

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Система моніторингу та аналізу рівня
захворюваності студентів з прогнозуванням
епідемічних ризиків»**

Виконав студент групи ВТ-21-1
ЗБУНЬ Каріна Сергіївна

Керівник роботи:
ВАКАЛЮК Тетяна Анатоліївна

Рецензент:
ЄФРЕМОВ Юрій Миколайович

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення
Тетяна Вакалюк
«26» лютого 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ЗБУНЬ Каріна Сергіївна**

Керівник роботи: **ВАКАЛЮК Тетяна Анатоліївна**

Тема роботи: **«Система моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків»,**

затверджена Наказом закладу вищої освіти від **«14» лютого 2025 р., №61/с**

Вихідні дані для роботи: **Предметом дослідження є технології та методи проєктування і розробки веб-орієнтованих систем моніторингу здоров'я. Об'єктом дослідження виступають процеси обробки та аналізу медичних даних у навчальному закладі.**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Вакалюк Т.А.	27.02.2025р.	06.03.2025р.
2	Вакалюк Т.А.	06.03.2025р.	24.04.2025р.
3	Вакалюк Т.А.	24.04.2025р.	05.06.2025р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра присвячена розробці веб-орієнтованої системи моніторингу здоров'я студентів та містить пояснювальну записку.

Пояснювальна записка до кваліфікаційної роботи складається з переліку умовних скорочень, вступу, трьох розділів, висновків, списку використаних джерел та додатку.

Текстова частина оформлена на 58 сторінках друкованого тексту. Список використаних джерел включає 26 найменувань і займає 4 сторінки. У роботі представлено 25 рисунків та 16 таблиць.

Загальний обсяг роботи становить 65 сторінок.

У випускній кваліфікаційній роботі розглядається задача побудови інформаційної системи для моніторингу медичних показників студентів у закладах освіти. Основну увагу приділено архітектурі системи, аналізу аналогічних рішень, вибору інструментальних засобів та реалізації функціональних модулів, зокрема системи авторизації, візуалізації аналітики, автоматичного сповіщення про критичні показники та прогнозування епідеміологічної ситуації.

Запропонована система базується на клієнт-серверній архітектурі. Клієнтська частина реалізована за допомогою ReactJS, серверна частина — з використанням NestJS, а зберігання даних забезпечується за допомогою MySQL. Аналітичний модуль із машинним навчанням реалізовано на мові Python з використанням бібліотек Scikit-learn.

Система дозволяє ефективно виявляти ризики, своєчасно реагувати на загрози та автоматизує медичний моніторинг у навчальному середовищі.

Ключові слова: МОНІТОРИНГ ЗДОРОВ'Я, МАШИННЕ НАВЧАННЯ, БАЗА ДАНИХ, ПРОГНОЗУВАННЯ, ВЕБ-ДОДАТОК.

					Житомирська політехніка.25.121.9.000 - ПЗ						
Змн.	Арк.	№ докум.	Підпис	Дата							
Розроб.		Збунь К.С.			Система моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків			Лім.	Арк.	Аркушів	
Перевір.		Вакалюк Т.А								2	73
Керівник								ФІКТ Гр. ВТ-21-1			
Н. контр.											
Зав. каф.											

ABSTRACT

The bachelor's qualification thesis is dedicated to the development of a web-based student health monitoring system and includes an explanatory note.

The explanatory note to the qualification thesis consists of a list of abbreviations, an introduction, three chapters, conclusions, a list of references, and an appendix.

The main text is presented on 58 pages of printed text. The list of references includes 26 sources and occupies 4 pages. The work contains 25 figures and 16 tables.

The total volume of the work is 65 pages.

The thesis addresses the task of building an information system for monitoring students' medical indicators in educational institutions. The focus is on the system's architecture, analysis of similar solutions, selection of development tools, and implementation of functional modules, including user authorization, data analytics visualization, automatic critical health alerts, and epidemiological risk forecasting.

The proposed system is based on a client-server architecture. The client-side is implemented using ReactJS, the server-side with NestJS, and data storage is handled by MySQL. The analytical module, powered by machine learning, is implemented in Python using the Scikit-learn library.

The system enables effective risk detection, timely response to threats, and automation of health monitoring in the educational environment.

Keywords: HEALTH MONITORING, MACHINE LEARNING, DATABASE, FORECASTING, WEB APPLICATION.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				4
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	7
ВСТУП.....	8
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ СИСТЕМИ МОНІТОРИНГУ ТА АНАЛІЗУ РІВНЯ ЗАХВОРЮВАНOSTІ СТУДЕНТІВ З ПРОГНОЗУВАННЯМ ЕПІДЕМІЧНИХ РИЗИКІВ.....	10
1.1. Визначення та постановка задачі	10
1.2. Аналіз аналогів програмного продукту.....	12
1.3. Визначення архітектури програмного забезпечення.....	17
1.4. Обґрунтування вибору інструментальних засобів та вимоги до програмного забезпечення.....	19
Висновки до першого розділу	22
РОЗДІЛ 2. ПРОЄКТУВАННЯ СИСТЕМИ МОНІТОРИНГУ ТА АНАЛІЗУ РІВНЯ ЗАХВОРЮВАНOSTІ СТУДЕНТІВ З ПРОГНОЗУВАННЯМ ЕПІДЕМІЧНИХ РИЗИКІВ.....	23
2.1. Визначення варіантів використання та об'єктно-орієнтованої структури системи	23
2.2. Розробка бази даних системи.....	34
2.3. Проектування та реалізація алгоритмів роботи системи	38
Висновки до другого розділу	43
РОЗДІЛ 3. ІТЕРФЕЙС ТА ПОРЯДОК РОБОТИ СИСТЕМИ МОНІТОРИНГУ ТА АНАЛІЗУ РІВНЯ ЗАХВОРЮВАНOSTІ СТУДЕНТІВ З ПРОГНОЗУВАННЯМ ЕПІДЕМІЧНИХ РИЗИКІВ	45
3.1. Порядок встановлення та налаштування параметрів системи	45

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

3.2. Структура інтерфейсу системи моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків.....	47
3.3. Тестування роботи програмного комплексу	56
Висновки до третього розділу	58
ВИСНОВКИ.....	60
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	61
ДОДАТКИ.....	65

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

TS – TypeScript.

БД – база даних.

API – Application programming interface, дослівно: Інтерфейс прикладного програмування.

REST API – Representational State Transfer Application Programming Interface.

EMR – Electronic Medical Record

EHR – Electronic Health Record

JSON – JavaScript Object Notation

JSONB – JSON Binary

ORM – Object-Relational Mapping

IoT – Internet of Things

npm – Node Package Manager

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. У сучасних реаліях цифровізації освіти особливого значення набуває розробка систем моніторингу стану здоров'я студентів. Своєчасний контроль медичних показників дозволяє не лише забезпечити індивідуальний підхід до здоров'я кожного студента, а й оперативно реагувати на можливі епідеміологічні загрози в навчальному середовищі.

Відсутність централізованої платформи для обліку, аналізу та візуалізації таких даних ускладнює процес прийняття обґрунтованих управлінських рішень. Це актуалізує потребу у впровадженні сучасної веб-системи, яка дозволить проводити аналітику медичних показників, автоматично виявляти критичні значення, прогнозувати спалахи захворюваності та надавати рекомендації щодо запобіжних заходів.

Метою випускної кваліфікаційної роботи є проєктування та розробка веб-орієнтованої системи моніторингу здоров'я студентів для закладів вищої освіти. Система має забезпечити збір, обробку, аналіз і візуалізацію медичних даних, автоматичне оповіщення відповідальних осіб про критичні стани, а також реалізацію алгоритмів прогнозування потенційних епідемій.

Для досягнення цієї мети в роботі розглянуто такі **завдання**:

- визначення основних функціональних вимог до системи та узагальнення її структури;
- аналіз існуючих аналогічних рішень і технологічних підходів;
- побудова архітектури, яка задовольняє вимоги масштабованості, безпеки та продуктивності;
- обґрунтування вибору інструментальних засобів для реалізації системи;
- розробка клієнтської, серверної та аналітичної частин системи;
- тестування функціональності та оцінка ефективності реалізованого рішення.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				8
Змн.	Арк.	№ докум.	Підпис	Дата		

Об'єктом дослідження виступають процеси обробки та аналізу медичних даних у навчальному закладі.

Предметом дослідження є технології та методи проектування і розробки веб-орієнтованих систем моніторингу здоров'я.

За матеріалами кваліфікаційної роботи опубліковано статтю у науковому періодичному виданні категорії А, яке індексується у Web of Science [1].

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ СИСТЕМИ МОНІТОРИНГУ ТА АНАЛІЗУ РІВНЯ ЗАХВОРЮВАНOSTІ СТУДЕНТІВ З ПРОГНОЗУВАННЯМ ЕПІДЕМІЧНИХ РИЗИКІВ

1.1. Визначення та постановка задачі

Основною метою є розробка веб-системи для моніторингу стану здоров'я студентів, яка дозволить автоматизувати процес збору, аналізу та візуалізації медичних показників, забезпечить швидке реагування на критичні стани та прогнозування можливих епідемій.

Сучасні навчальні заклади стикаються з проблемою відсутності ефективного інструменту для збору та аналізу даних про стан здоров'я студентів. Наявні системи або працюють у ручному режимі, або є недостатньо зручними для інтеграції в навчальний процес. Це ускладнює своєчасне прийняття рішень щодо карантинних заходів та контролю загальної епідеміологічної ситуації.

Також існують такі важливі проблеми. По-перше, спостерігається недостатня цифровізація процесів у навчальних закладах: більшість вищих навчальних закладів не мають централізованих платформ для збереження та аналізу медичних даних. По-друге, аналіз великих обсягів даних вручну є надто складним завданням — медичні працівники часто не мають можливості оперативно обробляти та аналізувати показники кожного студента, що може призводити до несвоєчасного виявлення проблем.

Для вирішення вказаних проблем необхідно розробити веб-сайт, який забезпечить зручне та швидке відображення даних про стан здоров'я студентів із бази даних. Ця система має забезпечувати такі можливості:

1. Візуалізація інформації — створення інтуїтивно зрозумілого інтерфейсу для перегляду та аналізу даних у вигляді графіків.
2. Динамічне оновлення даних — забезпечення автоматичного оновлення інформації на сторінці.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

3. Сповіщення про критичні стани — реалізація механізму автоматичного виявлення та відображення критичних значень стану здоров'я.
4. Багаторівневий доступ — забезпечення розмежованого доступу до даних для студентів та медичних працівників.
5. Прогнозування епідемічних ризиків — передбачає розробку інтелектуального алгоритму, що аналізує як накопичені, так і поточні медичні показники студентів з метою виявлення ймовірних спалахів захворюваності. У випадку фіксації підвищеного ризику система автоматично формує повідомлення, в якому зазначається рівень епідемічної загрози для конкретної академічної групи, та надсилає його відповідальним особам через електронну пошту.

Розробка системи передбачає проходження кількох ключових етапів. На першому з них буде спроектовано інтерфейс веб-застосунку: сформовано загальну концепцію дизайну та створено макети основних сторінок. Далі відбудеться реалізація архітектури клієнтської частини, яка включатиме структуру додатку та підключення необхідних бібліотек для забезпечення інтерактивної взаємодії.

Після цього розпочнеться розробка серверної частини, що відповідатиме за обробку запитів і взаємодію з базою даних. Буде створено REST API, який забезпечуватиме зв'язок між клієнтським інтерфейсом та внутрішніми сервісами. Також буде спроектовано реляційну базу даних для збереження медичних показників і параметрів довілля. Важливою вимогою є забезпечення обміну даними в реальному часі, оскільки сенсорні пристрої постійно надсилатимуть нові показники, які необхідно оперативно обробляти.

Ключовим етапом стане реалізація основного функціоналу системи. Передбачається розробка модулів для відображення, збору та обробки медичних даних. Окремо буде впроваджено аналітичний модуль із використанням алгоритмів машинного навчання для аналізу динаміки показників і прогнозування епідемічних ризиків. Крім того, буде реалізовано систему автоматичних сповіщень, яка

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

генеруватиме повідомлення у випадках виявлення критичних значень або підвищеної загрози захворюваності.

Завершальним етапом стане тестування функціональності системи. Базу даних буде заповнено згенерованими тестовими даними, наближеними до реальних, за попередній рік. Передбачається проведення ручного тестування кожного модуля з перевіркою їхньої роботи в умовах, максимально наближених до реальних. Особливу увагу буде приділено перевірці взаємодії між клієнтською та серверною частинами, щоб переконатися в стабільності, швидкодії та узгодженості всіх компонентів системи.

У результаті буде створено веб-систему, що забезпечить централізоване збереження, аналіз і прогнозування показників здоров'я студентів. Це дозволить адміністрації університету та медичному персоналу ефективно контролювати епідеміологічну ситуацію в навчальному закладі та своєчасно реагувати на можливі загрози.

Запровадження системи дозволить суттєво підвищити рівень цифровізації процесів моніторингу здоров'я студентів, а також значно зменшити навантаження на медичних працівників, які щоденно обробляють великий обсяг даних. Крім того, система забезпечить автоматизоване виявлення потенційних загроз і оперативне реагування на них, що є критично важливим у разі виникнення ознак спалаху захворюваності. Аналітичні можливості платформи сприятимуть покращенню загального рівня здоров'я студентів завдяки реалізації превентивних заходів, сформованих на основі даних. Таким чином, розробка системи сприятиме покращенню контролю стану здоров'я в університеті та дозволить своєчасно запобігати можливим епідеміям.

1.2. Аналіз аналогів програмного продукту

Розробка системи моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків є актуальним напрямом у сфері інформаційних технологій. Однак, у відкритому доступі немає готових рішень, що повністю відповідають концепції розроблюваного проекту. Найбільш схожими за

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

функціоналом системами можна вважати фітнес-платформи та системи електронних медичних карт (EMR/EHR). Проте жодна з них не орієнтована на специфіку освітнього середовища, де ключовим є не лише індивідуальний моніторинг, а й груповий аналіз для прийняття управлінських рішень. Система має враховувати особливості навчального процесу, розподіл студентів за академічними групами та необхідність централізованого доступу до даних.

Одним із популярних фітнес-рішень є Google Fit — платформа для відстеження активності та здоров'я користувачів [2] (рис. 1.1). Вона була створена компанією Google для збору даних із фітнес-пристроїв, смартфонів та носимих гаджетів. Google Fit інтегрується з багатьма сторонніми застосунками та сенсорами, що дозволяє отримувати різноманітну інформацію про фізичний стан користувачів.

Серед основних переваг Google Fit варто відзначити автоматичний збір даних: платформа здатна отримувати інформацію про фізичну активність користувача, таку як кількість кроків, витрачені калорії, пройденої дистанції та час активності, без потреби у ручному введенні. Ці дані надходять із вбудованих сенсорів смартфонів, розумних годинників, фітнес-трекерів та інших сумісних пристроїв, що підтримують API Google Fit. Крім того, Google Fit дозволяє відстежувати основні показники здоров'я користувача, що сприяє кращому розумінню його фізичного стану. Ще однією важливою перевагою є можливість візуалізації та аналітики: інформація подається у вигляді графіків і діаграм, що дає змогу аналізувати зміни стану здоров'я в динаміці. Історичні дані зберігаються і можуть бути використані для довгострокового аналізу фізичної активності та її впливу на здоров'я.

Попри значні переваги, платформа Google Fit має низку обмежень, які не дозволяють використовувати її як основу для системи моніторингу здоров'я в освітньому середовищі. Насамперед, Google Fit орієнтована виключно на індивідуальне використання. Її основна мета — відстеження фізичної активності конкретного користувача та надання йому персоналізованої статистики.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

Платформа не передбачає можливості централізованого збору та обробки даних для груп користувачів, що є критично важливим у контексті навчального закладу, де необхідно мати доступ до агрегованої інформації про стан здоров'я студентів певної групи, факультету чи всього університету.

Крім того, додаток не використовує зібрані дані для прогнозування можливих ризиків для здоров'я. У системі відсутні механізми аналітики, які б на основі накопиченої історичної інформації могли визначати тенденції до погіршення стану чи передбачати спалахи захворювань. Відтак, платформа не здатна виконувати функцію превентивного інструменту, що є ключовим у задачі своєчасного реагування на потенційні епідемічні загрози.

