Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка інформаційної панелі для моніторингу ключових показників діяльності

Виконав: студент IV курсу, групи СН-41
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

_____ Кучера Г. М.
(підпис) (прізвище та ініціали)

Керівник _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Шимчук Г. В.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль
2026

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«__» _червня_ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Кучері Галині Миколаївні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка інформаційної панелі для моніторингу ключових показників діяльності

Керівник роботи Боднарчук Ігор Орестович, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*14*» *травня* 2026 року *№ 4/9-239*

2. Термін подання студентом завершеної роботи 23 червня 2026 р.

3. Вихідні дані до роботи Літературні та інтернет джерела.

4. Зміст роботи (перелік питань, які потрібно розробити)
Анотація. Annotation. Перелік умовних одиниць, скорочень і термінів. Зміст. Вступ. Розділ 1. Аналіз предметної області та постановка завдання (1.1 Аналіз діяльності медичних закладів як об'єкта автоматизації; 1.2 Огляд та порівняльний аналіз існуючих інформаційно-аналітичних систем у медицині; 1.3 Обґрунтування вибору технологій та інструментальних засобів; 1.4 розробки Формування функціональних вимог до системи та постановка задач проектування; 1.5 Висновки до першого розділу), Розділ 2. Проектування інформаційно-аналітичної системи медичного закладу (2.1 Розробка структурно-функціональної схеми системи управління; 2.2 Моделювання бізнес-процесів взаємодії користувачів (реєстратор, лікар, пацієнт, директор); 2.3 Проектування логічної та фізичної структури бази даних медичного закладу; 2.4 Методи та алгоритми автоматизованого збору та обробки аналітичних даних; 2.5 Проектування архітектури інтерфейсу користувача (UI/UX) для різних ролей доступу; 2.6 Висновки до другого розділу), Розділ 3. Практична реалізація та аналіз результатів впровадження системи (3.1 Реалізація модулів адміністрування та керування персоналом і послугами клініки; 3.2. Розробка підсистеми реєстрації пацієнтів та планування медичних візитів; 3.3 Програмна реалізація аналітичного модуля візуалізації показників ефективності закладу; 3.5 Тестування системи та аналіз переваг її впровадження; 3.6 Висновки до третього розділу), 4. Безпека життєдіяльності, основи охорони праці (4.1 Вплив архітектури інтерфейсу інформаційної системи на психофізіологічний стан та безпеку праці медичного персоналу; 4.2 Ергономічні вимоги до організації робочого середовища розробника та користувачів аналітичних систем; 4.3 Висновки до четвертого розділу). Висновки. Перелік джерел. Додатки (Додаток А, Б, В, Г)

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
Тема дипломної роботи. Актуальність роботи. Цілі та завдання роботи. Об'єкт, предмет та методи дослідження. Технологічне обгрунтування та архітектура. Функціонал директора, лікаря, пацієнта та реєстратора. Безпека та технічні показники системи. Висновки до роботи.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці			

7. Дата видачі завдання 26 січня 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	26.01.2026	*Виконано*
2.	Підбір та опрацювання літературних джерел по темі	27.01.2026-16.02.2026	*Виконано*
	кваліфікаційної роботи		
3.	Виконання дослідження щодо методів обробки	17.02.2026-10.05.2026	*Виконано*
	аналітичних даних		
	Розроблення модулів та підсистем інформаційно-		
	аналітичної системи		
4.	Оформлення розділу «Аналіз предметної області та	11.05.2026-17.05.2026	*Виконано*
	постановка завдання»		
5.	Оформлення розділу «Проектування інформаційно-	18.05.2026-24.05.2026	*Виконано*
	аналітичної системи медичного закладу»		
6.	Оформлення розділу «Практична реалізація та аналіз	25.05.2026-31.05.2026	*Виконано*
	результатів впровадження системи»		
7.	Виконання завдання до підрозділу «Безпека	01.06.2026-08.06.2026	*Виконано*
	життєдіяльності»		
8.	Виконання завдання до підрозділу «Основи охорони	01.06.2026-08.06.2026	*Виконано*
	праці»		
9.	Оформлення кваліфікаційної роботи	09.06.2026-11.06.2026	*Виконано*
10.	Нормоконтроль	12.06.2026-15.06.2026	*Виконано*
11.	Перевірка на плагіат	16.06.2026	*Виконано*
12.	Попередній захист кваліфікаційної роботи	18.06.2026	*Виконано*
13.	Захист кваліфікаційної роботи	23.06.2026	

Студент _____ Кучера Г. М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка інформаційної панелі для моніторингу ключових показників діяльності // Кваліфікаційна робота освітнього рівня «Бакалавр» // Кучера Галина Миколаївна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СН-41 // Тернопіль, 2026 // С. – 62, рис. – 23, табл. – 1, додат. – 12, бібліогр. – 42.

Ключові слова: інформаційно-аналітична система, медичний центр, автоматизація медицини, візуалізація даних, PHP, MySQL, Chart.js, охорона праці.

Кваліфікаційна робота присвячена розробці та впровадженню комплексної інформаційно-аналітичної системи для автоматизації діяльності сучасного медичного закладу.

У першому розділі кваліфікаційної роботи проведено аналіз предметної області та сучасного стану автоматизації медичних послуг. Висвітлено проблеми обробки великих масивів даних та необхідність впровадження аналітичних інструментів для підтримки прийняття управлінських рішень. Розглянуто існуючі аналоги та обґрунтовано вибір технологічного стеку для розробки.

У другому розділі кваліфікаційної роботи виконано проектування архітектури системи. Описано структуру реляційної бази даних, розроблено діаграми варіантів використання та потоків даних. Сформовано концепцію UI/UX-дизайну, визначено ролі користувачів (директор, лікар, реєстратор, пацієнт) та спроектовано логіку їхньої взаємодії з платформою.

У третьому розділі кваліфікаційної роботи детально описано процес програмної реалізації системи з використанням мови PHP та бібліотеки Chart.js. Представлено реалізацію особистих кабінетів, модулів онлайн-запису, електронного архіву та інтерактивних звітів. Проведено функціональне

тестування системи, що підтвердило точність аналітичних розрахунків та стабільність роботи модулів.

У четвертому розділі кваліфікаційної роботи розглянуто питання безпеки життєдіяльності та охорони праці. Проаналізовано вплив інтерфейсу на психофізіологічний стан персоналу та встановлено ергономічні вимоги до організації робочих місць користувачів системи.

Об’єкт дослідження – процес розробки та функціонування інформаційно-аналітичної системи медичного центру. Предмет дослідження – методи, засоби та інструментарій для проектування, реалізації та аналізу ефективності медичних інформаційних систем.

ANNOTATION

Development of a Dashboard for Monitoring Key Performance Indicators // Bachelor's Qualification Work // Kuchera Halyna Mykolaivna // Ternopil Ivan Puluj National Technical University, Faculty of Computer and Information Systems and Software Engineering, Department of Computer Science, group SN-41 // Ternopil, 2026 // P. – 62, figs. – 23, tabs. – 1, annexes. – 12, bibliogr. – 42.

***Keywords*:** information and analytical system, medical center, medical automation, data visualization, PHP, MySQL, Chart.js, occupational health and safety.

The qualification work is devoted to the development and implementation of a comprehensive information and analytical system for automating the activities of a modern medical institution.

In the first chapter of the qualification work, the analysis of the subject area and the current state of medical services automation is carried out. The problems of processing large data sets and the need to implement analytical tools to support management decision-making are highlighted. Existing analogues are reviewed, and the choice of the technological stack for development is justified.

In the second chapter of the qualification work, the design of the system architecture is performed. The structure of the relational database is described, use case diagrams and data flow diagrams are developed. The concept of UI/UX design is formed, user roles (director, doctor, registrar, patient) are defined, and the logic of their interaction with the platform is designed.

In the third chapter of the qualification work, the process of software implementation of the system using PHP and Chart.js library is described in detail. The implementation of personal accounts, online appointment modules, digital archive, and interactive reports is presented. Functional testing of the system was conducted, which confirmed the accuracy of analytical calculations and the stability of the modules' operation.

In the fourth chapter of the qualification work, the issues of life safety and occupational health and safety are considered. The influence of the interface on the psychophysiological state of the personnel is analyzed, and ergonomic requirements for the organization of systems' users workplaces are established.

The object of research is the process of development and functioning of the information and analytical system of a medical center. The subject of research is the methods, tools, and software for the design, implementation, and performance analysis of medical information systems.

ПЕРЕЛІК УМОВНИХ ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Apache – вебсервер з відкритим вихідним кодом.

Chart.js –відкрита JavaScript-бібліотека для візуалізації даних за допомогою діаграм.

CSS (англ. Cascading Style Sheets) – каскадні таблиці стилів.

CSV-файли (англ. Comma-Separated Values) – текстовий формат, призначений для представлення табличних даних.

HTML (англ. HyperText Markup Language) – мова розмітки гіпертексту.

InnoDB – високопродуктивна система збереження даних (рушій) для СУБД MySQL.

JavaScript – мультипарадигмова мова програмування для створення інтерактивних вебсторінок.

JSON (англ. JavaScript Object Notation) – текстовий формат обміну даними.

MySQL – реляційна система керування базами даних.

PHP (англ. Personal Home Page / Hypertext Preprocessor) – мова програмування.

SQL-запит (англ. Structured Query Language) – запит до бази даних за допомогою структурованої мови запитів.

UI/UX (англ. User Interface / User Experience) – дизайн користувацького інтерфейсу та досвіду взаємодії.

XAMPP – кросплатформна збірка вебсервера, що містить Apache, MySQL та PHP.

лк. – люкс, одиниця вимірювання освітленості.

мм – міліметр, одиниця вимірювання довжини.

СУБД – система керування базами даних.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасних медичних закладах ефективність роботи значною мірою залежить від швидкості обробки інформації та доступу до актуальних аналітичних даних. Використання цифрових систем дозволяє автоматизувати процеси реєстрації пацієнтів, ведення медичної документації та формування звітності. Особливе значення мають інформаційні панелі (дашборди), які забезпечують моніторинг ключових показників діяльності закладу, зокрема завантаженості лікарів, кількості записів на прийом і рівня задоволеності пацієнтів. Тому розробка інформаційно-аналітичної системи медичного центру з інтегрованою панеллю моніторингу є актуальним завданням.

Мета роботи полягає у розробці інформаційно-аналітичної системи медичного центру з інтегрованою інформаційною панеллю для моніторингу ключових показників діяльності, яка забезпечує автоматизацію запису пацієнтів, ведення електронних медичних карт та формування аналітичної звітності.

Об’єктом дослідження є процеси управління та інформаційного супроводу діяльності медичного закладу.

Предметом дослідження є методи та програмні засоби розробки інформаційних систем і аналітичних панелей із використанням вебтехнологій PHP, MySQL та Chart.js.

Для досягнення поставленої мети необхідно вирішити такі **завдання**:

- проаналізувати предметну область та сформувати функціональні вимоги до системи;
- спроектувати структуру бази даних для збереження медичної та аналітичної інформації;
- розробити інтерфейс користувача для різних категорій користувачів;
- реалізувати інформаційну панель для моніторингу ключових показників діяльності медичного закладу;
- провести тестування системи та оцінити результати її впровадження;

- дослідити питання охорони праці та безпеки життєдіяльності при роботі з інформаційною системою.

Методи дослідження. У роботі використано методи системного аналізу, реляційного моделювання баз даних, вебпрограмування, UI/UX-проектування та тестування програмного забезпечення.

Практичне значення отриманих результатів полягає у створенні програмного продукту для автоматизації роботи медичного центру та моніторингу його основних показників діяльності. Розроблена система дозволяє скоротити час обробки інформації, підвищити точність аналітики та покращити процес прийняття управлінських рішень.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Аналіз діяльності медичних закладів як об’єкта автоматизації

Сучасний медичний заклад є складною структурою, де одночасно взаємодіють потоки пацієнтів, медичного персоналу та адміністративних ресурсів. Традиційні методи управління, що базуються на паперових носіях або розрізнених електронних таблицях, призводять до значних часових витрат на обробку інформації. Основними ланками, що потребують цифрової трансформації, є процеси реєстрації відвідувачів, ведення медичних карток, формування графіків роботи лікарів та збір статистичних показників для керівництва [39].

Діяльність реєстратури є первинним етапом взаємодії пацієнта з клінікою. При відсутності єдиної бази даних виникають помилки дублювання інформації, складність у пошуку існуючих амбулаторних карток та нераціональний розподіл часу прийому. Впровадження інструментів швидкого пошуку пацієнтів за прізвищем або номером телефону дозволяє реєстратору прискорити обслуговування в декілька разів, забезпечуючи наповнення системи первинними даними для подальшого аналізу.

Медичний персонал у своїй щоденній роботі потребує оперативного доступу до історії візитів та результатів досліджень. Можливість перегляду електронних архівів та завантажених файлів аналізів у профілі пацієнта виключає ризик втрати паперових документів. Окрім клінічної складової, важливим аспектом є контроль завантаженості кожного фахівця, що дозволяє оптимізувати робочий час та уникати великих черг у коридорах закладу [37].

Для керівника медичного центру об’єктом автоматизації стає управлінська звітність. Директор має володіти актуальними цифрами щодо загальної кількості зареєстрованих осіб, активних записів та середнього рейтингу закладу на основі відгуків. Як зазначається у концепції розвитку цифрової медицини,

“цифровізація охорони здоров'я має на меті створення умов для оперативного прийняття управлінських рішень на основі аналізу великих масивів даних” [18]. Саме аналітичний блок дозволяє виявити найбільш затребувані відділення та оцінити роботу персоналу на основі статистичних вибірок.

Процес аналізу діяльності закладу через цифрові інструменти включає збір даних про статуси прийомів (виконано, заплановано, скасовано). Наявність таких відомостей у реальному часі дає змогу оперативно реагувати на зміни в робочому процесі, наприклад, через блокування доступу пацієнтам, які систематично ігнорують візити. Це забезпечує дисципліну та фінансову стійкість організації.

Таким чином, автоматизація медичного закладу охоплює всі рівні: від низової ланки збору даних у реєстратурі до високорівневого аналізу ефективності директором. Створення цілісної інформаційно-аналітичної системи стає базовою умовою для підвищення якості надання послуг та спрощення внутрішньої комунікації між підрозділами.

1.2 Огляд та порівняльний аналіз існуючих інформаційно-аналітичних систем у медицині

В Україні використовується велика кількість програм для автоматизації роботи медичних закладів, які відносяться до медичних інформаційних систем. Вони підключаються до державної електронної системи охорони здоров’я eHealth, роботу якої координує Національна служба здоров'я України. Такі системи проходять перевірку перед підключенням, тому на практиці використовується багато різних рішень, що відрізняються між собою за функціями та способом організації роботи з даними [40]. Через це медичні заклади змушені обирати систему, яка не завжди повністю відповідає їхнім потребам.

Поширеним варіантом є хмарні системи, які працюють через браузер і не потребують встановлення на комп’ютер. Вони зручні тим, що забезпечують

доступ до даних з різних пристроїв і автоматично взаємодіють із державними сервісами. Проте такі системи часто мають перевантажений інтерфейс: користувач бачить багато функцій, які не використовуються щодня. У дослідженнях зазначається, що складний інтерфейс і велика кількість елементів можуть ускладнювати роботу та збільшувати час навчання персоналу [1]. У результаті працівники витрачають більше часу на виконання простих дій, наприклад запис пацієнта або перегляд історії візитів.

Більшість сучасних систем дозволяють зберігати інформацію про пацієнтів, прийоми лікарів та надані послуги. Однак можливості аналізу цих даних зазвичай обмежені. Наприклад, система може показати кількість прийомів або список пацієнтів, але для отримання більш детальної інформації необхідно використовувати додаткові інструменти або окремі модулі [17]. Це створює незручності для керівництва закладу, оскільки потрібна інформація не завжди доступна одразу і її доводиться збирати з різних джерел.

Окремо варто розглянути професійні системи управління, які мають більше можливостей для роботи з даними. Вони дозволяють будувати звіти, графіки та інші форми представлення інформації. Проте такі рішення зазвичай є складними у впровадженні та потребують значних витрат на налаштування і підтримку [42]. Для невеликих медичних центрів це є суттєвим обмеженням.

З урахуванням зазначеного, доцільним є створення власної системи, яка буде враховувати особливості роботи конкретного медичного закладу. Використання вебтехнологій дає можливість реалізувати простий інтерфейс для реєстратури та додати потрібні інструменти для аналізу даних без зайвих функцій. Такий підхід дозволяє зменшити залежність від сторонніх програм і забезпечити наявність тільки тих можливостей, які дійсно використовуються в роботі.

1.3 Обґрунтування вибору технологій та інструментальних засобів розробки

Для розробки інформаційної системи медичного закладу було обрано підхід, при якому система поділяється на серверну та клієнтську частини. Це дозволяє працювати з програмою через браузер без встановлення окремих застосунків на кожен комп’ютер. Обробка даних виконується на сервері, а користувач працює лише з інтерфейсом. Такий підхід спрощує підтримку системи та дозволяє використовувати її на різних пристроях.

Серверна частина реалізована з використанням мови програмування PHP. Вона широко застосовується для створення вебсистем, що працюють із базами даних. PHP добре взаємодіє з вебсерверами та дозволяє обробляти запити користувачів і виконувати операції з даними. У літературі зазначається, що ця мова має велику кількість готових рішень і активно використовується для створення динамічних вебдодатків [32].

