

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

**Пояснювальна записка**

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ  
ЗАСОБІВ ДЛЯ ПРОГНОЗУВАННЯ ПОГОДНИХ ХАРАКТЕРИСТИК  
RESEARCH AND SOFTWARE IMPLEMENTATION OF  
WEATHER FORECASTING TOOLS

Виконав(ла): студент(ка) 2 курсу, групи КНТ-114м

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

КУЛЬЧЕНКО Я.Е.

(ПРИЗВИЩЕ та ініціали)

Керівник ЗАЙКО Т.А.

(ПРИЗВИЩЕ та ініціали)

Рецензент КОРОТКИЙ О.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний університет «Запорізька політехніка»

Факультет КНТ  
Кафедра програмних засобів  
Ступінь вищої освіти магістр  
Спеціальність 121 Інженерія програмного забезпечення  
(код і найменування)

Освітня програма (спеціалізація) Інженерія програмного забезпечення  
(назва освітньої програми (спеціалізації))

**ЗАТВЕРДЖУЮ**

Завідувач кафедри ПЗ, д.т.н, проф.  
Сергій СУББОТІН  
“ ” 2025\_року

**З А В Д А Н Н Я**

**НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)**

КУЛЬЧЕНКА Ярослава Едуардовича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація засобів для прогнозування погодних характеристик. Research and Software Implementation of Weather Forecasting Tools

керівник проєкту (роботи) к.т.н., доцент, ЗАЙКО Тетяна Анатоліївна,  
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 30 ” вересня 2025 року № 447

2. Строк подання студентом проєкту (роботи) 02 грудня 2025 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів) \_\_\_\_\_

Слайди презентації

## 6. Консультанти розділів проєкту (роботи)

| Розділ              | ПРИЗВИЩЕ, ініціали та посада консультанта | Підпис, дата   |                           |
|---------------------|-------------------------------------------|----------------|---------------------------|
|                     |                                           | завдання видав | прийняв виконане завдання |
| 1-4 Основна частина | ЗАЙКО Т.А., доцент                        |                |                           |
| Нормоконтроль       | ДЕЙНЕГА Л.Ю., ст. викладач                |                |                           |

7. Дата видачі завдання “ 30 ” вересня 2025 року.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів дипломного проєкту (роботи)                             | Строк виконання етапів проєкту ( роботи ) | Примітка     |
|-------|----------------------------------------------------------------------|-------------------------------------------|--------------|
| 1     | Постановка завдання роботи.                                          | 1 тиждень                                 | Завдання, ТЗ |
| 2     | Аналіз предметної області.                                           | 1 тиждень                                 | Розділ 1     |
| 3     | Вибір мови програмування та інших технологій розробки.               | 2 тиждень                                 | Розділ 2     |
| 4     | Розробка структури програми.                                         | 3 тиждень                                 | Розділ 3     |
| 5     | Розробка програми.                                                   | 4-7 тижні                                 | Розділи 3,4  |
| 6     | Тестування та експериментальне дослідження програмного забезпечення. | 8 тиждень                                 | Розділ 4     |
| 7     | Оформлення пояснювальної записки та документів до неї.               | 9-10 тижні                                | Додатки      |
| 8     | Нормоконтроль та рецензування.                                       | 11 тиждень                                |              |
| 9     | Захист роботи.                                                       | 12 тиждень                                |              |

Студент(ка)

\_\_\_\_\_  
( підпис )      Ярослав КУЛЬЧЕНКО  
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

\_\_\_\_\_  
( підпис )      Тетяна ЗАЙКО  
(Ім'я ПРИЗВИЩЕ)

## РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:  
95 с., 4 табл., 30 рис., 3 дод., 17 джерел.

PYCHARM, PYTHON, МОВА ПРОГРАМУВАННЯ, ПОГОДА,  
ПРОГНОЗУВАННЯ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ХАРАКТЕРИСТИКА.

Об'єкт дослідження – процес розробки програмного забезпечення для прогнозування погоди.

Предмет дослідження – програмні засоби для прогнозування погодних характеристик.

Мета роботи – дослідження та розробка програмного забезпечення для прогнозування погодних характеристик.

Матеріали, методи та технічні засоби: мова програмування Python, середовище розробки PyCharm.

Результати. Розроблено програмне забезпечення для прогнозування погодних характеристик за допомогою мови програмування Python та середовища розробки PyCharm.

Практична цінність роботи полягає у розробці програмного забезпечення для прогнозування погодних характеристик.

Висновки. Виконано проєктування програмного забезпечення для прогнозування погодних характеристик. Розроблено програмне забезпечення для прогнозування погодних характеристик. Здійснено тестування розробленого програмного забезпечення для прогнозування погодних характеристик.

Галузь використання – прогнозування погоди.

## ABSTRACT

Explanatory note to the diploma qualifying work of the master: 95 pages, 4 tables, 30 figures, 3 appendixes, 17 sources.

PYCHARM, PYTHON, PROGRAMMING LANGUAGE, WEATHER, FORECASTING, SOFTWARE, CHARACTERISTICS.

The object of research is the process of developing software for weather forecasting.

The subject of the research is software for forecasting weather characteristics.

The purpose of this work is to develop the software for forecasting weather characteristics.

Materials, methods and technical tools: Python programming language, PyCharm development environment.

Results. The software tools for forecasting weather characteristics using the Python programming language and the PyCharm development environment have been developed.

The practical value of the work lies in the development of software for forecasting weather characteristics.

Conclusions. The software for forecasting weather characteristics has been designed. The software for forecasting weather characteristics has been developed. The developed software for forecasting weather characteristics has been tested.

Scope of use – forecasting weather.

## ЗМІСТ

|                                                                                                                    |    |
|--------------------------------------------------------------------------------------------------------------------|----|
|                                                                                                                    | С. |
| Перелік скорочень та умовних позначок.....                                                                         | 8  |
| Вступ.....                                                                                                         | 9  |
| 1 Аналіз предметної області.....                                                                                   | 11 |
| 1.1 Програмне забезпечення для прогнозування погодних характеристик..                                              | 11 |
| 1.2 Аналіз програмного забезпечення для прогнозування погодних характеристик.....                                  | 18 |
| 1.3 Висновки за розділом 1 .....                                                                                   | 26 |
| 2 Матеріали і методи.....                                                                                          | 28 |
| 2.1 Вибір мови програмування.....                                                                                  | 28 |
| 2.2 Вибір середовища розробки для створення програмного забезпечення для прогнозування погодних характеристик..... | 31 |
| 2.3 Розробка моделі програмного забезпечення для прогнозування погодних характеристик.....                         | 33 |
| 2.4 Висновки за розділом 2 .....                                                                                   | 36 |
| 3 Опис програми.....                                                                                               | 38 |
| 3.1 Структура програмного забезпечення для прогнозування погодних характеристик.....                               | 38 |
| 3.2 Функціонування програмного забезпечення для прогнозування погодних характеристик.....                          | 40 |
| 3.3 Особливості реалізації програмного забезпечення для прогнозування погодних характеристик.....                  | 43 |
| 3.4 Проєктування інтерфейсу програмного забезпечення для прогнозування погодних характеристик.....                 | 46 |
| 3.5 Висновки за розділом 3 .....                                                                                   | 48 |
| 4 Експлуатація, тестування та експериментальне дослідження програми.....                                           | 49 |
| 4.1 Призначення й умови застосування програми.....                                                                 | 49 |
| 4.2 Характеристики програми для прогнозування погодних характеристик.....                                          | 50 |
| 4.3 Інструкція по експлуатації програми.....                                                                       | 51 |

|       |                                                                                   |    |
|-------|-----------------------------------------------------------------------------------|----|
| 4.3.1 | Звернення до програми.....                                                        | 51 |
| 4.3.2 | Вхідні й вихідні дані.....                                                        | 51 |
| 4.3.3 | Повідомлення.....                                                                 | 51 |
| 4.4   | Виконання програмного забезпечення для прогнозування погодних характеристик.....  | 51 |
| 4.5   | Тестування програмного забезпечення для прогнозування погодних характеристик..... | 59 |
| 4.6   | Висновки за розділом 4.....                                                       | 59 |
|       | Висновки.....                                                                     | 60 |
|       | Перелік джерел посилання.....                                                     | 63 |
|       | Додаток А Технічне завдання.....                                                  | 65 |
|       | Додаток Б Фрагмент тексту програми.....                                           | 69 |
|       | Додаток В Слайди презентації.....                                                 | 88 |

## **ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК**

API – Application Programming Interface;

GRU – Gated Recurrent Unit;

LSTM – Long Short Term Memory;

SGD – Stochastic Gradient Descent;

ПД – погодні дані;

ПЗ – програмне забезпечення.

## ВСТУП

Актуальність програмного забезпечення для прогнозування погодних характеристик зумовлюється необхідністю підвищення точності та оперативності отримання метеорологічної інформації в умовах глобальних кліматичних змін. Сучасний розвиток суспільства та економіки значною мірою залежить від достовірних прогнозів погоди, оскільки вони є критично важливими для забезпечення безпеки транспортних перевезень, ефективного функціонування аграрного сектору, раціонального використання енергетичних ресурсів та планування виробничої діяльності в багатьох галузях [1], [2].

Програмні системи прогнозування, які базуються на чисельних моделях атмосфери, великих обсягах метеорологічних даних та алгоритмах машинного навчання, відкривають можливості для підвищення деталізації та достовірності прогнозів. Особливої ваги це набуває в контексті зростання частоти та інтенсивності екстремальних погодних явищ, що вимагають своєчасного попередження та мінімізації можливих соціально-економічних збитків [3], [4].

Таким чином, розроблення та впровадження програмного забезпечення для прогнозування погодних характеристик є актуальним як у науково-методологічному, так і в прикладному вимірі, оскільки воно сприяє зниженню ризиків, оптимізації використання ресурсів та підвищенню рівня адаптивності суспільства до кліматичних викликів [5], [6].

Проте деякі програми для прогнозування погодних характеристик можуть використовувати неефективні методи та моделі, що призводить до низької точності прогнозів. Важливим недоліком є висока залежність якості прогнозів від обсягу й достовірності вхідних даних. Ще однією проблемою є невизначеність у прогнозах, яка часто ускладнює їхнє практичне застосування. Також варто зазначити, що програмне забезпечення, орієнтоване на глобальні моделі, не завжди забезпечує достатню точність для локальних територій. Таким чином, головними недоліками програмних систем прогнозування

погоди виступають обмежена точність та локальна деталізація прогнозів, залежність від якості вхідних даних, висока ресурсомісткість та складність практичного використання результатів у різних сферах діяльності [1]-[6].

Тому актуальною є розробка програмного забезпечення для прогнозування погодних характеристик. У дипломній кваліфікаційній роботі магістра розв'язується актуальне завдання розробки та дослідження програмного забезпечення для прогнозування погодних характеристик.

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для прогнозування погодних характеристик;
- розробити модель програмного забезпечення для прогнозування погодних характеристик;
- здійснити проєктування програмного забезпечення для прогнозування погодних характеристик;
- створити програмне забезпечення для прогнозування погодних характеристик;
- виконати тестування розробленого програмного забезпечення для прогнозування погодних характеристик.

## **1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ**

### **1.1 Програмне забезпечення для прогнозування погодних характеристик**

Прогнозування погоди – це процес наукового передбачення стану атмосфери для певного місця і часу з використанням метеорологічних знань та сучасних технологій. Програмне забезпечення для прогнозування погодних характеристик виступає інструментом, що забезпечує обчислення, обробку та представлення результатів аналізу атмосферних процесів у зручному для користувачів вигляді. Такі програмні системи ґрунтуються на обробці великих масивів даних, що збираються з різних джерел, і дозволяють створювати прогнози з різними часовими та просторовими масштабами [1].

Сам процес прогнозування є складним поєднанням математичних рівнянь та алгоритмів, які розв'язуються за допомогою потужних обчислювальних систем. Визначення погоди для конкретної місцевості здійснюється шляхом моделювання параметрів атмосфери, серед яких температура, тиск, вологість, вітер та кількість опадів. Щоб забезпечити точність, ці параметри розраховуються для численних точок, розташованих у вигляді сітки, що дозволяє відтворити загальну картину погодних процесів. Чим менший інтервал між точками такої сітки, тим точнішим стає прогноз [1].

Важливою складовою є використання глобальних і регіональних моделей. Глобальні системи дають змогу оцінювати атмосферні процеси на великій території, тоді як регіональні зосереджуються на окремих географічних зонах, забезпечуючи вищу деталізацію. Для окремих явищ розроблені спеціалізовані моделі, які враховують характеристики морських хвиль, температуру поверхні океану чи поведінку вітру у висотних шарах атмосфери [1].

Джерелами інформації для таких моделей є дані, отримані з метеостанцій, супутників, радарів, буїв та зондів. Прилади вимірюють основні показники, серед яких температура, швидкість і напрямок вітру, кількість

опадів, а також більш складні характеристики. Зібрані дані надходять у реальному часі та стають основою для математичного аналізу, який виконується суперкомп'ютерами [1].

Прогноз погоди завжди орієнтований на майбутній час. Для короткострокових прогнозів використовується висока роздільна здатність моделей, що забезпечує більшу точність. Довгострокові прогнози менш деталізовані, однак дозволяють оцінити загальні тенденції. У практиці також застосовується короткочасне прогнозування на найближчі години, яке особливо ефективне для виявлення швидкоплинних атмосферних явищ, таких як раптові зливи чи грози [1].

Вимірювання погодних параметрів здійснюється у спеціальних одиницях, що належать до міжнародної системи, що забезпечує єдність у передачі та інтерпретації результатів. Частота оновлення даних залежить від обраної моделі та можливостей технічних засобів, проте чим частіше відбувається оновлення, тим оперативніше користувач отримує актуальну інформацію [1].

Таким чином, програмне забезпечення для прогнозування погодних характеристик є комплексною системою, що поєднує збір даних, їх аналіз, математичне моделювання та відображення результатів. Воно забезпечує точне та своєчасне відображення стану атмосфери і є невід'ємною частиною сучасного наукового та практичного підходу до метеорології [1].

Прогноз погоди завжди пов'язаний із певним рівнем невизначеності. Атмосфера є складною та мінливою системою, у якій дрібні похибки вимірювань неминуче впливають на результат. Інформація, що надходить від наземних станцій, супутників чи інших приладів, містить неточності, тому математичні моделі не здатні дати абсолютно правильну картину. Найбільш достовірними вважаються короткі прогнози, тоді як із плином часу їхня надійність суттєво зменшується, і на віддалене майбутнє передбачення майже завжди мають умовний характер [2], [3].

Щоб зменшити похибку, застосовуються ансамблеві методи. Суть такого підходу полягає у створенні не одного варіанта, а кількох альтернативних сценаріїв розвитку атмосфери. Якщо результати цих сценаріїв близькі між собою, можна говорити про більшу впевненість у передбаченні. Якщо ж розбіжності значні, довіра до такого прогнозу знижується [2], [3].

Прогнози можуть бути як загальними, що описують основні атмосферні параметри, так і спеціалізованими, розробленими для конкретних завдань. Наприклад, існують окремі передбачення для авіації, морської навігації, фермерського господарства чи навіть для спортивних занять на відкритому повітрі. Такі прогнози враховують детальніший набір показників і часто готуються окремими організаціями або приватними компаніями [2], [3].

Щоб зробити інформацію зрозумілою для широкого кола користувачів, результати обчислень подаються у доступній формі. Для цього застосовуються символи, карти, таблиці, графіки чи інтерактивні елементи в цифрових додатках. Сучасні сервіси дозволяють спостерігати за змінами погоди протягом доби в реальному часі, від руху хмар до початку опадів [2], [3].

Отримати прогноз можна різними шляхами: через мобільний застосунок, на екрані комп'ютера чи навіть на розумному годиннику. Проте для коректної роботи таких сервісів потрібне стабільне з'єднання з мережею. Окреме значення мають повідомлення про зміну погоди. Такі сповіщення містять попередження про небезпечні явища чи суттєві коливання умов, щоб користувач встиг відреагувати. Водночас попередження можуть бути налаштовані й для бажаних погодних подій, наприклад, для отримання інформації про сильний вітер чи опади, що цікавлять спортсменів [2], [3].

Через складність атмосфери різні моделі здатні давати відмінні результати. Тому важливим є не лише використання одного джерела, а й зіставлення кількох прогнозів. Порівняння допомагає зрозуміти загальну тенденцію та оцінити, який варіант буде найімовірнішим у конкретних умовах [2], [3].

Великим проривом у розвитку науки стало суттєве зростання точності прогнозів погоди, яке й надалі відбувається завдяки удосконаленню методів числового моделювання та збільшенню обсягів спостережень за атмосферою і пов'язаними з нею середовищами — океаном, поверхнею ґрунту, рослинним покривом чи крижаними масивами. Важливу роль у цьому відіграли супутникові вимірювання, а також прогрес у сфері обчислювальної техніки, що забезпечив можливість швидкого опрацювання величезних масивів даних. Кожна держава має спеціальну службу, яка займається моніторингом атмосфери та підготовкою прогнозів для урядових структур, підприємств і населення [3], [4].

Робота таких центрів охоплює кілька ключових етапів. Насамперед за допомогою спеціалізованого програмного забезпечення здійснюється збір даних, які надходять із наземних станцій, морських буїв, радіозондів, радарів, супутників, а також від приладів на літаках і кораблях. Усе це формує глобальну систему спостережень, фінансування якої розподіляється між країнами. Особливістю метеорології є відкритість обміну даними, яка зберігається навіть у часи політичних конфліктів [3], [4].

Після збору інформації відбувається її перевірка. Необхідно відсіяти хибні чи підозрілі вимірювання, що можуть виникати через технічні збої або інші похибки. Для цього застосовуються порівняння з сусідніми спостереженнями та з короткостроковими прогнозами. Лише після цього формується так званий початковий стан атмосфери, який максимально відповідає реальності. Це завдання складне, адже дані завжди мають похибки, не збираються одночасно та не покривають усі регіони світу [3], [4].

Далі запускається сам процес прогнозування за допомогою відповідного програмного забезпечення. Числові моделі розв'язують фізичні рівняння та відтворюють послідовні зміни атмосфери. Використовуються як глобальні схеми, що охоплюють увесь світ і дають картину на кілька діб наперед, так і регіональні моделі, які дозволяють отримати деталізованіші результати для окремих територій [4], [5].

Оскільки сучасні прогнози досягли високої точності, будь-які вдосконалення вимагають тривалого випробування, щоб переконатися у відсутності погіршень. При цьому необхідно враховувати й саму природу атмосфери, яка має різні рівні передбачуваності залежно від сезону та конкретних погодних умов. Деякі процеси легше піддаються прогнозуванню, тоді як інші залишаються значно більш мінливими та складними для точного опису [4], [5].

Точність прогнозів погоди суттєво змінюється залежно від параметра, що враховується, та тривалості випередження. Короткострокові прогнози температури зазвичай відзначаються мінімальними відхиленнями, а для вітру похибка часто залишається в межах прийнятного рівня, за винятком зон із потужними атмосферними збуреннями. Натомість кількість опадів, особливо грозових, виявляється набагато важче визначити з високою точністю, оскільки навіть невелике відхилення у вихідних даних здатне суттєво вплинути на кінцевий результат. Визначення точного місця виникнення шторму чи інтенсивності граду залишається складним навіть за кілька годин до події. Подібні труднощі виникають під час прогнозування снігопадів, коли температура ґрунту перебуває близько до нуля: незначне відхилення в показниках може змінити сам характер опадів. Туман також становить серйозну проблему для метеорологів, адже його поява безпосередньо залежить від вологості, яка надзвичайно мінлива. У випадку торнадо здебільшого вдається говорити лише про ймовірність їх формування, а не про точний прогноз [4], [5].

Своєчасність та якість прогнозів безпосередньо залежать від потужності обчислювальних програмних систем. Швидкість виконання операцій визначає, наскільки оперативно можна створити прогноз і передати його користувачам. Постійно постає питання оптимального використання доступних ресурсів, оскільки потрібно знайти баланс між двома вимогами: застосуванням найновіших спостережень, що відтерміновує початок розрахунків, та використанням складних моделей, які забезпечують точність,

але потребують більше часу на обчислення. Ефективність системи залежить від того, наскільки вдало організовано цей компроміс. На результат впливають кількість врахованих вимірювань, рівень складності алгоритмів обробки даних, роздільна здатність моделі та детальність рівнянь, що описують атмосферні процеси. Важливу роль відіграє й розмір ансамблів прогнозів: збільшення їх кількості може надати точніший результат, але водночас потребує більше ресурсів. Також необхідно зважати на розподіл обчислювальної потужності між атмосферними моделями та тими, що описують вплив погоди, аби уникнути втрати ефективності [4], [5].