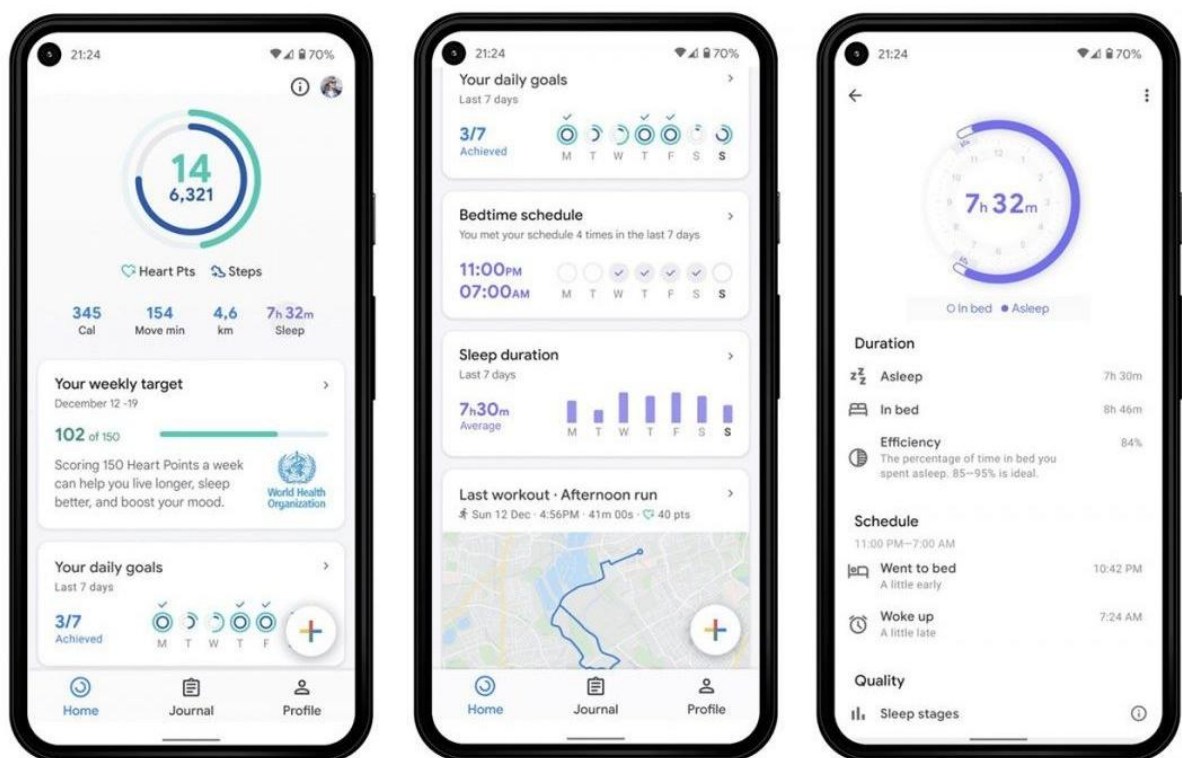


Рис. 1.1 - Google Fit

Іншим напрямком є системи електронних медичних карт (EMR/EHR), що використовуються для збереження, обробки та управління медичними даними пацієнтів у лікарнях, поліклініках та інших медичних закладах. Вони дозволяють автоматизувати медичний документообіг, підвищити точність діагностики та забезпечити швидкий доступ до інформації про стан здоров'я пацієнта.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

Основними перевагами систем електронних медичних карт є централізоване збереження та зручна доступність медичних даних, що забезпечує ефективну взаємодію між медичними працівниками та пацієнтами. Такі системи дозволяють автоматизувати значну частину медичних процесів і мінімізувати потребу в паперовій документації, що значно зменшує навантаження на персонал і знижує ризик втрати інформації.

Важливою особливістю є інтеграція з діагностичними приладами та лабораторіями: сучасні системи EMR/EHR здатні автоматично отримувати результати з медичних приладів, таких як рентген, комп'ютерна томографія, магнітно-резонансна томографія, а також з лабораторій, де проводяться клінічні аналізи. Інформація надходить безпосередньо в електронну картку пацієнта, що усуває необхідність ручного введення даних та суттєво зменшує ймовірність помилок. Завдяки цьому скорочується час обробки результатів аналізів, що підвищує швидкість реагування медичних працівників на можливі ускладнення.

Крім того, системи електронних медичних записів сприяють покращенню діагностики та прийняття клінічних рішень. Доступ до статистичних даних про захворювання дозволяє швидко виявляти тенденції розвитку хвороб у певній популяції та вчасно впроваджувати відповідні заходи. Вбудовані алгоритми підтримки прийняття рішень допомагають лікарям обирати оптимальні методи лікування на основі аналізу накопиченої інформації.

Попри всі переваги, системи електронних медичних карт мають і певні недоліки. Одним із головних викликів є висока вартість впровадження та підтримки таких рішень, оскільки створення комплексної EMR/EHR-системи потребує значних фінансових і технічних ресурсів для розробки, навчання персоналу та подальшого обслуговування. Крім того, більшість наявних рішень не адаптовані для специфіки використання в навчальних закладах, що обмежує їх ефективне застосування у сфері освіти.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

Порівняльна характеристика систем наведена нижче (табл. 1.1), де можна побачити ключові відмінності між Google Fit, EMR/EHR та системою моніторингу здоров'я студентів.

Таблиця 1.1

Порівняльна характеристика систем

Характеристика	EMR/EHR	Google Fit	Система моніторингу здоров'я студентів
Призначення	Збереження та обмін медичних записів пацієнтів	Відстеження фізичної активності індивідуального користувача	Моніторинг здоров'я студентів та прогнозування епідемій
Джерела даних	Лікарі, медичні заклади, лабораторні аналізи	Смартфони, фітнес-трекери, API інших додатків	Автоматичні сенсори
Прогнозування	Доступ до статистики захворювань дозволяє швидко визначати тенденції розвитку хвороб серед населення	Відсутнє	Використання історичних даних для аналізу та прогнозування епідемій
Адміністрування	Медичний персонал може вводити та використовувати дані під час планування лікування, пацієнти можуть переглядати свої дані	Користувач контролює свої дані самостійно	Адміністратори можуть аналізувати загальні тенденції, студенти можуть переглядати свої дані

Аналіз існуючих рішень показав, що Google Fit пропонує ефективний збір та візуалізацію фізичних показників, але не має централізованої моделі моніторингу груп користувачів. У свою чергу, EMR/EHR системи фокусуються на веденні медичних записів, проте не займаються профілактикою або прогнозуванням. Жодна з наявних платформ не забезпечує комплексного підходу до моніторингу здоров'я студентів, їхнього поточного стану та ймовірності поширення захворювань.

Таким чином, для вирішення задачі моніторингу здоров'я студентів необхідно розробити власну систему, яка поєднуватиме переваги аналогів з

можливістю прогнозування ризиків епідемій та централізованого керування даними студентів.

1.3. Визначення архітектури програмного забезпечення

Архітектура програмного забезпечення системи моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків базується на клієнт-серверному підході, що дозволяє розподілити функціональність між різними компонентами для забезпечення ефективності та гнучкості. Основними частинами системи є клієнтський інтерфейс, серверна частина, база даних, аналітичний модуль та система сповіщень.

Клієнтська частина відповідає за взаємодію користувачів із системою. Через веб-інтерфейс студенти та медичні працівники можуть переглядати медичні показники, аналізувати динаміку змін і отримувати відповідні повідомлення. Дизайн інтерфейсу орієнтований на зручність та швидкий доступ до ключової інформації, а інтеграція інструментів візуалізації дозволяє наочно відображати медичні показники та тенденції змін у стані здоров'я студентів.

Серверна частина виконує роль центрального обробника запитів, забезпечуючи зв'язок між клієнтським інтерфейсом, базою даних та аналітичним модулем. Сервер опрацьовує запити користувачів, виконує бізнес-логіку та забезпечує безперебійну взаємодію між усіма компонентами системи.

База даних є фундаментом усієї системи, оскільки в ній зберігаються всі ключові дані: медичні показники студентів та записи про екологічні умови в навчальних приміщеннях. Вона підтримує складні запити та забезпечує швидкий доступ до великих масивів інформації.

Аналітичний модуль займається обробкою накопичених даних і відповідає за формування прогнозів щодо можливих спалахів захворюваності. У його основі — алгоритми машинного навчання, які виявляють аномальні значення, аналізують динаміку змін показників і оцінюють рівень ризику для кожної академічної групи. Це дозволяє системі діяти проактивно, тобто не лише реагувати на вже зафіксовані критичні значення, а й передбачати потенційні загрози заздалегідь.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

Система сповіщень є важливою складовою, яка відповідає за інформування користувачів. Якщо у студента фіксуються критичні показники, система надсилає йому електронний лист із відповідним попередженням і рекомендацією звернутися до медичного пункту університету. Окрім цього, відповідальні особи (медики або адміністратори) отримують push- або email-сповіщення про всі випадки, що потребують уваги. Якщо система на основі аналізу даних формує прогноз про підвищений епідемічний ризик, адміністрація університету отримає відповідне сповіщення із зазначеним рівнем загрози. У такому разі можуть бути оперативно прийняті управлінські рішення, наприклад, про переведення окремих груп або навіть факультетів на дистанційне навчання для запобігання поширенню хвороби.

На рисунку нижче (рис. 1.2) представлена діаграма компонентів системи, яка ілюструє взаємодію між основними модулями. Вона допомагає краще зрозуміти, як система функціонує в цілому та як дані передаються між її складовими.

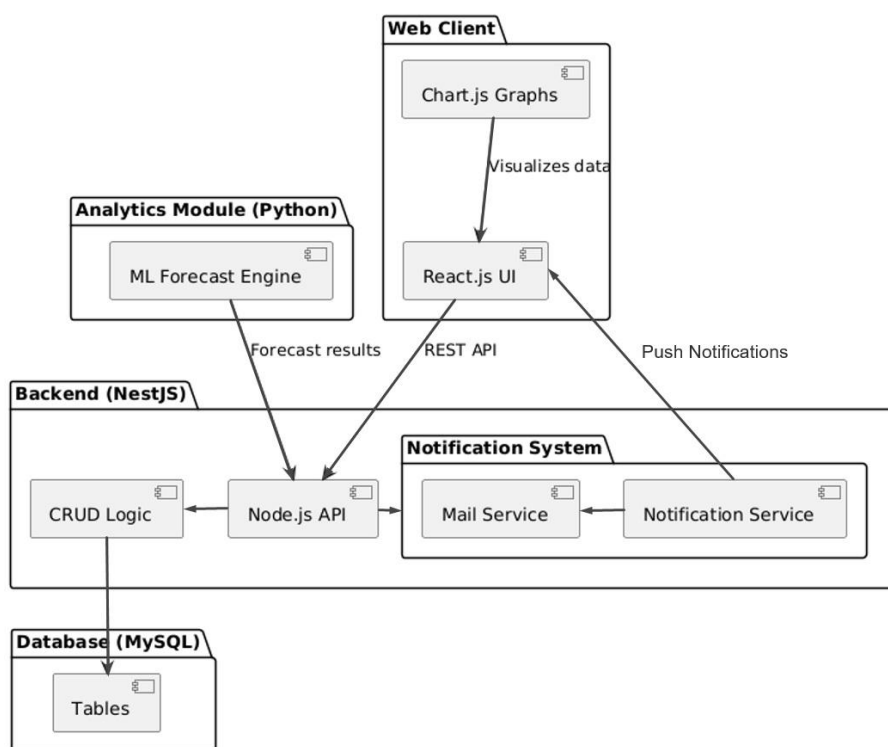


Рис. 1.2 - Діаграма компонентів системи.

Така архітектура забезпечує високу продуктивність, безпеку та гнучкість у подальшому розширенні функціоналу. Завдяки чітко визначеній взаємодії між компонентами система може ефективно відстежувати стан здоров'я студентів,

своєчасно виявляти потенційні ризики та запобігати можливим епідеміям у навчальному закладі.

1.4. Обґрунтування вибору інструментальних засобів та вимоги до програмного забезпечення

Система моніторингу та аналізу рівня захворюваності студентів розроблена з урахуванням функціональних вимог, що визначають вибір технологічного стеку. Основними критеріями стали продуктивність, масштабованість, зручність використання, а також можливість швидкої обробки та аналізу медичних даних.

Для реалізації серверної логіки в системі обрано фреймворк NestJS [3], який забезпечує зручну модульну структуру, підтримку масштабування та інтеграцію з сучасними інструментами. Його можливості дозволяють ефективно організувати обробку запитів, взаємодію з базою даних і реалізацію бізнес-логіки, включаючи контроль доступу до функціональних частин системи.

Для збору та збереження даних про здоров'я студентів у реальному часі необхідна стабільна база даних, що підтримує складні зв'язки між таблицями та здатна обробляти великі обсяги інформації. Було обрано MySQL, оскільки ця система керування базами даних забезпечує високу продуктивність, надійність і підтримку складних SQL-запитів, що відповідає вимогам проєкту.

Щоб забезпечити зручну візуалізацію медичних показників, система використовує React.js [4] у поєднанні з Chart.js [5]. Це дозволяє створювати динамічний інтерфейс із відображенням графіків та інтерактивних елементів, що полегшує аналіз даних для користувачів.

Оскільки одним із ключових завдань є аналіз історичних даних і прогнозування ризику спалахів захворюваності, для реалізації аналітичного модуля було обрано Python [6] з використанням бібліотеки Scikit-learn.

Щоб користувачі могли оперативно отримувати інформацію про критичні показники, у систему інтегровано механізм push- та email-сповіщень. Для цього використовується окремий сервіс сповіщень, який взаємодіє з поштовим модулем на основі бібліотеки @nestjs-modules/mailer та сокет-шлюзом.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				19
Змн.	Арк.	№ докум.	Підпис	Дата		

У процесі розробки використовувалося середовище WebStorm, яке забезпечує зручну інтеграцію з обраними технологіями, ефективний контроль коду та підтримку TypeScript [7]. Для тестування API-запитів застосовувався Postman, що дозволяє швидко перевіряти взаємодію між клієнтом і сервером та знаходити можливі помилки в роботі бекенду.

Окрім цього, для вибору бази даних було проведено аналіз популярних рішень, таких як MySQL [8] як основну реляційну СУБД проєкту, а також альтернативи — PostgreSQL [9] з розширеною підтримкою JSONB та MongoDB [10], орієнтовану на документо-орієнтоване зберігання. Детальне порівняння можливостей цих систем наведено нижче (табл. 1.2). PostgreSQL відзначається потужними аналітичними можливостями та розширеною підтримкою складних типів даних, зокрема JSONB, що робить її зручною для побудови складних запитів. MongoDB, у свою чергу, пропонує гнучкість у роботі з неструктурованими даними та масштабованість у розподілених системах, однак вимагає додаткової конфігурації для забезпечення безпеки та цілісності даних.

За результатами аналізу було обрано MySQL як оптимальний варіант для реалізації проєкту, оскільки вона поєднує в собі високу продуктивність при типовому навантаженні, підтримку реляційної моделі, гнучку структуру SQL-запитів, а також простоту в налаштуванні й інтеграції з серверною частиною системи, розробленою на базі NestJS. Такий вибір дозволяє забезпечити стабільну та ефективну роботу системи в умовах навчального закладу.

Таким чином, вибір кожного інструмента був здійснений з урахуванням його технічних можливостей, відповідності функціональним та нефункціональним вимогам системи, а також стабільності та підтримки в професійному середовищі розробників. Застосування перевірених технологій, таких як MySQL для зберігання даних, NestJS для побудови серверної логіки, React.js для фронтенду та Scikit-learn для реалізації алгоритмів машинного навчання, дозволяє створити надійну архітектуру, готову до масштабування та подальшого розвитку. Такий підхід гарантує не лише продуктивність і безпеку системи, а й забезпечує її гнучкість,

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				20
Змн.	Арк.	№ докум.	Підпис	Дата		

адаптивність до змін та ефективне вирішення завдань моніторингу здоров'я студентів, включаючи оперативне виявлення критичних станів та прогнозування потенційних епідеміологічних загроз.

Таблиця 1.2

Порівняння баз даних

Критерій	MongoDB	PostgreSQL	MySQL
Тип СУБД	Документна (NoSQL)	Реляційна (SQL)	Реляційна (SQL)
Структура даних	JSON-документи	Таблична, підтримка JSONB	Таблична, базова підтримка JSON
Масштабованість	Горизонтальна	Горизонтальна і вертикальна	Вертикальна (стабільна), з можливістю шардінгу
Продуктивність на складних запитах	Висока на неструктурованих даних	Висока продуктивність на складних аналітичних запитах	Висока продуктивність для більшості CRUD-операцій
Робота з великими обсягами даних	Оптимізована для великих неструктурованих обсягів	Сильна підтримка великих масивів даних	Надійна при помірному навантаженні, добре масштабується вертикально
Підтримка JSON	Основний формат зберігання	Так (JSONB — оптимізований для запитів та зберігання)	Так (є базова підтримка з можливістю фільтрації та індексації)
Безпека даних	Вимагає додаткової конфігурації	Розширені можливості шифрування та аудиту	Надійна безпека, розвинені механізми контролю доступу
Резервне копіювання та відновлення	Потребує налаштування зовнішніх сервісів	Вбудована підтримка реплікації та резервного копіювання	Добре підтримується, є перевірені інструменти для бекапу і реплікації

При визначенні функціональних вимог до системи, а також під час обґрунтування архітектурного підходу було проаналізовано сучасні дослідження у

сфері освітнього середовища, моніторингу здоров'я, IoT та машинного навчання в епідеміології [11–24], що охоплюють як технічні, так і педагогічні аспекти впровадження подібних інформаційних систем.