Для збереження даних використовується система керування базами даних MySQL. Вона дозволяє працювати зі структурованими даними та встановлювати зв’язки між таблицями. Це важливо для організації інформації про пацієнтів, прийоми та результати обстежень. У джерелах зазначається, що MySQL підтримує роботу з великими обсягами даних і має механізми оптимізації запитів, зокрема індекси [35].

Для розгортання локального середовища розробки використано пакет XAMPP, який включає вебсервер Apache, інтерпретатор PHP та систему керування базами даних MySQL. Це дозволяє запускати та тестувати систему на локальному комп’ютері без необхідності налаштування кожного компонента окремо. Використання XAMPP значно спрощує процес розробки та перевірки працездатності вебдодатка.

Клієнтська частина системи реалізована за допомогою HTML, CSS та JavaScript. Ці технології відповідають за відображення інформації та взаємодію з користувачем. Для відображення статистичних даних використовується

бібліотека Chart.js, яка дозволяє будувати графіки на основі даних системи. Це дає можливість відображати, наприклад, кількість записів або завантаженість лікарів у зручному вигляді. JavaScript також використовується для оновлення даних на сторінці без її повного перезавантаження [2].

Обраний набір технологій дозволяє створити систему, яка працює через браузер і не потребує встановлення на кожен комп'ютер. Це означає, що доступ до системи можна отримати з будь-якого пристрою. Такий підхід зменшує витрати на впровадження та спрощує оновлення, оскільки всі зміни виконуються на сервері [19].

1.4 Формування функціональних вимог до системи та постановка задач проектування

На основі аналізу діяльності медичного закладу та особливостей організації лікувального процесу сформовано функціональні вимоги до інформаційно-аналітичної системи. Розподіл функціоналу між ролями користувачів дозволяє забезпечити ефективну взаємодію між адміністрацією, медичним персоналом, реєстратурою та пацієнтами, а також мінімізувати дублювання операцій і підвищити швидкість обробки інформації.

Користувач із роллю "Директор" працює з адміністративно-аналітичним модулем та має доступ до повної інформації системи. Основними функціями цієї ролі є формування статистики записів за визначений період із можливістю вибору конкретних дат, перегляд розподілу записів за статусами у графічному вигляді, аналіз завантаженості лікарів та відділень, а також керування персоналом. Директор має можливість додавати та редагувати дані про лікарів і реєстраторів, керувати графіками роботи та періодами відпусток, переглядати оцінки й відгуки пацієнтів, створювати внутрішні повідомлення та наради для персоналу, а також здійснювати імпорт і експорт даних бази [20].

Роль "Реєстратура" орієнтована на виконання операційних задач, пов'язаних із обслуговуванням пацієнтів. Реєстратор забезпечує реєстрацію

нових пацієнтів у системі, створення записів на прийом до лікаря, перегляд актуального розкладу роботи медичного персоналу та надання інформації щодо наявності лікаря на конкретну дату або час. Також передбачено можливість оновлення базової інформації про записи пацієнтів та контроль актуальності даних.

Користувач із роллю “Лікар” працює з клінічним модулем системи та має доступ до електронних медичних карт пацієнтів. Основними функціями лікаря є перегляд персонального графіка прийомів, отримання списку пацієнтів на поточний день, внесення інформації про стан здоров’я під час прийому, перегляд історії звернень пацієнтів та ведення електронної медичної документації. Крім цього, лікар отримує повідомлення про внутрішні наради та адміністративні розпорядження.

Роль “Пацієнт” реалізує функції клієнтського модуля та забезпечує дистанційну взаємодію із закладом через онлайн-сервіси. Пацієнт має можливість записуватися на прийом до лікаря, переглядати інформацію про спеціалістів і перелік медичних послуг, отримувати доступ до результатів прийомів та рекомендацій лікаря, завантажувати результати лабораторних аналізів і залишати відгуки після відвідування спеціаліста.

Для реалізації інформаційної системи необхідно вирішити низку технічних задач проектування. Основною задачею є розробка структури бази даних, яка забезпечує взаємозв’язок між пацієнтами, лікарями, записами на прийом та результатами аналізів [20]. Також необхідно реалізувати механізм фільтрації даних за часовими проміжками для побудови статистичних звітів і графіків, інтегрувати бібліотеку Chart.js для візуалізації аналітичної інформації, створити систему внутрішніх повідомлень між директором та персоналом, реалізувати функції імпорту й експорту даних та забезпечити керування структурою відділень і графіками роботи лікарів через інтерфейс системи.

1.5 Висновки до першого розділу

У першому розділі розглянуто предметну область та проаналізовано підходи до організації роботи медичних інформаційних систем. На основі цього визначено основні вимоги до майбутньої системи для медичного закладу.

Аналіз розвитку електронної системи охорони здоров’я в Україні показав, що медичні заклади поступово переходять від паперових журналів до цифрового обліку даних через систему eHealth, яку координує Національна служба здоров'я України. Це дозволяє зменшити використання паперових носіїв і забезпечує доступ до інформації про пацієнтів у цифровому вигляді.

Огляд існуючих медичних інформаційних систем показав, що більшість рішень є універсальними і розраховані на різні типи закладів. Через це частина функцій не використовується в повсякденній роботі, а деякі необхідні можливості відсутні або реалізовані частково. Також виявлено, що не всі системи зручні для швидкої роботи персоналу та не завжди дозволяють отримувати потрібну аналітичну інформацію без додаткових модулів. Це створює потребу у створенні окремої системи, яка буде орієнтована на конкретний медичний заклад.

Для реалізації проекту обрано вебпідхід із використанням PHP та MySQL. Таке поєднання дозволяє організувати роботу з даними через браузер і забезпечує збереження всієї інформації в єдиній базі. Для побудови графіків і відображення статистики використовується Chart.js, що дозволяє відображати дані у вигляді діаграм і спрощує їх аналіз.

На основі аналізу роботи медичного закладу визначено функціональні можливості системи для різних ролей користувачів: директора, реєстратора, лікаря та пацієнта. Кожна роль має власний набір функцій, пов’язаних із їхніми щоденними задачами – від запису пацієнтів до перегляду медичної інформації та аналітики.

Також сформовано основні задачі проектування системи, серед яких:

- створення структури бази даних для зберігання інформації про пацієнтів, лікарів і записи на прийом;
- реалізація фільтрації даних за датами для побудови статистики;
- побудова графіків і діаграм для аналізу роботи закладу;
- реалізація внутрішнього обміну повідомленнями між користувачами системи;
- забезпечення можливості оновлення та експорту даних.

Отримані результати дають основу для наступного етапу роботи, де буде виконано проектування архітектури системи, структури бази даних та логіки взаємодії між її компонентами.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ІНФОРМАЦІЙНО-АНАЛІТИЧНОЇ СИСТЕМИ МЕДИЧНОГО ЗАКЛАДУ

2.1 Розробка структурно-функціональної схеми системи управління

Під час проектування інформаційно-аналітичної системи було визначено основні модулі системи, зв’язки між ними та функції користувачів. Для побудови системи використано клієнт-серверний підхід, при якому обробка даних виконується на сервері, а взаємодія користувача із системою здійснюється через браузер [10]. Такий підхід дозволяє організувати централізоване зберігання інформації та забезпечити доступ до системи з різних пристроїв.

Серверна частина реалізує обробку запитів користувачів, взаємодію з базою даних та перевірку прав доступу. Для зберігання інформації використовується MySQL, у якій зберігаються дані про пацієнтів, записи на прийом, графіки роботи лікарів, медичні висновки та відгуки. Клієнтська частина системи забезпечує окремі інтерфейси для директора, реєстратора, лікаря та пацієнта.

Загальна структура системи наведена на рисунку 2.1. Усі модулі взаємодіють через спільну базу даних, що дозволяє автоматично оновлювати інформацію між користувачами. Наприклад, після створення запису реєстратором інформація одразу відображається у графіку лікаря, а після завершення прийому змінений статус візиту стає доступним у звітах адміністрації.

Одним із основних компонентів є модуль управління та аналітики, який використовується директором медичного закладу. Цей модуль призначений для роботи зі статистикою та керування персоналом. Користувач має можливість переглядати кількість записів за обраний період, аналізувати завантаженість лікарів, переглядати оцінки пацієнтів та працювати з кадровою інформацією. Також модуль забезпечує створення внутрішніх повідомлень для персоналу та доступ до функцій імпорту й експорту даних [15].

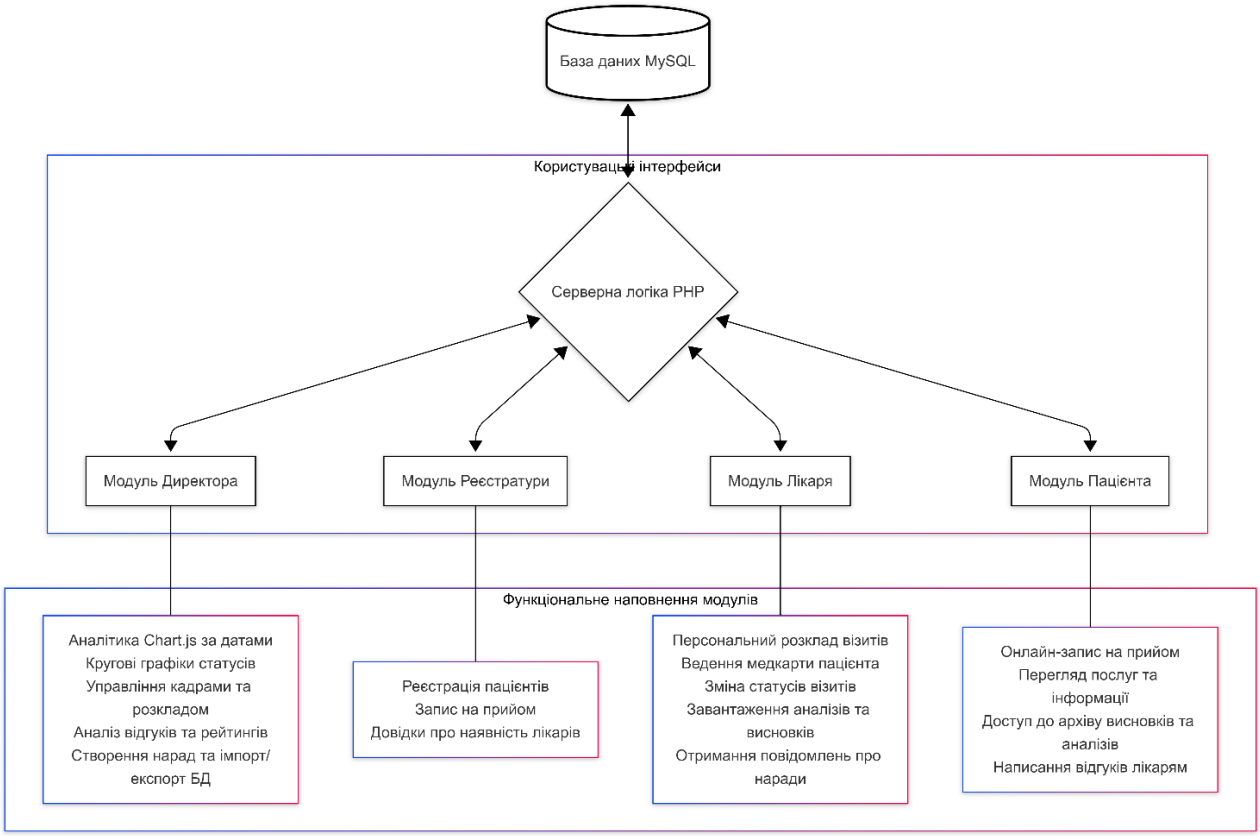

Рисунок 2.1 – Структурно-функціональна схема інформаційно-аналітичної системи медичного закладу

Модуль реєстратури використовується для щоденної роботи з пацієнтами. Через нього виконується реєстрація нових користувачів, створення записів на прийом та перегляд графіку лікарів. Реєстратор також може перевіряти інформацію про наявність лікаря на робочому місці відповідно до розкладу.

Модуль лікаря забезпечує роботу з електронними медичними картами та інформацією про прийоми. Лікар має доступ до списку пацієнтів, власного графіку та історії звернень. Під час прийому до системи можуть вноситися результати огляду, висновки та файли аналізів. Крім цього, лікар отримує повідомлення про наради або зміни у роботі закладу.

Модуль пацієнта призначений для онлайн-взаємодії із закладом. Через нього пацієнт може самостійно записатися на прийом, переглянути інформацію про лікарів та отримати доступ до результатів обстежень. Після завершення візиту користувач має можливість залишити відгук про роботу лікаря.

Для визначення доступу до функцій системи сформовано матрицю функціональних повноважень користувачів, яка наведена у таблиці 2.1. Вона показує, які операції доступні для кожної ролі системи [22].

Таблиця 2.1 – Матриця функціональних повноважень користувачів системи

Функціональна можливість системи	**Директор**	**Реєстратор**	**Лікар**	**Пацієнт**
Реєстрація нового пацієнта в базі	—	+	—	—
Створення запису на прийом	—	+	—	+
Перегляд розкладу та наявності лікарів	+	+	+	+
Зміна статусу візиту (Виконано/Скасовано)	—	—	+	—
Внесення даних у медкарту під час прийому	—	—	+	—
Завантаження файлів (аналізи/висновки)	—	—	+	—
Перегляд власних результатів та аналізів	—	—	—	+
Написання відгуку про лікаря	—	—	—	+
Аналітика: графіки завантаженості (Chart.js)	+	—	—	—
Управління кадрами, розкладом та відпустками	+	—	—	—
Створення нарад та зборів персоналу	+	—	—	—
Імпорт/Експорт та прямий доступ до БД	+	—	—	—

Структурна організація системи забезпечує взаємозв’язок між усіма модулями та дозволяє підтримувати актуальність інформації для кожного користувача. Завдяки цьому система може використовуватися як для щоденної роботи медичного персоналу, так і для аналізу діяльності закладу та контролю роботи персоналу.

2.2 Моделювання бізнес-процесів взаємодії користувачів (реєстратор, лікар, пацієнт, директор)

Для опису роботи інформаційно-аналітичної системи виконано моделювання основних бізнес-процесів, які виникають під час взаємодії користувачів із системою. Це дозволяє визначити порядок обробки даних, послідовність виконання дій та взаємозв’язки між модулями системи [9].

Одним із основних процесів є запис пацієнта на прийом та подальше обслуговування. Процес може бути ініційований як реєстратором, так і самим пацієнтом через онлайн-запис. Під час створення запису система перевіряє графік лікаря та наявність вільного часу. Після підтвердження інформація зберігається у базі даних зі статусом “Заплановано”. У день прийому лікар отримує доступ до списку пацієнтів, переглядає інформацію про візит та після завершення прийому вносить результати огляду до електронної медичної карти. Також лікар може додавати файли аналізів або висновки, які після збереження стають доступними пацієнту у власному кабінеті.

Іншим важливим процесом є аналітичний моніторинг роботи медичного закладу. Директор формує запит до системи, задаючи часовий проміжок для аналізу даних. Після цього система виконує обробку інформації про записи, статуси візитів та відгуки пацієнтів. Результати відображаються у вигляді графіків і діаграм за допомогою бібліотеки Chart.js. Це дозволяє переглядати завантаженість лікарів, кількість прийомів та скасованих записів. На основі отриманої інформації директор може змінювати графіки роботи персоналу або формувати внутрішні повідомлення для працівників.

Окремо реалізовано процес зворотного зв’язку. Після завершення прийому пацієнт має можливість залишити оцінку та відгук про роботу лікаря. Інформація автоматично зберігається у системі та використовується для формування рейтингу спеціалістів. Директор отримує доступ до всіх відгуків і може аналізувати рівень обслуговування в закладі [4].

У результаті моделювання бізнес-процесів визначено основні сутності системи: користувач, пацієнт, лікар, запис на прийом, медична карта, файл аналізів, відгук, розклад та нарада. На їх основі у наступних етапах проектування буде сформовано структуру бази даних і зв’язки між таблицями.

Послідовність взаємодії між користувачами та системою наведено на рисунку 2.2. Діаграма демонструє процес проходження інформації від моменту створення запису до формування аналітики для адміністрації закладу.

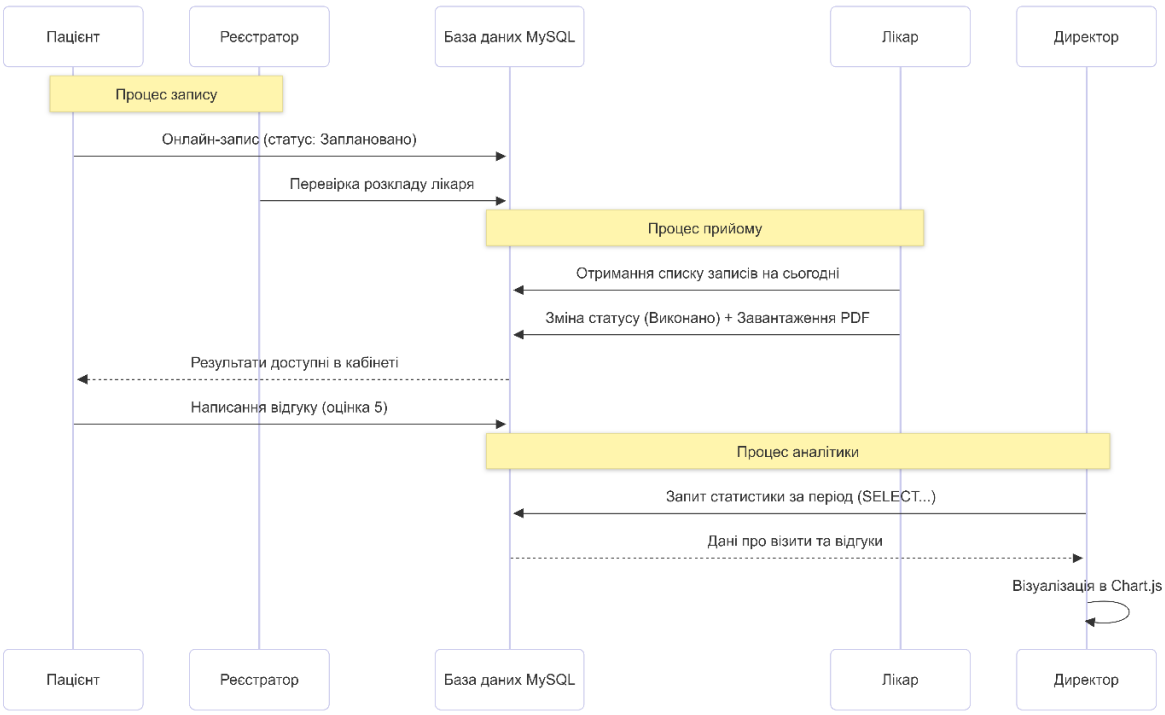

Рисунок 2.2 – Діаграма послідовності взаємодії користувачів із системою

Представлена діаграма дозволяє наочно показати послідовність взаємодії між користувачами системи та базою даних під час виконання основних операцій. Використання такого підходу спрощує аналіз потоків інформації та

допомагає визначити, які модулі беруть участь у кожному етапі обробки даних. Це також дає можливість виявити основні зв’язки між компонентами системи ще до етапу реалізації програмного коду, що спрощує подальше проектування структури бази даних та функціональних модулів системи [41].

