МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»
Кафедра комп'ютерних наук

ОСВІТНЬО ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»
галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма здобуття освіти: очна (денна)

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

на тему: «Проєктування інформаційної системи ведення електронної медичної картки пацієнта»

здобувачки Коваль Анни Юріївни _____

(Підпис)

Науковий керівник: PhD, ст. викладач, Вознюк Ярослав Юрійович _____

(Підпис)

Робота допущена до захисту перед екзаменаційною комісією з атестації здобувачів вищої освіти (ЕК)

Завідувач кафедри: к.е.н., доцент Тішков Б.О.

_____

Київ 2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»
Кафедра комп'ютерних наук

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»
галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

ПОГОДЖЕНО:
Керівник проєктної групи(гарант)
освітньо-професійної програми
_____Оксана ПОМАЗУН
"____" _____2026 р.

ЗАТВЕРДЖУЮ:
Завідувач кафедри
_____Богдан ТІШКОВ
"____" _____2026 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувачу вищої освіти ***Коваль Анні Юріївні***

очної (денної) форми здобуття освіти

на підготовку кваліфікаційної бакалаврської роботи
на тему: «*Проєктування інформаційної системи ведення електронної медичної картки пацієнта»*

Тему затверджено наказом ректора Університету від « *10* » *лютого 2026* р. № *255- ст*.

Кваліфікаційна бакалаврська робота виконується на матеріалах
приватного медичного центру

План кваліфікаційної бакалаврської роботи

Розділ І «Характеристика та аналіз предметної галузі**»**

Розділ ІІ «Розробка вимог і моделювання інформаційної системи»

Розділ ІІІ «Проєктування та реалізація компонентів системи**»**

Об'єктом дослідження виступають процеси оперування медичною документацією, запису пацієнтів і формування звітності в приватному медичному центрі.

Предмет дослідження охоплює методи і засоби побудови веб-орієнтованої системи електронної медичної картки з підтримкою інтеграції з ЕСОЗ.

Мета кваліфікаційної бакалаврської роботи полягає у створенні інформаційної системи електронної медичної картки для приватного медичного центру з технічною готовністю до підключення до ЕСОЗ.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:

У розділі I необхідно проаналізувати предметну область і виявити проблеми існуючого підходу до ведення медичної документації; дослідити організаційну структуру медичного центру та змоделювати основні бізнес-процеси; провести порівняльний аналіз існуючих програмних рішень для автоматизації медичних закладів; обґрунтувати вибір архітектури системи, мови програмування, фреймворку та системи керування базами даних.

У розділі II потрібно сформувати бізнес-вимоги, функціональні та нефункціональні вимоги до системи; розробити постановку задачі та описати алгоритм роботи системи з визначенням вхідної та вихідної інформації; розробити математичне забезпечення системи; побудувати діаграму прецедентів, діаграми послідовності, діаграму класів та діаграму трасування вимог засобами UML.

У розділі III необхідно спроєктувати інформаційне забезпечення системи включно з інфологічною та даталогічною моделями бази даних; спроєктувати технічне забезпечення системи та мережеву інфраструктуру; описати програмне забезпечення та архітектуру системи; реалізувати програмний прототип на стеку Python, Flask і PostgreSQL; провести функціональне тестування системи та оцінити результати реалізації.

Завдання підготував
науковий керівник_____ **Вознюк Ярослав Юрійович**

«11» лютого 2026 р.

Завдання одержав
здобувач _____ **Коваль Анна Юріївна**

«11» лютого 2026 р.

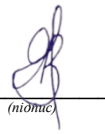
Відгук
про кваліфікаційну бакалаврську роботу
здобувача навчально-наукового інституту
«Інститут інформаційних технологій в економіці»
освітньо-професійної програми
«Комп'ютерні науки»

Коваль Анни Юріївни
на тему
«Проєктування інформаційної системи ведення електронної медичної картки пацієнта»

1. Актуальність теми: Тема кваліфікаційної бакалаврської роботи є актуальною в умовах цифрової трансформації сфери охорони здоров'я України та впровадження електронної системи охорони здоров'я. Автоматизація процесів ведення медичної документації, електронних медичних карток, запису пацієнтів і формування звітності сприяє підвищенню ефективності роботи медичних закладів, якості медичного обслуговування та забезпеченню відповідності сучасним вимогам законодавства.
2. Позитивні риси кваліфікаційної бакалаврської роботи: У роботі проведено ґрунтовний аналіз предметної області та існуючих програмних рішень у сфері медичних інформаційних систем. Авторкою сформовано бізнес-, функціональні та нефункціональні вимоги до системи, побудовано UML-моделі, виконано проєктування інформаційного, технічного та програмного забезпечення. Важливою перевагою роботи є реалізація програмного прототипу на основі сучасного технологічного стеку Python, Flask та PostgreSQL, а також проведення його функціонального тестування.
3. Наявність самостійних розробок автора: Авторкою самостійно виконано моделювання бізнес-процесів медичного центру, розроблено вимоги до інформаційної системи, створено інфологічну та даталогічну моделі бази даних, спроєктовано архітектуру програмного забезпечення та реалізовано програмний прототип інформаційної системи ведення електронної медичної картки пацієнта. Результати роботи підтверджують належний рівень володіння сучасними методами аналізу, проєктування та розроблення інформаційних систем
4. Цінність теоретичних висновків та практичних рекомендацій. Отримані результати мають практичне значення для автоматизації діяльності приватних медичних закладів. Запропоноване рішення дозволяє забезпечити ведення електронних медичних карток, управління розкладом прийомів, формування звітності та створює технічні передумови для інтеграції з ЕСОЗ. Теоретичні напрацювання можуть бути використані під час подальших досліджень і розроблення медичних інформаційних систем.
5. Наявність недоліків: Робота виконана на належному рівні. Як побажання до подальшого розвитку проєкту можна відзначити доцільність більш детального опрацювання механізмів інтеграції з зовнішніми медичними інформаційними

системами та розширення аналітичних можливостей системи. Зазначені зауваження не знижують загальної позитивної оцінки роботи.

6. Загальна оцінка кваліфікаційної бакалаврської роботи та її допущення до захисту перед ЕК. Кваліфікаційна бакалаврська робота Коваль Анни Юріївни виконана відповідно до встановлених вимог, характеризується логічною структурою, достатнім рівнем теоретичного обґрунтування та практичної реалізації. Робота заслуговує на позитивну оцінку, а її авторка — на присвоєння освітнього ступеня «бакалавр» за спеціальністю 122 «Комп'ютерні науки». Робота рекомендується до захисту перед Екзаменаційною комісією.

Науковий керівник 
(підпис)
_____ , PhD, Вознюк Ярослав Юрійович

"11" червня 2026р.

Рецензія
на кваліфікаційну бакалаврську роботу
здобувача вищої освіти

Коваль Анни Юріївни

Тема «Проєктування інформаційної системи ведення електронної медичної картки пацієнта»

1. Актуальність теми кваліфікаційної бакалаврської роботи і доцільність її розроблення

Тема кваліфікаційної бакалаврської роботи є актуальною в умовах цифровізації системи охорони здоров'я України. Впровадження електронних медичних карток у приватних медичних центрах є необхідною умовою інтеграції з державною електронною системою охорони здоров'я та підвищення якості медичного обслуговування. Розроблення доступного і функціонального рішення для невеликих приватних закладів має практичне значення для галузі в цілому і підтверджує доцільність проведеного дослідження.

2. Якість проведеного дослідження

Дослідження виконано на належному науково-технічному рівні. Здобувачка продемонструвала вміння аналізувати предметну область, формувати вимоги до інформаційної системи та застосовувати сучасні методи моделювання. У роботі проведено ґрунтовний порівняльний аналіз існуючих програмних рішень, розроблено комплекс вимог до системи та побудовано UML-моделі. Програмний прототип реалізовано із застосуванням актуального стеку технологій, що підтверджує достатній рівень теоретичної підготовки і практичних навичок здобувачки у сфері інформаційних технологій.

3. Позитивні риси кваліфікаційної бакалаврської роботи

Робота має чітку структуру і логічну послідовність викладу матеріалу. Здобувачка коректно застосувала методи моделювання бізнес-процесів, опису вимог до продукту та проєктування інформаційних систем. Реалізований програмний прототип охоплює основні функціональні модулі системи і може бути використаний як основа для повноцінного впровадження в медичному закладі. Архітектура системи технічно готова до підключення до державної системи охорони здоров'я без переписування системи. Здобувачка виявила ініціативність, уважність до деталей, відповідальне ставлення до виконання завдань та високий рівень до залученості.

4. Зауваження

Суттєвих зауважень до роботи немає. У подальшому доцільно розширити функціональність системи шляхом реалізації модуля виписування електронних рецептів з передачею до ЕСОЗ, особистого кабінету пацієнта, а також доповнення аналітичного модуля фінансовою звітністю.

5. Практична значимість висновків і рекомендацій

Розроблена інформаційна система має практичну цінність і може бути впроваджена в реальному медичному центрі. Вона автоматизує ведення електронних медичних карток, управління розкладом прийомів та формування аналітичної звітності. Отримані результати можуть бути використані при розробці подібних рішень для інших медичних закладів.

Місце роботи та посада рецензента
Науковий ступінь, учене звання (*за наявності*)

ТОВ «КОДЖИТАЙЗ СОЛЮШНЗ»,
Керівник практики від бази практики
Постоялкін В. М.

Підпис засвідчую: Постоялкін В. М.
(ПІБ, пі



РЕФЕРАТ

Кваліфікаційна бакалаврська робота містить 74 сторінки, 11 таблиць, 34 рисунки, список використаних джерел з 30 найменувань, 8 додатків.

«Проєктування інформаційної системи ведення електронної медичної картки пацієнта»

Об’єктом дослідження виступають процеси оперування медичною документацією, запису пацієнтів і формування звітності в приватному медичному центрі.

Предмет дослідження охоплює методи і засоби побудови веб-орієнтованої системи електронної медичної картки з підтримкою інтеграції з ЕСОЗ.

Мета кваліфікаційної роботи полягає у створенні інформаційної системи електронної медичної картки для приватного медичного центру з технічною готовністю до підключення до ЕСОЗ. Щоб досягти цієї мети, в тексті проаналізовано існуючі проблеми і наявні розробки, сформовано вимоги, побудовано моделі системи, спроєктовано її забезпечення та реалізовано і протестовано прототип.

У першому розділі проаналізовано предметну область приватного медичного центру, виявлено проблеми існуючого підходу до ведення документації, проведено порівняльний аналіз п'яти програмних рішень та обґрунтовано вибір технологій розробки. У другому розділі сформовано комплекс вимог до системи, розроблено постановку задачі та математичне забезпечення, побудовано UML-моделі поведінки і структури системи. У третьому розділі спроєктовано інформаційне, технічне та програмне забезпечення, реалізовано програмний прототип на стеку

Python, Flask і PostgreSQL та проведено функціональне тестування за десятьма тест-кейсами.

Практична значущість роботи полягає в тому, що розроблена система може бути впроваджена в реальному медичному закладі. Вона автоматизує ведення електронних медичних карток, управління розкладом прийомів та формування аналітичної звітності, а її архітектура забезпечує технічну готовність до підключення модулів інтеграції з ЕСОЗ і лабораторною інформаційною системою. Теоретична значущість визначається систематизацією підходів до проєктування медичних інформаційних систем з урахуванням вимог українського законодавства. Методична значущість полягає в тому, що розроблений підхід до проєктування системи ЕМК, включно з комплексом вимог, UML-моделями та архітектурними рішеннями, може бути використаний як методична основа при створенні подібних систем для інших медичних закладів.

Рік виконання кваліфікаційної бакалаврської роботи - 2026.

Рік захисту роботи - 2026.

Ключові слова: електронна медична картка, інформаційна система, медичний центр, ЕСОЗ, Flask, PostgreSQL, UML-моделювання.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ 4
ВСТУП 5
РОЗДІЛ 1. ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ 7
1.1 Характеристика предметної області та об’єкта дослідження 7
1.1.1 Організаційна структура об’єкта дослідження 8
1.1.2 Моделювання основних бізнес-процесів 10
1.1.3 Характеристика існуючої інформаційної системи 13
1.1.4 Обґрунтування необхідності вдосконалення системи 14
1.2 Аналіз літературних джерел та практичного досвіду використання інформаційних систем і технологій в предметній області 15
1.3 Обґрунтування вибору методів та технологій для створення інформаційної системи 18
РОЗДІЛ 2. РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ 20
2.1 Аналіз і специфікація вимог до інформаційної системи 20
2.1.1 Бізнес-вимоги 20
2.1.2 Функціональні вимоги 21
2.1.3 Нефункціональні вимоги 23
2.1.4 Специфікація вимог 24
2.2 Постановка задачі та опис алгоритму роботи системи 25
2.2.1 Постановка задачі 25
2.2.2 Алгоритм роботи системи 29
2.2.3 Математичне забезпечення системи 31
2.3 Моделювання інформаційної системи 32
2.3.1 Моделювання поведінки системи 32
2.3.2 Моделювання структури системи 35
2.3.3 Розподіл вимог за компонентами системи 37
РОЗДІЛ 3. ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИТЕМИ 41
3.1 Інформаційне забезпечення 41
3.1.1 Загальна характеристика інформаційного забезпечення 41

3.1.2 Побудова системи класифікації та кодування 42
3.1.3 Джерела вхідних даних та форми їх подання 44
3.1.4 Інфологічна модель бази даних 45
3.1.5 Даталогічна модель бази даних 46
3.2 Технічне забезпечення 49
3.2.1 Загальні положення та схема автоматизації 49
3.2.2 Структура комплексу технічних засобів 50
3.2.3 Опис робочого місця користувача 52
3.2.4 Схема мережі передачі даних 53
3.3 Програмне забезпечення 56
3.3.1 Засоби та технології розроблення 56
3.3.2 Архітектура та структура програмного забезпечення 58
3.3.3 Структура функціональних модулів системи 62
3.4 Результати реалізації інформаційної системи 66
3.4.1 Реалізація функціональних модулів системи 66
3.4.2 Тестування та оцінювання результатів роботи системи 70
ВИСНОВКИ 73
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 75
ДОДАТКИ 79

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

АРМ – Автоматизоване робоче місце

API – (Application Programming Interface) програмний інтерфейс застосунку

BPMN – (Business Process Model and Notation) нотація моделювання бізнес-процесів

БД – База даних

ЕМК – Електронна медична картка

ЕСОЗ – Електронна система охорони здоров’я

ІС – Інформаційна система

МКХ-10 – Міжнародна класифікація хвороб десятого перегляду

ORM – (Object-Relation Mapping) об’єктно-реляційне відображення

REST – (Representational State Transfer) архітектурний стиль передачі даних

СУБД – Система керування базами даних

TLS – (Transport Layer Security) протокол захисту транспортного рівня

UML – (Universally Unique Identifier) уніфікована мова моделювання

ВСТУП

Кожен день у приватній клініці реєструють нових пацієнтів, проводять прийоми, виписують рецепти і відправляють людей на аналізи. Якщо вся ця діяльність частково ведеться на папері, а частково в застарілій програмі без зв’язку з державними системами, персонал витрачає час на дублювання записів замість того, щоб займатися відвідувачами. Держава при цьому вимагає від лікувальних установ підключення до електронної системи охорони здоров’я - без цього неможливо виписати електронний рецепт чи взяти участь у програмах відшкодування вартості ліків. Саме така ситуація спостерігалася у закладі, що став об’єктом цього дослідження.

Науковці по всьому світу займаються питаннями автоматизації медичних закладів. Одні вивчають технічні стандарти передачі клінічних даних, інші аналізують досвід впровадження електронних карток у різних країнах. В Україні більшість наукових праць зосереджена на роботі з ЕСОЗ і наслідках медичної реформи. Але варіантів, орієнтованих саме на невеликі приватні заклади з обмеженим бюджетом у літературі практично немає.

Мета кваліфікаційної роботи полягає у створенні інформаційної системи електронної медичної картки для приватного медичного центру з технічною готовністю до підключення до ЕСОЗ. Щоб досягти цієї мети, в тексті проаналізовано існуючі проблеми і наявні розробки, сформовано вимоги, побудовано моделі системи, спроєктовано її забезпечення та реалізовано і протестовано прототип.

Об’єктом дослідження виступають процеси оперування медичною документацією, запису пацієнтів і формування звітності в приватному медичному центрі.

Предмет аналізу охоплює методи і засоби побудови веб-орієнтованої системи електронної медичної картки з підтримкою інтеграції з ЕСОЗ.

У проєкті застосовувались структурний і порівняльний аналіз, моделювання засобами UML, реляційне моделювання бази даних та прототипування інтерфейсу. Розроблена система вирішує практичні задачі конкретного закладу і може бути перенесена на інші подібні організації. Отримані результати систематизують підходи до проєктування медичних інформаційних систем в умовах українського законодавства.

Джерельну базу склали нормативно-правові акти у сфері електронної охорони здоров’я, публікації вітчизняних і зарубіжних дослідників та документація програмних засобів. Робота включає вступ, три розділи, висновки, список джерел і додатки. Перший розділ присвячений аналізу предметної області та обґрунтуванню технологій, другий охоплює вимоги і UML-моделі, третій містить опис забезпечення системи та результати реалізації.

РОЗДІЛ 1
ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Характеристика предметної області та об’єкта дослідження

У медичній практиці ефективність надання допомоги пацієнтам безпосередньо пов’язана з тим, наскільки оперативно обробляються інформаційні потоки. Протягом робочої зміни лікар генерує велику кількість документації, до якої належать оглядові протоколи, результати діагностики, лікувальні призначення, рецептурні бланки та направлення. Якщо це все накопичується лише на паперових носіях, діяльність установи стикається з критичними обмеженнями. Пошук потрібної картки в архіві забирає багато часу, існує постійний ризик фізичного псування документів, а кілька спеціалістів одночасно взагалі позбавлені можливості вивчати історію хвороби. До того ж передача паперових бланків між відділеннями викликає значні затримки.

Вітчизняний ринок приватних медичних послуг наразі перебуває в стані дуже високої конкуренції. Пацієнт під час вибору клініки оцінює не тільки кваліфікацію лікарів. Суттєвим чинником стає сервіс, зокрема наявність опції онлайн-запису, швидкість надання результатів лабораторних тестів та зручний особистий кабінет в мобільному додатку чи на сайті. Для забезпечення такого функціоналу сучасному медичному центру вкрай необхідна інтегрована інформаційна система, яка б дозволила об’єднати роботу всіх його відділень задля синхронізації щоденних внутрішніх операцій.

Окремим вагомим стимулом для модернізації виступають чинні норми. Зокрема, Постанова Кабінету Міністрів України №411 від 25 квітня 2018 року, прямо зобов’язує заклади реалізувати підключення до ЕСОЗ [1]. Без цього кроку технічно неможливо вести цифрові картки, генерувати е-рецепти або перевіряти чинні декларації пацієнтів із лікарями первинної ланки [2].

Базовим елементом для розгортання ІТ-інфраструктури клініки є електронна медична картка [3]. Саме цей інструмент забезпечує надійне збереження всієї клінічної інформації в цифровому вигляді. Завдяки ЕМК персонал має миттєвий доступ до анамнезу пацієнта, підрозділи оперативно обмінюються результатами досліджень, а сама установа безперешкодно взаємодіє з базою даних ЕСОЗ.

Як об’єкт у цьому дослідженні розглядається приватна багатопрофільна медична клініка, що надає послуги на комерційній основі та співпрацює із страховими компаніями. Проведений аналіз показав, що на початковому етапі робота закладу будувалася за застарілими принципами: документація залишалася паперовою, автоматизованого обміну даними між відділеннями не існувало, а інтеграція з ЕСОЗ була повністю відсутня.

Для оцінки ефективності функціонування медичного центру доцільно використовувати систему таких показників [4]:

1. загальна кількість прийнятих пацієнтів протягом місяця;
2. сумарний обсяг виконаних діагностичних процедур;
3. середній час очікування пацієнтом початку прийому;
4. рівень задоволеності відвідувачів (за результатами опитувань);
5. відсоток повторних звернень клієнтів до клініки;
6. коефіцієнт завантаженості робочого часу медичного персоналу.

Предметом автоматизації у цьому проєкті визначено процеси ведення електронних медичних карток, формування графіків роботи лікарів і реєстрації візитів, генерацію е-рецептів, швидкої передачі результатів аналізів із лабораторії, а також створення модулів аналітичної звітності для керівництва закладу.

1.1.1 Організаційна структура об’єкта дослідження

Організаційна структура медичного центру побудована на лінійно-функціональних принципах, що передбачає чітке закріплення за кожним

підрозділом конкретного кола повноважень та меж його відповідальності [5]. Графічне зображення схеми управління закладом представлено нижче на рисунку 1.1.

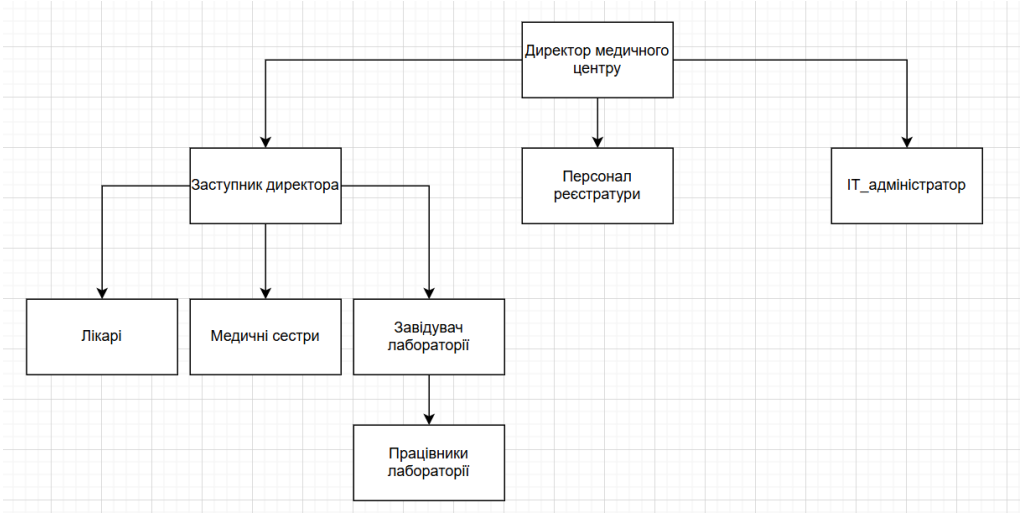


Рисунок 1.1 – Схема організаційної структури медичного закладу

Джерело власна розробка

Загальне керівництво адміністративною діяльністю здійснює директор установи. До його безпосередніх задач належить визначення векторів щодо автоматизації процесів та інтеграції сучасних програмних рішень у повсякденну практику. Головними користувачами ЕМК є лікарі, оскільки вони займаються оглядом хворих, встановленням діагнозів, розробкою плану лікування та виписуванням електронних рецептів, тому мають найширші права доступу в системі. Допоміжну функцію під час консультування пацієнтів покладено на медичних сестер, які забезпечують фіксацію первинних життєвих параметрів хворого. Робота цього персоналу в базі ЕМК обмежена: інтерфейс дозволяє їм оперативно додавати результати поточних замірів і переглядати зроблені лікарем призначення, проте технічно блокує будь-яку можливість змінювати чи редагувати встановлені діагнози. Персонал реєстратури здійснює перший контакт між відвідувачем та медичною установою. Співробітники цього відділу реєструють нових пацієнтів, формують електронні картки та ведуть запис на прийом. Від якості

їхньої роботи з системою значною мірою залежить точність даних у всій базі. Персонал лабораторії проводить діагностичні дослідження і передає їх результати до системи. Завдяки цьому лікар одразу отримує дані в картці пацієнта без зайвих паперових носіїв і без ризику, що результати загубляться або запізняться. IT-адміністратор забезпечує безперебійну роботу всієї технічної інфраструктури системи, керує обліковими записами користувачів, налаштовує права доступу та організовує резервне копіювання даних.

1.1.2 Моделювання основних бізнес-процесів

Для того, щоб виявити проблемні ділянки і сформувати вимоги до нової системи змодельовано три основні бізнес-процеси медичного центру у нотації BPMN [6].

Бізнес-процес 1. Реєстрація пацієнта та створення електронної медичної картки.

Процес починається із звернення пацієнта до реєстратури та пошуку його картки в базі. Якщо відвідувач звертається вперше, то персонал реєстратури заповнює електронну форму та створює картку. Оскільки система зберігає лише мінімальний набір даних, паралельно формується паперова папка для детальних нотаток лікарів. Після оформлення документів пацієнта записують на прийом, однак інтерфейс розкладу не відображає реального завантаження лікарів. Діаграму цього бізнес-процесу наведено на рисунку 1.2.

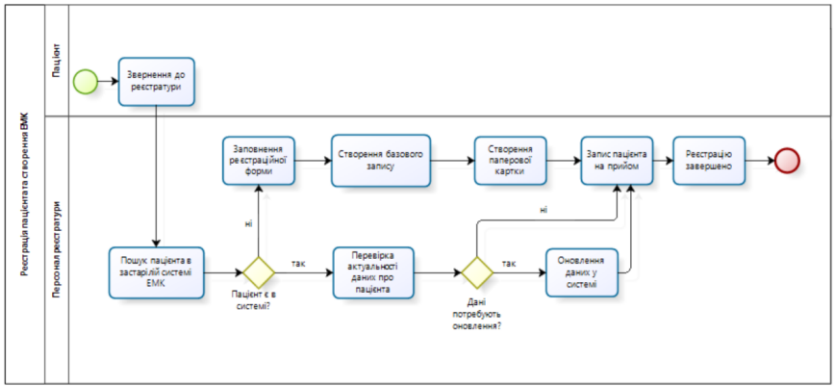


Рисунок 1.2 – Реєстрація пацієнта та створення електронної медичної картки

Джерело: власна розробка у Bizagi Process Modeler [7]

Виявлено наступні проблемні ділянки: дублювання даних між електронними і паперовими форматами, відсутність автоматичної валідації введених даних та виявлення дублікатів записів, а також відсутність інтеграції з ЕСОЗ.

Бізнес-процес 2. Проведення прийому лікарем та ведення медичної документації.

Медична сестра викликає пацієнта та вимірює показники стану здоров’я, фіксуючи їх лише на папері. Лікар шукає картку в системі, але бачить лише стислі нотатку без доступу до повної медичної історії, що змушує повторно збирати анамнез. Огляд і встановлення діагнозу фіксується в системі у скороченому вигляді, а направлення на аналізи та рецепти виписуються на паперових бланках через відсутність зв’язку з лабораторією та ЕСОЗ. На рисунку 1.3 зображено діаграму бізнес-процесу.

