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 ПРОЕКТ

на тему

ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ МЕДИЧНИМИ ДАНИМИ ЛІКАРНІ

здобувача Стеценка Артема Олеговича_____

Науковий керівник:
к.е.н., доцент
_____Помазун О.М.

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 проектної групи(гарант)
освітньо-професійної програми
_____Іванченко Г.Ф.
"____" _____2024 р.

ЗАТВЕРДЖУЮ:
Завідувач кафедри
_____Тішков Б.О.
"____" _____2024 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти ***Стеценка Артема Олеговича***

очної (денної) форми навчання

на підготовку кваліфікаційного бакалаврського проекту
на тему: *«Проєктування інформаційної системи управління медичними даними лікарні»*

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

Кваліфікаційний бакалаврський проект виконується на матеріалах
інтернет-ресурсів, технічної документації

План кваліфікаційного бакалаврського проекту
Розділ І Характеристика та аналіз предметної області

Розділ ІІ Розробка вимог і моделювання інформаційної системи
Розділ ІІІ Проєктування та реалізація прототипу, та модулів системи

Об'єкт дослідження інформаційна система, яка забезпечує управління медичними даними в межах лікарні
Предмет дослідження процеси та методи проєктування інформаційної системи управління медичними даними лікарні
Мета кваліфікаційного бакалаврського проекту розробка ефективної та надійної інформаційної системи, яка забезпечить збирання, зберігання, обробку та передачу медичної інформації.

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

У розділі I Дати характеристику предметної області управління медичними даними в лікарнях, проаналізувати цю предметну область та дослідити існуючі системи управління медичними даними.

У розділі II Розробка вимог і моделювання інформаційної системи управління медичними даними лікарні. Аналіз і специфікація вимог до цієї системи, постановка задачі та алгоритм її розв'язання, а також моделювання інформаційної системи

_____

_____

_____

_____

_____

У розділі III Проєктування та реалізація компонентів інформаційної системи управління медичними даними лікарні. Інформаційне забезпечення, технічне забезпечення, програмне забезпечення, результати реалізації інформаційної
системи.

_____

_____

_____

_____

Завдання підготував

науковий керівник _____ Помазун Оксана Миколаївна

"_____"_____ 2024 р.

Завдання одержав

студент _____ Стеценко Артем Олегович

"_____"_____ 2024 р.

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

Стеценка Атема Олеговича
на тему
«Проєктування інформаційної системи управління медичними даними лікарні»

1. Актуальність теми: «Проєктування інформаційної системи управління медичними даними лікарні» є актуальною темою через зростаючою потребою в удосконаленні систем управління медичною інформацією. У сучасних лікарнях, обсяг медичних даних зростає, тому необхідно забезпечити їхнє ефективне зберігання, швидкий доступ та обробку. Впровадження інформаційної системи допомагає автоматизувати процеси, знижує ризик помилок, поліпшує взаємодію між медичним персоналом та пацієнтами.
2. Позитивні риси кваліфікаційного бакалаврського проекту: проект проявляє високий рівень компетентності автора, який виявляється у глибокому аналізі предметної області та обґрунтуванні вибору методів розробки інформаційної системи. В проекті простежується логічна послідовність етапів розробки системи, а також наявні технічні та функціональні аспекти, що дозволяє зрозуміти важливість та практичну застосовність запропонованих рішень.
3. Наявність самостійних розробок автора автором було представлено інтерфейс системи який при мінімальній кількості тексту дозволяє зрозуміти який функціонал надає та чи інша частина системи опираючись на піктограми.
4. Цінність теоретичних висновків та практичних рекомендацій:важливість теоретичних висновків автора полягає в їхньому потенціалі вплинути на розвиток забезпечення управління медичними даними в лікарнях.
5. Наявність недоліків: Можливість провадження більш сучасного функціоналу та реалізовано не весь функціонал системи.
6. Загальна оцінка кваліфікаційного бакалаврського проекту та його допущення до захисту перед ЕК:_____

Науковий керівник

_____ доцент, к.е.н. Помазун О.М.
(підпис)

“___”_____ 20__ р.

Рецензія

на кваліфікаційний бакалаврський проект

Стеценка Артема Олеговича

тема

«Проєктування інформаційної системи управління медичними даними лікарні»

Актуальність теми кваліфікаційного бакалаврського проекту і доцільність його розроблення «Проєктування інформаційної системи управління медичними даними лікарні» є актуальною темою через зростаючою потребою в удосконаленні систем управління медичною інформацією.

Якість проведеного дослідження дослідження, яке проведено в рамках цього проекту, відзначається високою якістю і деталізацією. Автор проявив глибоке розуміння предметної області управління медичними даними в лікарнях і здійснили обґрунтований аналіз існуючих проблем та вимог до інформаційної системи.

Позитивні риси кваліфікаційного бакалаврського проекту автор демонструє інноваційний підхід до вирішення актуальних проблем у сфері медичної інформатики. Він враховує сучасні вимоги до систем управління медичними даними та включає в себе детальний аналіз технічних, функціональних та безпекових аспектів розробки інформаційної системи.

Зауваження автор виконав проектування системи зі сторони користувача, але система також має адміністративну частину.

Практична значимість висновків і рекомендацій Висновки та рекомендації, представлені у роботі, мають велику практичну значимість для медичних установ. Рекомендації, наведені у роботі, можуть бути використані для подальшої розробки та впровадження інформаційних систем у медичних закладах з метою покращення їх ефективності та продуктивності. Робота заслуговує на оцінку «Відмінно»

Місце роботи та посада рецензента Український державний університет імені Михайла Драгоманова, доцент кафедри інформаційних технологій і програмування факультету математики, інформатики та фізики

к.пед.н. доцент _____ Умрик М.А.

АНОТАЦІЯ

кваліфікаційного бакалаврського проекту студента 4 курсу
Навчально-наукового інституту «Інститут інформаційних технологій в економіці»
Стеценка Артема Олеговича, виконаної на тему:
«Проєктування інформаційної системи управління медичними даними лікарні»
Київ: кафедра інформаційних систем в економіці, 2024р.

Кваліфікаційний бакалаврський проект має на меті вирішення актуальної проблеми визначення потреб в управління медичними даними лікарні.. Ця проблема розглядається з використанням інформаційних технологій, і проект спрямований на її розв'язання та подальше удосконалення.

Кваліфікаційний бакалаврський проект складається з трьох розділів, які пов'язаних між собою.

В першому розділі дана характеристику предметної області управління медичними даними в лікарнях, проаналізувати цю предметну область та дослідити існуючі системи управління медичними даними.

Другий розділ є проектним і присвячений розробці вимог і моделювання інформаційної системи управління медичними даними лікарні. Аналіз і специфікація вимог до цієї системи, постановка задачі та алгоритм її розв'язання, а також моделювання інформаційної системи.

Третій розділ проєктування та реалізація компонентів інформаційної системи управління медичними даними лікарні. Інформаційне забезпечення, технічне забезпечення, програмне забезпечення, результати реалізації інформаційної системи.

Висновки містять рекомендації для покращення управління медичними даними в лікарнях, які спростять процеси збору, обробки та аналізу медичної інформації.

РЕФЕРАТ

Кваліфікаційний бакалаврський проект містить 98 сторінок, 13 таблиць, 64 рисунків, список літератури з 35 джерел посилань, 5 додатки.

«ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ МЕДИЧНИМИ ДАНИМИ ЛІКАРНІ»

Перелік ключових слів: інформаційна система, проєктування, прототип, діаграма, медичні дані.

Предметом дослідження є процес проєктування інформаційної системи управління медичними даними лікарні.

Об'єктом дослідження є структура та функціональні можливості інформаційної системи, призначеної для збору, зберігання, обробки та аналізу медичної інформації в лікарнях.

Мета кваліфікаційного бакалаврського проекту є розробка ефективної та надійної інформаційної системи яка спростить процес управління медичними даними та покращить надання медичних послуг в лікарнях

Завданням кваліфікаційного бакалаврського проекту є аналіз потреб та вимог користувачів, розробка та імплементація відповідної інформаційної системи, тестування її функціональності

Результатом досягнутим в процесі роботи є створення інформаційної системи, яка відповідає вимогам та потребам медичних закладів і може ефективно управляти медичними даними.

Одержані результати можуть бути використані для покращення ефективності та точності управління медичними даними в лікарнях, забезпечення більш високого рівня якості надання медичних послуг, а також для подальшого вдосконалення та розвитку інформаційних систем у сфері охорони здоров'я.

Рік виконання кваліфікаційного бакалаврського проекту – 2024.

Рік захисту кваліфікаційного бакалаврського проекту – 2024

ЗМІСТ

ВСТУП

Проєктування інформаційної системи управління медичними даними лікарні є актуальною темою у контексті сучасної охорони здоров’я.

Цей проект спрямований на впровадження ефективної системи, яка забезпечить збір, зберігання, обробку та доступ до медичної інформації, сприяючи підвищенню якості надання медичних послуг та оптимізації роботи медичного персоналу.

Мета полягає в створенні централізованого ресурсу, який дозволить легко відстежувати пацієнтську інформацію, медичні історії та лікування. Це сприятиме ефективному прийняттю рішень, підвищенню безпеки та покращенню обслуговування пацієнтів.

Основною метою проєктування такої системи є створення централізованої платформи, яка об'єднує всі необхідні дані для надання високоякісної медичної допомоги. Вона повинна забезпечувати користувачам швидкий доступ до медичних записів, діагностичних результатів інших важливих документів,

Об’єктом дослідження є інформаційна система управління медичними даними для лікарні. Це включає всі аспекти, пов'язані з організацією, зберіганням, обробкою та використанням медичних даних, які забезпечують надання медичних послуг.

Предмет дослідження є інформаційна система управління медичними даними для лікарні. При створенні системи управління медичними даними для збору та аналізу медичної інформації в лікарні, необхідно враховувати існуючі підходи та системи управління даними та розробляти нові рішення, які виправлять їх недоліки та покращать їх ефективність.

Було використано інструменти програмних засобів «Opcat», «DrawIo» та «Enterprise Architect» для проектування. В програмному середовищі Figma було створено прототип системи, для реалізації було використано мову програмування Python з фреймворком Django, базою даних MySQL.

РОЗДІЛ 1 ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Характеристика предметної галузі та об’єкта дослідження

Управління даними охорони здоров’я - процес зберігання, захисту та аналізу даних пацієнтів, отриманих із різноманітних джерел. З точки зору управління, медичні заклади в основному використовують такі технології, як електронні медичні записи (EMC) і системи CRM. EMC дозволяє лікарям в електронному вигляді записувати та зберігати інформацію про пацієнтів: вашу історію хвороби, діагнози, аналізи тощо. Це дає вам загальну картину того, ким ви є як пацієнт у будь-який момент. У свою чергу, окрім зручного зберігання даних, ви маєте можливість використовувати автоматизовані функції, такі як запис на прийом до лікаря, замовлення ліків тощо [1].

Технологія CRM допомагає відстежувати історію вашої взаємодії з практикою, збираючи та аналізуючи дані з різних джерел, таких як контакт-центри, соціальні мережі, mHealth (мобільна медицина) тощо. Усе це допомагає організаціям охорони здоров’я отримати 360-градусний огляд своїх пацієнтів. Він охоплює не лише інформацію про ваше здоров’я, а й ваші звички та вподобання. Разом ці технології створюють цілісне уніфіковане уявлення про пацієнта на одній консолі. Українські лікарні активно зайнялися медичними інформаційними системами (MIC). Ви отримуєте електронну медичну картку пацієнта, а лікар видає електронні інструкції та рецепти. Далі ви можете заявити сімейному лікарю онлайн і мати можливість спілкуватися з лікарем у будь-який час і в будь-якому місці за допомогою зручного мобільного додатку [2].

В наш час лікарні в Україні активно приєднуються до медичних інформаційних систем. З цими системами ви отримуєте електронну медичну картку, лікар може виписати електронні напрямки та рецепти. Після цього ви можете укласти декларації з сімейним лікарем в онлайн-режимі та користуватися зручними програмами для взаємодії з лікарем в будь-який час та в будь-якому місці.

Інформаційна система управління медичними даними в лікарні може виконувати різноманітні функції, спрямовані на забезпечення ефективного та безпечного управління медичною інформацією. Основні можливості такої системи включають:

- Зберігання медичних записів;
- управління реєстрацією пацієнтів;
- запис прийому пацієнта;
- управління ліками та рецептами.

Розглянемо кожен пункт більш детально:

Зберігання медичних записів: система дозволяє зберігати, організовувати та забезпечувати доступ до медичних даних пацієнтів.

Управління реєстрацією пацієнтів: система може служити для реєстрації нових пацієнтів та ведення бази даних про пацієнтів.

Запис прийому пацієнта: система дозволяє вести електронний журнал записів про прийом пацієнтів, включаючи дати, часи та обраного лікаря.

Управління ліками та рецептами: система може включати функції для виписування електронних рецептів, відстеження використання ліків та управління запасами медикаментів у лікарні.

Отже, створення такої інформаційної системи сприяє покращенню ефективності, точності та доступності медичної інформації, а також полегшують процеси управління лікарнею.

Методологія OPM, або Object Process Methodology, використовує системний підхід для аналізу, проектування та моделювання систем. Основна її ідея полягає у тому, що будь-яку систему можна розглядати як сукупність об'єктів, які взаємодіють та пройшли через різні процеси [3].

На системній діаграмі верхнього рівня показано головний процес інформаційної системи (рис.1.1). На цьому рівні визначені об’єкти та основні процеси. Тест скрипту діаграми знаходиться в додатку А.

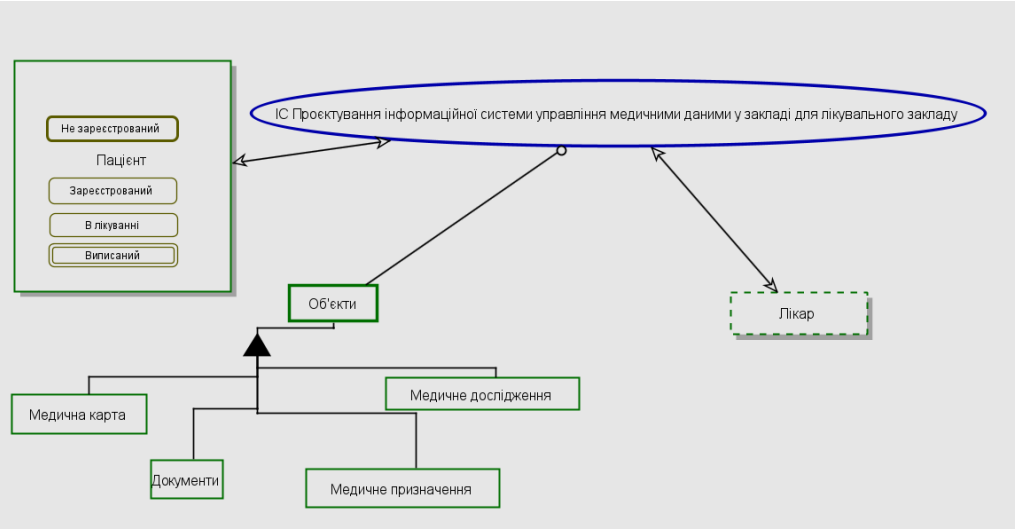


Рисунок 1.1 - Діаграма OPD верхнього рівня

Джерело: розроблено автором самостійно

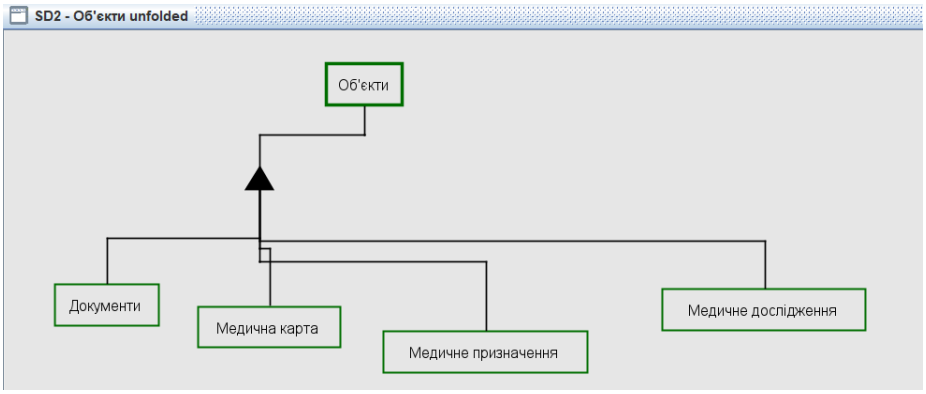


Рисунок 1.2 - механізм розгортання

Джерело: розроблено автором самостійно

На рис.1.3-1.8 відображаються діаграми нижчого рівня, що містять підпроцеси з асоційованими об’єктами та відповідними зв’язками.

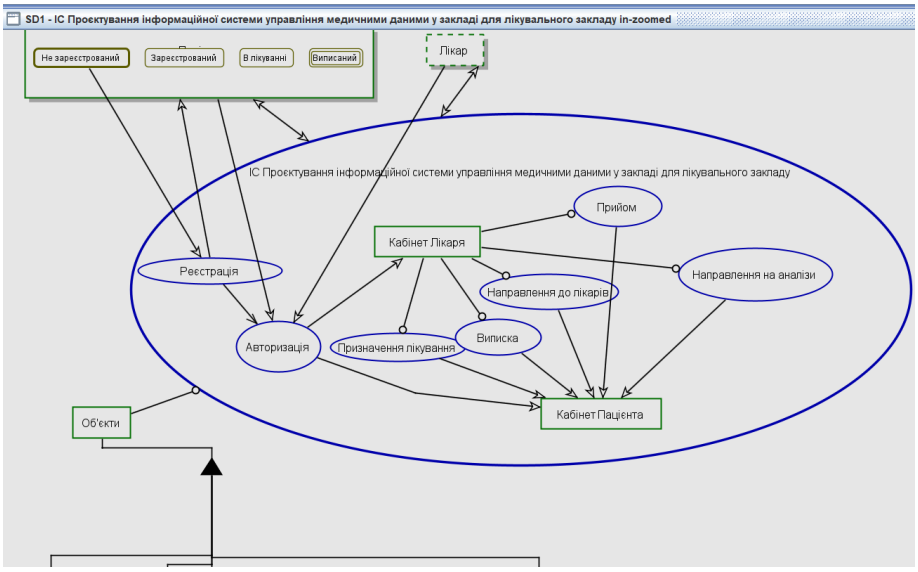


Рисунок 1.3 - Діаграма OPD нижнього рівня

Джерело: розроблено автором самостійно

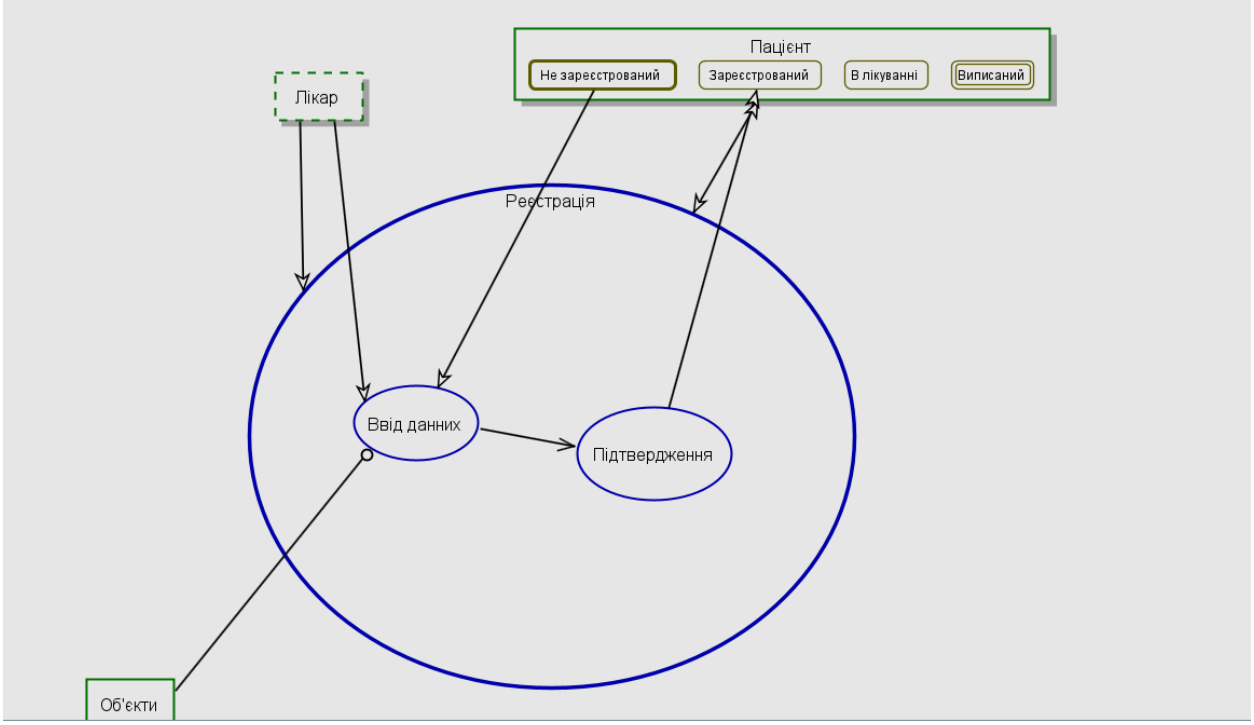


Рисунок 1.4 - Діаграма OPD нижнього рівня. Реєстрація

Джерело: розроблено автором самостійно

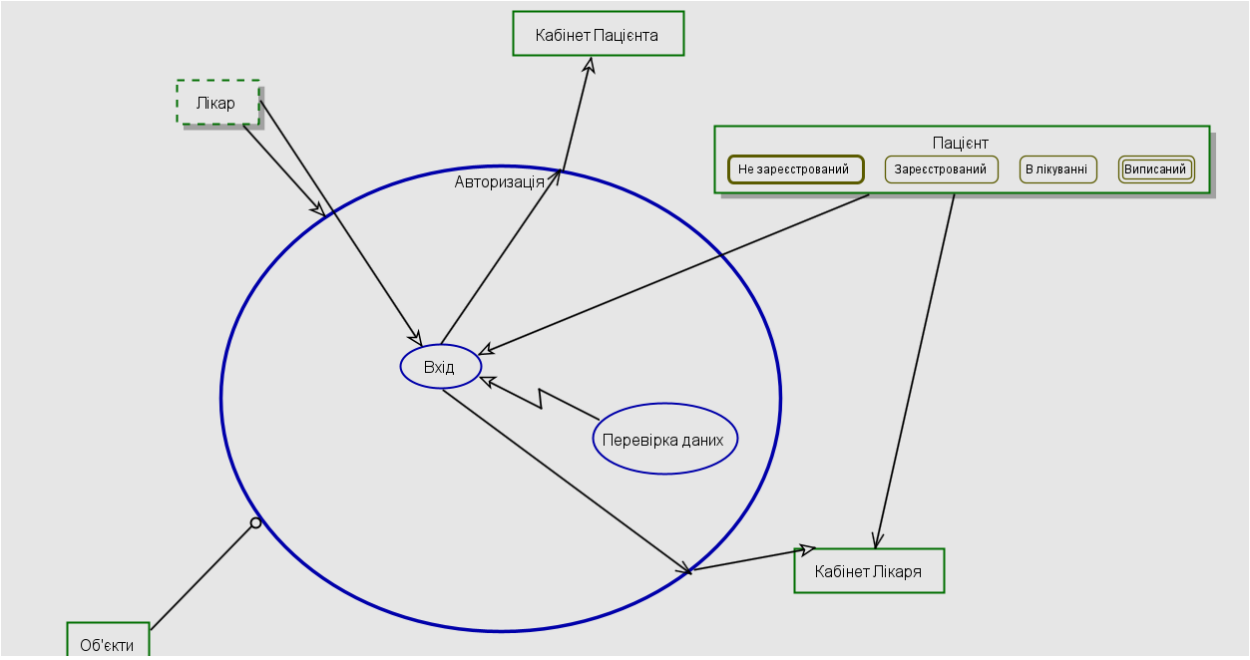


Рисунок 1.5 - Діаграма OPD нижнього рівня. Процес Авторизації

Джерело: розроблено автором самостійно

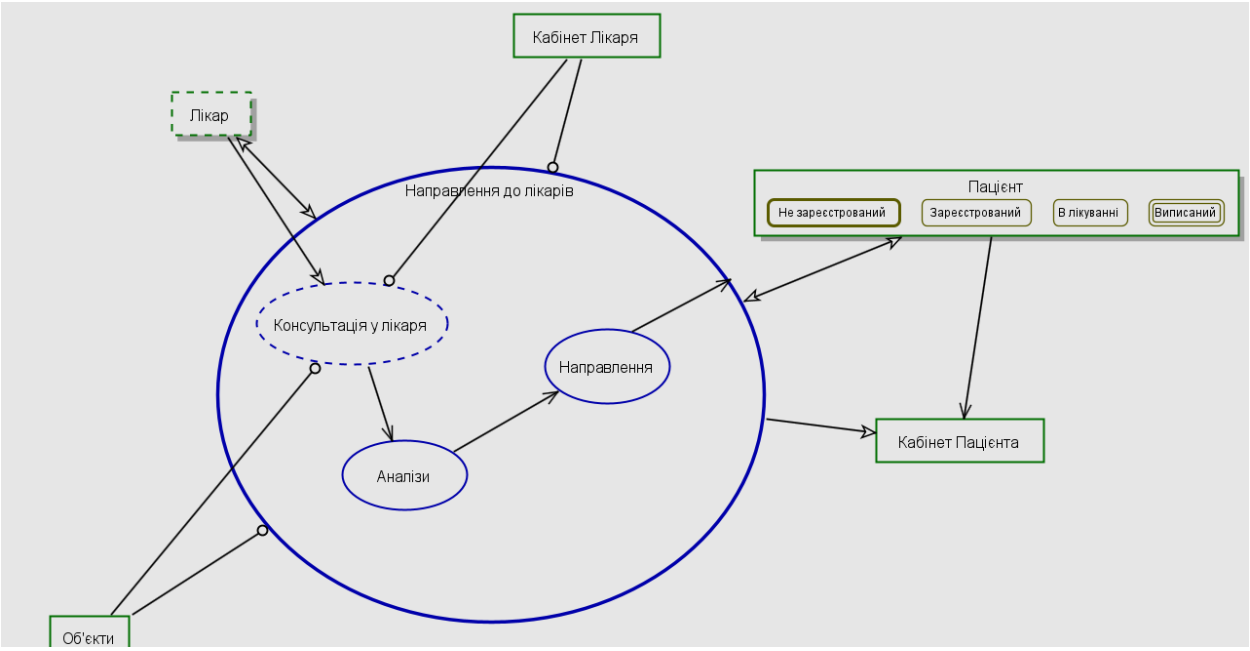


Рисунок 1.6 - Діаграма OPD нижнього рівня. Процес Направлення до лікарів

Джерело: розроблено автором самостійно

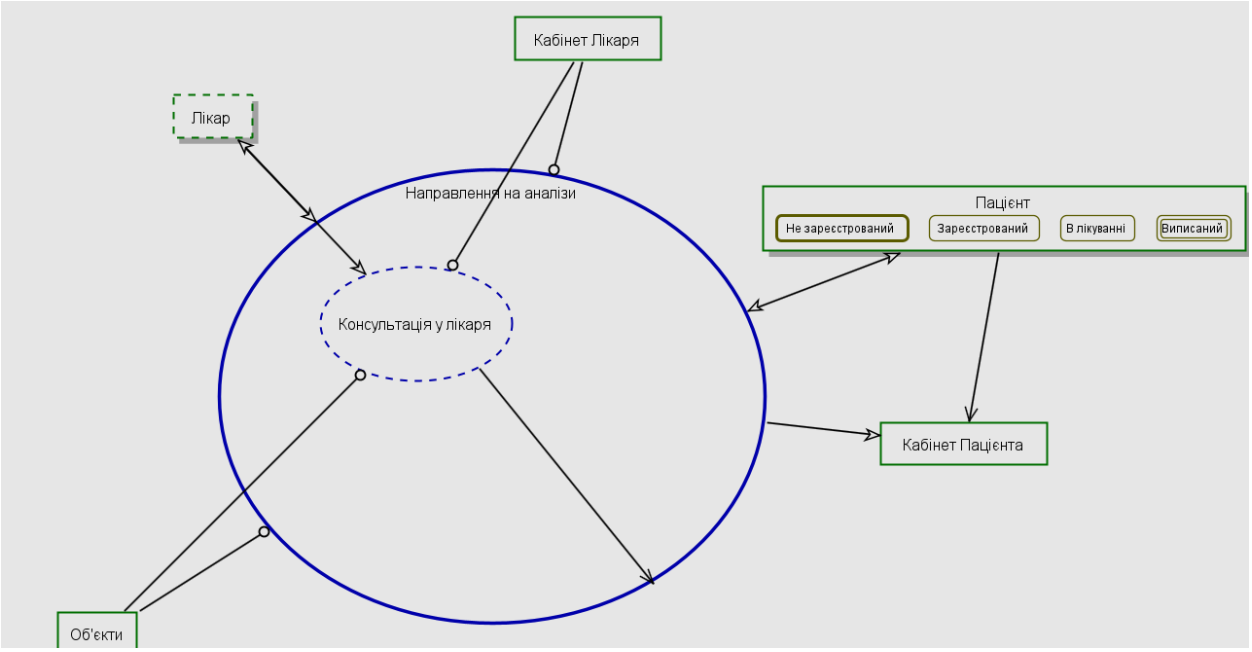


Рисунок 1.7 - Діаграма OPD нижнього рівня. Направлення на аналізи

Джерело: розроблено автором самостійно

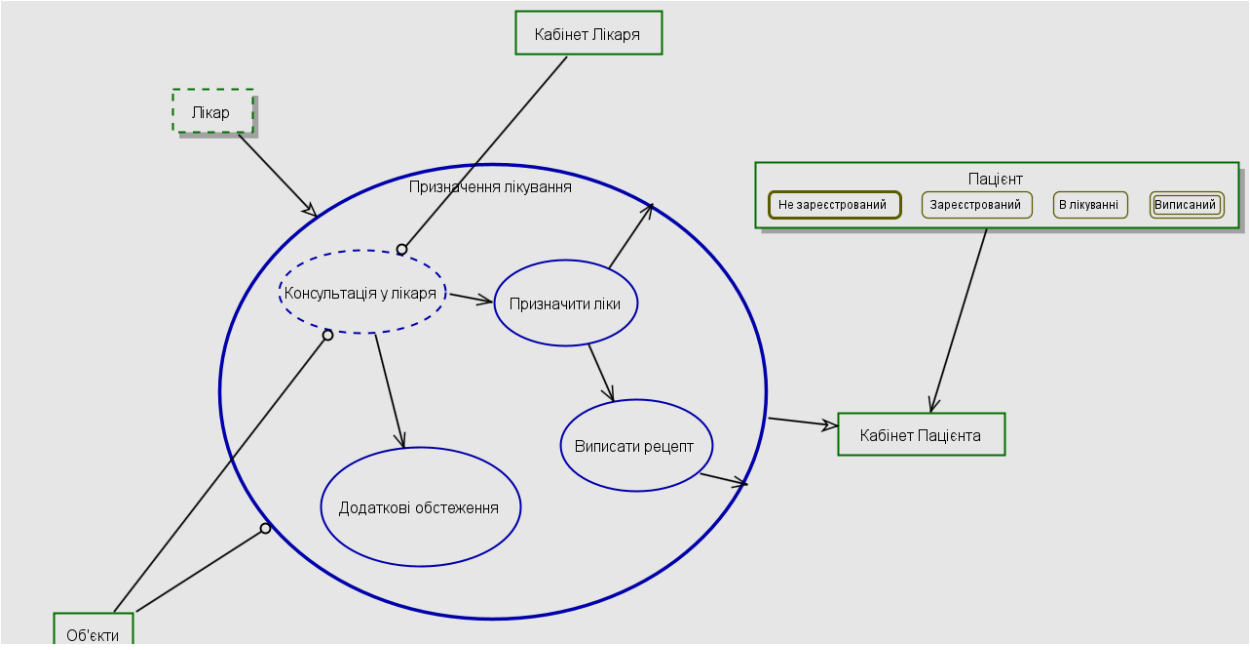


Рисунок 1.8 - Діаграма OPD нижнього рівня. Призначення лікування

Джерело: розроблено автором самостійно

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

Сьогодні існує багато інформаційних систем, які використовуються для управління медичними даними, кожна з яких має свої переваги та недоліки. Розглянемо деякі з них:

Salesforce Health Cloud - обліково-керівна система для організацій охорони здоров'я. Вона надає інструменти для ефективного ведення медичних записів та взаємодії з пацієнтами [4].

