кнопки відправляється callback-запит, який потрапляє до відповідного хендлера. У цьому хендлері виконується запит до бази даних, де зберігається повна інформація про події. Дані передаються через асинхронні SQL-функції, реалізовані у модулі роботи з базою даних. Отриманий результат використовується для формування наступного меню — вибору категорій участі.

Систему реєстрації реалізовано за допомогою FSM (Finite State Machine). У коді визначено клас станів, кожен з яких відповідає за певний етап діалогу: вибір події, вибір категорії, підтвердження реєстрації. FSM дозволяє обмежити введення користувача рамками поточного кроку, запобігти некоректним даним та неочікуваним діям. Наприклад, якщо користувач надсилає текст у момент, коли необхідно вибрати категорію за допомогою кнопок, система ігнорує невалідне введення та пропонує повторити вибір. Коли користувач обирає категорію, FSM викликає метод доступу до бази даних, де виконується SQL-запит на створення нового запису в таблиці Registrations. Код також передбачає перевірку на повторну реєстрацію: якщо користувач уже зареєстрований на подію, система повідомляє про це та не створює дубліката.

Адміністративний функціонал реалізовано окремими хендлерами, доступ до яких надається лише користувачам із відповідними правами. Перевірка відбувається на основі user_id, який порівнюється з переліком адміністраторів у базі даних або у конфігураційному файлі. Адміністратор може створювати нові події, редагувати наявні та переглядати списки зареєстрованих учасників. У коді реалізація створення події включає послідовний діалог через FSM: введення назви, дати, опису, місця проведення та інших параметрів. Після завершення всі дані передаються до функції створення запису в таблиці Events. Для роботи зі списками учасників використано асинхронні SELECT-запити, які повертають структури даних у вигляді списку словників, що далі форматуються у текстове повідомлення.

Структура кодової бази передбачає наявність окремого модуля для створення та генерації інлайн-клавіатур. У цьому модулі визначено функції, що динамічно формують кнопки залежно від отриманих даних. Наприклад, для кожної події створюється окрема кнопка з callback-даними виду event_{id}, а для категорій — category_{id}. Такий підхід дає змогу масштабувати систему, додаючи нові види кнопок і меню без модифікації основної логіки.

Окремої уваги заслуговує реалізація обробки помилок. У коді передбачено глобальний хендлер errors_handler, який перехоплює винятки, що можуть виникнути під час виконання SQL-запитів, взаємодії з Telegram API або обробки станів FSM. Помилки записуються до журналу логування з використанням стандартного модуля logging, що забезпечує можливість подальшого аналізу та оперативного усунення технічних проблем.

3.5 Реєстрація користувачів

Реєстрація користувачів забезпечує інтеграцію ідентифікаційних даних користувачів з базою даних та створює основу для взаємодії з системою. У контексті спортивних заходів реєстрація дозволяє пов’язати користувача з подіями, категоріями участі та іншими компонентами платформи. Реалізація потребує чіткої структури, валідації даних та інтеграції з модулем обробки станів.

На першому етапі система ідентифікує користувача через Telegram API, використовуючи user_id, який є ключем для збереження інформації в базі даних. Після запуску команди /start користувач автоматично створюється в таблиці Users, якщо його запису немає. У базі даних фіксується ідентифікатор Telegram, ім’я користувача, повне ім’я, дата першої взаємодії та статус користувача.

```python
async def profile(update: Update, context):
    session = SessionLocal()
    user = session.query(User).filter(
        User.telegram_id == update.message.from_user.id
    ).first()
    session.close()

    if not user:
        await update.message.reply_text(
            "Ви ще не зареєстровані!\nВикористайте команду /register"
        )
        return

    text = (
        f"👤 *Ваш профіль*\n"
        f"ПІБ: {user.fullname}\n"
        f"Телефон: {user.phone}\n"
        f"Вік: {user.age}\n"
        f"Роль: {user.role}"
    )

    await update.message.reply_text(text, parse_mode="Markdown")
```

Рис. 3.3 – базова реєстрація користувача(кодом)