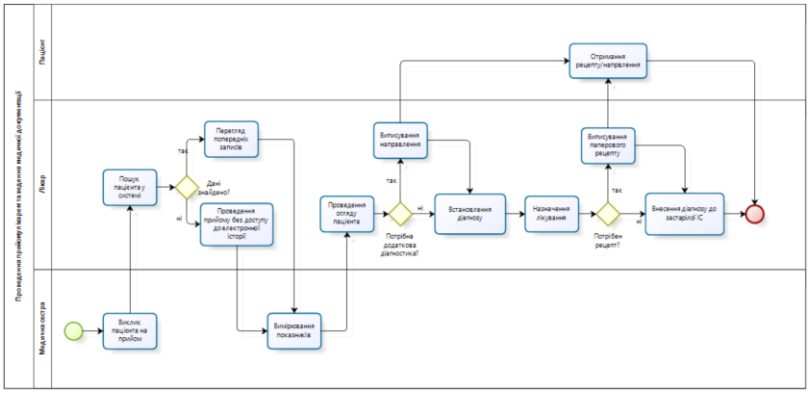


Рисунок 1.3 – Проведення прийому лікарем та ведення медичної документації

Джерело: власна розробка у Bizagi Process Modeler [7]

Проблемними ділянками є: відсутність повної електронної історії хвороби, паралельне ведення документації у двох форматах, паперові направлення і рецепти.

Бізнес-процес 3. Призначення та отримання результатів лабораторних досліджень.

Лікар виписує паперове направлення і передає його пацієнту. Лаборант вручну вносить дані пацієнта в систему, що несе ризик помилок при перенесенні. Після виконання аналізів результати не передаються автоматично до ЕМК, а лікар не отримує сповіщення про їх готовність. Пацієнт приносить паперові результати на наступний прийом, через що між проведенням дослідженням і корекцією лікування виникає суттєва затримка. Графічне зображення цього бізнес-процесу наведено на рисунку 1.4.

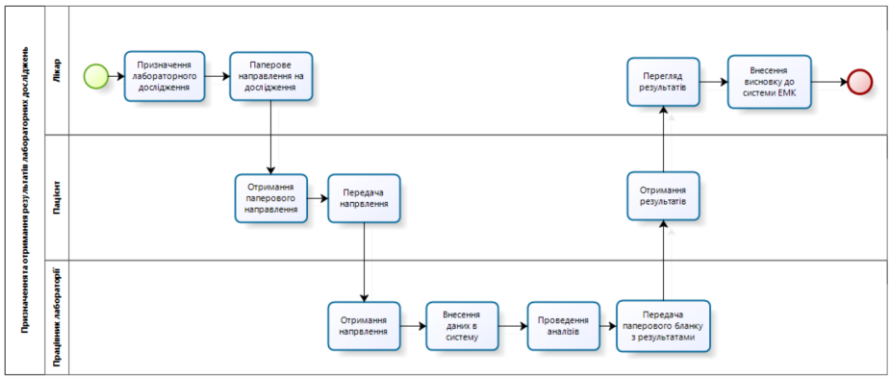


Рисунок 1.4 – Призначення та отримання результатів лабораторних досліджень

Джерело: власна розробка у Bizagi Process Modeler [7]

У цьому бізнес-процесі було знайдено наступні слабкі місця: повністю паперовий документообіг між лікарем та лабораторією, відсутність автоматичного сповіщення про готовність і електронного архіву лабораторних показників пацієнта.

1.1.3 Характеристика існуючої інформаційної системи

На момент дослідження в медичному закладі функціонувала локальна клієнт-серверна система, розроблена декілька років тому і яка з того часу практично не оновлювалася. ІС забезпечувала лише базові операції: реєстрація пацієнтів, планування прийомів і мінімальне ведення медичної картки.

Електронна картка хворого містить лише загальні відомості такі як: прізвище, ім’я, дату народження, контактні дані, а також короткі нотатки про попередні візити. Можливості для структурованого документування скарг, результатів огляду, анамнезу чи динаміки показників здоров’я в системі не передбачено. Через це лікарі змушені вести паралельну паперову документацію, що суттєво збільшувало час на оформлення кожного прийому.

Пошук пацієнта виконується повільно, автозаповнення і контекстні підказки відсутні. Поля введення не мають валідації, тобто система не перевіряє коректність номера телефону чи адреси електронної пошти, не виявляє дублікати записів і не попереджає персонал про можливі помилки. Внаслідок цього в базі нагромаджуються записи з некоректними або суперечливими даними.

Модуль планування прийомів відображає розклад лікарів лише в спрощеному вигляді без візуалізації реального навантаження протягом дня. Персонал реєстратури не може оперативно оцінити завантаженість конкретного спеціаліста, що ускладнює розподіл записів та призводить до нерівномірного завантаження лікарів.

Система функціонує виключно в локальній мережі закладу і не має жодних інтеграційних можливостей. Виписування рецептів формування направлень на лабораторні дослідження та отримання їх результатів відбуваються поза системою, тобто на паперових носіях. Передача інформації між підрозділами здійснюється ручну, що спричиняє затримки і помилки при перенесенні даних. Лікар не має електронного доступу до результатів аналізів та дізнається про них лише тоді, коли пацієнт приносить паперовий бланк на наступний прийом.

Аналітичні можливості системи відсутні повністю, тому керівництво закладу не може переглянути статистику завантаженості персоналу, структуру захворюваності чи фінансові показники у режимі реального часу. Формування будь-якої звітності потребує ручного збору та обробки даних, що займає значний час і знижує точність результатів.

З точки зору безпеки ІС також не відповідає сучасним вимогам. Розмежування прав доступу реалізовано поверхнево, журнал дій користувачів не ведеться, а резервне копіювання даних виконується нерегулярно та без автоматичної перевірки цілісності копій.

1.1.4 Обґрунтування необхідності вдосконалення системи

Аналіз поточного стану інформаційного забезпечення медичного центру показав, що існуюча система не відповідає ні потребам персоналу, ні вимогам законодавства, ні очікуванням клієнтів.

Найбільш критичною проблемою є відсутність інтеграції з ЕСОЗ. Без підключення до державної системи заклад не може виписувати рецепти, що позбавляє пацієнтів можливості скористатися програмою «Доступні ліки» та іншими державними програмами відшкодування вартості медикаментів [8]. У конкурентному середовищі приватної медицини це є суттєвим недоліком, який впливає на привабливість закладу для відвідувачів.

Паралельне ведення документації в паперовому та електронному форматах щодня забирає у персоналу час який міг би витрачатись на безпосередню роботу з пацієнтами. Лікар змушений двічі фіксувати одну й ту саму інформацію, а медична сестра записувати показники на папері, бо система не підтримує їх введення. Крім втрати часу, це породжує розбіжності між записами та підвищує ризик помилок.

Відсутність електронного обміну з лабораторією затягує процес лікування. Від моменту призначення аналізів до отримання їх результатів лікарем проходить

значно більше часу, ніж потрібно, оскільки весь цей шлях пацієнт долає самостійно з паперовими бланками. Жодних сповіщень система не генерує, тому медичний персонал дізнається про результати лише на повторному візиті хворого.

Керівництво закладу фактично позбавлене інструментів для аналізу роботи центру. Дані про завантаженість лікарів, популярність послуг чи фінансові показники можна отримати лише шляхом ручного збору інформації.

Впровадження нової інформаційної системи дозволить вирішити всі виявлені проблеми. Перехід на повністю електронний документообіг усуне дублювання роботи і знизить кількість помилок. Інтеграція з ЕСОЗ забезпечить виписування електронних рецептів і участь у державних програмах. Підключення до лабораторної системи скоротить час між призначенням і отриманням результатів досліджень. Модуль аналітики дозволить керівництву отримувати актуальні дані для управлінських рішень.

1.2 Аналіз літературних джерел та практичного досвіду використання інформаційних систем і технологій в предметній області

Тематика автоматизації медичних закладів та впровадження електронних карток активно розглядається в науковій літературі. Зарубіжні дослідники приділяють значну увагу архітектурним підходах до побудови ЕМК-систем, стандартам обміну даними, а також на питанням захисту персональної інформації пацієнтів. У вітчизняних роботах акцент зміщений на особливості впровадження електронних карток в умовах реформування охорони здоров’я України та специфіці інтеграції з ЕСОЗ. Навіть попри велику кількість публікацій, питання розробки доступних рішень для невеликих приватних закладів залишається недостатньо висвітленим.

З метою визначення функціональних вимог та архітектурних підходів при проєктуванні системи проведено аналіз п’яти існуючих програмних рішень.

Helsi є провідним українським сервісом електронної охорони здоров'я, який розроблений з урахуванням вимог українського законодавства [9]. Платформа забезпечує ведення структурованих медичних записів, виписування е-рецептів та направлень, онлайн-запис пацієнтів, інтеграцію з програмою «Доступні ліки», а також сервісом «Дія». Доступ здійснюється через браузер або мобільний застосунок. Базовий функціонал надається безкоштовно, що робить систему привабливою для закладів з обмеженим бюджетом. Серед переваг користувачі відзначають простоту підключення до медичної картки, зрозумілий інтерфейс та оперативну технічну підтримку.

Doctor Eleks – українська система для приватних клінік та медичних центрів із повною інтеграцією з ЕСОЗ [10]. Підтримує загальне ведення ЕМК, виписування електронних рецептів, формування направлень і автоматичну перевірку декларацій пацієнтів. Система має сучасну веб-орієнтовану архітектуру з можливістю хмарного або локального розгортання. Інтерфейс оптимізований для швидкої роботи лікарів і адаптований для різних пристроїв.

eClinicalWorks є одним із найбільш поширених комерційних рішень для електронного ведення медичної документації [11]. Система пропонує електронні форми, інтегровані модулі для дистанційного виписування рецептів та роботи з лабораторними результатами, а також інструменти підтримки клінічних рішень на основі алгоритмів штучного інтелекту. Попри широкий функціонал, платформа не має інтеграції з ЕСОЗ, що суттєво обмежує можливості її застосування в українських закладах.

Athenahealth обслуговує понад 160 тисяч лікарів і зберігає медичні дані більш ніж 117 мільйонів пацієнтів [12]. Основний продукт AthenaOne включає електронні медичні записи, інструменти для клінічної роботи і управління фінансовими потоками. Система має один із найбільш сучасних інтерфейсів серед інших ЕМК-рішень і оновлюється автоматично кожні 4-6 тижнів. Як і eClinicalWorks, платформа орієнтована на американський ринок і не має інтеграції з ЕСОЗ.

Veradigm розрахована на великі медичні заклади та обслуговує понад 180 тисяч медичних працівників [13]. Платформа включає повну ЕМК з підтримкою

всіх медичних напрямків, електронне призначення лікарів з перевіркою взаємодій препаратів та алергій, а також розвинутий аналітичний модуль із застосуванням ШІ. Veradigm розрахована на масштабні організації і вимагає початкових інвестицій, що фактично виключає її з переліку рішень, доступних для малих приватних закладів.

Порівняльний аналіз існуючих програмних рішень наведено у таблиці 1.1.

Таблиця 1.1 – Порівняльний аналіз систем електронних медичних карток

№	Можливості	Helsi	Doctor Eleks	eClinicalWorks	Athenahealth	Varadigm
1	Структуровані медичні записи	Так	Так	Так	Так	Так
2	Інтеграція з державними системами	Так	Так	Ні	Ні	Ні
3	Інтеграція з лабораторією	Так	Так	Так	Так	Так
4	Особистий кабінет пацієнта	Так	Так	Так	Так	Так
5	Аналітика та звітність	Так	Так	Так	Так	Так
6	Архітектура	Хмарна	Хмарна	Хмарна	Хмарна	Хмарна
7	Мобільний додаток	Так	Так	Так	Так	Так
8	Модель ліцензування	Freemium	Комерційна	Комерційна	Комерційна	Комерційна
9	Безкоштовна базова версія	Так	Ні	Ні	Ні	Ні
10	Кількість закладів	1800+	1700+	110000+	270000+	50000+
11	Відгуки користувачів	Позитивні	Позитивні	Змішані	Позитивні	Змішані

Джерело: власна розробка на основі даних з офіційних сайтів [9], [10], [11], [12], [13].

За результатами порівняння виявлено, що усі розглянуті системи побудовані на сучасних хмарних технологіях, підтримують веб-доступ і мають мобільні застосунки. Головною відмінністю між українськими та закордонними рішеннями є наявність інтеграції з ЕСОЗ: Helsi і Doctor Eleks повністю відповідають вимогам вітчизняного законодавства, тоді як американські рішення проєктувались під інший ринок і не мають необхідних інтеграцій. Найдоступнішим з фінансової точки зору є Helsi завдяки безкоштовному базовому функціоналу.

Водночас аналіз виявив, що жодна з розглянутих систем не задовольняє повною мірою потреби невеликого приватного закладу без додаткових витрат на налаштування. Це підтверджує доцільність розробки власного рішення, яке поєднає необхідний функціонал, інтеграцію з ЕСОЗ і доступність впровадження.

1.3 Обґрунтування вибору методів та технологій для створення інформаційної системи

Перед початком розробки визначено ключові критерії до технологічного стеку: система має працювати через браузер, підтримувати кілька категорій користувачів з різними правами, надійно зберігати медичні дані і мати готовність до підключення ЕСОЗ.

Платформу побудовано за трирівневою клієнт-серверною архітектурою з розділеними на рівень представлення, рівень бізнес-логіки та рівень даних [14]. Кожен рівень розвивається незалежно, серверний централізує перевірку прав доступу, а при зростанні навантаження окремі компоненти масштабуються без переписування системи. Інтерфейс реалізовано як веб-застосунок на Bootstrap [15], серверна частина включає API Gateway, сервіс авторизації і основний медичний модуль, рівень даних представлений PostgreSQL.

Серверну частина написано на Python з мікрофреймворком Flask [16]. Flask не нав’язує жорсткої структури проєкту, що дозволило організувати модулі відповідно до функціональних блоків системи. З’єднання з базою даних реалізовано через Flask-SQLAlchemy, авторизацію через Flask-Login, паролі зберігаються у захешованому вигляді засобами Werkzeug.

Для зберігання даних обрано PostgreSQL [17]. Медичні дані мають реляційну природу: між пацієнтами, лікарями, прийомами, рецептами і результатами аналізів існують чіткі зв’язки. PostgreSQL підтримує складні аналітичні запити і гарантує транзакційну обробку, що критично важливо для медичної системи. База включає

десять таблиць, окремо виділено журнал аудиту для фіксації всіх операцій відповідно до вимог захисту персональних даних.

Архітектуру системи змодельовано засобами UML відповідно до стандарту ISO/IEC 19514:2017. Побудовано діаграми прецедентів, послідовності, класів та трасування у середовищі Enterprise Architect [18]. До початку розробки в Figma створено інтерактивний прототип інтерфейсу, що дозволило перевірити логіку навігації ще до написання коду.

РОЗДІЛ 2
РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Аналіз і специфікація вимог до інформаційної системи

2.1.1 Бізнес-вимоги

Бізнес-вимоги відображають стратегічні цілі медичного центру, досягнення яких очікується після впровадження системи (рис. 2.1) [19].

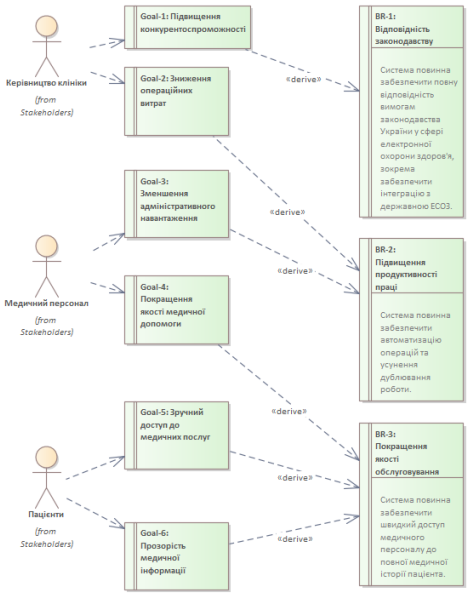


Рисунок 2.1 – Діаграма бізнес-вимог до ІС ЕМК

Джерело: власна розробка в Enterprise Architect [18]

Бізнес-вимога BR-1 стосується відповідності законодавству України у сфері електронної охорони здоров’я. Після впровадження системи всі рецепти виписуватимуться виключно через ЕСОЗ без паперових бланків, декларації

перевірятимуться автоматично, а участь у державних програмах медичного забезпечення дозволить розширити клієнтську базу закладу.

Бізнес-вимога BR-2 спрямована на підвищення продуктивності персоналу через автоматизацію рутинних операцій. Очікується скорочення часу на первинне введення даних, зменшення часу пошуку потрібної інформації лікарем під час прийому та збільшення кількості пацієнтів, яких один спеціаліст може прийняти протягом робочого дня без подовження робочого часу.

Бізнес-вимога BR-3 визначає підвищення якості обслуговування через забезпечення доступу персоналу до повної історії кожного пацієнта та надання зручного особистого кабінету. Показниками виконання є зниження частки неявок завдяки автоматичним нагадуванням і зростання рівня задоволеності.

2.1.2 Функціональні вимоги

Функціональні вимоги описують, що саме система повинна вміти робити [20]. Кожна з них прив’язана до конкретної бізнес-цілі через відношення «satisfy» (рис. 2.2).

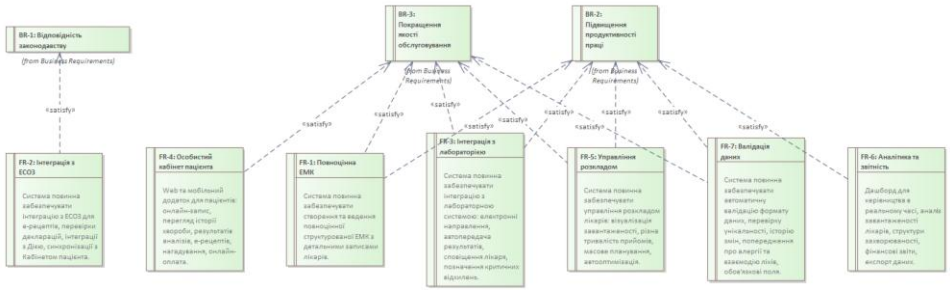


Рисунок 2.2 – Діаграма функціональних вимог до ІС ЕМК

Джерело: власна розробка в Enterprise Architect [18]

Функціональна вимога FR-1 охоплює ведення медичної документації в електронному вигляді. Лікар повинен мати змогу вносити скарги пацієнта,

результати огляду та призначення у структурованому вигляді з використанням шаблонів. Система також зберігає відомості про алергічні реакції і дозволяє прикріплювати файли до картки пацієнта.

Функціональна вимога FR-2 описує роботу з державною системою охорони здоров'я. Лікар виписує рецепт у системі, і він автоматично реєструється в ЕСОЗ. Паперові рецепти після впровадження цієї функції не використовуватимуться взагалі. Система також перевіряє наявність та статус декларації пацієнта при кожному зверненні.

Функціональна вимога FR-3 стосується роботи з лабораторією. Замість паперового направлення лікар формує електронне одразу в системі. Після того як лабораторія завершує дослідження, результати автоматично з'являються в медичній картці, а лікар отримує сповіщення.

Функціональна вимога FR-4 визначає можливості для клієнтів. Через особистий кабінет у браузері або мобільному застосунку пацієнт самостійно обирає зручний час і записується на прийом, переглядає свою медичну карту і завантажує електронні рецепти.

Функціональна вимога FR-5 описує управління розкладом. Персонал реєстратури бучить реальну завантаженість кожного лікаря на будь-який день, враховуючи тривалість різних типів консультацій, якщо ж виникає конфлікт у графіку, система про це одразу повідомляє.

Функціональна вимога FR-6 охоплює аналітику і звітність. Керівник закладу отримує зведені дані про кількість прийомів, статус захворюваності, завантаженість лікарів та фінансові показники. Звіти можна завантажити у зручному форматі.

Функціональні вимоги FR-7 описує валідацію даних. При введенні будь-якої інформації система перевіряє її формат, шукає можливі дублікати і автоматично повідомляє лікаря, якщо призначений препарат може викликати алергічну реакцію у конкретного пацієнта.

2.1.3 Нефункціональні вимоги

Нефункціональні вимоги визначають характеристики роботи системи незалежно від конкретного функціоналу [21]. Діаграму зображено на рисунку 2.3.

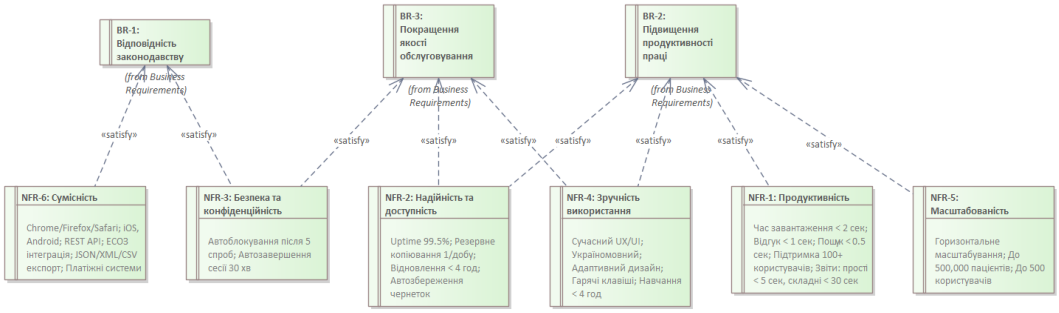


Рисунок 2.3 – Діаграма нефункціональних вимог до ІС ЕМК

Джерело: власна розробка в Enterprise Architect [18]

Нефункціональна вимога NFR-1 стосується швидкодії, коли в системі одночасно працює більше ста користувачів, то кожна сторінка повинна відкриватися не довше трьох секунд. Невеликі звіти формуються до п’яти секунд, а об’ємні - до тридцяти.

Нефункціональна вимога NFR-2 визначає стабільність роботи. Система має бути доступною не менше 99,5% робочого часу. Щоночі виконується резервне копіювання з перевіркою цілісності даних, а у разі збою відновлення завершується протягом чотирьох годин.

Нефункціональна вимога NFR-3 регулює захист персональних даних пацієнтів. Весь трафік між браузером і сервером шифрується через TLS [22]. Облікові записи з розширеними правами входять у систему через двофакторну автентифікацію. П’ять невдалих спроб авторизації поспіль призводять до автоматичного блокування.

Нефункціональна вимога NFR-4 стосується зручності освоєння. Новий співробітник розбирається з базовими функціями за п’ятнадцять хвилин, а повне навчання роботі з усіма модулями займає не більше чотирьох годин.

Нефункціональна вимога NFR-5 описує масштабованість. При зростанні навантаження окремі компоненти нарощуються незалежно один від одного. Система коректно працює при п’ятистах одночасних підключеннях і базі у п’ятсот тисяч записів.

Нефункціональна вимога NFR-6 визначає сумісність. Система однаково працює в усіх браузерах і надає REST API для зовнішніх інтеграцій. Архітектура передбачає можливість підключення стандарту обміну медичними даними HL7 FHIR у майбутньому [23].

2.1.4 Специфікація вимог

Описані вимоги забезпечують повне покриття потреб медичної установи, а також формують основу для архітектурного проєктування системи. Трасованість реалізуються через логічні зв’язки між різними рівнями вимог, де кожна функціональна та нефункціональна вимога має чіткий зв’язок із відповідними бізнес-вимогами. Критерій вимірюваності здійснюється шляхом визначення конкретних ключових показників, які стануть основою для подальшої оцінки успішності системи після її впровадження. Порядок пріоритетності вимог виконувався шляхом їх бізнес-цінності та обсягу ресурсів, необхідних для реалізації. Такий підхід дає змогу структурувати етапи розгортання системи, надаючи перевагу впровадженню найважливіших функціональних та нефункціональних аспектів, що мають найвищий рівень значущості.

Специфікацію вимог з ідентифікаторами, типами, статусами та пріоритетністю наведено у додатку А.

2.2 Постановка задачі та опис алгоритму роботи системи

2.2.1 Постановка задачі

Розробка системи ЕМК спрямована на автоматизацію процесів документування медичної інформації, управління прийомами та формування звітності в приватному медичному центрі. Необхідність зумовлена функціональною застарілістю наявного програмного забезпечення та відсутністю інтеграції з державними системами.

Система обробляє дані кількох ключових об’єктів: пацієнти медичного закладу, їхні медичні картки, електронні рецепти, результати лабораторних досліджень та показники роботи закладу. Інформаційний ресурс із медичними відомостями оновлюється при кожній взаємодії користувача із системою з очікуваним часом відповіді менше п’яти секунд. Графік роботи оновлюється в режимі реального часу при внесення будь-яких змін, фінансові показники формуються щодня, а комплексні звіти генеруються щомісяця.

Програмне забезпечення реалізує вісім основних блоків. Управління медичними картками охоплює створення, перегляд і редагування електронних карток із повною історією звернень. Запис на прийом забезпечує онлайн-бронювання часових слотів через особистий кабінет або реєстратуру. Управління розкладом включає формування і коригування графіку з автоматичним виявленням конфліктів. Фіксування життєвих показників дозволяє медичній сестрі вносити вимірювання перед прийомом. Виписування електронних рецептів здійснюється з автоматичною реєстрацією в ЕСОЗ. Управління лабораторними дослідженнями включає створення направлень і отримання результатів. Аналітика забезпечує формування звітів про роботу закладу. Інтеграція з державними системами дозволяє обмінюватися даними з ЕСОЗ та «Дією». Інформаційна модель системи наведена на рисунку 2.4.

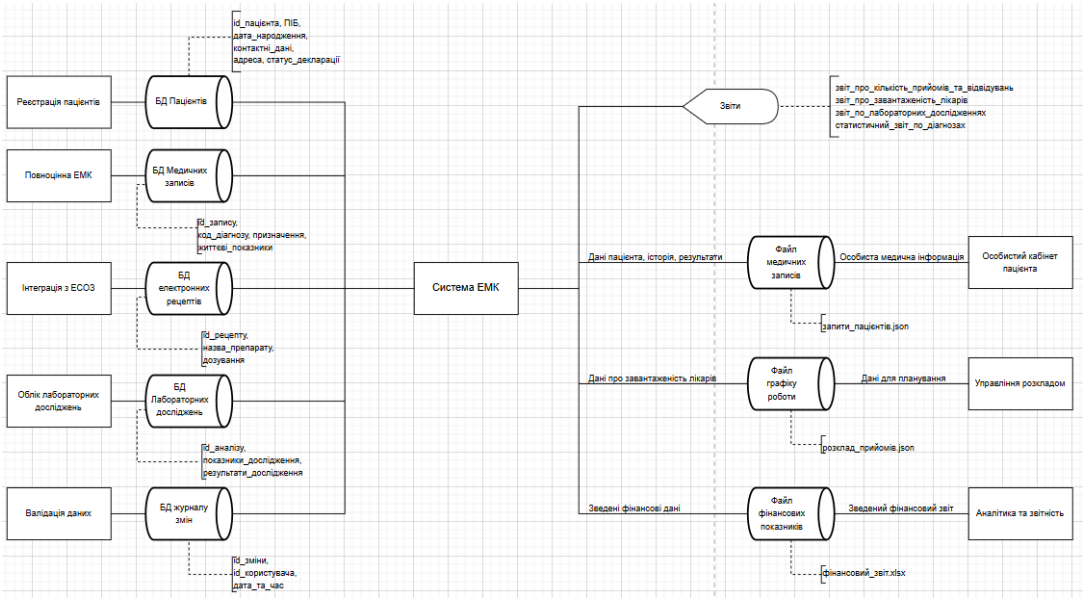


Рисунок 2.4 – Інформаційна модель системи ЕМК

Джерело: власна розробка

Система взаємодіє з кількома зовнішніми підсистемами. ЕСОЗ приймає електронні рецепти і надає інформацію про декларації пацієнтів та дані з «Дії». Лабораторна система отримує електронні направлення і передає результати досліджень у форматах HL7, JSON або XML протягом одного-трьох днів. Модуль фінансового обліку отримує дані про проведені прийоми для формування звітності. Особистий кабінет пацієнта синхронізується з основною системою для відображення актуальної інформації. Обмін між підсистемами відбувається через асинхронні API-виклики для некритичних операцій і синхронні для тих, що потребують негайної відповіді.