Перевага:

- Високий ступінь налаштування та гнучкості;
- інтеграція зі сторонніми додатками та медичними пристроями;
- здатність проводити детальний аналіз даних для покращення догляду за пацієнтами.

Недолік:

- Впровадження та підтримка систем коштує дорого;
- налаштування є складним і вимагає експертного впровадження.

Healthcare HubSpot - універсальний бізнес-інструмент, розроблений, щоб покращити спосіб спілкування медичних компаній зі своїми клієнтами [5].

Переваги:

- Легка інтеграція з маркетинговими інструментами;
- безкоштовні або доступні плани для малого бізнесу;
- може використовуватися для залучення пацієнтів і маркетингу.

Недоліки:

- Можливі обмеження для великих медичних закладів.
- може бути менше функціоналу порівняно з іншими CRM;

Pipedrive - проста та гнучка система CRM, яка дозволяє зберігати записи та взаємодіяти з клієнтами. Він має інструменти для управління відносинами та транзакціями [6].

Переваги:

- Легкий у використанні та зрозумілий інтерфейс;
- гнучка конфігурація та можливості розширення;
- можливість використання для відносин з пацієнтами.

Недоліки:

- Може вимагати налаштувань для медичного сектору;
- може бути менше функціоналу для великих організацій.

Microsoft Dynamics 365 for Healthcare - інтегрована система для управління відносинами з пацієнтами та клієнтами [7].

Переваги:

- Інтеграція з іншими продуктами Microsoft;
- велика спільнота користувачів Microsoft;
- розширені можливості аналітики.

Недоліки:

- Високі витрати та складність реалізації;
- може вимагати інших продуктів Microsoft для повного функціоналу.

Таблиця 1.1 - Порівняння систем

Номер	Характеристика	Salesforce Health Cloud	HubSpot for Healthcare	Pipedrive	Microsoft Dynamics 365 for Healthcare
1	Електронні медичні картки (EMR)	Так	Можливо	Можливо	Так
2	Лікарські призначення та рецепти	Так	Можливо	Ні	Так

Номер	Характеристика	Salesforce Health Cloud	HubSpot for Healthcare	Pipedrive	Microsoft Dynamics 365 for Healthcare
3	Управління та планування прийому	Так	Так	Можливо	Так
4	Керування станом пацієнтів	Так	Можливо	Можливо	Так

Джерело: розроблено автором самостійно

У першому розділі бакалаврського проєкту було досліджено предметну область. Розглянуто процеси, які використовуються в інформаційній системі за допомогою методології OPM, зображено діаграми верхнього та нижнього рівня з детальним описом кожної з них. Крім того, проведено порівняння існуючих систем.

РОЗДІЛ 2 РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ ПІДСИСТЕМИ

2.1 Аналіз і специфікація вимог до інформаційної підсистеми

SysML - мова моделювання систем, яка розширює можливості мови UML (Unified Modeling Language) для більш ефективного моделювання складних систем. Він надає спеціалізовані конструкції та діаграми для опису структури, поведінки, вимог та параметричних обмежень системи [8].

Бізнес-вимоги - умови, функціональність або особливості, які бізнес-система повинна мати. Вони є ключовим елементом успішного впровадження будь-якої системи, оскільки вони визначають, що саме повинна здійснювати система для досягнення поставлених цілей бізнесу [9].

Бізнес-вимоги:

- Доступу до своїх медичних даних онлайн;
- зручності та простоти використання системи для запису на прийоми та перегляду результатів аналізів;
- автоматизації процесів прийому пацієнтів;
- обробки медичної інформації та призначення лікування;
- підвищення ефективності роботи лікарні;
- покращення координації догляду за пацієнтами.

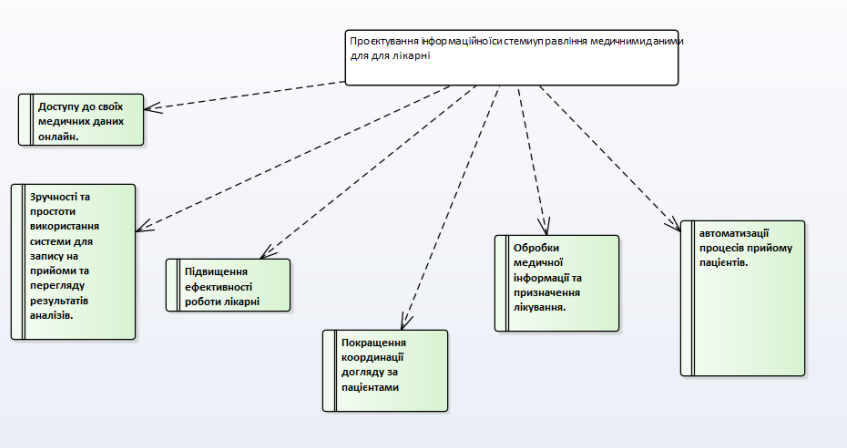


Рисунок 2.1 - Діаграма бізнес-вимог

Джерело: розроблено автором самостійно

2.1.2 Функціональні вимоги до системи

Функціональні вимоги до автоматизованої системи представлені на рисунку 2.2.

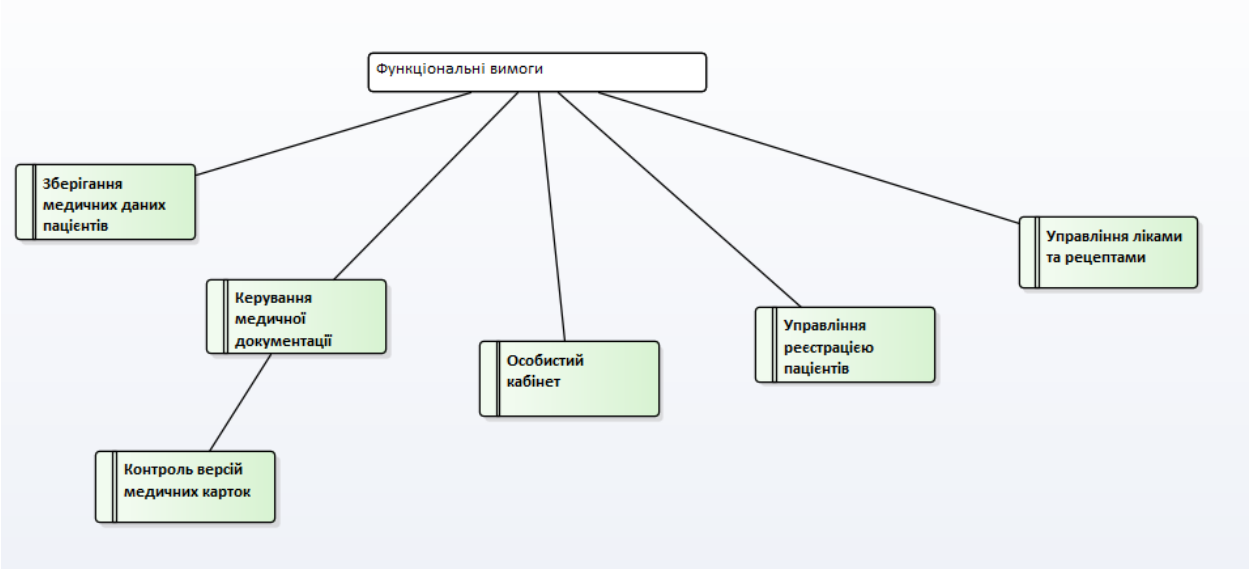


Рисунок 2.2 - Діаграма функціональних вимог

Джерело: розроблено автором самостійно

2.1.3 Нефункціональні вимоги

Нефункціональні вимоги до автоматизованої системи представлені на рисунку 2.3.

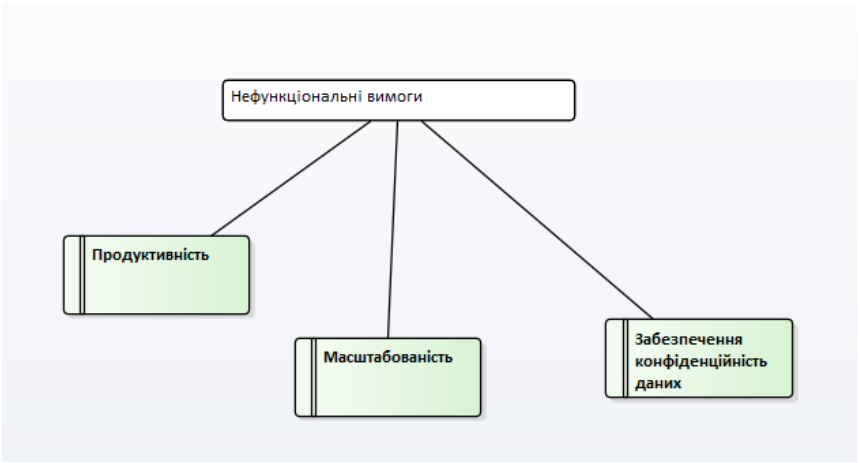


Рисунок 2.3 - Діаграма нефункціональні вимог

Джерело: розроблено автором самостійно

Джерелами вимог виступають пацієнти і лікарі.

На рисунку 2.4 представлено діаграму підсумкової кількості вимог

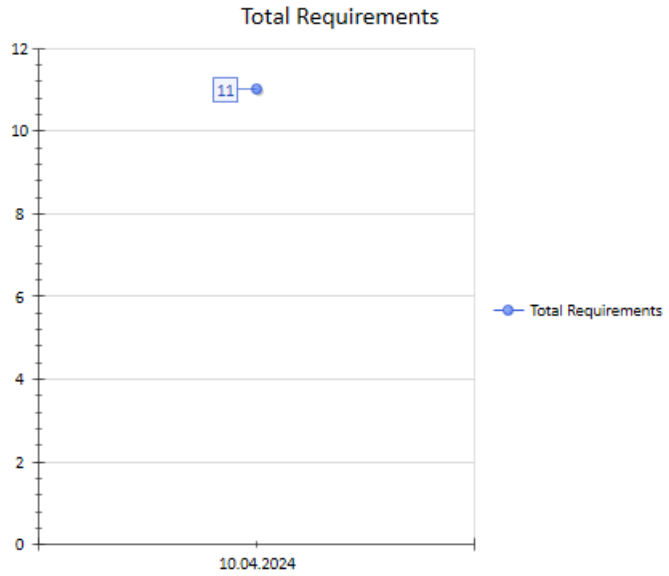


Рисунок 2.4 - Діаграма підсумкової кількості вимог

Джерело: розроблено автором самостійно

На рисунку 2.5 представлено діаграму розподілу всіх вимог за статусом

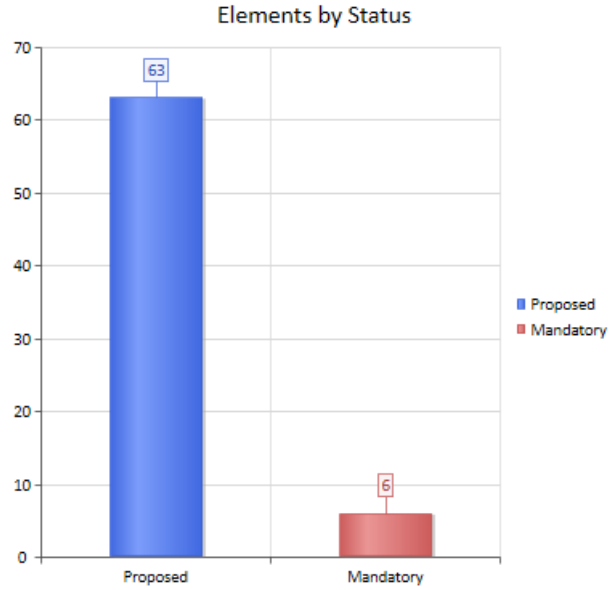


Рисунок 2.5 - Діаграма розподілу за статусом

Джерело: розроблено автором самостійно

На рисунку 2.6 представлено діаграму розподілу всіх вимог за пріоритетом.

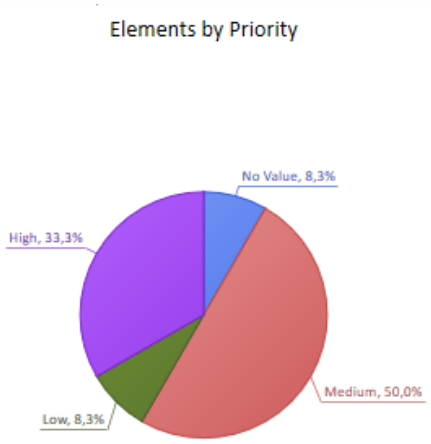


Рисунок 2.6 - діаграма розподілу вимог за пріоритетом

Джерело: розроблено автором самостійно

На рисунку 2.7 представлено діаграму розподілу всіх вимог за складністю.

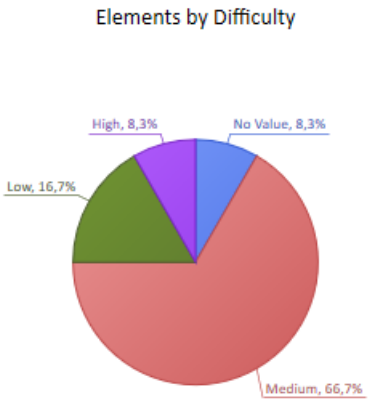


Рисунок 2.7 - діаграма розподілу вимог за складністю.

Джерело: розроблено автором самостійно

2.2 Постановка задачі та алгоритм розв'язання задачі

Призначення задачі полягає у створенні комплексної інформаційної системи, яка забезпечує збирання, зберігання, обробку та управління медичними даними в лікарні. Основна мета системи - покращення якості медичних послуг, оптимізація робочих процесів, підвищення ефективності

управління ресурсами та забезпечення своєчасного доступу до актуальної інформації для прийняття клінічних та адміністративних рішень, знизити операційні витрати, скоротити час обробки медичних даних, зменшити кількість помилок, оптимізувати використання ресурсів та покращити фінансовий контроль. У довгостроковій перспективі це призведе до підвищення ефективності роботи лікарні та покращення фінансових показників.

Проєктування програмного забезпечення, інтеграцію з існуючими системами, забезпечення захисту даних та надійності роботи системи, використання сучасних технологій для забезпечення масштабованості та гнучкості системи.

Періодичність розв’язання задачі та видачі інформації залежить від конкретного модуля:

- Реєстрація пацієнтів: в режимі реального часу;
- медичні записи: під час кожного прийому або медичної процедури;
- аналітичні звіти: за запитом, щомісячно, щоквартально.

Умови припинення автоматизованого розв’язання задачі Автоматизоване розв’язання задачі може бути припинене у випадках:

- Технічних збоїв або відмови обладнання;
- відсутності доступу до мережі або серверів;
- виявлення критичних помилок у програмному забезпеченні;
- зовнішніх загроз безпеці, які вимагають негайного втручання;

При розв’язанні задачі здійснюється керування наступними об’єктами: дані користувача, призначення ліків, виписування рецептів, виписування направлення до лікарів, виписування направлення на аналізи.

2.2.1. Вхідна інформація

Вхідні повідомлення інформаційної системи - дані, які подаються користувачем, адміністратором або зберігаються в базі даних.

Таблиця 2.1 - Перелік та опис вхідних повідомлень

№	Назва вхідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
1	Дані користувача	user_data	Екранна форма	При реєстрації чи зміні особистих даних	Користувач
2	Призначення ліків	medicine_patient	Файл/Паперовий документ/Екранна форма	Коли призначають ліки	Адміністратор
3	Виписування рецептів	w_recipes_patient	Файл/Паперовий документ/Екранна форма	Коли виписують рецепт	Адміністратор
4	Виписування направлення до лікарів	w_direction_doctor	Файл/Паперовий документ/Екранна форма	Коли виписують направлення до лікарів	Адміністратор
5	Виписування направлення на аналізи	w_analyses_patient	Файл/Паперовий документ/Екранна форма	Коли виписують направлення на аналізи	Адміністратор

Джерело: розроблено автором самостійно

Перелік і опис вихідних повідомлень

Для вихідної інформації використовується: дані пацієнтів, дані лікарів, призначені ліки, виписані рецепти, направлення до лікарів, направлення на аналізи.

2.2.2. Вихідна інформація

Таблиця 2.2. - Перелік та опис вихідних повідомлень

№	Назва вихідного повідомлення	Ідентифікатор	Форма подання	Періодичність видання	Користувачі інформації
1	Дані пацієнтів	profile_patient	Екранна форма	По запиту	Адміністратор
2	Дані лікарів	profile_doctor	Екранна форма	По запиту	Користувач
3	Призначені ліки	need_ medicine	Файл/Паперовий документ/Екранна форма	По запиту	Користувач
4	Виписані рецепти	need_recipes	Файл/Паперовий документ/Екранна форма	По запиту	Користувач
5	Направлення до лікарів	direction_doct or	Файл/Паперовий документ/Екранна форма	По запиту	Користувач
6	Направлення на аналізи	analyses_patie nt	Файл/Паперовий документ/Екранна форма	По запиту	Користувач

Джерело: розроблено автором самостійно

2.2.3. Використовувана та результатна інформація

Реєстрація пацієнтів (patient_registration): Містить дані про пацієнтів, які зареєстровані у медичному закладі, включаючи їх особисту інформацію та контактні дані.

Електронні медичні картки (electronic_medical_records): Зберігає всю медичну інформацію про пацієнтів, включаючи діагнози, історію лікування та результати аналізів.

Управління запасами медикаментів (medication_inventory): Містить дані про наявність, рух та запаси медикаментів у медичному закладі.

Лабораторні результати (lab_results): Містять результати лабораторних аналізів.

Графік прийому лікарів (doctor_schedules): Інформація про робочий графік лікарів, доступні дати та час для прийому пацієнтів.

Дані про призначення медикаментів (medication_prescriptions): Містять інформацію про призначення медикаментів.

Дані про історію візитів пацієнтів (patient_visit_history): Історія всіх візитів пацієнтів до лікарні, включаючи дати, причини візиту та результати обстежень.

Дані про рецепти на ліки (prescription_data): Записи про видані рецепти на медикаменти, включаючи дозування та інструкції для пацієнтів.

Дані про медичні консультації (medical_consultations): Записи про консультації з лікарями, включаючи дату, час та рекомендації лікарів.

Дані про виписки з лікарні (hospital_discharge_data): Дані про виписки пацієнтів з лікарні, включаючи дату виписки та рекомендації на післялікарняний період.

Таблиця 2.3. - Перелік масивів використовуваної інформації

Масив	Ідентифікатор	Максимальна кількість записів
Реєстрація пацієнтів	patient_registration	100,000
Електронні медичні картки	electronic_medical_records	1,000,000
Управління запасами медикаментів	medication_inventory	150,000
Масив	Ідентифікатор	Максимальна кількість записів

Лабораторні результати	lab_results	2,000,000
Графік прийому лікарів	doctor_schedules	50,000
Дані про призначення медикаментів	medication_prescriptions	1,000,000
Дані про історію візитів пацієнтів	patient_visit_history	1,000,000
Дані про рецепти на ліки	prescription_data	500,000
Дані про медичні консультації	medical_consultations	1,000,000
Дані про виписки з лікарні	hospital_discharge_data	700,000

Джерело: розроблено автором самостійно

Кожен масив має унікальний ідентифікатор і максимальну кількість записів, що допоможе оптимізувати процеси зберігання та обробки даних.

Таблиця 2.4. - Перелік масивів результатної інформації

Масив	Ідентифікатор	Максимальна кількість записів
Пацієнти	PAT	50,000
Лікарі	DOC	500
Медичні записи	MEDREC	1,000,000
Розклад прийомів	SCHED	100,000
Масив	Ідентифікатор	Максимальна кількість записів
Медичні процедури	MEDPROC	50,000
Лабораторні результати	LABRES	200,000

Виписки з лікарні	DISCH	20,000
Призначення ліків	PRES	200,000

Джерело: розроблено автором самостійно

2.2.4. Математичне забезпечення та алгоритм розв'язання задачі

Розрахувати відсоток від усіх виписаних ліків, скільки виписано з рецептом

$$\mathrm{Psr} = \left(\frac{\mathrm{Sr}}{\mathrm{Wvl}}\right) \times 100\%, \quad (2.1)$$

де P_{sr} - Відсоток виписаних ліків за рецептом,

Sr - Виписані ліки з рецептом,

Wvl - Всього виписаних ліків.

Розрахувати відсоток від усіх виписаних ліків, скільки виписані ліки без рецепта

$$\mathrm{Pbr} = \left(\frac{\mathrm{Br}}{\mathrm{Wvl}}\right) \times 100\%, \quad (2.2)$$

де Pbr - Відсоток виписаних ліків без рецепта,

Br - Виписані ліки без рецептом,

Wvl - Всього виписаних ліків.

Залежно від специфіки задачі, можуть бути використані інші параметри та формули.

Взаємозв'язок з іншими задачами можна прослідкувати на інформаційній моделі задачі.

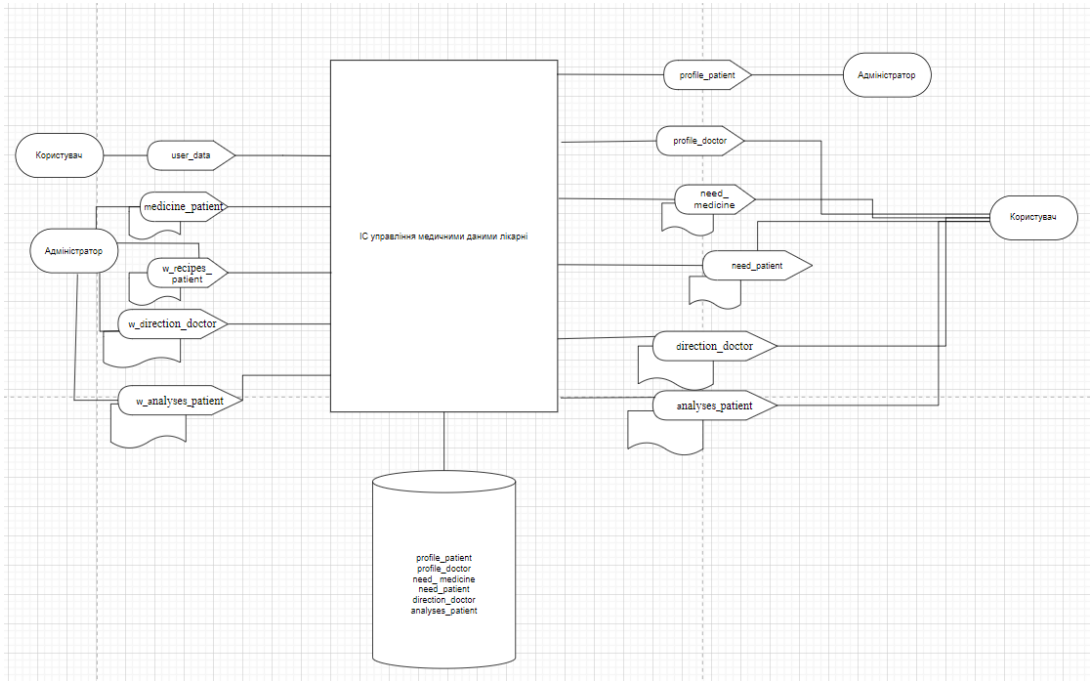


Рисунок 2.8 - Інформаційна модель задачі

Джерело: розроблено автором самостійно

2.3 Моделювання інформаційної підсистеми

Use-Case діаграма - діаграма UML, яка моделює функціональні вимоги системи, показуючи взаємодію між системою та її зовнішніми користувачами (акторами) [10]. Вона складається з акторів, Use-Case'ів (сценаріїв використання) та зв'язків між ними. Діаграма (рис. 2.9) допомагає розуміти, як система використовується і які функції вона повинна виконувати.

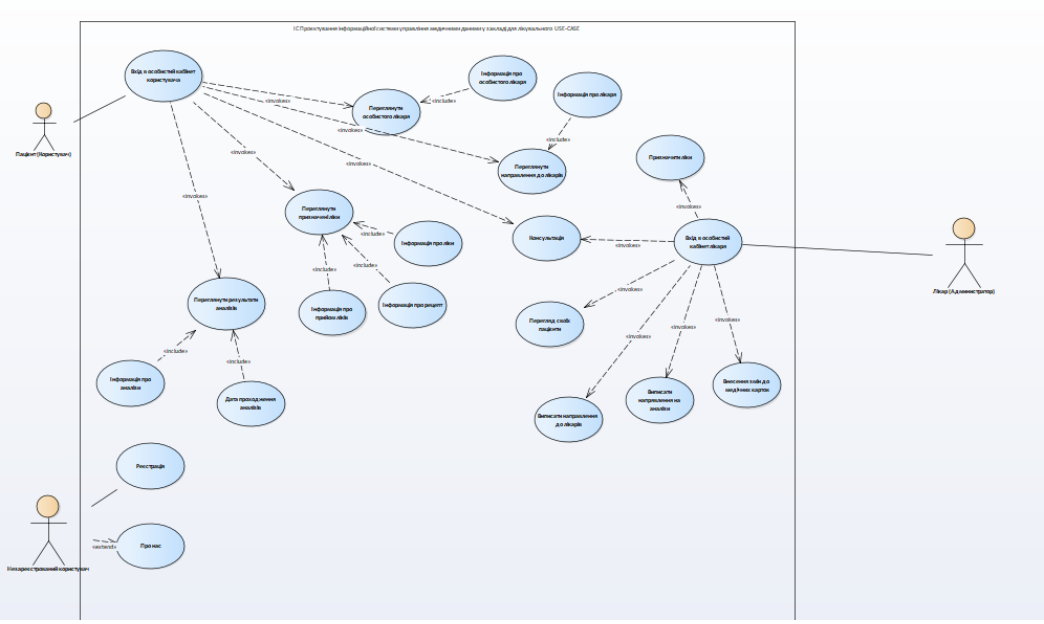


Рисунок 2.9 - Діаграма Use-Case

Джерело: розроблено автором самостійно

На рисунку 2.10-2.15 подано текстові сценарії для варіанта використання. Для підготовки сценаріїв використано розробник сценаріїв - Scenario Builder в середовищі Enterprise Architect.

