МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: РОЗРОБКА ПОГОДНОГО ПОМІЧНИКА НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ

Здобувача вищої освіти другого (магістерського) рівня,
ІІ курсу, групи КМІ-24-2, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології в економіці»



(підпис) Магури Маркіяна Романовича
(прізвище, ім'я, по батькові)

Науковий керівник 

(підпис) Ніжегородцев Владислав Олександрович, кан. пед. наук, доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Керівник освітньо-професійної програми 

(підпис) Омельчук Антон Анатолійович, канд. техн. наук
(прізвище, ім'я, по батькові, науковий ступінь, вчен звання)

Допущено до захисту за рішенням кафедри протокол від «02» січня 2026 № 10

Завідувач кафедри 

(підпис) Філоненко Михайло Миколайович, канд. фіз.-мат. наук, доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Реєстрація № 27 22.01.2026 (дата) (підпис відповідальної особи)

Результати перевірки до захисту (до захисту, на доопрацювання) 02.01.2026 (дата) (підпис)

Результат захисту 4 (добре) (оцінка) 05.02.2026 (дата) (підпис)

Голова комісії: (підпис) О.С. Дюбанов

Члени комісії:

(підпис) А.Ю. Горбовий

(підпис) М.М. Філоненко

(підпис) І.Д. Погорєловська

БЛАНК ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії,секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст кваліфікаційної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	24		05.02.2026 Дюбанов О.С.
2. Дотримання вимог щодо оформлення	від 0 до 10	8		Горбовий А.Ю.
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	3		Філоненко М.М. Погорєловська І.Д.
4. Рецензент	від 0 до 5	4		Суховецька Л.П.
РАЗОМ БАЛІВ	від 0 до 50	39		

Оцінювання кваліфікаційної роботи науковим керівником - 39 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалоюЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підпис)
78	C	4 (добре)	05.02.2026	Дюбанов О.С. Горбовий А.Ю. Філоненко М.М. Погорєловська І.Д. Суховецька Л.П.

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	3 (задовільно)
50-59	E	
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій
Кафедра комп'ютерних та інформаційних технологій і систем
Галузь знань 12 «Інформаційні технології»
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Інформаційні управляючі системи і технології в економіці»

Затверджую
Завідувач кафедри _____ Філоненко М.М.
«11» грудня 2024

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Магурі Маркіяну Романовичу

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи Розробка погодного помічника на основі штучного інтелекту
Затверджена наказом ректора від «10» грудня 2024 № 1852
2. Строк здачі закінченої роботи «02» січня 2026
3. Мета роботи Проєктування та програмна реалізація інтелектуального метеобота на базі платформи Telegram, який забезпечує користувачів достовірними метеорологічними даними у максимально доступній та зручній формі. Вибір месенджера зумовлений його високою популярністю, надійністю та гнучким інструментарієм для інтеграції зовнішніх сервісів через API.
Об'єкт дослідження Автоматизовані програмні агенти, що забезпечують користувачів актуальними прогнозами в режимі реального часу через звичні платформи обміну повідомленнями.
Предмет дослідження Надання миттєвих даних про поточний стан погоди.
Орієнтовна база дослідження Погодні додатки.
4. Вихідні дані до роботи Методи, алгоритми та програмні інструменти розробки інтелектуального чат-бота для збору, обробки та персоналізованої візуалізації метеоданих у середовищі Telegram.
5. Перелік графічного матеріалу Інтерфейс погодного помічника.
6. Зміст кваліфікаційної роботи (перелік питань за кожним розділом)
ВСТУП; РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ; АКТУАЛЬНІСТЬ ТЕМИ; ПОРІВНЯЛЬНИЙ АНАЛІЗ ПЛАТФОРМ ОБМІНУ ПОВІДОМЛЕННЯМИ ДЛЯ ІНТЕГРАЦІЇ ЧАТ-БОТІВ; ОСОБЛИВОСТІ РОЗРОБКИ ЧАТ-БОТІВ НА БАЗІ ПЛАТФОРМИ TELEGRAM; АЛГОРИТМ РОЗГОРТАННЯ ТА НАЛАШТУВАННЯ ЧАТ-БОТА В СЕРЕДОВИЩІ TELEGRAM; КОМПАРАТИВНИЙ АНАЛІЗ СЕРВІСІВ НАДАННЯ МЕТЕОРОЛОГІЧНИХ API; РОЗДІЛ 2. ВИБІР ЗАСОБІВ РЕАЛІЗАЦІЇ; РОЗДІЛ 3. ОПИС ТА ПРОГРАМНА РЕАЛІЗАЦІЯ. AI ПОГОДНИЙ ПОМІЧНИК; ВИСНОВКИ; СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ.

7. Календар план виконання роботи:

№ п/п	Назва етапів кваліфікаційноїроботи	Строк виконання етапів роботи	
		За планом	Фактично
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	До 15.12.2024	До 15.12.2024
2	**І розділ** (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	30.05.2025	30.05.2025
3	**ІІ розділ** (аналіз експериментальної частини про методики досліджень)	10.09.2025	10.09.2025
4	**ІІІ розділ** (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	20.11.2025	20.11.2025
5	**Висновки**	11.12.2025	11.12.2025
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	25.12.2025	25.12.2025
7	Отримання відгуку керівника	28.01.2026	28.01.2026
8	Отримання зовнішньої рецензії	30.01.2026	30.01.2026

8. Дата видачі завдання «11» грудня 2024

9. Керівник кваліфікаційної роботи _____ (підпис) Ніжегородцев В.О. (прізвище, ініціали)

10. Керівник освітньо-професійної програми _____ (підпис) Омельчук А.А. (прізвище, ініціали)

11. Завдання до виконання отримав здобувач вищої освіти _____ (підпис) Магура М.Р. (прізвище, ініціали)

АНОТАЦІЯ

РОЗРОБКА ПОГОДНОГО ПОМІЧНИКА НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ

У результаті проведеного теоретичного дослідження було обґрунтовано доцільність та технічну можливість створення метеорологічного чат-бота. Швидкий доступ до метеоданих через месенджери є критично важливим для широкого спектра галузей — від агросектору та будівництва до логістики й туризму. Інтеграція погодних сервісів у Telegram дозволяє задовольнити попит користувачів на отримання інформації в режимі «тут і зараз». Головною метою визначено створення інтуїтивно зрозумілого інтерфейсу для оперативного інформування про стан погоди. Для цього необхідно реалізувати стабільну інтеграцію з метеорологічними API, забезпечити персоналізацію даних (збереження локації користувача) та впровадити алгоритми обробки помилок для гарантування високої відмовостійкості системи. Telegram ідентифіковано як оптимальне середовище завдяки наявності потужного Telegram Bot API. Цей інструментарій надає розробнику гнучкі можливості для керування командами, надсилання медіаданих та взаємодії з користувачем, що значно спрощує процес програмної реалізації. Огляд існуючих аналогів (Weatherbot, Weatherman тощо) підтвердив високу затребуваність подібних рішень, водночас виявивши потенціал для вдосконалення функціонала, зокрема в частині надійності роботи та зручності кастомізації.

Кваліфікаційна робота містить 57 сторінок, 12 рисунків, 1 таблиця, список використаних джерел 37 найменувань.

Ключові слова: інформаційна система, погодний помічник, Telegram, API.

Рік захисту роботи 2026

ABSTRACT

DEVELOPMENT OF A WEATHER ASSISTANT BASED ON ARTIFICIAL INTELLIGENCE

As a result of the conducted theoretical research, the feasibility and technical feasibility of creating a meteorological chatbot were substantiated. Quick access to meteorological data via messengers is critically important for a wide range of industries - from the agricultural sector and construction to logistics and tourism. The integration of weather services in Telegram allows us to satisfy the user demand for information in the "here and now" mode. The main goal is to create an intuitive interface for prompt information about the weather. To do this, it is necessary to implement stable integration with meteorological APIs, ensure data personalization (saving the user's location) and implement error handling algorithms to guarantee high system fault tolerance. Telegram was identified as the optimal environment due to the presence of a powerful Telegram Bot API. This toolkit provides the developer with flexible capabilities for managing commands, sending media data and interacting with the user, which significantly simplifies the process of software implementation. A review of existing analogues (Weatherbot, Weatherman, etc.) confirmed the high demand for such solutions, while at the same time revealing the potential for improving the functionality, in particular in terms of reliability and ease of customization.

The qualification work contains 57 pages, 12 figures, 1 table, a list of sources used 37 items.

Keywords: information system, weather assistant, Telegram, API.

The year of the work is 2026

ЗМІСТ

ВСТУП 3

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 6

1.1. Актуальність теми 6

1.2. Порівняльний аналіз платформ обміну повідомленнями для інтеграції чат-ботів 7

1.3. Особливості розробки чат-ботів на базі платформи Telegram 10

1.4. Алгоритм розгортання та налаштування чат-бота в середовищі Telegram 12

1.5. Компаративний аналіз сервісів надання метеорологічних API 18

РОЗДІЛ 2. ВИБІР ЗАСОБІВ РЕАЛІЗАЦІЇ 24

РОЗДІЛ 3. ОПИС ТА ПРОГРАМНА РЕАЛІЗАЦІЯ. AI ПОГОДНИЙ ПОМІЧНИК 35

ВИСНОВКИ 52

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 54

ВСТУП

Актуальність теми. Метеорологічні умови є визначальним чинником у повсякденній діяльності людини, впливаючи на широкий спектр рішень — від побутової логістики до планування масштабних транспортних операцій. Попри високу доступність метеоданих у мережі Інтернет, проблема оперативного отримання точної та релевантної інформації залишається відкритою. Традиційні методи (вебсайти, мобільні додатки) часто вимагають зайвих маніпуляцій, що створює запит на більш нативні та зручні інструменти взаємодії, такі як розмовні інтерфейси.

Концепція рішення. Ефективним інструментом розв'язання цієї проблеми є впровадження спеціалізованих чат-ботів (метеоботів). Це автоматизовані програмні агенти, що забезпечують користувачів актуальними прогнозами в режимі реального часу через звичні платформи обміну повідомленнями. Використання таких ботів дозволяє персоналізувати контент на основі геолокації та вподобань користувача, перетворюючи пошук інформації на інтерактивний діалог.

Метою кваліфікаційної роботи є проєктування та програмна реалізація інтелектуального метеобота на базі платформи **Telegram**, який забезпечує користувачів достовірними метеорологічними даними у максимально доступній та зручній формі. Вибір месенджера Telegram зумовлений його високою популярністю, надійністю та гнучким інструментарієм для інтеграції зовнішніх сервісів через API.

Основні функціональні можливості розробки:

- **Моніторинг у реальному часі:** надання миттєвих даних про поточний стан погоди.
- **Деталізоване прогнозування:** формування погодинних та подобових прогнозів на основі інтеграції з надійними метеорологічними API.

- **Персоналізація за геолокацією:** автоматичне визначення

місцезнаходження користувача для надання локальних даних.

- **Система екстрених сповіщень:** автоматичне інформування про різкі зміни погодних умов та попередження про небезпечні метеоявища.
- **Аналітичний блок:** надання інформації про додаткові параметри (вологість, швидкість вітру, атмосферний тиск).

Об'єкт дослідження

Процес оперативного інформування населення про метеорологічні умови та небезпечні погодні явища за допомогою сучасних засобів комунікації (месенджерів).

Предмет дослідження

Методи, алгоритми та програмні інструменти розробки інтелектуального чат-бота для збору, обробки та персоналізованої візуалізації метеоданих у середовищі Telegram.

Мета роботи - проєктування та програмна реалізація автоматизованої інформаційної системи (метеобота) на платформі Telegram, яка забезпечує користувачів достовірними прогнозами погоди в режимі реального часу, враховуючи їхню геолокацію та індивідуальні налаштування сповіщень, для підвищення зручності доступу до критично важливої інформації.

Завдання дослідження

Для досягнення поставленої мети необхідно вирішити наступні завдання:

1. **Провести аналіз предметної області**, дослідивши сучасні джерела метеорологічних даних та популярні методи розробки розмовних інтерфейсів.
2. **Здійснити порівняльний огляд існуючих метеорологічних API** (таких як OpenWeatherMap, Weatherbit або Apple Weather) для вибору найбільш стабільного та точного джерела даних.
3. **Спроектувати архітектуру чат-бота**, включаючи логіку обробки запитів, структуру бази даних для зберігання налаштувань користувачів та

механізми інтеграції з зовнішніми сервісами.

4. **Вибрати оптимальний стек технологій** (наприклад, Python з бібліотеками aiogram або telebot) та обґрунтувати методи зберігання геолокаційних даних.
5. **Програмно реалізувати функціонал бота**, що включає миттєвий запит поточної погоди, формування погодинних/добових прогнозів та систему автоматичних сповіщень про суворі погодні умови.
6. **Провести комплексне тестування** розробленого продукту на предмет точності даних, швидкості відгуку та зручності взаємодії користувача з інтерфейсом (UI/UX).
7. **Оцінити ефективність впровадженого рішення** порівняно з традиційними метеорологічними сервісами.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Актуальність теми

Сучасна актуальність розробки метеорологічних ботів зумовлена трансформацією способів споживання контенту. Користувачі все частіше віддають перевагу отриманню персоналізованої інформації безпосередньо в месенджерах, що мінімізує часові витрати на перемикання між сторонніми мобільними застосунками та вебресурсами.

