

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

ОСВІТНЬО ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»

галузь знань 12 «Інформаційні технології»

спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЄКТ

на тему

WEB-застосунок обліку діяльності приватного медичного закладу

здобувача Мельника Дмитра Ігоровича _____

Науковий керівник:

д.е.н., професор

_____Мозгалі О.П.

**Проект допущений до захисту
перед екзаменаційною комісією з
атестації здобувачів вищої освіти
завідувач кафедри:**

к.е.н., доцент.

_____Тішков Б.О.

Київ 2024

Міністерство освіти і науки України
Київський національний економічний університет імені Вадима Гетьмана
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»

Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»

галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

ПОГОДЖЕНО:

Керівник проєктної групи(гарант)
освітньо-професійної програми

_____Іванченко Г.Ф.

“ _____ ” _____ 2024 р.

ЗАТВЕРДЖУЮ:

Завідувач кафедри

_____Тішков Б.О.

“ _____ ” _____ 2024 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти ***Мельника Дмитра Ігоровича***

очної (денної) форми навчання

на підготовку кваліфікаційного бакалаврського проєкту
на тему: «WEB-застосунок обліку діяльності приватного медичного закладу»

Тему затверджено наказом ректора Університету від «11»__ 03__ 2024 р. № 529-ст.

Кваліфікаційний бакалаврський проєкт виконується на матеріалах
наукових публікаціях та матеріалів з мережі інтернет

План кваліфікаційного бакалаврського проєкту

Розділ I Характеристика та аналіз предметної області

Розділ II Розробка вимог і моделювання веб-застосунку

Розділ III Проектування та реалізація компонентів системи

Об'єкт дослідження: веб-застосунок для обліку діяльності приватного медичного закладу.

Предмет дослідження: процеси автоматизації обліку діяльності приватного медичного закладу за допомогою розробленого веб-застосунку.

Мета кваліфікаційного бакалаврського проєкту: розроблення та впровадження веб-застосунку для автоматизації обліку діяльності приватного медичного закладу, отримання досвіду розроблення інформаційних систем.

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

У розділі I Дослідити принципи роботи та характеристики інформаційних систем підтримки діяльності медичних закладів, зокрема веб-застосунки обліку діяльності приватних медичних закладів, порівняти різні інформаційні системи з точки зору їх стійкості, швидкості та складності реалізації. Обрати оптимальну структуру веб-застосунку обліку діяльності приватного медичного закладу.

У розділі II Визначити та описати вимоги до веб-застосунку обліку діяльності приватного медичного закладу. Проаналізувати та навести специфікацію вимог до веб-застосунку обліку діяльності приватного медичного закладу. Розробити постановку задачі, архітектуру веб-застосунку обліку діяльності приватного медичного закладу. Визначити функціональні модулі системи, розробити схеми взаємодії між модулями системи.

У розділі III Здійснити проєктування та реалізувати компоненти веб-застосунку обліку діяльності приватного медичного закладу. Вибрати мову програмування та інструменти розробки для реалізації системи. Розробити програмне забезпечення для кожного модуля системи. Навести опис інформаційного та технічного забезпечення для реалізації веб-застосунку обліку діяльності приватного медичного закладу. Протестувати систему щодо наявності помилок та уразливостей.

Завдання підготував

науковий керівник _____

Мозгалі Ольга Петрівна

“11”__03__ 2024 р.

Завдання одержав

студент _____

Мельник Дмитро Ігорович

“11”__03__ 2024 р.

Відгук
про кваліфікаційний бакалаврський проєкт
здобувача навчально-наукового інституту
«Інститут інформаційних технологій в економіці»
освітньо-професійної програми
«Комп'ютерні науки»

Мельника Дмитра Ігоровича

на тему

«WEB-застосунок обліку діяльності приватного медичного закладу»

1. Актуальність теми: У сучасному світі, де технології розвиваються стрімкими темпами, приватні медичні заклади стикаються з постійною потребою в оптимізації своїх процесів та підвищенні ефективності роботи. WEB-застосунки для обліку діяльності медичних закладів можуть стати незамінним інструментом для досягнення цих цілей. Оскільки цей тип програмного забезпечення пропонує широкий спектр функцій, які дозволяють автоматизувати рутинні завдання, вести облік пацієнтів, медичних карток, фінансових операцій та багато іншого, й завдяки цьому керівництво закладу може зосередитися на більш стратегічних питаннях, а персонал може приділяти більше уваги безпосередньо пацієнтам. Крім того, даний WEB-застосунок може допомогти покращити комунікацію та співпрацю між різними відділами медичного закладу. Завдяки централізованому збору та доступності даних, всі зацікавлені сторони можуть мати актуальну інформацію про стан справ у закладі. Враховуючи всі ці переваги, WEB-застосунок обліку діяльності приватного медичного закладу стає все більш актуальним для приватних медичних закладів, які прагнуть до успіху в конкурентному середовищі.

2. Позитивні риси кваліфікаційного бакалаврського проєкту: В роботі було системно досліджено предметну область, включаючи принципи обліку медичної діяльності, архітектуру системи, методи та засоби розроблення веб-застосунків. Автором було застосовано актуальні методи аналізу та синтезу інформації, а також програмного забезпечення для розроблення та тестування інформаційної системи для обліку діяльності приватного медичного закладу. Розроблена WEB-застосунок обліку діяльності приватного медичного закладу може мати практичне застосування в медичній сфері.

3. Наявність самостійних розробок автора: Кваліфікаційний бакалаврський проєкт виконаний автором самостійно із застосуванням сучасного інструментарію. Автор самостійно дослідив методи та засоби розроблення WEB-застосунків та принципи обліку діяльності приватного медичного закладу; також ним було самостійно обрано програмні та технічні засоби й розроблено програмне забезпечення для реалізації WEB-застосунку обліку діяльності приватного медичного закладу. Крім того, автором було проведено первинне тестування WEB-застосунку та усунуто виявлені помилки.

4. Цінність теоретичних висновків та практичних рекомендацій:

Теоретичні висновки роботи можуть бути використані для розробки нових підходів до організації обліку в приватних медичних закладах. Розроблений WEB-застосунок має потенціал для комерційного використання й забезпечити підвищення ефективності роботи медичного закладу та якості надання медичних послуг, а також зниження витрат та економію часу за рахунок автоматизації рутинних завдань.

5. Наявність недоліків: Представлене проєктне рішення потребує подальшого вдосконалення й розвитку для використання у промисловій експлуатації. Також перед впровадженням в промислову експлуатацію необхідно буде провести тестування розробленого WEB-застосунку на проникнення, щоб оцінити його стійкість до можливих кібератак.

6. Загальна оцінка кваліфікаційного бакалаврського проєкту та його допущення до захисту перед ЕК: Загалом, представлений кваліфікаційний бакалаврський проєкт відповідає встановленим вимогам методичних вказівок щодо структури, обсягу та змісту, та рекомендується до захисту з **позитивною оцінкою**, а Мельнику Дмитру Ігоровичу може бути присвоєний освітньо-кваліфікаційний рівень «бакалавр» галузі знань 12 «Інформаційні технології» за спеціальністю 122 «Комп'ютерні науки».

Науковий керівник

(підпис)

д. е. н., Мозгаллі О.П.

“10” червня 2024 р.

Рецензія
на кваліфікаційний бакалаврський проєкт
Мельника Дмитра Ігоровича
на тему: «WEB-застосунок обліку діяльності приватного медичного закладу»

Актуальність теми кваліфікаційної бакалаврської роботи і доцільність її розроблення полягає у потребі створення та удосконалення веб-додатку обліку діяльності приватного медичного закладу. У зв'язку зі зростанням обсягів медичної інформації та потреби в ефективному управлінні нею, створення такої системи стає критичною необхідністю.

Якість проведеного дослідження характеризується високим рівнем теоретичного обґрунтування та практичної реалізації. Автор даної роботи провів ретельний аналіз предметної області та існуючих програмних засобів, забезпечив ґрунтовну базу для розробки додатку, використав сучасні технології та підходи у проєктуванні та впровадженні веб-застосунку.

Позитивні риси кваліфікаційної бакалаврської роботи полягають у комплексному підході до автоматизації процесів приватного медичного закладу. Реалізація веб-застосунку, що включає реєстрацію пацієнтів, обробку медичних записів, планування візитів та моніторинг лікування, значно підвищує ефективність роботи медичного персоналу. Зручний інтерфейс, інтеграція з існуючими базами даних та безпечний доступ до медичної інформації роблять систему корисною та надійною для користувачів.

Зауваження до кваліфікаційної бакалаврської роботи стосуються деяких аспектів її реалізації та перспектив розвитку. Незважаючи на загальну високу якість, робота могла б виграти від більш детального тестування на більших обсягах даних для оцінки продуктивності системи під навантаженням. Крім того, можливі труднощі в інтеграції з деякими існуючими медичними системами можуть вимагати додаткового часу та ресурсів для вирішення. Впровадження системи також може потребувати навчання медичного персоналу, що слід врахувати в плануванні ресурсів і часу.

Практична значущість висновків і рекомендацій полягає у спроможності підвищити ефективність управління та роботи медичного закладу через впровадження веб-застосунку. Вважаю, що робота виконана на високому професійному рівні, має теоретичну та практичну значущість й заслуговує на оцінку «Відмінно».

Місце роботи та посада рецензента: директор підприємства ТОВ «РЕД КАРЛОС»

Никитюк П. С.



АНОТАЦІЯ

студента 4 курсу

Навчально-наукового інституту «Інститут інформаційних технологій в економіці»

Мельника Дмитра Ігоровича, виконаної на тему:

«WEB-застосунок обліку діяльності приватного медичного закладу»

Київ: кафедра інформаційних систем в економіці, 2024р.

Актуальність даного кваліфікаційного бакалаврського проєкту полягає у потребі створення та удосконалення веб-додатку обліку діяльності приватного медичного закладу. У зв'язку зі зростанням обсягів медичної інформації та потреби в ефективному управлінні нею, створення такої системи стає критичною необхідністю.

Кваліфікаційний бакалаврський проєкт складається з трьох розділів, кожен з яких логічно впорядкований та взаємопов'язаний з іншими. Перший розділ присвячений аналізу предметної галузі, в якому проводиться характеристика об'єкта дослідження та аналіз існуючих програмних засобів. Другий розділ є проєктним та містить обґрунтування методу проєктування системи, розроблення її архітектури та алгоритму розв'язання задач. Третій розділ конструктивний і включає інформаційне, технічне та організаційне забезпечення системи.

У висновках надаються рекомендації щодо доцільності розроблення та впровадження веб-додатку обліку діяльності приватного медичного закладу, а також формулюються перспективи подальшого розвитку системи та можливі шляхи удосконалення.

Цей проєкт має важливе значення для оптимізації роботи медичних закладів та підвищення якості медичних послуг, а також є актуальним доповненням у галузі інформаційних технологій у сфері охорони здоров'я.

РЕФЕРАТ

Кваліфікаційний бакалаврський проєкт містить 68 сторінок, 9 таблиць, 16 рисунків, список джерел літератури та ресурсів з 10 найменувань та 16 додатків.

WEB-застосунок обліку діяльності приватного медичного закладу

Ключові слова: веб-застосунок, інформаційна система, додаток, приватний медичний заклад, автоматизація, облік, пацієнти, медичні записи, планування візитів, моніторинг лікування, адміністративні процеси, аналіз, ефективність, рекомендації, розвиток, розробка.

Об'єктом дослідження є веб-застосунок для обліку діяльності приватного медичного закладу. Цей застосунок спрямований на покращення процесів реєстрації пацієнтів, обробки медичних записів, планування візитів, моніторингу лікування та адміністрування. Мета полягає у розробці веб-застосунку та впровадженні цієї системи для підвищення ефективності медичного закладу та поліпшення якості медичних послуг.

Предметом дослідження є процеси автоматизації обліку діяльності приватного медичного закладу за допомогою розробленого веб-застосунку. Це включає вивчення методів, алгоритмів та технологій, необхідних для ефективного управління даними та процесами, пов'язаними з медичною практикою.

Мета кваліфікаційного бакалаврського проєкту полягає у розробленні та впровадженні веб-застосунку для автоматизації обліку діяльності приватного медичного закладу, а також отримання досвіду розроблення інформаційних систем. Розроблений застосунок має на меті полегшити адміністративні процеси та забезпечити зручний доступ до медичної інформації.

У процесі дослідження був проведений аналіз предметної галузі, наведені характеристики процесів ІТ-підтримки діяльності медичного закладу, розроблена архітектура та алгоритм роботи системи. Також було розроблено та впроваджено інформаційне, технічне та організаційне забезпечення для підсистеми обліку діяльності медичного закладу.

Результатом проєкту є веб-застосунок, який спростить роботу медичного закладу та поліпшить якість медичних послуг, які надаються пацієнтам.

Висновки проєкту містять рекомендації щодо впровадження застосунку та підтримки його подальшого розвитку. Реалізація цього проєкту сприятиме покращенню медичної допомоги та задоволенню потреб пацієнтів.

Рік виконання кваліфікаційного бакалаврського проєкту – 2024р.

Рік захисту кваліфікаційного бакалаврського проєкту – 2024р.

ВСТУП.....	3
РОЗДІЛ 1 ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ....	7
1.1 Характеристика предметної галузі та об'єкта дослідження.....	7
1.2 Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі.....	10
РОЗДІЛ 2 РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ВЕБ-ЗАСТОСУНКУ	
2.1 Аналіз і специфікація вимог до веб-застосунку	16
2.2 Постановка задачі та алгоритм розв'язання задачі	20
2.2.1. Характеристика задачі.....	20
2.2.2. Алгоритм розв'язання задачі	24
2.3 Моделювання веб-застосунку.....	27
2.3.1. Моделювання поведінки системи	27
2.3.2. Моделювання структури застосунку	32
2.3.3. Вимоги за компонентами системи	36
РОЗДІЛ 3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ.....	39
3.1 Інформаційне забезпечення	39
3.2 Технічне забезпечення.....	42
3.3 Програмне забезпечення	44
3.4 Результати реалізації веб-застосунку.....	47
ВИСНОВКИ	49
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	51
ДОДАТКИ.....	2

ВСТУП

Сучасна медична сфера стрімко розвивається, що вимагає вдосконалення методів управління та контролю за діяльністю медичних закладів. Ефективне функціонування медичного закладу залежить від чіткої організації робочих процесів, оперативного доступу до інформації та своєчасного прийняття управлінських рішень. Автоматизація обліку діяльності приватного медичного закладу та роботи його персоналу за допомогою інформаційної системи стає ключовим фактором в сучасному світі для підвищення його конкурентоспроможності, розвитку та ефективності.

Вибір теми «WEB-застосунок обліку діяльності приватного медичного закладу» обумовлений необхідністю підвищення ефективності управління та обліку в медичних установах, особливо в умовах швидко розвиваючогося ринку приватних медичних послуг. Інформаційні системи та застосунки, які їх реалізують, відіграють ключову роль у модернізації та автоматизації процесів, що дозволяє значно підвищити якість обслуговування пацієнтів, оптимізувати роботу медичного персоналу та знизити витрати.

Актуальність дослідження полягає в значущості впровадження сучасних інформаційних технологій у приватних медичних закладах. В умовах пандемій, зростаючої конкуренції та підвищених вимог пацієнтів до якості медичних послуг, приватні медичні установи потребують ефективних рішень для обліку та управління своєю діяльністю. Сучасний рівень розвитку інформаційних систем веб-застосунків дозволяє створювати комплексні рішення, що включають електронні медичні картки, системи управління записом на прийом, облік фінансових операцій тощо.

Об'єктом дослідження є веб-застосунок для обліку діяльності приватного медичного закладу. Предметом дослідження є процеси автоматизації обліку діяльності приватного медичного закладу за допомогою розробленого веб-застосунку.

Метою даного дослідження є розроблення та впровадження веб-застосунку для автоматизації обліку діяльності приватного медичного закладу, а також отримання досвіду розроблення інформаційних систем.

Існуючі підходи до автоматизації обліку в медичних закладах можна поділити на три групи, а саме: використання готових програмних продуктів (вибір рішення серед широкого спектру готових програмних продуктів, призначених для автоматизації роботи медичних закладів), розробка власного застосунку (може бути оптимальним рішенням для закладів зі специфічними потребами, дозволить створити систему, яка буде максимально відповідати потребам та специфіці роботи закладу) та використання хмарних сервісів, котрі стають все більш популярними для автоматизації роботи медичних закладів. Вони пропонують зручний доступ до інформаційної системи з будь-якого місця та пристрою без значних інвестицій у IT-інфраструктуру.

Ключовий зміст проєкту передбачає розробку функціоналу для лікарів, головного лікаря та адміністратора, що відповідає їхнім конкретним потребам та обов'язкам. Лікарі отримають можливість ефективного обліку своїх пацієнтів та подій, головний лікар буде мати повний доступ до інформації про лікарів, а адміністратор може ефективно керувати даними пацієнтів та персоналу.

Очікуваний результат розробки у створенні функціонального веб-додатку, який буде відповідати потребам приватного медичного закладу, забезпечить автоматизацію обліку його діяльності, покращить роботу персоналу, підвищить рівень обслуговування пацієнтів та сприятиме розвитку та успіху медичного закладу.

Для дослідження були використані наступні джерела:

1. Health Care Information Systems: A Practical Approach for Health Care Management [1].
2. Electronic Health Records: Understanding and Using Computerized Medical Records [2].
3. Медичні інформаційні системи: огляд можливостей і приклади використання [3].
4. Медична інформаційна система, що це [4].
5. Розробка медичної інформаційної системи для медичних закладів первинної ланки [5].
6. Проєктування баз і сховищ даних [6].
7. Проєктування систем оброблення інформації [7].

8. Інформаційні системи і технології в економіці [8].

9. Концепція та методології системного інжинірингу [9].

10. Комп'ютеризована система шляхів догляду (CCPs) для підтримки особистісно-орієнтованого, інтегрованого та проактивного догляду в умовах домашнього догляду [11].

11. Розробка та впровадження особистісно-орієнтованої, технологічно вдосконаленої моделі медичної допомоги для лікування хронічних станів: когортне дослідження [12].

