МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проєктами

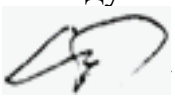
Кафедра програмного забезпечення автоматизованих систем

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма «Інженерія програмного забезпечення»

«Допущений до захисту»

Завідувач кафедри

 проф. Приходько С.Б.

«__» _____ 2026 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти «бакалавр»

на тему: Розробка телеграм-боту афіши розважальних заходів м. Миколаєва

Виконав: студент групи 4151

 Пацалюк Д.Р.

Керівник роботи:

к.т.н., доцент

 Макарова Л. М

м. Миколаїв – 2026 рік

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра програмного забезпечення автоматизованих систем
Спеціальність 121 «Інженерія програмного забезпечення»
Освітня програма «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____ доц. Макарова Л.М.
(підпис)
«_07_» ___04___ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «бакалавр»

Студенту Пацалюку Денису Руслановичу _____
(Прізвище, ім'я, по батькові)

1. Тема роботи: Розробка телеграм-боту афіши розважальних заходів м. Миколаєва____
_____
Керівник роботи: доцент кафедри ПЗАС, к.т.н., доцент Макарова Л. М. _____
Затверджені наказом ректора №275-уч від 07.04.2026 р.

2. Термін подання роботи: 01.06.2026 р. _____

3. Вихідні дані по роботі: _____
_____

4. Перелік питань, що належать до розробки (найменування розділів)_____
Титульний аркуш; Завдання на кваліфікаційну роботу; Анотація (українською та англійською мовами); Зміст; Перелік умовних позначень, символів, одиниць та термінів (при необхідності); Вступ; Аналіз практичної задачі інженерії програмного забезпечення; Проєкт програмного забезпечення; Результати розробки програмного забезпечення; Розділ з охорони праці; Висновки; Список використаних джерел; Додатки (технічне завдання, текст програми, опис програми, інструкція користувача, програма та методика випробувань програмного забезпечення).

5. Перелік презентаційних матеріалів _____
Титульний слайд, Актуальність та мета роботи, Огляд існуючих програмних рішень для розробки телеграм-боту, Функціональні вимоги до програмного забезпечення, Архітектура програмного забезпечення, Сценарії варіантів використання, Інформаційна модель телеграм-боту, Діаграма класів телеграм-боту, Діаграма послідовності, Засоби програмної реалізації, Фізична модель даних, Результати розробки, Висновки, Заключний слайд. _____
_____
_____
_____
_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4	Гурець Н.В., ст. викладач		

7. Дата видачі завдання 07.04.2025 р.

К А Л Е Н Д А Р Н И Й П Л А Н

Номер	Назва етапів роботи	Терміни виконання	Примітка
1.	Підготовка розділу ВСТУП	30.03.2026	ПП*
2.	Аналіз практичної задачі інженерії ПЗ	06.04.2026	ПП*
3.	Аналіз вимог до ПЗ	13.04.2026	ПП*
4.	Розробка архітектури ПЗ	20.04.2026	
5.	Розробка проекту ПЗ	04.05.2026	
6.	Реалізація та тестування ПЗ	11.05.2026	
7.	Підготовка розділу Результати розробки ПЗ	15.05.2026	
8.	Підготовка розділу з охорони праці	18.05.2026	ПП*
9.	Підготовка розділу ВИСНОВКИ	22.05.2026	
10.	Оформлення списку використаних джерел та додатків	25.05.2026	
11.	Подання на кафедру ПЗАС тексту остаточного варіанту роботи, підписаного її керівником, у роздрукованому та електронному форматі разом із заявами щодо самостійності виконання роботи та ідентичності друкованої та електронної версій роботи (Додатки 1 і 2 «Порядку здійснення заходів з перевірки робіт на наявність текстових збігів/ідентичності/схожості із використанням програмно-технічних засобів», який введений в дію наказом ректора НУК за №20 від 20.01.2020 р.)	01.06.2026	не пізніше, ніж за 14 діб до захисту (згідно п.4.1 зазначеного Порядку
12.	Підготовка презентації та доповіді	05.06.2026	
13.	Попередній захист роботи на засіданні кафедри ПЗАС	08.06.2026	
14.	Подання на кафедру ПЗАС електронних версій наступних документів у форматі pdf: кваліфікаційної роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК за №287-уч від 19.05.2020 р.); презентації доповіді, а також Авторського договору з додатком	15.06.2026	

* - за результатами переддипломної практики (ПП), яка була з 02.02.2026 до 22.03.2026 р.

Студент _____ Пацалюк Д.Р.
(підпис) (ПІБ)

Керівник роботи _____ Макарова Л.М.
(підпис) (ПІБ)

АНОТАЦІЯ

Кваліфікаційна робота присвячена розв’язанню практичної задачі інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов, розробці телеграм-боту афіши розважальних заходів м. Миколаєва, впровадження якого дозволить мешканцям та гостям Миколаєва у зручному вигляді отримувати інформацію щодо розважальних заходів, які проводяться у місті.

У кваліфікаційній роботі міститься аналіз практичної задачі інженерії програмного забезпечення, постановка задачі на розробку програмного забезпечення, проект програмного забезпечення, результати розробки програмного забезпечення та розділ з охорони праці.

Кваліфікаційна робота викладена на 87 сторінках друкованого тексту та містить 16 рисунків, 10 таблиць, 5 додатків та список використаних джерел з 22 найменувань.

ABSTRACT

The qualification work is dedicated to solving the practical task of software engineering, characterized by complexity and uncertainty of conditions, software development of the telegram-bot of the movie posters of Mykolaiv, the implementation of which will allow residents and guests of Mykolaiv to conveniently receive information about entertainment events held in the city.

The qualification work contains an analysis of the practical problem of software engineering, a statement of the problem for software development, a software project, the results of software development and a section on labor protection.

Qualification work is presented on the 87 pages of printed text and contains 16 figures, 10 tables, 5 appendices and a list of references from 22 names.

ЗМІСТ

ВСТУП

Кваліфікаційна робота присвячена розв’язанню практичної задачі інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов – розробці телеграм-боту афіши розважальних заходів м. Миколаєва, який допоможе мешканцям та гостям Миколаєва у зручному вигляді отримувати інформацію щодо розважальних заходів, які проводяться у місті.

Актуальність розробки телеграм-боту афіші розважальних заходів м. Миколаєва зумовлена зростанням потреби мешканців та гостей міста в оперативному отриманні актуальної інформації про культурні, освітні, спортивні та розважальні події, які проходять або будуть проходити у місті. Використання телеграм-боту як інформаційного сервісу дозволяє автоматизувати процес збору, обробки та поширення інформації про заходи, забезпечити зручний доступ до них через популярний месенджер, а також підвищити рівень інформованості користувачів. Розробка такого програмного продукту потребує інтеграції функцій отримання даних із різних джерел, їх структурування, пошуку, фільтрації та персоналізованого відображення відповідно до запитів користувачів.

Комплексність та невизначеність умов розв’язання поставленої задачі полягають у необхідності роботи з великою кількістю різнорідних та динамічних даних про заходи, що надходять із різних інформаційних ресурсів і можуть мати неповний, неструктурований або несвоєчасно оновлений характер. Додаткову складність створюють зміни в розкладі подій, скасування або перенесення заходів, відмінності у форматах подання інформації та непередбачуваність поведінки користувачів під час взаємодії з ботом. У зв’язку з цим розроблюваний програмний продукт повинен забезпечувати гнучку обробку даних, підтримувати актуальність інформації та коректно функціонувати в умовах постійних змін зовнішнього середовища.

Метою даної кваліфікаційної роботи є розробка телеграм-боту афіши розважальних заходів м. Миколаєва.

Для досягнення поставленої мети потрібно вирішити такі завдання:

– виконати аналіз практичної задачі інженерії програмного забезпечення з теми кваліфікаційної роботи;

– виконати аналіз існуючих програмних рішень для розробки телеграм-боту;

– виконати постановку задачі на розробку телеграм-боту афіши розважальних заходів м. Миколаєва;

– виконати аналіз вимог до телеграм-боту афіши розважальних заходів м. Миколаєва;

– розробити архітектуру телеграм-боту афіши розважальних заходів м. Миколаєва;

– розробити проект телеграм-боту афіши розважальних заходів м. Миколаєва.

Об’єктом кваліфікаційної роботи є процес розробки телеграм-боту афіши розважальних заходів м. Миколаєва.

1 АНАЛІЗ ПРАКТИЧНОЇ ЗАДАЧІ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ ТЕЛЕГРАМ-БОТУ АФІШИ РОЗВАЖАЛЬНИХ ЗАХОДІВ М. МИКОЛАЄВА

1.1 Аналіз практичної задачі інженерії програмного забезпечення

У сучасних умовах цифровізації суспільства значна частина інформації про культурні, спортивні, освітні та розважальні заходи поширюється через інтернет-ресурси та соціальні мережі. Для мешканців та гостей міста Миколаєва актуальною є проблема оперативного отримання повної та достовірної інформації про події, що відбуваються в місті, оскільки відомості про різні заходи часто розміщуються на різних вебсайтах, сторінках соціальних мереж, у телеграм-каналах та інших джерелах, що ускладнює їх пошук та потребує значних витрат часу користувачів на моніторинг інформаційного простору. Коротко розглянемо деякі з таких ресурсів.

Одним із таких ресурсів є портал «Культурний центр Миколаївщини», який містить афішу культурних подій області, новини закладів культури та інформацію про майбутні заходи. Перевагою ресурсу є наявність централізованої бази подій, однак для отримання повної картини користувачеві необхідно самостійно переглядати окремі сторінки закладів, новини та афішу, що потребує додаткових витрат часу [1]. Аналогічним ресурсом з точки зору як наявної інформації, так і переваг та недоліків, є сайт Обласного палацу культури [2]. Також поширеним джерелом інформації є спеціалізовані сайти афіш та продажу квитків, зокрема Teatr.org.ua, де публікуються відомості про концерти, вистави та гастрольні тури. Однак такі ресурси охоплюють лише окремі категорії заходів і не забезпечують єдиного інформаційного простору для всіх розважальних подій міста. До того ж, вони, зазвичай, публікують інформацію про розважальні події на всій території України, а не тільки у місті Миколаєві [3].

Крім вебсайтів, значна кількість анонсів поширюється через сторінки організацій у соціальних мережах Facebook, Instagram та Telegram-канали.

Інформація в них часто оновлюється оперативніше, проте користувачеві доводиться підписуватися на численні канали та сторінки, самостійно відстежувати нові публікації та шукати потрібні відомості серед великого обсягу повідомлень. До того ж, підписка на деякі Telegram-канали, зокрема, Telegram-канали «Куди піти? | Миколаїв» доступна тільки для користувачів версії Pro, яка є платною для користувачів Telegram [4.]

У результаті дані про події є розосередженими між багатьма джерелами, що ускладнює їх пошук та аналіз. Зазначені недоліки обумовлюють доцільність розробки телеграм-боту, який забезпечить централізований доступ до актуальної афіші розважальних заходів м. Миколаєва через єдиний інтерфейс взаємодії.

Одним із перспективних способів вирішення зазначеної проблеми є створення телеграм-боту афіші розважальних заходів. Використання платформи Telegram обумовлено її широкою популярністю серед користувачів, доступністю на різних пристроях та наявністю зручного програмного інтерфейсу для розробки ботів. Телеграм-бот може виконувати функції централізованого джерела інформації, забезпечуючи швидкий доступ до актуальної афіші заходів без необхідності відвідування великої кількості інформаційних ресурсів [5, 6].

Практична задача інженерії програмного забезпечення полягає у розробці програмної системи, яка забезпечуватиме збір, зберігання, обробку та надання користувачам інформації про розважальні заходи міста Миколаєва через інтерфейс телеграм-боту. Система повинна надавати можливість перегляду списку заходів, пошуку подій за категоріями та датами, отримання детальної інформації про конкретний захід, а також забезпечувати зручну взаємодію користувача з інформаційним сервісом.

Особливістю даної задачі є необхідність роботи з динамічними даними, які можуть регулярно змінюватися. Організатори заходів можуть змінювати дату, час, місце проведення або скасовувати події. Тому програмне забезпечення повинно підтримувати механізми оперативного оновлення інформації та забезпечувати актуальність даних, що надаються користувачам. Крім того, необхідно враховувати можливість розширення функціональних можливостей

системи, зокрема додавання нових категорій заходів, впровадження персоналізованих рекомендацій або системи сповіщень про майбутні події.

З точки зору інженерії програмного забезпечення задача характеризується комплексністю, оскільки потребує інтеграції декількох взаємопов’язаних компонентів: клієнтського інтерфейсу телеграм-боту, серверної частини, бази даних та засобів адміністрування інформаційного наповнення. Додаткову складність створює необхідність забезпечення надійності, зручності використання, швидкодії та можливості супроводу програмного продукту в процесі його експлуатації.

Таким чином, розробка телеграм-боту афіші розважальних заходів м. Миколаєва є актуальною практичною задачею інженерії програмного забезпечення, спрямованою на підвищення доступності інформації про події міста та створення зручного інструменту для взаємодії користувачів з актуальною афішею заходів.

1.2 Аналіз існуючих програмних рішень для розробки телеграм-боту

Окрему увагу при виборі платформи для реалізації чат-боту необхідно приділити аналізу сучасних месенджерів, які надають можливість створення програмних агентів для автоматизованої взаємодії з користувачами. Найбільш поширеними платформами є Telegram, Facebook Messenger, Discord та Viber. Кожна з них надає засоби для створення ботів, проте відрізняється рівнем відкритості програмного інтерфейсу, доступністю документації, зручністю розробки та особливостями цільової аудиторії.

Facebook Messenger є одним із найпопулярніших месенджерів у світі та підтримує інтеграцію з бізнес-сервісами і чат-ботами. Платформа дозволяє автоматизувати спілкування з користувачами, надсилати повідомлення та реалізовувати різноманітні сценарії взаємодії. Водночас використання Messenger тісно пов'язане з екосистемою Meta та сервісами Facebook, що ускладнює доступ

до функціоналу для частини користувачів. Крім того, політика платформи та доступність окремих інструментів можуть змінюватися залежно від вимог компанії-власника [7].

Discord є популярною платформою для спілкування спільнот за інтересами, зокрема у сфері комп'ютерних ігор та інформаційних технологій. Система надає розвинений програмний інтерфейс для створення ботів, які можуть реагувати на події, виконувати команди користувачів та взаємодіяти із зовнішніми сервісами. Однак основним середовищем використання Discord є сервери спільнот, тому його застосування для широкого інформування населення міста про розважальні заходи є менш доцільним через специфіку аудиторії та принципи організації взаємодії користувачів [8].

Месенджер Viber також підтримує створення ботів і бізнес-акаунтів. Його перевагою є значна популярність серед українських користувачів. Проте створення нових ботів на платформі Viber орієнтоване переважно на комерційне використання, а доступ до окремих сервісів може вимагати додаткових витрат. Це обмежує можливості використання платформи для навчальних і некомерційних проєктів [9].

Порівняльний аналіз показує, що найбільш доцільною платформою для реалізації афіші розважальних заходів є Telegram [5]. Серед його переваг слід виділити відкритий та добре документований Telegram Bot API, безкоштовне створення ботів, наявність великої кількості бібліотек для різних мов програмування, підтримку інтерактивних кнопок, меню, повідомлень різних типів та можливість інтеграції з базами даних і зовнішніми вебсервісами [6]. Крім того, Telegram є одним із найбільш популярних месенджерів в Україні, що забезпечує зручний доступ потенційних користувачів до розроблюваного сервісу без необхідності встановлення додаткового програмного забезпечення. Завдяки цим перевагам Telegram є оптимальним вибором для реалізації телеграм-боту афіші розважальних заходів м. Миколаєва.

На сьогодні існує значна кількість програмних рішень для створення телеграм-ботів, які відрізняються функціональними можливостями, складністю

розробки, вартістю використання та рівнем гнучкості налаштування. Умовно такі рішення можна поділити на три основні групи: конструктори ботів без програмування, спеціалізовані платформи для автоматизації бізнес-процесів та програмні фреймворки для повноцінної розробки ботів.

До першої групи належать конструктори ботів, такі як ManyChat [10], SendPulse [11] та Chatfuel [12]. Вони надають графічний інтерфейс для створення сценаріїв взаємодії з користувачем без необхідності написання програмного коду. Основними перевагами таких систем є швидкість створення бота та простота налаштування. Водночас вони мають обмежені можливості щодо реалізації складної бізнес-логіки, інтеграції з зовнішніми джерелами даних та розширення функціоналу. Крім того, використання деяких функцій може вимагати придбання платної підписки.

До другої групи належать платформи автоматизації, зокрема Make [13] (Integromat) та Zapier [14]. Вони дозволяють створювати автоматизовані сценарії обміну даними між Telegram та іншими інформаційними системами. Такі рішення забезпечують швидку інтеграцію з різноманітними сервісами, проте їх використання є доцільним переважно для реалізації простих інформаційних сервісів. При збільшенні кількості користувачів або ускладненні функціональних вимог вартість експлуатації таких платформ може суттєво зрости.

Найбільш гнучким підходом є використання програмних фреймворків та бібліотек для розробки телеграм-ботів. Серед найбільш поширених рішень можна виділити бібліотеки Telegram Bot API для мов програмування Python, JavaScript, Java та PHP. Використання таких інструментів дозволяє реалізувати довільну логіку роботи системи, інтегрувати бази даних, вебсервіси та зовнішні інформаційні ресурси. Розробник отримує повний контроль над архітектурою програмного забезпечення та може адаптувати його до конкретних вимог предметної області.

