МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

(повне найменування назва факультету)

дискретного аналізу та інтелектуальних систем

(повна назва кафедри)

Дипломна робота

СТВОРЕННЯ ВЕБ-ЗАСТОСУНКУ ДЛЯ ЛІКАРЕНЬ "ПЛАНУВАННЯ ПРИЙОМУ"

Виконав: студент групи ПМІ-43

спеціальності

122 «Комп'ютерні науки»

(шифр і назва спеціальності)

Коляса Е.П.

(підпис) (прізвище та ініціали)

Керівник Коркуна Н.М.

(підпис) (прізвище та ініціали)

Рецензент

(підпис) (прізвище та ініціали)

2023

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет Прикладної математики та інформатики _____
Кафедра Дискретного аналізу та інтелектуальних систем _____
Спеціальність 122 «Комп'ютерні науки»_____
(шифр і назва)

«ЗАТВЕРДЖУЮ»
Завідувач кафедри _____Притула М.М._____

"31 "серпня 2022 року

З А В Д А Н Н Я

НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

_____Колясі Ерасту Павловичу_____
(прізвище, ім'я, по батькові)

1 .Тема роботи Створення веб-застосунку для лікарень «Планування прийому»_____
_____
_____

керівник роботи асистент кафедри дискретного аналізу та інтелектуальних систем Коркуна Наталія Михайлівна_____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені Вченою радою факультету від **" 13" вересня 2022 року № 15** _____

2. Строк подання студентом роботи **13.06.2023р.** _____

3. Вихідні дані до роботи Документація по мові програмування C#, документація по ASP .NET Core MVC, документація по Microsoft SQL Server, вбудовані бібліотеки ASP .NET Core_____
_____
_____

4. Зміст дипломної роботи (перелік питань, які потрібно розробити) Аналіз існуючих__ продуктів на ринку, проаналізувати і обґрунтувати вибір технології, сформулювати___ вимоги до програмного продукту та побудувати діаграми, розробка веб-додатку_____
_____
_____
_____

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Порівняльна таблиця доступних на ринку продуктів, use-case таблиці із_____
функціональними вимогами, діаграма варіантів використання, діаграма_____
послідовності, ER-діаграма_____
_____

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання **31 серпня 2022 р.**

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1.	Ознайомлення з темою роботи	30.09.2022	
2.	Проведення аналізу існуючих програмних продуктів	28.10.2022	
3.	Вибір засобів розробки програмного продукту	11.11.2022	
4.	Визначення основних вимог до програмного продукту	02.12.2022	
5.	Побудова діаграм	30.12.2022	
6.	Детальне вивчення технології ASP.NET Core MVC	03.02.2023	
7.	Розробка системи реєстрації та автентифікації користувачів	23.02.2023	
8.	Реалізація системи створення, редагування та видалення зустрічей	17.03.2023	
9.	Реалізація системи перегляду календаря конкретного лікаря	31.03.2023	
10.	Реалізація системи відображення підтверджених та непідтверджених зустрічей	14.04.2023	
11.	Вдосконалення інтерфейсу користувача	28.04.2023	
12.	Наповнення даними бази даних	05.05.2023	
13.	Оформлення дипломної роботи	26.05.2023	

Студент _____ Коляса Е.П.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Коркуна Н.М.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Тема дипломної роботи полягає в розробці веб-додатку для здійснення електронного запису до лікаря та аналізі наявних програмних продуктів на ринку. У роботі було обрано технологію ASP.NET Core з використанням MVC архітектури і детально розглянуто її особливості, призначення складових та можливості використання.

Для досягнення поставленої мети, було виконано наступні кроки. Початково було проведено аналіз наявних програмних продуктів, популярних у медичній галузі. Цей аналіз дозволив зрозуміти потреби та очікування користувачів, а також переваги та недоліки існуючих рішень.

На основі проведеного аналізу було обґрунтовано вибір технології ASP.NET Core з використанням MVC архітектури для розробки програмного продукту. Ця технологія має великий потенціал і надає зручні інструменти для створення веб-додатків, зокрема в медичній сфері.

Особлива увага приділялась архітектурі та функціональним вимогам до програмного продукту. Було побудовано діаграми взаємодії, ER-моделі та інші візуальні засоби для чіткого представлення структури та функціональності додатку. Це дозволило краще розуміти потреби користувачів і забезпечити оптимальну організацію компонентів та їх взаємодію.

У процесі розробки було звернуто особливу увагу на систему реєстрації та автентифікації користувачів. Були використані надійні методи шифрування та зберігання паролів, а також механізми перевірки прав доступу до ресурсів додатку.

Одним із головних досягнень було реалізовано систему створення, редагування та видалення зустрічей. Користувачам надавалась зручна інтерфейсна можливість взаємодії з системою, що дозволяла легко керувати їх зустрічами та проводити необхідні зміни.

Підтвердження та перегляд зустрічей було реалізовано з урахуванням особливостей медичної галузі. Користувачам надавалась можливість переглядати всі свої зустрічі, а також відображати підтверджені та непідтверджені зустрічі для забезпечення більшої зручності та контролю.

У результаті успішного виконання дипломної роботи був створений функціональний веб-додаток для електронного запису до лікаря, який відповідає вимогам та включає ключові функції, необхідні для зручного та ефективного використання медичною галуззю. Робота також включала аналіз наявних програмних продуктів та обґрунтування вибору технології розробки, що підкреслює важливість та актуальність результатів дослідження.

ЗМІСТ

ВСТУП

У сучасному світі, коли технології займають все більш значне місце в нашому житті, інтернет-сервіси стають дедалі популярнішими в різних галузях. Інтернет та мобільні пристрої стали невід'ємною частиною нашого повсякдення, а також значними інструментами для розвитку бізнесу та покращення якості надання послуг. Однією з галузей є медицина, де вже давно активно використовуються онлайн-сервіси для покращення якості та швидкості надання медичної допомоги.

Одним з таких інструментів є застосунки для онлайн запису до лікаря. Ці застосунки дозволяють пацієнтам швидко та зручно записатись на прийом до лікаря, обираючи зручний для них час та дату, переглядаючи наявність вільних місць, отримуючи підтвердження запису та нагадування про прийом. Такі застосунки також дають змогу отримувати необхідну інформацію про медичний заклад та його працівників.

Одним з головних інструментів розвитку медичної галузі може слугувати сучасна модернізована система для автоматизації процесів у медичному закладі. Система автоматизації лікарні - це система, яка орієнтована на полегшення управління лікарнею, є багатофункціональною і легкою для модернізації та підтримки. Ціль автоматизації - збільшення ефективності управління медичним закладом та персоналом, зменшення часу на обслуговування пацієнтів без втрати якості сервісу. Помітні беззаперечні переваги лікарень з автоматизованими процесами перед іншими подібними медичними закладами:

- зручність та доступність. Пацієнти можуть записатись до лікаря в будь-який час та з будь-якого місця з доступом до Інтернету, що значно зменшує витрати часу та зусиль на пошук лікаря та запис до нього;
- можливість швидкої реєстрації та оновлення даних пацієнта. За допомогою застосунку пацієнти можуть швидко зареєструватись та оновити свої

особисті дані, що дозволяє забезпечити актуальність медичної інформації та зменшує ризик помилок під час надання медичної допомоги;

- підвищення ефективності роботи медичного закладу. Застосунок дозволяє медичному закладу ефективніше планувати та керувати своєю роботою, зменшує кількість відмов від прийому та сприяє збільшенню навантаження на лікарів;
- забезпечення конфіденційності медичної інформації. Застосунок дозволяє зберігати медичну інформацію у захищеному електронному вигляді, що забезпечує конфіденційність та безпеку даних пацієнтів;
- зменшення черг та очікування. Застосунок дозволяє пацієнтам зарезервувати свій час прийому та уникнути зайвого очікування на реєстрації або у лікаря;
- можливість відстеження стану здоров'я. За допомогою застосунку пацієнти можуть відстежувати свій стан здоров'я, приймати ліки та здійснювати контроль над своїм здоров'ям;

Всі ці переваги використання застосунку для онлайн запису до лікаря показують, що такий застосунок може забезпечити пацієнтам зручність та швидкий доступ до медичних послуг, а медичним закладам - ефективнішу роботу та підвищення якості надання медичної допомоги. Тому розробка та впровадження такого застосунку є актуальним та важливим завданням у сучасному медичному середовищі.

Мета дипломної роботи: проаналізувати доступні на ринку програмні продукти, створення ефективного та зручного застосунку для внутрішнього використання лікарень, який спростить роботу адміністраторів та працівників у реєстратурі.

Вимоги до системи:

- робота системи з вікна браузера;

- розділені модулі системи для роботи користувачів з різних груп (лікарів, пацієнтів, адміністраторів);

- дружній для користувача інтерфейс;

- відображення інформації про статус запису до лікаря в реальному часі;

Для досягнення поставленої мети необхідно вирішити наступні задачі:

- аналіз існуючих рішень.
- розробка структури програмного продукту.
- аналіз та вибір програмних засобів для реалізації.
- розробка програми згідно до вимог.

РОЗДІЛ 1. ОГЛЯД ВІДОМИХ РІШЕНЬ НА РИНКУ

1.1 «Helsi»

«Helsi» - це медична інформаційна система, яка включає в себе веб-сайт та мобільний додаток. Забезпечує зручний та простий інтерфейс для пацієнтів, який дозволяє швидко знайти необхідну інформацію та забронювати час візиту до лікаря, що особливо важливо в сучасному світі, де швидкість та зручність є ключовими факторами. Крім того, додаток «Helsi» дозволяє лікарям зберігати та обробляти медичну інформацію про пацієнтів, що дозволяє збільшити ефективність надання медичної допомоги та зменшити ризик помилок.

Додаток побудований на платформі React Native, що дозволяє розробникам створювати нативні мобільні додатки для Android та iOS за допомогою JavaScript.

Додаток «Helsi» використовує базу даних PostgreSQL для зберігання медичної інформації про пацієнтів, лікарів, прийомів, аналізів та іншої інформації. Для забезпечення безпеки та конфіденційності даних, база даних захищена паролем та шифруванням.

Для забезпечення безпеки та надійності додатку, розробники використовують різні технології, такі як використання SSL-шифрування для передачі даних між сервером та клієнтом, аутентифікацію користувачів за допомогою токенів та ідентифікаторів, перевірку валідності даних на стороні сервера та клієнта.

