

Міністерство освіти і науки України
Харківський національний університет імені В.Н. Каразіна
Факультет комп'ютерних наук

Рекомендовано до захисту:

протокол засідання кафедри №

___ від ___.____.2025 р.

Завідувач кафедри ІПСіТ

_____ Олешко О. І.

(підпис)

(прізвище та ініціали)

Пояснювальна записка

до дипломної роботи бакалавра

на тему Розробка сайту для моніторингу та діагностування стану дітей із відхиленнями зросту

Захищено на засіданні ЕК № _____

протокол № _____ від _____.____.2019 р.

Оцінка _____ / _____

Голова ЕК

(підпис)

(прізвище та ініціали)

Виконав: студент 4 курсу, групи КС- 41

Спеціальності:

122 Комп'ютерні науки

(шифр і назва спеціальності підготовки)

Мірошниченко Марія Олександрівна

(прізвище, ім'я та по батькові, підпис)



Керівник канд. тех. наук, доцент, Олешко О. І.

(ступень, звання, прізвище та ініціали, підпис)

Рецензент старший викладач Коршенко В.С

(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 інтелектуальних програмних систем і технологій

Рівень вищої освіти (освітньо-кваліфікаційний рівень) бакалавр

Спеціальність 122 Комп'ютерні науки

ЗАТВЕРДЖУЮ

В.о. завідувача кафедри ІПСіТ

_____ Олег ОЛЕШКО
підпис ім'я, прізвище

“ _____ ” _____ 2025 року

**З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ**

Мірошниченко Марії Олександрівни
(прізвище, ім'я, по батькові студента)

1. Тема роботи: «Розробка сайту для моніторингу та діагностування стану дітей із відхиленнями зросту»

керівник роботи канд. тех. наук, доцент, Олешко О. І.
(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 4 червня 2025 року

3. Перелік питань, які потрібно розробити:

Ознайомитися з особливостями цифрового моніторингу зросту дітей, включаючи міжнародні нормативи фізичного розвитку, провести аналіз існуючих веб-рішень для медичного моніторингу дітей, визначити вимоги до функціональності для дітей, що проходять гормональну терапію, розробити архітектуру веб-платформи на основі фреймворку Django, реалізувати функціональні модулі: реєстрація, управління дітьми, база норм, побудова графіків, нагадування, система порад, провести тестування веб-додатку, перевірити точність графіків та рекомендацій.

4. План роботи

№ з/п	Назви етапів роботи
1	Формулювання мети та повного переліку завдань для виконання кваліфікаційної роботи (КР)
2	Пошук та аналіз інформаційних джерел за темою КР (норми зросту, підхід до моніторингу, гормональна терапія).
3	Визначення вимог до функціональності системи та специфіки підтримки дітей на терапії.
4	Розробка моделі веб-платформи та архітектури проєкту.
5	Програмна реалізація модулів (аутентифікація, моніторинг, база норм, система порад, графіки, нагадування).
6	Проведення тестування веб-сайту та перевірка відображення даних на ростограми
7	Підсумок отриманих результатів та оформлення пакету документів для процедури захисту КР в екзаменаційній комісії ННІ КН та ІІІ.

5. Дата видачі завдання 10 жовтня 2025 року

Студент



(підпис)

М. О. Мірошніченко

(ініціали, прізвище)

Керівник роботи

(підпис)

О. І. Олешко

(ініціали, прізвище)

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи містить: 56 сторінок, 3 розділи, 24 рисунки, 5 таблиць, 1 додаток, 13 джерел за переліком посилань.

Робота присвячена створенню веб-додатку для батьків дітей із порушеннями зросту, що дозволяє здійснювати моніторинг фізичних показників, порівнювати їх з міжнародними та національними нормативами, отримувати персоналізовані рекомендації та нагадування.

Метою роботи є розробка локалізованої цифрової платформи, що дає змогу батькам зручно фіксувати та аналізувати зростові дані дитини.

У процесі дослідження використано методи порівняльного аналізу існуючих онлайн-інструментів моніторингу зросту, проектування архітектури веб-додатку за шаблоном MVT, а також розробку програмного забезпечення з використанням фреймворку Django, мови програмування Python, бібліотек Matplotlib/Plotly для візуалізації та SQLite для зберігання даних.

У результаті реалізовано веб-додаток із модульною архітектурою, підтримкою багатьох країн, можливістю побудови графіків. Особливістю рішення є адаптація під потреби користувачів, що знаходяться за кордоном, і використання українських нормативів розвитку.

Результати можуть бути використані в медичних закладах, навчальних установах, а також батьками для самостійного цифрового моніторингу розвитку дитини.

Ключові слова: ЗРОСТОВІ НОРМИ, ВЕБ-ДОДАТОК, DJANGO, MONITORING, ДИТИНА, ГРАФІКИ, PYTHON, SQLite, УКРАЇНА, ВІЗУАЛІЗАЦІЯ, ВООЗ, РОЗРОБКА, КОРИСТУВАЧ

ANNOTATION

The explanatory note to the qualification project includes: 56 pages, 3 sections, 24 figures, 5 tables, 3 appendices, and 24 references.

The project focuses on the development of a web application for parents of children with growth disorders. The application enables the convenient recording of physical indicators, comparison with international and national standards, personalized recommendations, and reminder functionality.

The aim of the project is to create a localized digital platform for monitoring children's height development with support for Ukrainian standards.

Research methods include comparative analysis of existing growth monitoring tools, architectural design using the MVT pattern, and software implementation using Django framework, Python language, data visualization libraries (Matplotlib/Plotly), and SQLite database.

The result is a modular web application that supports country-specific norms, allows for historical data visualization, and provides user-friendly tools for parents and caregivers. One of the key features is adaptability for users residing abroad and the use of Ukrainian development norms.

The solution may be used in medical and educational institutions as well as by parents for at-home digital child growth monitoring.

Keywords: GROWTH NORMS, WEB APPLICATION, DJANGO, MONITORING, CHILD, CHARTS, PYTHON, SQLITE, REMINDER SYSTEM, UKRAINE, VISUALIZATION, WHO, DEVELOPMENT, USER

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ ТА СКОРОЧЕНЬ.....	5
ВСТУП	6
1 ТЕОРЕТИЧНА ЧАСТИНА	8
1.1 Постановка задачі, об'єкт, предмет та мета	8
1.2 Огляд літератури за темою роботи.....	8
1.3 Розгляд аналогів	9
1.4 Визначення функціональних вимог	12
1.5 Архітектура системи	14
2 ПРАКТИЧНА ЧАСТИНА	22
2.1 Реалізація веб-додатку	22
2.2 Робота з базою даних	35
2.3 Робота системи у реальному режимі.....	39
2.4 Тестування	44
2.5 Висновки до розділу	46
3 ПІДСУМКОВИЙ АНАЛІЗ І ПЕРСПЕКТИВИ	48
3.1 Порівняння з поставленими завданнями	48
3.2 Обмеження та пропозиції розвитку.....	48
ВИСНОВКИ.....	53
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ	55
ДОДАТОК А – ЛІСТИНГИ ПРОГРАМНОГО КОДУ	57
ДОДАТОК Б – ГРАФІК ЗРОСТУ	64

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ ТА СКОРОЧЕНЬ

1. АДМІН-ПАНЕЛЬ – інтерфейс керування в Django для модераторів або розробників
2. API – Application Programming Interface – інтерфейс прикладного програмування, набір правил для взаємодії між програмними компонентами.
3. БД – база даних
4. ВООЗ – Всесвітня організація охорони здоров'я
5. ГР – графік розвитку (в контексті відображення зросту)
6. ORM – Object-Relational Mapping – спосіб взаємодії з базою даних через об'єктно-орієнтовану модель
7. СУБД – система управління базами даних
8. CSV – Comma-Separated Values – формат табличних даних, розділених комами
9. HTML – HyperText Markup Language – мова розмітки для створення веб-сторінок
10. HTTP – HyperText Transfer Protocol – протокол передавання даних у веб
11. JSON – JavaScript Object Notation – формат обміну структурованими даними
12. MVT – Model-View-Template – архітектурний шаблон, що використовується у Django
13. Ditina – модель, що описує основні атрибути дитини в базі даних
14. Measurement – модель для збереження вимірювань у системі
15. см – сантиметр – одиниця довжини
16. рік – одиниця часу для визначення віку
17. дата/час – формат запису для збереження моментів створення та оновлення об'єктів

ВСТУП

Стан здоров'я дітей є одним із головних індикаторів соціального добробуту суспільства. Серед найважливіших параметрів фізичного розвитку дитини — зріст та вага, які мають велике значення для діагностики потенційних захворювань або відхилень у розвитку [1]. Особливої уваги потребують діти із затримкою або порушеннями росту, які можуть бути як наслідком генетичних чинників, так і результатом гормонального дисбалансу. На сьогодні контроль росту часто здійснюється лікарями вручну за допомогою друкованих графіків, де відмічаються вимірювання дитини та порівнюються з нормами, що створює додаткові труднощі для батьків в умовах цифровізації охорони здоров'я [2].

У сучасній медичній практиці, як в Україні, так і за кордоном, все частіше застосовуються веб-технології для моніторингу фізіологічних параметрів людини. Проте більшість існуючих сервісів або не адаптовані до українських нормативів, або надто загальні й не орієнтовані на потреби батьків, які мають дітей із відхиленнями росту. У той час як офіційні паперові документи залишаються необхідною частиною медичної практики — відповідно до чинних норм МОЗ, — цифрові сервіси можуть значно підвищити зручність щоденного користування. Можливість мати усі записи під рукою, швидко переглядати графіки, бачити зміни дуже зручно для батьків. Електронна версія може стати ефективним доповненням до паперової форми. На жаль, на українському інформаційному ринку майже відсутні якісні локалізовані рішення, які задовольняли б ці потреби.

Проблема, яку вирішує дана робота, полягає у відсутності комплексного, локалізованого, зручного у користуванні та доступного інструменту для онлайн-моніторингу зросту дітей. Особливо це важливо для батьків, чії діти перебувають на спеціальному лікуванні, оскільки своєчасне виявлення аномалій у темпах росту дозволяє ефективно реагувати та коригувати терапію.

У межах дипломної роботи розроблено веб-сайт для моніторингу фізичного розвитку дітей, зокрема — росту, з можливістю порівняння з нормативами. Платформа дозволяє батькам вести облік даних, переглядати зведення у вигляді графіків, отримувати автоматичні рекомендації та нагадування. Для дітей, що проходять гормональну терапію, реалізовано персоналізовані поради.

Актуальність теми зумовлена поєднанням двох чинників: недостатньою цифровізацією дитячої медичної статистики в Україні та зростаючою потребою у гнучких онлайн-сервісах, що підтримують індивідуальні траєкторії розвитку дитини. В новизні роботи закладено створення універсального ресурсу, орієнтованого не лише на середньостатистичну норму, а й на спеціальні категорії користувачів, з підтримкою багатоетапного лікування і функціональністю, яка може бути доповнена рекомендаційними сервісами.

Результатом виконаної роботи є веб-застосунок на основі фреймворку Django, що реалізує функціональні модулі реєстрації, управління обліковими записами дітей, базу норм росту, графіки та систему рекомендацій.

1 ТЕОРЕТИЧНА ЧАСТИНА

1.1 Постановка задачі, об'єкт, предмет та мета

Мета дипломної роботи полягає у створенні сайту для батьків дітей із порушеннями росту, який дозволить зручно вносити дані про фізичні показники, порівнювати їх з нормативами та отримувати рекомендації. Об'єктом дослідження є фізичний розвиток дітей. Предметом є цифрові методи фіксації, аналізу та інтерпретації показників зросту в рамках веб-застосунку.

Завдання дослідження:

- провести аналіз існуючих рішень у сфері моніторингу росту;
- визначити вимоги до функціональності веб-застосунку;
- реалізувати веб-платформу із сучасним інтерфейсом користувача;
- забезпечити підтримку персоналізованих порад та адаптації для дітей, що проходять гормональну терапію;
- протестувати систему на надійність та точність роботи.

1.2 Огляд літератури за темою роботи

Моніторинг фізичного розвитку дітей є ключовим аспектом педіатричної практики, що дозволяє своєчасно виявляти відхилення від норми та запобігати можливим захворюванням. Одним з основних методів оцінки фізичного розвитку є використання центильних таблиць, які дозволяють порівнювати індивідуальні антропометричні показники дитини з нормативними значеннями для певного віку та статі [3], [4].

У міжнародній практиці широко використовуються стандарти фізичного розвитку, розроблені Всесвітньою організацією охорони здоров'я (ВООЗ). Ці стандарти були створені на основі даних, зібраних у багатьох країнах, і відображають оптимальні умови росту дітей. Вони включають графіки для різних вікових груп та статей, що дозволяє здійснювати точний моніторинг розвитку дитини.

У Польщі також існують онлайн-калькулятори для оцінки фізичного розвитку дітей, які базуються на центильних таблицях. Наприклад, калькулятор на сайті Medme.pl дозволяє батькам самостійно перевіряти та регулярно моніторити розвиток своєї дитини, вводючи актуальні дані та отримуючи результати у вигляді центильних показників.

Незважаючи на наявність таких інструментів, більшість існуючих рішень не підтримують українські нормативи, що створює труднощі для батьків в Україні. Однак зараз багато українців знаходяться за кордоном, і для них також важливо мати доступ до інструментів, які враховують українські норми розвитку, а також можливість порівняння з іншими країнами. У зв'язку з цим, актуальним є створення локалізованого, зручного у користуванні та доступного інструменту для онлайн-моніторингу зросту дітей, який би враховував українські нормативи та індивідуальні особливості розвитку дитини, а також дозволяв би вибір між різними країнами для більш точних порівнянь. Такий інструмент дозволив би батькам мати всі необхідні дані під рукою, що є особливо важливим в умовах цифровізації охорони здоров'я. Також важливо, щоб такий ресурс був доступний українською мовою.