Автоматизований режим роботи припиняється у таких випадках: відсутність зв’язку з ЕСОЗ під час операцій із рецептами, критичні збої бази даних, які унеможливлюють відновлення з резервної копії, втрата доступу до мережі понад тридцять хвилин, а також виявлення спроб несанкціонованого доступу до медичних даних. За таких обставин система переходить у режим обмеженої функціональності.

Вихідна інформація системи призначена для різних категорій користувачів і формується у відповідних форматах. Перелік і опис вихідних повідомлень наведено в таблиці 2.1.

Таблиця 2.1 – Перелік і опис вихідних повідомлень

№	Назва вихідного повідомлення	Ідентифікатор	Форма подання	Періодичність	Термін видання і час затримки	Користувачі інформації
1	Файл медичних записів	OUT_EMR	HTML, PDF	При кожному зверненні	< 5 секунд	Пацієнти, лікарі
2	Файл графіку роботи	OUT_SCHED	JSON, HTML	У реальному часі	< 2 секунд	Персонал реєстратури, лікарі
3	Файл фінансових показників	OUT_FIN	XLSX, PDF	Щоденно/на вимогу	< 30 секунд	Керівництво
4	Звіти про діяльність	OUT_REPORT	PDF, XLSX	Щомісяця	<5 хвилин	Керівництво

Джерело: власна розробка

Файл медичних записів OUT_EMR генерується у форматах HTML і PDF при кожному звернення з часом відповіді до п’яти секунд. Він містить повну медичну картку пацієнта, включаючи персональні дані, хронологію візитів із діагнозами і призначеннями, результати аналізів, активні рецепти та записи на майбутні прийоми. Користувачами цієї інформації є лікарі та пацієнти.

Файл графіку роботи OUT_SCHED готується у форматах JSON і HTML в режимі реального часу з затримкою не більше двох секунд. У його структурі відображається розклад лікарів із переліком часових слотів, їх поточним статусом і відсотком завантаженості. Кінцевими користувачами є персонал реєстратури та лікарі.

Файл фінансових показників OUT_FIN експортується у форматах XLSX і PDF щодня або на вимогу користувача з часом створення до тридцяти секунд. Він містить відомості про загальні доходи, кількість прийомів, середній чек, а також структуру захворюваності. Отримувачами виступає керівництво закладу.

Звіти про діяльність OUT_REPORT вивантажуються у форматах PDF і XLSX щомісяця з часом генерації до п’яти хвилин. Вони містять зведену статистику за звітний період, зокрема кількість пацієнтів, їх розподіл за віком і статтю, загальну кількість рецептів та проведених лабораторних досліджень. Користувачами є керівництво закладу.

Ескізи форм вихідних повідомлень наведено в Додатку Б.

Вхідна інформація системи надходить із різних джерел і використовується для наповнення бази даних та забезпечення роботи функціональних модулів. Перелік і опис вхідних повідомлень наведено в таблиці 2.2.

Таблиця 2.2 – Перелік і опис вхідних повідомлень

№	Назва вхідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
1	Дані пацієнтів	IN_PAT	Екранна форма	При реєстрації	Реєстратура
2	Медичні записи	IN_MED	Екранна форма	При кожному візиті	Лікарі
3	Дані з ЕСОЗ	IN_ESOZ	JSON	Періодично	ЕСОЗ
4	Результати досліджень	IN_LAB	HL7, JSON, XML	1-3 дні	Лабораторія
5	Дані для валідації	IN_VALID	Усі екранні форми	Постійно	Всі користувачі

Джерело: власна розробка

Дані пацієнтів IN_PAT вводяться персоналом реєстратури через екранну форму при першому зверненні. Включають прізвище, ім’я та по-батькові, дату народження, стать, ідентифікаційний код, номер телефону, адресу електронної пошти та адресу проживання.

Медичні записи IN_MED вносяться лікарем через інтерфейс системи під час кожного прийому. Включають скарги пацієнта, анамнез, результати огляду, попередній і остаточний діагноз за МКХ-10 [24] та призначене лікування.

Дані з ЕСОЗ IN_ESOZ надходять у форматі JSON в автоматичному режимі. Містять статус декларації пацієнта, відомості про лікаря первинної ланки, термін дії декларації та право на безоплатні ліки.

Результати досліджень IN_LAB передаються лабораторною системою у форматах HL7, JSON або XML протягом одного-трьох днів після виконання

аналізів. Містять номер направлення, найменування дослідження, значення показників із одиницями виміру та референсними діапазонами [25].

Дані для валідації IN_VALID формуються автоматично при кожній операції і включають тип операції, ідентифікатор користувача та дату і час виконання.

Ескізи форм вхідних повідомлень наведені в Додатку В.

2.2.2 Алгоритм роботи системи

Усі запити від користувачів проходять через API Gateway, який перенаправляє їх до відповідних сервісів. Алгоритм виписування електронного рецепті наведено на рисунку 2.5.

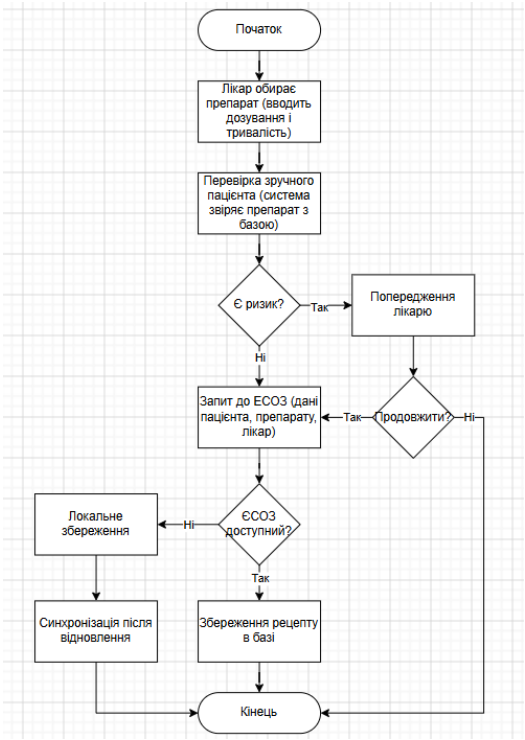


Рисунок 2.5 – Блок-схема алгоритму виписування електронного рецепту

Джерело: власна розробка

Кожна сесія починається з перевірки облікових даних через Auth Service. Введений пароль хешується і порівнюється із збереженим в базі, якщо ж дані збігаються, то відкривається сесія з правами доступу відповідно до ролі користувача. Проте п’ять невдалих спроб підряд призводять до автоматичного блокування облікового запису.

Коли пацієнт звертається вперше до закладу, персонал реєстратури шукає його за ідентифікаційним кодом, якщо запису немає, заповнюється реєстраційна форма. Кожне поле перевіряється при введенні: формат телефону, коректність електронної пошти, унікальність коду.

На прийомі лікар відкриває картку і бачить повну медичну історію разом із показниками, які медична сестра внесла до початку консультації. Діагноз обирається з довідника МКХ-10. При призначенні препарату система звіряє його з відомими алергіями пацієнта і показує попередження, якщо є ризик несумісності.

Виписування рецепту відбувається в кілька кроків. Спершу лікар обирає препарат і вказує дозування, далі система формує запит до електронної системи охорони здоров’я, звідти повертається унікальний ідентифікатор, і рецепт зберігається в базі з прив’язкою до медичного запису. Якщо ЕСОЗ у цей момент недоступний, рецепт зберігається локально і синхронізується автоматично після відновлення з’єднання.

Направлення до лабораторії лікар формує електронно прямо в системі. Integration Service передає його в лабораторну систему. Коли аналізи готові, вони передаються у форматі HL7 або JSON. Система зіставляє їх з направленням за ідентифікатором, додає до картки пацієнта і надсилає сповіщення лікарю.

При записі на прийом програмне забезпечення показує вільні слоти з урахуванням тривалості різних типів консультацій. Якщо виникає накладка в графіку, система одразу повідомляє персонал реєстратури і пропонує найближчий вільний час, а за день до прийому пацієнт отримує автоматичне нагадування.

2.2.3 Математичне забезпечення системи

Система використовує декілька математичних методів для обробки даних і підтримки функціональності модулів.

1. Розрахунок завантаженості лікарів

Відсоток завантаженості кожного лікаря за обраний період розраховується за формулою:

$$Z = \left(\frac{N_{\text{зайнятих}}}{N_{\text{загальних}}}\right) * 100\%, \quad (2.1)$$

де N_зайнятих - кількість зайнятих часових слотів, N_загальних - загальна кількість доступних слотів за той самий період. Цей показник використовується в аналітичному модулі і використовується керівництвом для оцінки ефективності використання робочого часу персоналу.

2. Виявлення конфліктів у розкладі

Для двох записів на прийом конфлікт виникає тоді, коли часові інтервали перетинаються. Перевірка виконується за умовою:

$$K = (t_A_1 < t_B_2)\text{^}(t_B_1 < t_A_2), \quad (2.2)$$

де t_A1 і t_A2 - час початку і завершення прийому A, t_B1 і t_B2 - час початку і завершення прийому B. Якщо K = true, то система блокує запис і повідомляє персонал про накладку в графіку.

3. Розрахунок середнього чека

Для формування фінансової звітності система розраховує середній дохід на прийом за обраний період:

$$C = \frac{D_{\text{загальних}}}{N_{\text{прийомів}}}, \quad (2.3)$$

де D_загальних - сума надходжень за період, N_прийомів - кількість завершених прийомів. Показник відображається в аналітичному дашборді поряд із загальною виручкою та структурою витрат.

4. Алгоритм хешування паролів

Для захисту облікових даних використовується алгоритм PBKDF2-SHA256 [26]. При реєстрації пароль перетворюється на хеш із застосуванням унікального випадкового ключа, що генерується окремо для кожного користувача:

H = PBKDF2(password, random_key, 26000 iterations, SHA-256)

При автентифікації введений пароль хешується з тим самим ключем і порівнюється зі збереженим значенням.

5. Перевірка контактних даних

Перевірка формату контактних даних виконується регулярними виразами на стороні сервера при кожній операцій запису. Номер телефону перевіряється за маскою +380XXXXXXXXX, електронна пошта - за стандартом RFC5322 [27]. Такий підхід унеможливлює збереження некоректних даних навіть у разі обходу клієнтської перевірки.

2.3 Моделювання інформаційної системи

2.3.1 Моделювання поведінки системи

Для визначення функціональних можливостей системи з точки зору користувачів побудовано діаграму прецедентів. Платформа взаємодіє з шістьма категоріями внутрішніх користувачів: пацієнт, лікар, медична сестра, працівник реєстратури, адміністратор системи та керівник медичного закладу. Зовнішніми акторами є ЕСОЗ та лабораторна система.

Система включає дев’ять основних прецедентів: «Управління ЕМК», «Виписування електронних рецептів», «Управління лабораторними дослідженнями», «Запис пацієнта на прийом», «Управління розкладом лікарів», «Генерування звітів та аналітики», «Валідація медичних даних», «Управління

користувачами», «Фіксування життєвих показників». Прецедент «Виписування електронних рецептів» включає «Інтеграція з ЕСОЗ» через відношення «include», оскільки кожен виписаний рецепт автоматично реєструється в державній системі. «Управління лабораторними дослідженнями» включає «Відправлення направлення до лабораторії» та «Отримання результатів з лабораторії» (рис. 2.6).

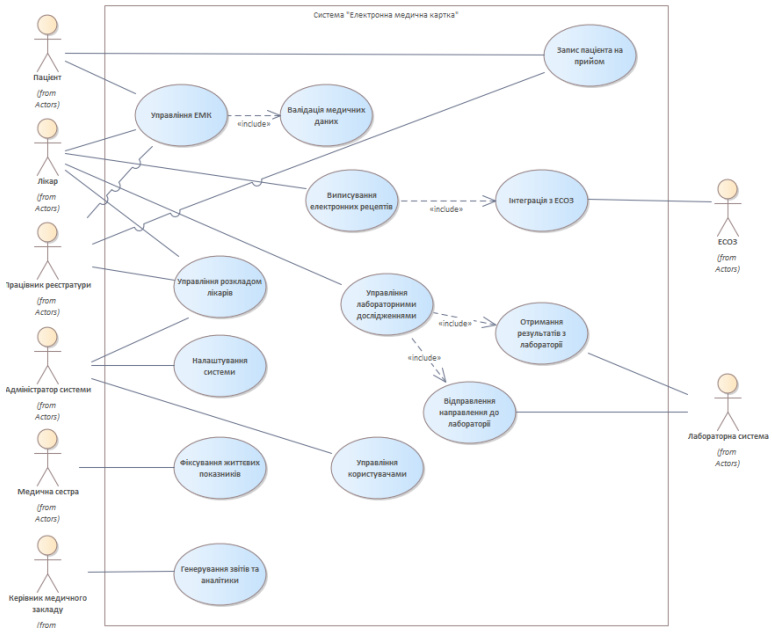


Рисунок 2.6 – Діаграма прецедентів системи ЕМК

Джерело: власна розробка в Enterprise Architect [18]

Для кожного прецеденту розроблено текстові сценарії з описом передумов, основного потоку виконання, альтернативних потоків та післяумов. Як приклад наведено сценарії для прецеденту «Фіксування життєвих показників» (рис. 2.7 - 2.10).

Пацієнт зареєстрований в системі	Pre-condition
Медсестра авторизована в системі	Pre-condition
Пацієнт присутній на прийомі	Pre-condition

Рисунок 2.7 – Передумови прецеденту «Фіксування життєвих показників»

Джерело: власна розробка в Enterprise Architect [18]

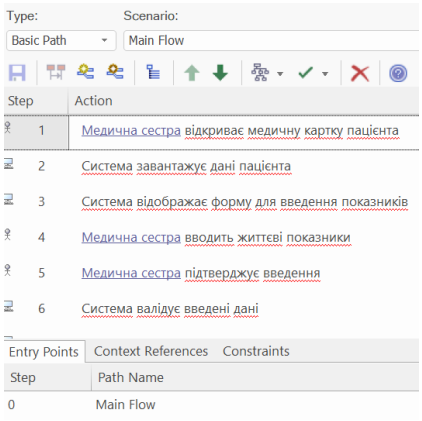


Рисунок 2.8 – Основний сценарій прецеденту «Фіксування життєвих показників»

Джерело: власна розробка в Enterprise Architect [18]

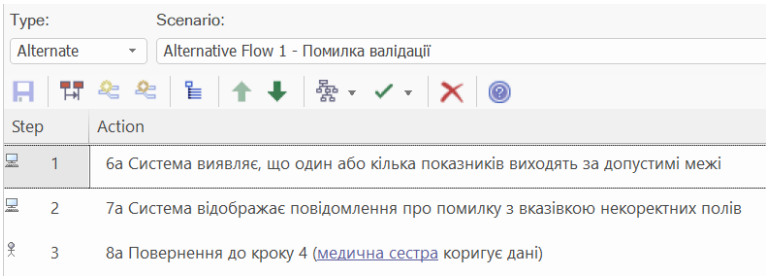


Рисунок 2.9 – Альтернативний сценарій «Помилка валідації»

Джерело: власна розробка в Enterprise Architect [18]

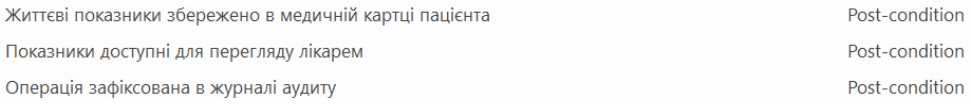


Рисунок 2.10 – Післяумови прецеденту «Фіксування життєвих показників»

Джерело: власна розробка в Enterprise Architect [18]

На основі текстових сценаріїв побудовано діаграми послідовності для кожного прецеденту, які відображають взаємодію між акторами та компонентами системи за шаблоном «Boundary → Control → Entity». Як приклад наведено діаграму послідовності для прецеденту «Фіксування життєвих показників» (рис. 2.11).

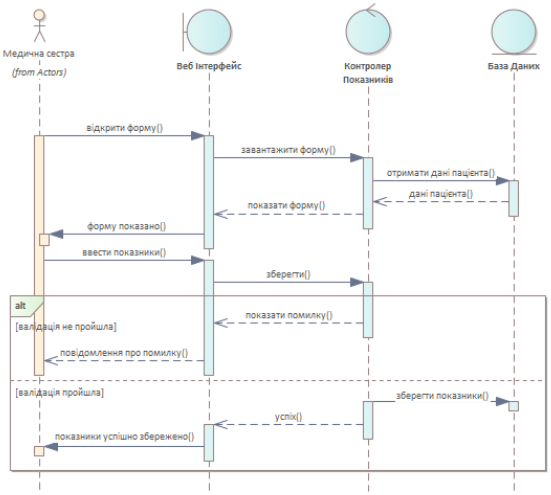


Рисунок 2.11 – Діаграма послідовності «Фіксування життєвих показників»

Джерело: власна розробка в Enterprise Architect [18]

Медична сестра надсилає запит на відкриття форми до веб-інтерфейсу, який далі звертається до контролера показників зі запитом на завантаження форми, контролер отримує дані пацієнта з бази і повертає заповнену форму. Після введення показників виконується збереження запису, і перевіряється коректність заповнених полів. Якщо дані некоректні, то веб-інтерфейс повертає повідомлення про помилку, якщо ж валідація пройшла, то показники зберігаються в базі даних і повертає підтвердження. Діаграми решти діаграм послідовності наведено у Додатку Г.

2.3.2 Моделювання структури системи

Модель системи представлено діаграмою класів, що відображає її статичну структуру системи відповідно до шаблону «Boundary → Control → Entity» (рис. 2.12).

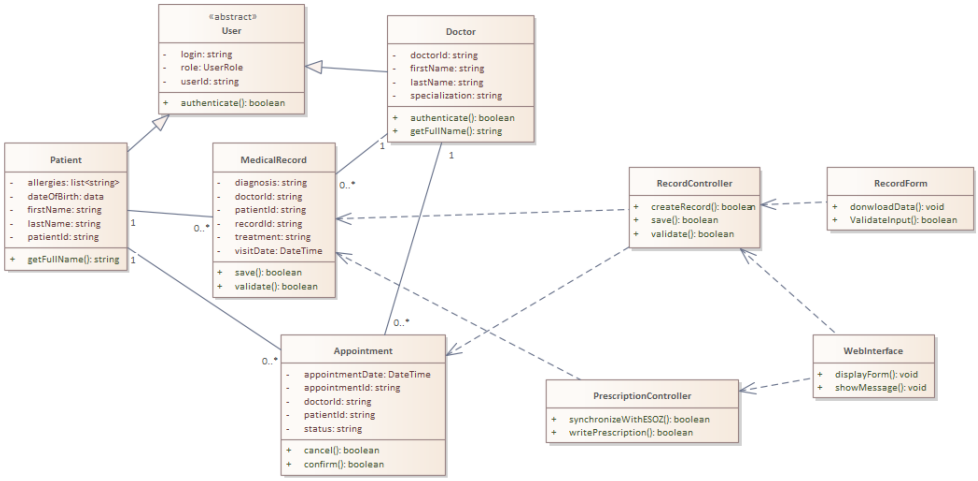


Рисунок 2.12 – Діаграма класів

Джерело: власна розробка в Enterprise Architect [18]

Класи сутності моделюють дані предметної галузі. Базовий клас User містить загальні атрибути всіх користувачів. Patient та Doctor успадковують User і додають специфічні атрибути. MedicalRecord зберігає діагноз, скарги та лікування. Appointment моделює запис на прийом.

Граничні класи забезпечують інтерфейс системи. WebInterface відображає форми та повідомлення. RecordForm валідує дані медичних записів.

Класи керування координують бізнес-логіку системи. RecordController управляє створенням записів та прийомів. PrescriptionController відповідає за рецепти та синхронізацію з ЕСОЗ.

На діаграмі класів відображено відношення узагальнення (generalization) між базовим класом User та його спеціалізаціями Patient та Doctor асоціації між класами-сутностями (Patient та Doctor мають відношення один-до-багатьох з MedicalRecord і Appointment), а також залежності (dependency) між граничними класами, класами керування та класами-сутностями відповідно до правила «Boundary → Control → Entity». Така структура забезпечує зв’язування елементів, спрощує тестування та подальше розширення функціональності системи.

2.3.3 Розподіл вимог за компонентами системи

Верифікацію проєктних рішень виконано за допомогою трасування функціональних вимог до прецедентів, класів системи та тест-кейсів. Такий підхід дозволяє чітко перевірити, чи повністю реалізовано кожну вимогу, визначити компоненти системи, а також забезпечити повне покриття тестами та вчасно виявити надлишкові елементи. Діаграму зображено на рисунку 2.13.

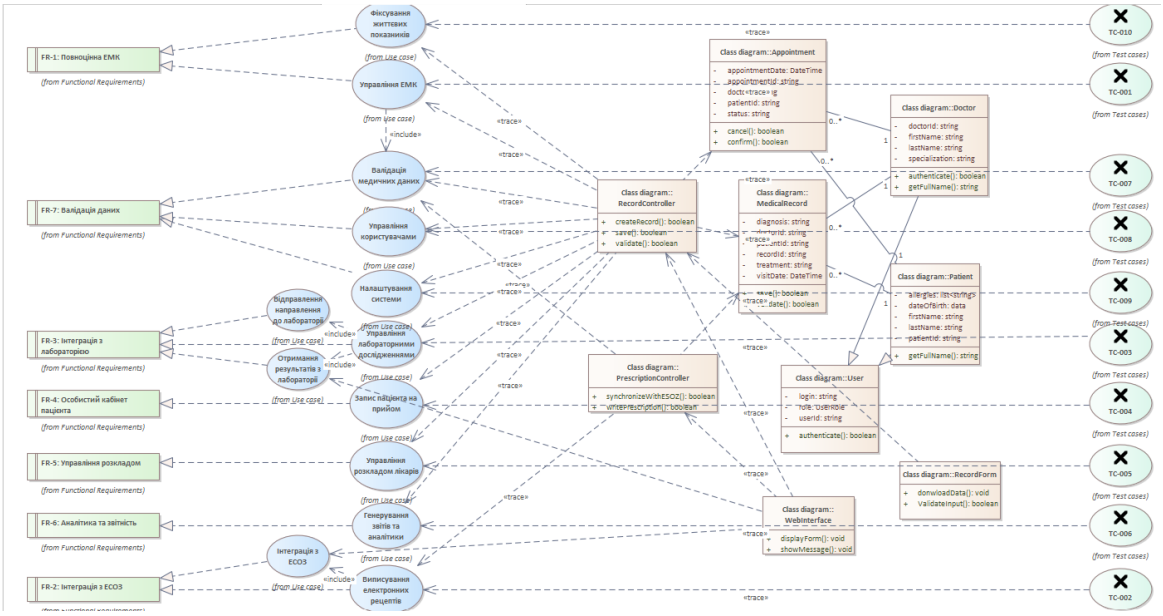


Рисунок 2.13 – Діаграма трасування вимог до системи

Джерело: власна розробка в Enterprise Architect [18]

Усі сім функціональних вимог виконуються через відповідні прецеденти. Вимога FR-1 реалізується за допомогою прецедентів «Управління ЕМК» та «Фіксування життєвих показників». Виконання FR-2 покладено на «Виписування електронних рецептів» та «Інтеграцію з ЕСОЗ». FR-3 охоплює роботу прецедентів «Відправлення направлення до лабораторії» та «Отримання результатів з лабораторії». Процедура «Запис на прийом» закриває потребу вимоги FR-4, а FR-5 реалізується через «Управління розкладом лікаря». FR-6 виконується за допомогою

«Генерування звітів та аналітики». Вимога FR-7 охоплює прецеденти «Валідація медичних даних», «Управління користувачами» та «Налаштування системи».

Матриця трасування підтверджує, що кожна функціональна вимога пов’язана щонайменше з одним прецедентом, завдяки чому жодна не залишилась без практичної реалізації (рис. 2.14).

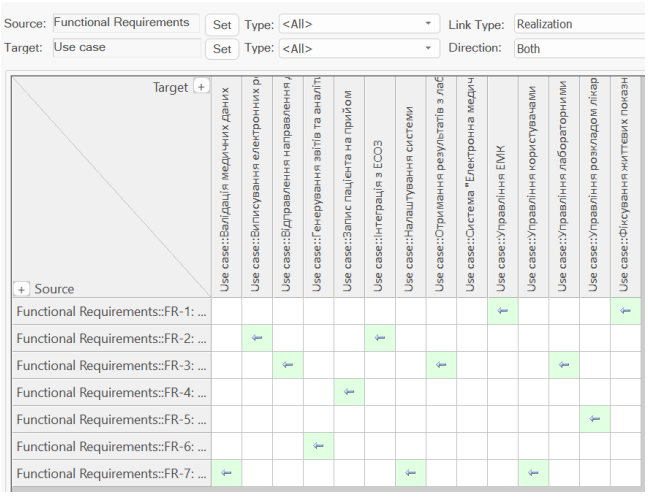


Рисунок 2.14 – Матриця трасування функціональних вимог до прецедентів

Джерело: власна розробка в Enterprise Architect [18]

Клас керування RecordController відповідає за більшість прецедентів системи, оскільки основна частина операцій стосується безпосередньої роботи з медичними записами та прийомами. PrescriptionController виділено в окремий клас через специфічну логіку інтеграції з ЕСОЗ та перевірку декларацій пацієнтів (рис. 2.15).

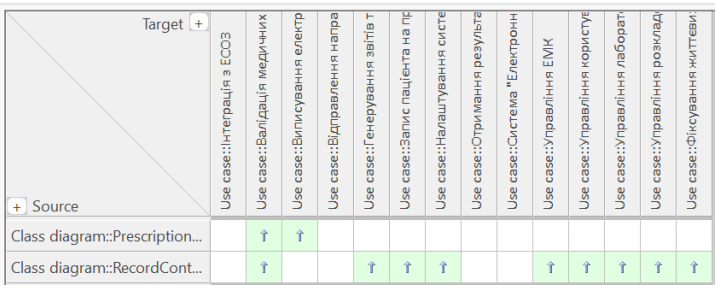


Рисунок 2.15 – Матриця трасування класів керування до прецедентів

Джерело: власна розробка в Enterprise Architect [18]

Для відображення інтерфейсу користувача в усіх прецедентах використовується WebInterface, який виступає єдиним граничним класом системи. При цьому RecordForm задіяний лише у тих сценаріях, де потрібно вводити та зберігати медичні дані пацієнтів (рис. 2.16).