Подальший розвиток програмних систем прогнозування залежить від роботи дослідницьких підрозділів національних метеорологічних служб, які активно співпрацюють між собою та з науковими установами. Залишається низка невирішених питань, пов'язаних з передбачуваністю атмосферних процесів, що робить цю сферу надзвичайно важливою для суспільства і виправдовує значні державні інвестиції. Розширення систем спостереження стимулює появу нових технологій, що мають великий потенціал для промисловості, особливо у космічній галузі та при створенні сучасних наземних приладів, таких як радари чи лідари [5], [6].

Прогнозування атмосферних процесів стало невід'ємною складовою сучасного суспільства. Його значення проявляється у можливості планувати сільськогосподарську діяльність, забезпечувати безпеку транспортних систем чи своєчасно реагувати на природні катастрофи. Надійність прогнозів зросла завдяки технічному прогресу, який забезпечив суттєве підвищення точності та швидкості аналізу даних. Сучасна метеорологія спирається на складні методи збору й обробки інформації про стан атмосфери, що дало змогу сформувати цілісне уявлення про закономірності її розвитку [5], [6].

Формування прогнозів починається з отримання відомостей про атмосферу з різноманітних джерел. Дані надходять із супутників, які передають інформацію про хмарність, температуру поверхні океанів та розвиток небезпечних явищ. Використовуються також радари, що фіксують

характер і переміщення опадів, а наземні станції вимірюють тиск, вологість та швидкість вітру. Доповнюють цю картину зонди, що піднімаються на значну висоту та визначають вертикальні профілі атмосфери, а також спостереження з літаків і морських суден, які заповнюють прогалини у віддалених регіонах. Зібраний масив інформації обробляється за допомогою спеціалізованого програмного забезпечення та математичних моделей, що відтворюють динаміку атмосфери. Розрахунки виконуються суперкомп'ютерами, здатними здійснювати величезну кількість операцій за короткий проміжок часу. Моделі глобального масштабу дозволяють визначати розвиток погодних процесів на декілька днів уперед, тоді як регіональні версії враховують локальні особливості. Для зменшення рівня невизначеності застосовуються ансамблеві методи, коли кілька симуляцій запускаються одночасно, щоб отримати більш надійний результат [5], [6].

Результати аналізу за допомогою спеціалізованого програмного забезпечення трансформуються у прогнози, які поширюються через різноманітні платформи — від карт і графіків до мобільних додатків. Успіхи в галузі пов'язані з появою високопродуктивних обчислювальних систем, розвитком штучного інтелекту та використанням супутників нового покоління. Суттєвою новацією стало також залучення інформації від особистих пристроїв та локальних метеостанцій, що розширює обсяг доступних даних. Водночас поєднання короткострокових прогнозів з довгостроковими кліматичними моделями допомагає враховувати вплив глобальних змін на локальні погодні процеси [5], [6].

Покращення методів прогнозування має практичне значення для різних сфер життя. Своєчасні повідомлення дають змогу підготуватися до небезпечних природних явищ, зменшуючи ризики для населення й економіки. Сільське господарство отримує можливість оптимізувати виробничі цикли, транспортні системи працюють більш безпечно, а галузь відновлюваної енергетики ефективніше планує свої ресурси. Зростання довіри до прогнозів зміцнило їхній статус як важливого інструмента прийняття рішень [5], [6].

Подальший розвиток цієї галузі пов'язується з використанням новітніх програмних засобів та обчислювальних технологій, включаючи квантові системи, які здатні здійснювати аналіз атмосферних даних з небаченою швидкістю. Інтеграція глобальних мереж і поширення розподілених обчислень обіцяє ще більш оперативне й точне відображення атмосферних процесів у майбутньому. Таким чином, поступове вдосконалення прогнозування сприяє підвищенню стійкості суспільства перед природними викликами та формує основу для більш передбачуваного майбутнього [5], [6].

Визначено, що програмне забезпечення для прогнозування погодних характеристик являє собою інструмент, що забезпечує обчислення, обробку та представлення результатів аналізу атмосферних процесів у зручному для користувачів вигляді. Такі програмні системи ґрунтуються на обробці великих масивів даних, що збираються з різних джерел, і дозволяють створювати прогнози з різними часовими та просторовими масштабами.

## **1.2 Аналіз програмного забезпечення для прогнозування погодних характеристик**

Проаналізуємо програмне забезпечення для прогнозування погодних характеристик [7]-[13].

Програмне забезпечення WeatherBug (рис. 1.1) позиціонується як зручна програма, орієнтована на користувачів, яким потрібна деталізована інформація про погодні умови. Функціональність програмної системи WeatherBug дозволяє отримувати прогнози для різних сценаріїв, зокрема для ситуацій, що потребують штормових попереджень, а також для відстеження факторів навколишнього середовища, наприклад рівня пилку. Завдяки розвиненій метеорологічній мережі забезпечується доступ до актуальних даних у реальному часі, що включають поточні показники, погодинні прогнози та більш тривалі сценарії розвитку погодних умов [7], [11].

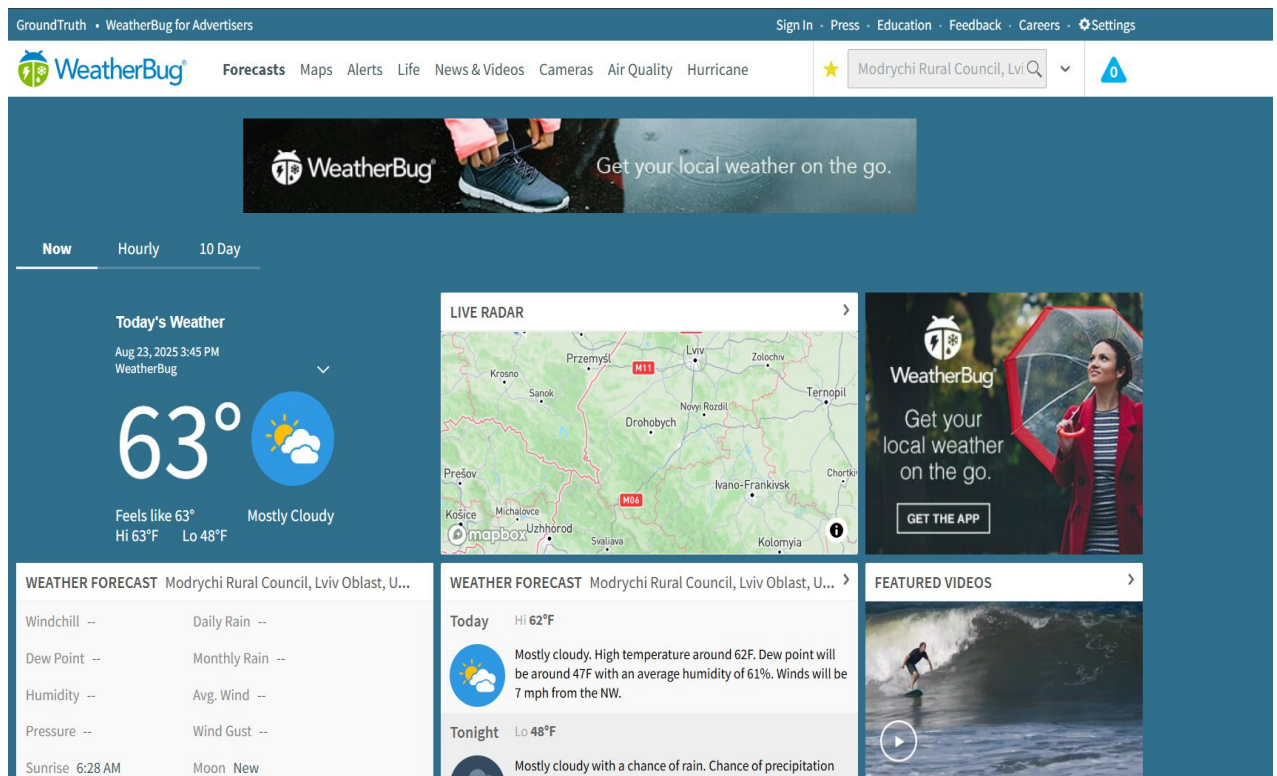


Рисунок 1.1 – Програмне забезпечення WeatherBug [11]

У програмному забезпеченні WeatherBug передбачено використання значної кількості інтерактивних карт, що відображають широкий спектр параметрів атмосфери. Серед них можна виділити карти, які демонструють зміну температури, силу і напрямок вітру, атмосферний тиск, рівень вологості, розподіл опадів і блискавок, а також дані з доплерівського радара. Така багатокomпонентна структура дозволяє формувати комплексне уявлення про погодні ситуацію, надаючи користувачам як практичні, так і науково значущі інструменти для аналізу [7], [11].

У результаті WeatherBug виступає не лише як інформаційний сервіс, а й як інтегроване середовище моніторингу, що поєднує простоту взаємодії та широкий набір інструментів для точного прогнозування та візуалізації погодних характеристик [7], [11].

Основними характеристиками програмного забезпечення WeatherBug є такі [7], [11]:

- інтерфейс: програмне забезпечення WeatherBug є простим у використанні, забезпечується зручна навігація та інтуїтивне відображення прогнозів [7], [11];

- прогнози погоди: програмне забезпечення WeatherBug надає детальні щогодинні та щоденні прогнози для різних регіонів [7], [11];

- радарні карти: програмне забезпечення WeatherBug надає інтерактивні карти з відображенням опадів, температури, напрямку вітру та штормових систем [7], [11];

- сповіщення: програмне забезпечення WeatherBug надсилає попередження про несприятливі погодні умови у реальному часі [7], [11];

- додаткові дані: програмне забезпечення WeatherBug включає інформацію про якість повітря, рівень пилку та екологічні показники [7], [11];

- підтримка платформ: програмне забезпечення WeatherBug доступне як мобільний додаток для iOS та Android, а також у вебверсії [7], [11];

- точність: забезпечує високу точність прогнозів завдяки використанню даних із різних метеорологічних джерел [7], [11];

- персоналізація: програмне забезпечення WeatherBug дозволяє користувачу налаштовувати локації та види даних, які користувач бажає відстежувати [7], [11];

- інтеграція: сумісність з іншими сервісами та можливість синхронізації з розумними пристроями [7], [11];

- безкоштовна та платна версії: основні функції програмного забезпечення WeatherBug доступні безкоштовно, додаткові можливості відкриваються у преміальній версії [7], [11].

Серед недоліків програмного забезпечення WeatherBug можна відзначити надмірну кількість рекламних повідомлень, що можуть заважати користувачам під час роботи з додатком. Також програма іноді демонструє нестабільність у роботі, зокрема повільне завантаження даних чи зависання на окремих пристроях. Додатковим мінусом є обмежений функціонал у безкоштовній версії, який змушує користувачів переходити на платну

підписку для отримання повного доступу. Окремі користувачі відзначають і неточність прогнозів у певних регіонах, що може знижувати довіру до результатів [7], [11].

Програмне забезпечення Ambient Weather Network (рис. 1.2) являє собою сучасне рішення для моніторингу метеорологічних показників, що може використовуватися як у побутових цілях, так і в професійній діяльності. Програмне забезпечення Ambient Weather Network забезпечує можливість отримання детальних прогнозів, своєчасних попереджень та постійного спостереження за погодними умовами в реальному часі. Його особливістю є доступ до масштабної мережі домашніх метеостанцій, розташованих як у близькому оточенні користувача, так і у віддалених куточках світу. Така інтеграція створює умови для формування більш точних і локалізованих прогнозів, що робить програмне забезпечення Ambient Weather Network універсальним інструментом для тих, хто потребує якісної та оперативної інформації про зміни погоди. [7], [12].

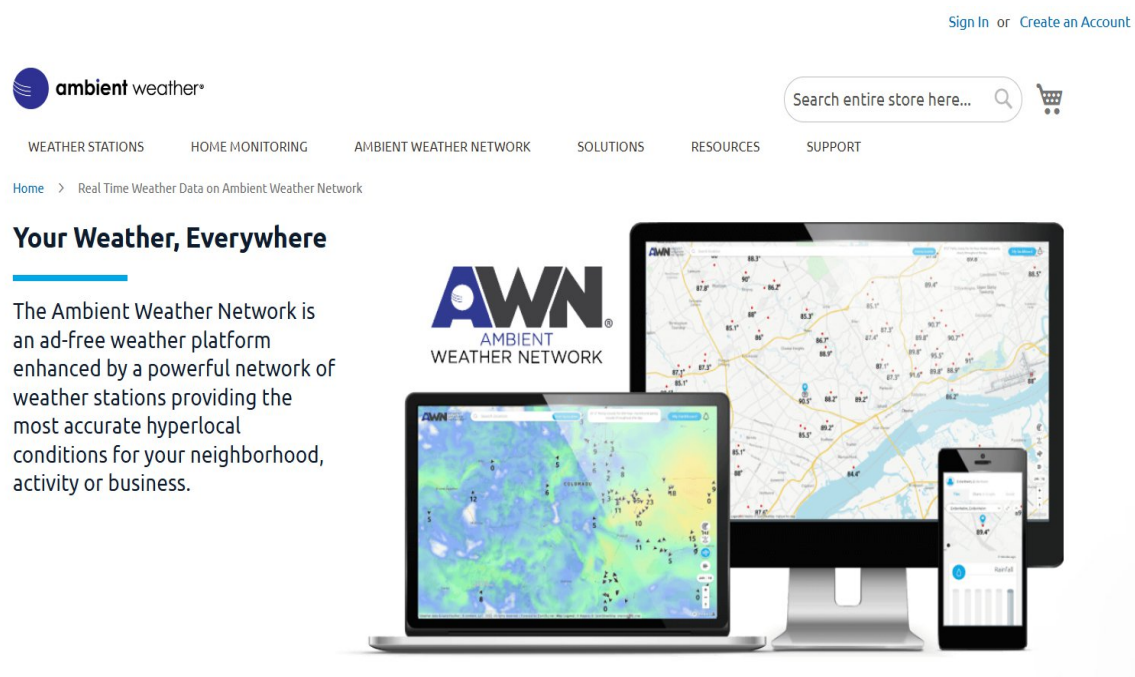


Рисунок 1.2 – Програмне забезпечення Ambient Weather Network [12]

Програмне забезпечення Ambient Weather Network має такі особливості [7], [12]:

- інтерфейс користувача: програмне забезпечення Ambient Weather Network надає зручна та інтуїтивна панель управління для перегляду погодних даних у режимі реального часу [7], [12];

- інтеграція з пристроями: програмне забезпечення Ambient Weather Network підтримує широкий спектр метеостанцій від Ambient Weather та інших сумісних систем [7], [12];

- доступність даних: програмне забезпечення Ambient Weather Network надає можливість користувачу перегляду інформації через вебпортал або мобільні додатки [7], [12];

- функції моніторингу: програмне забезпечення Ambient Weather Network надає детальне відображення температури, вологості, тиску, швидкості та напрямку вітру [7], [12];

- збереження історії: програмне забезпечення Ambient Weather Network підтримує накопичення даних з можливістю аналізу погодних змін у минулому [7], [12];

- інтеграція з сервісами: програмне забезпечення Ambient Weather Network підтримує підключення до Alexa, Google Assistant та інших розумних платформ [7], [12];

- персоналізація: програмне забезпечення Ambient Weather Network надає налаштування сповіщень і віджетів для відображення потрібної інформації [7], [12];

- спільний доступ: програмне забезпечення Ambient Weather Network надає можливість ділитися погодними даними з іншими користувачами та у відкритих мережах [7], [12].

Серед недоліків програмного забезпечення Ambient Weather Network можна відзначити обмежену функціональність без використання офіційних метеостанцій, іноді повільне оновлення інформації у мобільних додатках та залежність якості роботи від стабільності Інтернет з'єднання [7], [12].

Програмне забезпечення Weather Display (рис. 1.3) є програмою, створеною для підвищення ефективності роботи метеостанцій та забезпечення максимальної точності у відображенні погодних характеристик [7], [13].

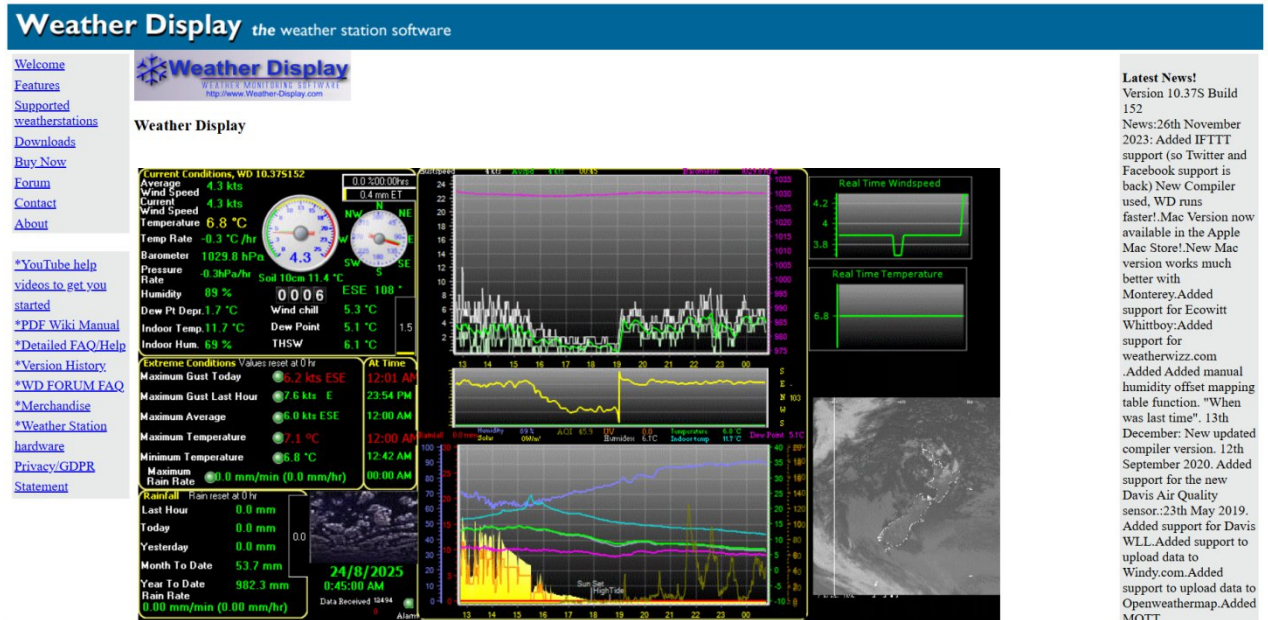


Рисунок 1.3 – Програмне забезпечення Weather Display [13]

Використання програмного забезпечення Weather Display дозволяє отримувати дані в реальному часі та формувати графіки, що відображають динаміку змін, що робить аналіз більш наочним і доступним. Завдяки підтримці великої кількості різноманітних моделей обладнання, яке випускається провідними виробниками, забезпечується гнучкість та універсальність у застосуванні. Додатковою перевагою є можливість завантаження інформації через протокол FTP, що відкриває шлях до збереження та поширення даних у віддалених ресурсах [7], [13].

Програмне забезпечення Weather Display передбачає сповіщення, які формуються з певною періодичністю залежно від обраного користувачем режиму, однак вони не надсилаються безперервно, що дозволяє уникнути надмірної кількості повідомлень. Сумісність з різними операційними системами, включаючи найбільш поширені версії Windows, macOS, Linux та

навіть Raspberry Pi, розширює можливості використання програми як для особистих потреб, так і для професійного застосування [7], [13].

Програмне забезпечення Weather Display має такі особливості [7], [13]:

- підтримка різних метеостанцій: програмне забезпечення Weather Display сумісне з великою кількістю моделей і виробників метеостанцій, що забезпечує гнучкість у використанні [7], [13];

- детальна візуалізація: програмне забезпечення Weather Display дозволяє створювати графіки, діаграми та інші засоби відображення даних у режимі реального часу [7], [13];

- інтеграція з онлайн сервісами: програмне забезпечення Weather Display може передавати дані на метеорологічні портали та в мережеві спільноти [7], [13];

- автоматичні звіти: є можливість генерування погодних звітів і статистики у зручних форматах [7], [13];

- налаштування інтерфейсу: користувач програмного забезпечення Weather Display може змінювати вигляд робочого середовища відповідно до власних потреб [7], [13];

- розширені аналітичні функції: програмне забезпечення Weather Display надає інструменти для аналізу історичних погодних даних та прогнозування тенденцій [7], [13];

- збереження історії: програмне забезпечення Weather Display підтримує накопичення даних з можливістю аналізу погодних змін у минулому [7], [13];

- сумісність з різними операційними системами: програмне забезпечення Weather Display може працювати на різних операційних системах [7], [13].

Серед недоліків програмного забезпечення Weather Display можна відзначити складність налаштування, що вимагає від користувачів певних технічних знань, обмежену кількість україномовних ресурсів для навчання та

підтримки, а також періодичні проблеми з коректним відображенням даних у разі використання нестандартних метеостанцій [7], [13].