1.3 Розгляд аналогів

Моніторинг фізичного розвитку дітей є важливою складовою педіатрії, і в даний час існує безліч онлайн-ресурсів, що пропонують калькулятори для оцінки зросту та ваги дітей. Однак більшість з цих рішень обмежуються лише підтримкою міжнародних стандартів або не враховують українські нормативи. Крім того, багато з цих інструментів не пропонують персоналізованих рекомендацій або інтеграції з календарем для автоматичних нагадувань.

1. Medme.pl — Kalkulator Siatek Centylowych

Medme.pl [5] є польським онлайн-калькулятором для оцінки фізичного розвитку дітей. Веб-сайт дозволяє батькам вводити дані про вік, стать, зріст та

вагу дитини, а також отримувати результат у вигляді центильних показників. Цей ресурс використовує польські нормативи для визначення відповідності фізичного розвитку дитини.

Переваги:

- Простий та інтуїтивно зрозумілий інтерфейс.
- Безкоштовний доступ до калькулятора.
- Підтримка польських нормативів для оцінки розвитку.

Недоліки:

- Відсутність підтримки українських нормативів, що є суттєвим обмеженням для українських користувачів.
- Відсутність персоналізованих рекомендацій, що не дозволяє забезпечити індивідуальний підхід до моніторингу розвитку.
- Обмежена функціональність: нагадувань.

More Than Height — Child Growth Tracker [6]

More Than Height — це міжнародний онлайн-калькулятор для моніторингу зросту дітей, який використовує стандарти Всесвітньої організації охорони здоров'я (ВООЗ). Цей калькулятор дозволяє порівнювати зрост дитини з міжнародними стандартами розвитку для різних вікових груп та статей.

Переваги:

- Використання міжнародних стандартів ВООЗ для оцінки зросту та ваги.
- Простий та зручний інтерфейс.
- Безкоштовний доступ до основних функцій.

Недоліки:

- Відсутність підтримки українських нормативів, що є суттєвим обмеженням для українських користувачів.

- Відсутність персоналізованих рекомендацій, що не дозволяє забезпечити індивідуальний підхід до моніторингу розвитку.

- Обмежена функціональність: нагадувань.

2. CDC — Child and Teen BMI Calculator [4]

Цей калькулятор, розроблений Центрами контролю та профілактики захворювань (CDC) США, дозволяє оцінити індекс маси тіла (ІМТ) дітей та підлітків. Він використовує стандарти CDC для різних вікових груп та статей.

Переваги:

- Використання офіційних американських стандартів для оцінки фізичного розвитку.
- Зручний для користувачів з США та інших країн, що орієнтуються на американські нормативи.

Недоліки:

- Відсутність українських нормативів, що робить ресурс малоефективним для користувачів з України.
- Обмежена функціональність: калькулятор орієнтований лише на оцінку ІМТ, без можливості побудови графіків росту чи порівняння з іншими стандартами.
- Відсутність персоналізованих рекомендацій та функцій для нагадувань про вимірювання.

3. Recepta.pl — Kalkulator Wagi Dziecka [7]

Recepta.pl пропонує калькулятор для оцінки ваги дітей за допомогою центильних таблиць та індексу маси тіла (ІМТ). Цей калькулятор використовується переважно в Польщі і підтримує польські нормативи.

Переваги:

- Підтримка польських нормативів для оцінки ваги та розвитку дітей.
- Інтуїтивно зрозумілий інтерфейс.

Недоліки:

- Не підтримує українські нормативи.
- Відсутність функцій персоналізованих рекомендацій або нагадувань.

1.4 Визначення функціональних вимог

Функціональні вимоги визначають, що саме має виконувати система, тобто її поведінку у відповідь на вхідні дані. Для веб-додатку спостереження за фізичним розвитком дітей, ці вимоги охоплюють можливості введення, обробки, зберігання та візуалізації даних, а також взаємодії користувача з системою. Вони були сформульовані на основі аналізу аналогічних рішень, потреб цільової аудиторії та технічних можливостей обраної платформи.

Основні функціональні вимоги

1. Реєстрація та аутентифікація користувачів:

Система повинна забезпечувати можливість створення облікових записів для батьків та інших користувачів, а також безпечний вхід до системи з використанням пароля.

2. Створення та управління профілями дітей:

Користувачі повинні мати змогу додавати, редагувати та видаляти профілі дітей, включаючи основну інформацію.

3. Введення та зберігання даних про фізичний розвиток:

Система повинна дозволяти введення показників росту дітей з можливістю зберігання цих даних у базі даних для подальшого аналізу.

4. Порівняння з нормативними показниками:

Веб-додаток має автоматично порівнювати введені дані з національними та міжнародними нормативами (наприклад, стандартами ВООЗ) та надавати відповідні висновки.

5. Візуалізація даних:

Система повинна будувати графіки росту дітей на основі введених даних.

6. Надання персоналізованих рекомендацій:

На основі аналізу даних, система повинна генерувати індивідуальні поради щодо фізичного розвитку дитини, враховуючи можливі відхилення від норми.

7. Імпорт та експорт даних:

Користувачі повинні мати можливість експортувати введені дані у форматі CSV або PDF, а також імпортувати дані.

8. Адаптивний інтерфейс користувача:

Інтерфейс веб-додатку має бути адаптованим для різних пристроїв та забезпечувати зручну навігацію.

9. Безпека та конфіденційність даних:

Веб-додаток має забезпечувати захист персональних даних користувачів відповідно до чинного законодавства про захист інформації.

У таблиці 1.1 наведено відповідність між сформульованими функціональними вимогами та реалізованим функціоналом веб-додатку:

Таблиця 1.1 – Відповідність функціональних вимог реалізованому функціоналу веб-додатку

№	Функціональна вимога	Реалізований функціонал веб-додатку
1	Реєстрація та аутентифікація користувачів	Створення облікових записів, вхід з паролем
2	Створення та управління профілями дітей	Додавання, редагування, видалення профілів
3	Введення та зберігання даних	Введення показників росту та ваги, збереження в базі даних
4	Порівняння з нормативами	Автоматичне порівняння з стандартами ВООЗ
5	Візуалізація даних	Побудова графіків росту та ваги
6	Персоналізовані рекомендації	Генерація порад на основі аналізу даних
7	Імпорт та експорт даних	Експорт у CSV/PDF, імпорт з інших джерел
8	Адаптивний інтерфейс	Підтримка різних пристроїв, зручна навігація
9	Система нагадувань	Надсилання нагадувань користувачам
10	Безпека та конфіденційність	Захист персональних даних користувачів відповідно до чинного законодавства про захист інформації

Ці функціональні вимоги були визначені з урахуванням потреб користувачів та технічних можливостей сучасних веб-технологій. Їх реалізація забезпечить ефективне використання веб-додатку для моніторингу фізичного розвитку дітей.

1.5 Архітектура системи

Архітектура веб-додатку — це концептуальна модель, яка визначає логічну структуру програмного забезпечення, його компоненти, зв'язки між ними та взаємодію з користувачем і зовнішніми сервісами. Архітектура є фундаментальною частиною процесу проектування, оскільки вона забезпечує основу для прийняття рішень щодо реалізації, масштабування, підтримки та розширення системи.

Однією з ключових вимог є розділення відповідальностей між компонентами, що дозволяє забезпечити зручність у підтримці, тестуванні й масштабуванні коду. Для цього широко використовуються архітектурні шаблони — наприклад, MVC (Model–View–Controller) і його похідні – MVT, що використовується в Django.

Шаблон MVT (Model – View – Template)

У проєктуванні майбутнього веб-додатку буде використано шаблон **Model – View – Template (MVT)**, який реалізовано у фреймворку Django — одному з найпопулярніших інструментів для створення масштабованих веб-застосунків [8].

Компоненти MVT:

1. Models (структура даних і база)

Модулі в `models.py` (лістинг А.1) описують сутності проєкту, зокрема клас `Ditina`, який містить такі поля, як ім'я, вік, стать, регіон, зріст, прийом гормональних препаратів тощо. Усі зв'язки реалізовані за допомогою вбудованої Object-Relational Mapping (ORM) Django, що дозволяє:

- уникнути SQL-запитів — доступ до даних реалізується методами на кшталт `Ditina.objects.all()` чи `filter(...)`;
- забезпечити валідацію та підтримку типів полів (вік, зріст — як `FloatField`);
- зручно масштабувати проєкт без зміни SQL-структури.

Модель `Ditina` відповідає за збереження інформації про кожну дитину. Вона містить такі поля:

- `ima` — ім'я дитини;
- `prizvische` — прізвище;
- `region` — країна проживання (обирається зі списку, що містить «Україна», «Польща», «Туреччина», «Японія», «США»);
- `vik` — вік у роках (дробові значення — допускаються);
- `zrist` — зріст у сантиметрах;
- `pryimaie_gormony` — булеве поле, що вказує, чи приймає дитина гормональні препарати;
- `data_stvorennia` — дата додавання запису;
- `data_onovlennia` — дата останнього редагування запису.

2. Views (обробка логіки та запитів)

Модель `Ditina` активно використовується у функціях контролерів. Наприклад, у лістингу А.3 показано функцію `dashboard_page_view`, яка відповідає за обробку форм та вивід списку дітей, збережених користувачем. Тут також реалізовано додавання нової дитини. View-функції розміщено у `views.py` (лістинг А.3) та `views_graph.py` (лістинг А.).

Вони відповідають за:

- додавання, редагування та видалення дитини (`dashboard_page_view`, `edit_ditina_view`, `delete_ditina_view`);
- побудову графіків на основі зібраних даних

- обробку форм, що надсилаються з HTML-інтерфейсу;
- вивід динамічного контенту в шаблони.

Запити обробляються відповідно до маршрутизації, описаної у `urls.py` (лістинг A.5).

3. Forms (введення та валідація)

Для введення інформації про дитину використовується форма `DitinaForm` (`forms.py` (лістинг A.2)) яка дозволяє безпосередньо працювати з моделлю. Вона обмежує набір полів, які заповнює користувач, та слугує основою для валідації даних при надсиланні.

Форма прив'язана до моделі `Ditina`:

- визначено типи полів, їхні віджети, а також текстові підказки для користувачів;
- передбачено валідацію (наприклад, від'ємний зріст або вік не проходить перевірку);
- реалізовано кастомні стилі форм (через HTML-атрибути класів).

4. Templates (шаблони HTML-інтерфейсу)

Шаблони знаходяться в теці `templates/` і реалізовані за допомогою Jinja-подібного синтаксису:

- використовується спадкування шаблонів (`base.html`);
- реалізовано цикли, умови, виведення таблиць, форм і графіків;
- структура відповідає принципу розділення логіки (`view`) і відображення (`template`).

5. Static (CSS, JS, зображення)

Фронтенд частина створена з використанням:

- фреймворку Bootstrap для адаптивного дизайну;
- власних стилів, розміщених у `static/css/`;

- підтримки зображень, кнопок, кольорів та інтерактивності.

6. Database (база даних SQLite)

У проєкті використовується SQLite як легка вбудована реляційна СУБД, що не потребує окремого серверного середовища. Створення таблиць реалізовано через міграції Django (makemigrations, migrate).

Це забезпечує:

- автоматичне оновлення структури бази;
- контроль історії змін;
- легку портативність структури при розгортанні.

7. Admin interface (адміністрування)

Адмін-панель Django — це потужний інструмент для внутрішнього управління даними. Вона читає метадані з ваших моделей і надає зручний інтерфейс для керування вмістом сайту. Рекомендується використовувати її як внутрішній інструмент управління організацією, а не як основний інтерфейс для кінцевих користувачів [9].

Переваги використання адмін-панелі Django

- Швидкість розробки: Можна швидко створити інтерфейс для управління даними без додаткового кодування.
- Гнучкість: Можливість налаштування інтерфейсу відповідно до потреб проєкту.
- Безпека: Вбудована система аутентифікації та авторизації забезпечує контроль доступу до даних.
- Інтеграція з моделями: Адмін-панель працює безпосередньо з моделями Django, що забезпечує узгодженість даних.

Файл admin.py (див. лістинг А.6) дозволяє:

- керувати даними дітей через зручний графічний інтерфейс Django Admin;

- реалізовано імпорт/експорт у CSV/JSON/Excel через пакет django-import-export;
- адміністратор має доступ до всіх CRUD-операцій та контролю валідації.

На рисунку 1.1 зображено реалізовану адмін-панель.

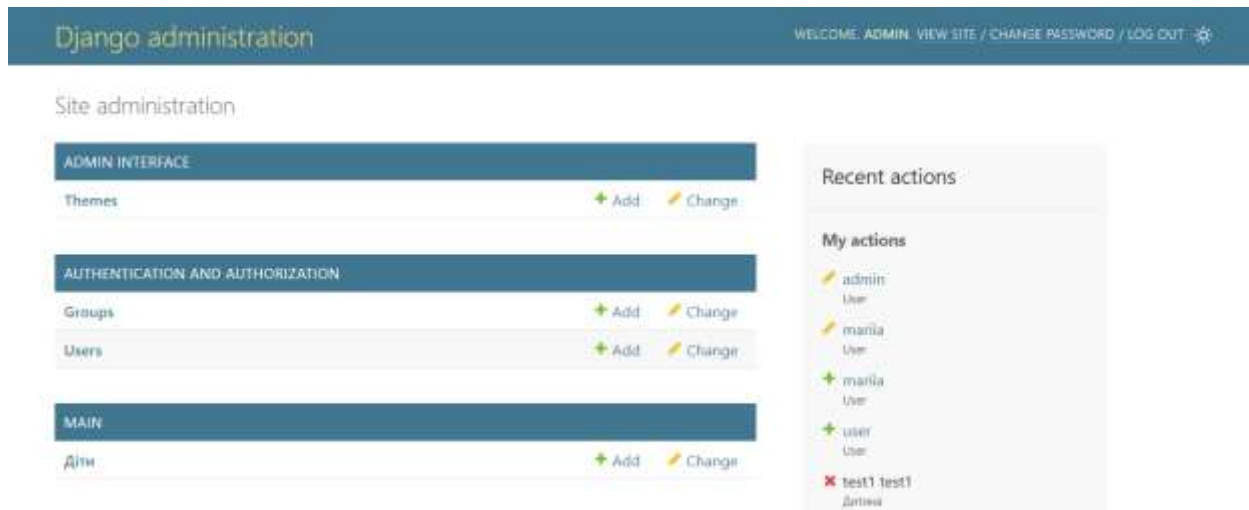


Рисунок 1.1 – Адмін-панель

Шаблон MVT дозволяє уникати змішування бізнес-логіки з відображенням, забезпечуючи високу гнучкість та забезпечує розділення занепокоєнь (separation of concerns), що є важливою умовою для написання коду [10].