Висновки до першого розділу

У першому розділі було визначено основну концепцію проєкту та його актуальність у сфері моніторингу здоров'я студентів. Проведено аналіз існуючих рішень, зокрема фітнес-платформ, систем електронних медичних карт (EMR/EHR) та інструментів збору медичних даних. Визначено їхні ключові переваги та недоліки у контексті використання для моніторингу стану здоров'я в освітніх закладах.

На основі цього аналізу сформульовано вимоги до власної системи, що включають автоматичний збір даних, прогнозування ризиків захворюваності та інтеграцію з аналітичними модулями. Обґрунтовано вибір клієнт-серверної архітектури, яка забезпечує високу продуктивність, гнучкість та масштабованість.

Також розглянуто основні інструментальні засоби та програмні технології, необхідні для реалізації системи. Це дозволить створити ефективне рішення для збору, аналізу та візуалізації медичних показників студентів, сприяючи оперативному реагуванню на потенційні загрози здоров'ю.

		<i>Збунь К.С.</i>			<i>Житомирська політехніка.25.121.9.000 - ПЗ</i>	<i>Арк.</i>
		<i>Вакалюк Т.А.</i>				
<i>Змн.</i>	<i>Арк.</i>	<i>№ докум.</i>	<i>Підпис</i>	<i>Дата</i>		<i>22</i>

РОЗДІЛ 2. ПРОЄКТУВАННЯ СИСТЕМИ МОНІТОРИНГУ ТА АНАЛІЗУ РІВНЯ ЗАХВОРЮВАНOSTІ СТУДЕНТІВ З ПРОГНОЗУВАННЯМ ЕПІДЕМІЧНИХ РИЗИКІВ

2.1. Визначення варіантів використання та об'єктно-орієнтованої структури системи

У цьому розділі подано детальний огляд варіантів використання системи, визначено основні ролі користувачів, їх функціональні потреби, а також описано загальну архітектуру системи на основі об'єктно-орієнтованого підходу. Проєктована система повинна відповідати як функціональним, так і нефункціональним вимогам, серед яких ключовими є: точність і своєчасність збору даних, можливість їх аналізу та прогнозування, а також зручне відображення результатів у вигляді інтерактивної інформації, зрозумілої для кінцевих користувачів.

Архітектурно система поділяється на три основні логічні модулі, кожен з яких виконує свою специфічну роль. Клієнтський модуль забезпечує взаємодію користувача з інформацією, отриманою з бази даних, і реалізується у вигляді зручного веб-інтерфейсу, через який користувач може переглядати свої медичні показники. Адміністративний модуль надає можливість медичним працівникам переглядати статистичні графіки, аналізувати зміни показників у динаміці та приймати обґрунтовані рішення на основі актуальних даних. Аналітичний модуль відповідає за обробку накопичених медичних записів, застосовує алгоритми машинного навчання для прогнозування ризику спалахів захворюваності та автоматично генерує сповіщення у випадках виявлення потенційної загрози.

У системі передбачено дві основні ролі користувачів. Перша — студент, який має доступ до персональних медичних даних, може переглядати історію власних записів, бачити зміни у показниках та отримувати автоматичні сповіщення у разі фіксації критичних значень. Друга роль — медичний працівник, який здійснює моніторинг загального стану студентів, аналізує динаміку захворюваності, взаємодіє з адміністрацією закладу освіти та приймає рішення щодо подальших дій

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				23
Змн.	Арк.	№ докум.	Підпис	Дата		

— наприклад, направлення студента на медичне обстеження чи ініціювання заходів при виявленні ризику епідемії.

Бізнес-вимоги:

1. Мета проєкту – створити платформу для моніторингу захворюваності, яка дозволить приймати обґрунтовані управлінські рішення в реальному часі.
2. Система має забезпечити швидкий доступ до аналітики та історії захворюваності.

Вимоги зовнішніх користувачів (Студент):

Студент має доступ до особистої інформації, що стосується стану здоров'я. Він повинен мати змогу переглядати історію власних медичних показників. У випадках, коли фіксуються критичні значення, система автоматично надсилає студенту сповіщення з відповідними рекомендаціями, зокрема щодо необхідності звернення до медичного пункту.

Вимоги зовнішніх користувачів (Медичний працівник):

Медичний працівник потребує ширшого функціоналу для виконання своїх обов'язків. Йому має бути надано доступ до даних усіх студентів. Крім того, система повинна надавати можливість перегляду зведеної статистики у зручному форматі — у вигляді графіків, що дозволяє швидко оцінювати загальний стан здоров'я в університеті. У разі виявлення критичних показників у студента або зростання захворюваності в окремій академічній групі, медичний працівник має оперативно отримувати відповідні сповіщення для вжиття подальших заходів.

Характеристика об'єкта комп'ютеризації

Навчальний заклад із великою кількістю студентів, медичним пунктом та адміністративним персоналом, який потребує системного контролю стану здоров'я студентів і прогнозування ризиків масових захворювань.

Функціональні вимоги

Функціональні вимоги до системи формують логічний ланцюг її основних можливостей, які охоплюють повний цикл роботи з медичними даними студентів

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				24
Змн.	Арк.	№ докум.	Підпис	Дата		

— від моменту доступу до системи й до прийняття рішень на основі прогнозної аналітики. На рисунку 2.3 зображено послідовну діаграму, яка ілюструє структуру функціональності у вигляді спрямованої стрілки, що підкреслює залежність кожного етапу від попереднього.



Рис. 2.3 - Схематичне зображення функціональних вимог до системи

Робота системи починається з реалізації рольового доступу, що дозволяє розмежувати функціонал між студентами та медичними працівниками відповідно до їхніх повноважень. Після авторизації користувачі взаємодіють з інтерфейсом, що дозволяє вводити або переглядати медичні показники. Всі зібрані дані автоматично зберігаються у базі й відображаються в системі у зручному вигляді.

Наступним етапом є візуалізація даних. Завдяки інтегрованим засобам побудови графіків користувачі можуть швидко аналізувати динаміку показників, виявляти відхилення та оцінювати загальний стан здоров'я. Для підвищення інформативності система реалізує механізм автоматичного позначення критичних значень, що дозволяє миттєво виявляти тривожні стани організму.

Після фіксації критичних параметрів активується система сповіщень, яка надсилає відповідні повідомлення у вигляді push або email. Це дозволяє оперативно інформувати як самих студентів, так і медичний персонал про необхідність реагування.

Фінальним елементом функціонального циклу є блок прогнозування, який ґрунтується на аналізі історичних медичних записів. Використовуючи алгоритми машинного навчання, система здатна оцінювати рівень епідеміологічного ризику для кожної навчальної групи та заздалегідь попереджати про можливість спалахів захворюваності.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				25
Змн.	Арк.	№ докум.	Підпис	Дата		

Таким чином, усі функціональні можливості системи тісно пов'язані між собою і утворюють замкнутий цикл, спрямований на автоматизацію моніторингу здоров'я студентів та підтримку управлінських рішень у навчальному закладі.

Нефункціональні вимоги

Нефункціональні вимоги визначають якісні характеристики системи, що впливають на зручність її використання, надійність, стабільність у роботі та здатність до масштабування. Вони не пов'язані безпосередньо з реалізацією функціоналу, однак відіграють ключову роль у забезпеченні цілісності й ефективності програмного забезпечення в умовах реального середовища експлуатації.

На рисунку 2.4 представлено узагальнену діаграму, яка візуалізує чотири основні напрями нефункціональних вимог: сприйняття, надійність, продуктивність та експлуатацію. Кожен із цих напрямів охоплює окремий аспект очікуваної поведінки системи під час її використання.

У блоці сприйняття акцент зроблено на доступності та зручності інтерфейсу для кінцевих користувачів. Система має бути інтуїтивно зрозумілою й адаптованою до різних категорій користувачів — студентів та медичних працівників — без необхідності в додатковому навчанні. Очікується, що час освоєння базового функціоналу не перевищуватиме 15 хвилин для студентів та 30 хвилин для медичного персоналу.

У свою чергу, надійність системи полягає у забезпеченні безвідмовної роботи протягом щонайменше 20 діб, а також у мінімізації кількості помилок — допустимою вважається лише одна помилка на 1000 запитів. Це дозволяє гарантувати стійкість програмного комплексу навіть у випадку активного використання великою кількістю користувачів.

Продуктивність передбачає стабільну роботу системи при одночасному підключенні щонайменше 100 активних користувачів. Такий показник є критично важливим для впровадження системи у великому навчальному закладі, де значна кількість студентів може використовувати платформу одночасно.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				26
Змн.	Арк.	№ докум.	Підпис	Дата		

Останній блок, експлуатація, пов'язаний зі здатністю системи до масштабування. Архітектура має бути побудована таким чином, щоб система могла ефективно працювати навіть при суттєвому зростанні обсягу медичних записів і кількості користувачів, не втрачаючи при цьому швидкодії та стабільності.

Узагальнене представлення цих вимог у вигляді блок-схеми наведено нижче.



Рис. 2.4 - Графічне представлення нефункціональних вимог системи

Таким чином, нефункціональні характеристики створюють підґрунтя для надійної, зручної у використанні та гнучкої системи, яка може бути ефективно впроваджена в умовах реального університетського середовища.

Системні вимоги

Для забезпечення стабільної та повнофункціональної роботи програмного забезпечення в умовах навчального закладу були визначені системні вимоги, які охоплюють як середовище виконання на стороні користувача, так і інфраструктуру серверної частини, включно з базою даних та аналітичними інструментами. Ці вимоги є критично важливими для коректного розгортання і подальшого супроводження системи, особливо в умовах обробки великого обсягу даних, характерного для масштабного університетського середовища.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				27
Змн.	Арк.	№ докум.	Підпис	Дата		

На рисунку 2.5 наведено діаграму, яка відображає основні технічні параметри, що мають бути дотримані при інсталяції та запуску програмного комплексу. Діаграма умовно поділена на дві основні частини: середовище виконання та вимоги до серверної складової.

Середовище виконання передбачає використання сучасних веббраузерів, таких як Chrome, Firefox або Edge, які підтримують останні версії JavaScript- і HTML-стандартів. Програмне забезпечення має бути сумісним із операційною системою Windows 10 або новішою. Мінімальна конфігурація клієнтського пристрою включає 2 ГБ оперативної пам'яті, процесор із тактовою частотою не менше 1.5 ГГц та щонайменше 500 МБ вільного місця на диску. Такий набір характеристик забезпечує коректне функціонування користувацького інтерфейсу навіть за умов обмежених апаратних ресурсів.

Що стосується серверної частини, вона реалізована з використанням фреймворку NestJS і побудована за принципом REST API для забезпечення зручної інтеграції клієнтської частини з аналітичним модулем і базою даних. У якості СУБД використовується MySQL, яка забезпечує надійне збереження та обробку структурованих даних. Аналітична підсистема, що відповідає за прогнозування епідеміологічних ризиків, реалізована на мові програмування Python з використанням бібліотек машинного навчання scikit-learn та TensorFlow.

Загальна структура вимог наведена нижче у вигляді схемного зображення, що відображає ключові технічні компоненти, необхідні для повноцінного розгортання та функціонування системи.

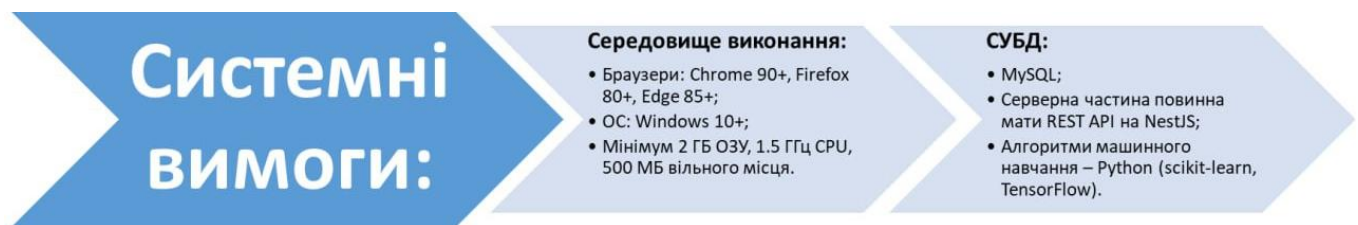


Рис. 2.5 - Візуалізація системних вимог до реалізації проєкту

Таким чином, дотримання наведених технічних умов дозволяє гарантувати коректну роботу всіх підсистем, забезпечити швидкий запуск і налаштування, а

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		28

також сприяти масштабуванню рішення у майбутньому без необхідності у суттєвих модифікаціях.

Варіанти використання

Аналіз вимог користувачів дав підстави для визначення варіантів використання системи моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків. Нижче зображено побудовану діаграму варіантів використання (рис. 2.6).

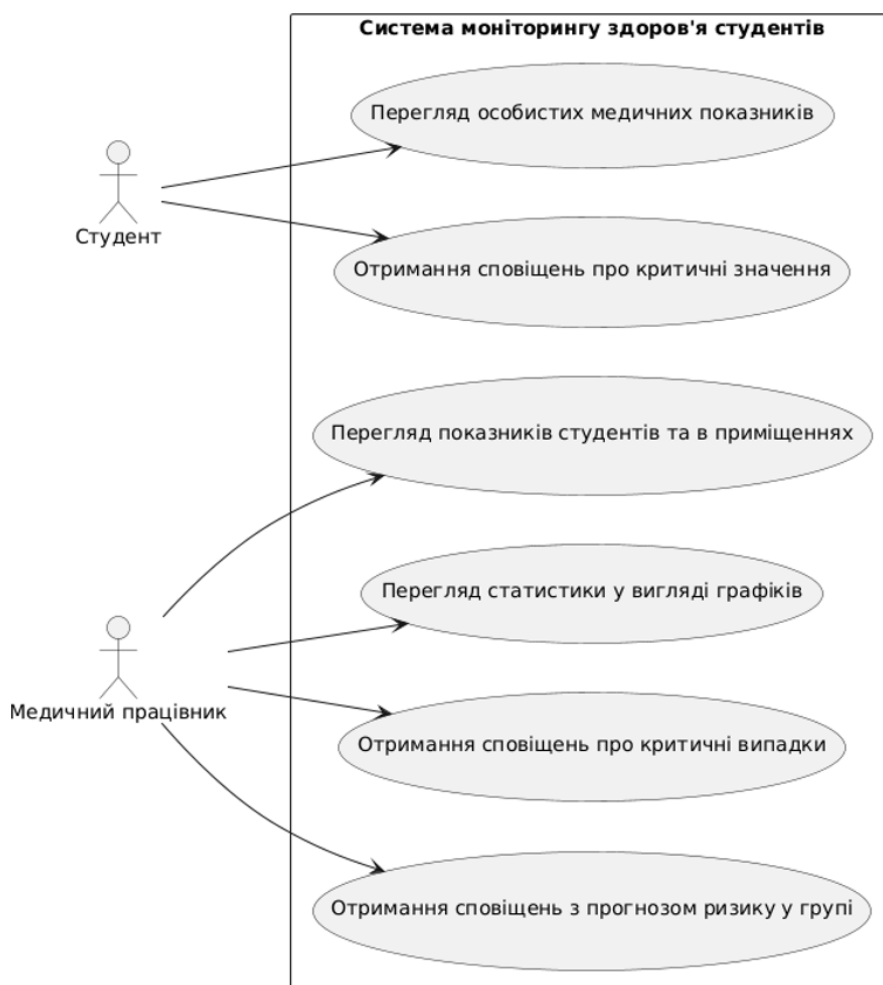


Рис. 2.6 - Діаграма варіантів використання

Опис діаграми варіантів використання наведено в таблицях (табл. 2.1–2.6). Кожна таблиця детально ілюструє конкретний сценарій взаємодії користувачів із системою відповідно до їх ролей. Це дозволяє чітко визначити функціональні можливості, доступні для кожного типу користувача, а також межі доступу до інформації.

Таблиця 2.1

Опис преценденту «Перегляд особистих медичних даних»

Назва варіанту використання	Перегляд особистих медичних даних
Основні актори	Студент
Короткий опис	Студент має змогу переглядати історію власних медичних показників, включаючи дати вимірювань, температурні значення, пульс, сатурацію та інші параметри.

Цей варіант реалізує базову функціональність для студентів, що дозволяє самостійно стежити за змінами свого стану здоров'я у системі.

Таблиця 2.2

Опис преценденту «Отримання сповіщень про критичні значення»

Назва варіанту використання	Отримання сповіщень про критичні значення
Основні актори	Студент
Короткий опис	У випадку фіксації критичних показників (наприклад, підвищеної температури, зниженої сатурації або тривожного пульсу) система автоматично генерує повідомлення, яке надсилається на електронну пошту студента. Повідомлення містить перелік вимірювань, а також короткі рекомендації щодо подальших дій, зокрема звернення до медичного пункту університету.

Цей сценарій забезпечує індивідуальне інформування студентів про потенційно небезпечні стани та стимулює вчасне звернення до медичного пункту.

Таблиця 2.3

Опис преценденту «Перегляд статистики у вигляді графіків»

Назва варіанту використання	Перегляд статистики у вигляді графіків
Основні актори	Медичний працівник
Короткий опис	Медичний працівник має можливість переглядати зведену статистику по студентських групах у форматі графіків для аналізу рівня захворюваності за певні періоди.