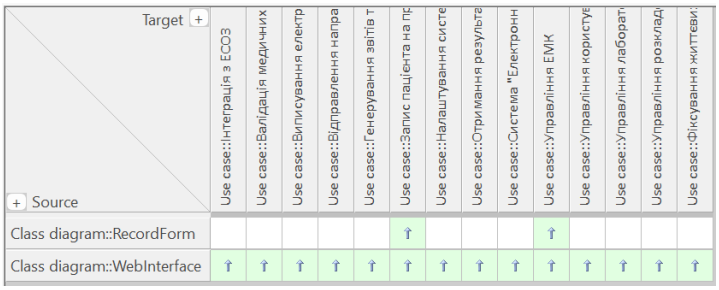


Рисунок 2.16 – Матриця трасування граничних класів до прецедентів

Джерело: власна розробка в Enterprise Architect [18]

Кожен основний прецедент покривається відповідним тест-кейсом у діапазоні від ТС-001 до ТС-010. Усі вони мають статус «To be specified», що означає необхідність їх детального опрацювання на етапі тестування із визначенням вхідних даних, очікуваних результатів та критеріїв прийняття (рис. 2.17).

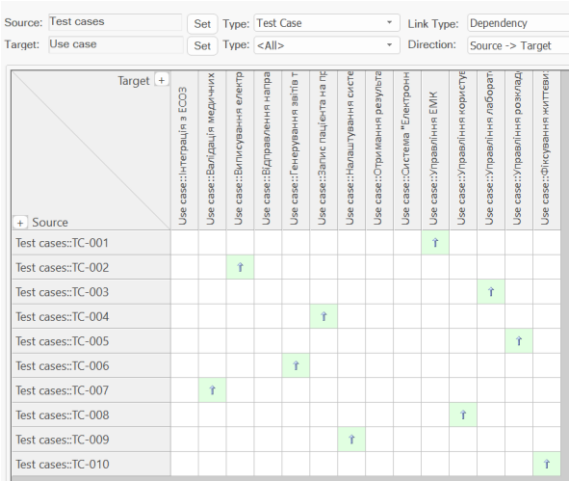


Рисунок 2.17 – Матриця трасування тест-кейсів до прецедентів

Джерело: власна розробка в Enterprise Architect [18]

Проведене трасування підтверджує три головні характеристики проєктних рішень. По-перше, забезпечено повноту покриття, оскільки всі сім функціональних

вимог реалізовані через прецеденти. По-друге, в архітектурі немає надлишкових компонентів, бо всі розроблені класи задіяні хоча б в одному ланцюжку зв’язків. По-третє, система має повне покриття тестами, адже для кожного варіанта використання створено свій тест-кейс.

РОЗДІЛ 3
ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1 Інформаційне забезпечення

3.1.1 Загальна характеристика інформаційного забезпечення

Базу даних, вхідну й вихідну документацію, нормативно-довідкові матеріали та методи кодування інформації відносять до інформаційного забезпечення системи ЕМК. Головна його мета полягає у тому, щоб безперебійно постачати всім функціональним модулям актуальні й точні відомості.

Загалом інформаційне забезпечення прийнято ділити на дві складові: позамашинну та внутрішньомашинну [28]. До позамашинної належать нормативні класифікатори й довідники, необхідні для щоденної роботи. Це, зокрема, міжнародний класифікатор хвороб МКХ-10 для фіксації діагнозів, реєстр лікарських засобів для оформлення рецептів, а також списки лікарських спеціальностей і видів лабораторних аналізів. Внутрішньомашинний компонент містить безпосередньо конфігураційні файли та реляційну базу даних, що складається з десяти таблиць.

Характер інформації в системі залежить від її призначення та способів використання. До оперативної категорії належать відомості, які оновлюються щодня: відкриті направлення, поточні медичні записи, та активний запис на прийом. Аналітичний сегмент формується на основі оперативних відомостей шляхом їх групування й підрахунку за певний проміжок часу, після чого ці дані відображаються у звітах для керівної ланки клініки. До архівного типу відносять повністю завершені медичні історії пацієнтів, редагування яких після закриття картки прийому стає неможливим.

Інформаційні потоки передаються як усередині архітектури, так і під час взаємодії із зовнішніми платформами. Внутрішній інформаційний обмін реалізовано між окремими модулями через спільну базу даних. Зовнішня взаємодія потрібна для надсилання е-рецептів до системи ЕСОЗ, верифікації пацієнтських декларацій, а також імпорту лабораторних результатів у загальноприйнятих форматах JSON або HL7.

Для збереження цілісності інформації використовуються механізми транзакційної обробки та зовнішні ключі на рівні таблиць бази даних. Безпека даних базується на розмежуванні прав згідно з ролями користувачів, шифруванні мережевого трафіку за допомогою протоколу TLS і криптографічному захисті паролів алгоритмом PBKDF2-SHA256. Будь-які коригування чи внесення нових медичних записів автоматично фіксуються журналом аудиту із прив’язкою до конкретного облікового запису та часу події, що повністю відповідає нормативним вимогам законодавства про захист персональних даних.

3.1.2 Побудова системи класифікації та кодування

У системі ЕМК кодуванню підлягають такі об’єкти, як пацієнти, лікарі, прийоми, медичні записи, діагнози, рецепти, направлення на дослідження, результати досліджень, життєві показники та ролі користувачів. Для кожного об’єкта визначено конкретний метод кодування, а також структуру коду.

Більшість об’єктів системи ідентифікуються числовим автоінкрементним ідентифікатором, який PostgreSQL генерує автоматично при створення нового запису. Такий підхід гарантує унікальність кожного запису без ручного введення і повністю виключає дублювання.

Діагнози кодуються відповідно до Міжнародної класифікації хвороб десятого перегляду МКХ-10, що представляє собою ієрархічну алфавітно-цифрову систему з кодами виду A00-Z99. Перша літера позначає клас хвороб, а наступні цифри

уточнюють захворювання. Це забезпечує сумісність із державними медичними реєстрами та ЕСОЗ.

Рецепти після виписування отримують внутрішній числовий ідентифікатор prescription_id, а після реєстрації в ЕСОЗ їм присвоюється зовнішній ідентифікатор esoz_id у форматі XXXX-XXXX-XXXX-XXXX [29]. Результати лабораторних досліджень передаються за стандартом HL7 і прив'язуються до направлення через order_id.

Ролі користувачів кодуються скороченими англійськими назвами посад, що відповідають функції користувача в системі, а саме doctor, admin, registrar. Роль зберігається у полі role таблиці users і визначає рівень доступу до даних.

Таблиця 3.1 – Система класифікації та кодування об'єктів системи ЕМК

№	Об'єкт	Ідентифікатор	Метод кодування	Структура коду	Формат	Приклад
1	Пацієнт	patient_id	Порядкове (числове)	Ціле число, автоінкремент	N	1, 2, 3...
2	Лікар	doctor_id	Порядкове (числове)	Ціле число, автоінкремент	N	1, 2, 3...
3	Прийом	appointment_id	Порядкове (числове)	Ціле число, автоінкремент	N	1, 2, 3...
4	Медичний запис	record_id	Порядкове (числове)	Ціле число, автоінкремент	N	1, 2, 3...
5	Діагноз	Код МКХ-10	Ієрархічний	1 літера + 2-4 цифри	AXXXX	J06.9
6	Рецепт (система)	prescription_id	Порядкове (числове)	Ціле число, автоінкремент	N	1, 2, 3...
7	Рецепт (ЕСОЗ)	esoz_id	Зовнішній	4 групи по 4 символи	XXXX-XXXX-XXXX-XXXX	1234-5678-9012-3456
8	Направлення	order_id	Порядкове (числове)	Ціле число, автоінкремент	N	1, 2, 3...
9	Результат аналізу	result_id	Порядкове (числове)	Ціле число, автоінкремент	N	1, 2, 3...
10	Роль користувача	role	Кодування скороченими назвами посад	Текстове значення	-	doctor, admin, registrar

Продовження таблиці 3.1

11	Життєві показники	vital_id	Порядкове (числове)	Ціле, автоінкремент	N	1, 2, 3 ...

Джерело: власна розробка

3.1.3 Джерела вхідних даних та форми їх подання

Вхідні інформаційні масиви для системи ЕМК формуються за допомогою двох типів джерел: первинних та вторинних. Роль первинних джерел виконує персонал медичної установи, який заповнює екранні форми веб-інтерфейсу в ручному режимі. Вторинними джерелами вважаються сторонні цифрові платформи, які інтегруються в систему та автоматично надсилають відомості через API, зокрема електронна система охорони здоров'я та лабораторна система. Перелік усіх джерел вхідної інформації міститься в таблиці 3.2.

Таблиця 3.2 – Джерела вхідних даних та форми їх подання

Джерело даних	Тип джерела	Форми подання даних	Коментарі
Персонал реєстратури	Первинне	Структуровані дані з веб-платформи	Персональні дані пацієнта, запис на прийом
Лікар	Первинне	Структуровані дані з веб-платформи	Медичний запис, діагноз МКХ-10, призначення
Медична сестра	Первинне	Структуровані дані з веб-платформи	Життєві показники пацієнта перед прийомом
ЕСОЗ	Вторинне	Напівтруктуровані дані, JSON через REST API	Статус декларації, ідентифікатор рецепту
Лабораторна система	Вторинне	Напівтруктуровані дані, HL7 або JSON	Результати лабораторних досліджень

Джерело: власна розробка

Побудова каналів, якими надходить інформація, повністю залежить від обраного типу джерела. Передача первинних відомостей організована через безпечний HTTPS-протокол між клієнтським браузером та сервером. Автоматичне завантаження вторинної інформації з бази ЕСОЗ відбувається за допомогою REST

API у форматі JSON безпосередньо в момент обробки рецептів чи перевірки статусів декларацій. Дані від лабораторної системи надходять у стандартах HL7 або JSON одразу після завершення аналізів та автоматично прив’язуються до відповідних електронних направлень завдяки наявності спільного ідентифікатора замовлення.

3.1.4 Інфологічна модель бази даних

Інфологічна модель відображає концептуальну структуру предметної області незалежно від конкретної СУБД. Вона описує інформаційні об’єкти системи, їх атрибути та зв’язки між ними.

Для побудови інфологічної моделі обрано реляційну модель, оскільки медичні дані мають чітку ієрархічну структуру з визначеними зв’язками між сутностями. На відміну від моделей зірка або сніжинка, які застосовують переважно в сховищах даних для аналітичної обробки, реляційна модель краще підходить для оперативних транзакційних систем, де важлива цілісність інформації і підтримка складних зв’язків між сутностями.

Основними інформаційними об’єктами системи є десять сутностей. Сутність «Користувач» зберігає облікові відомості всіх категорій персоналу системи і є базовою для сутностей «Пацієнт» та «Лікар». Сутність «Пацієнт» містить персональні та медичні дані хворого, включно з інформацією про алергії та статус декларації. Сутність «Запис на прийом» пов’язує пацієнта і лікаря через конкретну дату і час та фіксує статус прийому. Сутність «Медичний запис» містить клінічну інформацію про візит: діагноз за МКХ-10, скарги, результати огляду та призначене лікування. Сутність «Життєві показники» фіксує вимірювання стану здоров’я пацієнта перед прийомом. Сутність «Рецепт» зберігає відомості про призначені препарати та ідентифікатор реєстрації в ЕСОЗ. Сутність «Направлення на дослідження» фіксує запит лікаря до лабораторії. Сутність «Результат

дослідження» зберігає отримані від лабораторії показники з референсними діапазонами. Сутність «Журнал аудиту» фіксує всі операції зміни даних у системі.

Між сутностями визначено такі зв’язки. Користувач має один запис Пацієнта або Лікаря (1:1). Пацієнт та Лікар можуть мати багато Записів на прийом (1:N). Запис на прийом містить один Медичний запис (1:1..0), а також пов’язаний з одним набором Життєвих показників (1:1). Медичний запис може мати багато Рецептів та Направлень на дослідження (1:N). Направлення містить один або кілька Результатів досліджень (1:N). Кожна операція користувача фіксується в Журналі аудиту (N:1). ER-діаграму інфологічної моделі бази даних наведено на рисунку 3.1.

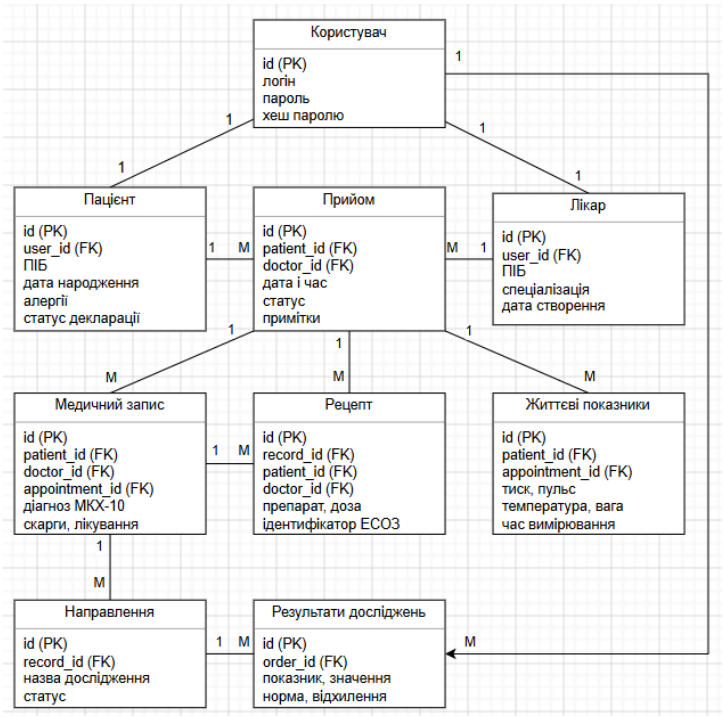


Рисунок 3.1 – Інфологічна модель бази даних системи ЕМК

Джерело: власна розробка

3.1.5 Даталогічна модель бази даних

Зберігання даних системи ЕМК реалізовано засобами реляційної СУБД PostgreSQL. Доцільність використання цієї бази даних підтверджується її

архітектурними можливостями. Платформа PostgreSQL дозволяє ефективно опрацьовувати складні інформаційні запити та забезпечує транзакційну ізольованість процесів, що є обов'язковою умовою для медичного софту, де збої під час збереження карток можуть призвести до серйозних наслідків. Безпека структури та зв'язків підтримується на рівні бази завдяки обмеженням зовнішніх ключів. Додатковим аргументом на користь цього вибору є те, що PostgreSQL є безкоштовним програмним забезпеченням, що дозволяє уникнути витрат на купівлю ліцензій та знижує загальну вартість впровадження системи в невеликих приватних клініках.

Внутрішня архітектура бази охоплює десять окремих таблиць, кожна з яких повністю відповідає критеріям третьої нормальної форми [30]. За автоматичне створення унікальних числових кодів для нових записів відповідають первинні ключі з налаштованим типом даних. Логічні зв'язки між сутностями системи встановлюються за допомогою зовнішніх ключів. Нижче наведено детальний опис полів для трьох базових таблиць проєкту, тому як структури інших внесені у Додаток Д.

Таблиця 3.3 – Структура таблиці «Користувачі» (users)

Назва поля (атрибута)	**Фізичне ім'я поля**	**Тип даних**	**Обмеження / Ключ**
Ідентифікатор користувача	user_id	SERIAL	PRIMARY KEY
Логін	login	VARCHAR (100)	NOT NULL, UNIQUE
Хеш паролю	password_hash	VARCHAR (255)	NOT NULL
Роль	role	VARCHAR (50)	NOT NULL
Дата створення	created_at	TIMESTAMP	DEFAULT NOW ()

Джерело: власна розробка

Таблиця 3.4 – Структура таблиці «Пацієнти» (patients)

Назва поля (атрибута)	**Фізичне ім'я поля**	**Тип даних**	**Обмеження / Ключ**
Ідентифікатор пацієнта	patient_id	SERIAL	PRIMARY KEY
Ідентифікатор користувача	user_id	INTEGER	FOREIGN KEY (user)
Ім'я	first_name	VARCHAR (100)	NOT NULL
Прізвище	last_name	VARCHAR (100)	NOT NULL

Продовження таблиці 3.4

По батькові	middle_name	VARCHAR (100)	-
Дата народження	date_of_birth	DATE	NOT NULL
Номер телефону	phone	VARCHAR (20)	-
Електронна пошта	email	VARCHAR (100)	-
Адреса проживання	address	VARCHAR (200)	-
Наявність алергій	allergies	TEXT	-
Статус декларації	declaration_status	BOOLEAN	DEFAULT FALSE
Дата створення	created_at	TIMESTAMP	DEFAULT NOW ()

Джерело: власна розробка

Таблиця 3.5 – Структура таблиці «Медичні записи» (medical_records)

Назва поля (атрибута)	**Фізичне ім'я поля**	**Тип даних**	**Обмеження / Ключ**
Ідентифікатор запису	record_id	SERIAL	PRIMARY KEY
Ідентифікатор пацієнта	patient_id	INTEGER	FOREIGN KEY (patients)
Ідентифікатор лікаря	doctor_id	INTEGER	FOREIGN KEY (doctor)
Ідентифікатор прийому	appointment_id	INTEGER	FOREIGN KEY (appointments)
Код діагнозу МКХ-10	diagnosis_code	VARCHAR (10)	NOT NULL
Опис діагнозу	diagnosis_description	TEXT	-
Скарги	complaints	TEXT	-
Результати огляду	examination_results	TEXT	-
Лікування	treatment	TEXT	-
Дата візиту	visit_date	TIMESTAMP	NOT NULL
Дата створення	created_at	TIMESTAMP	DEFAULT NOW ()

Джерело: власна розробка

Схематичне зображення наявних сутностей та зв'язків між ними наведено на рисунку 3.2. Тестовий набір даних для перевірки працездатності вміщено у Додатку Е.

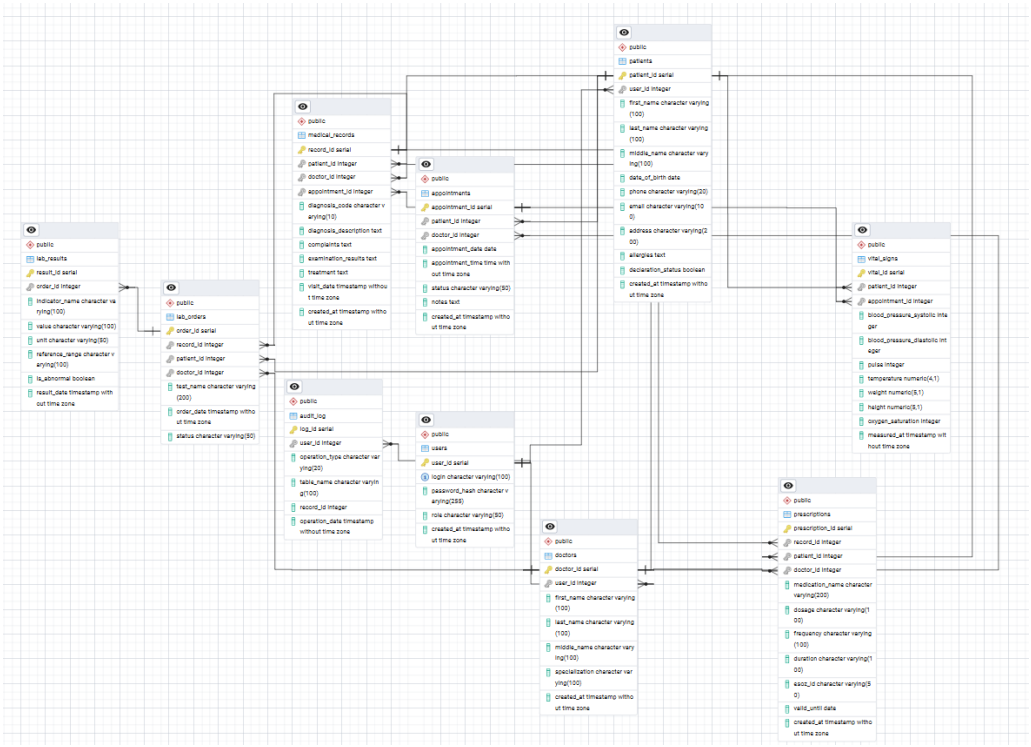


Рисунок 3.2 – Даталогічна модель бази даних системи ЕМК

Джерело: власна розробка

3.2 Технічне забезпечення

3.2.1 Загальні положення та схема автоматизації

Система ЕМК розгорнута у локальній мережі приватного медичного центру. Серверна частина встановлена на виділеній апаратній платформі у серверному приміщенні закладу, де розміщено базу даних PostgreSQL та веб-застосунок Flask API. Весь персонал отримує доступ до системи через браузер з підключених до локальної мережі робочих місць. Підключення до інтернету необхідне насамперед для обміну даними із зовнішніми платформами - ЕСОЗ та лабораторною інформаційною системою, а також для резервного каналу зв’язку через 4G/LTE у разі недоступності основного провайдера.

У закладу функціонує п’ять типів робочих місць із різним призначенням. Стійка реєстратури оснащена терміналом для реєстрації нових звернень і ведення розкладу прийомів. У кожному лікарському кабінеті встановлено АРМ для документування медичних записів і виписування рецептів. Процедурний кабінет має окреме робоче місце, де медична сестра фіксує показники стану пацієнта перед консультацією. Лабораторія підключена до системи для роботи з направленнями та внесення результатів аналізів. Кабінет керівництва має доступ до аналітичного модуля для перегляду звітів про роботу закладу. Загальну схему автоматизації наведено на рисунку 3.3.

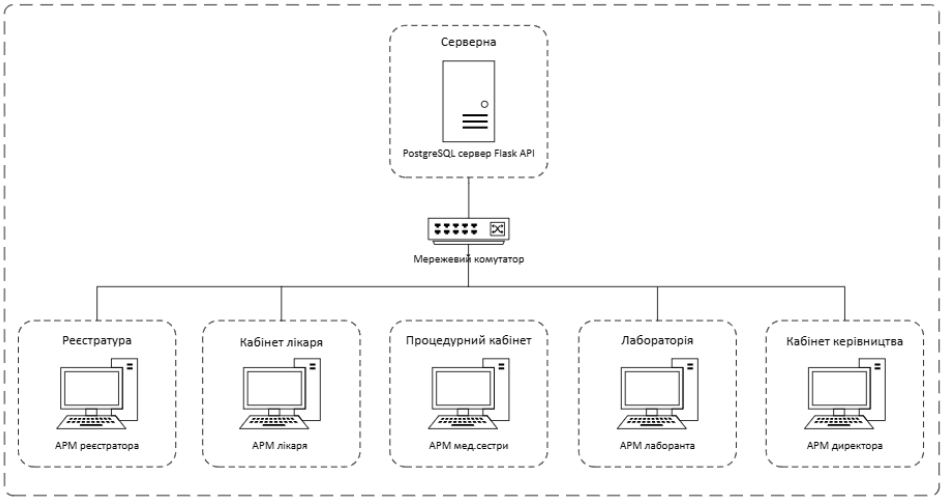


Рисунок 3.3 – Загальна схема автоматизації ІС ЕМК

Джерело: власна розробка

3.2.2 Структура комплексу технічних засобів

Апаратна архітектура системи ЕМК об’єднує центральний обчислювальний вузол, мережеву інфраструктуру, периферійні пристрої та автоматизовані робочі місця співробітників.

Головним вузлом інфраструктури виступає виділена апаратна платформа, яка встановлюється в замкненій комунікаційній шафі всередині серверного

приміщення клініки. На цьому обладнанні функціонує реляційна база даних PostgreSQL, а також розгорнуто веб-застосунок Flask API. Граничні апаратні параметри підібрані відповідно до нефункціональних вимог щодо швидкодії системи. Вони включають чотириядерний процесор із тактовою частотою від 2,4 ГГц, обсяг оперативної пам’яті від 8 ГБ, твердотільний накопичувач SSD ємністю не менше 500 ГБ, а також мережеву карту зі швидкістю передачі даних 1Гбіт/с. Для забезпечення безперервної роботи та виконання вимоги NFR-2 щодо відмовостійкості центральний вузол інтегровано з джерелом безперебійного живлення, яке дозволяє коректно завершити всі активні процеси у разі раптового знеструмлення будівлі.

Функціонал платформи розрахований на роботу п’яти автоматизованих робочих місць, кожне з яких відповідає певній ролі користувача, а саме АРМ реєстратора, АРМ лікаря, АРМ медичної сестри, АРМ лаборанта та АРМ директора. Стартова кількість місць визначається структурою медичного центру за принципом одного терміналу на кожну посаду, при цьому архітектура дозволяє підключати нові пристрої без модернізації центрального вузла.

Комунікаційна складова проєкту містить маршрутизатор із бездротовим модулем Wi-Fi для підключення до основного інтернет-каналу на швидкості 100Мбіт/с із виділеною статичною IP-адресою. Наявність постійної IP-адреси є критично важливою для успішної синхронізації та обміну даними з платформою ЕСОЗ. Для підвищення стабільності передбачено резервний 4G/LTE модем, який автоматично запускається під час збоїв на лінії головного провайдера. До складу мережі також входять гігабітний комутатор локальної мережі та апаратний міжмережевий екран, що блокує зовнішні загрози.

Допоміжне периферійне обладнання складається з мережевого принтера для друкування рецептів чи звітів, а також ноутбуків або планшетів, які використовуються для мобільної взаємодії з ЕМК через Wi-Fi за відсутності доступу до стаціонарного комп’ютера.

Безпека всього комплексу технічних засобів забезпечується на кількох рівнях одночасно. Серверні та комутаційні пристрої захищені від стороннього фізичного

втручання завдяки розміщенню у замкненому боксі. Мережевий трафік між клієнтськими веб-браузерами та центральним вузлом повністю шифрується за допомогою криптографічного протоколу TLS. Обмеження прав на перегляд інформації реалізовано на рівні програмного коду через систему користувацьких ролей, що задовольняє нормативні акти щодо безпеки персональних даних.

Технічний супровід та адміністрування розгорнутої інфраструктури покладається на одного ІТ-спеціаліста. До його обов’язків належать моніторинг апаратної платформи, контроль облікових записів персоналу та налаштування щонічного створення резервних копій бази даних.

3.2.3 Опис робочого місця користувача

Робочі місця системи ЕМК базуються на веб-орієнтованому підході. Вся взаємодія з інформаційним середовищем відбувається безпосередньо через браузер, завдяки чому відпадає потреба у розгортанні додаткових програмних модулів на локальних комп’ютерах. Таке рішення спрощує адміністрування мережі та мінімізує вимоги до обчислювальної потужності клієнтської техніки.

