Міністерство освіти і науки України
Харківський національний університет імені В.Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту

Рекомендовано до захисту:
протокол засідання кафедри
№ ___ від _____.____.2025 р.
Завідувач кафедри _____
_____ _____
(підпис) (прізвище та ініціали)

Пояснювальна записка

до кваліфікаційної роботи бакалавра

на тему «Розробка інформаційної системи раціонального планування розподілу електроенергії в системах енергопостачання великих міст в умовах її дефіциту»

Захищено на засіданні ЕК № _____
протокол № ____ від ____.____.2019 р.
Оцінка _____ / _____
Голова ЕК
_____ _____
(підпис) (прізвище та ініціали)

Виконав:
студент 4 курсу, групи ЗКС- 41
Спеціальності: 122 комп’ютерні науки
(шифр і назва спеціальності підготовки)
Биканов Артем Анатолійович
(прізвище, ім’я та по батькові, підпис)
Керівник Ph.D. Ковальчук Д.М.
(ступень, звання, прізвище та ініціали, підпис)
Рецензент _____
(ступень, звання, прізвище та ініціали, підпис)

Харків – 2025

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

Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра математичного моделювання та аналізу даних
Рівень вищої освіти (освітньо-кваліфікаційний рівень) бакалавр
Галузь знань: 12 Інформаційні технології
Спеціальність: 122 Комп'ютерні науки
Освітня програма: Комп'ютерні науки

ЗАТВЕРДЖУЮ
В.О. завідувача кафедри ____
Володимир СТРУКОВ
«_03_» _лютого_2025_ року

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

__Биканов Артем Анатолійович__
(прізвище, ім'я, по батькові студента)

1. Тема роботи Розробка інформаційної системи раціонального планування розподілу електроенергії в системах енергопостачання великих міст в умовах її дефіциту,

керівник роботи Ковальчук Дмитро Миколайович, старший викладач, Ph. D.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від "_16_" _квітня_2025 року №4101-5/962__

2. Строк подання студентом роботи 05 червня 2025

3. Перелік питань, які потрібно розробити: Аналіз предметної області. Вивчення літератури, огляд публікацій. Визначення вимог до інформаційної системи. Вибір технологій та програмних засобів реалізації. Проєктування та програмна реалізація інформаційної системи. Аналіз результатів та їх узагальнення. Написання пояснювальної записки, створення презентації та доповіді.

4. План роботи

№ з/п	Назви етапів роботи
1	Аналіз предметної області
2	Вивчення літератури, огляд публікацій
3	Визначення вимог до інформаційної системи
4	Вибір технологій та програмних засобів реалізації
5	Проєктування та програмна реалізація інформаційної системи
6	Аналіз результатів та їх узагальнення
7	Написання пояснювальної записки, створення презентації та доповіді

5. Дата видачі завдання 16.04.2025

Студент Биканов А. А.
ініціали, прізвище

підпис

Керівник роботи Ковальчук Д. М.
ініціали, прізвище

підпис

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

БД – База даних;

ЛЕП – Лінія електропередачі;

ІС – Інформаційна система;

ЕС – Енергетична система;

ЕМ – Електрична мережа;

ВСТУП

У сучасних умовах зростаючої енергетичної нестабільності проблема ефективного розподілу електроенергії набуває особливої актуальності. Системи енергопостачання великих міст часто стикаються з викликами, спричиненими як внутрішніми технічними обмеженнями, так і зовнішніми факторами — зокрема, воєнними діями, економічними кризами, кібератаками, нестачею палива чи обмеженим імпортом енергоресурсів. У таких умовах питання раціонального планування споживання та розподілу електроенергії перетворюється з інженерного завдання на стратегічну необхідність.

Актуальність дослідження полягає в інтеграції сучасних підходів до проектування інформаційних систем із практичними потребами енергетичної галузі. Застосування інформаційних технологій дає змогу оперативно обробляти великі обсяги даних, здійснювати прогнозування навантаження та приймати обґрунтовані рішення щодо перерозподілу ресурсів у реальному часі.

Мета даної роботи - проєктування та розробка інформаційної системи, що забезпечує раціональне планування та ефективний розподіл електроенергії в енергосистемах великих міст за умов її дефіциту. Така система має сприяти мінімізації енергетичних втрат, підвищенню стабільності функціонування критичної інфраструктури та зниженню соціальних ризиків, пов’язаних із можливими аваріями або знеструмленнями.

Для досягнення поставленої мети в роботі вирішуються такі основні завдання:

- провести аналіз предметної області, визначити структуру сучасних енергетичних систем великих міст та проаналізувати основні причини виникнення дефіциту електроенергії;

- дослідити існуючі методи раціонального планування та розподілу електроенергії, оцінити їх ефективність в умовах обмежених енергоресурсів;

- визначити функціональні та нефункціональні вимоги до інформаційної системи, яка має підтримувати процеси планування та керування розподілом електроенергії;

- спроєктувати архітектуру інформаційної системи, що дозволить забезпечити гнучкість, масштабованість і стійкість до змін в енергетичному навантаженні;

- розробити прототип інформаційної системи, що реалізує ключові функції: збір, обробку, аналіз даних і прийняття рішень щодо розподілу електроенергії;

- оцінити результати розробки за допомогою тестування та аналізу функціонування системи на прикладах сценаріїв дефіциту електроенергії.

Об’єкт дослідження є процеси планування та розподілу електроенергії в системах енергопостачання великих міст.

Предметом дослідження є методи та програмні засоби раціонального планування розподілу електроенергії в умовах її обмеженої доступності.

Наукова новизна роботи полягає у застосуванні методів раціонального планування до задач енергетичного розподілу в критичних умовах, а також у розробці інформаційної системи, яка здатна адаптуватися до динамічних змін у споживанні та доступності електроенергії.

Практичне значення полягає у можливості впровадження результатів роботи в існуючі системи управління енергопостачанням для підвищення їх ефективності, гнучкості та стійкості до надзвичайних ситуацій.

Структурно робота складається з вступу, трьох основних розділів та висновків. У вступі представлено актуальність роботи, сформульовано мету та відповідні завдання, об’єкт та предмет дослідження, наведено загальну структуру роботи. У першому розділі проведено аналіз предметної області, зокрема будови енергосистеми, причин виникнення дефіциту електроенергії та методів її раціонального розподілу. У другому розділі описано етапи проєктування інформаційної системи, визначено її вимоги та ключові компоненти. Третій розділ присвячений безпосередній розробці

інформаційної системи, включаючи вибір інструментів та опис реалізованих алгоритмів. Висновок висвітлює інформацію щодо підсумків дослідження, його наукової та практичної значущості, можливі перспективи подальшого розвитку.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

У цьому розділі розглянуто основні поняття предметної області. Описано будову електричної мережі та систематизовано її провідні ознаки. Крім цього, надано визначення поняття раціонального планування. Описано причини виникнення проблеми дефіциту електроенергії. Розділ завершується описом методів оптимізації.

1.1. Будова енергосистеми

Одним з провідних понять досліджуваної предметної області є енергосистема, це поняття визначається як сукупність електростанцій та мереж, необхідних для передачі, перетворення та розподілення електроенергії. Електрична станція являє собою комплекс обладнання, призначений для перетворення інших видів енергії в електричну. Електростанції можуть бути атомними, тепловими, гідравлічними, сонячними, вітровими, приливними, геотермальними, тощо. Постачання отриманої електроенергії відбувається за допомогою електричної мережі, що являє собою комплекс устаткування, необхідного для передачі та розподілу електроенергії. Електрична мережа включає лінії електропередачі, підстанції та розподільні пункти. Лінії електропередачі є тією частиною електричної мережі, яка виконує передачу електроенергії від генеруючих електроустановок до перетворюючих та розподільних пунктів. В залежності від способу передачі електроенергії лінії електропередачі розділяються на кабельні, які виконують передачу за допомогою кабелів, та на повітряні, які виконують передачу за допомогою дротів, що знаходяться на відкритому повітрі. Повітряні лінії є простішими у будівництві, проте у разі неможливості їхньої експлуатації прокладаються кабельні лінії, таким чином у разі необхідності подолання певних перешкод, як то автотраса чи залізниця, використовуються саме кабельні лінії електропередачі. Електричні підстанції відповідають за перетворення електроенергії, вони можуть змінювати

напругу або здійснювати інші перетворення. Розподільні пункти являють собою електроустановки, що відповідають за розподілення електроенергії між кінцевими споживачами без виконання перетворень.

Smart Grid, або розумні енергомережі, являють собою покоління електроенергетичних систем, яке поєднує класичну інфраструктуру з цифровими технологіями, зокрема інтернетом речей, машинним навчанням, обробкою даних в хмарних середовищах та з іншими технологіями. Smart Grid забезпечує двосторонню взаємодію між споживачем і постачальником електроенергії, що дає змогу адаптивно керувати попитом та підвищити енергоефективність. Ключові компоненти Smart Grid включає в якості компонентів інтелектуальні лічильники (Smart Meters), автоматизовані системи диспетчеризації, системи управління навантаженням, системи виявлення й локалізації аварій, цифрові підстанції, прилади інтернету речей та інші інструменти. Однією з переваг Smart Grid є можливість реального часу виявляти перевантаження, втрати, аварії, і вчасно реагувати. Також такі мережі підтримують інтеграцію відновлюваних джерел енергії, що є критично важливим у контексті енергетичного переходу та стійкості до дефіциту.