Таблиця 2.4

Опис преценденту «Перегляд показників студентів та в приміщеннях»

Назва варіанту використання	Перегляд показників студентів та в приміщеннях
Основні актори	Медичний працівник
Короткий опис	Медичний працівник має доступ до детального перегляду фізіологічних показників студентів (температура, пульс, сатурація) та екологічних параметрів у навчальних приміщеннях (температура повітря, вологість і тд.), де перебували ці студенти.

Цей варіант використання дозволяє здійснювати повноцінний моніторинг як індивідуального стану здоров'я студентів, так і умов довкілля в університетських кабінетах. Наявність такої інформації дає змогу виявляти можливі зовнішні чинники, що впливають на самопочуття студентів, та оперативно вживати відповідних заходів.

Таблиця 2.5

Опис преценденту «Отримання сповіщень з прогнозом ризику у групі»

Назва варіанту використання	Отримання сповіщень про ризику
Основні актори	Медичний працівник
Короткий опис	Система автоматично аналізує накопичені медичні записи за допомогою алгоритмів машинного навчання. У разі виявлення підвищеного ризику захворюваності в конкретній академічній групі, формується повідомлення з вказанням назви групи, рівня ризику та часових рамок. Це повідомлення надсилається медичному працівнику на електронну пошту.

Цей сценарій забезпечує своєчасне реагування на потенційні епідеміологічні загрози та дає можливість планувати превентивні заходи. Завдяки автоматизованому інформуванню працівник може оперативно оцінити ситуацію, провести додаткові обстеження або ініціювати локальні обмежувальні заходи. Такий підхід дозволяє уникнути масових спалахів захворюваності шляхом раннього втручання.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				31
Змн.	Арк.	№ докум.	Підпис	Дата		

Опис прецеденту «Отримання сповіщень про критичні випадки»

Назва варіанту використання	Отримання сповіщень про критичні випадки
Основні актори	Медичний працівник
Короткий опис	Система формує push-сповіщення на сайті при фіксації критичних показників у студента з зазначенням групи та імені.

Цей сценарій забезпечує швидке інформування про індивідуальні загрози для здоров'я, наприклад, підвищену температуру, критичне падіння сатурації або інші симптоми, що можуть свідчити про початок хвороби.

Повідомлення з'являється у верхній частині інтерфейсу незалежно від того, на якій сторінці перебуває користувач, що забезпечує його миттєву видимість. Крім того, система дублює сповіщення на email, щоб гарантувати, що інформація буде отримана навіть у разі неактивного сеансу користувача.

Щоб ефективно реалізувати такі функціональні можливості, необхідно організувати чітку внутрішню структуру системи, що відповідає сценаріям використання.

Реалізація кожного з варіантів використання потребує відповідного розділення логіки в межах серверної частини застосунку. У середовищі NestJS це досягається шляхом модульного підходу: функціональність розбивається на окремі модулі, кожен з яких відповідає за конкретну підсистему — наприклад, медичні показники, моніторинг стану середовища, прогнозування ризиків, надсилання повідомлень. Всередині модулів розміщуються сервіси, контролери й інші допоміжні класи, які разом реалізують повний життєвий цикл обробки даних.

Умовні «класи» системи (тобто логічно відокремлені компоненти) визначено відповідно до їх функціонального призначення. До них належать як моделі медичних або екологічних об'єктів, так і сервіси, відповідальні за аналітику, комунікацію або інтеграцію зі сторонніми системами. Це дозволяє забезпечити розширюваність і підтримуваність програмного коду.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				32
Змн.	Арк.	№ докум.	Підпис	Дата		

Короткий опис основних структурних елементів системи, які відповідають за реалізацію ключових задач, визначених у попередньому підрозділі, наведено нижче (табл. 2.7).

Таблиця 2.7

Опис класів системи

Клас	Опис
Person	Представлений у модулі person. Відповідає за користувача системи. Має ролі: студент, медичний працівник. Містить персональні дані, такі як ім'я, email, група навчання тощо.
PersonRecord	Міститься в модулі person-records. Зберігає медичні показники, отримані від студента: температура, пульс, сатурація, час запису, критичність результату.
Room	Визначений у модулі room. Представляє навчальне або лабораторне приміщення, в якому здійснюється моніторинг екологічних параметрів.
RoomRecord	Описаний у модулі room-records. Містить екологічні дані, зібрані в кабінетах (температура повітря, вологість, CO ₂ , тиск), пов'язані з конкретним Room.
ForecastService	Розташований у модулі forecast. Відповідає за запуск алгоритмів машинного навчання, передачу вхідних даних та обробку результатів прогнозу.
NotificationService	Знаходиться у модулі notification. Генерує та надсилає повідомлення у вигляді email або push-сповіщень відповідним користувачам.
MailService	Реалізовано в модулі mail. Відповідає за відправлення email-повідомлень через поштовий сервер. Використовується NotificationService як залежність.
SocketGateway	Розміщений у модулі socket. Забезпечує двосторонній зв'язок із клієнтом через WebSocket, використовується для передачі push-повідомлень медперсоналу в реальному часі.

2.2. Розробка бази даних системи

Для зберігання даних у системі моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків використовується реляційна система управління базами даних MySQL. Ця СУБД забезпечує високу продуктивність, стабільність роботи при значному навантаженні, підтримку транзакцій, складних SQL-запитів та зв'язків між сутностями, що є критично важливим для реалізації аналітичних функцій системи в реальному часі.

База даних побудована з урахуванням принципів нормалізації, чіткого розподілу відповідальності між таблицями, а також із можливістю масштабування та подальшого розширення. Структура БД охоплює ключові аспекти предметної області: облік користувачів, медичних показників студентів та параметрів навколишнього середовища в навчальних кабінетах.

Ось таблиця, що узагальнює опис структури бази даних системи моніторингу:

Таблиця 2.8

Структура бази даних

Назва таблиці	Призначення
person	Зберігає інформацію про студентів (ПІБ, група, email, роль тощо).
person_record	Містить медичні показники, зафіксовані під час вимірювань (температура, пульс, сатурація).
room	Описує приміщення, в яких здійснюється моніторинг (номер, тип аудиторії).
room_record	Зберігає екологічні параметри приміщень (температура повітря, вологість, CO ₂ , тиск).
user	Таблиця автентифікованих користувачів системи (логін, email, пароль).
role	Містить типи ролей у системі (студент, медичний працівник).

Кожна таблиця спроектована таким чином, щоб забезпечити логічну цілісність даних, можливість фільтрації за студентами, кабінетами, датами та типами критичних значень. Структура бази даних була побудована з урахуванням принципів нормалізації, що дозволяє уникнути дублювання інформації та спрощує обробку запитів з боку серверної частини системи.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				34
Змн.	Арк.	№ докум.	Підпис	Дата		

Окрему увагу приділено підтримці зв'язків між таблицями за допомогою зовнішніх ключів, що забезпечує узгодженість медичних, екологічних та аналітичних даних. Завдяки цьому система здатна швидко виконувати аналітичні запити та формувати вибірки для побудови графіків або прогнозування.

Детальний опис частини структурних таблиць, що використовуються у базі даних проєктованої системи, представлено нижче (табл. 2.9–2.12).

Для реалізації базової структури обліку користувачів системи використовується таблиця person. Вона містить усю необхідну особисту інформацію про студентів, включаючи ПІБ, електронну пошту, належність до навчальної групи, а також роль у системі. Ця таблиця є центральною у відношенні до інших, оскільки через зовнішні ключі з нею пов'язуються як медичні, так і екологічні показники. У таблиці також зберігається інформація, пов'язана з підтвердженням акаунта, що необхідна для контролю доступу (табл. 2.9).

Таблиця 2.9

Опис таблиці person

Поле	Тип	Призначення
id	int	Ідентифікатор
studentID	int	Індивідуальний номер студента
name	varchar(255)	Ім'я
surname	varchar(255)	Прізвище
patronymic	varchar(255)	По батькові
email	varchar(255)	Електронна пошта
studyGroup	varchar(255)	Група навчання
role	varchar(255)	Роль користувача
password	varchar(255)	Пароль
confirmCode	varchar(255)	Код підтвердження
confirmed	tinyint	Підтвердження облікового запису

Медичні дані, зібрані від студентів, зберігаються у таблиці person_record. Кожен запис містить фізіологічні показники, такі як температура тіла, пульс,

сатурація, а також відмітку про критичність та час створення. Крім цього, кожен запис прив'язується до студента через поле `personId`, і до конкретного приміщення через `roomId`, що дозволяє проводити просторово-часовий аналіз медичних подій. Саме з цієї таблиці формується основний масив даних для подальшого аналізу, відображення та прогнозування (табл. 2.10).

Таблиця 2.10

Опис таблиці `person_record`

Поле	Тип	Призначення
<code>id</code>	<code>int</code>	Ідентифікатор
<code>personId</code>	<code>int</code>	Зв'язок зі студентом (<code>person</code>)
<code>roomId</code>	<code>int</code>	ID приміщення, де проводилось вимірювання
<code>temperature</code>	<code>decimal</code>	Температура тіла
<code>saturation</code>	<code>decimal</code>	Насичення крові киснем
<code>heartRate</code>	<code>decimal</code>	Частота серцевих скорочень
<code>isCriticalResult</code>	<code>tinyint</code>	Чи є критичне відхилення
<code>createdAt</code>	<code>datetime(6)</code>	Дата створення

Щоб забезпечити просторову прив'язку екологічних даних, у системі створено таблицю `room`, яка описує усі навчальні приміщення, в яких встановлено сенсори. У таблиці зберігається базова інформація про кімнати: їх номер та тип (наприклад, лекційна зала або лабораторія) (табл. 2.11).

Таблиця 2.11

Опис таблиці `room`

Поле	Тип	Призначення
<code>id</code>	<code>int</code>	Ідентифікатор приміщення
<code>roomNumber</code>	<code>varchar(255)</code>	Номер кімнати
<code>roomType</code>	<code>varchar(255)</code>	Тип кімнати (практична аудиторія, лекційна зала)

Екологічні параметри, зафіксовані сенсорами в навчальних кабінетах, зберігаються у таблиці `room_record`. Кожен запис містить дані про температуру

повітря, вологість, рівень CO₂, атмосферний тиск, а також час фіксації. Записи зв'язуються з таблицею room, що дозволяє визначати, в яких саме приміщеннях фіксувалися ті чи інші умови. Ці дані використовуються як контекст для аналізу потенційного впливу навколишнього середовища на самопочуття студентів (табл. 2.12).

Таблиця 2.12

Опис таблиці room_record

Поле	Тип	Призначення
id	int	Ідентифікатор
roomId	int	Зв'язок із таблицею room
temperature	decimal	Температура повітря
humidity	decimal	Вологість
co2	decimal	Рівень CO ₂
pressure	decimal	Атмосферний тиск
createdAt	datetime(6)	Час фіксації показника

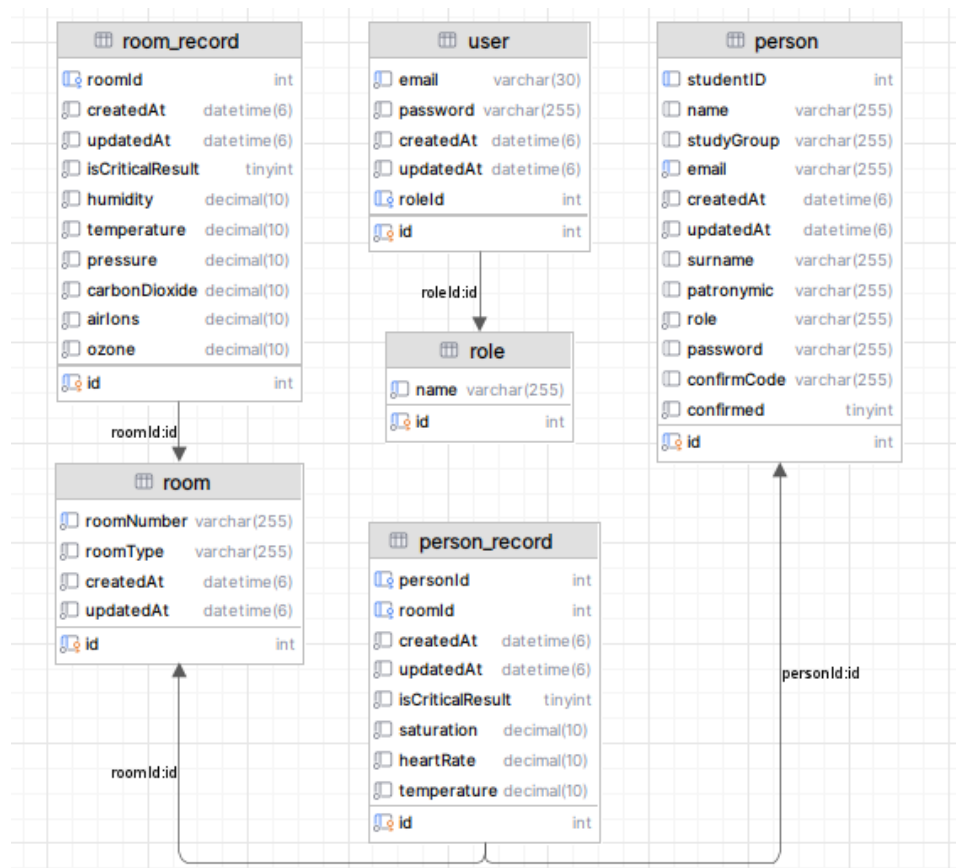


Рис. 2.7 - База даних системи

На рисунку (рис. 2.7) зображено діаграму бази даних системи, яка демонструє основні таблиці, їх поля та зв'язки. Ця діаграма є візуалізацією логічної моделі зберігання інформації, що використовується в роботі системи.

Таке проектування БД забезпечує швидкий доступ до даних, сумісність із ORM-інструментами, простоту масштабування та можливість гнучкої побудови аналітичних звітів без надмірного навантаження на систему.

2.3. Проектування та реалізація алгоритмів роботи системи

Система моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків реалізує набір алгоритмів і механізмів, що забезпечують автоматичне збирання, обробку, аналіз, візуалізацію медичних і екологічних даних, а також формування прогнозів щодо можливих спалахів захворюваності в навчальному середовищі.

Після входу на сайт користувач потрапляє на сторінку авторизації. В залежності від ролі, визначеної у базі даних, користувач отримує доступ до відповідного функціоналу. Для студентів вхід здійснюється за підтвердженою електронною адресою, а для медичних працівників та адміністраторів — за допомогою логіна та пароля.

Інтерфейс системи містить навігаційну панель із посиланнями на основні розділи: профіль, записи, статистика, кімнати, користувачі. У нижній частині панелі розміщено кнопку згортання, яка дозволяє адаптувати інтерфейс для пристроїв із меншою роздільною здатністю.

На сторінці перегляду записів усі критичні значення (наприклад, температура $> 36.8^{\circ}\text{C}$ або сатурація $< 94\%$) автоматично виділяються червоним кольором. Передбачено фільтрацію записів за критичністю ($\text{isCriticalResult} = 1$) або за автором запису — для зручності студента. Реалізація цього механізму відображення наведена у фрагменті коду (лістинг 2.1).

```
export const Element: FC<IPersonRecordsElementProps> = ({data}) => {  
  
    const adjustedDate = moment(data.createdAt).add(3, 'hours')
```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				38
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    return (
      <DivElement color={data.isCriticalResult ? COLORS.red :
COLORS.green2}>
        <Name>{data.person.name} {data.person.surname}
{data.person.patronymic}</Name>
        <CriticalIcon critical={data.isCriticalResult}/>
        <ParamElement label={'Група'}
value={data.person.studyGroup}/>
        <ParamElement label={'Кисень'} value={data.saturation}
units={''}/>
        <ParamElement label={'Частота серцебиття'}
value={data.heartRate} units={'уд./хв.'}/>
        <ParamElement label={'Температура тіла'}
value={Number(data.temperature).toFixed(1)} units={'°C'}/>
        <ParamElement label={'Кімната'}
value={data.room.roomNumber}/>
        <Date>
          {adjustedDate.format('HH:mm DD/MM/YYYY')}
        </Date>
      </DivElement>
    )
  }
}

```

Лістинг 2.1 - Компонент Element, який візуалізує медичні показники студента з урахуванням критичних значень

У розділах «Кімнати» та «Користувачі» представлено список приміщень і студентів відповідно. Для кожного елемента передбачено контекстне меню, за допомогою якого користувач може переглянути всі записи, пов'язані з відповідним об'єктом, а також перейти до індивідуальної статистики. Ця статистика автоматично агрегується для вибраного студента або кабінету та надає зручний доступ до відповідної інформації.

У розділі «Статистика» реалізовано побудову графіків, які відображають динаміку основних параметрів здоров'я та мікроклімату. Зокрема, відстежуються показники температури тіла, пульсу, сатурації крові, рівня вуглекислого газу в

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				39
Змн.	Арк.	№ докум.	Підпис	Дата		

приміщеннях, а також вологості повітря. Такий підхід дозволяє візуалізувати зміни стану здоров'я та умов середовища у зручному для аналізу форматі. Графіки інтерактивні та формуються на основі даних з таблиць `person_record` і `room_record`. Користувач може задати часовий інтервал або конкретну дату через календар, а також вибрати об'єкт для аналізу. Дані агрегуються на сервері (NestJS) та передаються у frontend (ReactJS) у вигляді REST-запитів, після чого відображаються за допомогою бібліотеки Chart.js. Фрагмент реалізації цієї логіки наведено нижче (лістинг 2.2).