2.3 Проектування логічної та фізичної структури бази даних медичного закладу

Проектування бази даних є одним із основних етапів створення інформаційної системи, оскільки саме база даних забезпечує збереження, оновлення та отримання інформації під час роботи користувачів. Для реалізації системи використано реляційну модель даних на основі MySQL, яка дозволяє організувати зв’язки між таблицями та забезпечити спільну роботу всіх модулів системи [27].

Під час проектування було визначено основні сутності, які використовуються у роботі медичного закладу. На їх основі сформовано структуру бази даних, що складається із взаємопов’язаних таблиць.

Таблиця users використовується для авторизації працівників системи. У ній зберігаються логіни, паролі та ролі користувачів. Це дозволяє розмежувати права доступу між директором, реєстратором та іншими працівниками.

Таблиця doctors містить інформацію про лікарів: прізвище та ім’я, спеціалізацію, відділення, графік роботи та періоди відпусток. Дані цієї таблиці використовуються під час формування розкладу та створення записів на прийом.

Таблиця patients призначена для збереження інформації про пацієнтів медичного закладу. У ній зберігаються персональні дані, необхідні для реєстрації та подальшого обслуговування пацієнта.

Центральною таблицею системи є appointments, яка містить інформацію про записи на прийом. Таблиця включає дату візиту, статус запису та зв’язки з лікарем і пацієнтом. Саме ці дані використовуються для побудови статистики та графіків у модулі аналітики [27].

Для збереження оцінок та коментарів пацієнтів використовується таблиця reviews. У ній містяться оцінка лікаря та текст відгуку. Інформація з цієї таблиці використовується для формування рейтингу спеціалістів та аналізу якості обслуговування. Таблиця files призначена для збереження файлів аналізів, висновків та іншої медичної документації. Кожен файл пов'язується з конкретним записом на прийом, що дозволяє пацієнту отримати доступ до результатів обстеження через систему. Для організації внутрішньої взаємодії між працівниками використовується таблиця meetings, у якій зберігається інформація про наради та повідомлення для персоналу.

Для підтримки зв'язків між таблицями у системі використано зовнішні ключі (Foreign Keys), що підтримуються механізмом InnoDB. Це дозволяє забезпечити цілісність даних та запобігти появі записів без відповідних зв'язків. Наприклад, запис на прийом не може бути створений без існування пацієнта та лікаря [23]. Логічна структура зв'язків між таблицями наведена на рисунку 2.3. ER-діаграма відображає основні сутності системи та взаємозв'язки між ними.

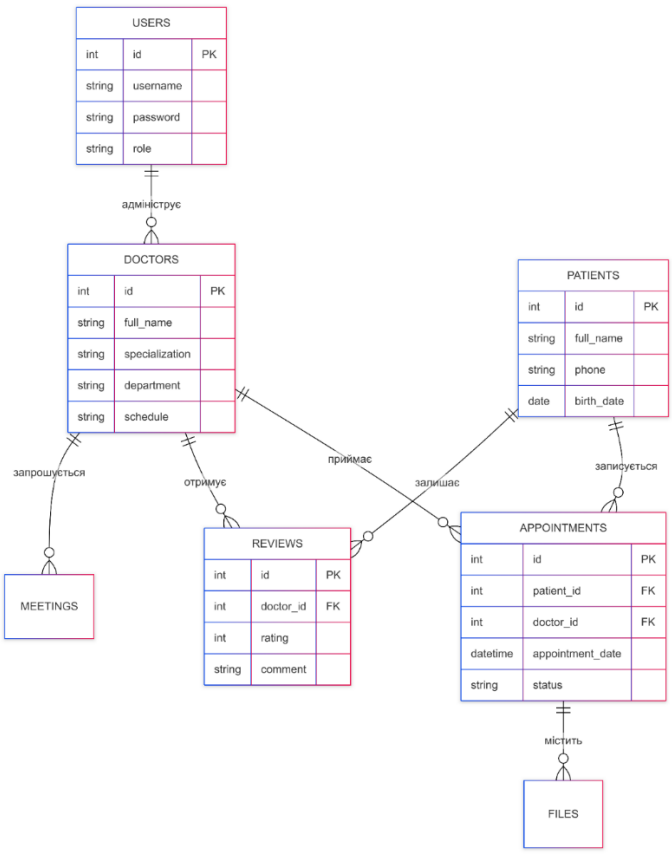

Рисунок 2.3 – ER-діаграма бази даних інформаційної системи

Представлена структура бази даних дозволяє організувати спільну роботу всіх модулів системи та забезпечує централізоване збереження інформації. Використання реляційної моделі спрощує обробку даних, формування звітів та побудову аналітики для керівництва медичного закладу.

2.4 Методи та алгоритми автоматизованого збору та обробки аналітичних даних

Ефективність інформаційно-аналітичної системи залежить від способів обробки накопичених даних та швидкості формування статистичної інформації. У розробленій системі реалізовано набір алгоритмів, які забезпечують автоматичний збір інформації з бази даних, її обробку та подальше відображення у вигляді графіків і діаграм [14].

Одним із основних алгоритмів є алгоритм фільтрації та агрегації даних для побудови графіків у Chart.js. Під час формування звітів директор задає початкову та кінцеву дати періоду аналізу. Після цього система виконує SQL-запит до таблиці appointments із використанням оператора BETWEEN, що дозволяє отримати записи лише за вибраний проміжок часу. Далі результати групуються за датами за допомогою GROUP BY DATE(appointment_date). Отримані дані обробляються PHP-скриптом, який формує масиви дат і відповідних значень кількості записів. Після перетворення у формат JSON інформація передається до бібліотеки Chart.js для побудови графіка.

Для побудови кругових діаграм реалізовано алгоритм розрахунку структури візитів за статусами. Система визначає загальну кількість записів у вибраному часовому проміжку та окремо підраховує кількість записів зі статусами “виконано”, “заплановано” та “скасовано”. Після цього для кожного статусу визначається відсоткове співвідношення.

Формула розрахунку частки записів (2.1) має вигляд:

$$S_i = \frac{n_i}{N} \cdot 100\%, \tag{2.1}$$

де n_i – кількість записів певного статусу, а N – загальна кількість записів за обраний період.

Отримані значення використовуються для побудови кругової діаграми, що дозволяє оцінити кількість виконаних або скасованих прийомів.

Для аналізу якості роботи персоналу реалізовано алгоритм автоматичного розрахунку рейтингу лікарів. Після кожного нового відгуку система перераховує середню оцінку на основі всіх записів у таблиці reviews. Розрахунок виконується методом середнього арифметичного.

Формула визначення середнього рейтингу лікаря (2.2) має вигляд:

$$R_{avg} = \frac{\sum_{i=1}^{n} r_i}{n}, \tag{2.2}$$

де r_i – оцінка окремого відгуку, а n – кількість усіх відгуків про лікаря.

Результат розрахунку автоматично відображається в аналітичному модулі директора поруч із інформацією про лікаря.

Для запобігання помилкам під час створення записів у системі реалізовано алгоритм перевірки доступності лікаря. Перед додаванням нового запису система перевіряє, чи не перебуває лікар у відпустці та чи не існує вже запису на той самий час. Якщо хоча б одна з перевірок завершується негативно, створення нового запису блокується. Це дозволяє уникнути дублювання прийомів та помилок у розкладі [3].

Окремо реалізовано метод імпорту та експорту даних. Система дозволяє формувати SQL-дампи або CSV-файли для резервного копіювання та подальшого аналізу інформації у сторонніх програмах. Такий підхід спрощує перенесення бази даних і забезпечує можливість збереження інформації у разі технічних збоїв.

Алгоритм формування аналітичного звіту наведено на рисунку 2.4. Блок-схема демонструє послідовність обробки даних – від введення параметрів пошуку до відображення графіка в аналітичному модулі системи.

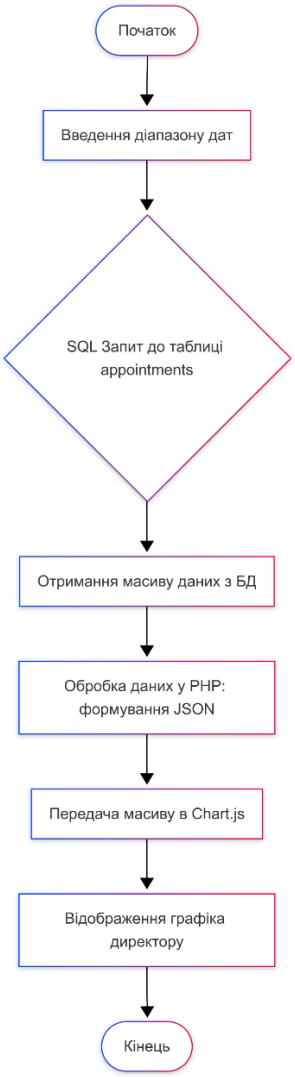

Рисунок 2.4 – Алгоритм формування аналітичного звіту

Представлений алгоритм демонструє послідовність автоматизованої обробки інформації у системі. Використання фільтрації, агрегації та візуалізації даних дозволяє швидко отримувати статистичну інформацію та спрощує аналіз роботи медичного закладу.

2.5 Проектування архітектури інтерфейсу користувача (UI/UX) для різних ролей доступу

Проектування інтерфейсу користувача та взаємодії з системою базується на розподілі функцій відповідно до ролей користувачів і спрощенні виконання щоденних операцій. Основна увага приділена швидкому доступу до ключових

функцій, мінімальній кількості переходів між сторінками та зрозумілій структурі елементів інтерфейсу [28]. Інтерфейс побудовано за принципом єдиної навігаційної панелі, яка змінюється залежно від ролі користувача після авторизації. Це дозволяє кожному користувачу бачити лише необхідний для його роботи функціонал без перевантаження зайвими елементами системи.

Інтерфейс директора призначений для контролю та аналізу діяльності медичного закладу. Основні розділи адміністративного модуля включають управління персоналом, перегляд статистики та роботу з аналітичними даними. У панелі керування доступні функції перегляду відгуків пацієнтів, редагування інформації про лікарів і реєстраторів, а також перегляд графіків завантаженості персоналу та відділень. Окремий блок містить аналітичні дані, сформовані на основі записів на прийом і відгуків пацієнтів. Також реалізовано модуль внутрішніх повідомлень та нарад, який дозволяє створювати події для персоналу та контролювати їх виконання. Додатково передбачено доступ до функцій експорту та імпорту інформації з бази даних.

Інтерфейс реєстратора орієнтований на роботу з пацієнтами та записами на прийом. Основні функції цього модуля зосереджені на швидкому створенні нових записів і пошуку інформації про пацієнтів. Реєстратор має доступ до графіків роботи лікарів, може створювати записи на прийом та перевіряти наявність вільного часу у розкладі лікаря. Також реалізовано функцію реєстрації нових пацієнтів у системі та оновлення базових даних щодо існуючих записів. Інтерфейс побудований таким чином, щоб мінімізувати кількість дій під час обслуговування одного відвідувача та прискорити роботу реєстратури.

Інтерфейс лікаря призначений для роботи з медичними даними пацієнтів та ведення електронної документації. Основна увага приділена швидкому доступу до персонального розкладу прийомів і електронних медичних карт. Лікар може переглядати список пацієнтів на поточний день, відкривати їхні медичні записи та вносити результати прийому. Також реалізовано можливість завантаження результатів аналізів, медичних висновків та інших документів у

систему. Окремий розділ містить повідомлення про внутрішні наради та зміни у роботі медичного закладу.

Інтерфейс пацієнта спрощений та орієнтований на самостійне використання системи без необхідності постійного звернення до реєстратури. Основними функціями особистого кабінету є запис на прийом до лікаря, перегляд історії відвідувань та доступ до результатів аналізів і медичних висновків. Пацієнт може переглядати інформацію про лікарів, ознайомлюватися з доступними медичними послугами та залишати відгуки після проходження прийому. Це забезпечує зручність взаємодії із закладом та підвищує рівень доступності медичних сервісів.

Для всіх ролей у системі реалізовано спільні елементи інтерфейсу, зокрема кнопку виходу із системи та індикатори стану виконання процесів. Інтерфейс адаптовано для роботи на різних типах пристроїв, включаючи настільні комп’ютери, планшети та мобільні телефони, що забезпечує доступ до системи незалежно від місця перебування користувача.

2.6 Висновки до другого розділу

У другому розділі виконано проектування архітектури та основних компонентів інформаційно-аналітичної системи медичного закладу. На основі аналізу вимог сформовано структуру взаємодії між модулями системи та визначено принципи їх роботи [26].

Розроблена структурно-функціональна схема базується на клієнт-серверній архітектурі, у якій серверна частина відповідає за обробку запитів, керування даними та взаємодію з базою даних. Визначено чіткий розподіл функцій між ролями користувачів: директором, реєстратором, лікарем та пацієнтом. Це дозволило розділити логіку системи на незалежні модулі та зменшити взаємні залежності між ними.

Моделювання бізнес-процесів дало змогу описати повний цикл роботи системи – від створення запису на прийом до отримання результатів лікування

та формування аналітичних звітів. Побудовані діаграми дозволили визначити послідовність дій користувачів і структуру обміну даними між компонентами системи, що спростило подальше проектування логіки програмної реалізації.

Проектування бази даних показало доцільність використання реляційної моделі та зв’язків між таблицями через зовнішні ключі. Це забезпечує узгодженість даних між записами пацієнтів, лікарів, візитів та результатів обстежень. Використання індексів за датами та статусами записів дозволяє швидко отримувати аналітичну інформацію, необхідну для побудови графіків і звітів у системі [24].

Окрему увагу приділено проектуванню інтерфейсу користувача. Розроблена структура UI забезпечує розділення доступу за ролями та спрощує взаємодію з системою. Кожен користувач отримує набір функцій, необхідних саме для його роботи, що зменшує кількість зайвих дій та покращує загальну організацію роботи в системі.

Отже, результати другого розділу формують повну основу для подальшої реалізації програмної частини системи. У наступному розділі буде виконано опис програмної реалізації, структури коду та тестування розробленого рішення в умовах наближених до реальних.

РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ ВПРОВАДЖЕННЯ СИСТЕМИ

3.1 Реалізація модулів адміністрування та керування персоналом і послугами клініки

Програмна реалізація адміністративних модулів системи виконана із використанням серверної мови PHP та системи керування базами даних MySQL. Основна логіка модулів побудована на взаємодії користувацьких форм із таблицями бази даних через SQL-запити та PHP-обробники [12].

Одним із ключових компонентів системи є модуль керування персоналом та розкладом. Для цього реалізовано інтерфейс роботи з таблицею doctors, який дозволяє додавати нових лікарів, редагувати спеціалізації, змінювати відділення та налаштовувати графік роботи. Окремо реалізовано механізм контролю відпусток. Перед створенням нового запису система автоматично перевіряє поля vacation_start та vacation_end. Якщо обрана дата потрапляє у період відпустки лікаря, створення запису блокується. Це дозволяє уникнути помилкового запису пацієнтів у недоступний час. Інтерфейс модуля керування персоналом наведено на рисунку 3.1. Програмний код для автоматичного налаштування робочих графіків лікарів та їх синхронізації з календарем системи представлено у вигляді готового сценарію auto_schedule.php у додатку Б (лістинг Б.1).