12. Аналіз і розробка веб-орієнтованих інформаційних систем охорони здоров'я в епоху промислової революції [13].

В результаті огляду літературних джерел було виявлено, що існуючі дослідження досить добре висвітлюють загальні принципи та підходи до створення медичних інформаційних систем, але практичні аспекти їх впровадження у приватних медичних закладах потребують подальшого дослідження та адаптації до специфічних умов.

Для вирішення поставлених завдань використовувалися наступні методи дослідження:

1. Аналіз існуючих рішень – проведено огляд і аналіз існуючих інформаційних систем та застосунків обліку та управління в медичних закладах. Цей аналіз дозволив ідентифікувати переваги та недоліки існуючих рішень і визначити ключові вимоги до нового веб-застосунку.

2. Проєктування застосунку – на основі отриманих даних розроблено концептуальну модель веб-застосунку, що включає в себе структуру бази даних, алгоритми обробки та зберігання інформації, а також інтерфейс користувача. Кожен етап проєктування був детально обґрунтований з урахуванням специфіки потреб приватних медичних закладів.

3. Використання алгоритмів роботи – розроблено алгоритми роботи застосунку для кожного з основних функціональних блоків, таких як облік пацієнтів, запис на прийом, облік фінансових операцій тощо. Ці алгоритми були ретельно протестовані та оптимізовані з метою забезпечення швидкодії та надійності системи.

4. Тестування системи – проведено комплексне тестування розробленої інформаційної системи веб-застосунку з метою перевірки її працездатності,

безпеки та відповідності вимогам. Тестування включало в себе модульне тестування окремих компонентів системи, інтеграційне тестування взаємодії між компонентами, а також приймальне тестування на реальних даних.

Ці методи дослідження дозволили забезпечити якісне проєктування та впровадження веб-застосунку обліку діяльності приватного медичного закладу, забезпечуючи високу ефективність та надійність її роботи.

У розробці веб-додатку було використано даних інструментарій:

- Клієнтська частина – для функціональної, безпечної та маштабованої розробки інтерфейсу системи було обрано TypeScript фреймворк Angular з використанням графічної бібліотеки PrimeNG.
- Серверна частина – для розробки API (Application Programming Interface) серверу було обрано прогресивний фреймворк Node.js – NestJS.
- Для зберігання даних було обрано NoSQL (Not only SQL) базу даних MongoDB з використанням бібліотеки Mongoose.
- В розробці веб-додатку використовувалась IDE (Integrated Development Environment) WebStorm. В якості інструменту проєктування графічного інтерфейсу використовувався онлайн сервіс Figma. Для тестування роботи маршрутів серверної частини було використано API платформу Postman.

Даний інструментарій дозволив зручно та ефективно розробити веб-застосунок, який відповідає всім поставленим цілям.

РОЗДІЛ 1 ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Характеристика предметної галузі та об'єкта дослідження

Предметна галузь охоплює комплекс об'єктів реального або уявного світу, що розглядається в контексті веб-застосунку і технологій для приватних медичних закладів. У нашому випадку предметна галузь включає різноманітні аспекти управління та обліку діяльності медичних установ, що забезпечує ефективне функціонування медичних послуг.

Предметна галузь – характеристика та аналіз ІТ підтримки діяльності медичних закладів, сукупність інформаційних систем, розроблених для автоматизації процесів управління в медичних закладах. Ці системи забезпечують електронний облік пацієнтів, управління записами на прийом, облік медичних послуг, фінансовий облік, а також інші адміністративні функції. Основні завдання, які вирішуються в рамках цієї галузі, включають:

- Підвищення ефективності роботи медичного персоналу.
- Поліпшення якості обслуговування пацієнтів.
- Зниження витрат на адміністрування.
- Забезпечення збереження та конфіденційності медичної інформації.

Облік приватного медичного закладу включає досить різні функції. В рамках даного проєкту, веб-застосунок буде мати такий функціонал як: звітність відвідувань пацієнтів, перегляд та можливість зміни інформації щодо лікарів та пацієнтів, керування подіями закладу.

Організаційна структура медичного закладу включає різні робочі місця: лікарі, адміністратори та головний лікар. Зв'язки між ними полягають у взаємодії щодо доступу до інформації та виконання різних функцій. Лікарі працюють з пацієнтами та вносять дані про їх здоров'я та зустрічі. Головний лікар має доступ до всієї інформації та може керувати подіями у закладі, створювати нові прийоми та додавати лікування відповідним діагнозам. Адміністратори керують інформацією про події закладу, пацієнтів та лікарів, ведуть документообіг, мають можливість додавати нові діагнози та лікування.

Система буде мати дані про пацієнтів, лікарів (вони ж і користувачі системи), діагнози, лікування, івенти, прийоми. Така структура забезпечує чіткий розподіл функцій та відповідальності між користувачами, що сприяє підвищенню ефективності роботи медичного закладу.

Основні функції та взаємозв'язки між рівнями користувачів:

- Адміністратор: забезпечує загальне адміністрування системи, підтримує актуальність даних про користувачів та пацієнтів, здійснює управління доступом.
- Головний лікар: контролює медичну діяльність, керує плануванням прийомів та заходів, відповідає за додавання нових медичних записів та діагнозів.
- Лікар: здійснює облік та ведення медичних записів своїх пацієнтів, складає звіти про проведені прийоми, пов'язані з лікуванням пацієнтів.

Передача та обробка інформації між різними робочими місцями відбувається шляхом заповнення звітів лікарів щодо прийомів пацієнтів, створення та обробки медичних карток пацієнтів, рецептів тощо. Це забезпечує оперативний доступ до актуальної інформації та дозволяє швидко приймати обґрунтовані рішення.

Основні сутності розроблюваної інформаційної системи веб-застосунку:

1. Пацієнти – зберігання даних про пацієнтів, включаючи їх медичні історії та особисту інформацію.
2. Лікарі – управління інформацією про лікарів, їхні розклади та спеціалізації.
3. Адміністратор – здійснює загальне управління системою та забезпечує підтримку її роботи.
4. Лікарські прийоми – опис результатів зустрічей лікарів з пацієнтами, формування звітності закладу.
5. Події закладу – відображення заходів у медичному закладі, таких як переоблік медичного персоналу та інші адміністративні події.
6. Діагнози – реєстрація та зберігання інформації про діагнози пацієнтів, що дозволяє лікарям вести історію хвороби та планувати лікування.
7. Лікування – облік призначених пацієнтам процедур, медикаментів та інших терапевтичних заходів.

Така організаційна структура дозволяє забезпечити високий рівень безпеки та конфіденційності медичної інформації, а також оптимізувати роботу кожного підрозділу медичного закладу. Це створює передумови для ефективного управління закладом, що в кінцевому рахунку підвищує якість надання медичних послуг та задоволеність пацієнтів.

Рішення в предметній галузі стосуються управління та оптимізації діяльності медичних закладів. Основні проблемні ситуації включають забезпечення точного обліку пацієнтів, оптимізацію розкладу прийомів, контроль за виконанням медичних процедур та збереження конфіденційності даних. Цілі розв'язання цих проблем включають підвищення ефективності роботи закладу, покращення якості медичних послуг та зниження витрат. Особами, що приймають рішення, є адміністратори, головні лікарі та інші управлінські кадри. Принципи узгодження індивідуальних переваг можуть включати групове прийняття рішень та консультації з медичним персоналом.

Фактори, що впливають на середовище прийняття рішень, включають технологічні, економічні та соціальні аспекти. Технологічні фактори охоплюють доступність і функціональність інформаційних систем. Економічні фактори включають вартість впровадження та обслуговування системи. Соціальні фактори охоплюють рівень підготовки та прийняття нових технологій персоналом.

Взаємозв'язки рішень стосуються того, як попередні рішення впливають на подальші дії та результати. Наприклад, вибір інформаційної системи впливає на методи обліку пацієнтів та управління медичними процедурами. Ці рішення мають враховувати попередній досвід і результати впровадження інших систем та методик.

Для прийняття обґрунтованих рішень необхідна внутрішня та зовнішня інформація, включаючи емпіричні дані, об'єктивну статистику та звіти. Джерела їх надходження включають медичні картки, внутрішні звіти закладу та зовнішні медичні бази даних. Методи збору та обробки інформації включають алгоритмізацію, евристику, імітацію та аналітичну методологію.

Ситуації прийняття рішень у медичних закладах можуть бути різних типів: закриті задачі, відкриті задачі або кризові ситуації. Види рішень можуть бути

детерміновані, прийняті за умов ризику, невизначеності або нечітких цілей. Ефективне управління інформаційними системами дозволяє своєчасно реагувати на зміни та приймати обґрунтовані рішення в різних ситуаціях. Саме веб-застосунок допоможе зручно та ефективно керувати такою системою.

1.2 Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі

Аналіз літературних джерел дозволяє оцінити сучасний стан розв'язання проблем, пов'язаних з інформаційними системами і технологіями в медичних закладах. Розглядаючи різні теорії та погляди, важливо провести критичний огляд відповідних літературних джерел, у тому числі зарубіжних, щоб отримати комплексне розуміння досліджуваної теми.

Інформаційні системи, включаючи застосунки, які їх реалізують, в медичних закладах є критично важливими для ефективного управління пацієнтами та адміністративними процесами. За даними багатьох досліджень, основні напрямки розвитку ІС включають електронні медичні картки (ЕМК), системи управління пацієнтами (СУП), телемедицину, а також рішення для підтримки прийняття рішень (СППР) [13] [17].

1. Електронні медичні картки (ЕМК) – сучасні ЕМК дозволяють зберігати всю інформацію про пацієнта в одному місці, забезпечуючи швидкий доступ до історії хвороби, діагнозів, результатів аналізів та лікування. Дослідження показують, що використання ЕМК знижує кількість помилок і підвищує ефективність медичних закладів.

2. Системи управління пацієнтами (СУП) – СУП автоматизують процеси запису пацієнтів, управління розкладом прийомів, обробки платежів і ведення медичної документації. Це значно полегшує роботу адміністративного персоналу та покращує взаємодію між різними відділами медичного закладу.

3. Рішення для підтримки прийняття рішень (СППР) – СППР використовуються для аналізу великих обсягів медичних даних і надання рекомендацій лікарям щодо діагностики та лікування пацієнтів. Ці системи

базуються на штучному інтелекті та машинному навчанні, що дозволяє підвищити точність і швидкість прийняття рішень.

Для критичного огляду літературних джерел були розглянуті наукові статті, дослідження та публікації відомих фахівців у галузі медичних інформаційних систем. Аналіз показав, що більшість дослідників згодні з тим, що впровадження ІС у медичних закладах сприяє покращенню якості медичних послуг і підвищенню ефективності управління. Проте існують і певні проблеми, такі як високі витрати на впровадження систем, складність їх інтеграції з існуючими процесами та необхідність навчання персоналу.

Аналіз практичного досвіду використання ІС та веб-додатків у медичних закладах показав, що найбільш успішними є ті заклади, які використовують комплексні системи, що охоплюють всі аспекти управління та обслуговування пацієнтів. Наприклад, впровадження ЕМК у клініках США значно підвищило рівень задоволеності пацієнтів і знизило адміністративні витрати. У країнах Європи активно використовуються системи телемедицини, що дозволяють пацієнтам отримувати консультації від лікарів з інших країн, що підвищує якість діагностики та лікування [14].

При аналізі літературних джерел і практичного досвіду були виявлені як позитивні, так і негативні аспекти використання інформаційних систем у медичних закладах. Серед позитивних аспектів – підвищення ефективності роботи, зниження кількості помилок, поліпшення якості медичних послуг. Серед негативних – високі витрати на впровадження, складність інтеграції з існуючими системами та необхідність постійного навчання персоналу.

Важливими проблемними питаннями, які потребують подальшого дослідження, є:

- Як знизити витрати на впровадження і підтримку застосунків у медичних закладах?
- Які методи інтеграції нових систем з існуючими процесами є найбільш ефективними?
- Як забезпечити високий рівень безпеки та конфіденційності медичних даних?

- Які підходи до навчання персоналу є найбільш дієвими для швидкого освоєння нових ІС?

На основі аналізу літературних джерел та практичного досвіду можна зробити висновок, що впровадження інформаційних систем у медичних закладах є необхідним і ефективним кроком для підвищення якості медичних послуг. Проте, для досягнення максимального ефекту, необхідно враховувати як позитивні, так і негативні аспекти, розробляти стратегії зниження витрат і забезпечення інтеграції нових систем. Особлива увага повинна приділятися питанням безпеки даних і навчання персоналу. Це створює передумови для подальшого дослідження та розробки ефективних рішень для управління медичними закладами за допомогою інформаційних систем [1] [3] [12] [13].

Розглянемо та порівняємо існуючі інформаційні системи для ІТ підтримки діяльності приватних медичних закладів. Ми зосередимося на таких важливих аспектах, як повнота функціональних можливостей ІС, функції та завдання прийняття рішень, на які орієнтована ІС, передбачені підходи до організації бази даних, програмного забезпечення, технологічно-організаційні аспекти та якість інтерфейсу.

Першою розглянемо ЕМСі – це медична інформаційна система в Україні, включена до переліку медичних інформаційних систем, рекомендованих Міністерством охорони здоров'я України. Система ЕМСі розроблена відповідно до стандартів ISO та МОЗ України і містить модулі: електронна медична картка пацієнта, лабораторія, медичні кадри та інші.

Основними автоматизованими системою елементами є документообіг, управління потоками пацієнтів, отримання діагностичних даних у зручному вигляді, облік персоналу, облік матеріальних цінностей для поліклінік, приватних практик, комерційної медицини, науково-дослідних інститутів, обласних клінічних центрів та інших медичних установ незалежно від форми власності, типів допомоги, що надається, кадрового складу і рівня організації.

Центральним елементом системи є пацієнт і його електронна медична карта, в якій зберігається вся інформація про історію хвороби: хто його обслуговував, які документи були створені, які послуги надані і які наступні плани [15].

Другою інформаційною системою для порівняння буде МЕДІНФОСЕРВІС. Дана інформаційна система охоплює автоматизацію лікувальних процесів амбулаторно-поліклінічних та стаціонарних лікувальних закладів, акредитований МОЗ та підключений до електронної системи охорони здоров'я «Е-Health». Обираючи МІС «МедІнфоСервіс» Ви отримуєте повний функціонал для передачі електронних медичних записів до ЕСОЗ з функцією імпорту даних з «Медичної карти амбулаторного хворого» та «Медичної карти стаціонарного хворого» до електронних медичних записів, а це означає, що лікарям не потрібно робити потрійну роботу, а достатньо лише внести дані в електронну документацію з подальшою можливістю імпорту та друку документів.

Функціонал робочого місця амбулаторії охоплює низку процесів, які необхідні для повноцінного функціонування поліклінічного закладу/відділення/кабінету та включає в себе роботу онлайн та офлайн реєстратури з можливістю формування електронної черги на прийом.

Модуль «Медична карта амбулаторного пацієнта» реалізовано у вигляді електронної медичної картки, яка містить дані про пацієнта (особисті, місце проживання, місце роботи, листок профоглядів, сигнальні позначки, фактори ризику та інші), анамнез життя, хірургічні втручання, алергологічний анамнез, непереносимість лікарських засобів та відвідування

Взаємодія з електронною системою охорони здоров'я дозволяє лікарям створювати декларації з пацієнтами, реєструвати електронні рецепти за програмою реімбурсації «Доступні ліки», виписувати/погашати електронні направлення, тощо [16].

В таблиці 1.1 наведено порівняльну характеристику наявних інформаційних систем.

Провівши детальний порівняльний аналіз існуючих інформаційних систем у сфері обліку медичних закладів наведемо декілька ключових висновків.

Таблиця 1.1– Порівняльна характеристика наявних інформаційних систем

Ключові аспекти	Наш проєкт	ЕМСi	МЕДІНФОСЕРВІС
Повнота функціональних можливостей ІС	+	+	+
ситуації, пов'язані з прийняттям рішень, функції та завдання прийняття рішень, на які орієнтована ІС	+	+	+
організація бази/сховища даних, програмного забезпечення	+	+	+
технологічно-організаційні аспекти	+	+	+
якість інтерфейсу «ІС – користувач»	+	+	–
склад і форма результатів	+	+	+
новітні методи, технології та підходи до розроблення ІС	+	+	–

Джерело: складено автором

Оцінюючи існуючі інформаційні системи, було визначено, що наш проєкт є цілком конкурентоспроможним та не поступається по функціоналу та іншим основним характеристикам аналогам на ринку.

Система ЕМСi є чудовим рішенням для великих закладів чи навіть цілих мереж. Має чудовий інтерфейс та цілу екосистему допоміжного програмного забезпечення та вражає функціональністю. В свою чергу, МЕДІНФОСЕРВІС активно розвивається та постійно оновлюється, потребує мінімальних витрат на встановлення та експлуатацію, має швидке реагування на звернення користувачів технічної підтримки та розробників медичної інформаційної системи.

Враховуючі вищезгадані висновки порівняння, можна виокремити конкретний функціонал, який в подальшому складе напрямок розвитку нашого веб-застосунку, наприклад, створення реєстру ліків закладу, впровадження системи відомостей щодо щеплень.

РОЗДІЛ 2 РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ВЕБ-ЗАСТОСУНКУ

2.1 Аналіз і специфікація вимог до веб-застосунку

Вимоги до веб-застосунку – це ключовий етап у процесі його розробки. Вони визначають умови, які повинний виконувати застосунок або її компоненти для досягнення поставлених цілей, задоволення стандартів, специфікацій та інших формальних документів. Цей процес є частиною системного інжинірингу, від якого залежить успішність усього проєкту.

Основною метою створення веб-застосунку є підвищення ефективності управління медичним закладом шляхом автоматизації процесів обліку пацієнтів, медичних записів, управління ресурсами та поліпшення взаємодії між різними відділами.