Проведений аналіз показав, що існуючі конструктори та платформи дозволяють швидко створювати прості телеграм-боти, однак мають обмеження щодо масштабованості та функціональності. Для забезпечення необхідної

гнучкості, можливості супроводу та подальшого розвитку системи доцільно реалізувати телеграм-бот афіші розважальних заходів як окремий програмний продукт із власною серверною частиною та базою даних.

Для реалізації телеграм-боту афіші розважальних заходів м. Миколаєва необхідна підтримка зберігання інформації про події, її пошуку за різними критеріями, адміністрування контенту та можливості подальшого розширення функціоналу. Зазначені вимоги важко реалізувати за допомогою конструкторів або платформ автоматизації без значних обмежень. Тому найбільш доцільним рішенням є розробка власного програмного продукту на основі Telegram Bot API із використанням сучасних засобів програмування та систем керування базами даних.

1.3 Постановка задачі на розробку телеграм-боту афіші розважальних заходів м. Миколаєва

На основі проведеного аналізу предметної області та існуючих програмних рішень було сформульовано функціональні та нефункціональні вимоги до телеграм-боту афіші розважальних заходів м. Миколаєва. Визначення вимог є важливим етапом розробки програмного забезпечення, оскільки дозволяє встановити перелік функцій системи та критерії якості її роботи.

Функціональні вимоги

Функціональні вимоги визначають набір можливостей, які повинна забезпечувати система для користувачів та адміністраторів.

Телеграм-бот повинен забезпечувати виконання таких функцій:

1. Надання користувачеві інформації про найближчі розважальні заходи м. Миколаєва.

2. Відображення списку заходів із зазначенням назви, дати, часу та місця проведення.

3. Перегляд детальної інформації про вибраний захід, включаючи опис, адресу проведення та контактні дані організаторів.

4. Пошук заходів за категоріями (концерти, вистави, фестивалі, спортивні події, виставки тощо).

5. Пошук заходів за датою проведення.

6. Відображення актуальної афіші на поточний день або визначений період часу.

7. Надання користувачеві посилань на офіційні ресурси заходів або сторінки продажу квитків.

8. Можливість перегляду інформації про місце проведення заходу.

9. Відображення головного меню та навігаційних кнопок для спрощення взаємодії користувача із системою.

10. Реєстрація та обробка команд користувача через інтерфейс Telegram.

11. Збереження інформації про заходи у базі даних.

12. Додавання нових заходів адміністратором системи.

13. Редагування відомостей про існуючі заходи.

14. Видалення неактуальних або скасованих заходів.

15. Забезпечення авторизованого доступу адміністратора до функцій керування контентом.

16. Формування відповідей користувачам відповідно до введених запитів.

17. Надання повідомлень про помилки або некоректно введені команди.

Нефункціональні вимоги

Нефункціональні вимоги визначають характеристики якості програмного забезпечення та обмеження щодо його функціонування.

До основних нефункціональних вимог належать:

1. Продуктивність. Час формування відповіді на запит користувача не повинен перевищувати декількох секунд за умови нормального навантаження на систему.

2. Надійність. Система повинна забезпечувати коректну роботу навіть у разі виникнення помилок введення або тимчасової недоступності окремих зовнішніх сервісів.

3. Доступність. Телеграм-бот повинен бути доступним для користувачів цілодобово через платформу Telegram.

4. Зручність використання. Інтерфейс взаємодії має бути зрозумілим для користувачів без спеціальної підготовки та не вимагати вивчення складних інструкцій.

5. Масштабованість. Архітектура програмного забезпечення повинна забезпечувати можливість збільшення кількості користувачів та обсягу інформації про заходи без суттєвої зміни структури системи.

6. Супроводжуваність. Програмний код повинен мати модульну структуру, що спрощує внесення змін та подальший розвиток програмного продукту.

7. Безпека. Система повинна забезпечувати захист адміністративних функцій від несанкціонованого доступу та безпечне зберігання даних.

8. Сумісність. Телеграм-бот повинен коректно функціонувати на всіх пристроях, що підтримують офіційний клієнт Telegram.

9. Портативність. Серверна частина системи повинна мати можливість розгортання на різних операційних системах та хостингових платформах.

10. Актуальність даних. Інформація про заходи повинна своєчасно оновлюватися та відповідати фактичному стану подій.

Таким чином, визначені функціональні та нефункціональні вимоги формують основу для проєктування архітектури телеграм-боту афіші розважальних заходів м. Миколаєва та безпосередньо розробки телеграм-боту афіші розважальних заходів м. Миколаєва.

Технічне завдання на розробку телеграм-боту афіши розважальних заходів м. Миколаєва наведено у Додатку А.

2 РОЗРОБКА ТЕЛЕГРАМ-БОТУ АФІШИ РОЗВАЖАЛЬНИХ ЗАХОДІВ М. МИКОЛАЄВА

2.1 Аналіз вимог до телеграм-боту афіши розважальних заходів м. Миколаєва

2.1.1 Діаграма варіантів використання телеграм-боту

Для аналізу вимог до програмного забезпечення доцільно використовувати діаграму варіантів використання (Use Case Diagram), яка відображається засобами мови моделювання UML. Діаграма варіантів використання призначена для графічного відображення функціональних можливостей системи та взаємодії користувачів із нею. Вона дозволяє визначити основних акторів системи, перелік їхніх дій та межі функціонування програмного продукту. Використання такої діаграми на етапі аналізу вимог сприяє кращому розумінню потреб користувачів, виявленню основних сценаріїв роботи системи та формуванню вимог до її реалізації [15].

У телеграм-боті афіши розважальних заходів м. Миколаєва, що розробляється в рамках кваліфікаційної роботи, можна виділити два основні актори:

– Користувач – особа, яка використовує телеграм-бот для отримання інформації про розважальні заходи;

– Адміністратор – особа, яка здійснює наповнення та підтримку актуальності інформаційної бази заходів.

Для актора Користувач передбачені такі варіанти використання:

1. Запуск телеграм-боту.
2. Перегляд головного меню.
3. Перегляд списку найближчих заходів.
4. Перегляд заходів за категоріями.
5. Пошук заходів за датою.

6. Перегляд детальної інформації про захід.

7. Перегляд місця проведення заходу.

8. Отримання посилання на офіційний ресурс або сторінку продажу квитків.

9. Отримання довідкової інформації про роботу бота.

Для актора Адміністратор передбачені такі варіанти використання:

1. Авторизація в системі адміністрування.

2. Перегляд переліку заходів, що зберігаються в базі даних.

3. Додавання нового заходу вручну.

4. Редагування інформації про захід.

5. Видалення заходу.

6. Налаштування джерел даних для отримання інформації про події.

7. Додавання нового джерела даних.

8. Редагування параметрів джерела даних.

9. Видалення джерела даних.

10. Запуск імпорту інформації із зовнішніх джерел.

11. Перегляд імпортованих заходів.

12. Модерація імпортованих заходів перед публікацією.

13. Перегляд статистики використання телеграм-боту.

Діаграма варіантів використання телеграм-боту афіші розважальних заходів м. Миколаєва для актора Користувач наведена на рисунку 2.1.

На діаграмі варіантів використання користувача відображено основні функціональні можливості телеграм-боту афіші розважальних заходів м. Миколаєва. З попередніх варіантів використання для Користувача було прибрано такі варіанти використання: 1. Запуск телеграм-боту та 2. Перегляд головного меню, оскільки це технічні дії, а не бізнес-функції. Також було зроблено один головний варіант використання «Переглянути інформацію про захід», який буде включатися з різних способів пошуку.

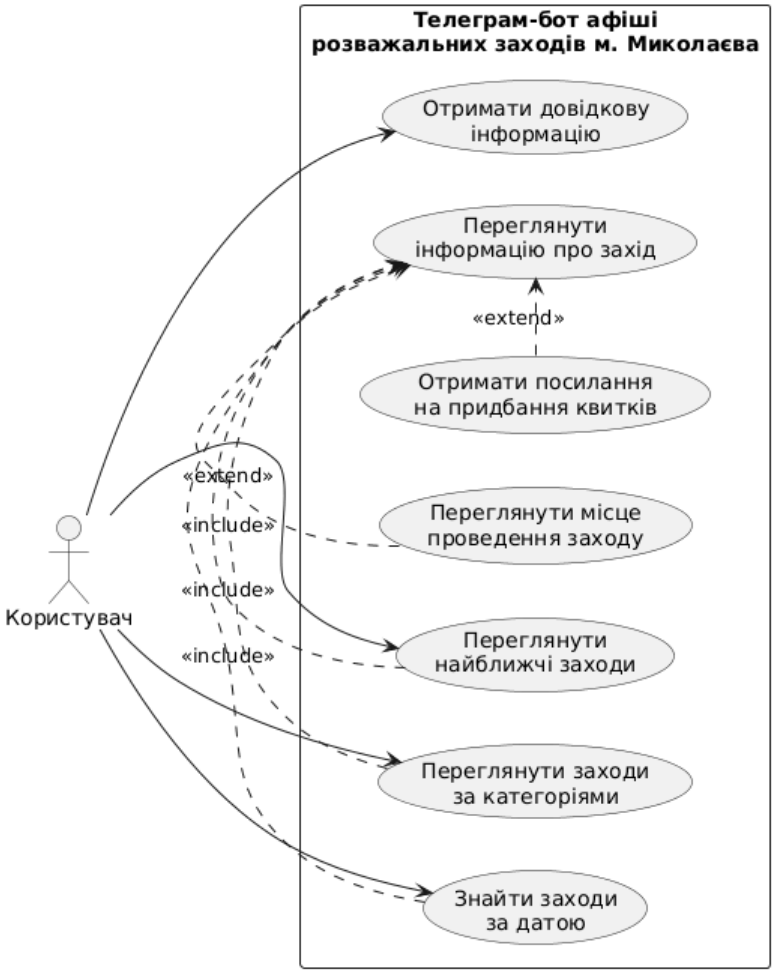


Рисунок 2.1 – Діаграма варіантів використання телеграм-боту афіші розважальних заходів м. Миколаєва для актора Користувач

Користувач може переглядати найближчі заходи, здійснювати пошук подій за категоріями або датою проведення, а також отримувати довідкову інформацію щодо роботи сервісу. Результатом виконання операцій пошуку є перегляд детальної інформації про обраний захід, тому відповідні варіанти використання пов'язані відношенням «include». Під час перегляду інформації про захід користувач додатково може переглянути місце проведення події або перейти за посиланням для придбання квитків, що відображено за допомогою відношення «extend». Така структура діаграми дозволяє наочно представити основні сценарії взаємодії користувача з телеграм-ботом.

Діаграма варіантів використання телеграм-боту афіші розважальних заходів м. Миколаєва для актора Адміністратор наведена на рисунку 2.2.

Рисунок 2.2 – Діаграма варіантів використання телеграм-боту афіші розважальних заходів м. Миколаєва для актора Адміністратор

Таким чином, Адміністратор забезпечує як ручне наповнення бази даних заходів, так і автоматизоване отримання інформації із зовнішніх джерел, що дозволяє підтримувати актуальність афіші та зменшити трудомісткість супроводу системи. Налаштувати джерела даних включає операції додавання, редагування та видалення джерел. Модерувати імпортовані заходи є необов'язковим розширенням перегляду імпортованих даних. Усі адміністративні операції вимагають попередньої авторизації.

2.1.2 Специфікації варіантів використання телеграм-боту

Виходячи з аналізу діаграм варіантів використання для Користувача та Адміністратора, маємо загалом 20 варіантів використання. Наведемо специфікації варіантів використання для шести з них.

Для користувача:

UC-01 Перегляд найближчих заходів

Основний сценарій роботи бота. Через нього користувач фактично знайомиться з афішею.

UC-02 Пошук заходів за категорією

Демонструє механізм фільтрації та роботи з даними.

UC-03 Перегляд інформації про захід

Центральний сценарій, який використовується після пошуку або перегляду списку заходів.

Для адміністратора:

UC-04 Додавання заходу

Класична CRUD-операція та один із головних сценаріїв адміністрування.

UC-05 Імпорт заходів із зовнішніх джерел

Сценарій, який відображає особливість отримання даних із зовнішніх джерел інформації.

UC-06 Модерація імпортованих заходів

Показує взаємодію між автоматичним імпортом та власною базою даних.

Специфікації перелічених варіантів використання наведено у таблицях 2.1 – 2.6.

Таблиця 2.1 – Специфікація варіанту використання UC-01 Перегляд найближчих заходів

Поле	Опис
Ідентифікатор	UC-01
Назва	Перегляд найближчих заходів
Актор	Користувач
Передумови	Користувач відкрив телеграм-бот
Основний сценарій	1. Користувач обирає пункт «Найближчі заходи». 2. Бот отримує дані із бази даних. 3. Бот формує список найближчих подій. 4. Бот відображає список заходів користувачу.
Альтернативний сценарій	У базі даних відсутні актуальні заходи. Бот виводить відповідне повідомлення.
Післяумови	Користувач отримав перелік найближчих заходів.

Таблиця 2.2 – Специфікація варіанту використання UC-02 Пошук заходів за категорією

Поле	Опис
Ідентифікатор	UC-02
Назва	Пошук заходів за категорією
Актор	Користувач
Передумови	Користувач відкрив телеграм-бот
Основний сценарій	1. Користувач обирає пункт «Категорії». 2. Бот відображає перелік категорій. 3. Користувач обирає категорію. 4. Бот виконує пошук заходів у вибраній категорії. 5. Бот відображає результати пошуку.
Альтернативний сценарій	У вибраній категорії заходи відсутні. Бот повідомляє користувача про відсутність результатів.
Післяумови	Користувач отримав перелік заходів обраної категорії.

Таблиця 2.3 – Специфікація варіанту використання UC-03 Перегляд інформації про захід

Поле	Опис
Ідентифікатор	UC-03
Назва	Перегляд інформації про захід
Актор	Користувач
Передумови	Користувач отримав список заходів
Основний сценарій	1. Користувач обирає захід зі списку. 2. Бот отримує інформацію про захід із бази даних. 3. Бот відображає назву, опис, дату, час, місце проведення та посилання на квитки.
Альтернативний сценарій	Інформація про захід відсутня або була видалена. Бот виводить повідомлення про помилку.
Післяумови	Користувач отримав детальну інформацію про обраний захід.

Таблиця 2.4 – Специфікація варіанту використання UC-04 Додавання заходу

Поле	Опис
Ідентифікатор	UC-04
Назва	Додавання заходу
Актор	Адміністратор
Передумови	Адміністратор успішно авторизований у системі
Основний сценарій	1. Адміністратор обирає функцію додавання заходу. 2. Система відображає форму введення даних. 3. Адміністратор заповнює необхідні поля. 4. Система перевіряє коректність введених даних. 5. Інформація зберігається у базі даних.
Альтернативний сценарій	Введені некоректні або неповні дані. Система повідомляє про помилку та пропонує повторити введення.
Післяумови	Новий захід додано до бази даних.

Таблиця 2.5 – Специфікація варіанту використання UC-05 Імпорт заходів із зовнішніх джерел

Поле	Опис
1	2
Ідентифікатор	UC-05
Назва	Імпорт заходів із зовнішніх джерел
Актор	Адміністратор
Передумови	Адміністратор авторизований, джерела даних налаштовані
Основний сценарій	1. Адміністратор запускає процедуру імпорту. 2. Система звертається до зовнішніх джерел даних.

Кінець таблиці 2.5

1	2
	3. Система отримує інформацію про заходи. 4. Отримані дані зберігаються у тимчасовому сховищі для перевірки.
Альтернативний сценарій	Джерело даних недоступне або виникла помилка з'єднання. Система реєструє помилку та повідомляє адміністратора.
Післяумови	Імпортовані заходи доступні для перегляду та модерації.

Таблиця 2.6 – Специфікація варіанту використання UC-06 Модерація імпортованих заходів

Поле	Опис
Ідентифікатор	UC-06
Назва	Модерація імпортованих заходів
Актор	Адміністратор
Передумови	Виконано імпорт заходів із зовнішніх джерел
Основний сценарій	1. Адміністратор переглядає список імпортованих заходів. 2. Адміністратор перевіряє коректність інформації. 3. Адміністратор підтверджує або відхиляє кожний захід. 4. Підтверджені заходи додаються до основної бази даних.
Альтернативний сценарій	Виявлено дублікати або некоректні дані. Адміністратор відхиляє відповідні записи.
Післяумови	У базі даних залишаються лише перевірені та актуальні заходи.

Розроблені специфікації варіантів використання покривають усі ключові бізнес-процеси телеграм-боту афіші розважальних заходів м. Миколаєва.

2.2 Архітектура телеграм-боту афіши розважальних заходів м. Миколаєва

Архітектурну структуру програмної системи та взаємозв’язки між її основними модулями відображає діаграма компонентів. Вона дозволяє визначити фізичні частини системи, їх відповідальність та взаємодію [16].

У розроблюваній системі виділено такі основні компоненти:

– клієнтська частина у вигляді Telegram Bot API;

– серверна частина;

– база даних;

– модуль імпорту даних із зовнішніх джерел;

– адміністративний модуль керування даними.