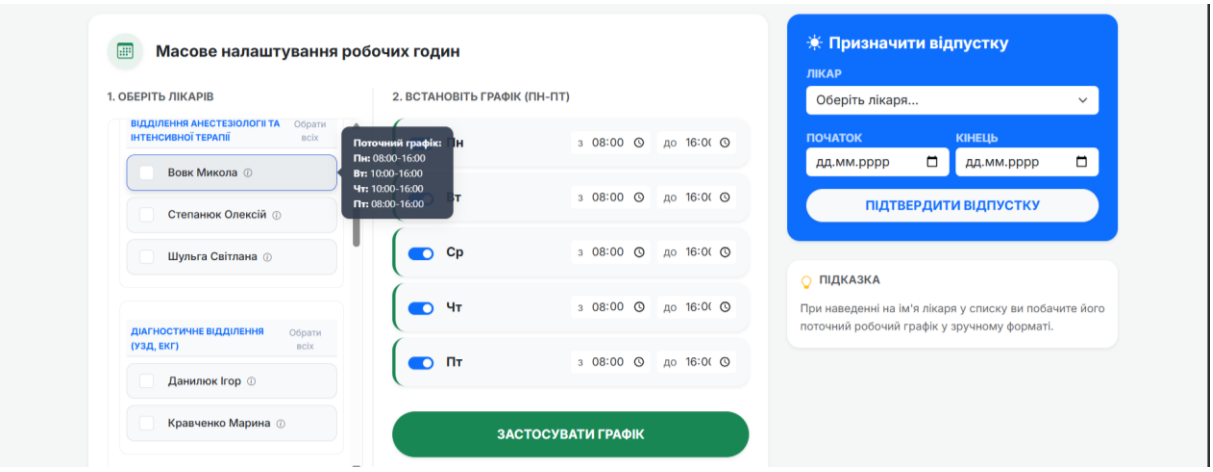

Рисунок 3.1 – Модуль керування персоналом та спеціалізаціями

Для організації внутрішньої взаємодії між працівниками реалізовано модуль нарад та внутрішніх повідомлень. Після заповнення форми директором PHP-обробник приймає введені дані та зберігає їх у таблиці meetings. Після цього інформація автоматично відображається у кабінетах лікарів. Такий підхід дозволяє організувати централізоване інформування персоналу без використання сторонніх сервісів. Реалізація інтерфейсу створення нарад наведена на рисунку 3.2. Алгоритм створення нарад та збереження їх тем у базі даних міститься у файлі save_meeting.php (додаток Б). Структуру SQL-запиту, який виводить активні збори для лікарів у реальному часі, наведено в додатку В (лістинг В.4).

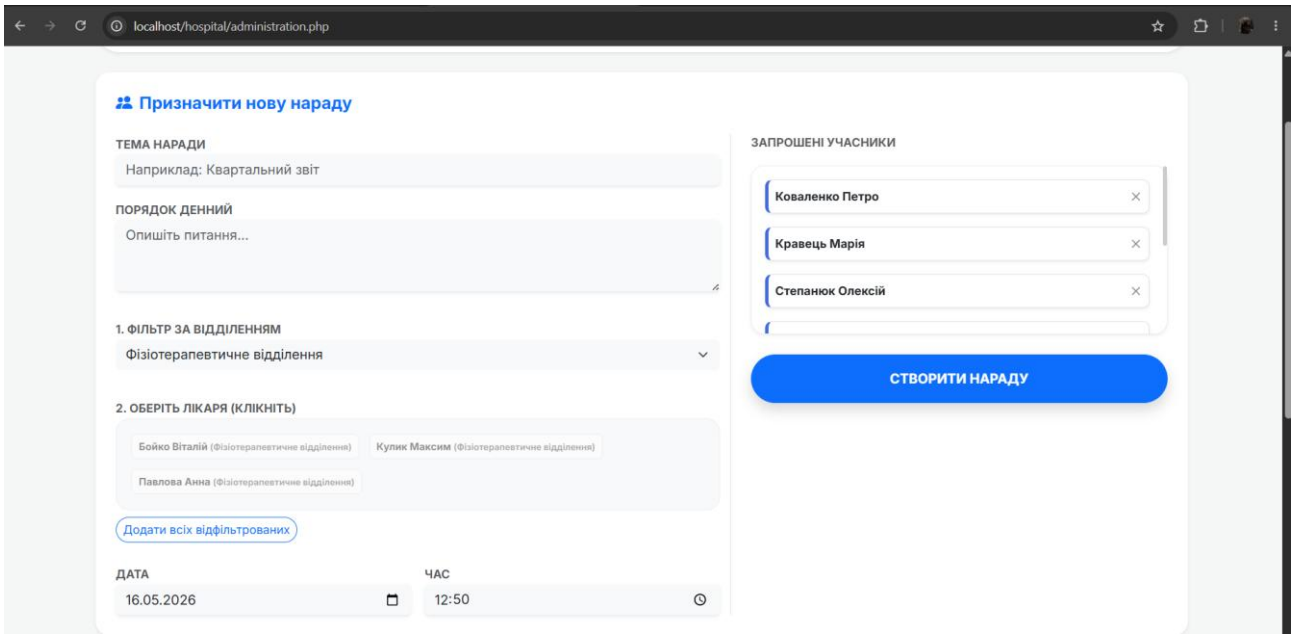

Рисунок 3.2 – Реалізація системи внутрішніх нарад у кабінеті директора

Окремим компонентом системи є модуль моніторингу репутації та відгуків. У цьому блоці система автоматично виконує обробку оцінок пацієнтів та формує середній рейтинг лікарів на основі записів таблиці reviews. Директор має можливість переглядати всі коментарі пацієнтів та аналізувати інформацію щодо роботи персоналу. Для відображення статистики використовуються графіки та інформаційні блоки, сформовані за допомогою Chart.js. Приклад роботи модуля наведено на рисунку 3.3. Розрахунок середньої оцінки та

формування рейтингу лікарів автоматично виконується на стороні бази даних. Оптимізований SQL-запит із використанням функції математичного усереднення AVG() наведено у додатку В (лістинг В.2).

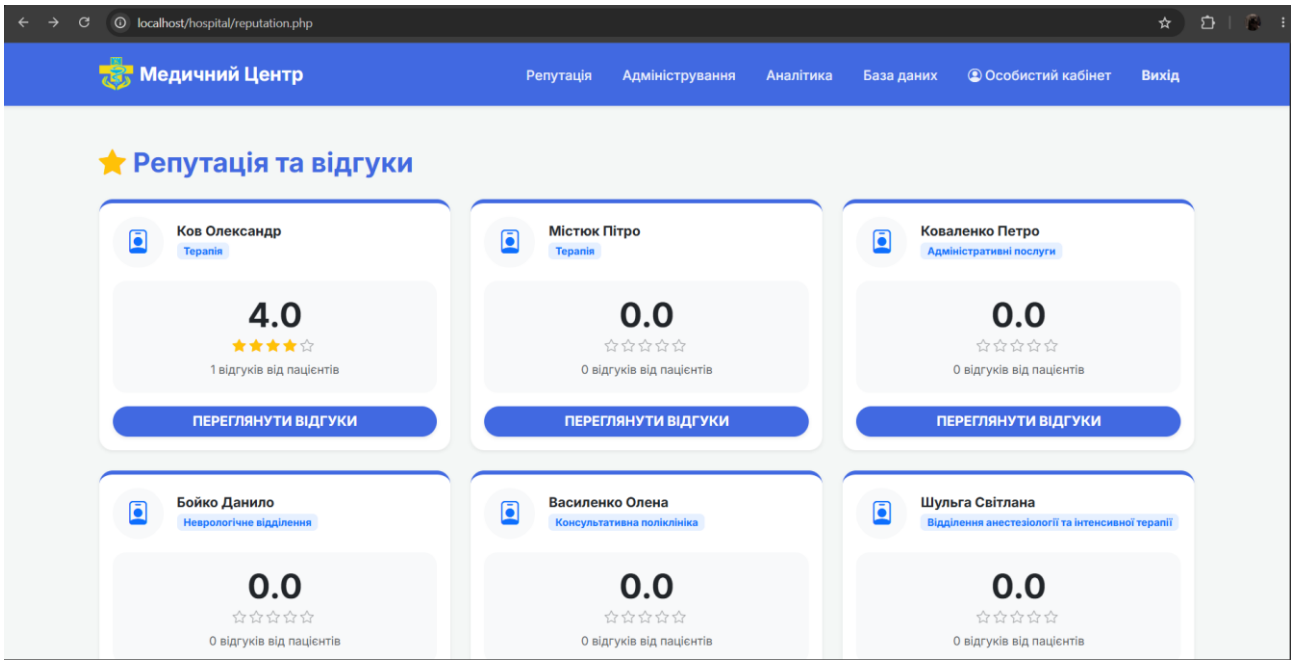

Рисунок 3.3 – Інтерфейс аналізу репутації та загальних показників закладу

Для роботи з резервними копіями та технічним обслуговуванням системи реалізовано модуль адміністрування бази даних. Через цей модуль директор може виконувати експорт таблиць у SQL-формат та імпорт даних із зовнішніх файлів. Це дозволяє створювати резервні копії бази даних та переносити інформацію між системами. Усі операції виконуються через PHP-сценарії, які взаємодіють із сервером бази даних. Інтерфейс технічного модуля представлено на рисунку 3.4. Програмні інструменти для експорту великих масивів даних у формат CSV та коректного збереження українського тексту реалізовано за допомогою спеціальних сценаріїв, базові компоненти яких містяться у додатку В.

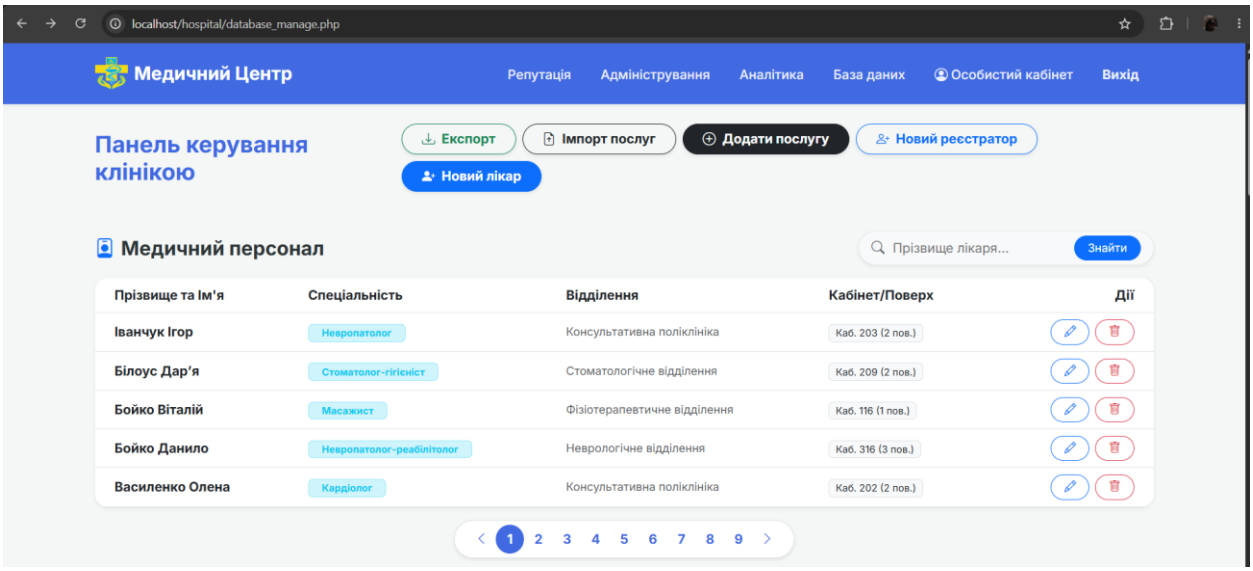

Рисунок 3.4 – Модуль адміністрування, експорту та імпорту бази даних

Для працівників реєстратури реалізовано окремий операційний модуль із спрощеним інтерфейсом. Основна увага приділена швидкому пошуку пацієнтів та створенню записів на прийом. Інтерфейс містить форму реєстрації нових пацієнтів, систему перевірки доступності лікаря та інструменти перегляду розкладу. Це дозволяє мінімізувати час обслуговування відвідувачів та спрощує роботу реєстратора. Робоче місце реєстратури наведено на рисунку 3.5.

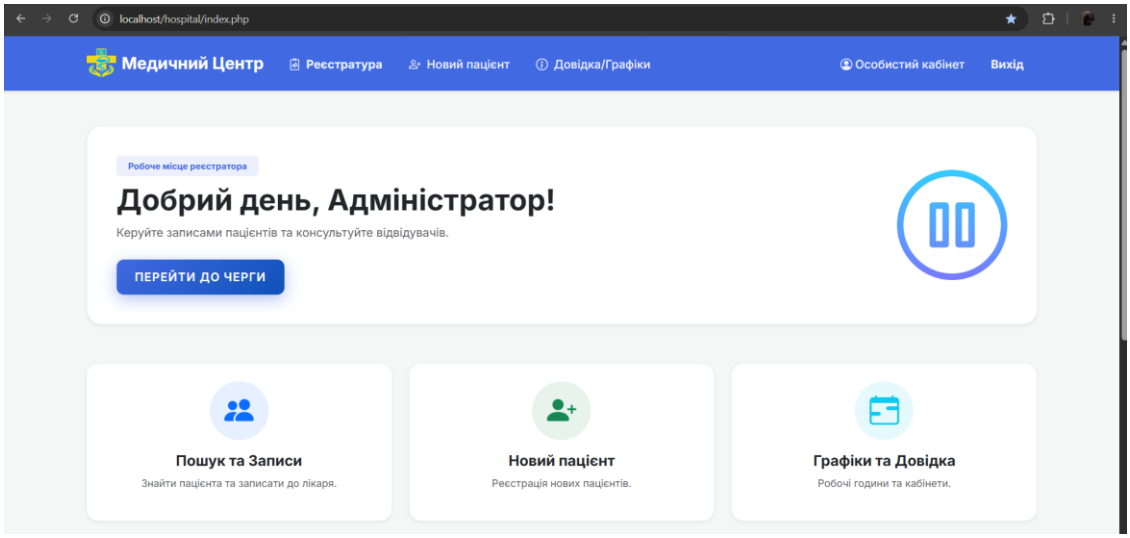

Рисунок 3.5 – Робоче місце реєстратора: інтерфейс пошуку та запису пацієнтів

Реалізовані адміністративні модулі забезпечують централізоване керування персоналом, записами пацієнтів та внутрішніми процесами клініки.

Використання PHP та MySQL дозволило організувати взаємодію між інтерфейсом користувача та базою даних і забезпечити обробку інформації в режимі реального часу [11].

3.2. Розробка підсистеми реєстрації пацієнтів та планування медичних візитів

Підсистема реєстрації пацієнтів та планування медичних візитів реалізована як окремий функціональний модуль, що забезпечує взаємодію між пацієнтом, реєстратором та лікарем. Основне призначення підсистеми полягає у створенні записів на прийом, перевірці доступності лікарів та збереженні інформації у таблиці appointments бази даних [21].

Для реєстрації нових пацієнтів у системі реалізовано вебформу, через яку вводяться персональні дані користувача: прізвище та ім’я, номер телефону, дата народження та адреса проживання. Після відправлення форми PHP-обробник виконує перевірку коректності введених даних, зокрема перевірку заповнення обов’язкових полів і формату номера телефону. Після успішної перевірки формується SQL-запит типу INSERT INTO patients, який додає нового пацієнта до бази даних. Інтерфейс додавання нового пацієнта наведено на рисунку 3.6.

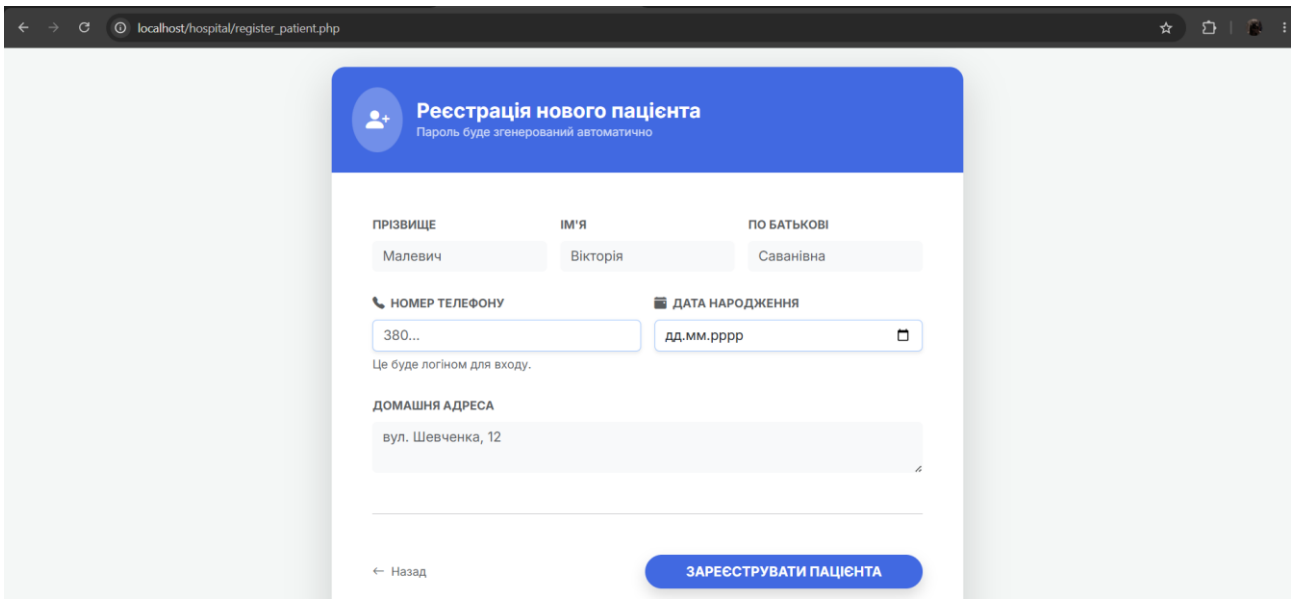

Рисунок 3.6 – Форма реєстрації нового пацієнта в інформаційній системі

Створення запису на прийом реалізовано у вигляді послідовного алгоритму перевірки та збереження інформації. Після вибору лікаря та послуги система отримує інформацію з таблиць doctors та hospital_services. Далі PHP-контролер перевіряє, чи не існує вже запису на той самий час у вибраного лікаря. Для цього використовується SQL-запит із перевіркою поля appointment_time.

