НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
(повна назва інституту/факультету)

кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ
(повна назва кафедри)

«До захисту допущено»
Завідувач кафедри БМК

_____ Світлана АЛХІМОВА
(підпис) (ініціали, ПРІЗВИЩЕ

“___” __*червня*__2025р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою
«Комп’ютерні технології в біології та медицині»
зі спеціальності *122 «Комп’ютерні науки»*
на тему: ***Веб-застосунок “Виписка електронних рецептів”***

Виконав: студент **IV** курсу, групи _БС-14__

ВОВК АНДРІЙ МИКОЛАЙОВИЧ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник: *Коваленко Олександр Сергійович,*
професор каф. БМК, д.м.н., професор _____
(посада, науковий ступінь, вчене звання, прізвище, ім’я, по ініціали) (підпис)

Рецензент: *асистент каф. біомедичної інженерії, асистент,*
Царенко Микола Андрійович _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут (факультет) БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
(повна назва)

Кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ
(повна назва)

Рівень вищої освіти – перший (бакалаврський)
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Комп'ютерні технології в біології та медицині»
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри БМК
_____ Світлана АЛХІМОВА
«____» __*квітень*___2025 р.

ЗАВДАННЯ
на дипломну роботу студенту

ВОВКУ АНДРІЮ МИКОЛАЙОВИЧУ

(прізвище, ім'я, по батькові)

1. Тема роботи ***Веб-застосунок "Виписка електронних рецептів"***

керівник роботи
Коваленко Олександр Сергійович, д.м.н., професор
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «26»_*травня*_ 2025 р. №1758-*с*_
в

2. Термін подання студентом роботи ***22-23 травня 2025 року***

3. Вихідні дані: Аналіз наявних додатків-аналогів, дані медичних та технічних досліджень.

4. Зміст роботи : Проаналізувати наявні на базі практики літературу та існуючі рішення за темою ДР. Проаналізувати та обрати засоби реалізації веб-застосунку та засоби захисту інформації. Провести розрахунок економічного ефекту за темою ДР. Зробити загальні висновки та визначитись з остаточною темою ДР до наказу.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): у роботі представлено 19 рисунків та 1 таблиця.

6. Консультанти розділів роботи[1*]

[1*] Якщо визначені консультанти. Консультантом не може бути зазначено наукового керівника магістерської дисертації.

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Дипломної роботи	Іляш Ольга Ігорівна, доктор економічних наук, професор	07.04.25 *виконано*	04.05.25

7. Дата видачі завдання ***06 квітня 2025р.***

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Отримати завдання за темою ДР на практику	До 24.12.2024р.	***виконано***
2	Переддипломна практика. Виконання практичної частини ДР та додаткових розділів	1-4 тиждень за графіком практики	***виконано***
3	Виконання основних розділів ДР (Вступ,; Розділ 1 аналоги програмного забезпечення. Огляд літературних джерел; Розділ 2 засоби реалізації програмного веб-застосунку для виписки рецептів ; Розділ 3 обґрунтування вибору програмного середовища / побудови нейронної мережі / класифікації вибірки машинного навчання для реалізації поставленої задачі; Розділ 4 інформаційне забезпечення. Програмна реалізація та методика роботи веб-застосунки для виписки електронних рецептів; Розділ 5 узагальнення результатів роботи;)	Протягом усієї практики	
4	Перевірка ДР науковим керівником	до 20.05.2025	
5	Подання в електронному вигляді ДР на перевірку нормоконтролера та плагіат (StrikePlagiarism.com). Доопрацювання ДР	Не пізніше 23.05.2025	
6	Отримати відгук від керівника ДР	Не пізніше 05.06.2025	
7	Надати пакету документів по ДР та супровідних до неї документів на засідання кафедри		
8	Подання ДР рецензенту. Отримання рецензії.	09.06 – 13.06.2025	
9	Захист ДР в ЕК	*16.06 - -21.06.2025*	

Студент _____ **Андрій ВОВК**
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник ДР _____ **Олександр КОВАЛЕНКО**
(підпис) (ім'я, ПРІЗВИЩЕ)

Нормоконтролер _____ **Галина КОРНІЄНКО**
(підпис) (ім'я, ПРІЗВИЩЕ)

АНОТАЦІЯ

Дипломна робота за темою «Веб-застосунок “Виписка електронних рецептів”» виконана студентом кафедри біомедичної кібернетики ФБМІ *Вовка Андрія Миколайовича зі спеціальності 122 «Комп’ютерні науки» за освітньо-професійною програмою «Комп’ютерні технології в біології та медицині» та* складається зі: вступу; 5 розділів , висновків до кожного з цих розділів; загальних висновків; списку використаних джерел, який налічує 25 джерел. Загальний обсяг роботи 78 сторінок.