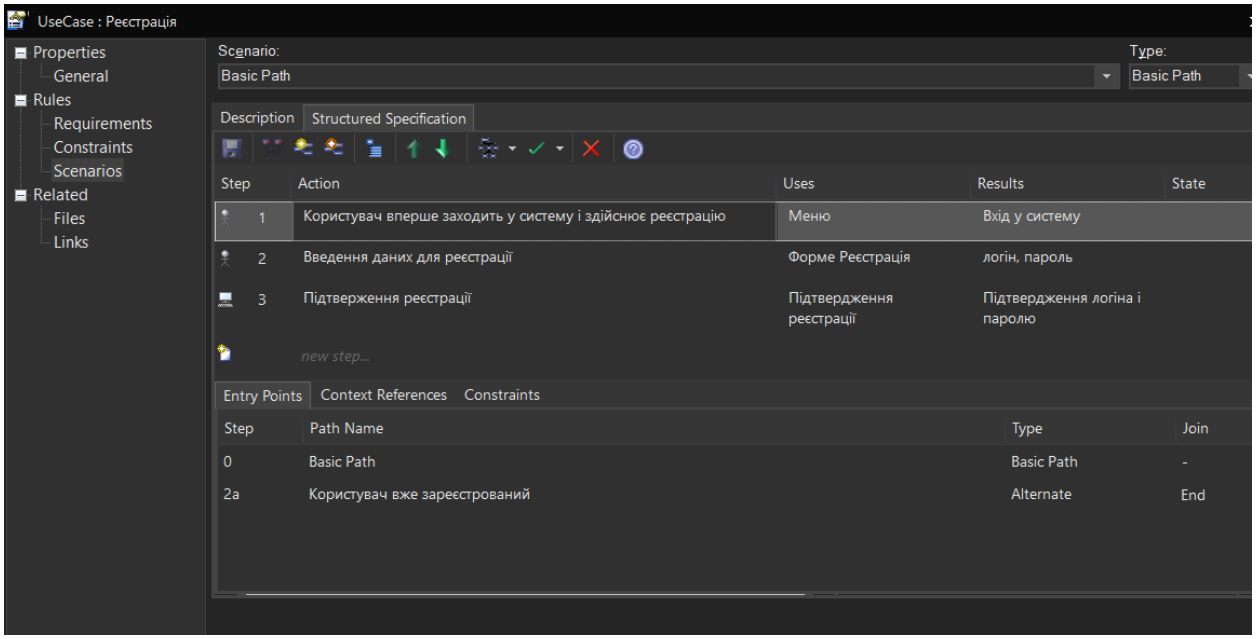


Рисунок 2.10 - Тестовий сценарій для варіанту використання «Реєстрація»

Джерело: розроблено автором самостійно

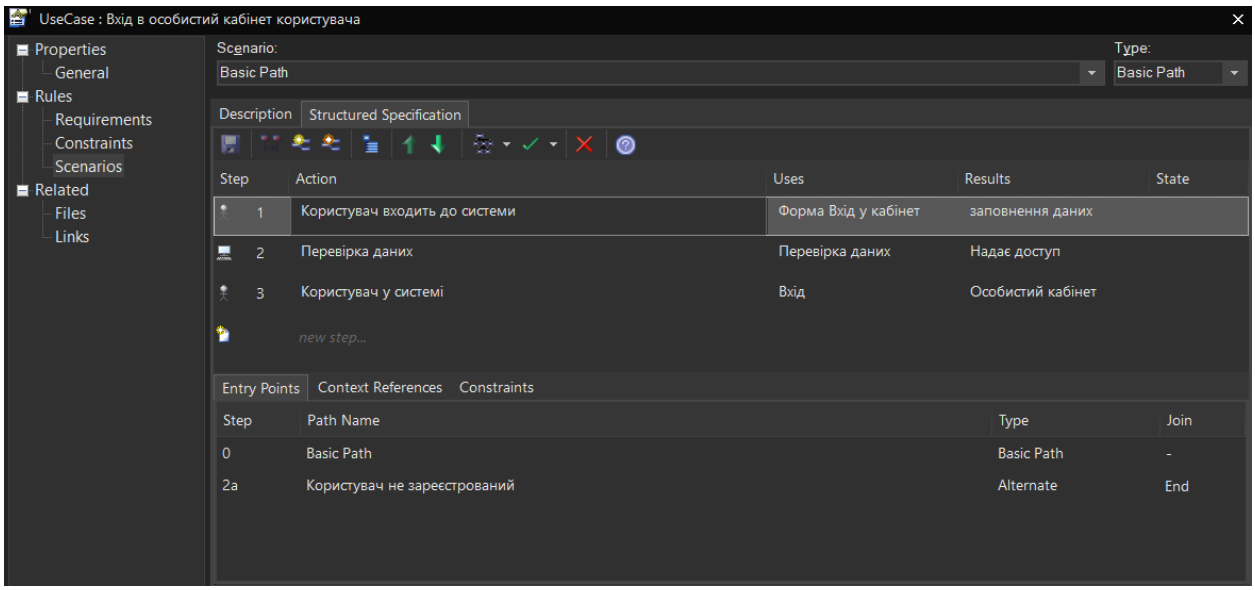


Рисунок 2.11 - Тестовий сценарій для варіанту використання «Вхід в особистий кабінет користувача»

Джерело: розроблено автором самостійно

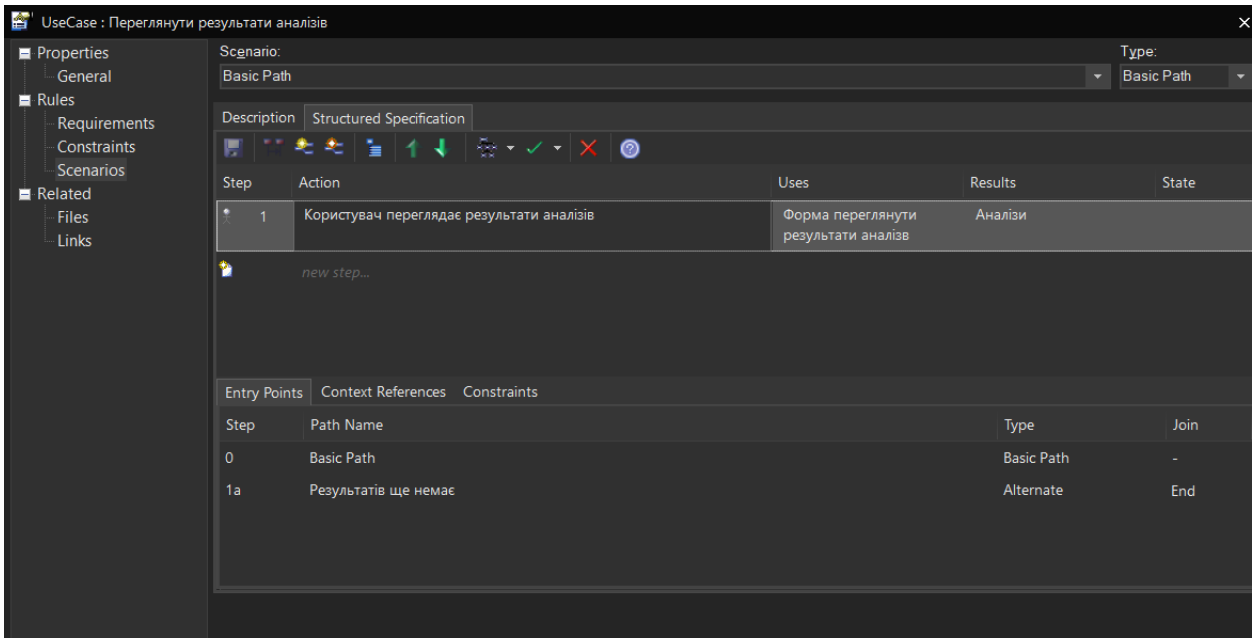


Рисунок 2.12 - Тестовий сценарій для варіанту використання «Переглянути результати аналізів»

Джерело: розроблено автором самостійно

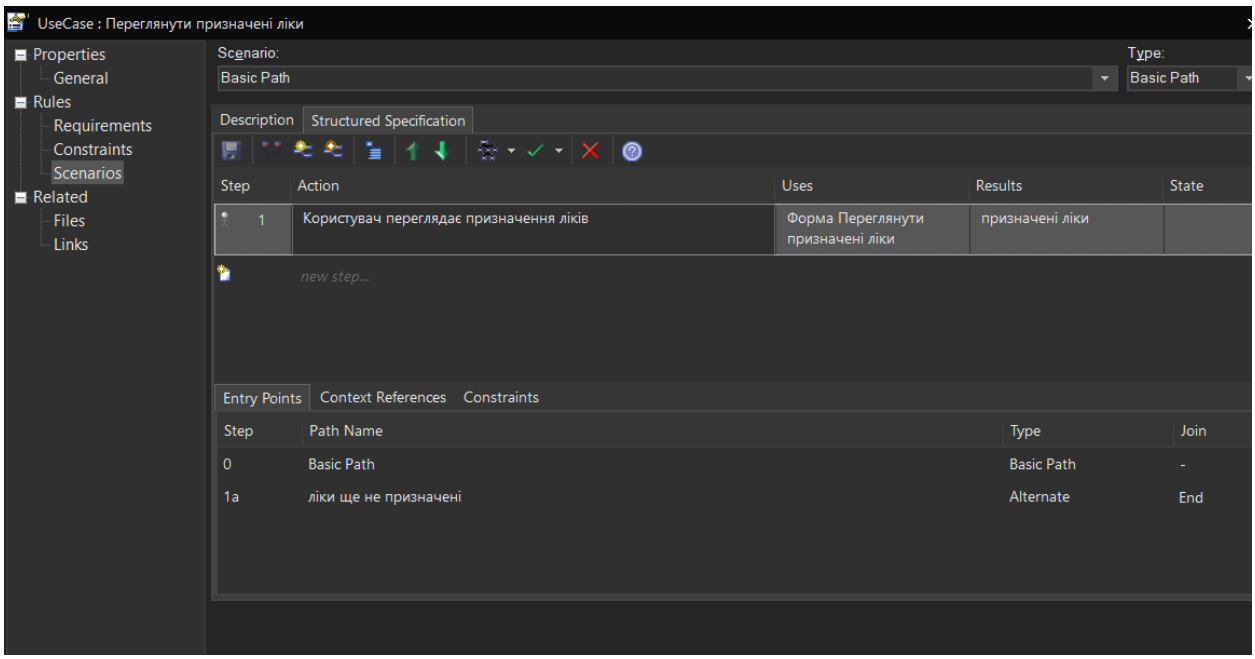


Рисунок 2.13 - Тестовий сценарій для варіанту використання «Переглянути призначені ліки»

Джерело: розроблено автором самостійно

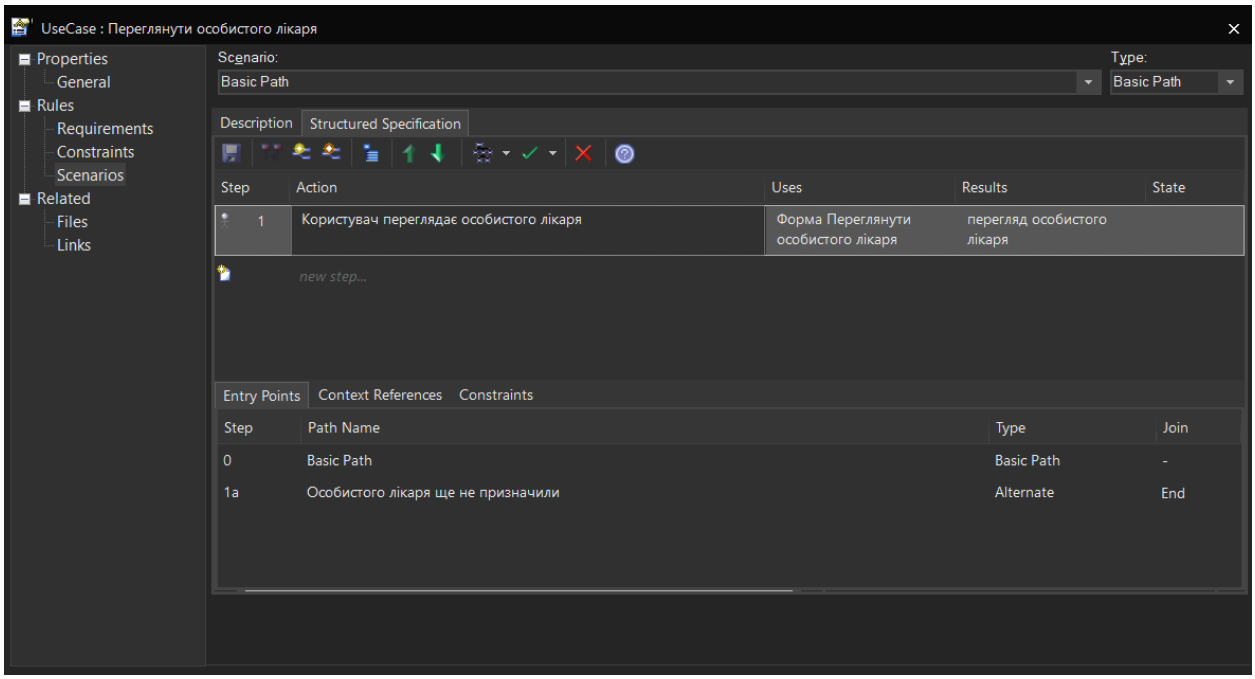


Рисунок 2.14 - Тестовий сценарій для варіанту використання «Переглянути особистого лікаря»

Джерело: розроблено автором самостійно

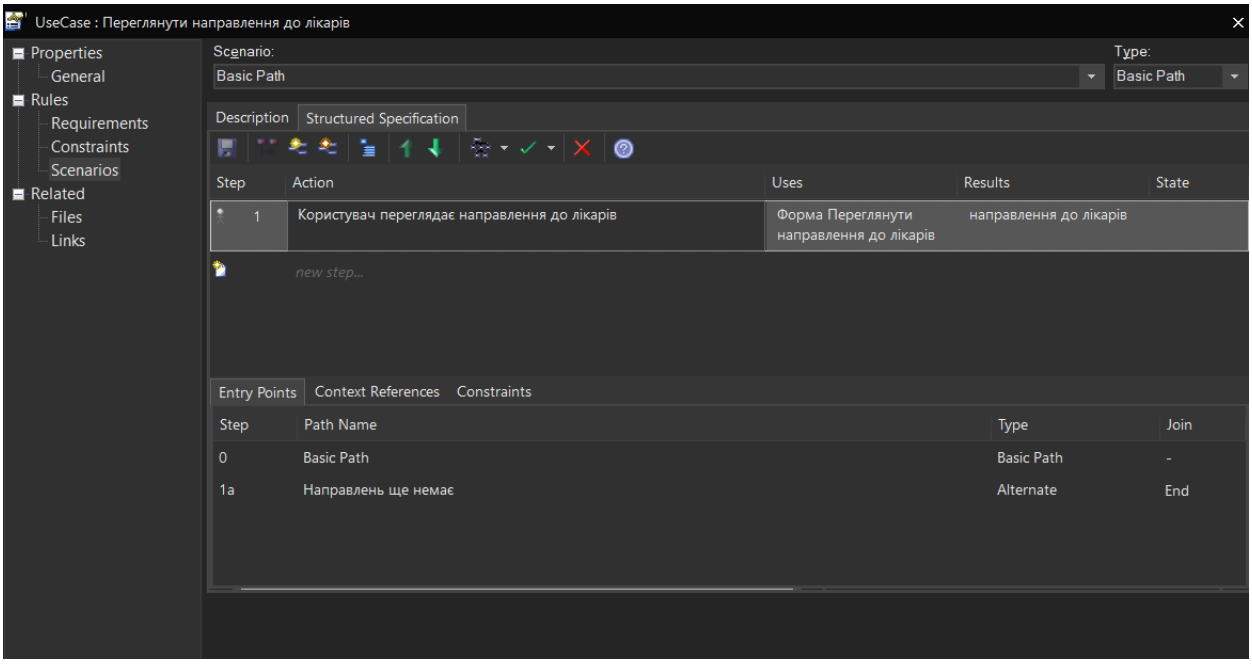


Рисунок 2.15 - Тестовий сценарій для варіанту використання «Переглянути направлення до лікарів»

Джерело: розроблено автором самостійно

На діаграмах послідовності (рис. 2.16-2.21) подано модель системних прецедентів акторів при взаємодії з системою

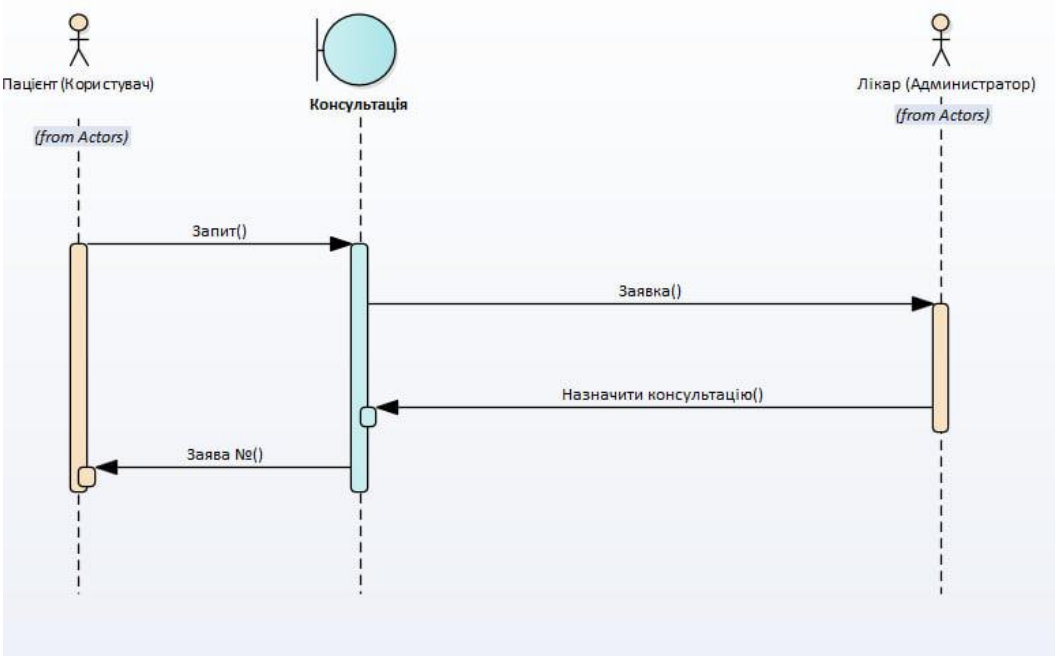


Рисунок 2.16 - Діаграма послідовності Консультація

Джерело: розроблено автором самостійно

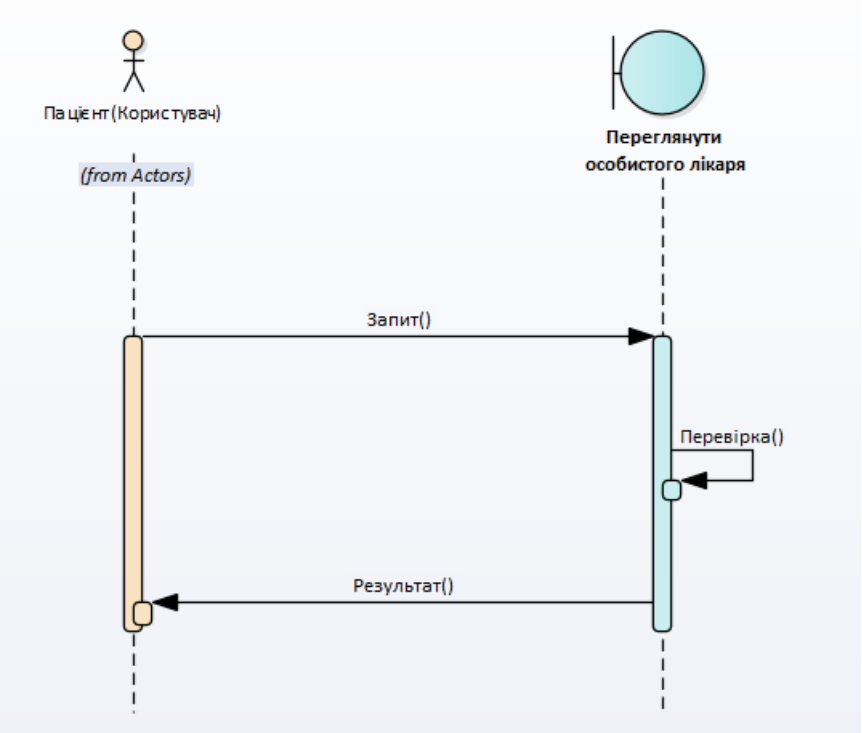


Рисунок 2.17 - Діаграма послідовності особистий лікар

Джерело: розроблено автором самостійно

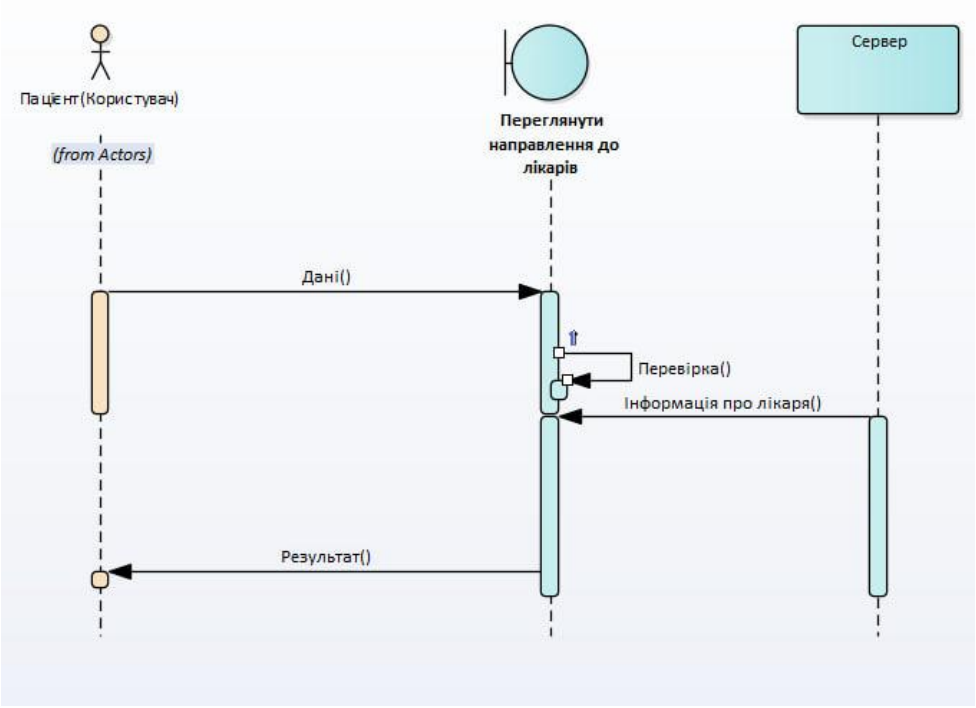


Рисунок 2.18 - Діаграма послідовності Переглянути направлення до лікаря

Джерело: розроблено автором самостійно

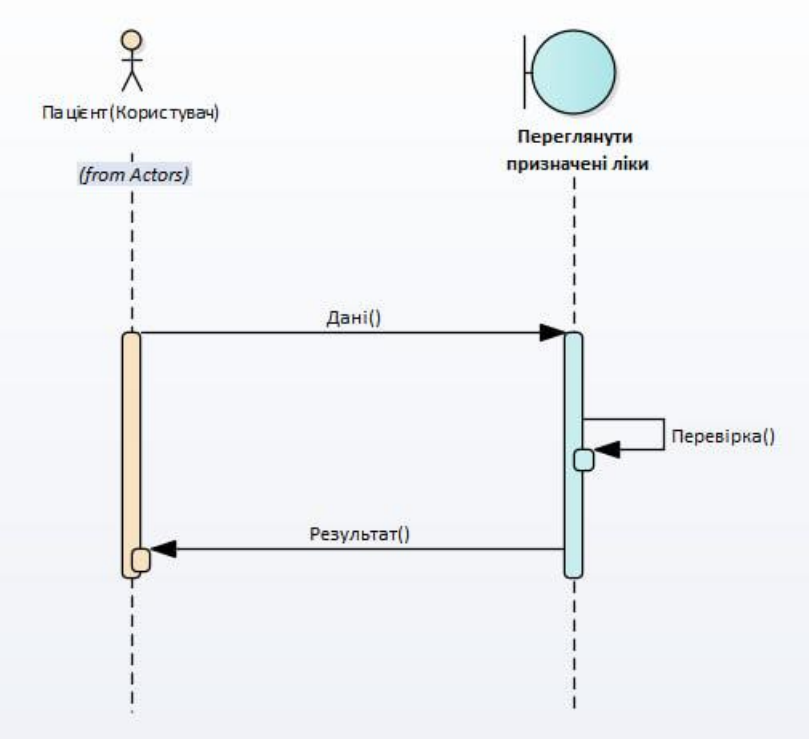


Рисунок 2.19 - Діаграма послідовності Переглянути призначення ліків

Джерело: розроблено автором самостійно

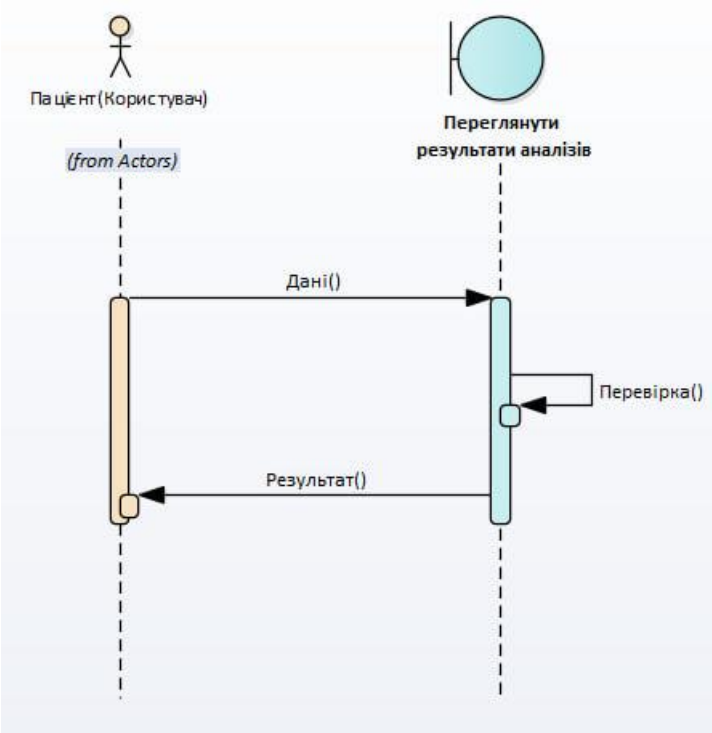


Рисунок 2.20 - Діаграма послідовності Переглянути результати аналізів

Джерело: розроблено автором самостійно

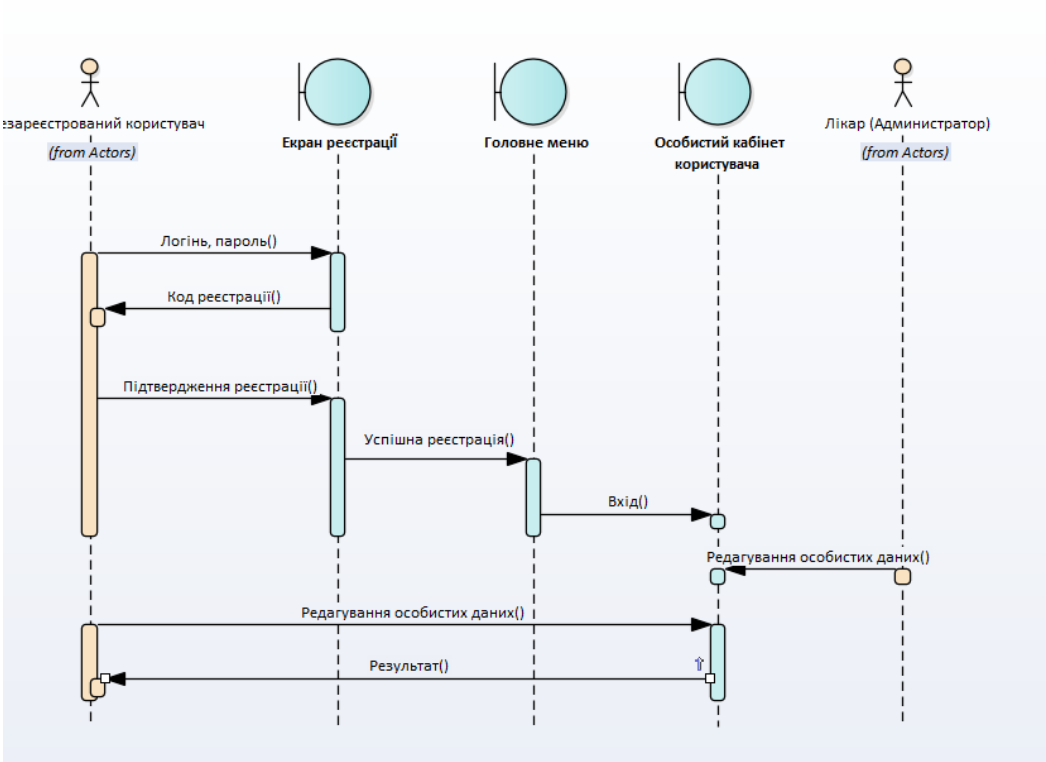


Рисунок 2.21 - Діаграма послідовності Реєстрація

Джерело: розроблено автором самостійно

2.3.2 Моделювання структури системи

Діаграми внутрішніх блоків SysML допомагають визначати структуру системи та її компонентів, їх властивості та взаємозв'язки, що сприяє кращому розумінню систем (рис. 2.22) [11].