В умовах глобальних кліматичних змін та зростання частоти аномальних погодних явищ, оперативний доступ до верифікованих метеоданих стає критично важливим. Метеоботи дозволяють автоматизувати процес інформування, надаючи не лише загальні прогнози, а й адаптивні сповіщення, що базуються на геолокації та індивідуальних налаштуваннях користувача.

Окрім приватного використання, метеорологічні сервіси мають стратегічне значення для галузей, що критично залежать від природних умов:

- Сільське господарство: для планування польових робіт та захисту врожаю.
- Логістика та транспорт: для оптимізації маршрутів та гарантування безпеки перевезень.
- Туризм та сфера послуг: для підвищення якості обслуговування клієнтів та оперативного коригування планів заходів.

І найголовніше. Враховуючи повномасштабне вторгнення країни 404, необхідність в актуальному погодного помічнику є над актуальною.

Еволюція методів штучного інтелекту та обробки природної мови (NLP) дозволила значно вдосконалити інтерактивну складову таких ботів. Сучасні розробки здатні сприймати складні запити природною мовою, що забезпечує високий рівень залученості та покращує користувацький досвід (UX). Отже, поєднання метеорологічного аналізу з інструментами автоматизації залишається

перспективним напрямом, що постійно розвивається разом із технологіями передачі даних.

1.2. Порівняльний аналіз платформ обміну повідомленнями для інтеграції чат-ботів

Вибір середовища для розгортання метеорологічного бота залежить від функціональних можливостей платформи та доступних інструментів розробки. Розглянемо найбільш релевантні месенджери та їхні технічні особливості:

1. Платформа Telegram

Telegram є кросплатформним хмарним месенджером, що характеризується високим рівнем безпеки та розвиненою екосистемою для автоматизації [2].

- Технічні переваги: Використання спеціалізованого Telegram Bot API дозволяє створювати складні програмні рішення для оперативного надання прогнозів та сповіщень.
- Функціональні особливості: Платформа підтримує вбудовані (inline) режими, що дає змогу користувачам отримувати метеодані безпосередньо у діалоговому вікні без переходу на зовнішні ресурси. Наявність системи команд та вебхуків (webhooks) забезпечує високу швидкість обробки запитів.

2. Платформа Facebook Messenger

Facebook Messenger — це комунікаційний сервіс, глибоко інтегрований у соціальну екосистему Meta. Платформа орієнтована на максимальну інтерактивність та персоналізацію контенту.

- Технічні переваги: Розробка здійснюється через Messenger Platform, яка надає розширені можливості для маркетингової аналітики та взаємодії з аудиторією.
- Функціональні особливості: Сервіс пропонує набір елементів інтерфейсу, таких як швидкі відповіді (quick replies) та постійні меню (persistent menus). Ці інструменти дозволяють автоматизувати процес вибору локації або

типу прогнозу, роблячи взаємодію з ботом більш інтуїтивною.

3. Платформа WhatsApp

WhatsApp є одним із найбільш масових месенджерів у світі, що акцентує увагу на приватності та безпеці передачі даних.

- Технічні переваги: Розробка автоматизованих рішень базується на WhatsApp Business API, що дозволяє інтегрувати сервіси розсилки метеорологічних прогнозів та термінових сповіщень.
- Функціональні обмеження: Платформа вирізняється суворою політикою модерації та складним процесом верифікації розробників. Використання ботів регулюється жорсткими регламентами, що дещо ускладнює процес розгортання та масштабування некомерційних проєктів.

4. Платформа Slack

Slack орієнтований на корпоративний сегмент та автоматизацію робочих процесів усередині команд.

- Технічні переваги: Гнучкий Slack API дозволяє створювати глибоко інтегрованих ботів, що стають частиною робочого середовища.
- Функціональні особливості: Завдяки використанню інтерактивних компонентів (кнопок, діалогових вікон та меню), метеоботи в Slack можуть надавати персоналізовані звіти безпосередньо в канали розробки або планування, підвищуючи зручність для бізнес-користувачів.

5. Платформа Discord

Discord є провідним інструментом для ігрових спільнот та тематичних серверів, що пропонує широкі можливості для кастомізації.

- Технічні переваги: Використання Discord API надає розробникам доступ до розширеного функціоналу керування серверами.
- Функціональні особливості: Підтримка власних слеш-команд (slash commands) та реакцій дозволяє створювати максимально залучаючий інтерфейс.

Метеобот у цьому середовищі може не лише інформувати, а й інтегруватися в ігрові сценарії або події спільноти.

Обґрунтування вибору платформи

Підсумовуючи результати компаративного аналізу, слід зазначити, що вибір оптимального месенджера визначається цільовою аудиторією та сценаріями використання (use cases):

1. Масовий сегмент та персональне використання: Найбільш раціональним є вибір Telegram або Facebook Messenger. Ці платформи мають найнижчий поріг входження для розробника та забезпечують охоплення широкої бази користувачів.

2. Корпоративний та професійний сегмент: Для автоматизації сповіщень всередині робочих груп пріоритетним є Slack, зважаючи на його орієнтованість на продуктивність.

3. Спеціалізовані спільноти: Discord є найкращим рішенням для нішевих груп, де важлива інтерактивність та специфічні налаштування доступу.

Узагальнюючи вищевикладене, можна стверджувати, що метеоботи є універсальним інструментом із широким вектором застосування: від задоволення персональних потреб користувачів до розв’язання стратегічних завдань у сферах агропромислового комплексу, логістики, енергетики та управління надзвичайними ситуаціями.

Стрімка еволюція месенджерів, таких як **Telegram, Facebook Messenger, WhatsApp, Slack** та **Discord**, значно спростила процес інтеграції інтелектуальних агентів у повсякденну комунікацію. Кожна з розглянутих платформ надає унікальний набір інструментарію та API-можливостей, що дозволяє проектувати інтерактивні системи з високим рівнем залученості користувачів.

Проведений аналіз дозволяє зробити такі стратегічні висновки щодо вибору платформи:

1. **Для індивідуального використання:** Найбільш перспективними є **Telegram** та **Facebook Messenger** завдяки їхній масовості, доступності та низькому порогу входження для розробки.

2. **Для спеціалізованих середовищ:** Платформи **Slack** (корпоративний сектор) та **Discord** (ігрові та тематичні спільноти) є більш доцільними з огляду на їхні розширені можливості інтеграції в робочі та групові процеси.

Отже, вибір середовища розгортання метеобота має базуватися на комплексному аналізі конкретного сценарію використання (use case) та специфічних вимог цільової аудиторії до функціональності та зручності інтерфейсу.

З огляду на стрімкий прогрес у галузі штучного інтелекту, метеоботи трансформуються у високоефективні системи з глибокою персоналізацією та адаптивними інтерфейсами. Подальший розвиток цієї технології відкриває значні перспективи як для приватного використання, так і для бізнес-сектору, дозволяючи оптимізувати операційні процеси та суттєво покращити користувацький досвід (User Experience).

Впровадження таких інструментів є стратегічно важливим кроком для автоматизації інформування та підвищення якості взаємодії з даними в реальному часі.

Враховуючи результати проведеного компаративного аналізу, зокрема гнучкість налаштувань, простоту інтеграції та відкритість API, для реалізації даного проєкту обрано платформу **Telegram**.

1.3. Особливості розробки чат-ботів на базі платформи Telegram

Telegram є одним із провідних хмарних месенджерів, що поєднує високу функціональність (групові комунікації, передача медіафайлів великого обсягу) із розвиненою інфраструктурою для автоматизації. Ключовим інструментом для

розробників є потужний Bot API, який дозволяє створювати інтелектуальні сервіси з високим рівнем інтерактивності [2].

Переваги платформи:

1. **Масштабність аудиторії:** Стабільне зростання кількості активних користувачів (близько 1000 млн станом на 2025 рік) забезпечує широке охоплення потенційної аудиторії та легкість впровадження продукту.

2. **Доступність розробки:** Telegram Bot API вирізняється низьким порогом входження завдяки якісній документації та логічній структурі, що значно прискорює процес проєктування та розгортання.

3. **Розширений UI-інструментарій:** Наявність вбудованих (inline) клавіатур, підтримка різноманітних медіаформатів та можливість редагування повідомлень дозволяють створювати ергономічні та візуально привабливі інтерфейси.

4. **Високий рівень захищеності:** Використання протоколів шифрування та механізмів двофакторної автентифікації гарантує безпеку даних, що є критичним при обробці персональної або транзакційної інформації.

Обмеження платформи (Cons):

1. **Складність монетизації:** Порівняно з іншими платформами, Telegram має менш розвинені вбудовані інструменти для прямої комерціалізації сервісів.

2. **Аналітичні обмеження:** Базовий інструментарій платформи надає обмежений обсяг статистичних даних, що вимагає інтеграції сторонніх сервісів для глибокого аналізу поведінки користувачів.

3. **Специфіка пошуку:** Відсутність централізованого каталогу або механізму рекомендацій ботів усередині застосунку ускладнює органічне залучення нових користувачів.

4. **Екосистемна інтеграція:** Хоча платформа підтримує інтеграцію із зовнішніми службами, деякі функціональні зв'язки можуть бути менш глибокими порівняно з корпоративними месенджерами (наприклад, Slack).

Використання Telegram як бази для розробки метеобота є стратегічно виправданим рішенням для проєктів, орієнтованих на активну взаємодію з користувачем. Попри певні обмеження в аналітиці та маркетинговому просуванні, технічна гнучкість та безпека платформи роблять її оптимальним вибором для реалізації розмовних інтерфейсів

1.4. Алгоритм розгортання та налаштування чат-бота в середовищі Telegram

Процес створення інтелектуального агента на базі Telegram формалізований та складається з декількох послідовних етапів, що охоплюють як адміністративну реєстрацію, так і безпосередню програмну реалізацію [3]:

1. **Реєстрація облікового запису:** Первинним етапом є наявність верифікованого аккаунта в месенджері Telegram, що слугує платформою для подальшої взаємодії з інструментами розробника.

2. **Реєстрація бота через сервіс BotFather:** Використання спеціалізованого системного бота BotFather для ініціалізації нового об'єкта. Цей етап включає визначення глобального імені (Name) та унікального технічного ідентифікатора (Username).

3. **Генерація API-токена:** Після завершення реєстрації система видає унікальний ідентифікатор — API Token. Цей маркер є ключовим елементом автентифікації, який забезпечує безпечну взаємодію між сервером розробника та серверами Telegram Bot API.

4. **Програмна реалізація логіки:** Написання програмного коду з використанням обраної мови програмування (наприклад, **Python**) та спеціалізованих фреймворків (таких як python-telegram-bot, aiogram або Telegraf для Node.js). Це дозволяє реалізувати функціонал обробки вхідних запитів та формування відповідей.

5. **Деплоймент (розгортання):** Перенесення програмного коду на віддалений сервер або хмарну платформу (наприклад, **AWS, Google Cloud,**

Heroku). Це забезпечує безперебійну роботу бота в режимі 24/7.

6. **Альфа-тестування:** Проведення контрольних запитів безпосередньо через інтерфейс клієнта Telegram для перевірки базової працездатності та коректності відгуку системи.

7. **Ітеративне вдосконалення:** Етап комплексного тестування, спрямований на оптимізацію функціоналу, виправлення помилок та покращення користувацького досвіду (UX).

Хоча технічна процедура створення бота є стандартизованою та відносно простою, розробка високоефективного продукту потребує детального проєктування архітектури, врахування сценаріїв користувацької поведінки та постійної підтримки актуальності бази даних.

У межах дослідження було проведено огляд функціональних можливостей популярних метеорологічних сервісів, представлених на платформі Telegram. Серед найбільш релевантних аналогів було виокремлено наступні:

1. **MeteoBot** — спеціалізований програмний агент, що забезпечує користувачів метеоданими у реальному часі.

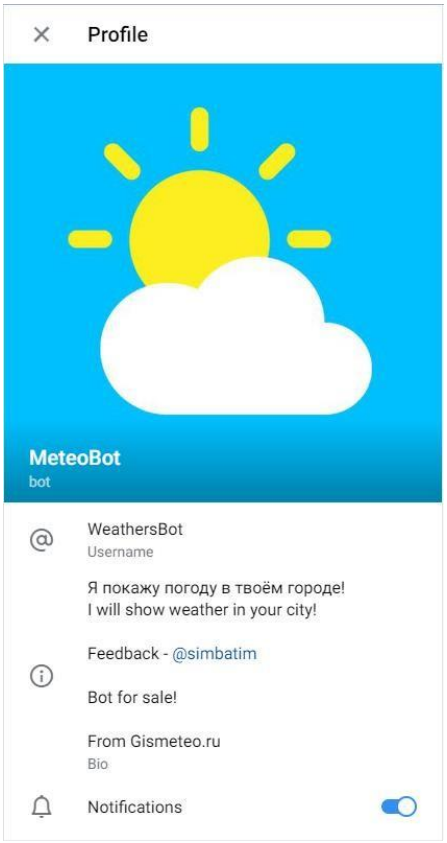


Рис.1.1. MeteoBot

Функціональні особливості: Сервіс підтримує моніторинг поточного стану атмосфери та надає детальні звіти.