Програмна система також взаємодіє із зовнішніми інформаційними джерелами, з яких отримуються відомості про розважальні заходи. Діаграму компонентів телеграм-боту афіши розважальних заходів м. Миколаєва представлено на рисунку 2.3.

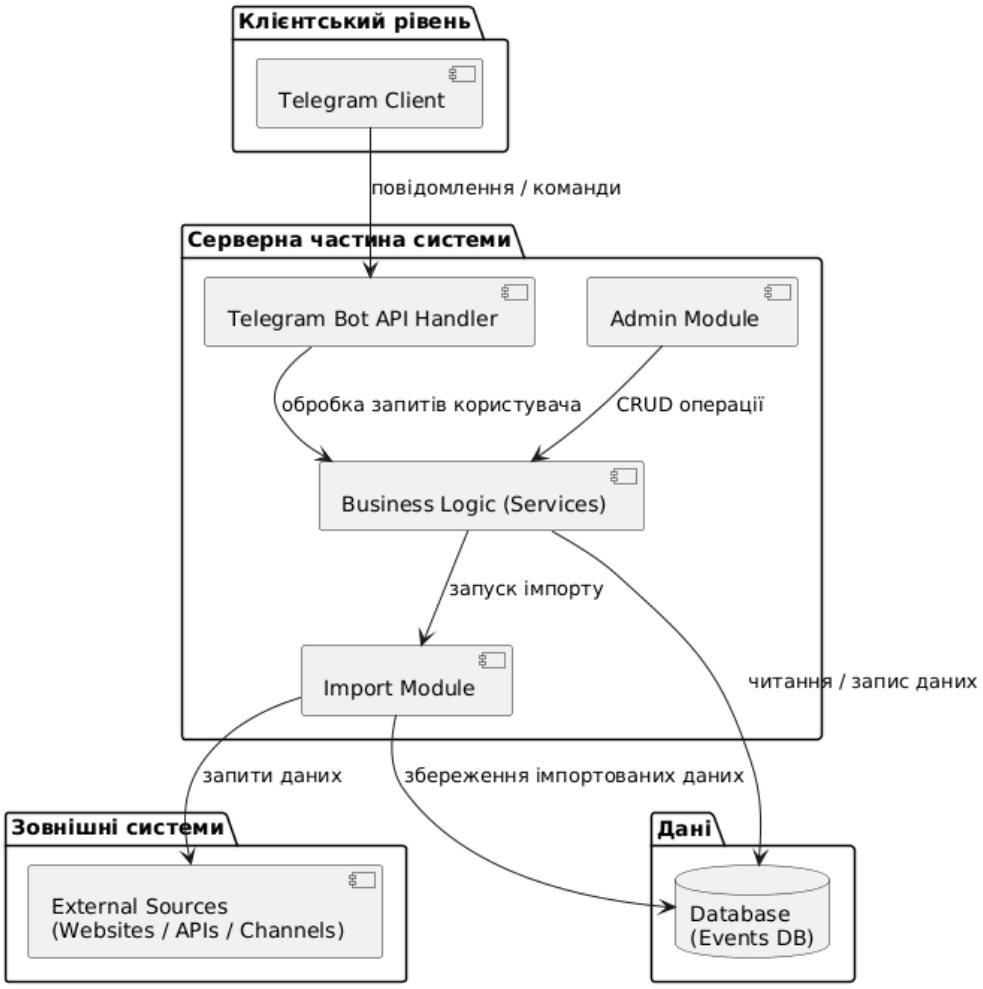


Рисунок 2.3 – Діаграма компонентів телеграм-боту афіши розважальних заходів м. Миколаєва

Діаграма розгортання відображає фізичну структуру програмної системи та спосіб її розміщення на апаратних або хмарних вузлах. Вона демонструє

взаємодію між клієнтськими пристроями користувачів, серверною частиною телеграм-боту, базою даних та зовнішніми інформаційними джерелами [17].

У розроблюваній програмній системі користувачі взаємодіють із телеграм-ботом через мобільні або десктопні пристрої з встановленим месенджером Telegram. Запити надходять до хмарного або серверного середовища, де розгорнуто сервер додатка, що містить логіку обробки повідомлень, модуль адміністрування та модуль імпорту даних.

База даних розміщується на окремому сервері або в хмарному сховищі для забезпечення надійності та масштабованості. Модуль імпорту взаємодіє із зовнішніми джерелами даних через HTTP-запити до вебсайтів, API або інформаційних каналів. Така архітектура забезпечує гнучкість, масштабованість і незалежність від конкретного апаратного забезпечення.

Діаграму розгортання телеграм-боту афіши розважальних заходів м. Миколаєва представлено на рисунку 2.4.

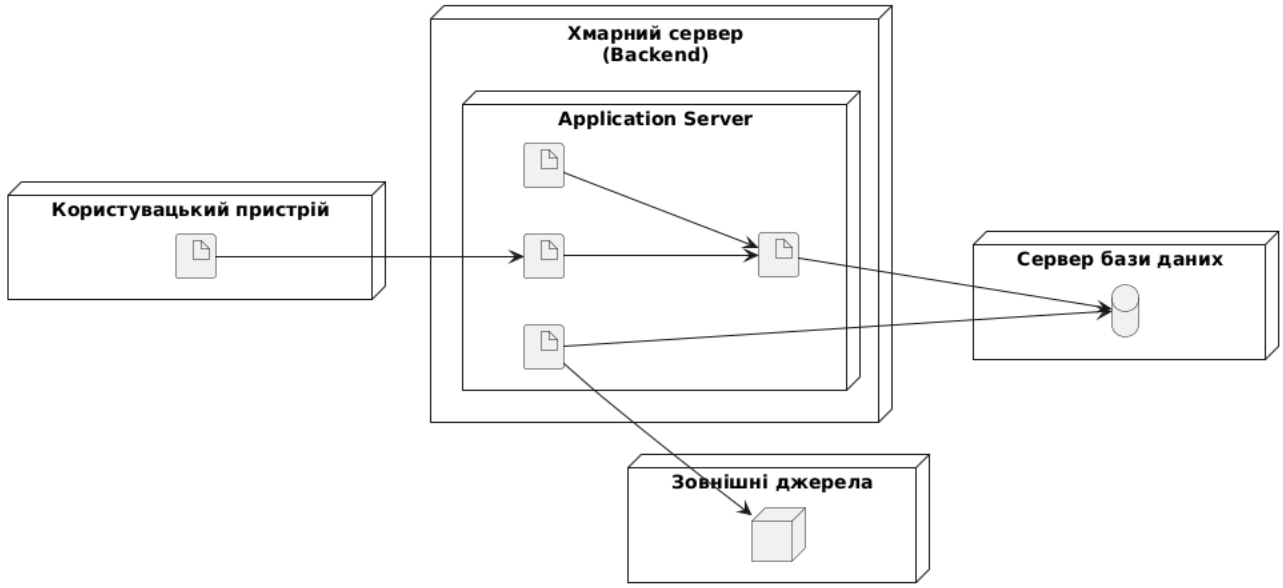


Рисунок 2.4 – Діаграма розгортання телеграм-боту афіши розважальних заходів м. Миколаєва

Із представленої діаграми розгортання видно, що Telegram працює як клієнтська платформа; бот – це серверний застосунок; база даних винесена

окремо, що додає масштабованість програмній системі; імпорт працює через зовнішні API або сайти; адміністративна частина представляє собою не окремий сервер, а модуль бекенду.

2.3 Проект телеграм-боту афіши розважальних заходів м. Миколаєва

2.3.1 Ескізний проект телеграм-боту

Сценарії варіантів використання

Після діаграми варіантів використання та специфікацій наступним кроком проєктування телеграм-боту афіши розважальних заходів м. Миколаєва будуємо діаграми діяльності для ключових бізнес-процесів системи. Діаграму діяльності для варіанту використання «Пошук та перегляд інформації про захід» наведено на рисунку 2.5.

Наведена діаграма діяльності відображає послідовність дій, які виконуються користувачем та системою під час пошуку розважального заходу. Процес починається із запуску бота користувачем, після чого користувач обирає спосіб пошуку заходів. Система виконує пошук у базі даних та відображає знайдені результати. У разі наявності заходів користувач може переглянути детальну інформацію про обрану подію. Якщо заходів не знайдено, система повідомляє про це користувача та завершує процес.

Далі побудуємо діаграму діяльності для варіанту використання «Імпорт та модерація заходів із зовнішніх джерел», оскільки саме вона відображає особливість обраної архітектури (власна БД + зовнішні джерела інформації). Діаграму наведено на рисунку 2.6.

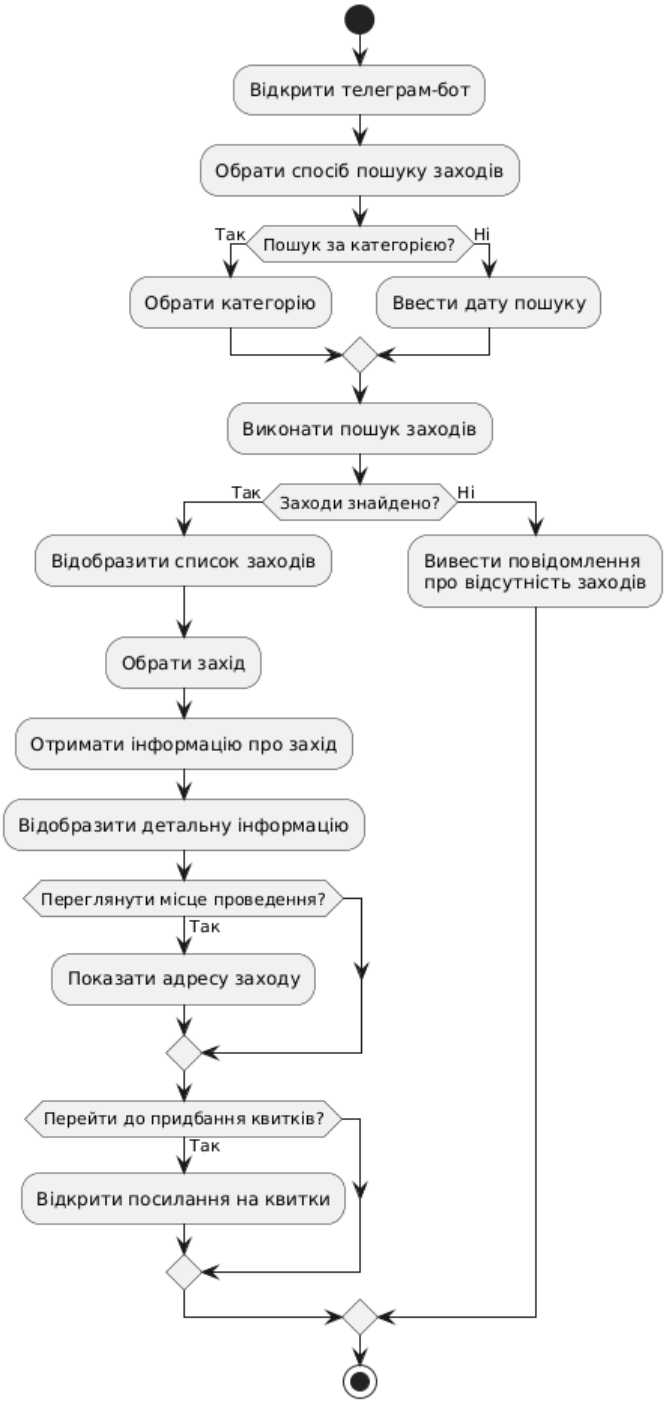


Рисунок 2.5 – Діаграма діяльності для варіанту використання «Пошук та перегляд інформації про захід»

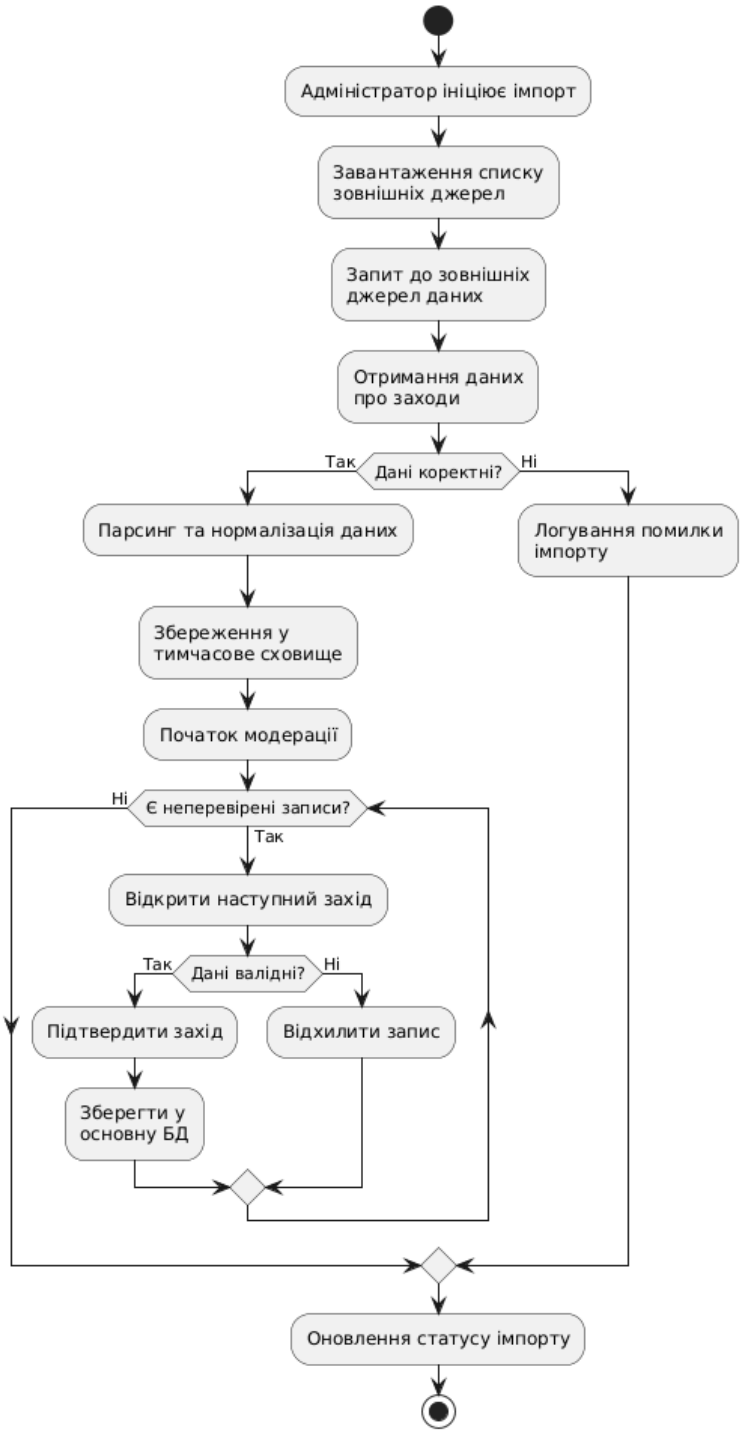


Рисунок 2.6 – Діаграма діяльності для варіанту використання «Імпорт та модерація заходів із зовнішніх джерел»

Наведена діаграма діяльності відображає процес автоматизованого отримання інформації про розважальні заходи з зовнішніх інформаційних ресурсів, її первинної обробки та подальшої перевірки адміністратором.

Процес починається із ініціації імпорту адміністратором системи. Далі система звертається до налаштованих зовнішніх джерел даних (вебсайти, сторінки подій, канали або API) та виконує отримання інформації про заходи. Отримані дані проходять етап парсингу та попередньої обробки, після чого зберігаються у тимчасовому сховищі (черзі імпорту).

Наступним етапом є модерація імпортованих даних адміністратором. Адміністратор переглядає кожен захід, перевіряє коректність інформації, усуває дублікати та некоректні записи. У разі підтвердження захід переноситься до основної бази даних системи. Якщо запис є помилковим або неактуальним, він відхиляється та видаляється з черги імпорту.

Завершення процесу відбувається після обробки всіх імпортованих записів. Таким чином, забезпечується баланс між автоматизацією збору даних та контролем їх якості з боку адміністратора.

Інформаційна модель

Інформаційну модель телеграм-боту афіши розважальних заходів м. Миколаєва представлено ER-діаграмою, яка призначена для формалізованого опису структури бази даних та взаємозв’язків між основними сутностями предметної області. Вона дозволяє визначити, які дані зберігаються в системі, як вони структуровані та яким чином взаємодіють між собою.

Інформаційну модель телеграм-боту наведено на рисунку 2.7.

У розроблюваній системі телеграм-боту афіші розважальних заходів м. Миколаєва основними сутностями є «Захід», «Категорія», «Джерело даних», «Імпортований захід» та «Адміністратор». Сутність «Захід» є центральною та містить основну інформацію про події: назву, опис, дату, час та місце проведення.

Кожен захід належить до однієї категорії, що дозволяє виконувати фільтрацію та пошук подій. Джерела даних використовуються для автоматизованого імпорту інформації із зовнішніх ресурсів. Імпортовані заходи проходять етап перевірки та модерації перед тим, як потрапити до основної бази

даних. Адміністратор здійснює керування контентом системи, включаючи додавання, редагування та видалення даних.