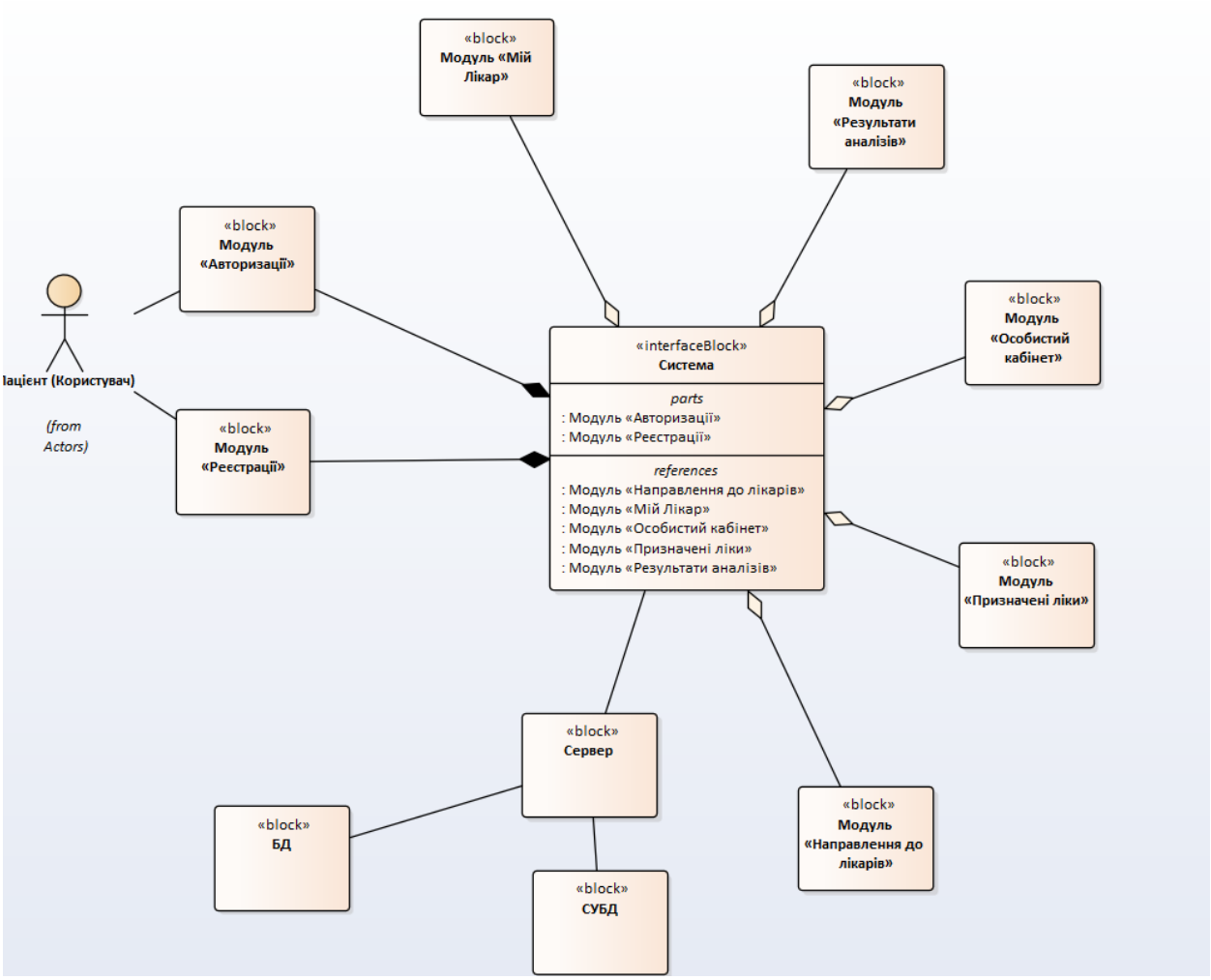


Рисунок 2.22 - Діаграма внутрішніх блоків

Джерело: розроблено автором самостійно

На діаграмі класів-сутностей (рис. 2.23) відображено основні класи-сутності:

- user_data (Дані користувача);
- personal_doctor (Особистий лікар);
- w_recipes_patient (Рецепти пацієнта);
- medicine_patient (Ліки пацієнта);
- w_direction_doctor (Направлення до лікарів);
- w_analyses_patient (Аналізи пацієнта);
- doctor_data (Дані лікарів).

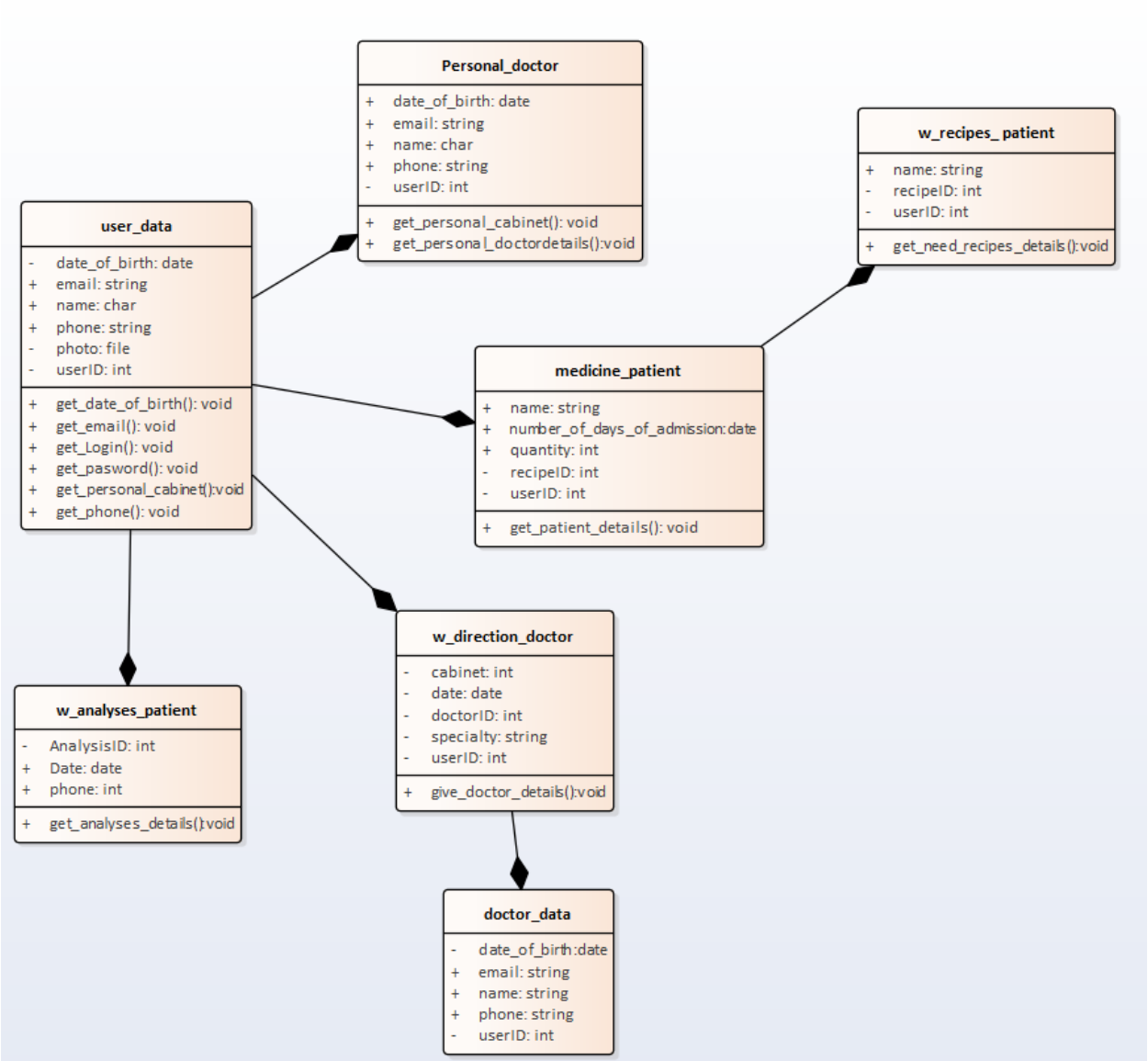


Рисунок 2.23 - Діаграма класів-сутностей

Джерело: розроблено автором самостійно

На діаграмі (рис. 2.24) подані класи керування.

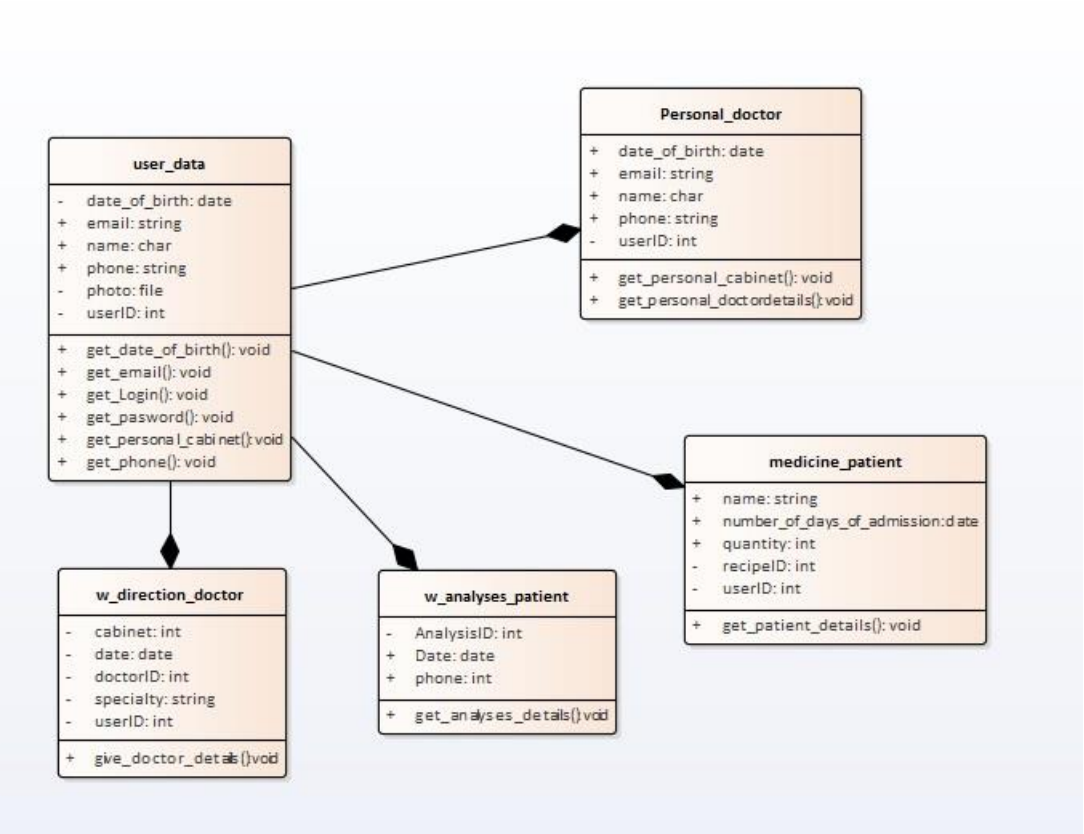


Рисунок 2.24 - Діаграма керування

Джерело: розроблено автором самостійно

На рис. 2.25 представлені граничні класи, за допомогою яких можна змоделювати інтерфейси системи.

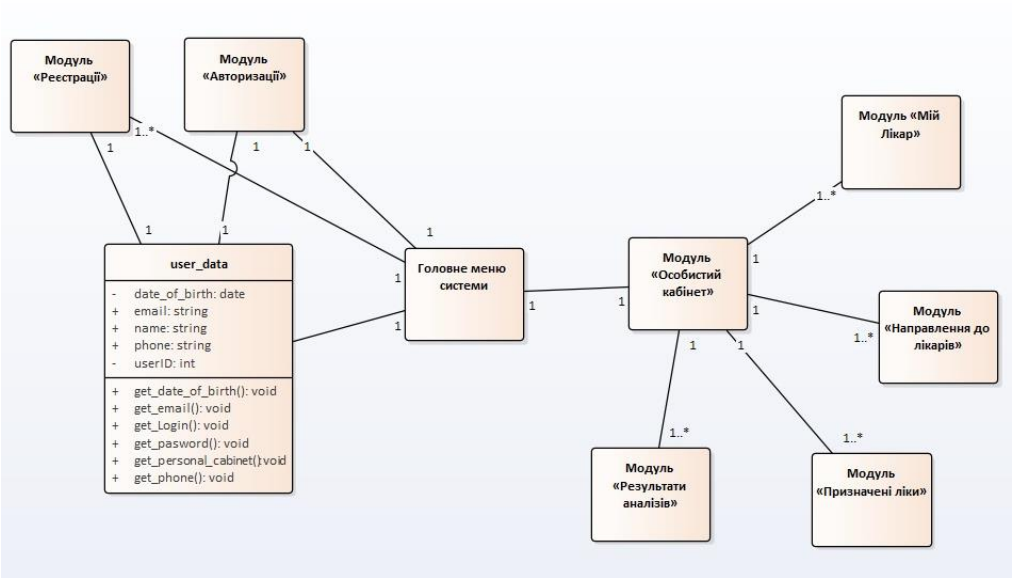


Рисунок 2.25 - Діаграма граничних класів

Джерело: розроблено автором самостійно

Перевірка вимог до розробки системи автоматизації та їх залежність від компонентів системи показана на діаграмі трасування(рис. 2.26), яка показує високорівневі функціональні вимоги та прецеденти, а також відповідність між ними.

Діаграми трасування застосовуються для визначення та аналізу різноманітних зв'язків [12].

Матриця взаємозв’язків , доповнює діаграму трасування як показано на рисунку 2.27.

Матриця зв’язків - інструмент для візуалізації того, як вимоги пов’язані між собою та з іншими елементами моделі у візуально привабливій матриці або електронній таблиці [13].

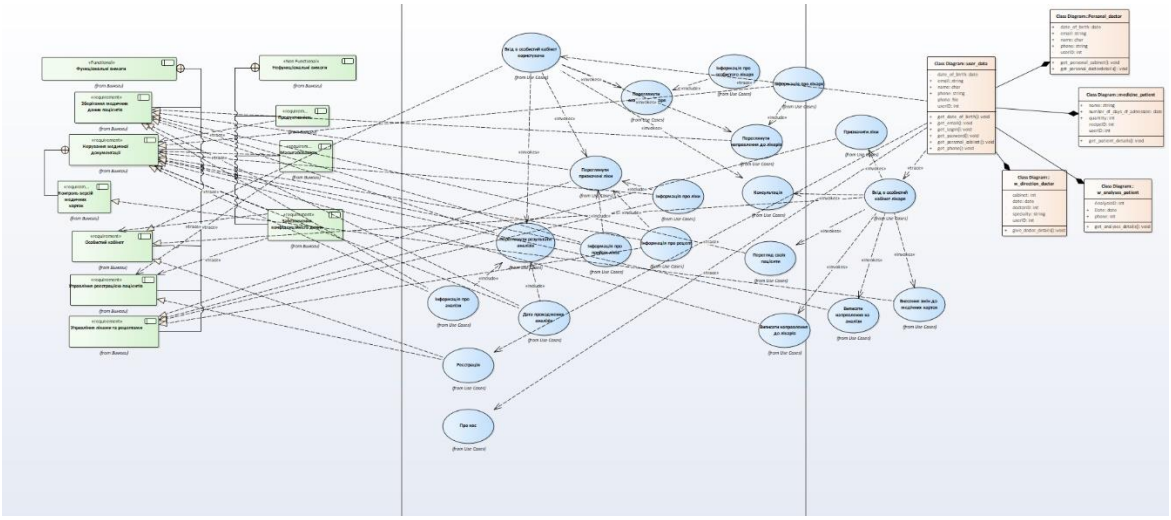


Рисунок 2.26 - Діаграма трасування

Джерело: розроблено автором самостійно

Рисунок 2.27 - Матриця взаємозв’язків прецедентів та вимог

Джерело: розроблено автором самостійно

В даному розділі бакалаврського проекту було розглянуто вимоги, діаграму Use-Case, текстові сценарії, діаграми послідовності, діаграму внутрішніх блоків, діаграму класів, діаграму керування, діаграму трасування та матрицю взаємозв’язків, описано вхідні та вихідні дані, математичні формули і алгоритм розв’язання задачі.

РОЗДІЛ 3 ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1 Інформаційне забезпечення

3.1.1 Загальна характеристика інформаційного забезпечення

Інформаційне забезпечення інформаційної системи управління медичними даними для лікарні включає: електронну медичну картку, ефективне функціонування системи, інтерфейси користувача, бази даних та покращують якість медичних послуг.

Передання інформації може здійснюватися як вручну, так і автоматично. Для цього використовуються наступні методи:

- Локальні мережі (LAN): для обміну інформацією між різними відділами лікарні.
- Інтернет: для безпечного передання даних з однієї установи до іншої або до центральних баз даних.
- Електронна пошта та спеціалізовані медичні платформи: для комунікації з зовнішніми організаціями.
- Системи класифікації та кодування включає: код зареєстрованого користувача, код лікарів, код рецептів, код ліків та код аналізів.

Система класифікації та кодування - сукупність методів, правил та процесів, що використовуються для впорядкування і ідентифікації різних об'єктів чи інформації за допомогою умовних позначень, або кодів. Класифікація передбачає розподіл об'єктів на групи за певними ознаками, а кодування - присвоєння цим групам або об'єктам унікальних кодів. Коди можуть бути цифровими, буквено-цифровими або штриховими, що спрощує їх автоматизоване зчитування та обробку [14].

Проектування форм первинних документів включає: реєстрації пацієнта, форма медичного огляду, форма призначення лікування, медичні карти

пацієнтів: структура, що включає всі необхідні поля для введення особистих даних, історії хвороби, діагнозів та призначень.

Проектування форм первинних документів полягає у створенні структурованих документів, які служать для фіксації та обліку різних господарських операцій і подій [15].

До внутрішньо машинної інформації належать таблиці бази даних. До вхідної інформації входять такі таблиці як: user_data medicine_patient w_recipes_ patient w_direction_doctor w_analyses_patient. До вихідної інформації входять такі таблиці як: profile_patient profile_doctor need_ medicine need_recipes direction_doctor analyses_patient

Внутрішньо машинна інформація - сукупність даних, які обробляються та зберігаються всередині інформаційної системи або обчислювальної машини. Вона включає всі типи даних, що використовуються для виконання програмних алгоритмів, управління системними процесами, а також для забезпечення функціонування різних прикладних програм [16].

До зовнішньо машинної інформації належать : Реклама, брошури

Зовнішньо машинна інформація - дані, які зберігаються поза обчислювальною системою та використовуються для подальшої обробки і введення в систему [17].

Дані зберігаються в базі даних, яка може бути розміщена у хмарному середовищі або на власному сервері.

Схема інформаційного забезпечення показана на рисунку 3.1.

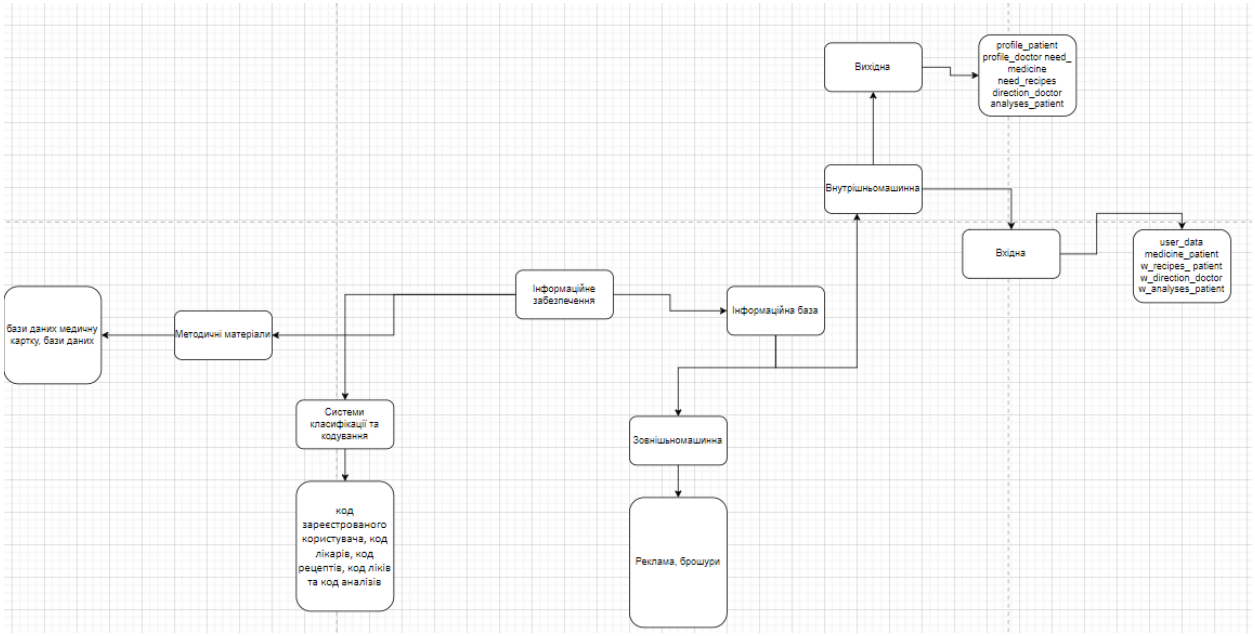


Рисунок 3.1 - Схема інформаційного забезпечення

Джерело: розроблено автором самостійно

3.1.2 Організація збору і передавання первинної інформації

Первинна інформація в лікувальному закладі може включати: Дані про пацієнтів (медичні картки, анамнез, результати аналізів, діагнози, плани лікування). Дані про медичний персонал (кваліфікація, розклад, спеціалізація). Дані про матеріально-технічне забезпечення (медичне обладнання, лікарські засоби). Дані про адміністративну діяльність (фінансові звіти, графіки роботи, статистика).

Первинна інформація - дані, які отримані безпосередньо з джерела, що відображає реальний об'єкт або подію в момент їхнього існування [18].

Збір даних здійснюється з різних джерел, таких як електронні медичні картки, лабораторні інформаційні системи, адміністративні системи та інші. Для цього можуть використовуватися різні методи: Анкетування та опитування медичного персоналу. Витяг даних з існуючих систем та баз даних. Використання сенсорних пристроїв та медичних приладів для автоматичного збору даних.

Зібрана інформація повинна бути передана в інформаційну систему управління медичними даними. Це включає: Перетворення паперових даних в електронний формат шляхом сканування та розпізнавання тексту. Інтеграція різних інформаційних систем через API або інші методи інтеграції. Використання безпечних каналів передачі даних для забезпечення конфіденційності та цілісності інформації.

3.1.3 Побудова системи класифікації та кодування

Системи класифікації та кодування - методи і правила, які використовуються для систематизації об'єктів та призначення їм унікальних кодів для полегшення обробки, зберігання та пошуку інформації [19].

Алфавітно-цифровий код - система кодування, яка використовує як літери алфавіту (від A до Z), так і цифри (від 0 до 9) для представлення даних. Такі коди використовуються для ідентифікації, зберігання і передачі інформації в зручній і компактній формі [20].

1. Код зареєстрованого користувача (пацієнта)

Метод кодування

Алфавітно-цифровий код: Цей метод забезпечує унікальність і дозволяє включати різні атрибути.

Структура коду

Формат коду: ПАЦ-YYYYMMDD-XXXX

ПАЦ: Префікс, що означає "пацієнт".

YYYYMMDD: Дата народження пацієнта.

XXXX: Чотиризначний унікальний ідентифікатор, що генерується автоматично.

Довжина коду: 4 символи (префікс) + 8 символів (дата народження) + 4 символи (ідентифікатор) = 16 символів.

2. Код лікарів

Метод кодування

Алфавітно-цифровий код: Включає спеціалізацію лікаря та рік початку роботи.

Структура коду

Формат коду: ЛІК-SSS-YYYY-XXXX

ЛІК: Префікс, що означає "лікар".

SSS: Спеціалізація лікаря (наприклад, KRD для кардіологів, NRL для неврологів).

YYYY: Рік початку роботи.

XXXX: Чотиризначний унікальний ідентифікатор.

Довжина коду: 3 символи (префікс) + 3 символи (спеціалізація) + 4 символи (рік) + 4 символи (ідентифікатор) = 14 символів.

3. Код рецептів

Метод кодування

Алфавітно-цифровий код: Включає дату виписки та унікальний ідентифікатор.

Структура коду

Формат коду: РЕЦ-YYYYMMDD-XXXX

РЕЦ: Префікс, що означає "рецепт".

YYYYMMDD: Дата виписки рецепта.

XXXX: Чотиризначний унікальний ідентифікатор.

Довжина коду: 3 символи (префікс) + 8 символів (дата) + 4 символи (ідентифікатор) = 15 символів.

Приклад: РЕЦ-20240530-0456

4. Код ліків

Метод кодування

Алфавітно-цифровий код: Включає код ліків за АТХ і унікальний ідентифікатор.

Структура коду

Формат коду: ЛІК-АТХ-XXXX

ЛІК: Префікс, що означає "ліки".

АТХ: Код ліків згідно з Анатомо-терапевтичною хімічною класифікаційною системою (АТС).

XXXX: Чотиризначний унікальний ідентифікатор.

Довжина коду: 3 символи (префікс) + 7 символів (АТХ) + 4 символи (ідентифікатор) = 14 символів.

5. Код аналізів

Метод кодування

Алфавітно-цифровий код: Включає дату проведення аналізу та унікальний ідентифікатор.

Структура коду

Формат коду: АНЛ-YYYYMMDD-XXXX

АНЛ: Префікс, що означає "аналіз".

YYYYMMDD: Дата проведення аналізу.

XXXX: Чотиризначний унікальний ідентифікатор.

Довжина коду: 3 символи (префікс) + 8 символів (дата) + 4 символи (ідентифікатор) = 15 символів.

3.1.4 Проектування форм первинних документів, машинограм та відеокадрів

У контексті медичних закладів первинні документи фіксують всі етапи взаємодії з пацієнтами.

Форма медичного огляду фіксує результати огляду пацієнта лікарем. Вона є важливим документом для діагностування захворювань та планування лікування.