Переваги:

– система хмарного сховища: «Helsi» використовує хмарне сховище для зберігання конфіденційної медичної інформації, що забезпечує високий рівень захисту даних та знижує ризик їхньої втрати.

– автоматизована система запису на прийом: «Helsi» має функціонал автоматичного запису на прийом до лікаря, що дозволяє уникнути зайвих черг та запобігти затримкам у наданні медичної допомоги.

– швидкий доступ до медичної інформації: «Helsi» забезпечує швидкий та зручний доступ до медичної інформації, що може допомогти лікарям приймати більш обґрунтовані рішення.

– інтеграція зі сторонніми системами: «Helsi» може бути легко інтегрований з іншими медичними системами, що полегшує обмін медичною інформацією та покращує якість медичної допомоги.

– зручний та інтуїтивно зрозумілий інтерфейс: «Helsi» має зручний та інтуїтивно зрозумілий інтерфейс, що полегшує використання додатку та забезпечує комфорт користувачів.

Недоліки:

– недостатній рівень захисту від кібератак: хоча «Helsi» використовує хмарне сховище для зберігання даних, додаток може бути вразливий до кібератак, що може призвести до втрати конфіденційної медичної інформації.

– залежність від Інтернет-підключення: «Helsi» потребує Інтернет-підключення для роботи, що може створювати проблеми у випадку відсутності Інтернету або його нестабільності.

– високі вимоги до обладнання: «Helsi» вимагає відносно потужного обладнання та оперативної пам'яті для оптимальної роботи, що може створювати проблеми для користувачів з менш потужними комп'ютерами або мобільними пристроями.

Узагальнюючи, додаток «Helsi» - це продуктивна та надійна медична інформаційна система, яка забезпечує зручну та швидку взаємодію між пацієнтами та лікарями. Він має безпечну та надійну архітектуру, використовує ряд технологій та інструментів для покращення користувацького досвіду та працює на різних мобільних пристроях та операційних системах.

1.2 «Medcard24»

«Medcard24» - це мобільний додаток, який побудований на основі різних технологій та мов програмування. Він дозволяє користувачам створювати та

зберігати електронні медичні картки. Додаток дозволяє вести детальний медичний журнал, який містить інформацію про відвідування лікаря, проведені процедури, результати аналізів та призначення лікарських засобів.

Крім того, «Medcard24» надає користувачам доступ до бази даних про лікарів, клініки та інші медичні заклади в їхньому регіоні, що дозволяє легко знайти та записатися на прийом до лікаря, переглянути відгуки пацієнтів про лікарів та більш детально дізнатися про послуги, які надаються в цих закладах.

«Medcard24» має функцію нагадування про прийоми до лікаря, призначення лікарських засобів та проведення процедур. Користувачі можуть встановлювати свої нагадування та налаштовувати їх під себе.

Застосунок допомагає зробити медичну інформацію більш доступною та зручною для зберігання та оновлення, а також дозволяє користувачам зосередитися на своєму здоров'ї та добробуті, забезпечуючи їм зручність та безпеку.

Для зберігання медичної інформації додаток використовує базу даних. Для забезпечення безпеки та конфіденційності медичних даних користувачів, «Medcard24» використовує шифрування даних на стороні сервера.

Для забезпечення зручного та ефективного використання додатку, використовуються різні технології та інструменти, такі як RESTful API для обміну даними між додатком та сервером, технологія нотифікацій для нагадування про прийоми лікаря та інші події, а також діалогові вікна для введення та редагування медичної інформації.

«Medcard24» також використовує різні сервіси, такі як Google Maps для знаходження лікарів та медичних закладів на мапі, а також дозволяє зберігати та керувати медичними документами в електронному вигляді.

Переваги:

– зручність використання - додаток має інтуїтивно зрозумілий інтерфейс, який дозволяє користувачам з легкістю зберігати та керувати своєю медичною інформацією.

– безпека даних - застосунок використовує шифрування даних на стороні сервера для забезпечення конфіденційності та захисту медичної інформації користувачів.

– можливість знаходження лікарів та медичних закладів в регіоні користувача за допомогою сервісів, таких як Google Maps.

– нотифікації - додаток дозволяє налаштовувати нагадування про прийоми лікаря та інші події, що допомагає користувачам пам'ятати про важливі дати та події.

Недоліки:

– не всі медичні заклади можуть бути представлені в додатку, що може обмежити можливості користувачів.

– інформація, що зберігається в додатку, може бути не повністю сумісна з системами електронної медичної документації, що використовуються у медичних закладах.

– застосунок може вимагати певних дозволів для доступу до деяких функцій, що може викликати занепокоєння з точки зору приватності користувачів.

– важливо мати належні знання та досвід у використанні додатку, щоб забезпечити правильне та безпечне зберігання медичної інформації.

Загалом, «Medcard24» - це корисний та зручний додаток, що допомагає зберігати та керувати медичною інформацією, а також знаходити лікарів та медичні заклади в регіоні користувача. Проте, користувачам слід мати на увазі, що додаток має свої обмеження та необхідно мати належні знання та досвід у використанні його функцій.

1.3 «Health24»

«Health24» - це додаток, призначений для моніторингу здоров'я та зберігання медичної інформації користувачів. Додаток пропонує широкий спектр функцій, включаючи пошук лікарів, запис на прийом, створення медичної картки, відстеження симптомів та хронологію захворювань. «Health24» є доступним для завантаження на пристрої з операційною системою iOS та Android.

Технічно, додаток побудований з використанням фреймворку React Native, що дозволяє розробникам створювати мобільні додатки для двох операційних систем одночасно. «Health24» використовує Firebase в якості хмарної платформи для зберігання та обробки даних, а також для аутентифікації користувачів та надання доступу до різних функцій додатку.

Додаток містить багато різноманітних функцій, що включають в себе пошук лікарів за різними параметрами, запис на прийом до обраного лікаря, створення медичної картки з основними даними про користувача та його медичну історію, відстеження симптомів та хронологію захворювань. Крім того, «Health24» дозволяє користувачам отримувати поради та рекомендації щодо здорового способу життя, таких як вправи та дієти.

Додаток має інтуїтивний та зрозумілий інтерфейс, що дозволяє користувачам легко користуватися різними функціями та зручно знаходити необхідну інформацію. «Health24» також забезпечує захист конфіденційної медичної інформації користувачів за допомогою шифрування та інших заходів безпеки.

Переваги:

- масштабованість та готовність до використання в різних країнах та медичних системах;
- легкий та інтуїтивно зрозумілий інтерфейс користувача;
- можливість зберігання медичної історії та легкий доступ до неї;
- зручний механізм запису на прийом та отримання медичних послуг;
- наявність медичної експертизи та консультацій онлайн;

Недоліки:

- наявність помилок та іноді повільна робота додатку;
- недостатня стабільність під час використання на деяких мобільних пристроях;
- обмежена можливість налаштування додатку під індивідуальні потреби користувача;
- відсутність деяких функцій, що можуть бути корисні користувачам, таких як моніторинг фізичної активності або статистика про здоров'я;

В цілому, «Health24» є високоякісним та корисним додатком для користувачів, які прагнуть піклуватися про своє здоров'я та зберігати медичну інформацію в одному місці. Додаток забезпечує легкий доступ до медичних послуг та зручний механізм запису на прийом, що дозволяє значно зекономити час та зусилля. Крім того, «Health24» допомагає користувачам зберігати свою медичну історію, яка може бути використана в подальшому при необхідності консультації з лікарем.

1.4 Порівняння відомих рішень

Таблиця 1.1 - Порівняння доступних на ринку продуктів

Назва / Властивість	**Helsi**	**Medcard24**	**Health24**
Демо-версія	ТАК	ТАК	ТАК
Технічна підтримка	ТАК	ТАК	ТАК
Працює у хмарі	ТАК	ТАК	ТАК
Мобільний додаток	ТАК	ТАК	ТАК
Інтуїтивно зрозумілий інтерфейс	ТАК	ТАК	ТАК
Може працювати без доступу до мережі Інтернет	НІ	ТАК	НІ
Система сумісна з сторонніми сервісами	ТАК	НІ	ТАК

Працює на різних операційних системах	ТАК	ТАК	ТАК
Безкоштовна система	НІ	ТАК	НІ

Проаналізувавши найбільш популярні додатки, які присутні на ринку можна дійти висновку, що в плані функціональності першість отримує додаток «Medcard24». Він відповідає більшості вимогам ринку, є стабільним, користується популярністю серед клієнтів та за рахунок своєї цінової політики доступний для медичних закладів із великим та малим фінансуванням.

РОЗДІЛ 2. ОГЛЯД ВИКОРИСТАНИХ ТЕХНОЛОГІЙ

2.1 ASP.NET Core with MVC

Microsoft .NET — програмна технологія, запропонована фірмою Microsoft як платформа для створення як звичайних програм, так і веб-застосунків. Багато в чому є продовженням ідей та принципів, покладених в технологію Java. Кожна бібліотека (збірка) в .NET має свідчення про свою версію, що дозволяє усунути можливі конфлікти між різними версіями збірок. .NET — крос-платформова технологія, поділяється на дві основні частини — середовище виконання (по суті віртуальна машина) та інструментарій розробки. Середовища розробки .NET-програм: Visual Studio .NET (C++, C#, J#), SharpDevelop, Borland Developer Studio (Delphi, C#) тощо. Середовище Eclipse має додаток для розробки .NET-програм. Застосовні програми також можна розроблювати в текстовому редакторі та використовувати консольний компілятор.[3]

ASP.NET Core - це фреймворк від Microsoft, який розроблений для побудови веб-додатків під операційну систему Windows, Linux та macOS. Основною метою цього фреймворку є спрощення процесу створення веб-додатків за допомогою

забезпечення платформи з відкритим кодом, яка має високу швидкість та продуктивність.[7]

ASP.NET Core забезпечує багато можливостей для розробки веб-додатків, таких як маршрутизація, валідація введення, автентифікація та авторизація, використання пакетів NuGet, робота з базами даних та багато іншого. В додаток до цього, ASP.NET Core підтримує багато різних форматів даних, таких як JSON та XML, що дозволяє легко обмінюватись даними з іншими додатками та службами.[7]

Крім того, ASP.NET Core забезпечує простоту та гнучкість в розробці веб-додатків. За допомогою структурованих директорій та добре організованої структури проекту, розробники можуть легко розбити свій код на логічні блоки та швидко знайти необхідний код для зміни або розширення.

ASP.NET Core with MVC є однією з найбільш популярних технологій веб-розробки, яка використовується для створення високоякісних, масштабованих та безпечних веб-додатків.