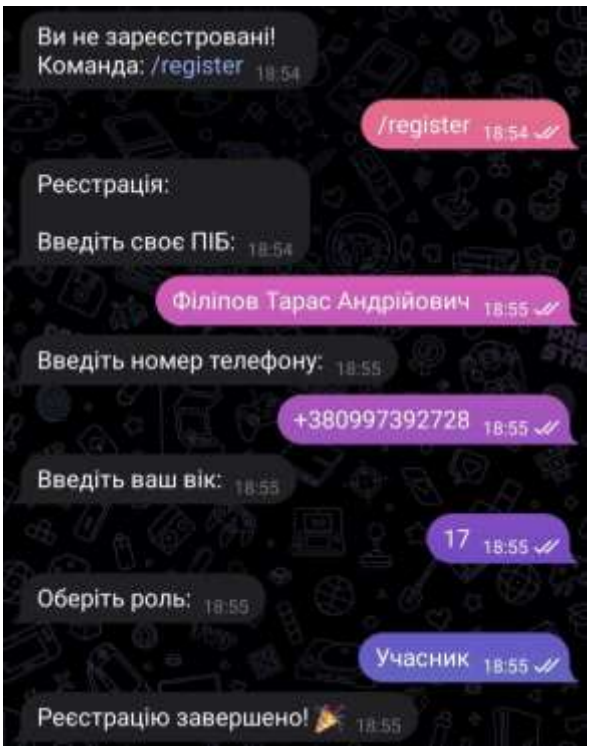

Рис. 3.4 – базова реєстрація користувача

Після базової реєстрації користувач може переглядати спортивні події і проходити процедуру реєстрації. Для цього використовується Finite State Machine (FSM), який забезпечує вибір події, категорії участі та підтвердження реєстрації. Застосування FSM дозволяє контролювати логіку дій та запобігає випадкам, коли користувач може некоректно взаємодіяти з ботом.

Рис. 3.5 – вибір ролі користувача

Процес реєстрації починається з вибору події, список якої формується динамічно на основі інформації з таблиці Events. Кожна подія відображається у вигляді інтерактивної кнопки, що містить унікальний callback-ідентифікатор. Коли користувач натискає кнопку, система переходить до наступного стану FSM, у якому здійснюється вибір категорії участі. Категорії зберігаються в таблиці Categories і пов’язані з подією через зовнішній ключ. Вибір категорії також оформлений у вигляді кнопок, генерованих у момент запиту до бази даних.

Після вибору категорії система перевіряє наявність реєстрації в таблиці Registrations, щоб уникнути дублювання записів. Якщо користувач вже зареєстрований, бот надсилає повідомлення і пропонує переглянути свої реєстрації або вибрати інший захід. Якщо реєстрації немає, система переходить до завершального етапу, де користувачеві пропонується підтвердження участі. У разі підтвердження в базу даних додається новий запис із прив’язкою до user_id, event_id та category_id. Запис доповнюється інформацією про дату реєстрації та статус участі.

Важливо забезпечити безпеку та захист даних користувача. Усі SQL-запити реалізовано у параметризованому форматі, що запобігає SQL-ін’єкціям. Крім цього, усі запити до бази даних виконуються у асинхронному режимі, що дозволяє ботові обробляти одночасні запити багатьох користувачів.

Загальна логіка реєстрації користувачів також передбачає підтримку механізму скасування або зміни реєстрацій. Ця функція може бути розширена. Наприклад, адміністратор може у будь-який момент викликати список учасників і змінювати статус їхніх заявок. Усі зміни фіксуються у таблиці Registrations, що забезпечує історію змін.

Під час розробки механізму реєстрації важлива генерація інформаційних повідомлень, які надсилаються користувачу після успішної

реєстрації. Повідомлення містять підтвердження участі, інформацію про дату проведення, обрану категорію та інструкції.

Механізм реєстрації користувачів у Telegram-боті поєднує автоматичну ідентифікацію через Telegram API, інтеграцію з базою даних, реалізацію FSM та механізми валідації. Система забезпечує зручність використання, безпечність, стійкість до помилок та чітку логіку. Завдяки цьому Telegram-бот може обробляти великі обсяги реєстрацій, підтримувати спортивні заходи та забезпечувати інтерактивну взаємодію з користувачами.