Порівняльну характеристику програмного забезпечення для прогнозування погоди наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для прогнозування погоди

| Критерій порівняння             | WeatherBug [11] | Ambient Weather Network [12] | Weather Display [13] |
|---------------------------------|-----------------|------------------------------|----------------------|
| Підтримка різних платформ       | +               | +-                           | +-                   |
| Наявність мобільного застосунку | +               | +-                           | -                    |
| Інтеграція з метеостанціями     | -               | +                            | +                    |
| Можливості налаштування         | -               | +-                           | +                    |
| Зручність у використанні        | +               | +-                           | -                    |
| Наявність додаткових функцій    | +               | +-                           | +                    |

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для прогнозування погоди. Проте деякі програмні засоби для прогнозування погодних характеристик можуть використовувати неефективні методи та моделі, що призводить до низької точності прогнозів. Важливим недоліком є висока залежність якості прогнозів від обсягу й достовірності вхідних даних. Ще однією проблемою є невизначеність у прогнозах, яка часто ускладнює їхнє практичне застосування. Також варто зазначити, що програмне забезпечення, орієнтоване на глобальні моделі, не завжди забезпечує достатню точність для локальних територій. Таким чином, головними недоліками програмних систем прогнозування погоди виступають обмежена точність та локальна деталізація прогнозів, залежність від якості вхідних даних, висока ресурсомісткість та складність

практичного використання результатів у різних сферах діяльності. Тому актуальною є розробка програмного забезпечення для прогнозування погодних характеристик.

При розробці програмного забезпечення для прогнозування погодних характеристик необхідно забезпечити такі функціональні вимоги:

- підтримка можливості введення користувачем вхідних даних (широти та довготи) для подальшого прогнозування погодних характеристик;
- підтримка можливості надсилання користувачем запиту до метеосервісу для отримання погодних даних;
- можливість обробки даних та прогнозування погодних характеристик;
- можливість виведення в екранну форму результатів щодо прогнозування температури, вологості, швидкості вітру та інших характеристик;
- підтримка можливості прогнозування погодних характеристик для інших місць (повторного прогнозування при інших вхідних даних);
- підтримка можливості збереження результатів прогнозування.

### **1.3 Висновки за розділом 1**

Визначено, що програмне забезпечення для прогнозування погодних характеристик являє собою інструмент, що забезпечує обчислення, обробку та представлення результатів аналізу атмосферних процесів у зручному для користувачів вигляді. Такі програмні системи ґрунтуються на обробці великих масивів даних, що збираються з різних джерел, і дозволяють створювати прогнози з різними часовими та просторовими масштабами.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для прогнозування погоди. Проте деякі програмні засоби для прогнозування погодних характеристик можуть використовувати неефективні методи та моделі, що призводить до низької

точності прогнозів. Важливим недоліком є висока залежність якості прогнозів від обсягу й достовірності вхідних даних. Ще однією проблемою є невизначеність у прогнозах, яка часто ускладнює їхнє практичне застосування. Також варто зазначити, що програмне забезпечення, орієнтоване на глобальні моделі, не завжди забезпечує достатню точність для локальних територій. Таким чином, головними недоліками програмних систем прогнозування погоди виступають обмежена точність та локальна деталізація прогнозів, залежність від якості вхідних даних, висока ресурсомісткість та складність практичного використання результатів у різних сферах діяльності. Тому актуальною є розробка програмного забезпечення для прогнозування погодних характеристик.

Сформульовано функціональні вимоги до програмного забезпечення для прогнозування погодних характеристик.

## 2 МАТЕРІАЛИ І МЕТОДИ

### 2.1 Вибір мови програмування

При розробці програмного забезпечення для прогнозування погодних характеристик було використано мову програмування Python [14], [15].

Мова програмування Python є ефективною для реалізації сучасних обчислювальних рішень та наукових досліджень завдяки поєднанню гнучкості та універсальності. Її застосування охоплює широкий спектр сфер — від аналітики даних і математичного моделювання до веброзробки та автоматизації складних процесів. Висока виразність синтаксису дозволяє створювати алгоритмічні конструкції різного рівня складності, забезпечуючи при цьому ефективність реалізації завдань, що потребують обробки великих обсягів інформації [14], [15].

Велика кількість відкритих бібліотек і модулів забезпечує швидку інтеграцію різноманітних інструментів для обчислень, статистичного аналізу та машинного навчання, що значно прискорює розробку програмного забезпечення і розширює його функціональні можливості. Python активно використовується у сфері штучного інтелекту та нейронних мереж, а також у візуалізації та обробці просторових даних, що відкриває широкі перспективи для створення аналітичних панелей та наукових візуалізацій [14], [15].

Важливим чинником популярності мови програмування Python є активна спільнота розробників, яка забезпечує постійний розвиток екосистеми, підтримку та оновлення бібліотек, а також сприяє впровадженню інноваційних рішень у наукову та прикладну діяльність. Саме завдяки поєднанню простоти, масштабованості та широкого набору інструментів Python став ключовим середовищем для розробки програмних систем, здатних ефективно працювати з великими даними, прогнозними моделями та інтелектуальними технологіями [14], [15].

Вибір мови програмування Python для створення програмного забезпечення, призначеного для прогнозування погодних характеристик,

обґрунтовується низкою технічних та науково-практичних особливостей. Мова вирізняється наявністю потужного набору бібліотек і фреймворків, призначених для обробки даних, чисельного моделювання та реалізації алгоритмів машинного навчання. Використання інструментів на кшталт NumPy, Pandas, SciPy, scikit-learn, TensorFlow та PyTorch дозволяє ефективно працювати з великими обсягами метеорологічної інформації та створювати прогностичні моделі різного рівня складності [14], [15].

Значний вплив на поширення Python має багата екосистема бібліотек і фреймворків, що охоплюють практично всі напрями сучасної інформатики. Для чисельних обчислень та наукового моделювання активно застосовуються спеціалізовані пакети, які дають змогу виконувати операції з великими масивами даних, працювати з багатовимірними структурами та реалізовувати методи статистичного аналізу. У галузі машинного навчання та штучного інтелекту Python став базовою платформою завдяки наявності інструментарію для побудови, навчання та оптимізації нейронних мереж. Не менш важливою є його роль у візуалізації інформації та обробці просторових даних, що забезпечує широкі можливості для створення інтерфейсів, аналітичних панелей і наукових візуалізацій [14], [15].

Також значним чинником є можливості візуалізації результатів за допомогою бібліотек Matplotlib, Seaborn, Plotly, що дає змогу відтворювати графіки й карти розподілу погодних параметрів для наукового аналізу та практичного використання кінцевими користувачами [14], [15].

Велике значення має інтеграція Python із геоінформаційними системами та інструментами роботи з просторовими даними. Бібліотеки GeoPandas, Rasterio, netCDF4 та подібні забезпечують обробку спеціалізованих форматів кліматичних і супутникових даних, що є критично важливим для точності прогнозів [14], [15].

Окрім технічних переваг, мова відзначається простотою синтаксису, що робить її доступною для дослідників і метеорологів, які не завжди мають високий рівень знань у програмуванні. Активна спільнота розробників сприяє

швидкому обміну досвідом, появі нових інструментів і підтримці відкритих наукових проєктів [14], [15].

Таким чином, Python поєднує в собі простоту використання, потужність і універсальність, що обґрунтовує його застосування для розробки програмного забезпечення прогнозування погоди, яке ґрунтується на обробці великих даних, чисельних моделях і технологіях штучного інтелекту [14], [15].

Обґрунтування вибору мови програмування для реалізації програмного забезпечення для прогнозування погодних характеристик наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для реалізації програмного забезпечення для прогнозування погодних характеристик

| Критерій порівняння мов програмування                               | Мова програмування |      |    |
|---------------------------------------------------------------------|--------------------|------|----|
|                                                                     | Python             | Java | R  |
| Бібліотеки для аналізу даних                                        | +                  | +-   | +  |
| Підтримка машинного навчання                                        | +                  | +-   | +- |
| Засоби візуалізації результатів                                     | +                  | -    | +- |
| Зручність для створення ПЗ для прогнозування погодних характеристик | +                  | -    | -  |
| Низький поріг входу для користувачів                                | +                  | +-   | +- |

Отже, для реалізації програмного забезпечення для прогнозування погодних характеристик обрано мову програмування Python, яка вирізняється широким набором бібліотек і фреймворків для роботи з даними, математичного моделювання та машинного навчання. Крім того, ця мова відзначається простотою синтаксису, що робить її доступною для дослідників

і фахівців з метеорології, які не завжди мають глибокі знання у сфері програмної інженерії. Активна спільнота розробників сприяє швидкому обміну знаннями, появі нових інструментів та підтримці відкритих наукових проєктів.

## **2.2 Вибір середовища розробки для створення програмного забезпечення для прогнозування погодних характеристик**

В якості середовища розробки для створення програмного забезпечення для прогнозування погодних характеристик було обрано середовище PyCharm [16], [17].

Середовище розробки PyCharm може розглядатися як високо функціональний інструмент для роботи з мовою Python, який забезпечує не лише зручність програмування, але й комплексне управління проєктами різного рівня складності. Інтеграція з численними технологіями та підтримка різноманітних інструментів дозволяють одночасно виконувати написання коду, його тестування та налагодження, підвищуючи ефективність робочого процесу. Інтелектуальні механізми автодоповнення та підказок сприяють скороченню часу розробки і мінімізації помилок. Середовище також оптимізоване для командної роботи завдяки інтеграції з системами контролю версій і віддаленими репозиторіями, що полегшує спільну розробку та управління проєктами. Особлива увага приділяється роботі з даними, що робить PyCharm актуальним у наукових дослідженнях і прикладних проєктах, де обробляються великі масиви інформації. У результаті PyCharm формується як сучасне середовище, здатне поєднувати простоту використання з широкими функціональними можливостями та забезпечувати ефективність розробки складних програмних систем [16], [17].

Вибір середовища розробки PyCharm для створення програмних рішень, спрямованих на прогнозування погодних характеристик,

зумовлюється комплексом переваг, що підвищують ефективність процесу розробки та зручність роботи з Python [16], [17].

Середовище розробки PyCharm забезпечує розширені можливості для аналітичної обробки та моделювання, що спрощує управління великими обсягами метеорологічних даних. Підтримка численних бібліотек і фреймворків, серед яких NumPy, Pandas, Matplotlib та scikit-learn, створює умови для реалізації алгоритмів обробки інформації та побудови точних прогнозних моделей [16], [17].

Важливою характеристикою PyCharm є ефективність налагодження та тестування коду, що дозволяє своєчасно виявляти помилки та підвищувати стабільність програмного забезпечення. Це особливо критично у сфері прогнозування погоди, де навіть незначні неточності можуть впливати на результативність моделей [16], [17].

Середовище розробки PyCharm також інтегрується з системами контролю версій, хмарними сервісами та інструментами керування проєктами, що спрощує організацію колективної розробки, обмін результатами та оперативне вдосконалення програмних рішень. Інтелектуальні підказки, автодоповнення коду та зручна навігація забезпечують прискорення робочого процесу та зменшують ймовірність синтаксичних помилок, роблячи середовище PyCharm особливо цінним для науково-технічних проєктів великого масштабу [16], [17].

Таким чином, середовище розробки PyCharm являє собою оптимальне середовище для розробки програмного забезпечення прогнозування погоди, об'єднуючи гнучкість, продуктивність і широкий набір інструментів для ефективної роботи з Python та обробки великих даних [16], [17].

Обґрунтування вибору середовища розробки для створення програмного забезпечення для прогнозування погодних характеристик наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення для прогнозування погодних характеристик

| Критерій порівняння середовищ розробки | Середовища розробки |                  |        |
|----------------------------------------|---------------------|------------------|--------|
|                                        | PyCharm             | Jupyter Notebook | Spyder |
| Засоби налагодження та тестування      | +                   | –                | +–     |
| Автодоповнення та підказки             | +                   | +–               | +      |
| Інтеграція з системами контролю версій | +                   | –                | +–     |
| Ефективність для командної роботи      | +                   | –                | +–     |
| Швидкість розробки великих систем      | +                   | –                | +–     |

Таким чином, для створення програми для прогнозування погодних характеристик обрано середовище розробки PyCharm, що поєднує у собі інструменти для повноцінної розробки програмного забезпечення, зокрема, підтримку масштабних проєктів, інтеграцію з бібліотеками, засоби налагодження, управління версіями та ефективну організацію командної роботи.

### **2.3 Розробка моделі програмного забезпечення для прогнозування погодних характеристик**

З метою можливості розробки програмного забезпечення для прогнозування погодних характеристик створимо відповідну модель (рис. 2.1), що являє собою багаторівневу систему, у якій відбувається взаємодія між

користувачем, застосунком, локальним сховищем та зовнішнім метеосервісом. Основна ідея такої моделі полягає в забезпеченні безперервного циклу обробки даних: від отримання початкової інформації до формування прогнозних результатів, що подаються користувачеві у зручній та зрозумілій формі.

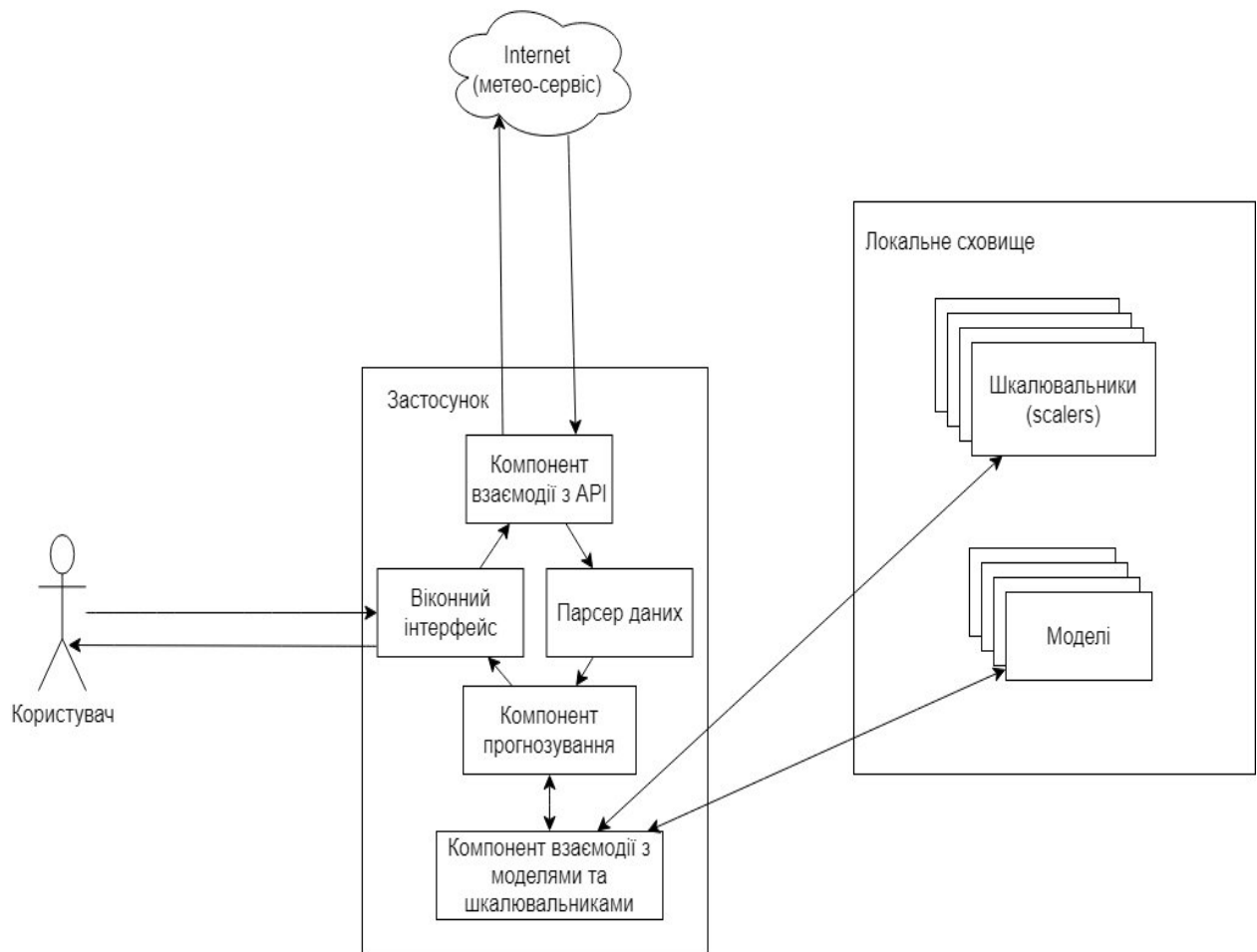


Рисунок 2.1 – Модель програмного забезпечення для прогнозування погодних характеристик

У центрі моделі перебуває застосунок, який виконує роль координатора всіх процесів. Він об'єднує в собі кілька взаємопов'язаних компонентів. Компонент взаємодії з API відповідає за отримання метеорологічних даних з Інтернет сервісів, встановлюючи зв'язок із зовнішніми джерелами та забезпечуючи передачу запитів і прийом відповідей у стандартизованих форматах. Дані, що надходять від метеосервісів, обробляються парсером, який

трансформує їх у внутрішній формат системи, придатний для подальшого аналізу та використання прогнозними алгоритмами.

Іншою складовою є віконний інтерфейс, завдяки якому користувач отримує доступ до функціональних можливостей застосунку. Саме через нього відбувається введення параметрів, вибір налаштувань, запуск розрахунків та перегляд результатів у вигляді графіків, таблиць чи картографічних візуалізацій. Віконний інтерфейс формує міст між користувачем і системою, роблячи роботу з програмним забезпеченням інтуїтивно зрозумілою та доступною навіть для тих, хто не має спеціальної підготовки у сфері програмування чи математичного моделювання.

Ключове місце у структурі займає компонент прогнозування, який реалізує алгоритмічні та математичні методи аналізу. Саме тут виконується обробка даних, застосування статистичних і машинних моделей, а також побудова прогнозів на основі історичних спостережень і поточних показників. Для коректної роботи цього модуля важливою є взаємодія із локальним сховищем, де зберігаються попередньо підготовлені моделі та шкалювальники. Компонент взаємодії з моделями та шкалювальниками забезпечує узгодження роботи алгоритмів із даними, виконуючи стандартизацію та нормалізацію вхідної інформації. Завдяки цьому система може застосовувати різні варіанти моделей, порівнювати їхню ефективність та підбирати оптимальний підхід для конкретних умов.

Локальне сховище у цій архітектурі функціонує як база знань, що зберігає математичні моделі та шкалювальники, необхідні для коректної обробки вхідних даних. Завдяки використанню локальних ресурсів забезпечується швидкий доступ до необхідних інструментів без потреби постійного звернення до зовнішніх джерел, що підвищує автономність системи та зменшує залежність від стабільності Інтернет з'єднання.

Взаємодія між усіма елементами моделі має циклічний характер. Користувач ініціює роботу, використовуючи інтерфейс застосунку. Застосунок звертається до метеосервісу для отримання актуальних даних,

після чого передає їх до парсера. Оброблені дані потрапляють у компонент прогнозування, де виконуються необхідні обчислення з урахуванням моделей і шкалювальників із локального сховища. Результати повертаються до інтерфейсу й відображаються у зручному форматі, створюючи завершений цикл обміну інформацією між користувачем і системою.

У такій моделі забезпечується баланс між роботою з великими обсягами інформації, гнучкістю інтеграції та точністю прогнозів. Поєднання Інтернет сервісів, локального сховища та розвиненого функціоналу застосунку дозволяє реалізувати ефективне середовище для досліджень, практичних розробок та прикладного використання у сфері метеорології.

Таким чином, створено модель програмного забезпечення для прогнозування погодних характеристик, яка подана у вигляді багаторівневої системи, у якій відбувається взаємодія між користувачем, застосунком, локальним сховищем та зовнішнім метеосервісом. Основна ідея такої моделі полягає в забезпеченні безперервного циклу обробки даних: від отримання початкової інформації до формування прогнозних результатів, що подаються користувачеві у зручній та зрозумілій формі. Поєднання Інтернет сервісів, локального сховища та розвиненого функціоналу застосунку дозволяє реалізувати ефективне середовище для досліджень, практичних розробок та прикладного використання у сфері метеорології.

## **2.4 Висновки за розділом 2**

Для реалізації програмного забезпечення для прогнозування погодних характеристик обрано мову програмування Python, яка вирізняється широким набором бібліотек і фреймворків для роботи з даними, математичного моделювання та машинного навчання. Крім того, ця мова відзначається простотою синтаксису, що робить її доступною для дослідників і фахівців з метеорології, які не завжди мають глибокі знання у сфері програмної

інженерії. Активна спільнота розробників сприяє швидкому обміну знаннями, появі нових інструментів та підтримці відкритих наукових проєктів.

Для створення програми для прогнозування погодних характеристик обрано середовище розробки PyCharm, що поєднує у собі інструменти для повноцінної розробки програмного забезпечення, зокрема, підтримку масштабних проєктів, інтеграцію з бібліотеками, засоби налагодження, управління версіями та ефективну організацію командної роботи.