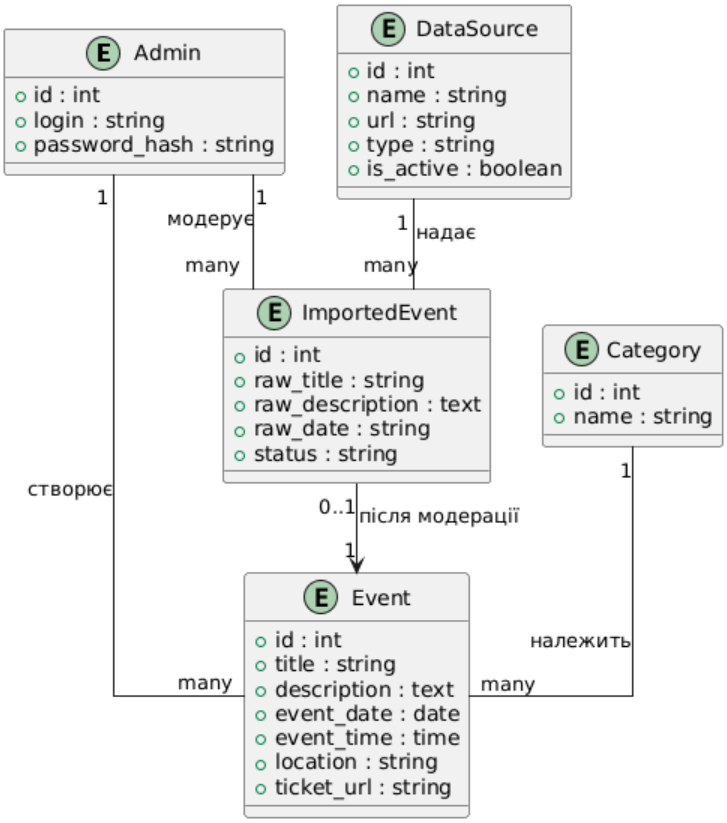


Рисунок 2.7 – Інформаційна модель телеграм-боту афіши розважальних заходів м. Миколаєва

Побудована інформаційна модель добре відображає гібридну архітектуру програмної системи, де є власна БД, зовнішні джерела інформації та етап модерації, а також логіку системи, де є перевірені дані (Event) та чернетка або «сирі дані» (ImportedEvent).

У системі реалізовано розділення даних на імпортовані та підтверджені заходи. Імпортовані записи проходять етап модерації адміністратором і лише після перевірки потрапляють до основної таблиці заходів, що забезпечує контроль якості даних.

Інтерфейс програмного забезпечення

UX-частина програмної системи телеграм-боту афіші розважальних заходів м. Миколаєва визначає структуру взаємодії користувача з системою через інтерфейс месенджера Telegram. Оскільки Telegram-бот не має графічного інтерфейсу у класичному розумінні, взаємодія реалізується через текстові повідомлення, кнопки та інтерактивні меню.

Основною метою UX-дизайну є забезпечення простоти навігації, мінімальної кількості кроків для отримання інформації та інтуїтивного доступу до функцій системи. Інтерфейс побудований на основі головного меню з можливістю перегляду найближчих заходів, пошуку за категоріями та перегляду детальної інформації про події.

Для адміністратора передбачено окремий набір команд для керування заходами та запуску імпорту даних. Таким чином, UX-структура забезпечує розділення функцій користувача та адміністратора і підвищує зручність використання системи.

Прототип екрану з головним меню телеграм-боту афіші розважальних заходів м. Миколаєва наведено на рисунку 2.8.

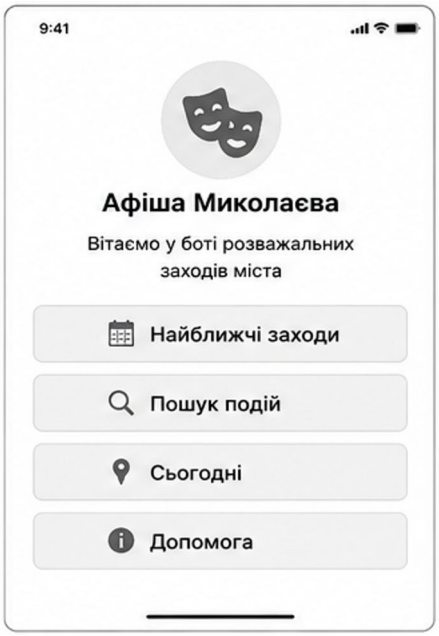


Рисунок 2.8 – Прототип екрану з головним меню телеграм-боту афіші розважальних заходів м. Миколаєва

Прототип екрану панелі адміністратора телеграм-боту афіші розважальних заходів м. Миколаєва наведено на рисунку 2.9.

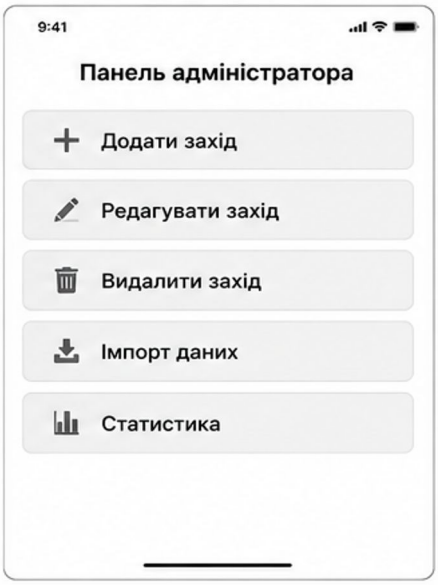


Рисунок 2.9 – Прототип екрану панелі адміністратора телеграм-боту афіші розважальних заходів м. Миколаєва

UI системи побудований на компонентному підході з використанням повторно використаних каркасних компонентів, що дозволяє прискорити проектування інтерфейсу та забезпечити однорідність користувацького досвіду.

2.3.2 Технічний проект телеграм-боту

Діаграма класів

Діаграма класів відображає логічну структуру програмної системи телеграм-боту афіші розважальних заходів м. Миколаєва та взаємозв’язки між основними класами. Вона демонструє об’єктно-орієнтовану модель реалізації програмної системи, включаючи класи для обробки запитів користувачів, роботи з базою даних, управління заходами та імпорту даних із зовнішніх джерел.

Діаграму класів телеграм-боту афіші розважальних заходів м. Миколаєва , наведено на рисунку 2.10.

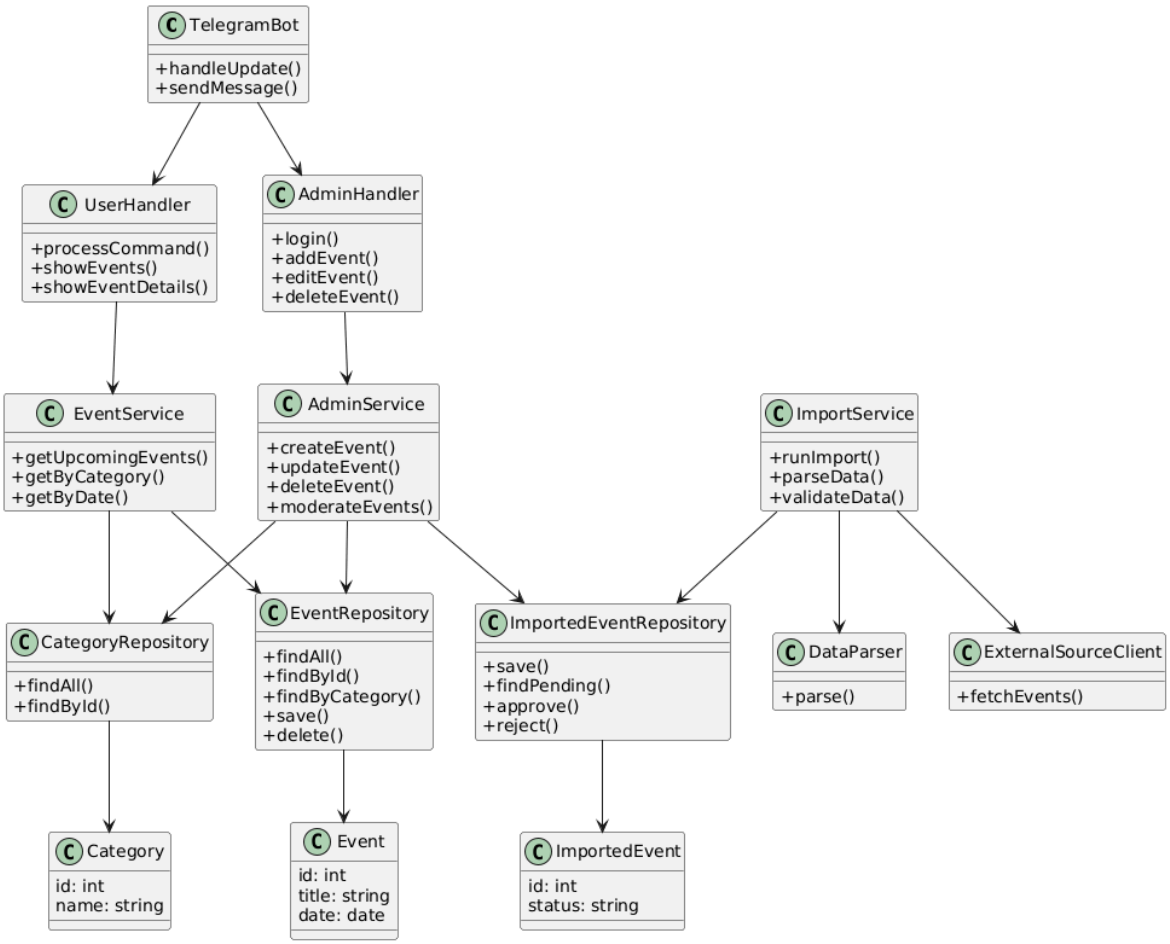


Рисунок 2.10 – Діаграма класів телеграм-боту афіші розважальних заходів м. Миколаєва

Основними класами системи є клас TelegramBot, який відповідає за взаємодію з Telegram API, клас EventService, який реалізує бізнес-логіку роботи з подіями, клас EventRepository для доступу до бази даних, а також класи ImportService та ExternalSourceClient, які забезпечують отримання та обробку даних із зовнішніх джерел.

Адміністративна частина реалізована через клас AdminService, який забезпечує функції додавання, редагування та видалення заходів, а також модерацію імпортованих даних. Така структура забезпечує розділення відповідальностей та спрощує супровід і масштабування системи.

Категорії виділені в окремий репозиторій як довідникова сутність для забезпечення можливості розширення системи та незалежного управління класифікацією заходів.

Специфікації основних класів

1. Клас TelegramBot

Призначення: забезпечує взаємодію між користувачем та системою через Telegram API.

Основні методи:

– handleUpdate() – обробка вхідного повідомлення;

– sendMessage() – відправка повідомлення користувачу;

– registerHandlers() – реєстрація обробників команд.

2. Клас UserHandler

Призначення: обробка команд користувача та формування запитів до сервісного шару.

Основні методи:

– processCommand() – визначення типу запиту;

– showEvents() – отримання списку заходів;

– showEventDetails() – перегляд детальної інформації;

– searchEvents() – пошук заходів.

3. Клас AdminHandler

Призначення: обробка адміністративних команд та керування даними.

Основні методи:

– login() – авторизація;

– addEvent() – додавання заходу;

– editEvent() – редагування заходу;

– deleteEvent() – видалення заходу;

– moderateImportedEvents() – модерація імпорту.

4. Клас EventService

Призначення: реалізація бізнес-логіки роботи з заходами.

Основні методи:

– getUpcomingEvents() – найближчі заходи;

– getByCategory() – фільтр за категорією;

– getByDate() – пошук за датою;

– getEventById() – отримання конкретного заходу.

5. Клас AdminService

Призначення: реалізація бізнес-логіки адміністрування системи.

Основні методи:

– createEvent() – створення заходу;

– updateEvent() – оновлення даних;

– deleteEvent() – видалення заходу;

– moderateEvent() – підтвердження імпортованих даних.

6. Клас ImportService

Призначення: забезпечення імпорту даних із зовнішніх джерел.

Основні методи:

– runImport() – запуск процесу імпорту;

– parseData() – обробка отриманих даних;

– validateData() – перевірка коректності;

– saveImportedEvents() – збереження результатів.

7. Клас EventRepository

Призначення: доступ до бази даних заходів.

Основні методи:

– findAll() – отримання всіх заходів;

– findById() – пошук за ID;

– findByCategory() – фільтрація;

– save() – збереження;

– delete() – видалення.

8. Клас ImportedEventRepository

Призначення: управління імпортованими (неперевіреними) заходами.

Основні методи:

– save() – збереження імпортованого запису;

– findPending() – отримання неперевірених;

– approve() – підтвердження;

– reject() – відхилення.

9. Клас CategoryRepository

Призначення: управління довідником категорій заходів.

Основні методи:

– findAll() – всі категорії;

– findById() – пошук категорії.

Діаграми послідовності

Діаграма послідовності відображає взаємодію між об’єктами системи у часі під час виконання варіанту використання.

Діаграму послідовності для варіанту використання «Перегляд найближчих заходів», яка показує порядок обміну повідомленнями між користувачем, Telegram-ботом, сервісним шаром та базою даних, наведено на рисунку 2.11.

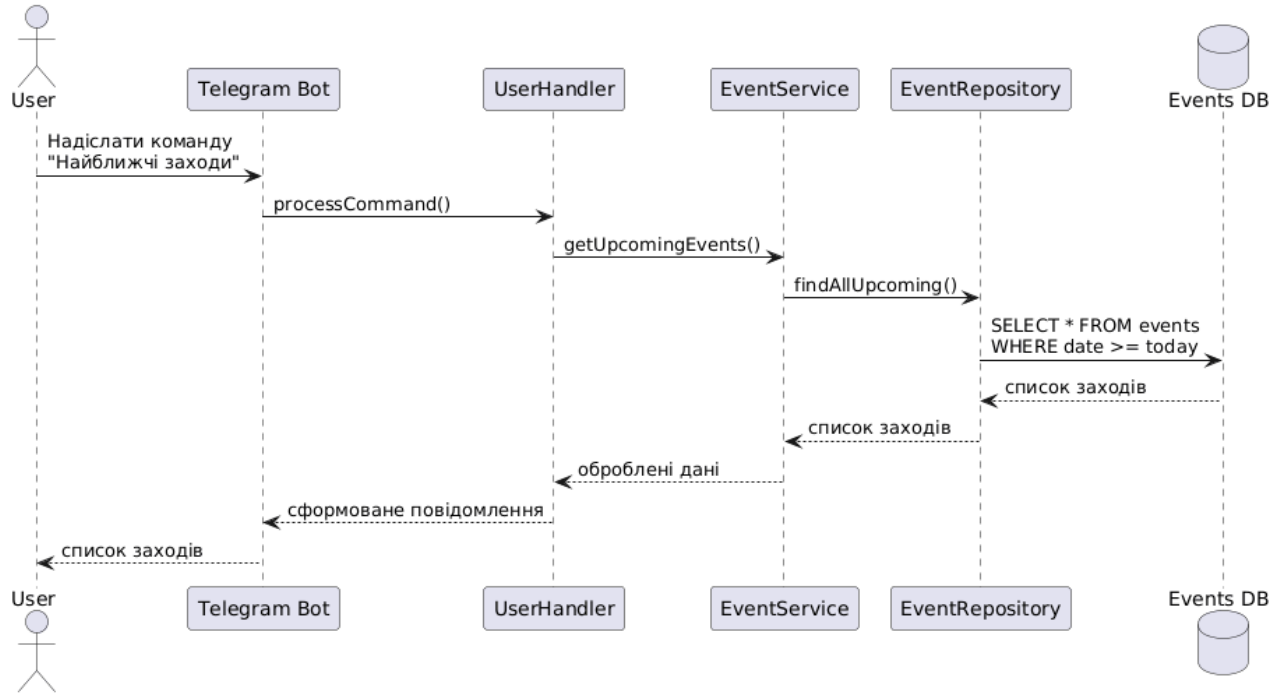


Рисунок 2.11 – Діаграма послідовності для варіанту використання «Перегляд найближчих заходів»

Процес починається з ініціювання запиту користувачем у Telegram. Запит надходить до обробника бот-системи, який передає його до сервісного рівня.

Сервіс виконує запит до бази даних через репозиторій, отримує список актуальних заходів та повертає його назад до бота. Після цього бот формує відповідь і надсилає її користувачу.

Діаграму послідовності для варіанту використання «Імпорт заходів із зовнішніх джерел», яка відображає взаємодію між компонентами системи під час автоматизованого отримання та обробки інформації про розважальні заходи, наведено на рисунку 2.12.