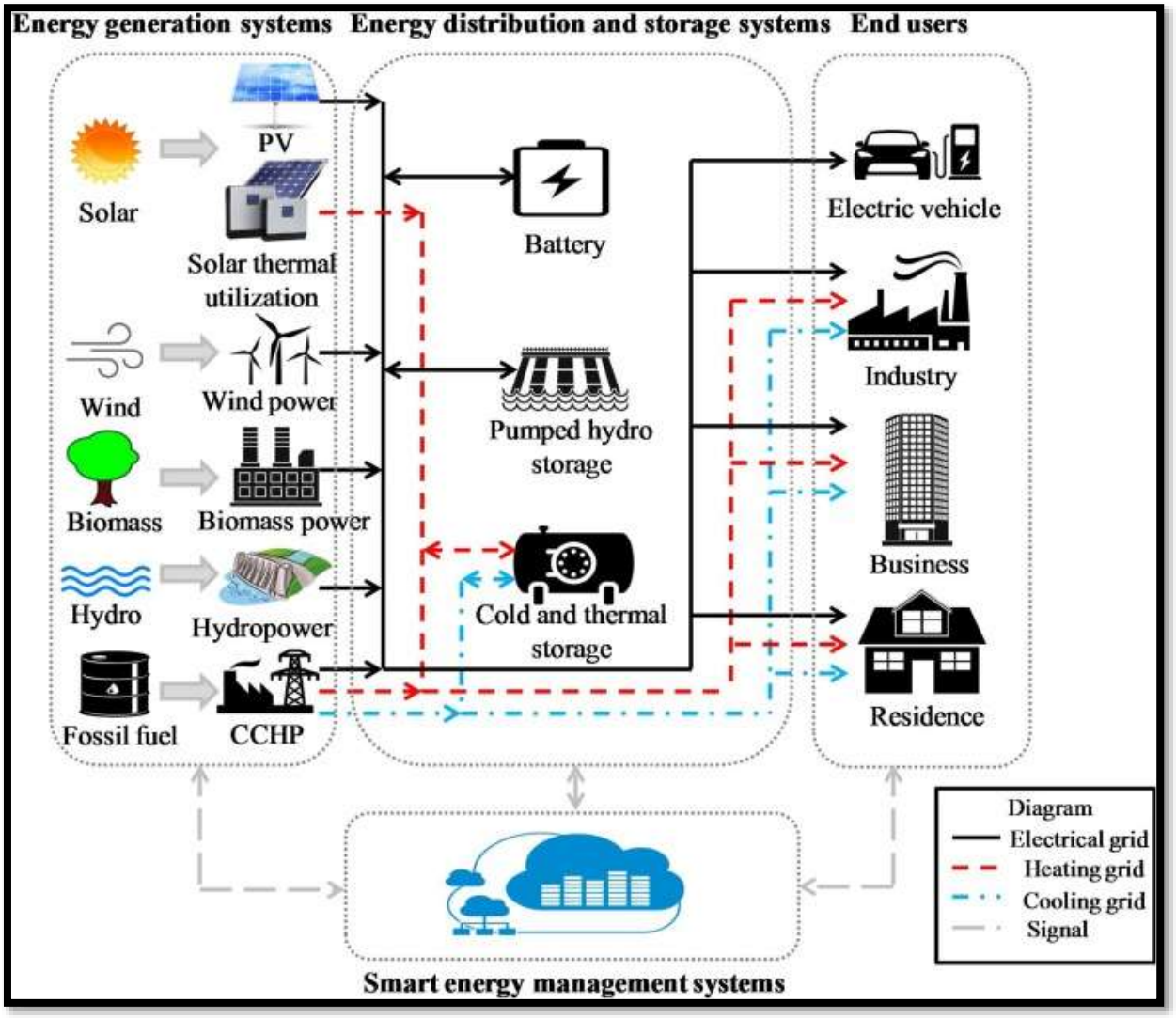

Рисунок 1.1 – Схема енергосистеми [3]

1.2. Причини виникнення дефіциту електроенергії

Дефіцит електроенергії – являє собою ситуацію, яка виникає коли попит на перевищує наявну кількість електроенергії або ж коли можливість її постачання не є достатньою. Така ситуація може виникати з різних причин. Причиною нестачі електроенергії може бути непередбачений ріст споживання електроенергії або аварії на електростанціях, які призводять до зменшення генеруючої потужності цих станцій. Причиною неможливості постачання електроенергії до кінцевого споживача можуть стати аварії пошкодження елементів електричної мережі.

Лінії електропередачі можуть бути пошкоджені внаслідок короткого замикання або інших чинників. Повітряні лінії унаслідок того що знаходяться на відкритому просторі є вразливими перед екстремальними погодними умовами, в свою чергу кабельні лінії можуть виходити з ладу через непридатний стан зовнішньої оболонки кабелю.

Іншим чинником дефіциту електроенергії є аварії на підстанціях та розподільних пунктах. Трансформаторні підстанції можуть виходити з ладу через унутрішні поломки, неналежний ремонт або обслуговування, пошкодження ізоляції внаслідок відсутності постійної напруги та з інших причин. Аварії в мережі часто супроводжуються каскадним ефектом: відмова одного елементу спричиняє перерозподіл навантаження й поширення відмов по системі [14]. Саме каскадні події спричиняють розлади, що переходять в блекаути [15].

У разі дефіциту електроенергії можуть бути застосовані резервні потужності, у разі нестачі резервної потужності можуть відбуватися тимчасові відключення електроенергії для деяких споживачів. Нестача електроенергії призводить до необхідності раціонального планування її розподілу.

1.3. Методи раціонального планування розподілу електроенергії

Раціональне планування розподілу електроенергії передбачає необхідність ретельного спостереження за станом енергосистеми та аналіз і прогнозування попиту на електроенергію. Моніторинг за енергоспоживанням може проводитися за допомогою датчиків, лічильників та сенсорів інтернету речей. З огляду на це має бути реалізована система моніторингу енергоспоживання. Традиційно архітектура системи моніторингу складається з трьох частин. Перша частина включає сенсори інтернету речей, які збирають дані. Далі ці дані за допомогою спеціальних протоколів пересилаються до наступного рівня, який включає бази даних та програмне забезпечення для обробки інформації. Передача даних може бути здійснена

за допомогою різних технологій, так до сенсорів першого рівня можуть бути додані вай-фай модулі або, до прикладу, радіомодеми, також є можливим використання технології Ethernet. Третій рівень включає програмне забезпечення, яке дозволяє переглядати та вносити дані.

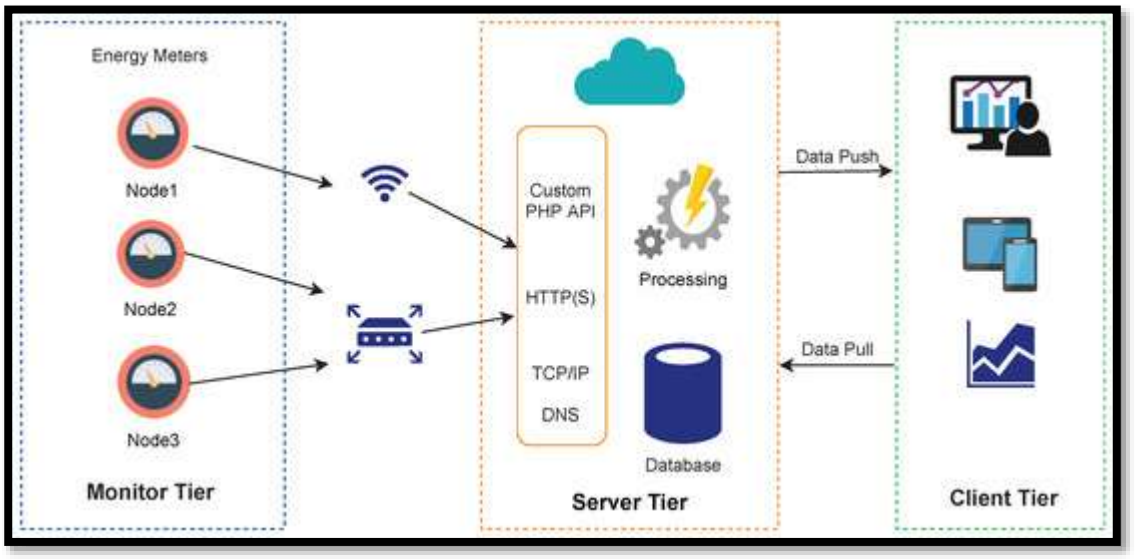

Рисунок 1.2 – Архітектура системи моніторингу [5]

При цьому розумні датчики можуть спростити моніторинг та прийняття рішень щодо розподілення електроенергії. Також є можливим використання обладнання для оптимізації використання електроенергії кінцевим споживачем, що також сприяє заощадженню в умовах дефіциту. Оскільки причиною неможливості надати необхідну кількість електроенергії часто стають аварії, необхідно також мати можливість реєструвати аварійні випадки.

Проте окрім моніторингу необхідною є також можливість прогнозувати навантаження для керування обладнанням, розподілення резервів і вибору режиму енергосистеми. Таким чином саме прогнозування дає вихідну інформацію для подальшого керування та прийняття рішень. В свою чергу прогнозування може базуватися на даних, отриманих під час моніторингу стану енергосистеми. Тому задачу прогнозування можна представити як аналіз часових рядів, адже цей метод дозволяє обробляти

великі об’єми даних зібрані під час моніторингу, які пов’язані зі споживанням електроенергії в минулому, та шукати в цих даних тренди і закономірності. Одна з основних задач аналізу часового ряду полягає в тому, що дано ряд спостережень $\{x_t\}_{t=1}^{T}$ випадкової величини $\xi(t)$, при цьому необхідно встановити значення наступного елемента ряду [4]. Метод аналізу часових рядів може допомогти прогнозувати енергоспоживання районами міст та підприємствам. Однак прогноз електроспоживання вимагає також врахувати значну кількість випадкових факторів, тому у разі якщо метод аналізу числових рядів не даватиме необхідної точності прогнозування, необхідно буде поєднати його з іншими методами для забезпечення необхідної точності.