Для веб-застосунку обліку діяльності приватного медичного закладу важливо визначити наступні бізнес-вимоги:

- Реєстрація та облік пацієнтів – можливість реєстрації нових пацієнтів, зберігання особистої та медичної інформації, історії візитів, діагнозів та рекомендацій.
- Управління лікарями та персоналом – додавання, редагування та видалення профілів лікарів, назначення робочих графіків, доступ до медичних даних та звітів.
- Звітність відвідувань пацієнтів – автоматизована формація звітів за відвідуваннями пацієнтів, відображення статистики за періодами, типами відвідувань та лікарями.
- Ефективне керування подіями закладу – фіксація, аналіз та моніторинг подій в медичному закладі, таких як переоблік персоналу, надходження нового обладнання чи організація навчальних семінарів.
- Документообіг та адміністративна робота – система повинна підтримувати ефективний документообіг, включаючи створення, редагування, зберігання та обмін медичною документацією, рецептами, договорами та іншою адміністративною інформацією.

- Забезпечення безпеки даних – захист конфіденційної інформації пацієнтів, використання сучасних методів шифрування та захисту від несанкціонованого доступу.

- Мобільний доступ та інтеграція – можливість роботи з системою з мобільних пристроїв, інтеграція з іншими медичними системами, лабораторіями та страховими компаніями для автоматизації процесів.

- Інтерфейс користувача – інтуїтивно зрозумілий та дружній інтерфейс, який спрощує навчання персоналу, покращує продуктивність роботи та зменшує ймовірність помилок.

Ці бізнес-вимоги відображають комплексні потреби медичного закладу у системі обліку діяльності, яка буде надійною, ефективною та гнучкою для адаптації до змін у робочих процесах та вимогах користувачів.

У процесі створення веб-застосунку для обліку діяльності приватного медичного закладу, основна увага приділяється не лише забезпеченню потрібного функціоналу, а й гарантуванню її надійності, безпеки та ефективності. Ця система має відповідати вимогам не тільки медичної сфери, але й сучасних стандартів ІТ, щоб забезпечити комфорт та зручність роботи для всього персоналу медичного закладу. На відповідь на ці потреби, нижче представлені функціональні вимоги до розроблюваної системи:

- Авторизація та аутентифікація – користувачі матимуть можливість реєстрації та входу в якості лікарів, адміністраторів та головного лікаря. Саме це забезпечить захист особистих даних пацієнтів за допомогою шифрування.

- Управління пацієнтською базою – користувач системи буде надана можливість додавання, редагування та видалення профілів пацієнтів з основними даними, контактною інформацією, медичною історією. Зберігання історії відвідувань, результатів обстежень, аналізів та лікування для кожного пацієнта.

- Робота з медичним персоналом – ця функція надає інструмент адміністрації інформаційної системи створення, редагування та видалення профілів лікарів з основними даними, спеціалізацією, графіком роботи. Назначення, редагування та скасування прийомів пацієнтів, управління чергами.

- Календар та планування – відображення індивідуальних графіків роботи лікарів та загального графіка роботи закладу. Онлайн-планування прийомів пацієнтів, автоматичне оновлення графіка при зміні робочих змін.
- Документообіг – можливість для користувача системи створення, редагування, зберігання медичних документів, рецептів, висновків з можливістю приєднання файлів. Електронний обмін документами між робочими місцями з можливістю стеження за статусами документів.
- Адаптивний інтерфейс користувача – інтуїтивно зрозумілий дизайн, підтримка різних типів пристроїв та розмірів екранів.

✓ FR1	FunctionalRequirement	Proposed	High	Medium
✓ FR2	FunctionalRequirement	Proposed	Medium	High
✓ FR3	FunctionalRequirement	Proposed	Low	Medium
✓ FR4	FunctionalRequirement	Proposed	High	Medium
✓ FR5	FunctionalRequirement	Proposed	Medium	Low
✓ FR6	FunctionalRequirement	Proposed	Medium	Medium

Рисунок 2.1 – Таблиця специфікації функціональних вимог

Джерело: складено автором

У рамках розробки веб-застосунку для приватного медичного закладу, не менш важливим є забезпечення якості та надійності системи. Нефункціональні вимоги визначають критерії, що впливають на загальну продуктивність, безпеку та зручність використання системи. Вони охоплюють аспекти, такі як швидкодія, масштабованість, доступність, стабільність та інші, які визначають загальний рівень задоволення від використання системи як медичним персоналом, так і пацієнтами. Нижче наведено детальний перелік нефункціональних вимог до системи.

- Ведення лікарської звітності – користувачам веб-застосунку повинно бути надано набір інструментів для ведення обліку лікарської діяльності, оформлення звітів прийомів пацієнтів;
- Безпекові міри – додаток повинен мати механізми регулярної аутентифікації та авторизації користувачів, а також захист від несанкціонованого доступу та зломів.
- Перегляд та робота з інформацією системи – користувачам має бути надано доступ до зручного перегляду інформації системи. Це включає перегляд інформації пацієнтів, лікарів, прийомів та подій.
- Зручний інтерфейс – додаток повинен забезпечувати легкий у сприйнятті, ефективний у роботі та гнучкий інтерфейс для роботи з даними.

✔ NFR1	NoFunctionalRequirement	Proposed	Medium	Medium
✔ NFR2	NoFunctionalRequirement	Proposed	High	Medium
✔ NFR3	NoFunctionalRequirement	Proposed	Medium	Low
✔ NFR4	NoFunctionalRequirement	Proposed	Low	Medium

Рисунок 2.2 – Таблиця специфікації нефункціональних вимог

Джерело: складено автором

Ці нефункціональні вимоги гарантують надійність, безпеку та ефективність роботи веб-додатку в приватному медичному закладі.

Вимоги до веб-додатку є основою для її подальшої розробки і впровадження. Вони повинні бути чіткими, зрозумілими і повними, щоб забезпечити ефективність роботи системи та задоволення потреб користувачів. Визначення правильних вимог і їх належне документування є критичним етапом, від якого залежить успішність проєкту загалом.

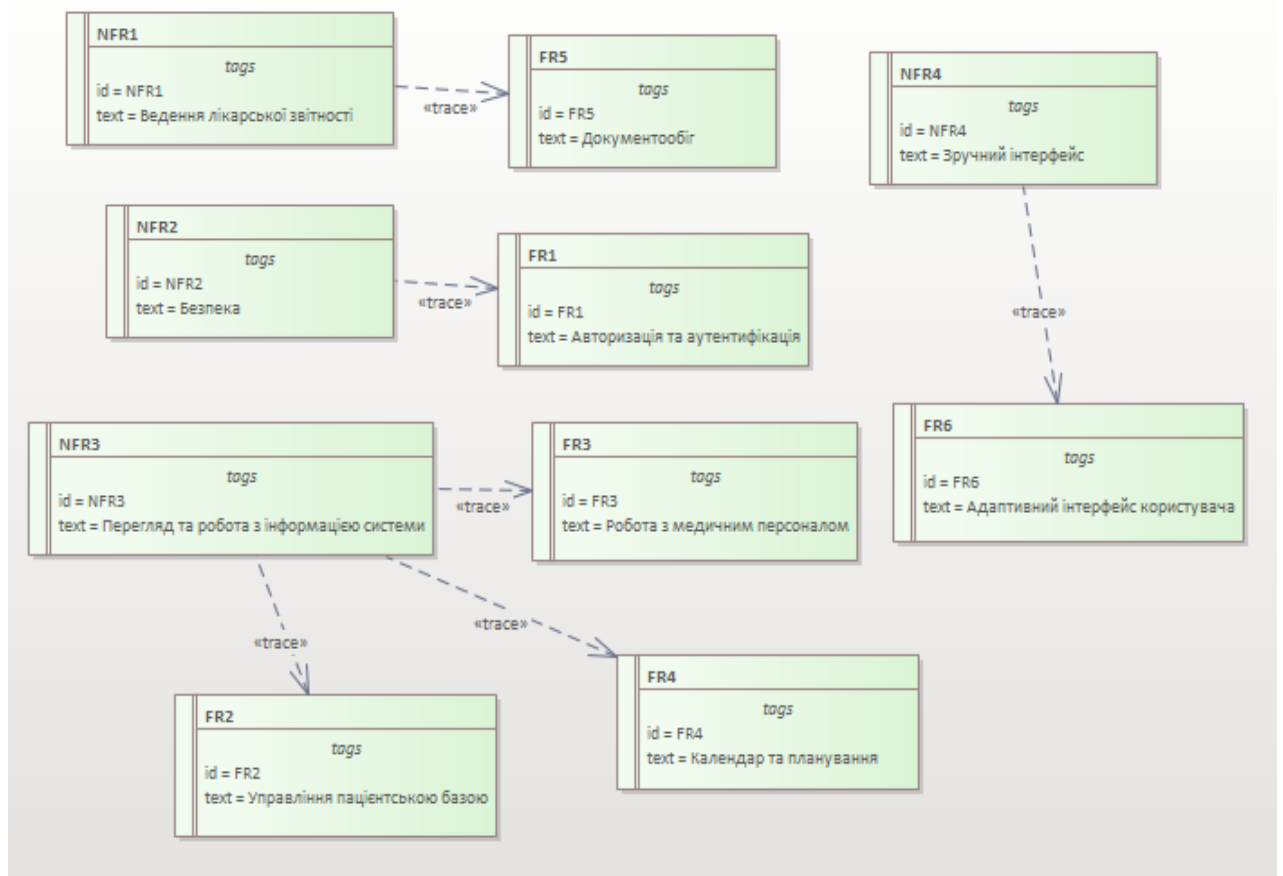


Рисунок 2.3 - Діаграма вимог з поданням ієрархії вимог

Джерело: складено автором

2.2 Постановка задачі та алгоритм розв’язання задачі

2.2.1. Характеристика задачі

Призначення задачі полягає в розробці веб-додатку для обліку приватного медичного закладу, яка має на меті полегшити та оптимізувати роботу лікарів, головного лікаря та адміністратора. Техніко-економічна сутність полягає в покращенні робочих процесів, підвищенні ефективності та якості надання медичних послуг, а також в забезпеченні ефективного управління медичним закладом. Обґрунтування використання електронно-обчислювальних машин (ЕОМ) базується на їхній здатності автоматизувати процеси обліку, забезпечити зручний доступ до інформації та зменшити ймовірність помилок у веденні обліку.

З використанням ЕОМ можливе централізоване управління даними, автоматичне оновлення інформації, швидке та точне виконання розрахунків і звітів, а також забезпечення високого рівня безпеки та конфіденційності даних.

Це дозволить зменшити кількість ручної роботи, мінімізувати ризик людських помилок та покращити загальну ефективність медичного закладу.

До переліку об'єктів, за управління якими розв'язується задача, входять:

- Пацієнти (реєстрація, облік медичної історії).
- Лікарі (управління профілями, графіками роботи).
- Медичні записи (зберігання, редагування, обмін).
- Адміністративні процеси (управління ресурсами, звітність, документообіг).

Вихідна інформація включає в себе дані про пацієнтів (особиста інформація, медична історія), дані про лікарів, звіти зустрічей та прийомів пацієнтів, інформацію про події в закладі. Ця інформація використовується для ефективного управління та надання медичних послуг.

Розв'язання задачі відбувається в режимі реального часу. Періодичність видачі вихідної інформації може бути різною в залежності від конкретного запиту чи обставин, але в основному інформація має бути доступною миттєво.

Автоматизоване розв'язання задачі може припинитися у разі технічних збоїв, відсутності електропостачання, або при необхідності проведення технічного обслуговування системи.

Дана задача тісно пов'язана з задачами обліку медичних послуг, управління ресурсами (лікарі, персонал), а також може мати зв'язки з фінансовим обліком закладу.

Розподіл дій між персоналом та технічними засобами передбачає доступ до системи для лікарів, головного лікаря та адміністратора з відповідним рівнем доступу. Технічні засоби забезпечують збереження та обробку інформації, а персонал відповідає за правильне введення даних, моніторинг роботи системи та вирішення можливих проблем.

Інформаційна модель задачі включає взаємодію між різними компонентами системи, такими як база даних пацієнтів, модулі управління медичними записами, модулі обробки та аналізу даних, а також інтерфейси для користувачів (лікарів, адміністративного персоналу). Модель повинна забезпечувати ефективну обробку даних, зберігання та захист інформації, а також інтеграцію з зовнішніми системами.

Адміністрація закладу відповідає за підтримку технічної частини інформаційної системи, моніторинг стабільної роботи веб-застосунку. До технічних засобів можна віднести сервери та програмне забезпечення.

Користувачам системи доступні такі можливості:

- Перегляд інформації інформаційної системи веб-застосунку – користувачі можуть переглядати різні типи інформації, зокрема профілі пацієнтів, історію медичних записів, графіки роботи лікарів, звіти та адміністративні документи. Це забезпечує швидкий доступ до необхідної інформації для прийняття обґрунтованих рішень.
- Додання нової інформації до системи – користувачі мають можливість додавати нові записи до системи, такі як нові профілі пацієнтів, медичні записи, результати обстежень, нові події або заходи в закладі. Це дозволяє підтримувати актуальність та повноту даних.
- Редагування існуючих даних – користувачі можуть редагувати вже існуючі записи, вносячи необхідні корективи чи оновлення. Це включає оновлення медичної інформації, корекцію помилок, зміну графіків роботи лікарів та інші адміністративні зміни.

На рисунку 2.4 продемонстровано основні об'єкти інформаційної системи веб-додатку та зв'язки між ними для кращого розуміння функціонування процесів.

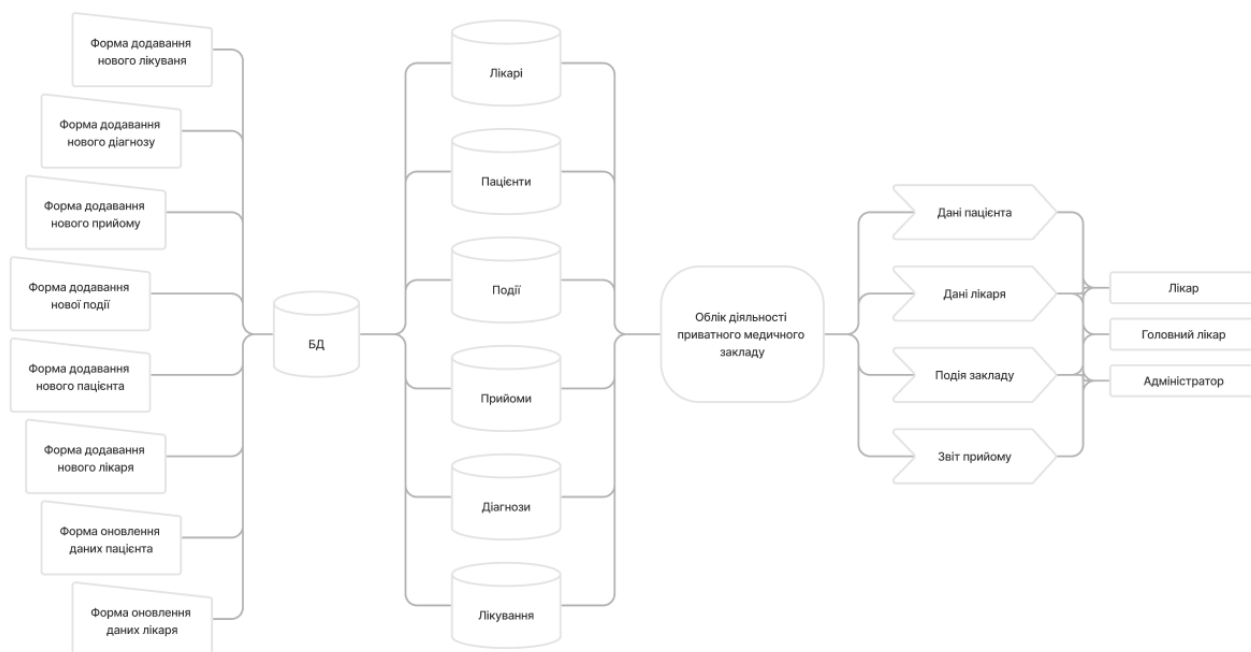


Рисунок 2.4 – Інформаційна модель задачі

Вхідна інформація в систему є ключовим елементом, що дозволяє веб-додатку виконувати свої завдання з обліку приватного медичного закладу. Призначенням вхідної інформації є надання системі необхідних даних для ведення медичного обліку, планування прийому лікарів, та управління ресурсами закладу. В таблиці 2.1 наведено перелік та опис вхідних повідомлень, що застосовується для розв'язання поставлених задач.

Таблиця 2.1 – Перелік і опис вхідних повідомлень

№ з/п	Назва вхідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
1.	Оформлення нового відвідування пацієнта	INP-001	Електронна форма на веб-сайті	За потребою	Веб-застосунок обліку діяльності медзакладу
2.	Оформлення звіту відвідування пацієнта	INP-002	Електронна форма на веб-сайті	За потребою	Веб-застосунок обліку діяльності медзакладу
3.	Створення нової події закладу	INP-003	Електронна форма на веб-сайті	За потребою	Веб-застосунок обліку діяльності медзакладу
4.	Редагування інформації пацієнтів	INP-004	Електронна форма на веб-сайті	За потребою	Веб-застосунок обліку діяльності медзакладу
5.	Редагування інформації лікарів	INP-005	Електронна форма на веб-сайті	За потребою	Веб-застосунок обліку діяльності медзакладу
6.	Додання нових пацієнтів	INP-006	Електронна форма на веб-сайті	За потребою	Веб-застосунок обліку діяльності медзакладу
7.	Додання нових лікарів	INP-007	Електронна форма на веб-сайті	За потребою	Веб-застосунок обліку діяльності медзакладу
8.	Додання нових діагнозів	INP-008	Електронна форма на веб-сайті	За потребою	Веб-застосунок обліку діяльності медзакладу
9.	Додання нових лікувань	INP-009	Електронна форма на веб-сайті	За потребою	Веб-застосунок обліку

№ з/п	Назва вхідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
					діяльності медзакладу

Джерело: складено автором

Вихідна інформація є результатом роботи веб-додатку та відображає дані про діяльність приватного медичного закладу. Призначенням вихідної інформації є забезпечення зручного та інформативного представлення даних для прийняття управлінських рішень, формування звітів та забезпечення користувачів актуальною інформацією.