За допомогою ASP.NET Core with MVC, розробники можуть створювати веб-додатки з використанням шаблону проектування Model-View-Controller (MVC). Цей шаблон дозволяє розбити додаток на три частини: модель, яка відповідає за роботу з даними, представлення, яке відображає дані користувачеві, та контролер, який відповідає за обробку запитів користувача та зв'язок між моделлю та представленням.[1]

Загалом, ASP.NET Core with MVC є потужною технологією для розробки веб-додатків, що забезпечує продуктивність, масштабованість, безпеку та гнучкість. Вона підтримує різні операційні системи та бази даних, що дозволяє розробникам створювати веб-додатки, що задовольняють потреби різних клієнтів та користувачів.

2.1.1 Model (MVC)

У моделі Model-View-Controller (MVC) модель (Model) є компонентом, який відповідає за управління даними та логікою бізнес-процесів. Модель забезпечує зберігання та доступ до даних, а також реалізує бізнес-логіку, яка опрацьовує ці дані.[8]

Основна ідея моделі полягає в тому, щоб забезпечити гнучкість та використання даних незалежно від інших компонентів додатку. Це дозволяє використовувати одну і ту ж модель у різних контекстах, наприклад, на стороні клієнта та на сервері.

У ASP.NET Core with MVC, модель може бути представлена класом або групою класів, які представляють собою дані та логіку для роботи з цими даними. Модель може бути підключена до бази даних, файлової системи або інших джерел даних.[8]

Крім того, модель може мати різні типи зв'язків з іншими компонентами MVC. Наприклад, модель може взаємодіяти з представленням (View) шляхом відправки даних в шаблони відображення (view templates) для відображення на стороні клієнта. Модель також може взаємодіяти з контролером (Controller), приймаючи та оброблюючи дані від користувача.

Один з ключових аспектів моделі в ASP.NET Core with MVC - це розділення моделі від представлення та контролера. Це дозволяє розробникам зосередитися на роботі з даними та бізнес-логікою, не залежно від того, як вони будуть відображені користувачам або як їх введуть користувачі.

ASP.NET Core with MVC також підтримує валідацію даних на рівні моделі. Це дозволяє розробникам забезпечити правильність та безпеку даних, що передаються в додаток, перед тим, як вони будуть збережені в базі даних.

Загалом, модель є важливим компонентом веб-додатків, розроблених з використанням технології ASP.NET Core with MVC. Вона дозволяє розробникам

ефективно працювати з даними та бізнес-логікою, що допомагає забезпечити якість та безпеку веб-додатків.

2.1.2 View (MVC)

View є одним з ключових компонентів шаблону проектування Model-View-Controller (MVC) в технології ASP.NET Core. У контексті MVC, View відповідає за відображення даних користувачу та їх взаємодію з ним. Вона відображає дані, які зберігаються в моделі, та дозволяє користувачу взаємодіяти з цими даними за допомогою різних елементів вводу, таких як кнопки, поля вводу та інші.[8]

View є одним з найбільш важливих компонентів у MVC, оскільки вона забезпечує інтерфейс, через який користувач може взаємодіяти з додатком. ASP.NET Core with MVC забезпечує підтримку різноманітних форматів відображення, таких як HTML, XML, JSON, PDF та інші. View може бути розроблена з використанням шаблонних механізмів, таких як Razor, який дозволяє відокремлювати логіку відображення від бізнес-логіки.[8]

Один з ключових аспектів View в ASP.NET Core with MVC - це розділення View від контролера та моделі. Це дозволяє розробникам зосередитися на відображенні даних, не залежно від того, як ці дані були отримані.

View не повинна містити жодного коду, що стосується бізнес-логіки або моделі. Вона лише відображає дані, які передаються з контролера та моделі. Це робить View дуже простим та легким для розуміння та підтримки.

ASP.NET Core with MVC також забезпечує підтримку механізму роутингу, який дозволяє встановлювати зв'язок між URL-адресою та відповідною View. Це дозволяє розробникам створювати логічну структуру веб-додатка та забезпечувати легкість навігації для користувачів.

Крім того, ASP.NET Core with MVC дозволяє розробникам використовувати часткові View, що дає змогу використовувати повторно фрагменти коду та спрощувати процес розробки та підтримки веб-додатків.

Загалом, View є важливим компонентом веб-додатків, розроблених з використанням технології ASP.NET Core with MVC. Вона дозволяє розробникам ефективно відображати дані та інтерфейс користувача, що допомагає забезпечити якість та зручність веб-додатків.

2.1.3 Controller (MVC)

Контролер (Controller) - це один з компонентів архітектури Model-View-Controller (MVC). Це центральна точка для обробки даних та взаємодії з моделлю (Model) та представленням (View). У контексті MVC, Controller відповідає за обробку запитів від користувачів, звернених до сервера.[8]

Контролер отримує вхідні дані від користувача та інших джерел, обробляє їх та відправляє відповідні команди моделі і представленню для здійснення необхідних дій. Контролер також відповідає за валідацію вхідних даних та управління станом додатку.[8]

ASP.NET Core with MVC забезпечує підтримку створення контролерів з різноманітними типами даних, такими як рядки, числа, об'єкти JSON та інші. Контролер може бути розроблений з використанням різних механізмів, таких як атрибути, які дозволяють задати додаткові налаштування для контролера та його методів.[8]

Контролер може бути реалізований в будь-якій мові програмування та зазвичай містить методи для обробки різних запитів, таких як GET, POST, PUT, DELETE та інших.

Контролер також забезпечує підтримку механізму фільтрації, який дозволяє виконувати спільну логіку для багатьох контролерів. Фільтри можуть бути використані для встановлення прав доступу, кешування даних та інших завдань.[8]

Загалом, контролер є важливим компонентом веб-додатків, розроблених з використанням технології ASP.NET Core with MVC. Він дозволяє розробникам ефективно обробляти запити від користувачів та забезпечувати якість та зручність.

2.2 Microsoft SQL Server

Microsoft SQL Server - це система управління базами даних (СУБД) від корпорації Microsoft. Вона є однією з найпопулярніших реляційних СУБД на ринку та широко використовується для зберігання та управління даними в багатьох організаціях та компаніях.[9]

Основні функції Microsoft SQL Server:

- Зберігання та організація даних. Microsoft SQL Server забезпечує зручний та ефективний спосіб зберігання та організації даних в таблицях, що включають стовпці та рядки з даними.
- Управління даними. Microsoft SQL Server дозволяє виконувати широкий спектр операцій управління даними, таких як додавання, видалення, редагування та пошук даних.
- Захист даних. Microsoft SQL Server надає високий рівень захисту даних завдяки механізмам аутентифікації, авторизації та шифрування даних.
- Робота з даними в режимі реального часу. Microsoft SQL Server може працювати з даними в режимі реального часу, що дозволяє швидко та ефективно збирати та аналізувати дані в режимі реального часу.
- Робота з розподіленими системами. Microsoft SQL Server може працювати з розподіленими системами та включає механізми реплікації, що дозволяє забезпечити доступ до даних в різних місцях.
- Безперервність роботи. Microsoft SQL Server має механізми, що дозволяють забезпечувати безперервність роботи в разі відмови системи або інших аварійних ситуацій.

Загалом, Microsoft SQL Server є потужною та надійною СУБД, що має широкий функціонал та може задовольнити потреби різних бізнес-секторів. Вона підтримує багато мов програмування та інструментів розробки, таких як T-SQL, .NET Framework, Entity Framework, Integration Services та багато інших.

Одним з ключових переваг Microsoft SQL Server є його масштабованість. Система може працювати з великим обсягом даних та забезпечувати швидкий доступ до них навіть при високому навантаженні.

Крім того, Microsoft SQL Server є дуже гнучкою та може бути налаштована під потреби конкретної організації або проекту. Система підтримує різні рівні доступу до даних та механізми забезпечення їх цілісності та конфіденційності.

Незважаючи на це, Microsoft SQL Server не є бездоганною СУБД і має свої недоліки. Одним з них є висока вартість ліцензування, яка може бути проблемою для менших компаній або проектів з обмеженим бюджетом. Крім того, Microsoft SQL Server не є платформонезалежною СУБД та працює тільки на ОС Windows.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ

3.1 Ролі користувачів

Проаналізувавши складову медичних закладів можна виділити наступні ролі користувачів в системі: адміністратор, лікар, пацієнт, гість.

Адміністратор – спеціально навчена людина, яка слідкує за системою, може наділяти ролями різних користувачів, відповідає за стабільність системи. Може переглядати усіх зареєстрованих у системі лікарів та пацієнтів. Адміністратор може створити обліковий запис іншого адміністратора. При зверненні пацієнта створює зустріч із лікарем. Може переглядати календар зустрічей усіх лікарів. Також може скасовувати та редагувати зустріч в календарі лікаря та пацієнта.

Лікар - фахівець з повною вищою медичною освітою, що в установленому законом порядку постійно займається підтримкою або відновленням людського здоров'я, через запобігання (профілактика), розпізнавання (діагностика) та лікування захворювань і травм. В системі лікар може переглядати календар усіх своїх зустрічей із пацієнтами. Перегляд зустрічей можна здійснювати за

конкретним днем, тижнем або місяцем. Також, лікар може підтверджувати та скасовувати зустріч із свого календаря. Лікар може переглядати конкретну зустріч і всі її деталі, а також вносити дані про діагноз після проведеної зустрічі.

Пацієнт - людина, яка отримує медичну допомогу. Пацієнт звертається до адміністратора лікарні і записується на зустріч із лікарем. В системі пацієнт може переглядати календар своїх зустрічей в конкретний день, тиждень або місяць а також побачити, чи підтвердив зустріч лікар чи ні. Також пацієнт може переглянути конкретну зустріч із лікарем та всі її деталі.

Гість – це не залогований або не зареєстрований користувач у системі. Гість може зареєструватись або увійти у систему через свій обліковий запис.

3.2 Use case

Use case - у розробці програмного забезпечення та системному проектуванні це опис поведінки системи, як вона відповідає на зовнішні запити. Іншими словами, різновид використання описує, «хто» і «що» може зробити з розглянутою системою. Методика різновидів використання застосовується для виявлення вимог до поведінки системи, відомих також як функціональні вимоги.

У більшості випадків Use Case описує, що робить система, а не як. Власне, цього правила і варто дотримуватися, створюючи такі сценарії.