3.6 Створення та адміністрування спортивних заходів

Функціонал створення та адміністрування спортивних заходів є ключовим компонентом Telegram-бота, оскільки він дозволяє організаторам формувати розклад, визначати структуру подій та керувати реєстраціями учасників. Адміністрування може охоплювати широкий спектр завдань - від створення нових подій до модерації списків учасників та опрацювання рішень. Реалізація цього функціоналу вимагає підходу до формування структури, інтеграції з базою даних та контролю доступу.

Ключовим принципом є розмежування прав доступу між адміністраторами та користувачами. Програмне забезпечення використовує механізм перевірки user_id, що гарантує доступ до адміністративних функцій лише авторизованим організаторам. При кожному виклику команди бот перевіряє права користувача, що унеможливлює втручання у дані та забезпечує стабільність роботи системи.

```python
async def menu_buttons(update: Update, context: ContextTypes.DEFAULT_TYPE):
    query = update.callback_query
    await query.answer()

    action = query.data

    if action == "menu_events":
        await query.edit_message_text("📅 Команда: /events")
        return

    if action == "menu_events_manage":
        await query.edit_message_text(
            "🛠 Управління подіями:\n"
            "/create_event – створити\n"
            "/edit_event – редагувати\n"
            "/delete_event – видалити"
        )
        return

    if action == "menu_profile":
        await query.edit_message_text("👤 Команда: /profile")
        return

    if action == "menu_stats":
        await query.edit_message_text("📊 Команда: /stats")
        return
```

Рис. 3.6 – управління подіями візуалізовано кодом

Процес створення заходу реалізовано через діалогову взаємодію адміністратора з ботом, що дозволяє вводити всю необхідну інформацію. За допомогою системи станів (FSM) адміністратор вводить назву, опис, місце, дату та час, а також параметри. Переходячи між станами, бот виконує валідацію введених даних, перевіряючи формат дати, наявність текстових елементів та довжину опису. Це мінімізує помилки і гарантує, що інформація є валідною та повною.

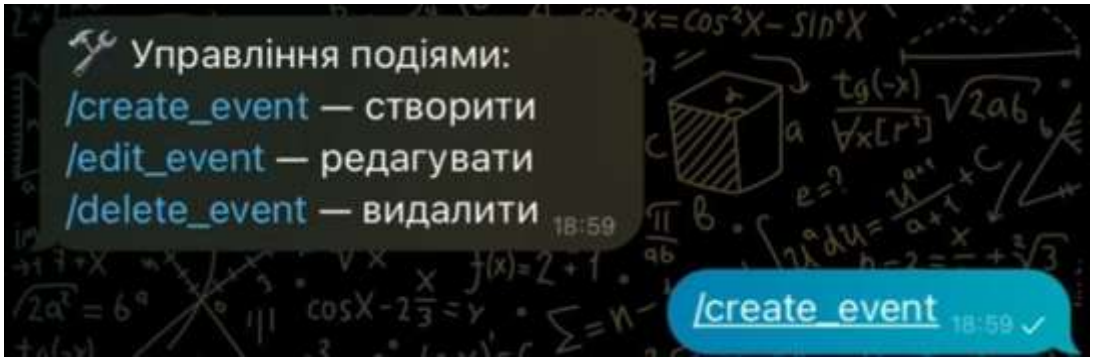

Рис. 3.7 – створення спортивного заходу

Після введення основної інформації Telegram-бот надсилає повідомлення з пропозицією підтвердити створення заходу, що дозволяє перевірити введену інформацію перед внесенням її в базу даних.

```python
async def create_event_start(update: Update, context: ContextTypes.DEFAULT_TYPE):
    if not is_organizer(update):
        await update.message.reply_text("❌ Доступ лише для організатора або адміна.")
        return ConversationHandler.END

    await update.message.reply_text("Введіть назву події:")
    return TITLE


async def event_title(update: Update, context):
    context.user_data["new_event"] = {"title": update.message.text}
    await update.message.reply_text("Короткий опис:")
    return DESC


async def event_desc(update: Update, context):
    context.user_data["new_event"]["description"] = update.message.text
    await update.message.reply_text("Локація:")
    return LOCATION
```

Рис. 3.8 – створення спортивного заходу візуалізація коду

У разі підтвердження бот створює новий запис у таблиці Events, який отримує унікальний ідентифікатор і зберігає параметри заходу, а також дату створення та статус активності. У разі помилки адміністратор може скасувати створення або повернутися до кроку, що забезпечує гнучкість.