Таблиця 2.2 – Перелік і опис вихідних повідомлень

№ з/п	Назва вихідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
1.	Сторінка з карткою пацієнта	OUT-001	Сторінка у веб-застосунку	Після оновлення веб-сайту	Веб-застосунок обліку діяльності
2.	Сторінка з карткою лікаря	OUT-002	Сторінка у веб-застосунку	Після оновлення веб-сайту	Веб-застосунок обліку діяльності
3.	Панель з подіями закладу	OUT-003	Сторінка у веб-застосунку	Після оновлення веб-сайту	Веб-застосунок обліку діяльності
4.	Звіт лікаря	OUT-004	Модальне вікно у веб-застосунку	Після оновлення веб-сайту	Веб-застосунок обліку діяльності

Джерело: складено автором

2.2.2. Алгоритм розв’язання задачі

Однією з ключових математичних моделей в системі є обробка і зберігання інформації про розклад прийому лікарів та інші медичні події. Для цього використовуються математичні формули для роботи з датами та часом.

Розрахунок кількості прийнятих пацієнтів 1-лікарями за дату d :

$$P_{ld} = \sum_i ldi,$$

де i – це певний пацієнт, d – дата/проміжок часу, l – лікар медичного закладу.

Розрахунок кількості прийнятих пацієнтів І-лікарями за період часу d: $P_l = \sum_d P_{ld}$.

Розрахунок кількості прийнятих пацієнтів по всіх лікарях закладу:

$$P = \sum_l P_l.$$

Основні алгоритми функціонування системи, з основними етапами, наведено нижче:

- Оформлення звіту лікаря щодо прийому пацієнта:
 1. Отримання вхідних даних з форми від користувача.
 2. Підтвердження відправки даних користувачем.
 3. Валідація отриманих даних системою.
 4. Відправка даних до бази даних.
- Додання нового суб'єкта до бази даних закладу:
 1. Користувач натискає відповідний пункт в боковому меню застосунку.
 2. Обирає суб'єкт який хоче додати (пацієнта, лікаря, звіт чи інше).
 3. Вводить у форму відповідну інформацію.
 4. Підтверджує введені дані.
 5. Система проводить валідацію даних.
 6. Система відправляє дані до бази даних.
- Перегляд інформації:
 1. Користувач переходить на відповідну сторінку додатку.
 2. Система підвантажує дані з бази даних та відображає їх користувачу.
 3. Користувач переглядає відповідні дані.
- Оновлення інформації пацієнтів та лікарів:
 4. Користувач переходить на відповідну сторінку додатку.
 5. Натискає кнопку оновлення даних.
 6. Система відображає користувачу модальне вікно з формою для вводу нових даних.
 7. Користувач вводить дані та підтверджує відправку.
 8. Система валідує дані та відправляє до бази даних.

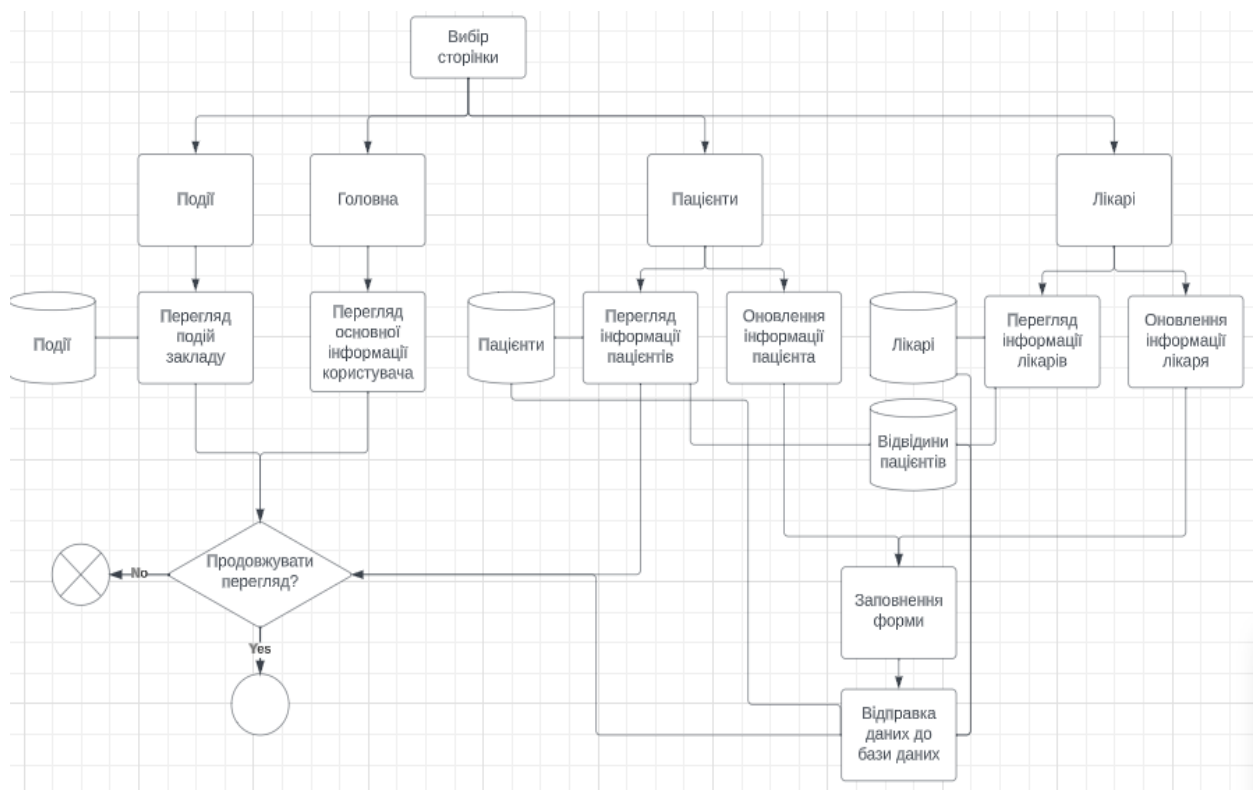


Рисунок 2.5 – Схема алгоритму обліку діяльності приватного медичного закладу

Джерело: складено автором

Дана схема відображає весь спектр та послідовність операцій, що відбуваються в веб-застосунку. Вона ілюструє дії, які може виконувати користувач, що допомагає краще зрозуміти концепцію інформаційної системи та спрощує в подальшому розробку веб-застосунку.

Створення діаграм алгоритмів є важливим етапом у процесі розробки, оскільки вони сприяють розумінню, комунікації та ефективності системи, що в результаті призводить до успішного впровадження та експлуатації веб-додатку для обліку приватного медичного закладу.

2.3 Моделювання веб-застосунку

2.3.1. Моделювання поведінки системи

Діаграма прецедентів є однією з ключових частин проєктування системи, яка дозволяє визначити основні взаємодії між користувачами та системою. Цей інструмент допомагає зрозуміти, як різні категорії користувачів будуть взаємодіяти з системою, та визначити основні функціональні можливості системи з огляду на її бізнес-вимоги.

Діаграма прецедентів допомагає ідентифікувати ключові прецеденти, які система повинна підтримувати, що в свою чергу визначає потрібний функціонал для коректної роботи системи. Цей інструмент дозволяє чітко визначити, як користувачі будуть взаємодіяти з системою, що допомагає уникнути непорозумінь та недоречностей у процесі розробки. Діаграма прецедентів допомагає визначити різні рівні доступу до функціоналу системи для різних категорій користувачів, що є важливим для забезпечення безпеки та конфіденційності даних. Вона надає чітке уявлення про те, як система буде інтегрована в бізнес-процеси медичного закладу, допомагаючи забезпечити їх ефективність та оптимізацію.

Діаграма прецедентів веб-додатку обліку діяльності приватного медичного закладу описує дії користувачів, таких як лікар, головний лікар та адміністратор:

- Редагування інформації – даний прецедент дозволяє користувачам редагувати інформації в системі залежно від рівня дозволу. Адміністратор має дозвіл на редагування всіх даних системи, головний лікар – пацієнтів, діагнозів та лікувань, а лікар, може редагувати прийоми пацієнтів;
- Пошук інформації – цей прецедент дозволяє користувачам системи здійснювати пошук необхідних даних за унікальними ідентифікаторами та переглядати результати;
- Логін – прецедент, що демонструє можливість авторизації користувачів, шляхом заповнення форми логіну;
- Перегляд інформації – цей прецедент демонструє можливість перегляду користувачами системи даних. В залежності від дозволу користувачу надається доступ до даних: адміністратор має абсолютний доступ, головний лікар доступ

до даних всіх лікарів та пацієнтів, а лікар – доступ до своїх пацієнтів та їхніх відвідин.

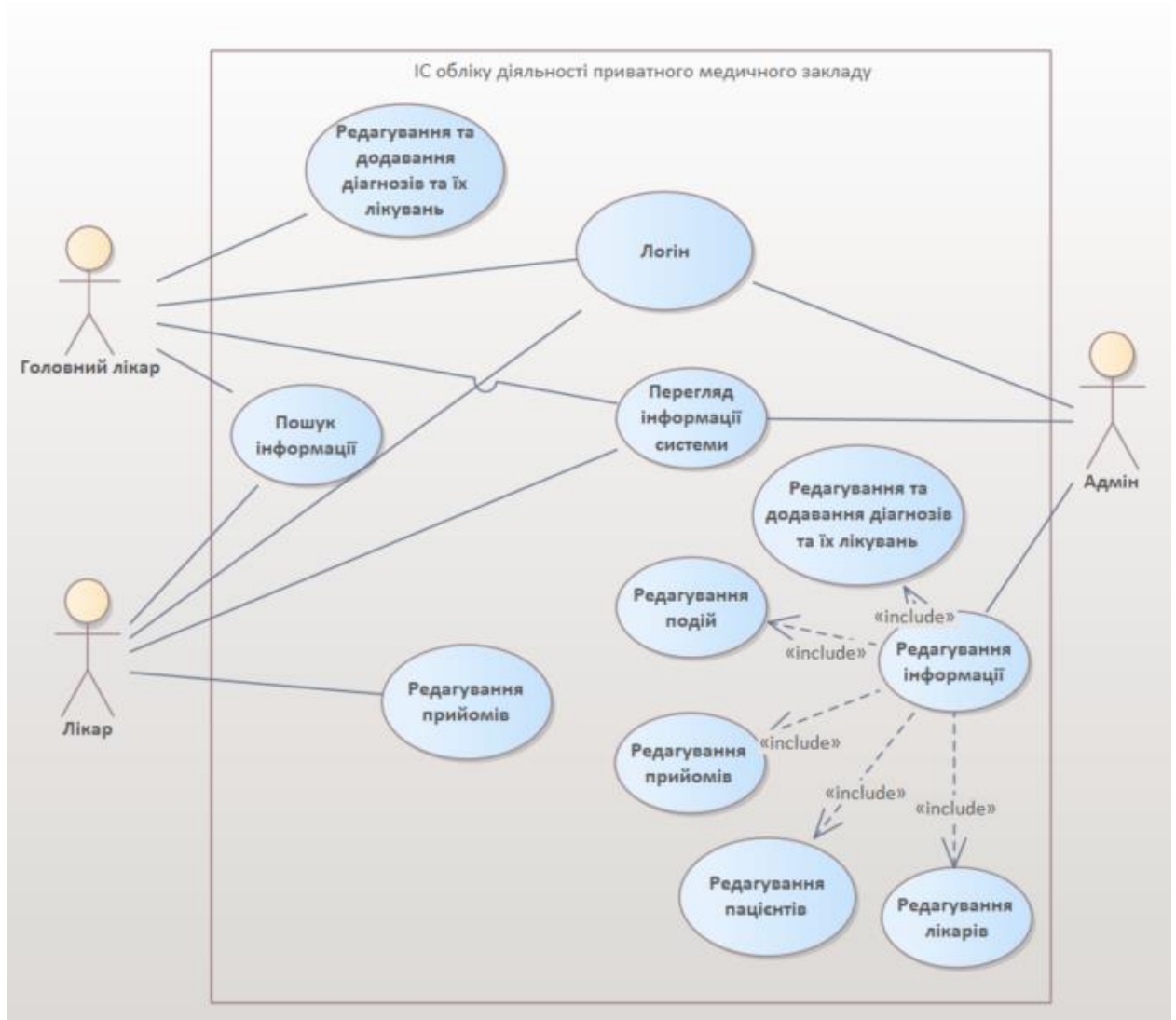


Рисунок 2.6 – Діаграма прецендентів

Джерело: складено автором

1. Користувач заповнює форму пошуку інформації:

Нормальний сценарій:

- 1) користувач вводить пошуковий запит;
- 2) після введення запиту, користувач натискає кнопку «пошук»;
- 3) система здійснює пошук на стороні серверу;
- 4) сервер повертає результати пошуку на клієнт;
- 5) система надає користувачеві результати пошуку.

Альтернативний сценарій:

- якщо система нічого не знаходить – вона повідомляє про це користувача відповідним повідомленням.

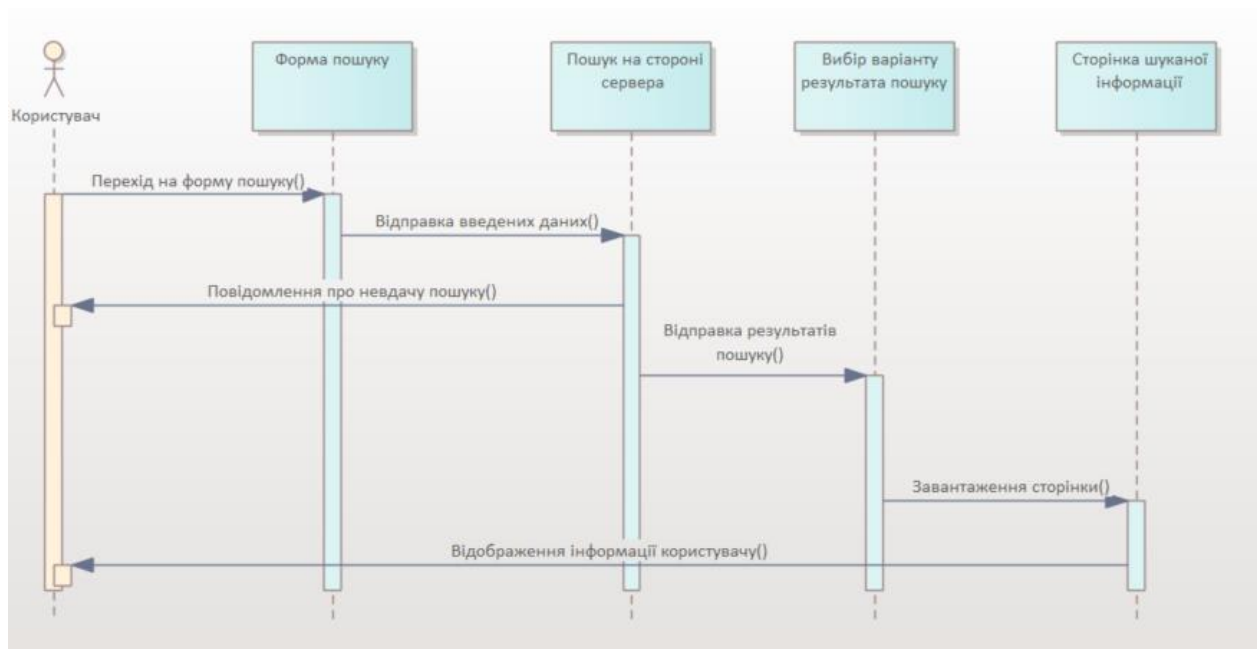


Рисунок 2.7 – Діаграма послідовності прецеденту «Пошук інформації»

Джерело: складено автором

2. Користувач здійснює вхід у систему:

Нормальний сценарій:

- 1) Користувач переходить на сайт;
- 2) Користувач вводить дані у форму логіну;
- 3) Натиснення користувачем кнопки відправки;
- 4) Верифікація на стороні серверу;
- 5) Перехід користувача у систему.

Альтернативний сценарій:

- користувач вводить неправильні дані;
- система не знаходить даного користувача.

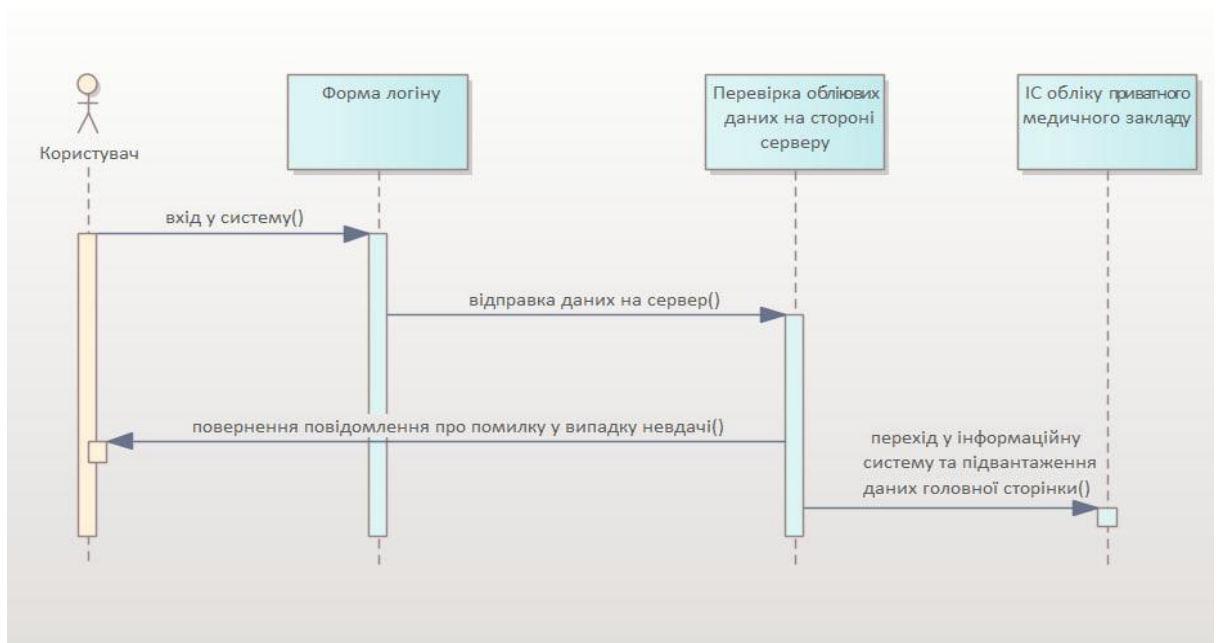


Рисунок 2.8 – Діаграма послідовності прецеденту «Входу у систему»

Джерело: складено автором

3. Користувач переглядає веб-сторінку інформаційної системи:

Нормальний сценарій:

- 1) Користувач за допомогою панелі навігації обирає веб-сторінку;
- 2) Система переадресовує користувача на дану сторінку та відображає її.