Створено модель програмного забезпечення для прогнозування погодних характеристик, яка подана у вигляді багаторівневої системи, у якій відбувається взаємодія між користувачем, застосунком, локальним сховищем та зовнішнім метеосервісом. Основна ідея такої моделі полягає в забезпеченні безперервного циклу обробки даних: від отримання початкової інформації до формування прогнозних результатів, що подаються користувачеві у зручній та зрозумілій формі. Поєднання Інтернет сервісів, локального сховища та розвинуеного функціоналу застосунку дозволяє реалізувати ефективне середовище для досліджень, практичних розробок та прикладного використання у сфері метеорології.

### 3 ОПИС ПРОГРАМИ

#### 3.1 Структура програмного забезпечення для прогнозування погодних характеристик

Структуру програмного забезпечення для прогнозування погодних характеристик наведено на рис. 3.1.

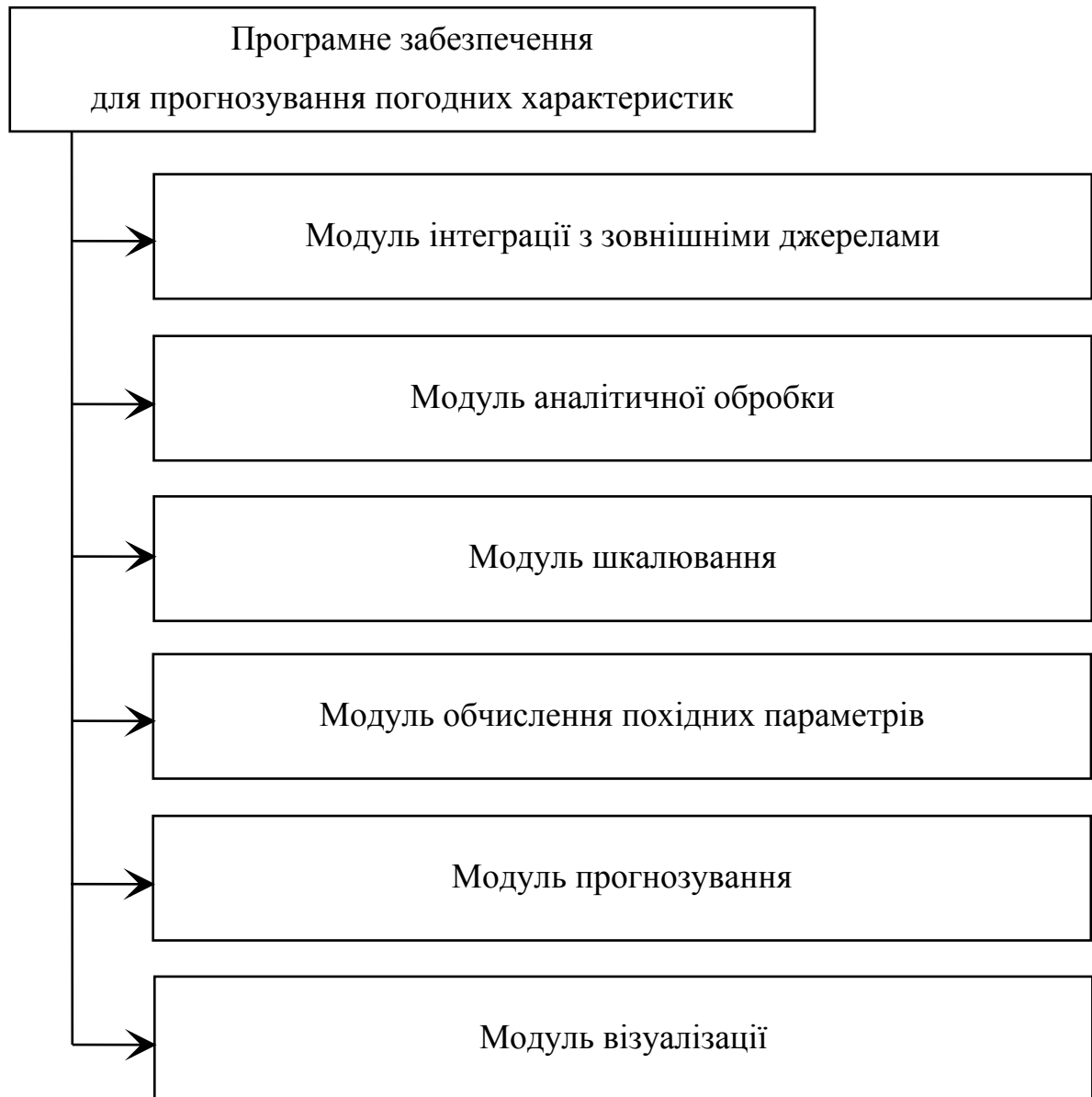


Рисунок 3.1 – Структура програмного забезпечення для прогнозування погодних характеристик

Структура програмного забезпечення для прогнозування погодних характеристик формується як цілісна система, у якій кожен структурний компонент виконує взаємопов'язану роль, забезпечуючи послідовний перехід від введення вихідних даних до представлення кінцевого результату у зручному для користувача вигляді. Основними модулями програмного забезпечення для прогнозування погодних характеристик є такі: модуль інтеграції з зовнішніми джерелами, модуль аналітичної обробки, модуль шкалювання, модуль обчислення похідних параметрів, модуль прогнозування, модуль візуалізації.

У модулі інтеграції з зовнішніми джерелами отримані координати місцевості, яка становить інтерес для прогнозування, передаються до системи, що забезпечує зв'язок із погодними API. Цей модуль здійснює HTTP-запити для отримання історичних даних за останні дні, а також прогнозних значень на найближчий час. Таким чином, створюється інформаційна база, необхідна для статистичного аналізу та побудови моделей передбачення.

У модулі аналітичної обробки виконується розпакування масивів інформації, їхнє структурування та розрахунок статистичних показників, таких як середні значення температури, максимальні й мінімальні параметри вологості чи атмосферного тиску. Завдяки цьому вхідні дані набувають придатності для подальшої роботи в межах прогнозних алгоритмів.

У модулі шкалювання виконується перетворення значень даних для забезпечення правильного співвідношення параметрів для застосування у прогнозних моделях. Таке попереднє нормування підвищує точність і стабільність результатів моделювання.

У модулі обчислення похідних параметрів здійснюється формування інтегральних показників, зокрема, накопиченої кількості опадів, які можуть мати вагоме значення при побудові точніших моделей прогнозування. Таким чином, створюється набір характеристик, який відображає не лише миттєві значення, а й тенденції у динаміці погодних процесів.

Модуль прогнозування працює на основі алгоритмів машинного навчання. Його завдання полягає у використанні підготовлених і нормованих даних для формування прогнозних значень температури, вологості та інших метеорологічних характеристик. Результати використання цього модулю безпосередньо визначають практичну цінність програмного забезпечення.

Модуль візуалізації відповідає за введення даних та представлення прогнозних показників у графічному інтерфейсі. За допомогою цього модуля користувач вводить координати місцевості, яка становить інтерес для прогнозування. Саме ці дані стають базовою відправною точкою, оскільки географічні параметри визначають характер подальших обчислень і вибір інформаційних ресурсів. Завдяки цьому модулю користувач отримує доступ не лише до числових значень, але й до їх наочного відображення у вигляді графіків, діаграм чи картографічних моделей. Такий підхід сприяє інтуїтивному сприйняттю результатів і підвищує практичну придатність програмного продукту.

Отже, структурна організація програмного забезпечення вибудовується як багаторівнева система з шести модулів, які взаємодіють між собою, утворюючи єдиний алгоритмічний цикл — від введення координат користувачем до наочного представлення прогнозу.

### **3.2 Функціонування програмного забезпечення для прогнозування погодних характеристик**

Функціонування програмного забезпечення для прогнозування погодних характеристик подамо за допомогою схеми, зображеної на рис. 3.2.

Функціонування програмного забезпечення для прогнозування погодних характеристик відбувається у вигляді послідовного алгоритму, який поєднує завантаження моделей, роботу з користувачем, обробку метеорологічних даних і формування кінцевого прогнозу.



Рисунок 3.2 – Схема функціонування програмного забезпечення для прогнозування погодних характеристик

Початковим етапом є ініціалізація системи, у межах якої активуються всі необхідні ресурси та здійснюється підготовка моделей, призначених для прогнозування окремих погодних параметрів. На цьому етапі формується основа для подальшої роботи, адже саме завантажені алгоритми визначають точність і стабільність результатів.

Після цього створюється графічне вікно з полями для введення, де користувачеві пропонується зазначити координати місцевості, для якої необхідно отримати прогноз погоди. Отримані дані стають відправною точкою для наступних операцій, оскільки вони дозволяють звернутися до відповідних джерел інформації. Далі відбувається комунікація з метеорологічним API, що забезпечує доступ до архівних показників за попередні дні, необхідних для якісного моделювання.

Завантажені від сервісу дані проходять процес парсингу та структурованого аналізу. На цьому етапі виконується їх попереднє впорядкування, виділення ключових параметрів і приведення до формату, придатного для обробки прогнозними моделями. Після здійснення цих дій система переходить безпосередньо до етапу прогнозування. Моделі на основі отриманих даних генерують передбачення основних метеорологічних характеристик, серед яких швидкість вітру, температура повітря, кількість опадів, рівень хмарності та відносна вологість.

Результати розрахунків у подальшому відображаються у вікні програми, що дає можливість користувачеві ознайомитися з прогнозом у зрозумілій та зручній формі. Після цього система пропонує виконати повторне прогнозування. У випадку, якщо користувач виявляє зацікавленість у новому запиті, процес повертається до етапу введення координат і розпочинається повторно. Якщо ж потреби у новому прогнозі немає, робота програми завершується.

Таким чином, функціонування програмного забезпечення вибудовується як замкнений цикл, у межах якого забезпечується гнучке поєднання користувацької взаємодії з автоматизованою обробкою даних та

застосуванням прогнозних моделей, що дозволяє отримувати актуальні результати для будь-якої заданої місцевості.

### **3.3 Особливості реалізації програмного забезпечення для прогнозування погодних характеристик**

Реалізація програмного забезпечення для прогнозування погодних характеристик вибудовується як комплексна система, у якій кожен етап взаємодіє з іншими та формує цілісний процес отримання прогнозу. На початковому рівні організовується введення даних користувачем, де передбачено використання графічного інтерфейсу із віконною структурою. Саме в ньому користувач зазначає координати місцевості, після чого система проводить перевірку правильності введених значень і в разі некоректного вводу надає відповідні повідомлення. Такий підхід мінімізує можливість виникнення помилок на ранніх етапах та створює умови для якісної подальшої обробки інформації.

Подальший етап стосується взаємодії з зовнішніми сервісами, а саме з метеорологічними API. Для цього здійснюється попередній вибір найбільш доцільного API, що забезпечує доступ до необхідних метеорологічних даних у максимально актуальному вигляді. У процесі роботи з API враховується специфіка його відповіді, зокрема можливість виникнення помилок або нестандартних форматів, що вимагає реалізації спеціальних механізмів для аналізу та адаптації отриманої інформації.

Обробка даних, отриманих від API, відіграє ключову роль, оскільки саме на цьому етапі відбувається структурування, узгодження й підготовка метеорологічних показників до подальшого використання. Для цього застосовуються алгоритми, що дозволяють не лише впорядкувати дані, а й здійснити попередній аналіз, виділяючи з них найбільш значущі параметри для майбутнього прогнозу.

Після цього для кожного погодного показника підбираються оптимальні методи приведення до узгодженого діапазону, що гарантує коректність обчислень у прогнозних моделях. Реалізація такого механізму включає використання окремих шкалювальників для кожного параметра, що дозволяє зберігати індивідуальність підходу до різних типів даних і забезпечує високу точність результатів.

Прогнозування здійснюється на основі моделей машинного навчання, серед яких можуть використовуватися як глибинні нейронні мережі, так і рекурентні архітектури типу LSTM. Для кожного метеорологічного параметра передбачено застосування окремої моделі, що забезпечує спеціалізоване налаштування та зменшує похибку. Тренування цих моделей відбувається на історичних наборах даних, що дозволяє досягти оптимальної продуктивності під час роботи з новою інформацією. У процесі виконання програми потрібні моделі завантажуються динамічно, що підвищує гнучкість і забезпечує швидке реагування на користувацькі запити.

Результати прогнозування мають бути відображені в зручній і зрозумілій формі. Тому особлива увага приділяється візуалізації вихідних даних, яка реалізується через графічний інтерфейс. Такий підхід дозволяє користувачеві не лише отримати цифрові значення параметрів, але й побачити їх на графіках чи діаграмах, що значно підвищує наочність і спрощує сприйняття інформації.

Завдяки зручному інтерфейсу створюються умови для простого введення даних і швидкого доступу до результатів, що робить програмне забезпечення інтуїтивно зрозумілим навіть для тих, хто не має глибоких технічних знань. Таким чином, реалізація системи базується на поєднанні ефективних алгоритмів, гнучких методів машинного навчання та дружнього користувацького середовища, що у комплексі забезпечує точність, зручність і практичну цінність прогнозів.

Особливості реалізації програмного забезпечення для прогнозування погодних характеристик зумовлюються необхідністю інтеграції декількох

функціональних компонентів, кожен із яких виконує власне завдання та водночас забезпечує безперервність загального процесу. Робота починається зі створення механізму введення даних, що передбачає розробку графічного інтерфейсу для отримання від користувача координат широти та довготи. На цьому етапі важливо передбачити коректність вводу, оскільки помилкові значення здатні вплинути на точність подальших результатів. Тому у програмі реалізується процедура перевірки введених параметрів, яка попереджає користувача про помилки та дозволяє їх виправити. Для побудови інтерфейсу застосовується спеціалізоване програмне забезпечення, наприклад бібліотека `tkinter`, що дає змогу створювати прості та водночас функціональні віконні форми.

В основі системи функціонує модуль прогнозування, де застосовуються алгоритми машинного навчання. Залежно від завдань і характеристик даних можуть використовуватися нейронні мережі глибинного навчання чи рекурентні моделі. Для кожного погодного параметра передбачається окрема модель, що підвищує точність передбачення. Ефективність прогнозування забезпечується шляхом попереднього тренування моделей на історичних метеорологічних наборах даних, де враховуються складні залежності між різними параметрами.

Результати роботи моделі доводяться до користувача у зрозумілому вигляді за допомогою спеціального інтерфейсу виводу, який наочно відображає прогнозовані значення. Візуалізація даних може бути реалізована у вигляді таблиць, графіків чи діаграм, що робить інформацію більш доступною та зрозумілою. Відповідний модуль інтегрується з іншими компонентами системи, забезпечуючи цілісність програмного продукту та безперервність обміну даними.

Таким чином, особливості реалізації програмного забезпечення полягають у послідовному формуванні взаємопов'язаних модулів, від введення координат користувачем і збирання метеорологічних даних до їх математичної обробки, прогнозування та наочного подання результатів.

Комплексний характер реалізації дозволяє створити систему, здатну не лише забезпечити точність передбачень, а й надати користувачеві зручний інструмент для взаємодії з програмою.

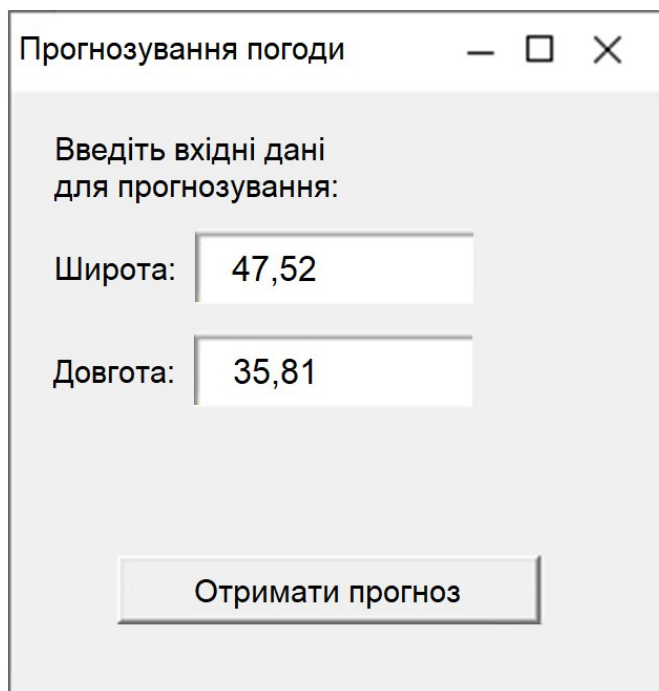
### **3.4 Проєктування інтерфейсу програмного забезпечення для прогнозування погодних характеристик**

Проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для прогнозування погодних характеристик виконано з урахуванням вимог технічного завдання. При розробленні програмного забезпечення для прогнозування погодних характеристик враховані функціональні вимоги до програми:

- підтримка можливості введення користувачем вхідних даних (широти та довготи) для подальшого прогнозування погодних характеристик;
- підтримка можливості надсилання користувачем запиту до метеосервісу для отримання погодних даних;
- можливість обробки даних та прогнозування погодних характеристик;
- можливість виведення в екранну форму результатів щодо прогнозування температури, вологості, швидкості вітру та інших характеристик;
- підтримка можливості прогнозування погодних характеристик для інших місць (повторного прогнозування при інших вхідних даних);
- підтримка можливості збереження результатів прогнозування.

Інтерфейс головної форми програмного забезпечення для прогнозування погодних характеристик наведено на рис. 3.3.

Інтерфейс форми виведення результатів прогнозування погодних характеристик наведено на рис. 3.4.



Прогнозування погоди

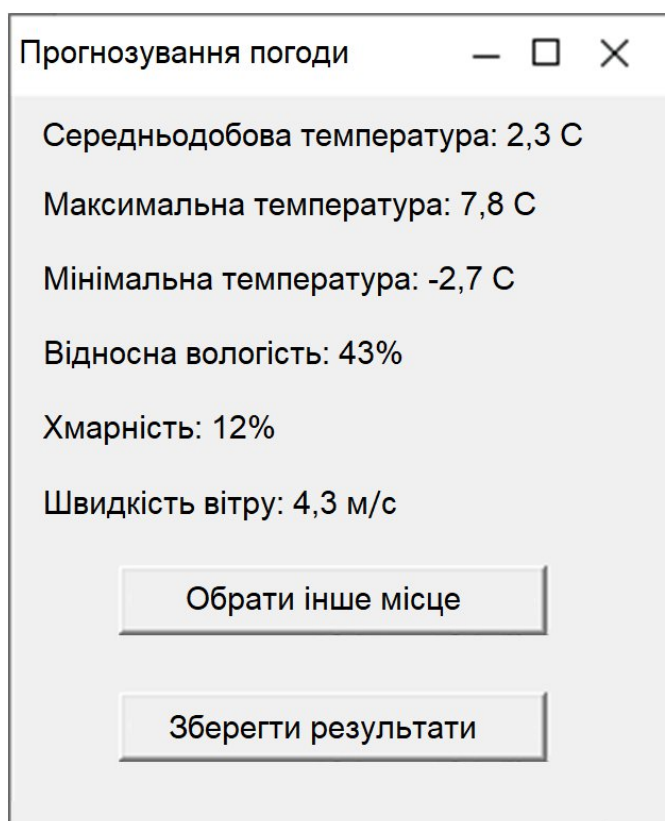
Введіть вхідні дані  
для прогнозування:

Широта: 47,52

Довгота: 35,81

Отримати прогноз

Рисунок 3.3 – Інтерфейс головної форми програмного забезпечення для прогнозування погодних характеристик



Прогнозування погоди

Середньодобова температура: 2,3 С

Максимальна температура: 7,8 С

Мінімальна температура: -2,7 С

Відносна вологість: 43%

Хмарність: 12%

Швидкість вітру: 4,3 м/с

Обрати інше місце

Зберегти результати

Рисунок 3.4 – Інтерфейс форми виведення результатів прогнозування погодних характеристик

### 3.5 Висновки за розділом 3

Розроблено програмне забезпечення для прогнозування погодних характеристик.

Запропоновано структуру програмного забезпечення для прогнозування погодних характеристик. Визначено, що кожен структурний компонент виконує взаємопов'язану роль, забезпечуючи послідовний перехід від введення вихідних даних до представлення кінцевого результату у зручному для користувача вигляді. Основними модулями програмного забезпечення для прогнозування погодних характеристик є такі: модуль інтеграції з зовнішніми джерелами, модуль аналітичної обробки, модуль шкалювання, модуль обчислення похідних параметрів, модуль прогнозування, модуль візуалізації.

Описано функціонування програмного забезпечення для прогнозування погодних характеристик. Визначено, що функціонування програми відбувається у вигляді послідовного алгоритму, який поєднує завантаження моделей, роботу з користувачем, обробку метеорологічних даних і формування кінцевого прогнозу.

Описано особливості реалізації програмного забезпечення для прогнозування погодних характеристик.

Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для прогнозування погодних характеристик.

## **4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ**

### **4.1 Призначення й умови застосування програми**

Програма призначена для прогнозування погодних характеристик. Програма написана на мові Python, яка вирізняється широким набором бібліотек і фреймворків для роботи з даними, математичного моделювання та машинного навчання. В якості середовища розробки для створення програми для прогнозування погодних характеристик обрано PyCharm, що поєднує у собі інструменти для повноцінної розробки програмного забезпечення, зокрема, підтримку масштабних проєктів, інтеграцію з бібліотеками, засоби налагодження, управління версіями та ефективну організацію командної роботи.