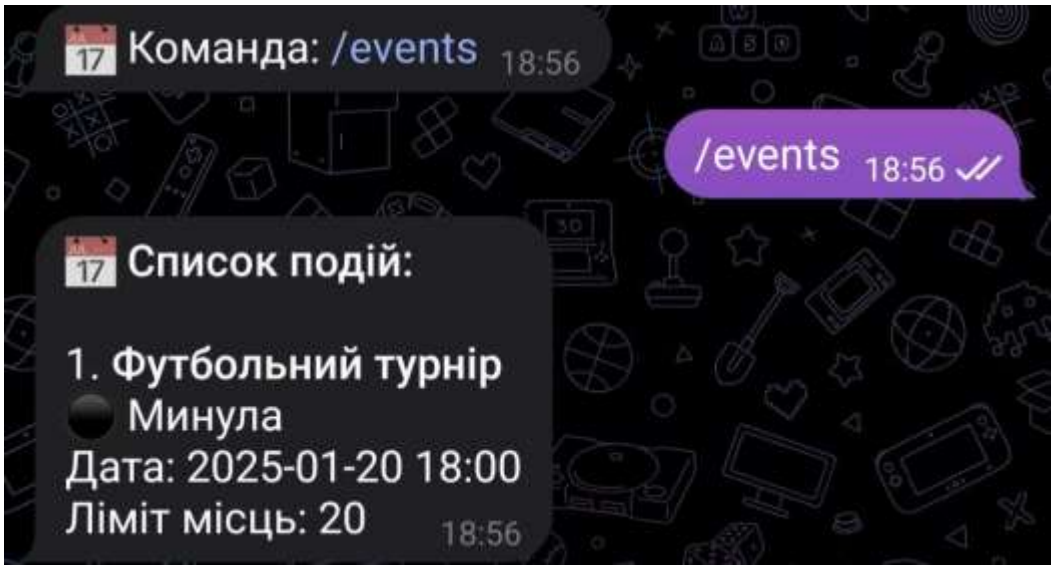

Рис. 3.9 – вже створенний спортивний захід

Після створення заходу адміністратор може переходити до формування категорій участі, які зберігаються в таблиці Categories і пов’язуються з подією через зовнішній ключ. Процес додавання категорій також реалізовано через FSM, що дозволяє вводити назву, опис та обмеження категорії. Валідація введених параметрів гарантує, що дані будуть коректно збережені.

Функціонал передбачає можливість перегляду та модерації списку учасників. Telegram-бот запитує таблицю Registrations та формує структурований перелік зареєстрованих користувачів. Адміністратор може переглядати інформацію про учасників, їхні категорії, статуси реєстрації та дату подання заявки. У разі потреби організатор може змінити статус заявки. Логіка зміни статусу реалізована через окремий хендлер, що дозволяє здійснювати керування записами у базі даних.

Система надає адміністраторам інструменти для інформування всіх зареєстрованих учасників або окремих категорій. Повідомлення надсилаються через цикл, який перебирає всі відповідні user_id та викликає метод Telegram API. Щоб не перевищувати обмеження Telegram щодо кількості повідомлень, код використовує затримку між відправленнями або чергу.

Модуль створення та адміністрування спортивних заходів використовує FSM, асинхронну обробку запитів та параметризовані SQL-функції для забезпечення точності, безпеки та зручності адміністрування. Підхід гарантує ефективність роботи Telegram-бота під час проведення спортивних заходів та дозволяє організаторам керувати усіма аспектами подій.