Альтернативний сценарій:

- Даний користувач не має прав для перегляду інформації на цій сторінці.

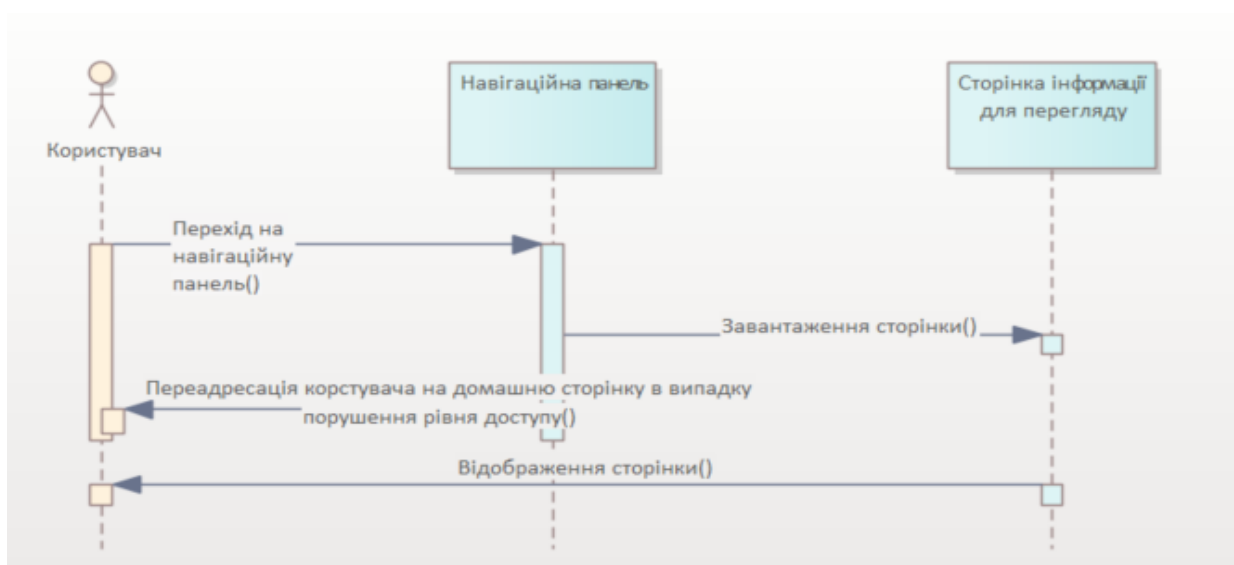


Рисунок 2.9 – Діаграма послідовності прецеденту «Перегляд веб-сторінок»

Джерело: складено автором

4. Користувач редагує інформацію у системі:

Нормальний сценарій:

- 1) Користувач обирає в навігаційні панелі меню редагування інформації;
- 2) Користувач вводить дані у форму;
- 3) Після введення даних, натискає кнопку підтвердження;
- 4) Валідація на стороні клієнта;
- 5) Збереження оновленої інформації у базі даних;
- 6) Повідомлення про успішну зміну даних.

Альтернативний сценарій:

- Користувач ввів дані неправильно;
- Користувач не має права на редагування даної інформації.

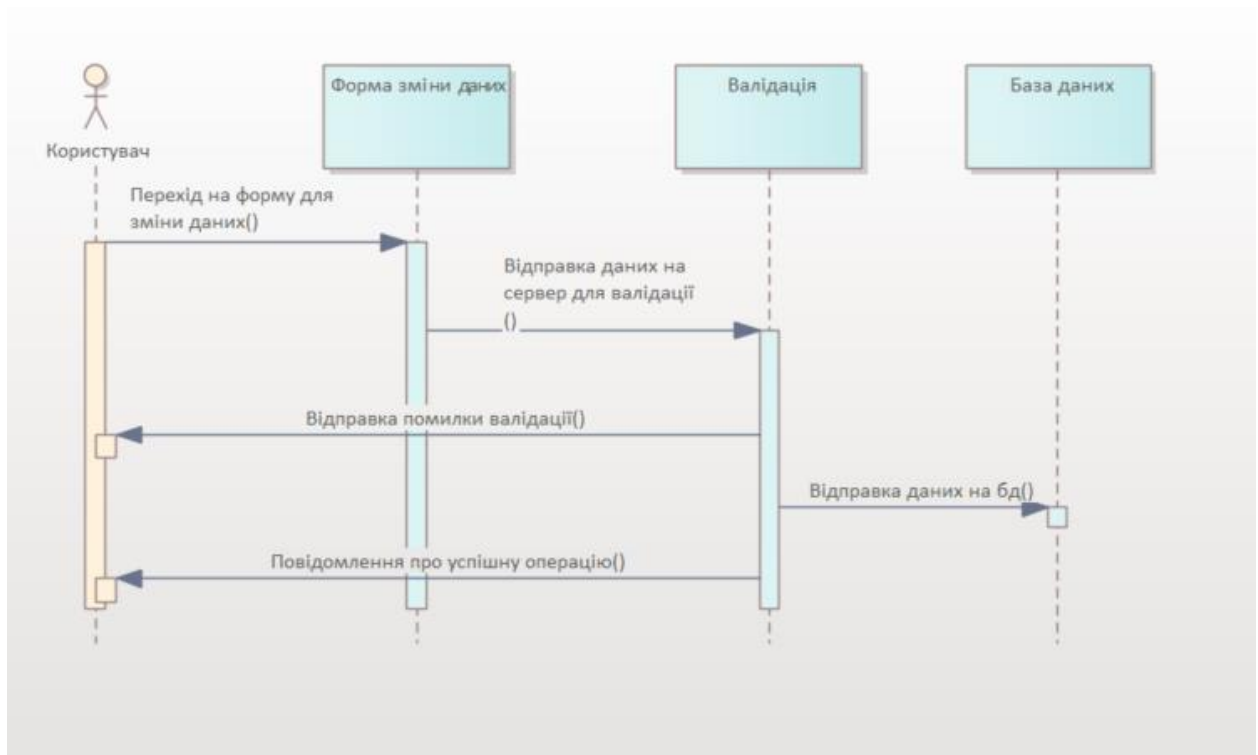


Рисунок 2.10 – Діаграма послідовності прецеденту «Зміна даних у системі»

Джерело: складено автором

Отже, для кожного з продемонстрованих прецедентів було визначено базовий та альтернативний сценарій виконання процесів прецедентів.

2.3.2. Моделювання структури застосунку

Моделювання структури дозволяє чітко визначити, як компоненти системи взаємодіють між собою, що сприяє кращому розумінню її архітектури. Ієрархічна структура блоків полегшує розробку, оскільки окремі компоненти можуть розроблятися паралельно та інтегруватися пізніше. Зрозуміла структура системи допомагає забезпечити її легкість підтримки та майбутнього розвитку, зокрема внесення змін або додавання нового функціоналу.

Діаграма визначення блоків

Діаграма визначення блоків системи розбиває веб-застосунок на основні компоненти та підсистеми, які взаємодіють між собою для забезпечення функціональності системи.

Головний блок діаграми «Медична система» містить такі блоки: «Сторінки», «База даних», «Сервер» та «Адміністарція». Кожен з даних блоків виконує свою функцію у системі.

Сторінки веб-застосунку приватного медичного закладу – це інтерфейси, які надають користувачеві можливість взаємодії з системою. Їхнє призначення – відображення інформації та надання інтерфейсів для редагування та створення нових даних, таких як відвідини пацієнтів, події, тощо.

Сервер виконує роль центру логічних операцій веб-додатку. Він є проміжною ланкою між клієнтською стороною (користувачем) та іншими допоміжними сервісами, такими як база даних. Сервер здійснює обробку та валідацію отриманих від користувача даних, опрацьовує запити до бази даних та відсилає їх на клієнтську сторону для подальшого показу на сторінках застосунку. Сервер здійснює функцію безпеки системи та встановлює рівні доступу користувачів.

База даних системи обліку приватного медичного закладу має функцію ефективного, зручного та надійного зберігання даних та операцій з ними. До таких даних належить інформація про: лікарів закладу, пацієнтів та історії їхніх відвідин, прийомів, подій, тощо.

Адміністрація у інформаційній системі закладу виконує роль наглядача над системою та її процесами. Вона наглядає над цілісністю даних, спостерігає за діяльністю лікарів, може переглядати інформації користувачів. Адміністрація

може додавати, видаляти та змінювати користувачів системи. Адміністратори здійснюють контроль над подіями у закладі.

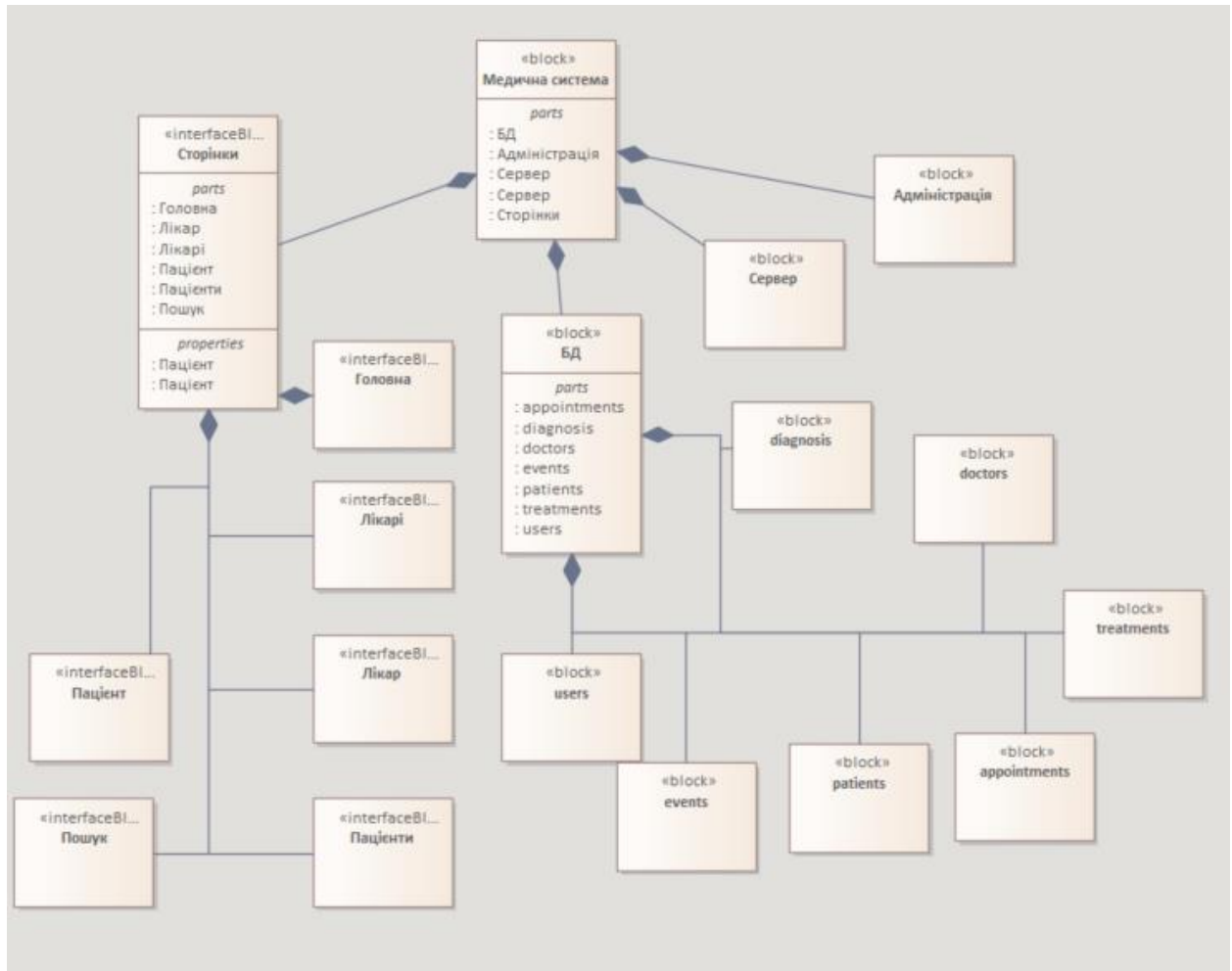


Рисунок 2.11 – Діаграма визначення блоків

Джерело: складено автором

Діаграма внутрішніх блоків

Діаграма внутрішніх блоків дозволяє детально розглянути внутрішню структуру системи, ідентифікувати основні компоненти, підсистеми та їх взаємодію.

Користувач системи може обрати відповідний пункт меню на панелі навігації у сторінці пацієнта. Далі він може ввести дані у надану йому форму, надаючи дані про прийом. Після цього, він підтверджує відправку даних на сервер, де ці дані верифікуються та обробляються. Після успішної обробки даних, сервер відправляє їх до бази даних на збереження та надсилає на клієнтську сторону повідомлення про успіх операції. В результаті, користувач системи

отримує повідомлення про успішне створення запису пацієнта на прийом до лікаря.

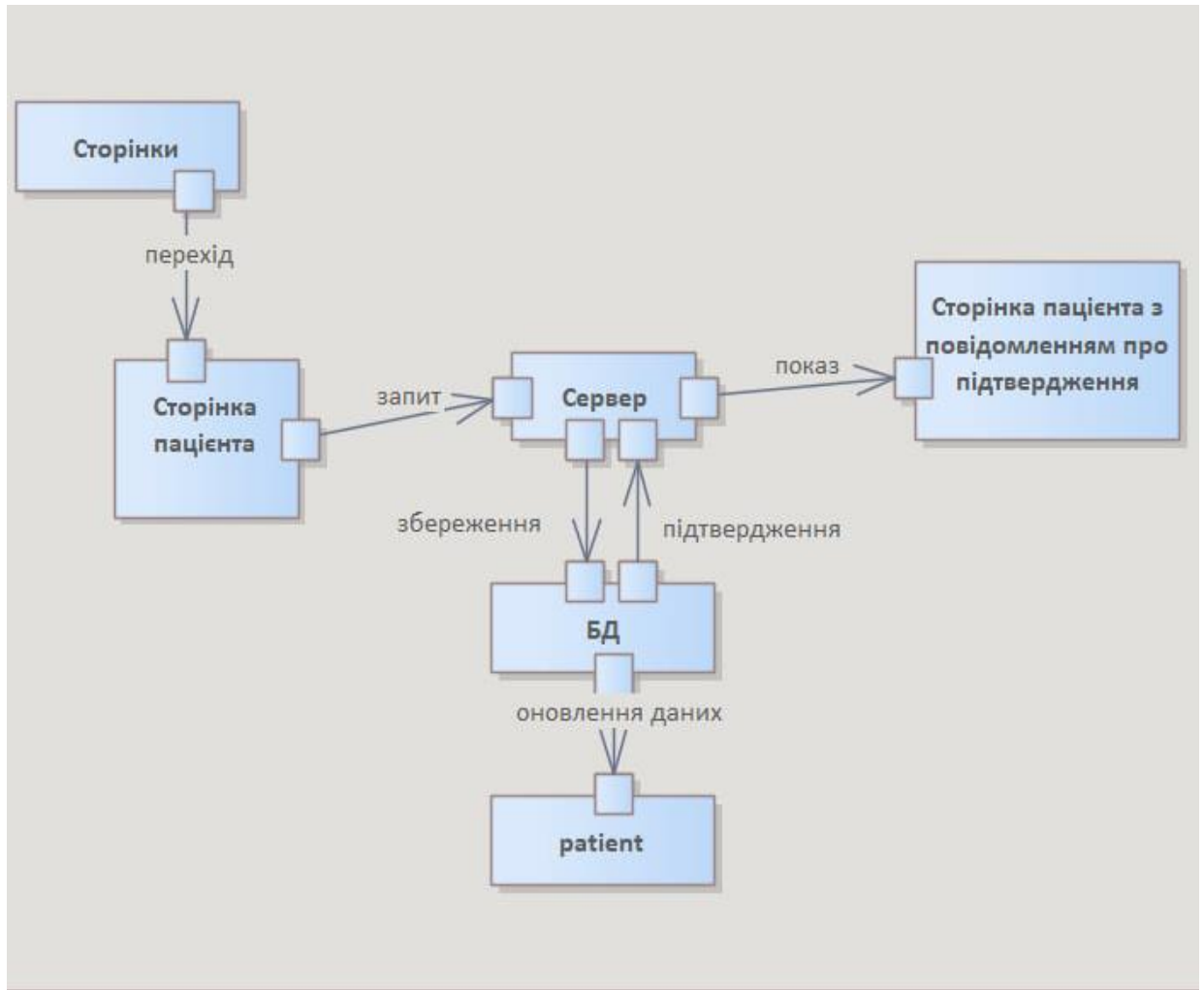


Рисунок 2.12 – Діаграма внутрішніх блоків обробки процесу заповнення форми «оформлення прийому у лікаря»

Джерело: складено автором

Діаграма класів

Діаграма класів – це структурна діаграма, яка відображає статичну структуру системи, ідентифікує класи, їх атрибути та методи, а також зв'язки між класами.

Діаграма класів дозволяє зорієнтуватися в структурі системи, визначити відповідальності класів та зв'язки між ними. Це сприяє оптимізації розробки, забезпеченню гнучкості системи та підтримці її масштабування. Крім того, вона слугує основою для подальшого проєктування та реалізації системи, забезпечуючи її стабільність та надійність.