Use-case потрібні:

- Для замовників. Кожен use-case несе кінцеву бізнес-цінність, зрозумілу замовнику. То ж навіть технічно не підкована людина може переконатися у реалізації того чи іншого use-case у системі. Наявність готового сценарію дає можливість замовнику своєчасно підтвердити старт подальшої роботи тестувальників та команди розробників.
- Для розробників. Зручність полягає у розлогому описі основного й альтернативного потоку подій. Уся інформація зображена максимально структуровано та зрозуміло з урахуванням кінцевого

результату. Use-case ідеально підходять у ситуаціях складних сценаріїв.

– Для тестувальників. Це чудова база для формування тестових сценаріїв – test case. Use-case, за замовчуванням, є тестованими вимогами із зазначеною метою і шляхом її досягнення.

Варіанти застосування use-case у процесі розробки залежать від використовуваної методології. В одних методологіях все, що потрібно, – це короткий огляд use-case. А в інших – сценарії використання ускладнюються і змінюються вже у процесі розробки.

У деяких методологіях вони можуть початися як короткі бізнес-сценарії, а вже згодом розвинутися у детальні системні сценарії використання, щоб потім перерости у надзвичайно детальні та вичерпні тести.

Таблиця 3.1 - Таблиця use-case "Account Management System"

№	РОЛЬ	USE-CASE	ОПИС
1	Гість	Зареєструватися	Зареєструватися у систему
2		Увійти	Увійти до системи
3	Лікар/Пацієнт	Вийти	Вийти із системи
4	Адміністратор	Вийти	Вийти із системи
5		Створити новий обліковий запис адміністратора	Додати ім’я, електронну пошту, придумати пароль

Таблиця 3.2 - Таблиця use-case "Appointment Management System"

№	РОЛЬ	USE-CASE	ОПИС
1	Адміністратор	Вибрати календар лікаря	Переглянути всі дані про зустрічі з обраним лікарем у календарі
2		Створити зустріч	Обрати лікаря і заповнити поля в модальному вікні: назва, опис, пацієнт, дата і час, тривалість
3		Редагувати зустріч	Обрати зустріч і в модальному вікні змінити одне або більше полів: назва, опис, пацієнт, дата і час, тривалість
4		Скасувати зустріч	Обрати зустріч і видалити її із календаря лікаря і пацієнта
5	Лікар	Переглянути зустрічі	Відкрити календар і переглянути всі призначені зустрічі
6		Підтвердити зустріч	Змінити статус зустрічі із очікуваного на підтверджене
7		Скасувати зустріч	Обрати зустріч і видалити її із календаря
8		Поставити діагноз	Обрати зустріч і в модальному вікні заповнити поле: діагноз

9	Пацієнт	Переглянути зустрічі	Відкрити календар і переглянути всі призначені зустрічі

3.3 UML-діаграми

UML (Unified Modeling Language) — уніфікована мова моделювання, що використовується розробниками програмного забезпечення для візуалізації процесів та роботи систем.

Складання діаграм за допомогою UML — це чудовий спосіб допомогти іншим швидко зрозуміти складну ідею чи структуру.

3.3.1 Діаграма варіантів використання

Діаграма варіантів використання — (діаграма прецедентів, сценарій використання, use case) — дозволяє уявити типи ролей та їх взаємодію із системою. Проте не показує порядок виконання кроків. Зображує функціональні вимоги (те, що система може зробити) з точки зору користувача.

Діаграми варіантів використання складаються з 4 об’єктів: актор, прецедент (варіант використання), система, зв’язок.

Актор. Поняття когось, або чогось, що взаємодіє з системою, але не належить до неї (знаходиться за межами системи). Зазвичай позначається у вигляді стилізованого чоловічка.

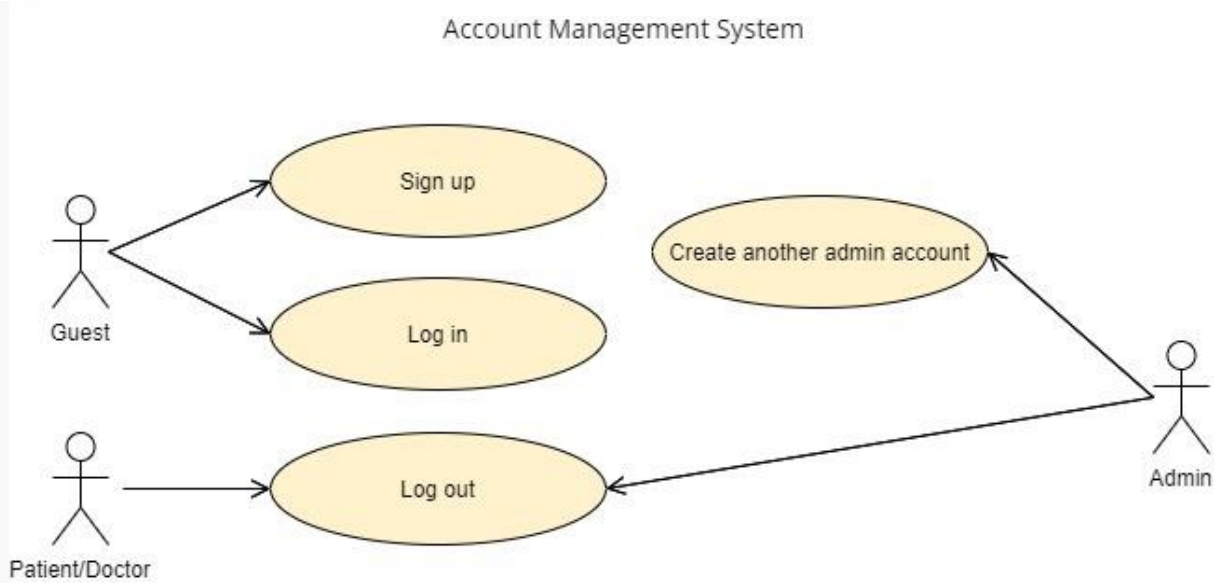
Випадок використання (прецедент). Прецеденти визначають очікувану поведінку та відповідають на питання, що робить система. Представляють набір можливих функцій, дій або завдань. Зображується у вигляді еліпса з назвою дії (дієсловом) у ньому. Прецедент вказує, що має трапитись, проте не відповідає на запитання, як це має статись.

Система. Те, що моделюється. Це може бути сайт, мобільний додаток або навіть модуль програмного забезпечення. Зображується у вигляді прямокутника з назвою системи у верхній частині.

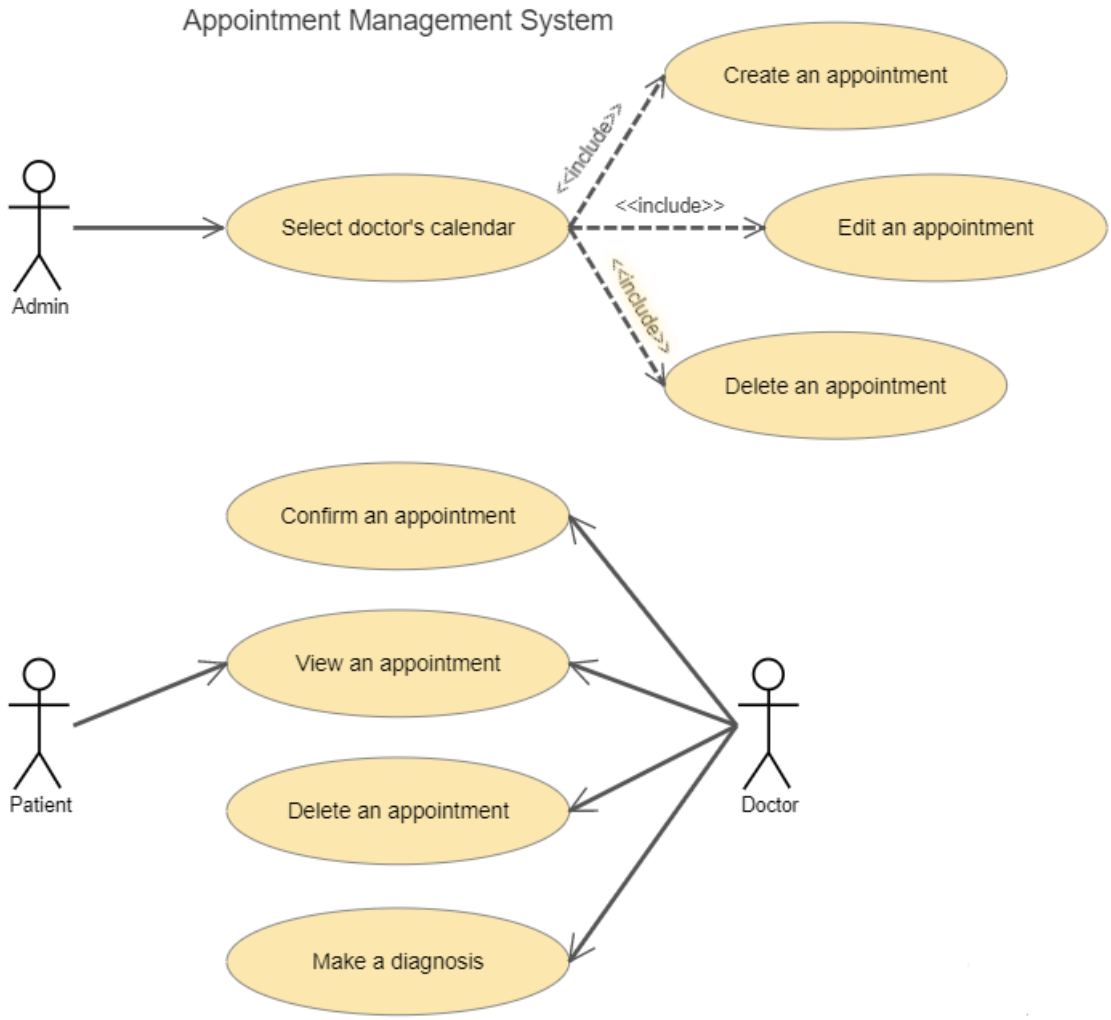
Зв'язки (відношення). Усі актори мають бути пов'язані з прецедентом. Проте не усі прецеденти повинні бути пов'язані з акторами. Частіше всього для зображення зв'язку використовується суцільна лінія.