Актуальність теми. Актуальність теми зумовлена цифровізацією медичної сфери в Україні та впровадженням електронної системи охорони здоров’я. Електронні рецепти сприяють оптимізації медичних послуг, підвищенню їх доступності, автоматизації виписування ліків і зменшенню помилок у взаємодії між лікарями, пацієнтами та аптеками.

Мета і завдання роботи.

Метою роботи є розробка веб-застосунку для виписки електронних рецептів, який забезпечить зручне, безпечне та ефективне формування, передачу й обробку медичних призначень відповідно до стандартів цифрової медицини та вимог захисту персональних даних, зокрема ДСТУ ISO 17523:2016. Реалізація такого інструменту покликана оптимізувати взаємодію між лікарями, пацієнтами та аптечними закладами, скоротити кількість помилок і підвищити загальну доступність медичних послуг.

Її досягнення передбачає вирішення наступних завдань:

1. Аналіз вітчизняних та зарубіжних джерел щодо цифровізації охорони здоров’я та впровадження електронних рецептів.
2. Обґрунтування вибору архітектури, технологій і стандартів для реалізації системи.
3. Проєктування та реалізація безпечної структури бази даних для зберігання інформації про пацієнтів, рецепти та візити.

4. Розробка функціоналу для реєстрації користувачів, авторизації, створення, перегляду та обробки електронних рецептів.
5. Забезпечення високого рівня інформаційної безпеки, зокрема шифрування, контроль доступу та захист персональних даних.

Використані методи. У роботі застосовано комплекс сучасних методів і технологій, зокрема: аналіз літературних джерел та програмних аналогів, моделювання структури бази даних із використанням ORM (SQLAlchemy), побудову веб-застосунку на основі фреймворку Flask (Python), розробку користувацького інтерфейсу за допомогою HTML, Jinja2 та Tailwind CSS, створення адаптивних форм з валідацією через Flask-WTF. Для забезпечення інформаційної безпеки впроваджено механізми авторизації, хешування паролів, CSRF-захисту, контроль доступу за ролями користувачів, а також реалізовано захищене зберігання даних у PostgreSQL. Усі компоненти спроєктовані відповідно до стандарту ДСТУ ISO 17523:2016.

Отримані результати. . У результаті виконаної роботи розроблено повнофункціональний веб-застосунок “Виписка електронних рецептів”, який дозволяє лікарям створювати рецепти, пацієнтам — отримувати призначення та QR-коди для аптек, а провізорам — здійснювати видачу лікарських засобів. Реалізовано систему реєстрації, авторизації та розмежування прав доступу для трьох типів користувачів. Забезпечено відповідність системи сучасним вимогам інформаційної безпеки, структуровано базу даних відповідно до медичних стандартів, протестовано функціонал на відповідність вимогам. Система готова до впровадження в умовах приватних клінік або пілотних програм цифрової медицини в Україні.

Апробація результатів роботи. Не передбачено.

Публікації. Не предбачено.

Бібліографічний опис ДР

Вовк А.М. [Веб-застосунок “Виписка електронних рецептів”] :

дипломна роб. бакалавра : 122 Комп’ютері науки / Вовк Андрій Миколайович. – Київ, 2025. – 80 с

ANNOTATION

The bachelor's thesis titled *“* E-Prescription Web Application*”* was completed by Andrii Mykolaiovych Vovk, a student of the Department of Biomedical Cybernetics at the Faculty of Biomedical Engineering, specializing in 122 "Computer Science" under the educational and professional program "Computer Technologies in Biology and Medicine." The thesis includes: an introduction; 5 chapters with individual conclusions; general conclusions; a list of 25 references. The total length of the thesis is 78 pages.

Relevance of the Topic

The relevance of the topic is driven by the digitalization of the healthcare sector in Ukraine and the implementation of an electronic healthcare system. Electronic prescriptions contribute to the optimization of medical services, improvement of their accessibility, automation of medication issuance, and reduction of errors in interactions between doctors, patients, and pharmacies.

Purpose and Objectives of the Study

The purpose of the study is to develop a web application for issuing electronic prescriptions that ensures convenient, secure, and efficient creation, transmission, and processing of medical prescriptions in accordance with digital healthcare standards and data protection requirements, particularly ДСТУ ISO 17523:2016. This tool aims to optimize interactions between doctors, patients, and pharmacies, reduce errors, and enhance overall accessibility of healthcare services.

To achieve this goal, the following tasks were defined:

1. Analyze domestic and international sources on healthcare digitalization and the implementation of electronic prescriptions.
2. Justify the choice of system architecture, technologies, and applicable standards.

3. Design and implement a secure database structure for storing patient, prescription, and visit information.

4. Develop functionalities for user registration, authorization, creation, viewing, and processing of electronic prescriptions.