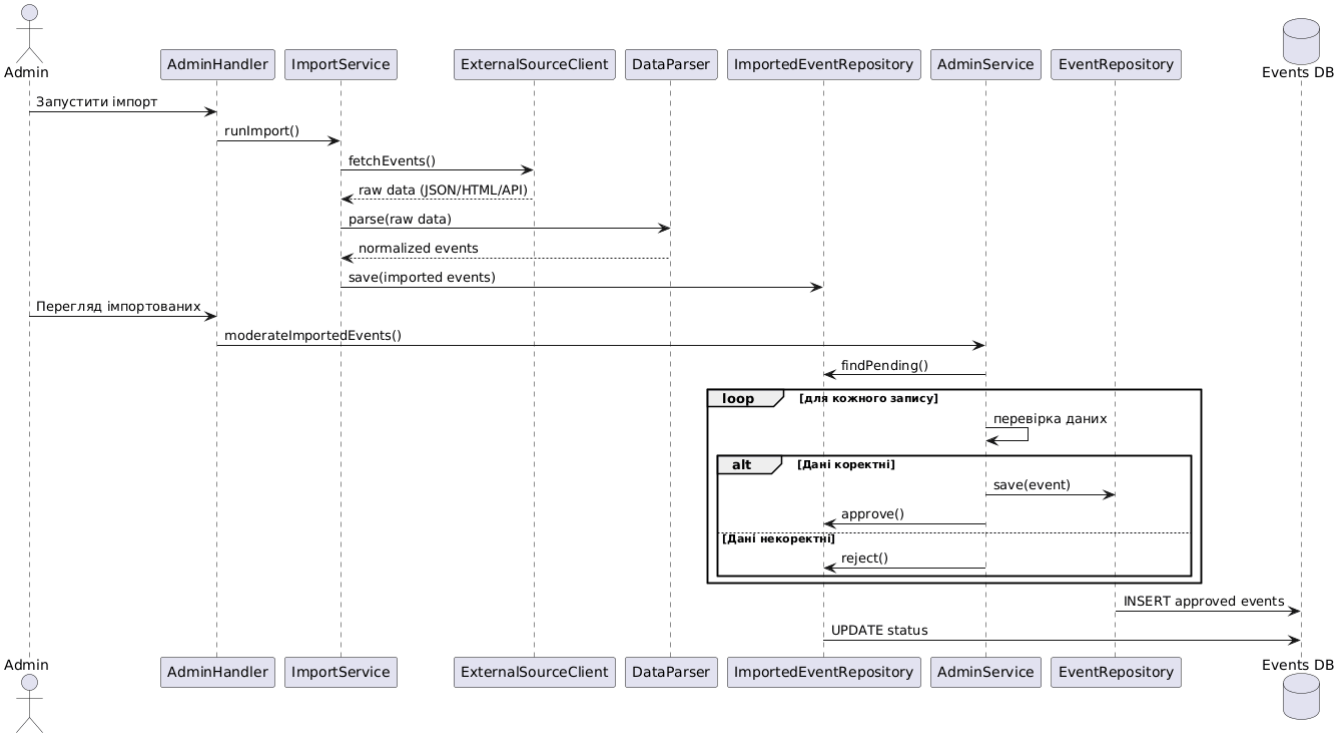


Рисунок 2.12 – Діаграма послідовності для варіанту використання «Перегляд найближчих заходів»

Процес ініціюється адміністратором, який запускає процедуру імпорту через адміністративний модуль. Далі система звертається до зовнішніх джерел даних через спеціалізований сервіс, отримує сирі дані про події та передає їх у модуль парсингу. Після цього дані нормалізуються та зберігаються у тимчасовому сховищі як імпортовані записи.

Наступним етапом є перевірка та модерація даних адміністратором. Після підтвердження записи переносяться до основної бази даних системи. У разі

відхилення некоректні дані видаляються або ігноруються. Такий підхід забезпечує контроль якості інформації та гнучкість інтеграції із зовнішніми ресурсами.

2.3.3 Робочий проект телеграм-боту

Обґрунтування вибору засобів програмної реалізації

Для реалізації телеграм-бота афіші розважальних заходів м. Миколаєва було обрано стек технологій, що включає мову програмування Python, фреймворк aiogram для розробки Telegram-ботів, систему керування базами даних MySQL, ORM-бібліотеку SQLAlchemy, а також бібліотеки Requests і Beautiful Soup для отримання та обробки інформації із зовнішніх джерел. Обраний стек є сучасним, безкоштовним, широко використовується у промисловій розробці та забезпечує можливість створення масштабованого програмного продукту.

Основною мовою програмування обрано Python. Дана мова характеризується простим синтаксисом, високою читабельністю програмного коду та великою кількістю бібліотек для розробки вебзастосунків, ботів і сервісів автоматизації. Важливою перевагою Python є підтримка асинхронного програмування, що дозволяє ефективно обробляти велику кількість одночасних запитів користувачів без суттєвого навантаження на сервер. Крім того, Python має розвинену спільноту розробників і добре документовані програмні інструменти, що спрощує процес створення та супроводу програмного забезпечення [18].

Для реалізації функціональності телеграм-бота було обрано фреймворк aiogram. Даний фреймворк є одним із найпопулярніших засобів розробки Telegram-ботів мовою Python та підтримує всі можливості Telegram Bot API. Фреймворк побудований на основі асинхронної моделі виконання, що забезпечує високу продуктивність та швидке реагування бота на дії користувачів. Серед переваг aiogram слід відзначити підтримку маршрутизації повідомлень, механізмів фільтрації команд, машин станів для реалізації складних сценаріїв взаємодії та модульну архітектуру застосунку. Це дозволяє організувати структуру програмного коду відповідно до сучасних принципів інженерії програмного забезпечення [19].

Для зберігання інформації про заходи, категорії, адміністраторів та результати імпорту обрано систему керування базами даних MySQL. Вибір MySQL обумовлений її високою продуктивністю, надійністю та широким поширенням у веброзробці. Система підтримує реляційну модель даних, механізми забезпечення цілісності інформації, індексування та ефективне виконання SQL-запитів. Важливою перевагою є також наявність безкоштовної Community-версії та підтримка більшістю сучасних інструментів розробки [20].

Для взаємодії програмного коду з базою даних використовується бібліотека SQLAlchemy. Вона реалізує підхід ORM (Object-Relational Mapping), який дозволяє працювати з даними у вигляді об’єктів мови програмування замість написання великої кількості SQL-запитів вручну. Використання ORM підвищує читабельність коду, спрощує супровід системи та зменшує ймовірність виникнення помилок при роботі з базою даних. Крім того, SQLAlchemy підтримує різні СУБД, що забезпечує можливість подальшого масштабування програмного продукту та перенесення його на інші платформи з мінімальними змінами програмного коду [21].

Однією з ключових функцій системи є отримання інформації про розважальні заходи із зовнішніх вебресурсів. Для реалізації даного функціоналу доцільно використовувати бібліотеки Requests та Beautiful Soup. Бібліотека Requests забезпечує зручне надсилання HTTP-запитів до вебсерверів та отримання HTML-вмісту сторінок. У свою чергу, Beautiful Soup дозволяє аналізувати HTML-документи, знаходити необхідні елементи та вилучати структуровану інформацію про події, дати проведення, місця проведення та посилання на джерела. Поєднання цих бібліотек дозволяє реалізувати механізм автоматичного імпорту заходів із зовнішніх джерел.

Фізична модель даних

Фізична модель відображає структуру таблиць реляційної бази даних, їх атрибути, первинні та зовнішні ключі, а також зв'язки між сутностями. Модель реалізована з використанням СУБД MySQL та забезпечує зберігання інформації

про заходи, категорії, адміністраторів, зовнішні джерела даних і результати імпорту заходів для подальшої модерації. Фізичну модель бази даних телеграм-бота афіші розважальних заходів м. Миколаєва наведено на рисунку 2.13

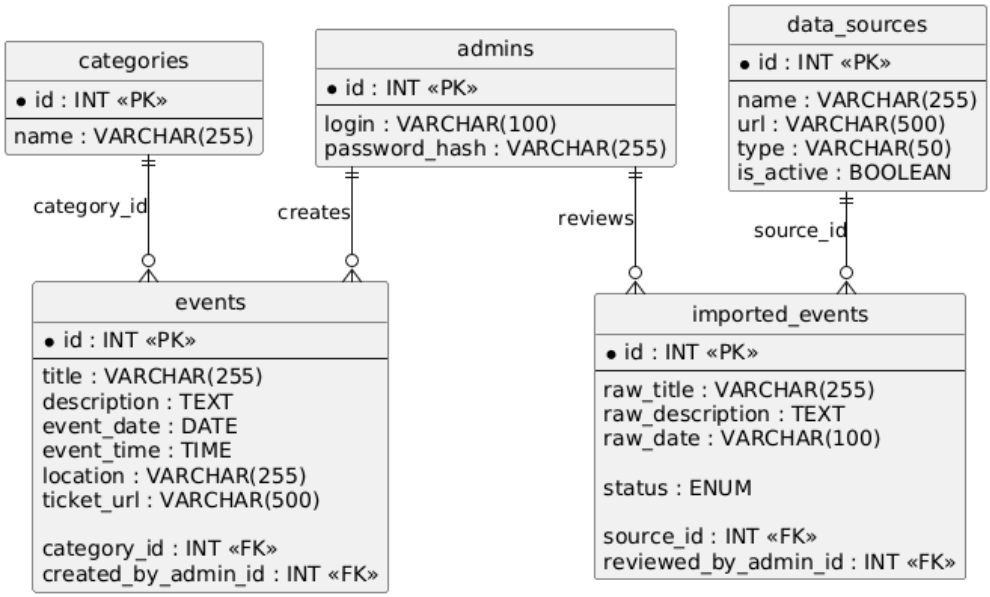


Рисунок 2.13 – Фізична модель бази даних телеграм-бота афіші розважальних заходів м. Миколаєва

Реалізація та тестування програмного забезпечення

Розробка телеграм-бота афіші розважальних заходів м. Миколаєва виконувалася із застосуванням мови програмування Python та фреймворку aiogram. Для зберігання даних використовувалася система керування базами даних MySQL, а взаємодія між програмним кодом та базою даних реалізована за допомогою ORM-бібліотеки SQLAlchemy. Отримання інформації із зовнішніх джерел здійснюється засобами бібліотек Requests та Beautiful Soup.

Програмне забезпечення реалізовано відповідно до багаторівневої архітектури, яка включає рівень представлення, рівень бізнес-логіки та рівень доступу до даних. Рівень представлення представлений інтерфейсом Telegram-бота, який забезпечує взаємодію користувачів та адміністраторів із системою через повідомлення та інтерактивні кнопки. Рівень бізнес-логіки відповідає за обробку запитів користувачів, формування результатів пошуку, управління

заходами та виконання процедур імпорту даних. Рівень доступу до даних забезпечує зберігання, пошук та оновлення інформації у базі даних.

У процесі реалізації було створено модулі керування заходами, категоріями, джерелами даних та адміністраторами системи. Для користувачів реалізовано функції перегляду найближчих заходів, пошуку за категоріями, перегляду детальної інформації про події та отримання актуальної афіші на обрану дату. Для адміністраторів реалізовано функції додавання, редагування та видалення заходів, запуску імпорту даних із зовнішніх джерел та модерації отриманої інформації перед її публікацією.

Після завершення реалізації було проведено тестування програмного забезпечення. Основною метою тестування була перевірка правильності роботи функціональних можливостей системи та виявлення можливих помилок у процесі взаємодії користувачів із телеграм-ботом. Тестування виконувалося методом функціонального тестування на основі розроблених варіантів використання.

У процесі тестування було перевірено коректність виконання таких сценаріїв:

– запуск бота та відображення головного меню;

– перегляд списку найближчих заходів;

– пошук заходів за категоріями;

– перегляд детальної інформації про захід;

– авторизація адміністратора;

– додавання нового заходу;

– редагування та видалення існуючих заходів;

– імпорт даних із зовнішніх джерел;

– модерація імпортованих заходів.

Для прикладу тестування розглянемо функцію Додавання нового заходу, оскільки вона містить кілька полів із різними обмеженнями. Розглянемо поле «Назва заходу». Вимоги до цього поля є такими:

– поле обов'язкове;

– довжина від 3 до 255 символів;

– допускаються літери, цифри, пробіли та розділові знаки;

– порожнє значення не допускається.

Класи еквівалентності для поля «Назва заходу» наведено у таблиці 2.7.

Таблиця 2.7 – Класи еквівалентності для поля «Назва заходу»

ID класу	Опис класу	Тип класу
EC1	Назва довжиною від 3 до 255 символів	Правильний
EC2	Порожнє значення	Неправильний
EC3	Назва довжиною менше 3 символів	Неправильний
EC4	Назва довжиною більше 255 символів	Неправильний

Тести для правильних класів еквівалентності поля «Назва заходу» наведено у таблиці 2.8, а тести для неправильних класів еквівалентності – у таблиці 2.9.

Таблиця 2.8 – Тести для правильних класів еквівалентності

№	Клас	Тестові дані	Очікуваний результат
1	EC1	Концерт Jazz Night	Захід успішно створено
2	EC1	Кінофестиваль 2026	Захід успішно створено
3	EC1	Театральна вистава "Лісова пісня"	Захід успішно створено

Таблиця 2.9 – Тести для неправильних класів еквівалентності

№	Клас	Тестові дані	Очікуваний результат
1	EC2	"" (порожній рядок)	Повідомлення про обов'язковість заповнення поля
2	EC3	А	Повідомлення про недостатню довжину назви
3	EC3	«Д	Повідомлення про недостатню довжину назви
4	EC4	Рядок довжиною 256 символів	Повідомлення про перевищення максимальної довжини

Результати тестування показали, що всі основні функції програмного забезпечення працюють відповідно до визначених вимог. Під час перевірки було підтверджено коректність взаємодії між телеграм-ботом, серверною частиною та базою даних. Також було встановлено, що програмна система забезпечує

правильне збереження та відображення інформації про заходи, а механізм імпорту та модерації дозволяє підтримувати актуальність афіші.

Для оцінювання якості програмного забезпечення додатково було виконано перевірку стійкості до некоректного введення даних. Результати перевірки показали, що система коректно обробляє помилкові запити користувачів та запобігає виникненню критичних помилок під час роботи.

Таким чином, реалізований телеграм-бот забезпечує виконання всіх поставлених функціональних вимог, має модульну архітектуру та може бути використаний як інструмент інформування мешканців та гостей м. Миколаєва про актуальні розважальні заходи.

Випробування програмного забезпечення

Після тестування було виконано випробування програмного забезпечення згідно з Програмою та методикою випробування телеграм-боту афіши розважальних заходів м. Миколаєва, яка наведена у Додатку Д.

Результати випробування показали, що всі основні функції програмної системи працюють відповідно до визначених функціональних вимог. У процесі перевірки не було виявлено критичних помилок, які б унеможливлювали використання телеграм-бота за призначенням. Отримані результати підтверджують коректність реалізації основних сценаріїв взаємодії користувачів та адміністраторів із системою.

3 РЕЗУЛЬТАТИ РОЗРОБКИ ТЕЛЕГРАМ-БОТУ АФІШИ РОЗВАЖАЛЬНИХ ЗАХОДІВ М. МИКОЛАЄВА

В результаті виконання кваліфікаційної роботи було розв’язано практичну задачу інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов – розроблено телеграм-бот афіши розважальних заходів м. Миколаєва, який надасть мешканцям та гостям Миколаєва інформацію про розважальні заходи, які проводяться у місті. Програмний продукт призначений для централізованого отримання інформації про концерти, театральні вистави, фестивалі, кінопокази та інші культурно-розважальні події, що відбуваються у місті.

Під час розробки було виконано аналіз предметної області, визначено функціональні та нефункціональні вимоги до програмного забезпечення, спроєктовано архітектуру системи, структуру бази даних та інтерфейс взаємодії користувачів із системою. Результатом виконаних робіт стало створення програмного продукту, який забезпечує автоматизований доступ до афіші заходів через популярний месенджер Telegram.

Розроблений бот реалізовано із використанням мови програмування Python та фреймворку aiogram. Для зберігання даних використовується реляційна база даних MySQL. Архітектура програмного забезпечення побудована відповідно до принципів багаторівневої організації програмних систем і включає рівень представлення, рівень бізнес-логіки та рівень доступу до даних.

На етапі проєктування було розроблено діаграму компонентів, яка відображає взаємодію між основними складовими системи. До складу програмного забезпечення входять модуль Telegram-бота, модуль обробки бізнес-логіки, модуль імпорту інформації із зовнішніх джерел та модуль доступу до бази даних. Взаємодія між компонентами забезпечує отримання користувацьких запитів, обробку інформації та формування відповідей у режимі реального часу.

Основним способом взаємодії користувача із системою є використання інтерфейсу Telegram-бота. Після запуску бота користувач отримує доступ до головного меню, яке містить основні функції системи. Зокрема, реалізовано можливість перегляду найближчих заходів, пошуку подій за категоріями, перегляду детальної інформації про конкретний захід та отримання довідкової інформації щодо роботи системи.

Для спрощення навігації інтерфейс побудовано на основі інтерактивних кнопок Telegram. Такий підхід дозволяє мінімізувати кількість текстових команд, які необхідно вводити користувачу, та підвищує зручність використання програмного продукту. Приклад реалізації головного меню телеграм-бота наведено на рисунку 3.1.

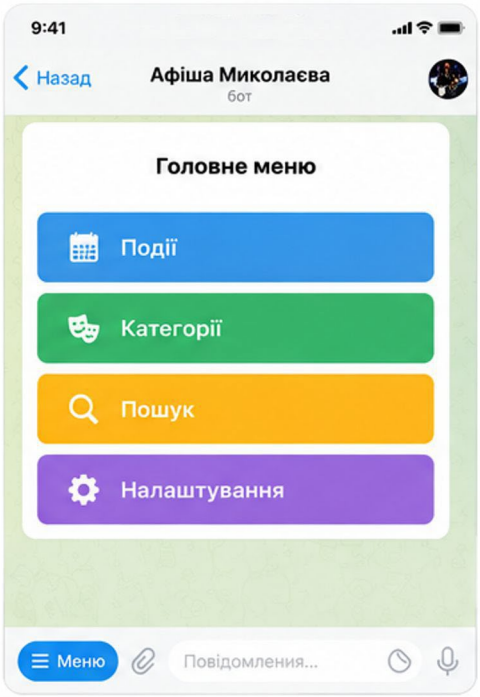


Рисунок 3.1 – Головне меню телеграм-бота афіши розважальних заходів м. Миколаєва

Однією з ключових функцій системи є перегляд інформації про заходи. Після вибору відповідної команди користувач отримує перелік найближчих подій із зазначенням їх назв, дат проведення та місць проведення. Для кожного заходу передбачено можливість перегляду детальної інформації, включаючи опис події

та посилання на ресурс для придбання квитків або отримання додаткової інформації. Приклад інтерфейсу перегляду заходів наведено на рисунку 3.2.