Для стабільного відображення інтерфейсу апаратні засоби мають задовольняти базові критерії. Достатніми параметрами є процесор із частотою від 1,6 ГГц, оперативна пам’ять від 4 ГБ, дисковий простір не менше 64 ГБ та монітор із роздільною здатністю від 1366x768 точок. Веб-застосунок однаково стабільно працює в середовищі сучасних версій Google Chrome, Mozilla Firefox або Microsoft Edge під управлінням будь-якої поширеної операційної системи.

Базова комплектація кожного робочого місця охоплює монітор, клавіатуру та мишу. Особливістю АРМ реєстратора є наявність мережевого принтера для роздрукування направлень та іншої паперової документації.

Обмін даними для стаціонарних комп’ютерів організовано через дротові лінії Ethernet на швидкості 1Гбіт/с, що забезпечує високу пропускну здатність і

відсутність затримок при роботі з базою даних. Робоче місце директора поєднує дротовий Ethernet та бездротовий Wi-Fi для зручного перегляду аналітики з мобільних пристроїв у межах закладу. Захист інформації під час передачі мережею забезпечується протоколом TLS без використання апаратного шифрування каналів. Склад технічних засобів усіх категорій робочих місць наведено у таблиці 3.6.

Таблиця 3.6 – Склад технічних засобів автоматизованих робочих місць

АРМ	**Комп'ютер**	**Монітор**	**Клавіатура**	**Миша**	**Принтер**	**Спосіб підключення**
Персонал реєстратури	1	1	1	1	1	Ethernet
Лікар	1	1	1	1	-	Ethernet
Медична сестра	1	1	1	1	-	Ethernet
Персонал лабораторії	1	1	1	1	-	Ethernet
Директор	1	1	1	1	-	Ethernet/Wi-Fi

Джерело: власна розробка

3.2.4 Схема мережі передачі даних

Побудова мережевої інфраструктури медичного центру базується на моделі зірка із застосуванням центрального комутатора локальної мережі. Організація підключення всіх компонентів системи через єдиний комунікаційний вузол спрощує адміністрування та дозволяє оперативно інтегрувати нові пристрої без модернізації наявної структури.

Внутрішня локальна мережа об'єднує сім основних пристроїв: апаратну платформу з розгорнутими PostgreSQL і Flask API, п'ять стаціонарних робочих місць та мережевий принтер. З'єднання пристроїв реалізовано дротовим кабелем витої пари категорії Cat 6, що забезпечує швидкість обміну інформацією на рівні

1Гбіт/с. Для мобільного доступу з ноутбуків або планшетів задіяно бездротовий Wi-Fi модуль маршрутизатора.

Вихід у глобальну мережу організовано через маршрутизатор. Основний канал надається провайдером зі швидкістю 100 Мбіт/с та статичною IP-адресою. Наявність фіксованої адреси є обов'язковою вимогою для стабільної взаємодії з ЕСОЗ, оскільки зовнішні сервери звертаються до вузла клініки за чітко визначеною точкою входу. Резервна лінія зв'язку побудована на базі 4G/LTE модема, який автоматично вмикається при збоях основного провайдера. Загальний комплект комунікаційного обладнання складається з одного маршрутизатора з підтримкою Wi-Fi, одного гігабітного комутатора локальної мережі, одного модема та апаратного міжмережевого екрана.

Особливості передачі трафіку залежать від посадових обов'язків кожної групи користувачів. Персонал реєстратури надсилає текстові дані екранних форм для оформлення пацієнтів та керування розкладом протягом восьми годин на добу за п'ятиденного робочого графіку тому обсяг його трафіку є мінімальним. Лікар формує структуровані медичні записи, надсилає запити до ЕСОЗ для виписування рецептів та отримує файли лабораторних досліджень протягом робочої зміни. Медична сестра завантажує числові відомості про життєві показники пацієнтів переважно в ранкові години прийому. Персонал лабораторії отримує електронні направлення та надсилає результати аналізів у форматах HL7 або JSON протягом восьми годин щодня. Директор робить запити на аналітичні звіти у форматах PDF або XLSX за потребою.

Взаємодія із зовнішніми системами спирається на інтернет-канал. Запити до ЕСОЗ передаються через REST API у форматі JSON з обов'язковим шифруванням трафіку за протоколом TLS. Результати аналізів надходять від лабораторної інформаційної системи у форматах HL7 або JSON після підтвердження виконання кожного дослідження.

Стабільність та безпека мережевої інфраструктури досягається комплексом захисних заходів. Резервування лінії зв'язку через 4G/LTE гарантує безперервність зовнішніх інтеграцій під час аварій основного провайдера. Джерело безперебійного

живлення захищає мережеві пристрої від стрибків напруги та раптового відключення електроживлення, забезпечуючи автономну роботу комунікаційного вузла щонайменше тридцять хвилин. Міжмережевий екран блокує спроби несанкціонованого підключення та забезпечує стійкість до зовнішніх мережевих загроз. Гарантійний термін експлуатації комунікаційного обладнання становить не менше трьох років, планові сервісні роботи виконуються раз на рік. Схему мережі передачі даних наведено на рисунку 3.4.

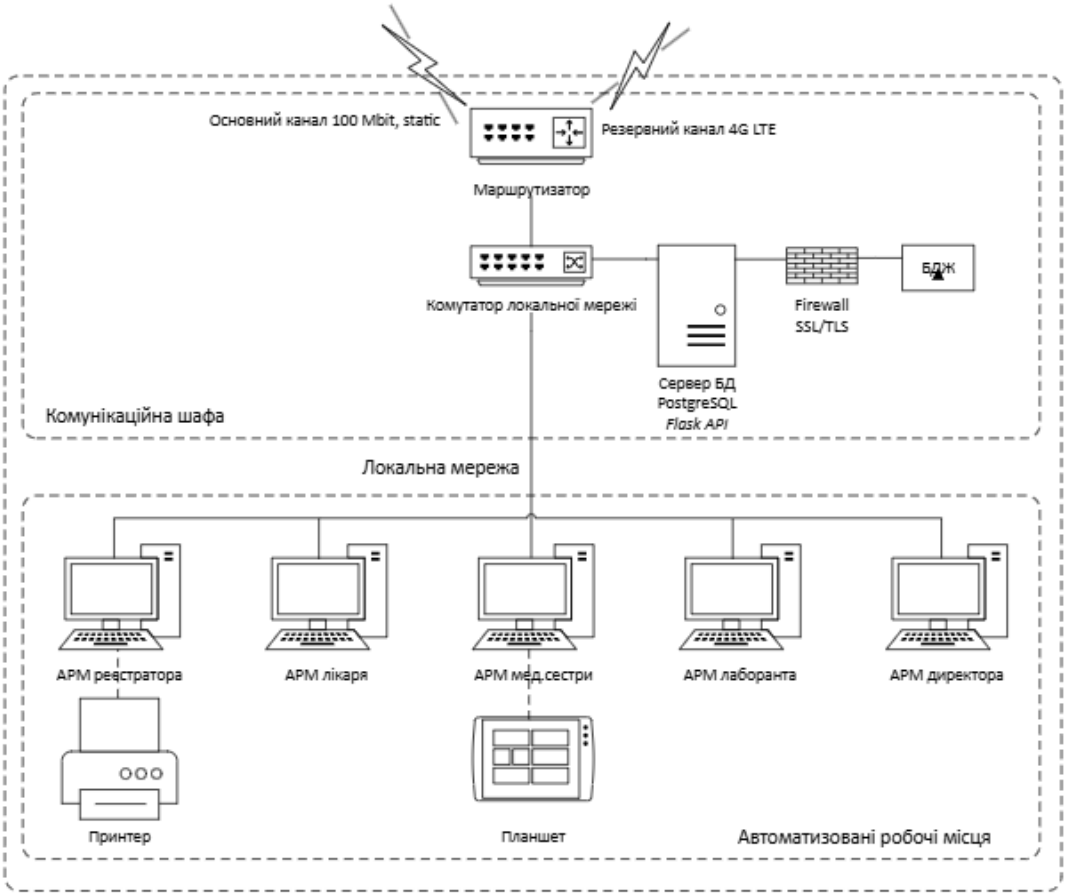


Рисунок 3.4 – Схема мережі передачі даних

Джерело: власна розробка

3.3 Програмне забезпечення

3.3.1 Засоби та технології розроблення

Програмну реалізацію системи ЕМК виконано на основі сучасного стеку технологій. Усі інструменти обрано з урахуванням затверджених функціональних та нефункціональних вимог до цього проєкту.

Серверну частину додатка написано мовою Python із використанням мікрофреймворку Flask. Цю мову обрано завдяки зрозумілому синтаксису, наявності великої кількості готових бібліотек та успішному досвіду створення схожих веб-ресурсів. Фреймворк Flask не обмежує структуру каталогів, завдяки чому вдалося зручно розподілити внутрішні модулі системи. Для керування маршрутами застосовано механізм Blueprint, який допомагає групувати обробники запитів у самостійні блоки.

Для взаємодії з базою даних підключено бібліотеку Flask-SQLAlchemy. Вона дозволяє керувати таблицями через стандартні класи Python без ручного написання складних SQL-запитів. Авторизацію користувачів та захист приватних веб-сторінок забезпечує розширення Flask-Login. Безпека облікових записів досягається шляхом збереження паролів у захешованому вигляді, що реалізовано засобами бібліотеки Werkzeug за допомогою алгоритму PBKDF2-SHA256.

Клієнтську частину створено за допомогою стандартного набору HTML, CSS та JavaScript, а для швидкої побудови адаптивного дизайну інтегровано фреймворк Bootstrap 5. Генерація екранних форм відбувається на стороні сервера через рушій Jinja2, який вбудовано у Flask. Описаний підхід забезпечує стабільне відображення візуальних елементів на різних пристроях медичного персоналу без потреби встановлення додаткового софту на комп’ютери користувачів.

Для тривалого збереження інформаційних масивів задіяна реляційна СУБД PostgreSQL. Комунікація між веб-сервером та базою даних координується через

драйвер psycopg2. Усі параметри підключення до бази винесені у файл конфігурації config.py, що значно спрощує перенесення коду між етапами розробки та промислової експлуатації.

Написання коду здійснювалося в редакторі Visual Studio Code з плагінами для підтримки Python та розпізнавання тегів Jinja2. Фіксація версій зовнішніх бібліотек ведеться за допомогою менеджера pip у текстовому файлі requirements.txt. Щоб уникнути системних конфліктів між пакетами різних програм, розробка проходила всередині ізольованого віртуального оточення Python venv. Для контролю ходу розробки, фіксації проміжних кроків та збереження історії проєкту було розгорнуто локальний репозиторій на базі системи Git. Зведений перелік головних програмних засобів та використаних бібліотек представлено в таблиці 3.7.

Таблиця 3.7 – Програмні засоби та бібліотеки системи ЕМК

Засіб / Бібліотека	Версія	Призначення
Python	3.13	Мова програмування серверної частини
Flask	3.1.1	Веб-фреймворк для побудови API і маршрутизації
Flask-SQLAlchemy	2.0.49	ORM для роботи з базою даних
Flask-Login	0.6.3	Управління сесіями та автентифікацією
Werkzeug	3.1.8	Хешування паролів, утиліти HTTP
psycopg2	2.9.12	Драйвер підключення до PostgreSQL
Jinja2	3.1.6	Шаблонізатор для генерації HTML
Bootstrap	5.3.0	CSS-фреймворк для адаптивного інтерфейсу
PostgreSQL	16	Реляційна СУБД для зберігання даних
Visual Studio Code	1.122.1	Середовище розробки
Git	2.51.1	Система контролю версій

Джерело: власна розробка

Нижче наведено приклад вихідного коду, що містить опис класу користувача з підключенням функції авторизації від інструменту Flask-Login:

```
from flask_sqlalchemy import SQLAlchemy
from flask_login import UserMixin
from datetime import datetime

db = SQLAlchemy()
```

```
class User(UserMixin, db.Model):
    __tablename__ = 'users'
    user_id = db.Column(db.Integer, primary_key=True)
        login = db.Column(db.String(100), unique=True,
nullable=False)
    password_hash = db.Column(db.String(255), nullable=False)
    role = db.Column(db.String(50), nullable=False)
    created_at = db.Column(db.DateTime, default=datetime.utcnow)

    def get_id(self):
        return str(self.user_id)
```

Джерело: власна розробка

Функція get_id повертає текстове значення представлення ідентифікатора користувача, що є обов'язковим критерієм для інтеграції з класом UserMixin із бібліотеки Flask-Login. Конфігурація unique=True для стовпця login блокує появу дублікатів імен користувачів на рівні бази даних.

3.3.2 Архітектура та структура програмного забезпечення

Програмна архітектура системи ЕМК базується на трирівневій клієнт-серверній моделі. Весь програмний комплекс розділено на три повністю незалежні сегменти: рівень відображення, рівень обробки бізнес-процесів та рівень збереження інформації. Завдяки такому підходу з'являється можливість оновлювати кожен рівень окремо, надійно ізолювати механізми перевірки прав доступу на стороні сервера, а також гнучко масштабувати систему в моменти максимальних навантажень.

Рівень відображення функціонує як традиційний веб-інтерфейс, створений за допомогою технологій HTML, Bootstrap та Jinja2. Користувачі надсилають стандартні HTTP-запити і взаємодіють із платформою через звичайний браузер. Рівень бізнес-логіки базується на інструментах Flask і розбитий на чотири автономні компоненти Blueprint, а саме auth, patients, appointments та analytics. Кожен із цих модулів відповідає за свій спектр завдань та комунікує з базою даних за допомогою ORM-моделей. Рівень збереження даних організовано в СУБД PostgreSQL, яка містить десять таблиць, а зв'язок із кодом налаштовано через бібліотеку Flask-SQLAlchemy. Схематичне зображення архітектури програмного забезпечення наведено на рисунку 3.5. Конфігурацію каталогів створеного проєкту продемонстровано на рисунку 3.6.

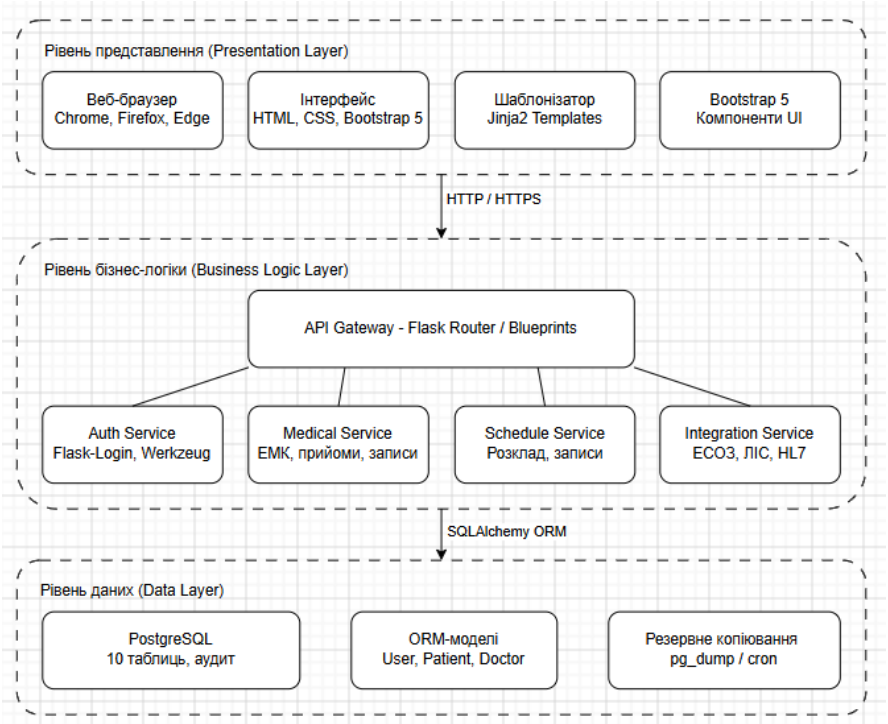


Рисунок 3.5 – Архітектура програмного забезпечення системи ЕМК

Джерело: власна розробка

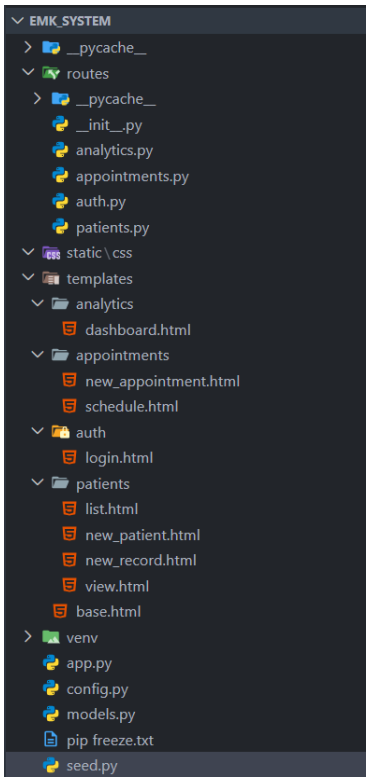


Рисунок 3.6 – Структура каталогів проєкту системи ЕМК

Джерело: власна розробка

Головним скриптом для запуску додатка є файл app.py, який контролює старт Flask-застосунку, відкриває підключення до бази даних, активує розширення Flask-Login та реєструє всі Blueprint-модулі. Конфігураційний файл config.py зберігає параметри з'єднання з PostgreSQL та секретний ключ для захисту сесій. Файл models.py містить описи всіх ORM-моделей, які представлені у коді як класи Python. Нижче наведено фрагмент файлу app.py, що демонструє реєстрацію модулів і налаштування автентифікації:

```
from routes.auth import auth
from routes.analytics import analytics
from routes.appointments import appointments
from routes.patients import patients
from flask import Flask, redirect, url_for
from flask_login import LoginManager
from config import Config
from models import db, User
```

```
app = Flask(__name__)
app.config.from_object(Config)

db.init_app(app)

login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = 'auth.login'
login_manager.login_message = 'Будь ласка, увійдіть в систему'

@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))

@app.route('/')
def index():
    return redirect(url_for('auth.login'))

app.register_blueprint(auth, url_prefix='/auth')
app.register_blueprint(patients, url_prefix='/patients')
app.register_blueprint(appointments, url_prefix='/appointments')
app.register_blueprint(analytics, url_prefix='/analytics')

if __name__ == '__main__':
    app.run(debug=True)
```

Джерело: власна розробка

За допомогою декоратора login_manager.user_loader налаштовується спеціальна функція, яка вилучає об’єкт користувача з бази даних за його ідентифікатором під час кожного звернення. Параметр login_view фіксує адресу сторінки для входу, куди система Flask-Login буде автоматично направляти неавторизованих гостей при спробі відкрити захищені сторінки ЕМК.

Кругообіг даних між рівнями побудовано за таким принципом. Веб-браузер генерує HTTP-запит до Flask, після чого відповідний Blueprint перехоплює його та звертається до моделей SQLAlchemy для отримання чи оновлення записів у PostgreSQL. На основі цього результат сервер формує фінальну сторінку за допомогою шаблонізатора Jinja2 і відправляє готовий HTML-код назад у браузер користувача.

3.3.3 Структура функціональних модулів системи

Програмний комплекс системи ЕМК містить чотири функціональні модулі, які спроєктовані як відокремлені блоки Blueprint у межах Flask. Кожен підрозділ відповідає за конкретну предметну область додатка і здійснює обмін даними через загальні моделі ORM та єдину базу даних.

Блок автентифікації (auth) керує процесами авторизації користувачів та безпечного завершення робочих сесій. Під час авторизації лікар або персонал реєстратури вносить свої ідентифікаційні дані, після чого модуль аналізує їх за допомогою Flask-Login, здійснює кодування пароля та порівнює результат із записом у таблиці users. Якщо перевірка успішна, запускається користувацька сесія з урахуванням прав доступу конкретної ролі. Рольова модель визначає перелік доступних сторінок та операцій. Автоматичне блокування активується після п’яти невдалих спроб авторизації поспіль. Нижче наведено кодовий фрагмент обробника входу:

```
@auth.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        login = request.form.get('login')
        password = request.form.get('password')
        user = User.query.filter_by(login=login).first()
```

```
            if user and check_password_hash(user.password_hash,
password):
                login_user(user)
                return redirect(url_for('patients.list_patients'))
            flash('Невірний логін або пароль')
        return render_template('auth/login.html')
```

Джерело: власна розробка

Модуль ведення пацієнтів (patients) містить основні інструменти для обробки електронних медичних карток. Його функціонал дозволяє відображати загальний список людей із можливістю текстового пошуку за ім’ям або прізвищем, відкривати повну історію хвороб із хронологічним списком візитів, фіксувати поточні життєві показники та реєструвати нових осіб у базі. Під час відкриття папки пацієнта система паралельно вилучає всі записи у зворотному часовому порядку, а також відображає останні збережені параметри здоров’я. Нижче наведено приклад обробника для відкриття медичної картки:

```
    @patients.route('/<int:patient_id>')
    @login_required
    def view_patient(patient_id):
        patient = Patient.query.get_or_404(patient_id)
                                        records =
MedicalRecord.query.filter_by(patient_id=patient_id).order_by(
            MedicalRecord.visit_date.desc()).all()
                                         vitals =
VitalSigns.query.filter_by(patient_id=patient_id).order_by(
            VitalSigns.measured_at.desc()).first()
                  return render_template('patients/view.html',
patient=patient, records=records, vitals=vitals)
```

Джерело: власна розробка

Захист цього маршруту забезпечується декоратором login_required, який обмежує доступ для неавторизованих користувачів. Спеціальний метод get_or_404 знаходить потрібний рядок або повертає помилку з кодом 404, якщо запитуваного пацієнта немає у базі даних.

Модуль ведення розкладу (appointments) координує роботу з графіком прийомів на вибраний день та створює нові записи. Під час відображення календаря програма відсікає зайві елементи за вказаною датою та сортує інформацію за часом, що дає змогу персоналу реєстратури бачити точний графік роботи лікарів. Під час реєстрації нового візиту додаток зчитує з форми ідентифікатори пацієнта та спеціаліста, фіксує обраний час і записує новий рядок у таблицю appointments.

Аналітичний блок (analytics) відповідає за підготовку зведених статистичних звітів про роботу медичної установи. Він підраховує сумарну кількість зареєстрованих пацієнтів, створених записів та працюючих лікарів, а також визначає найбільш завантажених спеціалістів за допомогою агрегуючих запитів. Нижче наведено приклад формування головного дашборду:

```
    @analytics.route('/')
    @login_required
    def dashboard():
        total_patients = Patient.query.count()
        total_appointments = Appointment.query.count()
                                        completed                    =
Appointment.query.filter_by(status='completed').count()
        doctors_count = Doctor.query.count()
        top_doctors = db.session.query(
            Doctor.last_name,
            Doctor.specialization,
            func.count(Appointment.appointment_id).label('total')
                  ).join(Appointment,      Doctor.doctor_id      ==
Appointment.doctor_id)\
                  .group_by(Doctor.doctor_id,      Doctor.last_name,
Doctor.specialization)\
```

```
        .order_by(func.count(Appointment.appointment_id).desc())\
        .limit(5).all()
    return render_template('analytics/dashboard.html',
                           total_patients=total_patients,
                           total_appointments=total_appointments,
                           completed=completed,
                           doctors_count=doctors_count,
                           top_doctors=top_doctors)
```

Джерело: власна розробка

Взаємодія між усіма компонентами побудована на основі спільних моделей ORM, які прописані у файлі models.py. До прикладу, блок patients оперує сутностями Patients, MedicalRecord та VitalSigns, блок розкладу взаємодіє з Appointment, Patient та Doctor, а аналітичний модуль збирає інформацію з кількох таблиць одночасно. Цей підхід створює єдине джерело актуальної інформації та унеможливлює розсинхронізацію даних між окремими компонентами системи. Графічне представлення діаграми компонентів наведено на рисунку 3.7.

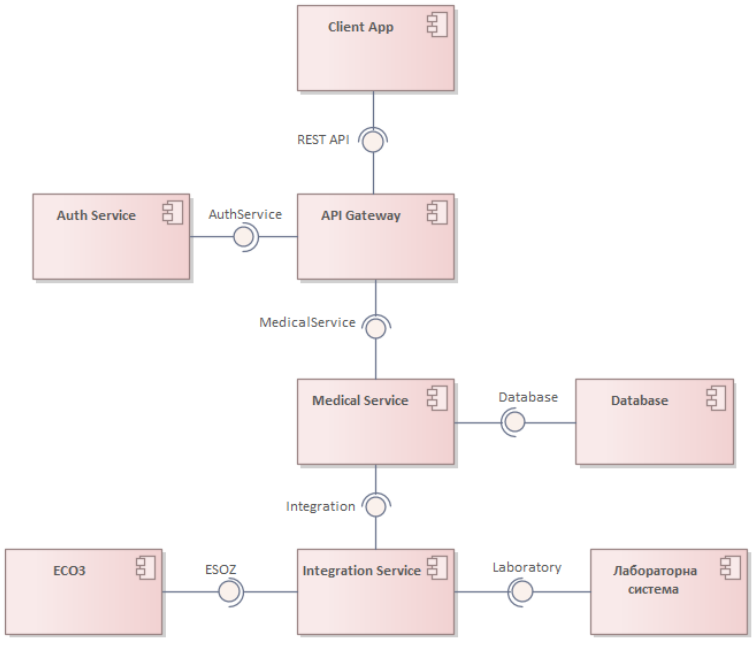


Рисунок 3.7 – Діаграма компонентів системи ЕМК

Джерело: власна розробка в Enterprise Architect [18]

3.4 Результати реалізації інформаційної системи

3.4.1 Реалізація функціональних модулів системи

У результаті розробки реалізовано чотири основні функціональні модулі системи ЕМК: модуль автентифікації, модуль управління пацієнтами, модуль розкладу прийомів та аналітичний модуль. Для тестування використано набір даних із п’яти пацієнтів, двох лікарів та п’яти записів на прийом, сформований за допомогою скрипту seed.py.

Модуль автентифікації забезпечує захищений вхід до системи з перевіркою облікових даних. Сторінка містить поля для введення логіна і пароля, а також кнопку підтвердження. Після успішної автентифікації система перенаправляє користувача до списку пацієнтів відповідно до його ролі. У лівій частині екрана розміщено навігаційну панель з розділами Пацієнт, Розклад та Аналітика, яка залишається доступною на всіх сторінках системи (рис. 3.8).