5. Ensure a high level of information security, including encryption, access control, and protection of personal data.

Methods Used

The study employed a set of modern methods and technologies, including: literature and software solution analysis, database structure modeling using ORM (SQLAlchemy), web application development with the Flask (Python) framework, user interface creation using HTML, Jinja2, and Tailwind CSS, and adaptive form development with validation via Flask-WTF. To ensure information security, mechanisms for authorization, password hashing, CSRF protection, and role-based access control were implemented. Data is securely stored in PostgreSQL. All components were designed in accordance with the ДСТУ ISO 17523:2016 standard.

Results Obtained

As a result of the study, a fully functional *"* E-Prescription Web Application*"* was developed. It allows doctors to issue prescriptions, patients to receive prescriptions and QR codes for pharmacies, and pharmacists to dispense medications. A complete user registration, authorization, and role-based access system was implemented. The application meets modern information security requirements, features a well-structured database compliant with medical standards, and its functionality was tested for compliance. The system is ready for deployment in private clinics or pilot digital healthcare programs in Ukraine.

Dissemination of Results

Not provided.

Publications

Not provided.

Bibliographic Reference

Vovk, A. M. *[Web Application for Electronic Prescription Issuance]*: Bachelor's thesis in Computer Science (122) / Andrii Mykolaiovych Vovk. – Kyiv, 2025. – 80p.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних

LIMS – лабораторна інформаційна система

ПЗ – програмне забезпечення

API – Application Programming Interface, інтерфейс прикладного програмування, який дозволяє системам взаємодіяти між собою.

БД – База даних, це структурована колекція даних, яка зберігається та управляється комп'ютерною системою.

CSS – Cascading Style Sheets, мова стилів для оформлення зовнішнього вигляду HTML-сторінок.

ДСТУ – Державний стандарт України.

DOM – Document Object Model, об'єктна модель документа HTML або XML.

EHR – Electronic Health Record, електронна медична карта пацієнта.

Flask – Мікрофреймворк для створення веб-додатків мовою Python.

HTML – HyperText Markup Language, мова гіпертекстової розмітки документів для вебу.

HTTP – HyperText Transfer Protocol, протокол передачі гіпертексту, який використовується для комунікації між веб-клієнтом і сервером.

JWT – JSON Web Token, формат безпечної передачі інформації між сторонами.

JSON – JavaScript Object Notation, текстовий формат для обміну даними.

ORM – Object-Relational Mapping, технологія перетворення даних з бази даних у вигляд об'єктів у коді програми.

PDF – Portable Document Format, формат файлів для збереження документа з фіксованим оформленням.

POST/GET – методи HTTP-запитів для відправки (POST) або отримання (GET) даних із сервера.

SQL – Structured Query Language, мова структурованих запитів для роботи з реляційними базами даних.

UI – User Interface, користувацький інтерфейс.

UX – User Experience, досвід користувача під час взаємодії з системою.

URL – Uniform Resource Locator, адреса ресурсу в Інтернеті.

UUID – Universally Unique Identifier, унікальний ідентифікатор для об'єктів.

WTF – WTForms, бібліотека Python для створення веб-форм.

PostgreSQL — СКБД з відкритим кодом, що підтримує SQL і розширення.

ERD — діаграма "сутність-зв'язок", яка відображає структуру вашої бази даних.

ВСТУП

Актуальність теми дипломної роботи безпосередньо пов’язана з глобальними процесами цифровізації в Україні, зокрема в медичній сфері. В умовах швидкого розвитку електронної системи охорони здоров'я впровадження електронних рецептів є одним із найважливіших кроків на шляху до оптимізації медичних послуг, підвищення їх доступності та якості. Важливим аспектом є автоматизація процесу виписки лікарських засобів, що дозволяє не тільки зменшити кількість людських помилок, а й значно прискорити взаємодію між лікарями, пацієнтами та аптечними закладами.

Мета і завдання роботи

Метою роботи є розробка веб-застосунку для виписки електронних рецептів, який забезпечує зручне, безпечне та ефективне формування і зберігання медичних призначень відповідно до сучасних стандартів цифрової медицини та вимог захисту персональних даних.

Її досягнення передбачає вирішення наступних завдань:

1. Аналіз вітчизняних та зарубіжних джерел щодо цифровізації охорони здоров’я та електронних рецептів.
2. Обґрунтування вибору архітектури, технологій і стандартів для реалізації системи.
3. Проєктування структури бази даних для безпечного зберігання інформації про пацієнтів та рецепти.
4. Розробка функціоналу створення та передачі електронних рецептів.
5. Забезпечення захисту даних відповідно до стандарту ДСТУ ISO 17523:2016.