Ключові переваги: Реалізована система превентивних сповіщень про критичні зміни погоди та небезпечні метеорологічні явища. Важливою характеристикою бота є багатомовний інтерфейс, що розширює коло потенційних користувачів та забезпечує доступність інформації на міжнародному рівні.

2. Бот @SynoptykBot — універсальний інформаційний сервіс, що забезпечує глобальне покриття метеорологічними даними.

- **Функціональні можливості:** Система генерує комплексні звіти, що охоплюють моніторинг поточної погодної ситуації, деталізовані погодинні прогнози, а також середньострокове планування терміном до 5 діб.
- **Технічні переваги:** Бот інтегрує функціонал превентивного

інформування про небезпечні атмосферні явища (штормові попередження, різкі перепади температур). Завдяки впровадженій багатомовності та відсутності географічних обмежень у пошуку, сервіс демонструє високу адаптивність до потреб користувачів у різних регіонах світу.

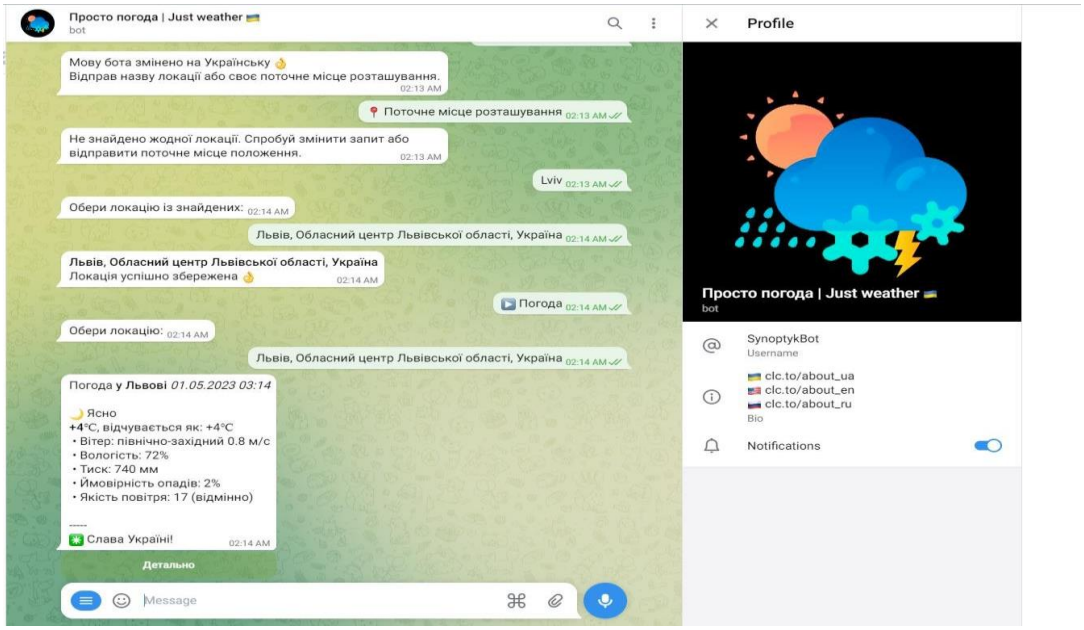


Рис. 1.2. Запуск бота

3. **Бот WeatherUKR_bot** — спеціалізований інформаційний ресурс, що вирізняється поглибленою деталізацією метеорологічних показників.

- **Функціональні можливості:** Головною особливістю сервісу є розширений горизонт прогнозування, який, окрім поточного моніторингу та погодинної деталізації, охоплює довгостроковий прогноз терміном до 15 діб.
- **Технічні переваги:** Система оснащена модулем оперативного оповіщення про небезпечні метеорологічні умови. Багатомовна підтримка та висока точність формування звітів роблять цей інструмент ефективним для довгострокового планування діяльності користувача в різних географічних зонах.

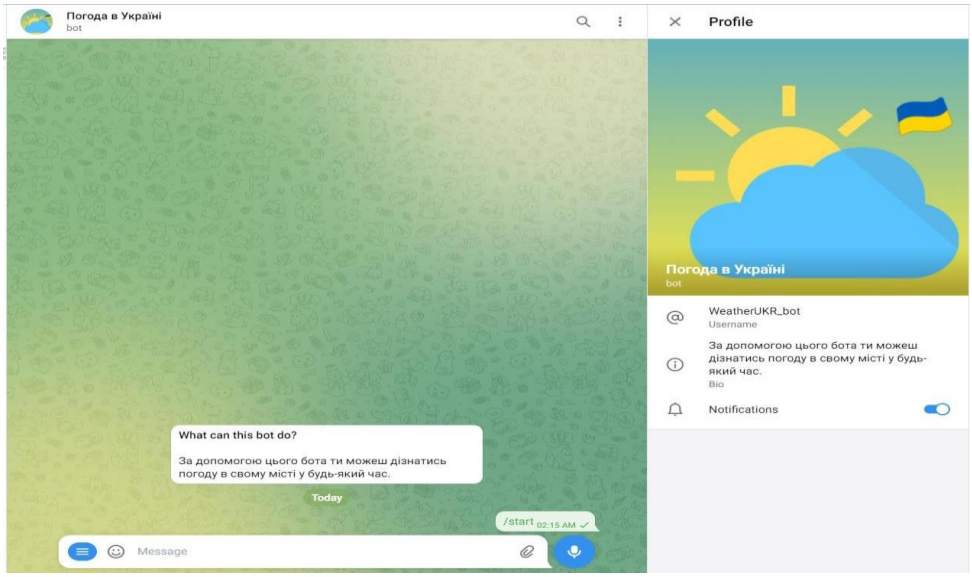


Рис. 1.3 WeatherUKR_bot

4. **Бот ForecastUkraine_bot** — інтерактивний сервіс метеорологічного моніторингу, адаптований для надання верифікованих прогнозів.

Функціональні можливості: Програмний продукт забезпечує доступ до розширених метеоданих, що включають актуальний стан погоди, деталізовані погодинні зміни та середньострокові прогнози з глибиною планування до 5 діб.

Технічні переваги: В системі реалізовано алгоритм автоматичної генерації попереджень про аномальні та небезпечні погодні явища. Завдяки підтримці мультиязичності та зручній структурі подання інформації, бот є ефективним інструментом для оперативного та стратегічного планування повсякденної діяльності.

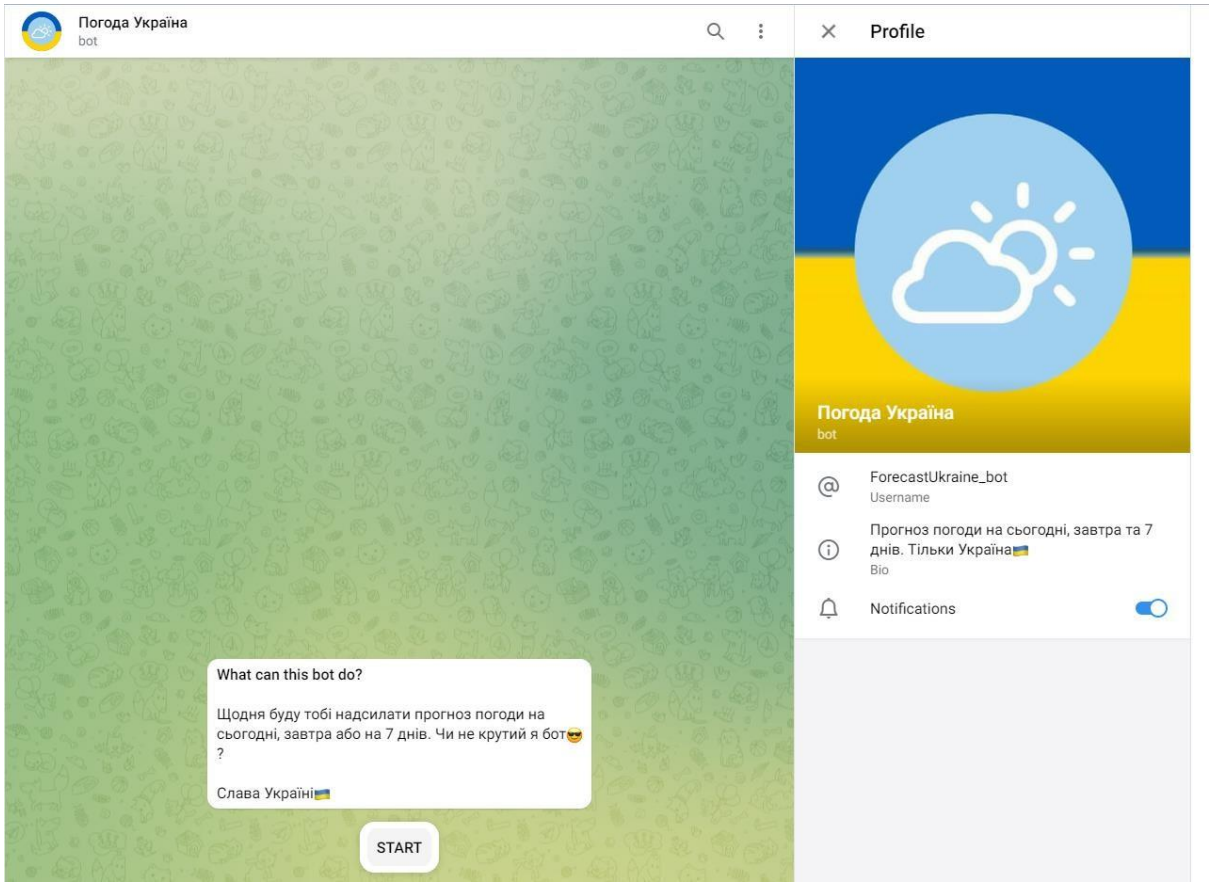


Рис.1.4 ForecastUkraine_bot

Резюмуючи огляд існуючих рішень, слід зазначити, що більшість метеорологічних ботів мають подібний функціональний набір: моніторинг поточного стану атмосфери, генерацію коротко- та середньострокових прогнозів, а також розсилку екстрених сповіщень. Проте, попри візуальну схожість, детальний аналіз виявив низку критичних аспектів:

1. **Варіативність джерел даних:** Точність та достовірність інформації суттєво залежать від обраного постачальника метеоданих (API). Різні сервіси демонструють неоднаковий рівень похибки для певних географічних регіонів.

2. **Технічна стабільність:** Практична експлуатація наявних аналогів свідчить про часті випадки некоректної роботи алгоритмів, затримки у відгуку інтерфейсу або неактуальність наданих сповіщень.

3. **Персоналізація:** Значна частина ботів пропонує шаблонні рішення, які не завжди враховують специфічні запити користувача щодо глибини

аналітики або зручності UI/UX. З огляду на виявлені недоліки та нестабільність роботи існуючих сервісів, постає доцільність розробки авторського метеорологічного бота. Створення власного програмного продукту дозволить самостійно обрати найбільш релевантні API, оптимізувати архітектуру для стабільної роботи та реалізувати гнучкий функціонал, що максимально відповідає сучасним вимогам точності та надійності.

1.5. Компаративний аналіз сервісів надання метеорологічних API

Для забезпечення коректного функціонування метеоботи потребують інтеграції з надійними зовнішніми джерелами даних. Процес вибору оптимального сервісу (API-провайдера) є критичним етапом, що безпосередньо впливає на точність та швидкість роботи системи. Розглянемо найбільш поширені рішення:

1. Сервіс OpenWeatherMap

OpenWeatherMap позиціонується як одна з найбільш універсальних та популярних платформ для отримання метеорологічних даних за допомогою програмного інтерфейсу (API).

Функціональні можливості: Сервіс надає розширений спектр параметрів, що охоплюють як базові показники (температура, атмосферний тиск, рівень вологості), так і складні метеодані (швидкість та напрямок вітру, хмарність, ультрафіолетовий індекс тощо).

Технічні особливості: Платформа підтримує генерацію як поточних даних у режимі реального часу, так і короткострокових та довгострокових прогнозів. Завдяки високій частоті оновлення бази знань та підтримці формату JSON, інтеграція з цим сервісом дозволяє будувати динамічні та відмовостійкі архітектури чат-ботів.

Переваги та недоліки:

- Широкий спектр даних доступний через API.
- Безкоштовний та платні плани для різних обсягів запитів та функцій.
- Можливі обмеження в безкоштовному плані щодо обсягу запитів та швидкості оновлення даних.

2. Сервіс Weatherbit

Weatherbit є високотехнологічною платформою, що спеціалізується на наданні метеорологічних API з акцентом на низьку затримку (latency) та високу точність даних.

Функціональні можливості: Система підтримує розгалужену структуру запитів, що дозволяє отримувати не лише актуальні метеопоказники та середньострокові прогнози, а й доступ до глибоких архівів історичних погодних даних. Це робить сервіс ефективним для проведення ретроспективного аналізу.

Умови експлуатації та масштабованість: Платформа пропонує гнучку систему тарифних планів. Наявність безкоштовного рівня доступу (Free Tier) дозволяє проводити розробку та тестування прототипів (MVP), тоді як комерційні плани забезпечують розширені ліміти на кількість запитів за хвилину та гарантовану технічну підтримку, що важливо для високонавантажених систем.

Переваги та недоліки:

- Різні типи запитів, включаючи поточну погоду, прогнози та історичні дані.
- Безкоштовний план та платні плани з розширеними можливостями.
- Можливі обмеження в безкоштовному плані щодо обсягу запитів та деталізації даних.