Вибір архітектурного підходу

Оскільки розробка стосується роботи з динамічними даними — такими як антропометричні показники дітей — важливо обрати таку архітектуру, яка дозволяє:

- зручно організовувати і зберігати дані (Model),
- обробляти запити з урахуванням логіки користувача (View),
- гнучко та швидко оновлювати інтерфейс (Template).

На рисунку 1.2 зображено загальну архітектуру веб-додатку, що демонструє взаємозв'язок між моделлю, представленням та шаблоном у контексті шаблону MVT.

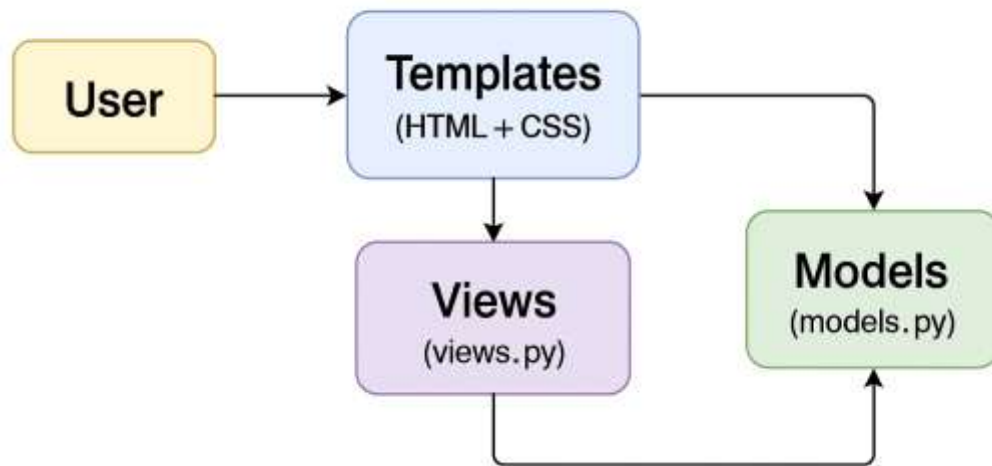


Рисунок 1.2 – Загальна архітектура веб-додатку

Шаблон MVT оптимально підходить для реалізації медичного або напівмедичного веб-додатку, оскільки забезпечує підтримку складних запитів, реалізацію графіків, роботу з користувачами і персоналізовані рекомендації без зайвої складності архітектури.

Використання ORM в архітектурі Django

Важливою частиною архітектури є ORM (Object-Relational Mapping) — механізм, який дозволяє працювати з базою даних через об'єкти мови програмування Python, без необхідності писати SQL-запити вручну. У Django ORM є складовою Model-компонента і забезпечує:

- створення таблиць у БД на основі Python-класів;
- автоматичне оновлення схеми бази через міграції;
- виконання запитів за допомогою простих методів (наприклад, `.filter()`, `.get()`).

Переваги використання Django ORM:

1. **Швидкість розробки:** Django ORM дозволяє розробникам швидко створювати та змінювати структуру бази даних без необхідності писати SQL-запити. Це особливо корисно на початкових етапах розробки, коли структура даних може часто змінюватися
2. **Підтримка міграцій:** Міграції в Django дозволяють відстежувати зміни в моделях та автоматично застосовувати їх до бази даних. Це забезпечує зручне управління версіями схеми бази даних та спрощує командну розробку.
3. **Безпека:** Django ORM автоматично захищає від SQL-ін'єкцій, оскільки всі запити до бази даних проходять через безпечні методи. Це зменшує ризик вразливостей у веб-додатку
4. **Портативність:** Завдяки абстрагуванню від конкретної СУБД, Django ORM дозволяє легко змінювати базу даних (наприклад, з SQLite на PostgreSQL) без значних змін у коді
5. **Зручна робота з відносинами:** Django ORM підтримує різні типи відносин між моделями (один-до-одного, один-до-багатьох, багато-до-багатьох), що дозволяє ефективно моделювати складні структури даних

За допомогою ORM можна описати логіку, де один користувач може створити багато записів для дітей, а кожна дитина — мати багато вимірювань. Така архітектура особливо корисна для медичних застосунків, які потребують точного контролю структур і залежностей.

Модульність та розширюваність

У рамках запропонованої архітектури кожен компонент є ізольованим:

- Моделі не мають логіки представлення;
- Представлення не містять SQL-коду;
- Шаблони не мають бізнес-логіки.

Це дозволяє розробникам зосереджуватись на окремих частинах системи без впливу на інші, здійснювати командну розробку, адаптувати застосунок під інші цільові групи (наприклад, лікарів або фахівців з розвитку).

Обмеження та альтернативи

Хоча Django ORM має багато переваг, існують ситуації, коли використання сирих SQL-запитів може бути доцільнішим:

1. Складні запити:

Для дуже складних або специфічних запитів ORM може бути обмеженим, і написання сирого SQL може бути ефективнішим

2. Оптимізація продуктивності:

У випадках, коли необхідна максимальна продуктивність, сирі SQL-запити можуть забезпечити кращу оптимізацію.

Однак Django надає можливість використовувати сирі SQL-запити через метод `raw()` або модуль `django.db.connection`, що дозволяє поєднувати переваги ORM та прямого SQL у межах одного додатку.

2 ПРАКТИЧНА ЧАСТИНА

2.1 Реалізація веб-додатку

Опис реалізованого функціоналу

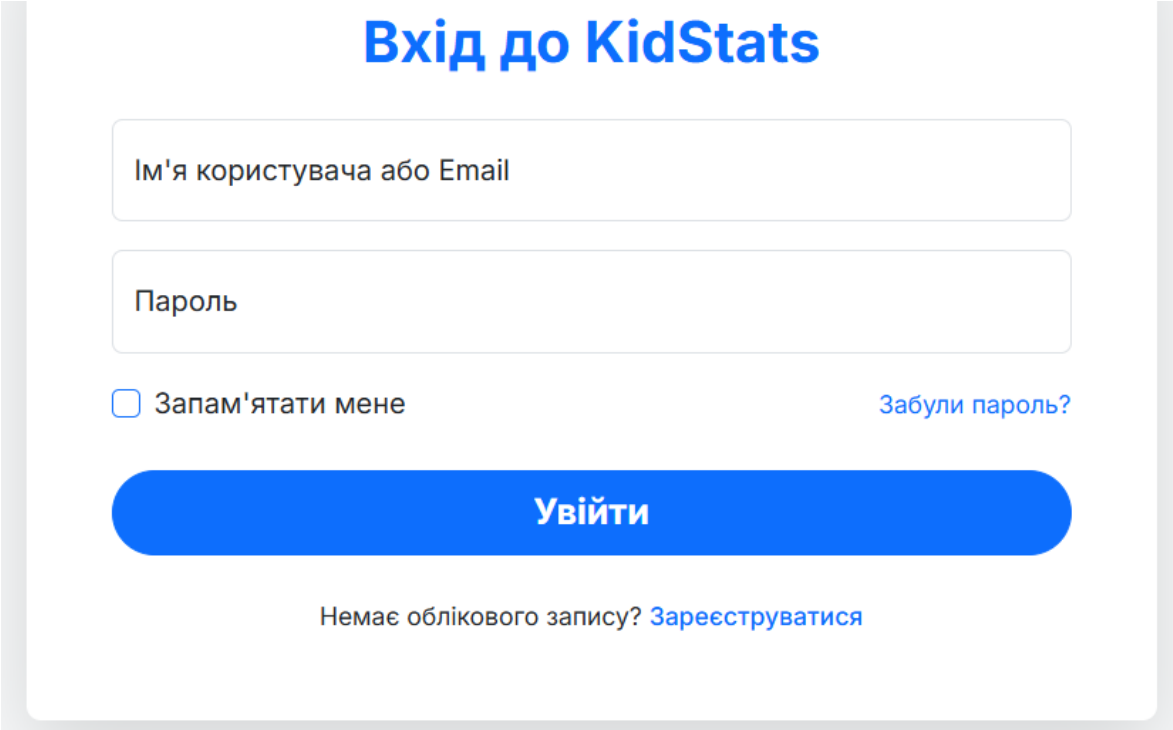
Реалізований веб-додаток містить набір функцій, що забезпечують повноцінне ведення даних про фізичний розвиток дітей, включаючи автентифікацію користувачів, створення анкет дитини, облік зросту, візуалізацію та обмін інформацією через зручні формати.

1. Робота з користувачами

У веб-додатку реалізовано базову систему керування користувачами:

Реєстрація та логін

Користувач має можливість створити обліковий запис через стандартну форму, після чого отримує доступ до кабінету. При вході перевіряються логін і пароль відповідно до системи аутентифікації Django. Сторінка входу представлена на рисунку 2.1.



Вхід до KidStats

Ім'я користувача або Email

Пароль

☐ Запам'ятати мене [Забули пароль?](#)

Увійти

Немає облікового запису? [Зареєструватися](#)

Рисунок 2.1 – Вхід в аккаунт

Ролі користувачів

Після входу кожному користувачеві надається відповідний рівень доступу адміністратор має повні права: може додавати, редагувати, видаляти записи про дітей, бачити всі профілі, використовувати функції імпорту/експорту, а звичайний користувач має доступ лише до перегляду власних записів.

Інтерфейс логіну використовує базову форму з перевіркою правильності введення даних. Всі маршрути із захищеним доступом перевіряють автентифікацію перед обробкою запиту.

Список користувачів в адмін-панелі представлено на рисунку 2.2.

	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
☐	admin	admin@gmail.com			✓
☐	maria	mariamiro.1234.mm@gmail.com	Maria	Miroshnychenko	✗
☐	user				✗

3 users

Рисунок 2.2 – Список користувачів в адмін-панелі

2. Робота з профілями дітей

Усі операції створення, редагування, перегляду та видалення профілів дітей реалізовано у двох режимах: через головний інтерфейс веб-додатку (для користувачів) та через адміністративну панель Django (для адміністраторів).

Такий підхід забезпечує як доступність для звичайного користувача, так і розширене керування даними з боку адміністратора або медичного працівника.

Для цього реалізовано повний набір CRUD — це акронім, що позначає чотири базові операції, які виконуються над даними в інформаційних системах:

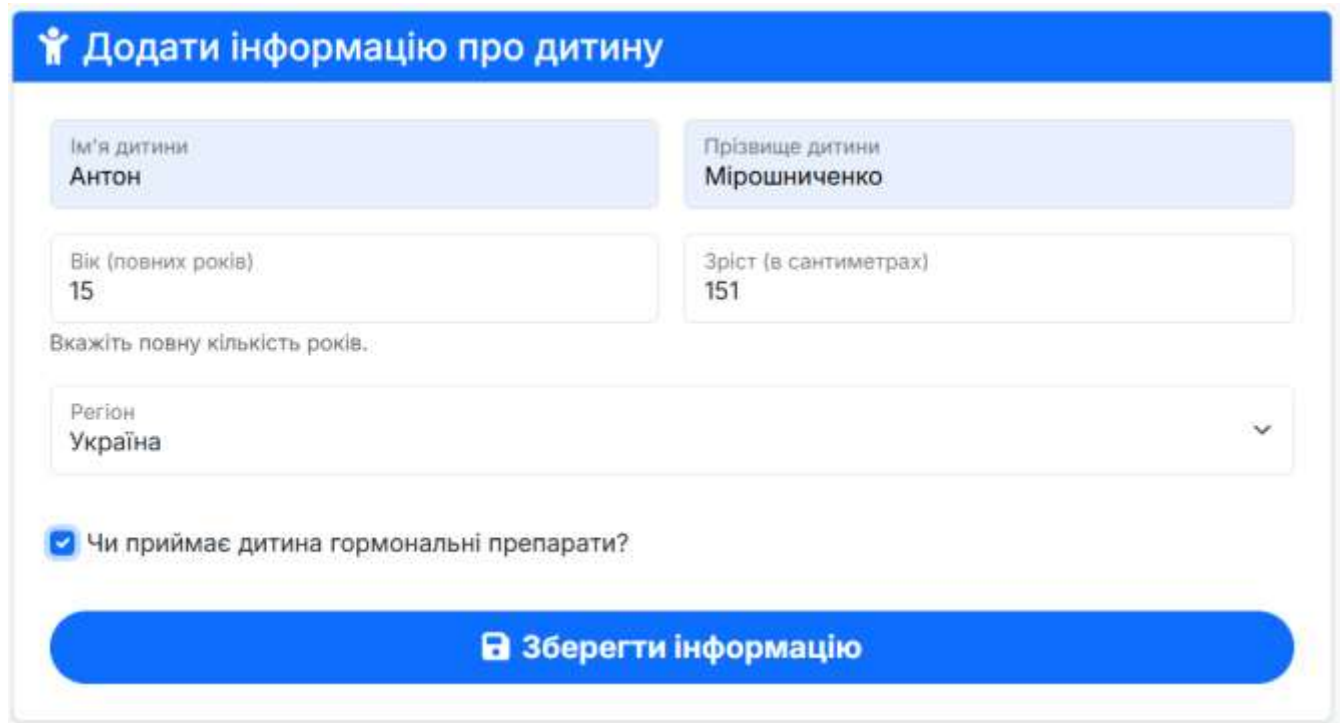
- Create (Створення): додавання нових записів до бази даних.
- Read (Читання): отримання або перегляд існуючих даних.
- Update (Оновлення): зміна або редагування наявних записів.
- Delete (Видалення): видалення записів з бази даних [11].