3.7 Аналіз роботи та можливості розвитку

Аналіз роботи Telegram-бота є важливим етапом оцінювання та визначення перспектив його вдосконалення. Розроблена система допомагає спортивним менеджерам, зменшує навантаження на організаторів та забезпечує інструмент для реєстрації на заходи, отримання інформації та взаємодії з організаторами. Тестування показало, що Telegram-бот працює стабільно та адаптується до різних форматів спортивних заходів.

Система використовує Telegram, що забезпечує доступність бота для користувачів. Інтерфейс на інлайн-клавіатурах та структурованих меню мінімізує кількість введення тексту та зменшує ймовірність помилок. Логіка переходів між командами, реалізована за допомогою FSM, спрощує структуру діалогів. Здатність обробляти великий обсяг запитів забезпечується Aiogram та оптимізацією доступу до бази даних PostgreSQL.

Реалізація асинхронних SQL-запитів, логування помилок, контролю доступу та параметризованих операцій з базою даних дозволяє забезпечити безпеку та зберегти цілісність даних. Тестування показало, що бот обробляє типові сценарії використання, масову реєстрацію, надсилання сповіщень та адміністративні операції. Система адаптується до конфігурацій серверного середовища.

Telegram-бот покращує процес управління реєстраціями та комунікацією з учасниками. За допомогою документованих процесів адміністратори можуть створювати події, керувати категоріями, перевіряти списки учасників та надсилати сповіщення. Це спрощує процедури та мінімізує людський фактор. Бот підвищує оперативність ухвалення рішень та покращує управління спортивними заходами.

Реалізована архітектура дозволяє розширювати функціонал та додавати нові модулі без змін програми. Автономні модулі хендлерів, окремі компоненти для роботи з базою даних, централізований модуль конфігурацій

та використання .env забезпечують гнучкість та можливість інтеграції сервісів у майбутньому.

Аналітичні модулі дозволять автоматично формувати статистику за учасниками, подіями та категоріями, допомагаючи визначати рейтинги, порівнювати динаміку участі та аналізувати ефективність заходів. Інтеграція з геолокаційними сервісами може спростити пошук місця проведення події або дозволити боту надсилати нагадування з урахуванням місця розташування користувача.

Додавання системи електронних квитків або QR-кодів спростить реєстрацію на місці проведення заходу та дозволить пришвидшити процеси контролю та обліку, а також знизити черги. Інтеграція з платіжними сервісами дозволила б організаторам приймати оплату через Telegram.

Розроблений Telegram-бот відповідає поставленим завданням, забезпечує зручну та стабільну взаємодію з користувачами, а також надає потужний інструмент управління спортивними подіями. Гнучкість системи, модульність архітектури та можливість інтеграції рішень роблять її платформою для розвитку та масштабування як у спортивній сфері, так і в інших галузях.

ВИСНОВКИ

У дипломній роботі я провів дослідження, зробив аналіз і створив Telegram-бота для організації спортивних подій. Це дало змогу виконати поставлене завдання та досягти мети. Розроблена система поєднує сучасні технології, автоматизовані інструменти та гнучку структуру, що спрощує взаємодію між організаторами та учасниками спортивних заходів. Створений продукт поліпшує реєстрацію, спілкування, інформування та керування, а також зменшує час і витрати на ці дії.

Теоретичний аналіз показав важливість теми, оскільки зараз цифровізація спорту вимагає інструментів для швидкої організації спортивних заходів. Популярність Telegram робить ботів придатними для автоматизації спортивного менеджменту. У вступі я визначив проблеми в організації спортивних подій, такі як складна обробка заявок, відсутність спільної системи комунікації, різні підходи до зберігання даних і неефективна реєстрація. Тому створення Telegram-бота стало своєчасним рішенням, яке відповідає потребам галузі та технологічним можливостям.

У першому розділі я проаналізував теорію та особливості цифрової організації спортивних подій. Я дослідив ключові елементи керування, роль мобільних платформ у спілкуванні та використання месенджерів. Також я розглянув структуру Telegram API, функції ботів і підходи до їх впровадження в бізнес-процеси. Важливою умовою було питання безпеки та захисту даних. Аналіз літератури показав, що Telegram-боти допомагають автоматизувати організацію спортивних заходів і покращують спілкування з учасниками.