Таким чином для прогнозування може бути застосований також і регресійні методи, оскільки саме вони дозволяють також врахувати зовнішні фактори, як то погодні умови та зміни в електроспоживанні. Методи регресійного аналізу є найпоширенішими статистичними інструментами для виявлення залежностей між змінними [12, с. xvii]. Регресійний аналіз дозволяє представити прогнозовану величину як залежну змінну Y, значення якої визначається на основі незалежних змінних X_i .

Значення залежної змінної обраховується за наступною формулою:

$$Y = \beta_0 + \sum_{\iota=1}^{n} \beta_\iota X_\iota + \varepsilon, \tag{1.1}$$

де β_ι – коефіцієнти регресії;

ε – похибка.

В найпростішому випадку можна застосувати простий метод прогнозування, сутність якого полягає в тому щоби визначити функцію

$y = a + b * x$, де y відповідає значенням часового ряду, а x - часу. Параметри a і b можна підібрати за методом найменших квадратів:

$$a = \frac{\sum_{i=0}^{n} y_i + b \sum_{i=1}^{n} x_i}{n}, \tag{1.2}$$

$$b = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}, \tag{1.3}$$

де $\bar{x}, \bar{y}$ – середні значення.

Однак ця модель ігноруватиме сезонність та циклічність описуваного явища, адже енергоспоживання залежить від часу та від пори року. Також ця модель є вразливою до впливу випадкових коливань. Застосування метод найменших квадратів також не даватиме задовільних результатів, якщо модель, що передбачає апроксимацію прямою, не підходить до явища, що розглядається [8, с.8]. Проте таку модель можна реалізувати для порівняння її результатів з результатами складніших моделей.

У разі якщо споживана потужність перевищуватиме потужність, що генерується, виникне дефіцит, що в свою чергу призведе до зниження частоти струму. Однак прогнози споживання електроенергії дозволяють завчасно прийняти рішення щодо використання резервних потужностей або відключення частини кінцевих споживачів. Розробка систем раннього попередження енергетичних криз є важливою складовою економічної, соціальної та екологічної сталості [13, с.2].

На промислових підприємствах споживання електроенергії, як правило, досягає свого максимуму в ранковий та вечірній період, в середині дня споживання є меншим, вночі сягає мінімуму. Крім цього споживання також

залежить від пори року, так взимку споживання зростає через потребу в опалюванні приміщень, влітку – за рахунок витрат на зниження температури приміщень. Динаміку споживання електроенергії протягом року можна спостерігати на графіку.

Рисунок 1.3 – Графік споживання електроенергії протягом року [7]

Оскільки споживання електроенергії протягом доби та протягом року є нерівномірним, для раціонального планування доцільно визначити піки споживання електроенергії, адже це дозволяє застосувати певні методи заощадження електроенергії за рахунок регулювання споживаної потужності. Так, на машинобудівних і приладобудівних заводах можна робити переключення індукційних установок і термопечей у режим підігріву на час максимуму навантаження [6, с. 448]. На підприємствах у разі нестачі резервних потужностей відключення споживачів відбувається чергами [6].

Визначити час максимального та мінімального навантаження можна на основі спостереження за попередньою динамікою енергоспоживання певних районів та підприємств, а також за рахунок застосування обраних методів прогнозування.

У разі якщо прогнозоване енергоспоживання перевищує наявні потужності, доцільно також спрогнозувати витрати, що можуть бути спричинені відключенням певної частки промислових споживачів, адже відключення призведе до простою промислового устаткування. Виходячи зі спрогнозованих збитків можна приймати рішення про відключення тих чи інших кінцевих споживачів у разі нестачі резервної потужності.

Таким чином статистична від статистичної моделі вимагається здатність враховувати специфіку предметної області, а саме сезонність, циклічність та наявність випадкових коливань. Однак при цьому від моделі вимагається також простота реалізації та обчислень. З огляду на це доцільним убачається застосування статистичної моделі ARIMA, яка передбачає усунення впливу випадкових коливань за рахунок використання ковзного середнього, а також використання автокореляції для виявлення повторюваних у часі закономірностей. Також ця модель не є надто складною в реалізації, адже для багатьох мов програмування вже впроваджено бібліотеки та залежності, що допомагають працювати з цією моделлю. Моделі ARIMA є, можливо, найбільш часто використаними при прогнозуванні майбутніх значень [11, с.54].

ARIMA поєднує три компоненти: авторегресію (AR), інтегрування (I) та ковзне середнє (MA). Компонент AR описує залежність поточного значення від його попередніх значень. Компонент I забезпечує стаціонарність ряду, тобто усунення впливу тренду шляхом диференціювання. Компонент MA дозволяє враховувати залежність поточного значення від попередніх помилок прогнозу.

Типова модель ARIMA(p,d,q) має три параметри:

- **p** — порядок авторегресії (кількість попередніх значень);
- **d** — порядок інтегрування (кількість диференціювань для стаціонарності);
- **q** — порядок ковзного середнього.

Головна формула моделі ARIMA має вигляд:

$$\Phi_p(B)(1-B)^d y_t = \theta_q(B)\varepsilon_t, \tag{1.4}$$

де y_t — значення часового ряду в момент часу ;

$By_t = y_{t-1}$ — оператор зсуву;

$(1-B)^d$ —оператор диференціювання порядку d;

$\Phi_q(B) = 1 - \sum_{i=1}^{p} -\theta_i B^i$— поліном авторегресії порядку p;

$\theta_q(B) = 1 + \sum_{i=1}^{q} \theta_i B^i$— поліном ковзного середнього порядку q;

ε_t — випадкова помилка (шумова компонента).

Поліном авторегресії дозволяє змоделювати вплив попередніх значень часового ряду на поточне значення. Цей компонент дозволяє врахувати інерційність системи. За p = 1 поточне значення y_t залежатиме від попереднього значення y_{t-1} з коефіцієнтом Φ_{t-1}.

Поліном ковзного середнього моделює вплив відхилення реальних значень від прогнозованих у попередні моменти часу на поточне значення. Так за q = 1 поточне значення y_t враховує не лише поточну помилку ε_t, але й попередню ε_{t-1}, з відповідним коефіцієнтом θ_{t-1}. Цей компонент згладжує випадкові коливання, що виникають у наборі даних.

Таким чином, модель ARIMA включає диференціювання вхідного ряду для досягнення стаціонарності, після чого застосовується комбінація AR і MA компонент для опису залишкової структури. ARIMA дозволяє не тільки прогнозувати значення енергоспоживання на основі історичних даних, а й адаптуватися до змін у структурі даних, наприклад до сезонних коливань чи трендів. Реалізація ARIMA може бути значно спрощена завдяки наявності бібліотек у мовах Python (statsmodels), Java (Workday TimeSeries), R (forecast) тощо.

2 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

У цьому розділі розглянуто основні теоретичні поняття дослідження. Надано визначення інформаційної системи та систематизовано провідні ознаки цього поняття. Крім цього, надано визначення поняття раціонального планування. Сформульовано основні вимоги до ПЗ, що розробляється. Розділ завершується описом методики дослідження.

2.1. Поняття інформаційної системи

Інформаційна система визначається як набір взаємопов’язаних компонентів, призначений для збирання, обробки, зберігання та розповсюдження інформації, а також для отримання зворотного зв’язку для досягнення певної цілі [1, с. 4].

Щодо класифікації інформаційних систем зазвичай застосовується так звана пірамідальна модель. Сутність цієї моделі полягає в тому, що інформаційні системи розподіляються на п’ять типів, які в свою чергу ієрархічно розподіляються на три рівні [2]. Найнижчій рівень називається операційним рівнем, він включає системи автоматизації виробництва (process control systems) та системи обробки транзакцій (transaction processing systems). Системи автоматизації виробництва використовуються для того аби відстежувати та контролювати фізичні процеси, при цьому для збору даних використовуються сенсори. Системи такого типу можуть використовуватися в енергосистемі для збирання даних щодо споживання електроенергії за допомогою лічильників, датчиків та сенсорів інтернету речей. Системи обробки транзакцій використовуються для того щоб збирати, аналізувати та оновлювати дані, які стосуються транзакцій, при цьому на відміну від попереднього типу інформаційних систем сенсори не використовуються. Інформаційна система такого типу може допомогти при збиранні інформації, яка не може бути зібрана за допомогою датчиків інтернету речей, як то звіти.

Середній рівень, який має назву рівень середньої ланки управління (middle management level), включає два типи інформаційних систем, а саме системи підтримки прийняття рішень (decision support systems) та інформаційні системи управління (management information systems). Інформаційні системи підтримки прийняття рішень використовуються для збору та аналізу інформації з метою створення детальних звітів, які призначені для спрощення процесу прийняття рішень. Інформаційні системи такого типу можуть використовуватись в енергосистемі для унаочнення динаміки енергоспоживання шляхом побудови графіків. Інформаційні системи підтримки прийняття рішень в енергетиці дозволяють ефективно реагувати на кризи постачання та забезпечувати стабільність в умовах дефіциту [9]. Інформаційні системи управління збирають та оброблюють інформацію з баз даних і таким чином спрощюють процес управління підприємством. Такі інформаційні системи можуть використовуватись для обробки великого об’єму інформації, яка стосується споживання електроенергії та зберігається в базах даних

Останній рівень має назву виконавчий рівень (executive level), він включає лише один тип інформаційних систем, а саме виконавчі інформаційні системи (executive information systems). Ці системи збирають інформацію з унутрішніх та зовнішніх баз даних, а також інших джерел, для того щоб спростити аналіз становища, визначення головних тенденцій та прийняття стратегічних рішень. Інформаційна система такого типу може бути використана для збору та аналізу великого об’єму даних, що надходять з різних джерел, а також для побудування графіків, прогнозування споживаної потужності в короткостроковій та довгостроковій перспективі з використанням обраних методів прогнозування, для обрахування потенційних збитків від можливого відключення промислових споживачів та для прийняття рішень відносно використання резервних потужностей та відносно заощадження електроенергії у разі якщо прогнозоване споживання електроенергії перевищуватиме наявні потужності. Інтеграція інформаційних

технологій в управлінські процеси дозволяє підвищити ефективність використання ресурсів, особливо в умовах нестачі та невизначеності [10].