У діаграмі варіантів використання існує 4 типи зв'язків:

- Асоціація. Звичайний зв'язок актора та прецеденту. Позначається суцільною лінією без напису
- Розширення. Показує додаткову функціональність або можливий не обов'язковий варіант поведінки системи. Базовий прецедент має сенс сам по собі, не залежить від розширюючого і може існувати без нього. Відношення розширення позначається пунктирною лінією зі звичайним вказівником, що вказує на базовий прецедент та написом <<extend>>.
- Включення. Показує, що поведінка одного прецеденту включається як складовий компонент у послідовність поведінки іншого прецеденту. Відношення включення використовується для уникнення дублювання однакових прецедентів та додає функціональність, не вказану в базовому. Відношення позначається пунктирною лінією зі стрілкою та написом <<include>>, що вказує на включений варіант.
- Генералізація. Генералізація (успадкування, узагальнення) це батьківсько-дочірні відношення. Генералізація позначається суцільною лінією з трикутним не зафарбованим вказівником, що вказує на прецедент-предок.



”Рисунок 3.1 – use case діаграма “Account Management System” застосунку”



“Рисунок 3.2 – use case діаграма “Appointment Management System”застосунку”

3.3.2 Діаграма послідовності

Діаграма послідовності (Sequence Diagram) — показує часові особливості передачі і прийому повідомлень об’єктами. Впорядкованість за часом слід розуміти як послідовність дій і не плутати з часовими діаграмами.

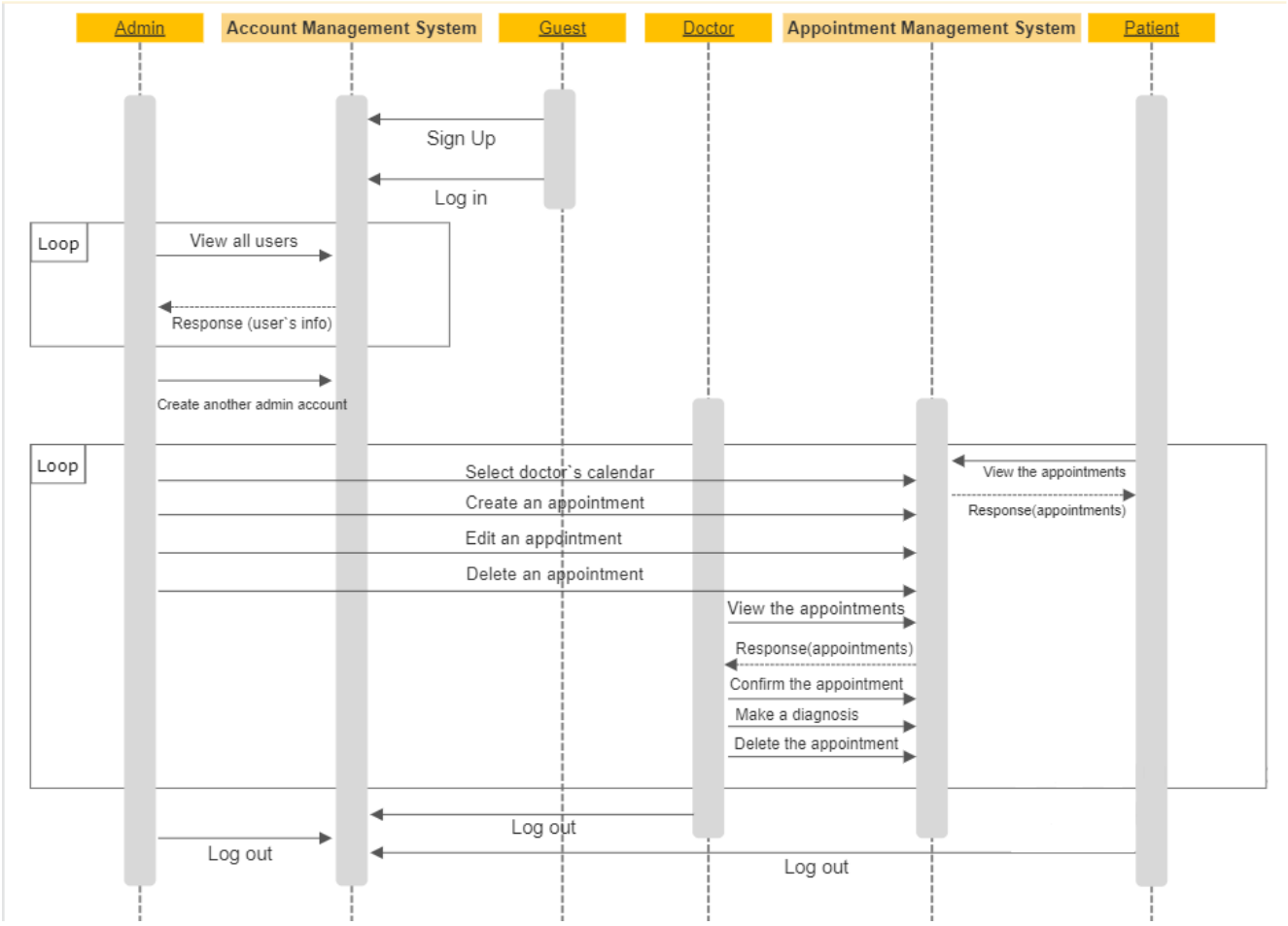
Позначення діаграми послідовності:

Лінія життя починається з об’єкта-прямокутника (голова) та зображується вертикальною пунктирною лінією (стеблом). Вона служить для позначення періоду часу, протягом якого об’єкт існує в системі. Якщо об’єкт існує в системі постійно, то його лінія життя повинна продовжуватися по всій площині діаграми зверху донизу.

Смуга активації (фокус управління) — тонкий прямокутник на лінії життя, протягом якого елемент виконує операцію. Довжина прямокутника вказує на тривалість перебування об’єктів в активному режимі.

Повідомлення (виклики) з’являються в послідовному порядку на лінії життя. Повідомлення зображується за допомогою стрілок. Початок стрілки завжди торкається лінії життя відправника та лінії життя об’єкта, що приймає повідомлення.

Фрагмент використовується, якщо процеси утворюють цикл або вимагають виконання умов для його закінчення. Він складається з вікна (рамки) та оператора фрагмента (напису) в п’ятикутнику зверху зліва.



"Рисунок 3.3 – Діаграма послідовності застосунку"

3.3.3 ER-діаграма

ER-модель (Entity-relationship model або Entity-relationship diagram) – це семантична модель даних, яка призначена для спрощення процесу проектування бази даних. З ER-моделі можуть бути породжені всі види баз даних: реляційні, ієрархічні, мережні, об'єктні. В основі ER-моделі лежать поняття "сутність", "зв'язок" та "атрибут".

ER-модель – це представлення бази даних у вигляді наочних графічних діаграм. ER-модель візуалізує процес, що визначає деяку предметну область. Діаграма "сутність-зв'язок" – це діаграма, яка представляє в графічному вигляді сутності, атрибути і зв'язки.

Сутність в базі даних – це будь-який об'єкт в базі даних, який можна виділити виходячи з суті предметної області для якої розробляється ця база даних. Розробник

бази даних повинен вміти правильно визначати сутності. Будь-яка сутність позначається у вигляді прямокутника з назвою всередині нього.

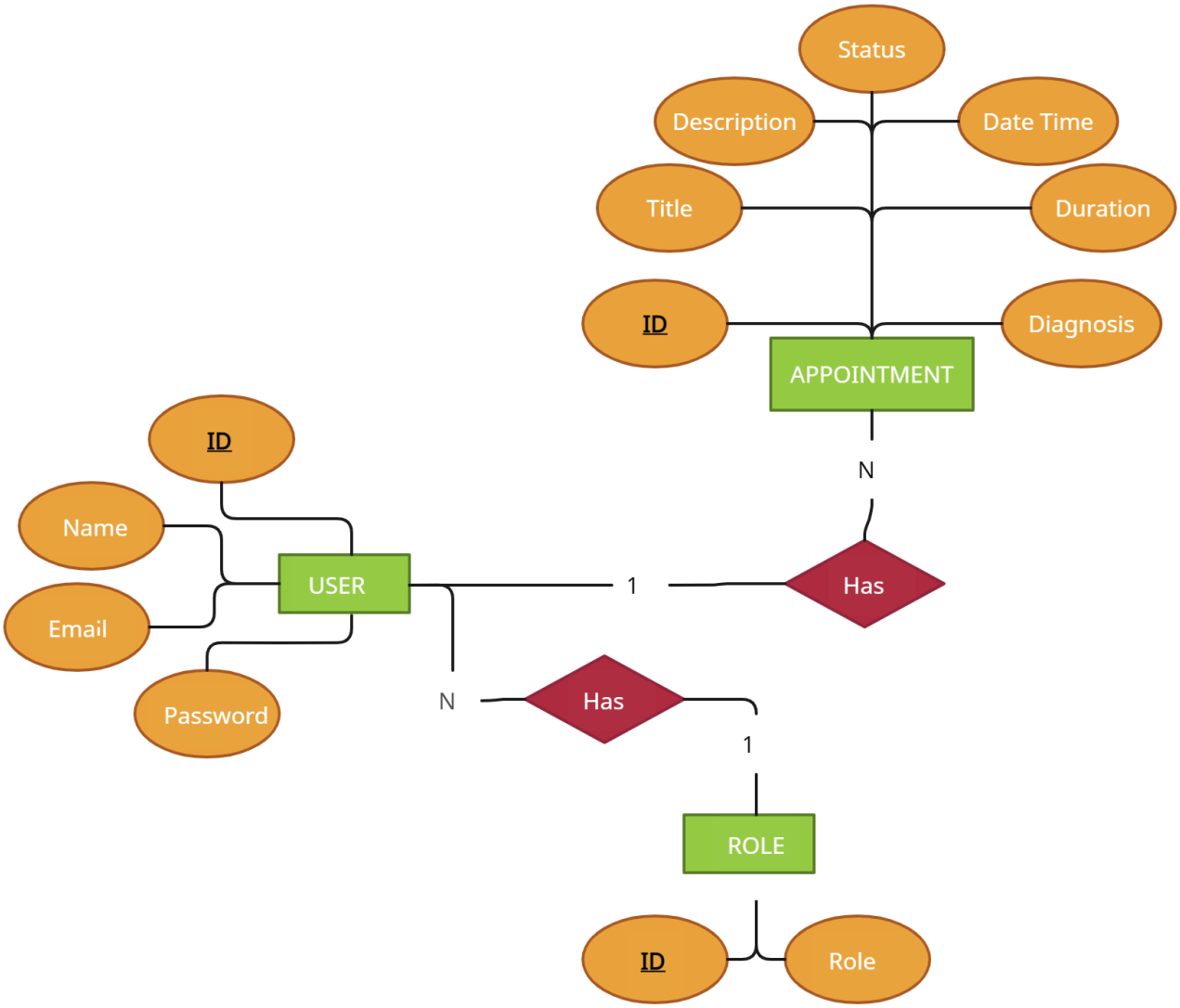
Кожен тип сутності має певний набір атрибутів. Атрибути призначені для опису конкретної сутності.