Ці операції є основою роботи будь-якого веб-додатку, який працює з базами даних. Вони відповідають ключовим функціональним сценаріям користувача: додати нову інформацію, переглянути її, змінити або видалити.

У Django реалізація CRUD здійснюється через view-функції, форми, шаблони та ORM-запити:

Створення профілю дитини

Через інтерфейс веб-додатку користувач заповнює форму з полями: ім'я, прізвище, вік, зріст, стать, регіон, гормональна терапія. Всі поля проходять перевірку в *forms.py* (лістинг А.2) перед збереженням у базу. Форма для введення даних про дитину через основний інтерфейс зображена на рисунку 2.3.



Додати інформацію про дитину

Ім'я дитини
Антон

Прізвище дитини
Мірошніченко

Вік (повних років)
15

Зріст (в сантиметрах)
151

Вкажіть повну кількість років.

Регіон
Україна

☒ Чи приймає дитина гормональні препарати?

Зберегти інформацію

Рисунок 2.3 – Форма для додавання дитини (через сайт)

Через адміністративну панель адміністратор може створити новий запис у відповідному розділі, використовуючи ті ж самі поля, що автоматично генеруються Django Admin. Аналогічна форма, реалізована в адмін-панелі, наведена на рисунку 2.4.

Add Дитина

Ім'я:	
Прізвище:	
Region:	----- ▼
Вік:	
Зріст (у см):	
☐ Приймає гормони	

Рисунок 2.4 – Форма для додавання дитини (через адмін-панель)

Редагування та перегляд

Через сайт існуючі записи можна оновити, при переході до редагування дані автоматично підставляються у форму.

Процес оновлення даних через веб-інтерфейс ілюструє рисунок 2.5.

Додати інформацію про дитину

Ім'я дитини Нікіта	Прізвище дитини Лісюк
Вік (повних років) 12	Зріст (в сантиметрах) 105

Вкажіть повну кількість років.

Region
Туреччина

☒ Чи приймає дитина гормональні препарати?

Оновити інформацію

Скасувати

Рисунок 2.5– Оновлення вже існуючих даних (через сайт)

Через адміністративну панель можна відкрити запис та змінити безпосередньо в адмін-інтерфейсі.

В адмін-панелі зміни до запису вносяться так, як показано на рисунку 2.6.

Change Дитина

Кучерявий Денис HISTORY

Ім'я:	Денис
Прізвище:	Кучерявий
Регіон:	Україна
Вік:	16
Зріст (у см):	144
☒ Приймає гормони	

SAVE Save and add another Save and continue editing Delete

Рисунок 2.6 – Редагування дитини в адмін-панелі

Видалення запису

Підтвердження видалення запису супроводжується діалоговим вікном (рисунок 2.7).

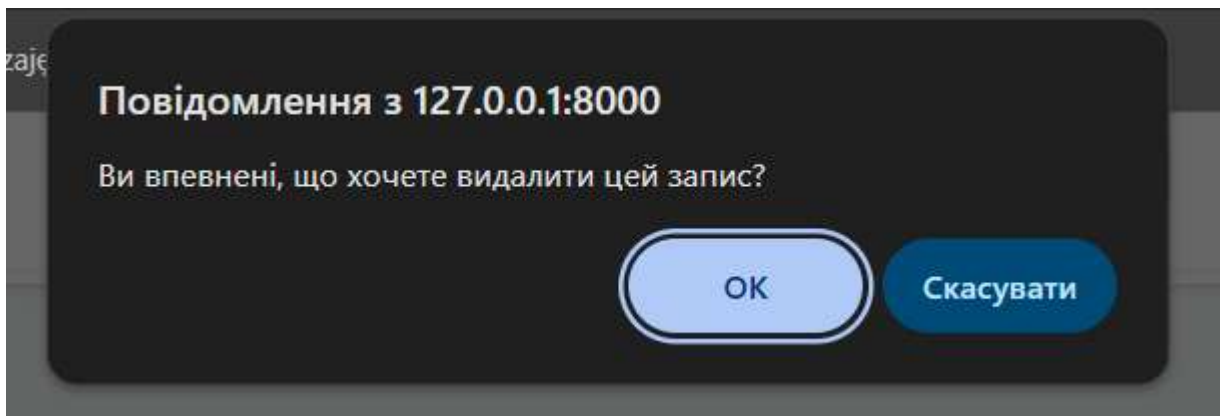


Рисунок 2.7 – Підтвердження видалення дитини.

У Django Admin видалення здійснюється стандартним способом — через інтерфейс таблиці або картки об'єкта згідно з рисунком 2.8.

Select Дитина to change

Action: 3 of 8 selected

☐	ID	ІМ'Я	ПРИЗВИЩЕ	ВІК	ЗРІСТ (У СМ)
☒	17	Олег	Коваль	8	130
☒	13	Дмитро	Ковальчук	12	150
☐	11	Іванка	Котира	16	131
☒	10	Денис	Кучерявий	16	144

Рисунок 2.8 – Видалення запису в адмін-панелі

Імпорт та експорт даних

Операції імпорту та експорту даних є важливою частиною функціоналу веб-додатку, що забезпечує зручність використання системи, особливо в контексті медичних даних, які потребують надійного зберігання, резервного копіювання та гнучкого введення.

Призначення та переваги:

1. **Захист від втрати даних**

Експорт даних у форматах Comma-Separated Values (CSV) або JavaScript Object Notation (JSON) дозволяє регулярно створювати резервні копії. Це важливо в разі технічних збоїв, переїзду на новий сервер або потреби відновлення даних після видалення чи помилки користувача.

2. Масове введення інформації

Імпорт з Excel дозволяє швидко перенести вже наявні списки дітей до веб-додатку без потреби вручну створювати кожен запис. Це зручно, наприклад, для лікарів, шкіл або великих сімей, які вже мають дані у табличному вигляді.

3. Передача даних між установами

CSV та JSON — формати, які легко інтегруються в інші системи. Це відкриває можливість обміну інформацією між лікарями, батьками або медичними центрами.

4. Аналітика та звітність

Дані, експортовані з веб-додатку, можна обробити у зовнішніх аналітичних системах — наприклад, у Google Sheets, Excel або Power BI — для створення додаткових звітів і візуалізацій.

Технічна реалізація:

Для реалізації функціоналу імпорту та експорту використано модуль `django-import-export`, який дозволяє адміністратору працювати з даними напряму через вбудовану адмін-панель Django.

Імпорт з Excel (XLSX)

У веб-додатку реалізовано можливість завантажити файл із записами про дітей, які буде автоматично оброблено та збережено в базі даних.

Це дозволяє:

- швидко перенести велику кількість записів;
- зменшити кількість помилок при ручному введенні;
- зекономити час при первинному заповненні системи.

Експорт у CSV / JSON

Адміністратор може натиснути кнопку "Експортувати", обрати формат та отримати файл зі структурованими даними. Усі поля моделі Ditina включаються в експорт автоматично, але за потреби можна залишити тільки вибрані. Механізм експорту реалізовано у вигляді, що подано на рисунку 2.9.

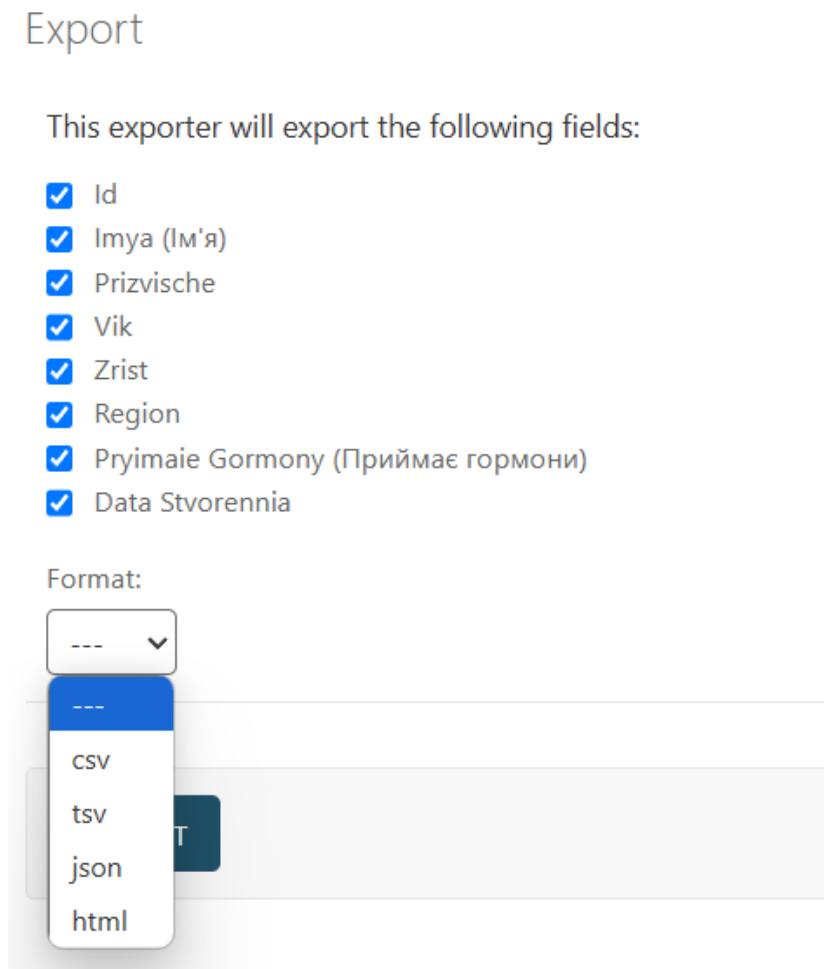


Рисунок 2.9 – Експорт даних у панелі адміністратора

Кодова реалізація:

У файлі admin.py (лістинг А.6) оголошено ресурсний клас DitinaResource. Цей клас зв'язує модель Ditina з бібліотекою import_export. Підключення моделі до системи імпорту/експорту відображено на рисунку 2.10.

```
@admin.register(Ditina)
class DitinaAdmin(ImportExportModelAdmin):
    resource_class = DitinaResource
    list_display = ('id', 'imya', 'prizvische', 'vik', 'zrist', 'region', 'pryimaie_gormony')
```

Рисунок 2.10 – Реєстрація моделі з підтримкою імпорту/експорту

Таким чином, адміністратор має доступ до повноцінного інтерфейсу обміну даними прямо в Django Admin, без додаткових інструментів.

Візуалізація даних

Однією з ключових функцій веб-додатку є графічне представлення даних, що значно полегшує аналіз фізичного розвитку дитини як для батьків, так і для медичних працівників.

Інтерфейс користувача (UI) та реалізація сторінок

Для реалізації візуальної частини застосовано Bootstrap — це безкоштовний, відкритий фронтенд-фреймворк, який допомагає розробникам швидко створювати адаптивні, мобільно-орієнтовані вебсайти та вебдодатки [12].

Основні характеристики Bootstrap:

- Мобільно-орієнтований дизайн: Bootstrap спочатку розроблявся з урахуванням потреб мобільних пристроїв, забезпечуючи оптимальне відображення на екранах різних розмірів.
- 12-колонкова сітка: Гнучка система сіток дозволяє легко створювати адаптивні макети, які автоматично підлаштовуються під розмір екрана.
- Готові компоненти: Надає набір попередньо стилізованих елементів інтерфейсу, таких як кнопки, форми, навігаційні панелі, модальні вікна та інші.
- JavaScript-плагіни: Включає інтерактивні компоненти, такі як каруселі, випадаючі меню, підказки та інші, які працюють без необхідності використання сторонніх бібліотек.

- Підтримка Sass: Можливість налаштування стилів за допомогою змінних, міксинів та інших функцій препроцесора Sass.

Цей фреймворк дозволив легко реалізувати адаптивний дизайн, використовуючи готові компоненти, такі як navbar, cards, buttons, tables, form-control тощо. Крім того, частина стилів оформлена вручну з використанням CSS-класів для кольорів, відступів і тіней.

Головна сторінка (Home)

Головна сторінка веб-додатку KidStats оформлена з використанням CSS-фреймворку Bootstrap. Це забезпечує адаптивність для різних пристроїв та сучасний вигляд. Інтерфейс простий і зрозумілий — у верхній частині розташоване навігаційне меню (navbar) з логотипом і посиланнями на головні розділи: "Головна", "Панель", "Про нас". Якщо користувач ще не ввійшов в аккаунт, то відображається кнопка "Увійти".

Центральна частина сторінки містить велику промо-секцію (jumbotron), що містить основне повідомлення для користувача: «Слідкуйте та аналізуйте ріст вашої дитини». Нижче розміщено пояснення про переваги платформи — зручність, безпечність і функціональність.

Для дій користувача передбачено дві основні кнопки:

- «Додати нове вимірювання» (синя кнопка btn-primary) — веде до форми введення ростових даних;
- «Перейти до панелі» (сіра кнопка btn-secondary) — перенаправляє в особистий кабінет користувача.

Нижній колонтитул містить копірайт та фразу із сердечком і підписом про місію платформи — «щоб допомогти відстежувати розвиток вашої дитини».

Таке оформлення відповідає сучасним практикам UX/UI, забезпечує простоту взаємодії та візуально привабливу структуру, зосереджену на основних функціях. Інтерфейс головної сторінки платформи зображено на рисунку 2.11.