3. Сервіс AccuWeather

AccuWeather є одним із найбільш авторитетних світових провайдерів, що спеціалізується на наданні метеорологічних даних високої точності з використанням власних прогнозних моделей.

Функціональні можливості: Окрім стандартних метрик поточного стану

та прогнозних значень, сервіс надає доступ до специфічних типів даних, зокрема радарних зображень високої роздільної здатності та супутникової візуалізації метеофронтів. Це дозволяє реалізувати в чат-боті розширений графічний контент.

Інфраструктура доступу: Як і конкурентні рішення, AccuWeather використовує багаторівневу модель доступу. Безкоштовна версія API (Limited Free Trial) підходить для ознайомлення з функціоналом, проте комерційні ліцензії є необхідними для отримання доступу до преміальних функцій та обробки значних масивів даних без затримок.

Переваги та недоліки:

- Точні погодові дані та прогнози на кілька днів.
- Безкоштовний план та платні плани для розширених функцій.
- Обмеження в безкоштовному плані щодо обсягу запитів та доступу до певних функцій.

4. Сервіс Weather Underground

Weather Underground представляє собою унікальну метеорологічну платформу, функціонування якої базується на інтеграції даних з розгалуженої мережі приватних та професійних метеостанцій (PWS) по всьому світу.

Функціональні можливості: Завдяки використанню краудсорсингових даних, сервіс забезпечує доступ до гіперлокальних метеопоказників. Набір даних включає деталізовані прогнози, глибокі масиви історичної інформації, радарні знімки та специфічні дані про стан земної поверхні.

Інструментарій розробника: Платформа пропонує гнучку модель взаємодії через API. Поряд із безкоштовним рівнем доступу для некомерційних та тестових проєктів, існують преміальні плани, що знімають обмеження на кількість запитів та відкривають доступ до високоточних аналітичних інструментів і додаткових функціональних модулів.

Переваги та недоліки:

- Доступ до розширеного набору погодових даних.

- Безкоштовний план та платні плани з додатковими функціями.
- Обмеження в безкоштовному плані щодо обсягу запитів та доступу до деяких функцій.

Ось варіант перефразування, де особливу увагу приділено актуальному статусу сервісу та його інтеграції в екосистему Apple, що критично важливо для актуальності вашого дослідження:

5. Сервіс Dark Sky

Dark Sky довгий час вважався еталонним інструментом для отримання гіперлокальних метеоданих, що базувалися на передових алгоритмах короткострокового прогнозування (nowcasting).

Функціональні можливості: Платформа спеціалізувалася на наданні надточних прогнозів з високою дискретністю (до хвилини), а також забезпечувала доступ до радарної візуалізації та погодинної аналітики атмосферних змін. **Сучасний статус та обмеження:** Слід зазначити, що після придбання сервісу корпорацією **Apple**, стратегія надання послуг зазнала суттєвих змін. З 2022 року Dark Sky офіційно припинив реєстрацію нових розробників та інтегрував свої технології у пропрієтарний фреймворк **Apple WeatherKit**. Наразі оригінальний API Dark Sky є недоступним для нових проєктів, а його функціонал підтримується лише в межах екосистеми Apple для існуючих клієнтів або через нові інструменти розробки компанії.

Переваги та недоліки:

- Гіперлокальна погодова інформація з точними прогнозами.
- Потужні функції, включаючи радарні зображення та годинні прогнози.
- Припинено прийом нових користувачів та перехід під керівництво Apple.

Зазначені сервіси репрезентують собою потужний інструментарій для інтеграції верифікованої метеоінформації в програмні продукти будь-якої складності. Використання їхніх API дозволяє розробникам автоматизувати

процес отримання актуальних показників стану атмосфери, забезпечуючи високу точність та релевантність даних у кінцевих додатках.

Процес ідентифікації оптимального API-провайдера має базуватися на комплексному аналізі наступних критеріїв:

Технічні вимоги проєкту: необхідна глибина прогнозування та специфікація метеорологічних параметрів.

Інтенсивність експлуатації: очікуваний обсяг запитів до сервера та здатність системи до масштабування.

Функціональне наповнення: доступність додаткових інструментів, таких як радарні карти, історичні архіви або гіперлокальні сповіщення.

Економічна доцільність: відповідність вартості платних тарифів бюджету розробки та ефективність безкоштовних рівнів доступу (Free Tiers).

Таким чином, остаточне рішення щодо вибору постачальника даних є балансом між технічними можливостями платформи та стратегічними цілями розроблювального погодного бота.

Висновки до розділу 1

Підсумовуючи проведене дослідження, можна стверджувати, що метеоботи на платформі Telegram є високоефективним інструментом для оперативного доступу до погодних даних у глобальному масштабі. Сфера їхнього застосування охоплює широкий спектр — від задоволення персональних потреб користувачів до інтеграції в бізнес-процеси, логістику та туристичні сервіси.

Telegram було ідентифіковано як надійну екосистему для розгортання таких інтелектуальних агентів. Платформа пропонує розробникам гнучкий інструментарій, підтримку численних мов програмування та розвинені бібліотеки, що суттєво знижує технічні бар’єри для реалізації власних рішень.

Аналіз існуючих аналогів показав наступне:

Незважаючи на велику кількість доступних ботів, кожен з них має унікальний набір функцій, орієнтований на певні сценарії використання та регіони.

Критичним аспектом залишається точність та верифікація даних, оскільки надійність прогнозів безпосередньо залежить від обраного метеорологічного провайдера (API).

Користувачі часто змушені проводити порівняльне тестування сервісів для пошуку рішення, що забезпечує найвищу коректність даних для їхньої локації.

Загалом, впровадження погодних ботів у месенджери забезпечує зручний, персоналізований та технологічний підхід до моніторингу навколишнього середовища. Це дозволяє користувачам ефективно планувати діяльність у реальному часі, що робить розробку таких систем актуальним і перспективним напрямком у сфері інформаційних технологій.

РОЗДІЛ 2. ВИБІР ЗАСОБІВ РЕАЛІЗАЦІЇ

Для програмної реалізації метеорологічного чат-бота було обрано Python — високорівневу об’єктно-орієнтовану мову програмування. Створена Гвідо ван Россумом у 1990 році, мова отримала свою назву як омаж британському комедійному шоу «Flying Circus Monty Python», а не на честь плазуна, як прийнято вважати [7]. Синтезувавши кращі концепції таких мов, як C, C++, Java, ABC та Modula-3, Python став потужним інструментом з низкою стратегічних переваг:

Відкритий початковий код: Забезпечує прозорість розробки та можливість вільного використання широкої бази бібліотек.

Лаконічний синтаксис: Мінімізує кількість шаблонного коду, що підвищує читабельність та прискорює процес написання скриптів.

Потужний математичний апарат: Наявність спеціалізованих інструментів для складних обчислень та обробки даних.

Модульна архітектура: Можливість підключення величезної кількості зовнішніх модулів для розширення функціоналу.

Кросплатформеність: Програми, написані на Python, є переносними та стабільно функціонують у середовищах Windows, macOS, Android, iOS, а також у різноманітних дистрибутивах UNIX/Linux.

Серед ключових особливостей, що виділяють Python серед інших мов програмування, слід зазначити:

1. **Інтерактивний режим:** Мова підтримує інтерпретований підхід, що дозволяє виконувати вирази безпосередньо після введення. Це значно спрощує етап налагодження коду та тестування окремих алгоритмів у реальному часі.

2. **Досконала ООП-модель:** Об’єктно-орієнтований підхід у Python реалізований надзвичайно логічно та елегантно. Попри певну специфічність порівняно з іншими мовами, ця парадигма забезпечує високу потужність при проектуванні складних програмних структур, дозволяючи створювати гнучкий та масштабований код.

Додатковими технологічними перевагами Python, що зумовили вибір цієї мови для розробки метеобота, є:

Підтримка функціональної парадигми: Мова дозволяє використовувати елементи функціонального програмування (такі як лямбда-вирази, функції вищого порядку map, filter), що забезпечує гнучкість при обробці потоків погодних даних.

Модульність та пакетна архітектура: Програмна структура Python базується на ієрархії модулів, які об'єднуються в пакети. Це не тільки спрощує підтримку коду, але й дозволяє інтегрувати компоненти, написані іншими мовами програмування (наприклад, C/C++), для оптимізації критичних за швидкістю ділянок.

Інтроспекція об'єктів: Потужний механізм аналізу, який дозволяє під час виконання програми отримувати повну інформацію про внутрішню структуру та властивості будь-якого об'єкта, що значно полегшує налагодження та динамічну розробку.

Механізми обробки винятків: Розвинена система перехоплення та обробки помилок, що гарантує відмовостійкість бота при некоректних відповідях від зовнішніх API.

Ітератори та генератори: Ефективні інструменти для роботи з послідовностями даних, що дозволяють оптимізувати використання оперативної пам'яті при обробці великих масивів метеорологічної інформації.

Декоратори: Елегантний інструмент для модифікації поведінки функцій, що широко використовується у фреймворках для Telegram-ботів (наприклад, для реєстрації обробників повідомлень). Окремої уваги заслуговує **розширена стандартна бібліотека**. Філософія «batteries included» забезпечує розробника всім необхідним інструментарієм безпосередньо «з коробки», включаючи модулі для реалізації HTTP-серверів та клієнтів, що є фундаментальним при роботі з мережевими запитами до погодних сервісів.

BotFather

BotFather є фундаментальним інструментом адміністрування, розробленим

командою Telegram для забезпечення життєвого циклу користувацьких ботів. Він функціонує як централізована панель управління, що дозволяє розробнику взаємодіяти з платформою на рівні конфігурації та реєстрації програмних об’єктів [6].

До ключових функціональних можливостей сервісу належать:

Реєстрація та ідентифікація: BotFather дозволяє ініціалізувати необмежену кількість ботів у межах одного облікового запису розробника. Головною умовою реєстрації є створення унікального системного імені (username), яке має бути автентичним у загальній базі платформи.

Кастомізація інтерфейсу: За допомогою набору команд сервіс дозволяє редагувати зовнішні параметри чат-бота, зокрема встановлювати профільне зображення (аватар), формулювати опис (Description) та розділ «Про бота» (About), що безпосередньо впливає на користувацький досвід (UX).

Управління параметрами: Окрім реєстрації, інструмент забезпечує можливість оновлення технічних налаштувань, генерації API-токенів та керування дозволами бота (наприклад, доступ до груп або режим конфіденційності).

Етапи реєстрації та технічної ініціалізації бота

Процедура створення програмного об’єкта в середовищі Telegram за допомогою сервісу **BotFather** формалізована та складається з трьох ключових етапів:

1. Призначення екранного імені (Display Name): Визначення назви бота, яка буде візуалізуватися в інтерфейсі чатів та списку контактів користувачів. Це ім’я не обов’язково має бути унікальним і може містити довільні символи.

2. Генерація унікального ідентифікатора (Username): Створення технічного імені бота, яке є ключем для його пошуку в глобальній мережі Telegram. До цього параметру висуваються суворі вимоги:

Унікальність: ім'я не повинно дублювати вже існуючі в системі записи.

Синтаксис**:** дозволено використання лише латинських літер, цифр та символу підкреслення.

Формат: обов’язкове закінчення на слово «bot» (наприклад, weather_informer_bot). У разі невідповідності критеріям BotFather відхиляє запит і потребує повторного введення.

3. Отримання API-токена: Після успішної валідації назви та ідентифікатора система генерує унікальний буквено-цифровий код (токен). Цей маркер є основним інструментом автентифікації, що надає повний програмний доступ до управління ботом та дозволяє здійснювати взаємодію з Telegram Bot API.

Telegram Bot API

Функціонування месенджера Telegram базується на власному протоколі шифрування MTProto. Для спрощення розробки сторонніх сервісів було впроваджено Telegram Bot API — високорівневу надбудову, яка виступає посередником між програмістом та складним протоколом MTProto. Це дозволяє розробнику взаємодіяти з серверами Telegram за допомогою стандартних HTTPS-запитів, не заглиблюючись у технічні деталі шифрування [4].

Механізми отримання оновлень

Для синхронізації бота з діями користувачів передбачено два основні методи отримання даних (Updates):

- **Long Polling:** Додаток ініціює запити до сервера Telegram з певною періодичністю, перевіряючи наявність нових подій. Цей метод є оптимальним для етапу розробки та тестування.
- **Webhooks:** Сервер Telegram автоматично надсилає HTTP-сповіщення на сервер розробника відразу після виникнення події. Це забезпечує мінімальну затримку та вищу ефективність у промисловій експлуатації.

Структура та ідентифікація запитів

Кожен програмний об'єкт ідентифікується унікальним буквено-цифровим маркером (API Token), наприклад: 123456:ABC-DEF1234ghIkl-mno5678pqRs. Токен є ключем для формування URL-адреси запиту за наступним шаблоном: https://api.telegram.org/bot<token>/METHOD_NAME

Система підтримує методи GET та POST. Для передачі параметрів

використовуються наступні типи контенту (Content-Type):

1. URL query string — передача даних безпосередньо в рядку запиту.

2. application/x-www-form-urlencoded — стандартне кодування форм.

3. application/json — основний формат для передачі структурованих даних (окрім медіафайлів).