Перший клас «Event» має атрибути eventId, title, description. Методи класу надають можливість створення нових подій та отримання інформації про вже існуючі. Клас пов'язаний з класом «Doctor».

Наступний клас «Doctor» містить атрибути, такі як doctorId, name, phone, post тощо. Методи даного класу дозволяють створювати нових лікарів, оновлювати інформацію існуючих, видаляти лікарів з системи та отримувати інформації про них. Він також пов'язаний з класами «User» та «Appointment».

Клас «Appointment» містить атрибути appointmentId, doctorId, patientId тощо. Методи даного класу надають можливість створення нових, редагування існуючих та отримання інформації про відвідини. Цей клас пов'язаний з класами «Doctor», «Patient», «Diagnosis» та «Treatment».

Клас «Patients», що містить такі атрибути: patientId, name, gender тощо. Методи цього класу дозволяють отримати інформацію про пацієнта закладу, додати нового та редагувати наявного. Цей клас має зв'язок з класом «Appointment» та «Doctor».

Клас «Diagnosis» містить атрибути title, description, тощо. Методи даного класу надають можливість створення нових, редагування існуючих та отримання інформації про діагнози. Цей клас пов'язаний з класом «Appointment».

Останній клас «Treatment», що містить такі атрибути: title, description, duration тощо. Методи цього класу дозволяють отримати весь спектр необхідних операцій з лікуваннями, які призначає лікар під час візиту. Цей клас має зв'язок з класом «Appointment».

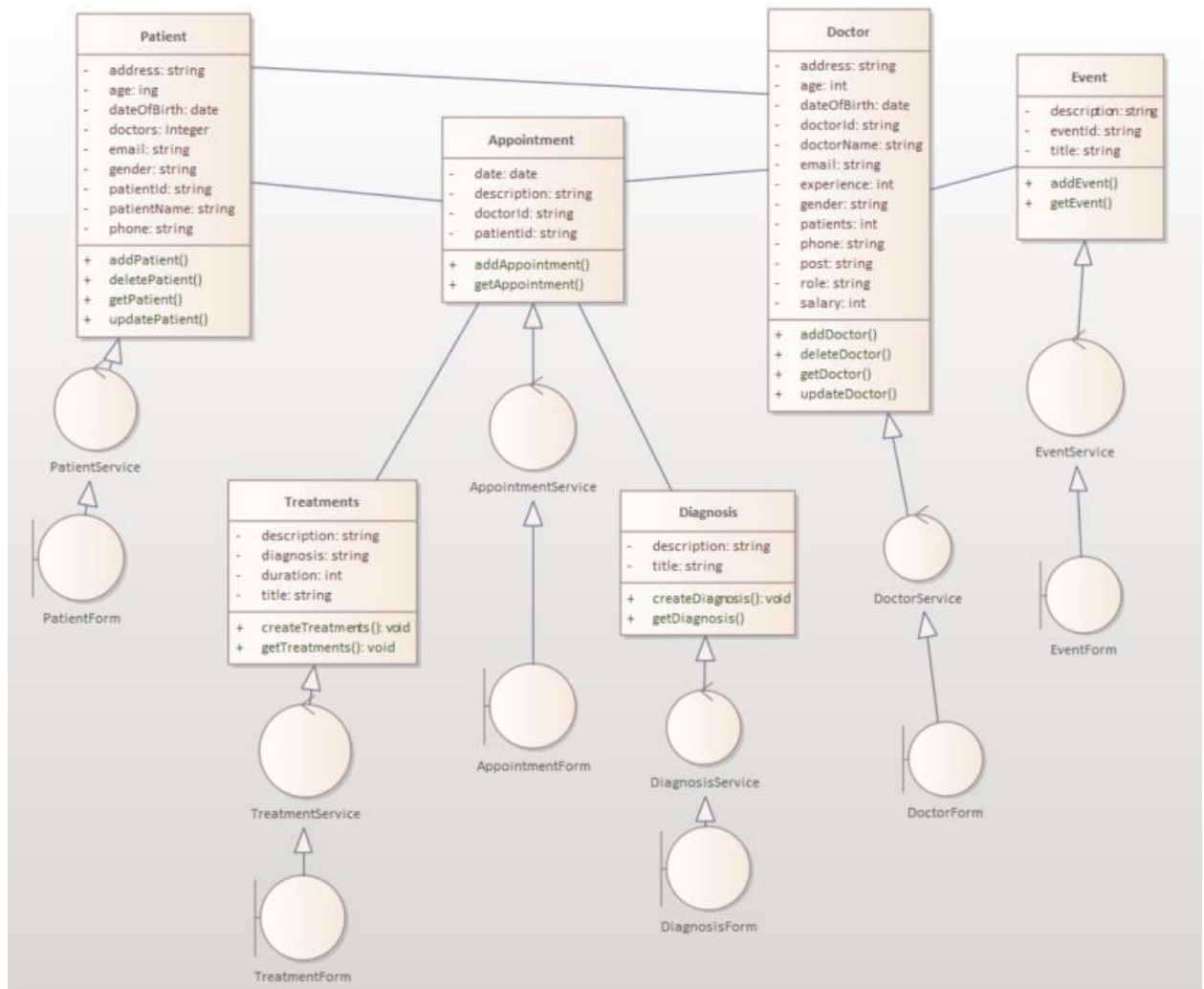


Рисунок 2.13 – Діаграма класів

Джерело: складено автором

Отримавши змодельовану структуру веб-застосунку можна побачити загальні принципи ієрархії системи та її компонентів, що суттєво полегшить розробку додатку в подальшому.

2.3.3. Вимоги за компонентами системи

Трасування вимог до системи є важливим етапом у процесі розробки, який забезпечує відповідність реалізованої системи вимогам, встановленим на початкових стадіях. Цей процес включає у себе встановлення взаємозв'язків між різними елементами системи та початковими вимогами до неї.

На початковому етапі визначаються ключові вимоги до системи веб-додатку, такі як функціональні та нефункціональні вимоги, бізнес-вимоги та інші. Після цього кожна вимога трасується через всі стадії життєвого циклу розробки, від аналізу та проєктування до реалізації та тестування, щоб переконатися, що вона врахована в кінцевому продукті.

Трасування вимог допомагає забезпечити повноту та узгодженість реалізованої системи з початковими вимогами, а також виявити потенційні прогалини або конфлікти між різними частинами системи. Цей процес є важливим для забезпечення якості та надійності розробленого веб-додатку, а також для задоволення потреб користувачів та стейкхолдерів.

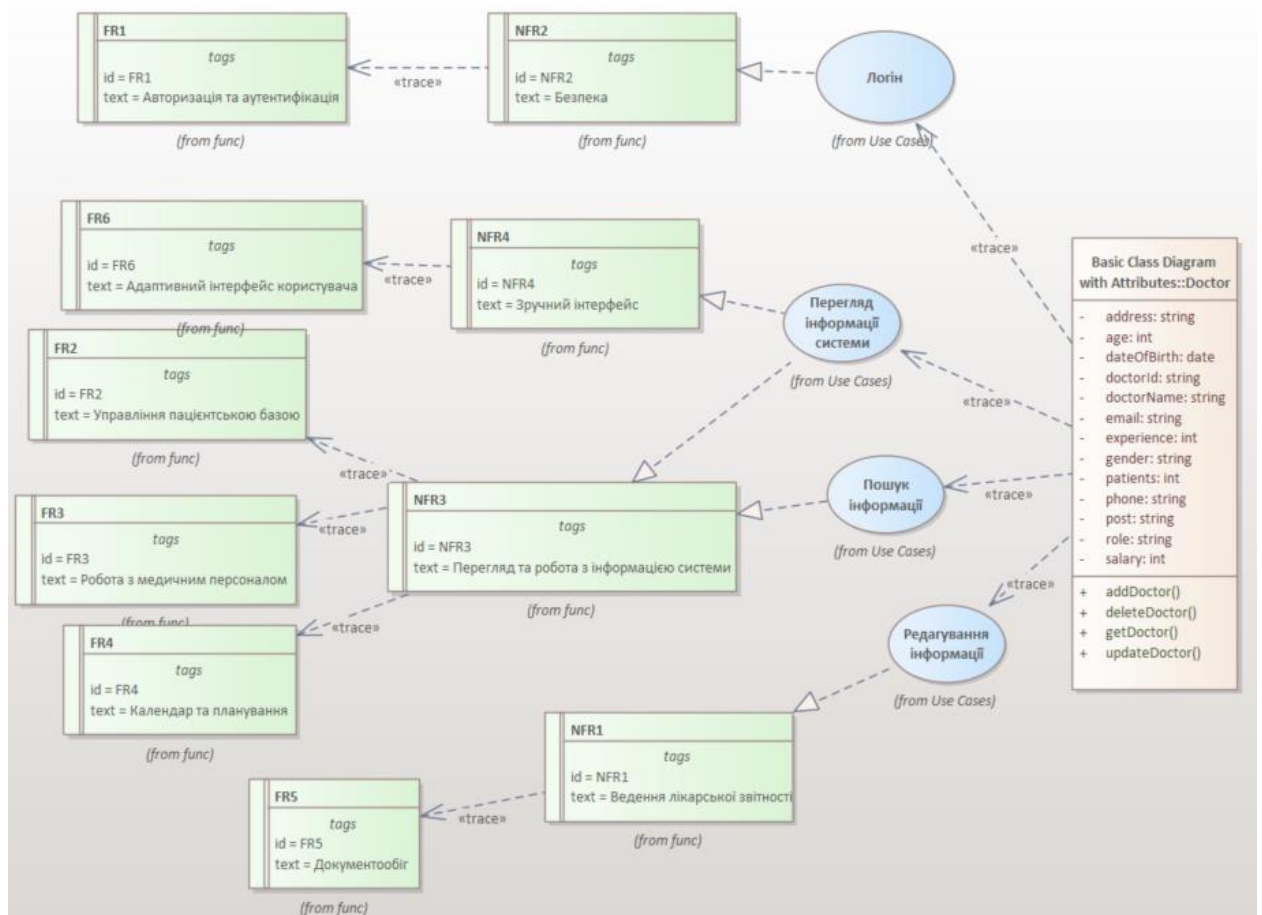


Рисунок 2.14 – Діаграма трасування вимог веб-додатку

Джерело: складено автором

Target +	func::NFR1	func::NFR2	func::NFR3	func::NFR4
+ Source				
Use Cases::Логін		↑		
Use Cases::Перегляд інформації системи			↑	↑
Use Cases::Пошук інформації			↑	
Use Cases::Редагування інформації	↑			

Рисунок 2.15 – Матриця взаємозв’язків діаграми трасування вимог веб-додатку

Джерело: складено автором

Дані вимоги стануть основою в проєктуванні та допоможуть в подальшій ефективній розробці веб-застосунку обліку діяльності приватного медичного закладу.

РОЗДІЛ 3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1 Інформаційне забезпечення

Інформаційне забезпечення системи передбачає використання комплексу елементів, які сприяють оптимальному функціонуванню та обслуговуванню веб-додатку для обліку приватного медичного закладу.

Для зберігання даних використовується MongoDB, яка є NoSQL базою даних, що забезпечує гнучкість і масштабованість.

Вибір MongoDB обґрунтований її здатністю ефективно працювати з великими обсягами даних, гнучкою моделлю зберігання, високою продуктивністю та простотою інтеграції з Node.js через NestJS.

Інфологічна модель бази даних є високорівневим описом структури даних, що використовується в інформаційній системі веб-застосунку. Цей рівень абстракції не враховує конкретних технічних реалізацій, таких як типи баз даних чи конкретний запит, але фокусується на сутностях та відносинах між ними. В даному проєкті база даних містить такі документи як лікар, пацієнт, подія, прийом пацієнта, діагноз та лікування.

Далі представлені описи всіх атрибутів об'єктів веб-застосунку (див. табл. 3.1-3.4).

Таблиця 3.1 – Опис атрибутів інформаційного об'єкту «Пацієнт»

Найменування	Ідентифікатор у системі	Тип даних	Бізнес-правила	
			Поле ключове	Обов'язкове поле
Унікальний ідентифікатор	patientId	String	+	так
Ім'я	firstName	String	—	так
Прізвище	lastName	String	—	так
Стать	gender	String	—	так
Дата народження	birthDate	Date	—	так
Електронна пошта	email	String	—	так
Номер мобільного телефону	phone	String	—	так

Найменування	Ідентифікатор у системі	Тип даних	Бізнес-правила	
			Поле ключове	Обов'язкове поле
Адреса	address	String	—	так
Опис	description	String	—	ні
Лікарі	doctors	Array	—	ні
Прийоми у лікаря	appointments	Array	—	ні
Діагнози	diagnosis	Array	—	ні

Джерело: складено автором

Таблиця 3.2 – Опис атрибутів інформаційного об'єкту «Лікар»

Найменування	Ідентифікатор у системі	Тип даних	Бізнес-правила	
			Поле ключове	Обов'язкове поле
Унікальний ідентифікатор	doctortId	String	+	так
Ім'я	firstName	String	—	так
Прізвище	lastName	String	—	так
Стать	gender	String	—	так
Дата народження	birthDate	Date	—	так
Електронна пошта	email	String	—	так
Номер мобільного телефону	phone	String	—	так
Адреса	address	String	—	так
Посада	post	String	—	так
Досвід	experience	Number	—	так
Рівень доступу	role	String	—	так
Опис	description	String	—	ні
Пацієнти	patients	Array	—	ні
Прийоми пацієнтів	appointments	Array	—	ні

Джерело: складено автором

Таблиця 3.3 – Опис атрибутів інформаційного об'єкту «Прийоми»

Найменування	Ідентифікатор у системі	Тип даних	Бізнес-правила	
			Поле ключове	Обов'язкове поле
Унікальний ідентифікатор	appointmentId	String	+	так
Лікар	doctor	Object	—	так
Пацієнт	patient	Object	—	так
Дата	date	Date	—	так
Час	time	Date	—	так

Найменування	Ідентифікатор у системі	Тип даних	Бізнес-правила	
			Поле ключове	Обов'язкове поле
Статус	status	String	—	так
Ціна	cost	Number	—	ні
Лікування	treatment	Object	—	ні
Висновок	conclusion	String	—	ні
Діагноз	diagnosis	String	—	ні

Джерело: складено автором

Таблиця 3.4 – Опис атрибутів інформаційного об'єкту «Подія»

Найменування	Ідентифікатор у системі	Тип даних	Бізнес-правила	
			Поле ключове	Обов'язкове поле
Унікальний ідентифікатор	eventId	String	+	так
Заголовок	title	String	—	так
Дата	date	Date	—	так
Час	time	Date	—	так
Опис	description	String	—	ні

Джерело: складено автором

Таблиця 3.5 – Опис атрибутів інформаційного об'єкту «Діагноз»

Найменування	Ідентифікатор у системі	Тип даних	Бізнес-правила	
			Поле ключове	Обов'язкове поле
Унікальний ідентифікатор	diagnosisId	String	+	так
Заголовок	title	String	—	так
Опис	description	String	—	так

Таблиця 3.6 – Опис атрибутів інформаційного об'єкту «Лікування»

Найменування	Ідентифікатор у системі	Тип даних	Бізнес-правила	
			Поле ключове	Обов'язкове поле
Унікальний ідентифікатор	treatmentId	String	+	так
Заголовок	title	String	—	так
Тривалість	duration	Number	—	так
Діагноз	diagnosis	Object	—	так
Опис	description	String	—	так

Джерело: складено автором

3.2 Технічне забезпечення

Веб-додаток обліку діяльності приватного медичного закладу передбачає наявність віддаленого орендованого серверу, на якому, передбачається, що буде розміщено ядро системи, база даних та всі необхідні для неї дані. Доступ клієнтів до системи буде здійснюватися через використання веб-додатку з доступом до мережі Інтернет, що дозволить користувачам системи (лікарям та адміністрації), легко отримувати доступ до інформації.

Основні переваги такого підходу:

1. Відсутність необхідності у власному серверному обладнанні – медичному закладу не потрібно купувати, встановлювати та обслуговувати власні сервери, що значно знижує початкові інвестиції.

2. Зменшення витрат на технічне обслуговування – хостинг-провайдер бере на себе всі технічні питання, включаючи оновлення програмного забезпечення, моніторинг та усунення несправностей.

3. Гнучкість у масштабуванні ресурсів – віддалений хостинг дозволяє швидко збільшити або зменшити обчислювальні ресурси в залежності від потреб медичного закладу, що забезпечує оптимальне використання ресурсів.

4. Підготовленість до пікових навантажень – легко масштабувати серверні потужності для обробки пікових навантажень, наприклад, під час епідемій або інших надзвичайних ситуацій.

5. Безперервність роботи – віддалені хостинг-провайдери зазвичай забезпечують високу доступність своїх послуг за допомогою резервних систем живлення, охолодження та інфраструктури.

6. Рівень безпеки – провайдери віддаленого хостингу інвестують значні ресурси у забезпечення безпеки даних, включаючи захист від кібератак, шифрування даних та контроль доступу [18].

7. Мобільні рішення – багато хостинг-провайдерів пропонують рішення, оптимізовані для мобільних пристроїв, що дозволяє лікарям і персоналу працювати зі смартфонів та планшетів.

8. Віддалений доступ – лікарі та адміністративний персонал можуть отримувати доступ до системи з будь-якої точки світу, що забезпечує гнучкість і

зручність роботи, особливо у випадках дистанційної роботи або телемедицини [17].