```
public async personDashboard(params: any):
Promise<PersonRecord[]> {

    const {day, id, month, year} = params;

    const startDate = new Date(year, month - 1, day);

    const finishDate = addDays(startDate, 1);

    const qb = this.personRecordsRepository

        .createQueryBuilder('record')

        .leftJoinAndSelect('record.room', 'room')

        .leftJoinAndSelect('record.person', 'person')

        .where('record.createdAt BETWEEN :startDate AND :finishDate',
{startDate, finishDate});

    if (id) {

        qb.andWhere('person.id = :id', {id});

    }

    qb.orderBy('record.createdAt', 'ASC');

    return qb.getMany();

}
```

Лістинг 2.2 - Отримання зведених записів студента за вказану дату для формування графіків

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				40
Змн.	Арк.	№ докум.	Підпис	Дата		

Прогнозування епідемічного ризику

В окремому модулі реалізовано алгоритм прогнозування ризику захворюваності для кожної навчальної групи. Прогноз формується на основі:

- частоти фіксації критичних медичних показників;
- темпів зростання негативної динаміки;
- екологічних умов у кабінетах (наприклад, CO₂ або вологість).

Для побудови прогнозу використовується модель машинного навчання, навчена за історичними даними. Вона запускається сервером через інтеграцію з Python-модулем, що використовує бібліотеки scikit-learn і pandas [25]. Цей модуль ініціює підключення до моделі, передає параметри та обробляє результат прогнозу. Приклад реалізації відповідного сервісу подано нижче (лістинг 2.3).

```
export class ForecastService {
  async runForecast(): Promise<any> {
    const scriptPath = resolve(__dirname,
'../../../../../../analytics/predict.py');
    const pythonPath =
'C:\\Users\\Kara\\WebstormProjects\\diploma\\venv\\Scripts\\python.exe';

    const pythonProcess = spawn(pythonPath, [scriptPath]);
    return new Promise((resolve, reject) => {
      let result = '';
      let error = '';
      pythonProcess.stdout.on('data', (data) => {
        result += data.toString(); });
      pythonProcess.stderr.on('data', (data) => {
        error += data.toString(); });
      pythonProcess.on('close', (code) => {
        if (code !== 0) {
          reject(new Error(`Python process exited with
code ${code}: ${error}`));
        } else {
          try {
            resolve(JSON.parse(result));
          }
        }
      });
    });
  }
}
```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				41
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        } catch (e) {
            reject(new Error(`Invalid JSON from
Python: ${result}`)); })
        });
    });
}

```

Лістинг 2.3 - Сервіс запуску алгоритму прогнозування ризику захворюваності

У разі перевищення заздалегідь встановлених порогів ризику система автоматично формує сповіщення. Email-повідомлення надсилаються відповідним адміністраторам та студентам, а на панелі керування з'являється попередження у реальному часі. Таким чином, реалізується реактивна стратегія протидії можливим епідеміологічним загрозам.

Контроль доступу та матриця прав

У системі реалізовано матричну модель доступу на основі ролей. Це дозволяє чітко розмежувати, які дані і функції доступні кожній категорії користувачів. У таблиці 2.13 наведено рівні доступу, де:

- 0 — відсутній доступ;
- 1 — доступ до власних даних;
- 2 — повний доступ до всіх відповідних даних.

Таблиця 2.13

Матриця доступу до функцій системи

Функція	Студент	Мед. працівник
Перегляд особистого профілю	1	0
Перегляд списку студентів	0	2
Перегляд записів усіх студентів	0	2
Перегляд списку навчальних кабінетів	0	2
Перегляд записів по кабінетах	0	2
Перегляд статистики у вигляді графіків	1	2
Повідомлення про критичні показники	1	2
Повідомлення про можливу епідемію захворюваності	0	2

Усі основні сценарії взаємодії користувачів із системою, зокрема вхід до особистого кабінету, перегляд і внесення медичних записів, аналіз даних, надсилання сповіщень — реалізовано у вигляді чітко структурованих алгоритмів. Їхня робота залежить від ролі користувача: студенти мають доступ лише до власних записів, медичні працівники можуть переглядати всі медичні показники.

На рисунку (рис. 2.8) представлено узагальнену діаграму активності, яка демонструє основні гілки логіки системи в залежності від ролі користувача.

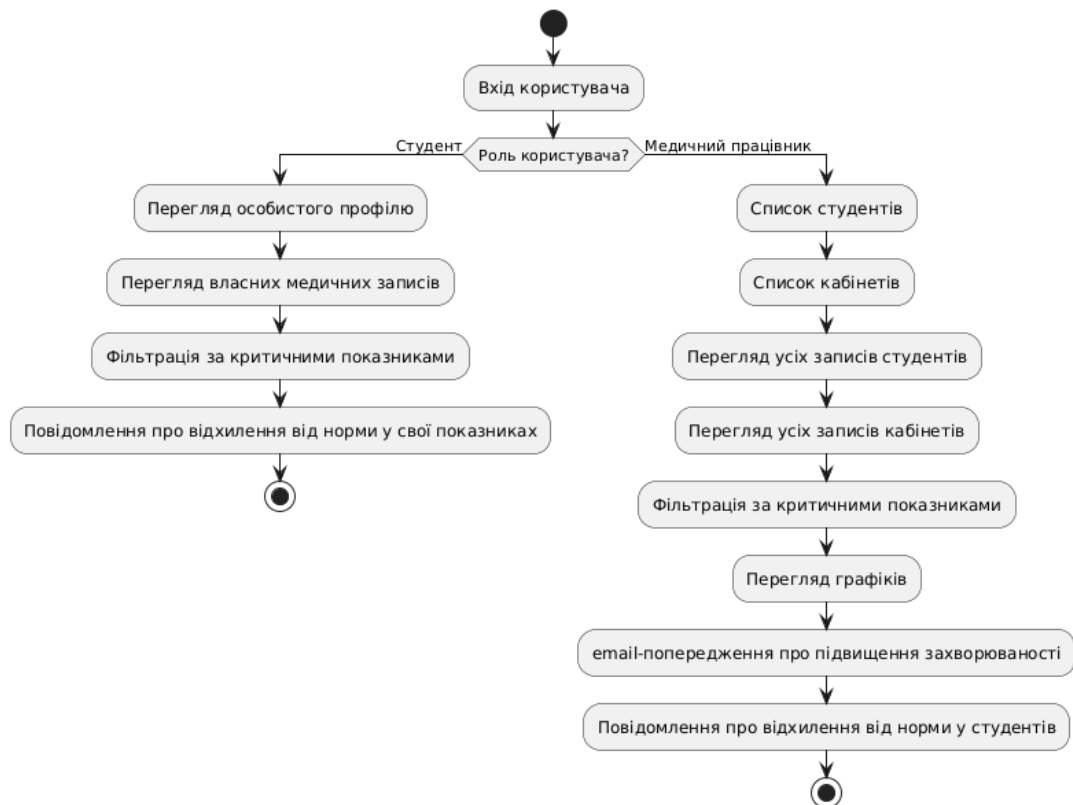


Рис. 2.8 - Діаграма активності

Висновки до другого розділу

У процесі проєктування системи моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків було визначено основні варіанти використання програмного продукту відповідно до ролей користувачів: студент, медичний працівник та адміністратор. На основі цього було спроектовано об'єктно-орієнтовану структуру системи та визначено взаємодію між її ключовими компонентами.

Особливу увагу приділено розробці реляційної бази даних, яка забезпечує збереження медичних показників, даних про користувачів, параметрів приміщень, результатів прогнозування та логів оповіщення. Структура БД побудована з урахуванням вимог до масштабованості, надійності та підтримки аналітичних обчислень.

Також було описано основні алгоритми роботи системи: збирання медичних даних, виявлення критичних значень, побудова статистичних графіків, запуск моделей прогнозування епідеміологічного ризику, а також автоматичне сповіщення користувачів у випадках перевищення допустимих порогів. Узагальнена діаграма активності відображає послідовність дій користувачів та логіку взаємодії між клієнтською і серверною частинами системи.

Таким чином, на основі аналізу вимог, функціональності та можливих сценаріїв використання було закладено архітектурну основу системи, що дозволяє ефективно контролювати рівень захворюваності в навчальному середовищі та своєчасно реагувати на потенційні епідемічні загрози.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

РОЗДІЛ 3. ІТЕРФЕЙС ТА ПОРЯДОК РОБОТИ СИСТЕМИ МОНІТОРИНГУ ТА АНАЛІЗУ РІВНЯ ЗАХВОРЮВАНOSTІ СТУДЕНТІВ З ПРОГНОЗУВАННЯМ ЕПІДЕМІЧНИХ РИЗИКІВ

3.1. Порядок встановлення та налаштування параметрів системи

Розроблена система моніторингу та аналізу рівня захворюваності студентів із прогнозуванням епідемічних ризиків реалізована як веб-застосунок із клієнтською частиною (frontend), серверною частиною (backend), базою даних та аналітичним модулем прогнозування. Встановлення та запуск системи здійснюється локально або на серверному середовищі з підтримкою Node.js, MySQL та Python.

Перед запуском необхідно встановити програмне забезпечення:

- Node.js (версія 18 або новіша);
- MySQL Server (версія 8 або новіша);
- Python 3.10+ з бібліотеками pandas, scikit-learn;
- Менеджер пакетів npm або yarn;
- Git для клонування репозиторію.

Після підготовки середовища користувач має клонувати репозиторій системи з GitHub або іншого сховища коду. У структурі проєкту передбачено кілька основних каталогів: server (серверна частина), web-client (фронтенд), analytics (аналітичний модуль) та допоміжні конфігураційні файли.

Серверна частина реалізована на базі фреймворку NestJS. У її конфігураційному файлі .env зазначаються параметри підключення до бази даних (лістинг 3.1).

```
DB_TYPE=mysql
DB_HOST=127.0.0.1
DB_PORT=3336
DB_DATABASE=healthmonitoring
DB_USERNAME=root
DB_ROOT_PASSWORD=rootpass
DB_SSL=false
```

Лістинг 3.1 - Параметри підключення до бази даних

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				45
Змн.	Арк.	№ докум.	Підпис	Дата		

Після цього встановлюються залежності проєкту за допомогою команди `npm install`, і запускається сам сервер у режимі розробки (`npm run start:dev`). Сервер обробляє REST-запити, здійснює підключення до бази даних, логіку авторизації та взаємодію з аналітичним модулем.

Фронтенд частина розташована в директорії `web-client` і створена з використанням `React.js`. Також встановлюються залежності, і система запускається локально за допомогою `npm run dev`. Користувач отримує доступ до інтерфейсу за адресою, яку відкриває Vite (`http://localhost:5173`).

Окремий компонент системи — аналітичний модуль, розміщений у директорії `analytics`. Він відповідає за підготовку та запуск моделей машинного навчання, які прогнозують ризики захворюваності у навчальних групах. У цьому модулі встановлюються Python-залежності, створюється віртуальне середовище, після чого виконується скрипт навчання моделі. Збережена модель використовується для подальшого прогнозування та формування результатів.

Після налаштування компонентів необхідно ініціалізувати базу даних. Структура бази відповідає описанам у розділі 2.4 таблицям. Вона може бути створена за допомогою ORM-інструментів, передбачених у NestJS. Після створення таблиць і зв'язків база даних заповнюється тестовими даними, щоб перевірити працездатність функціоналу в умовах, наближених до реального використання.

Після запуску системи користувачі мають змогу переходити до веб-інтерфейсу, авторизовуватись у системі відповідно до своєї ролі, переглядати медичні показники, вивчати статистику та отримувати сповіщення.

Таким чином, система моніторингу налаштовується шляхом встановлення необхідного середовища, конфігурації компонентів, ініціалізації бази даних та запуску окремих модулів. Її структура дозволяє як локальне використання, так і розгортання на сервері або у хмарному середовищі. У якості альтернативи ручному налаштуванню середовища може бути використано Docker-контейнери, які дозволяють автоматизувати процес розгортання і забезпечити портативність конфігурацій [26].

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				46
Змн.	Арк.	№ докум.	Підпис	Дата		

3.2. Структура інтерфейсу системи моніторингу та аналізу рівня захворюваності студентів з прогнозуванням епідемічних ризиків

Інтерфейс розробленої системи є інтуїтивно зрозумілим і побудований на принципах адаптивного дизайну, що забезпечує комфортну взаємодію користувачів із системою як на великих екранах, так і на мобільних пристроях. Користувачі з різними ролями мають доступ до різного функціоналу в межах одного інтегрованого веб-застосунку. Усі сторінки мають єдину стилістичну концепцію та логічн

Після переходу на сайт користувач потрапляє на сторінку входу, де здійснюється авторизація. Студенти входять за допомогою email-адреси, яка попередньо підтверджується, а медичні працівники — через стандартну форму з логіном і паролем. Інтерфейс авторизації представлено на рисунку нижче (рис. 3.9).

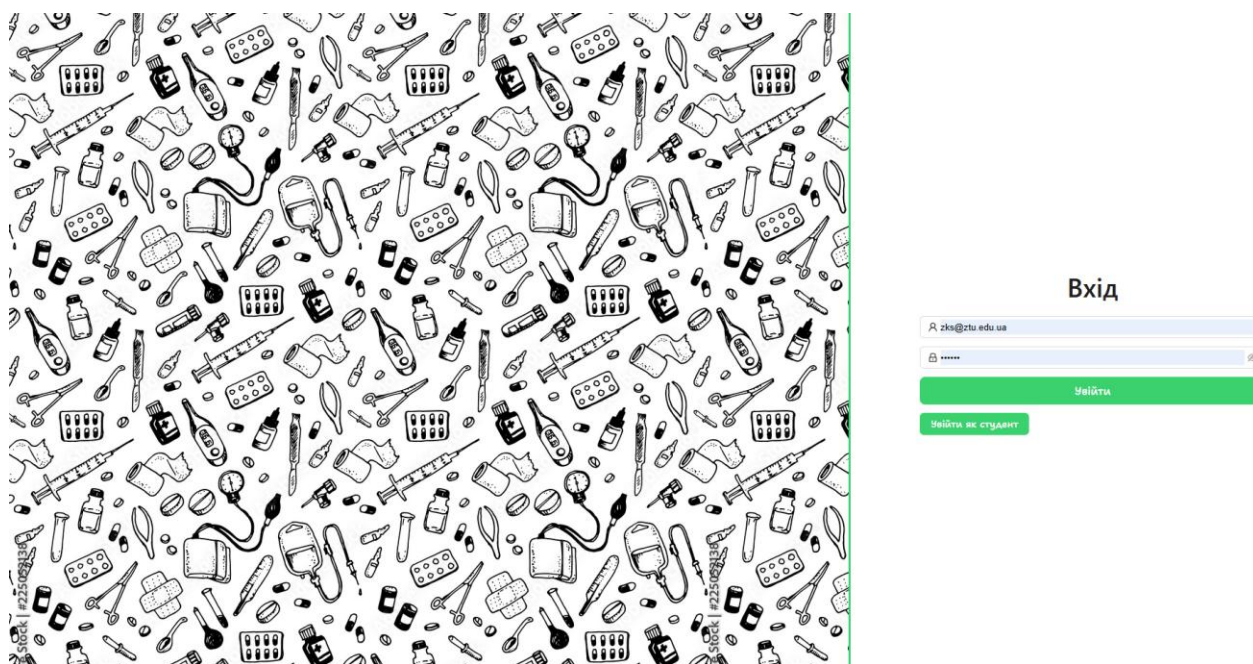


Рис. 3.9 - Сторінка входу

Після авторизації користувач потрапляє на сторінку з записами, і має змогу здійснювати переходи до основних розділів системи. У лівій частині розміщується меню з переходами до основних розділів системи: «Кабінети», «Студенти», «Записи», «Статистика», «Профіль» тощо. Внизу панелі знаходиться кнопка згортання, що дозволяє зменшити навігаційне меню для збільшення робочої

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				47
Змн.	Арк.	№ докум.	Підпис	Дата		

області — це особливо зручно під час перегляду графіків або великих таблиць на пристроях із невеликою роздільною здатністю.