4. multipart/form-data — спеціалізований формат для завантаження та передачі файлів.

Обробка відповідей сервера

Результати виконання запитів повертаються у форматі **JSON**. Структура стандартної відповіді включає наступні поля:

- ok (boolean) — індикатор успішності операції.
- result — об'єкт, що містить запитані дані у разі успіху.
- description (string) — текстовий опис результату або деталізація помилки.
- error_code (integer) — числовий код помилки (повертається лише при ok: false).

Така модель забезпечує високу прозорість взаємодії та спрощує обробку виняткових ситуацій під час розробки алгоритмів бота.

Функціональні методи та об'єктна модель Telegram Bot API

Для керування логікою бота та взаємодії з платформою розробнику доступний широкий набір методів, що реалізують базовий та розширений функціонал [4]:

- **getUpdates**: базовий метод для отримання вхідних подій шляхом періодичного опитування сервера (Long Polling).
- **setWebhook**: інтегрує бота із зовнішньою URL-адресою для автоматичного отримання оновлень через HTTPS POST-запити у форматі JSON.
- **deleteWebhook**: видаляє прив'язку до домену, що є необхідною умовою для відновлення роботи через getUpdates.
- **sendMessage**: основний метод для надсилання текстових повідомлень.
- **sendLocation**: передає географічні координати, що критично важливо

для погодних сервісів.

- **getFile**: забезпечує доступ до завантажених на сервер медіафайлів.

Механізми обробки вхідних даних

Telegram дозволяє розробникам використовувати стандартні серверні конфігурації платформи або розгортати власні локальні сервери (Bot API server) для зняття обмежень на обсяг переданих даних.

Основні типи об'єктів (JSON Data Types)

Дані в API представлені у вигляді структурованих JSON-об'єктів, що описують сутності системи:

Тип об'єкта	Функціональне призначення
User	Містить інформацію про конкретного користувача або іншого бота.
Chat	Описує простір взаємодії (приватний діалог, група або канал).
Message	Комплексний об'єкт, що містить текст, метадані та вкладення.
MessageId	Унікальний цілочисельний ідентифікатор повідомлення в межах чату.
Animation	Обробка анімованих файлів (GIF або відео у форматі H.264 без аудіодоріжки).
Audio / Voice	Диференціація між музичними файлами та короткими голосовими повідомленнями.

Важливою особливістю архітектури є те, що методи getUpdates та setWebhook є взаємовиключними. При активному веб-хуку Telegram автоматично припиняє накопичення оновлень у черзі для Long Polling. У разі недоступності сервера розробника (невдалий статус HTTP-запиту), система робить декілька повторних спроб, після чого тимчасово призупиняє розсилку до відновлення стабільного з'єднання.

Чому це важливо для розробки:

1. **Конфлікт методів:** Якщо ви тестуєте бота локально через getUpdates, переконайтеся, що раніше не був встановлений веб-хук, інакше бот не реагуватиме.

2. **Об'єкт Chat:** У ньому міститься id, за яким ваш бот буде надсилати прогноз погоди конкретному користувачу.

3. **Об'єкт Message:** Саме з нього ваш код буде витягувати назву міста або локацію для запиту до метеосервісу.

Функціональні можливості та об'єктна модель Telegram Bot API

Для взаємодії з платформою розробнику надається широкий перелік методів, що дозволяють керувати логікою бота та обробляти вхідні дані [4]. Серед базових методів слід виділити:

- **getUpdates** — отримання черги вхідних подій;
- **setWebhook** — встановлення зв'язку з певним URL-доменом для автоматичного отримання оновлень;
- **deleteWebhook** — анулювання поточної прив'язки до веб-хуку;
- **sendMessage** — надсилання текстової інформації користувачу;
- **sendLocation** — трансляція географічних координат;
- **getFile** — отримання доступу до завантажених об'єктів.

Хоча більшість рішень використовує стандартну хмарну конфігурацію Telegram, архітектура API дозволяє розробникам за потреби переходити на власні локальні сервери для розширення лімітів та функціоналу.

Механізми отримання оновлень

Існує сувора диференціація між методами отримання даних. Метод **getUpdates** базується на технології *long polling* (довготривале опитування), тоді як **setWebhook** дозволяє серверу Telegram самостійно ініціювати POST-запити з JSON-пакетами на адресу розробника.

Важливо враховувати, що ці механізми є взаємовиключними: доки активний веб-хук, отримання даних через getUpdates стає неможливим. У разі виникнення помилок при передачі через веб-хук, Telegram здійснює обмежену

кількість повторних спроб, після чого припиняє відправку до відновлення стабільного з'єднання.

Структура даних та основні класи

Усі типи даних у Bot API представлені у формі JSON-об'єктів. Нижче наведено ключові типи, що формують основу архітектури повідомлень:

- **User** — ідентифікує конкретного користувача або програмного агента (бота);
- **Chat** — описує середовище взаємодії (приватний діалог, групу чи канал);
- **Message** — комплексний об'єкт, що містить зміст повідомлення та його метадані;
- **MessageId** — унікальний цілочисельний індекс повідомлення;
- **Animation** — об'єкт для роботи з анімованим контентом (GIF або відео без звуку);
- **Audio / Voice** — дозволяють розрізняти повноцінні музичні файли та короткі голосові записи.

Чому це важливо для розробки:

1. **Тип Chat:** У вашому коді саме з цього об'єкта буде вилучатися chat_id для надсилання прогнозу.

2. **Тип Voice:** Якщо ви захочете додати функцію запиту погоди голосом, вам доведеться працювати саме з цим типом даних.

3. **Метод deleteWebhook:** Цей метод часто використовується на початку розробки, щоб скинути старі налаштування і почати тестувати бота локально.

API OpenWeatherMap

OpenWeatherMap API — це універсальний хмарний сервіс, що забезпечує програмний доступ до глобальних масивів метеорологічної інформації. Використання даного API дозволяє інтегрувати в додатки динамічні дані про стан атмосфери, ретроспективні показники та аналітичні прогнози для будь-якої географічної точки планети [16].

Ключовий функціонал сервісу:

- **Моніторинг поточної ситуації:** Надання деталізованої інформації про стан погоди в режимі реального часу. Набір параметрів включає не лише базові показники температури, а й вектор швидкості вітру, рівень вологості, атмосферний тиск та індекси хмарності.
- **Прогностична аналітика:** Генерація короткострокових та середньострокових прогнозів. API дозволяє отримувати дані про очікувану динаміку атмосферних явищ, включаючи ймовірність опадів та температурні коливання на кілька діб вперед.
- **Гнучка географічна прив'язка:** Система підтримує кілька методів ідентифікації локації — за назвою населеного пункту, унікальним кодом міста або шляхом передачі точних географічних координат (широти та довготи), що забезпечує високу точність локалізації запитів.

Переваги:

Застосування даного API у процесі розробки метеорологічного бота обумовлено низкою стратегічних переваг, що забезпечують високу якість підсумкового програмного продукту:

- **Глобальне охоплення та універсальність:** Сервіс оперує масштабною мережею даних, що охоплює практично всі географічні регіони світу. Це дозволяє створювати масштабовані додатки з підтримкою глобального позиціонування, незалежно від локації користувача.
- **Мультипараметрична структура даних:** Платформа надає вичерпний набір метеорологічних показників — від базових (температура, вологість, атмосферний тиск) до специфічних (векторний аналіз вітру, рівень хмарності, ймовірність опадів). Такий обсяг інформації дозволяє реалізувати складну функціональність: від миттєвих звітів до систем раннього попередження про зміну погодних умов.
- **Низький поріг входження та якісна документація:** Архітектура API вирізняється логічністю та простотою інтеграції. Наявність структурованої технічної документації та готових прикладів (SDK) дозволяє мінімізувати часові

витрати на етапі імплементації сервісу в програмний код.

- **Розширений аналітичний інструментарій:** Окрім стандартних запитів, OpenWeatherMap пропонує доступ до історичних метеоархівів та засобів графічної візуалізації. Це дає можливість розробнику будувати не просто інформаційні, а потужні аналітичні системи з ретроспективним аналізом та наочним відображенням метеорологічних трендів.

Недоліки:

Поряд із суттєвими перевагами, впровадження OpenWeatherMap API супроводжується певними технологічними та експлуатаційними обмеженнями, які необхідно враховувати при проектуванні архітектури бота:

- **Квотування та ліміти запитів:** Безкоштовний рівень доступу (Free Tier) має жорсткі обмеження щодо кількості запитів за хвилину та добу. Для високонавантажених систем або проектів із великою аудиторією виникає потреба в переході на комерційні тарифні плани для забезпечення стабільного доступу до розширених масивів даних.
- **Динамічність та потенційна похибка даних:** Специфіка метеорологічних спостережень передбачає високу швидкість зміни атмосферних фронтів. Через затримки в оновленні бази даних на серверах провайдера іноді можуть виникати ситуації короткочасної невідповідності прогнозу реальним погодним умовам.
- **Нерівномірність географічного покриття:** Хоча сервіс є глобальним, щільність метеостанцій у віддалених або малонаселених регіонах може бути недостатньою. Це призводить до зниження точності прогнозів або отримання неповних метеоданих для певних локацій.
- **Ризики сторонньої залежності:** Робота додатка критично залежить від стабільності зовнішньої інфраструктури OpenWeatherMap. Б

Будь-які технічні збої на стороні провайдера, проведення профілактичних робіт або зміни в політиці доступу до API можуть безпосередньо вплинути на працездатність чат-бота.

Загалом, OpenWeatherMap API виступає надійним та потужним

фундаментом для розробки метеорологічних сервісів. Проте для створення якісного програмного продукту розробнику необхідно впроваджувати механізми обробки помилок, кешування даних та враховувати ліміти безкоштовного доступу на етапі проектування.

РОЗДІЛ 3. ОПИС ТА ПРОГРАМНА РЕАЛІЗАЦІЯ. AI ПОГОДНИЙ ПОМІЧНИК

Мета проекту: Створення інтелектуального Telegram бота, який надає детальну інформацію про погоду та генерує персоналізовані рекомендації на основі передової мовної моделі Mistral AI.

Основна ідея: Об'єднати точні погодні дані з потужностями штучного інтелекту для надання користувачам не просто інформації про погоду, а розумних, контекстних рекомендацій щодо одягу, активностей та здоров'я.

Актуальність: У сучасному світі люди потребують швидкого доступу до актуальної інформації та розумних порад. Інтеграція AI технологій з популярними месенджерами дозволяє надавати персоналізовані рекомендації прямо в зручному інтерфейсі Telegram.

Концепція

AI Погодний Помічник - це Telegram бот, який:

1. Отримує погодні дані з OpenWeatherMap API для будь-якого міста світу
2. Аналізує умови за допомогою Mistral AI мовної моделі
3. Генерує рекомендації щодо:
 - Одягу залежно від погоди
 - Активностей, які підходять для поточних умов
 - Здоров'я та безпеки
 - Спеціальних попереджень

Унікальність

Відмінності від звичайних погодних ботів:

- Звичайні боти: Тільки температура та опис
- Наш бот: Температура + AI рекомендації + персоналізація

Переваги AI підходу:

- Розумний аналіз всіх погодних параметрів одночасно

- Контекстні рекомендації, адаптовані під конкретні умови
- UA Природна українська мова в рекомендаціях
- Швидка генерація через хмарний API

Цільова аудиторія

- Люди, які планують активності на день
- Подорожуючі, які потребують інформації про погоду
- Спортсмени та любителі активного відпочинку
- Користувачі, які цінують розумні технології

Опис проєкту

Функціональні вимоги

Основні функції:

1. Отримання погодних даних для будь-якого міста
2. Відображення детальної інформації про погоду
3. Генерація AI рекомендацій через Mistral API
4. Статистика використання бота
5. Команди для навігації та допомоги

Додаткові функції:

- Обробка помилок та fallback режими
- Індикатори завантаження
- Красиве форматування повідомлень
- Підтримка української мови

Нефункціональні вимоги

Продуктивність:

- Швидкість відповіді: < 5 секунд
- Обробка помилок: graceful degradation
- Надійність: робота навіть при часткових збоях API

Безпека:

- Захист API ключів

- Валідація введених даних

- Обмеження довжини повідомлень

Масштабованість:

- Підтримка багатьох користувачів одночасно

- Асинхронна обробка запитів

- Готовність до розширення функціоналу

Технічні характеристики

Мова програмування: Python 3.8+

Фреймворк: aiogram 3.3.0

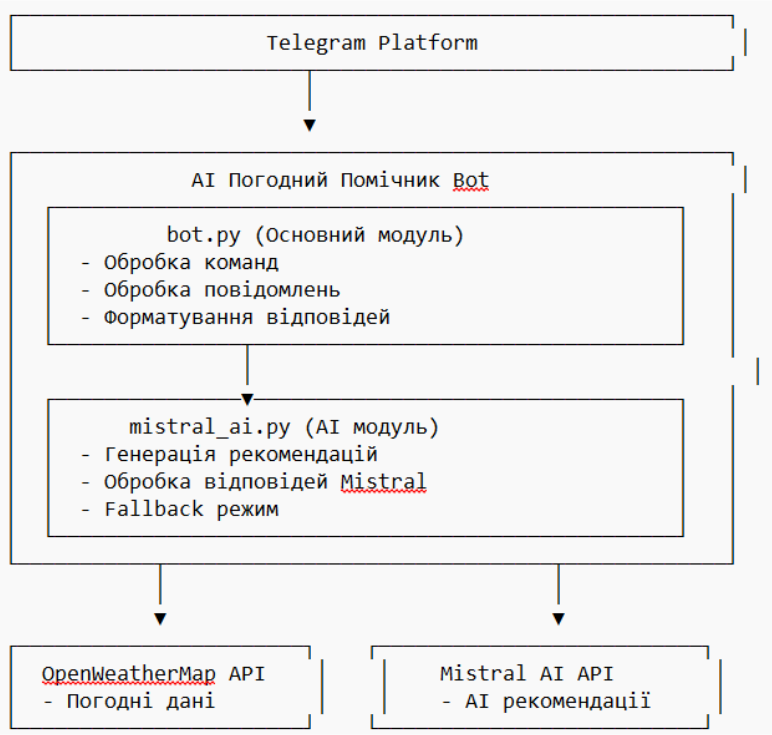
AI модель: Mistral AI (mistral-small-latest)