Серед причин нестачі електроенергії можуть погодні умови (бурі, сніг, спека), зношеність обладнання, помилки персоналу, відсутність належних засобів контролю кібератаки. Багато з цих факторів є прогнозованими, і таким чином система раннього сповіщення може стати у нагоді для запобігання дефіциту електроенергії.

Для запобігання нестачі електроенергії інформаційні системи можуть виявляти тренди деградації обладнання забезпечувати автоматизоване відключення аварійних ділянок, моделювати сценарії навантаження для попередження перевантажень, впроваджувати системи раннього сповіщення про можливості нестачі електроенергії (early warning systems). Це сприяє підвищенню надійності енергетичної інфраструктури та її здатності функціонувати в умовах нестабільного постачання.

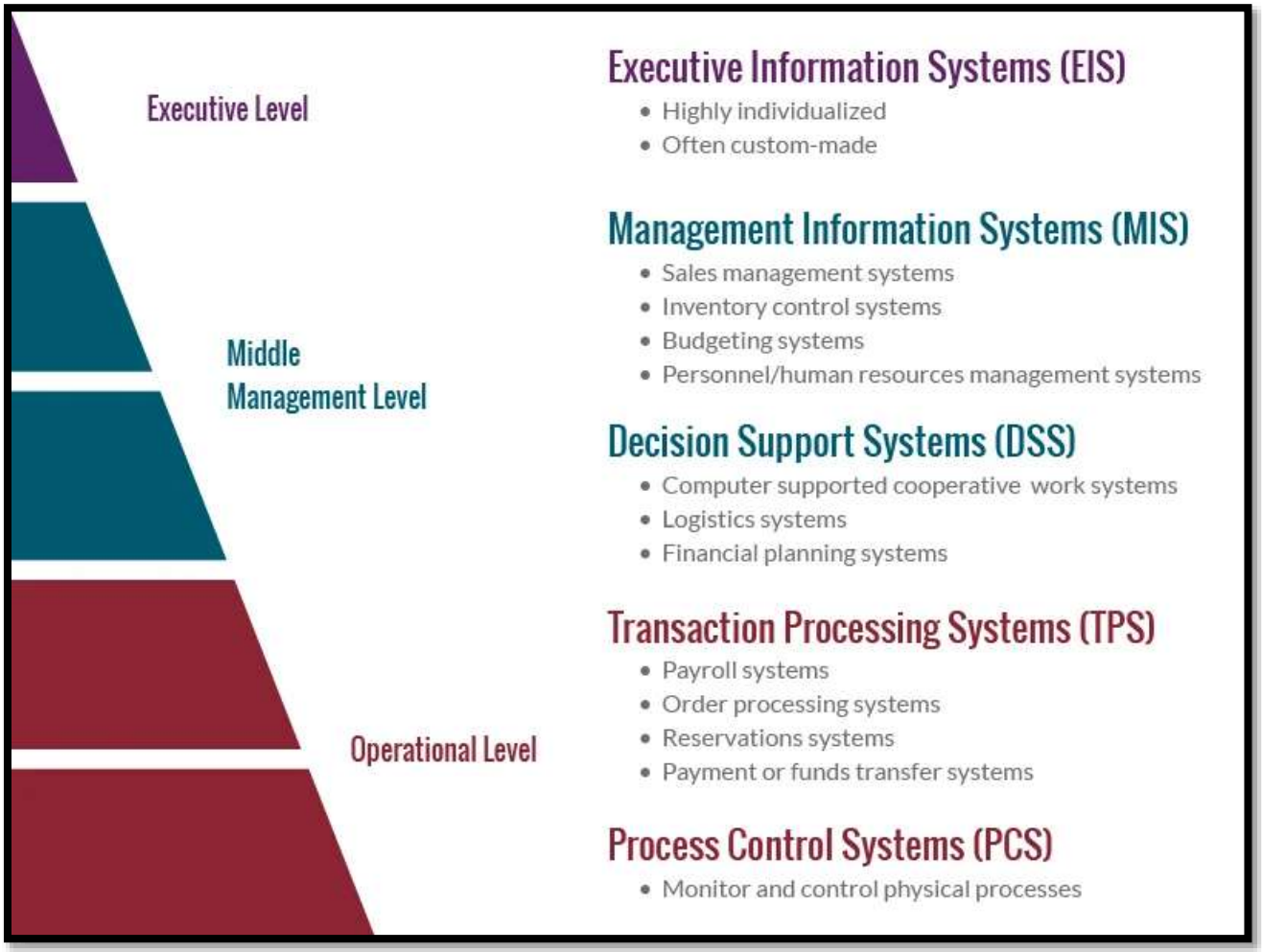

Рисунок 2.1 – Пірамідальна модель класифікації інформаційних систем [2]

2.2 Визначення вимог до інформаційної системи

Як вже було зазначено від інформаційної системи, що проектується, вимагається здатність збирати та аналізувати великий об’єм даних, що надходять від різних джерел, як то сенсорів та датчиків інтернету речей, а також вимагається здатність проводити обчислення з отриманими даними, а саме прогнозувати майбутні значення, проводити перевірки на наявність трендів та закономірностей в наборах даних, обчислювати потенційні витрати через відключення промислових споживачів і унаочнювати отриману інформацію у вигляді графіків з метою спрощення прийняття рішень щодо розподілу електроенергії.

Споживання електроенергії вимірюватиметься лічильниками, але зберігатиметься в базах даних, тому від інформаційної системи буде вимагатися можливість одержувати інформацію з баз даних. Крім цього від обраної мови програмування буде вимагатись наявність певних рішень щодо роботи зі статистичними моделями, а також можливість інтегрувати в проєкт нові методи прогнозування, планування та моніторингу. Таким чином якщо буде прийнято рішення щодо застосування нової моделі, то має бути можливість інтегрувати в проєкт певні залежності, які спростять реалізацію цих моделей, адже ці моделі можуть бути достатньо складні в реалізації.

Крім цього деякі моделі передбачають виконання досить складні обчислень, адже багато моделей використовують трудомісткі чисельні методи, зокрема при використанні методів інтегрально обчислення для знаходження взаємної кореляції та автокореляції. Таким чином обрані інструменти мусять економити обчислювальні потужності при виконанні прогнозів.

Основна мета інформаційної системи — забезпечення ефективного моніторингу, аналізу та прогнозів споживання електроенергії в умовах обмеженого постачання. Для цього система має включати такі функціональні модулі як модуль збору даних, що запроваджує автоматизоване зчитування показників з інтелектуальних лічильників та підтримку протоколів передачі даних; аналітичний модуль**,** що запроваджує побудову графіків споживання і відслідковування трендів, використання алгоритмів аналізу даних та статистичних моделей ARIMA, а також порівняльний аналіз по регіонах, споживачах та часу; модуль прогнозування, що запроваджує можливість робити прогнози споживання на коротко- та середньостроковий період, обробляти сезонні та випадкові коливання, а також виводити довірчі інтервали для майбутніх значень. Також вимагається впровадження системи раннього сповіщення, яка повідомляє операторам про перевищення порогів у випадках перевищення порогів, несправностей або аномалій у споживанні.

Крім цього має бути реалізована адміністративна панель у вигляді інтерфейсу користувача, що дозволить переглядати статистику споживання електроенергії та унаочнювати показники на графіках. Для того аби реалізувати можливість передачі продукту слід розглянути контейнеризацію проєкту.

Запровадження цих функцій забезпечить ефективне управління енергетичними ресурсами, вчасне реагування на критичні ситуації, а також підвищить прозорість та прогнозованість у роботі системи.

3 РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

У цьому розділі згідно з визначеними вимогами до проєкту було визначено основні інструменти розробки інформаційної системи, як то бібліотеки, залежності, мови програмування та фреймворки. Описано основні етапи розробки інформаційної системи, а також рішення. Крім цього, надано опис основних методів.

3.1. Визначення інструментів розробки

Як вже було визначено в попередньому розділі інформаційна система мусить надавати можливість отримувати інформацію з баз даних та виконувати її обробку. При цьому можуть стати у нагоді певні готові рішення щодо статистичних моделей та інших методів прогнозування.

На початку розробки інформаційної системи розглядалася мова програмування Python, оскільки вона надає низку готових рішень, які стосуються деяких статистичних моделей та методів машинного навчання. Також могли б стати у нагоді бібліотеки, які допомагають унаочнити дані, як до прикладу mathplotlib. Проте python є інтерпретованою мовою програмування, і програми, написані цією мовою, як правило, працюють повільніше ніж аналогічні програми, написані компільованими мовами програмування. Це може бути важливим при обробці великого об’єму статистичних даних, що можуть стосуватися динаміки споживання електроенергії.