Другий розділ описує технічні аспекти створення Telegram-бота. Я опрацював структуру програми, яка базується на фреймворку Aiogram і підтримує велике навантаження під час реєстрацій. Я розглянув модель бази даних PostgreSQL, яка забезпечує стабільність та ефективність роботи з інформацією. Значну увагу приділено UML-діаграмам, які графічно показують бізнес-процеси та взаємодію компонентів системи. У розділі також

вивчено питання масштабування та продуктивності. Аналіз показує, що розроблена структура є гнучкою та готовою до розширення.

У практичному розділі я реалізував Telegram-бота, що підтвердило ефективність обраних технологій. Я детально розглянув структуру програми, логіку обробки команд, структуру станів (FSM) і реалізацію реєстрації користувачів і керування спортивними заходами. Практична частина показала, що Aiogram і PostgreSQL забезпечують швидку взаємодію з базою даних, а модульний підхід полегшує підтримку та масштабування. Функціонал бота включає створення подій, керування категоріями, реєстрацію учасників, формування списків, модерацію заявок і надсилання сповіщень. Це підтверджує, що система відповідає вимогам і вирішує проблеми.

Оцінка ефективності показала, що Telegram-бот покращує організаційні процеси, зменшує навантаження на адміністраторів і робить користування зручним. Система автоматизує рутинні дії, усуває помилки, спрощує обробку даних і створює спільний інформаційний простір для всіх учасників спортивних заходів. Аналіз довів, що бот може добре працювати в різних умовах і має можливості для розширення.

Можливості вдосконалення Telegram-бота великі. Вони включають модулі аналітики, інтеграцію з геолокацією, систему електронних квитків і QR-кодів, платіжні системи, розширений інтерфейс адміністратора та персоналізацію взаємодії з користувачами. Це дасть змогу системі вийти за межі базових функцій і стати багатофункціональною цифровою платформою для спортивних організацій.

Отже, розроблений Telegram-бот підтвердив свою важливість, функціональність, надійність і ефективність як інструмент для організації спортивних заходів. Система може використовуватися у спортивних клубах, федераціях, навчальних закладах і на заходах різних масштабів. Дипломна робота підтвердила важливість використання сучасних цифрових технологій

у спортивному менеджменті та відкрила можливості для розвитку таких систем.

СПИСОК ВИКОРАСТАНИХ ДЖЕРЕЛ

1. Telegram Messenger LLP. Telegram Bot API Documentation. – Режим доступу: https://core.telegram.org/bots/api
2. Mamkin A., Aiogram Documentation. – Режим доступу: https://docs.aiogram.dev
3. PostgreSQL Global Development Group. PostgreSQL Documentation. – Режим доступу: https://www.postgresql.org/docs
4. Fowler M. Patterns of Enterprise Application Architecture. – Addison-Wesley, 2002. – 560 p.
5. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. – Addison-Wesley, 1994. – 416 p.
6. Martin R. Clean Code: A Handbook of Agile Software Craftsmanship. – Prentice Hall, 2008. – 464 p.
7. Hunt A., Thomas D. The Pragmatic Programmer. – Addison-Wesley, 2019. – 352 p.
8. Sommerville I. Software Engineering. 10th ed. – Pearson Education, 2015. – 820 p.
9. ISO/IEC 25010:2011 Systems and software engineering — System and software quality models.
10. Pressman R. Software Engineering: A Practitioner’s Approach. – McGraw-Hill, 2014. – 936 p.
11. Кравець П. Розробка чат-ботів у Telegram: сучасні інструменти та можливості // Інформаційні технології. – 2021. – №2. – С. 44–52.
12. Фісенко І. М. Використання месенджерів у цифровому середовищі // Цифрові технології XXI століття. – Київ, 2020. – С. 113–120.
13. Бойко О. В. Автоматизація управління спортивними подіями // Спортивний вісник. – 2019. – №4. – С. 89–94.
14. Воронін В. Основи організації спортивних заходів. – Львів: ЛДУФК, 2018. – 224 с.