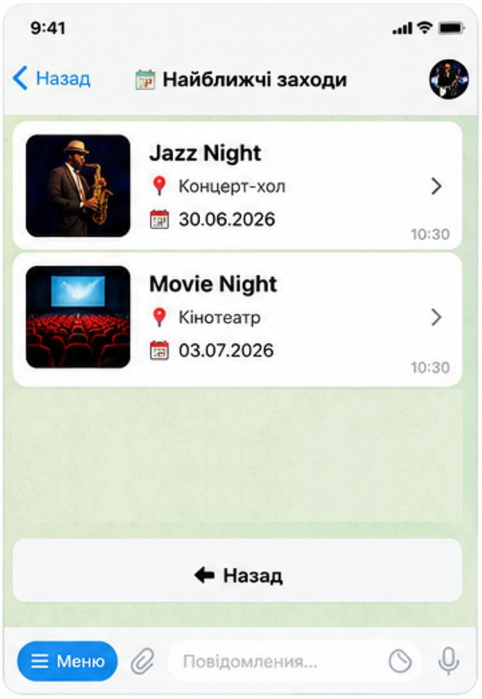


Рисунок 3.2 – Інтерфейс перегляду заходів телеграм-бота афіши розважальних заходів м. Миколаєва

Для забезпечення актуальності інформації у системі реалізовано механізм імпорту даних із зовнішніх джерел. Отримані записи автоматично завантажуються до спеціального сховища імпортованих даних, після чого проходять процедуру модерації. Такий підхід дозволяє запобігти потраплянню до афіші недостовірної або застарілої інформації.

Окрему увагу приділено реалізації адміністративних функцій. Для адміністраторів системи передбачено спеціальний режим роботи, який дозволяє виконувати керування інформаційним наповненням бота. Адміністратор може додавати нові заходи, редагувати існуючі записи, видаляти неактуальну інформацію, запускати процедуру імпорту даних та виконувати модерацію отриманих відомостей. Приклад інтерфейсу адміністративної панелі наведено на рис. 3.3.

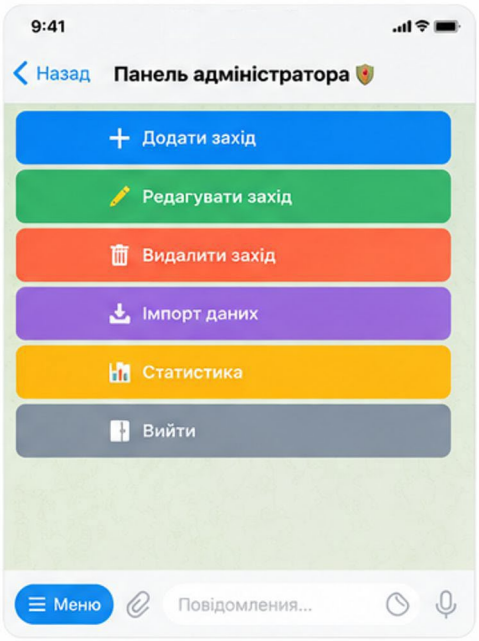


Рисунок 3.3 – Інтерфейс адміністративної панелі телеграм-бота афіши розважальних заходів м. Миколаєва

З метою забезпечення надійності роботи програмного забезпечення було реалізовано механізми перевірки коректності введених даних. Зокрема, передбачено контроль обов'язкових полів, перевірку допустимих форматів дат і часу, а також обробку помилок, які можуть виникати під час взаємодії із зовнішніми джерелами даних. У випадку виникнення помилкових ситуацій користувач або адміністратор отримує відповідне повідомлення з поясненням причини помилки.

Для зберігання інформації про заходи, категорії, адміністраторів та результати імпорту було розроблено реляційну базу даних. Структура бази даних забезпечує цілісність інформації та підтримує зв'язки між основними сутностями системи. Використання реляційної моделі дозволяє ефективно виконувати пошук, сортування та фільтрацію даних, необхідних для формування афіші.

Після завершення реалізації програмного забезпечення було проведено його тестування. Тестування здійснювалося на основі функціональних вимог, сформованих на етапі аналізу предметної області. Перевірялися основні сценарії роботи користувачів та адміністраторів, включаючи запуск бота, перегляд заходів, пошук за категоріями, додавання нових подій та виконання імпорту інформації.

Результати тестування підтвердили коректність функціонування всіх реалізованих модулів програмної системи. У процесі тестування було встановлено, що програмний продукт успішно виконує обробку користувацьких запитів, забезпечує збереження даних у базі даних та коректно відображає інформацію про заходи. Також було підтверджено працездатність механізму модерації імпортованих записів та правильність реалізації адміністративних функцій.

Отримані результати свідчать про досягнення поставленої мети кваліфікаційної роботи. Розроблений телеграм-бот забезпечує швидкий та зручний доступ до інформації про розважальні заходи м. Миколаєва, сприяє централізації даних із різних джерел та дозволяє підтримувати актуальність афіші за рахунок механізмів імпорту й модерації інформації. Створений програмний продукт може бути використаний як основа для подальшого розвитку інформаційного сервісу міських заходів та розширення його функціональних можливостей.

4 ОХОРОНА ПРАЦІ

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності [22].

Мета охорони праці – забезпечення безпечних, нешкідливих і сприятливих умов праці через вирішення багатьох складних завдань, основними з яких є:

- проектування підприємств, технологічних процесів і конструювання обладнання з обов'язковим виконанням вимог охорони праці;

- знаходження оптимальних співвідношень між різними факторами виробничого середовища, що дозволяє забезпечити мінімум несприятливого впливу їх на здоров'я працівників;

- встановлення, законодавче оформлення визначених норм кожного з несприятливих або небезпечних факторів, систематичний контроль за їх застосуванням;

- розробка конкретних заходів щодо покращення умов праці та забезпечення її безпеки на основі застосування у виробництві новітніх досягнень науки і техніки;

- застосування раціональних засобів захисту працівників від впливу несприятливих факторів виробничого середовища, а також втілення організаційних заходів, які нейтралізують або послаблюють ступінь їх впливу на організм людини;

- розробка та застосування методів і засобів оцінки ефективності заходів з охорони праці, що плануються і здійснюються.

Основним законом в галузі охорони праці є Закон України «Про охорону праці». Дія Закону України «Про охорону праці» поширюється на всіх юридичних та фізичних осіб, які відповідно до законодавства використовують найману працю, та на всіх працюючих.

4.1 Аналіз шкідливих та небезпечних факторів на робочому місці

Під час укладання трудових договорів роботодавець повинен проінформувати працівника під розписку про умови праці та про наявність на його робочому місці небезпечних і шкідливих виробничих факторів, які ще не усунуто, можливі наслідки їх впливу на здоров'я та про права працівника на пільги і компенсації за роботу в таких умовах відповідно до законодавства і колективного договору.

Шкідливий виробничий фактор – виробничий фактор, вплив якого може призвести до погіршення стану здоров'я, зниження працездатності працівника.

Небезпечний виробничий фактор – виробничий фактор, дія якого за певних умов може призвести до травм або іншого раптового погіршення здоров'я працівника. В таблиці 4.1 перераховані шкідливі та небезпечні виробничі фактори.

Таблиця 4.1 – Шкідливі та небезпечні виробничі фактори

Найменування фактору	Можливі джерела його виникнення	Характер дії
Небезпека ураження електричним струмом	Мережа живлення	Небезпечний
Пожежонебезпечність приміщень	Наявність матеріалів, що загоряються і джерел запалення	Небезпечний та шкідливий
Електромагнітне випромінювання в тому числі і рентгенівське	ЕПТ (Дисплей с джерелом рентгенівського, радіочастотного, ультрафіолетового й інфрачервоного випромінювання та випромінювання звукового діапазону)	Шкідливий
Несприятлива освітленість	Недостатнє штучне і природне освітлення	Шкідливий
Психофізіологічні напруження	Монотонність праці, перенапруженість зорових аналізаторів, розумова напруженість, незручність і статичність пози	Шкідливий

Тривала і інтенсивна робота за комп’ютером може стати джерелом важких професійних та звичайних захворювань таких як:

- Неправильна постава – це призводить до подальшого розвитку викривлення хребта, сколіозу, лордозу, кіфозу, а як результат голові болі, болі в області шиї і всього хребта, болі в області тазу.

- Порушення фокусування – є наслідком напруженої роботи за монітором комп’ютера, може бути викликана перенапруженням очних м’язів.

- Тунельний синдром або синдром зап’ястного каналу – проявляється після напруженої роботи за ПК.

- Короткозорість – розвивається через те, що екран монітора за контрастом вище ніж навколишні об’єкти.

- Сухість очей – при погляді на джерело світла людина починає менше моргати та виникає «обсушування» рогівки ока, що призводить до болів.

- Геморой – причиною якого стає застій крові у венах.

4.2 Заходи щодо зменшення шкідливих факторів на робочому місці

Перш ніж починати роботу за ПК рекомендується обладнати робоче місце, а саме:

- Висоту робочої поверхні столу відрегулювати в межах 680-800 мм, при відсутності регулювання висота робочої поверхні столу повинна бути 725 мм.

- Робочий стіл повинен мати простір для ніг висотою не менше 600 мм, шириною не менше 500 мм, глибиною на рівні колін не менше 450 мм і для витягнутих не менше 650 мм.

- Клавіатуру розташувати на поверхні столу на відстані 100 – 300 мм від краю, зверненого до користувача, або на спеціальній регульованій по висоті стільниці.

- Рівень очей при вертикально розташованому екрані має припадати на центр або 2/3 висоти екрану, лінія погляду повинна бути перпендикулярна центру екрану.

Рекомендації під час роботи за ПК:

- Для зменшення негативного впливу на здоров'я працівників виробничих факторів необхідно застосовувати регламентовані перерви.

- Тривалість безперервної роботи за візуальним дисплейним терміналом без регламентованої перерви не повинна перевищувати 2 години.

- Під час перерви не варто читати або дивитися телевізор.

- У робочому приміщенні, де розташовані комп'ютери, щодня потрібно виконувати вологе прибирання. Також приміщення потрібно провітрювати щогодини.

- Необхідно слідкувати за станом екрану монітора, який має бути без пилу та плям.

- Не рекомендовано використовувати звичайний телевізор замість монітора.

- У процесі роботи рекомендується періодично (приблизно раз на 20-30 хвилин) переводити погляд з екрана на найбільш віддалений предмет у кімнаті, а ще краще – на віддалений об'єкт за вікном.

- Якщо з'явилося відчуття втоми, напруження, сонливості, тяжкості в очах, потрібно припинити роботу та хоча б трохи відпочити.

- Потрібно слідкувати за поставою: ноги повинні твердо стояти на підлозі чи на спеціальній підставці: стегна повинні бути під прямим кутом до тулуба, а гомілки – під прямим кутом до стегон; сидіти потрібно прямо або злегка нахилившись вперед; відстань від очей до екрана монітора – не менше 55-60 см; центр екрану має знаходитися на рівні очей чи трохи нижче; рекомендується хоча б раз на день виконувати гімнастику для очей.

- Щоб уникнути синдрому «сухого ока» рекомендовано моргати кожні 3-5 секунд для зволоження рогівки ока.

Одна з причин, яка викликає втому очей при роботі за комп’ютером – постійне мерехтіння і світлова пульсація картинки на моніторі.

Вправи для зняття втоми очей:

- Розітріть долоні, щоб у них з’явилося відчуття тепла, складіть їх навхрест, прикладіть до повік, передаючи таким чином їх енергію тепла до очей. Затримайтеся в такому положенні на 30-40 секунд. Після закрийте очі і повільно натискайте на повіки кінчиками пальців, по 20 разів.

- Закрийте очі, зробіть по 10 обертаючих рухів очними яблуками за годинниковою стрілкою і проти неї. Після цього, заплющивши очі, відкрийте очі. Зробити п’ять повторень.

- Зробіть кілька постукуючих легких рухів пальцями двох рук по голові, почавши з чола і поступово перейшовши до потилиці.

- Моргніть очима два рази, сильно і різко примружтеся і різко відкрийте очі. Зробіть 10 разів.

- Сильно розітріть долоні, щоб відчути теплоту, а після погладжуючими рухами п’ять разів натисніть на очні яблука.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи було розв’язано практичну задачу інженерії програмного забезпечення, що характеризується комплексністю та невизначеністю умов – розроблено телеграм-бот афіши розважальних заходів м. Миколаєва.

Усі поставлені завдання було виконано у повному обсязі, а саме:

– виконано аналіз практичної задачі інженерії програмного забезпечення з теми кваліфікаційної роботи;

– виконано аналіз існуючих програмних рішень для розробки телеграм-боту;

– виконано постановку задачі на розробку телеграм-боту афіши розважальних заходів м. Миколаєва;

– виконано аналіз вимог до телеграм-боту афіши розважальних заходів м. Миколаєва;

– розроблено архітектуру телеграм-боту афіши розважальних заходів м. Миколаєва;

– розроблено проект телеграм-боту афіши розважальних заходів м. Миколаєва.

Використання розробленого телеграм-боту афіши розважальних заходів м. Миколаєва як інформаційного сервісу забезпечить зручний доступ до афіши через популярний месенджер, а також підвищити рівень інформованості мешканців та гостей міста.

Також у кваліфікаційній роботі було виконано розділ з охорони праці.

Отже, мета кваліфікаційної роботи, яка була поставлена на початку роботу, досягнута.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Культурний центр Миколаївщини. URL: https://pkm.mk.ua/ (дата звернення: 24.03.2026).

2. Обласний палац культури. URL: https://www.odk.net.ua/ (дата звернення: 24.03.2026).

3. Афіша Миколаїв – вистави та концерти. URL: https://teatr.org.ua/cities/mikolayiv (дата звернення: 24.03.2026).

4. Telemetrio. Куди піти? | Миколаїв. URL: https://telemetr.io/uk/channels/1398380751-mykolaiv_to_go (дата звернення: 24.03.2026).

5. Telegram Messenger. URL: https://telegram.org/ (дата звернення: 27.03.2026).

6. Telegram Bot API. URL: https://core.telegram.org/bots/api (дата звернення: 27.03.2026).

7. Messenger Brings People Together All Over the World. URL: https://about.fb.com/news/2017/04/messenger-f8/ (дата звернення: 27.03.2026).

8. Discord. Bots & Companion Apps. URL: https://docs.discord.com/developers/platform/bots (дата звернення: 27.03.2026).

9. Viber Developers Hub. URL: https://developers.viber.com/ (дата звернення: 27.03.2026).

10. ManyChat. Social smarter, not harder. URL: https://manychat.com/ (дата звернення: 27.03.2026).

11. SendPulse. URL: https://sendpulse.ua/features/chatbot (дата звернення: 27.03.2026).

12. Chatfuel. Stop Replying. Start Running. URL: https://chatfuel.com/ (дата звернення: 27.03.2026).

13. Make. The visual AI automation platform. URL: https://www.make.com/ (дата звернення: 27.03.2026).

14. How to get started with Telegram on Zapier. URL: https://help.zapier.com/hc/en-us/articles/16700016131085-How-to-get-started-with-Telegram-on-Zapier (дата звернення: 27.03.2026).

15. Діаграма варіантів використання. URL: https://dou.ua/forums/topic/40575/ (дата звернення: 12.04.2026).

16. 1. Component Diagram Tutorial. What is a UML component diagram? URL: https://www.lucidchart.com/pages/uml-component-diagram (дата звернення: 12.04.2026).

17. Deployment Diagram Tutorial. URL: https://www.lucidchart.com/pages/uml-deployment-diagram (дата звернення: 12.04.2026).

18. Python. URL: https://www.python.org/ (дата звернення: 12.04.2026).

19. aiogram. URL: https://aiogram.dev/ (дата звернення: 12.04.2026).

20. MySQL. URL: https://www.mysql.com/products/mysql/ (дата звернення: 12.04.2026).

21. SQLAlchemy – The Database Toolkit for Python. URL: https://www.sqlalchemy.org/ (дата звернення: 12.04.2026).

22. Про охорону праці : Закон України від 14.10.1992 № 2694-XII. Редакція від 12.09.2025. URL: https://zakon.rada.gov.ua/laws/show/2694-12 (дата звернення: 20.05.2026).

Додаток А Технічне завдання на розробку телеграм-боту афіші розважальних заходів м. Миколаєва

Вступ

Найменування програмного забезпечення: телеграм-бот афіші розважальних заходів м. Миколаєва. Область застосування – месенджер Telegram.

1 Підстава для розробки

Підставою для розробки програмного забезпечення є завдання на кваліфікаційну роботу на здобуття ступеня вищої освіти "бакалавр", виданого кафедрою ПЗАС НУК імені адмірала Макарова.