Фреймворк Maven дозволяє інтегрувати готові рішення в проєкт, написаний мовою Java, в тому числі можна додати залежності, що можуть спростити реалізацію складних алгоритмів, які стосуються статистичних моделей або машинного навчання. Особливість цього фреймворку полягає в тому, що програми, написані з його використанням, самі можуть бути включені в наступні проєкти на правах залежності.

У відкритому доступ вдалося знайти програми з відкритим вихідним кодом, які написані мовою Java з використанням Maven та можуть спростити певні етапи розробки. Так було знайдено проєкт, що дозволяє прогнозувати майбутні значення часових рядів з використанням статистичної моделі ARIMA, яка в іншому випадку була б складною в реалізації. Окрім цього Maven дозволяє при компіляції створити jar-файл програми, який потім може бути використаний при контейнеризації проєкту. Спеціальні налаштування також дозволяють додати в сам jar-файл залежності, що використовувались в проєкті, що знов може допомогти при використанні докеру.

Таким чином при виборі інструментарію розробки перевагу було надано мові програмування Java. У якості системи управління базами даних був обраний PostgreSQL, адже він підтримує реалізацію функціоналу, що виходить за межі мови SQL. Також було вирішено реалізувати контейнеризацію проєкту, для того аби мати можливість передачі програмного продукту.

3.2. Опис методів інформаційної системи

На початку роботи програма повинна отримати дані зі споживання електроенергії певного району протягом певного періоду. Це реалізується за допомогою доданої до проєкту залежності postgresql. До бази даних надсилається SQL-запит, результат зберігається у вигляді списку. Функція connectWithRetry() передбачає певну кількість повторних спроб встановити зв’язок у разі невдачі. Це є важливим з огляду на можливу контейнеризацію проєкту, адже база даних може завантажитися пізніше ніж програма, що призведе до неможливості з’єднання. Аналогічним чином реалізовано також отримання даних, що відображаються в інтерфейсі користувача.

```
    List<ConsumptionRecord> result = new ArrayList<>();

    String query = """
        SELECT e.consumption_date, e.consumption_kwh
        FROM energy_consumption e
        JOIN districts d ON d.id = e.district_id
        WHERE d.name = ?
        ORDER BY e.consumption_date;
        """;

    try (Connection conn = connectWithRetry(URL, USER, PASSWORD);
         PreparedStatement stmt = conn.prepareStatement(query))
{

        stmt.setString(1, districtName);
        try (ResultSet rs = stmt.executeQuery()) {
            while (rs.next()) {
                LocalDate date =
rs.getDate("consumption_date").toLocalDate();
                double kwh = rs.getDouble("consumption_kwh");
                result.add(new ConsumptionRecord(date, kwh));
            }
        }

    } catch (SQLException e) {
        e.printStackTrace();
    }


     public static Connection connectWithRetry(String url, String
user, String password) {
    int attempts = 10;
    while (attempts-- > 0) {
        try {
            return DriverManager.getConnection(url, user,
password);
        } catch (SQLException e) {
            System.out.println("Waiting");
            try {
                Thread.sleep(3000);
            } catch (InterruptedException ignored) {}
        }
    }
    throw new RuntimeException("Failed DB connection");
}
```

Після отримання даних виконується обробка, виконується аналіз та прогнозування майбутніх значень.

```
    // Побудова лінійної регресії: y = a + b * x
double sumX = 0, sumY = 0, sumXY = 0, sumXX = 0;

for (int i = 0; i < n; i++) {
    double x = i;
    double y = series.get(i);
    sumX += x;
    sumY += y;
    sumXY += x * y;
    sumXX += x * x;
}

double b = (n * sumXY - sumX * sumY) / (n * sumXX - sumX * sumX);
double a = (sumY - b * sumX) / n;

double nextX = n;
return a + b * nextX;
```

Як було зазначено в розділі 1.3 коефіцієнти a і b визначаються за методом найменших квадратів. Крім регресійного аналізу для прогнозування застосовується також статистична модель ARIMA, яку було реалізовано за допомогою залежності.

```
    int forecastSize = 10;
int p = 3;
int d = 0;
int q = 3;
int P = 1;
int D = 1;
int Q = 0;
int m = 0;
ArimaParams params = new ArimaParams(p, d, q, P, D, Q, m);
ForecastResult arimaForecast = dataProcessing.arima(timeSeries,
forecastSize, params);
for (int i = 0; i < forecastSize; i++)
    System.out.printf("\nARIMA forecast: %.2f",
arimaForecast.getForecast()[i]);
```

Крім цього також було враховано довірчий інтервал цієї моделі, який демонструє те, в яких межах можуть коливатися відхилення реальних даних від спрогнозованих. З огляду на важливість цієї інформації довірчі інтервали також було додано на грфік.

Було реалізовано обчислення автокореляції та ковзного середнього для усунення впливу випадкових коливань і для того аби відстежувати фактор циклічності.

```
     public static List<Double> autocorrelation(List<Double> data)
{
    int lag = 0;
    List<Double> result = new ArrayList<>();
    List<Double> x;
    List<Double> y;
    double meanX, meanY, numerator, denomX, denomY;

    while(data.size() - 1 > lag){
        x = data.subList(0, data.size() - lag);
        y = data.subList(lag, data.size());
        meanX =
x.stream().mapToDouble(Double::doubleValue).average().orElse(0.0
);
        meanY =
y.stream().mapToDouble(Double::doubleValue).average().orElse(0.0
);

        numerator = 0; denomX = 0; denomY = 0;
        for (int i = 0; i < x.size(); i++) {
            double dx = x.get(i) - meanX;
            double dy = y.get(i) - meanY;
            numerator += dx * dy;
            denomX += dx * dx;
            denomY += dy * dy;
        }
        result.add(numerator / Math.sqrt(denomX * denomY));
        lag ++;
    }
    return result;
}
```

```
     public static List<Double> movingAverage(List<Double> data,
int windowSize) {
    List<Double> result = new ArrayList<>();
    for (int i = 0; i <= data.size() - windowSize; i++) {
        double sum = 0;
        for (int j = 0; j < windowSize; j++) {
            sum += data.get(i + j);
        }
        result.add(sum / windowSize);
    }
    return result;
}
```

Після завершення обробки даних, динаміка споживання електроенергії, обчислені показники, прогнози, а також їхній довірчий інтервал подаються на графіку для того аби уможливити аналіз положення та впровадження плану раціонального розподілу енергії.

```
     XYSeriesCollection dataset = new XYSeriesCollection();
dataset.addSeries(actualSeries);
dataset.addSeries(forecastSeries);
dataset.addSeries(upperSeries);
dataset.addSeries(lowerSeries);
dataset.addSeries(averageSeries);
dataset.addSeries(autocorrelationSeries);

JFreeChart chart = ChartFactory.createXYLineChart(
        "Споживання електроенергії",
        "День",
        "кВт*год",
        dataset,
        PlotOrientation.VERTICAL,
        true, true, false);

XYPlot plot = chart.getXYPlot();
```

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було досягнуто поставлену мету - розроблено інформаційну систему, що забезпечує раціональне планування та ефективний розподіл електроенергії в умовах її дефіциту для великих міських енергетичних систем.

На основі аналізу предметної області було виявлено ключові фактори, що призводять до енергетичного дефіциту, серед яких технічний знос обладнання, зростання енергоспоживання, геополітичні загрози та відсутність належного прогнозування навантаження. Досліджено структуру сучасних енергосистем, зокрема їхню залежність від централізованих та децентралізованих джерел енергії.

У ході дослідження розглянуто сучасні методи раціонального планування розподілу електроенергії, серед яких важливе місце займають математичні моделі оптимізації, алгоритми прогнозування та методи машинного навчання. Визначено, що поєднання алгоритмічних підходів з сучасними інформаційними технологіями дозволяє значно підвищити ефективність управління енергоспоживанням.

Спроєктована та реалізована інформаційна система охоплює процеси збору, зберігання, обробки та візуалізації даних. Було визначено вимоги до системи, обрано відповідні інструменти розробки, реалізовано базовий функціонал із підтримкою прийняття рішень щодо розподілу ресурсів у режимі, наближеному до реального часу.

Результати тестування підтвердили здатність розробленої системи адаптуватися до зміни вхідних параметрів та надавати користувачеві обґрунтовані рекомендації щодо енергорозподілу, з урахуванням пріоритетності об'єктів та обмеженості доступних ресурсів.

Практичне значення роботи полягає у можливості впровадження запропонованої системи в реальні енергетичні інфраструктури з метою

підвищення стійкості, прозорості та керованості розподілу електроенергії, особливо у кризових умовах або в періоди пікового навантаження.

Отже, результати дослідження можуть бути використані як основа для подальшого розвитку адаптивних та інтелектуальних систем енергетичного менеджменту в міському середовищі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Zemmouchi-Ghomari, L. The basic concepts of information systems. Intechopen. 2021. 19 с.

2. Florida`s stem university. URL: https://online.fit.edu/degrees/undergraduate/cis/bachelor-science-computer-information-systems/5-types-of-information-systems/ (Дата звернення: 29.05.2025).

3. ScienceDirect. URL: https://www.sciencedirect.com/science/article/abs/pii/S2210670720305916 (Дата звернення: 02.06.2025).

4. Аналіз числових рядів / Яровий А. Т., Страхов Є. М. Одеса 2019. 109 с.

5. A Low-Cost Energy Monitoring System with Universal Compatibility and Real-Time Visualization for Enhanced Accessibility and Power Savings. / Khan, H.R. та ін. Sustainability 2024. URL: https://doi.org/10.3390/su16104137.

6. Шкрабець Ф. П. Основи електропостачання 2012. 463 с.