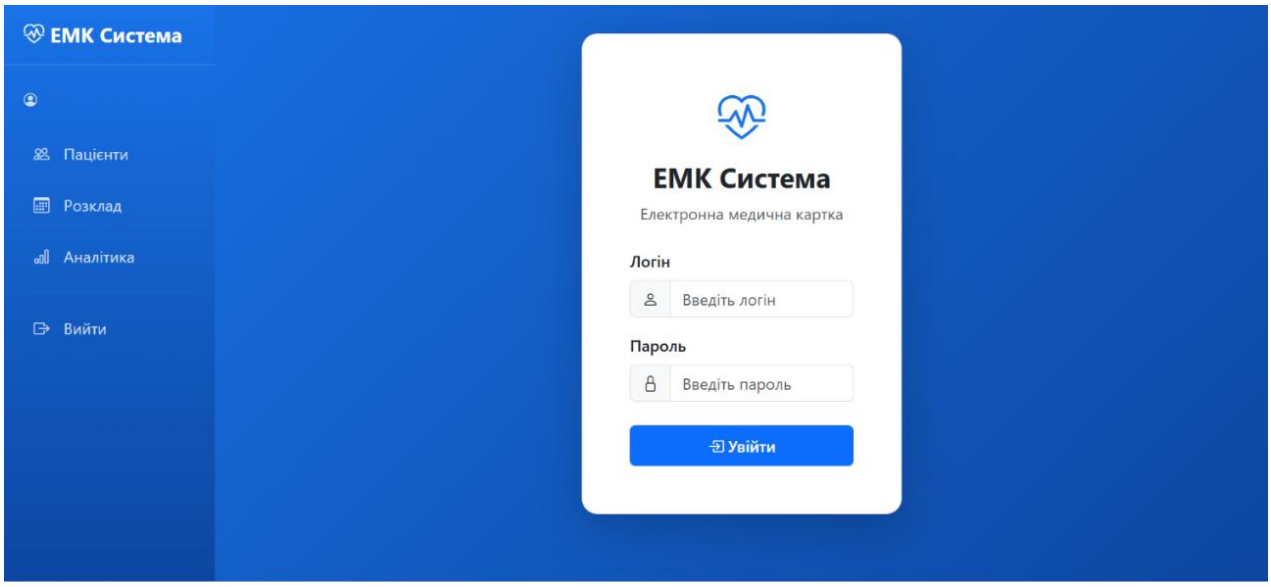


Рисунок 3.8 – Сторінка входу до системи ЕМК

Джерело: власна розробка

Модуль управління пацієнтами реалізує перегляд реєстру пацієнтів, пошук за прізвищем або ім’ям, відображення детальної картки пацієнта та створення нових медичних записів. На головній сторінці відображається таблиця з п’ятьма тестовими хворими. Для кожного з них вказано ПІБ, дату народження, контактний телефон, наявність алергій та статус декларації. Алергії виділено червоною міткою. Статус декларації позначається зеленим кольором, якщо вона «Активна» або сірим, якщо «Відсутня» (рис. 3.9).

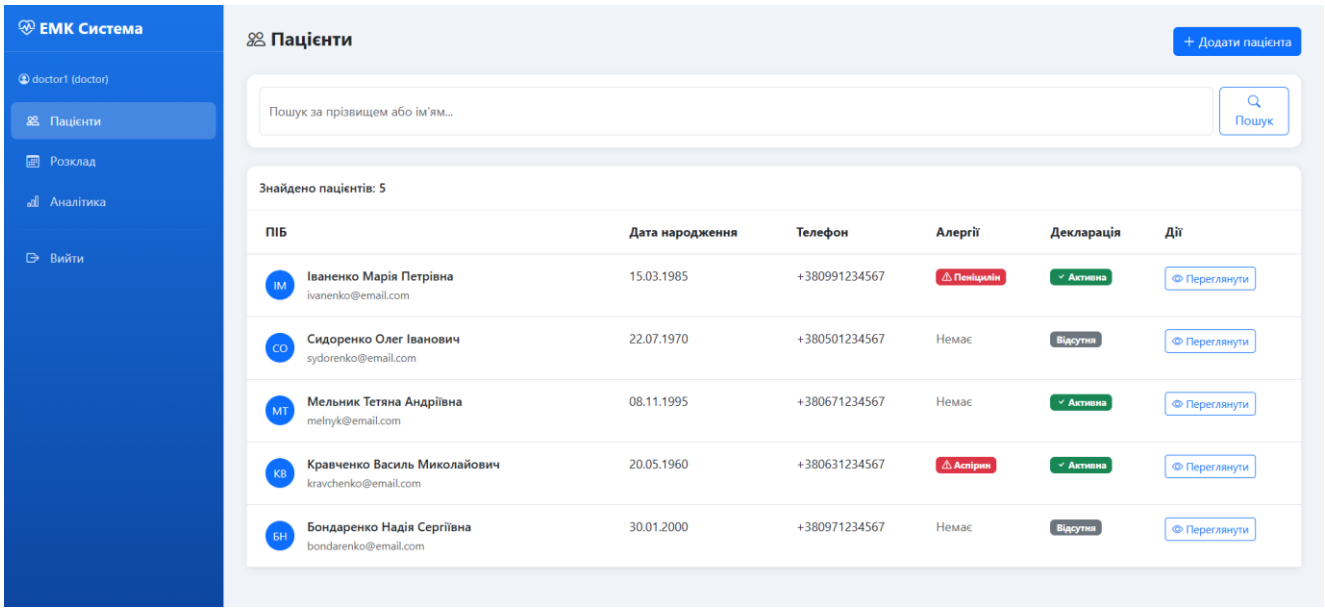


Рисунок 3.9 – Список пацієнтів системи ЕМК

Джерело: власна розробка

При відкритті картки пацієнта відображаються дві панелі. Ліва містить особисті дані: дату народження, телефон, email, адресу, статус декларації та алергії. Права панель показує останні зафіксовані життєві показники з відміткою часу вимірювання. Нижче розміщено хронологічну історію медичних записів (рис. 3.10).

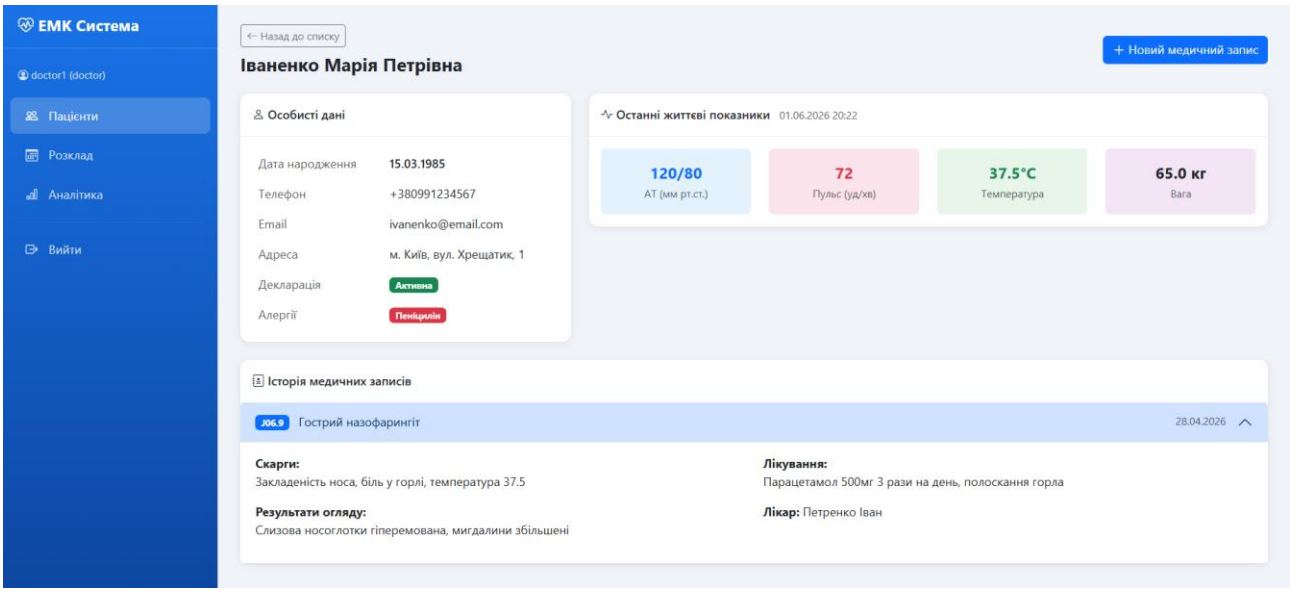


Рисунок 3.10 – Картка пацієнта з медичними записами та життєвими показниками

Джерело: власна розробка

Форма створення нового медичного запису містить поля для вибору лікаря зі списку, введення коду діагнозу за МКХ-10, опису діагнозу, скарг пацієнта, результатів огляду та призначеного лікування. Всі вони мають підказки у вигляді прикладів заповнення. Ті, які позначені зірочкою є обов'язковими. Після натискання кнопки «Зберегти запис» дані записуються до бази і стають доступними в картці пацієнта (рис. 3.11).

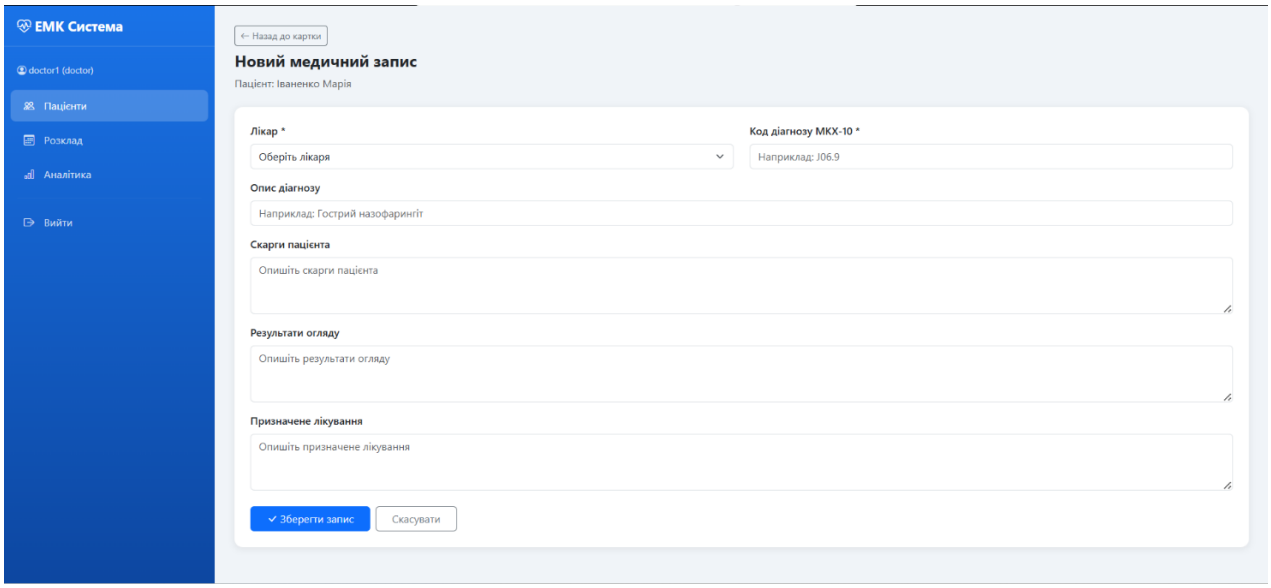


Рисунок 3.11 – Форма створення нового медичного запису

Джерело: власна розробка

Модуль розкладу прийомів відображає записи на обраний день у табличному вигляді з сортуванням часу. Для дати 05.05.2026 система показує три записи. Також відображається час, ім’я пацієнта з посиланням на його картку, лікар, спеціальність та статус. Всі записи мають статус «Заплановано» (рис. 3.12).

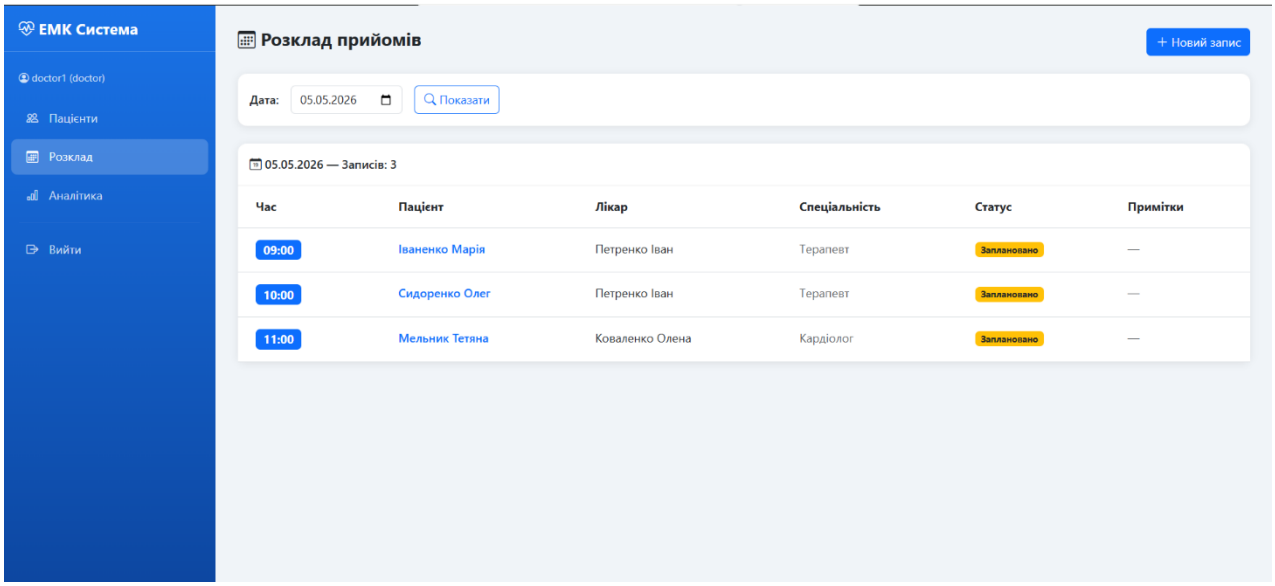


Рисунок 3.12 – Розклад прийомів системи ЕМК

Джерело: власна розробка

Аналітичний модуль формує зведену статистику роботи закладу. У верхній частині дашборду розміщено чотири інформаційні картки: всього пацієнтів, всього прийомів, завершених прийомів та лікарів у системі. Нижче відображається рейтинг лікарів за кількість прийомів, а у правій частині – загальну статистику (рис. 3.13).

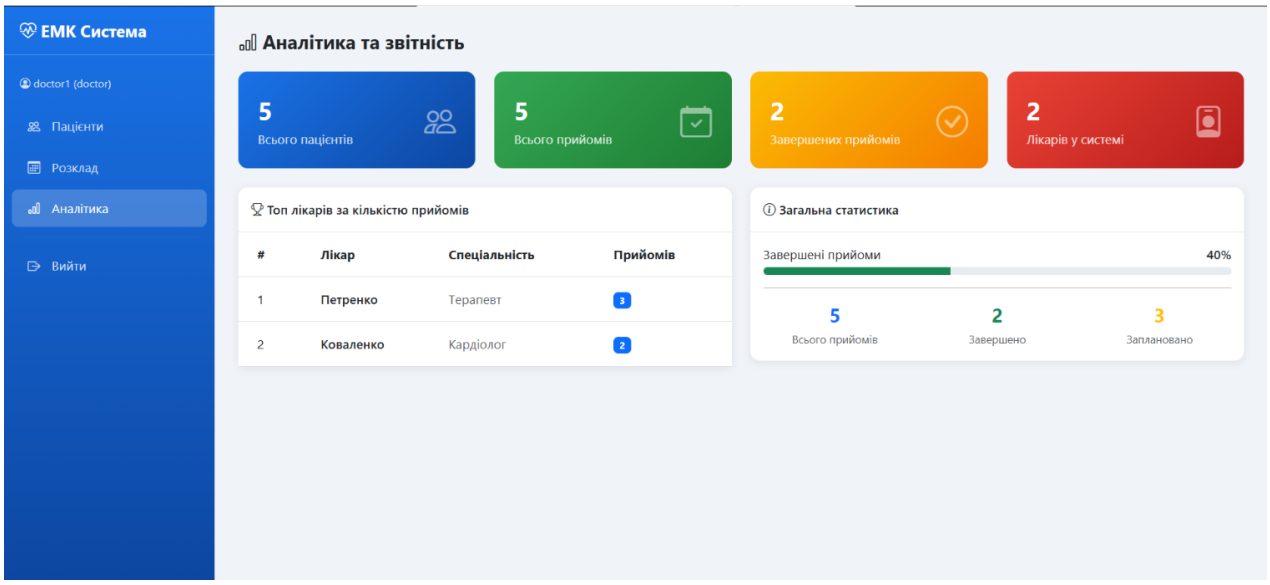


Рисунок 3.13 – Аналітичний дашборд системи ЕМК

Джерело: власна розробка

Практична цінність реалізованого рішення полягає в автоматизації ключових процесів медичного центру. Система замінює паперовий документообіг електронним веб-інтерфейсом, забезпечує централізоване зберігання медичних даних та надає керівництву інструменти для аналізу роботи закладу. Реалізований прототип охоплює основний функціонал і може бути розширений шляхом підключення модулів інтеграції з ЕСОЗ та лабораторною системою.

3.4.2 Тестування та оцінювання результатів роботи системи

Тестування системи ЕМК проводилось шляхом функціональної перевірки основних сценаріїв роботи на тестовому наборі даних. Мета – підтвердити, що реалізований функціонал відповідає визначеним вимогам, і дані обробляються коректно в кожному модулі.

Тестовий набір сформовано скриптом seed.py і містить п’ять пацієнтів, двох лікарів, п’ять записів на прийом, три медичні записи та три набори життєвих показників. Перевірка виконувалась на локальному сервері Flask з підключенням

до PostgreSQL. Для кожного сценарію визначено вхідні дані, очікуваний і фактичний результат (табл. 3.8).

Таблиця 3.8 – Результати тестування функціональних модулів системи ЕМК

Модуль	**Сценарій**	**Вхідні дані**	**Очікуваний результат**	**Фактичний результат**	**Статус**
Автентифікація	Вхід з коректним и даними	Login: doctor1; Password: password123	Перехід на список пацієнтів	Перехід виконано	Виконано
Автентифікація	Вхід з невірним паролем	Login, doctor1, password, wrong	Повідомлення про помилку	«Невірний логін або пароль»	Виконано
Пацієнти	Відображе ння реєстру	Авторизовани й користувач	Таблиця з пацієнтами	Записи відображено	Виконано
Пацієнти	Пошук за прізвищем	Запит «Іваненко»	Відфільтрований список	Пацієнта знайдено	Виконано
Пацієнти	Перегляд картки	patient_id=1	Дані, показники, записи	Картка відображена	Виконано
Пацієнти	Створення медичного запису	Лікар, діагноз, скарги	Запис збережено в БД	Запис з’явився в картці	Виконано
Розклад	Перегляд на дату	Дата 05.05.2026	Три записи за часом	Записи відображено	Виконано
Розклад	Новий запис	Пацієнт, лікар, час	Запис збережено	З’явився в розкладі	Виконано
Аналітика	Дашборд	Авторизовани й користувач	Зведена статистика	Дані відображені коректно	Виконано
Аналітика	Рейтинг лікарів	Дані прийомів	Сортування за кількістю	Петренко - 3, Коваленко - 2	Виконано

Джерело: власна розробка

Всі десять сценаріїв завершились успішно. Система правильно обробляє вхідні дані, зберігає їх у базі та відображає результати відповідно до очікувань.

З обмежень реалізованого прототипу варто зазначити відсутність прямої інтеграції з ЕСОЗ і лабораторною системою – ці компоненти закладені в архітектурі, але потребують підключення до реальних API. Серед напрямків

подальшого розвитку є реалізація модуля електронних рецептів, особистий кабінет пацієнта та розширена фінансова звітність.

ВИСНОВКИ

У ході виконання бакалаврського проєкту повністю досягнуто поставлену мету та вирішено всі технічні завдання. Аналіз діяльності медичного центру виявив чотири основні проблеми: одночасне ведення паперових звітів та цифрових документів, відсутність зв’язку з платформою ЕСОЗ для генерації рецептів, передача аналізів лише на бланках і брак аналітики для керівництва. Огляд п’яти ринкових аналогів підтвердив, що готові програми вимагають великих витрат на адаптацію. Через це виникла потреба створити власну систему.

На основі проведеного аналізу сформовано три бізнес-вимоги, сім функціональних і шість нефункціональних вимог. Для розробки софту обрано мову Python та веб-фреймворк Flask. База даних побудована на PostgreSQL. Цей стек технологій гарантує надійність та безпеку збереження медичних даних.

Під час проєктування засобами UML побудовано діаграму прецедентів з дев’ятьма варіантами використання, шість діаграм послідовності та діаграму класів за шаблоном «Boundary-Control-Entity». Матриця трасування підтвердила повне покриття вимог архітектурою без зайвих елементів. Також створено математичні алгоритми для розрахунку завантаженості персоналу і пошуку конфліктів у розкладі лікаря.

Базу даних реалізовано у СУБД PostgreSQL із десяти таблиць у третій нормальній формі для захисту від дублювання інформації. Окремо додано таблицю журналу аудиту для фіксації дій користувачів. Мережева інфраструктура використовує топологію зірки з основним каналом 100 Мбіт/с, резервним зв’язком 4G/LTE та шифруванням трафіку через TLS.

Прототип програми містить чотири Blueprint-модулі: автентифікація з рольовим доступом, робота з електронними медичними картками, управління розкладом та аналітична звітність. Інтерфейс є повністю україномовним, адаптивним і відкривається в браузері без встановлення програм. Під час перевірки

системи успішно пройдено десять тест-кейсів, які підтвердили стабільність запису даних.

Практичне значення проєкту полягає в можливості розгортання створеного додатка в реальній клініці. Розроблена програма оптимізує щоденні завдання лікарів або адміністраторів. Вона допомагає повністю відмовитися від паперу. Гнучка архітектура залишає систему технічно готовою до майбутнього підключення модулів ЕСОЗ та автоматичного імпорту результатів з лабораторій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Деякі питання електронної системи охорони здоров’я : Постанова Кабінету Міністрів України від 25 квіт. 2018 р. № 411 : станом на 14 берез. 2024 р. URL: https://zakon.rada.gov.ua/laws/show/411-2018-%D0%BF (дата звернення: 11.05.2026).

2. Що входить до обов'язків лікаря первинної ланки : офіційний сайт / Міністерство охорони здоров'я України. URL: https://moz.gov.ua/uk/scho-vhodit-do-obovjazkiv-likarja-pervinnoi-lanki (дата звернення: 03.06.2026).

3. Електронна медична карта: як впроваджуватимуть : Головний медичний портал / Медична платформа. URL: https://medplatforma.com.ua/article/1769-elektronna-medichna-karta-yak-vprovadjuvatimut (дата звернення: 03.06.2026).

4. Що таке KPI: ключові показники ефективності : блог / Hurma System. URL: https://hurma.work/blog/shho-take-kpi-klyuchovi-pokazniki-efektivnosti/ (дата звернення: 03.06.2026).

5. Бази даних в інформаційних системах: Електронна медична картка : інформаційний ресурс / StudFile. URL: https://studfile.net/preview/10714248/page:17/ (дата звернення: 03.06.2026).

6. Що таке BPMN-діаграма і навіщо вона потрібна з прикладами : блог / IAmPM. URL: https://iampm.club/ua/blog/shho-take-bpmn-diagrama-i-navishho-vona-potribna-z-prikladami-2/ (дата звернення: 03.06.2026).

7. Generating documentation: Bizagi Studio Help Platform / Bizagi. URL: https://help.bizagi.com/platform/en/index.html?generating_documentation.htm (дата звернення: 03.06.2026).

8. Де доступні ліки : офіційний сайт / Міністерство охорони здоров'я України. URL: https://moz.gov.ua/uk/dev-dostupni-liki (дата звернення: 03.06.2026).

9. Helsi.me - сервіс запису до лікарів та збереження медичних даних: офіційний сайт / Медична інформаційна система Helsi. URL: https://helsi.me/ (дата звернення: 03.06.2026).

10. Doctor Eleks - медична інформаційна система для клінік та лабораторій: офіційний сайт. URL: https://doctor.eleks.com/ (дата звернення: 03.06.2026).

11. eClinicalWorks: Cloud EHR & Practice Management Solutions: official website. URL: https://www.eclinicalworks.com/ (дата звернення: 03.06.2026).

12. athenahealth: Electronic Health Records, Practice Management & Medical Billing: official website. URL: https://www.athenahealth.com/ (дата звернення: 03.06.2026).

13. Veradigm: Healthcare Data Systems & EHR Solutions : official website. URL: https://veradigm.com/ (дата звернення: 03.06.2026).

14. Client-Server Architecture: What it is and how it works : техн. стаття / QA Start Up ; QATestLab Training Center. URL: https://training.qatestlab.com/blog/technical-articles/client-server-architecture/ (дата звернення: 03.06.2026).

15. Що таке Bootstrap і для чого він потрібен: огляд фреймворку : блог / Навчальний IT-центр Foxminded. URL: https://foxminded.ua/bootstrap/ (дата звернення: 03.06.2026).

16. Lightweight and flexible Flask: fundamentals for software web development: блог / PNN Soft. URL: https://pnn.com.ua/ua/blog/detail/lightweight-and-flexible-flask-fundamentals-for-software-web-development (дата звернення: 04.06.2026).

17. PostgreSQL Documentation: official website / The PostgreSQL Global Development Group. URL: https://www.postgresql.org/docs/ (дата звернення: 04.06.2026).

18. Documenting an Enterprise Architecture: Enterprise Architect User Guide / Sparx

Systems.URL:https://sparxsystems.com/enterprise_architect_user_guide/17.1/guide_books/ea_documenting_an_enterprise_architecture.html (дата звернення: 04.06.2026).

19. Що таке документ бізнес-вимог (Business Requirements Document): вебсайт / Спільнота «Business Analysis in Ukraine». URL:https://www.ba.in.ua/2023/11/24/shho-take-dokument-biznes-vymog-business-requirements-document/ (дата звернення: 04.06.2026).

20. Функціональні вимоги до програмного забезпечення : вебсайт / TQM systems. URL: https://tqm.com.ua/ua/kompleksna-avtomatyzatsiia-biznesu/funkcionalni-vimogi (дата звернення: 04.06.2026).

21. Non-Functional Requirements: Examples, Types, Approaches : техн. стаття / QA Start Up; QATestLab Training Center. URL:https://training.qatestlab.com/blog/technical-articles/non-functional-requirements-examples-types-approaches/(дата звернення: 04.06.2026).

22. Що таке протокол TLS, як він працює та від чого захищає: блог / Хостинг-провайдер Cityhost. URL: https://cityhost.ua/uk/blog/scho-take-protokol-tls-yak-vin-pracyu-ta-vid-chogo-zahischa.html (дата звернення: 04.06.2026).

23. Стандарти: офіційний сайт/ HL7 Україна. URL:https://hl7.org.ua/standards/ (дата звернення: 04.06.2026).

24. МКХ-10-АМ : Електронний медичний довідник / Doctor Guides. URL: https://doctorguides.com.ua/mkxam (дата звернення: 04.06.2026).

25. Що таке референтні значення аналізів? : медична стаття / Інтермедика. URL: https://intermedica.com.ua/ua/labarticles/tpost/my1zbflj51-scho-take-referensn-znachennya-analzv (дата звернення: 04.06.2026).

26. PBKDF2 : матеріали з Вікіпедії - вільної енциклопедії. URL: https://uk.wikipedia.org/wiki/PBKDF2 (дата звернення: 04.06.2026).

27. Що таке стандарт електронної пошти RFC 5322 : техн. стаття / Вебпортал «Ma.Zori». URL: https://ma.zori.cx.ua/ukraincyam/shho-take-standart-elektronnoi-poshti-rfc-5322.html (дата звернення: 04.06.2026).