На ER-діаграмі атрибути позначаються у вигляді еліпсу з назвою всередині еліпсу. Якщо атрибут є первинним ключем, то його назву підкреслюють.

Зв’язок – це деяке відношення між двома типами сутностей. В ER-моделі зв’язки позначаються у вигляді ромба. Всередині ромба вказується дієслово, яке визначає характер взаємодії між сутностями.

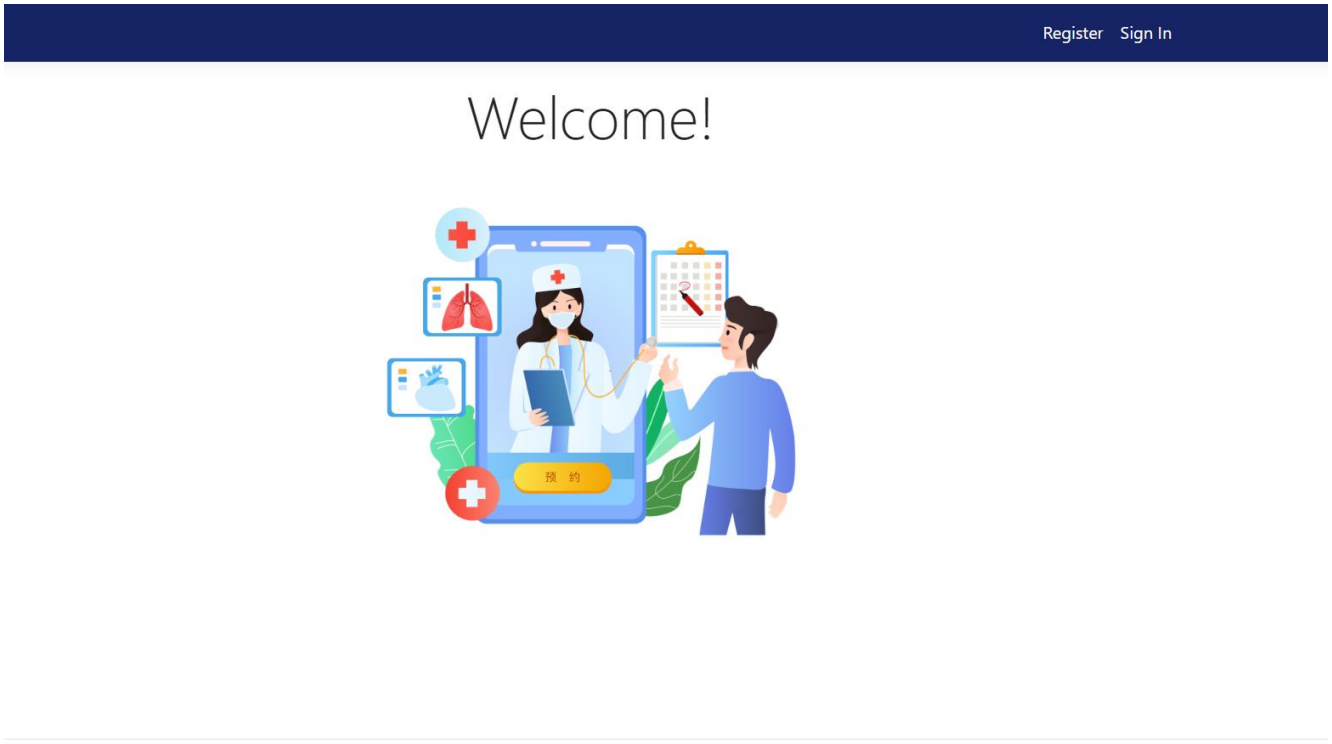
Розрізняють 3 типи зв’язків між сутностями:

- “один до одного” або 1:1. Це означає, що одному екземпляру сутності може відповідати тільки один екземпляр іншої сутності;
- “один до багатьох” або 1:M. Це означає, що одному екземпляру сутності може відповідати будь-яка кількість (M) екземплярів іншої сутності. Якщо відоме значення максимальної кількості екземплярів, то це значення вказується замість символу M;
- “багато до багатьох” або M:N. Це означає, що декільком екземплярам однієї сутності може відповідати декілька екземплярів іншої сутності.

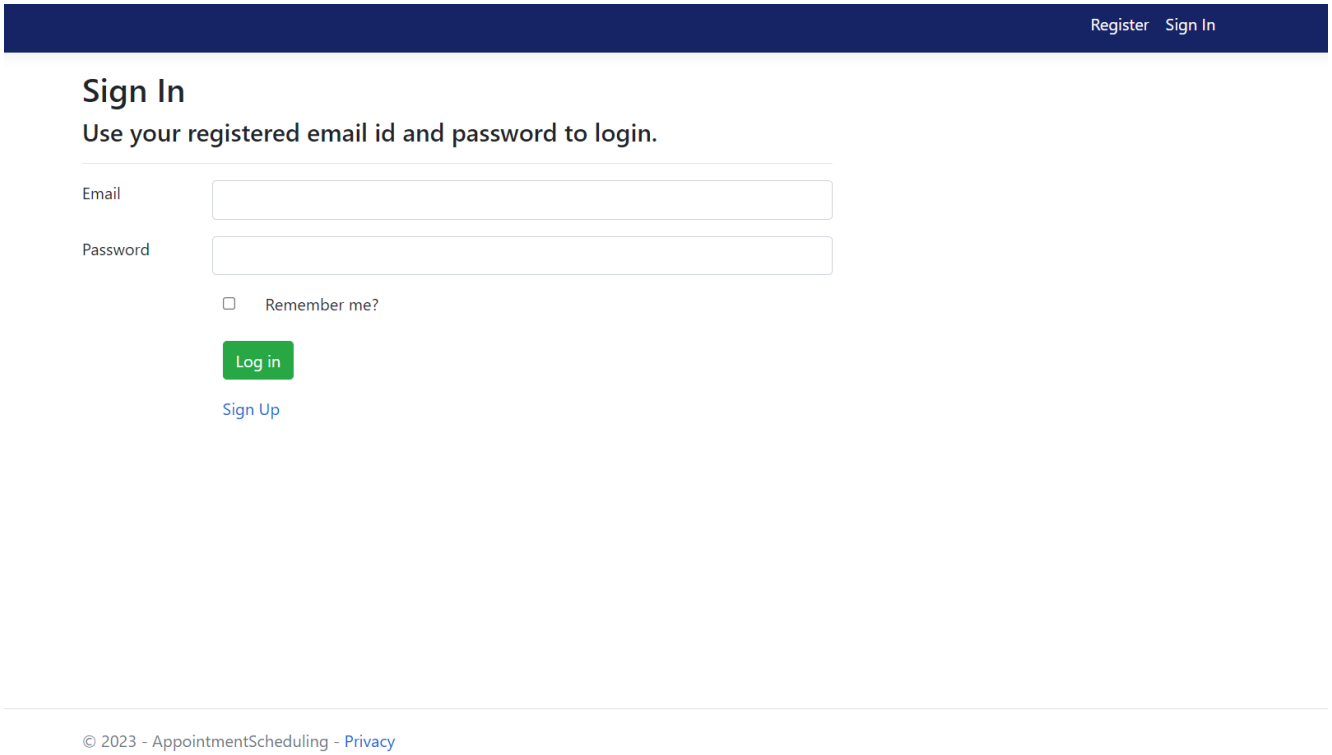


“Рисунок 3.4 – ER-діаграма застосунку”