Використані методи. У роботі застосовано комплекс сучасних методів і технологій, зокрема: аналіз літературних джерел та програмних аналогів, моделювання структури бази даних із використанням ORM (SQLAlchemy),

побудову веб-застосунку на основі фреймворку Flask (Python), розробку користувацького інтерфейсу за допомогою HTML, Jinja2 та Tailwind CSS, створення адаптивних форм з валідацією через Flask-WTF. Для забезпечення інформаційної безпеки впроваджено механізми авторизації, хешування паролів, CSRF-захисту, контроль доступу за ролями користувачів, а також реалізовано захищене зберігання даних у PostgreSQL. Усі компоненти спроєктовані відповідно до стандарту ДСТУ ISO 17523:2016.

Отримані результати. **.** У результаті виконаної роботи розроблено повнофункціональний веб-застосунок “Виписка електронних рецептів”, який дозволяє лікарям створювати рецепти, пацієнтам — отримувати призначення та QR-коди для аптек, а провізорам — здійснювати видачу лікарських засобів. Реалізовано систему реєстрації, авторизації та розмежування прав доступу для трьох типів користувачів. Забезпечено відповідність системи сучасним вимогам інформаційної безпеки, структуровано базу даних відповідно до медичних стандартів, протестовано функціонал на відповідність вимогам. Система готова до впровадження в умовах приватних клінік або пілотних програм цифрової медицини в Україні.

Апробація результатів роботи**.** Не передбачено.

Публікації. Не передбачено.

Структура роботи

Дипломна робота за темою «Веб-застосунок “Виписка електронних рецептів”» виконана студентом *Вовком Андрієм Миколайовичем зі спеціальності 122 «Комп’ютерні науки» за освітньо-професійною програмою «Комп’ютерні технології в біології та медицині»,* побудована за класичним типом та викладена на 80 сторінках машинописного тексту. Вона складається з: вступу; 5 розділів, висновків до кожного з цих розділів; загальних висновків; списку використаних джерел, який налічує 25 джерел. В роботі представлено 19 рисунків, 1 таблиця.

РОЗДІЛ 1
АНАЛОГИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ. ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ

У цьому розділі здійснено детальний аналіз існуючих програмних рішень, призначених для виписки електронних рецептів, як на національному, так і на міжнародному рівні. Розглянуто державну українську систему eHealth, міжнародну платформу з відкритим кодом OpenEMR, комерційне хмарне рішення DrChrono (США), а також локальну систему MedicsDocAssistant. Оцінено їхню функціональність, архітектурні підходи, відповідність нормативним вимогам та можливості інтеграції. Виявлено переваги та недоліки кожного рішення, що дозволило обґрунтувати необхідність створення власного веб-застосунку, адаптованого до українського законодавства, стандартів інформаційної безпеки та потреб користувачів.

1.1 Існуючі програмні рішення для виписки електронних рецептів

У сучасних умовах цифровізації охорони здоров’я особливої актуальності набувають інформаційні системи, що забезпечують безпечну та ефективну виписку електронних рецептів. Їх широке впровадження сприяє зменшенню паперового документообігу, підвищенню точності медичних призначень та забезпеченню зручності для пацієнтів. На цьому етапі важливо провести ретельний аналіз наявних програмних рішень як в Україні, так і за кордоном, щоб сформувати вимоги до власного веб-застосунку та уникнути повторення типових недоліків.

Система eHealth (Україна)

Державна система eHealth є головним офіційним інструментом цифрової трансформації медичної галузі в Україні. Вона була впроваджена в межах медичної реформи та адмініструється Національною службою здоров’я

України (НСЗУ). Одним із ключових її компонентів є модуль електронного рецепта**,** який забезпечує повноцінний процес призначення лікарських засобів і їх подальшого відпуску в аптеках, підключених до системи [8][13].

Функціонування модуля рецепта реалізується через медичні інформаційні системи (МІС), що діють як проміжна ланка між лікарем і центральною базою даних. Авторизація лікаря в системі здійснюється через кваліфікований електронний підпис (КЕП)**,** що забезпечує відповідність національному законодавству у сфері електронного документообігу [6].

Серед переваг eHealth можна виокремити:

1. Централізоване зберігання рецептів, що унеможливлює їх втрату;
2. Автоматичну верифікацію лікаря, пацієнта та аптеки через інтеграцію з державними реєстрами;
3. Контроль за обігом підконтрольних речовин згідно з Постановою КМУ №589;
4. Відповідність вимогам ДСТУ ISO/HL7 13606 щодо структури клінічних документів [6].

Проте, система має й низку обмежень. Вона є закритою для кастомізації — лікарі не можуть змінити логіку формування рецептів, адаптувати інтерфейс під потреби приватного закладу чи розгорнути рішення на власному сервері. У разі технічних збоїв у МІС або центральній базі зникає можливість виписки нових рецептів, що становить ризик для безперервного лікування пацієнтів.