Програмне забезпечення для прогнозування погодних характеристик забезпечує виконання таких функцій:

- підтримка можливості введення користувачем вхідних даних (широти та довготи) для подальшого прогнозування погодних характеристик;
- підтримка можливості надсилання користувачем запиту до метеосервісу для отримання погодних даних;
- можливість обробки даних та прогнозування погодних характеристик;
- можливість виведення в екранну форму результатів щодо прогнозування температури, вологості, швидкості вітру та інших характеристик;
- підтримка можливості прогнозування погодних характеристик для інших місць (повторного прогнозування при інших вхідних даних);
- підтримка можливості збереження результатів прогнозування.

Для експлуатації програмного забезпечення для прогнозування погодних характеристик необхідно таке апаратне та програмне забезпечення:

- процесор із тактовою частотою не нижче 2,0–2,5 ГГц;

- оперативна пам'ять обсягом не менше 8 ГБ;
- жорсткий диск з вільним місцем не менше 20 ГБ;
- стабільне Інтернет з'єднання для завантаження метеорологічних даних з відкритих джерел чи API;
- операційна система, середовище розробки PyCharm, встановлений інтерпретатор Python 3.8 або вище, бібліотеки Python для роботи з даними та прогнозування: NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow.

Функціональні характеристики розробленого програмного забезпечення для прогнозування погодних характеристик наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення для прогнозування погодних характеристик наведено у додатку Б.

## **4.2 Характеристики програми для прогнозування погодних характеристик**

Програмне забезпечення для прогнозування погодних характеристик дозволяє виконувати автоматизовану обробку метеорологічних даних та формування прогнозів на основі аналітичних моделей. Його робота ґрунтується на використанні алгоритмів машинного навчання, які дозволяють виявляти закономірності у великих обсягах інформації. Система забезпечує зручний інтерфейс для взаємодії з користувачем, що дає можливість швидко отримувати результати прогнозів у зручному вигляді. Програмне забезпечення для прогнозування погодних характеристик відзначається високою точністю розрахунків, адаптивністю до зміни вихідних даних та здатністю до інтеграції з іншими програмними системами. Особливістю є можливість візуалізації результатів, що спрощує аналіз та робить інформацію зрозумілою для широкого кола користувачів. Також важливою рисою є підтримка різних форматів даних, що дозволяє використовувати програму в різних умовах та для різних цілей.

### **4.3 Інструкція по експлуатації програми**

#### **4.3.1 Звернення до програми**

Для запуску програмного забезпечення для прогнозування погодних характеристик необхідно запустити відповідний ру-файл.

#### **4.3.2 Вхідні й вихідні дані**

Вхідними даними до програмного забезпечення для прогнозування погодних характеристик є інформація про координати місця для прогнозування погоди (широта та довгота) та ін.

Вихідними даними програмного забезпечення для прогнозування погодних характеристик є прогнозовані значення погоди, зокрема, температури, вологості, швидкості вітру та інші характеристики.

#### **4.3.3 Повідомлення**

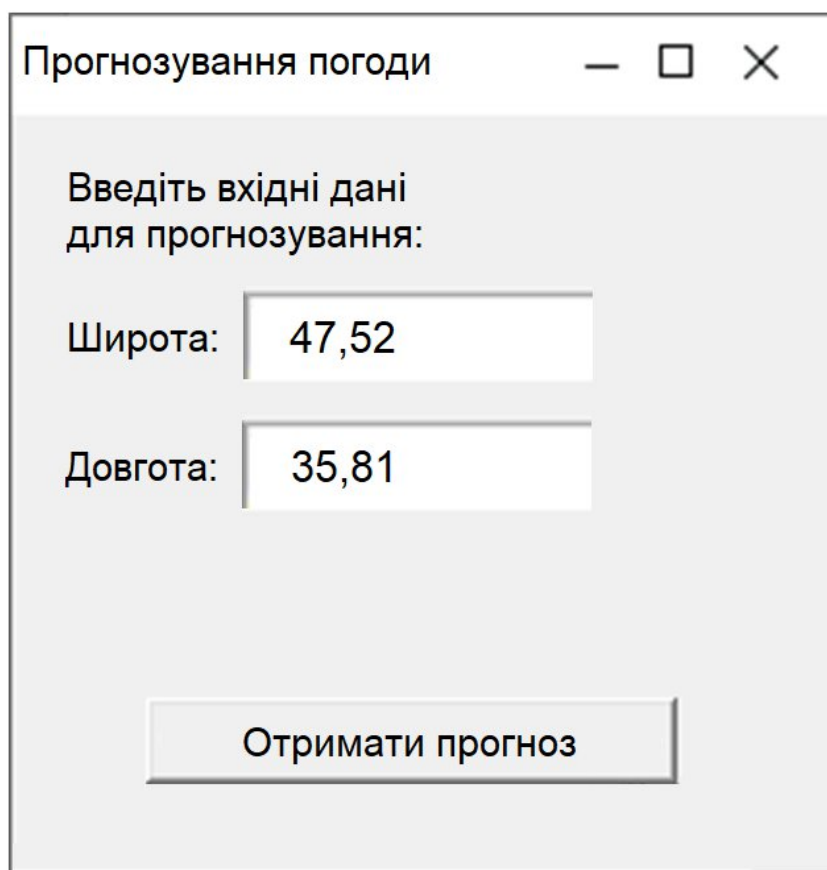
Користувач програмного забезпечення для прогнозування погодних характеристик може отримати такі повідомлення: повідомлення про некоректні дані координат місця для прогнозування погоди.

### **4.4 Виконання програмного забезпечення для прогнозування погодних характеристик**

Програму реалізовано як набір модулів для можливості використання в інших програмних системах. Для зручності роботи кінцевого користувача з програмою також реалізовано візуальний інтерфейс користувача. При цьому у версії з візуальним інтерфейсом реалізовано основні можливості програми, зокрема, введення користувачем вхідних даних та прогнозування погодних

характеристик. У консольній версії реалізовано додатково можливість завантаження даних з метеосервісу, відображення погодних графіків та ін.

Після запуску програмного забезпечення для прогнозування погодних характеристик користувачу відображається головна форма (рис. 4.1), де необхідно ввести вхідні дані – широту та довготу.



Прогнозування погоди

Введіть вхідні дані  
для прогнозування:

Широта: 47,52

Довгота: 35,81

Отримати прогноз

Рисунок 4.1 – Інтерфейс головної форми програмного забезпечення для прогнозування погодних характеристик

Після введення вхідних даних необхідно натиснути кнопку Отримати прогноз, внаслідок чого отримуються та обробляються дані з метеосервісу. Після цього результати прогнозування температури, вологості, швидкості вітру та інших характеристик відображаються у відповідній формі (рис. 4.2).

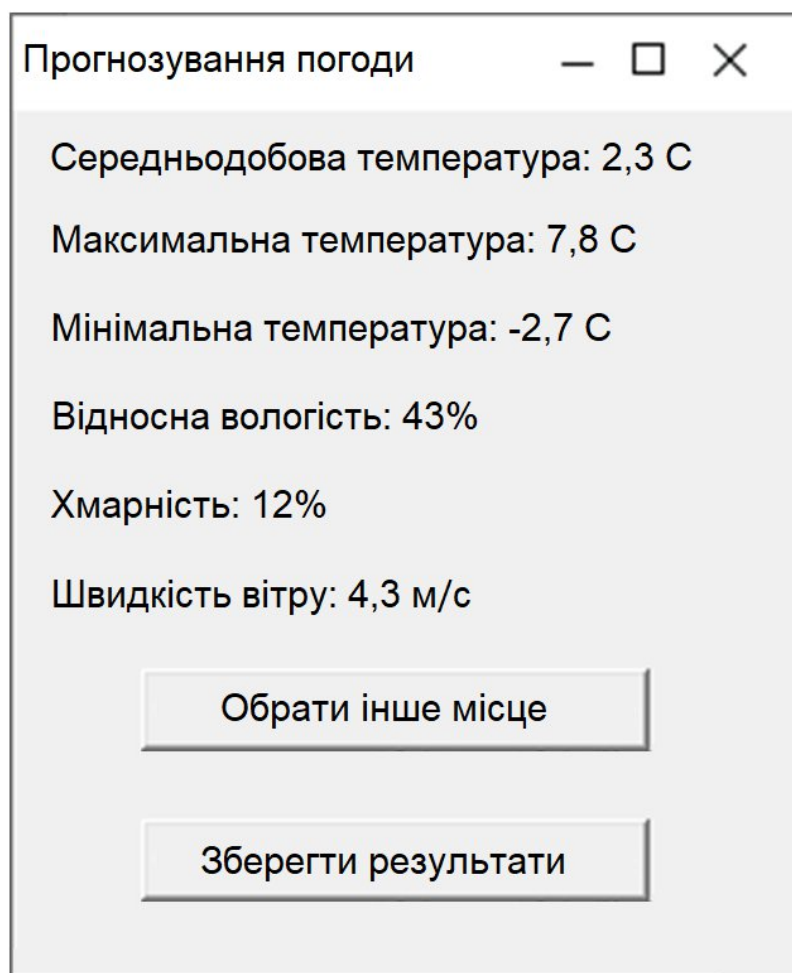


Рисунок 4.2 – Результати прогнозування погодних характеристик

Для прогнозування погодних характеристик для інших місць (повторного прогнозування при інших вхідних даних) користувачу необхідно натиснути кнопку **Обрати інше місце**. Для збереження результатів прогнозування необхідно натиснути відповідну кнопку.

За допомогою реалізованих програмних модулів також можна проводити експериментальне дослідження прогнозування погодних характеристик шляхом використання різних інтелектуальних моделей.

На рис. 4.3–4.9 наведено результати прогнозування температури за заданими наборами даних за допомогою згорткової нейронної мережі (рис. 4.3), глибокої нейромоделі Deep NN (рис. 4.4), моделі GRU (рис. 4.5), моделі LSTM (рис. 4.6), моделі LSTM та шарами Dropout (рис. 4.7), моделі GRU та шарами Dropout (рис. 4.8), моделі GRU та алгоритму SGD (рис. 4.9).

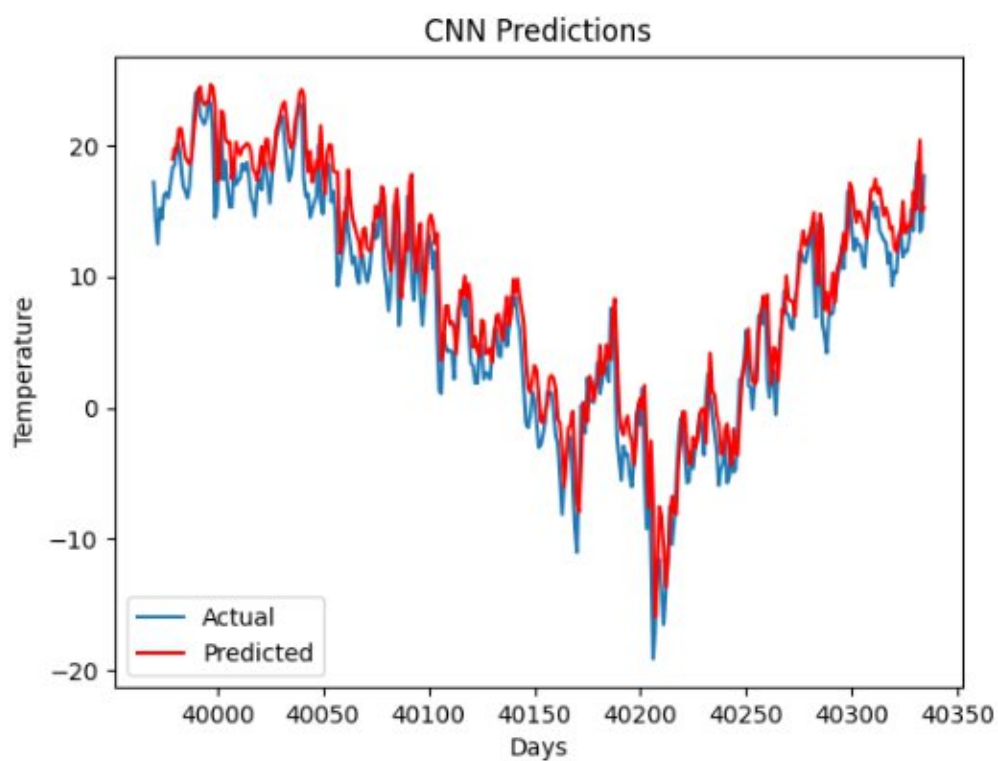


Рисунок 4.3 – Результати прогнозування температури за допомогою згорткової нейронної мережі

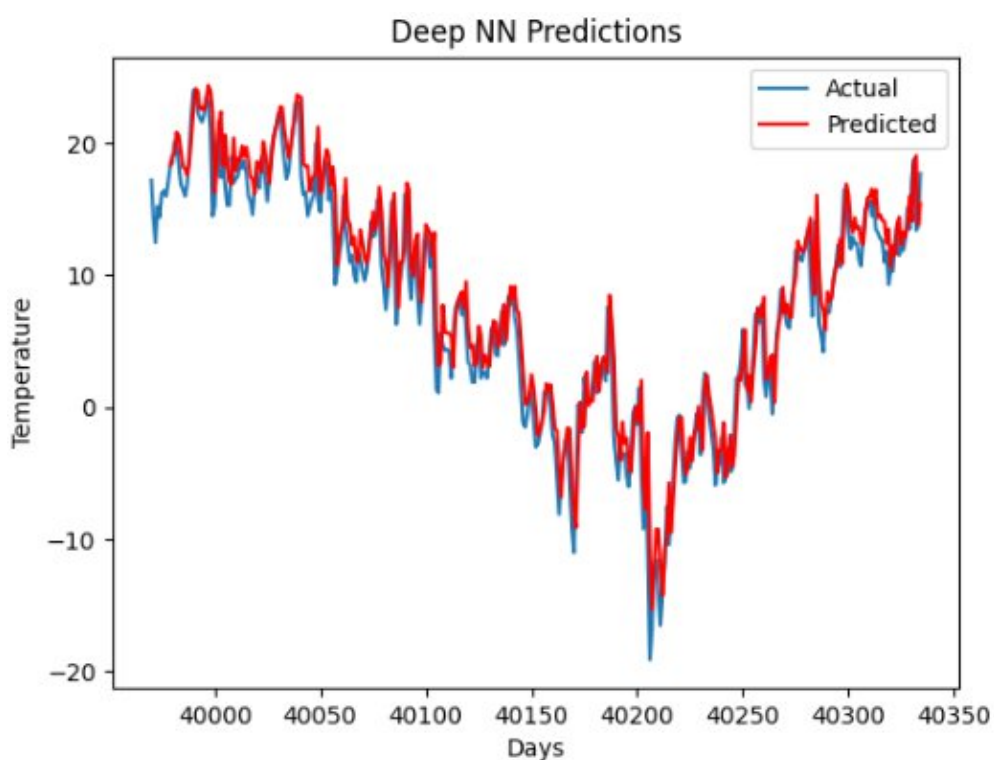


Рисунок 4.4 – Результати прогнозування температури за допомогою нейромоделі Deep NN

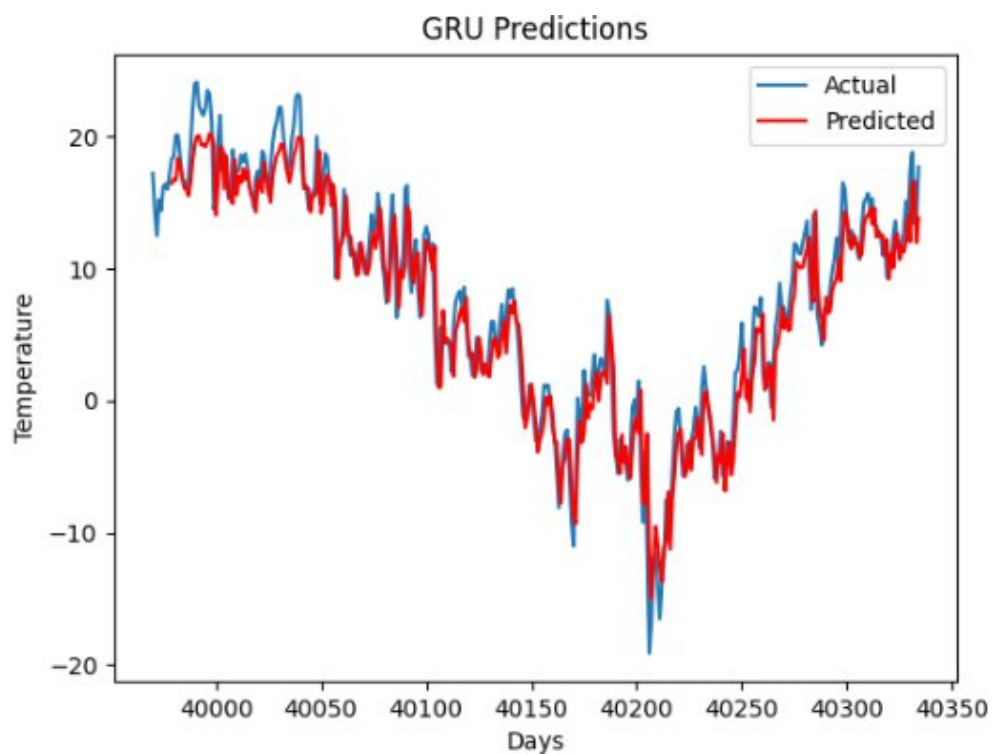


Рисунок 4.5 – Результати прогнозування температури за допомогою нейронної мережі GRU

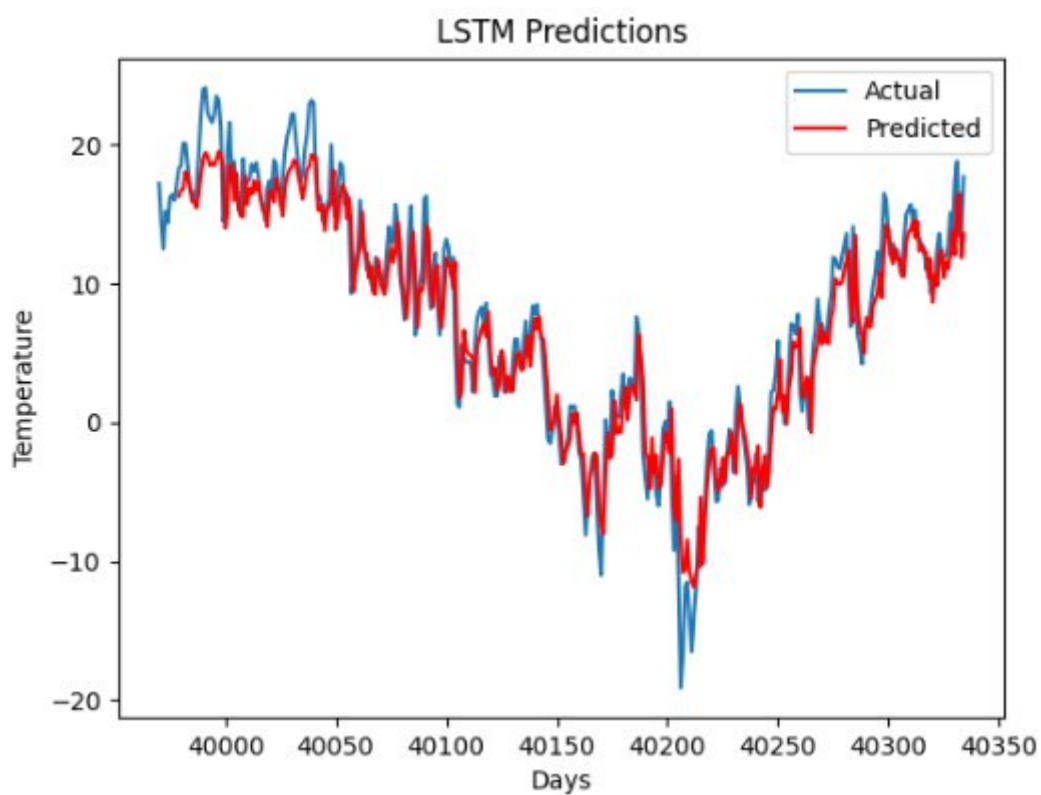


Рисунок 4.6 – Результати прогнозування температури за допомогою нейронної мережі LSTM