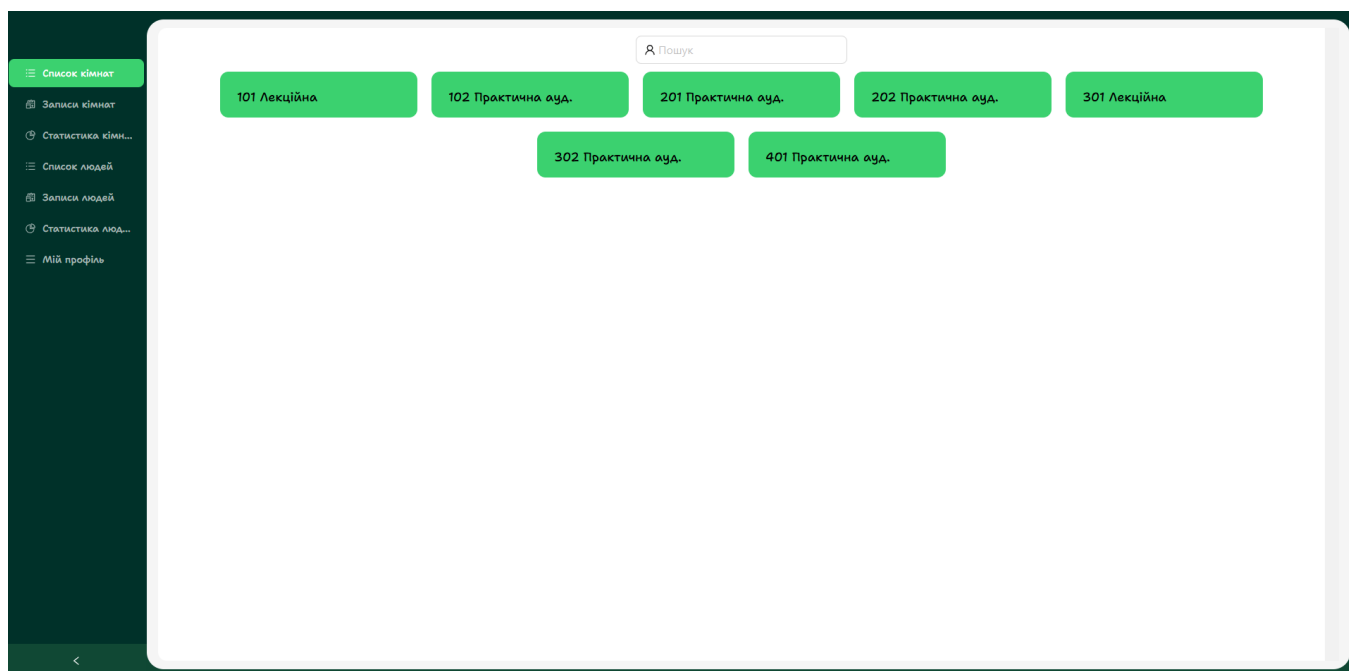


Рис. 3.10 - Сторінка зі списком кабінетів

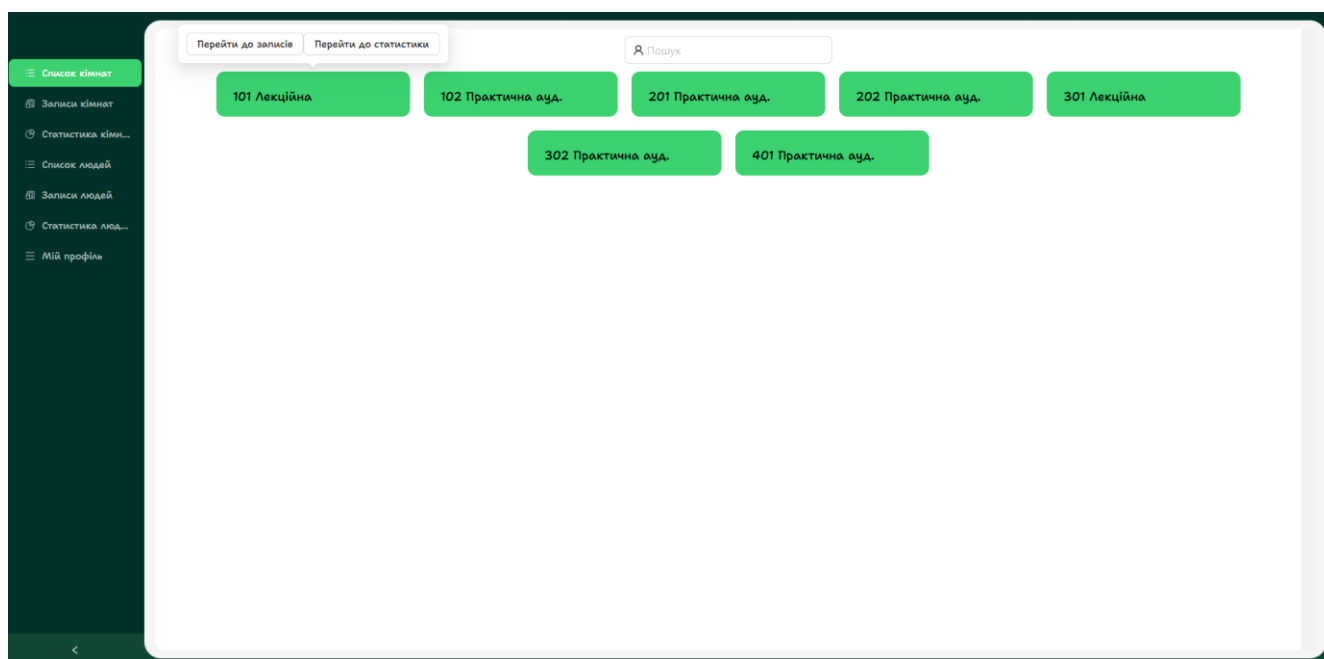


Рис. 3.11 - Можливість переглядати дані за певним кабінетом

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

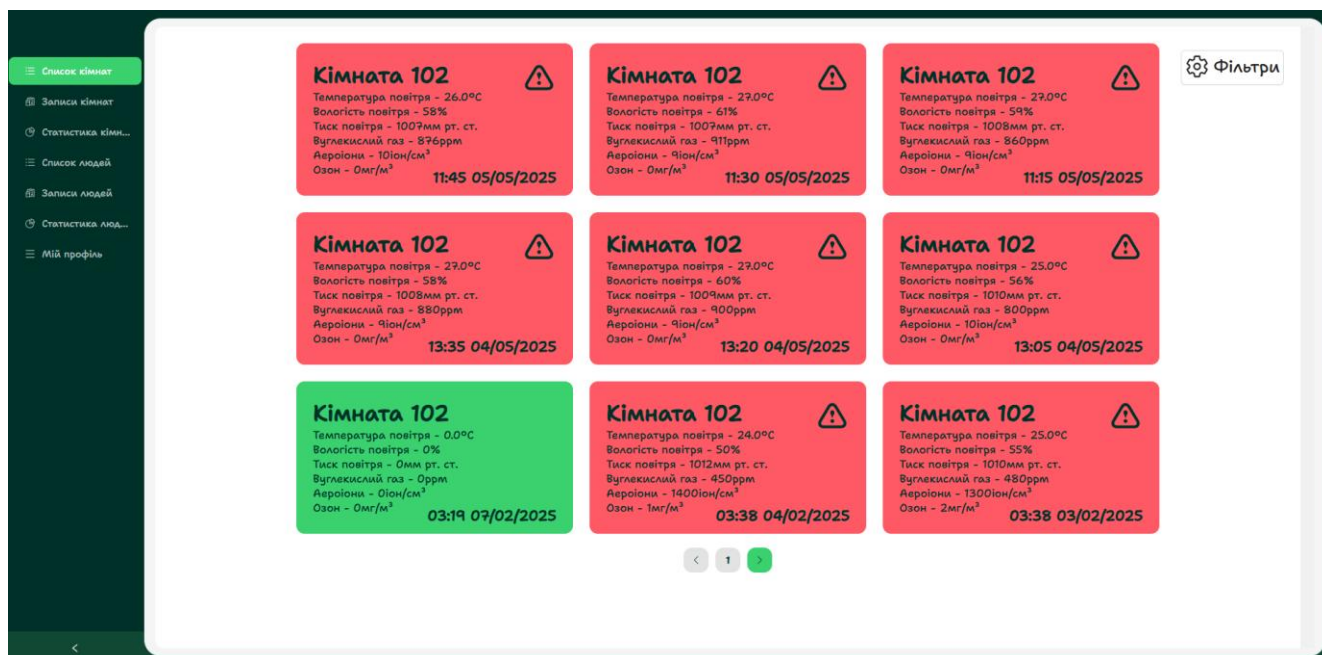


Рис. 3.12 - Сторінка записів по кімнаті



Рис. 3.13 - Сторінка з записами даних кімнат

Розділ «Кімнати» (рис. 3.10) відображає перелік приміщень, де проводиться моніторинг екологічних параметрів. Кожен запис включає номер кабінету та його тип. При натисканні відкривається статистика конкретного приміщення.

Користувач має змогу переглянути дані за конкретним кабінетом, представлені у вигляді карток з вказаними значеннями температури повітря, вологості, CO₂ тощо (рис. 3.11-3.12).

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				49
Змн.	Арк.	№ докум.	Підпис	Дата		

Загальний перелік усіх зібраних даних з кімнат представлено на рисунку нижче (рис. 3.13). Він надає змогу швидко відстежити екологічну ситуацію в університетських приміщеннях.

На вкладці «Студенти» (рис. 3.14) медичний працівник може переглянути перелік усіх зареєстрованих студентів. Кожен запис включає ПІБ та навчальну групу. При натисканні на конкретну особу відкривається сторінка з медичними показниками студента, отриманими в результаті вимірювань.

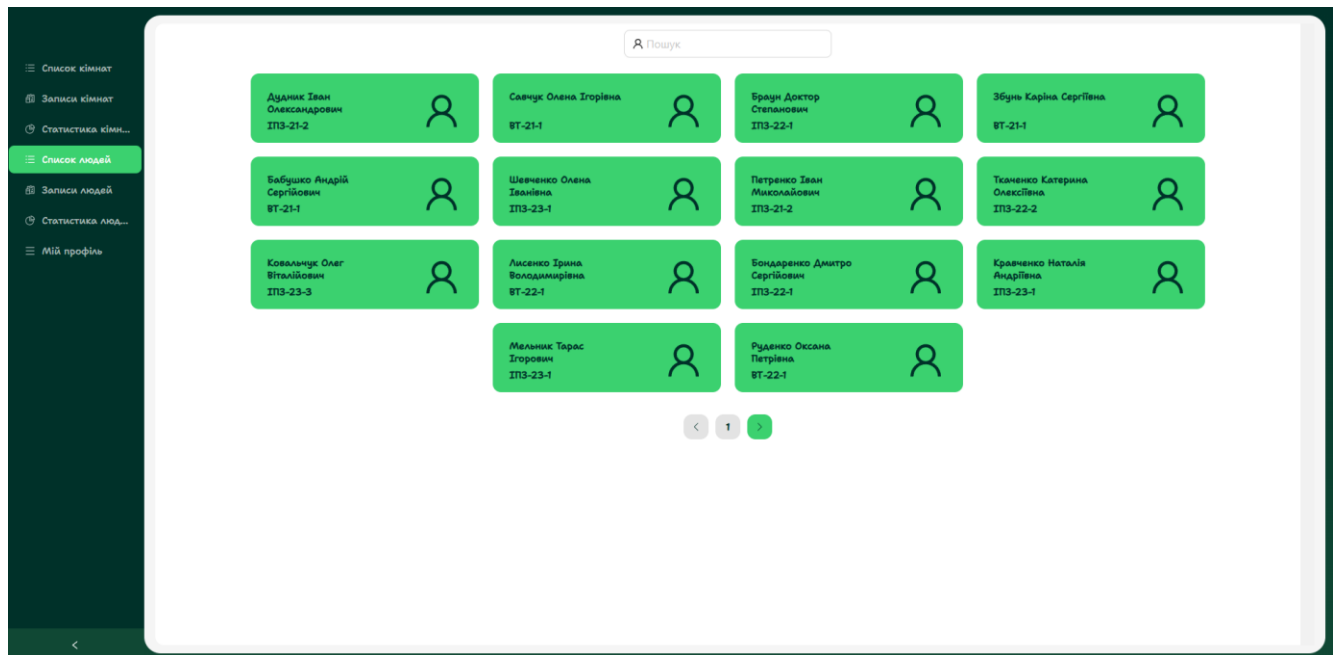


Рис. 3.14 - Перелік студентів

Сторінка записів студентів (рис. 3.15) показує усі вимірювання, що стосуються конкретного користувача. Дані включають температуру тіла, сатурацію, пульс, а також відмітку про критичність. Записи з критичними значеннями виділяються червоним кольором (рис. 3.16), що дає змогу швидко звернути увагу на потенційно небезпечні випадки.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				50
Змн.	Арк.	№ докум.	Підпис	Дата		



Рис. 3.15 - Записи студентів



Рис. 3.16 - Лише записи з критичними показниками

В окремому розділі представлені інтерактивні графіки (рис. 3.17), що візуалізують динаміку зміни медичних показників. Користувач може змінювати часовий діапазон за допомогою календаря, а також переглядати збільшені версії графіків (рис. 3.18) для детальнішого аналізу.

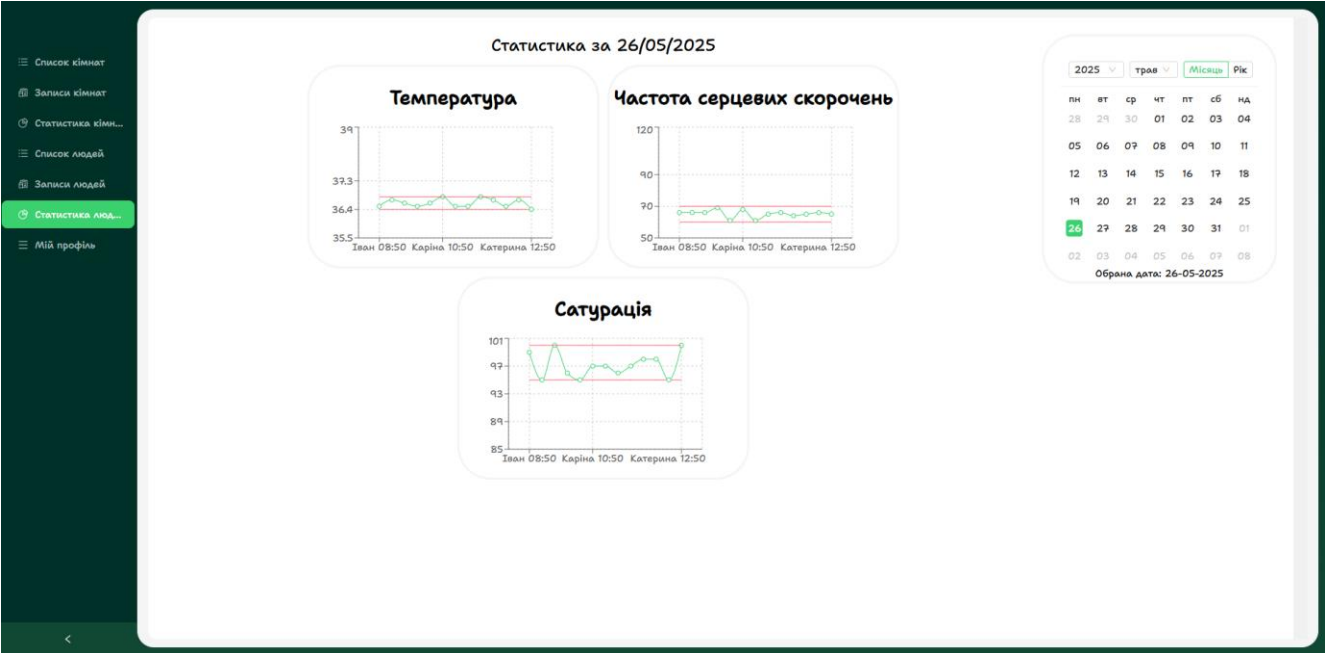


Рис. 3.17 - Графіки показників

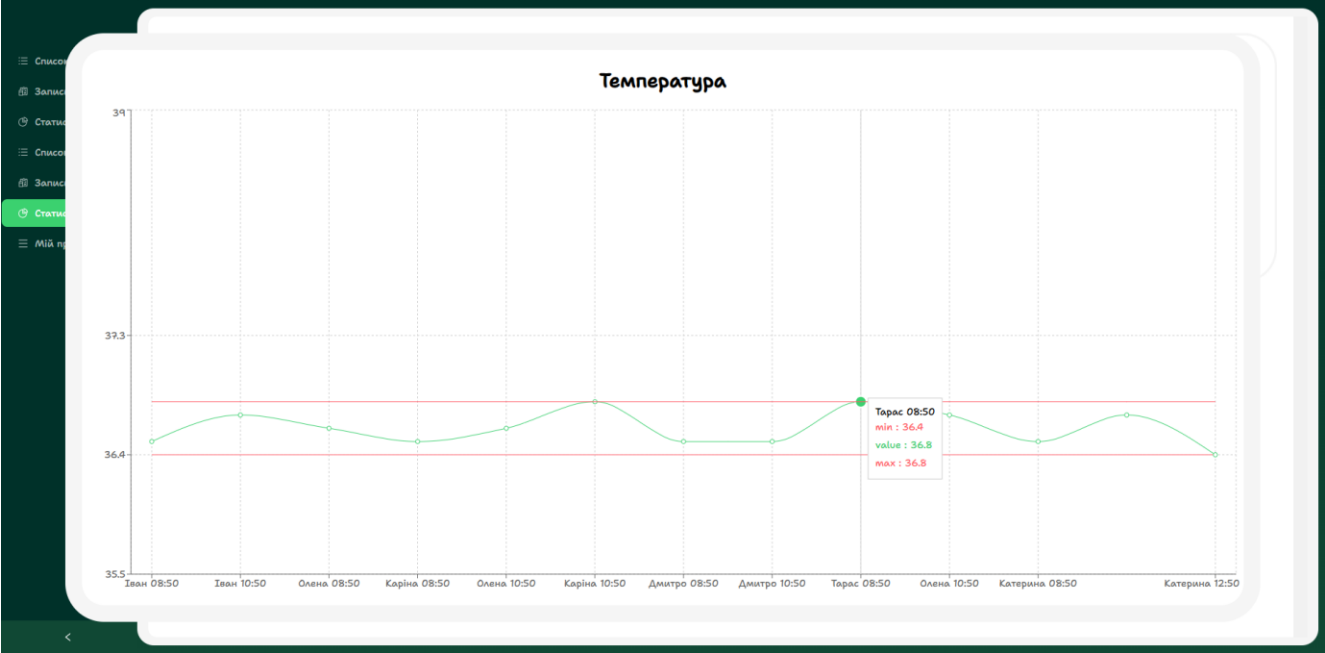


Рис. 3.18 - Збільшений графік

При перегляді даних за конкретною особою (рис. 3.19-3.20) або приміщенням статистичні графіки автоматично адаптуються до відповідної вибірки. Це дає змогу відстежувати стан здоров'я окремої особи або екологічні умови в певній аудиторії.