OpenEMR — міжнародна система з відкритим кодом

OpenEMR (Open-source Electronic Medical Record) — одна з найбільш поширених систем для управління електронними медичними записами. Вона сертифікована відповідно до стандартів ONC (Office of the National Coordinator for Health Information Technology, США) і активно використовується в медичних установах різних країн [9].

1. Модуль призначення медикаментів в OpenEMR дозволяє:
2. Виписувати рецепти та пов’язувати їх із діагнозами;
3. Застосовувати шаблони призначень;
4. Враховувати взаємодію ліків;
5. Вести історію призначень і змін.

Однією з ключових переваг системи є відкритий код, що надає можливість глибокої модифікації та інтеграції з локальними рішеннями. Програмісти можуть адаптувати інтерфейс під мову, валюту, національні стандарти без обмежень, що є важливою перевагою порівняно з закритими комерційними платформами.

Попри це, система має низку недоліків:

1. Застарілий дизайн інтерфейсу;
2. Високий поріг входу для нових користувачів;
3. Вимога окремого налаштування серверної інфраструктури;
4. Відсутність інтеграції з українською системою eHealth, що робить використання OpenEMR в нашій країні можливим лише у закритих приватних медичних установах або як прототип [9].

DrChrono (США) — комерційна хмарна система

DrChrono є SaaS-рішенням (Software as a Service), яке поєднує функції електронної медичної картки, управління розкладом, взаємодії з пацієнтами та фінансового обліку [1]. Один з найбільш розвинених модулів — eRx (Electronic Prescription) — дозволяє формувати призначення, перевіряти їх на відповідність дозування, протипоказання та алергії, автоматично надсилати їх до партнерських аптек.

Система відповідає стандартам HIPAA (Health Insurance Portability and Accountability Act) і підтримує:

1. Повноцінний цифровий підпис;
2. Персоналізовані профілі лікарів;

3. Мобільний додаток (особливо для iPad/iPhone);
4. Інструменти аналітики та фінансового прогнозування.

Основними недоліками DrChrono є її закритість для сторонньої кастомізації, англомовний інтерфейс, відсутність підтримки локальних стандартів (таких як ДСТУ ISO 17523:2016) і висока вартість ліцензійного обслуговування, що значно обмежує її застосування в українських реаліях [1][6].

MedicsDocAssistant — локальне рішення для внутрішніх потреб

Це програмне забезпечення призначене для розгортання у лікарнях та поліклініках з локальною мережею. Воно передбачає:

1. Виписку електронних рецептів;
2. Ведення медичної карти пацієнта;
3. Зв’язок між рецептом і діагнозом;
4. Генерацію внутрішніх звітів.

MedicsDocAssistant адаптовано під українське мовне середовище, має інтерфейс українською та російською мовами, що є суттєвим для державних медичних закладів. Водночас система не має веб-версії, тому не передбачає віддаленого доступу, що обмежує її масштабованість. У разі відсутності IT-відділу виникають труднощі з оновленням, налаштуванням чи інтеграцією з аптечними системами [7].

1.2 Нормативно-правова база, що регламентує виписку електронних рецептів

Розробка та впровадження електронних систем у медичній галузі в Україні відбувається у межах чинної нормативної бази. Серед ключових документів, що регламентують процес виписки електронних рецептів, варто відзначити:

1. ДСТУ ISO 17523:2016 «Інформатика охорони здоров’я. Вимоги до електронного рецепта» — встановлює вимоги до структури, безпеки та зберігання даних рецепта [6].
2. Наказ МОЗ №1971 від 31.08.2022 — регламентує правила функціонування електронного рецепта в Україні [13].

Висновок з розділу 1

Проведений аналіз існуючих програмних рішень у сфері виписки електронних рецептів засвідчив значну різноманітність підходів до організації цифрового медичного документообігу. Державна система eHealth демонструє високий рівень централізації та інтеграції з національними реєстрами, однак її функціональність обмежена і не передбачає індивідуальної адаптації до потреб медичних закладів. OpenEMR, як міжнародне рішення з відкритим кодом, має потенціал для локалізації, проте вимагає значних технічних ресурсів для впровадження. Комерційна платформа DrChrono забезпечує багатий функціонал і відповідність міжнародним стандартам, але через високу вартість і відсутність локалізації непридатна для використання в Україні. Локальні системи, такі як MedicsDocAssistant, орієнтовані на внутрішнє використання, мають обмежену масштабованість та не підтримують сучасні веб-технології.