Якщо обраний часовий проміжок є вільним, система створює новий запис у таблиці appointments зі статусом planned. У випадку конфлікту користувач отримує повідомлення про недоступність обраного часу. Це дозволяє уникнути дублювання записів та помилок у графіку лікарів [13]. Інтерфейс створення запису на прийом наведено на рисунку 3.7. Алгоритм генерації сітки доступного часу прийому та перевірки слотів на відсутність накладання розкладів реалізовано у модулі appointment.php, код якого детально описано у додатку Б (лістинг Б.2).

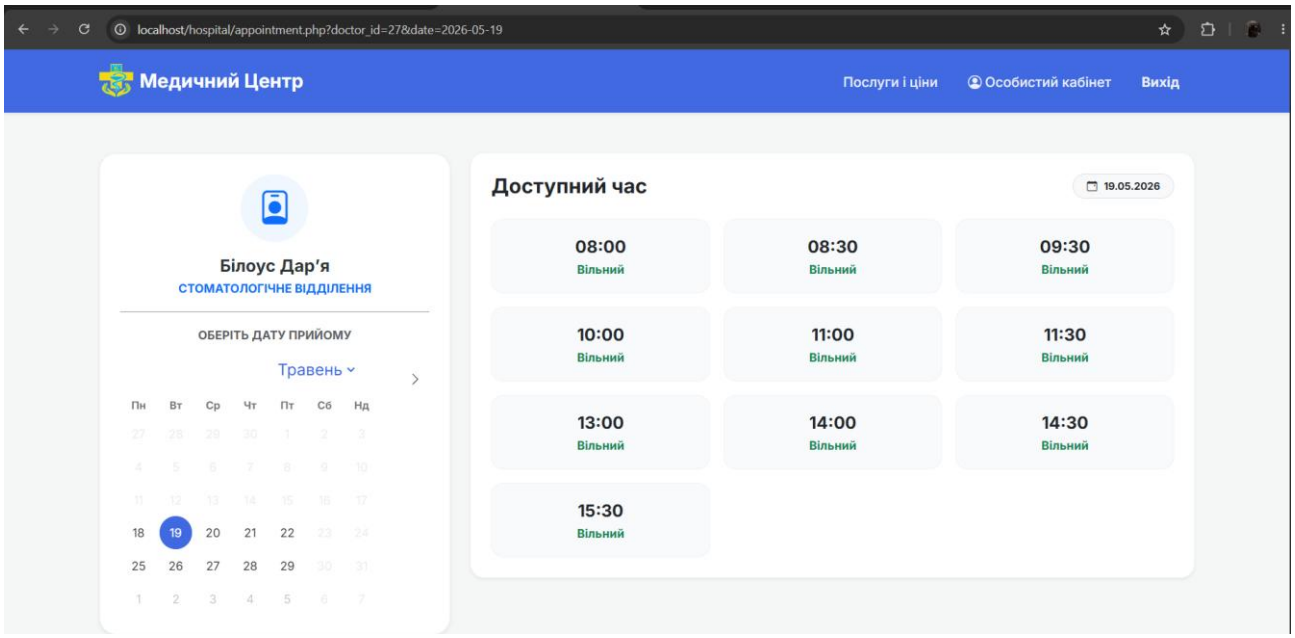

Рисунок 3.7 – Інтерфейс оформлення запису пацієнта до спеціаліста

Окремо реалізовано модуль самостійного запису пацієнта через особистий кабінет. Після авторизації система автоматично отримує дані пацієнта із сесії користувача через змінну $_SESSION['user_id'], що дозволяє уникнути

повторного введення персональної інформації. Пацієнту необхідно лише обрати лікаря, дату та доступний час прийому.

Такий підхід спрощує взаємодію користувача із системою та зменшує навантаження на реєстратуру. Інтерфейс особистого кабінету пацієнта представлено на рисунку 3.8.

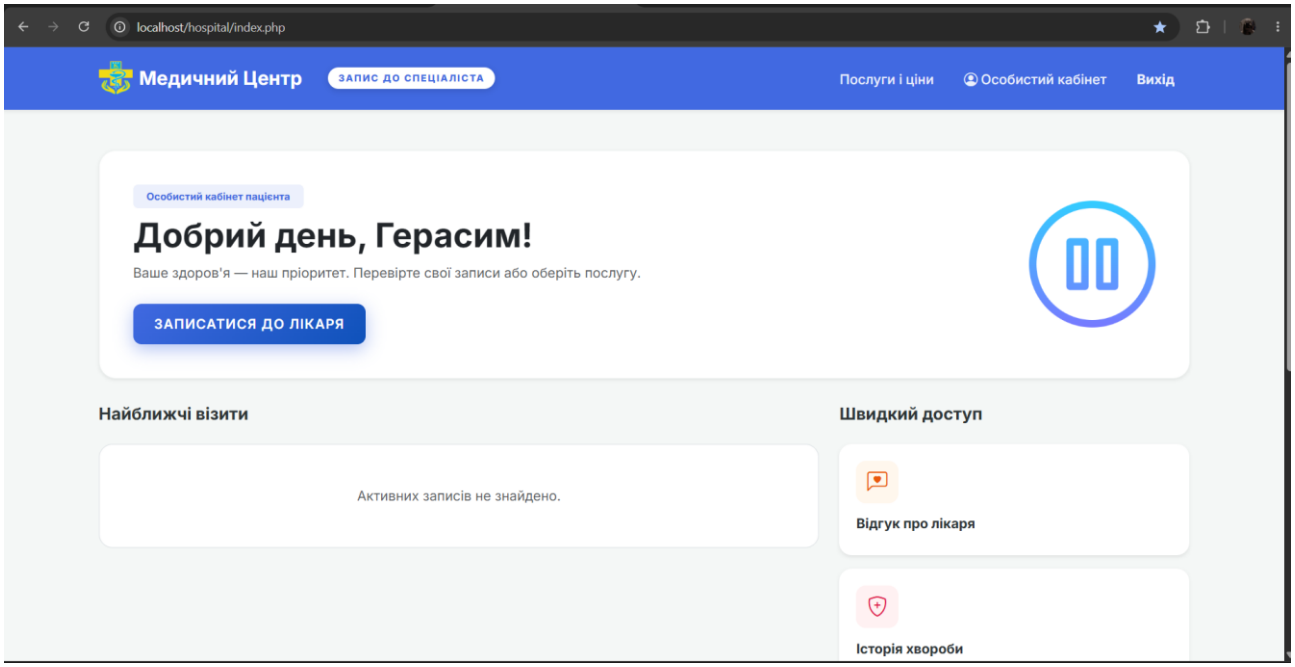

Рисунок 3.8 – Особистий кабінет пацієнта: функціонал онлайн-запису

Після створення запису інформація автоматично відображається у журналі прийомів лікаря. Для кожного запису система підтримує зміну статусу візиту: “заплановано”, “виконано” або “скасовано”. Зміна статусу виконується через PHP-обробники та SQL-запити типу UPDATE appointments.

Після встановлення статусу «виконано» лікар отримує можливість додати медичний висновок або результати аналізів пацієнта. Ці дані автоматично стають доступними у кабінеті пацієнта. Робочий журнал лікаря наведено на рисунку 3.9.

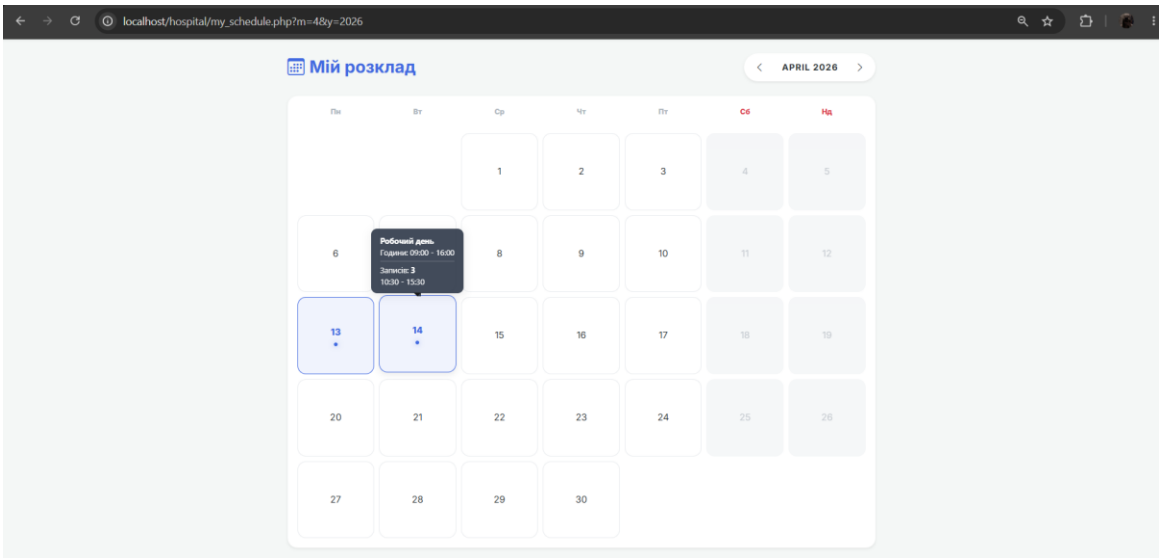

Рисунок 3.9 – Журнал прийомів лікаря з відображенням статусів візитів

Реалізована підсистема забезпечує автоматизацію процесів реєстрації пацієнтів та планування прийомів, дозволяє уникати конфліктів у розкладі та забезпечує синхронізацію інформації між усіма користувачами системи.

3.3 Програмна реалізація аналітичного модуля візуалізації показників ефективності закладу

Аналітичний модуль реалізовано як окремий компонент інформаційної системи, призначений для збору, обробки та візуалізації статистичних даних медичного закладу. Основою роботи модуля є взаємодія серверної частини на PHP із базою даних MySQL та бібліотекою Chart.js, яка використовується для побудови інтерактивних графіків і діаграм [34].

Для забезпечення комплексного аналізу діяльності клініки реалізовано механізм збору статистики за визначений часовий проміжок. Користувач задає початкову та кінцеву дати звітного періоду, після чого система виконує SQL-запит до таблиці appointments. Для групування записів за днями використовується функція DATE_FORMAT, що дозволяє формувати часові ряди для побудови лінійних графіків.

Після виконання запиту PHP-скрипт формує масиви даних і конвертує їх у формат JSON. На стороні клієнта ці дані передаються у Chart.js для автоматичного оновлення графіків без перезавантаження сторінки. Такий підхід відповідає принципам побудови динамічних вебзастосунків [30].

Для аналізу структури прийомів реалізовано кругову діаграму, яка відображає співвідношення статусів візитів: planned, completed та cancelled. Система автоматично підраховує кількість записів для кожного статусу та формує набір значень для побудови Pie Chart.

Окремо реалізовано механізм автоматичного розрахунку рейтингу лікарів. Для цього використовується SQL-функція AVG() над оцінками пацієнтів у таблиці reviews. Після відкриття аналітичної панелі система автоматично виконує перерахунок середнього балу кожного лікаря та відображає результат у вигляді рейтингових карток [29]. Збір аналітичної інформації та її підготовка для головного дашборду директора виконується сервером за допомогою скрипта analytics.php (додаток А, лістинг А.1). Код побудови інтерактивних діаграм у браузері за допомогою мови JavaScript наведено в лістингу А.2 додатка А.

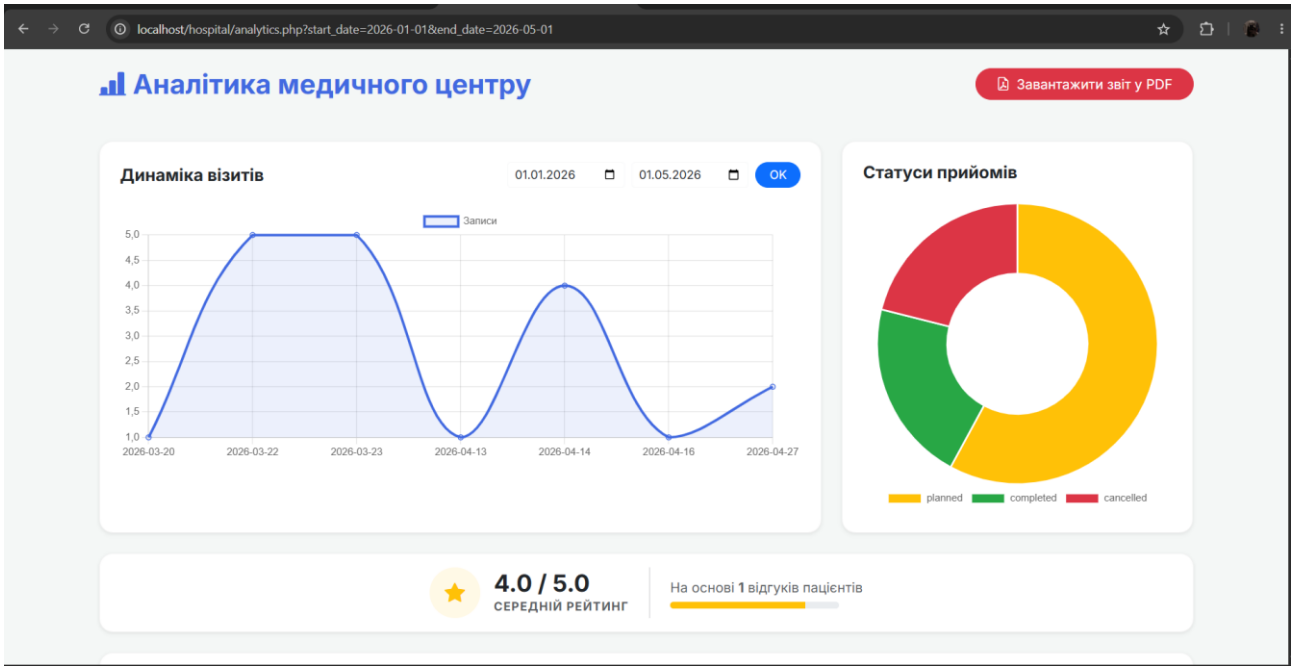

Рисунок 3.10 – Комплексна панель аналітики: динаміка записів, структура статусів та рейтинги лікарів

Для аналізу навантаження на персонал реалізовано модуль порівняльної статистики лікарів. Алгоритм системи виконує групування записів за полем doctor_id та підраховує кількість прийомів для кожного лікаря.

Після обробки даних результати передаються у Chart.js для побудови стовпчикового графіка типу bar. Такий підхід дозволяє порівнювати інтенсивність роботи різних спеціалістів та аналізувати розподіл навантаження між відділеннями медичного закладу [5]. Базовий SQL-запит для згрупованої вибірки даних і аналізу завантаженості медичних відділень представлено у додатку В (лістинг В.1). Алгоритм, який автоматично формує друковані звіти та шкали ефективності у форматі PDF, наведено в додатку А (лістинг А.3).

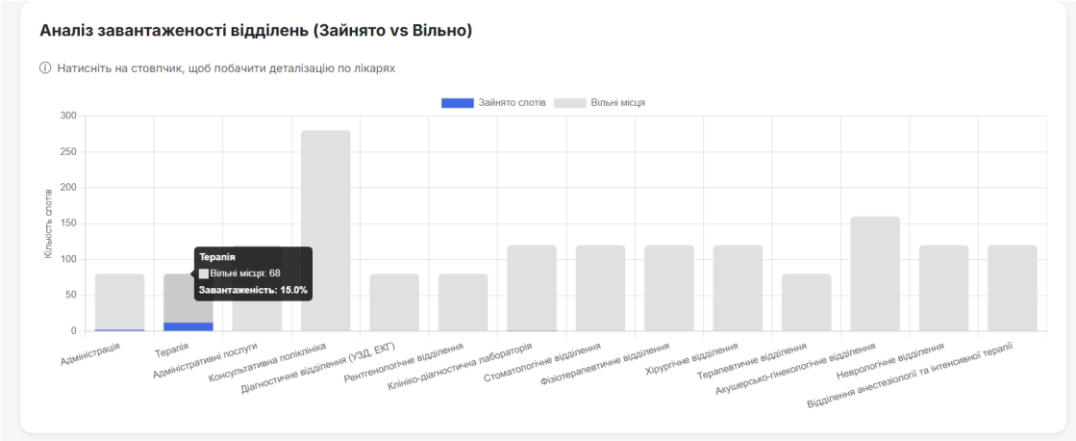

Рисунок 3.11 – Стовпчиковий графік порівняльного аналізу завантаженості лікарів закладу

Усі елементи аналітичного модуля інтегровані із системою фільтрації даних. При зміні часового діапазону система автоматично повторно виконує SQL-запити та оновлює статистику у графіках і таблицях. Це відповідає підходам побудови інтерактивних вебсистем.

Додатково реалізовано функцію експорту звітів у формат CSV. PHP-скрипти формують файл на основі поточної вибірки з бази даних, що дозволяє подальше використання даних у табличних процесорах або для архівування.

3.4 Реалізація функціональних модулів лікаря, пацієнта та інформаційних сервісів

Цей підрозділ описує програмну реалізацію клінічного блоку системи та сервісів самообслуговування, які забезпечують повний цикл взаємодії користувача з медичним закладом – від авторизації до отримання результатів лікування.

Інтерфейс лікаря спроектовано з орієнтацією на швидкий доступ до медичних даних та мінімізацію адміністративних дій під час прийому пацієнтів. Основна взаємодія реалізована через роботу з таблицями пацієнтів і записів на прийом, що дозволяє лікарю переглядати історію відвідувань та додавати нові медичні записи. На рисунку 3.13 зображено робоче місце лікаря з доступом до електронної медичної карти та архіву пацієнтів.