Погодний API: OpenWeatherMap

Архітектура: Модульна, асинхронна

Архітектура

Загальна архітектура



Компоненти системи

1. Презентаційний рівень (Presentation Layer)

Модуль: `bot.py`

Відповідальність:

- Взаємодія з користувачем через Telegram
- Обробка команд та повідомлень
- Форматування вихідних повідомлень
- Відображення індикаторів завантаження

Ключові функції:

- `cmd_start()` - привітання
- `cmd_help()` - допомога
- `cmd_stats()` - статистика
- `cmd_about()` - інформація
- `handle_message()` - обробка запитів

2. Бізнес-логіка (Business Logic Layer)

Модуль: `bot.py` (клас WeatherBot)

Відповідальність:

- Отримання погодних даних
- Форматування повідомлень
- Управління статистикою
- Координація між компонентами

Ключові методи:

- `get_weather_data(city)` - запит до OpenWeatherMap
- `format_weather_message()` - форматування

3. AI обробка (AI Processing Layer)

Модуль: `mistral_ai.py`

Відповідальність:

- Генерація рекомендацій через Mistral AI
- Обробка та очищення відповідей
- Fallback режим при недоступності API

- Парсинг та структурування рекомендацій

Ключові методи:

- `generate_recommendations()` - генерація
- `_clean_markdown()` - очищення
- `_truncate_text()` - обрізання
- `_fallback_recommendations()` - резерв

4. Інтеграційний рівень (Integration Layer)

API інтеграції:

- OpenWeatherMap API (HTTP GET)
- Mistral AI API (HTTP POST)
- Telegram Bot API (через aiogram)

Обробка:

- HTTP запити через `requests`
- Асинхронна обробка через `aiogram`
- Обробка помилок та таймаутів

Потік даних

```
Користувач → Telegram → bot.py
        │
        ├──→ OpenWeatherMap API → Погодні дані
        │
        └──→ mistral_ai.py → Mistral API → AI рекомендації
                    │
                    └──→ Обробка → Форматування
                       │
                       └──→ Користувач
```

Патерни проектування

Використані патерни:

1. Singleton - глобальний екземпляр WeatherBot
2. Strategy - різні стратегії рекомендацій (AI vs Fallback)

3. Template Method - шаблон обробки запиту

4. Adapter - адаптація API відповідей до формату Telegram

План розробки

Етап 1: Планування та проєктування

Завдання:

- Визначення вимог та функціоналу
- Вибір технологій та API
- Проектування архітектури
- Створення структури проекту

Результат:

- Технічне завдання
- Архітектурна схема
- Структура файлів

Етап 2: Базовий функціонал

Завдання:

- Налаштування Telegram бота
- Інтеграція OpenWeatherMap API
- Базові команди (/start, /help)
- Обробка текстових повідомлень
- Відображення погодних даних

Результат:

- Працюючий бот з базовим функціоналом
- Отримання та відображення погоди

Етап 3: AI інтеграція

Завдання:

- Налаштування Mistral AI API
- Створення модуля mistral_ai.py
- Генерація промптів для AI
- Обробка відповідей Mistral
- Інтеграція з основним ботом

Результат:
- AI рекомендації працюють
- Інтеграція з Mistral API

Етап 4: Оптимізація та виправлення

Завдання:
- Обробка помилок
- Fallback режими
- Оптимізація довжини повідомлень
- Видалення Markdown форматування
- Покращення форматування

Результат:
- Стабільна робота
- Обробка всіх помилок
- Оптимізовані повідомлення

Етап 5: Документація та тестування

Завдання:
- Написання документації
- Створення README
- Тестування функціоналу
- Виправлення виявлених помилок

Результат:

- Повна документація
- Протестований бот
- Готовий до використання

<u>Майбутні етапи (Можливі розширення)</u>

Етап 6: Розширення функціоналу

- База даних для статистики
- Історія запитів
- Улюблені міста
- Прогнози погоди

Етап 7: Оптимізація

- Кешування результатів
- Моніторинг та аналітика
- Покращення продуктивності

Процес розробки

Підхід: Ітеративна розробка з поступовим додаванням функцій

Етапи розробки:

1. Аналіз вимог
 - Визначення цілей проекту
 - Аналіз цільової аудиторії
 - Вибір технологій
2. Проектування
 - Архітектура системи
 - Структура коду
 - API інтеграції

3. Реалізація
 - Написання коду
 - Інтеграція компонентів
 - Базове тестування

4. Тестування
 - Функціональне тестування
 - Тестування помилок
 - Тестування продуктивності

5. Оптимізація
 - Виправлення помилок
 - Покращення продуктивності
 - Рефакторинг коду

6. Документація
 - Написання документації
 - Створення прикладів
 - Інструкції з використання

Інструменти розробки

Середовище розробки:
- Python 3.8+
- Віртуальне середовище (venv)
- Редактор коду (VS Code / PyCharm)

Бібліотеки:
- `aiogram` - Telegram бот фреймворк
- `requests` - HTTP клієнт
- `logging` - логування

API сервіси:

- Telegram Bot API
- OpenWeatherMap API
- Mistral AI API

Версійність

Версія 1.0.0 (Початкова)

- Базовий функціонал
- Отримання погоди
- Прості рекомендації

Версія 2.0.0 (Поточна)

- Інтеграція Mistral AI
- Розумні рекомендації
- Покращена обробка помилок
- Оптимізація повідомлень

Версія 3.0.0 (Планується)

- База даних
- Кешування
- Розширена статистика
- Додаткові функції

Процес тестування

Типи тестування:

1. Мануальне тестування

- Тестування всіх команд
- Тестування з різними містами
- Тестування обробки помилок

2. Інтеграційне тестування
- Тестування API інтеграцій
- Тестування потоку даних
- Тестування взаємодії компонентів

3. Тестування продуктивності
- Швидкість відповіді
- Обробка навантаження
- Використання ресурсів

Стандарти коду:
- PEP 8 стиль кодування
- Документовані функції
- Зрозумілі назви змінних
- Модульна структура

Метрики успіху

Технічні метрики

- Швидкість відповіді: < 5 секунд
- Успішність запитів: > 95%
- Обробка помилок: 100% випадків
- Доступність: 24/7

Функціональні метрики

- Підтримка всіх міст світу
- Генерація AI рекомендацій
- Підтримка української мови

- Зручний інтерфейс

Користувацькі метрики

- Кількість активних користувачів
- Кількість запитів на день
- Середня кількість міст на користувача
- Рівень задоволеності

Навчальна цінність

Отримані навички:

- Python програмування (ООП, асинхронність)
- Робота з API (HTTP запити, обробка відповідей)
- Розробка Telegram ботів (aiogram)
- Інтеграція AI сервісів (Mistral API)
- Обробка помилок та винятків
- Логування та діагностика

AI/ML концепції

Вивчені концепції:

- Мовні моделі та їх застосування
- Генерація тексту через API
- Промпт інжиніринг
- Контекстна обробка даних
- Рекомендаційні системи

Інженерні практики

Застосовані практики:

- Модульна архітектура
- Розділення відповідальності

- Обробка помилок (graceful degradation)
- Конфігурація через змінні середовища
- Документація коду та проекту

Майбутній розвиток

Короткострокові покращення

- База даних для збереження статистики
- Кешування погодних даних
- Покращення обробки помилок
- Додаткові команди

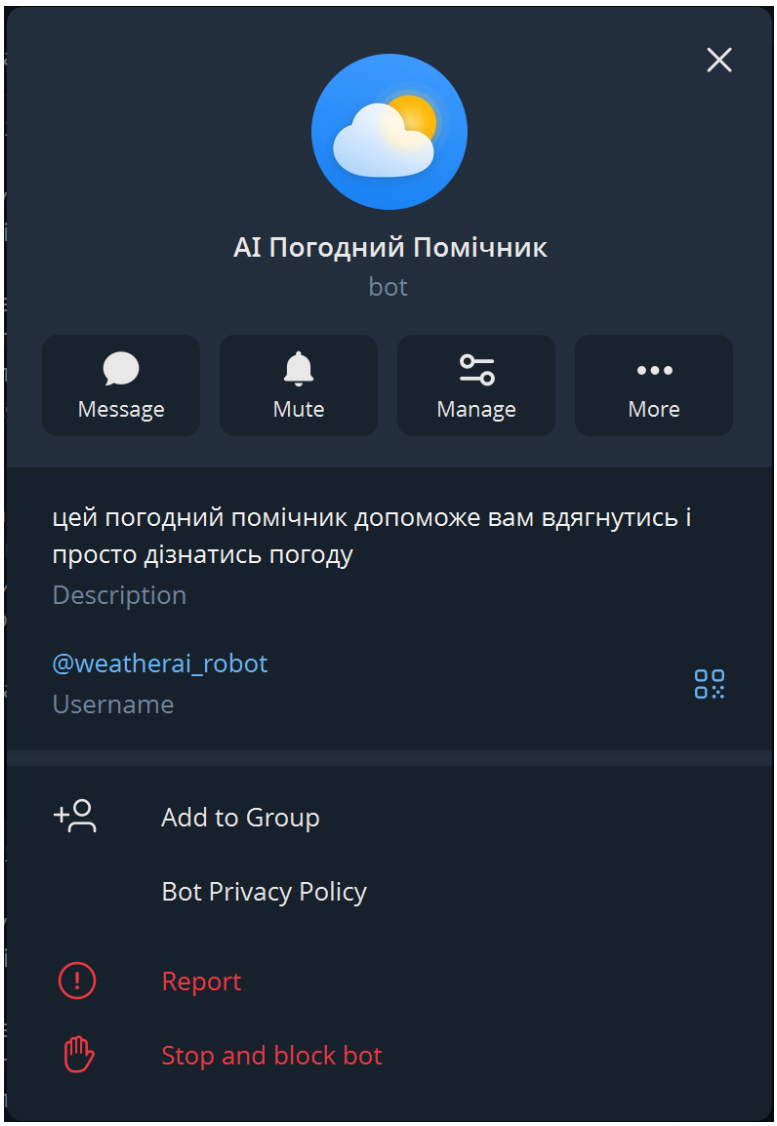
Довгострокові цілі

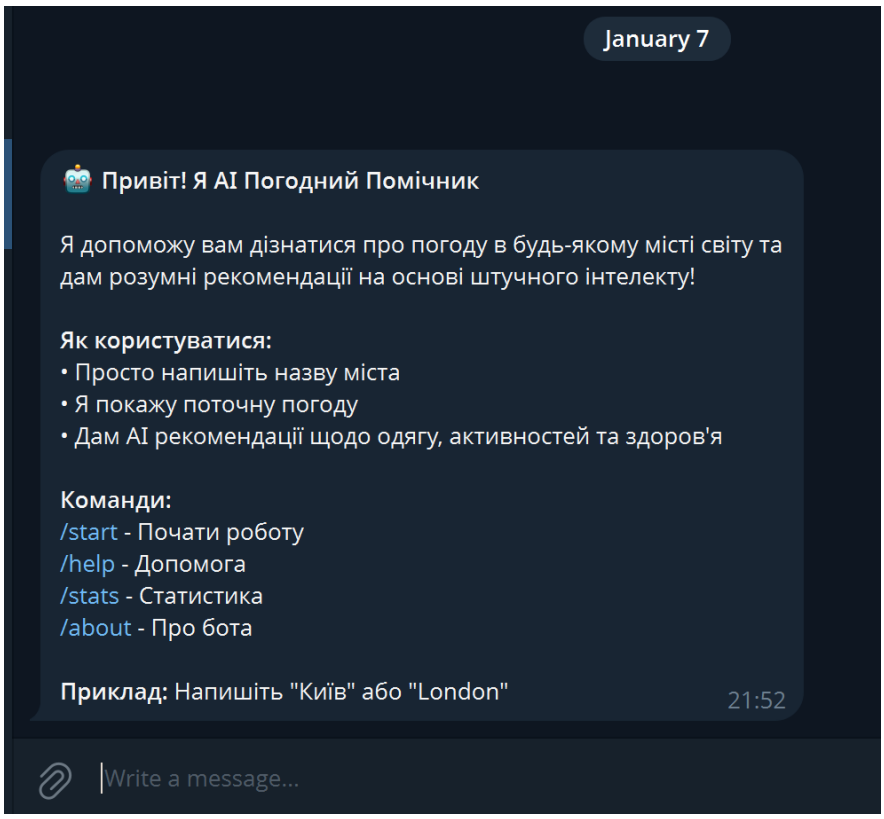
- Мультимовність (підтримка різних мов)
- Прогнози погоди на кілька днів
- Персоналізація на основі історії
- Інтеграція з іншими сервісами
- Webhook замість polling
- Моніторинг та аналітика