Отже, жодне з проаналізованих рішень не відповідає одночасно вимогам гнучкості, безпеки, доступності, відповідності національним стандартам і можливості масштабування. Це обґрунтовує доцільність створення нового веб-застосунку, який поєднає переваги наявних систем, відповідатиме стандарту ДСТУ ISO 17523:2016, забезпечуватиме зручність як для лікарів, так і для пацієнтів.

РОЗДІЛ 2
ЗАСОБИ РЕАЛІЗАЦІЇ ПРОГРАМНОГО ВЕБ-ЗАСТОСУНКУ ДЛЯ ВИПИСКИ РЕЦЕПТІВ

У цьому розділі детально розкрито та обґрунтовано вибір засобів реалізації програмного продукту для задачі автоматизації процесу виписки електронних рецептів лікарем. Виходячи з постановки задачі дипломної роботи, у розділі проаналізовано та обґрунтовано вибір мови програмування, фреймворків, бібліотек, форматів зберігання й обробки даних, структуру та організацію бази даних, особливості моделювання користувацьких інтерфейсів, шкали й кольорові рішення, принципи валідації, а також засоби забезпечення безпеки та захисту інформації.

Також у розділі розглянуто архітектуру веб-застосунку, структуру моделей і форм, деталі побудови взаємозв’язаних сутностей бази даних, використання шаблонізаторів, засоби стилізації інтерфейсу та забезпечення адаптивності дизайну (з урахуванням сучасних колірних моделей, шрифтів і градієнтів для підвищення зручності та доступності), вибір форматів представлення медичної інформації відповідно до ДСТУ ISO 17523:2016 та практик галузі.

Окремо обґрунтовано вибір стеку технологій (Python, Flask, PostgreSQL, SQLAlchemy, Tailwind CSS, WTForms), а також підходів до структуризації даних, валідації вхідної інформації, розмежування доступу за ролями користувачів і організації ефективної взаємодії між усіма компонентами системи. Таким чином, розділ комплексно висвітлює теоретичні й практичні аспекти реалізації програмного забезпечення для створення надійного, захищеного та зручного веб-застосунку для автоматизованої роботи з електронними рецептами.

2.1. Вибір мови програмування та серверного фреймворку

Для реалізації веб-застосунку “Виписка електронних рецептів” було обрано мову програмування Python — одну з найпоширеніших і найзручніших мов сучасності для розробки веб-застосунків, наукових, освітніх та комерційних програмних систем. Python відзначається лаконічністю синтаксису, високою читабельністю коду, широкою бібліотечною екосистемою та активною міжнародною спільнотою[11].

Окрім цього, Python є кросплатформною мовою з відкритим вихідним кодом, що дозволяє уникнути ліцензійних обмежень та значно спростити розгортання продукту на різних серверних середовищах[4]. Такий вибір забезпечує швидке прототипування функціоналу, зручність масштабування й подальшого супроводу системи, а також можливість легкого навчання нових учасників команди.

В якості серверного фреймворку для побудови бізнес-логіки застосунку використано Flask — сучасний мікрофреймворк для Python, який ідеально підходить для створення web-систем з індивідуальною архітектурою, гнучкими вимогами до логіки та унікальним інтерфейсом.

Flask надає мінімальний базовий функціонал (маршрутизація, шаблони, робота із сесіями), не нав’язує суворих структур і дозволяє інтегрувати сторонні розширення відповідно до конкретних потреб проєкту [2]. Головною перевагою такого підходу є можливість створювати масштабовані та легко модифіковані інформаційні системи — саме те, що потрібно для сучасної медичної IT-інфраструктури.

Вибір Flask додатково обґрунтовується якісною документацією, активною спільнотою, можливістю реалізації RESTful API, широкою підтримкою для інтеграції із зовнішніми бібліотеками (ORM, захист, аутентифікація, робота з PDF/Excel, email, тестування) та легкістю навчання для нових розробників.

Також важливо, що Flask гарно підходить для побудови багаторівневих,

рольових і безпечних веб-застосунків, що відповідають вимогам галузі охорони здоров’я.

2.2. Організація серверної логіки та архітектура проєкту

Серверна логіка веб-застосунку “Виписка електронних рецептів” побудована відповідно до принципів розділення відповідальності (Separation of Concerns) та модульності, що є базовими у сучасній розробці інформаційних систем. Це дозволяє не лише підвищити якість та читабельність коду, а й значно спростити тестування, масштабування та подальший супровід проєкту[14].