7. Всеукраїнська енергетична асамблея. URL: https://uaea.com.ua/dysp/ee-cons.html (Дата звернення: 05.06.2025).

8. Jeffrey S. Simonoff. Handbook of Regression Analysis. Hoboken, 2013. с. 236.

9. Reddy B. S., Assenza G. Energy Efficiency and Climate Change. Modelling, Measurement and Policy. Berlin, Springer, 2009. 250 с.

10. Laudon K. C., Laudon J. P. Management Information Systems: Managing the Digital Firm. 16th ed. Boston, Pearson, 2020. 648 с.

11. Ozturk S., Ozturk F. Forecasting energy consumption of Turkey by ARIMA model // Journal of Asian Scientific Research, 2018, 8(2). с.52–60. URL: https://archive.aessweb.com/index.php/5003/article/download/3874/6093

12. Draper R., Smith H., Applied Regression Analysis 3rd ed., New York, Wiley, 1998. 659 с.

13. Early Warning Systems for World Energy Crises // Sustainability. 2024, Vol.16 (6). с. 1–20.

14. A Survey on Power System Blackout and Cascading Events: Research Motivations and Challenges // MDPI Energies, 2019, Vol.12 (4). с. 1–25.
15. URL: https://www.geni.org/globalenergy/library/media_coverage/EnergyCentral/EnergyPulse/NortheastBlackout/SystemBlackoutCausesAndCures/index.shtml?utm_source=chatgpt.com (Дата звернення: 08.06.2025).

ДОДАТОК А. ПРИКЛАДИ ВИХІДНОГО КОДУ

Лістинг 1 – Main

```
     package org.example;

import com.workday.insights.timeseries.arima.struct.ArimaParams;
import
com.workday.insights.timeseries.arima.struct.ForecastResult;
import org.example.ui.userInterface;

import java.util.List;

public class Main {

    public static void main(String[] args) {
        userInterface.main();
        List<Double> timeSeries =
dataFetching.getConsumptionByDistrict("Центральний");
        System.out.printf("Time series: " + timeSeries);
        double max = 210;

        if (timeSeries.size() < 2) {
            System.out.println("Not enough data.");
            return;
        }

        double avg =
timeSeries.stream().mapToDouble(Double::doubleValue).average().o
rElse(0.0);
        System.out.printf("\nMean value: %.2f\n", avg);

        int window = 2;
        List<Double> movingAvg =
dataProcessing.movingAverage(timeSeries, window);
        System.out.println("Moving average (window = " + window
+ "): " + movingAvg);

        String trend = dataProcessing.trend(timeSeries);
        System.out.println("General trend: " + trend);

        List<Double> correlation =
dataProcessing.autocorrelation(timeSeries);
        System.out.printf("Autocorrelation: " + correlation);

        double regression =
dataProcessing.regressionAnalysis(timeSeries);
        System.out.printf("\nRegression analysis forecast: %.2f",
regression);
```

```
        if(regression > max)
            System.out.printf("\nDeficit predicted");

        int forecastSize = 10;
        int p = 3;
        int d = 0;
        int q = 3;
        int P = 1;
        int D = 1;
        int Q = 0;
        int m = 0;
        ArimaParams params = new ArimaParams(p, d, q, P, D, Q,
m);
        ForecastResult arimaForecast =
dataProcessing.arima(timeSeries, forecastSize, params);
        for (int i = 0; i < forecastSize; i++)
            System.out.printf("\nARIMA forecast: %.2f",
arimaForecast.getForecast()[i]);

        return;
    }
}
```

Лістинг 2 – Клас dataFetching

```
     package org.example;

import java.sql.*;

import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;

public class dataFetching {
    private static final String URL =
"jdbc:postgresql://localhost:5432/postgres";
    private static final String USER = "postgres";
    private static final String PASSWORD = "root";

    /*static String url = System.getenv("DB_URL");
    static String user = System.getenv("DB_USER");
    static String pass = System.getenv("DB_PASSWORD");*/

    public static List<Double> getConsumptionByDistrict(String
districtName) {
        List<Double> result = new ArrayList<>();

        String query = """
            SELECT e.consumption_date, e.consumption_kwh
            FROM energy_consumption e
            JOIN districts d ON d.id = e.district_id
            WHERE d.name = ?
            ORDER BY e.consumption_date;
            """;

        try (Connection conn = connectWithRetry(URL, USER,
PASSWORD);
             PreparedStatement stmt =
conn.prepareStatement(query)) {

            stmt.setString(1, districtName);
            try (ResultSet rs = stmt.executeQuery()) {
                while (rs.next()) {
                    double kwh = rs.getDouble("consumption_kwh");
                    result.add(kwh);
                }
            }

        } catch (SQLException e) {
            e.printStackTrace();
        }

        return result;
    }
```

```
    public static List<ConsumptionRecord> getRecords(String
districtName) {
        List<ConsumptionRecord> result = new ArrayList<>();

        String query = """
            SELECT e.consumption_date, e.consumption_kwh
            FROM energy_consumption e
            JOIN districts d ON d.id = e.district_id
            WHERE d.name = ?
            ORDER BY e.consumption_date;
            """;

        try (Connection conn = connectWithRetry(URL, USER,
PASSWORD);
             PreparedStatement stmt =
conn.prepareStatement(query)) {

            stmt.setString(1, districtName);
            try (ResultSet rs = stmt.executeQuery()) {
                while (rs.next()) {
                    LocalDate date =
rs.getDate("consumption_date").toLocalDate();
                    double kwh = rs.getDouble("consumption_kwh");
                    result.add(new ConsumptionRecord(date, kwh));
                }
            }

        } catch (SQLException e) {
            e.printStackTrace();
        }

        return result;
    }

    public static Connection connectWithRetry(String url, String
user, String password) {
        int attempts = 10;
        while (attempts-- > 0) {
            try {
                return DriverManager.getConnection(url, user,
password);
            } catch (SQLException e) {
                System.out.println("Waiting");
                try {
                    Thread.sleep(3000);
                } catch (InterruptedException ignored) {}
            }
        }
        throw new RuntimeException("Failed DB connection");
    }
```

```
    public static class ConsumptionRecord {
        private LocalDate date;
        private double consumptionKwh;

        public ConsumptionRecord(LocalDate date, double
consumptionKwh) {
            this.date = date;
            this.consumptionKwh = consumptionKwh;
        }

        public LocalDate getDate() {
            return date;
        }

        public double getConsumptionKwh() {
            return consumptionKwh;
        }

        @Override
        public String toString() {
            return date + " : " + consumptionKwh + " кВт·год";
        }
    }
}
```