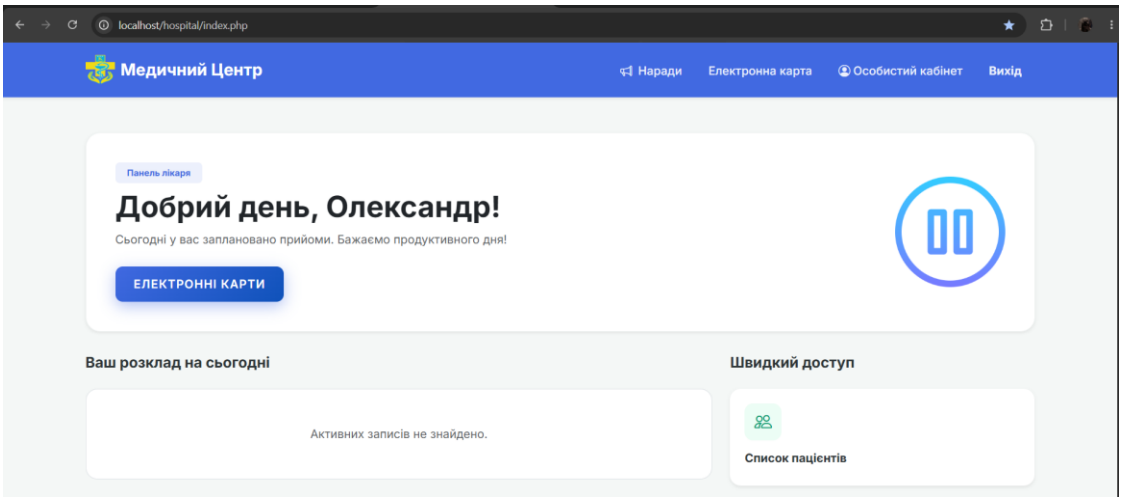

Рисунок 3.13 – Інтерфейс лікаря: електронна медична карта та архів пацієнтів

Завантаження медичних файлів реалізовано через серверну обробку PHP, яка перевіряє формат файлів, зберігає їх на сервері та фіксує шлях у базі даних. Це дозволяє організувати контрольований доступ до медичної документації та забезпечує зв’язок між пацієнтом і його медичними результатами [8].

Особистий кабінет пацієнта є основним інтерфейсом взаємодії користувача із системою. Після авторизації система ідентифікує користувача

через user_id, що дозволяє відображати персональні дані, історію візитів та результати лікування.

Модуль відгуків реалізовано через форму оцінювання, яка зберігає дані у таблиці відгуків. Після додавання нового запису система автоматично перераховує середній рейтинг лікаря, який використовується в аналітичному модулі керівника. На рисунку 3.14 зображено інтерфейс пацієнта з функціями перегляду історії лікування та системою зворотного зв'язку. Щоб обмежити можливість залишати неактуальні відгуки, система перевіряє дату прийому. SQL-запит, який вираховує дозволений ліміт у 48 годин за допомогою функції DATEDIFF(), наведено у додатку В (лістинг В.3).

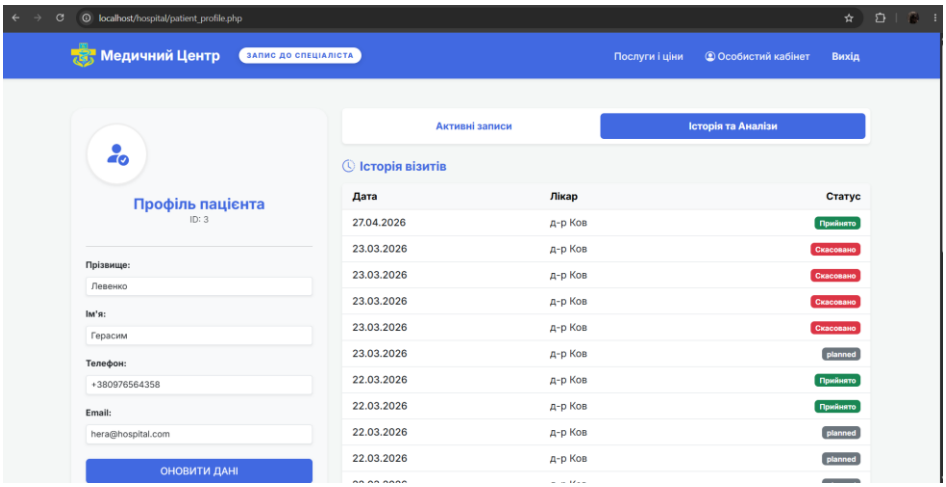

Рисунок 3.14 – Особистий кабінет пацієнта: історія лікування та відгуки

Додатково реалізовано інформаційні сервіси, які включають перегляд переліку медичних послуг та графіків роботи лікарів. Дані про послуги формуються на основі довідника, що дозволяє пацієнту ознайомитися з вартістю та типами процедур перед записом. Графік роботи лікарів формується з урахуванням їхнього статусу присутності, що дозволяє уникати конфліктів при плануванні візитів. Цей функціонал відображено на рисунку 3.15. Для безпечного входу в кабінети реалізовано модуль автентифікації на основі перевірки паролів password_verify (додаток Г, лістинг Г.2). Скрипт відновлення

доступу за допомогою відправки одноразових кодів на email описано в лістингу Г.3 додатка Г.

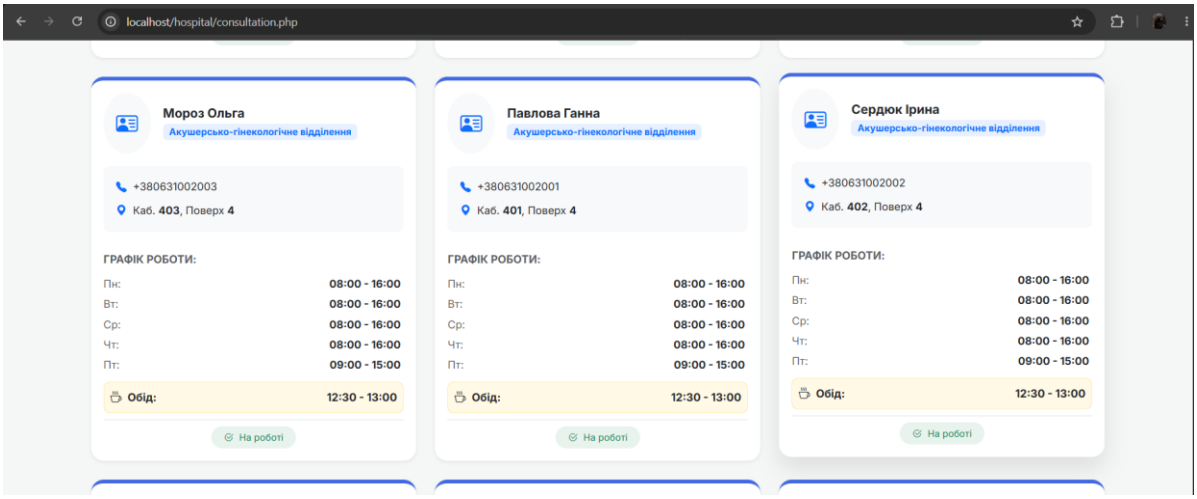

Рисунок 3.15 – Інформаційний модуль: послуги та графіки роботи лікарів

Окремим компонентом системи є модуль внутрішніх повідомлень про наради. Він працює за принципом вибірки даних із таблиці зустрічей, де перевіряється участь конкретного лікаря. У разі збігу система відображає інформацію про тему та час проведення наради, що забезпечує внутрішню комунікацію між адміністрацією та персоналом [16].

3.5 Тестування системи та аналіз переваг її впровадження

Завершальним етапом розробки є тестування інформаційно-аналітичної системи та перевірка її працездатності в умовах, наближених до реальної експлуатації. Основною метою тестування було підтвердження коректності роботи всіх функціональних модулів, а також перевірка взаємодії між серверною частиною, базою даних і клієнтським інтерфейсом [25].

Тестування виконувалося на рівні окремих сценаріїв користувачів: директора, реєстратора, лікаря та пацієнта. Перевірялися процеси авторизації, створення та редагування записів, формування медичних даних, а також

відображення аналітичної інформації. Особливу увагу приділено перевірці цілісності даних при одночасному доступі декількох користувачів.

Було встановлено, що система коректно обробляє створення записів на прийом та виключає дублювання часових слотів завдяки перевіркам на рівні SQL-запитів. Це забезпечує стабільність роботи розкладу лікарів навіть при високому навантаженні.

Аналітичні модулі, реалізовані на основі Chart.js, перевірялися шляхом порівняння автоматично згенерованих результатів із контрольними розрахунками. Отримані дані повністю співпали, що підтверджує коректність агрегації інформації з таблиці appointments.

Окремо протестовано модуль збереження та доступу до медичних файлів. Перевірка показала, що завантажені документи коректно зберігаються на сервері та прив’язуються до відповідних записів пацієнтів. Доступ до файлів обмежено відповідно до ролей користувачів, що виключає несанкціонований перегляд даних.

Порівняльний аналіз ручного та автоматизованого підходів до формування звітності показав суттєве скорочення часу отримання аналітики. Формування статистики тепер виконується автоматично без участі адміністративного персоналу.

Розрахунок середнього рейтингу лікарів на основі відгуків пацієнтів дозволяє в режимі реального часу оцінювати якість надання медичних послуг. Відповідні дані оновлюються в адміністративній панелі системи [25].

Модуль внутрішніх нарад забезпечує автоматичне інформування лікарів про заплановані збори, що зменшує ризик втрати інформації та покращує внутрішню комунікацію між адміністрацією та персоналом.

Особистий кабінет пацієнта забезпечує доступ до історії візитів, результатів лікування та функції онлайн-запису.

Окремо проведено стрес-тестування функцій резервного копіювання та відновлення бази даних. Було перевірено створення SQL-дампу та його відновлення на резервному сервері, що підтвердило готовність системи до

міграції та аварійного відновлення даних. Процес успішної верифікації адміністративних дій та підтвердження цілісності даних при експорті зафіксовано на рисунку 3.16. Результати тестування безпеки підтвердили надійний захист даних від зовнішнього втручання. Безпечне підключення до бази даних та налаштування технології захисту від SQL-ін'єкцій наведено у додатку Г (лістинг Г.1).

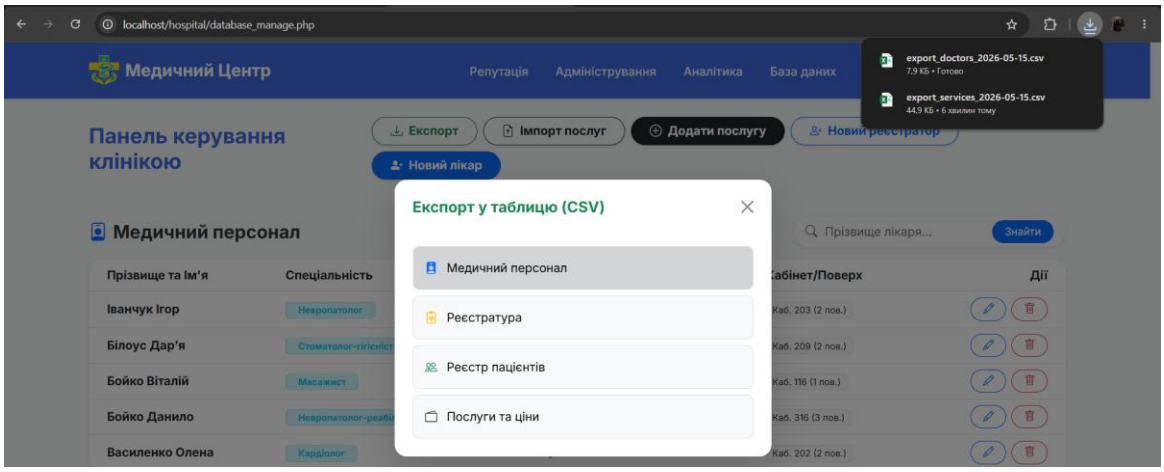

Рисунок 3.16 – Тестування модуля адміністрування та експорту бази даних

За результатами тестування встановлено, що розроблена інформаційно-аналітична система повністю відповідає вимогам технічного завдання, коректно виконує всі основні функції та може бути впроваджена в умовах реального медичного закладу.

3.6 Висновки до третього розділу

У третьому розділі виконано практичну реалізацію інформаційно-аналітичної системи медичного закладу та перевірено її роботу в умовах тестування. Отримані результати підтверджують коректність реалізації основних функціональних модулів.

Реалізовано серверну частину системи на мові PHP із взаємодією з реляційною базою даних MySQL. Забезпечено роботу з основними таблицями

системи, що дозволило об’єднати адміністративні та медичні дані в єдину структуру та забезпечити їх узгоджену обробку.

Розроблено аналітичний модуль на основі Chart.js, який забезпечує візуалізацію основних показників роботи закладу. Реалізовано формування графіків завантаженості, структури візитів та розрахунок середнього рейтингу лікарів на основі відгуків пацієнтів.

Створено функціональні модулі для основних ролей користувачів. Забезпечено підтримку процесів реєстрації пацієнтів, онлайн-запису, ведення медичних записів, перегляду результатів обстежень, а також внутрішніх повідомлень між персоналом.

Проведене тестування підтвердило коректність роботи алгоритмів обробки даних, відсутність критичних помилок у взаємодії модулів та стабільність роботи системи при виконанні операцій з базою даних, включаючи резервне копіювання та відновлення.

У результаті впровадження системи досягається скорочення часу на виконання адміністративних операцій, автоматизація формування звітності та покращення доступу до медичної інформації для всіх категорій користувачів.

Таким чином, розроблена інформаційно-аналітична система відповідає вимогам технічного завдання та може бути використана в умовах реального медичного закладу.

РОЗДІЛ 4. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Вплив архітектури інтерфейсу інформаційної системи на психофізіологічний стан та безпеку праці медичного персоналу

У сучасних умовах цифровізації медицини інтерфейс користувача інформаційної системи стає невід'ємною частиною робочого середовища медичного персоналу. Лікарі, реєстратори та адміністратори щоденно взаємодіють із програмним забезпеченням, тому якість його архітектури безпосередньо впливає на продуктивність праці, рівень стресу та загальний психофізіологічний стан користувачів. Неправильно спроектований інтерфейс може спричиняти перевантаження уваги, зорову втому та зниження швидкості прийняття рішень, що є критичним у медичній сфері. На рисунку 4.1 показано вплив колірної гами інтерфейсу на емоційний стан користувача.

Рисунок 4.1 – Психологічний вплив колірної гами інтерфейсу на емоційний стан оператора [7]

При розробці системи було обрано спокійну синьо-білу кольорову гаму. Такий вибір обґрунтований тим, що синій колір у психології сприймається як

стабільний і заспокійливий, він асоціюється з довірою, безпекою та концентрацією. Це особливо важливо в умовах медичного середовища, де користувач працює під час високого навантаження та обмеженого часу на прийняття рішень. Білий фон у поєднанні з темним текстом забезпечує високий рівень контрастності, що покращує читабельність інформації та зменшує навантаження на зоровий апарат.

Додатково важливим є дотримання ергономічних вимог до візуального відображення інформації. Надмірна кількість яскравих елементів або неконтрастних кольорів може викликати швидку втому очей та зниження концентрації уваги. Тому інтерфейс системи розроблено з урахуванням принципів візуальної простоти та мінімалізму.

Складна багаторівнева структура меню часто є причиною так званого “техностресу”, коли користувач витрачає більше часу на пошук функцій, ніж на виконання основної роботи. Для вирішення цієї проблеми у системі реалізовано принцип мінімальної кількості кліків та спрощену навігаційну структуру на основі карткового (блочного) дизайну. Такий підхід дозволяє групувати функції за логічними категоріями та забезпечує швидкий доступ до найважливіших операцій. На рисунку 4.2 наведено порівняння впливу складного та оптимізованого інтерфейсу на когнітивний ресурс користувача.

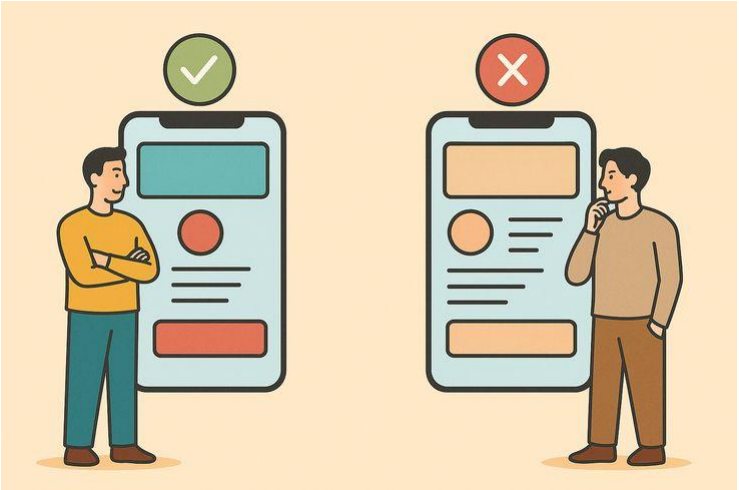

Рисунок 4.2 – Порівняння впливу складного та оптимізованого інтерфейсу на когнітивний ресурс користувача [6]

Використання спрощеної структури інтерфейсу зменшує навантаження на короткочасну пам’ять користувача. Це дозволяє медичному персоналу швидше орієнтуватися в системі та зменшує кількість помилок під час введення або пошуку інформації. Відповідно до принципів когнітивної ергономіки, зменшення кількості одночасно оброблюваних елементів підвищує ефективність роботи та знижує ризик інформаційного перевантаження.