2 Призначення розробки

2.1 Експлуатаційне призначення

Експлуатаційним призначенням програмного забезпечення є допомога мешканцям та гостям Миколаєва у зручному вигляді отримувати інформацію щодо розважальних заходів, які проводяться у місті.

2.2 Функціональне призначення

Функціональним призначенням програмного забезпечення є: надання користувачеві інформації про найближчі розважальні заходи м. Миколаєва, перегляд детальної інформації про вибраний захід, включаючи опис, адресу проведення та контактні дані організаторів, пошук заходів за категоріями (концерти, вистави, фестивалі, спортивні події, виставки тощо) та датою проведення; додавання нових заходів адміністратором системи.

3 Вимоги до програмного забезпечення

3.1 Вимоги до функціональних характеристик

3.1.1 Вимоги до складу виконуваних функцій

1. Надання користувачеві інформації про найближчі розважальні заходи м. Миколаєва.

2. Відображення списку заходів із зазначенням назви, дати, часу та місця проведення.

3. Перегляд детальної інформації про вибраний захід, включаючи опис, адресу проведення та контактні дані організаторів.

4. Пошук заходів за категоріями (концерти, вистави, фестивалі, спортивні події, виставки тощо).

5. Пошук заходів за датою проведення.

6. Відображення актуальної афіші на поточний день або визначений період часу.

7. Надання користувачеві посилань на офіційні ресурси заходів або сторінки продажу квитків.

8. Можливість перегляду інформації про місце проведення заходу.

9. Відображення головного меню та навігаційних кнопок для спрощення взаємодії користувача із системою.

10. Реєстрація та обробка команд користувача через інтерфейс Telegram.

11. Збереження інформації про заходи у базі даних.

12. Додавання нових заходів адміністратором системи.

13. Редагування відомостей про існуючі заходи.

14. Видалення неактуальних або скасованих заходів.

15. Забезпечення авторизованого доступу адміністратора до функцій керування контентом.

16. Формування відповідей користувачам відповідно до введених запитів.

17. Надання повідомлень про помилки або некоректно введені команди.

3.1.2 Вимоги до організаційних вхідних та вихідних даних

Дані зберігаються у базі даних з довільним доступом.

Вхідні дані: Захід, Категорія, Джерело даних, Імпортований захід.

Вихідні дані: інформації про вибраний захід, включаючи опис, адресу проведення та контактні дані організаторів.

3.2 Нефункціональні вимоги

1. Продуктивність. Час формування відповіді на запит користувача не повинен перевищувати декількох секунд за умови нормального навантаження на систему.

2. Надійність. Система повинна забезпечувати коректну роботу навіть у разі виникнення помилок введення або тимчасової недоступності окремих зовнішніх сервісів.

3. Доступність. Телеграм-бот повинен бути доступним для користувачів цілодобово через платформу Telegram.

4. Зручність використання. Інтерфейс взаємодії має бути зрозумілим для користувачів без спеціальної підготовки та не вимагати вивчення складних інструкцій.

5. Масштабованість. Архітектура програмного забезпечення повинна забезпечувати можливість збільшення кількості користувачів та обсягу інформації про заходи без суттєвої зміни структури системи.

6. Супроводжуваність. Програмний код повинен мати модульну структуру, що спрощує внесення змін та подальший розвиток програмного продукту.

7. Безпека. Система повинна забезпечувати захист адміністративних функцій від несанкціонованого доступу та безпечне зберігання даних.

8. Сумісність. Телеграм-бот повинен коректно функціонувати на всіх пристроях, що підтримують офіційний клієнт Telegram.

9. Портативність. Серверна частина системи повинна мати можливість розгортання на різних операційних системах та хостингових платформах.

10. Актуальність даних. Інформація про заходи повинна своєчасно оновлюватися та відповідати фактичному стану подій.

3.3 Вимоги до надійності

Програмне забезпечення повинно функціонувати без збоїв при умові безперервної роботи серверу на якому міститься ПЗ, стабільній роботі самого месенджеру Telegram та стабільному підключенні до мережі Internet.

У випадку введення користувачем некоректної команди, бот видасть повідомлення про помилку.

3.3 Вимоги до умов експлуатації

3.3.1 Кліматичні умови експлуатації

Кліматичні умови експлуатації, при яких повинні забезпечуватися задані характеристики, повинні відповідати вимогам, що пред'являються до технічних засобів в частині умов їх експлуатації.

3.3.2 Вимоги до кваліфікації користувача

Необхідний рівень підготовки користувача:

– мінімальні навики володіння комп’ютером або смартфоном;
– вміння користуватися будь-якою пошуковою системою мережи Internet;
– мінімальні навички роботи з месенджером Telegram.

3.4 Вимоги до складу та параметрів технічних засобів

Персональний комп'ютер використовується для розробки, адміністрування та тестування телеграм-бота. Комп’ютер повинен мати наступну мінімальну комплектацію:

– Процесор: 2-ядерний процесор із тактовою частотою 2,0 ГГц;
– Оперативна пам'ять: 4 ГБ;
– Вільне місце на диску: 2 ГБ;
– Мережева підтримка: Підключення до мережі Інтернет;
– Монітор: Роздільна здатність не менше 1366×768;
– Пристрої введення: клавіатура, миша;

Вимоги до смартфона

Смартфон використовується для взаємодії користувача з телеграм-ботом через застосунок Telegram. Смартфон повинен мати наступну мінімальну комплектацію:

– Оперативна пам'ять: 2 ГБ;

– Вільне місце: 200 МБ;

– Мережа: Доступ до Інтернету (Wi-Fi або мобільна мережа);

– Операційна система: Android 8.0 / iOS 15;

– Програмне забезпечення: встановлений застосунок Telegram.

4 Вимоги до програмної документації

Програмна документація повинна містити:

- технічне завдання;
- код програми;
- опис програми;
- керівництво користувача;
- програму та методику випробувань.

5 Стадії та етапи розробки

Стадії та етапи розробки представлені в таблиці А.1.

Таблиця А.1 – Стадії та етапи розробки програмного забезпечення

Номер п/п	Назва етапів роботи	Термін виконання
1	Аналіз практичної задачі інженерії програмного забезпечення	12.02.2026
2	Аналіз вимог до програмного забезпечення	09.03.2026
3	Розробка архітектури програмного забезпечення	06.04.2026
4	Розробка проєкту програмного забезпечення	14.04.2026
5	Реалізація та тестування програмного забезпечення	30.04.2026

6 Порядок контролю та прийому

Контроль за аналізом та проектуванням кожної окремої частини програмного забезпечення відбувається на кожному етапі з урахуванням вимог, визначених у технічному завданні. Кожна стадія розробки повинна бути представлена в зазначені строки та узгоджена із замовником.

Прийом проводиться відповідно до програми і методики випробувань і документується за допомогою протоколу проведення випробувань. У разі знаходження помилок під час прийому програмного забезпечення, складається акт про знайдені помилки, який підписується представниками замовника і розробника затверджується керівником організації-замовника та організації-розробника. Розробник повинен, на протязі не більше ніж 2 тижні, виправити зазначені помилки і оповістити замовника про повторне проведення перевірки.

Додаток Б Код телеграм-боту афіші розважальних заходів м. Миколаєва

```
-- ========================================
-- БАЗА ДАНИХ: Telegram Event Bot
-- ========================================

CREATE DATABASE IF NOT EXISTS event_bot
CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;

USE event_bot;

-- ========================================
-- ТАБЛИЦЯ КАТЕГОРІЙ
-- ========================================

CREATE TABLE categories (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL UNIQUE
);

-- ========================================
-- ТАБЛИЦЯ АДМІНІСТРАТОРІВ
-- ========================================

CREATE TABLE admins (
    id INT AUTO_INCREMENT PRIMARY KEY,
    login VARCHAR(100) NOT NULL UNIQUE,
    password_hash VARCHAR(255) NOT NULL
);

-- ========================================
-- ОСНОВНІ ЗАХОДИ
-- ========================================

CREATE TABLE events (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    event_date DATE NOT NULL,
    event_time TIME,
    location VARCHAR(255),
    ticket_url VARCHAR(500),

    category_id INT,
    created_by_admin_id INT NULL,

    source_import_id INT NULL,

    FOREIGN KEY (category_id) REFERENCES categories(id)
```

```
        ON DELETE SET NULL,

    FOREIGN KEY (created_by_admin_id) REFERENCES admins(id)
        ON DELETE SET NULL
);

-- ========================================
-- ДЖЕРЕЛА ДАНИХ
-- ========================================

CREATE TABLE data_sources (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    url VARCHAR(500) NOT NULL,
    type VARCHAR(50),
    is_active BOOLEAN DEFAULT TRUE
);

-- ========================================
-- ІМПОРТОВАНІ ЗАХОДИ (STAGING)
-- ========================================

CREATE TABLE imported_events (
    id INT AUTO_INCREMENT PRIMARY KEY,
    raw_title VARCHAR(255),
    raw_description TEXT,
    raw_date VARCHAR(100),

    status ENUM('pending', 'approved', 'rejected') DEFAULT
'pending',

    source_id INT,
    reviewed_by_admin_id INT NULL,

    FOREIGN KEY (source_id) REFERENCES data_sources(id)
        ON DELETE CASCADE,

    FOREIGN KEY (reviewed_by_admin_id) REFERENCES admins(id)
        ON DELETE SET NULL
);

-- ========================================
-- ІНДЕКСИ ДЛЯ ШВИДКОГО ПОШУКУ
-- ========================================

CREATE INDEX idx_events_date ON events(event_date);
CREATE INDEX idx_events_category ON events(category_id);
CREATE INDEX idx_imported_status ON imported_events(status);

-- ========================================
-- ПРИКЛАДИ ДОВІДНИКІВ
-- ========================================
```

```
INSERT INTO categories (name) VALUES
('Концерти'),
('Вистави'),
('Фестивалі'),
('Вечірки'),
('Кіно');
```

main.py

```
import asyncio
import logging

from aiogram import Bot, Dispatcher
from aiogram.client.default import DefaultBotProperties
from aiogram.enums import ParseMode
from aiogram.filters import Command
from aiogram.types import Message

from config.settings import BOT_TOKEN
from services.event_service import EventService
from services.import_service import ImportService
from repositories.event_repository import EventRepository
from database.connection import SessionLocal

logging.basicConfig(
    level=logging.INFO,
    format="%(asctime)s | %(levelname)s | %(name)s |
%(message)s"
)

logger = logging.getLogger(__name__)

bot = Bot(
    token=BOT_TOKEN,
    default=DefaultBotProperties(
        parse_mode=ParseMode.HTML
    )
)

dp = Dispatcher()


def get_event_service() -> EventService:
    session = SessionLocal()

    repository = EventRepository(session)

    return EventService(
        repository=repository
    )

@dp.message(Command("start"))
async def start_command(message: Message):
```

```
    await message.answer(
        """
<b>Афіша розважальних заходів м. Миколаєва</b>

Доступні команди:

/events - список заходів
/categories - категорії
/import - імпорт подій
/help - довідка
"""
    )

@dp.message(Command("events"))
async def events_command(message: Message):

    service = get_event_service()

    try:

        events = service.get_upcoming_events()

        if not events:

            await message.answer(
                "Наразі заходи відсутні."
            )

            return

        response = []

        for event in events:

            response.append(
                f"🎭 <b>{event.title}</b>\n"
                f"🗓️ {event.event_date}\n"
                f"📍 {event.location}\n"
            )

        await message.answer(
            "\n\n".join(response)
        )

    except Exception as ex:

        logger.exception(ex)

        await message.answer(
            "Помилка отримання даних."
        )
```

```
@dp.message(Command("categories"))
async def categories_command(message: Message):

    await message.answer(
        """
🎭 Концерти
🎬 Кіно
🎪 Фестивалі
🎭 Театр
"""
    )

@dp.message(Command("import"))
async def import_command(message: Message):

    import_service = ImportService()

    try:

        imported_count = (
            await import_service.import_events()
        )

        await message.answer(
            f"Імпортовано {imported_count} записів."
        )

    except Exception as ex:

        logger.exception(ex)

        await message.answer(
            "Помилка під час імпорту."
        )

@dp.message(Command("help"))
async def help_command(message: Message):

    await message.answer(
        "Використовуйте доступні команди меню."
    )

async def main():

    logger.info("Bot started")

    await dp.start_polling(bot)

if __name__ == "__main__":

    asyncio.run(main())
```

```
services/event_service.py
from datetime import date

from repositories.event_repository import EventRepository

class EventService:

    def __init__(
        self,
        repository: EventRepository
    ):
        self.repository = repository

    def get_upcoming_events(self):

        return self.repository.find_upcoming()

    def get_event_by_id(self, event_id: int):

        if event_id <= 0:
            raise ValueError(
                "Invalid event id"
            )

        return self.repository.find_by_id(
            event_id
        )

    def create_event(
        self,
        title: str,
        description: str,
        event_date,
        location: str
    ):

        self._validate_event(
            title,
            event_date
        )

        return self.repository.create(
            title=title,
            description=description,
            event_date=event_date,
            location=location
        )

    def update_event(
        self,
        event_id: int,
        **kwargs
```

```
    ):

        event = self.get_event_by_id(
            event_id
        )

        if not event:
            raise Exception(
                "Event not found"
            )

        return self.repository.update(
            event_id,
            **kwargs
        )

    def delete_event(
        self,
        event_id: int
    ):

        return self.repository.delete(
            event_id
        )

    def _validate_event(
        self,
        title,
        event_date
    ):

        if not title:
            raise ValueError(
                "Title is required"
            )

        if len(title) < 3:
            raise ValueError(
                "Title too short"
            )

        if event_date < date.today():
            raise ValueError(
                "Invalid event date"
            )
```

services/import_service.py

```
import logging

import requests

from bs4 import BeautifulSoup
```

```
logger = logging.getLogger(__name__)

class ImportService:

    SOURCES = [
        "https://example.com/events"
    ]

    async def import_events(self):

        total_imported = 0

        for source in self.SOURCES:

            try:

                events = self._load_source(
                    source
                )

                total_imported += len(events)

                logger.info(
                    "Imported %s events "
                    "from %s",
                    len(events),
                    source
                )

            except Exception as ex:

                logger.exception(ex)

        return total_imported

    def _load_source(
        self,
        url: str
    ):

        response = requests.get(
            url=url,
            timeout=10
        )

        response.raise_for_status()

        soup = BeautifulSoup(
            response.text,
            "html.parser"
        )
```

```
        result = []

        event_blocks = soup.select(
            ".event-item"
        )

        for item in event_blocks:

            title = item.get_text(
                strip=True
            )

            result.append(
                {
                    "title": title
                }
            )

        return result
```

models/event.py

```
from sqlalchemy import Column
from sqlalchemy import Integer
from sqlalchemy import String
from sqlalchemy import Date
from sqlalchemy import Time
from sqlalchemy import Text
from sqlalchemy.orm import declarative_base

Base = declarative_base()

class Event(Base):

    __tablename__ = "events"

    id = Column(Integer, primary_key=True)

    title = Column(String(255), nullable=False)

    description = Column(Text)

    event_date = Column(Date)

    event_time = Column(Time)

    location = Column(String(255))

    ticket_url = Column(String(500))
```

database/connection.py

```
from sqlalchemy import create_engine
```

```
from sqlalchemy.orm import sessionmaker

DATABASE_URL = (
    "mysql+pymysql://root:password@localhost/event_bot"
)

engine = create_engine(DATABASE_URL)

SessionLocal = sessionmaker(
    autocommit=False,
    autoflush=False,
    bind=engine
)


repositories/event_repository.py
from models.event import Event

class EventRepository:

    def __init__(self, session):
        self.session = session

    def get_all(self):
        return self.session.query(Event).all()

    def get_by_id(self, event_id):
        return (
            self.session
            .query(Event)
            .filter(Event.id == event_id)
            .first()
        )

    def add(self, event):
        self.session.add(event)
        self.session.commit()


services/import_service.py
import requests
from bs4 import BeautifulSoup

class ImportService:

    def import_events(self, url):

        response = requests.get(url)

        soup = BeautifulSoup(
            response.text,
            "html.parser"
        )
```

```
events = []

for item in soup.select(".event"):

    events.append({
        "title": item.text.strip()
    })

return events
```

Додаток В Опис телеграм-боту афіши розважальних заходів м. Миколаєва

1 Загальні відомості

1.1 Позначення і найменування програми

Найменування програмного забезпечення – телеграм-бот афіші розважальних заходів м. Миколаєва

Позначення – Telegram-bot of the movie posters of Mykolaiv.

1.2 Вимоги до складу та параметрів технічних засобів

Персональний комп'ютер використовується для розробки, адміністрування та тестування телеграм-бота. Комп’ютер повинен мати наступну мінімальну комплектацію:

– Процесор: 2-ядерний процесор із тактовою частотою 2,0 ГГц;

– Оперативна пам'ять: 4 ГБ;

– Вільне місце на диску: 2 ГБ;

– Мережева підтримка: Підключення до мережі Інтернет;

– Монітор: Роздільна здатність не менше 1366×768;

– Пристрої введення: клавіатура, миша;

Вимоги до смартфона

Смартфон використовується для взаємодії користувача з телеграм-ботом через застосунок Telegram. Смартфон повинен мати наступну мінімальну комплектацію:

– Оперативна пам'ять: 2 ГБ;

– Вільне місце: 200 МБ;

– Мережа: Доступ до Інтернету (Wi–Fi або мобільна мережа);

– Операційна система: Android 8.0 / iOS 15;

– Програмне забезпечення: встановлений застосунок Telegram.