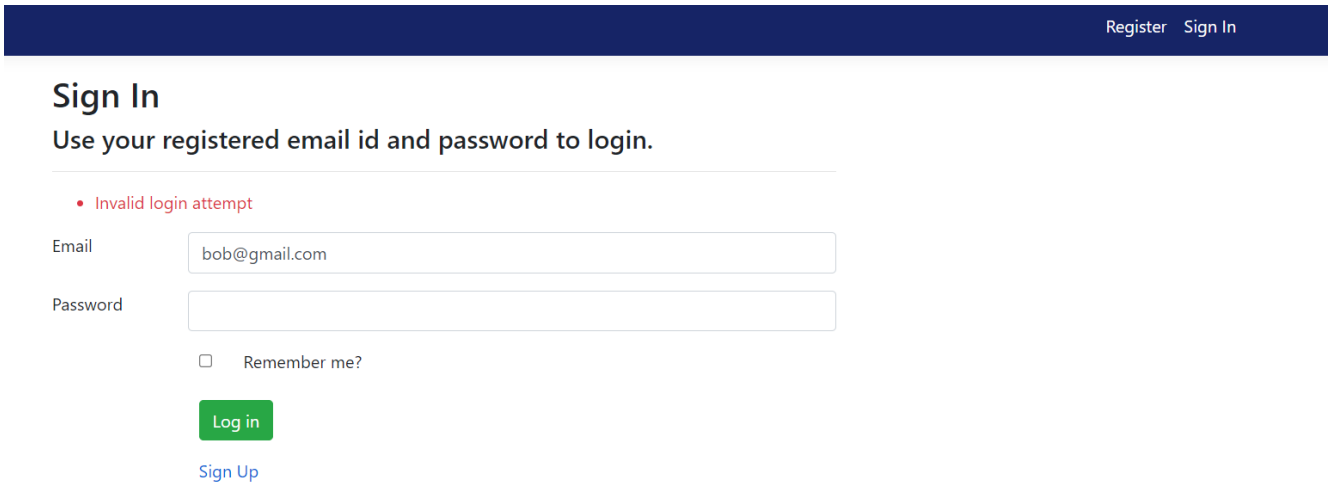
3.4 Вигляд



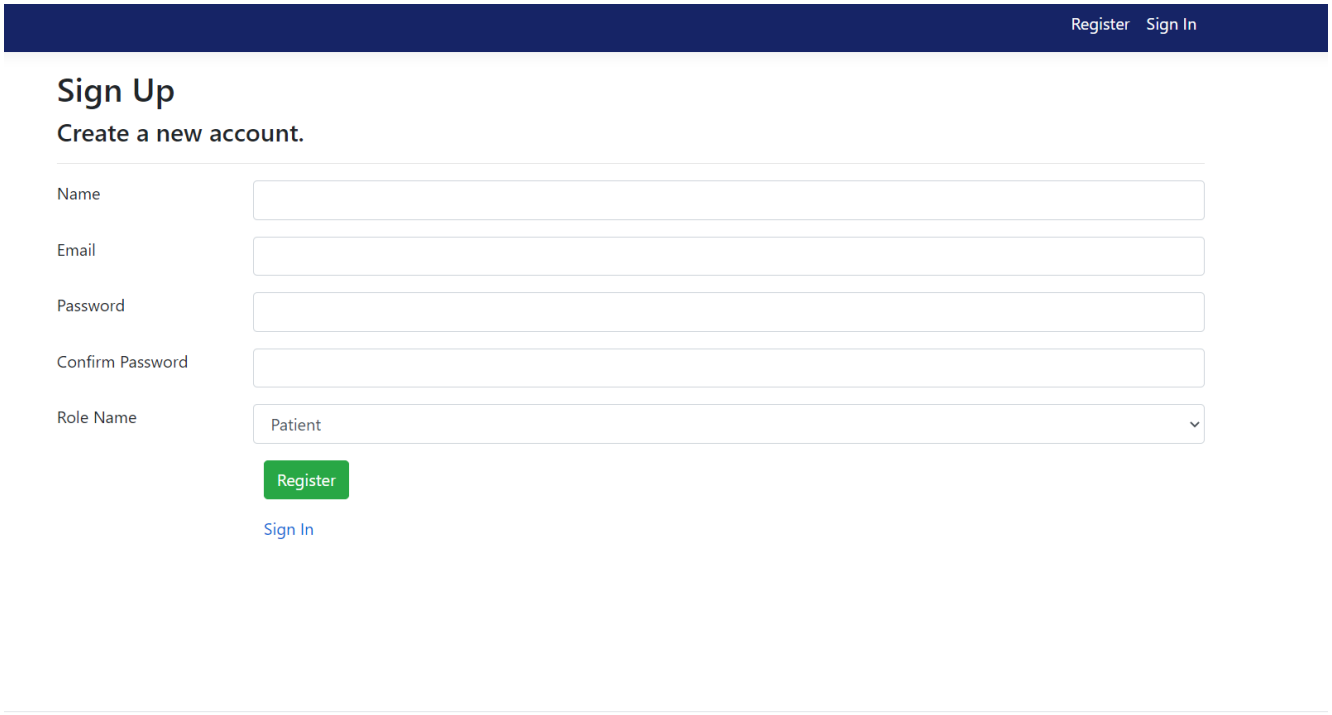
”Рисунок 3.5 - Головна сторінка"



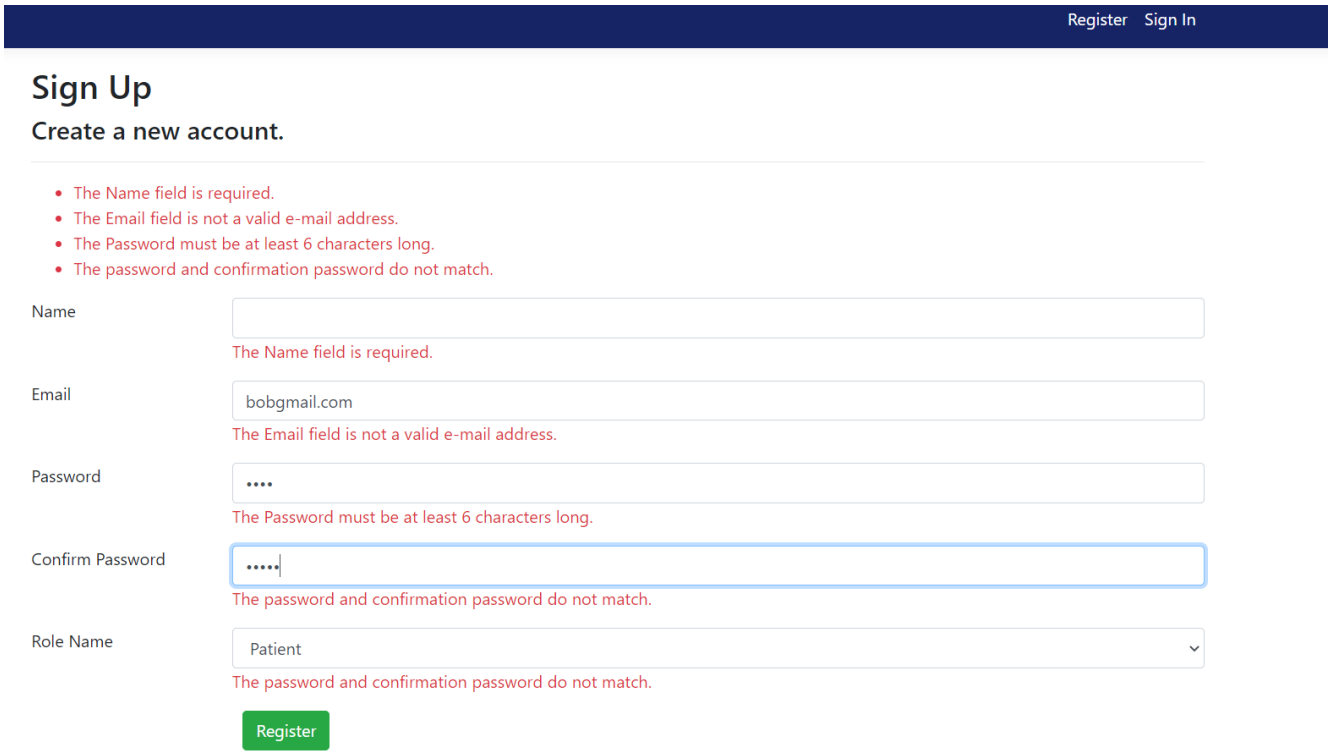
"Рисунок 3.6 - Сторінка входу"



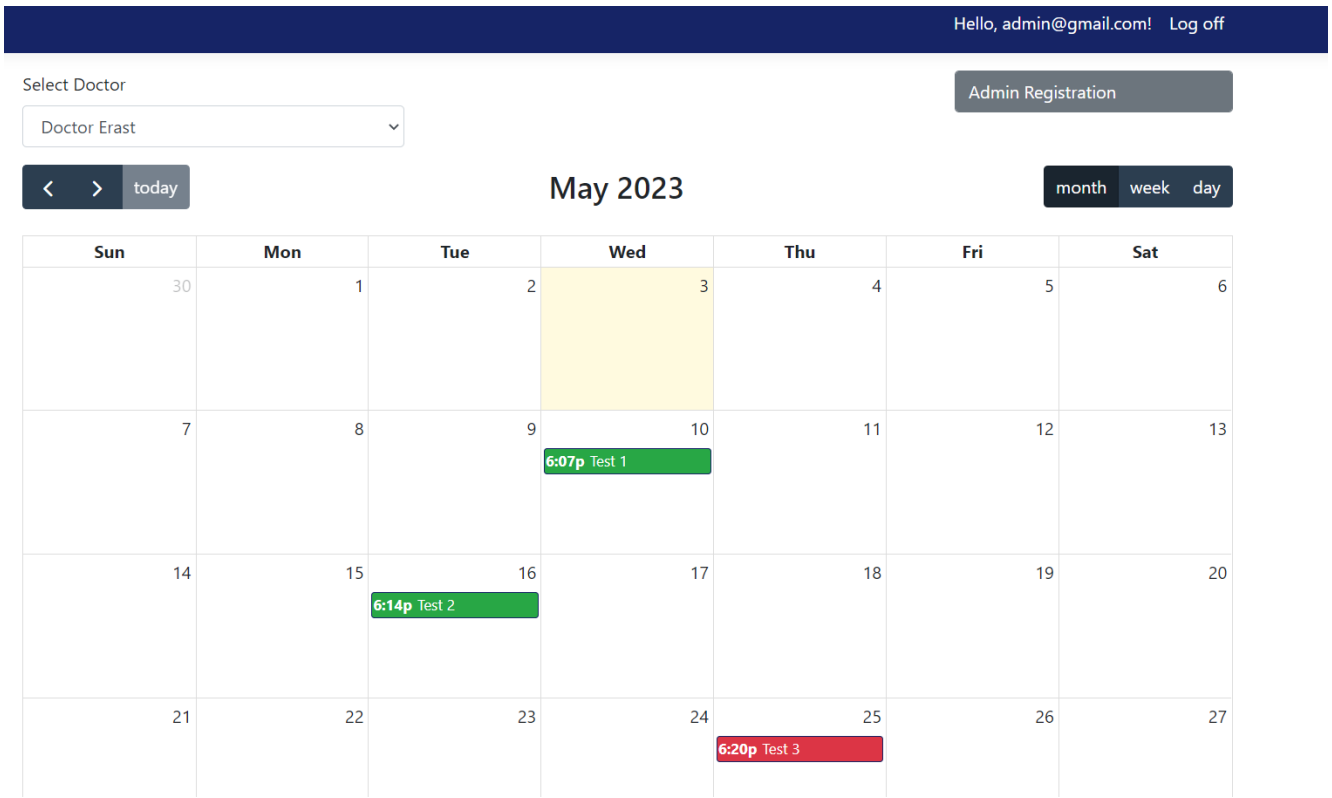
"Рисунок 3.7 - Сторінка входу із некоректно введеними даними"



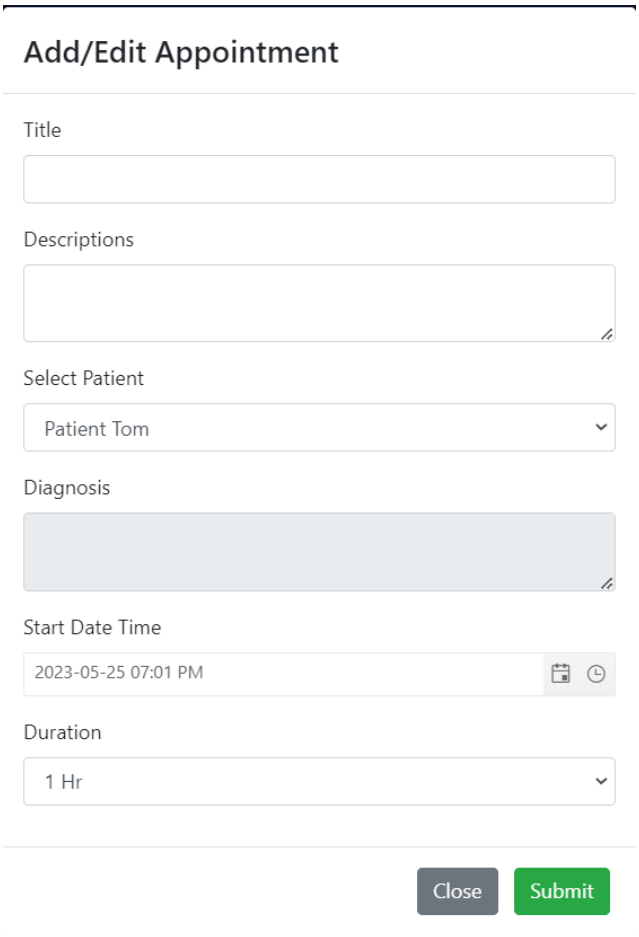
"Рисунок 3.8 - Сторінка реєстрації"