Робота за комп’ютером у медичних інформаційних системах також пов’язана з ризиком розвитку комп’ютерного зорового синдрому та інших професійних захворювань, зокрема порушень зору та опорно-рухового апарату. Для мінімізації цих ризиків у системі застосовано чітку структуру розміщення елементів, оптимальні розміри шрифтів та раціональне використання простору інтерфейсу [33]. На рисунку 4.3 наведено вплив параметрів екрана та інтерфейсу на втомлюваність очей під час роботи.

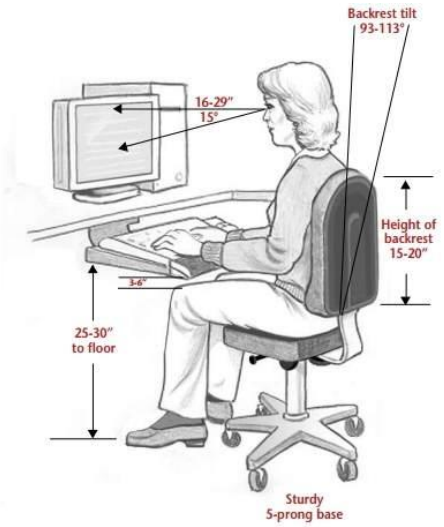

Рисунок 4.3 – Вплив параметрів екрана та інтерфейсу на втомлюваність очей під час роботи [31]

Окрему увагу слід приділити швидкодії системи, оскільки затримки у відгуку інтерфейсу негативно впливають на психологічний стан користувача. Повільна робота програмного забезпечення викликає фрустрацію, підвищує рівень стресу та може призводити до помилок у роботі, особливо в умовах великої кількості пацієнтів. Тому оптимізація SQL-запитів та використання

легких серверних і клієнтських рішень є важливим фактором забезпечення стабільної роботи системи.

Також важливим аспектом є зменшення когнітивного навантаження через логічну організацію інформації. Групування функцій за ролями користувачів (лікар, реєстратор, адміністратор, пацієнт) дозволяє уникнути перевантаження інтерфейсу зайвими елементами та підвищує зручність роботи.

Таким чином, можна зробити висновок, що правильно спроектована архітектура інтерфейсу інформаційної системи позитивно впливає на психофізіологічний стан медичного персоналу, знижує рівень стресу, мінімізує ризик професійних помилок та підвищує загальну ефективність роботи медичного закладу. Це підтверджує важливість врахування ергономічних та психологічних аспектів при розробці сучасних медичних інформаційних систем.

4.2 Ергономічні вимоги до організації робочого середовища розробника та користувачів аналітичних систем

Ефективність функціонування інформаційно-аналітичної системи залежить не лише від програмного забезпечення, а й від умов, у яких працюють розробник та кінцеві користувачі системи – лікарі, реєстратори та адміністративний персонал. Організація робочого місця повинна відповідати вимогам ДСанПіН 3.3.2.007-98 та міжнародним стандартам ергономіки.

Робоче місце користувача аналітичної системи повинно забезпечувати можливість вільної зміни пози під час роботи. Робоче крісло має бути обладнане підйомним механізмом для регулювання висоти сидіння та кута нахилу спинки, що дозволяє знизити статичне навантаження на хребет. Висота робочої поверхні столу повинна бути в межах 680–800 мм, а простір для ніг має забезпечувати комфортне розташування користувача: висота не менше 600 мм та ширина не менше 500 мм [36].

Монітор є основним засобом відображення інформації, тому його неправильне розташування може призводити до перенапруження м’язів шиї та

зорової втоми. Дистанція від очей користувача до екрана повинна становити 600–700 мм, але не менше 500 мм. Верхня межа монітора має знаходитися приблизно на рівні очей, а погляд користувача повинен бути спрямований трохи нижче горизонтальної лінії. Монітор необхідно розміщувати таким чином, щоб уникнути появи відблисків від вікон або освітлювальних приладів.

Для забезпечення стабільної роботи медичного персоналу та серверного обладнання необхідно підтримувати оптимальні параметри виробничого середовища. Рівень освітлення на робочому столі повинен становити 300–500 лк. Найбільш ефективним є поєднання загального та місцевого освітлення. Температура повітря у приміщенні повинна бути в межах 22–25 °C при відносній вологості 40–60 %. Недотримання цих параметрів негативно впливає як на працездатність людини, так і на стабільність роботи комп’ютерної техніки.

Аналітична діяльність пов’язана з високою концентрацією уваги та тривалою роботою за комп’ютером, тому важливе значення має правильна організація режиму праці та відпочинку. Відповідно до норм охорони праці, при восьмигодинному робочому дні рекомендується робити перерви тривалістю 5–10 хвилин через кожну годину роботи за комп’ютером. Під час перерв доцільно виконувати вправи для очей, шиї та кистей рук. На рисунку 4.4 зображено комплекс вправ для профілактики стомлюваності при тривалій роботі з інформаційними системами.

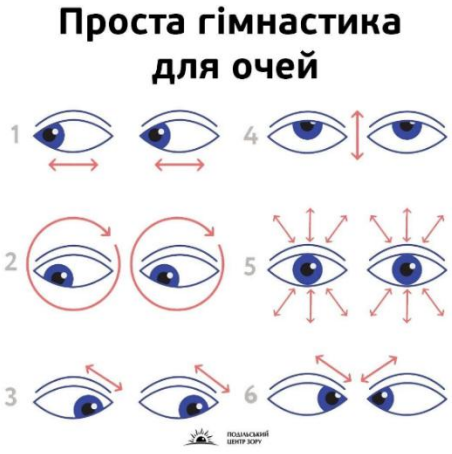

Рисунок 4.4 – Комплекс вправ для профілактики стомлюваності при тривалій роботі з інформаційними системами [38]

Оскільки інформаційно-аналітична система передбачає використання серверного обладнання та декількох робочих станцій, особлива увага приділяється електробезпеці. Усі штепсельні розетки повинні бути заземленими, а кабелі живлення та мережеві дроти мають бути надійно закріплені в кабель-каналах для запобігання механічним пошкодженням і травмуванню персоналу. Доступ до серверного обладнання необхідно обмежити для сторонніх осіб з метою забезпечення безпеки експлуатації та захисту даних.

Таким чином, дотримання ергономічних вимог при організації робочих місць користувачів інформаційної системи дозволяє мінімізувати ризики професійних захворювань, підвищити продуктивність праці медичного персоналу та забезпечити безпечні умови роботи в медичному закладі.

4.3 Висновки до четвертого розділу

У четвертому розділі було розглянуто комплекс питань, пов'язаних із забезпеченням безпеки життєдіяльності та охорони праці при розробці та експлуатації інформаційно-аналітичної системи медичного центру. На основі проведеного аналізу зроблено наступні висновки:

Встановлено, що архітектура інтерфейсу користувача (UI/UX) безпосередньо впливає на психофізіологічний стан медичного персоналу. Використання раціональної колірної гами, мінімізація когнітивного навантаження та забезпечення швидкого відгуку системи є необхідними умовами для запобігання професійному вигоранню та зниження ризику помилок при роботі з пацієнтами.

Визначено ергономічні вимоги до організації робочого місця, що включають правильне налаштування меблів, параметри освітлення та мікроклімату, а також дотримання норм розміщення моніторів. Дотримання цих стандартів дозволяє мінімізувати ризики виникнення комп'ютерного зорового синдрому та захворювань опорно-рухового апарату у розробників та лікарів.

Обґрунтовано необхідність суворого дотримання електробезпеки та впровадження регламентованого режиму праці та відпочинку. Регулярні перерви для виконання гімнастики очей та фізичних вправ є критично важливими для збереження високого рівня працездатності при тривалій роботі з аналітичними даними.

Дотримання розроблених рекомендацій з охорони праці забезпечує створення безпечного та продуктивного середовища, що сприяє ефективному впровадженню та тривалій експлуатації розробленого програмного продукту.

ВИСНОВКИ

У дипломній роботі проведено повний цикл аналізу, проектування, програмної реалізації та тестування інформаційно-аналітичної системи для медичного центру. На основі виконаного дослідження зроблено наступні загальні висновки:

У ході аналізу предметної області встановлено, що сучасні медичні заклади потребують інтегрованих рішень, які поєднують оперативний облік із глибокою аналітичною підтримкою. Дослідження існуючих аналогів підтвердило актуальність розробки системи, здатної автоматизувати роботу реєстратури, лікарів та забезпечити керівництво інструментами візуалізації ключових показників ефективності (KPI) закладу.

Розроблено багаторівневу архітектуру системи та структуру реляційної бази даних, що складається з 14 таблиць. Це забезпечило надійне збереження медичних даних, розкладів та відгуків пацієнтів. Спроектований інтерфейс користувача базується на ролевій моделі доступу, що дозволило розмежувати функціонал для директора, лікарів, реєстраторів та пацієнтів, забезпечуючи високу ергономічність та зручність використання.

Практична реалізація системи виконана з використанням мови програмування PHP та СУБД MySQL. Впроваджений аналітичний модуль на базі бібліотеки Chart.js дозволив автоматизувати процес формування інтерактивних звітів щодо завантаженості персоналу та репутаційного рейтингу лікарів. Створений особистий кабінет пацієнта з модулем онлайн-запису та цифровим архівом аналізів значно підвищив рівень сервісу та прозорість взаємодії з клієнтами.

Комплексне тестування системи підтвердило її стабільність, відсутність колізій при плануванні візитів та високу швидкість обробки запитів до бази даних. Впровадження системи в діяльність медичного центру дозволяє мінімізувати вплив людського фактору на етапі реєстрації, скоротити час на

підготовку управлінської звітності та оптимізувати внутрішні комунікації за допомогою модуля нарад.

Аналіз безпеки життєдіяльності та охорони праці довів, що розроблений програмний продукт відповідає вимогам ергономіки та психофізіології праці. Організація робочих місць згідно з визначеними стандартами, а також дотримання режимів праці та відпочинку, гарантує збереження здоров’я персоналу та забезпечує тривалу і безпечну експлуатацію системи в реальних умовах.

Таким чином, мета дипломної роботи досягнута повністю, а розроблена система є готовим до впровадження програмним рішенням, що має високий потенціал для автоматизації процесів у медичній галузі.

ПЕРЕЛІК ДЖЕРЕЛ

1. A. M. Alasmary, M. A. Alotaibi. Human-Centric Issues in eHealth Applications: A Review. – 2021 [Електронний ресурс]. Режим доступу: https://arxiv.org/abs/2104.01426 (дата звернення: 22.05.2026).

2. Apache Friends. XAMPP Documentation [Електронний ресурс]. Режим доступу: https://www.apachefriends.org/index.html (дата звернення: 22.05.2026).

3. Bauer, C. Springer Handbook of Database Technologies / C. Bauer, G. King. – Springer, 2024. – 650 p.

4. Bell, A. E. Database Concepts / A. E. Bell, D. M. Kroenke. – 9th ed. – Pearson, 2022. – 448 p.

5. Chart.js. Official Documentation [Електронний ресурс]. Режим доступу: https://www.chartjs.org/docs/latest/ (дата звернення: 22.05.2026).

6. Cognitive Load Theory in Interface Design: The Science of Effortless User Experiences. [Електронний ресурс]. Режим доступу: https://uk.pinterest.com/pin/1005428685573750175 (дата звернення: 22.05.2026).

7. Color in UI design (Part 2): Aspects and Meaning of Color on Human Emotions. [Електронний ресурс]. Режим доступу: https://uk.pinterest.com/pin/1107463364629748716 (дата звернення: 22.05.2026).

8. Connolly, T. Database Systems: A Practical Approach to Design, Implementation, and Management / T. Connolly, C. Begg. – 7th ed. – Pearson, 2022. – 1440 p.

9. Coronel, C. Database Systems: Design, Implementation, & Management / C. Coronel, S. Morris. – 14th ed. – Cengage Learning, 2023. – 832 p.

10. Dennis, A. Systems Analysis and Design / A. Dennis, B. Wixom, R. Roth. – 8th ed. – Hoboken, NJ : Wiley, 2021. – 448 p.

11. Dias, R. P. Modern Database Design and Applications / R. P. Dias, A. L. Ferreira. – Springer, 2024. – 380 p.

12. Doyle, M. Beginning PHP 8: Learn to Code, Create Websites, and Build Dynamic Web Applications / M. Doyle. – 5th ed. – Apress, 2021. – 720 p.

13. Duckett, J. PHP & MySQL: Server-side Web Development / J. Duckett. – Wiley, 2022. – 688 p.

14. Elmasri, R. Fundamentals of Database Systems / R. Elmasri, S. B. Navathe. – 8th ed. – Pearson, 2022. – 1280 p.

15. Erl, T. Service-oriented architecture: concepts, technology, and design / T. Erl. – Pearson Education India, 2022. – 792 p.

16. García-Molina, H. Database Systems: The Complete Book / H. García-Molina, J. D. Ullman, J. Widom. – 2nd ed. – Pearson, 2022. – 1248 p.

17. The use of abstract Moore automaton to control the sensors of a service-oriented alarm and emergency notification network / Olha Kryazhych, Victoria Itskovych, Kateryna Iushchenko, Veronika Hrytsyshyna, Danylo Bruvier, Vyacheslav Nykytyuk, Ihor Bodnarchuk // Scientific Journal of TNTU. – Tern. : TNTU, 2023. – Vol 109. – No 1. – P. 111–120.

18. MDN Web Docs. PHP Guide (latest documentation and references). Mozilla [Електронний ресурс]. Режим доступу: https://developer.mozilla.org/en-US/docs/Glossary/PHP (дата звернення: 22.05.2026).

19. Minnick, J. Responsive Web Design with HTML 5 & CSS / J. Minnick. – Cengage Learning, 2020. – 612 p.

20. National Library of Medicine. Healthcare Data Management and Electronic Health Records: Design and Implementation Approaches (review article). 2021 [Електронний ресурс]. Режим доступу: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8273493/ (дата звернення: 22.05.2026).

21. Nixon, R. Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5 / R. Nixon. – 6th ed. – O'Reilly Media, 2021. – 834 p.

22. Oestereich, B. Developing Software with UML: Object-oriented analysis and design in practice / B. Oestereich. – Pearson Education, 2022. – 352 p.

23. Oracle. MySQL 8.0 Reference Manual / Oracle. – 2023 [Електронний ресурс]. Режим доступу: https://dev.mysql.com/doc/refman/8.0/en/ (дата звернення: 22.05.2026).

24. Ramakrishnan, R. Database Management Systems / R. Ramakrishnan, J. Gehrke. – 4th ed. – McGraw-Hill Education, 2022. – 1065 p.

25. PostgreSQL Global Development Group. PostgreSQL 16 Documentation [Електронний ресурс]. Режим доступу: https://www.postgresql.org/docs/16/ (дата звернення: 22.05.2026).

26. Satzinger, J. W. Systems Analysis and Design in a Changing World / J. W. Satzinger, R. B. Jackson, S. D. Burd. – 8th ed. – Cengage Learning, 2022. – 525 p.

27. Sharma, P. Modern Database Management Systems: Concepts and Applications / P. Sharma, S. Sharma. – Springer, 2022. – 415 p.

28. Shneiderman, B. Designing the User Interface: Strategies for Effective Human-Computer Interaction / B. Shneiderman, C. Plaisant, M. Cohen [et al.]. – 7th ed. – Pearson, 2024. – 624 p.

29. Simpson, K. You Don’t Know JS Yet: Scope & Closures / K. Simpson. – 2nd ed. – O’Reilly Media, 2020–2021. – 180 p.

30. Traversy, B. Modern JavaScript From the Beginning / B. Traversy. – Packt Publishing, 2021. – 420 p.

31. Working from Home: Eye Strain & Remote Workers Complete Guide. [Електронний ресурс]. Режим доступу: https://poradumo.com.ua/53464-efektivna-zariadka-dlia-ochei-pri-roboti-z-komputerom (дата звернення: 22.05.2026).

32. Zandstra, M. PHP 8 Objects, Patterns, and Practice / M. Zandstra. – 6th ed. – Apress, 2021. – 579 p.

33. Березюк О. В. Безпека життєдіяльності : навч. посіб. / О. В. Березюк, М. С. Лемешев. – Вінниця : ВНТУ, 2011. – 168 с.

34. Боднарчук, І., & Харченко, О. ОЦІНЮВАННЯ ЯКОСТІ АРХІТЕКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ВИКОРИСТАННЯМ АЛГОРИТМУ ПРОСТОГО ВИБОРУ (14 ж).

35. Боднарчук, І. О. (2015). Методи і засоби проектування архітектури програмного забезпечення з врахуванням вимог якості.

36. Грибан В. Г. Охорона праці : навч. посіб. / В. Г. Грибан, О. В. Негодченко. – Київ : Центр учбової літератури, 2009. – 280 с.

37. Закон України «Про захист персональних даних» [Електронний ресурс]. Режим доступу: https://zakon.rada.gov.ua/laws/show/2297-17 (дата звернення: 22.05.2026).

38. Комплекс вправ для очей та розминки при роботі за комп'ютером. Facebook [Електронний ресурс]. Режим доступу: https://www.facebook.com/photo.php?fbid=2773502776231540&set=a.797392051246275&id=100029264634335 (дата звернення: 22.05.2026).

39. Концепція розвитку цифрової охорони здоров'я до 2025 року [Електронний ресурс]. Режим доступу: https://zakon.rada.gov.ua/laws/show/1671-2020-%D1%80 (дата звернення: 22.05.2026).