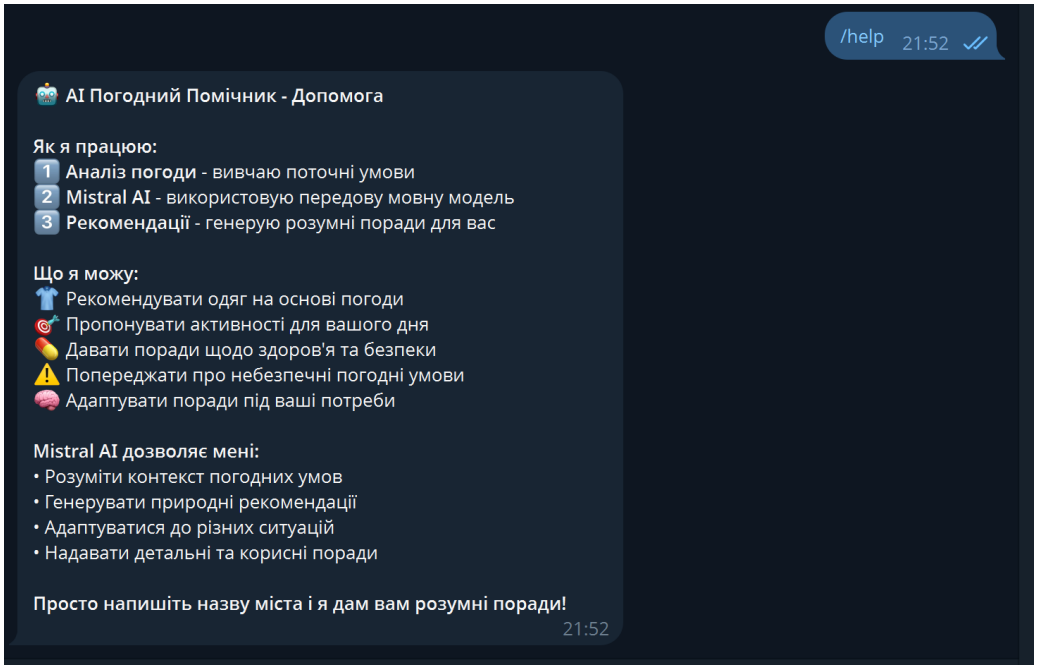
Можливі розширення

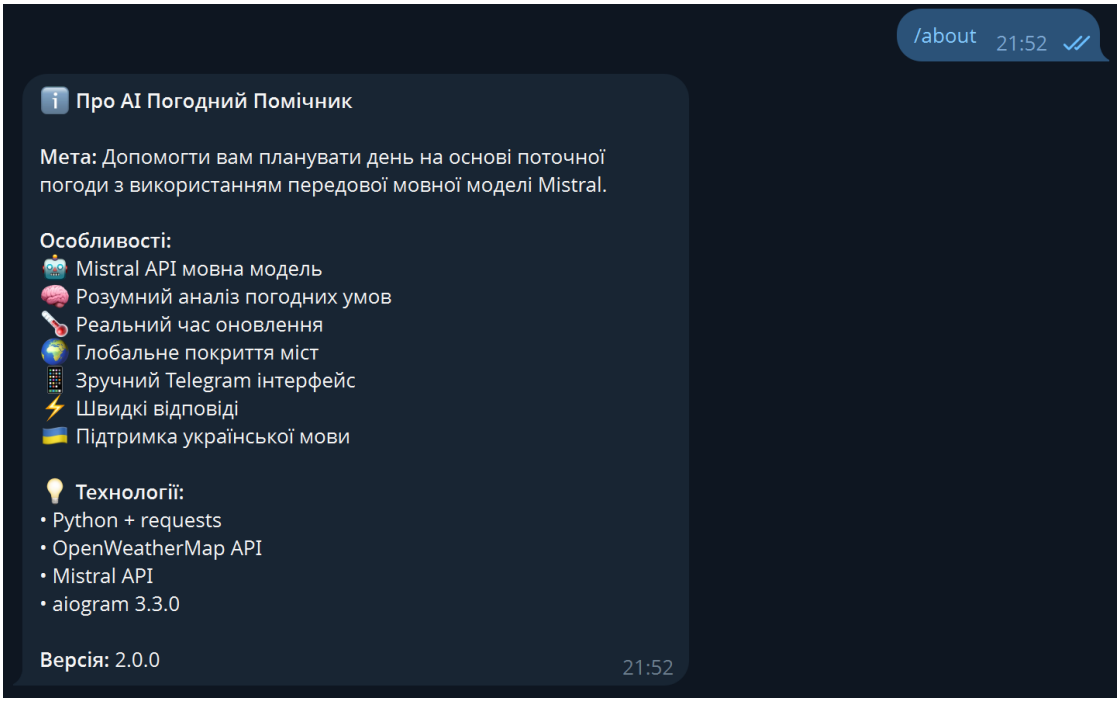
- Веб-інтерфейс
- Мобільний додаток
- Інтеграція з календарем
- Уведомлення про зміни погоди
- Рекомендації для подорожей

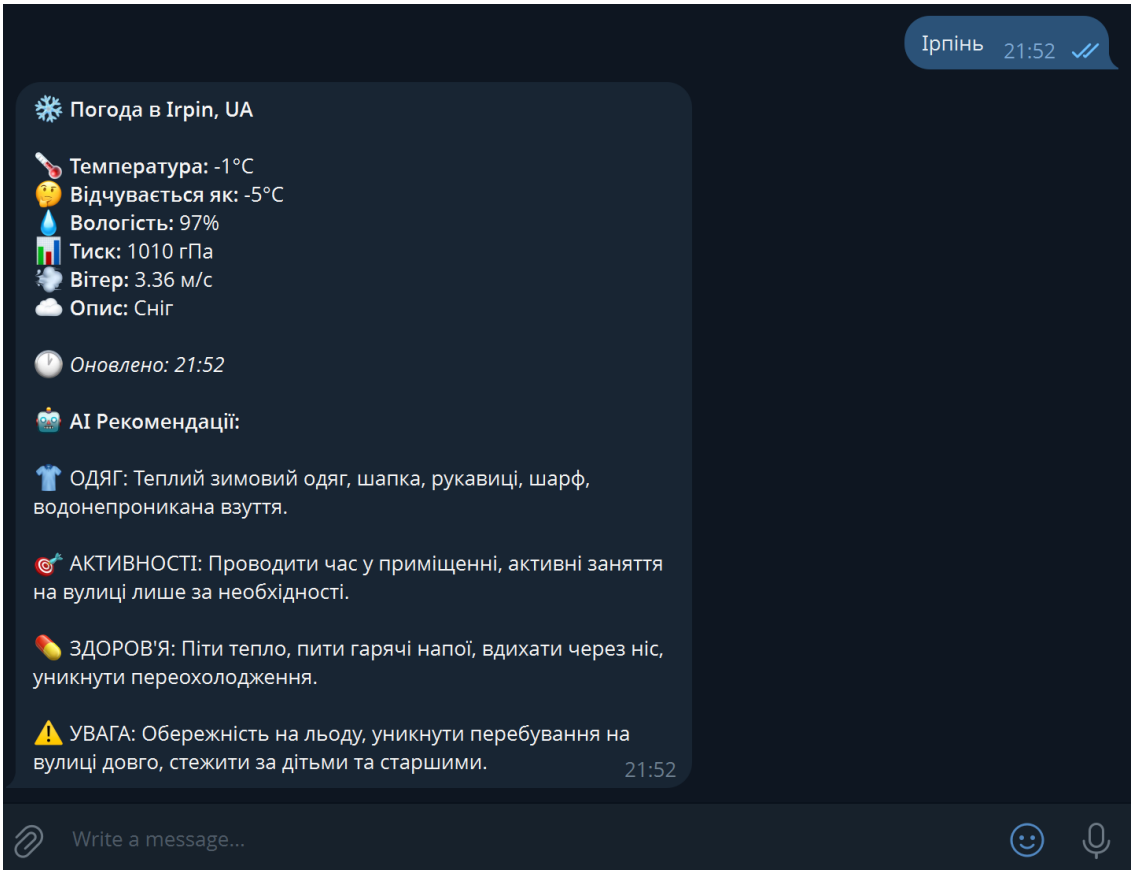
На наступних рисунках наведено скрини екрана розробленого AI погодного помічника.

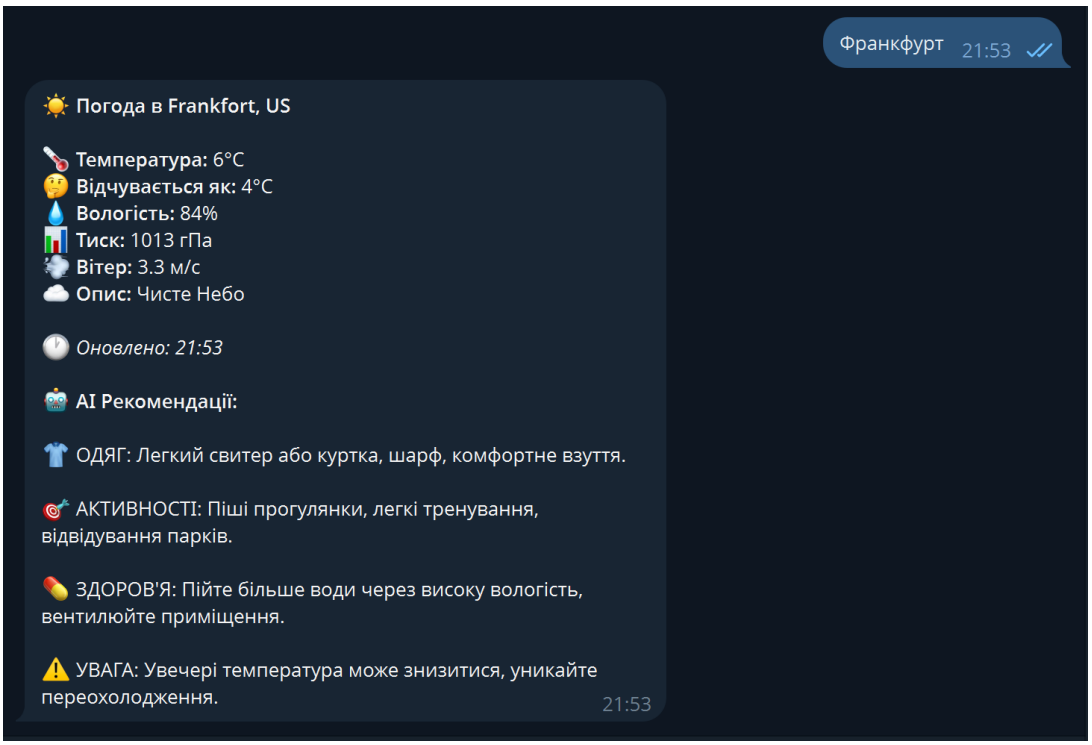

AI Погодний Помічник
bot
Message
Mute
Manage
More
цей погодний помічник допоможе вам вдягнутись і просто дізнатись погоду
Description
@weatherai_robot
Username
Add to Group
Bot Privacy Policy
Report
Stop and block bot

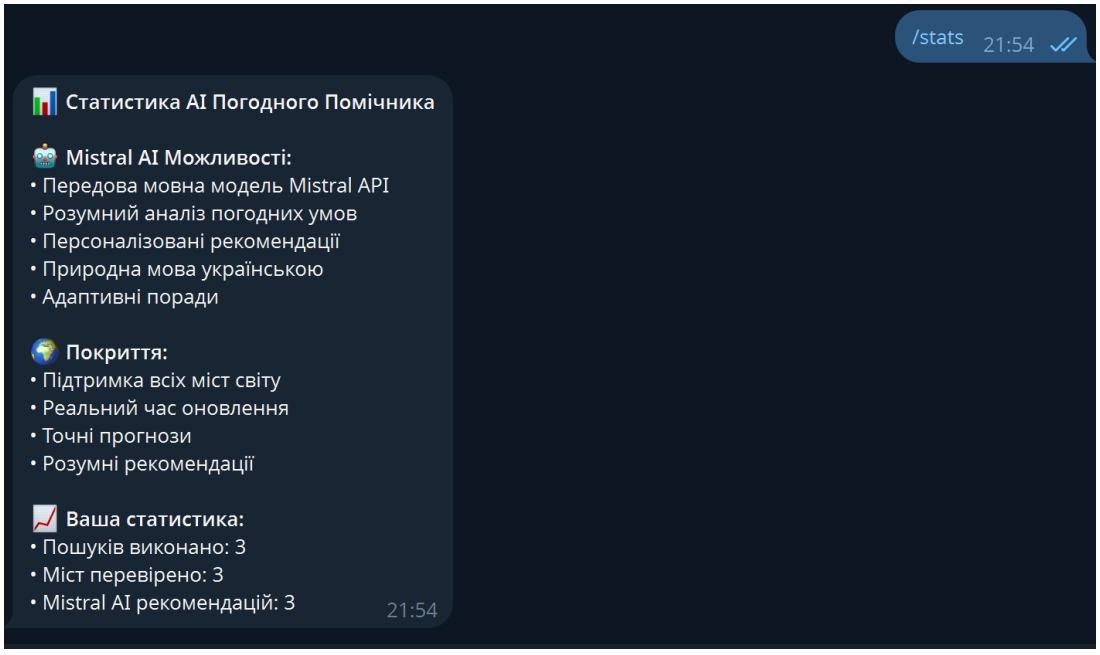

January 7
🤖 Привіт! Я AI Погодний Помічник
Я допоможу вам дізнатися про погоду в будь-якому місті світу та дам розумні рекомендації на основі штучного інтелекту!
Як користуватися:
• Просто напишіть назву міста
• Я покажу поточну погоду
• Дам AI рекомендації щодо одягу, активностей та здоров'я
Команди:
/start - Почати роботу
/help - Допомога
/stats - Статистика
/about - Про бота
Приклад: Напишіть "Київ" або "London"
21:52
Write a message...