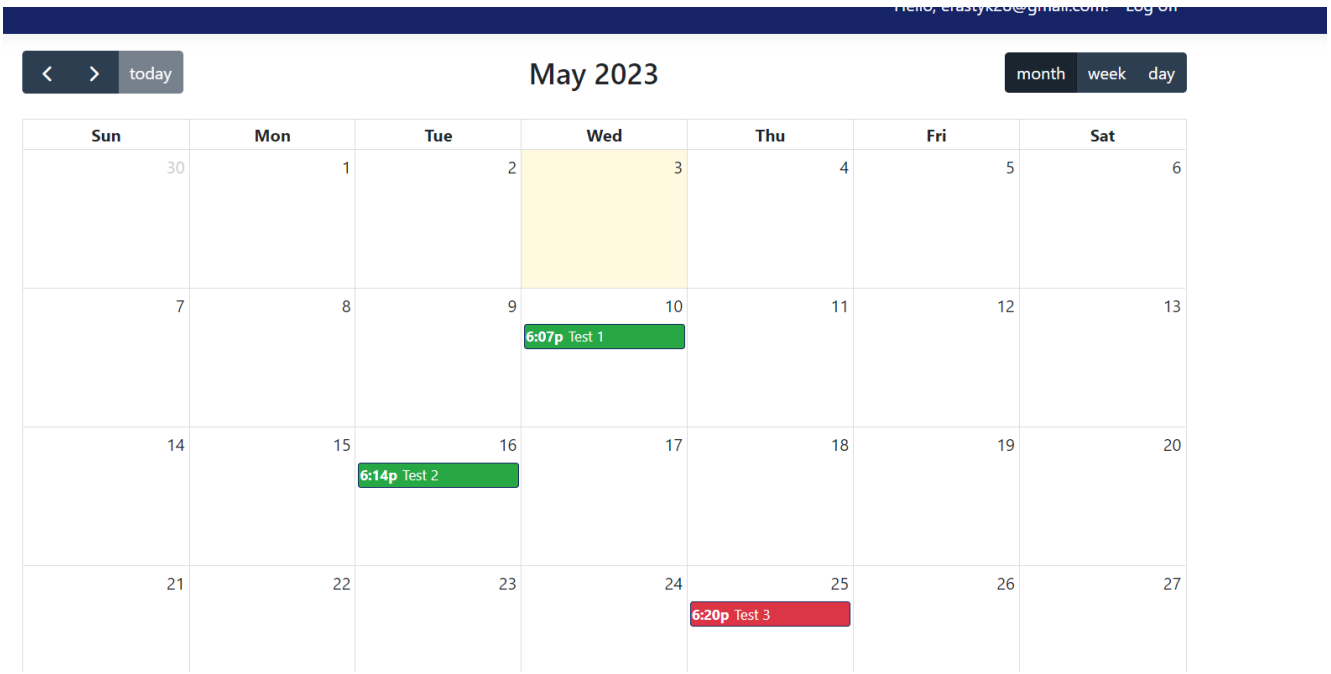
"Рисунок 3.9 - Сторінка реєстрації із некоректно введеними даними"



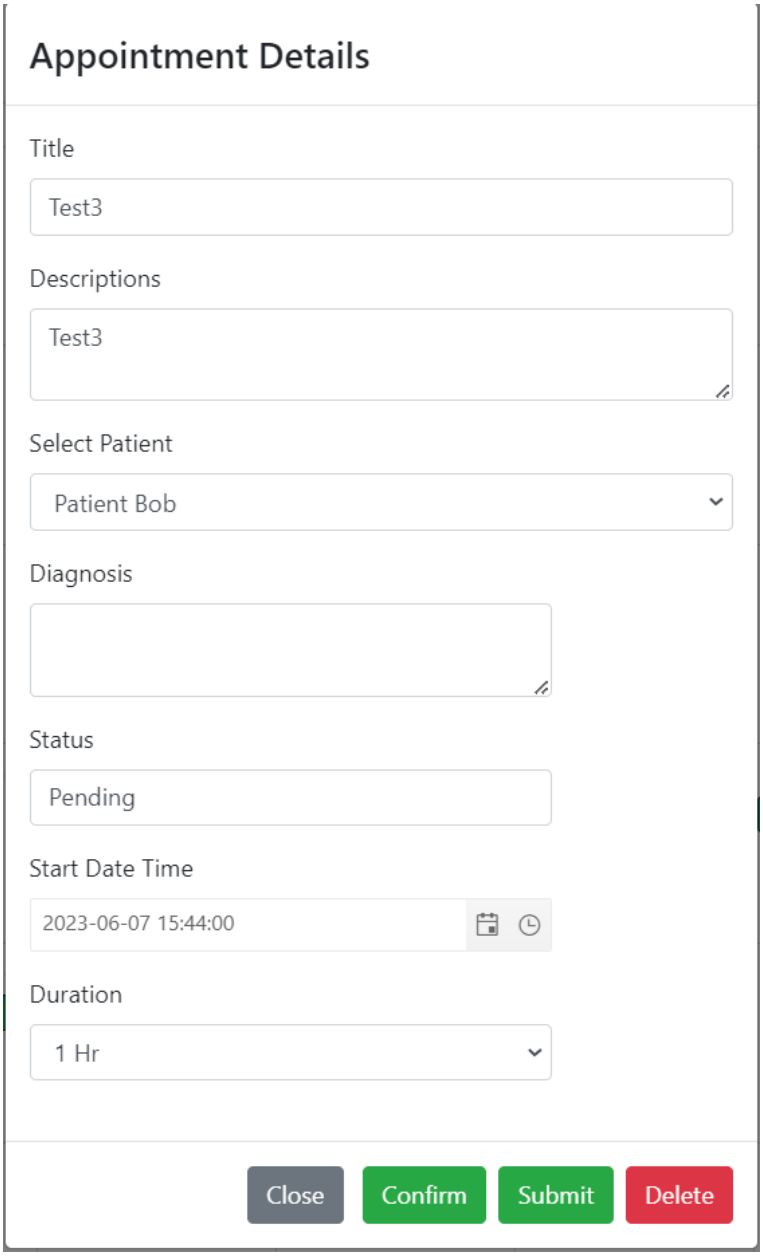
“Рисунок 3.10 - Головна сторінка адміністратора”



"Рисунок 3.11 - Модальне вікно для створення зустрічі"



"Рисунок 3.12 - Головна сторінка лікаря/пацієнта"



"Рисунок 3.13 - Модальне вікно для перегляду деталей, підтвердження зустрічі та внесення діагнозу для лікаря"

ВИСНОВКИ

Дипломна робота присвячена розробці веб-додатку, а також аналізу існуючих на ринку готових програмних продуктів для здійснення електронного запису до лікаря.

Для реалізації даного завдання було вибрано технологію ASP.NET Core with MVC, розглянуто основні її особливості, архітектуру, призначення основних складових та можливості їх використання.

На основі розглянутих матеріалів було створено проект. Лабораторний практикум охоплює такі основні аспекти створення додатків: побудова архітектури додатку, робота з даними, створення інтерфейсу користувача, комунікація між компонентами додатку.

В ході виконання дипломної роботи було зроблено наступне:

1. Здійснено аналіз існуючих продуктів на ринку, що є популярними в медичній галузі;

2. Проаналізовано та обґрунтовано вибір засобів розробки програмного продукту;

3. Написано вимоги та побудовано діаграми для спрощення розробки додатку;

4. Створено веб-додаток для вирішення поставленої проблеми;

5. Реалізовано систему реєстрації та автентифікації;

6. Реалізовано систему створення, редагування та видалення зустрічі;

7. Реалізовано систему перегляду всіх зустрічей;

8. Реалізовано відображення підтверджених та не підтверджених зустрічей;

9. Виконано тестування програмного продукту;

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Адам Фрімен, Професійний ASP .NET Core MVC 2(сьоме видання) - Apress, 25 Жовтня, 2017

2. Джеффрі Ріхтер, CLR через C#(четверте видання) - Microsoft Press, 25 Листопада, 2005

3. Ендрю Троелсен, Професійний C# : з .NET та .NET Core(восьме видання) – Apress, 21 Листопада, 2017

4. Посилання на офіційний сайт додатку «Helsi» [Електронний ресурс] – Режим доступу: https://helsi.me/

5. Посилання на офіційний сайт додатку «Medcard24» [Електронний ресурс] – Режим доступу: https://portal.medcard24.net/

6. Посилання на офіційний сайт додатку «Health24» [Електронний ресурс] – Режим доступу: https://h24.ua/

7. ASP.NET Core documentation [Електронний ресурс] – Режим доступу: https://learn.microsoft.com/en-us/dotnet/

8. ASP.NET Core MVC documentation [Електронний ресурс] – Режим доступу: https://learn.microsoft.com/en-us/aspnet/core/mvc/

9. Microsoft SQL Server documentation [Електронний ресурс] – Режим доступу: https://learn.microsoft.com/en-us/sql/sql-server/

10. StackOverflow [Електроний ресурс] – Режим доступу: https://stackoverflow.com/

11. Метаніт [Електроний ресурс] – Режим доступу: https://metanit.com

12. Wikipedia [Електроний ресурс] – Режим доступу: https://wikipedia.org

13. MSDN – мережа розробників Microsoft[Електроний ресурс] – Режим доступу: https://msdn.microsoft.com/uk-ua

14. BestProg [Електронний ресурс] – Режим доступу: https://www.bestprog.net/uk/sitemap_ua/

15. DOU [Електронний ресурс] – Режим доступу: https://dou.ua/forums/topic/40575/

16. Udemy [Електронний ресурс] – Режим доступу: https://www.udemy.com/