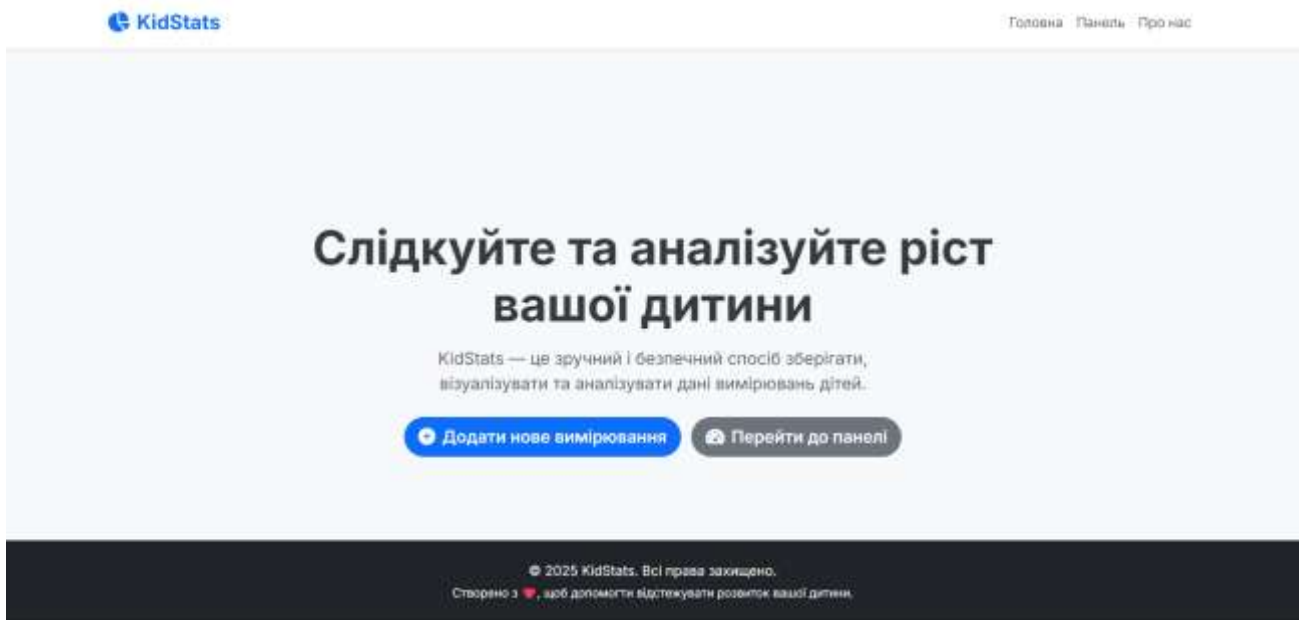


Рисунок 2.11 – Головна сторінка веб-додатку KidStats

Сторінка «Про нас»

Ця сторінка має статичну структуру та також створена за допомогою компонентів Bootstrap (включно з cards, icons). Містить заголовок «Про KidStats» та опис місії платформи. Серед розділів:

- Наша Місія — пояснення призначення сервісу, наголос на простоті, безпечності та підтримці батьків;
- Як це працює — короткий огляд функціоналу (збереження, аналіз, візуалізація даних вимірювань, інтерфейс кабінету).
- Наша команда — інформація про осіб, які стоять за створенням платформи.

У верстці використано іконки, картки, сіткову структуру Bootstrap. Розділ виконує функцію довіри та демонструє прозорість проєкту.

Така сторінка виконує не лише інформаційну, а й комунікаційну функцію, допомагаючи користувачам зрозуміти мету розробки, довіритися сервісу та зорієнтуватися в його можливостях. Інформаційна сторінка про платформу KidStats представлена на рисунку 2.12.



Рисунок 2.12 – Сторінка «Про нас»

Графіки росту

Система автоматично будує графіки на основі внесених користувачем даних про зріст і вагу дитини. Вісь X зазвичай відображає вік (у роках або місяцях), а вісь Y — відповідне значення (см або кг). Кожна точка на графіку — це результат одного вимірювання, а з'єднання точок дозволяє побачити тренд розвитку. Завдяки графічному відображенню користувач легко бачить, як змінюється фізичний стан дитини з часом.

Порівняння з нормативами

На графіках розвитку (ГР) накладаються нормативні криві, засновані на міжнародних стандартах ВООЗ (World Health Organization) та українських медичних таблицях зросту, де це доступно.

Дані користувача автоматично порівнюються з цими нормами. Якщо параметри виходять за межі прийнятного діапазону (нижче 3-го або вище 97-го центиллю), система відображає це кольоровими маркерами. У таких випадках також генеруються текстові рекомендації.

Адаптивна візуалізація

Графіки побудовано з використанням бібліотек Plotly та Matplotlib, що дозволяє генерувати як інтерактивні графіки для браузера (з масштабуванням, наведенням, фільтрами), так і статичні зображення для експорту у PDF чи друку. Веб-додаток будує графік зросту на основі даних користувача (див. Додаток Б).

Окрім графічного представлення даних, веб-додаток реалізує функціональність автоматичних порад для користувача. Поради генеруються на основі порівняння введених даних про зріст дитини з нормативними значеннями для відповідного віку. Це дає можливість користувачам оперативно отримати текстові висновки щодо стану фізичного розвитку дитини.

Основні особливості реалізації:

- Система аналізує введене значення зросту дитини у порівнянні з очікуваним середнім значенням для її віку;
- У разі відхилення від середнього рівня (менше або більше, ніж ± 1 стандартне відхилення), система формує текстову рекомендацію;
- Якщо дитина перебуває на гормональній терапії, виводиться спеціальне попередження з іконкою про необхідність консультації з лікарем;
- Текстові блоки рекомендацій відображаються динамічно — в залежності від введених параметрів, статусу дитини та медичних позначок.

У випадку, коли зріст дитини нижчий за середній для даного віку. Система інформує про очікуване значення та рекомендує звернення до педіатра. (див. Рисунок 2.13)

Якщо зріст дитини перебуває у межах норми. Система виводить текстове підтвердження цього та порівнює з середнім значенням. (див. Рисунок 2.14) Коли зріст дитини на нижній межі норми. Рекомендація зосереджена на необхідності спостереження за динамікою змін протягом наступних місяців (див. Рисунок 2.15).

 **Увага:** Дитина приймає гормональні препарати. Це може суттєво впливати на ріст і розвиток. **Обов'язково проконсультуйтеся з лікарем** для індивідуальної оцінки та регулярного моніторингу.

 **Зріст нижче середнього** для дитини вашого віку (131.0 см при 16.0 роках). Очікуване значення: 172.9 см. **Рекомендується звернутися до педіатра** для консультації та оцінки розвитку.

Рисунок 2.13 – Повідомлення, коли зріст нижче середнього

 **Увага:** Дитина приймає гормональні препарати. Це може суттєво впливати на ріст і розвиток. **Обов'язково проконсультуйтеся з лікарем** для індивідуальної оцінки та регулярного моніторингу.

 **Зріст у межах норми:** 150.0 см при 12.0 роках. Середнє значення: 149.1 см.

Рисунок 2.14 – Повідомлення, коли зріст дитини в нормі

 **Увага:** Дитина приймає гормональні препарати. Це може суттєво впливати на ріст і розвиток. **Обов'язково проконсультуйтеся з лікарем** для індивідуальної оцінки та регулярного моніторингу.

Зріст на нижній межі норми: 142.0 см при 12.0 роках. Очікуваний середній зріст: 148.1 см. **Слідкуйте за динамікою** росту дитини впродовж наступних місяців.

Рисунок 2.15 – Повідомлення, коли зріст на нижній межі норми

2.2 Робота з базою даних

БД (База даних) у веб-додатку реалізована за допомогою вбудованої системи ORM фреймворку Django, яка забезпечує об'єктно-реляційне відображення (Object-Relational Mapping). Завдяки ORM, усі взаємодії з базою даних реалізуються через Python-класи, без необхідності прямого використання SQL.

У веб-додатку використано легку реляційну систему управління базами даних (СУБД) SQLite, що не потребує окремого серверного середовища, є зручною для локального розгортання та тестування, а також підтримує основні реляційні зв'язки, обмеження і транзакції.

1. Модельна частина

Структура бази даних визначається у файлі `models.py` (лістинг А.1), де описано основні сутності веб-додатку:

- `User` – стандартна модель Django, яка містить облікову інформацію: логін, пароль, email, роль.

- Ditina – (відповідає Child) модель, яка представляє дитину з відповідними атрибутами: ім'я, прізвище, вік, зріст, регіон, гормональна терапія.

2. Основні поля моделі Ditina

Модель Ditina, яка є центральною у структурі веб-додатку, представляє окрему дитину з її персональними та медичними характеристиками. Клас описано у файлі `models.py` (лістинг A.1), і він використовується для збереження, редагування та відображення даних.

У таблиці 2.1 наведено опис основних полів моделі, що використовуються в базі даних веб-додатку.

Таблиця 2.1 – Опис ключових полів

Поле	Тип	Опис
<code>imya</code>	CharField (до 100 символів)	Ім'я дитини.
<code>prizvische</code>	CharField	Прізвище дитини.
<code>region</code>	CharField з вибором	Країна/регіон (Україна, Польща, Туреччина, Японія, США).
<code>vik</code>	PositiveIntegerField	Вік дитини у повних роках.
<code>zrist</code>	PositiveIntegerField	Зріст дитини у сантиметрах.
<code>pryimaie_gormony</code>	BooleanField	Чи проходить дитина гормональну терапію (True/False).
<code>data_stvorennia</code>	DateTimeField	Дата створення запису (автоматично при додаванні).
<code>data_onovlennia</code>	DateTimeField	Дата останнього оновлення запису (оновлюється при кожній зміні).

До особливостей реалізації можемо віднести:

`REGION_CHOICES` — дозволяє обмежити вибір регіону лише до заздалегідь визначених значень, що спрощує локалізацію нормативів росту.

Meta клас:

- `ordering = ['prizvische', 'imya']` — задає сортування у таблицях та формах за прізвищем і іменем.
- `verbose_name`, `verbose_name_plural` — назви, що використовуються в адміністративній панелі.

Метод `__str__()` — повертає зрозуміле представлення запису у вигляді: "Прізвище Ім'я".

3. Керування базою даних через термінал та обробка даних

Окрім стандартного взаємодії з базою даних через графічний інтерфейс (форми, панель адміністратора), веб-додаток підтримує повноцінну роботу з ORM через термінал. Це надзвичайно важливо для швидкого тестування, резервного копіювання, переносу даних та розширеної аналітики. На рисунку 2.14 показано команду, яка використовується для запуску інтерактивної оболонки Django.

```
j>python manage.py shell
```

Рисунок 2.16 – Команда для відкриття інтерактивної оболонки

У цьому режимі можна: створювати нові об'єкти моделей (додавати нових дітей); змінювати вже існуючі записи; видаляти об'єкти; здійснювати запити з умовами фільтрації; переглядати SQL-запити, які виконує ORM.

Для додавання нового запису до бази даних можна використовувемо Python-об'єкт класу `Ditina`. Після виклику методу `save()` об'єкт автоматично зберігається в базі даних. Рисунок 2.15 демонструє процес створення нового об'єкта моделі за допомогою ORM у терміналі.

```
>>> from main.models import Ditina
>>> d = Ditina(imya='Олер', prizvische='Коваль', vik=8, zrist=130, region='UKR')
>>> d.save()
```

Рисунок 2.17 – Створення об'єкта через ORM у терміналі

За допомогою ORM можна зчитувати інформацію з бази даних у вигляді Python-об'єктів. На рисунку 2.16 представлено приклад отримання всіх об'єктів моделі та застосування фільтрації за регіоном.

```
>>> Ditina.objects.all()
<QuerySet [ <Ditina: Коваль Олер>, <Ditina: Ковальчук Дмитро>, <Ditina: Котира Іванка>, <Ditina:
: Кучерявий Денис>, <Ditina: Лисюк Нікіта>, <Ditina: Оливкова Оля>, <Ditina: Сковорода Антон>,
<Ditina: Щур Петро]>
>>> Ditina.objects.filter(region='UKR')
<QuerySet [ <Ditina: Коваль Олер>, <Ditina: Ковальчук Дмитро>, <Ditina: Кучерявий Денис>, <Diti
na: Щур Петро]>
```

Рисунок 2.18 – Отримання всіх об'єктів та фільтрація за регіоном

ORM Django дозволяє застосовувати умови до запитів — аналогічно SQL, але у зручному об'єктно-орієнтованому стилі. Рисунок 2.17 ілюструє використання фільтрації для вибору об'єктів із зростом не менше 140 см.

```
>>> Ditina.objects.filter(zrist__gte=140)
<QuerySet [<Ditina: Ковальчук Дмитро>, <Ditina: Кучерявий Денис>, <Ditina: Оливкова Оля>, <Ditina: Сковорода Антон>, <Ditina: Щур Петро>]>
```

Рисунок 2.19 – Фільтрація: Зріст ≥ 140 см

На рисунку 2.18 показано приклад сортування об'єктів за віком у порядку спадання.

```
>>> Ditina.objects.order_by('-vik')
<QuerySet [<Ditina: Щур Петро>, <Ditina: Кучерявий Денис>, <Ditina: Сковорода Антон>, <Ditina: Оливкова Оля>, <Ditina: Котира Іванка>, <Ditina: Ковальчук Дмитро>, <Ditina: Лисюк Нікіта>, <Ditina: Коваль Олер>]>
```

Рисунок 2.20 – Сортування: за спаданням віку

У таблиці 2.2 представлено основні переваги використання об'єктно-реляційного відображення (ORM) у фреймворку Django.