2.2.1 Архітектурний підхід та структура директорій

Проєкт структуровано у вигляді набору логічних модулів, кожен із яких відповідає за окрему функціональну частину системи. Основними складовими є:

1. Головний модуль застосунку (app/__init__.py): відповідає за створення Flask-додатку, підключення конфігурацій, ініціалізацію розширень (ORM, сесій, захисту), реєстрацію blueprint-ів для окремих підсистем (наприклад, doctor, patient, pharmacy).
2. Модулі моделей (app/models/): містять класи ORM SQLAlchemy, що відображають структуру таблиць у базі даних (User, Pharmacy, Medication, Visit, Prescription).
3. Модулі форм (app/forms/): визначають усі форми (реєстрації, логіну, виписки рецепта, запису на прийом, обробки візиту) з валідацією, валідаторами та спеціальною логікою для кожної ролі.
4. Маршрути та обробники (app/blueprints/): кожна роль або функціональний розділ має власний blueprint, що забезпечує ізоляцію логіки й полегшує масштабування (наприклад, doctor.py для маршрутизації лікарів).

5. Шаблони (app/templates/): містять Jinja2-шаблони для генерації динамічних сторінок, розбиті за розділами системи та наслідуванням від базового шаблону.
6. Статичні ресурси (app/static/): включають скомпільовані CSS-файли (Tailwind), зображення, іконки, кастомні JS-файли.

Крім того, структура доповнюється директоріями для конфігурацій, файлів міграцій бази даних та налаштувань безпеки.

2.2.2. Взаємодія компонентів

Взаємодія між модулями здійснюється через чітко визначені інтерфейси — API-функції, blueprint-и та шаблони. Кожен маршрут обробляється контролером, який отримує дані з форми, проводить валідацію, звертається до моделі та повертає відрендерений HTML-шаблон з результатами або повідомленням про помилку. Завдяки цьому забезпечується прозорість логіки, можливість легко додавати нові функції та змінювати бізнес-правила без порушення структури основного коду.

Такий підхід повністю відповідає сучасним практикам розробки веб-застосунків та забезпечує високу якість і надійність системи в експлуатації[14].

2.3. Вибір та організація бази даних, моделювання структури даних

У системах електронної охорони здоров’я ключову роль відіграє коректне, безпечне та структуроване зберігання медичних і облікових даних. Для цього у веб-застосунку “Виписка електронних рецептів” обрано реляційну систему управління базами даних PostgreSQL — одну з найбільш потужних, стабільних і гнучких СУБД з відкритим кодом[10]. PostgreSQL забезпечує повну підтримку ACID-транзакцій, багаторівневу ізоляцію даних, сучасні механізми безпеки та масштабованість, що особливо важливо для медичних інформаційних систем із великою кількістю користувачів і одночасних

запитів[10].

2.3.1 Об’єктно-реляційне відображення (ORM) з SQLAlchemy

Взаємодія Python-додатку з базою даних реалізована через бібліотеку SQLAlchemy, яка є найпопулярнішою ORM (Object-Relational Mapping) для Python[12]. ORM дозволяє описувати структуру таблиць БД як класи Python, що робить код прозорим, легким для тестування, розширення й підтримки.

Кожна сутність (користувач, аптека, візит, рецепт, препарат) моделюється окремим класом із набором атрибутів і методів для читання, запису, оновлення та видалення даних. Також ORM автоматично керує зовнішніми ключами та зв’язками між таблицями, забезпечуючи цілісність даних навіть при складних операціях[12].

2.3.2 Проєктування моделі даних

Структура БД створювалася відповідно до принципів нормалізації та стандартів медичних інформаційних систем.

Основні таблиці:

1. User (Користувачі): містить поля для зберігання унікального логіна, ПІБ, хешованого пароля, ролі (пацієнт, лікар, провізор), секретного коду лікаря, зв’язку з аптекою для провізорів.
2. Pharmacy (Аптеки): містить ідентифікатор та назву аптеки, потенційно — адресу, контакти.
3. Medication (Препарати): зберігає інформацію про ліки — назва, код, форма випуску.
4. Visit (Візити): фіксує взаємодію пацієнта з лікарем (дати, скарги, діагноз, рекомендації, зв’язок із пацієнтом і лікарем).
5. Prescription (Рецепти): основна таблиця, що містить усі деталі електронного рецепта — пацієнт, лікар, візит, призначений препарат, дозування, інструкції, алергії, діагноз, термін дії, статус, аптека для відпуску.

Кожна таблиця має унікальний ідентифікатор (id), а всі зв’язки між

сутностями виконуються через зовнішні ключі, що забезпечує цілісність та послідовність даних у системі.

2.3.3. Безпека та резервування даних

Система базується на сучасних практиках захисту БД: використання ролей і прав доступу в PostgreSQL, шифрування паролів користувачів, регулярне створення резервних копій (backup) та аудит змін. Це мінімізує ризики втрати, несанкціонованого доступу чи компрометації чутливої медичної інформації[10],[12].

Такий підхід до організації даних є запорукою стабільної, безпечної та легко масштабованої медичної системи, яка відповідає найкращим світовим практикам проєктування електронних інформаційних сервісів.