Рисунок 3.2 - Форма медичного огляду

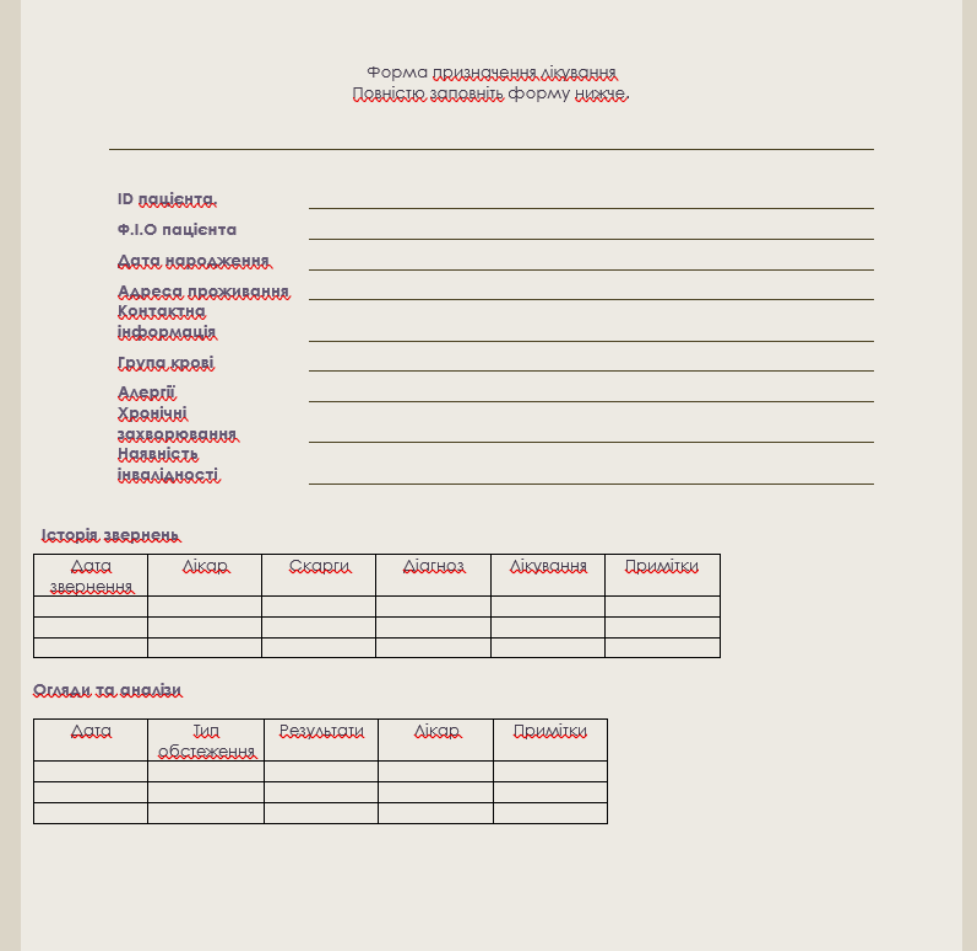
Джерело: розроблено автором самостійно

Форма призначення лікування використовується для фіксації лікарських призначень та рекомендацій. Вона є важливою для забезпечення безперервності та правильності лікувального процесу.

Рисунок 3.3 - Форма призначення лікування

Джерело: розроблено автором самостійно

Медична картка пацієнта є центральним документом, який містить повну інформацію про медичну історію пацієнта.



Форма призначення лікування
Повністю заповніть форму нижче.

ID пацієнта
Ф.І.О пацієнта
Дата народження
Адреса проживання
Контактна інформація
Група крові
Алергії
Хронічні захворювання
Наявність інвалідності

Історія звернень

Дата звернення	Лікар	Скарги	Діагноз	Лікування	Примітки

Огляди та аналізи

Дата	Тип обстеження	Результати	Лікар	Примітки

Рисунок 3.4 - Медична картка пацієнта

Джерело: розроблено автором самостійно

3.1.5 Структура інформаційних масивів

Структура інформаційних масивів - організація та упорядкування даних, які зберігаються в інформаційній системі. Вона включає визначення складу, змісту та способу представлення даних в пам'яті обчислювальної системи. Структурування інформаційних масивів забезпечує ефективне зберігання, доступ, управління та обробку даних [21].

Таблиця 3.1 - інформаційний масив user_data

Інформаційний масив user_data

Найменування масиву - user_data

Ідентифікатор масиву - ID

Найменування носія інформації - Жорсткий диск

Максимальний об’єм масиву - 100,000 записів

Довжина запису - 224 символів (або байтів)

Метод організації - База даних

Ключі упорядкування - Код користувача

Ідентифікатор індексного масиву - userID

Найменування	Ідентифікатор у програмі	Тип даних	PK/FK	Умова на значення	Обов’язкове поле	Індексне поле
Код користувача	userID	9(16)	PK	>0	Так	ІНД
Дата народження	date_of_birth	date(8)	-	-	Так	-
Пошта	email	X(45)	-	-	Так	-
Телефон	phone	9(10)	-	-	Так	-
Ф.І.О.	name	X(45)	-	-	Так	-
Фото	Photo	BLOB(100)	-	-	Так	-

Джерело: розроблено автором самостійно

Таблиця 3.2 - інформаційний масив Personal_doctor

Інформаційний масив Personal_doctor

Найменування масиву - Personal_doctor

Ідентифікатор масиву - doctorID

Найменування носія інформації - Жорсткий диск

Максимальний об’єм масиву - 100,000 записів

Довжина запису - 177 символів (або байтів)

Метод організації - База даних

Ключі упорядкування - Код лікаря

Ідентифікатор індексного масиву - doctorID

Найменування	Ідентифікатор у програмі	Тип даних	PK/FK	Умова на значення	Обов’язкове поле	Індексне поле
Код лікаря	doctorID	9(14)	PK	>0	Так	ІНД
Дата народження	date_of_birth	date(8)	-	-	Так	-
Пошта	email	9(10)	-	-	Так	-
Телефон	phone	X(45)	-	-	Так	-

Найменування	Ідентифікатор у програмі	Тип даних	PK/FK	Умова на значення	Обов'язкове поле	Індексне поле
Ф.І.О.	name	BLOB (100)	-	-	Так	-

Джерело: розроблено автором самостійно

Таблиця 3.3 - інформаційний масив analyses_patient

Інформаційний масив analyses_patient

Найменування масиву - analyses_patient

Ідентифікатор масиву -AnalysisID

Найменування носія інформації -Жорсткий диск

Максимальний об'єм масиву -100,000 записів

Довжина запису -98 символів (або байтів)

Метод організації - База даних

Ключі упорядкування - Код анализу

Ідентифікатор індексного масиву - AnalysisID

Найменування	Ідентифікатор у програмі	Тип даних	PK/FK	Умова на значення	Обов'язкове поле	Індексне поле
Код аналізу	AnalysisID	9(15)	PK	>0	Так	ІНД
Дата	Date	8	-	-	Так	-
Назва	name_analysis	X(45)	-	-	Так	-
Код лікаря	doctorID	9(14)	-	>0	Так	-
Код користувача	userID	9(16)	-	>0	Так	-

Джерело: розроблено автором самостійно

Таблиця 3.4 - w_analyses_patient

Інформаційний масив w_analyses_patient

Найменування масиву - w_analyses_patient

Ідентифікатор масиву - AnalysisID

Найменування носія інформації -Жорсткий диск

Максимальний об'єм масиву -100,000 записів

Довжина запису -82 символів (або байтів)

Метод організації - База даних

Ключі упорядкування - Код анализу

Ідентифікатор індексного масиву - AnalysisID

Найменування	Ідентифікатор у програмі	Тип даних	PK/FK	Умова на значення	Обов’язкове поле	Індексне поле
Код аналізу	AnalysisID	9(15)	PK	>0	Так	ІНД
Дата	Date	date(8)	-	-	Так	-
Назва	name_analysis	X(45)	-	-	Так	-
Код лікаря	doctorID	9(14)	-	>0	Так	-

Джерело: розроблено автором самостійно

Таблиця 3.5 - w_recipes_ patient

Інформаційний масив w_recipes_ patient

Найменування масиву - w_recipes_ patient

Ідентифікатор масиву - recipesID

Найменування носія інформації -Жорсткий диск

Максимальний об’єм масиву -100,000 записів

Довжина запису -74 символів (або байтів)

Метод організації - База даних

Ключі упорядкування - Код рецепту

Ідентифікатор індексного масиву - recipesID

Найменування	Ідентифікатор у програмі	Тип даних	PK/ FK	Умова на значення	Обов’язкове поле	Індексне поле
Код анализу	recipesID	9(15)	PK	>0	Так	ІНД
Назва	name	X(45)	-	-	Так	-
Код лікаря	doctorID	9(14)	-	>0	Так	-

Джерело: розроблено автором самостійно

Таблиця 3.6 - medicine_patient

Інформаційний масив medicine_patient

Найменування масиву - medicine_patient

Ідентифікатор масиву - medicine_ID

Найменування носія інформації -Жорсткий диск

Максимальний об’єм масиву -100,000 записів

Довжина запису -180 символів (або байтів)

Метод організації - База даних

Ключі упорядкування - Код ліків

Ідентифікатор індексного масиву - medicine_ID

Найменування	Ідентифікатор у програмі	Формат	PK/FK	Умова на значення	Обов’язкове поле	Індексне поле
Код ліків	medicine_ID	9(14)	PK	>0	Так	ІНД
Назва	name	X(45)	-	-	Так	-
Код користувача	userID	9(16)	-	>0	Так	-
Кількість днів прийому	number_of_days_of_admission	X(45)	PK	-	Так	ІНД
Кількість	quantity	X(45)	-	-	Так	-
Код рецепту	recipeID	9(15)	-	>0	Так	-

Джерело: розроблено автором самостійно

Таблиця 3.7 - doctor_data

Інформаційний масив doctor_data

Найменування масиву - doctor_data

Ідентифікатор масиву - doctor_ID

Найменування носія інформації -Жорсткий диск

Максимальний об’єм масиву -100,000 записів

Довжина запису -226 символів (або байтів)

Метод організації - База даних

Ключі упорядкування - ID

Ідентифікатор індексного масиву - doctor_data

Найменування	Ідентифікатор у програмі	Тип даних	PK/FK	Умова на значення	Обов’язкове поле	Індексне поле
Код лікаря	doctorID	9(14)	PK	>0	Так	ІНД
Дата народження	date_of_birth	date(8)	-	-	Так	-
Пошта	email	9(10)	-	-	Так	-
Телефон	phone	X(45)	-	-	Так	-
Ф.І.О.	name	X(45)	-	-	Так	-
Фото	Photo	BLOB(100)	-	-	Так	-

Джерело: розроблено автором самостійно

Таблиця 3.8 - w_direction_doctor

Інформаційний масив w_direction_doctor

Найменування масиву - w_direction_doctor

Ідентифікатор масиву - doctor_ID

Найменування носія інформації -Жорсткий диск

Максимальний об’єм масиву -100,000 записів

Довжина запису -112 символів (або байтів)

Метод організації - База даних

Ключі упорядкування - Код лікаря

Ідентифікатор індексного масиву - doctor_ID

Найменування	Ідентифікатор у програмі	Формат	PK/FK	Умова на значення	Обов’язкове поле	Індексне поле
Код лікаря	doctorID	9(14)	PK	>0	Так	ІНД
Кабінет	cabinet	X(45)	-	-	Так	-
Дата	date	date(8)	-	-	Так	-
Спеціальність	specialty	X(45)	-	-	Так	-

Джерело: розроблено автором самостійно

3.1.6 Вибір СКБД

Для проектування інформаційної системи управління медичними даними для лікарні обрано СКБД MySQL.

MySQL є найпопулярнішою у світі базою даних з відкритим кодом [22].

Переваги:

- Висока продуктивність та масштабованість;
- Відкритий вихідний код, що дозволяє безкоштовного використання та модифікації;
- Простота встановлення та використання;
- Широка сумісність
- Безпека

3.1.7 Інфологічна модель бази даних

Інфологічна модель бази даних - концептуальне уявлення про дані, яке відображає інформацію про предметну область в абстрактній, логічно структурованій формі. Вона використовується для моделювання сутностей, атрибутів і зв'язків між ними без прив'язки до конкретної реалізації або технології зберігання даних. Інфологічна модель є ключовою частиною

проектування бази даних, забезпечуючи ясне розуміння інформаційних потреб і сприяючи ефективному створенню фізичної моделі бази даних [23]

При проектуванні інформаційної системи управління медичними даними для лікарні було спроектовано інфологічну модель бази даних.

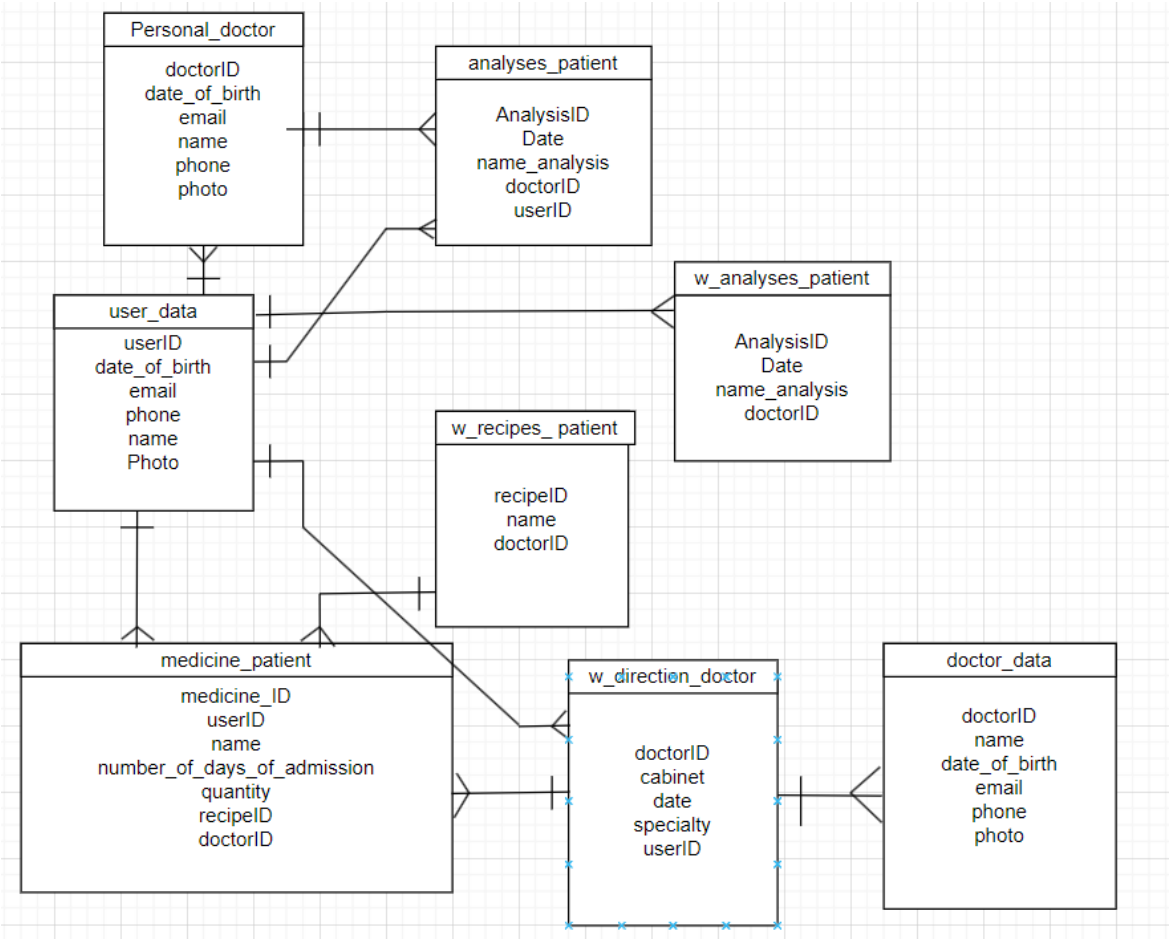


Рисунок 3.5 – Інфологічна модель бази даних

Джерело: розроблено автором самостійно

Інфологічна модель показює сутності та атрибути, а саме:

- user_data - userID - date_of_birth, email, phone, name, Photo;
- Personal_doctor - doctorID, date_of_birth, email, phone, name, Photo;
- analyses_patient - AnalysisID, Date, name_analysis, doctorID, userID;
- w_analyses_patient - AnalysisID, Date, name_analysis, doctorID;
- w_recipes_ patient - recipesID, name, doctorID;

- medicine_patient - medicine_ID, name-userID, number_of_days_of_admission, quantity. recipeID-doctorID;
- doctor_data - doctorID, date_of_birth-email, phone-name, Photo, date_of_birth;
- w_direction_doctor - doctorID, cabinet-date, specialty, userID.

3.1.8 Даталогічна модель бази даних

Даталогічна модель бази даних - модель логічного рівня, яка відображає структуру даних і логічні зв'язки між ними без прив'язки до конкретного способу їх фізичного зберігання. Вона детально описує всі типи зв’язків між даними, такі як один-до-одного, один-до-багатьох, та багато-до-багатьох, а також визначає всі атрибути та обмеження цілісності [24].

При проектуванні інформаційної системи управління медичними даними для лікарні було спроектовано даталогічну модель бази даних.

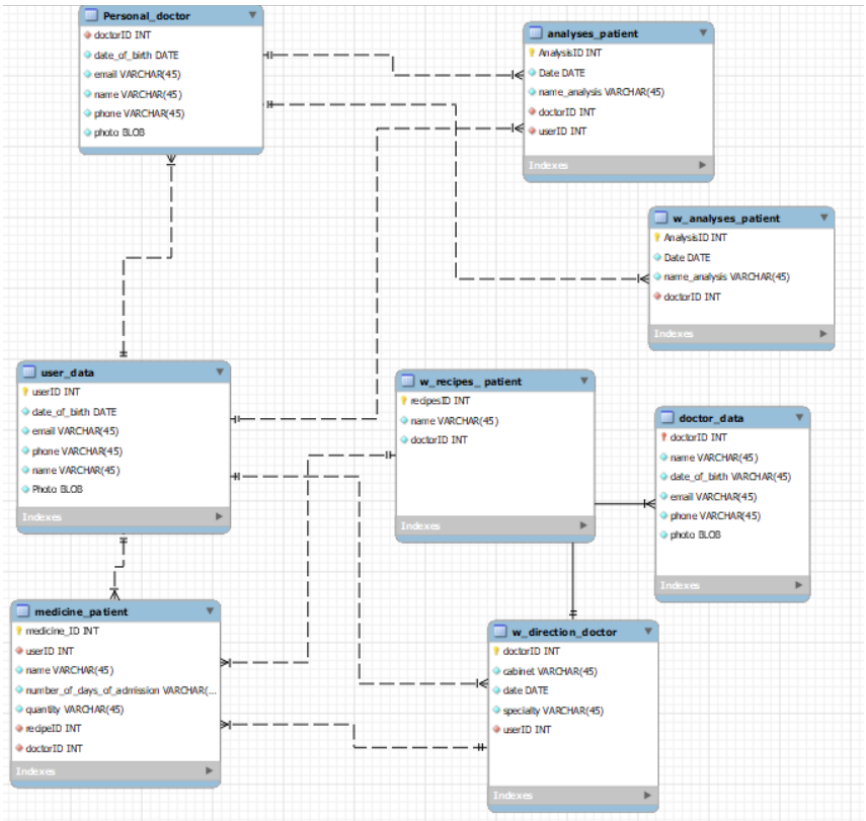


Рисунок 3.6 - даталогічна модель бази даних

Джерело: розроблено автором самостійно

3.2 Технічне забезпечення

Інформаційна система управління медичними даними лікарні передбачає наявність серверу на якому буде розташована на система, база даних та необхідні файли для її функціонування. Доступ клієнтів до системи буде здійснюватися через використання пристрою з доступом до мережі Інтернет.

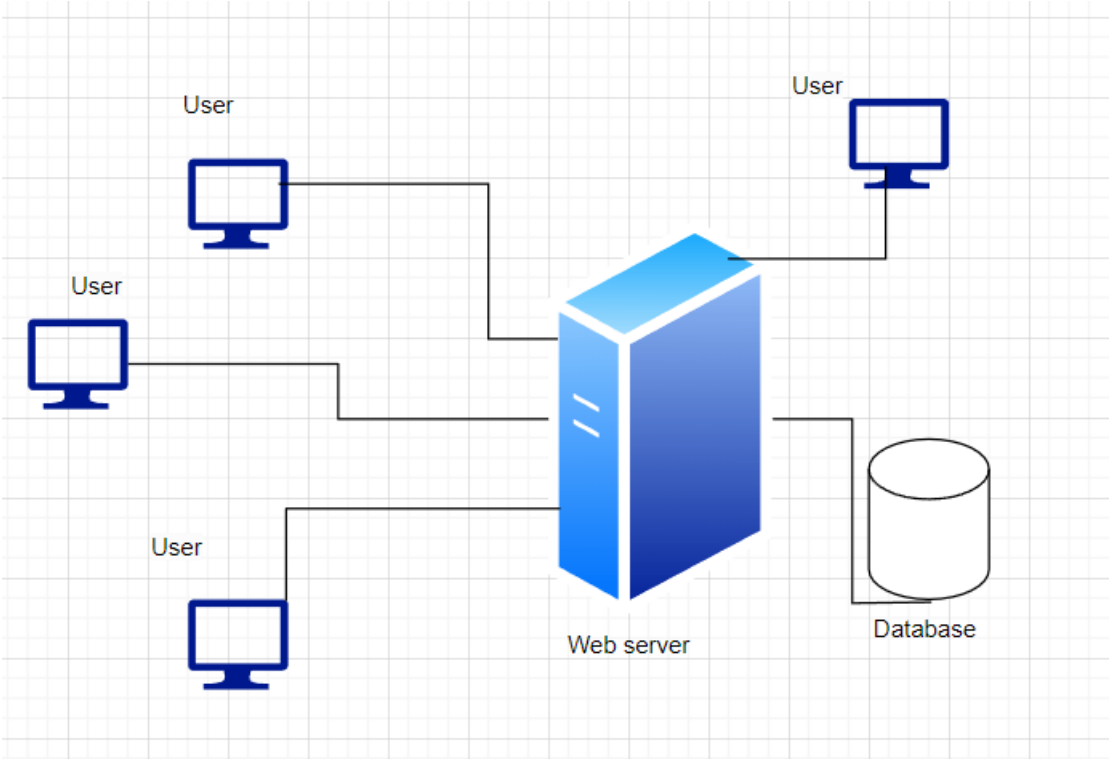


Рисунок 3.7 - Схема взаємодії технічних засобів в системі

Джерело: розроблено автором самостійно

До складу комплексу технічних засобів системи управління медичними даними лікарні входять пристрої та технології, які забезпечують ефективне функціонування всієї системи. Ці засоби включають комп'ютери для медичного персоналу, сервери для зберігання і обробки даних, мережеве обладнання для забезпечення зв'язку між різними компонентами системи, а також різні програмні продукти, що забезпечують управління медичними даними. Окрім цього, використовуються пристрої для введення даних, такі як сканери та медичні термінали, а також засоби для забезпечення безпеки даних, включаючи системи резервного копіювання.

Для захисту сервера від механічних дій потрібно розмістити в спеціально обладнаних приміщеннях, таких як серверні кімнати. Використання серверних стойок і шаф забезпечує додатковий захист від ударів і механічних пошкоджень, а антивібраційні платформи зменшують вплив вібрацій, які можуть негативно вплинути на обладнання.

Для запобігання термічному впливу серверні приміщення повинні бути оснащені надійними системами кондиціонування і вентиляції, які підтримують оптимальну температуру. Датчики температури дозволяють постійно моніторити стан середовища і своєчасно реагувати на зміни. Використання резервних систем охолодження є необхідним для запобігання перегріву в разі виходу з ладу основної системи.

Для захист від електромагнітного впливу потрібно використовувати спеціальні захищені шафи для серверів. Потрібно використовувати екрановані кабелі для передачі даних і живлення, щоб зменшити ризик проникнення електромагнітних перешкод. Фільтри живлення додатково захищають обладнання від електромагнітних імпульсів.

Захист від несанкціонованого доступу. Програмний захист, наприклад, за допомогою багатофакторної аутентифікації для підвищення рівня безпеки при доступі до інформаційних систем.

Для запобігання несанкціонованому доступу до серверів необхідно впроваджувати системи контролю доступу. Відеоспостереження і системи реєстрації доступу дозволяють відстежувати всі відвідування і оперативно реагувати на підозрілі дії. Фізичне розмежування серверного обладнання в окремих кімнатах або зонах, доступ до яких мають тільки авторизовані особи. Потрібно встановити системи сигналізації.

Схема мережі передачі даних.

Загальна структура мережі включає в себе локальну мережу (LAN) та підключення до Інтернету. Використовуються Ethernet з UTP-8 кабелями для підключення користувачів до маршрутизаторів і веб-сервера. Вибір обумовлений високою швидкістю передачі даних і стабільністю з'єднання.

Ethernet через UTP-8 кабелі для локальної мережі. Можливе підключення до Інтернету через оптоволокно UT-11. Використовуються маршрутизатори Cisco або аналогічні за характеристиками для надійності і продуктивності. Мінімум 100 Мбіт/с для внутрішніх з'єднань і 1 Гбіт/с для з'єднання з Інтернетом. Статичний IP для веб-сервера для забезпечення стабільного доступу.

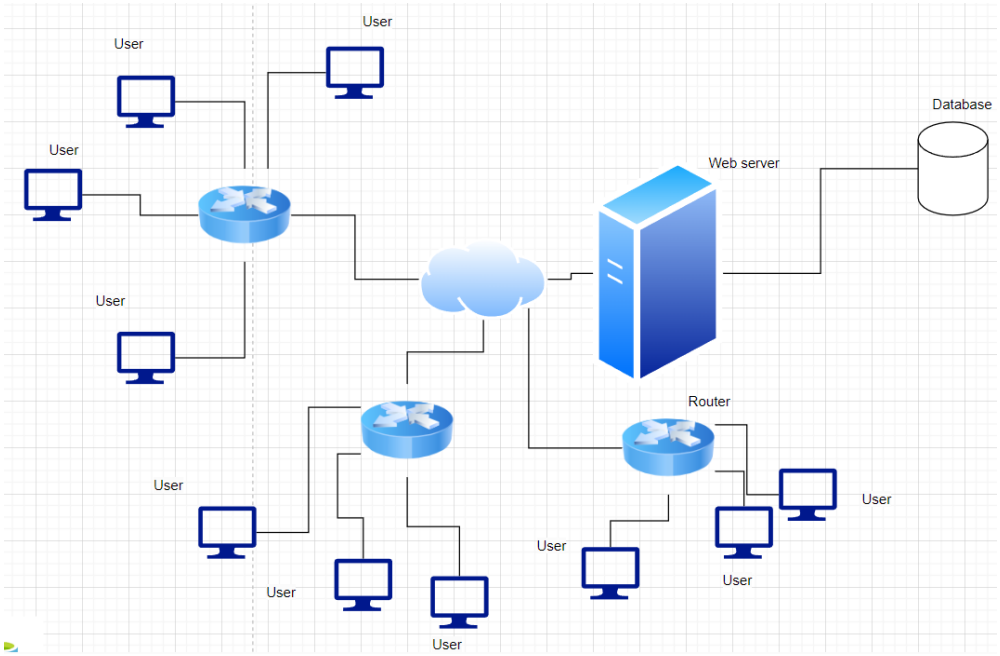


Рисунок 3.8 - Схема мережі передачі даних

Джерело: розроблено автором самостійно

3.3 Програмне забезпечення

Структура програмного забезпечення інформаційної системи включає кілька важливих компонентів, а саме:

- системне програмне забезпечення;
- прикладне програмне забезпечення;
- програмну документацію.

Системне програмне забезпечення

Забезпечує функціонування всіх прикладних програм і включає в себе операційні системи для серверів і робочих станцій, системи управління базами

даних для зберігання медичної інформації, а також засоби для забезпечення безпеки та захисту даних [25]:

Операційні системи, які керують апаратними ресурсами комп'ютера та забезпечують інтерфейс для взаємодії з користувачем і програмами. Прикладами є Windows, Linux, macOS.

Драйвери пристроїв, які дозволяють операційній системі взаємодіяти з апаратними компонентами, такими як принтери, відеокарти, мережеві адаптери.

Прикладне програмне забезпечення дозволяє користувачам виконувати специфічні завдання, що безпосередньо задовольняють їхні потреби.

Структура програмного забезпечення представлена у вигляді схеми

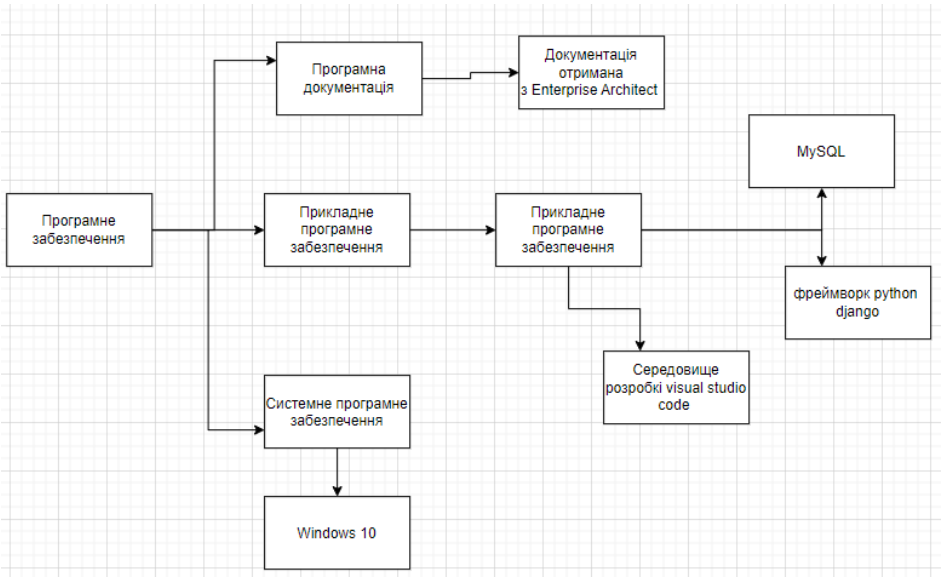


Рисунок 3.9 - Схема структури програмного забезпечення системи

Джерело: розроблено автором самостійно

Проєктування інформаційної системи управління медичними даними лікарні проводилось за допомогою Python та бази даних MYSQL. За допомогою середовищ Enterprise Architect і DrawIo було побудовано схеми і діаграми

Enterprise Architect - платформа, яка полегшує моделювання, аналіз, проектування та створення складних програмних систем. Його метою є

створення процесів UML, специфікацій, моделей для бази даних та інших діаграм, які візуалізують складні системи та керують ними [26].