Таблиця 2.2– Переваги використання ORM у Django

№	Перевага
1	Не потрібно знати SQL для базових операцій
2	Висока безпека — ORM захищає від SQL-ін'єкцій
3	Простота та читабельність синтаксису
4	Інтеграція з формами та валідацією
5	Легка підтримка міграцій і змін у моделі
6	Доступ до структури БД у вигляді об'єктів Python
7	Підтримка агрегацій, фільтрацій, анотацій, raw SQL при потребі
8	Можливість створення резервних копій та тестування

Завдяки ORM-функціоналу через термінал система є зручною для супроводу навіть без використання інтерфейсу. Вона підходить як для початківців, які не знайомі з SQL, так і для досвідчених користувачів, яким

потрібна швидка кастомна аналітика або обробка даних. Це робить веб-додаток гнучким, масштабованим і зручним у використанні.

2.3 Робота системи у реальному режимі

Функціонування веб-додатку в реальному часі передбачає інтерактивну взаємодію користувача із системою через зручний веб-інтерфейс. Користувачі можуть здійснювати всі основні дії — автентифікацію, додавання дітей, внесення вимірювань, перегляд графіків — без додаткових технічних знань.

Типовий сценарій взаємодії користувача:

- **Вхід у систему (логін)**

Користувач переходить на сторінку авторизації, вводить облікові дані (логін і пароль). Система перевіряє права доступу — звичайний користувач бачить лише своїх дітей, адміністратор має повний доступ.

- **Додавання нового профілю дитини**

Після входу користувач переходить у розділ "Додати дитину". Вносить: ім'я, прізвище, вік, регіон, зріст, ознаку прийому гормональної терапії. Дані зберігаються у базі за допомогою ORM.

- **Перегляд графіків росту**

Користувач переходить до візуалізації:

- побудова графіка росту на основі введених вимірювань;
- порівняння з нормативами;
- у разі відхилення — система може видати попередження або рекомендації.

На рисунку 2.19 зображено діаграму послідовностей, яка відображає кроки взаємодії між користувачем, формою введення даних, обробником запиту (View) та моделлю при додаванні нового запису про дитину у веб-додатку.

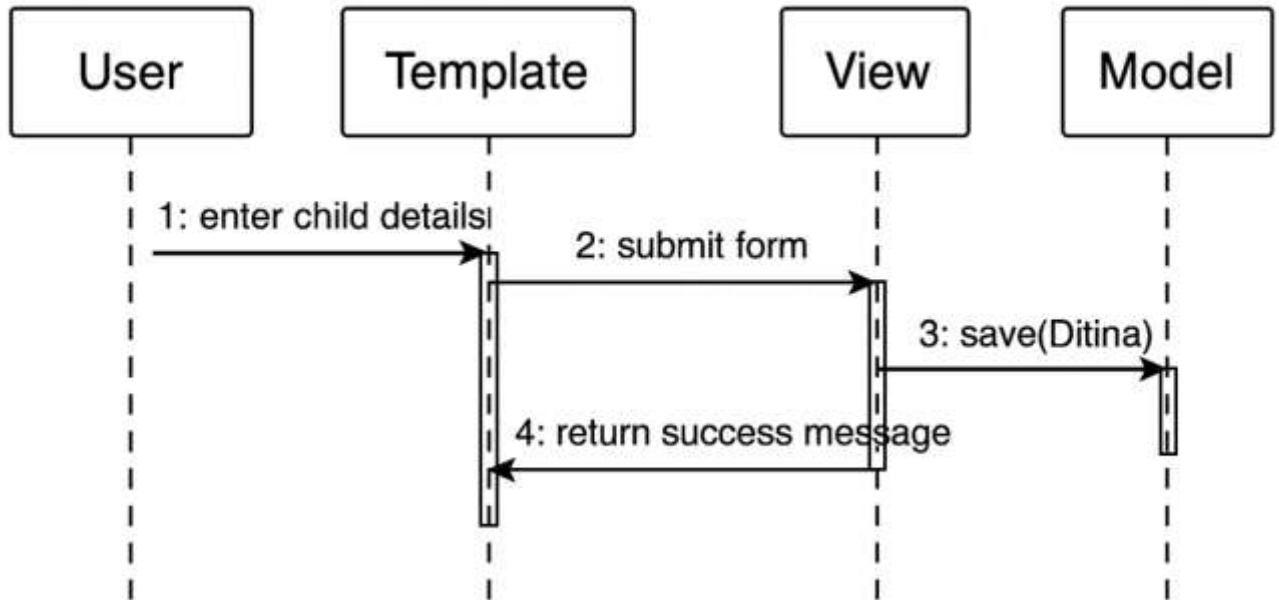


Рисунок 2.21 – Послідовність додавання дитини до системи

На UML-діаграмі показано послідовність повідомлень між основними компонентами веб-додатку під час додавання нової дитини:

1. Користувач ініціює дію через форму.
2. Інтерфейс (Template) збирає дані.
3. View приймає запит, обробляє форму.
4. Model (ORM) зберігає об'єкт Ditina.
5. Після збереження — відповідь повертається до користувача з повідомленням про успішне додавання.

Рисунок 2.20 ілюструє процес автентифікації адміністратора. Вказано, як відбувається обробка логіну — від введення облікових даних до верифікації через службові компоненти та повернення результату.

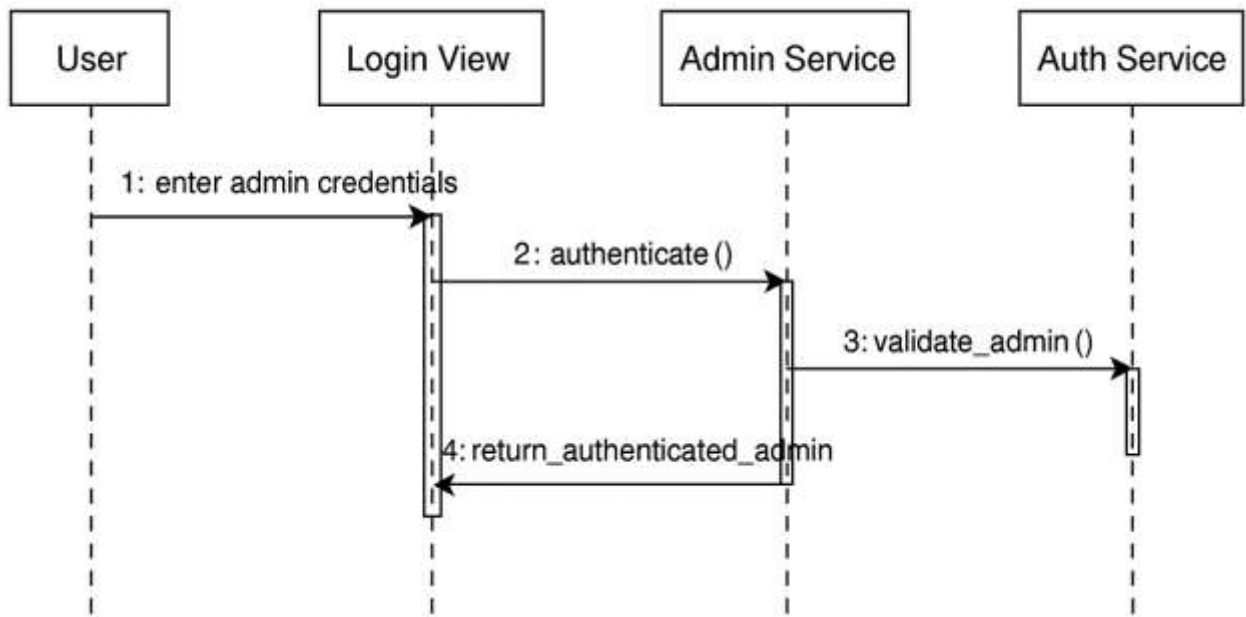


Рисунок 2.22 – Діаграма послідовностей автентифікації адміністратора у веб-додатку

На діаграмі показано типовий сценарій авторизації адміністратора у веб-додатку:

1. User (користувач) надсилає логін та пароль через форму на сторінці входу.
2. Login View приймає введені дані та ініціює виклик методу `authenticate()` для обробки авторизації.
3. Admin Service передає дані далі до Auth Service, де перевіряється справжність облікових даних через `validate_admin()`.
4. У разі успішної перевірки система повертає об'єкт автентифікованого адміністратора.

Опис переваг взаємодії:

До переваг системи можна віднести повну автоматизацію:

- усі дії виконуються без перезавантаження сторінки;
- простий і адаптивний інтерфейс (Bootstrap);
- збереження всіх даних із фіксацією дати та часу;
- усе працює за інтуїтивною логікою.

Сценарії використання (Use Case)

Щоб краще зрозуміти логіку взаємодії користувача з системою, нижче наведено типові сценарії використання (Use Cases), які охоплюють основні дії батьків або опікунів у процесі моніторингу фізичного розвитку дітей. Всі сценарії реалізовано в системі за допомогою відповідних view-функцій [A.3] та маршрутів [A.5].

UC1: Реєстрація нового користувача

Основний сценарій починається з реєстрації на платформі. Актором у цьому випадку виступає батько/внб або опікун дитини. Користувач переходить на сторінку реєстрації та заповнює відповідну форму, де вказує ім'я, електронну пошту та пароль.

UC2: Додавання дитини до профілю

Після успішної реєстрації та входу до особистого кабінету користувач має змогу додати дані про дитину. Для цього він переходить на сторінку «Панель» чи «Додати нове вимірювання». У формі необхідно вказати ім'я, прізвище, вік у роках (можливі дробові значення, наприклад, 2.5), зріст у сантиметрах, країну проживання та чи приймає дитина гормональну терапію. Після натискання кнопки «Зберегти» ці дані зберігаються в базі даних (модель Dіtina — Лістинг A.1), а дитина відображається у списку записів.

UC3: Перегляд графіка

Користувач обирає конкретний запис із переліку дітей і натискає «Показати графік». У відповідь система будує індивідуальний графік зросту на основі внесених даних. На графіку відображено середньостатистичні дані (типові вимірювання) та точка, що відповідає параметрам дитини. Якщо дитина приймає гормони, система додатково виводить текстове попередження з рекомендацією проконсультуватися з лікарем. Також формується висновок про те, чи зростова динаміка перебуває в межах норми, нижче або вище за середні показники для обраного віку.

На рисунку 2.21 продемонстровано інтерфейсну кнопку «Показати графік», яка ініціює побудову візуалізації даних про зростання чи вагу дитини. Вона є частиною функціоналу відображення аналітики у веб-додатку.

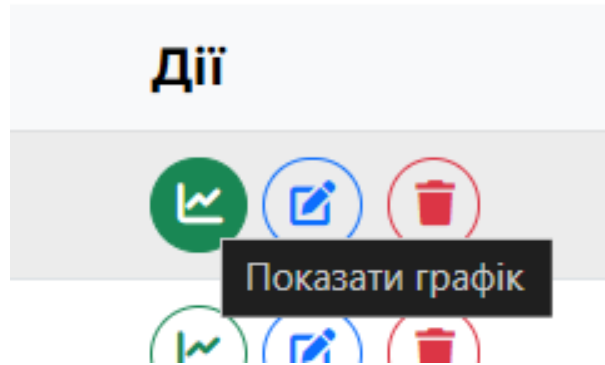


Рисунок 2.23 – Кнопка «Показати графік»

UC4: Редагування та оновлення даних

У разі зміни фізичних параметрів або необхідності виправлення даних користувач може відредагувати запис. Для цього потрібно натиснути «Редагувати» навпроти відповідного запису на сторінці дашборду. Система відкриває заповнену форму, в якій можна оновити будь-яке з полів: вік, зріст, ім'я або інше. Після підтвердження змін оновлені дані автоматично зберігаються в базі, а графік при повторному перегляді відображає вже нову точку.

Рисунок 2.22 показує реалізацію кнопки «Редагувати», яка дозволяє внести зміни до даних дитини. Цей елемент розташований у таблиці записів і активує форму редагування, доступну лише авторизованим користувачам.

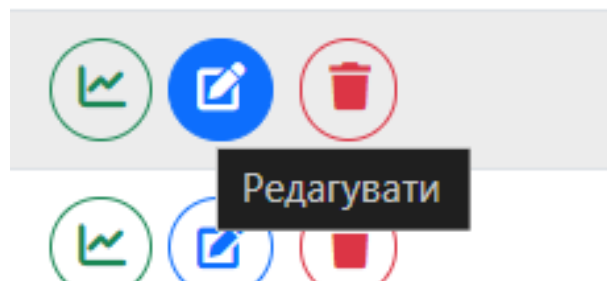


Рисунок 2.24 – Кнопка «Редагувати»

2.4 Тестування

Тестування програмного забезпечення (ПЗ) — це процес перевірки та оцінки якості програмного продукту з метою виявлення дефектів і забезпечення відповідності вимогам користувачів та специфікаціям. Для веб-додатків, тестування є критично важливим для забезпечення стабільності, безпеки та зручності використання системи [13].

Для перевірки надійності функціонування веб-додатку було проведено ручне тестування основного функціоналу. Метою тестування є перевірка того, що всі компоненти працюють згідно з технічними вимогами, без помилок при взаємодії користувача з інтерфейсом, базою даних та графічною підсистемою.

Тестування охоплювало як позитивні сценарії (нормальна поведінка при правильному введенні даних), так і негативні (неповні, некоректні, граничні значення).

Методика тестування

- Тестування проводилось локально в середовищі розробника.
- Використано облікові записи з різними ролями (звичайний користувач, адміністратор).
- Сценарії перевіряли роботу шаблонів, форм, графіків, імпорту, а також валідність даних.

У таблиці 2.3 наведено результати тестування ключових функціональних сценаріїв веб-додатку, що охоплюють операції автентифікації, взаємодії з базою даних, побудови графіків та обробки форм.