Рис. 3.19 - Записи по конкретній людині

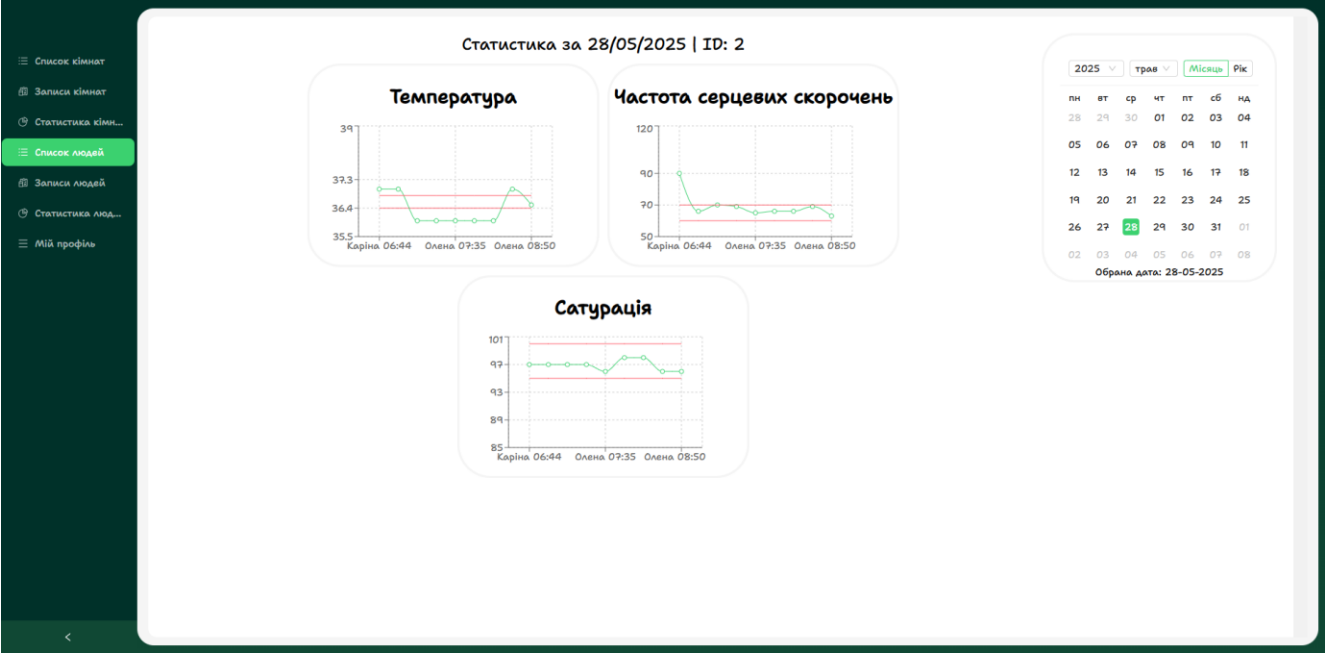


Рис. 3.20 - Графіки показників конкретної людини

Система підтримує адаптивну верстку, тому всі сторінки коректно відображаються на мобільних пристроях. Інтерфейс автоматично підлаштовується під розміри вкладки (рис. 3.21).

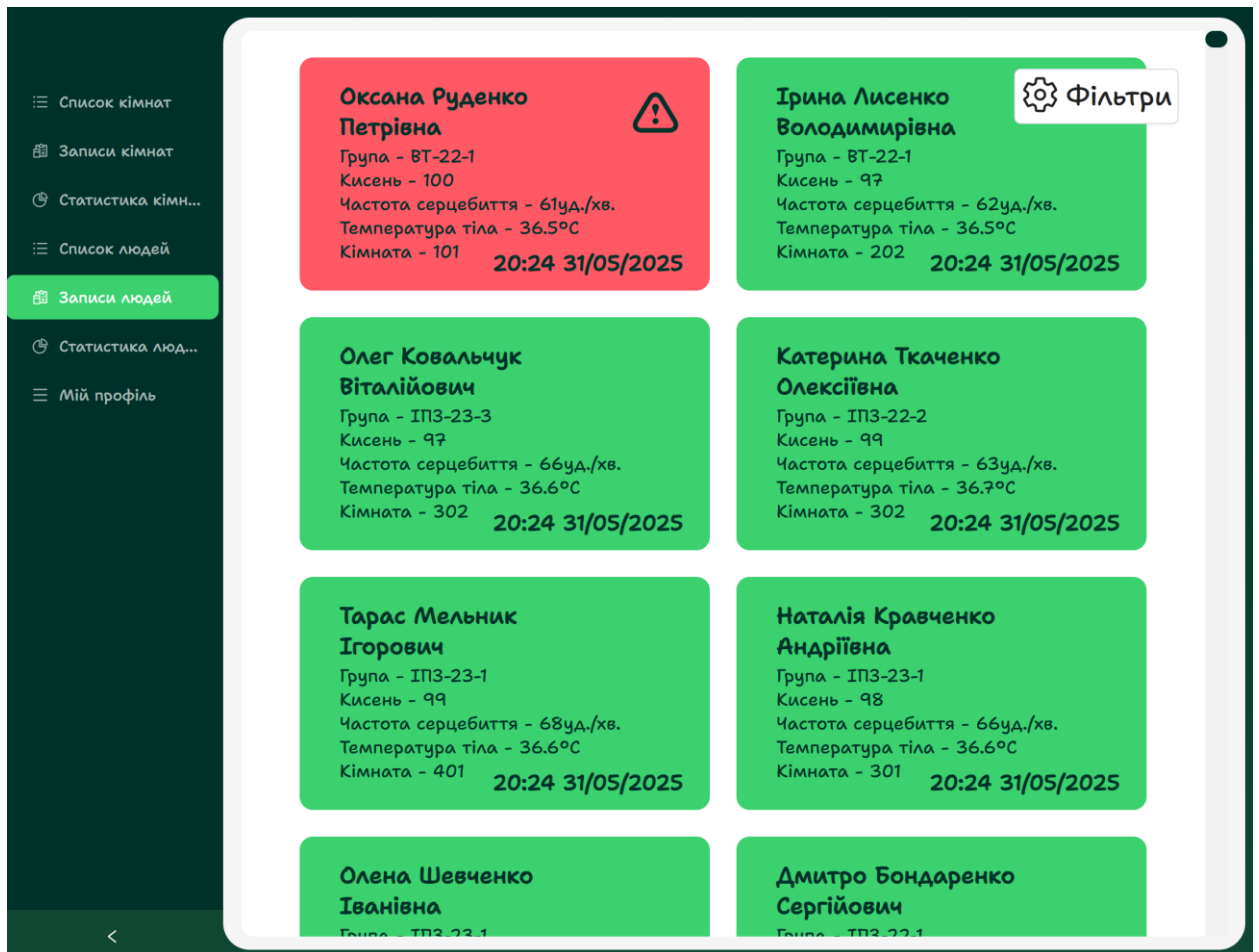


Рис. 3.21 - Сайт адаптується під розміри вкладки



Рис. 3.22 - Повідомлення про захворюваність у групі

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				54
Змн.	Арк.	№ докум.	Підпис	Дата		

Після запуску алгоритму прогнозування медичний працівник отримує push-сповіщення з повідомленням про виявлену підвищену захворюваність у певній академічній групі (рис. 3.22). Таке сповіщення з'являється у верхній частині екрана незалежно від активної сторінки, що дозволяє оперативно зреагувати на загрозу.

Одночасно із push-сповіщенням, система надсилає листа на електронну пошту медичного працівника (рис. 3.23). У листі зазначено назву академічної групи та рівень ризику. Це забезпечує дублювання важливої інформації та можливість її збереження для подальшого аналізу.

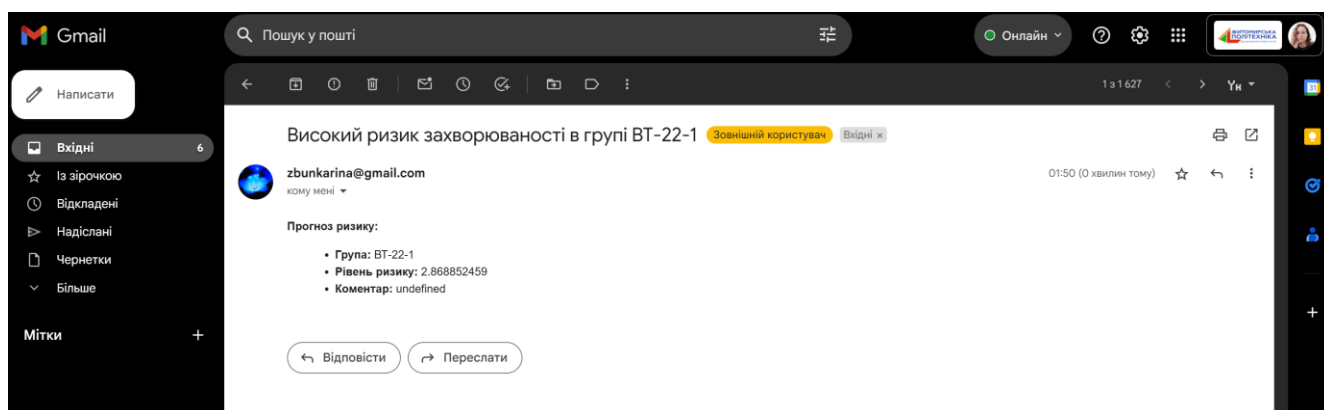


Рис. 3.23 - Лист про захворюваність у групі



Рис. 3.24 - Повідомлення про критичні показники в студента

У разі фіксації критичних медичних показників у конкретного студента (наприклад, висока температура або низька сатурація), на панелі адміністратора

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				55
Змн.	Арк.	№ докум.	Підпис	Дата		

відображається push-сповіщення (рис. 3.24) з іменем студента та його групою. Сповіщення з'являється поверх інтерфейсу незалежно від того, на якій сторінці перебуває користувач, що забезпечує своєчасне реагування.

Коли система фіксує критичні значення, студент автоматично отримує email (рис. 3.25) із переліком його показників, а також рекомендацією звернутися до медичного пункту. Це сприяє своєчасному реагуванню самого студента на початок критичного стану.

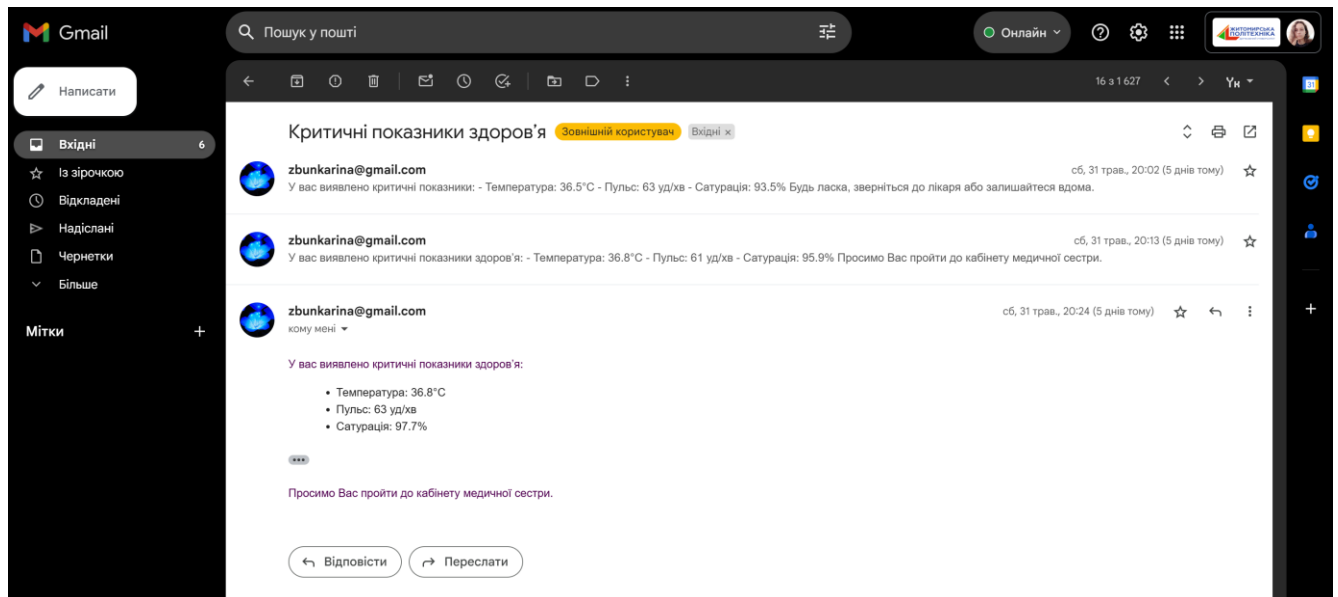


Рис. 3.25 - Лист на пошті хворого студента

Таким чином, структура інтерфейсу охоплює усі ключові функціональні можливості системи — перегляд медичних та екологічних даних, та є зрозумілою для кінцевих користувачів з будь-яким рівнем підготовки. Всі елементи організовані логічно та доступні через просту навігаційну модель.

3.3. Тестування роботи програмного комплексу

Тестування розробленої системи є важливою складовою процесу розробки, що дає змогу виявити та своєчасно усунути помилки ще до етапу впровадження. Основну увагу було зосереджено на ручному тестуванні інтерфейсу користувача, перевірці API-запитів за допомогою Postman, а також тестуванні алгоритмів прогнозування й системи візуалізації.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				56
Змн.	Арк.	№ докум.	Підпис	Дата		

У таблиці нижче (табл. 3.1) подано узагальнення чотирьох ключових тестових сценаріїв, які охоплюють основні модулі системи:

Таблиця 3.1

Результати тестування основних сценаріїв

	1	2	3	4
Назва сценарію	Вхід у систему з підтвердженням обліковим записом студента	Перегляд графіків за студентами	Отримання email-повідомлення при критичному показнику	Прогнозування ризику епідемії
Передумови	Підтверджено email-адресу студента	Існують історичні записи	Створено критичний запис 38.5, перевірити email	Історичні критичні записи присутні
Кроки	Ввести email, натиснути «Увійти»	Перейти в розділ 'Студенти', обрати студента, перейти в 'Статистика'	Зберегти запис із температурою 38.5, перевірити email	Запустити прогноз через Postman
Очікуваний результат	Відображення панелі користувача	Відображення графіків за останній місяць	Отримано лист з попередженням	Система надсилає лист при підвищеному ризику
Фактичний результат	Вхід виконано успішно. Помилка 500 при першій спробі — виправлено	Графіки були порожніми	Повідомлення не надійшло	Прогноз не створено, помилка доступу до model.pkl
Коментар	Відсутнє поле confirmed у БД	Не було групування по днях, додано агрегацію на backend	Функція викликала некоректно, виправлено	Шлях до моделі виправлено — прогноз працює

Під час перевірки аналітичного модуля було засвідчено, що точне налаштування API та правильне групування даних за часом є критичними для достовірного виведення статистики. Це особливо важливо, оскільки графіки відіграють ключову роль у прийнятті рішень медичним персоналом.

Також було підтверджено важливість коректної інтеграції між середовищами Node.js та Python. Аналітична модель прогнозування захворюваності працює на основі історичних даних і дозволяє формувати попередження для кожної академічної групи. Після виправлення шляху до моделі система почала стабільно генерувати прогнози.