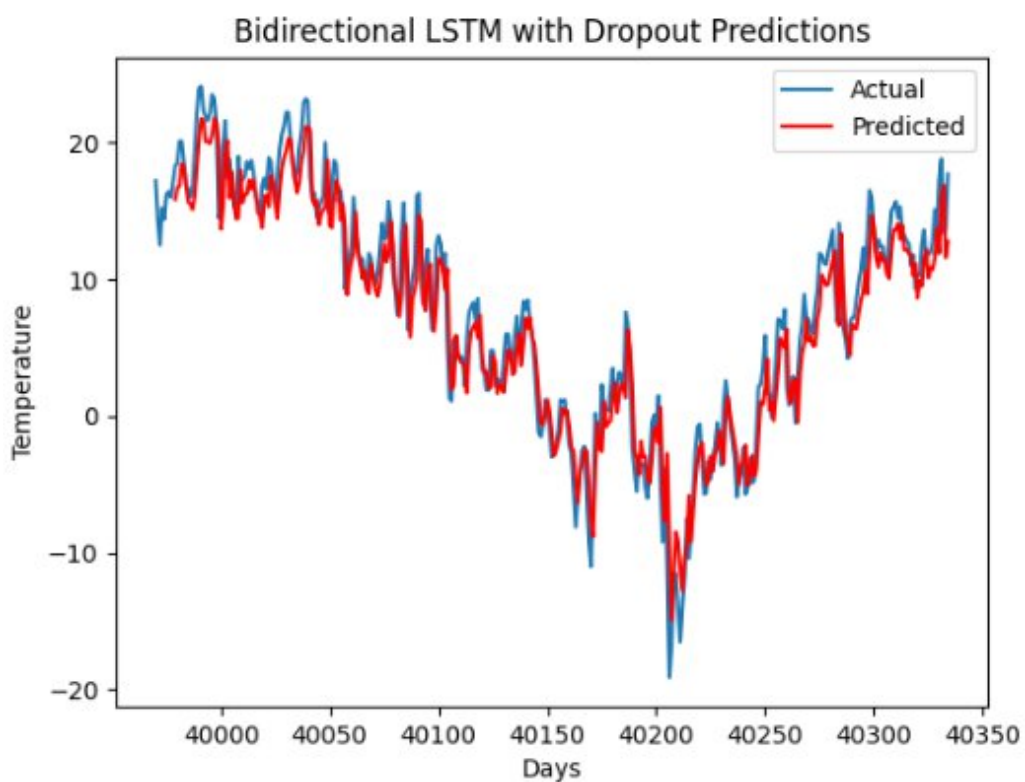


Рисунок 4.7 – Результати прогнозування температури за допомогою моделі LSTM та шарами Dropout

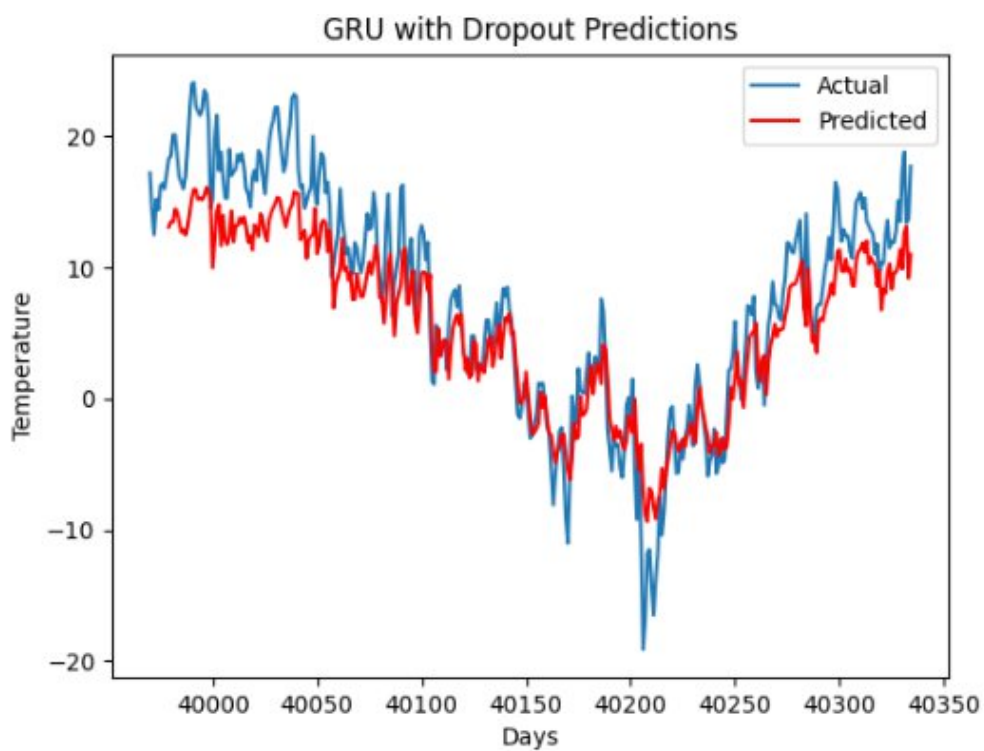


Рисунок 4.8 – Результати прогнозування температури за допомогою моделі GRU та шарами Dropout

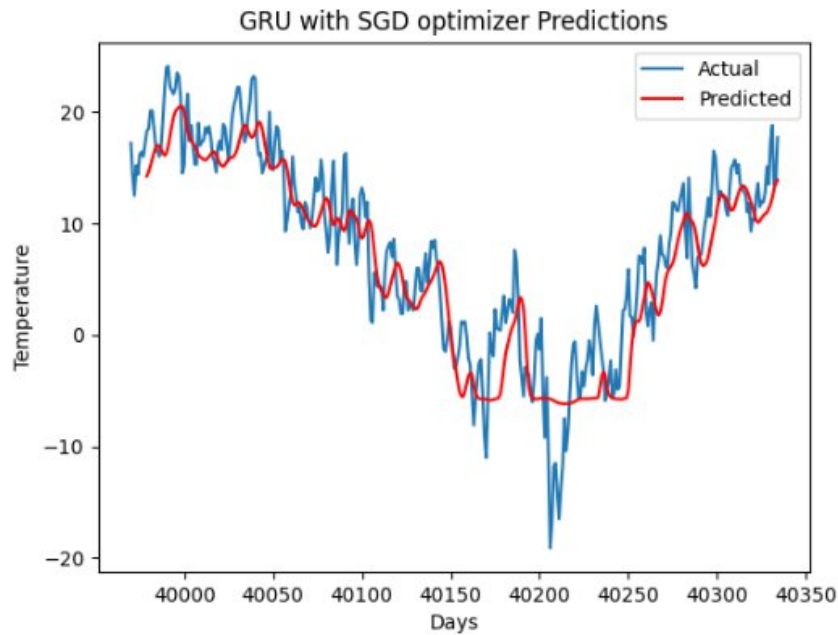


Рисунок 4.9 – Результати прогнозування температури за допомогою моделі GRU та алгоритму SGD

У таблиці 4.1 наведено порівняння ефективності використання нейромоделей для прогнозування температурних характеристик.

Таблиця 4.1 – Порівняння ефективності використання нейромоделей для прогнозування температурних характеристик

| Метод                                | Середньо-квадратичне відхилення | Час прогнозування (мс) | Швидкість навчання | Складність моделі | Використання ресурсів |
|--------------------------------------|---------------------------------|------------------------|--------------------|-------------------|-----------------------|
| Згорткова нейромережа (CNN)          | 8,62                            | 23,18                  | Середня            | Середня           | Середні               |
| Глибока нейромережа (Deep NN)        | 5,91                            | 19,93                  | Середня            | Висока            | Високі                |
| Нейромережа GRU                      | 5,28                            | 20,47                  | Середня            | Середня           | Середні               |
| Нейромодель LSTM                     | 5,71                            | 22,54                  | Середня            | Висока            | Високі                |
| Модель LSTM із шарами Dropout        | 5,67                            | 20,87                  | Середня            | Висока            | Середні               |
| Модель GRU із шарами Dropout         | 12,98                           | 11,08                  | Середня            | Середня           | Середні               |
| Модель GRU з алгоритмом навчання SGD | 13,47                           | 17,81                  | Середня            | Середня           | Середні               |

З таблиці 4.1 видно, що використання нейромоделей є досить ефективним для прогнозування погодних характеристик.

За результатами дослідження можна відзначити, що використання згорткових нейромереж виявляється доцільним у випадках, коли необхідно аналізувати просторові залежності, характерні для кліматичних карт чи супутникових знімків, проте висока ефективність таких моделей досягається лише за умови наявності великих обсягів даних для навчання. Рекурентні архітектури демонструють сильні сторони у роботі з часовими рядами, адже дозволяють враховувати послідовність показників, проте їхня здатність відтворювати довготривалі залежності є обмеженою.

Глибокі нейронні мережі створюють основу для моделювання складних взаємозв'язків між параметрами, що робить їх потужним інструментом для метеорологічних задач. Однак складність архітектури потребує значних обчислювальних потужностей і часу для навчання. Розширені варіанти, як-от нейромоделі LSTM із використанням шарів Dropout, дозволяють враховувати інформацію як із минулого, так і з майбутнього напрямків послідовності, що підвищує точність прогнозів, але вимагає ще більших ресурсів і має складнішу структуру. Використання регуляризації в глибоких мережах дає можливість уникнути перенавчання й підвищити ефективність навіть при роботі з обмеженими наборами даних, однак успіх такого підходу значною мірою залежить від коректності налаштування параметрів.

Ансамблеві підходи, які поєднують LSTM із глибокими мережами, що дає змогу досягти високої точності та компенсувати слабкі сторони окремих моделей, хоча це й вимагає значних обчислювальних витрат.

Таким чином, використання різних нейромоделей у сфері прогнозування погоди дає змогу знаходити баланс між точністю, швидкістю обчислень та вимогами до ресурсів. Кожна архітектура має власну сферу застосування, і вибір конкретного підходу залежить від характеру даних, доступних ресурсів і поставлених цілей у прогнозуванні.

#### **4.5 Тестування програмного забезпечення для прогнозування погодних характеристик**

Виконано тестування програмного забезпечення для прогнозування погодних характеристик.

Виявлено, що програма для прогнозування погодних характеристик функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги та успішно виконує свою основну задачу підтримки процесу організації харчування домашніх тварин.

Розроблене програмне забезпечення дозволяє забезпечувати автоматизацію процесів, пов'язаних з підтримкою організації харчування домашніх тварин.

#### **4.6 Висновки за розділом 4**

Описано програмне забезпечення для прогнозування погодних характеристик. Виконано тестування розробленого програмного забезпечення для прогнозування погодних характеристик. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою організації харчування домашніх тварин.

## ВИСНОВКИ

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для прогнозування погодних характеристик.

Визначено, що програмне забезпечення для прогнозування погодних характеристик являє собою інструмент, що забезпечує обчислення, обробку та представлення результатів аналізу атмосферних процесів у зручному для користувачів вигляді. Такі програмні системи ґрунтуються на обробці великих масивів даних, що збираються з різних джерел, і дозволяють створювати прогнози з різними часовими та просторовими масштабами.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для прогнозування погоди. Проте деякі програмні засоби для прогнозування погодних характеристик можуть використовувати неефективні методи та моделі, що призводить до низької точності прогнозів. Важливим недоліком є висока залежність якості прогнозів від обсягу й достовірності вхідних даних. Ще однією проблемою є невизначеність у прогнозах, яка часто ускладнює їхнє практичне застосування. Також варто зазначити, що програмне забезпечення, орієнтоване на глобальні моделі, не завжди забезпечує достатню точність для локальних територій. Таким чином, головними недоліками програмних систем прогнозування погоди виступають обмежена точність та локальна деталізація прогнозів, залежність від якості вхідних даних, висока ресурсомісткість та складність практичного використання результатів у різних сферах діяльності. Тому актуальною є розробка програмного забезпечення для прогнозування погодних характеристик.

Сформульовано функціональні вимоги до програмного забезпечення для прогнозування погодних характеристик.

Для реалізації програмного забезпечення для прогнозування погодних характеристик обрано мову програмування Python, яка вирізняється широким

набором бібліотек і фреймворків для роботи з даними, математичного моделювання та машинного навчання. Крім того, ця мова відзначається простотою синтаксису, що робить її доступною для дослідників і фахівців з метеорології, які не завжди мають глибокі знання у сфері програмної інженерії. Активна спільнота розробників сприяє швидкому обміну знаннями, появі нових інструментів та підтримці відкритих наукових проєктів.

Для створення програми для прогнозування погодних характеристик обрано середовище розробки PyCharm, що поєднує у собі інструменти для повноцінної розробки програмного забезпечення, зокрема, підтримку масштабних проєктів, інтеграцію з бібліотеками, засоби налагодження, управління версіями та ефективну організацію командної роботи.

Наукова новизна роботи полягає у тому, що створено модель програмного забезпечення для прогнозування погодних характеристик, яка подана у вигляді багаторівневої системи, у якій відбувається взаємодія між користувачем, застосунком, локальним сховищем та зовнішнім метеосервісом. Основна ідея такої моделі полягає в забезпеченні безперервного циклу обробки даних: від отримання початкової інформації до формування прогнозних результатів, що подаються користувачеві у зручній та зрозумілій формі. Поєднання Інтернет сервісів, локального сховища та розвиненого функціоналу застосунку дозволяє реалізувати ефективне середовище для досліджень, практичних розробок та прикладного використання у сфері метеорології.

Практична цінність полягає у тому, що розроблено програмне забезпечення для прогнозування погодних характеристик.

Запропоновано структуру програмного забезпечення для прогнозування погодних характеристик. Визначено, що кожен структурний компонент виконує взаємопов'язану роль, забезпечуючи послідовний перехід від введення вихідних даних до представлення кінцевого результату у зручному для користувача вигляді. Основними модулями програмного забезпечення для прогнозування погодних характеристик є такі: модуль

інтеграції з зовнішніми джерелами, модуль аналітичної обробки, модуль шкалювання, модуль обчислення похідних параметрів, модуль прогнозування, модуль візуалізації.

Описано функціонування програмного забезпечення для прогнозування погодних характеристик. Визначено, що функціонування програми відбувається у вигляді послідовного алгоритму, який поєднує завантаження моделей, роботу з користувачем, обробку метеорологічних даних і формування кінцевого прогнозу.

Описано особливості реалізації програмного забезпечення для прогнозування погодних характеристик.

Виконано проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для прогнозування погодних характеристик.

Виконано тестування розробленого програмного забезпечення для прогнозування погодних характеристик. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє забезпечити автоматизацію процесів, пов'язаних з підтримкою організації харчування домашніх тварин.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. What is a weather forecast and how does it work. Meaning, duration, accuracy, more [Electronic resource]. – Access mode: <https://windy.app/blog/what-is-a-weather-forecast-and-how-it-works.html>.
2. Introduction to weather forecasting [Electronic resource]. – Access mode: <https://www.encyclopedie-environnement.org/en/air-en/introduction-weather-forecasting-2/>.
3. The Science and Evolution of Weather Forecasting [Electronic resource]. – Access mode: <https://blog.worldweatheronline.com/weather-api/the-science-and-evolution-of-weather-forecasting/>.
4. Meteorology: How Weather Forecasting Works [Electronic resource]. – Access mode: <https://www.sciencenewstoday.org/meteorology-how-weather-forecasting-works>.
5. How Does Weather Forecasting Work [Electronic resource]. – Access mode: <https://www.scienceabc.com/innovation/how-does-weather-forecasting-work.html> /.
6. How a Weather Forecast is Made [Electronic resource]. – Access mode: <https://www.pagasa.dost.gov.ph/learning-tools/how-weather-forecast-made>.
7. The Best Weather Station Software [Electronic resource]. – Access mode: <https://www.theraingauge.com/weather-station-software/>.
8. Best Free Weather Software For Windows [Electronic resource]. – Access mode: <https://listoffreeware.com/best-free-weather-software/>.
9. Best Weather Intelligence Software [Electronic resource]. – Access mode: <https://sourceforge.net/software/weather-intelligence/>.
10. The 7 Best Software for Weather Forecasting [Electronic resource]. – Access mode: <https://www.saidhasyim.com/best-product/software/weather-forecasting-tools/>.
11. WeatherBug [Electronic resource]. – Access mode: <https://www.weatherbug.com/>.

12. Ambient Weather Network [Electronic resource]. – Access mode: <https://ambientweather.com/ambient-weather-network>.

13. Weather Display [Electronic resource]. – Access mode: <https://www.weather-display.com/>.

14. Python with Statement [Electronic resource]. – Access mode: <https://realpython.com/python-with-statement/>.

15. Learn Python with Tutorials, Exercises, and Quizzes [Electronic resource]. – Access mode: <https://pynative.com/>.

16. PyCharm. Feel the Difference in Data Science and AI/ML Workflows [Electronic resource]. – Access mode: <https://www.jetbrains.com/pycharm/data-science/>.

17. JetBrains PyCharm [Electronic resource]. – Access mode: <https://www.techspot.com/downloads/7705-pycharm.html>.

**ДОДАТОК А**  
**Технічне завдання**

## **Вступ**

Програмне забезпечення може використовуватися для прогнозування погодних характеристик.

### **А.1 Підстава для розробки**

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація засобів для прогнозування погодних характеристик», затверджене наказом Національного університету «Запорізька політехніка» № 447 від 30 вересня 2025 р.

### **А.2 Призначення розробки**

Програмний продукт призначений для прогнозування погодних характеристик.

### **А.3 Основні вимоги до програми, що розробляється**

#### **А.3.1 Вимоги до функціональних характеристик**

Програмне забезпечення для прогнозування погодних характеристик забезпечує виконання таких функцій:

- підтримка можливості введення користувачем вхідних даних (широти та довготи) для подальшого прогнозування погодних характеристик;
- підтримка можливості надсилання користувачем запиту до метеосервісу для отримання погодних даних;
- можливість обробки даних та прогнозування погодних характеристик;

- можливість виведення в екранну форму результатів щодо прогнозування температури, вологості, швидкості вітру та інших характеристик;
- підтримка можливості прогнозування погодних характеристик для інших місць (повторного прогнозування при інших вхідних даних);
- підтримка можливості збереження результатів прогнозування.

### **A.3.2 Вимоги до інтерфейсу програми**

Інтерфейс програмного забезпечення для прогнозування погодних характеристик повинен бути зручним для користувачів.

### **A.3.3 Вимоги до надійності**

Програмне забезпечення для прогнозування погодних характеристик повинно забезпечити надійне функціонування.

### **A.3.4 Умови експлуатації**

Для експлуатації програмного забезпечення для прогнозування погодних характеристик необхідна наявність персонального комп'ютера.

### **A.3.5 Вимоги до складу та параметрів технічний засобів**

Для експлуатації програмного забезпечення для прогнозування погодних характеристик необхідно таке апаратне та програмне забезпечення:

- процесор із тактовою частотою не нижче 2,0–2,5 ГГц;
- оперативна пам'ять обсягом не менше 8 ГБ;
- жорсткий диск з вільним місцем не менше 20 ГБ;

- стабільне Інтернет з'єднання для завантаження метеорологічних даних з відкритих джерел чи API;
- операційна система, середовище розробки PyCharm, встановлений інтерпретатор Python 3.8 або вище, бібліотеки Python для роботи з даними та прогнозування: NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow.

### **A.3.6 Вимоги до маркування і пакування**

Програма для прогнозування погодних характеристик може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення для прогнозування погодних характеристик».

### **A.4 Вхідні дані до роботи**

Вхідними даними до програмного забезпечення для прогнозування погодних характеристик є інформація про координати місця для прогнозування погоди (широта та довгота) та ін.

Вихідними даними програмного забезпечення для прогнозування погодних характеристик є прогнозовані значення погоди, зокрема, температури, вологості, швидкості вітру та інші характеристики.