1.3 Засоби програмної реалізації

Для реалізації телеграм-бота афіші розважальних заходів м. Миколаєва було обрано стек технологій, що включає мову програмування Python, фреймворк aiogram для розробки Telegram-ботів, систему керування базами даних MySQL, ORM-бібліотеку SQLAlchemy, а також бібліотеки Requests і Beautiful Soup для отримання та обробки інформації із зовнішніх джерел.

2 Функціональне призначення

2.1 Функціональне призначення

Функціональним призначенням програмного забезпечення є: надання користувачеві інформації про найближчі розважальні заходи м. Миколаєва, перегляд детальної інформації про вибраний захід, включаючи опис, адресу проведення та контактні дані організаторів, пошук заходів за категоріями (концерти, вистави, фестивалі, спортивні події, виставки тощо) та датою проведення; додавання нових заходів адміністратором системи.

2.2 Функціональні обмеження

Для роботи з телеграм-ботом необхідно мати підключення до мережі Інтернет та встановлений додаток Telegram при використанні на смартфоні або комп’ютері.

3 Опис логічної структури

3.1 Структура програмного забезпечення

Програмне забезпечення складається з наступних частин:

– сервер, на якому запущений скрипт програмного забезпечення;

– сервер Телеграму, з яким спілкується телеграм-бот, завдяки Telegram Bot API;

– пристрій користувача із встановленим додатком Telegram та підключенням до мережі Інтернет.

3.2 Алгоритм роботи програмного забезпечення

У програмному забезпеченні користувачі взаємодіють із телеграм-ботом через мобільні або десктопні пристрої з встановленим месенджером Telegram. Запити надходять до хмарного або серверного середовища, де розгорнуто сервер додатка, що містить логіку обробки повідомлень, модуль адміністрування та модуль імпорту даних.

База даних розміщується на окремому сервері або в хмарному сховищі для забезпечення надійності та масштабованості. Модуль імпорту взаємодіє із зовнішніми джерелами даних через HTTP-запити до вебсайтів, API або інформаційних каналів.

Додаток Г Інструкція користувача телеграм-боту афіши розважальних заходів м. Миколаєва

Послідовність роботи користувача з телеграм-ботом

Вітальне повідомлення (Welcome). Перехід: запуск бота командою /start.

Призначення: після першого запуску користувач отримує вітальне повідомлення та коротку інформацію про призначення телеграм-бота. Також відображається головне меню з основними командами. Вітальний екран телеграм-боту наведено на рисунку Г.1.

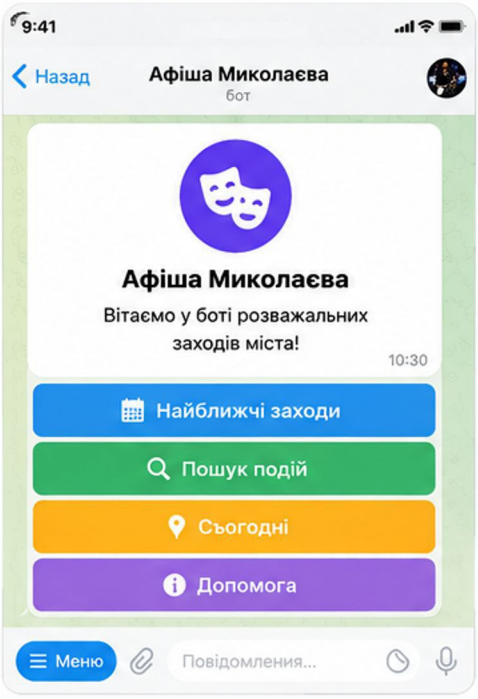


Рисунок Г.1 – Вітальний екран телеграм-боту

Дії користувача:

– перейти до перегляду заходів;

– виконати пошук;

– переглянути заходи на сьогодні;

– отримати довідку.

Головне меню. Перехід: після виконання команди /start або повернення з інших розділів.

Призначення: головне меню є центральною точкою навігації та забезпечує доступ до всіх основних функцій телеграм-бота. Головне меню телеграм-боту наведено на рисунку Г.2.

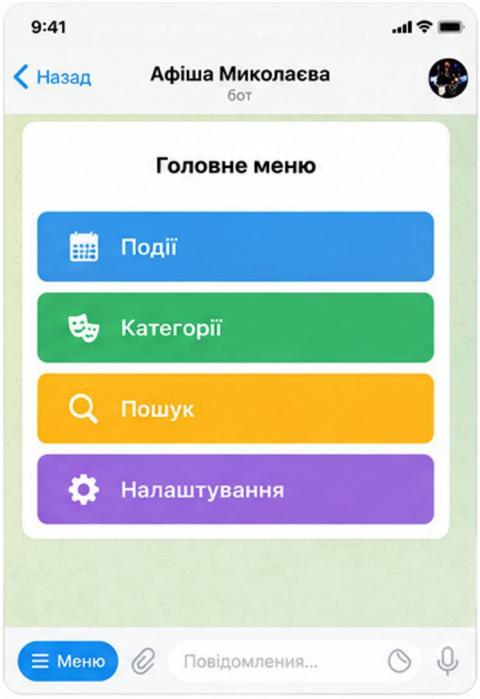


Рисунок Г.2 – Головне меню телеграм-боту

Доступні розділи:

– Події;

– Категорії;

– Пошук;

– Налаштування.

Список заходів. Перехід: кнопка «Події».

Призначення: відображається перелік найближчих розважальних заходів із короткою інформацією: назва, дата, місце проведення. Список заходів телеграм-боту наведено на рисунку Г.3.

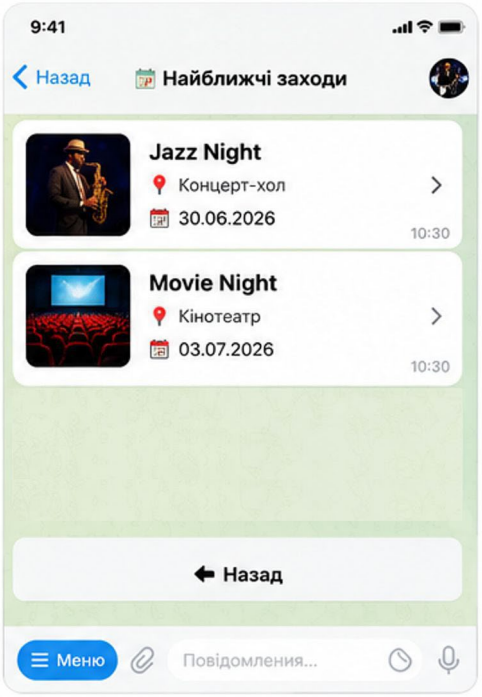


Рисунок Г.3 – Список заходів телеграм-боту

Користувач може вибрати будь-який захід для перегляду детальної інформації.

Детальна інформація про захід. Перехід: вибір конкретного заходу зі списку.

Призначення: на екрані відображаються повна назва заходу, дата, час, місце проведення, опис, посилання на придбання квитків (за наявності). Детальна інформація про захід наведена на рисунку Г.4.

Після ознайомлення користувач може повернутися до списку заходів.

Категорії. Перехід: кнопка «Категорії».

Призначення: користувач обирає категорію заходів, наприклад, концерти, театр, кіно, фестивалі, вечірки. Категорії телеграм-боту наведено на рисунку Г.5.

Після вибору категорії відображаються лише відповідні заходи.

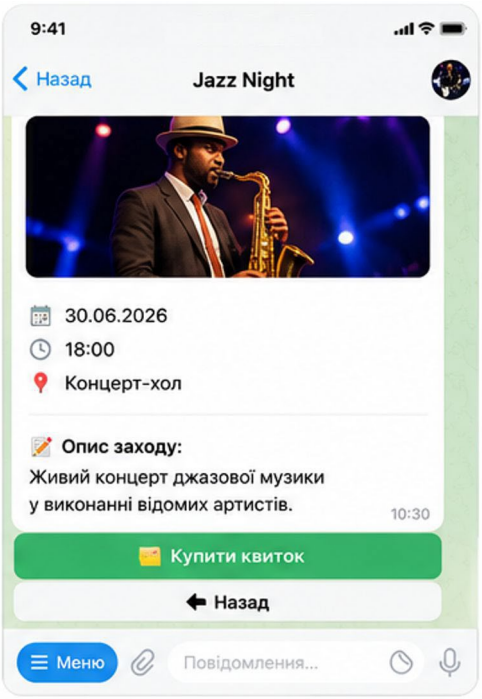


Рисунок Г.4 – Детальна інформація про захід

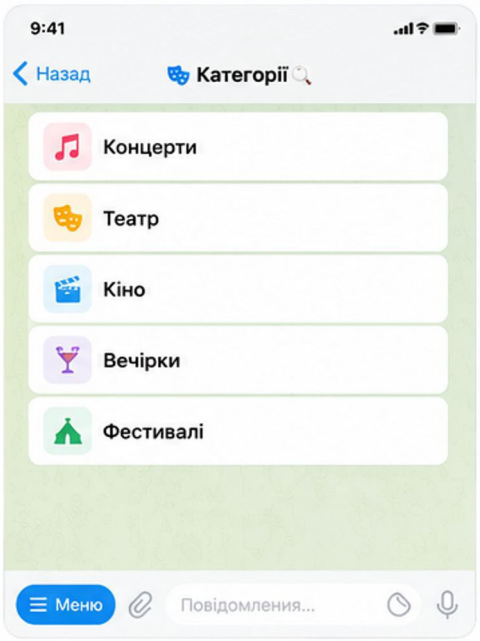


Рисунок Г.5 – Категорії телеграм-боту

Пошук. Перехід: кнопка «Пошук».

Призначення: дозволяє виконати пошук заходів за ключовими словами або іншими критеріями. Після введення запиту система формує список знайдених заходів. Результати пошуку у телеграм-боті наведено на рисунку Г.6.

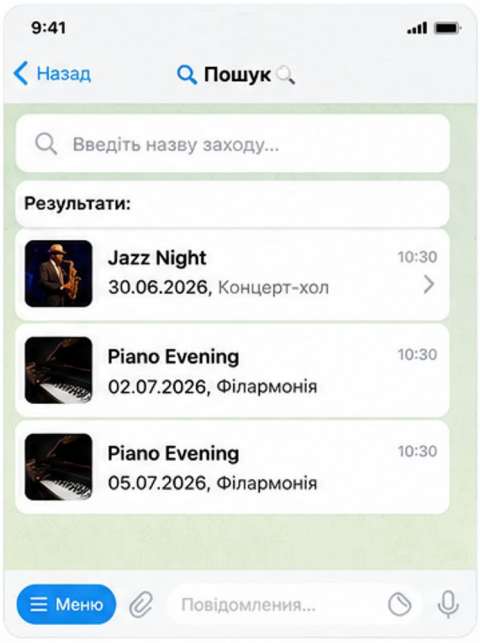


Рисунок Г.6 – Результати пошуку у телеграм-боті

Послідовність роботи адміністратора

Панель адміністратора. Перехід: після успішної авторизації адміністратора.

Призначення: адміністратор отримує доступ до функцій керування інформаційним наповненням системи. Панель адміністратора телеграм-боту наведено на рисунку Г.7.

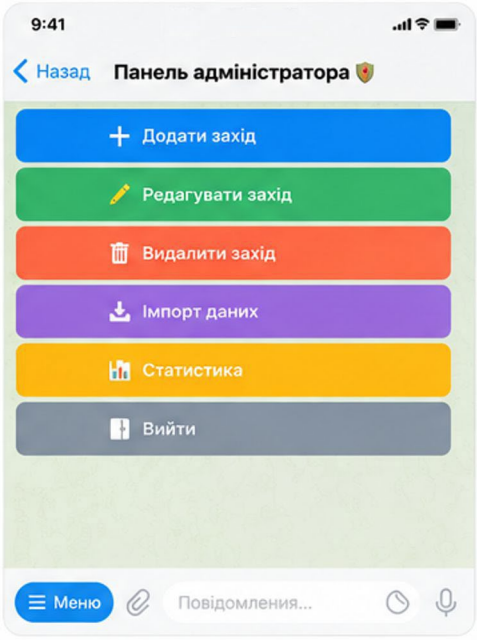


Рисунок Г.7 – Панель адміністратора телеграм-боту

Адміністратору доступні такі операції:

– додавання нового заходу;

– редагування заходу;

– видалення заходу;

– запуск імпорту;

– перегляд статистики.

Імпорт та модерація. Перехід: команда «Імпорт».

Призначення: після запуску імпорту система отримує інформацію із зовнішніх джерел та відображає її адміністратору для перевірки. Панель «Імпорт та модерація» телеграм-боту наведено на рисунку Г.8.

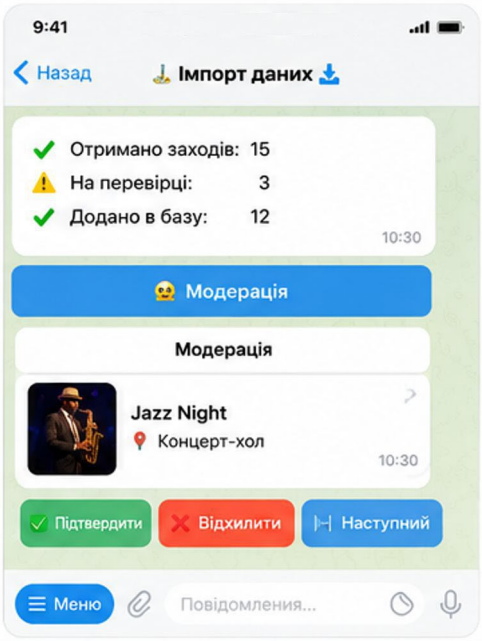


Рисунок Г.8 – Панель «Імпорт та модерація» телеграм-боту

Для кожного запису адміністратор може: підтвердити імпорт, відхилити запис, перейти до наступного заходу.

Після підтвердження інформація автоматично додається до основної афіші та стає доступною користувачам.

Додаток Д Програма та методика випробування телеграм-боту афіши розважальних заходів м. Миколаєва

1 Об’єкт випробувань

Об’єктом випробувань є телеграм-бот афіші розважальних заходів м. Миколаєва.

Область застосування ПЗ – месенджер Telegram.

2 Мета випробувань

Перевірка відповідності характеристик та вимог розробленого програмного забезпечення, викладених у технічному завданні, перевірка працездатності даного програмного забезпечення, приведення прикладу роботи та отримання відповідних навичок.

3 Вимоги до програмного забезпечення

При проведенні випробувань, можливості програмного забезпечення підлягають перевірці на відповідність вимогам, викладених у пункті «Вимоги до функціональних характеристик» технічного завдання.

4 Вимоги до програмної документації

Програмна документація повинна включати в себе наступні документи:

- технічне завдання;
- код програми;
- опис програми;
- керівництво користувача;
- програму та методику випробувань.

5 Склад та порядок випробувань

Вимоги до складу та параметрів технічних і програмних засобів вказані в Додатку А.

Порядок проведення випробувань:

– виконуються контрольні тести в довільному порядку;

– робиться аналіз отриманих результатів і встановлюється відповідність програмного продукту вимогам і системним документам.

При виявлені помилок у роботі програмного продукту складається їх перелік і обговорюється термін їх виправлення розробником. Після цього замовник проводить повторне тестування в повному обсязі (можливе використання нових чи додаткових тестів).

6 Методи випробувань

Методом випробування розробленого програмного забезпечення є тестування чорної скриньки, яке представляє процес виконання програмного забезпечення з метою виявлення помилок. Кроки процесу задаються тестами.

Для перевірки працездатності розробленого телеграм-бота було проведено функціональне тестування основних сценаріїв використання системи. Результати тестування наведено в таблиці Д.1.

Таблиця Д.1 – Методика випробування телеграм-бота

№	Тестовий сценарій	Вхідні дані	Очікуваний результат	Фактичний результат
1	2	3	4	5
1	Запуск бота	Команда /start	Відображається привітальне повідомлення та головне меню	Відображено головне меню
2	Перегляд найближчих заходів	Натискання кнопки «Найближчі заходи»	Відображається список актуальних заходів	Список заходів відображено

Кінець таблиці Д.1

1	2	3	4	5
3	Пошук за категорією	Вибір категорії «Концерти»	Відображаються заходи обраної категорії	Відображено відповідні заходи
4	Перегляд детальної інформації	Вибір конкретного заходу	Відображається повна інформація про захід	Інформацію відображено коректно
5	Авторизація адміністратора	Введення коректних облікових даних	Виконується вхід до панелі адміністратора	Вхід виконано успішно
6	Додавання нового заходу	Заповнення форми нового заходу	Захід зберігається у базі даних	Дані збережено
7	Редагування заходу	Зміна опису заходу	Оновлена інформація зберігається	Дані оновлено
8	Видалення заходу	Вибір команди видалення	Захід видаляється з бази даних	Запис видалено
9	Імпорт даних із зовнішнього джерела	Запуск процедури імпорту	Дані завантажуються до черги модерації	Дані імпортовано
10	Модерація імпортованого заходу	Підтвердження імпортованого запису	Захід додається до афіші	Захід опубліковано
11	Некоректна команда користувача	Введення невідомої команди	Виводиться повідомлення про помилку	Повідомлення відображено
12	Пошук за відсутньою категорією	Категорія без заходів	Виводиться повідомлення про відсутність результатів	Повідомлення відображено