Таким чином, проведене тестування підтвердило загальну працездатність системи. Виявлені помилки були типовими технічними недоопрацюваннями на етапі ранньої розробки та були успішно усунені. Основні модулі — візуалізація статистики, генерація повідомлень і прогнозування — функціонують відповідно до поставлених вимог, що дозволяє рекомендувати систему до впровадження в умовах навчального закладу.

Висновки до третього розділу

У третьому розділі було здійснено розгортання та налаштування всіх компонентів розробленої системи моніторингу здоров'я студентів, зокрема клієнтської частини на React.js, серверної частини на NestJS, аналітичного модуля з машинним навчанням на Python, а також бази даних MySQL. Надано детальну інструкцію з інсталяції залежностей, конфігурації середовища та запуску системи.

Далі було розглянуто структуру інтерфейсу веб-застосунку. Для кожної з ролей — студент і медичний працівник — представлено доступний функціонал, інтерфейс авторизації, перегляду записів, статистичних графіків та сторінки взаємодії з екологічними даними приміщень. Продемонстровано адаптивність інтерфейсу до мобільних пристроїв і зручну навігацію між розділами системи.

Особливу увагу приділено тестуванню системи, у ході якого було перевірено коректність роботи інтерфейсу, точність обробки REST-запитів, генерацію push- і email-сповіщень, а також запуск алгоритмів прогнозування. Усі виявлені проблеми

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				58
Змн.	Арк.	№ докум.	Підпис	Дата		

були успішно усунуті, що свідчить про працездатність та надійність розробленого програмного продукту.

У результаті виконаних робіт система підтвердила свою готовність до використання для збору, аналізу, візуалізації медичних показників студентів і своєчасного виявлення епідемічних ризиків.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				59
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У ході виконання кваліфікаційної роботи бакалавра було спроектовано та реалізовано веб-орієнтовану систему моніторингу стану здоров'я студентів, яка дозволяє збирати, обробляти, аналізувати та візуалізувати медичні дані з можливістю автоматичного прогнозування ризиків виникнення епідеміологічних загроз у навчальних закладах.

На основі аналізу предметної області були сформульовані основні функціональні та нефункціональні вимоги до системи, враховуючи ролі кінцевих користувачів: студентів, медичних працівників та адміністрації. У результаті було побудовано модель варіантів використання та спроектовано об'єктно-орієнтовану структуру системи, яка охоплює всі ключові компоненти взаємодії між користувачами та внутрішніми сервісами.

Особливу увагу було приділено проектуванню бази даних, яка забезпечує збереження медичних показників, екологічних параметрів у приміщеннях, результатів аналітики та логів автоматичних сповіщень. Розроблені таблиці дозволяють зберігати історію захворюваності, виявляти критичні відхилення та аналізувати динаміку за різними показниками. Також реалізовано зберігання прогнозних моделей та їх точності, що дає змогу оцінити ефективність алгоритмів машинного навчання.

У роботі описано послідовність роботи з системою для кожної ролі користувача, алгоритми авторизації, внесення медичних даних, фільтрації записів, формування статистики, запуску прогнозування та автоматичного інформування студентів і персоналу у разі підвищених ризиків. Було побудовано відповідні діаграми прецедентів, активності та класів, які відображають загальну архітектуру та логіку функціонування системи.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Вакалюк, Т. А., Спірін, О. М., Нікітчук, Т. М., & Збунь, К. С. (2023). Концептуальна модель здоров'язбережувального інформаційно-цифрового середовища закладу вищої освіти у період карантинних обмежень. Інформаційні технології і засоби навчання, 96(4), 58–71.
<https://doi.org/10.33407/itlt.v96i4.5362>
2. Google Fit Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://www.google.com/fit>
3. NestJS Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.nestjs.com/>
4. React.js Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://react.dev/>
5. Chart.js Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://www.chartjs.org/docs/latest/>
6. Python [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.python.org/3/>
7. TypeScript: JavaScript With Syntax For Types Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://www.typescriptlang.org>
8. MySQL Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://dev.mysql.com/doc/>
9. PostgreSQL [Електронний ресурс] – Режим доступу до ресурсу: <https://www.postgresql.org/>
10. MongoDB [Електронний ресурс] – Режим доступу до ресурсу: <https://www.mongodb.com/>
11. Data Mining for Wearable Sensors in Health Monitoring Systems: A Review of Recent Trends and Challenges [Електронний ресурс] – Режим доступу до ресурсу: <https://www.mdpi.com/1424-8220/13/12/17472?frbrVersion=5>

		Збунь К.С.			Житомирська політехніка. 25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				61
Змн.	Арк.	№ докум.	Підпис	Дата		

12. О.П.Буйницька, "Структурно-функційна модель інформаційно-освітнього середовища університету", ITLT, вип. 69, вип. 1, с. 268–278, Лют 2019.
<https://doi.org/10.33407/itlt.v69i1.2313>.
13. Z. Yu, W. Xu, and L. Yu, "Constructing an Online Sustainable Educational Model in COVID-19 Pandemic Environments," Sustainability, vol. 14, no. 6, p. 3598, Mar. 2022, doi: 10.3390/su14063598.
14. Meredith Carroll, Summer Lindsey, Maria Chaparro & Brent Winslow (2021) "An applied model of learner engagement and strategies for increasing learner engagement in the modern educational environment", Interactive Learning Environments, 29:5, 757-771, doi: 10.1080/10494820.2019.1636083
15. Araka, Eric, et al. "A Conceptual Educational Data Mining Model for Supporting Self-Regulated Learning in Online Learning Environments." In Handbook of Research on Equity in Computer Science in P-16 Education, edited by Jared Keengwe and Yune Tran, 278-292. Hershey, PA: IGI Global, 2021.
<https://doi.org/10.4018/978-1-7998-4739-7.ch016>
16. Shaurya, S. Som and A. Rana, "IoT Based Educational Model for Better Teaching-Learning Environment," 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2020, pp. 824-828, doi: 10.1109/ICRITO48877.2020.9197852.
17. G. Villar-Mayuntupa, P. M. Choque, F. A. Rojas and D. A. Fernández, "E-Learning: Virtual Classroom in Moodle as a collaborative environment for the knowledge of the Institutional Educational Model in teachers of a university in Lima North," 2020 IEEE World Conference on Engineering Education (EDUNINE), Bogota, Colombia, 2020, pp. 1-4, doi: 10.1109/EDUNINE48860.2020.9149545.
18. M. G. Silva da & H. S. Ferreira, (2020). "Model of Educational Reconstruction as a Theoretical and Methodological contribution to the Design of Environments of Science Teaching and Learning". Investigations in Science Education, 25(1), 262–281. <https://doi.org/10.22600/1518-8795.ienci2020v25n1p262>.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				62
Змн.	Арк.	№ докум.	Підпис	Дата		

19. В.О. Попова "Сприятливе здоров'язбережувальне освітнє середовище в умовах нової української школи». Альманах «QN»: зб. наук. пр. студентів III Всеукр. студент. наук.-практ. інтернет-конф. «Студентський науковий вимір проблем природничо-математичної освіти в контексті інтеграції України до єдиного європейського і світового освітнього простору". Глухів; Суми : Винниченко М.Д., 2021. Вип. 11. С. 177-182.
20. Т. В. Лутаєва, І. С. Сабатовська-Фролкіна, В. Ю. Фесенко, "Педагогічні умови створення здоров'язбережувального освітнього середовища", Сучасні тенденції спрямовані на збереження здоров'я людини : зб. наук. пр. присвячено пам'яті професора О. В. Пешкової, м. Харків, 21-22 квіт. 2022 р. Харків, 2022. Вип. 3. С. 146-150.
21. Morghan Hartmann, Umair Sajid Hashmi, Ali Imran. "Edge Computing in Smart Health Care Systems: Review", Challenges and Research Directions. [Електронний ресурс]. Доступно: <https://onlinelibrary.wiley.com/doi/ampdf/10.1002/ett.3710>.
22. С.Г. Дудко, Дидактичні засади формування здоров'язбережувального навчального середовища початкової школи: дис. на здобуття наук. ступ. канд. пед. наук: 19.00.09. К., 2015. 258 с.
23. І. Харченко, І. Шищенко, "Інформаційно-освітнє середовище закладу вищої освіти як підґрунтя для формування інформаційно-цифрової культури майбутніх фахівців". Людинознавчі студії. Серія «Педагогіка». No 13(45), 2021, 78–84, doi: <https://doi.org/10.24919/2413-2039.13/45.11>.
24. Т. М. Nikitchuk, Т. А. Vakaliuk, О. V. Andreiev, О. L. Korenivska, V. V. Osadchyi, М. G. Medvediev "Mathematical model of the base unit of the biotechnical system as a type of edge devices". XIV International Conference on Mathematics, Science and Technology Education. Journal of Physics: Conference Series, 2288 (2022), 012004, IOP Publishing doi:10.1088/1742-6596/2288/1/012004.

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				63
Змн.	Арк.	№ докум.	Підпис	Дата		

25. Pandas Documentation [Електронний ресурс] – Режим доступу до ресурсу:
<https://pandas.pydata.org/docs/>
26. Docker Documentation [Електронний ресурс] – Режим доступу до ресурсу:
<https://docs.docker.com/>

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

ДОДАТКИ

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

Лістинг `servic`y `PersonRecordsService`:

```

import {Injectable} from '@nestjs/common';
import {InjectRepository} from '@nestjs/typeorm';
import {PersonRecord} from '../entities';
import {DeleteResult, Repository, UpdateResult} from 'typeorm';
import {CrudOperations} from '@shared/interfaces';
import {GetAll} from '@shared/interfaces';
import {CreatePersonRecordRequest} from '../requests';
import {every, inRange} from 'lodash';
import {addDays} from 'date-fns';
import {GetRoomRecords} from '../../room-records/requests';
import {FindOptionsWhere} from 'typeorm/find-options/FindOptionsWhere';
import {NotificationService} from '../../notification/services';
import {PersonService} from "../../person/services";

@Injectable()

export class PersonRecordsService implements
CrudOperations<PersonRecord> {

  constructor(
    @InjectRepository(PersonRecord)
    private readonly personRecordsRepository:
Repository<PersonRecord>,

    private readonly personService: PersonService,
    private readonly notificationService: NotificationService,
  ) {}

  public async create(entity: CreatePersonRecordRequest & {
createdAt?: Date, updatedAt?: Date }): Promise<PersonRecord> {

    const {heartRate, personId, roomId, saturation, temperature,
createdAt, updatedAt} = entity;

```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				66
Змн.	Арк.	№ докум.	Підпис	Дата		

```

const isCriticalResult = this.isCriticalResults(entity);

const newRecord = this.personRecordsRepository.create({
  // eslint-disable-next-line @typescript-eslint/ban-ts-comment
  // @ts-expect-error
  person: personId,
  room: roomId,
  saturation,
  heartRate,
  temperature,
  isCriticalResult,
  createdAt,
  updatedAt,
}) as PersonRecord;

if (isCriticalResult) {
  const person = await this.personService.getOne(personId);
  await this.notificationService.notifyCriticalResult(person,
newRecord);
}

return this.personRecordsRepository.save(newRecord);
}

public async getOne(id: number): Promise<PersonRecord> {
  return this.personRecordsRepository.findOneBy({id});
}

public async getByPersonId(id: number):
Promise<GetAll<PersonRecord>> {
  const where: FindOptionsWhere<PersonRecord> = {person: {id}};
  const [personRecords, count] = await
this.personRecordsRepository.findAndCount({
    relations: {
      person: true,

```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				67
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        room: true,
    },
    where,
    order: {
        createdAt: 'DESC',
    },
});
return {
    data: personRecords,
    maxPage: count,
};
}

```

```

public async getAll(params: GetRoomRecords):
Promise<GetAll<PersonRecord>> {
    const where: FindOptionsWhere<PersonRecord> = {};
    if (params.onlyCritical) {
        where.isCriticalResult = true;
    }
    const [personRecords, count] = await
this.personRecordsRepository.findAndCount({
    relations: {
        person: true,
        room: true,
    },
    where,
    order: {
        createdAt: 'DESC',
    },
});
return {
    data: personRecords,

```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				68
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        maxPage: count,
    };
}

public async edit(fields: Partial<PersonRecord>):
Promise<UpdateResult> {
    return this.personRecordsRepository.update({id: fields.id},
fields);
}

public async delete(id: number): Promise<DeleteResult> {
    return this.personRecordsRepository.delete({id});
}

public isCriticalResults(fields: CreatePersonRecordRequest) {
    const {heartRate, saturation, temperature} = fields;

    return !every([
        inRange(saturation, 95, 100),
        inRange(heartRate, 60, 70),
        inRange(temperature, 36.4, 36.8),
    ]);
}

public async personDashboard(params: any): Promise<PersonRecord[]>
{
    const {day, id, month, year} = params;
    const startDate = new Date(year, month - 1, day);
    const finishDate = addDays(startDate, 1);

    const qb = this.personRecordsRepository
        .createQueryBuilder('record')

```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				69
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        .leftJoinAndSelect('record.room', 'room')

        .leftJoinAndSelect('record.person', 'person')

        .where('record.createdAt BETWEEN :startDate AND :finishDate',
{startDate, finishDate});

    if (id) {
        qb.andWhere('person.id = :id', {id});
    }

    qb.orderBy('record.createdAt', 'ASC');

    return qb.getMany();
}
}

```

Лістинг файлу predict.py:

```

import os
import pandas as pd
import joblib
from datetime import datetime
from data_loader import load_data

df = load_data()
df['createdAt'] = pd.to_datetime(df['createdAt'])
df['dayofyear'] = df['createdAt'].dt.dayofyear

features = ['temperature', 'saturation', 'heartRate', 'dayofyear']
df['label'] = ((df['temperature'] > 37.5) | (df['saturation'] < 94) |
(df['heartRate'] > 100)).astype(int)

script_dir = os.path.dirname(os.path.abspath(__file__))
model_path = os.path.join(script_dir, 'model.pkl')

model = joblib.load(model_path)

```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

```
df['predicted'] = model.predict(df[features])

result = df.groupby('studyGroup')['predicted'].mean().reset_index()
result['riskLevel'] = result['predicted'] * 100
print(result[['studyGroup', 'riskLevel']].to_json(orient='records'))
```

Лістинг файлу train.py:

```
import os
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
import joblib
from data_loader import load_data

df = load_data()

df['createdAt'] = pd.to_datetime(df['createdAt'])
df['dayofyear'] = df['createdAt'].dt.dayofyear
df['label'] = ((df['temperature'] > 37.5) | (df['saturation'] < 94) |
(df['heartRate'] > 100)).astype(int)

features = ['temperature', 'saturation', 'heartRate', 'dayofyear']
X = df[features]
y = df['label']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)

script_dir = os.path.dirname(os.path.abspath(__file__))
model_path = os.path.join(script_dir, 'model.pkl')
```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				71
Змн.	Арк.	№ докум.	Підпис	Дата		

```

joblib.dump(clf, model_path)

# joblib.dump(clf, 'model.pkl')
print("☑ Model trained and saved as model.pkl")

```

Лістинг pages/StudentRecords/Data.tsx

```

import {FC} from "react"
import useSWR from "swr";
import {COLORS, REQUESTS, SIZES} from "../../shared/constants";
import {DetailsContainer} from "../../shared/styles";
import styled from "styled-components";
import {ParamElement} from "../../shared/ui";
import moment from "moment/moment";

export const Data: FC = () => {
  const {data} = useSWR(REQUESTS.STUDENT_RECORDS)
  if (!data) return null

  return (
    <DetailsContainer>
      {data.data.map((element: any, index: number) => <Element
key={index} data={element}/>)}
    </DetailsContainer>
  )
}

type ElementProps = {
  data: any
}

const Element: FC<ElementProps> = ({data}) => {
  const adjustedDate = moment(data.createdAt).add(3, 'hours')
  return (
    <ElementBlock danger={data.isCriticalResult}>

```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				72
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        <ParamElement label={'Кімната'} value={data.room.roomNumber}/>
        <ParamElement label={'Частота серцебиття'}
value={data.heartRate} units={'уд./хв.'}/>
        <ParamElement label={'Кисень'} value={data.saturation}
units={' '}/>
        <ParamElement label={'Температура тіла'}
value={Number(data.temperature).toFixed(1)} units={'°C'}/>
        <Date>
            {adjustedDate.format('HH:mm DD/MM/YYYY')}
        </Date>
    </ElementBlock>
)
}

```

```

const Date = styled.div`
    font-weight: bold;
    position: absolute;
    font-size: 1rem;
    right: ${SIZES.md};
    bottom: ${SIZES.sm};
`

```

```

const ElementBlock = styled.div<{ danger: boolean }>`
    position: relative;
    padding: ${SIZES.md};
    padding-bottom: ${SIZES.lg};
    border-radius: ${SIZES.md};
    background: ${props => props.danger ? COLORS.red : COLORS.green2}
`;
`

```

		Збунь К.С.			Житомирська політехніка.25.121.9.000 - ПЗ	Арк.
		Вакалюк Т.А.				73
Змн.	Арк.	№ докум.	Підпис	Дата		