**ДОДАТОК Б**  
**Фрагмент тексту програми**

```

tmptr_avg = []
tmptr_max = []
tmptr_min = []
hmdt = []
drw = []
prcptn = []
snig = []
cldcvr = []
wnd = []
prcptn_akrc = 0

```

```

tmptr_avgscld = []
tmptr_maxscld = []
tmptr_minscld = []
hmdtscld = []
drwscld = []
prcptnscld = []
snigscld = []
cldcvrscld = []
wndscld = []

```

```

tmptr_avg_prd = 0
tmptr_max_prd = 0
tmptr_min_prd = 0
hmdt_prd = 0
drw_prd = 0
prcptn_prd = 0
snig_prd = 0
cldcvr_prd = 0
wnd_prd = 0

```

```

def dany_preparation(dany):
    krnt_dtf = dtftime.now()
    ndys_ago = krnt_dtf - timedelta(days=wndow_sz - 1)
    frmt_dtf = ndys_ago.strftime("%Y-%m-%d") + 'T00:00'
    dany = dany['hourly']
    strt_indx = dany['time'].indx(frmt_dtf)
    for indx in range(wndow_sz):
        tmptrden = dany['tmptr_2m'][strt_indx + indx *
24:strt_indx + (indx + 1) * 24]
        tmptr_avg.append(sum(tmptrden) / len(tmptrden))
        tmptr_min.append(min(tmptrden))
        tmptr_max.append(max(tmptrden))

        hmdtden = dany['relative_hmdt_2m'][strt_indx + indx *
24:strt_indx + (indx + 1) * 24]

```

```

        hmdt.append(sum(hmdtden) / len(hmdtden) / 100)

        drwden = dany['drw_point_2m'][strt_indx + indx *
24:strt_indx + (indx + 1) * 24]
        drw.append(sum(drwden) / len(drwden))

        prcptnden = dany['prcptn'][strt_indx + indx *
24:strt_indx + (indx + 1) * 24]
        prcptn.append(sum(prcptnden))

        snigden = dany['snig_depth'][strt_indx + indx *
24:strt_indx + (indx + 1) * 24]
        snig.append(sum(snigden) / len(snigden))

        cldcvrden = dany['cloud_cover'][strt_indx + indx *
24:strt_indx + (indx + 1) * 24]
        cldcvr.append(sum(cldcvrden) / len(cldcvrden) / 100)

        wndden = dany['wnd_speed_10m'][strt_indx + indx *
24:strt_indx + (indx + 1) * 24]
        wnd.append(sum(wndden) / len(wndden))

    akrc = 0
    for indx in range(int(len(prcptn))):
        akrc = akrc + prcptn[indx]
        prcptn[indx] = akrc
    prcptn_akrc = akrc

def scale_dany():
    global tmptr_avgscld
    global tmptr_maxscld
    global tmptr_minscld
    global hmdtscl
    global drwscld
    global prcptnscld
    global snigscld
    global cldcvrscld
    global wndscld

    tmptr_avgscld = tmptr_sclr.transform([tmptr_avg])
    tmptr_maxscld = tmptr_sclr.transform([tmptr_max])
    tmptr_minscld = tmptr_sclr.transform([tmptr_min])
    hmdtscl = tmptr_sclr.transform([hmdt]).reshape(-1, wndow_sz)
    drwscld = drw_sclr.transform([drw])
    prcptnscld = prcptn_sclr.transform([prcptn])
    snigscld = snig_sclr.transform([snig])
    cldcvrscld = cldcvr_sclr.transform([cldcvr])
    wndscld = wnd_sclr.transform([wnd])

```

```

def prdict():
    global tmptr_avg_prd
    global tmptr_max_prd
    global tmptr_min_prd
    global hmdt_prd
    global drw_prd
    global prcptn_prd
    global snig_prd
    global cldcvr_prd
    global wnd_prd

    prd1 = tmptr_md11.prdict(tmptr_avgscld).reshape(-1, wndow_sz)
    prd2 = tmptr_md12.prdict(tmptr_avgscld)[0, 0]
    prd = prd1 * 0.2 + prd2 * 0.8
    tmptr_avg_prd = tmptr_sclr.inverse_transform(prd)[0, wndow_sz
- 1]

    prd1 = tmptr_md11.prdict(tmptr_minscld).reshape(-1, wndow_sz)
    prd2 = tmptr_md12.prdict(tmptr_minscld)[0, 0]
    prd = prd1 * 0.2 + prd2 * 0.8
    tmptr_min_prd = tmptr_sclr.inverse_transform(prd)[0, wndow_sz
- 1]

    prd1 = tmptr_md11.prdict(tmptr_maxscld).reshape(-1, wndow_sz)
    prd2 = tmptr_md12.prdict(tmptr_maxscld)[0, 0]
    prd = prd1 * 0.2 + prd2 * 0.8
    tmptr_max_prd = tmptr_sclr.inverse_transform(prd)[0, wndow_sz
- 1]

    prd = hmdt_md1.prdict(hmdtsclد).reshape(-1, wndow_sz)
    prd = prd.reshape(-1, wndow_sz)
    hmdt_prd = hmdt_sclr.inverse_transform(prd)[0, wndow_sz - 1]

    prd = drw_md1.prdict(drwscld).reshape(-1, wndow_sz)
    prd = prd.reshape(-1, wndow_sz)
    drw_prd = drw_sclr.inverse_transform(prd)[0, wndow_sz - 1]

    prd = prcptn_md1.prdict(prcptnscld).reshape(-1, wndow_sz)
    prd = prd.reshape(-1, wndow_sz)
    prcptn_prd = prcptn_sclr.inverse_transform(prd)[0, wndow_sz -
1] - prcptn_akrc
    prcptn_prd = max(0, prcptn_prd)

    prd = snig_md1.prdict(snigscld).reshape(-1, wndow_sz)
    prd = prd.reshape(-1, wndow_sz)
    snig_prd = snig_sclr.inverse_transform(prd)[0, wndow_sz - 1]

```

```

    prd = cldcvr_mdl.predict(cldcvrsclد)
    prd = prd.reshape(-1, wndow_sz)
    cldcvr_prd = cldcvr_sclr.inverse_transform(prd)[0, wndow_sz -
1]

    prd = wnd_mdl.predict(wndscld).reshape(-1, wndow_sz)
    prd = prd.reshape(-1, wndow_sz)
    wnd_prd = wnd_sclr.inverse_transform(prd)[0, wndow_sz - 1]

def on_click():
    try:
        lttd = float(entry_lat.get())
        longtdd = float(entry_long.get())
        respns = get_meteo_dany(lttd, longtdd)
    except ValueError:
        lbler.config(text="Incorrect data")
        return

    if lttd > 180 or lttd < -180 or longtdd > 180 or longtdd < -
180:
        lbler.config(text="Incorrect data")
        return

    if respns.status_code == 200:
        dany = respns.json()
        dany_preparation(dany)
        lbler.config(text="")
    else:
        lbler.config(text=f"Error: {respns.status_code},
{respns.text}")
        return

    scale_dany()
    prdict()

df = pd.read_csv(file_path)

df = df[klmnm][30207:] / 100.0
dfdanny = []
for i in range(0, len(df), 24):
    wndow = df[i:i + wndow_sz]
    seredn = sum(wndow) / len(wndow)
    dfdanny.append(seredn)
df = dfdanny

tstindx = round(len(df) * 0.8)
tstsz = len(df) - tstindx

```

```

trndany = df[0:tstindx]

Xtrn = []
ytrn = []

for group_indx in range(int(len(trndany)-wndow_sz)):
    Xtrn.append(trndany[group_indx:group_indx+wndow_sz])
    ytrn.append([trndany[group_indx+wndow_sz]])

sclr_X = MinMaxSclr()
sclr_y = MinMaxSclr()

Xtrn = sclr_X.fittrnsfrm(np.array(Xtrn)).reshape(-1, wndow_sz)
ytrn = sclr_y.fittrnsfrm(np.array(ytrn).reshape(-1, 1))

joblib.dump(sclr_X, '..\\sclrs\\cldcvr.pkl')

if not os.path.exists mdl_path):
    mdl = Sequential()
    mdl.add(GRU(units=16, activation='relu',
return_sequences=True, input_shape=(wndow_sz, 1)))
    mdl.add(GRU(8, activation='relu', return_sequences=True))
    mdl.add(GRU(4, activation='relu', return_sequences=True))
    mdl.add(Dense(1, activation='sigmoid'))
    mdl.compile(optimizer='adam', loss='MSE')

    mdl.fit(Xtrn, ytrn, epochs=40, batch_sz=3, verbose=1)
    mdl.save(mdl_path)
else:
    mdl = keras.models.load_model(mdl_path)

def nrlntwrkrolling_predictions(dany):
    predictions = []

    for strt in range(len(dany)-wndow_sz):
        d = sclr_X.transform(np.array([dany[strt:strt +
wndow_sz]])).reshape(-1, wndow_sz)
        prd = mdl.predict(d)
        prd = sclr_y.inverse_transform([[prd[0,wndow_sz-
1,0]]])[0, 0]
        predictions.append(prd)
    return predictions

frcst_stps = tstsz-wndow_sz

predictions =
nrlntwrkrolling_predictions(df[tstindx:tstindx+tstsz])
actlvls = df[tstindx:tstindx+tstsz][-frcst_stps:]

```

```

mse = MSE(actlvls, prdictions)
print(f"Середньоквадратичне відхилення(MSE): {mse}")
i = df[tstindx:tstindx+tstsz].indx
plt.plot(range(tstindx, tstindx+tstsz),
df[tstindx:tstindx+tstsz], label='Act1')
plt.plot(range(tstindx+wndow_sz, tstindx+tstsz), prdictions,
label='Prdicted', color='red')
plt.xlabel('Dtf')
plt.ylabel('Cldcvr')
plt.legend()
plt.show()

df = pd.read_csv(file_path)

df = df[klmnm][30207:]
dfdanny = []
for i in range(0, len(df), 24):
    wndow = df[i:i + wndow_sz]
    seredn = sum(wndow) / len(wndow)
    dfdanny.append(seredn)
df = dfdanny

tstindx = round(len(df) * 0.8)
tstsz = len(df) - tstindx

trndany = df[0:tstindx]

Xtrn = []
ytrn = []

for group_indx in range(int(len(trndany)-wndow_sz)):
    Xtrn.append(trndany[group_indx:group_indx+wndow_sz])
    ytrn.append([trndany[group_indx+wndow_sz]])

sclr_X = MinMaxSclr()
sclr_y = MinMaxSclr()

Xtrn = sclr_X.fittrnsfrm(np.array(Xtrn)).reshape(-1, wndow_sz)
ytrn = sclr_y.fittrnsfrm(np.array(ytrn).reshape(-1, 1))

joblib.dump(sclr_X, '..\\sclrs\\drw.pkl')

if not os.path.exists mdl_path):
    mdl = Sequential()
    mdl.add(GRU(units=26, activation='relu',
return_sequences=True, input_shape=(wndow_sz, 1)))
    mdl.add(GRU(16, activation='relu', return_sequences=True))

```

```

mdl.add(GRU(8, activation='relu', return_sequences=True))
mdl.add(GRU(5, activation='relu', return_sequences=True))
mdl.add(Dense(1, activation='linear'))
mdl.compile(optimizer='adam', loss='MSE')

mdl.fit(Xtrn, ytrn, epochs=80, batch_sz=3, verbose=1)
mdl.save(mdl_path)
else:
    mdl = keras.models.load_model(mdl_path)

def nrlntwrkrolling_predictions(dany):
    predictions = []

    for strt in range(len(dany)-window_sz):
        d = sclr_X.transform(np.array([dany[strt:strt +
window_sz]])).reshape(-1, window_sz)
        prd = mdl.predict(d)
        prd = sclr_y.inverse_transform([[prd[0,window_sz-
1,0]]])[0, 0]
        predictions.append(prd)
    return predictions

frcst_stps = tstsz-window_sz

predictions =
nrlntwrkrolling_predictions(df[tstindx:tstindx+tstsz])
actlvls = df[tstindx:tstindx+tstsz][-frcst_stps:]
mse = MSE(actlvls, predictions)
print(f"Середньоквадратичне відхилення(MSE): {mse}")
i = df[tstindx:tstindx+tstsz].index
plt.plot(range(tstindx, tstindx+tstsz),
df[tstindx:tstindx+tstsz], label='Act1')
plt.plot(range(tstindx+window_sz, tstindx+tstsz), predictions,
label='Prdicted', color='red')
plt.xlabel('Dtf')
plt.ylabel('Drw Point')
plt.legend()
plt.show()

tstindx = round(len(df) * 0.8)
tstsz = len(df) - tstindx

trndany = df[0:tstindx]

Xtrn = []
ytrn = []

```

```

for group_indx in range(int(len(trndany)-window_sz)):
    Xtrn.append(trndany[group_indx:group_indx+window_sz])
    ytrn.append([trndany[group_indx+window_sz]])

sclr_X = MinMaxSclr()
sclr_y = MinMaxSclr()

Xtrn = sclr_X.fittrnsfrm(np.array(Xtrn)).reshape(-1, window_sz)
ytrn = sclr_y.fittrnsfrm(np.array(ytrn).reshape(-1, 1))

joblib.dump(sclr_X, '..\\sclrs\\hmdt.pkl')

if not os.path.exists mdl_path):
    mdl = Sequential()
    mdl.add(GRU(units=16, activation='relu',
return_sequences=True, input_shape=(window_sz, 1)))
    mdl.add(GRU(8, activation='relu', return_sequences=True))
    mdl.add(GRU(4, activation='relu', return_sequences=True))
    mdl.add(Dense(1, activation='sigmoid'))
    mdl.compile(optimizer='adam', loss='MSE')

    mdl.fit(Xtrn, ytrn, epochs=40, batch_sz=3, verbose=1)
    mdl.save(mdl_path)
else:
    mdl = keras.models.load_model(mdl_path)

def nrlntwrkrolling_predictions(dany):
    predictions = []

    for strt in range(len(dany)-window_sz):
        d = sclr_X.transform(np.array([dany[strt:strt +
window_sz]])).reshape(-1, window_sz)
        prd = mdl.predict(d)
        prd = sclr_y.inverse_transform([[prd[0,window_sz-
1,0]]])[0, 0]
        predictions.append(prd)
    return predictions

frcst_stps = tstsz-window_sz

predictions =
nrlntwrkrolling_predictions(df[tstindx:tstindx+tstsz])
actlvls = df[tstindx:tstindx+tstsz][-frcst_stps:]
mse = MSE(actlvls, predictions)
print(f"Середньоквадратичне відхилення (MSE): {mse}")
i = df[tstindx:tstindx+tstsz].index
plt.plot(range(tstindx, tstindx+tstsz),
df[tstindx:tstindx+tstsz], label='Act1')

```

```

plt.plot(range(tstindx+wndow_sz, tstindx+tstsz), prdictions,
label='Prdicted', color='red')
plt.xlabel('Dtf')
plt.ylabel('Hmdt')
plt.legend()
plt.show()

for group_indx in range(int(len(trndany)-wndow_sz)):
    set = trndany[group_indx:group_indx + wndow_sz].copy()
    akrc = 0
    for ind in range(len(set)):
        akrc = akrc + set[ind]
        set[ind] = akrc
    Xtrn.append(set)
    ytrn.append([trndany[group_indx+wndow_sz]+akrc])

sclr_X = MinMaxSclr()
sclr_y = MinMaxSclr()

Xtrn = sclr_X.fittrnsfrm(np.array(Xtrn)).reshape(-1, wndow_sz)
ytrn = sclr_y.fittrnsfrm(np.array(ytrn).reshape(-1, 1))

if not os.path.exists mdl_path):
    mdl = Sequential()
    mdl.add(GRU(units=50, activation='relu',
return_sequences=True, input_shape=(wndow_sz, 1)))
    mdl.add(GRU(units=25, activation='relu',
return_sequences=True))
    mdl.add(GRU(16, activation='relu', return_sequences=True))
    mdl.add(GRU(8, activation='relu', return_sequences=True))
    mdl.add(GRU(5, activation='relu', return_sequences=True))
    mdl.add(Dense(1, activation='linear'))
    mdl.compile(optimizer='adam', loss='MSE')

    mdl.fit(Xtrn, ytrn, epochs=20, batch_sz=10, verbose=1)
    mdl.save(mdl_path)
else:
    mdl = keras.mdls.load_mdl(mdl_path)

tstsz = 365
def nrlntwrkrolling_prdictions(dany):
    prdictions = []

    for strt in range(len(dany)-wndow_sz):
        set = dany[klmnm][strt:strt + wndow_sz].values.copy()
        set_sum = sum(dany[klmnm][strt:strt + wndow_sz].values)
        akrc = 0

```

```

        for ind in range(len(set)):
            akrc = akrc + set[ind]
            set[ind] = akrc
        d = sclr_X.transform(np.array([set])).reshape(-1,
window_sz)
        prd = mdl.predict(d)
        prd = sclr_y.inverse_transform([[prd[0,window_sz-
1,0]]])[0, 0] - set_sum
        if prd < 0:
            prd = 0
        prdictions.append(prd)
    return prdictions

frcst_stps = tstsz-window_sz

prdictions =
nrlntwrkrolling_prdictions(df_filled[tstindx:tstindx+tstsz])
actlvls = df_filled[tstindx:tstindx+tstsz][klmnm].iloc[-
frcst_stps:]
mse = MSE(actlvls, prdictions)
print(f"Середньоквадратичне відхилення(MSE): {mse}")

plt.plot(df_filled[tstindx:tstindx+tstsz].indx,
df_filled[tstindx:tstindx+tstsz][klmnm], label='Act1')
plt.plot(actlvls.indx, prdictions, label='Prdicted', color='red')
plt.xlabel('Dtf')
plt.ylabel('Prcptn')
plt.legend()
plt.show()

for group_indx in range(int(len(trndany)-window_sz)):
    Xtrn.append(trndany[group_indx:group_indx+window_sz])
    ytrn.append([trndany[group_indx+window_sz]])

sclr_X = MinMaxSclr()
sclr_y = MinMaxSclr()

Xtrn = sclr_X.fittrnsfrm(np.array(Xtrn)).reshape(-1, window_sz)
ytrn = sclr_y.fittrnsfrm(np.array(ytrn).reshape(-1, 1))

joblib.dump(sclr_X, '..\\sclrs\\snig.pkl')

if not os.path.exists(mdl_path):
    mdl = Sequential()
    mdl.add(Dense(units=50, activation='relu',
input_shape=(window_sz, 1)))
    mdl.add(Dense(units=25, activation='relu'))

```

```

mdl.add(Dense(units=20, activation='relu'))
mdl.add(Dense(16, activation='relu'))
mdl.add(Dense(16, activation='relu'))
mdl.add(Dense(16, activation='relu'))
mdl.add(Dense(8, activation='relu'))
mdl.add(Dense(8, activation='relu'))
mdl.add(Dense(8, activation='relu'))
mdl.add(Dense(5, activation='relu'))
mdl.add(Dense(1, activation='linear'))
mdl.compile(optimizer='adam', loss='MSE')

mdl.fit(Xtrn, ytrn, epochs=100, batch_sz=10, verbose=1)
mdl.save(mdl_path)
else:
    mdl = keras.models.load_model(mdl_path)

def nrlntwrkrolling_predictions(dany):
    predictions = []

    for strt in range(len(dany)-window_sz):
        d = sclr_X.transform(np.array([dany[klmnm][strt:str +
window_sz].values])).reshape(-1, window_sz)
        prd = mdl.predict(d)
        prd = sclr_y.inverse_transform([[prd[0,window_sz-
1,0]]])[0, 0]
        predictions.append(prd)
    return predictions

frcst_stps = tstsz-window_sz

predictions =
nrlntwrkrolling_predictions(df_filled[tstindx:tstindx+tstsz])
actlvls = df_filled[tstindx:tstindx+tstsz][klmnm].iloc[-
frcst_stps:]
mse = MSE(actlvls, predictions)
print(f"Середньоквадратичне відхилення(MSE): {mse}")

plt.plot(df_filled[tstindx:tstindx+tstsz].indx,
df_filled[tstindx:tstindx+tstsz][klmnm], label='Act1')
plt.plot(actlvls.indx, predictions, label='Prdicted', color='red')
plt.xlabel('Dt f')
plt.ylabel('Snig Depth')
plt.legend()
plt.show()

df = pd.read_csv(file_path)

```

```

df = df[30207:]
df = df[9::24][klmnm]
tstindx = round(df.values.shape[0] * 0.8)
tstsz = df.values.shape[0] - tstindx

trndany = df[0:tstindx].values

Xtrn = []
ytrn = []

for group_indx in range(int(len(trndany)-wndow_sz)):
    Xtrn.append(trndany[group_indx:group_indx+wndow_sz])
    ytrn.append([trndany[group_indx+wndow_sz]])

sclr_X = MinMaxScaler()
sclr_y = MinMaxScaler()

Xtrn = sclr_X.fittrnsfrm(np.array(Xtrn)).reshape(-1, wndow_sz)
ytrn = sclr_y.fittrnsfrm(np.array(ytrn).reshape(-1, 1))

joblib.dump(sclr_X, '..\\sclrs\\wnd.pkl')

if not os.path.exists mdl_path):
    mdl = Sequential()
    mdl.add(GRU(units=20, activation='relu',
return_sequences=True, input_shape=(wndow_sz, 1)))
    mdl.add(GRU(16, activation='relu', return_sequences=True))
    mdl.add(GRU(16, activation='relu', return_sequences=True))
    mdl.add(GRU(16, activation='relu', return_sequences=True))
    mdl.add(GRU(16, activation='relu', return_sequences=True))
    mdl.add(GRU(8, activation='relu', return_sequences=True))
    mdl.add(GRU(8, activation='relu', return_sequences=True))
    mdl.add(Dense(1, activation='sigmoid'))
    mdl.compile(optimizer='adam', loss='MSE')

    mdl.fit(Xtrn, ytrn, epochs=100, batch_sz=5, verbose=1)
    mdl.save(mdl_path)
else:
    mdl = keras.mdls.load_md1(mdl_path)

def nrlntwrkrolling_prdictions(dany):
    prdictions = []

    for strt in range(len(dany)-wndow_sz):
        d = sclr_X.transform(np.array([dany[strt:strt +
wndow_sz].values])).reshape(-1, wndow_sz)
        prd = mdl.prdict(d)