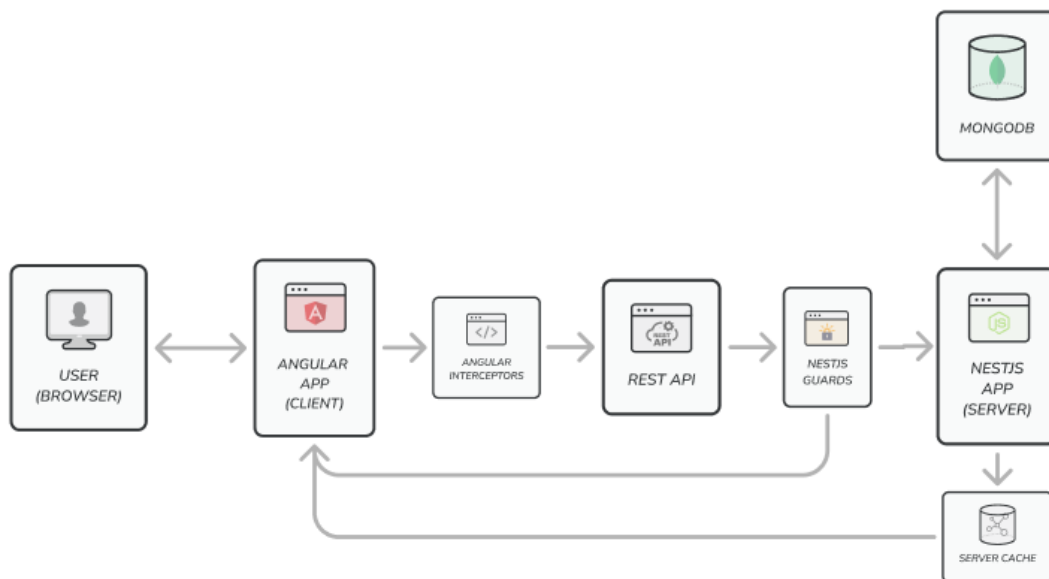


Рисунок 3.1 – Загальна схема автоматизації

Джерело: складено автором

- User (Browser) – це безпосередньо користувач, який користується браузером для взаємодії з системою.
- Angular app (Client) – клієнтська частина, яка взаємодіє з користувачем та сервером, є графічним представленням системи.
- Angular interceptors – перехоплювачі на стороні клієнта, які здійснюють налаштування запитів на сервер, в тому числі й додавання токенів безпеки до HTTP-запиту.
- REST API – це інтерфейс, котрий слугує мостом між такими двома частинами як: клієнт та сервер.
- NestJS guards – це захист серверу від неавторизованих користувачів. Також здійснює перевірку на рівень доступу користувача.
- NestJS app (Server) – серверна частина системи. Здійснює обробку даних, взаємодію з базою даних та клієнтом.
- Server cache – кеш, який зберігається на стороні серверу.
- MongoDB – база даних, яка здійснює широкий спектр взаємодії з даними, зберігає всю необхідну інформацію системи.

3.3 Програмне забезпечення

Обрання відповідних технологій та інструментів має вирішальне значення на шляху до створення ефективного та надійного програмного продукту. Застосування сучасних та перевірених засобів дозволяє досягти високої продуктивності, гнучкості та зручності у використанні системи. Цей вибір визначає не лише швидкість розвитку, а й майбутні можливості розширення та підтримки продукту.

Для розробки веб-додатку обліку приватного медичного закладу було використано такі технології:

1. Angular – популярний фреймворк для розробки веб-додатків, що базується на TypeScript. Він надає широкий набір інструментів для швидкої та ефективної розробки клієнтського застосунку. Angular пропонує компонентний підхід до розробки, де весь інтерфейс розділяється на невеликі, незалежні компоненти, що полегшує розуміння та підтримку коду. Він також має вбудовану підтримку для роботи з HTTP-запитами, маршрутизацією, валідацією форм та іншими типовими завданнями веб-розробки. Angular дозволяє створювати потужні односторінкові додатки з багатим функціоналом та високою продуктивністю.

2. PrimeUI – набір готових компонентів та інтерфейсів користувача для створення веб-додатків з використанням HTML, CSS і JavaScript. Він базується на популярному фреймворку PrimeFaces. PrimeUI містить широкий вибір компонентів, таких як таблиці, форми, панелі, меню, графіки та багато іншого, що робить його ідеальним інструментом для швидкого створення сучасних та стильних веб-додатків.

3. Node.js – відкрите середовище виконання JavaScript, яке дозволяє створювати швидкі та масштабовані серверні застосунки. Node.js базується на рушії JavaScript V8, що робить його дуже швидким та ефективним у виконанні коду. Він дозволяє використовувати JavaScript як мову програмування для серверного та клієнтського коду, що спрощує розробку та підтримку проєктів.

4. NestJS – це прогресивний фреймворк для розробки серверних додатків на Node.js, побудований на основі TypeScript. NestJS спирається на популярні патерни проєктування, такі як Dependency Injection та модульність, що дозволяє розробникам легко організовувати та керувати складністю коду. Він також має вбудовану підтримку для розробки API, обробки запитів та відповідей, аутентифікації, авторизації та багатьох інших функцій, що робить його ідеальним вибором для створення серверних застосунків.

5. MongoDB – документоорієнтована база даних, яка зберігає дані у вигляді документів у форматі JSON. Вона відрізняється від традиційних реляційних баз даних тим, що не вимагає строгої схеми даних, що робить її дуже гнучкою та зручною для роботи з нереляційними даними. MongoDB має широкий функціонал, включаючи можливість проводити операції CRUD (створення, читання, оновлення, видалення) з даними, агрегаційні операції, текстовий пошук, індексацію та інші.

6. Mongoose – це об'єктно-документний моделювальник для Node.js та MongoDB. Він надає простий та елегантний спосіб взаємодії з базою даних MongoDB, дозволяючи визначати моделі даних за допомогою схем та використовувати їх для взаємодії з базою даних. Mongoose дозволяє розробникам визначати структуру та валідувати дані, що зберігаються в базі даних, а також виконувати різноманітні операції, такі як створення, зчитування, оновлення та видалення даних, зручним та зрозумілим способом.

7. Figma – веб-сервіс для дизайну та прототипування інтерфейсів користувача. Цей інструмент дозволяє дизайнерам, розробникам та іншим учасникам команди спільно працювати над проєктами в реальному часі, без необхідності встановлення додаткових програм на комп'ютер. Figma має широкий функціонал, включаючи можливість створювати векторні малюнки, макети, макети інтерфейсів, а також робити їх взаємодійними прототипами.

8. WebStorm – це інтегроване середовище розробки від компанії JetBrains, призначене для розробки веб-додатків на JavaScript, HTML та CSS, а також для роботи з Node.js. Воно надає розробникам широкий набір інструментів для підтримки різноманітних технологій та рамок, таких як Angular. WebStorm має вбудовані інструменти для автоматичного завершення коду, візуального

редактора CSS та HTML, інструментів для роботи з системами контролю версій, такими як Git, і багато інших корисних функцій, що допомагають розробникам під час написання, тестування та налагодження коду.

9. Postman – це інструмент для тестування та розробки API, який надає можливість створювати, тестувати та документувати HTTP-запити та відповіді. Він дозволяє розробникам швидко та легко взаємодіяти з API, надсилати різні види запитів (GET, POST, PUT, DELETE тощо), передавати параметри, налаштовувати заголовки, створювати автоматизовані тести та багато іншого.

Використання цих технологій дозволило швидко та ефективно реалізувати всі цілі розробки веб-додатку:

- Безпека – NestJS дозволяє використовувати різноманітні заходи безпеки, такі як JWT-автентифікація, маркерування місць вразливостей та використання бібліотек для валідації введених даних. Це дозволяє підвищити безпеку серверної частини додатку та захистити її від різних видів атак.

- Ефективність роботи - Angular дозволяє реалізувати ефективний та продуктивний фронтенд, забезпечуючи швидку та реактивну взаємодію з користувачем. NestJS, з іншого боку, забезпечує швидке та ефективне виконання серверної логіки завдяки використанню Node.js та оптимізованих архітектурних рішень. MongoDB забезпечує ефективне зберігання та доступ до даних завдяки своїй документоорієнтованій архітектурі, що дозволяє швидко виконувати операції зберігання та доступу до інформації.

- Масштабованість – NestJS дозволяє легко масштабувати серверну частину додатку завдяки своїм архітектурним рішенням та підтримці Node.js. MongoDB, як документоорієнтована база даних, також дозволяє гнучко масштабувати сховище даних в залежності від потреб проєкту.

- Безперебійність – Node.js забезпечує асинхронне та подійно-орієнтоване програмування, що дозволяє системі обробляти багато запитів одночасно без блокування. MongoDB та Mongoose забезпечують надійне зберігання та доступ до даних, що сприяє безперебійному функціонуванню системи.

- Зручність розробки – WebStorm надає зручне та продуктивне середовище розробки для роботи з JavaScript, HTML та CSS, що полегшує написання та налагодження коду. Postman допомагає тестувати та взаємодіяти з API зручним

та інтуїтивно зрозумілим способом, що сприяє швидкій розробці та налагодженню серверних компонентів.

3.4 Результати реалізації веб-застосунку

Основою виконаної роботи є розробка та впровадження веб-застосунку для обліку діяльності приватного медичного закладу. Основні функціональні можливості додатку включають автоматизацію реєстрації пацієнтів, обробку медичних записів, планування візитів, моніторинг лікування, а також адміністративними процесами.

Використані в ході впровадження набори вхідних даних наведено у окремому Додатку до ВБП. Набори даних включають фрагменти реальних медичних записів, дані про пацієнтів, графіки роботи лікарів та іншу необхідну інформацію.

Результати тестування показали, що розроблений веб-застосунок відповідає поставленим вимогам, забезпечує високу точність обробки даних та зручність використання для медичного персоналу. Застосунок успішно інтегрований з існуючими базами даних та забезпечує безпечний доступ до медичної інформації.

Аналіз потенційної практичної цінності отриманих результатів показав, що впровадження даного додатку може значно підвищити ефективність роботи медичного закладу, скоротити час на обробку даних та знизити ризики помилок. У порівнянні з існуючими на ринку альтернативами, запропонована система є більш гнучкою та адаптованою до специфічних потреб приватного медичного закладу.

Результати аналізу ступеня готовності до реального використання та перспектив впровадження показали, що підсистема є готовою до пілотного впровадження в медичних закладах. На основі отриманих даних були надані рекомендації щодо подальших досліджень та удосконалень, зокрема щодо розширення функціоналу системи для підтримки додаткових медичних процесів та інтеграції з іншими медичними системами.

Потенційна практична цінність даного веб-додатку полягає в її здатності оптимізувати роботу медичного закладу, покращити якість обслуговування пацієнтів, знизити витрати на адміністрування та підвищити точність обробки медичних даних. Це сприятиме підвищенню задоволеності пацієнтів, скороченню часу на надання медичних послуг та покращенню загального рівня медичного обслуговування.

Один з основних напрямків розвитку веб-додатку полягає у розширенні її функціоналу. Зокрема, можна додати можливості:

- Розширення функцій електронної медичної картки – включення додаткових модулів для детального обліку історії хвороби, включаючи лабораторні результати, рентгенівські знімки та інші медичні зображення.
- Інтеграції з лабораторними та іншими медичними системами – це дозволить автоматично отримувати результати аналізів та обстежень, що значно скоротить час на обробку даних та знизить ризик помилок.
- Аналіз даних та звітність – розширення можливостей аналізу даних та створення детальних звітів допоможе адміністрації медичного закладу приймати обґрунтовані управлінські рішення. Система може включати інструменти для аналізу фінансових показників, продуктивності медичного персоналу, задоволеності пацієнтів та інших ключових метрик.

Розроблений веб-додаток має значний потенціал для підвищення ефективності роботи приватних медичних закладів та може бути рекомендована для впровадження в інших організаціях. Її перспективи розвитку включають розширення функціоналу, розробку мобільного додатку, інтеграцію з іншими системами, застосування машинного навчання та підвищення безпеки даних. Впровадження цих удосконалень дозволить ще більше оптимізувати роботу медичних закладів, покращити якість медичних послуг та забезпечити високу задоволеність пацієнтів.

ВИСНОВКИ

У процесі розробки веб-застосунку для обліку діяльності приватного медичного закладу було досягнуто головної мети проєкту – створення ефективного застосунку, який забезпечує автоматизацію ключових процесів медичного закладу, підвищуючи його ефективність та зручність для медичного персоналу та пацієнтів.

Функціональні можливості веб-застосунку:

- Автоматизація реєстрації пацієнтів – реалізовані форми для швидкої та зручної реєстрації пацієнтів, що дозволяє зменшити час на цей процес та знизити ймовірність помилок.
- Обробка медичних записів – система забезпечує зручне введення, зберігання та обробку медичних записів, що підвищує точність та доступність медичної інформації.
- Планування візитів - інтегрований календар дозволяє легко планувати візити пацієнтів, оптимізуючи розклад лікарів та зменшуючи час очікування.
- Моніторинг лікування – система дозволяє відстежувати прогрес лікування пацієнтів, забезпечуючи лікарів актуальною інформацією для прийняття рішень.
- Адміністративні процеси – автоматизація адміністративних задач, таких як облік лікарів, забезпечує ефективне управління діяльності працівників закладу.

Веб-застосунок розроблений з використанням технологій, таких як NestJS, Angular, MongoDB, що забезпечує високу продуктивність, масштабованість та надійність. Підсистема успішно інтегрована з існуючими базами даних закладу, забезпечуючи безпечний та швидкий доступ до медичної інформації. Реалізовані механізми шифрування та автентифікації користувачів забезпечують високий рівень безпеки медичних даних.

До переваг можна віднести:

- Підвищення ефективності – впровадження підсистеми значно підвищує ефективність роботи медичного закладу, знижує адміністративні витрати та покращує якість обслуговування пацієнтів.

- Гнучкість та адаптованість – система є гнучкою та легко адаптується до специфічних потреб приватного медичного закладу, що робить її конкурентоздатною на ринку.

- Зручність використання – інтуїтивно зрозумілий інтерфейс та зручність використання підсистеми сприяють швидкому освоєнню персоналом та підвищують продуктивність роботи.

Недоліком є потреба в постійній підтримці. Для забезпечення стабільної роботи системи необхідно регулярно проводити технічне обслуговування та оновлення.

Рекомендації щодо вдосконалення застосунку:

- Розширення функціоналу – впровадження додаткових функцій, таких як телемедицина, підтримка мобільних додатків, автоматичне нагадування про прийом ліків та візити, інтеграція з системами управління електронними медичними записами (EMR) та лабораторними інформаційними системами (LIS) значно покращить можливості системи та підвищить її корисність для медичного персоналу та пацієнтів.

- Автоматизація аналітичних процесів: Впровадження інструментів для автоматизованого аналізу даних, які допоможуть медичним закладам робити прогнози, оцінювати ефективність лікування та виявляти можливі проблеми на ранніх стадіях.

У процесі виконання кваліфікаційного бакалаврського проєкту я спроєктував та розробив ключові модулі підсистеми, включаючи клієнтську та серверну частини веб-застосунку. Провів тестування системи, аналіз її ефективності.

Виконання кваліфікаційного бакалаврського проєкту надало мені цінний досвід у розробці інформаційних систем, вдосконні технічних, покращило організаційні навички, а також зробити свій внесок у підвищення ефективності роботи приватних медичних закладів через створення зручної та функціонального веб-застосунку.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Health care information systems [Електронний ресурс] / Health Care Information Systems_ A Practical Approach for Health Care Management. Second Edition. – 2022. – Режим доступу до ресурсу: <https://www.slideshare.net/slideshow/health-care-information-systems-a-practical-approach-for-health-care-management-second-edition/253456213>.
2. Internet Archive [Електронний ресурс] / Electronic health records : understanding and using computerized medical records. – 2012. – Режим доступу до ресурсу: <https://archive.org/details/electronichealth00rich>.
3. Evergreens [Електронний ресурс] / Медичні інформаційні системи: огляд можливостей і приклади використання– 2020. – Режим доступу до ресурсу: <https://evergreens.com.ua/ua/articles/medical-information-systems.html>.
4. Health24 [Електронний ресурс] / Медична інформаційна система, що це? – 2021. – Режим доступу до ресурсу: <https://blog.h24.ua/uk/shho-take-mis/>.
5. Електронний архів КНУТД [Електронний ресурс] / Розробка медичної інформаційної системи для медичних закладів первинної ланки – 2021. – Режим доступу до ресурсу: <https://er.knutd.edu.ua/handle/123456789/19799>.
6. Проектування баз і сховищ даних: навч. посіб. – Ситник Н.В. К.:2004.
7. Проектування систем оброблення інформації: навч. посіб. – Береза А.М., Козак І.А. К.: КНЕУ, 2008.
8. ClickIT [Електронний ресурс] / Web Application Architecture: The Latest Guide 2024 – 2024. – Режим доступу до ресурсу: <https://www.clickittech.com/devops/web-application-architecture/>.
9. Інформаційні системи і технології в економіці: навч.посіб. - М.А. Сендзюка, М.Б. Вітер К. :КНЕУ, 2011.
10. Концепція та методології системного інжинірингу: навч.посіб. - Денісова О. О. :КНЕУ, 2019.
11. Dubuc N, Brière S, Corbin C, N'Bouke A, Bonin L, Delli-Colli N. Computerized care-pathways (CCPs) system to support person-centered, integrated, and proactive care in home-care settings. Inf Health Soc Care. 2021 – Режим доступу до ресурсу: . <https://doi.org/10.1177/2057158519856495>.

12. Petersen CL, Weeks WB, Norin O, Weinstein JN. Development and implementation of a person-centered, technology-enhanced care model for managing chronic conditions: cohort study. JMIR Mhealth Uhealth. 2019. – Режим доступу до ресурсу: <https://mhealth.jmir.org/2019/3/e11082/>.

13. Journal of Physics: Conference. Analysis and Design of Web-Based Health Service Information Systems (E-Health), in the Industrial Revolution Era 4.0. 2021 – Режим доступу до ресурсу: <https://iopscience.iop.org/article/10.1088/1742-6596/1764/1/012067/pdf>.

14. McCarthy, D. B., & Eastman, D. L. Electronic Medical Records and Their Impact on Patient Satisfaction and Healthcare Efficiency. Journal of Medical Informatics. 2020 – Режим доступу до ресурсу: [https://www.jmirs.org/article/S1939-8654\(20\)30011-4/abstract](https://www.jmirs.org/article/S1939-8654(20)30011-4/abstract).