/help 21:52
🤖 AI Погодний Помічник - Допомога
Як я працюю:
1️⃣ Аналіз погоди - вивчаю поточні умови
2️⃣ Mistral AI - використовую передову мовну модель
3️⃣ Рекомендації - генерую розумні поради для вас
Що я можу:
👕 Рекомендувати одяг на основі погоди
🎯 Пропонувати активності для вашого дня
💊 Давати поради щодо здоров'я та безпеки
⚠️ Попереджати про небезпечні погодні умови
🧠 Адаптувати поради під ваші потреби
Mistral AI дозволяє мені:
• Розуміти контекст погодних умов
• Генерувати природні рекомендації
• Адаптуватися до різних ситуацій
• Надавати детальні та корисні поради
Просто напишіть назву міста і я дам вам розумні поради!
21:52


/about 21:52
ℹ️ Про AI Погодний Помічник
Мета: Допомогти вам планувати день на основі поточної погоди з використанням передової мовної моделі Mistral.
Особливості:
🤖 Mistral API мовна модель
🧠 Розумний аналіз погодних умов
🌡️ Реальний час оновлення
🌍 Глобальне покриття міст
📱 Зручний Telegram інтерфейс
⚡ Швидкі відповіді
🇺🇦 Підтримка української мови
💡 Технології:
• Python + requests
• OpenWeatherMap API
• Mistral API
• aiogram 3.3.0
Версія: 2.0.0
21:52


Ірпінь 21:52
❄️ Погода в Irpin, UA
🌡️ Температура: -1°C
🤔 Відчувається як: -5°C
💧 Вологість: 97%
📊 Тиск: 1010 гПа
💨 Вітер: 3.36 м/с
☁️ Опис: Сніг
🕐 Оновлено: 21:52
🤖 AI Рекомендації:
👕 ОДЯГ: Теплий зимовий одяг, шапка, рукавиці, шарф, водонепроникана взуття.
🎯 АКТИВНОСТІ: Проводити час у приміщенні, активні заняття на вулиці лише за необхідності.
💊 ЗДОРОВ'Я: Піти тепло, пити гарячі напої, вдихати через ніс, уникнути переохолодження.
⚠️ УВАГА: Обережність на льоду, уникнути перебування на вулиці довго, стежити за дітьми та старшими.
21:52
Write a message...


Франкфурт 21:53
☀️ Погода в Frankfort, US
🌡️ Температура: 6°C
🤔 Відчувається як: 4°C
💧 Вологість: 84%
📊 Тиск: 1013 гПа
💨 Вітер: 3.3 м/с
☁️ Опис: Чисте Небо
🕐 Оновлено: 21:53
🤖 AI Рекомендації:
👕 ОДЯГ: Легкий светер або куртка, шарф, комфортне взуття.
🎯 АКТИВНОСТІ: Піші прогулянки, легкі тренування, відвідування парків.
💊 ЗДОРОВ'Я: Пійте більше води через високу вологість, вентилюйте приміщення.
⚠️ УВАГА: Увечері температура може знизитися, уникайте переохолодження.
21:53


/stats 21:54
📊 Статистика AI Погодного Помічника
🤖 Mistral AI Можливості:
• Передова мовна модель Mistral API
• Розумний аналіз погодних умов
• Персоналізовані рекомендації
• Природна мова українською
• Адаптивні поради
🌍 Покриття:
• Підтримка всіх міст світу
• Реальний час оновлення
• Точні прогнози
• Розумні рекомендації
📈 Ваша статистика:
• Пошуків виконано: 3
• Міст перевірено: 3
• Mistral AI рекомендацій: 3
21:54

ВИСНОВКИ

У результаті проведеного теоретичного дослідження було обґрунтовано доцільність та технічну можливість створення метеорологічного чат-бота. Основні результати аналізу зводяться до наступних положень:

1. **Обґрунтування актуальності:** Швидкий доступ до метеоданих через месенджери є критично важливим для широкого спектра галузей — від агросектору та будівництва до логістики й туризму. Інтеграція погодних сервісів у Telegram дозволяє задовольнити попит користувачів на отримання інформації в режимі «тут і зараз».

2. **Формулювання мети та завдань:** Головною метою визначено створення інтуїтивно зрозумілого інтерфейсу для оперативного інформування про стан погоди. Для цього необхідно реалізувати стабільну інтеграцію з метеорологічними API, забезпечити персоналізацію даних (збереження локації користувача) та впровадити алгоритми обробки помилок для гарантування високої відмовостійкості системи.

3. **Вибір платформи розробки:** Telegram ідентифіковано як оптимальне середовище завдяки наявності потужного Telegram Bot API. Цей інструментарій надає розробнику гнучкі можливості для керування командами, надсилання медіаданих та взаємодії з користувачем, що значно спрощує процес програмної реалізації.

4. **Аналіз ринкових рішень:** Огляд існуючих аналогів (Weatherbot, Weatherman тощо) підтвердив високу затребуваність подібних рішень, водночас виявивши потенціал для вдосконалення функціонала, зокрема в частині надійності роботи та зручності кастомізації.

Підсумовуючи, проведений аналіз сфер застосування, функціональних вимог та технологічних можливостей платформи створює надійний теоретичний фундамент. Це дозволяє перейти до наступного етапу роботи — вибору технологічного стека та безпосереднього проектування й реалізації архітектури чат-бота.

Розроблений AI погодний помічник - це успішний приклад інтеграції сучасних AI технологій з практичними застосуваннями. проєкт демонструє:

Технічну компетентність - використання сучасних технологій

Практичну цінність - реальне застосування для користувачів

Якість коду - чистий, документований, модульний код

Надійність - обробка помилок та fallback режими

Масштабованість - готовність до розширення

Проєкт готовий до використання та може слугувати основою для подальшого розвитку та розширення функціоналу.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1.What Is Chatbot: DefinitionandGuidance [Електронний ресурс] – Режим доступу до ресурсу: https://sendpulse.com/support/glossary/chatbot

2. Telegram [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Telegram_(software)

3. TelegramBot. Basic functionality. Flies separately, cutlets separately. (Part 2) [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/481354/

4. Telegram Bot API [Електронний ресурс] – Режим доступу до ресурсу: https://core.telegram.org/bots/api/

5. PyCharm [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/PyCharm

6. Instruction how to work with BotFather bot [Електронний ресурс] – Режим доступу до ресурсу: https://botcreators.ru/blog/botfather- instrukciya/

7. Python [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Python_(programming_language)

8. Welcome to aiogram’s documentation! [Електронний ресурс] – Режим доступу до ресурсу: https://docs.aiogram.dev/en/latest/

9. Welcome to GeoPy’s documentation! [Електронний ресурс] – Режим доступу до ресурсу: https://geopy.readthedocs.io/en/stable/

10. A Quick Guide to the Python Requests Library [Електронний ресурс] – Режим доступу до ресурсу: https://pythonru.com/biblioteki/kratkoe- rukovodstvo-po-biblioteke- python-requests

11. PyCharm Vs Visual Studio Code For Python Development [Електронний ресурс]–Режим доступу до ресурсу: https://www.c-sharpcorner.com/article/pycharm-vs-visual-studio-code-for-python- development/

12. Pycharm vs Visual Studio Code [for Python Development] [Електронний ресурс] – Режим доступу до ресурсу: https://www.tabnine.com/blog/pycharm- vs-visual-studio-code/

13.Comparing Python to Other Languages [Електронний ресурс] – Режим доступу до ресурсу: https://www.python.org/doc/essays/comparisons/

14. Python vs Java: What's The Difference? [Електронний ресурс] – Режим доступу до ресурсу: https://www.bmc.com/blogs/python-vs-java/

15. Bot API Reference [Електронний ресурс] – Resource access mode: https://tlgrm.ru/docs/bots/api

16. API OpenWeatherMap [Електронний ресурс] – Режим доступу до ресурсу: https://openweathermap.org/api.

17. Types of UML Diagrams [Електронний ресурс] – Режим доступу до ресурсу: https://www.lucidchart.com/blog/types-of-UML-diagrams.

18. Alvarado, R., & Smith, J. (2021). Weather data integration in smart city applications: Opportunities and challenges. Journal of Urban Informatics, 5(2), 112–128. https://doi.org/10.1016/j.jui.2021.02.004

19. Anderson, K., & Maxwell, J. (2019). The role of APIs in web-based information systems: A review. Information Systems Frontiers, 21(6), 1357–1373. https://doi.org/10.1007/s10796-019-09901-3

20. Bäumer, J., Künzer, L., & Schultz, J. (2020). Open data for weather applications: Case study of OpenWeather API. Environmental Modelling & Software, 134, 104814. https://doi.org/10.1016/j.envsoft.2020.104814

21. Chowdhury, A., Rahman, M., & Saha, R. (2022). Application of weather APIs in smart agriculture: Enhancing crop yield prediction. Computers and Electronics in Agriculture, 198, 107083. https://doi.org/10.1016/j.compag.2022.107083

22. Datt, R., & Patel, P. (2021). Development of real-time web applications using RESTful APIs: An empirical study. International Journal of Computer Applications, 183(46), 1–7. https://doi.org/10.5120/ijca2021921683

23. Gohil, H., & Vala, K. (2021). Designing weather dashboards using OpenWeather API and JavaScript frameworks. International Research Journal of Engineering and Technology, 8(5), 1264–1270.

24. Gupta, V., & Singh, R. (2020). The significance of APIs in modern software ecosystems. Journal of Software Engineering Research and Development,

8(1), 1–12. https://doi.org/10.1186/s40411-020-00075-3

25. Hansen, J., Rueda, C., & Romero, A. (2019). Real-time weather information systems: A comparative analysis of mobile and web-based platforms. International Journal of Information Management, 47, 98–106. https://doi.org/10.1016/j.ijinfomgt.2019.01.009

26. Kumar, A., & Kaur, S. (2023). User satisfaction analysis of weather forecasting applications. International Journal of Human–Computer Interaction, 39(5), 423–436. https://doi.org/10.1080/10447318.2022.2109831

27. Li, F., & Zhao, Y. (2022). Cloud-based meteorological forecasting: Leveraging APIs for global accessibility. Future Generation Computer Systems, 128, 72–83. https://doi.org/10.1016/j.future.2021.09.004

28. Mishra, R., & Rajan, A. (2021). A case study on the effectiveness of OpenWeather API in developing responsive applications. International Journal of Applied Engineering Research, 16(3), 145–152.

29. Nguyen, T., & Pham, H. (2020). Impact of mobile weather applications on daily decision-making. Journal of Information Technology and Management, 21(4), 341–352. https://doi.org/10.1007/s10799-020-00315-4

30. OpenWeather. (2024). OpenWeather API documentation. OpenWeather Ltd. https://openweathermap.org/api

31. Patel, M., Shah, K., & Desai, R. (2021). Disaster management systems with real-time weather forecasting using APIs. Procedia Computer Science, 192, 2541–2549. https://doi.org/10.1016/j.procs.2021.09.026

32. Ramesh, P., & Sethi, V. (2020). REST API-based data exchange models in environmental monitoring systems. Journal of Environmental Informatics, 35(1), 45–57. https://doi.org/10.3808/jei.202000409

33. Ray, D., & Kar, A. (2022). Mobile-first design in weather forecasting apps: Enhancing accessibility. Information and Management, 59(7), 103672. https://doi.org/10.1016/j.im.2022.103672

34. Roy, A., & Dutta, S. (2021). Statistical evaluation of weather forecast accuracy in developing countries. Meteorological Applications, 28(5),

e2033. https://doi.org/10.1002/met.2033

35. • Singh, P., & Kaur, H. (2023). Tourism and weather dashboards: An empirical investigation. Journal of Tourism Technology, 14(3), 217–231. https://doi.org/10.1108/JTT-02-2022-0009

36. Thomas, L., & George, B. (2020). Responsive web design practices for weather data visualization. International Journal of Web Engineering and Technology, 15(2), 122–138. https://doi.org/10.1504/IJWET.2020.109847

37. Williams, A., & Brown, E. (2022). API-driven applications: Improving performance and scalability in cloud-native systems. IEEE Access, 10, 24567–24580. https://doi.org/10.1109/ACCESS.2022.3151027