40. Національна служба здоров'я України. Електронна система охорони здоров’я eHealth: принципи функціонування та вимоги до МІС. – 2023 [Електронний ресурс]. Режим доступу: https://ehealth.gov.ua (дата звернення: 22.05.2026).

41. Порядок функціонування електронної системи охорони здоров'я [Електронний ресурс]. Режим доступу: https://zakon.rada.gov.ua/laws/show/411-2018-%D0%BF (дата звернення: 22.05.2026).

42. Харчеко, О., Боднарчук, І., & Яцишин, В. (2013). Експерта система проектування архітектури програмного забезпечення. Комп’ютерні технології друкарства, (29), 10-26.

Додатки

Додаток А

Програмна реалізація інформаційної панелі, аналітики та алгоритмів візуалізації

Лістинг А.1 – Скрипт збору та агрегації статистичних даних інформаційної панелі (analytics.php)

```
<?php
include 'db.php';
session_start();

if (!isset($_SESSION['role']) || $_SESSION['role'] !== 'Director')
{
    header("Location: login.php");
    exit();
}

// Обробка дат для формування динамічних часових зрізів
$start_date = $_GET['start_date'] ?? date('Y-m-d', strtotime('-7
days'));
$end_date = $_GET['end_date'] ?? date('Y-m-d');

// Агрегація динаміки візитів за обраний період (лінійний графік)
$stmtVisits = $pdo->prepare("
    SELECT DATE(appointment_date) as d, COUNT(*) as count
    FROM appointments
    WHERE appointment_date BETWEEN ? AND ?
    GROUP BY DATE(appointment_date)
    ORDER BY d ASC
");
$stmtVisits->execute([$start_date, $end_date]);
$days_data = $stmtVisits->fetchAll(PDO::FETCH_KEY_PAIR);

// Моніторинг та підрахунок статусів прийомів (кругова діаграма)
$status_data = $pdo->query("
    SELECT status, COUNT(*) as count
    FROM appointments
    GROUP BY status
")->fetchAll(PDO::FETCH_KEY_PAIR);

// Обчислення репутаційного KPI: середній рейтинг клініки на
основі відгуків
$stmtRating = $pdo->query("SELECT AVG(rating) as avg_rating,
COUNT(*) as total_reviews FROM doctor_reviews");
$rating_info = $stmtRating->fetch(PDO::FETCH_ASSOC);
$average_rating = number_format($rating_info['avg_rating'] ?? 0,
1);
$total_reviews = $rating_info['total_reviews'] ?? 0;
```

```
// Збір масивів для Drilldown-інфографіки (Завантаженість: План vs
Факт)
$dept_stats = $pdo->query("
    SELECT dep.id, dep.name, COUNT(DISTINCT d.id) * 40 as
total_capacity, COUNT(a.id) as booked_slots
    FROM departments dep
    LEFT JOIN doctors d ON d.department_id = dep.id
    LEFT JOIN appointments a ON a.doctor_id = d.id AND a.status !=
'cancelled'
    GROUP BY dep.id
")->fetchAll(PDO::FETCH_ASSOC);

$doctors_stats = $pdo->query("
    SELECT d.id as doc_id, d.last_name, d.department_id, 20 as
total_capacity, COUNT(a.id) as booked_slots
    FROM doctors d
    LEFT JOIN appointments a ON a.doctor_id = d.id AND a.status !=
'cancelled'
    GROUP BY d.id
")->fetchAll(PDO::FETCH_ASSOC);
?>
```

Лістинг А.2 – Алгоритм ініціалізації та інтерактивної візуалізації діаграм за допомогою Chart.js

```
// Візуалізація гістограми завантаженості медичних відділень з
підтримкою Drilldown
let drilldownChart;
const ctx =
document.getElementById('drilldownChart').getContext('2d');

function renderChart(labels, bookedData, capacityData, labelName)
{
    if (drilldownChart) drilldownChart.destroy();

    // Обчислення обсягу вільних слотів для стекової гістограми
    const freeData = capacityData.map((cap, i) => Math.max(0, cap
- bookedData[i]));

    drilldownChart = new Chart(ctx, {
        type: 'bar',
        data: {
            labels: labels,
            datasets: [
                {
                    label: 'Зайнято слотів',
                    data: bookedData,
                    backgroundColor: '#4169E1',
                    borderRadius: 5,
                },
                {
                    label: 'Вільні місця',
```

```
                    data: freeData,
                    backgroundColor: '#e0e0e0',
                    borderRadius: 5,
                }
            ]
        },
        options: {
            maintainAspectRatio: false,
            responsive: true,
            scales: { x: { stacked: true }, y: { stacked: true,
beginAtZero: true } },
            onClick: (e, activeEls) => {
                // Перехід на рівень лікарів при кліку на стовпчик
відділення
                if (activeEls.length > 0 && labelName === 'Записи
по відділеннях') {
                    const index = activeEls[0].index;
                    const deptId = deptData[index].id;
                    const deptName = deptData[index].name;
                    showDoctorChart(deptId, deptName);
                }
            }
        }
    });
}
```

Лістинг А.3 – Алгоритм динамічного рендерингу друкованої форми звіту ефективності в PDF (generate_pdf.php)

```
<?php
require_once 'dompdf/vendor/autoload.php';
include 'db.php';
use Dompdf\Dompdf;
use Dompdf\Options;

// Запит на підрахунок відсоткового співвідношення плану та факту
завантаженості
$stmtDept = $pdo->prepare("
    SELECT dep.id, dep.name, COUNT(DISTINCT d.id) * 40 as
total_capacity, COUNT(a.id) as booked_slots
    FROM departments dep
    LEFT JOIN doctors d ON d.department_id = dep.id
    LEFT JOIN appointments a ON a.doctor_id = d.id AND
a.appointment_date BETWEEN ? AND ? AND a.status != 'cancelled'
    GROUP BY dep.id
");
$stmtDept->execute([$_GET['start_date'], $_GET['end_date']]);
$dept_stats = $stmtDept->fetchAll(PDO::FETCH_ASSOC);

// Формування HTML-шаблону з прогрес-барами для рендерингу в PDF
$html = '<body><h3>Аналіз завантаженості відділень</h3><table>';
foreach ($dept_stats as $row) {
```

```
    $percent = $row['total_capacity'] > 0 ?
round(($row['booked_slots'] / $row['total_capacity']) * 100, 1) :
0;
    $html .= '<tr>
        <td>' . $row['name'] . '</td>
        <td>' . $percent . '%</td>
        <td><div style="background:#eee; width:100px;
height:10px;"><div style="background:#4169E1; height:10px; width:'
. $percent . '%;"></div></div></td>
    </tr>';
}
$html .= '</table></body>';

$options = new Options();
$options->set('defaultFont', 'DejaVu Sans'); // Забезпечення
підтримки кирилиці
$dompdf = new Dompdf($options);
$dompdf->loadHtml($html);
$dompdf->render();
$dompdf->stream("Efficiency_Report.pdf", ["Attachment" => true]);
?>
```

Додаток Б

Програмна реалізація модулів планування, адміністрування та автоматизації розкладу

Лістинг Б.1 – Алгоритм автоматичного масового налаштування робочих годин (auto_schedule.php)

```
<?php
include 'db.php';
session_start();

if ($_SERVER['REQUEST_METHOD'] === 'POST' &&
isset($_POST['target_doctors'])) {
    $doctor_ids = $_POST['target_doctors'];
    $days_selected = $_POST['days'] ?? [];

    // Мапінг українських інтерфейсних днів тижня на колонки
реляційної схеми БД
    $map = [
        'Пн' => ['mon_start', 'mon_end'], 'Вт' => ['tue_start',
'tue_end'],
        'Ср' => ['wed_start', 'wed_end'], 'Чт' => ['thu_start',
'thu_end'],
        'Пт' => ['fri_start', 'fri_end']
    ];

    foreach ($doctor_ids as $doc_id) {
        $values = ['doctor_id' => $doc_id];
        foreach ($map as $key => $cols) {
            if (in_array($key, $days_selected)) {
                $values[$cols[0]] = $_POST["start_$key"];
                $values[$cols[1]] = $_POST["end_$key"];
            } else {
                $values[$cols[0]] = null; // Позначення вихідного
дня
                $values[$cols[1]] = null;
            }
        }

        // Безпечний запис у БД з умовою ON DUPLICATE KEY UPDATE
для уникнення дублювання рядків
        $sql = "INSERT INTO doctor_schedules
                (doctor_id, mon_start, mon_end, tue_start,
tue_end, wed_start, wed_end, thu_start, thu_end, fri_start,
fri_end)
                VALUES (:doctor_id, :mon_start, :mon_end,
:tue_start, :tue_end, :wed_start, :wed_end, :thu_start, :thu_end,
:fri_start, :fri_end)
                ON DUPLICATE KEY UPDATE
```

```
                mon_start=VALUES(mon_start),
mon_end=VALUES(mon_end),
                tue_start=VALUES(tue_start),
tue_end=VALUES(tue_end),
                wed_start=VALUES(wed_start),
wed_end=VALUES(wed_end),
                thu_start=VALUES(thu_start),
thu_end=VALUES(thu_end),
                fri_start=VALUES(fri_start),
fri_end=VALUES(fri_end)";

        $pdo->prepare($sql)->execute($values);
    }
    header("Location: administration.php?schedule_success=1");
}
?>
```

Лістинг Б.2 – Програмна логіка перевірки зайнятості та бронювання тайм-слотів прийому (appointment.php)

```
<?php
// Отримання робочих годин лікаря для обраної користувачем дати
$day_of_week_en = date('l', strtotime($selected_date));
$day_map = ['Monday' => 'mon', 'Tuesday' => 'tue', 'Wednesday' =>
'wed', 'Thursday' => 'thu', 'Friday' => 'fri'];
$is_off_day = true;

if (isset($day_map[$day_of_week_en])) {
    $prefix = $day_map[$day_of_week_en];
    $stmtSch = $pdo->prepare("SELECT {$prefix}_start,
{$prefix}_end FROM doctor_schedules WHERE doctor_id = ?");
    $stmtSch->execute([$doctor_id]);
    $sch = $stmtSch->fetch();
    if ($sch && $sch["{$prefix}_start"]) {
        $work_start = $sch["{$prefix}_start"];
        $work_end = $sch["{$prefix}_end"];
        $is_off_day = false;
    }
}

// Генерація часової сітки та верифікація доступності кожного
слоту
$current_time = strtotime($work_start);
$end_time = strtotime($work_end);
$interval = 30; // Тривалість прийому — 30 хвилин

while ($current_time < $end_time) {
    $t_str = date("H:i", $current_time);

    // Перевірка наявності існуючого активного запису на цей час у
базі даних
```

```
    $check = $pdo->prepare("SELECT id FROM appointments WHERE
doctor_id = ? AND appointment_date = ? AND appointment_time = ?
AND status != 'cancelled'");
    $check->execute([$doctor_id, $selected_date, $t_str]);

    if (!$check->fetch()) {
        // Оформлення вільного слоту для запису
    }
    $current_time = strtotime("+$interval minutes",
$current_time);
}
?>
```

Додаток В

Структура аналітичних, системних та захищених SQL-запитів до реляційної бази даних

Лістинг В.1 – Запит для багатовимірного Drilldown-аналізу та зіставлення планової спроможності з фактом завантаженості відділень

```
SELECT
    dep.id,
    dep.name,
    COUNT(DISTINCT d.id) * 40 as total_capacity, -- Розрахунок
умовної планової місткості відділення
    COUNT(a.id) as booked_slots                  -- Підрахунок
фактично зайнятих слотів пацієнтами
FROM departments dep
LEFT JOIN doctors d ON d.department_id = dep.id
LEFT JOIN appointments a ON a.doctor_id = d.id AND a.status !=
'cancelled'
GROUP BY dep.id;
```

Лістинг В.2 – Запит агрегації та сортування лікарей за середнім репутаційним показником (KPI)

```
SELECT
    d.id, d.first_name, d.last_name, dep.name as dept_name,
    AVG(r.rating) as avg_rating,         -- Обчислення середньої
оцінки на основі відгуків
    COUNT(r.id) as reviews_count         -- Загальна кількість
надісланих відгуків пацієнтів
FROM doctors d
JOIN departments dep ON d.department_id = dep.id
LEFT JOIN doctor_reviews r ON d.id = r.doctor_id
GROUP BY d.id
ORDER BY avg_rating DESC;                -- Сортування рейтингу за
спаданням
```

Лістинг В.3 – Складний аналітичний запит верифікації часових лімітів для надання пацієнту прав на відгук (обмеження в 48 годин)

```
SELECT DISTINCT d.id, d.first_name, d.last_name, dep.name as
dept_name, a.appointment_date
FROM appointments a
JOIN doctors d ON a.doctor_id = d.id
JOIN departments dep ON d.department_id = dep.id
WHERE a.patient_id = ?
  AND a.status = 'completed'             -- Прийом має бути
успішно завершеним
```

```
  AND a.appointment_date <= CURDATE()
  AND DATEDIFF(CURDATE(), a.appointment_date) <= 2; -- Різниця дат
не повинна перевищувати 2 дні
```

Лістинг В.4 – Запит для моніторингу активних оперативних нарад для лікаря з урахуванням 15-хвилинного буфера

```
SELECT COUNT(*)
FROM meetings
WHERE FIND_IN_SET(?, doctor_ids)          -- Пошук ідентифікатора
лікаря у рядку з розділювачами
  AND meeting_datetime >= (NOW() - INTERVAL 15 MINUTE); --
Поточний час з урахуванням інтервалу проведення
```

Додаток Г

Компоненти забезпечення безпеки, автентифікації користувачів та розмежування прав доступу

Лістинг Г.1 – Налаштування безпечного з'єднання з СКБД за допомогою технології PDO (db.php)

```
<?php
$host = 'localhost';
$db   = 'hospital_db';
$user = 'root';
$pass = '';
$charset = 'utf8mb4';

$dsn = "mysql:host=$host;dbname=$db;charset=$charset";
$options = [
    // Увімкнення режиму генерації виключень при помилках для
запобігання витоку структури БД
    PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
    // Встановлення дефолтного формату вибірки даних у вигляді
асоціативних масивів
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    // Вимкнення емуляції підготовлених запитів для забезпечення
апаратного захисту від SQL-ін'єкцій
    PDO::ATTR_EMULATE_PREPARES   => false,
];

try {
     $pdo = new PDO($dsn, $user, $pass, $options);
} catch (\PDOException $e) {
     throw new \PDOException($e->getMessage(), (int)$e-
>getCode());
}
?>
```

Лістинг Г.2 – Модуль автентифікації користувачів із використанням безпечного механізму password_verify (login.php)

```
<?php
include 'db.php';
session_start();

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $login_input = trim($_POST['email']);
    $password = $_POST['password'];

    // Пошук облікового запису за допомогою параметризованого
безпечного запиту
```

```
    $stmt = $pdo->prepare("
        SELECT u.*, r.role_name
        FROM users u
        JOIN roles r ON u.role_id = r.id
        WHERE u.email = ?
    ");
    $stmt->execute([$login_input]);
    $user = $stmt->fetch();

    // Криптографічна верифікація введеного пароля з його
хешованим аналогом у БД
    if ($user && password_verify($password, $user['password'])) {
        $_SESSION['user_id'] = $user['id'];
        $_SESSION['role'] = $user['role_name']; // Фіксація
рольової моделі в сесії

        header("Location: index.php");
        exit();
    } else {
        $error = "Невірний логін (email/телефон) або пароль";
    }
}
?>
```

Лістинг Г.3 – Трикроковий алгоритм відновлення доступу за одноразовими OTP-кодами за допомогою SMTP-сервера PHPMailer (forgot_password.php)

```
<?php
include 'db.php';
session_start();
use PHPMailer\PHPMailer\PHPMailer;

function sendMail($toEmail, $code) {
    $mail = new PHPMailer(true);
    try {
        $mail->isSMTP();
        $mail->Host       = 'smtp.gmail.com';
        $mail->SMTPAuth   = true;
        $mail->Username   = 'gooshhhka@gmail.com';
        $mail->Password   = 'arsjjfvidmpuuahi'; // Програмний
захищений 16-значний ключ доступу
        $mail->SMTPSecure = PHPMailer::ENCRYPTION_STARTTLS;
        $mail->Port       = 587;
        $mail->CharSet    = 'UTF-8';

        $mail->setFrom('gooshhhka@gmail.com', 'Медичний Центр');
        $mail->addAddress($toEmail);
        $mail->isHTML(true);
        $mail->Subject = 'Код відновлення пароля';
        $mail->Body    = "Ваш одноразовий код підтвердження:
<b>$code</b>";
```

```
        $mail->send();
        return true;
    } catch (Exception $e) {
        return false;
    }
}

// Крок 3: Якщо код вірний — безпечне хешування та оновлення
пароля користувача
if (isset($_POST['set_password'])) {
    $new_pass = $_POST['new_password'];
    if (strlen($new_pass) >= 6 && $new_pass ===
$_POST['confirm_password']) {
        // Хешування за сучасним криптографічним стандартом Argon2
/ Bcrypt за замовчуванням
        $password_hash = password_hash($new_pass,
PASSWORD_DEFAULT);

        $stmt = $pdo->prepare("UPDATE users SET password = ? WHERE
id = ?");
        $stmt->execute([$password_hash,
$_SESSION['reset_user_id']]);

        unset($_SESSION['reset_code'], $_SESSION['reset_user_id'],
$_SESSION['reset_step']);
        $_SESSION['reset_step'] = 4; // Статус успішного оновлення
    }
}
?>
```