Drawio - безкоштовний онлайн-інструмент для створення діаграм і схем. Він дозволяє користувачам легко створювати різноманітні графічні схеми, включаючи блок-схеми, діаграми потоків, організаційні діаграми, моделі баз даних та багато іншого. Drawio інтегрується з різними хмарними сервісами, такими як Google Drive, Dropbox, OneDrive, що дозволяє зберігати та спільно використовувати діаграми [27].

Важливу роль відіграють системи керування базами даних, які зберігають та організовують медичні записи. Вибір може падати на MySQL, оскільки він забезпечує високу продуктивність, масштабованість та безпеку даних. Крім того, MySQL має відкритий код і активну спільноту розробників, що дозволяє швидко вирішувати будь-які технічні проблеми та підтримувати систему в актуальному стані. Для забезпечення ще більшої надійності можна використовувати реплікацію баз даних, яка дозволяє зберігати копії даних на декількох серверах, знижуючи ризики втрати інформації [28].

Python - мова програмування, яку використовують для розробки вебзастосунків, програмного забезпечення, машинного навчання. Python застосовують для вирішення робочих завдань у компаніях Google, Instagram, Facebook, IBM, NASA, Dropbox, Netflix та інших. Розробники цінують цю мову програмування за простоту у вивченні, ефективність та мультиплатформність. Крім того, Python має велику різноманітність бібліотек і фреймворків, які сприяють процесу розробки. Основними перевагами Python є його велика спільнота розробників, багатозадачність і можливість створювати різні типи додатків, включаючи веб-додатки, наукові додатки, системи штучного інтелекту тощо, які ідеально підходять для багатьох проектів [29].

«Django» - високорівнева структура Python, яка заохочує швидку розробку та чистий, прагматичний дизайн. Створений досвідченими розробниками, він усуває більшість проблем із веб-розробкою [30].

Прикладне програмне забезпечення

Прикладне програмне забезпечення для такої системи можна класифікувати на кілька категорій:

- програмне забезпечення загального призначення;
- методо орієнтоване програмне забезпечення;
- проблемно-орієнтоване програмне забезпечення;
- програмне забезпечення для роботи глобальних мереж;
- програмне забезпечення для організації обчислювального процесу.

Програмне забезпечення загального призначення включає базові інструменти та додатки, необхідні для щоденної роботи користувачів і адміністраторів системи [31]. Це можуть бути текстові редактори, електронні таблиці, програми для обробки зображень, а також системи управління документами, наприклад, графічний редактор Figma. Ці програми забезпечують загальні функції, необхідні для роботи з медичними даними, створення звітів та обміну інформацією між співробітниками лікарні.

Методоорієнтоване програмне забезпечення складається з інструментів, які використовуються для розробки та тестування методів обробки медичних даних [32]. Це можуть бути статистичні пакети, засоби для аналізу даних та інструменти для моделювання і симуляції медичних процесів. Таке програмне забезпечення допомагає розробникам створювати алгоритми для аналізу медичних даних, оцінки ефективності лікування та прогнозування стану пацієнтів.

Проблемно-орієнтоване програмне забезпечення включає спеціалізовані програми, розроблені для виконання конкретних завдань в рамках інформаційної системи управління медичними даними [33]. Це можуть бути системи для обробки і зберігання електронних медичних записів, програми для управління картками пацієнтів, лабораторними результатами, діагностичними зображеннями та іншими медичними даними. Таке програмне забезпечення забезпечує виконання специфічних функцій, необхідних для

медичної практики, і дозволяє інтегрувати різні джерела даних в єдину систему.

Програмне забезпечення для роботи глобальних мереж необхідне для забезпечення надійного та безпечного обміну медичними даними між різними установами та користувачами [34]. Це включає засоби для управління мережевими підключеннями, забезпечення безпеки даних, шифрування та автентифікації користувачів.

Програмне забезпечення для організації обчислювального процесу включає системи управління базами даних, серверне програмне забезпечення та засоби для адміністрування обчислювальних ресурсів [35]. Вибір СКБД є критично важливим, оскільки воно має бути інтегроване з іншими компонентами системи і забезпечувати високу продуктивність, надійність та безпеку зберігання медичних даних. СКБД повинна підтримувати функції резервного копіювання, відновлення даних і забезпечувати доступ до даних в режимі реального часу.

Програмна документація

Основна мета програмної документації для системи управління медичними даними лікарні полягає в забезпеченні зрозумілості, доступності та ефективного використання системи для користувачів. Ця документація включає в себе:

- Формуляр, де описуються основні характеристики програми, її комплектність, відомості про експлуатацію;
- відомості про призначення програми, області її використання для розв’язання завдань;
- обмеження на застосування мінімальної конфігурації технічних засобів;
- керівництво системного програміста, де надаються відомості для перевірки, забезпечення функціонування й налаштування програми відповідно до умов конкретного застосування;
- опис контрольного прикладу.

Формуляр

Формуляр для Системи Управління Медичними Даними Лікарні

Основні характеристики програми:

- Назва програми: [Назва програми] ;
- цільове призначення: Управління та обробка медичних даних пацієнтів лікарні, забезпечення зручного доступу до інформації для медичного персоналу;

Функціонал:

- Зберігання та оновлення медичних записів пацієнтів;
- система управління термінами та умовами зберігання даних відповідно до законодавства;
- модуль для запису результатів обстежень та діагностичних процедур;
- Інтеграція з медичним обладнанням для автоматичного збору даних;
- захист від несанкціонованого доступу до медичної інформації.

Обмеження на застосування мінімальної конфігурації технічних засобів спричинений необхідністю економії ресурсів. У системі управління медичними даними лікарні, такі обмеження можуть включати мінімальні вимоги до обладнання, наприклад, обов'язкову підтримку певного рівня обробки даних та безпеки, а також достатній обсяг пам'яті для зберігання та обробки великого обсягу медичних інформаційних даних. Такі обмеження можуть враховувати можливість роботи системи на застарілих або менш продуктивних пристроях, що дозволяє медичним установам зменшити витрати на обслуговування і підтримку технічних систем.

Відомості про призначення програми

Система дозволяє забезпечити ефективного управління медичними даними, поліпшити якість медичних послуг та оптимзувати робочий процес.

Керівництво системного програміста

Керівництво системного програміста для системи управління медичними даними лікарні має надати докладну інформацію щодо налаштування та забезпечення функціонування програми відповідно до конкретних вимог та умов застосування. Керівництво включає основні етапи:

- Аналіз вимог: Потрібно почати з детального аналізу вимог щодо системи управління медичними даними лікарні. Це включає розуміння функціональних потреб, рівня безпеки, швидкодії та інших ключових аспектів;
- вибір технологій: Вибрати відповідні технології для розробки програми, які задовольнять вимоги проекту та гарантують безпеку медичних даних;
- розробка архітектури системи: Розробити архітектуру програмної системи, враховуючи модульність, масштабованість та інші аспекти, що сприятимуть ефективному управлінню медичними даними;
- реалізація та програмування: Розробити програмний код, відповідно до вимог та архітектури системи. Забезпечте належну документацію коду для полегшення майбутнього розвитку та підтримки;
- моніторинг та оновлення: Потрібно постійно моніторинг роботи системи та вчасно вносити оновлення та покращення для забезпечення безперебійного функціонування.

Опис контрольного прикладу

Контрольний приклад для системи управління медичними даними лікарні може включати наступні етапи та функціональні вимоги:

- Аутентифікація і авторизація: Перевірка ідентифікаційних даних користувачів перед наданням доступу до системи. Користувачі можуть бути лікарями, медичним персоналом, адміністраторами тощо. Різні рівні доступу можуть бути надані в залежності від ролі користувача;
- збереження медичних даних пацієнтів: Можливість введення, збереження та оновлення медичних даних пацієнтів, таких як анамнез, діагнози, лікування, результати обстежень тощо;

- конфіденційність та безпека даних: Забезпечення конфіденційності медичних даних пацієнтів шляхом шифрування, регулярного аудиту безпеки та використання механізмів контролю доступу;
- моніторинг і аналіз даних: Функції для моніторингу стану пацієнтів, відслідковування лікувального процесу та аналізу медичних даних для виявлення тенденцій та покращення якості надання медичних послуг. Також були розроблені тест-кейси, які зображено в додатку В.

3.4 Результати реалізації інформаційної підсистеми

Користувач інформаційної системи - пацієнт.

Коли користувач відкриває ІС, він потрапляє на сторінку головного меню (рис. 3.10). Натиснувши «Про нас», ви можете дізнатися інформацію про систему (рис. 3.11). Користувачі можуть входити в особисті облікові записи як пацієнт так і лікар.

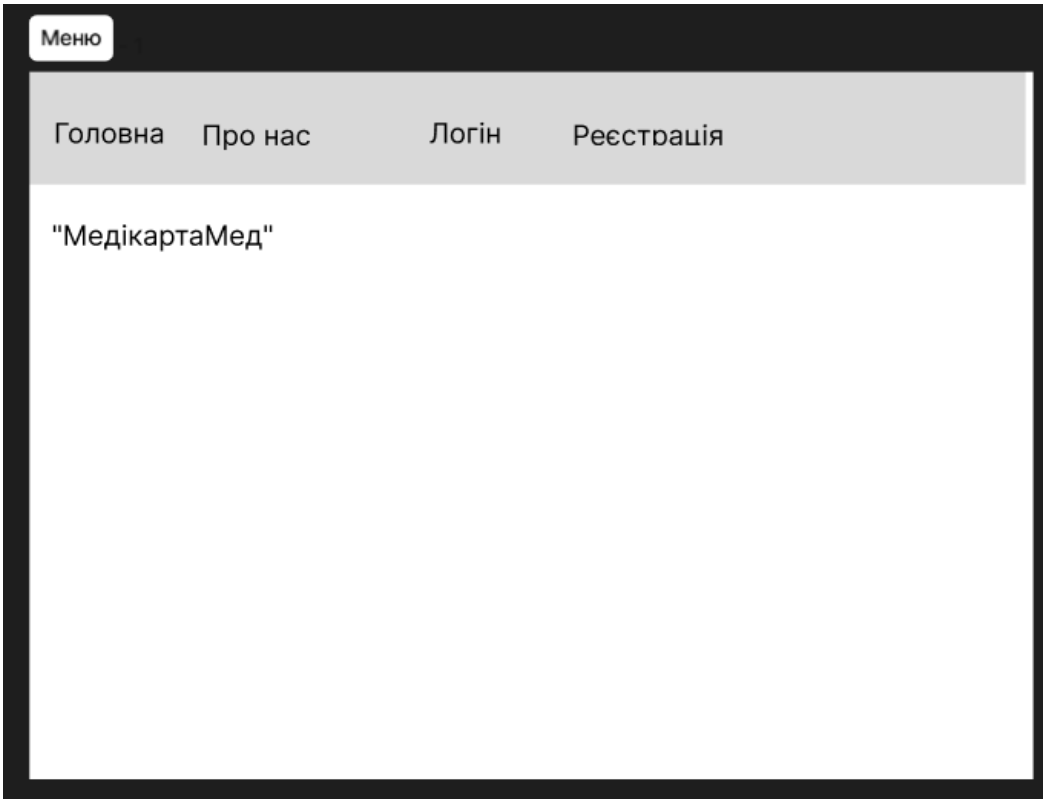


Рисунок 3.10 - Головне меню

Джерело: розроблено автором самостійно

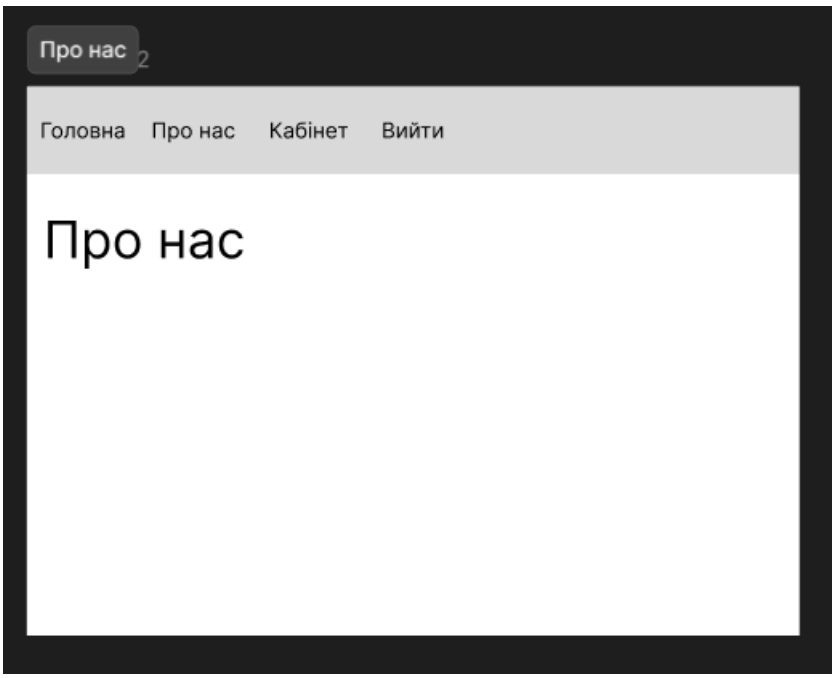


Рисунок 3.11 - Про нас

Джерело: розроблено автором самостійно

Для входу в особистий кабінет як пацієнт (рис. 3.12) потрібно ввести логін та пароль. Також новий користувач може реєструватися (рис. 3.13).

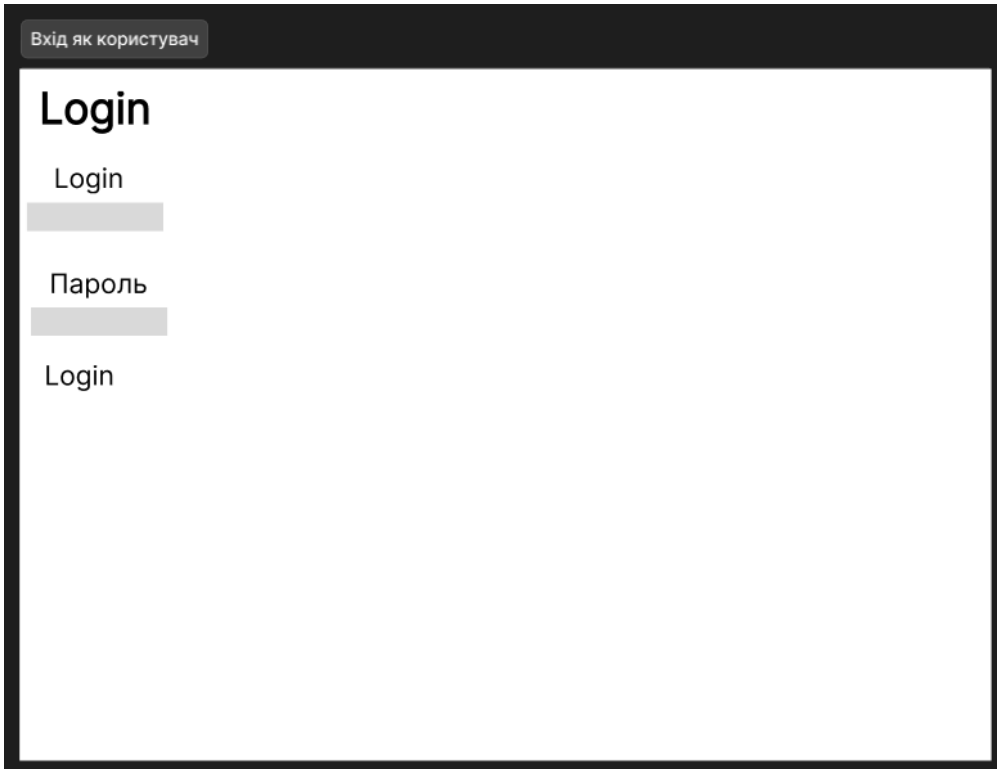


Рисунок. 3.12 - Вхід як пацієнт

Джерело: розроблено автором самостійно

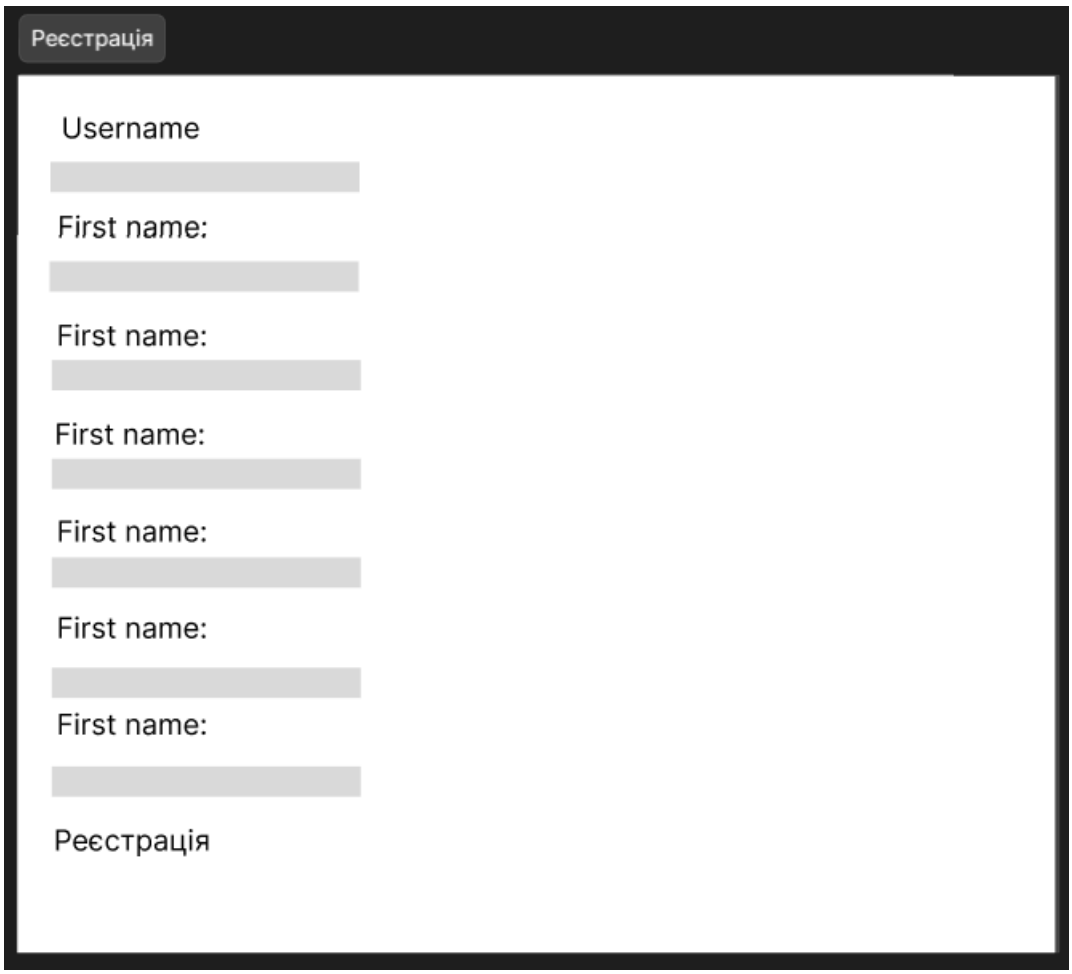


Рисунок. 3.13 – Реєстрація

Джерело: розроблено автором самостійно

В особистому кабінеті пацієнта можна подивитися свої дані, сімейного лікаря (рис. 3.14) натиснувши на "Мій лікар", направлення до лікарів (рис. 3.15), призначені ліки (рис.3.16) і результати аналізів (рис. 3.17)

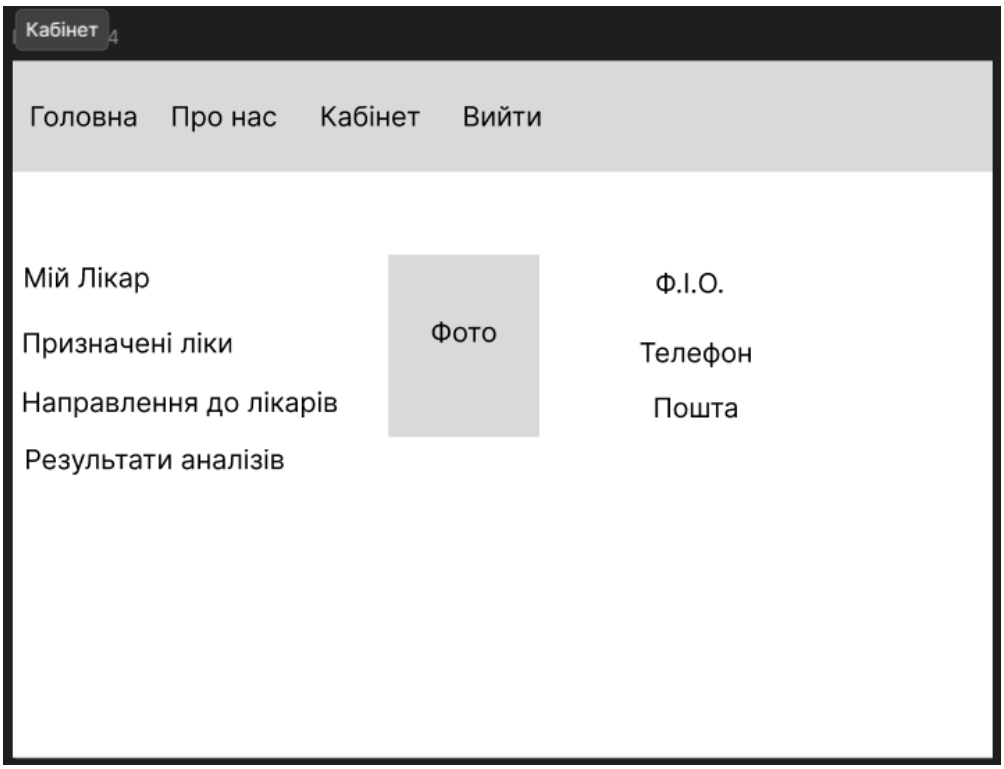


Рисунок. 3.14 - особистий кабінет

Джерело: розроблено автором самостійно

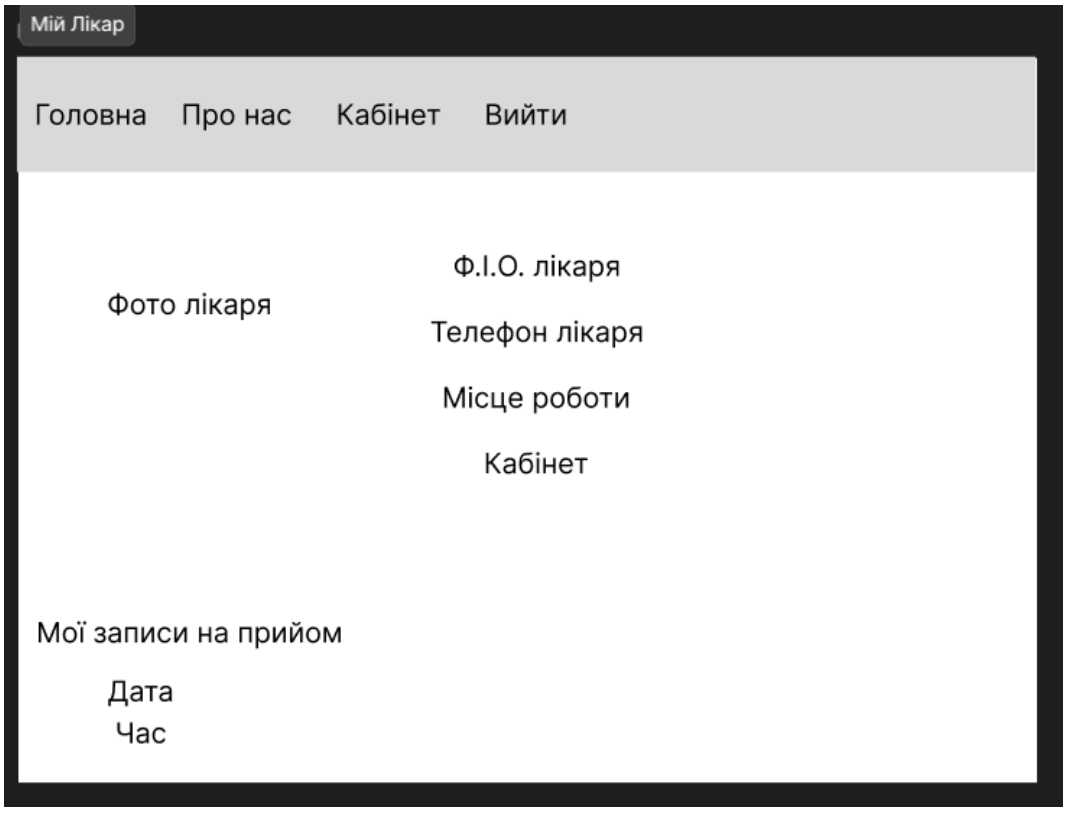


Рисунок. 3.15 - Мій лікар

Джерело: розроблено автором самостійно

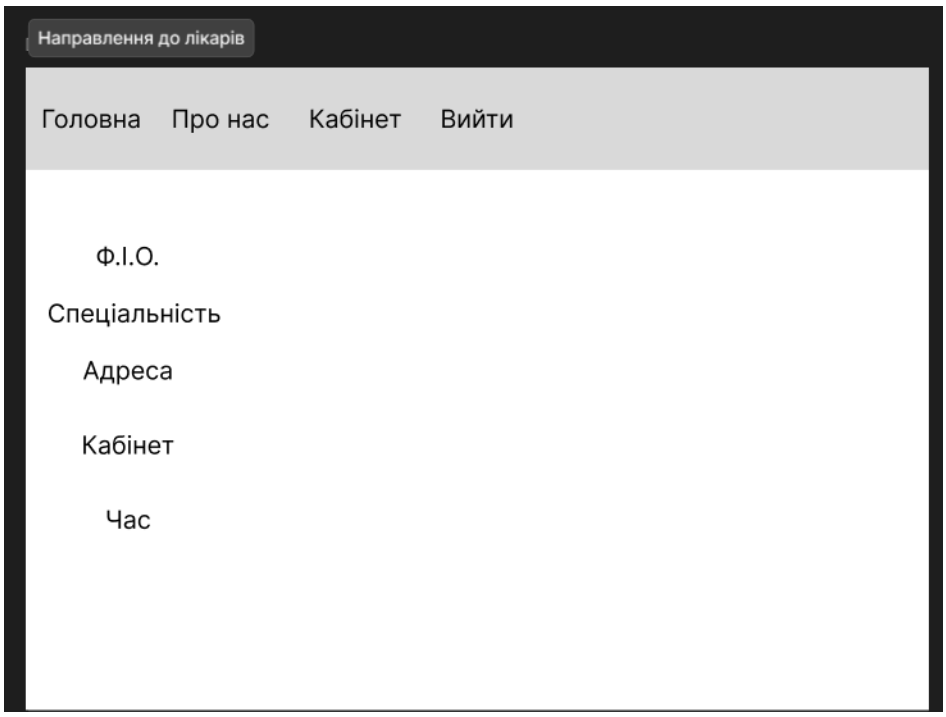


Рисунок. 3.16 - Направлення до лікарів

Джерело: розроблено автором самостійно

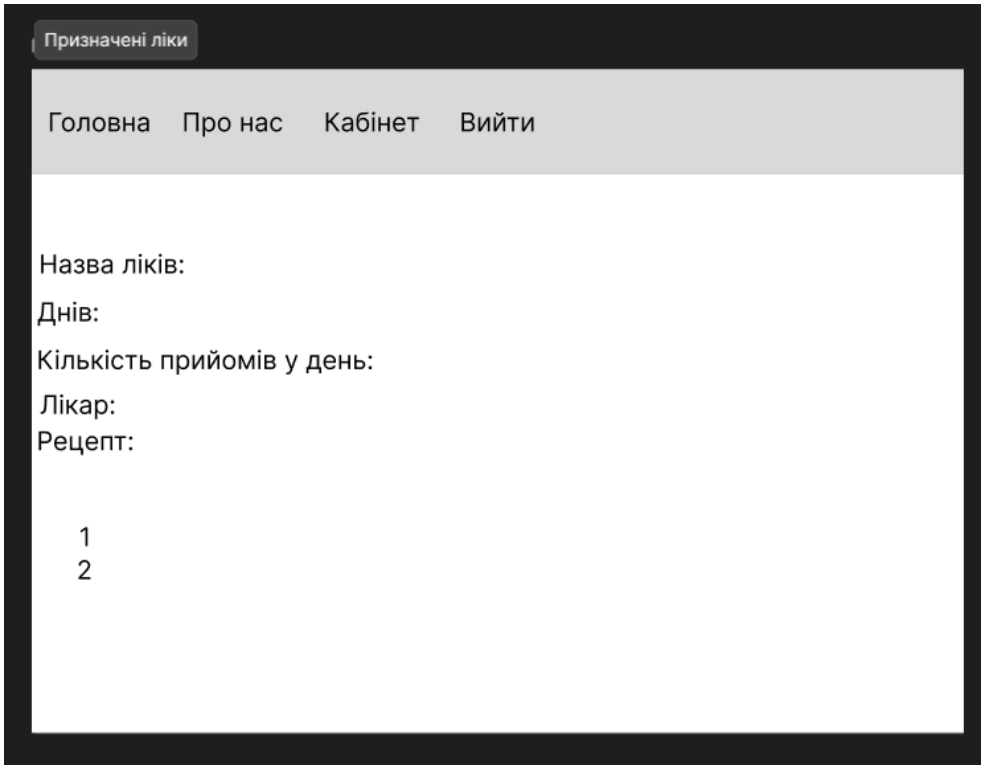


Рисунок. 3.17 - Призначені ліки

Джерело: розроблено автором самостійно

В особистому кабінеті лікаря можна подивитися свої дані (рис. 3.18), пацієнтів (рис. 3.19), направлення до лікарів (рис. 3.20), призначені ліки (рис.3.21) і направлення на аналізи (рис.3.22).