```

```

        prd = sclr_y.inverse_transform([[prd[0,window_sz-
1,0]]])[0, 0]
        prdictions.append(prd)
    return prdictions

frcst_stps = tstsz-window_sz

prdictions =
nrlntwrkrolling_prdictions(df[tstindx:tstindx+tstsz])
actlvls = df[tstindx:tstindx+tstsz].iloc[-frcst_stps:]
mse = MSE(actlvls, prdictions)
print(f"Середньоквадратичне відхилення(MSE): {mse}")

plt.plot(df[tstindx:tstindx+tstsz].indx,
df[tstindx:tstindx+tstsz], label='Act1')
plt.plot(actlvls.indx, prdictions, label='Prdicted', color='red')
plt.xlabel('Dtf')
plt.ylabel('Wnd Speed')
plt.legend()
plt.show()

tstindx = round(df_filled.values.shape[0] * 0.8)

trndany = df_filled['TAVG'][0:tstindx].values

Xtrn = []
ytrn = []

for group_indx in range(int(len(trndany)-window_sz)):
    Xtrn.append(trndany[group_indx:group_indx+window_sz])
    ytrn.append([trndany[group_indx+window_sz]])

sclr_X = MinMaxSclr()
sclr_y = MinMaxSclr()

Xtrn = sclr_X.fittrnsfrm(np.array(Xtrn)).reshape(-1, window_sz)
ytrn = sclr_y.fittrnsfrm(np.array(ytrn).reshape(-1, 1))

if not os.path.exists mdl_path):
    mdl = Sequential()
    mdl.add(Conv1D(32, 2, activation='relu',
input_shape=(window_sz, 1)))
    mdl.add(MaxPooling1D(2))
    mdl.add(Flatten())
    mdl.add(Dense(32, activation='relu'))
    mdl.add(Dense(1))
    mdl.compile(optimizer='adam', loss='MSE')

```

```

        mdl.fit(Xtrn, ytrn, epochs=6, batch_sz=10, verbose=1)
        mdl.save(mdl_path)
    else:
        mdl = keras.models.load_model(mdl_path)

    tstsz = 365
    klmnm = 'TAVG'
    def nrlntwrkrolling_predictions(dany):
        predictions = []

        for strt in range(len(dany)-window_sz):
            d = sclr_X.transform(np.array([dany[klmnm][strt:str +
window_sz].values])).reshape(-1, window_sz)
            prd = mdl.predict(d)
            prd = sclr_y.inverse_transform([[prd[0,0]])][0, 0]
            predictions.append(prd)
        return predictions

    frfst_steps = tstsz-window_sz

    strtchs = time.time()
    predictions =
nrlntwrkrolling_predictions(df_filled[tstindx:tstindx+tstsz])
    endchs = time.time()
    calculationchs = endchs - strtchs
    print("Час: " + str(calculationchs))

    actlvls = df_filled[tstindx:tstindx+tstsz][klmnm].iloc[-
frfst_steps:]
    mse = MSE(actlvls, predictions)
    print(f"Середньоквадратичне відхилення(MSE): {mse}")

    plt.plot(df_filled[tstindx:tstindx+tstsz].indx,
df_filled[tstindx:tstindx+tstsz][klmnm], label='Act1')
    plt.plot(actlvls.indx, predictions, label='Prdicted', color='red')
    plt.xlabel('Days')
    plt.ylabel('Tmpt')
    plt.title('CNN Predictions')
    plt.legend()
    plt.show()

    tstindx = round(df_filled.values.shape[0] * 0.8)

    trndany = df_filled['TAVG'][0:tstindx].values

    Xtrn = []

```

```

ytrn = []

for group_indx in range(int(len(trndany)-wndow_sz)):
    Xtrn.append(trndany[group_indx:group_indx+wndow_sz])
    ytrn.append([trndany[group_indx+wndow_sz]])

sclr_X = MinMaxSclr()
sclr_y = MinMaxSclr()

Xtrn = sclr_X.fittrnsfrm(np.array(Xtrn)).reshape(-1, wndow_sz)
ytrn = sclr_y.fittrnsfrm(np.array(ytrn).reshape(-1, 1))

if not os.path.exists mdl_path):
    mdl = Sequential()
    mdl.add(Dense(wndow_sz, input_dim=wndow_sz,
activation='relu'))
    mdl.add(Dense(16, activation='relu'))
    mdl.add(Dense(16, activation='relu'))
    mdl.add(Dense(8, activation='relu'))
    mdl.add(Dense(5, activation='relu'))
    mdl.add(Dense(1, activation='linear'))
    mdl.compile(optimizer='adam', loss='MSE')

    mdl.fit(Xtrn, ytrn, epochs=20, batch_sz=10, verbose=1)
    mdl.save(mdl_path)
else:
    mdl = keras.mdls.load_mdl(mdl_path)

tstsz = 365
klmnm = 'TAVG'
def nrlntwrkrolling_prdictions(dany):
    prdictions = []

    for strt in range(len(dany)-wndow_sz):
        prd =
mdl.prdict(sclr_X.transform(np.array([dany[klmnm][strt:strtwndo
w_sz].values])).reshape(-1, wndow_sz))
        prd = sclr_y.inverse_transform(prd)[0, 0]
        prdictions.append(prd)
    return prdictions

frcst_stps = tstsz-wndow_sz
strtchs = time.time()
prdictions =
nrlntwrkrolling_prdictions(df_filled[tstindx:tstindx+tstsz])
endchs = time.time()
calulationchs = endchs - strtchs
print("Чac: " + str(calulationchs))

```

```

actlvls = df_filled[tstindx:tstindx+tstsz][klmnm].iloc[-
frcst_stps:]
mse = MSE(actlvls, prdictions)
print(f"Середньоквадратичне відхилення(MSE): {mse}")

plt.plot(df_filled[tstindx:tstindx+tstsz].indx,
df_filled[tstindx:tstindx+tstsz][klmnm], label='Act1')
plt.plot(actlvls.indx, prdictions, label='Prdicted', color='red')
plt.xlabel('Days')
plt.ylabel('Tmpt')
plt.legend()
plt.show()

for group_indx in range(int(len(trndany)-wndow_sz)):
    Xtrn.append(trndany[group_indx:group_indx+wndow_sz])
    ytrn.append([trndany[group_indx+wndow_sz]])

sclr_X = MinMaxScaler()
sclr_y = MinMaxScaler()

Xtrn = sclr_X.fit_transform(np.array(Xtrn)).reshape(-1, wndow_sz)
ytrn = sclr_y.fit_transform(np.array(ytrn)).reshape(-1, 1))

if not os.path.exists mdl_path):
    mdl = Sequential()
    mdl.add(LSTM(units=50, activation='relu',
return_sequences=True, input_shape=(wndow_sz, 1)))
    mdl.add(LSTM(units=50, activation='relu',
return_sequences=True))
    mdl.add(LSTM(16, activation='relu', return_sequences=True))
    mdl.add(LSTM(16, activation='relu', return_sequences=True))
    mdl.add(LSTM(16, activation='relu', return_sequences=True))
    mdl.add(LSTM(8, activation='relu', return_sequences=True))
    mdl.add(LSTM(5, activation='relu', return_sequences=True))
    mdl.add(Dense(1, activation='linear'))
    mdl.compile(optimizer='adam', loss='MSE')

    mdl.fit(Xtrn, ytrn, epochs=15, batch_sz=10, verbose=1)
    mdl.save(mdl_path)
else:
    mdl = keras.models.load_model(mdl_path)

tstsz = 365
klmnm = 'TAVG'
def nrlntwrkrolling_predictions(dany):
    prdictions = []

```

```

        for strt in range(len(dany)-wndow_sz):
            d = sclr_X.transform(np.array([dany[klmnm][strt:str +
wndow_sz].values])).reshape(-1, wndow_sz)
            prd = mdl.predict(d)
            prd = sclr_y.inverse_transform([[prd[0,wndow_sz-
1,0]]])[0, 0]
            prdictions.append(prd)
        return prdictions

frcst_stps = tstsz-wndow_sz
strtchs = time.time()
prdictions =
nrlntwrkrolling_prdictions(df_filled[tstindx:tstindx+tstsz])
endchs = time.time()
calculationchs = endchs - strtchs
print("Час: " + str(calculationchs))
actlvls = df_filled[tstindx:tstindx+tstsz][klmnm].iloc[-
frcst_stps:]
mse = MSE(actlvls, prdictions)
print(f"Середньоквадратичне відхилення (MSE): {mse}")

plt.plot(df_filled[tstindx:tstindx+tstsz].indx,
df_filled[tstindx:tstindx+tstsz][klmnm], label='Act1')
plt.plot(actlvls.indx, prdictions, label='Prdicted', color='red')
plt.xlabel('Days')
plt.ylabel('Tmpttr')
plt.legend()
plt.show()

Xtrn = sclr_X.fittrnsfrm(np.array(Xtrn)).reshape(-1, wndow_sz, 1)
ytrn = sclr_y.fittrnsfrm(np.array(ytrn).reshape(-1, 1))

if not os.path.exists(mdl_path):
    mdl = Sequential()
    mdl.add(Bidirectional(LSTM(units=50, activation='relu',
return_sequences=True, input_shape=(wndow_sz, 1))))
    mdl.add(Bidirectional(LSTM(16, activation='relu',
return_sequences=True)))
    mdl.add(Dropout(0.2))
    mdl.add(Bidirectional(LSTM(16, activation='relu',
return_sequences=True)))
    mdl.add(Dropout(0.2))
    mdl.add(Bidirectional(LSTM(8, activation='relu',
return_sequences=True)))
    mdl.add(Bidirectional(LSTM(5, activation='relu',
return_sequences=True)))

```

```

mdl.add(Dense(1, activation='linear'))
mdl.compile(optimizer='adam', loss='MSE')

mdl.fit(Xtrn, ytrn, epochs=15, batch_sz=10, verbose=1)
mdl.save(mdl_path)
else:
    mdl = keras.models.load_model(mdl_path)

tstsz = 365
klmnm = 'TAVG'
def nrlntwrkrolling_predictions(dany):
    predictions = []

    for strt in range(len(dany)-window_sz):
        d = sclr_X.transform(np.array([dany[klmnm][strt:str +
window_sz].values])).reshape(-1, window_sz)
        prd = mdl.predict(d)
        prd = sclr_y.inverse_transform([[prd[0,window_sz-
1,0]]])[0, 0]
        predictions.append(prd)
    return predictions

frcst_stps = tstsz-window_sz
strtchs = time.time()
predictions =
nrlntwrkrolling_predictions(df_filled[tstindx:tstindx+tstsz])
endchs = time.time()
calculationchs = endchs - strtchs
print("Час: " + str(calculationchs))
actlvls = df_filled[tstindx:tstindx+tstsz][klmnm].iloc[-
frcst_stps:]
mse = MSE(actlvls, predictions)
print(f"Середньоквадратичне відхилення (MSE): {mse}")

plt.plot(df_filled[tstindx:tstindx+tstsz].indx,
df_filled[tstindx:tstindx+tstsz][klmnm], label='Act1')
plt.plot(actlvls.indx, predictions, label='Prdicted', color='red')
plt.xlabel('Days')
plt.ylabel('Tmpt')
plt.legend()
plt.show()

```

**ДОДАТОК В**  
**Слайди презентації**

Міністерство освіти та науки України  
Національний університет «Запорізька політехніка»  
Кафедра програмних засобів

## Дослідження та програмна реалізація засобів для прогнозування погодних характеристик

Виконав: Ярослав КУЛЬЧЕНКО  
ст. групи КНТ-114м

Керівник: Тетяна ЗАЙКО  
к.т.н., доцент

Рисунок В.1 – Слайд 1

## Об'єкт, предмет та мета роботи

- **Об'єкт дослідження** – процес розробки програмного забезпечення для прогнозування погоди.
- **Предмет дослідження** – програмні засоби для прогнозування погодних характеристик.
- **Метою роботи** є дослідження та розробка програмного забезпечення для прогнозування погодних характеристик.

2

Рисунок В.2 – Слайд 2

## Завдання роботи

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для прогнозування погодних характеристик;
- розробити модель програмного забезпечення для прогнозування погодних характеристик;
- здійснити проєктування програмного забезпечення для прогнозування погодних характеристик;
- створити програмне забезпечення для прогнозування погодних характеристик;
- виконати тестування розробленого програмного забезпечення для прогнозування погодних характеристик.

3

Рисунок В.3 – Слайд 3

## Порівняння програмного забезпечення для прогнозування погоди

| Критерій порівняння             | WeatherBug | Ambient Weather Network | Weather Display |
|---------------------------------|------------|-------------------------|-----------------|
| Підтримка різних платформ       | +          | +-                      | +-              |
| Наявність мобільного застосунку | +          | +-                      | -               |
| Інтеграція з метеостанціями     | -          | +                       | +               |
| Можливості налаштування         | -          | +-                      | +               |
| Зручність у використанні        | +          | +-                      | -               |
| Наявність додаткових функцій    | +          | +-                      | +               |

4

Рисунок В.4 – Слайд 4

## Вимоги до програмного забезпечення

При розробці програмного забезпечення для прогнозування погодних характеристик необхідно забезпечити такі функціональні вимоги:

- підтримка можливості введення користувачем вхідних даних (широти та довготи) для подальшого прогнозування погодних характеристик;
- підтримка можливості надсилання користувачем запиту до метеосервісу для отримання погодних даних;
- можливість обробки даних та прогнозування погодних характеристик;
- можливість виведення в екранну форму результатів щодо прогнозування температури, вологості, швидкості вітру та інших характеристик;
- підтримка можливості прогнозування погодних характеристик для інших місць (повторного прогнозування при інших вхідних даних);
- підтримка можливості збереження результатів прогнозування.

5

Рисунок В.5 – Слайд 5

## Порівняння мов програмування

| Критерій порівняння мов програмування                               | Мова програмування |      |    |
|---------------------------------------------------------------------|--------------------|------|----|
|                                                                     | Python             | Java | R  |
| Бібліотеки для аналізу даних                                        | +                  | +-   | +  |
| Підтримка машинного навчання                                        | +                  | +-   | +- |
| Засоби візуалізації результатів                                     | +                  | -    | +- |
| Зручність для створення ПЗ для прогнозування погодних характеристик | +                  | -    | -  |
| Низький поріг входу для користувачів                                | +                  | +-   | +- |

6

Рисунок В.6 – Слайд 6

## Порівняння середовищ розробки

| Критерій порівняння середовищ розробки | Середовища розробки |                  |        |
|----------------------------------------|---------------------|------------------|--------|
|                                        | PyCharm             | Jupyter Notebook | Spyder |
| Засоби налагодження та тестування      | +                   | –                | +–     |
| Автодоповнення та підказки             | +                   | +–               | +      |
| Інтеграція з системами контролю версій | +                   | –                | +–     |
| Ефективність для командної роботи      | +                   | –                | +–     |
| Швидкість розробки великих систем      | +                   | –                | +–     |

Рисунок В.7 – Слайд 7

## Модель програмного забезпечення для прогнозування погодних характеристик

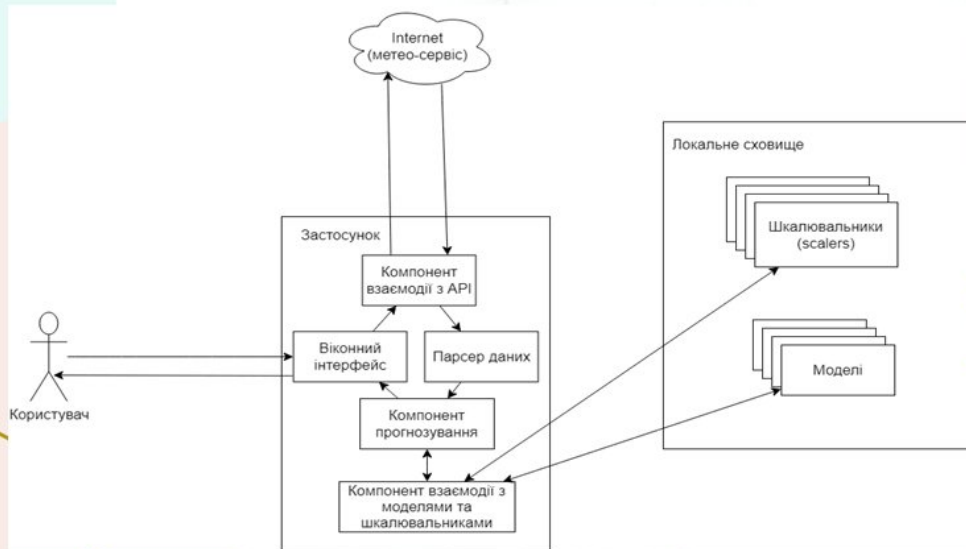
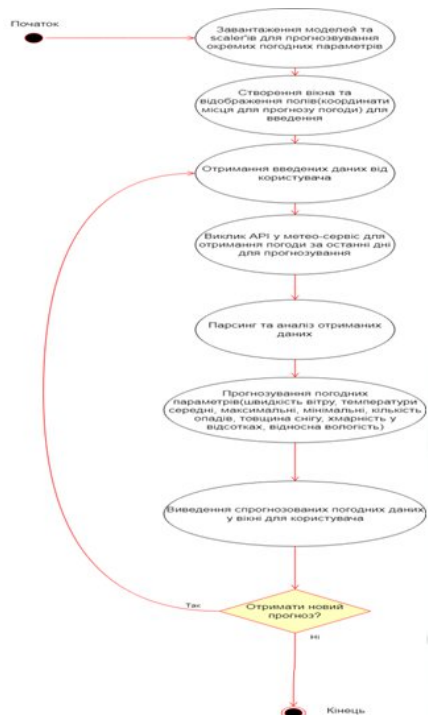


Рисунок В.8 – Слайд 8

# Структура програмного забезпечення



Рисунок В.9 – Слайд 9



## Схема функціонування програми

Рисунок В.10 – Слайд 10

## Робота з програмою. Введення вхідних даних. Отримання результатів прогнозування погоди

Прогнозування погоди

Введіть вхідні дані для прогнозування:

Широта:

Довгота:

Отримати прогноз

Прогнозування погоди

Середньодобова температура: 2,3 C  
Максимальна температура: 7,8 C  
Мінімальна температура: -2,7 C  
Відносна вологість: 43%  
Хмарність: 12%  
Швидкість вітру: 4,3 м/с

Обрати інше місце

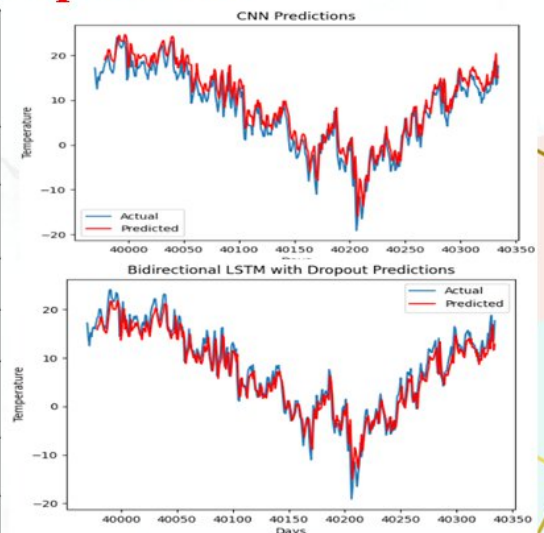
Зберегти результати

11

Рисунок В.11 – Слайд 11

## Експериментальне дослідження прогнозування погодних характеристик

| Метод                                | Середньо-квадратичне відхилення | Час прогнозування (мс) | Швидкість навчання | Складність моделі | Використання ресурсів |
|--------------------------------------|---------------------------------|------------------------|--------------------|-------------------|-----------------------|
| Згортова нейромережа (CNN)           | 8,62                            | 23,18                  | Середня            | Середня           | Середні               |
| Глибока нейромережа (Deep NN)        | 5,91                            | 19,93                  | Середня            | Висока            | Високі                |
| Нейромережа GRU                      | 5,28                            | 20,47                  | Середня            | Середня           | Середні               |
| Нейромодель LSTM                     | 5,71                            | 22,54                  | Середня            | Висока            | Високі                |
| Модель LSTM із шарами Dropout        | 5,67                            | 20,87                  | Середня            | Висока            | Середні               |
| Модель GRU із шарами Dropout         | 12,98                           | 11,08                  | Середня            | Середня           | Середні               |
| Модель GRU з алгоритмом навчання SGD | 13,47                           | 17,81                  | Середня            | Середня           | Середні               |



12

Рисунок В.12 – Слайд 12

## Висновки

В ході виконання кваліфікаційної роботи було проаналізовано та досліджено процес розробки програмного забезпечення для прогнозування погодних характеристик.

Новизна роботи полягає в тому, що створено модель програмного забезпечення для прогнозування погодних характеристик, яка подана у вигляді багаторівневої системи, у якій відбувається взаємодія між користувачем, застосунком, локальним сховищем та зовнішнім метеосервісом.

Практичне значення роботи полягає в тому, що розроблено програмне забезпечення для прогнозування погодних характеристик.

Усі завдання кваліфікаційної роботи повністю виконано.

13

Рисунок В.13 – Слайд 13