Таблиця 2.3 — Результати тестування функціоналу

№	Назва тесту	Вхідні дані	Очікуваний результат	Результат	Примітка
1	Реєстрація користувача	Коректні email, пароль, підтвердження	Створення нового облікового запису	Успішно	
2	Вхід у систему	Дійсні email та пароль	Перенаправлення на головну сторінку	Успішно	
3	Вхід із неправильним паролем	Вірний email, помилковий пароль	Повідомлення про помилку автентифікації	Успішно	
4	Додавання дитини	Ім'я, прізвище, вік, зріст, регіон	Новий запис збережено	Успішно	
5	Додавання дитини без імені	Пусте поле "Ім'я"	Повідомлення про помилку (форма не збережена)	Успішно	Перевірка валідації
6	Побудова графіка	Валідний профіль з вимірюваннями	Виведення графіка росту	Успішно	
7	Імпорт з Excel	Файл зі структурованими даними	Додано кілька нових записів	Успішно	Перевірено через адмін-панель
8	Вивід запиту з бази через shell	Ditina.objects.all()	Повертає всі об'єкти	Успішно	Тестування ORM
9	Видалення дитини	Обрати запис → "Видалити"	Запис видалено з бази	Успішно	
10	Введення граничного віку (0)	Вік = 0	Форма не зберігається, валідація	Успішно	Перевірка мінімального значення
11	Вхід неавторизованого користувача	Без логіну	Редірект на форму входу	Успішно	Перевірка захисту доступу

Висновки з тестування

- Усі критичні функції системи були протестовані та працюють стабільно.

- Валідація форм успішно запобігає введенню некоректних або неповних даних.
- Графіки коректно виводяться на основі динамічних запитів до бази.
- Імпорт/експорт через Django Admin працює згідно зі специфікацією.
- Захист доступу реалізований — неавторизовані користувачі не мають доступу до закритих маршрутів.
- Додаток залишається стабільним при роботі з десятками об'єктів, що є задовільним для локального середовища.

2.5 Висновки до розділу

У результаті реалізації практичної частини було створено повнофункціональний веб-додаток. Основна мета полягала в наданні користувачам інструменту для обліку, аналізу, візуалізації та зберігання даних про ріст дитини з можливістю подальшої обробки цих даних у зручних форматах.

Під час виконання практичної частини було реалізовано:

1. Функціонал автентифікації та ролей користувачів, що забезпечує контроль доступу до даних і можливість розділення прав між адміністратором і звичайним користувачем;
2. Модуль управління профілями дітей, включаючи базові CRUD-операції (створення, перегляд, редагування, видалення) через як основний інтерфейс, так і адміністративну панель;
3. Систему імпорту та експорту даних у форматах Excel, CSV, JSON із використанням бібліотеки `django-import-export`, що дозволяє здійснювати резервне копіювання та масове завантаження інформації;
4. Візуалізацію графіків росту та ваги на основі внесених даних із порівнянням з нормативами, з автоматичним маркуванням відхилень і рекомендаціями;

5. Інтерфейс, розроблений із використанням фреймворку Bootstrap, що забезпечує адаптивність і зручність для користувача;
6. Роботу з базою даних через ORM Django, включаючи використання моделей, фільтрацію, запити та інтерактивну роботу через термінал;
7. Діаграми послідовностей та приклади сценаріїв використання (Use Case), що відображають типову логіку взаємодії з веб-додатком;
8. Проведено ручне тестування ключових функціональних сценаріїв, результати якого показали коректну роботу системи, ефективну валідацію, стабільність під час обробки даних і захищеність доступу.

Таким чином, практична частина підтвердила функціональність та ефективність розробленого веб-додатку в контексті його реального застосування. Система забезпечує інтуїтивно зрозумілий користувацький досвід, зручне управління даними, можливість візуального аналізу та розширення функцій у майбутньому. Всі реалізовані компоненти працюють стабільно та відповідають технічному завданню, що дозволяє зробити висновок про успішну реалізацію практичної складової кваліфікаційної роботи.

3 ПІДСУМКОВИЙ АНАЛІЗ І ПЕРСПЕКТИВИ

3.1 Порівняння з поставленими завданнями

У процесі реалізації веб-додатку для моніторингу фізичного розвитку дітей було досягнуто цільової мети – створено інструмент, що дозволяє батькам зручно фіксувати, аналізувати та інтерпретувати показники зросту дітей відповідно до національних та міжнародних стандартів.

Об'єктом дослідження виступав процес цифрового моніторингу фізичного розвитку дитини, а предметом – методи реалізації такого моніторингу в межах веб-застосунку.

Фінальна реалізація повністю охоплює всі функціональні напрями, визначені на початковому етапі. Зокрема, реалізовано адаптивний інтерфейс, обробку індивідуальних параметрів дітей, інтеграцію з базою нормативів, функції імпорту та експорту даних, а також побудову динамічних графіків зросту.

Виконання завдань відповідає початково поставленим цілям дослідження. Розроблений веб-додаток охоплює як технічні, так і прикладні аспекти цифрового моніторингу росту, є придатним до використання в реальних умовах та має потенціал для масштабування. Це свідчить про практичну значущість реалізованого рішення, а також про ефективне досягнення мети дипломної роботи.

3.2 Обмеження та пропозиції розвитку

У процесі розробки та тестування веб-додатку було виявлено обмеження та функціональні можливості, які не були реалізовані у фінальній версії, але залишаються актуальними для подальшого розвитку. Також деякі аспекти мають технічні або логічні обмеження, що впливають на гнучкість системи.

1. Нереалізовані функціональні можливості

Автоматичні email-нагадування

Хоча в системі передбачено запис часу останнього вимірювання, функція нагадувань користувачу про наступне вимірювання (наприклад, кожні 3 або 6 місяців) не реалізована. Планувалося використати механізм періодичних задач (наприклад, Celery у зв'язці з Redis) для:

- надсилання email-нагадувань;
- створення повідомлень у веб-інтерфейсі;
- персоналізованих рекомендацій відповідно до віку дитини.

Причиною цього послушували брак часу на реалізацію асинхронної логіки та відсутність підтримки SMTP-конфігурації на етапі розробки.

Лікарська (медична) панель

У проєкті розглядалася ідея створення окремої ролі "лікаря", яка дозволяла:

- мати доступ до кількох дітей одночасно;
- призначати рекомендації;
- бачити позначки критичних відхилень від норм.

Причина відсутності: відсутність часу на реалізацію окремої системи прав доступу та інтерфейсу для лікарів. У поточній версії є лише 2 типи ролей: звичайний користувач і адміністратор.

2. Технічні обмеження

Використання SQLite

У якості системи управління базою даних обрано SQLite, що є оптимальним варіантом для прототипу або невеликого проєкту. Однак ця СУБД має обмеження:

- не підтримує багатокористувацький режим з великою кількістю паралельних записів;
- обмежений функціонал у порівнянні з PostgreSQL або MySQL;
- не підходить для великомасштабного продакшену.

Потенційне рішення: при розгортанні у хмарі або на сервері доцільно перейти на PostgreSQL, що краще масштабується та підтримує розширені функції.

3. Можливості для майбутньої реалізації

У поточній реалізації веб-платформа виконує базові функції моніторингу зросту дітей, однак для подальшого підвищення її ефективності, зручності та масштабованості, доцільно передбачити поступове розширення функціоналу.

Модель користувача (User)

У поточній реалізації використовуються стандартні можливості Django для автентифікації користувачів. Кожен користувач створює та управляє власними записами Ditina. У майбутньому доцільно розширити модель користувача (через AbstractUser) і додати такі поля:

- роль — дозволить розрізняти звичайного користувача (батька/матір) і лікаря;
- місце роботи/медустанова — для лікарів;
- телефон — для зворотного зв'язку або сповіщень.

Це відкриє можливість реалізації розширених сценаріїв взаємодії між користувачами, що мають різні ролі.

Модель вимірювань

Оскільки дитина протягом часу проходить кілька вимірювань, модель Ditina має потенціал для нормалізації даних через окрему модель Vymiryuvannia. Очікувана структура:

- ditina (ForeignKey) — посилання на дитину;
- data_vymiryuvannia — дата вимірювання;
- zrist — фактичний зріст;
- vaga — вага;
- komentar — додаткові нотатки.

Ця модель дозволить створити повноцінний часовий ряд і забезпечити більш точний аналіз тенденцій розвитку.

У таблиці 3.1 наведено орієнтовний план розвитку системи з розподілом по напрямках удосконалення, очікуваних результатах та технічних аспектах реалізації.

План розвитку передбачає поетапне вдосконалення проєкту в залежності від потреб користувачів, фінансування та зацікавленості медичної спільноти. Основна мета полягає в тому, щоб платформа стала не лише засобом реєстрації зростових показників, а й інтелектуальним помічником для батьків та лікарів.

Таблиця 3.1 — Орієнтовний план розвитку системи

Напрямок розвитку	Опис
1. Розширення функціональності	Додавання можливості введення додаткових параметрів (вага, ІМТ, обвід голови), розрахунок індексу гармонійного розвитку, статистика по дітях.
2. Автоматичні email-нагадування	Реалізація механізму регулярних сповіщень (на електронну пошту) про необхідність вимірювань або медичних візитів.
3. Впровадження ролей користувачів	Відокремлення прав доступу: «батько/мати», «лікар», «адміністратор». Для лікарів – можливість перегляду декількох дітей, створення рекомендацій.
4. Інтеграція медичних нормативів	Додавання офіційних українських та міжнародних (WHO, CDC) нормативів зросту й ваги з вибором типу шкали у профілі.
5. Підтримка багатомовності	Реалізація інтерфейсу українською, польською, англійською мовами. Вибір мови – автоматично або вручну у налаштуваннях.
6. Адаптація до мобільних пристроїв	Оптимізація зовнішнього вигляду під смартфони й планшети, створення адаптивної верстки з використанням Bootstrap.
7. Візуалізація динаміки	Реалізація графіків у форматі порівняння з кількома дітьми, динаміка росту за роками, heatmap, boxplot та інші методи візуального аналізу.
8. Зовнішній API для інтеграції	Створення REST API, що дозволить зовнішнім сервісам або мобільним додаткам підключатися до бази й отримувати/передавати дані.
9. Перехід на PostgreSQL	Для стабільності при масштабуванні — замінити SQLite на PostgreSQL. Це дозволить працювати з великою кількістю користувачів одночасно.
10. Безпека та GDPR	Додавання повноцінного захисту персональних даних згідно з GDPR: згода на обробку, шифрування даних, опції експорту й видалення профілю.
11. Додаток для мобільних пристроїв	У перспективі — розробка нативного мобільного додатку на Android/iOS з офлайн-режимом та синхронізацією з веб-сервісом.
12. Панель адміністратора	Більш розширений функціонал управління користувачами, перегляд активності, генерація звітів, блокування профілів.
13. Інтеграція з медичними платформами	Можливість обміну даними з електронними медичними картками (ЕМК), Medstar, Helsi, eHealth та ін.
14. Оцінка динаміки з використанням ML	У перспективі — впровадження машинного навчання для автоматичного виявлення аномалій та передбачення відхилень у розвитку.

ВИСНОВКИ

Актуальність обраної теми підтверджується зростаючою потребою в цифрових інструментах для моніторингу фізичного розвитку дітей, особливо в умовах недостатньої цифровізації медичних сервісів в Україні та актуальності для родин, які перебувають за кордоном. Розробка веб-додатку для моніторингу та діагностування стану дітей із відхиленнями зросту має важливе практичне значення для батьків, лікарів і освітніх установ.

Було проведено огляд літератури щодо фізіологічних нормативів розвитку дитини, розглянуто існуючі онлайн-сервіси та калькулятори, проаналізовано їх переваги й обмеження. Визначено, що більшість рішень не враховують українські нормативи й не мають інструментів для індивідуалізації, що посилило мотивацію до розробки нового рішення. Також були сформульовані функціональні вимоги до системи та запропоновано архітектуру на основі шаблону MVT фреймворку Django.

Далі було реалізовано повноцінний веб-додаток, який охоплює реєстрацію користувачів, введення та редагування даних про дітей, побудову графіків зросту, візуалізацію, імпорт/експорт даних, автоматичне порівняння з нормативами, рекомендації та систему ролей. Система побудована з використанням Bootstrap, SQLite, Plotly та ORM Django. Окремо реалізовано адмін-панель, що дозволяє адміністратору зручно керувати записами. Проведено тестування, що підтвердило стабільність і правильність роботи ключових функцій.

Зроблено співставлення реалізованого функціоналу з поставленими завданнями. Було виявлено відповідність між запланованим і досягнутим, а також зазначено обмеження, пов'язані з використанням SQLite, відсутністю автоматичних нагадувань та ролі лікаря. Запропоновано план розвитку системи, що охоплює багатомовність, перехід на PostgreSQL, інтеграцію з медичними

платформами, розширення ролей і навіть впровадження елементів машинного навчання.

Таким чином, поставлену мету — створення зручного та локалізованого веб-додатку для моніторингу зросту дітей із можливістю персоналізації — було досягнуто. Основна функціональність працює стабільно, користувацький інтерфейс є адаптивним, база даних оптимізована для збереження даних у реальному часі. Робота продемонструвала важливість поєднання медичних стандартів із сучасними веб-технологіями.