15. EMCImed. [Електронний ресурс] 2024 – Режим доступу до ресурсу: <https://emci.ua>.

16. Інформаційні технології для медицини [Електронний ресурс] 2024 – Режим доступу до ресурсу: <https://www.infomed.ck.ua/>.

17. Shachi Sinha; Akash Deep Rai; Monica Subashini M Health Monitoring and Doctor on Call System. 2022 – Режим доступу до ресурсу: <https://ieeexplore.ieee.org/document/9792689>.

18. Garrett Wallace Brown, Gemma Bridge, Jessica Martini. The role of health systems for health security: a scoping review revealing the need for improved conceptual and practical linkages. 2022 – Режим доступу до ресурсу: <https://globalizationandhealth.biomedcentral.com/articles/10.1186/s12992-022-00840-6>.

19. How to Build a Doctor Appointment Scheduling Software: Key Features, Challenges to Consider, and Monetization Techniques [Електронний ресурс] // svitla. – 2023. – Режим доступу до ресурсу: <https://svitla.com/blog/doctor-appointment-software-development>.

20. Shantanu K. A Data Model for a Medical Appointment Booking App [Електронний ресурс] / Shantanu Kher // Vertabelo. – 2017. – Режим доступу до ресурсу: <https://vertabelo.com/blog/the-doctor-will-see-you-soon-a-data-model-for-a-medical-appointment-booking-app/>.

21. Optimizing Daily Service Scheduling for Medical Diagnostic Equipment Considering Patient Satisfaction and Hospital Revenue [Электронный ресурс] / D.Gang, L. Xinyue, H. Hui, O. Xiaoling // MDPI. – 2018. – Режим доступа до ресурсу: <https://www.mdpi.com/2071-1050/10/9/3349>.

ДОДАТКИ

1. Прототип сторінки «Логін».

Log In

Email

Password

Submit

2. Прототип сторінки «Головна».

Search

Events

Patients

Doctors

Add

Cabinet

Log out

SealMed

A

Upcoming event

No event

Upcoming appointments

Date	Time	Patient	Doctor	
10 June 2024	12:00	Olga Fierro	Tanya Kalulu	

3. Прототип сторінки «Лікарі».

Search

Events

Patients

Doctors

Add

Cabinet

Log out

SealMed

A

ID	First name	Last name	Post	Email	Phone	
6655886745dcd31baee39b84	Ronnie	Ferrary	Therapist	ronfe@gmail.com	2357436754	+
665e13259448be4d9a7de3fe	Emilio	Varga	Doctor	email@gmail.com	2346	+
665f38ed512802127a23bc0f	Tanya	Kalulu	Likar	tanlik@gmail.com	135645376	+

4. Прототип сторінки «Пацієнти».

☰ Search Events Patients Doctors Add Cabinet Log out	SealMed					A
	ID	First name	Last name	Email	Phone	
	6655e81330438c40dfcea400	Olga	Fierro	olf@gmail.com	2362236754	+
	665e182a1e5a6ffaba14b506	Ostap	Glushka	234f@gmail.com	146143657	+

5. Прототип форми додання нової події.

SealMed

Search

Events

Patients

Doctors

Add

Cabinet

Log out

Title

Description

Time

Select a time

Date

Submit

6. Прототип форми додання нового лікарського прийому.

SealMed

Search

Events

Patients

Doctors

Add

Cabinet

Log out

Patient

Select a patient

Olga Fierro

Ostap Glushka

Date

Time

Select a time

Submit

7. Зразок вихідної інформації у вигляді звіту лікарського прийому

Appointment info



Appointment ID: 26u2055423g4

Patient: Greta Rumberg

Doctor: Alberto Franco

Date: 17.04.2024

Time: 2:00 pm

Status: Carried out

The cost of appointment: \$ 300

Diagnosis

Diagnosis

Conclusion:

Doctor's conclusion.

8. Структура документу колекції «appointments».

```
_id: ObjectId('665ec16cd503a2d786dc2993')
▼ patient : Object
  id : "6655e81330438c40dfcea400"
  name : "Olga Fierro"
▼ doctor : Object
  id : "6655886745dcd31baee39b84"
  name : "Ronnie Ferrary"
date : 2024-06-05T21:00:00.000+00:00
time : "12:00"
status : true
__v : 0
```

9. Структура документу колекції «diagnosis».

```
_id: ObjectId('665db50374c5bb45f8f10142')
title : "test diagnosis"
description : "description for test diagnosis"
__v : 0
```

10. Структура документу колекції «doctors».

```
_id: ObjectId('6655847f0568f7b81d1a5fe4')
firstName : "Alberto"
lastName : "Vespucci"
birthDate : 1980-03-10T11:03:31.000+00:00
post : "Administrator"
role : "admin"
gender : "male"
email : "albvespucci@gmail.com"
phone : "4422115763"
address : "14 Julien Street"
experience : 15
salary : 2000
description : ""
password : "$2b$07$rMVC/1jKNlfXLfuT.I9BguJJv6/w6IsVPLrCBtdp23EKr8SCBuaCq"
```

11. Структура документу колекції «events».

```
_id: ObjectId('6659ec182f17d0e650b8bf5c')
title : "Test event"
description : "Test event description"
date : 2024-06-15T11:03:31.000+00:00
time : "12:00"
```

12. Структура документу колекції «treatments».

```

_id: ObjectId('665dbfb374c5bb45f8f1014b')
title: "treatment for test"
description: "description treatment"
diagnosis: Array (1)
  ▾ 0: Object
      id: "665db50374c5bb45f8f10142"
      title: "test diagnosis"
duration: 1
__v: 0

```

13. Структура документу колекції «patients».

```

_id: ObjectId('6655e81330438c40dfcea400')
firstName: "Olga"
lastName: "Fierro"
birthDate: 1988-03-10T11:03:31.000+00:00
gender: "female"
email: "olf@gmail.com"
phone: "2362236754"
address: "some address"
description: "New test patient"
doctors: Array (2)
appointments: Array (3)
diagnosis: Array (empty)
treatments: Array (empty)
__v: 0

```

14. Код налаштування маршрутизації клієнту

```

import { Routes } from '@angular/router';
import { AuthLayoutComponent } from './layouts/auth-layout/auth-layout.component';
import { HomeComponent } from './pages/home-page/home-page.component';
import { AuthGuard } from './core/guards/auth.guard';
import { ContentLayoutComponent } from './layouts/content-layout/content-layout.component';
import { AddAppointmentPageComponent } from './pages/add-appointment-page/add-appointment-page.component';
import { AddDiagnosisPageComponent } from './pages/add-diagnosis-page/add-diagnosis-page.component';
import { AddDoctorPageComponent } from './pages/add-doctor-page/add-doctor-page.component';
import { AddEventPageComponent } from './pages/add-event-page/add-event-page.component';
import { AddPatientPageComponent } from './pages/add-patient-page/add-patient-page.component';
import { AddReportPageComponent } from './pages/add-report-page/add-report-page.component';
import { AddTreatmentPageComponent } from './pages/add-treatment-page/add-treatment-page.component';
import { DoctorsPageComponent } from './pages/doctors-page/doctors-page.component';
import { PatientsPageComponent } from './pages/patients-page/patients-page.component';
import { DoctorPageComponent } from './pages/doctor-page/doctor-page.component';

```

```
export const routes: Routes = [
  {
    path: "",
    component: ContentLayoutComponent,
    canActivate: [AuthGuard],
    children: [
      {path: "",
        component: HomePageComponent,
      },
      {path: 'doctors',
        component: DoctorsPageComponent,
      },
      {
        path: 'doctors/:id',
        component: DoctorPageComponent
      },
      {
        path: 'patients',
        component: PatientsPageComponent,
      },
      {
        path: 'add',
        children: [
          {
            path: 'appointment',
            component: AddAppointmentPageComponent,
          },
          {
            path: 'diagnosis',
            component: AddDiagnosisPageComponent,
          },
          {
            path: 'doctor',
            component: AddDoctorPageComponent,
          },
          {
            path: 'event',
            component: AddEventPageComponent,
          },
          {
            path: 'patient',
            component: AddPatientPageComponent,
          },
          {path: 'report',
```

```

        component: AddReportPageComponent,
      },
      {path: 'treatment',
        component: AddTreatmentPageComponent,
      },
    ],
  },
],
},
{path: 'signin',
  component: AuthLayoutComponent,
},
];

```

15. Модуль сторінки додання нового лікарського прийому

```

import { Component, OnDestroy, OnInit } from '@angular/core';
import {
  FormBuilder,
  FormGroup,
  FormsModule,
  ReactiveFormsModule,
  Validators,
} from '@angular/forms';
import { Subject, takeUntil } from 'rxjs';
import { Router } from '@angular/router';
import { MessageService } from 'primeng/api';
import { DoctorService } from '../core/services/doctor.service';
import { Patient } from '../core/interfaces/patient.interface';
import { Doctor } from '../core/interfaces/doctor.interface';
import { PatientService } from '../core/services/patient.service';
import { AppointmentService } from '../core/services/appointment.service';
import { ButtonModule } from 'primeng/button';
import { CalendarModule } from 'primeng/calendar';
import { CardModule } from 'primeng/card';
import { DropdownModule } from 'primeng/dropdown';
import { InputNumberModule } from 'primeng/inputnumber';
import { InputTextModule } from 'primeng/inputtext';
import { InputTextareaModule } from 'primeng/inputtextarea';
import { TimeEnum } from '../shared/time.enum';
@Component({
  selector: 'app-add-appointment-page',
  standalone: true,
  imports: [
    ButtonModule,
    CalendarModule,
    CardModule,

```

```

    DropdownModule,
    FormsModule,
    InputNumberModule,
    InputTextModule,
    InputTextareaModule,
    ReactiveFormsModule,
  ],
  templateUrl: './add-appointment-page.component.html',
  styleUrls: ['./add-appointment-page.component.scss'],
})

export class AddAppointmentPageComponent implements OnInit, OnDestroy {
  appointmentForm!: FormGroup;
  timeOptions: string[] = [];
  patients: Patient[] = [];
  selectPatients: string[] = [];
  doctors: Doctor[] = [];
  selectDoctors: string[] = [];
  private destroy$ = new Subject<void>();

  constructor(
    private readonly formBuilder: FormBuilder,
    private readonly router: Router,
    private readonly messageService: MessageService,
    private readonly doctorService: DoctorService,
    private readonly patientService: PatientService,
    private readonly appointmentService: AppointmentService,
  ) {}

  ngOnInit(): void {
    this.appointmentForm = this.formBuilder.group({
      patient: ['', [Validators.required]],
      doctor: ['', [Validators.required]],
      date: ['', [Validators.required]],
      time: ['', [Validators.required]],
    });

    const enumToStringArray = (
      enumObject: Record<string, string>,
    ): string[] => {
      return Object.keys(enumObject).map((key) => enumObject[key]);
    };

    this.timeOptions = enumToStringArray(TimeEnum);

    this.patientService
      .getAllPatients()

```

```

.pipe(takeUntil(this.destroy$))
.subscribe({
  next: (patients) => {
    this.patients = patients;
    this.selectPatients = this.patients.map(
      (patient) => patient.firstName + ' ' + patient.lastName,
    );
  },
});

this.doctorService
.getDoctors()
.pipe(takeUntil(this.destroy$))
.subscribe({
  next: (doctors) => {
    this.doctors = doctors;
    this.selectDoctors = this.doctors.map(
      (doctor) => doctor.firstName + ' ' + doctor.lastName,
    );
  },
});
}

ngOnDestroy() {
  this.destroy$.next();
  this.destroy$.complete();
}

onSubmit(): void {
  if (this.appointmentForm.valid) {
    const { patient, doctor, date, time } = this.appointmentForm.value;

    const docName = doctor.split(' ');
    const selectedDoctor = this.doctors.find(
      (doc) => doc.firstName === docName[0] && doc.lastName === docName[1],
    );

    const patName = patient.split(' ');
    const selectedPatient = this.patients.find(
      (pat) => pat.firstName === patName[0] && pat.lastName === patName[1],
    );

    const appointmentDto = {
      patient: {
        id: selectedPatient!._id,

```

```

    name: patient,
  },
  doctor: {
    id: selectedDoctor!._id,
    name: doctor,
  },
  date: date.toISOString(),
  time,
  status: true,
};

```

```

this.appointmentService
.createAppointment(appointmentDto)
.pipe(takeUntil(this.destroy$))
.subscribe({
  next: () => {
    this.router.navigateByUrl('/');
    this.messageService.add({
      severity: 'success',
      summary: 'Success',
      detail: 'Success!',
    });
  },
  error: (error) => {
    this.messageService.add({
      severity: 'error',
      summary: 'Error',
      detail: error.error.message,
    });
  },
});
}
}
}

```

16. Сервіс «appointments» на стороні серверу.

```

import { BadRequestException, Injectable } from "@nestjs/common";
import { InjectModel } from "@nestjs/mongoose";
import { Appointment } from "../schemas/appointment.schema";
import mongoose, { Model } from "mongoose";
import { CreateAppointmentDto } from "../dto/create-appointment.dto";
import { AppointmentResponse, NewAppointmentResponse } from "../appointment.interfaces";
import { CreateReportDto } from "../dto/create-report.dto";
import { DoctorService } from "../doctor/doctor.service";
import { PatientService } from "../patient/patient.service";

```

```

@Injectable()
export class AppointmentService {
  constructor(
    @InjectModel(Appointment.name) private appointmentModel: Model<Appointment>,
    private doctorService: DoctorService,
    private patientService: PatientService
  ) {}

  async createAppointment(createAppointmentDto: CreateAppointmentDto):
  Promise<NewAppointmentResponse> {
    const newAppointment = new this.appointmentModel(createAppointmentDto);
    const savedAppointment = await newAppointment.save();

    if (savedAppointment) {
      await this.handleDoctorPatientRelationship(createAppointmentDto, savedAppointment._id.toString());
    }

    return { result: !!savedAppointment };
  }

  async createReport(appointmentId: string, createReportDto: CreateReportDto):
  Promise<NewAppointmentResponse> {
    const updatedAppointment = await this.appointmentModel.findByIdAndUpdate(
      appointmentId,
      createReportDto,
      { new: true }
    );

    if (!updatedAppointment) {
      throw new BadRequestException('Error with report formatting');
    }

    return { result: !!updatedAppointment };
  }

  async getAppointments(): Promise<AppointmentResponse[]> {
    return this.appointmentModel.find();
  }

  async getAppointmentById(appointmentId: string) {
    if (!mongoose.isValidObjectId(appointmentId)) {
      throw new BadRequestException('Invalid appointment ID format');
    }

    return this.appointmentModel.findById(appointmentId);
  }
}

```

```
}
```

```
async getAppointmentForReport(doctorId: string): Promise<AppointmentResponse[]> {  
  return this.appointmentModel.find({ 'doctor.id': doctorId, status: true });  
}
```

```
async getPatientAppointments (patientId: string) {  
  return this.appointmentModel.find({ 'patient.id': patientId });  
}
```

```
async getDoctorAppointments (doctorId: string) {  
  return this.appointmentModel.find({ 'doctor.id': doctorId });  
}
```

```
async getUpcomingAppointments() {  
  
  const currentDate = new Date();  
  const endDate = new Date();  
  endDate.setDate(currentDate.getDate() + 7);  
  return this.appointmentModel.find({  
    date: { $gte: currentDate, $lte: endDate }  
  }).sort({ appointmentDate: 1 });  
}
```

```
async getDoctorAppointmentsById(doctorId: string) {  
  return this.appointmentModel.find({ 'doctor.id': doctorId,  
  });  
}
```

```
async getUpcomingAppointmentsById(doctorId: string) {  
  if (!mongoose.isValidObjectId(doctorId)) {  
    throw new BadRequestException('Invalid doctor ID format');  
  }  
  const currentDate = new Date();  
  const endDate = new Date();  
  endDate.setDate(currentDate.getDate() + 7);  
  return this.appointmentModel.find({ _id: doctorId,  
    date: { $gte: currentDate, $lte: endDate }  
  }).sort({ appointmentDate: 1 });  
}
```

```
private async handleDoctorPatientRelationship(createAppointmentDto: CreateAppointmentDto,  
appointmentId: string) {  
  const { doctor, patient } = createAppointmentDto;  
  const doctorPatientExists = await this.doctorService.doesPatientExist(doctor.id, patient.id);
```

```
const patientDoctorExists = await this.patientService.doesDoctorExist(patient.id, doctor.id);
if (!doctorPatientExists) {
  await this.doctorService.addPatient(doctor.id, patient);
}
if (!patientDoctorExists) {
  await this.patientService.addDoctor(patient.id, doctor);
}

await this.doctorService.addAppointment(doctor.id, appointmentId);
await this.patientService.addAppointment(patient.id, appointmentId);
}
}
```

Имя пользователя: Інформаційних систем в економіці Шкуратовська Те...	ID проверки: 1016345317
Дата проверки: 11.06.2024 04:26:26 EEST	Тип проверки: Doc vs Internet + Library
Дата отчета: 11.06.2024 06:48:27 EEST	ID пользователя: 100005745

Название файла: Мельник_Д_ІН-404

Количество страниц: 55 Количество слов: 9084 Количество символов: 75752 Размер файла: 1.20 MB ID файла: 1016146806

9.51% Совпадения

Наибольшее совпадение: 2.12% с источником из Библиотеки (ID файла: 1016121992)

6.36% Источники из Интернета 244 Страница 57

7.63% Источники из Библиотеки 437 Страница 58

0% Цитат

Исключение цитат выключено

Исключение списка библиографических ссылок выключено

0% Исключений

Нет исключенных источников

Модификации

Обнаружены модификации текста. Подробная информация доступна в онлайн-отчете.

Замененные символы 1

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

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

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЄКТ

на тему

WEB-застосунок обліку діяльності приватного медичного закладу

здобувача Мельника Дмитра Ігоровича

Науковий керівник:

д.е.н., професор

Мозгаллі О.П.

Проект допущений до захисту
перед екзаменаційною комісією з
атестації здобувачів вищої освіти
завідувач кафедри:
к.е.н., доцент.

Тішков Б.О.

Київ 2024