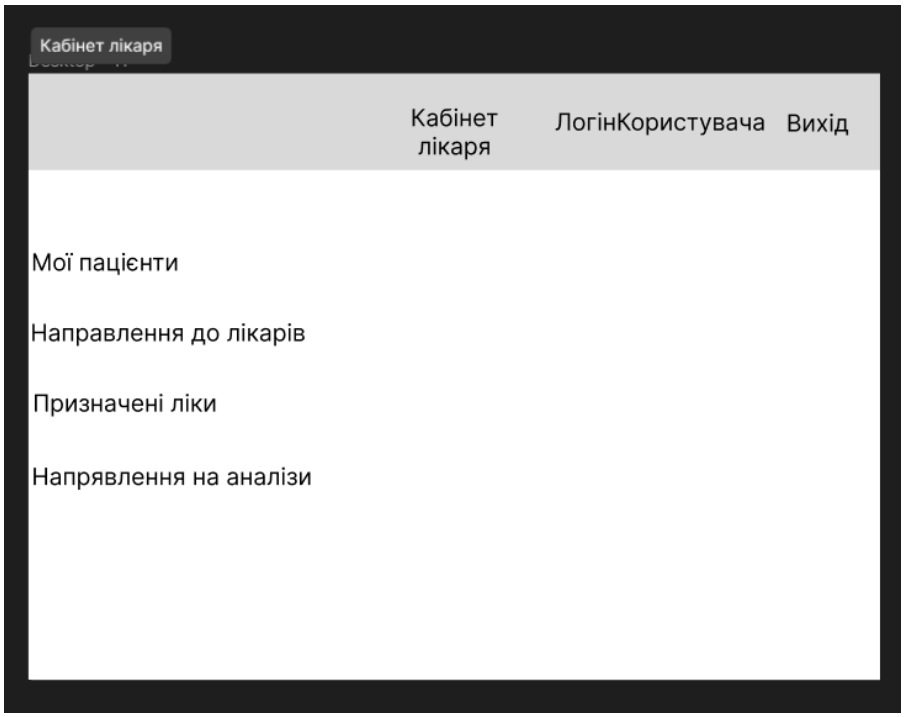


Рисунок 3.18 - Кабінет лікаря

Джерело: розроблено автором самостійно

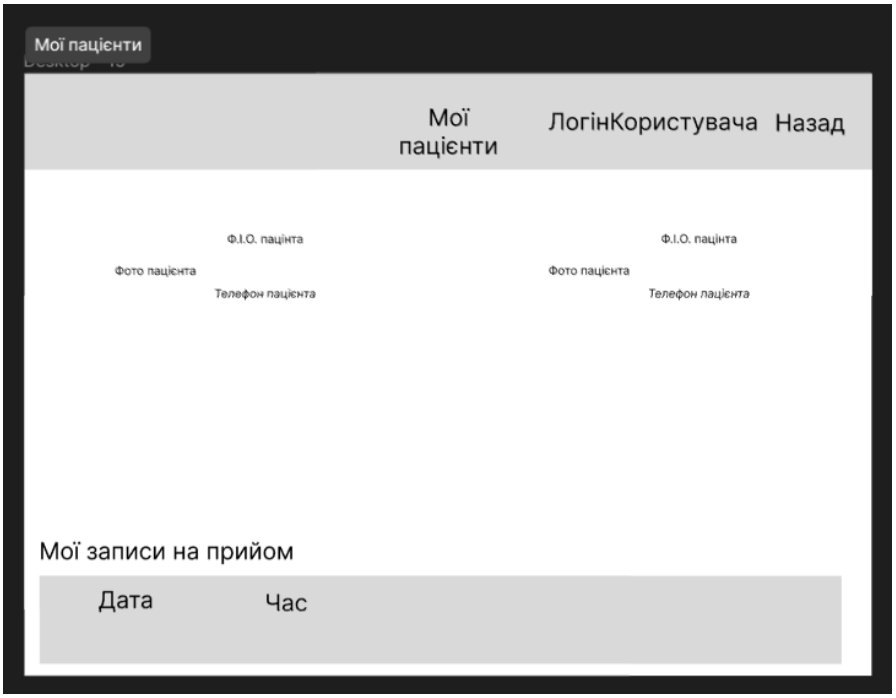


Рисунок 3.19 - Мої пацієнти

Джерело: розроблено автором самостійно

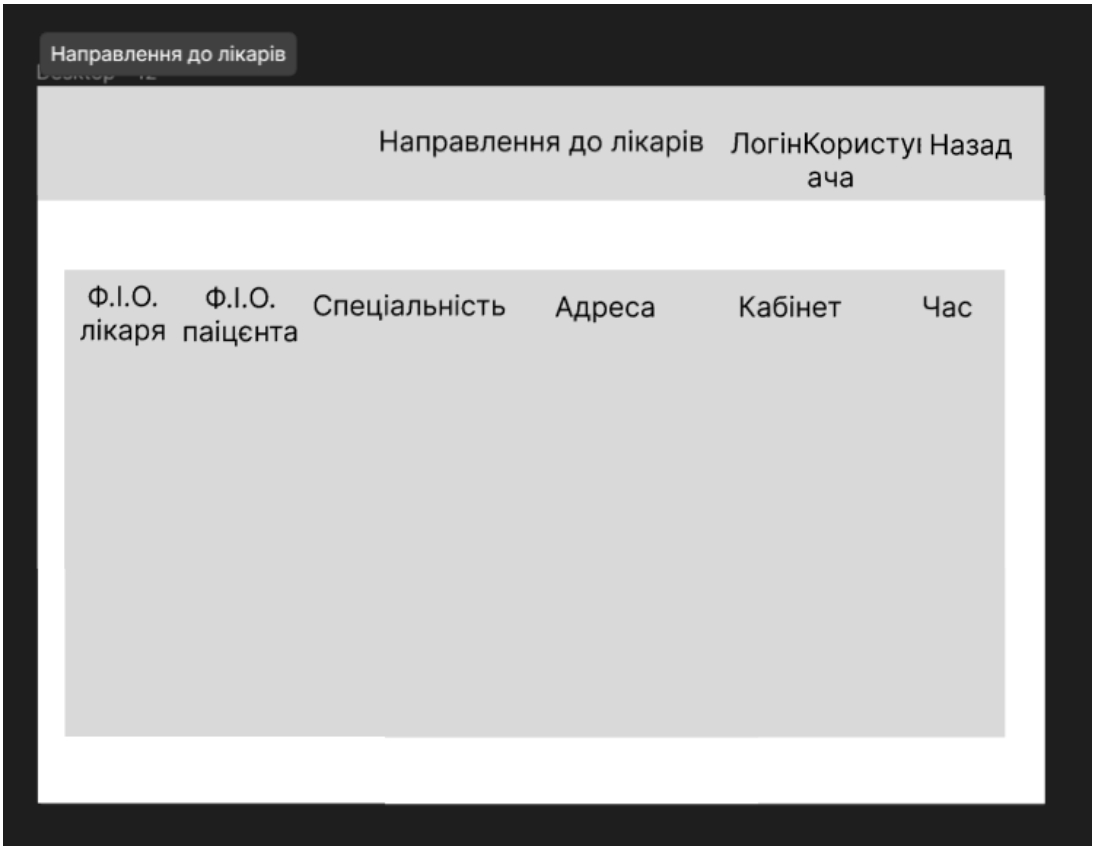


Рисунок - 3.20 - Направлення до лікарів

Джерело: розроблено автором самостійно

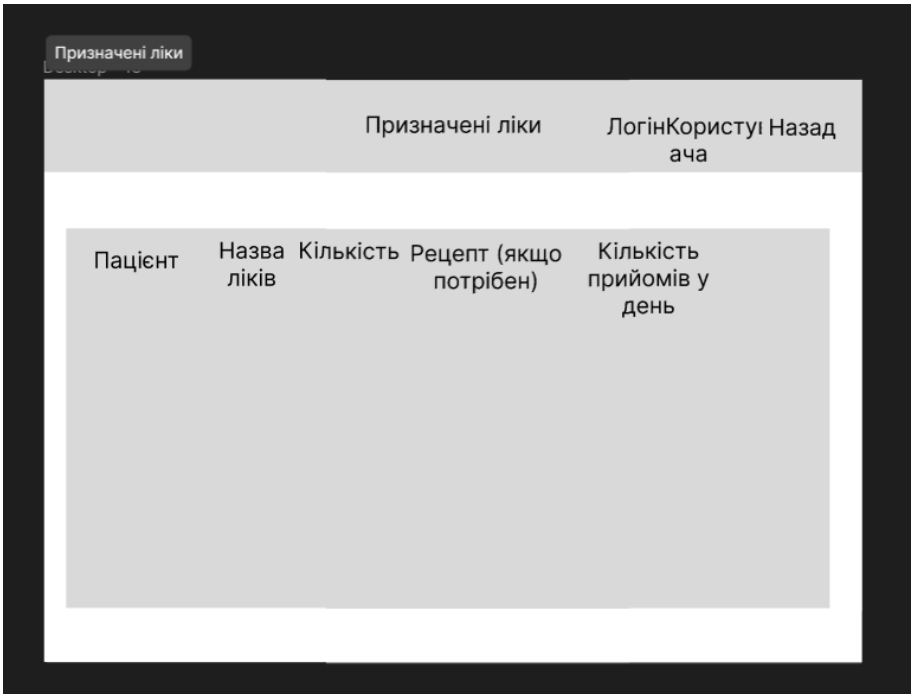


Рисунок 3.21 - Призначені ліки

Джерело: розроблено автором самостійно

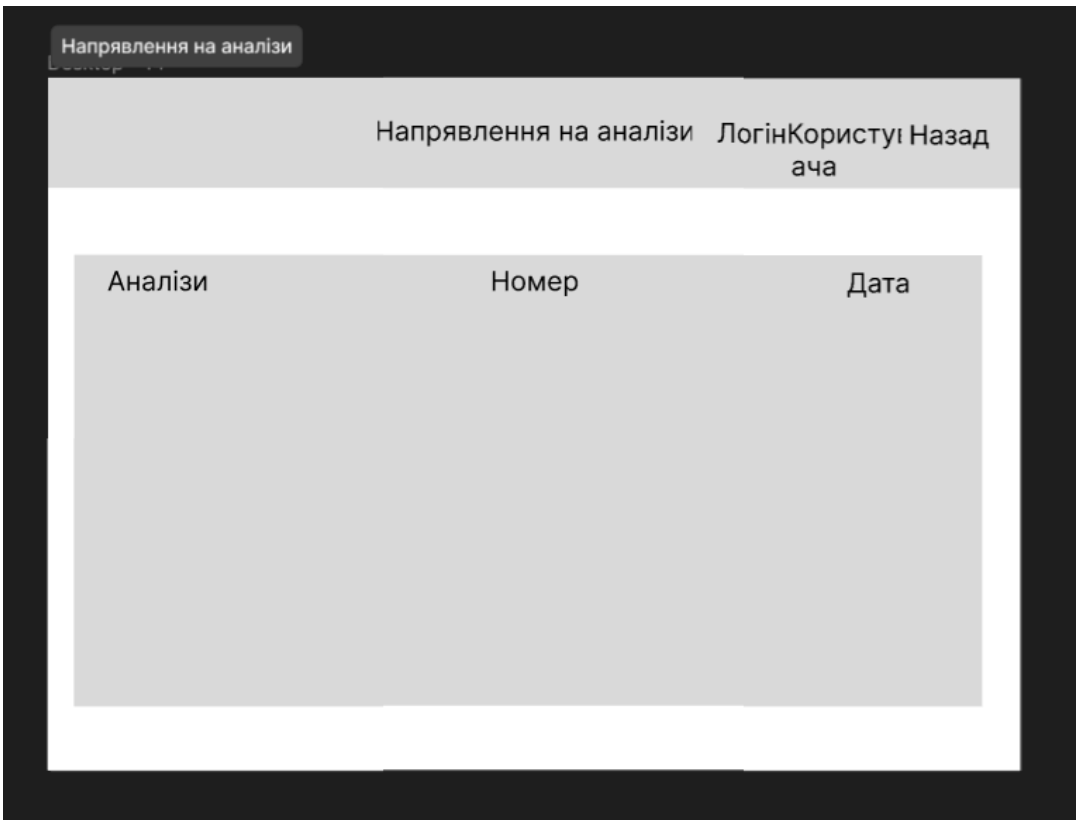


Рисунок 3.21 - Направлення на аналізи

Джерело: розроблено автором самостійно

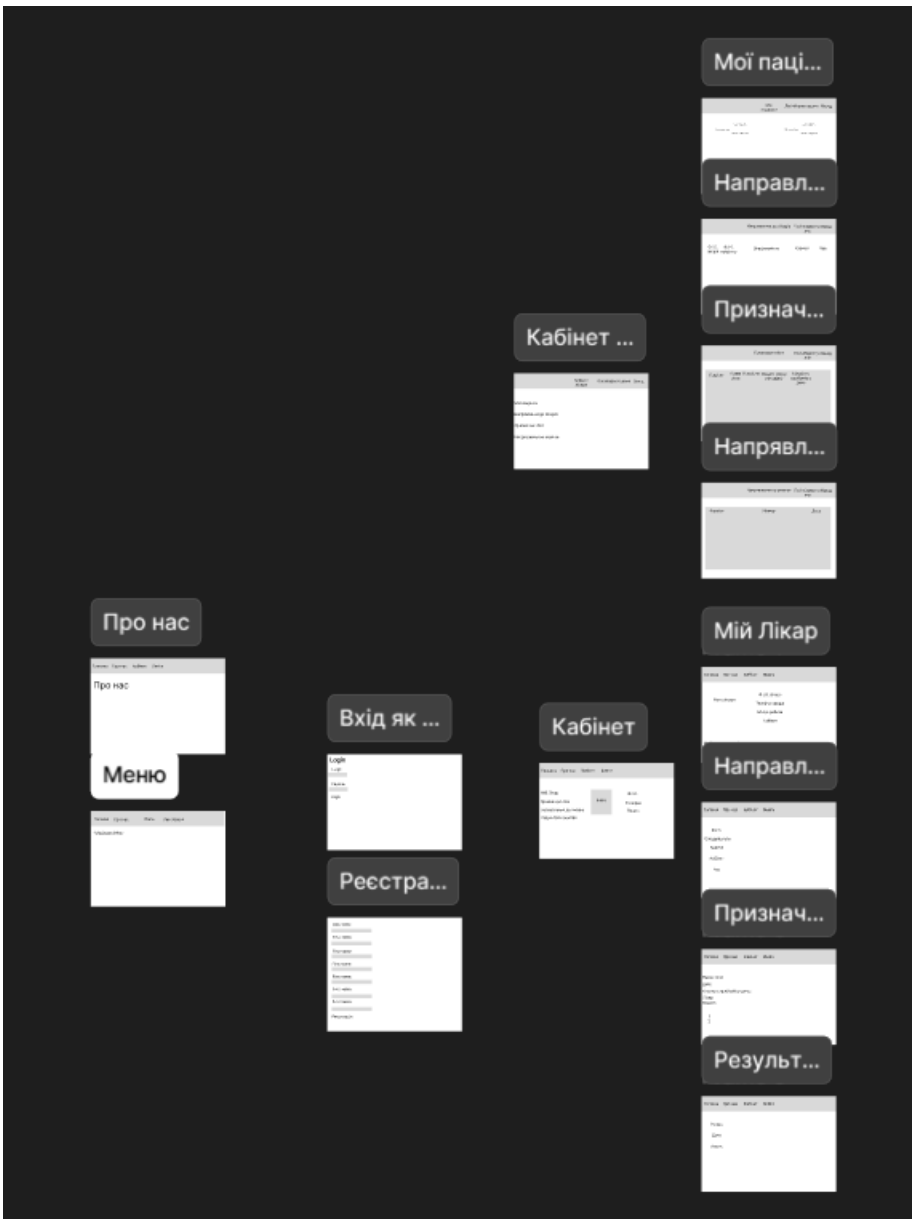


Рисунок 3.22 Направлення на аналізи

Джерело: розроблено автором самостійно

Реалізація системи на основі прототипу включає в себе функціональні сторінки: головну сторінку, форму «Про нас», форму «Реєстрації, форму» «Авторизації», особистий кабінет, форму «Призначені ліки».

Рисунок 3.23 – головна сторінка

Джерело: розроблено автором самостійно

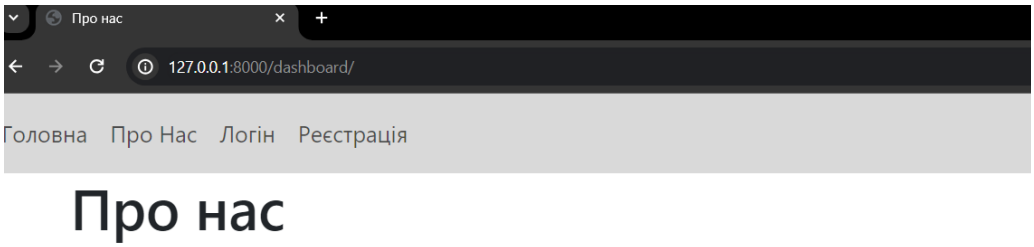


Рисунок 3.24 – сторінка «Про нас»

Джерело: розроблено автором самостійно

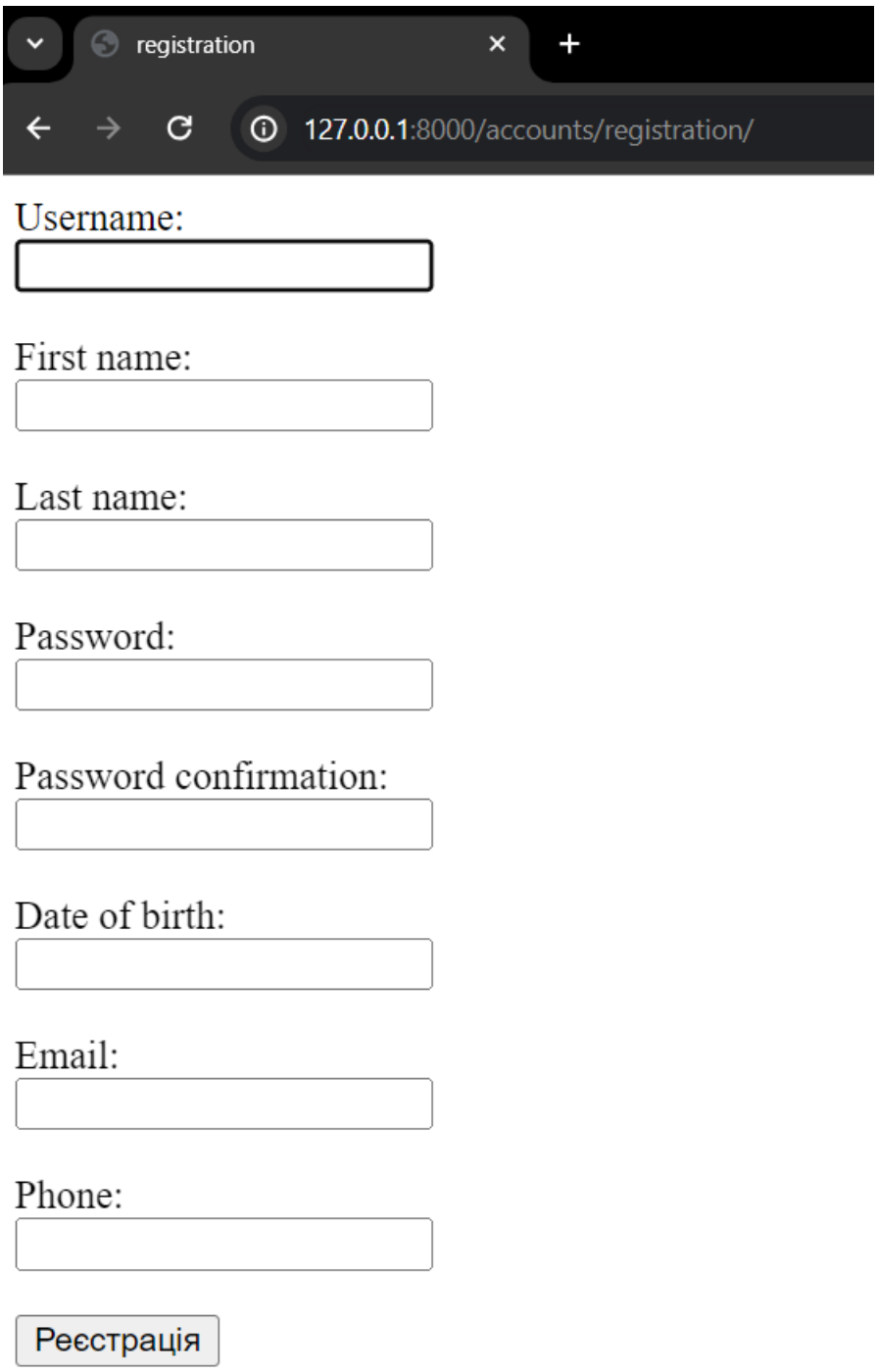


Рисунок 3.25 – Реєстрація

Джерело: розроблено автором самостійно

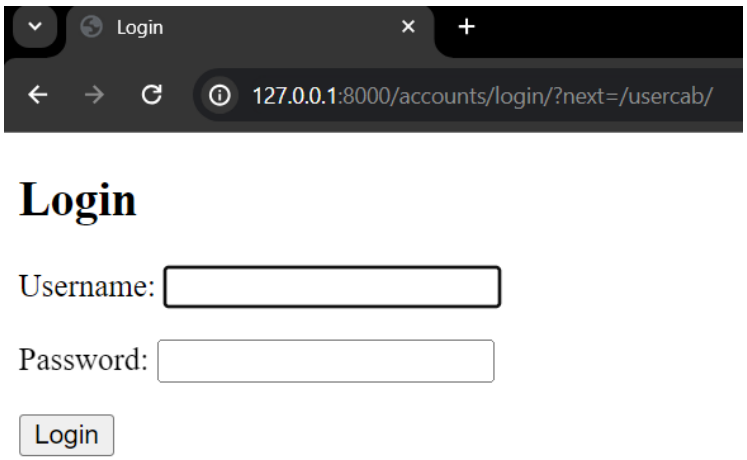


Рисунок 3.26 – вхід у систему

Джерело: розроблено автором самостійно

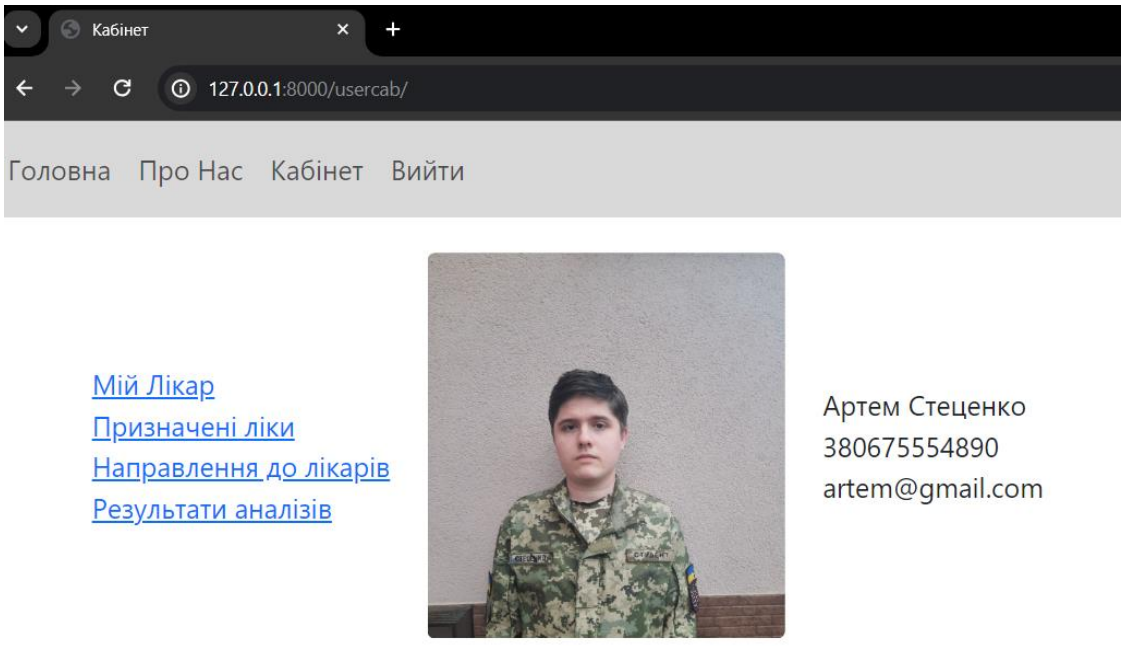


Рисунок 3.27 – особистий кабінет

Джерело: розроблено автором самостійно

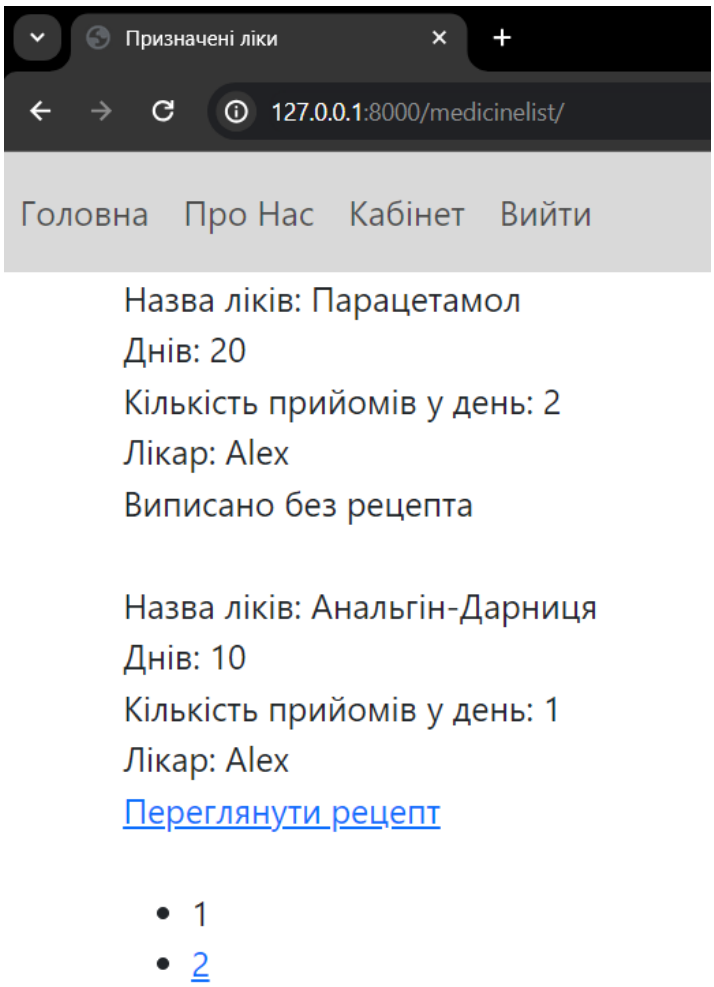


Рисунок 3.28 - форма «Призначені ліки»

Джерело: розроблено автором самостійно

В третьому розділі було описано загальну характеристику інформаційного забезпечення, організація збору і передавання первинної інформації, системи класифікації та кодування, проектування форм первинних документів, машинограм та відеокадрів, структуру інформаційних масивів, вибір СКБД, інфологічну модель бази даних, даталогічну модель бази даних, технічне забезпечення, схему мережі передачі даних, програмне забезпечення та результати реалізації інформаційної підсистеми.

ВИСНОВКИ

Ефективна інформаційна система управління медичними даними сприяє підвищенню якості медичних послуг, оптимізації внутрішніх процесів лікарні, зниженню операційних витрат та підвищенню задоволеності пацієнтів. Вона також дозволяє лікарям зосередитись на головному - наданні високоякісної медичної допомоги, залишаючи рутинні адміністративні задачі автоматизованим процесам.