Лістинг 3 – Клас dataProcessing

```
     package org.example;

import com.workday.insights.timeseries.arima.struct.ArimaParams;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.plot.PlotOrientation;
import org.jfree.chart.plot.XYPlot;
import org.jfree.chart.renderer.xy.XYLineAndShapeRenderer;
import org.jfree.data.xy.XYSeries;
import org.jfree.data.xy.XYSeriesCollection;

import com.workday.insights.timeseries.arima.Arima;
import
com.workday.insights.timeseries.arima.struct.ForecastResult;

import javax.swing.*;
import java.awt.*;
import java.util.ArrayList;
import java.util.List;

public class dataProcessing {
    public static List<Double> movingAverage(List<Double> data,
int windowSize) {
        List<Double> result = new ArrayList<>();
        for (int i = 0; i <= data.size() - windowSize; i++) {
            double sum = 0;
            for (int j = 0; j < windowSize; j++) {
                sum += data.get(i + j);
            }
            result.add(sum / windowSize);
        }
        return result;
    }

    public static String trend(List<Double> data) {
        int up = 0, down = 0;
        for (int i = 1; i < data.size(); i++) {
            if (data.get(i) > data.get(i - 1)) up++;
            else if (data.get(i) < data.get(i - 1)) down++;
        }

        if (up > down) return "Increasing";
        else if (down > up) return "Decreasing";
        else return "Neutral";
    }

    public static List<Double> autocorrelation(List<Double> data)
{
```

```
        int lag = 0;
        List<Double> result = new ArrayList<>();
        List<Double> x;
        List<Double> y;
        double meanX, meanY, numerator, denomX, denomY;

        while(data.size() - 1 > lag){
            x = data.subList(0, data.size() - lag);
            y = data.subList(lag, data.size());
            meanX =
x.stream().mapToDouble(Double::doubleValue).average().orElse(0.0
);
            meanY =
y.stream().mapToDouble(Double::doubleValue).average().orElse(0.0
);

            numerator = 0; denomX = 0; denomY = 0;
            for (int i = 0; i < x.size(); i++) {
                double dx = x.get(i) - meanX;
                double dy = y.get(i) - meanY;
                numerator += dx * dy;
                denomX += dx * dx;
                denomY += dy * dy;
            }
            result.add(numerator / Math.sqrt(denomX * denomY));
            lag ++;
        }
        return result;
    }

    public static double regressionAnalysis(List<Double> series)
{
        int n = series.size();
        if (n < 2) {
            System.out.println("Not enough data.");
            return series.get(n - 1);
        }

        // Побудова лінійної регресії: y = a + b * x
        double sumX = 0, sumY = 0, sumXY = 0, sumXX = 0;

        for (int i = 0; i < n; i++) {
            double x = i;
            double y = series.get(i);
            sumX += x;
            sumY += y;
            sumXY += x * y;
            sumXX += x * x;
        }

        double b = (n * sumXY - sumX * sumY) / (n * sumXX - sumX
* sumX);
```

```
        double a = (sumY - b * sumX) / n;

        double nextX = n;
        return a + b * nextX;
    }

    public static ForecastResult arima(List<Double> series, int
forecastSize, ArimaParams params) {
        double[] data = new double[series.size()];
        for (int i = 0; i < series.size(); i++)
            data[i] = series.get(i);

        ForecastResult forecastResult =
Arima.forecast_arima(data, forecastSize, params);
        return forecastResult;
    }

    public static void visualize(List<Double> series,
ForecastResult forecast, int window, int forecastSize) {
        XYSeries actualSeries = new XYSeries("Енергоспоживання");
        XYSeries forecastSeries = new XYSeries("Прогноз");
        XYSeries upperSeries = new XYSeries("Верхня межа
довірчого інтервалу");
        XYSeries lowerSeries = new XYSeries("Нижня межа
довірчого інтервалу");
        XYSeries averageSeries = new XYSeries("Ковзне середнє
(вікно = " + window + ")");
        XYSeries autocorrelationSeries = new
XYSeries("Автокореляція");
        List<Double> average = movingAverage(series, 3);
        List<Double> autocorrelation = autocorrelation(series);

        for (int i = 0; i < series.size(); i++)
            actualSeries.add(i, series.get(i));

        for (int i = 0; i < forecastSize; i++) {
            forecastSeries.add(series.size() + i,
forecast.getForecast()[i]);
            upperSeries.add(series.size() + i,
forecast.getForecastUpperConf()[i]);
            lowerSeries.add(series.size() + i,
forecast.getForecastLowerConf()[i]);
        }

        for (int i = 0; i < series.size() - window; i++){
            averageSeries.add(i, average.get(i));
            autocorrelationSeries.add(i,  autocorrelation.get(i));
        }

        XYSeriesCollection dataset = new XYSeriesCollection();
        dataset.addSeries(actualSeries);
        dataset.addSeries(forecastSeries);
```

```
        dataset.addSeries(upperSeries);
        dataset.addSeries(lowerSeries);
        dataset.addSeries(averageSeries);
        dataset.addSeries(autocorrelationSeries);

        JFreeChart chart = ChartFactory.createXYLineChart(
                "Споживання електроенергії",
                "День",
                "кВт*год",
                dataset,
                PlotOrientation.VERTICAL,
                true, true, false);

        XYPlot plot = chart.getXYPlot();

        XYLineAndShapeRenderer renderer = new
XYLineAndShapeRenderer();

        renderer.setSeriesLinesVisible(0, true);
        renderer.setSeriesShapesVisible(0, true);
        renderer.setSeriesPaint(0, Color.BLUE);

        renderer.setSeriesLinesVisible(1, true);
        renderer.setSeriesShapesVisible(1, true);
        renderer.setSeriesPaint(1, Color.RED);
        renderer.setSeriesShape(1, new
java.awt.geom.Ellipse2D.Double(-4, -4, 8, 8));

        renderer.setSeriesLinesVisible(2, true);
        renderer.setSeriesShapesVisible(2, true);
        renderer.setSeriesPaint(2, Color.GREEN);

        renderer.setSeriesLinesVisible(3, true);
        renderer.setSeriesShapesVisible(3, true);
        renderer.setSeriesPaint(3, Color.YELLOW);

        renderer.setSeriesLinesVisible(4, true);
        renderer.setSeriesShapesVisible(4, true);
        renderer.setSeriesPaint(4, Color.WHITE);

        renderer.setSeriesLinesVisible(5, true);
        renderer.setSeriesShapesVisible(5, true);
        renderer.setSeriesPaint(5, Color.BLACK);

        plot.setRenderer(renderer);

        JFrame frame = new JFrame("Графік");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.add(new ChartPanel(chart));
        frame.pack();
        frame.setLocationRelativeTo(null);
        frame.setVisible(true);
```

```
        }
}
```

Лістинг 4 – Клас userInterface

```
package org.example.ui;

import com.workday.insights.timeseries.arima.struct.ArimaParams;
import
com.workday.insights.timeseries.arima.struct.ForecastResult;
import org.example.dataFetching;
import org.example.dataProcessing;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.util.List;

public class userInterface extends JFrame {

    private JComboBox<String> districtSelector;
    private JTextArea outputArea;
    private JButton loadButton;

    public userInterface() {
        super("Енергоспоживання по районах");
        initUI();
    }

    private void initUI() {
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setSize(500, 400);
        setLayout(new BorderLayout());

        JPanel topPanel = new JPanel();
        topPanel.add(new JLabel("Оберіть район:"));

        districtSelector = new JComboBox<>(new String[] {
                "Центральний", "Південний", "Північний"
        });
        topPanel.add(districtSelector);

        loadButton = new JButton("Отримати дані");
        loadButton.addActionListener(this::loadData);
        topPanel.add(loadButton);

        add(topPanel, BorderLayout.NORTH);

        outputArea = new JTextArea();
        outputArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(outputArea);
        add(scrollPane, BorderLayout.CENTER);

        setLocationRelativeTo(null);
```

```
        setVisible(true);
    }

    private void loadData(ActionEvent e) {
        String selectedDistrict = (String)
districtSelector.getSelectedItem();
        List<Double> values =
dataFetching.getConsumptionByDistrict(selectedDistrict);
        List<dataFetching.ConsumptionRecord> records =
dataFetching.getRecords(selectedDistrict);
        int window = 2;
        int forecastSize = 10;
        int p = 3;
        int d = 0;
        int q = 3;
        int P = 1;
        int D = 1;
        int Q = 0;
        int m = 0;
        ArimaParams params = new ArimaParams(p, d, q, P, D, Q,
m);
        ForecastResult arimaForecast =
dataProcessing.arima(values, forecastSize, params);
        dataProcessing.visualize(values, arimaForecast, window,
forecastSize);

        outputArea.setText("");
        if (records.isEmpty()) {
            outputArea.append("Немає даних для району: " +
selectedDistrict);
        } else {
            outputArea.append("Енергоспоживання району \"" +
selectedDistrict + "\":\n\n");
            for (dataFetching.ConsumptionRecord r : records) {
                outputArea.append(r.toString() + "\n");
            }
        }
    }

    public static void main() {
        SwingUtilities.invokeLater(userInterface::new);
    }
}
```

Лістинг 5 – Файл pom.xml

```
     <?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
    <artifactId>TimeSeriesAnalysis</artifactId>
    <version>1.0-SNAPSHOT</version>

    <build>
        <plugins>
            <plugin>
                <artifactId>maven-assembly-plugin</artifactId>
                <configuration>
                    <archive>
                        <manifest>

<mainClass>org.example.Main</mainClass>
                        </manifest>
                    </archive>
                    <descriptorRefs>
                        <descriptorRef>jar-with-
dependencies</descriptorRef>
                    </descriptorRefs>
                </configuration>
                <executions>
                    <execution>
                        <id>make-assembly</id>
                        <phase>package</phase>
                        <goals>
                            <goal>single</goal>
                        </goals>
                    </execution>
                </executions>
            </plugin>
        </plugins>
    </build>



    <properties>
        <maven.compiler.source>18</maven.compiler.source>
        <maven.compiler.target>18</maven.compiler.target>
    </properties>

    <dependencies>
        <dependency>
```

```
            <groupId>org.jfree</groupId>
            <artifactId>jfreechart</artifactId>
            <version>1.5.3</version>
        </dependency>
        <dependency>
            <groupId>com.workday</groupId>
            <artifactId>timeseries-forecast</artifactId>
            <version>1.1.1</version>
        </dependency>
        <dependency>
            <groupId>org.postgresql</groupId>
            <artifactId>postgresql</artifactId>
            <version>42.7.2</version>
        </dependency>
    </dependencies>
</project>
```