28. Декомпозиція вимог та моделювання систем : електронний навчальний посібник / Платформа «StudFile». URL: https://studfile.net/preview/7675922/page:10/ (дата звернення: 04.06.2026).

29. Повідомлення від ЕЗОС: що потрібно знати пацієнтам : роз'яснення / Міністерство охорони здоров'я України. URL: https://moz.gov.ua/uk/povidomlennya-vid-esoz-sho-potribno-znati-paciyentam (дата звернення: 04.06.2026).

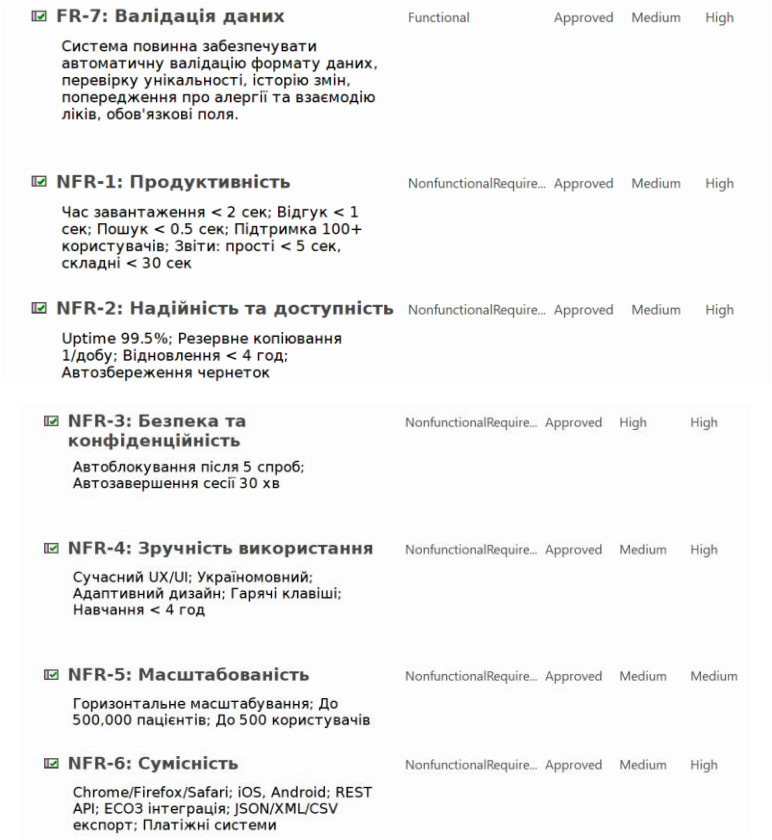
30. The normalization of relationships during the DB designing: навчальний посібник / Вебпортал «Relational databases». URL: https://rdb.dp.ua/uk/chapter_03 (дата звернення: 04.06.2026).

ДОДАТКИ

Додаток А

Специфікація вимог до системи

BR-1: Відповідність законодавству Система повинна забезпечити повну відповідність вимогам законодавства України у сфері електронної охорони здоров'я, зокрема забезпечити інтеграцію з державною ЕСОЗ.	BusinessRequirement	Approved	High	High
BR-2: Підвищення продуктивності праці Система повинна забезпечити автоматизацію операцій та усунення дублювання роботи.	BusinessRequirement	Approved	Medium	High
BR-3: Покращення якості обслуговування Система повинна забезпечити швидкий доступ медичного персоналу до повної медичної історії пацієнта.	BusinessRequirement	Approved	Medium	High
FR-1: Повноцінна ЕМК Система повинна забезпечувати створення та ведення повноцінної структурованої ЕМК з детальними записами лікарів.	Functional	Approved	High	High
FR-2: Інтеграція з ЕСОЗ Система повинна забезпечувати інтеграцію з ЕСОЗ для е-рецептів, перевірки декларацій, інтеграції з Дією, синхронізації з Кабінетом пацієнта.	Functional	Approved	High	High
FR-3: Інтеграція з лабораторією Система повинна забезпечувати інтеграцію з лабораторною системою: електронні направлення, автопередача результатів, сповіщення лікаря, позначення критичних відхилень.	Functional	Approved	High	High
FR-4: Особистий кабінет пацієнта Web та мобільний додаток для пацієнтів: онлайн-запис, перегляд історії хвороби, результатів аналізів, е-рецептів, нагадування, онлайн-оплата.	Functional	Approved	High	High
FR-5: Управління розкладом Система повинна забезпечувати управління розкладом лікарів: візуалізація завантаженості, різна тривалість прийомів, масове планування, автооптимізація.	Functional	Approved	Medium	High
FR-6: Аналітика та звітність Дашборд для керівництва в реальному часі, аналіз завантаженості лікарів, структури захворюваності, фінансові звіти, експорт даних.	Functional	Approved	Medium	Medium



Додаток А.1 – Специфікація вимог до системи

Джерело: власна розробка в Enterprise Architect

Додаток Б

Ескізи форм вихідних документів

Фінансовий звіт

(OUT_FIN_001)

Медичний заклад:_____

Період:_____ (ДД.ММ.РРРР-ДД.ММ.РРРР)

1. Загальна інформація

Кількість прийомів:_____

Кількість пацієнтів:_____

Кількість платних послуг: _____

2. Доходи

Стаття доходів	Сума, грн
Процедури	
Платні прийоми	
Лабораторні дослідження	

3. Витрати

Стаття витрат	Сума, грн
Заробітна плата персоналу	
Обладнання	
Медикаменти	

4. Фінансовий результат

Показник	Значення
Прибуток/Збиток	
Рентабельність (%)	

Дата: _____ (ДД.ММ.РРРР)

Додаток Б.1 – Фінансовий звіт

Джерело: власна розробка

Графік роботи лікаря

Медичний заклад:_____

Період:_____ (ДД.ММ.РРРР-ДД.ММ.РРРР)

1. Основна інформація

Лікар: _____(DOC-01-0023)

Спеціальність: _____

Кабінет: _____

Контактний номер:_____

2. Щотижневий графік роботи

День тижня	**Час роботи**	**Перерва**
Понеділок		
Вівторок		
…		
Неділя		

3. Додаткова інформація

Відпустка/Відрядження: _____

Особливі умови праці: _____

Дата: _____ (ДД.ММ.РРРР)

Додаток Б.2 – Графік роботи лікаря

Джерело: власна розробка

Додаток В

Ескізи форм вхідних документів

Картка пацієнта

Основні дані

ID пацієнта: _____ (PAT-00012457)
ПІБ:_____
Дата народження:_____
Стать:_____
Контактний телефон:_____
Адреса проживання:_____

Дані декларації

Статус декларації:_____
ID лікаря: _____ (DOC-01-0023)
Дата підписання:_____ (ДД.ММ.РРРР)

Медичні записи

Останній візит: _____ (ДД.ММ.РРРР)
Попередній діагноз:_____ (J00)
Призначене лікування: _____

Рекомендації лікаря:_____
_____
_____
_____

Додаток В.1 – Картка пацієнта

Джерело: власна розробка

Результати досліджень

1. Загальні відомості

Пацієнт: _____ (PAT-00012457)
Дата народження:_____ (ДД.ММ.РРРР)
Стать:_____
Лаб. № замовлення: _____ (LAB-01-20251202-0045)
Дата замовлення: _____ (ДД.ММ.РРРР)

2. Основні показники

Тип дослідження:_____

Показник	Значення	Одиниці	Референтні межі
Лейкоцити	6.60	г/л	3.9-10.2
Гемоглобін	125	г/л	120-156
Тромбоцити	329	г/л	166-389
Лімфоцити	40.60	%	20.0-44.0

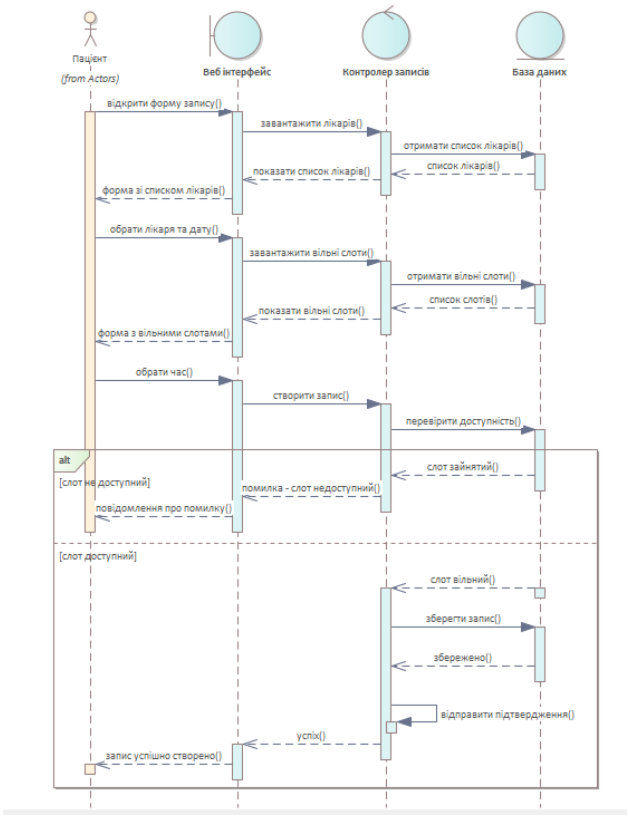
3. Коментар лабораторії:_____
_____
_____
_____

Додаток В.2 – Результати досліджень

Джерело власна розробка

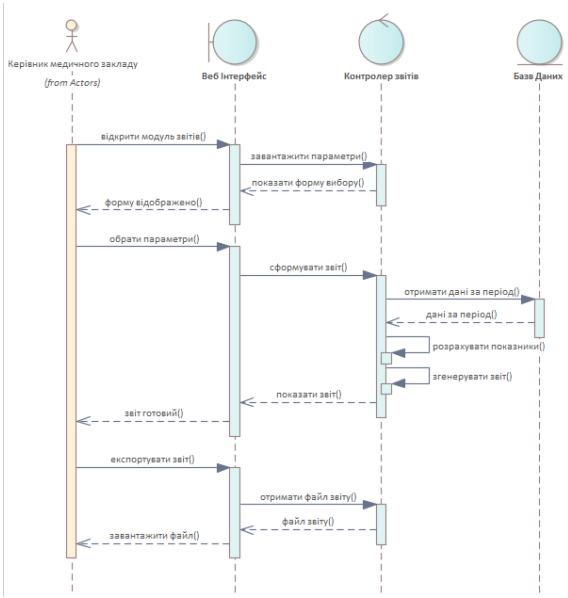
Додаток Г

Діаграми послідовності



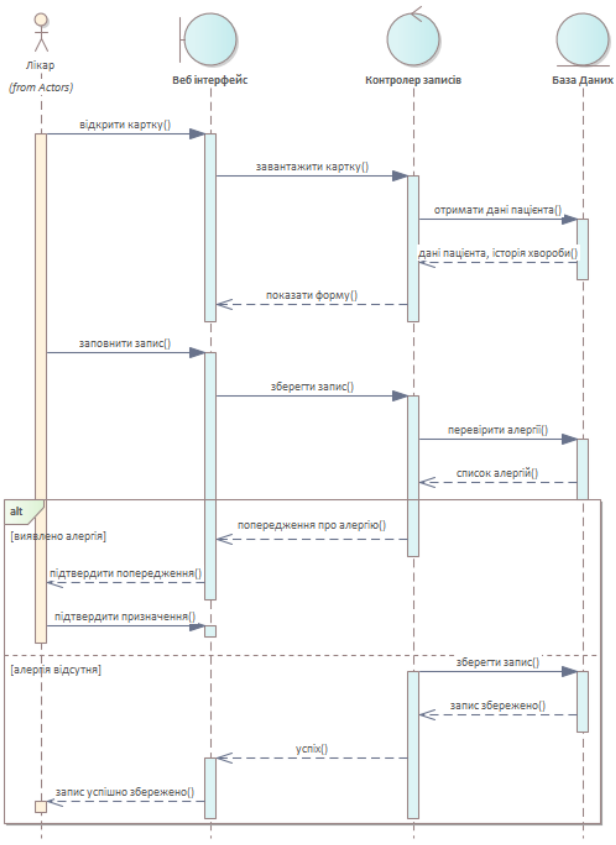
Додаток Г.1 – Діаграма послідовності «Запис пацієнта на прийом»

Джерело: власна розробка в Enterprise Architect



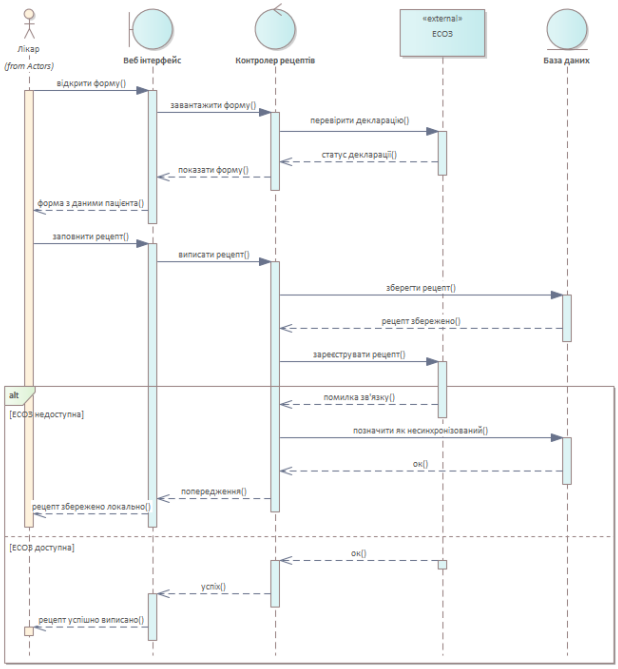
Додаток Г.2 – Діаграма послідовності «Формування звітів та аналітики»

Джерело: власна розробка в Enterprise Architect



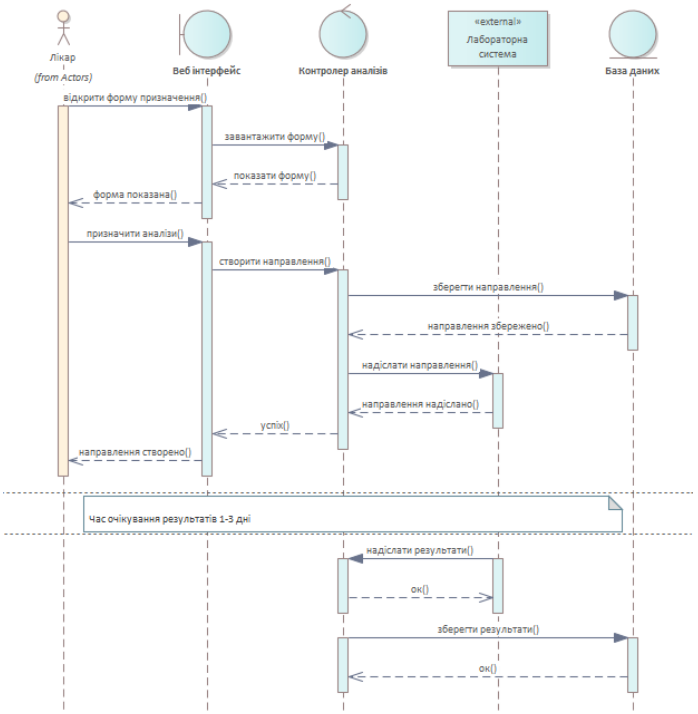
Додаток Г.3 – Діаграма послідовності «Створення медичного запису»

Джерело: власна розробка в Enterprise Architect



Додаток Г.4 – Діаграма послідовності «Виписування електронного рецепту»

Джерело: власна розробка в Enterprise Architect



Додаток Г.5 – Діаграма послідовності «Отримання результатів досліджень»

Джерело: власна розробка в Enterprise Architect

Додаток Д

Структури таблиць бази даних системи ЕМК

Таблиця Д.1 – Структура таблиці «Лікарі» (doctors)

Назва поля	Фізичне ім’я поля	Тип даних	Обмеження / Ключ
Ідентифікатор лікаря	doctor_id	SERIAL	PRIMARY KEY
Ідентифікатор користувача	user_id	INTEGER	FOREIGN KEY (users)
Ім’я	first_name	VARCHAR (100)	NOT NULL
Прізвище	last_name	VARCHAR (100)	NOT NULL
По батькові	middle_name	VARCHAR (100)	-
Спеціалізація	specialization	VARCHAR (100)	NOT NULL
Дата створення	created_at	TIMESTAMP	DEFAUL NOW ()

Джерело: власна розробка

Таблиця Д.2 – Структура таблиці «Записи на прийом» (appointments)

Назва поля	Фізичне ім’я поля	Тип даних	Обмеження / Ключ
Ідентифікатор прийому	appointment_id	SERIAL	PRIMARY KEY
Ідентифікатор пацієнта	patient_id	INTEGER	FOREIGN KEY (patients)
Ідентифікатор лікаря	doctor_id	INTEGER	FOREIGN KEY (doctors)
Дата прийому	appointment_date	DATE	NOT NULL
Час прийому	appointment_time	TIME	NOT NULL
Статус	status	VARCHAR (50)	DEFAUL ‘scheduled’
Примітки	notes	TEXT	-
Дата створення	created_at	TIMESTAMP	DEFAUL NOW ()

Джерело: власна розробка

Таблиця Д.3 – Структура таблиці «Життєві показники» (vital_signs)

Назва поля	Фізичне ім’я поля	Тип даних	Обмеження / Ключ
Ідентифікатор запису	vital_id	SERIAL	PRIMARY KEY
Ідентифікатор пацієнта	patient_id	INTEGER	FOREIGN KEY (patients)
Ідентифікатор прийому	appointment_id	INTEGER	FOREIGN KEY (appointments)
Систолічний тиск	blood_pressure_systolic	INTEGER	-
Діастолічний тиск	blood_pressure_diastolic	INTEGER	-
Пульс	Pulse	INTEGER	-
Температура	temperature	NUMERIC (4, 1)	-
Вага	weight	NUMERIC (5, 1)	-

Продовження Таблиці Д.3

Зріст	height	NUMERIC (5, 1)	-
Час вимірювання	measured_id	TIMESTAMP	DEFAUL NOW ()

Джерело: власна розробка

Таблиця Д.4 – Структура таблиці «Рецепти» (prescriptions)

Назва поля	**Фізичне ім'я поля**	**Тип даних**	**Обмеження / Ключ**
Ідентифікатор рецепту	prescription_id	SERIAL	PRIMARY KEY
Ідентифікатор запису	record_id	INTEGER	FOREIGN KEY (medical_records)
Ідентифікатор пацієнта	patient_id	INTEGER	FOREIGN KEY (patients)
Ідентифікатор лікаря	doctor_id	INTEGER	FOREIGN KEY (doctors)
Назва препарату	medication_name	VARCHAR (200)	NOT NULL
Дозування	dosage	VARCHAR (100)	NOT NULL
Частота прийому	frequency	VARCHAR (100)	-
Тривалість прийому	duration	VARCHAR (100)	-
Ідентифікатор ЕСОЗ	esoz_id	VARCHAR (50)	-
Дійсний до	valid_until	DATE	-
Дата створення	created_at	TIMESTAMP	DEFAUL NOW ()

Джерело: власна розробка

Таблиця Д.5 – Структура таблиці «Направлення на дослідження» (lab_orders)

Назва поля	**Фізичне ім'я поля**	**Тип даних**	**Обмеження / Ключ**
Ідентифікатор направлення	order_id	SERIAL	PRIMARY KEY
Ідентифікатор запису	record_id	INTEGER	FOREIGN KEY (medical_records)
Ідентифікатор пацієнта	patient_id	INTEGER	FOREIGN KEY (patients)
Ідентифікатор лікаря	doctor_id	INTEGER	FOREIGN KEY (doctors)
Назва дослідження	test_name	VARCHAR (200)	NOT NULL
Дата направлення	order_date	TIMESTAMP	DEFAUL NOW ()
Статус	status	VARCHAR (50)	DEFAUL 'pending'

Джерело: власна розробка

Таблиця Д.6 – Структура таблиці «Результати досліджень» (lab_results)

Назва поля	Фізичне ім’я поля	Тип даних	Обмеження / Ключ
Ідентифікатор результату	Result_id	SERIAL	PRIMARY KEY
Ідентифікатор направлення	order_id	INTEGER	FOREIGN KEY (lab_orders)
Назва показника	indicator_name	VARCHAR (100)	NOT NULL
Значення	value	VARCHAR (100)	-
Одиниці виміру	until	VARCHAR (50)	-
Референтний діапазон	reference_range	VARCHAR (100)	-
Відхилення від норми	is_abnormal	BOOLEAN	DEFAUL FALSE
Дата результату	result_date	TIMESTAMP	DEFAUL NOW ()

Джерело: власна розробка

Таблиця Д.7 – Структура таблиці «Журнал аудиту» (audit_log)

Назва поля	Фізичне ім’я поля	Тип даних	Обмеження / Ключ
Ідентифікатор запису	log_id	SERIAL	PRIMARY KEY
Ідентифікатор користувача	user_id	INTEGER	FOREIGN KEY (USERS)
Тип операції	operation_type	VARCHAR (20)	NOT NULL
Назва таблиці	table_name	VARCHAR (100)	NOT NULL
Ідентифікатор запису	record_id	INTEGER	-
Дата операції	operation_date	TIMESTAMP	DEFAUL NOW ()

Джерело: власна розробка

Додаток Е

Екранні форми таблиць бази даних системи ЕМК

	user_id [PK] integer	login character varying (100)	password_hash character varying (255)	role character varying (50)	created_at timestamp without time zone
1	43	doctor1	scrypt:32768:8:1$snCIDhD0rra...	doctor	2026-06-01 20:22:06.281544
2	44	admin	scrypt:32768:8:1$m50ntfxDQM...	admin	2026-06-01 20:22:06.28155
3	45	registrar	scrypt:32768:8:1$oFjwYaDZbn...	registrar	2026-06-01 20:22:06.281552
4	46	doctor2	scrypt:32768:8:1$cKeErJABc2c...	doctor	2026-06-01 20:22:06.281553

Рисунок Е.1 – Тестові дані таблиці users

Джерело: власна розробка

	patient_id [PK] integ	user_id integer	first_name character var	last_name character varyir	middle_name character varying (	date_of_birth date	phone character varying (	email character varying (100)	address character varying (200)	allergies text	declaration_status boolean	created_at timestamp without time zone
1	52	[null]	Марія	Іваненко	Петрівна	1985-03-15	+380991234567	ivanenko@email.com	м. Київ, вул. Хрещатик, 1	Пеніцилін	true	2026-06-01 20:22:06.298675
2	53	[null]	Олег	Сидоренко	Іванович	1970-07-22	+380501234567	sydorenko@email.com	м. Київ, вул. Шевченка, 15	[null]	false	2026-06-01 20:22:06.298678
3	54	[null]	Тетяна	Мельник	Андріївна	1995-11-08	+380671234567	melnyk@email.com	м. Київ, вул. Лесі Українки, 7	[null]	true	2026-06-01 20:22:06.298679
4	55	[null]	Василь	Кравченко	Миколайович	1960-05-20	+380631234567	kravchenko@email.com	м. Київ, вул. Полтавська, 3	Аспірин	true	2026-06-01 20:22:06.29868
5	56	[null]	Надія	Бондаренко	Сергіївна	2000-01-30	+380971234567	bondarenko@email.com	м. Київ, вул. Саксаганського, ...	[null]	false	2026-06-01 20:22:06.298681

Рисунок Е.2 – Тестові дані таблиці patients

Джерело: власна розробка

	record_id [PK] integer	patient_id integer	doctor_id integer	appointment_id integer	diagnosis_code character varying (	diagnosis_description text	complaints text	examination_results text	treatment text	visit_date timestamp without time zone	created_at timestamp without time zone
1	34	52	21	52	J06.9	Гострий назофарингіт	Закладеність но...	Слизова носоглотки гіпере...	Парацетамол 500м...	2026-04-28 09:00:00	2026-06-01 20:22:06.320649
2	35	55	22	55	I10	Артеріальна гіпертен...	Головний біль, за...	АТ 160/100 мм рт.ст., пульс ...	Еналаприл 10мг 2 р...	2026-04-29 10:00:00	2026-06-01 20:22:06.320652
3	36	53	21	53	K29.7	Гастрит неуточнений	Біль у животі, ну...	Живіт м'який, болючий в епі...	Омепразол 20мг вр...	2026-04-30 11:00:00	2026-06-01 20:22:06.320653

Рисунок Е.3 – Тестові дані таблиці medical_records

Джерело: власна розробка

	doctor_id [PK] integer	user_id integer	first_name character varying (100)	last_name character varying (100)	middle_name character varying (100)	specialization character varying (100)	created_at timestamp without time zone
1	21	43	Іван	Петренко	Олексійович	Терапевт	2026-06-01 20:22:06.294491
2	22	46	Олена	Коваленко	Василівна	Кардіолог	2026-06-01 20:22:06.294493

Рисунок Е.4 – Тестові дані таблиці doctors

Джерело: власна розробка

	appointment_id [PK] integer	patient_id integer	doctor_id integer	appointment_date date	appointment_time time without time zone	status character varying (50)	notes text	created_at timestamp without time zone
1	52	52	21	2026-05-05	09:00:00	scheduled	[null]	2026-06-01 20:22:06.311247
2	53	53	21	2026-05-05	10:00:00	scheduled	[null]	2026-06-01 20:22:06.31125
3	54	54	22	2026-05-05	11:00:00	scheduled	[null]	2026-06-01 20:22:06.311251
4	55	55	21	2026-05-06	09:30:00	completed	[null]	2026-06-01 20:22:06.311253
5	56	56	22	2026-05-06	14:00:00	completed	[null]	2026-06-01 20:22:06.311253

Рисунок Е.5 – Тестові дані таблиці appointments

Джерело: власна розробка

	vital_id [PK] integer	patient_id integer	appointment_id integer	blood_pressure_systolic integer	blood_pressure_diastolic integer	pulse integer	temperature numeric (4,1)	weight numeric (5,1)	height numeric (5,1)	oxygen_saturation integer	measured_at timestamp without time zone
1	27	52	52	120	80	72	37.5	65.0	168.0	98	2026-06-01 20:22:06.329646
2	28	55	55	160	100	88	36.6	85.0	175.0	97	2026-06-01 20:22:06.329648
3	29	53	53	125	82	76	36.8	78.0	180.0	99	2026-06-01 20:22:06.329649