У першому розділі бакалаврського проєкту було досліджено предметну область. Розглянуто процеси, які використовуються в інформаційній системі за допомогою методології ОРМ, зображено діаграми верхнього та нижнього рівня з детальним описом кожної з них. Крім того, проведено порівняння існуючих систем.

У другому розділі бакалаврського проєкту було розглянуто вимоги, діаграму Use-Case, текстові сценарії, діаграми послідовності, діаграму внутрішніх блоків, діаграму класів, діаграму керування, діаграму трасування та матрицю взаємозв’язків, описано вхідні та вихідні дані, математичні формули і алгоритм розв’язання задачі.

В третьому розділі було описано загальну характеристику інформаційного забезпечення, організація збору і передавання первинної інформації, системи класифікації та кодування, проектування форм первинних документів, машинограм та відеокадрів, структуру інформаційних масивів, вибір СКБД, інфологічну модель бази даних, даталогічну модель бази даних, технічне забезпечення, схему мережі передачі даних, програмне забезпечення та результати реалізації інформаційної підсистеми

Завдяки застосуванню передових технологій і методів проєктування, можна створити систему, яка забезпечує збереження та організацію медичних даних, швидкому доступу до них та оптимізації процесів управління.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Що таке ЕМС [Електронний ресурс] - https://medplatforma.com.ua/article/1971-elektronn-medichn-zapisi-yak-vesti (дата звернення 10.05.2024).

2. Що таке CRM [Електронний ресурс] - Режим доступу до ресурсу: https://nethunt.ua/blog/shcho-takie-crm-sistiema-povnii-ghid-po-viboru-crm-dlia-pochatkivtsiv/ (дата звернення 10.05.2024).

3. Методологія OPM [Електронний ресурс] https://study.com/academy/lesson/what-is-object-process-methodology.html#:~:text=The%20Object%20Process%20Methodology%20(OPM,completing%20a%20task%20or%20goal (дата звернення 10.05.2024).

4. Salesforce Health Cloud Електронний ресурс] - Режим доступу до ресурсу: ресурсу: https://www.salesforce.com/products/health-cloud/overview/ (дата звернення 28.04.2024).

5. Healthcare HubSpot [Електронний ресурс] - Режим доступу до ресурсу: https://www.hubdew.com/blog/hubspot-for-healthcare#:~:text=Unfortunately%2C%20traditional%20communication%20methods%20can,businesses%20connect%20with%20their%20clients. (дата звернення 28.11.2023).

6. Pipedrive [Електронний ресурс] - Режим доступу до ресурсу: https://www.pipedrive.com/ (дата звернення 28.04.2024).

7. Microsoft Dynamics 365 for Healthcare: [Електронний ресурс] - Режим доступу до ресурсу: https://www.microsoft.com/en-us/dynamics-365 (дата звернення 28.04.2024).

8. UML [Електронний ресурс] - https://visuresolutions.com/uk/довідник-mbse/що-таке-mbse/ (дата звернення 30.04.2024).

9. Що таке бізнес-вимоги? [Електронний ресурс] - https://visuresolutions.com/uk/вимоги-з-посібником-Word-Excel/найкращі-шаблони-документів-для-бізнес-вимог/#:~:text=Бізнес-вимоги%20-

%20це%20специфікації%2C,також%20будь-які%20технічні%20обмеження (дата звернення 30.04.2024).

10. Що таке use case [Електронний ресурс] - https://training.qatestlab.com/blog/technical-articles/what-is-a-use-case-and-what-are-they-for/ (дата звернення 30.04.2024).

11. SysML [Електронний ресурс] - https://sysml.org/sysml-faq/what-is-internal-block-diagram.html (дата звернення 30.04.2024).

12. Traceability Diagram [Електронний ресурс] - https://sparxsystems.com/enterprise_architect_user_guide/16.1/the_application_desktop/create_traceability_diagrams.html (дата звернення 30.04.2024).

13. Relationship Matrix [Електронний ресурс] - https://sparxsystems.com/enterprise_architect_user_guide/15.2/guidebooks/tools_ba_relationship_matrix.html (дата звернення 30.04.2024).

14. Загальні поняття класифікації та кодування [Електронний ресурс] - https://library.if.ua/book/100/6843.html (дата звернення 30.04.2024).

15. Методика проектування форм [Електронний ресурс] - https://elearning.sumdu.edu.ua/free_content/lectured:de1c9452f2a161439391120eef364dd8ce4d8e5e/20160217112601/281795/index.html#p3 (дата звернення 30.04.2024).

16. Зміст внутрішньо машинного інформаційного забезпечення [Електронний ресурс] - https://studfile.net/preview/9830218/page:16/ (дата звернення 30.04.2024).

17. Зовнішньо машинна інформація [Електронний ресурс] - https://pidru4niki.com/15980912/informatika/tehnologiyi_stvorennya_mashinnoyi_pozamashinnoyi_informatsiynoyi_bazi (дата звернення 30.04.2024)

18. Види інформації [Електронний ресурс] - https://pidru4niki.com/1957041139748/marketing/vidi_informatsiyi_marketingovi_informatsiyni_sistemi (дата звернення 25.05.2024)

19. Системи класифікації та кодування [Електронний ресурс] - https://library.if.ua/book/94/6495.html (дата звернення 25.05.2024)

20. Що таке буквено-цифрові символи? Значення буквено-цифрового коду та деякі поширені приклади алфавітно-цифрового коду [Електронний ресурс] - https://uk.milestoblog.com/what-are-alphanumeric-characters (дата звернення 25.05.2024)

21. Види інформаційних масивів [Електронний ресурс] - https://studfile.net/preview/7367629/page:26/ (дата звернення 10.05.2024)

22. Getting Started with MySQL [Електронний ресурс] - https://dev.mysql.com/doc/mysql-getting-started/en/ (дата звернення 10.05.2024)

23. Концептуальна модель бази даних [Електронний ресурс] - https://studopedia.net/14_37659_kontseptualna-model-bazi-danih.html (дата звернення 12.05.2024)

24. Концептуальна модель бази даних, Даталогічна модель бази даних - [Електронний ресурс] - https://subject-book.com/informatika/konceptualna-model-bazi-danix-datalogichna-model-bazi-danix-rozroblennya-programnogo-zastosuvannya-dlya-vedennya-repertuaru-kinoteatru.html (дата звернення 12.04.2024)

25. Системне програмне забезпечення [Електронний ресурс] - https://step.org.ua/konspekt/pz/tema1 (дата звернення 12.05.2024)

26. Enterprise Architect [Електронний ресурс] - https://sparxsystems.com/ (дата звернення 12.05.2024)

27. Draw.io - огляд сервіса [Електронний ресурс] - https://vchymo.com/application/Drawio (дата звернення 12.05.2024)

28. MySQL [Електронний ресурс] - https://vchymo.com/application/Drawio (дата звернення 12.05.2024)

29. Python [Електронний ресурс] - https://www.python.org/ (дата звернення 12.05.2024)

30. Django [Електронний ресурс] - https://www.djangoproject.com/ (дата звернення 08.06.2024)

31. Програмне забезпечення загального призначення [Електронний ресурс] -

https://yevshan.com.ua/info/004/content/content3.html#:~:text=ППП%20загального%20призначення-%20це%20універсальні,введення%20та%20редагування%20текстових%20даних (дата звернення 08.06.2024)

32. Method Oriented software [Електронний ресурс] - https://www.edx.org/learn/object-oriented-programming/the-georgia-institute-of-technology-introduction-to-object-oriented-programming-with-java-iii-exceptions-data-structures-recursion-and-guis?index=product&queryID=c1dfb72772929e04a758604983b626d0&position=1&results_level=first-level-results&term=Method+Oriented+software&objectID=course-348daf5a-eba8-4bf4-97c8-1034ab83b464&campaign=Introduction+to+Object-Oriented+Programming+with+Java+III%3A+Exceptions%2C+Data+Structures%2C+Recursion%2C+and+GUIs&source=edX&product_category=course&placement_url=https%3A%2F%2Fwww.edx.org%2Fsearch (дата звернення 08.06.2024)

33. Problem-oriented software [Електронний ресурс] - https://www.researchgate.net/publication/3189836_Problem_Oriented_Software_Engineering_Solving_the_Package_Router_Control_Problem (дата звернення 08.06.2024)

34. Програмне забезпечення для роботи глобальних мереж [Електронний ресурс] - http://www.tsatu.edu.ua/kn/course/prohramne-zabezpechennja-i-proektuvannja-lokalnyh-i-hlobalnyh-kompjuternyh-merezh/ (дата звернення 08.06.2024)

35. Програмне забезпечення для організації обчислювального процесу [Електронний ресурс] - https://www.researchgate.net/publication/3189836_Problem_Oriented_Software_Engineering_Solving_the_Package_Router_Control_Problem (дата звернення 08.06.2024)

ДОДАТКИ

Додаток А

Скрипт діаграми OPD верхнього рівня

Пацієнт is physical.
Пацієнт can be Не зареєстрований, Зареєстрований, В лікуванні, or Виписаний.
Не зареєстрований is initial.
Виписаний is final.
Лікар is environmental and physical.
Об'єкти consists of Медична карта, Медичне дослідження, Медичне призначення, and Документи.
ІС Проєктування інформаційної системи управління медичними даними у закладі для лікувального закладу requires Об'єкти.
ІС Проєктування інформаційної системи управління медичними даними у закладі для лікувального закладу affects Пацієнт and Лікар.

Скрипт процесу Системи

Пацієнт is physical.
Пацієнт can be Не зареєстрований, Зареєстрований, В лікуванні, or Виписаний.
Не зареєстрований is initial.
Виписаний is final.
Лікар is environmental and physical.
Об'єкти consists of Медична карта, Медичне дослідження, Медичне призначення, and Документи.
ІС Проєктування інформаційної системи управління медичними даними у закладі для лікувального закладу requires Об'єкти.
ІС Проєктування інформаційної системи управління медичними даними у закладі для лікувального закладу affects Пацієнт and Лікар.
ІС Проєктування інформаційної системи управління медичними даними у закладі для лікувального закладу zooms into Прийом, Направлення на аналізи, Реєстрація, Направлення до лікарів, Авторизація, Виписка, and Призначення лікування, as well as Кабінет Лікаря and Кабінет Пацієнта.

Прийом requires Кабінет Лікаря.
Прийом yields Кабінет Пацієнта.
Направлення на аналізи requires Кабінет Лікаря.
Направлення на аналізи yields Кабінет Пацієнта.
Реєстрація Авторизація.
Реєстрація consumes Не зареєстрований Пацієнт.
Реєстрація yields Пацієнт.
Направлення до лікарів requires Кабінет Лікаря.
Направлення до лікарів yields Кабінет Пацієнта.
Авторизація consumes Лікар and Пацієнт.
Авторизація yields Кабінет Лікаря and Кабінет Пацієнта.
Виписка requires Кабінет Лікаря.
Виписка yields Кабінет Пацієнта.
Призначення лікування requires Кабінет Лікаря.
Призначення лікування yields Кабінет Пацієнта.

Скрипт процесу «Авторизація»

Пацієнт is physical.
Пацієнт can be Не зареєстрований, Зареєстрований, В лікуванні, or Виписаний.
Не зареєстрований is initial.
Виписаний is final.
Пацієнт relates to Кабінет Лікаря.
Лікар is environmental and physical.
Авторизація requires Об'єкти.
Авторизація consumes Лікар.
Авторизація yields Кабінет Лікаря and Кабінет Пацієнта.
Авторизація zooms into Вхід and Перевірка даних.
Вхід Авторизація.
Вхід Авторизація.
Вхід consumes Лікар and Пацієнт.
Перевірка даних invokes Вхід.

Скрипт процесу «Направлення до лікарів»

Пацієнт is physical.
Пацієнт can be Не зареєстрований, Зареєстрований, В лікуванні, or Виписаний.
Не зареєстрований is initial.
Виписаний is final.
Пацієнт relates to Кабінет Пацієнта.
Лікар is environmental and physical.
Направлення до лікарів requires Об'єкти and Кабінет Лікаря.
Направлення до лікарів affects Пацієнт and Лікар.
Направлення до лікарів yields Кабінет Пацієнта.
Направлення до лікарів zooms into Консультація у лікаря, Направлення, and Аналізи.
Консультація у лікаря is environmental.
Консультація у лікаря Аналізи.
Консультація у лікаря requires Кабінет Лікаря and Об'єкти.
Консультація у лікаря consumes Лікар.
Направлення Направлення до лікарів.
Аналізи Направлення.

Скрипт процесу «Направлення на аналізи»

Пацієнт is physical.
Пацієнт can be Не зареєстрований, Зареєстрований, В лікуванні, or Виписаний.
Не зареєстрований is initial.
Виписаний is final.
Пацієнт relates to Кабінет Пацієнта.
Лікар is environmental and physical.
Направлення на аналізи requires Об'єкти and Кабінет Лікаря.
Направлення на аналізи affects Пацієнт and Лікар.
Направлення на аналізи yields Кабінет Пацієнта.
Направлення на аналізи zooms into Консультація у лікаря.

Консультація у лікаря is environmental.
Консультація у лікаря Направлення на аналізи.
Консультація у лікаря requires Кабінет Лікаря and Об'єкти.
Консультація у лікаря consumes Лікар.

Скрипт процесу «Призначення лікування»

Пацієнт is physical.
Пацієнт can be Не зареєстрований, Зареєстрований, В лікуванні, or Виписаний.
Не зареєстрований is initial.
Виписаний is final.
Пацієнт relates to Кабінет Пацієнта.
Лікар is environmental and physical.
Призначення лікування requires Об'єкти.
Призначення лікування consumes Лікар.
Призначення лікування yields Кабінет Пацієнта.
Призначення лікування zooms into Консультація у лікаря, Призначити ліки, Виписати рецепт, and Додаткові обстеження.
Консультація у лікаря is environmental.
Консультація у лікаря Додаткові обстеження.
Консультація у лікаря Призначити ліки.
Консультація у лікаря requires Об'єкти and Кабінет Лікаря.
Призначити ліки Виписати рецепт.
Призначити ліки Призначення лікування.
Виписати рецепт Призначення лікування.

Додаток Б

Специфікація бізнес-вимог для проєктування інформаційної системи управління медичними даними лікарні

Забезпечення конфіденційність даних Доступ до даних повинен бути доступний лише для авторизованих користувачів.	requirement	Mandatory	High	High
Зберігання медичних даних пацієнтів	requirement	Mandatory	Medium	High
Керування медичної документації Створення та редагування медичних карток. Сканування та завантаження документів.	requirement	Mandatory	Medium	High
Контроль версій медичних карток Відстеження змін	requirement	Proposed	Low	Low
Масштабованість Система повинна бути здатною масштабуватися для підтримки зростаючої кількості користувачів і даних.	requirement	Proposed	Low	Medium

Особистий кабінет Особистий кабінет пацієнтів і лікарів	requirement	Proposed	Medium	Medium
Продуктивність Система повинна бути здатною обробляти запити в режимі реального часу.	requirement	Mandatory	Medium	Medium
Управління ліками та рецептами Електронний рецепт. Контроль за рецептами	requirement	Mandatory	Medium	Medium
Управління реєстрацією пацієнтів Запис на прийом до лікаря	requirement	Mandatory	Medium	High

Додаток В

Код створення таблиць в mysql

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;

SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- Schema Hospital

-- Schema Hospital

CREATE SCHEMA IF NOT EXISTS `Hospital` DEFAULT CHARACTER SET utf8 ;

USE `Hospital` ;

-- Table `Hospital`.`user_data`

CREATE TABLE IF NOT EXISTS `Hospital`.`user_data` (

  `userID` INT NOT NULL,

  `date_of_birth` DATE NOT NULL,

  `email` VARCHAR(45) NOT NULL,

  `phone` VARCHAR(45) NOT NULL,

  `name` VARCHAR(45) NOT NULL,

  `Photo` BLOB NOT NULL,

  PRIMARY KEY (`userID`))

ENGINE = InnoDB;

-- Table `Hospital`.`Personal_doctor`

CREATE TABLE IF NOT EXISTS `Hospital`.`Personal_doctor` (

  `doctorID` INT NOT NULL,

  `date_of_birth` DATE NOT NULL,

  `email` VARCHAR(45) NOT NULL,

  `name` VARCHAR(45) NOT NULL,

  `phone` VARCHAR(45) NOT NULL,

  `photo` BLOB NOT NULL,

  CONSTRAINT `3`

    FOREIGN KEY (`doctorID`)

    REFERENCES `Hospital`.`user_data` (`userID`)

    ON DELETE CASCADE

    ON UPDATE CASCADE)

ENGINE = InnoDB;

-- Table `Hospital`.`w_recipes_ patient`
```

```
CREATE TABLE IF NOT EXISTS `Hospital`.`w_recipes_ patient` (
  `recipesID` INT NOT NULL,
  `name` VARCHAR(45) NOT NULL,
  `doctorID` INT NOT NULL,
  PRIMARY KEY (`recipesID`))
ENGINE = InnoDB;
-- Table `Hospital`.`w_direction_doctor`
CREATE TABLE IF NOT EXISTS `Hospital`.`w_direction_doctor` (
  `doctorID` INT NOT NULL,
  `cabinet` VARCHAR(45) NOT NULL,
  `date` DATE NOT NULL,
  `specialty` VARCHAR(45) NOT NULL,
  `userID` INT NOT NULL,
  PRIMARY KEY (`doctorID`),
  INDEX `5_idx` (`userID` ASC) VISIBLE,
  CONSTRAINT `5`
    FOREIGN KEY (`userID`)
    REFERENCES `Hospital`.`user_data` (`userID`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
-- Table `Hospital`.`medicine_patient`
CREATE TABLE IF NOT EXISTS `Hospital`.`medicine_patient` (
  `medicine_ID` INT NOT NULL,
  `userID` INT NOT NULL,
  `name` VARCHAR(45) NOT NULL,
  `number_of_days_of_admission` VARCHAR(45) NOT NULL,
  `quantity` VARCHAR(45) NOT NULL,
  `recipeID` INT NOT NULL,
  `doctorID` INT NOT NULL,
  INDEX `64_idx` (`recipeID` ASC) VISIBLE,
  PRIMARY KEY (`medicine_ID`),
  INDEX `77_idx` (`doctorID` ASC) VISIBLE,
  CONSTRAINT `2`
    FOREIGN KEY (`userID`)
    REFERENCES `Hospital`.`user_data` (`userID`)
    ON DELETE CASCADE
```

```
    ON UPDATE CASCADE,
  CONSTRAINT `64`
    FOREIGN KEY (`recipeID`)
    REFERENCES `Hospital`.`w_recipes_ patient` (`recipesID`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `77`
    FOREIGN KEY (`doctorID`)
    REFERENCES `Hospital`.`w_direction_doctor` (`doctorID`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
-- Table `Hospital`.`w_analyses_patient`
CREATE TABLE IF NOT EXISTS `Hospital`.`w_analyses_patient` (
  `AnalysisID` INT NOT NULL,
  `Date` DATE NOT NULL,
  `name_analysis` VARCHAR(45) NOT NULL,
  `doctorID` INT NOT NULL,
  PRIMARY KEY (`AnalysisID`),
  INDEX `9_idx` (`doctorID` ASC) VISIBLE,
  CONSTRAINT `9`
    FOREIGN KEY (`doctorID`)
    REFERENCES `Hospital`.`Personal_doctor` (`doctorID`)
   ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
-- Table `Hospital`.`doctor_data`
CREATE TABLE IF NOT EXISTS `Hospital`.`doctor_data` (
  `doctorID` INT GENERATED ALWAYS AS () VIRTUAL,
  `name` VARCHAR(45) NOT NULL,
  `date_of_birth` VARCHAR(45) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  `phone` VARCHAR(45) NOT NULL,
  `photo` BLOB NOT NULL,
  PRIMARY KEY (`doctorID`),
  CONSTRAINT `3`
    FOREIGN KEY (`doctorID`)
```

```
    REFERENCES `Hospital`.`w_direction_doctor` (`doctorID`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
-- Table `Hospital`.`analyses_patient`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `Hospital`.`analyses_patient` (
  `AnalysisID` INT NOT NULL,
  `Date` DATE NOT NULL,
  `name_analysis` VARCHAR(45) NOT NULL,
  `doctorID` INT NOT NULL,
  `userID` INT NOT NULL,
  PRIMARY KEY (`AnalysisID`),
  INDEX `5_idx` (`userID` ASC) VISIBLE,
  INDEX `8_idx` (`doctorID` ASC) VISIBLE,
  CONSTRAINT `5`
    FOREIGN KEY (`userID`)
    REFERENCES `Hospital`.`user_data` (`userID`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `8`
    FOREIGN KEY (`doctorID`)
    REFERENCES `Hospital`.`Personal_doctor` (`doctorID`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Додаток Д

Код програми

Views.py

```
from django.shortcuts import render, redirect
from django.contrib.auth.decorators import login_required
from django.views import View
from django.views.generic import ListView
from django.shortcuts import render
from django.http import Http404
from .models import MedicinePatient, WRecipesPatient, UserData
from django.contrib.auth.mixins import LoginRequiredMixin

def index(request):
    return render(request, 'likarnya/index.html')

def dashboard(request):
    return render(request, 'likarnya/dashboard.html')

class RecipeView(View):
    template_name = "likarnya/recipe.html"
    model = WRecipesPatient

    def get(self, request, pk):
        recipe = self.model.objects.filter(pk=pk)

        if recipe.count() == 0:
            raise Http404

        recipe = recipe.select_related().first()

        return render(request, self.template_name,
                      context={"recipe": recipe})

class MedicPatientListView(ListView,LoginRequiredMixin):
    login_url = "login"
```

```
    model = MedicinePatient
    template_name = "likarnya/medicinelist.html"
    paginate_by = 2
    def get_queryset(self):
        user_id = self.request.user.id
        return self.model.objects.filter(userID__user__id=user_id)


class UsercabView(View,LoginRequiredMixin):
    login_url = "login"
    template_name = "likarnya/usercab.html"
    def get(self, request):
        return render(request, self.template_name,{"userdata": UserData.objects.filter(user=request.user).first()})
```

urls.py

```
from django.urls import path, include
from . import views

urlpatterns = [
    path('', views.index, name='index'),
    path('dashboard/', views.dashboard, name='dashboard'),
    path('medicinelist/', views.MedicPatientListView.as_view(), name='medicinelist'),
    path('recipe/<int:pk>/', views.RecipeView.as_view(), name='recipe'),
    path('usercab/', views.UsercabView.as_view(), name='usercab'),
]
```

models.py

```
from django.db import models
from django.contrib.auth.models import User

class PersonalDoctor(models.Model):
    doctorID = models.AutoField(primary_key=True)
    date_of_birth = models.DateField()
    email = models.EmailField(max_length=45)
    name = models.CharField(max_length=45)
    phone = models.CharField(max_length=45)
    photo = models.BinaryField(blank=True, null=True)

class UserData(models.Model):
    userID = models.AutoField(primary_key=True)
    user = models.OneToOneField (User,on_delete=models.CASCADE, null=True)
```

```
    date_of_birth = models.DateField()
    email = models.EmailField(max_length=45)
    phone = models.CharField(max_length=45)
    photo = models.BinaryField(blank=True, null=True)
    def __str__(self) -> str:
        return f"{self.user.username}"

class AnalysesPatient(models.Model):
    analysisID = models.AutoField(primary_key=True)
    date = models.DateField()
    name_analysis = models.CharField(max_length=45)
    doctorID = models.ForeignKey(PersonalDoctor, on_delete=models.CASCADE)
    userID = models.ForeignKey(UserData, on_delete=models.CASCADE)

class DoctorData(models.Model):
    doctorID = models.AutoField(primary_key=True)
    name = models.CharField(max_length=45)
    date_of_birth = models.DateField()
    email = models.EmailField(max_length=45)
    phone = models.CharField(max_length=45)
    photo = models.BinaryField(blank=True, null=True)
    def __str__(self) -> str:
        return f"{self.name}"
class WDirectionDoctor(models.Model):
    doctorID = models.AutoField(primary_key=True)
    doctor=models.ForeignKey (DoctorData,on_delete=models.CASCADE, null=True)
    cabinet = models.CharField(max_length=45)
    date = models.DateField()
    specialty = models.CharField(max_length=45)
    userID = models.ForeignKey(UserData, on_delete=models.CASCADE)
    def __str__(self) -> str:
        return f"{self.doctor.name}"

class WAnalysesPatient(models.Model):
    analysisID = models.AutoField(primary_key=True)
    date = models.DateField()
    name_analysis = models.CharField(max_length=45)
```

```
    doctorID = models.ForeignKey(WDirectionDoctor, on_delete=models.CASCADE)


class WRecipesPatient(models.Model):
    recipesID = models.AutoField(primary_key=True)
    name = models.CharField(max_length=45)
    doctorID = models.ForeignKey(WDirectionDoctor, on_delete=models.CASCADE)


class MedicinePatient(models.Model):
    medicine_ID = models.AutoField(primary_key=True)
    userID = models.ForeignKey(UserData, on_delete=models.CASCADE)
    name = models.CharField(max_length=45)
    number_of_days_of_admission = models.CharField(max_length=255)
    quantity = models.CharField(max_length=45)
    recipesID = models.ForeignKey(WRecipesPatient, on_delete=models.CASCADE, null=True, blank=True)
    doctorID = models.ForeignKey(WDirectionDoctor, on_delete=models.CASCADE)
    def __str__(self) -> str:
        return f"{self.name}"
```

usercab.html

```
{% extends "common/base.html" %}

{% block title %}Кабінет{% endblock title %}
 {% load static %}
{% block content %}
<table>
    <tr>
        <td>
            <a href="#"><div>Мій Лікар</div></a>
            <a href="{% url 'medicinelist' %}"><div>Призначені ліки</div></a>
            <a href="#"><div>Направлення до лікарів</div></a>
            <a href="#"><div>Результати аналізів</div></a>
        </td>
        <td>
                <img style="width: 250px; display: block; border-radius: 25px; padding: 20px;" src="{% static 'images/photo.png' %}">
        </td>
        <td>
            <div>{{user.first_name}} {{user.last_name}}</div>
            <div>{{userdata.phone}}</div>
```

```
        <div>{{userdata.email}}</div>
      </td>
    </tr>
</table>

{% endblock content %}
```

recipe.html

```
{% extends "common/base.html" %}

{% block title %}Рецепт{% endblock title %}

{% block content %}
<div> {{recipe.name}} </div>
<div> {{recipe.doctorID.doctor.name}} </div>
{% endblock content %}
```

medicinelist.html

```
{% extends "common/base.html" %}
{% block title %}Призначені ліки{% endblock title %}
{% block content %}

{% for object in object_list %}
<div style="width: 100%;">
   <div style="display: inline-block; vertical-align: middle;">
      <div>
         <div>Назва ліків: {{object.name}}</div>
         <div>Днів: {{object.number_of_days_of_admission}}</div>
         <div>Кількість прийомів у день: {{object.quantity}}</div>
         <div>Лікар: {{object.doctorID.doctor.name}}</div>
         {% if object.recipesID %}
            <a href="{% url 'recipe' object.recipesID.pk %}"> <div> Переглянути рецепт</div> </a>
         {% else %}
            <div> Виписано без рецепта </div>
         {% endif %}

      </div>
   </div>
</div><br>          {% endfor %}
```

```
<div>
   <ul>
      {% for p in paginator.page_range %}

      {% if page_obj.number == p %}
      <li>
         <div>{{p}}</div>
      </li>
      {% else %}
      <li>
         <a href="?page={{p}}">
            <div>{{p}}</div>
         </a>
      </li>
      {% endif %}

      {% endfor %}
   </ul>
</div>
{% endblock content %}
```

index.hml

```
{% extends "common/base.html" %}

{% block title %}Головна сторінка{% endblock title %}

{% block content %}
<h1>"МедікартаМед"</h1>

{% endblock content %}
```

dashboard.hml

```
{% extends "common/base.html" %}
{% block title %}Про нас{% endblock title %}
{% block content %}
<h1>Про нас</h1>
{% endblock content %}
```

UNICHECK
by Turnitin

Ім'я користувача:
Інформаційних систем в економіці Шкуратовська Те...

ID перевірки:
1016352528

Дата перевірки:
12.06.2024 14:39:00 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
12.06.2024 15:00:32 EEST

ID користувача:
100005745

Назва документа: Стеценко_А_ІН_404

Кількість сторінок: 76 Кількість слів: 7484 Кількість символів: 63187 Розмір файлу: 3.28 MB ID файлу: 1016156247

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

13.8%
Схожість

Найбільша схожість: 4.29% з джерелом з Бібліотеки (ID файлу: 1016148653)

4.88% Джерела з Інтернету 263 ..... Сторінка 78

13.4% Джерела з Бібліотеки 461 ..... Сторінка 80

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи 10

Підозріле форматування 20 сторінок

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 ПРОЕКТ

на тему

ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ МЕДИЧНИМИ ДАНИМИ ЛІКАРНІ

здобувача Стеценка Артема Олеговича

Науковий керівник:
к.е.н., доцент
Помазун О.М.

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

Київ 2024