Подальші напрями розвитку включають впровадження мобільного додатку, підтримку персоналізованих графіків, автоматичні сповіщення, ролі для медичних працівників та повноцінну інтеграцію з електронними медичними картками. Це дозволить зробити веб-додаток універсальним рішенням для цифрового моніторингу фізичного розвитку дитини.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Medialt Clinic. Центильні таблиці: як визначити гармонійний розвиток дитини [Електронний ресурс]. — URL: <https://medialt.clinic/blog/rozvitok-ditini/centylni-tablytsi> (дата звернення: 12.02.2025).
2. World Health Organization. WHO Child Growth Standards [Електронний ресурс]. — Доступно: <https://www.who.int/tools/child-growth-standards/standards> (дата звернення: 21.02.2025).
3. CDC. WHO Growth Charts [Електронний ресурс]. — Доступно: <https://www.cdc.gov/growthcharts/who-growth-charts.htm> (дата звернення: 03.03.2025).
4. Medme.pl. Kalkulator Siatek Centylowych dla dzieci 0–18 lat [Електронний ресурс]. — Доступно: <https://www.medme.pl/kalkulatory/siatki-centylowe> (дата звернення: 12.03.2025).
5. Medme.pl. Kalkulator Siatek Centylowych dla dzieci 0–18 lat [Електронний ресурс]. — Доступно: <https://www.medme.pl/> (дата звернення: 20.03.2025).
6. More Than Height. Child Growth Calculator [Електронний ресурс]. — Доступно: <https://v2.morethanheight.com/en/calculator-charts/> (дата звернення: 02.04.2025).
7. Recepta.pl. Kalkulator Wagi Dziecka [Електронний ресурс]. — Доступно: <https://recepta.pl/artykuly/kalkulator-wagi-dziecka-sprawdz-czy-twoje-dziecko-odpowiednio-sie-rozwija> (дата звернення: 12.04.2025).
8. Django Project. Django design philosophies [Електронний ресурс]. — Доступно: <https://docs.djangoproject.com/en/5.2/misc/design-philosophies/> (дата звернення: 23.04.2025).
9. Django Project. Django Admin interface documentation [Електронний ресурс]. — Доступно: <https://docs.djangoproject.com/en/5.2/ref/contrib/admin/> (дата звернення: 02.05.2025).

10. GeeksforGeeks. Django Project MVT Structure [Електронний ресурс]. — Доступно: <https://www.geeksforgeeks.org/django-project-mvt-structure/> (дата звернення: 10.05.2025).

11. Welcome IT Professional. De CRUD-matrix uitgelegd [Електронний ресурс]. — Доступно: <https://uk.itpedia.nl/2023/03/25/de-crud-matrix-uitgelegd/> (дата звернення: 10.05.2025).

12. Hostinger. What is Bootstrap? [Електронний ресурс]. — Доступно: <https://www.hostinger.com/tutorials/what-is-bootstrap> (дата звернення: 21.05.2025).

13. Вікіпедія. Тестування програмного забезпечення [Електронний ресурс]. — Доступно: https://uk.wikipedia.org/wiki/Тестування_програмного_забезпечення (дата звернення: 28.05.2025).

ДОДАТОК А – ЛІСТИНГИ ПРОГРАМНОГО КОДУ

Лістинг А.1 — Модель Ditina (models.py)

```
class Ditina(models.Model):
    REGION_CHOICES = [
        ('UKR', 'Україна'),
        ('PL', 'Польща'),
        ('TU', 'Туреччина'),
        ('JA', 'Японія'),
        ('USA', 'США'),
    ]

    imya = models.CharField(max_length=100, verbose_name="Ім'я")
    prizvische = models.CharField(max_length=100, verbose_name='Прізвище')
    region = models.CharField(max_length=4, choices=REGION_CHOICES,
verbose_name='Регіон')
    vik = models.PositiveIntegerField(verbose_name='Вік')
    zrist = models.PositiveIntegerField(verbose_name='Зріст (у см)')
    pryimaie_gormony = models.BooleanField(default=False,
verbose_name='Приймає гормони')
    data_stvorennia = models.DateTimeField(auto_now_add=True,
verbose_name='Дата створення')
    data_onovlennia = models.DateTimeField(auto_now=True,
verbose_name='Дата оновлення')
```

Лістинг А.2 — Форма DitinaForm (forms.py)

```
class DitinaForm(forms.ModelForm):
    class Meta:
        model = Ditina
```

```
fields = ['imya', 'prizvische', 'region', 'vik', 'zrist', 'pryimaie_gormony']
```

Лістинг А.3 — View для обробки форми (views.py)

```
def dashboard_page_view(request):
    form_submitted = False
    if request.method == 'POST':
        form = DitinaForm(request.POST)
        form_submitted = True
        if form.is_valid():
            form.save()
            messages.success(request, 'Інформацію про дитину успішно додано!')
            return redirect('dashboard')
    else:
        form = DitinaForm()

    dity_list = Ditina.objects.all()

    context = {
        'form': form,
        'dity_list': dity_list,
        'form_submitted': form_submitted,
    }
    return render(request, 'dashboard.html', context)
```

Лістинг А.4 — Побудова графіка (views_graph.py)

```
def random_graph(request, ditina_id):
    base_data = [
        (0.0, 44.2, 46.1, 48.0, 49.9, 51.8, 53.7, 55.6),
        (0.25, 48.9, 50.8, 52.8, 54.7, 56.7, 58.6, 60.6),
```

```

(0.5, 52.4, 54.4, 56.4, 58.4, 60.4, 62.4, 64.4),
(1.0, 68.6, 71.0, 73.4, 75.7, 78.1, 80.5, 82.9),
(2.0, 78.0, 81.0, 84.1, 87.1, 90.2, 93.2, 96.3),
(3.0, 85.0, 88.7, 92.4, 96.1, 99.8, 103.5, 107.2),
(4.0, 90.7, 94.9, 99.1, 103.3, 107.5, 111.7, 115.9),
(5.0, 96.1, 100.7, 105.3, 110.0, 114.6, 119.2, 123.9),
(6.0, 101.17, 106.10, 111.02, 115.95, 120.88, 125.80, 130.73),
(7.0, 105.88, 111.16, 116.45, 121.73, 127.02, 132.31, 137.59),
(8.0, 110.32, 115.97, 121.62, 127.27, 132.91, 138.56, 144.21),
(9.0, 114.53, 120.54, 126.55, 132.57, 138.58, 144.59, 150.60),
(10.0, 118.66, 125.03, 131.41, 137.78, 144.15, 150.53, 156.90),
(11.0, 122.92, 129.65, 136.38, 143.11, 149.84, 156.57, 163.30),
(12.0, 127.82, 134.91, 142.00, 149.08, 156.17, 163.25, 170.34),
(13.0, 133.76, 141.19, 148.62, 156.04, 163.47, 170.90, 178.33),
(14.0, 140.10, 147.80, 155.49, 163.18, 170.87, 178.57, 186.26),
(15.0, 145.55, 153.35, 161.15, 168.96, 176.76, 184.57, 192.37),
(16.0, 149.58, 157.35, 165.13, 172.90, 180.67, 188.44, 196.21),
(17.0, 152.23, 159.87, 167.52, 175.16, 182.81, 190.45, 198.09),
(18.0, 153.73, 161.20, 168.68, 176.15, 183.62, 191.09, 198.56),
]

```

```

region_corrections = {
    'UKR': 0.0,
    'PL': -2.0,
    'TU': -1.0,
    'JA': -3.0,
    'USA': +1.5
}

```

```

try:
    ditina = Ditina.objects.get(id=ditina_id)
    component_age = float(ditina.vik)
    component_height = float(ditina.zrist)
    user_name = f"{ditina.imya} {ditina.prizvische}".strip()
    region = ditina.region or 'UKR'
except (Ditina.DoesNotExist, ValueError, TypeError) as e:
    logger.error(f"Помилка отримання дитини: {e}")
    component_age = None
    component_height = None
    user_name = 'Ваша дитина'
    region = 'UKR'

correction = region_corrections.get(region, 0.0)
logger.debug(f"Region: {region}, Correction: {correction}")

ages = [x[0] for x in base_data]
curves = {
    '-3CB': [round(x[1] + correction, 2) for x in base_data],
    '-2CB': [round(x[2] + correction, 2) for x in base_data],
    '-1CB': [round(x[3] + correction, 2) for x in base_data],
    'Середній': [round(x[4] + correction, 2) for x in base_data],
    '+1CB': [round(x[5] + correction, 2) for x in base_data],
    '+2CB': [round(x[6] + correction, 2) for x in base_data],
    '+3CB': [round(x[7] + correction, 2) for x in base_data],
}

```

```

style_map = {
    '-3CB': ('#999999', 'dot'),
    '-2CB': ('#ff9900', 'dot'),
    '-1CB': ('#cc0000', 'dot'),
    'Середній': ('#1f77b4', 'solid'),
    '+1CB': ('#2ca02c', 'dot'),
    '+2CB': ('#17becf', 'dot'),
    '+3CB': ('#9467bd', 'dot'),
}

```

```

data = []
for label, values in curves.items():
    color, dash = style_map[label]
    trace = go.Scatter(
        x=ages,
        y=values,
        mode='lines',
        name=label,
        line=dict(color=color, width=2, dash=dash)
    )
    data.append(trace)

```

if component_age is not None and component_height is not None:

```

data.append(go.Scatter(
    x=[component_age],
    y=[component_height],
    mode='markers',

```

```

        marker=dict(size=16,          color='#e83e8c',          line=dict(width=2,
color='black')),
        name=user_name,
        hoverinfo='skip' # опціонально: прибирає підказку при наведенні
    ))

```

```

layout = go.Layout(
    title='Графік: Вік — Зріст із кривими нормативів',
    xaxis=dict(title='Вік (роки)', dtick=1, range=[0, 18]),
    yaxis=dict(title='Зріст (см)'),
    plot_bgcolor='#f8f9fa',
    paper_bgcolor='#f8f9fa',
    legend=dict(font=dict(size=12)),
    margin=dict(l=60, r=30, t=60, b=60),
)

```

```

fig = go.Figure(data=data, layout=layout)
graph_html = fig.to_html(full_html=False, include_plotlyjs='cdn')

```

Лістинг А.6 – Реєстрація моделі Ditina в адмін-панелі з можливістю експорту (файл admin.py)

```

class DitinaResource(resources.ModelResource):
    imya = fields.Field(attribute='imya', column_name="Ім'я")
    pryimaie_gormony = fields.Field(
        attribute='pryimaie_gormony',
        column_name="Приймає гормони",
        widget=BooleanWidget()
    )

```

```

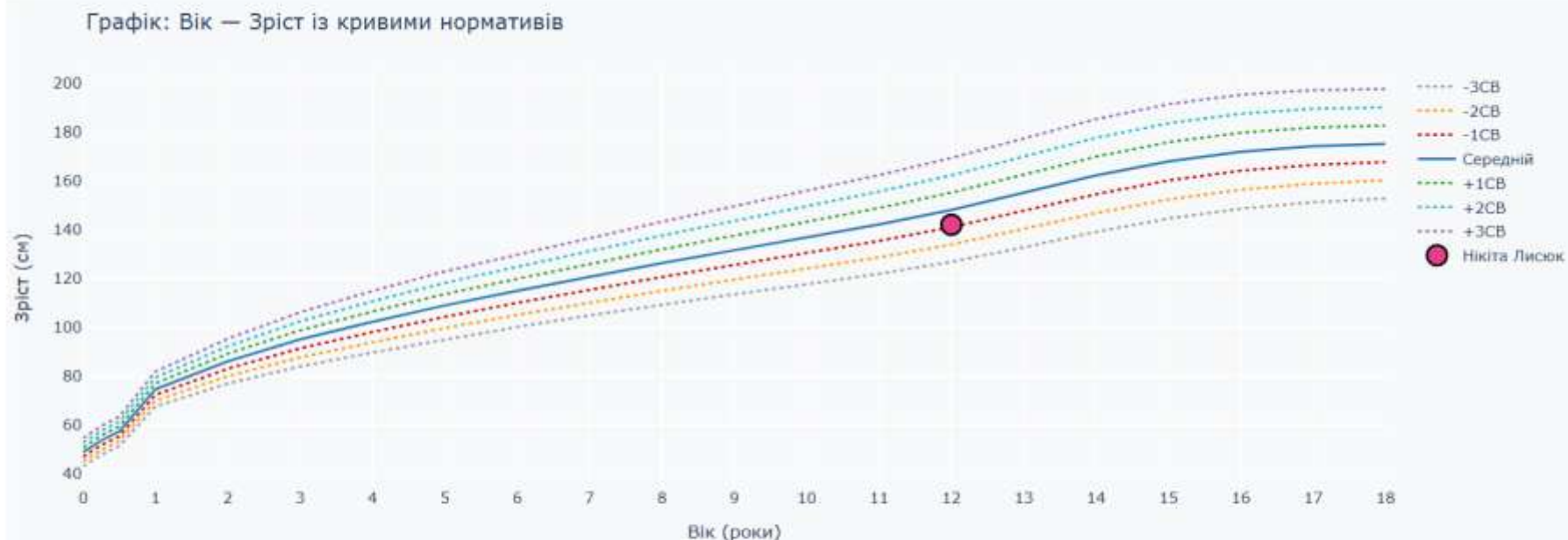
class Meta:
    model = Ditina
    fields = ('id', 'imya', 'prizvische', 'vik', 'zrist', 'region', 'pryimaie_gormony',
'data_stvorennia')
    export_order = fields

def before_export(self, queryset, *args, **kwargs):
    for obj in queryset:
        obj.region = obj.get_region_display()

@admin.register(Ditina)
class DitinaAdmin(ImportExportModelAdmin):
    resource_class = DitinaResource
    list_display = ('id', 'imya', 'prizvische', 'vik', 'zrist', 'region',
'pryimaie_gormony')

```

ДОДАТОК Б – ГРАФІК ЗРОСТУ



Увага: Дитина приймає гормональні препарати. Це може суттєво впливати на ріст і розвиток. **Обов'язково проконсультуйтеся з лікарем** для індивідуальної оцінки та регулярного моніторингу.

Зріст на нижній межі норми: 142.0 см при 12.0 роках. Очікуваний середній зріст: 148.1 см. **Слідкуйте за динамікою** росту дитини впродовж наступних місяців.