Рисунок Е.6 – Тестові дані таблиці vital_signs

Джерело: власна розробка

Додаток Ж

Екранні форми прототипу системи електронної медичної картки в середовищі Figma

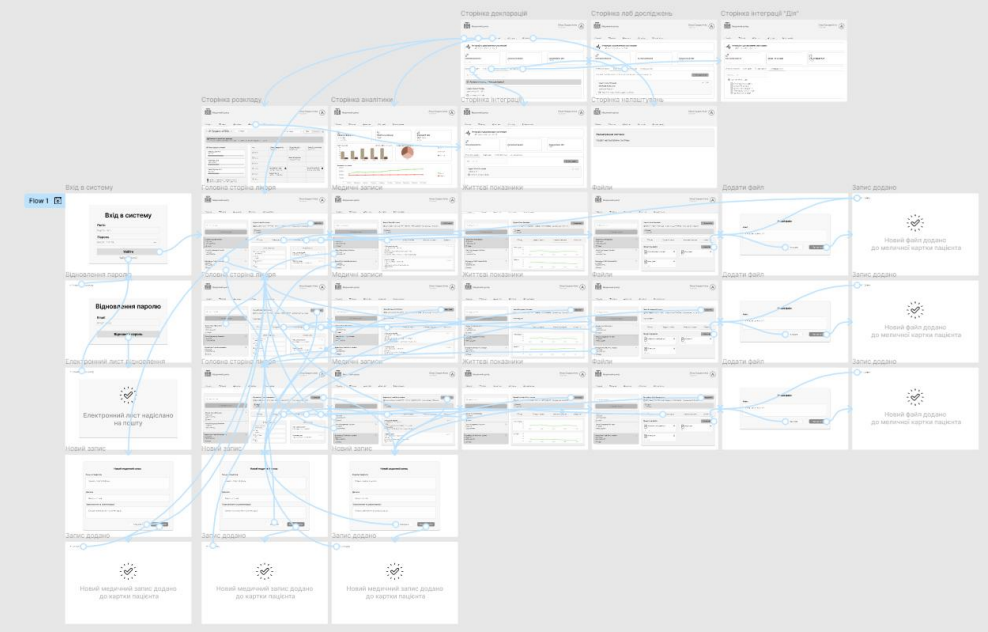


Рисунок Ж.1 – Прототип системи

Джерело: власна розробка

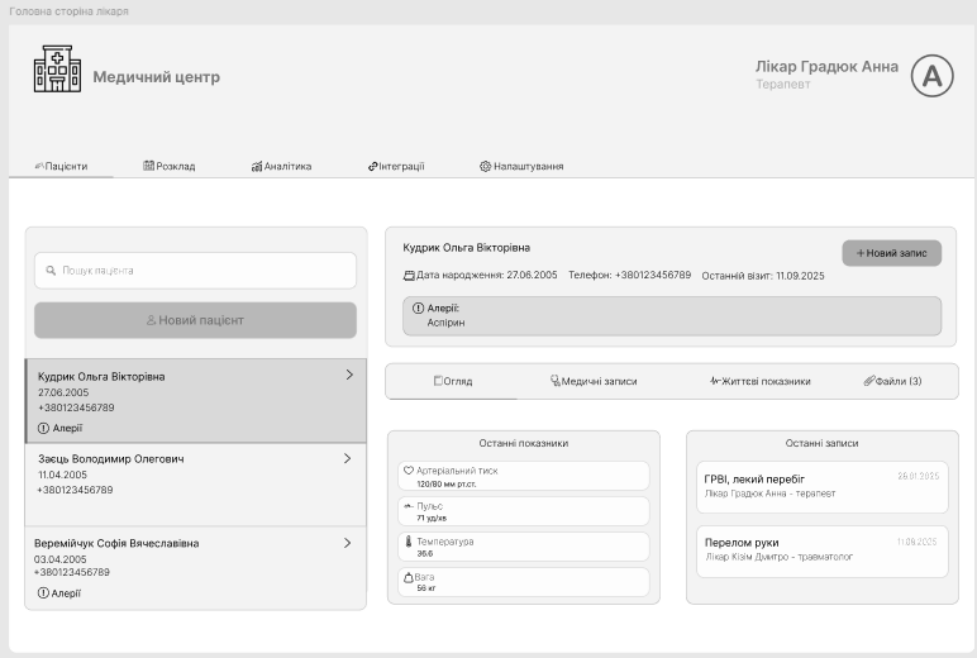


Рисунок Ж.2 – Головна сторінка лікаря

Джерело: власна розробка

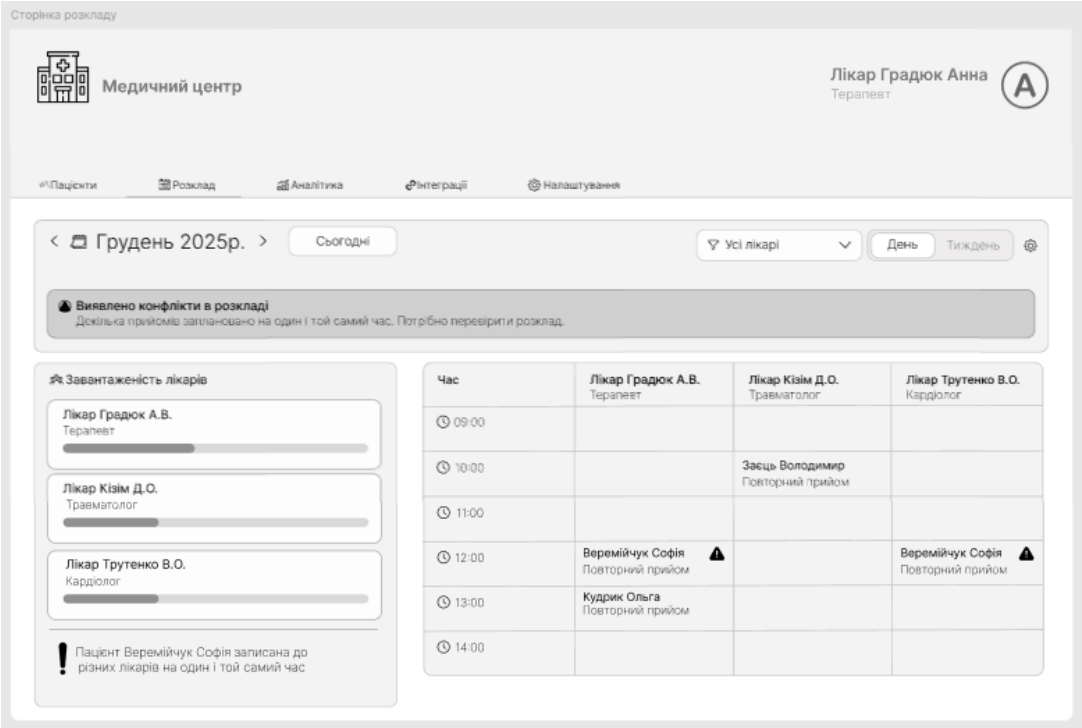


Рисунок Ж.3 – Сторінка навантаження лікарів

Джерело: власна розробка

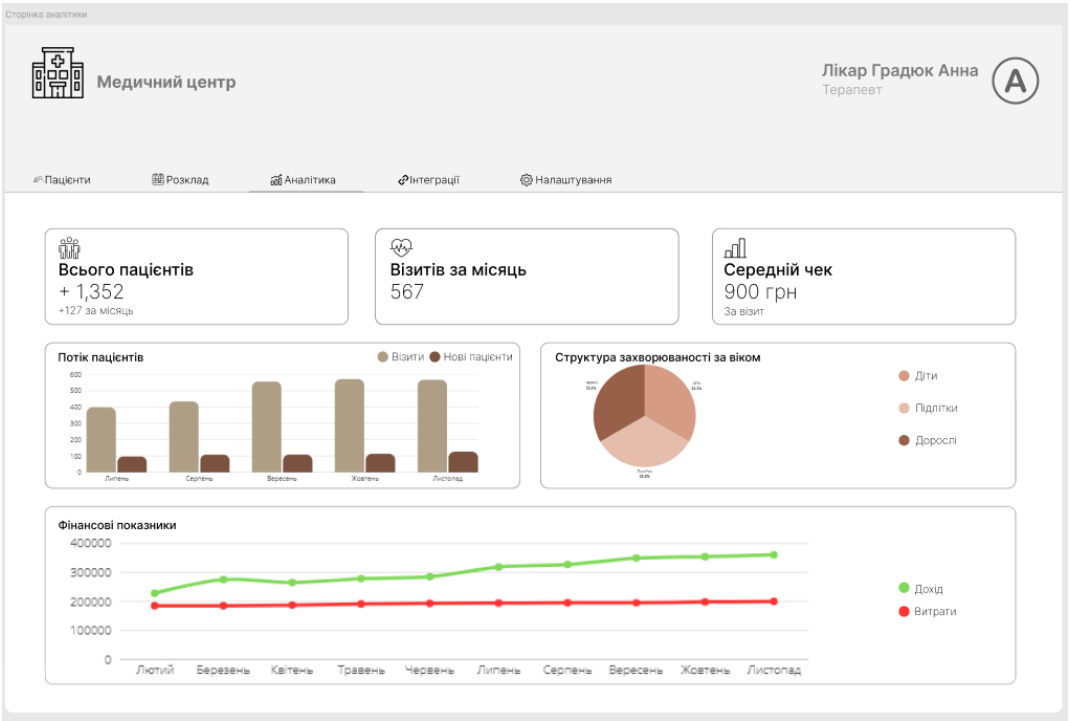


Рисунок Ж.4 – Сторінка аналітики

Джерело: власна розробка

Додаток З

Лістинг програмного коду системи ЕМК

Лістинг 3.1 – Файл «app.py»

```
from routes.auth import auth
from routes.analytics import analytics
from routes.appointments import appointments
from routes.patients import patients
from flask import Flask, redirect, url_for
from flask_login import LoginManager
from config import Config
from models import db, User

app = Flask(__name__)
app.config.from_object(Config)

db.init_app(app)

login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = 'auth.login'
login_manager.login_message = 'Будь ласка, увійдіть в систему'

@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))

@app.route('/')
def index():
    return redirect(url_for('auth.login'))

app.register_blueprint(auth, url_prefix='/auth')
app.register_blueprint(patients, url_prefix='/patients')
app.register_blueprint(appointments, url_prefix='/appointments')
```

```
app.register_blueprint(analytics, url_prefix='/analytics')

if __name__ == '__main__':
    app.run(debug=True)
```

Джерело: власна розробка

Лістинг 3.2 – Файл «models.py» (фрагмент)

```
from flask_sqlalchemy import SQLAlchemy
from flask_login import UserMixin
from datetime import datetime

db = SQLAlchemy()

class User(UserMixin, db.Model):
    __tablename__ = 'users'
    user_id = db.Column(db.Integer, primary_key=True)
        login = db.Column(db.String(100), unique=True,
nullable=False)
    password_hash = db.Column(db.String(255), nullable=False)
    role = db.Column(db.String(50), nullable=False)
    created_at = db.Column(db.DateTime, default=datetime.utcnow)

    def get_id(self):
        return str(self.user_id)

class Patient(db.Model):
    __tablename__ = 'patients'
    patient_id = db.Column(db.Integer, primary_key=True)
                user_id = db.Column(db.Integer,
db.ForeignKey('users.user_id'))
    first_name = db.Column(db.String(100), nullable=False)
    last_name = db.Column(db.String(100), nullable=False)
    middle_name = db.Column(db.String(100))
```

```
        date_of_birth = db.Column(db.Date, nullable=False)
        phone = db.Column(db.String(20))
        email = db.Column(db.String(100))
        address = db.Column(db.String(200))
        allergies = db.Column(db.Text)
        declaration_status = db.Column(db.Boolean, default=False)
        created_at = db.Column(db.DateTime, default=datetime.utcnow)
    class MedicalRecord(db.Model):
        __tablename__ = 'medical_records'
        record_id = db.Column(db.Integer, primary_key=True)
                    patient_id       =       db.Column(db.Integer,
db.ForeignKey('patients.patient_id'))
                     doctor_id       =       db.Column(db.Integer,
db.ForeignKey('doctors.doctor_id'))
        appointment_id = db.Column(
           db.Integer, db.ForeignKey('appointments.appointment_id'))
        diagnosis_code = db.Column(db.String(10), nullable=False)
        diagnosis_description = db.Column(db.Text)
        complaints = db.Column(db.Text)
        examination_results = db.Column(db.Text)
        treatment = db.Column(db.Text)
        visit_date = db.Column(db.DateTime, nullable=False)
        created_at = db.Column(db.DateTime, default=datetime.utcnow)
                    patient      =      db.relationship('Patient',
backref='medical_records')
        doctor = db.relationship('Doctor', backref='medical_records')
```

Джерело: власна розробка

Лістинг 3.3 - Файл «config.py»

```
import os
class Config:
    SECRET_KEY = 'emk-secret-key-2025'
```

```
                                    SQLALCHEMY_DATABASE_URI              =
'postgresql://postgres:postgres@localhost:5432/emk_db'
        SQLALCHEMY_TRACK_MODIFICATIONS = False
```

Джерело: власна розробка

Лістинг 3.4 – Файл «auth.py»

```
    from flask import Blueprint, render_template, redirect, url_for,
request, flash
    from flask_login import login_user, logout_user, login_required
    from werkzeug.security import check_password_hash
    from models import User

    auth = Blueprint('auth', __name__)

    @auth.route('/login', methods=['GET', 'POST'])
    def login():
        if request.method == 'POST':
            login = request.form.get('login')
            password = request.form.get('password')
            user = User.query.filter_by(login=login).first()
               if  user  and  check_password_hash(user.password_hash,
password):
                login_user(user)
                return redirect(url_for('patients.list_patients'))
            flash('Невірний логін або пароль')
        return render_template('auth/login.html')

    @auth.route('/logout')
    @login_required
    def logout():
        logout_user()
        return redirect(url_for('auth.login'))
```

Джерело: власна розробка

Лістинг 3.5 – Файл «patients.py» (фрагмент)

```
    from flask import Blueprint, render_template, request, redirect,
url_for, flash
    from flask_login import login_required, current_user
    from models import db, Patient, Doctor, MedicalRecord,
Appointment, VitalSigns
    from datetime import datetime

    patients = Blueprint('patients', __name__)

    @patients.route('/')
    @login_required
    def list_patients():
        search = request.args.get('search', '')
        if search:
            all_patients = Patient.query.filter(
                (Patient.last_name.ilike(f'%{search}%')) |
                (Patient.first_name.ilike(f'%{search}%'))
            ).all()
        else:
            all_patients = Patient.query.all()
                  return      render_template('patients/list.html',
patients=all_patients, search=search)

    @patients.route('/<int:patient_id>')
    @login_required
    def view_patient(patient_id):
        patient = Patient.query.get_or_404(patient_id)
                                          records                  =
MedicalRecord.query.filter_by(patient_id=patient_id).order_by(
            MedicalRecord.visit_date.desc()).all()
```

```
                                        vitals                    =
VitalSigns.query.filter_by(patient_id=patient_id).order_by(
            VitalSigns.measured_at.desc()).first()
                  return      render_template('patients/view.html',
patient=patient, records=records, vitals=vitals)
```

Джерело: власна розробка

Лістинг 3.6 – Файл «appointments.py»

```
    from flask import Blueprint, render_template, request, redirect,
url_for, flash
    from flask_login import login_required
    from models import db, Appointment, Patient, Doctor
    from datetime import datetime, date

    appointments = Blueprint('appointments', __name__)

    @appointments.route('/')
    @login_required
    def schedule():
                    date_str       =       request.args.get('date',
date.today().strftime('%Y-%m-%d'))
            selected_date   =   datetime.strptime(date_str,   '%Y-%m-
%d').date()
        all_appointments = Appointment.query.filter_by(
            appointment_date=selected_date
        ).order_by(Appointment.appointment_time).all()
        doctors = Doctor.query.all()
        return render_template('appointments/schedule.html',
                               appointments=all_appointments,
                               selected_date=selected_date,
                               doctors=doctors)

    @appointments.route('/new', methods=['GET', 'POST'])
```

```
    @login_required
    def new_appointment():
        if request.method == 'POST':
            appointment = Appointment(
                patient_id=request.form.get('patient_id'),
                doctor_id=request.form.get('doctor_id'),
                appointment_date=datetime.strptime(
                      request.form.get('appointment_date'), '%Y-%m-
%d').date(),
                appointment_time=datetime.strptime(
                                request.form.get('appointment_time'),
'%H:%M').time(),
                notes=request.form.get('notes')
            )
            db.session.add(appointment)
            db.session.commit()
            flash('Запис успішно створено')
            return redirect(url_for('appointments.schedule'))
        patients = Patient.query.all()
        doctors = Doctor.query.all()
        return render_template('appointments/new_appointment.html',
                               patients=patients, doctors=doctors)
```

Джерело: власна розробка

Лістинг 3.7 – Файл «analytics.py»

```
    from flask import Blueprint, render_template
    from flask_login import login_required
    from models import db, Patient, Appointment, MedicalRecord,
Doctor
    from sqlalchemy import func

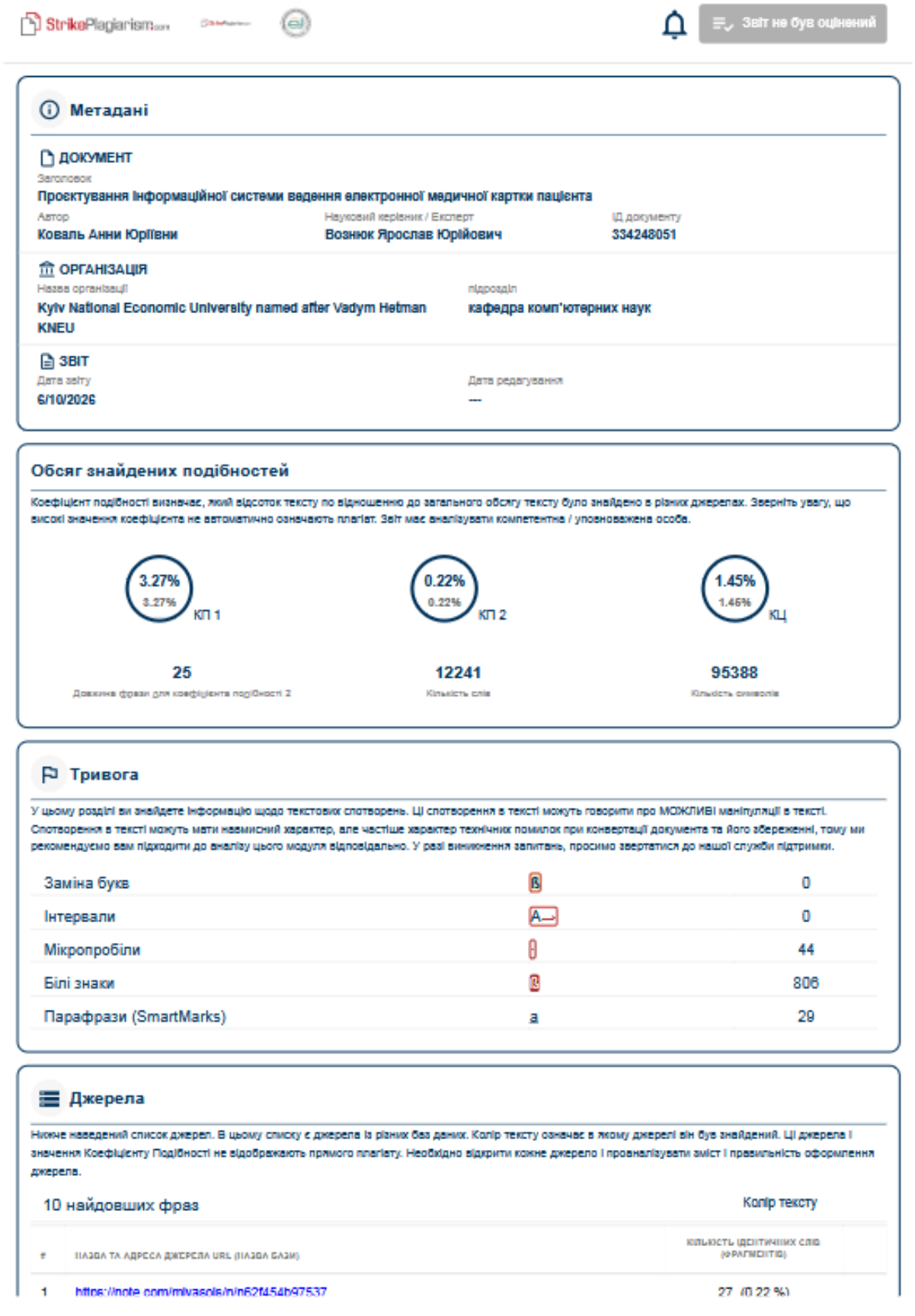
    analytics = Blueprint('analytics', __name__)
```

```
@analytics.route('/')
@login_required
def dashboard():
    total_patients = Patient.query.count()
    total_appointments = Appointment.query.count()
                                        completed                    =
Appointment.query.filter_by(status='completed').count()
    doctors_count = Doctor.query.count()
    top_doctors = db.session.query(
        Doctor.last_name,
        Doctor.specialization,
        func.count(Appointment.appointment_id).label('total')
              ).join(Appointment,      Doctor.doctor_id      ==
Appointment.doctor_id)\
              .group_by(Doctor.doctor_id,      Doctor.last_name,
Doctor.specialization)\
     .order_by(func.count(Appointment.appointment_id).desc())\
     .limit(5).all()
    return render_template('analytics/dashboard.html',
                           total_patients=total_patients,
                          total_appointments=total_appointments,
                           completed=completed,
                           doctors_count=doctors_count,
                           top_doctors=top_doctors)
```

Джерело: власна розробка

Додаток К

Короткий звіт подібності



StrikePlagiarism

Звіт не був оцінений

Метадані

ДОКУМЕНТ

Заголовок
Проектування інформаційної системи ведення електронної медичної картки пацієнта

Автор
Коваль Анни Юріївни

Науковий керівник / Експерт
Вознюк Ярослав Юрійович

ІД документу
334248051

ОРГАНІЗАЦІЯ

Назва організації
Kyiv National Economic University named after Vadym Hetman KNEU

підрозділ
кафедра комп'ютерних наук

ЗВІТ

Дата звіту
6/10/2026

Дата редагування

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

3.27% 3.27% КП 1

0.22% 0.22% КП 2

1.45% 1.45% КЦ

25
Довжина фрази для коефіцієнта подібності 2

12241
Кількість слів

95388
Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв	B	0
Інтервали	A	0
Мікропробіли		44
Білі знаки		806
Парафрази (SmartMarks)	a	29

Джерела

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз | Колір тексту

#	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://note.com/milvasolo/n/n62f454b97537	27 (0.22 %)

2	https://lr.kneu.edu.ua/bitstreams/066dbca0-6afa-4cf4-a181-c3e68a2aab6f/download	23 (0.19 %)
3	https://lr.kneu.edu.ua/bitstreams/8211ece0-ff9a-4fb1-bcb1-e03ba1f543d7/download	21 (0.17 %)
4	Method of dynamic targeting of a heterogeneous swarm of striking unmanned aerial vehicles based on iterative procedures of «assignment problem» Шовкошитний Ігор Іванович, Василенко Ольга Анатоліївна;	21 (0.17 %)
5	Проектування веб-сервісу онлайн-кіноафіші [redacted] 6/22/2025 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра комп'ютерних наук)	21 (0.17 %)
6	https://lr.kneu.edu.ua/server/api/core/bitstreams/add9ce43-9c83-4412-bccd-41d87f874e83/content	20 (0.16 %)
7	https://lr.kneu.edu.ua/bitstreams/f2d2da61-14eb-4ba9-b82f-2663cc9ae485/download	20 (0.16 %)
8	https://lr.kneu.edu.ua/bitstreams/8211ece0-ff9a-4fb1-bcb1-e03ba1f543d7/download	19 (0.16 %)
9	https://lr.kneu.edu.ua/bitstreams/4559c31c-9da2-4bfa-b1bd-f336a89d8c4b/download	18 (0.15 %)
10	https://lr.kneu.edu.ua/server/api/core/bitstreams/add9ce43-9c83-4412-bccd-41d87f874e83/content	15 (0.12 %)

База даних RefBooks (0.25 %)

#	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
джерело: Paperity		
1	Method of dynamic targeting of a heterogeneous swarm of striking unmanned aerial vehicles based on iterative procedures of «assignment problem» Шовкошитний Ігор Іванович, Василенко Ольга Анатоліївна;	21 (1) (0.17 %)
2	Legal protection of personal data in the field of health care in Ukraine О. М. Коваль,І. В. Діордіца, І. А. Коваленко;	10 (1) (0.08 %)

Домашня база даних (0.38 %)

#	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
3	Проектування веб-сервісу онлайн-кіноафіші [redacted] 6/22/2025 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра комп'ютерних наук)	42 (4) (0.34 %)
4	Розробка підсистеми для функціонування IT-підприємства [redacted] 6/14/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра інформаційних систем в економіці)	5 (1) (0.04 %)

Програма обміну базами даних (0.24 %)

#	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

6	КПІ_2025_121_МАМОНЕНКО [redacted] 5/6/2025 Ukrainian national aviation university (Криворізький Фаховий коледж)	15 (3) (0.12 %)
8	122_Smillansky_Mykhailo_Yuriyovich_It2025 [redacted] 6/2/2025 National University of Food Technologies (Кафедра інформаційних технологій, штучного інтелекту і кібербезпеки)	14 (2) (0.11 %)

Інтернет (2.39 %)

#	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
7	https://ir.kneu.edu.ua/bitstreams/066dbca0-6afa-4cf4-a181-c3e88a2aab6f/download	79 (6) (0.65 %)
8	https://ir.kneu.edu.ua/bitstreams/8211ece0-ff9a-4fb1-bcb1-e03ba1f543d7/download	60 (4) (0.49 %)
9	https://ir.kneu.edu.ua/server/api/core/bitstreams/add9ce43-9c83-4412-bccd-41d87f874e83/content	43 (3) (0.35 %)
10	**https://note.com/miyasols/n/n62f454b97537**	**27 (1) (0.22 %)**
11	https://www.codersarts.com/post/how-to-create-login-system-using-flask	23 (2) (0.19 %)
12	https://ir.kneu.edu.ua/bitstreams/f2d2da61-14eb-4ba9-b82f-2663cc9ae485/download	20 (1) (0.16 %)
13	https://ir.kneu.edu.ua/bitstreams/4559c31c-9da2-4bfa-b1bd-f336a89d8c4b/download	18 (1) (0.15 %)
14	https://gist.github.com/bkdinoop/6698956	12 (1) (0.1 %)
15	https://ir.kneu.edu.ua/bitstreams/57721939-fba5-4699-afde-625c26f42ccb/download	6 (1) (0.05 %)
16	https://ru.essays.club/%D0%A2%D0%BE%D1%87%D0%BD%D1%8B%D0%B5-%D0%BD%D0%B0%D1%83%D0%BA%D0%B8/%D0%9A%D0%BE%D0%BC%D0%BC%D1%83%D0%BD%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B8-%D0%B8-%D1%81%D0%B2%D1%8F%D0%B7%D1%8C/%D0%97%D0%B0%D1%85%D0%B8%D1%81%D1%82-%D1%96%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D1%97-%D0%B7-%D0%BE%D0%B1%D0%BC%D0%B5%D0%B6%D0%B5%D0%BD%D0%B8%D0%BC-470343.html	5 (1) (0.04 %)

Список прийнятих фрагментів

#	ЗМІСТ	КІЛЬКІСТЬ ОДІЛАКОВИХ СЛІВ (ФРАГМЕНТІВ)