ДОДАТОК Б. РЕЗУЛЬТАТИ ОБРОБКИ ДАНИХ

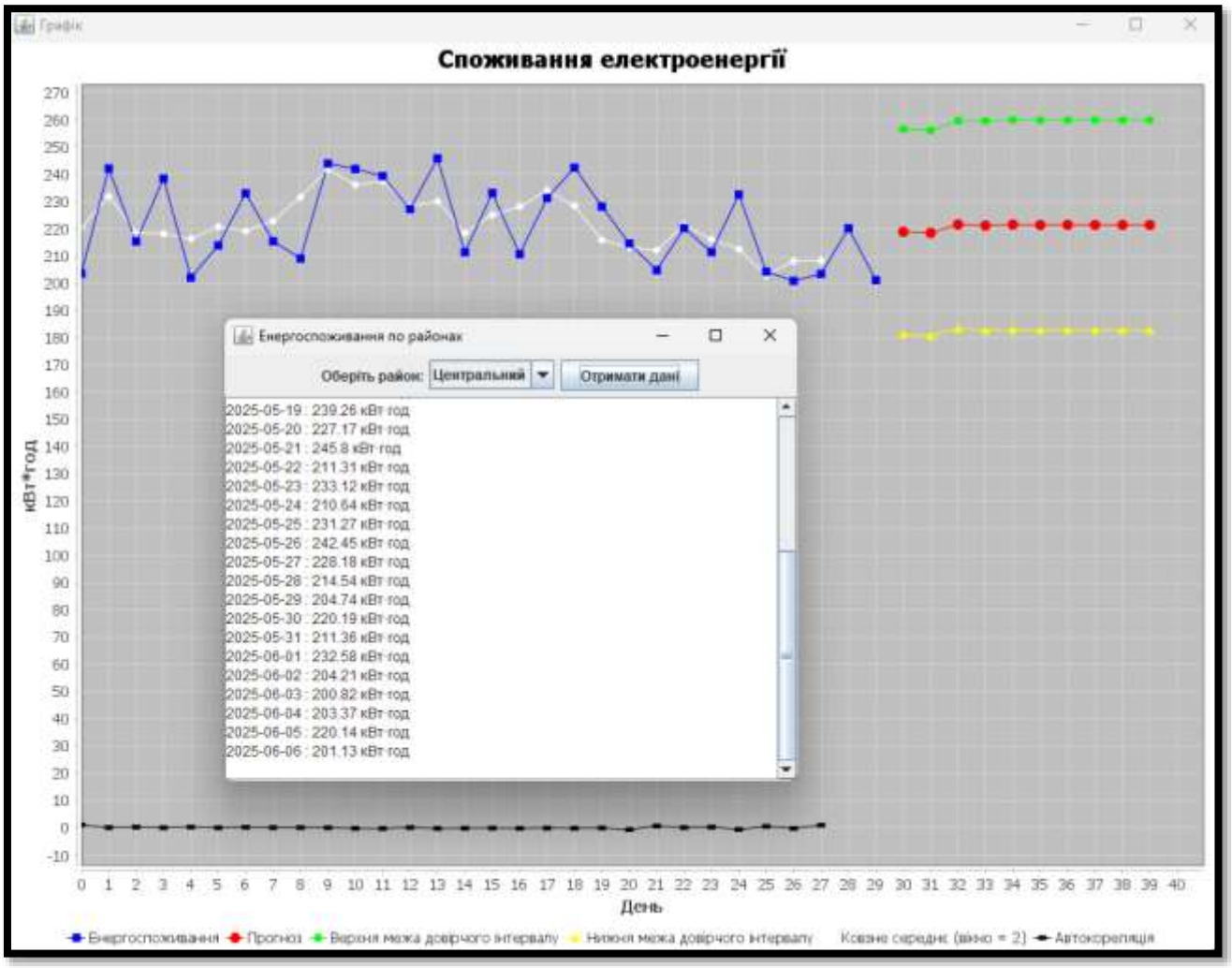

Рисунок Б.1 – Графік за результатами обробки даних

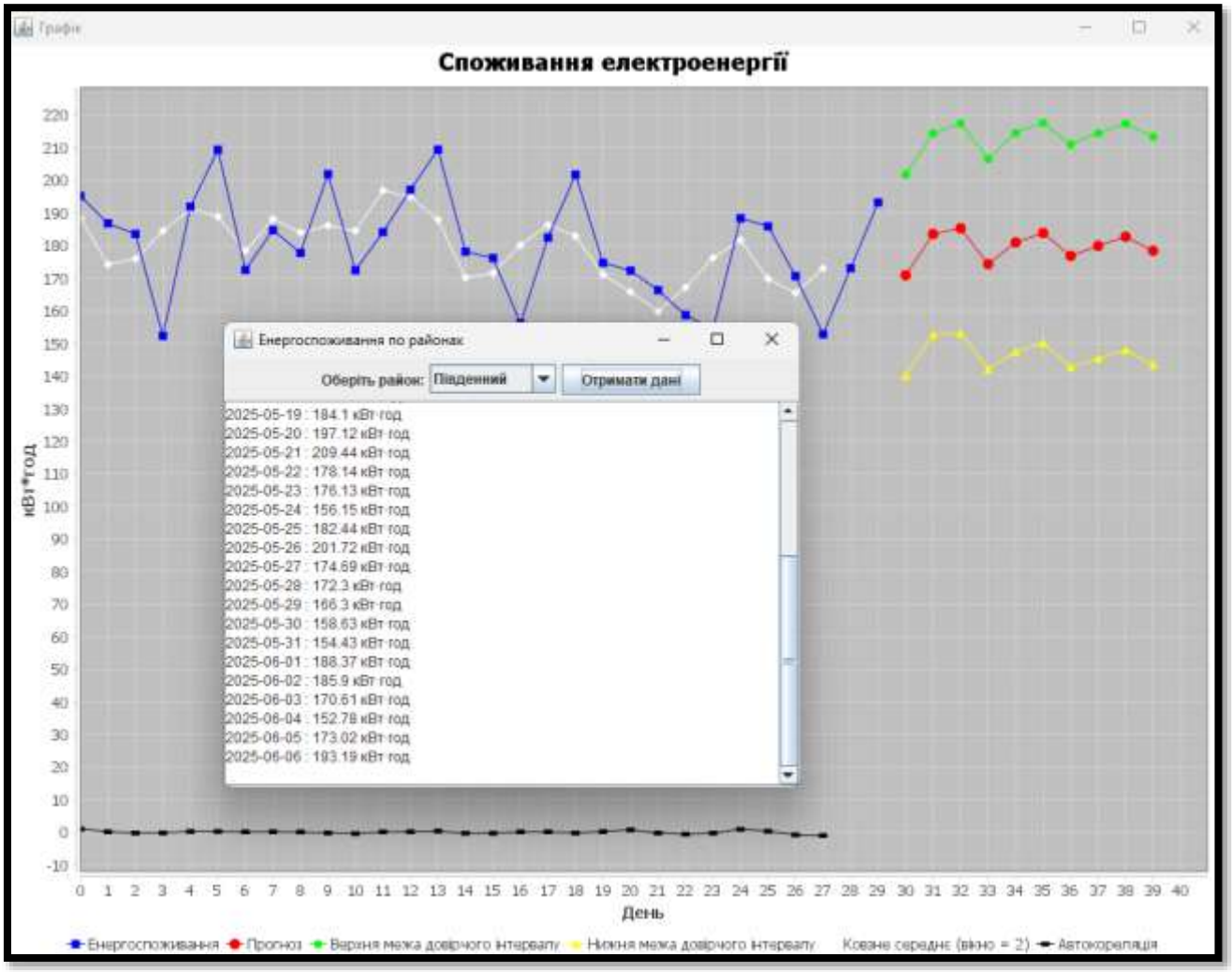

Рисунок Б.2 – Графік за результатами обробки даних

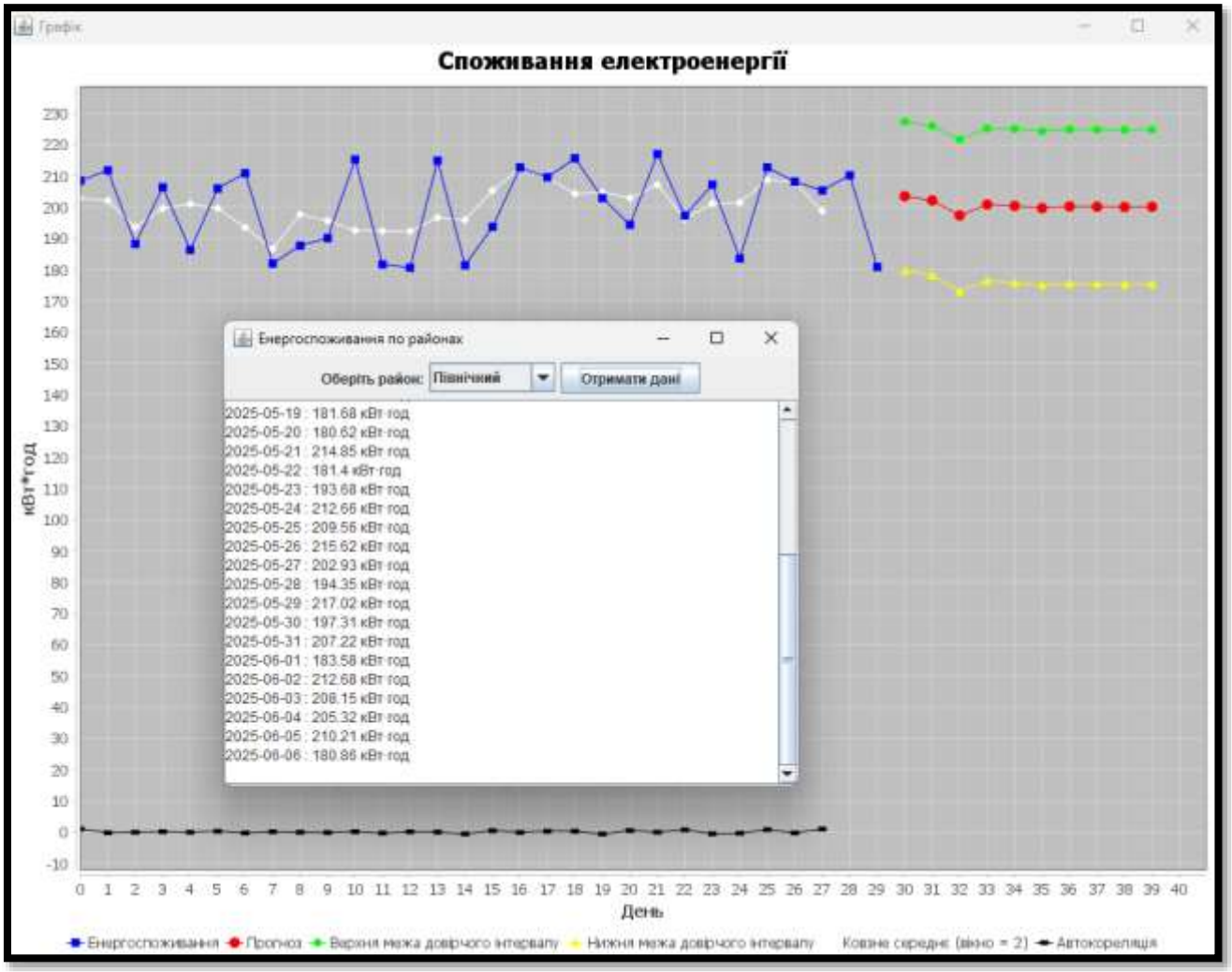

Рисунок Б.3 – Графік за результатами обробки даних