15. Дудко С. М. Цифровізація спортивної сфери: тенденції та виклики // Наука і спорт. – 2022. – №3. – С. 33–41.
16. Google Developers. OAuth 2.0 for Client Applications. – Режим доступу: https://developers.google.com/identity
17. Python Software Foundation. Python Documentation. – Режим доступу: https://docs.python.org
18. Kurose J. F., Ross K. W. Computer Networking: A Top-Down Approach. – Pearson, 2017. – 864 p.
19. Stallings W. Operating Systems: Internals and Design Principles. – Pearson, 2018. – 896 p.
20. Штефан Л., Олійник В. Інформаційні системи організації спортивних заходів // Системний аналіз. – 2021. – С. 77–84.
21. Docker Inc. Docker Documentation. – Режим доступу: https://docs.docker.com
22. Redis Labs. Redis Documentation. – Режим доступу: https://redis.io/documentation
23. Jadon R., Dogra D. A Survey of Chatbot Implementation Techniques // AI Review. – 2020. – Vol. 53. – P. 3191–3219.
24. Shum H.-Y., He X., Li D. From Eliza to XiaoIce: Challenges and Opportunities in Conversational AI // Science. – 2018. – Vol. 359. – P. 78–83.
25. Ray S. Python Programming: A Modular Approach. – Oxford University Press, 2019. – 640 p.
26. Kuznetsov M. Modern Methods of Event Management in Sports. – Berlin: Springer, 2021. – 274 p.
27. ISO/IEC 27001:2013 Information technology - Security techniques - Information security management systems.
28. Internet Engineering Task Force (IETF). RFC 8259: The JavaScript Object Notation (JSON) Data Interchange Format.
29. Van Rossum G. The History of Python // ACM Computing Surveys. – 2020.

30. Zignani M., Quadri C., Gaito S. “Telegram as a Social Network: Patterns and Analysis” // IEEE International Conference on Communications. – 2019.

Додаток А

```python
class User(Base):
    __tablename__ = "users"

    id = Column(Integer, primary_key=True, index=True)
    telegram_id = Column(Integer, unique=True, index=True)

    fullname = Column(String, nullable=False)
    phone = Column(String, nullable=False)
    age = Column(Integer, nullable=False)

    role = Column(String, default="Учасник")

    # Важливо: повинно бути "registrations"
    registrations = relationship("Registration", back_populates="user")

    def __repr__(self):
        return f"<User {self.fullname} ({self.role})>"
```

```python
class Event(Base):
    __tablename__ = "events"

    id = Column(Integer, primary_key=True, index=True)

    title = Column(String, nullable=False)
    description = Column(Text, nullable=False)
    location = Column(String, nullable=False)
    date = Column(DateTime, nullable=False)

    seats = Column(Integer, default=10)

    creator_id = Column(Integer, ForeignKey("users.id"))
    creator = relationship("User")

    # правильна назва і збіг back_populates
    registrations = relationship("Registration", back_populates="event")

    def __repr__(self):
        return f"<Event {self.title} on {self.date}>"
```

```python
class Registration(Base):
    __tablename__ = "registrations"

    id = Column(Integer, primary_key=True, index=True)

    user_id = Column(Integer, ForeignKey("users.id"))
    event_id = Column(Integer, ForeignKey("events.id"))

    # Обидва зв'язки тепер збігаються з моделями User та Event
    user = relationship("User", back_populates="registrations")
    event = relationship("Event", back_populates="registrations")

    def __repr__(self):
        return f"<Registration user={self.user_id} event={self.event_id}>"
```

```python
class Log(Base):
    __tablename__ = "logs"

    id = Column(Integer, primary_key=True, index=True)

    user_id = Column(Integer)
    action = Column(String, nullable=False)
    timestamp = Column(DateTime, default=datetime.datetime.~~utcnow~~)

    def __repr__(self):
        return f"<Log {self.action} at {self.timestamp}>"
```

```python
class Reminder(Base):
    __tablename__ = "reminders"

    id = Column(Integer, primary_key=True, index=True)

    event_id = Column(Integer, ForeignKey("events.id"))
    remind_time = Column(DateTime, nullable=False)
    type = Column(String)

    def __repr__(self):
        return f"<Reminder event={self.event_id} type={self.type}>"
```