2.4. Реалізація форм, валідація та обробка даних користувача

Однією з найважливіших частин веб-застосунку “Виписка електронних рецептів” є коректна, безпечна й зручна для користувача робота з формами. Саме через форми відбувається взаємодія користувача із системою: реєстрація, логін, подання запитів на прийом, оформлення і отримання рецептів тощо.

Для цього у застосунку використовується бібліотека WTForms, яка забезпечує просту та надійну інтеграцію форм із Flask через розширення Flask-WTF[2],[14].

2.4.1 Класи форм і їх структура

Кожна форма визначається як окремий Python-клас, у якому перераховано всі необхідні поля (input, select, textarea, checkbox) із вбудованою валідацією:

1. LoginForm: містить поля для логіна і пароля, захищений CSRF-токеном, обробляє вхід у систему.
2. RegisterForm: включає поля для логіна, ПІБ, вибору ролі, аптеки (для провізора), секретного коду лікаря, пароля та

підтвердження пароля. Передбачає розгалужену валідацію: перевірка унікальності логіна, мінімальна довжина, співпадіння паролів, правильність коду лікаря для ролі “лікар”, коректний вибір аптеки для провізора.

3. PrescriptionForm: створена для лікаря, дозволяє обрати пацієнта, препарат, дозування, вказати інструкції, вагу, зріст, алергії, діагноз, термін дії рецепта, аптеку для відпуску, статус “лікарняної формули”.

4. VisitRequestForm: дає можливість пацієнту обрати лікаря, дату прийому, вказати скарги.

5. VisitProcessForm: дозволяє лікарю вказати діагноз, рекомендації та одразу виписати рецепт, якщо потрібно.

Усі поля мають відповідні валідатори (DataRequired, Length, EqualTo, Optional), що гарантує коректність даних ще до передачі їх на сервер.

2.4.2 Валідація та захист

Кожна форма автоматично отримує CSRF-токен, який захищає від атак типу Cross-Site Request Forgery. Додатково валідація на рівні сервера дозволяє уникнути потрапляння у базу невалідних або потенційно шкідливих даних, що особливо важливо для систем з обробкою медичної інформації[2],[14].

Валідація включає:

1. Перевірку наявності обов’язкових полів.
2. Перевірку формату й довжини текстових полів.
3. Логічні обмеження (наприклад, неможливість реєстрації лікаря без коректного секретного коду).
4. Валідацію співпадіння паролів.
5. Перевірку правильності дати, формату імені, вибору зі списків.
6. Відображення детальних повідомлень про помилки для користувача.

Завдяки такому підходу забезпечується висока якість і надійність введених даних, мінімізуються людський фактор і ймовірність помилок, що є критично важливим у медичних системах[14].

2.5. Система ролей користувачів, авторизація та контроль доступу

Надійна система авторизації та контроль доступу є ключовими для будь-якого сучасного веб-застосунку, особливо у медичних системах, де йдеться про захист персональних даних пацієнтів, медичну та облікову інформацію. У веб-застосунку “Виписка електронних рецептів” реалізовано багаторівневу модель ролей користувачів, яка забезпечує гнучке розмежування прав доступу відповідно до виконуваних функцій у системі[2],[14].

2.5.1 Рольова модель (RBAC)

У системі впроваджено класичний підхід Role-Based Access Control (RBAC), відповідно до якого кожному користувачу при реєстрації присвоюється одна з трьох ролей:

1. Пацієнт — може реєструватись, записуватися на прийом до лікаря, переглядати свої візити та рецепти, отримувати QR-коди для аптеки.
2. Лікар — має розширений доступ: може переглядати пацієнтів, обробляти візити, створювати та редагувати електронні рецепти, вносити діагнози та рекомендації.
3. Провізор — отримує доступ до списку рецептів, що надійшли до конкретної аптеки, може змінювати статус рецепта (“відпущено”, “відмова”), але не бачить медичної історії пацієнтів.

2.5.2 Технічна реалізація авторизації

Для управління сесіями, збереженням стану входу та перевіркою ролей використовується бібліотека Flask-Login, яка інтегрується з моделлю користувача та автоматично обробляє вхід/вихід, “запам’ятати мене”,

перенаправлення неавторизованих користувачів на сторінку логіну.

Контроль доступу реалізується за допомогою спеціальних декораторів (@login_required) і додаткових перевірок ролі у функціях-обробниках. Наприклад, створення рецепта доступне лише лікарям, а підтвердження відпуску — лише провізорам.

Вся логіка системи контролює не лише доступ до сторінок, а й до функціональних елементів інтерфейсу: відповідні кнопки, посилання й форми відображаються тільки тим користувачам, яким дозволено виконувати ці дії згідно з їхньою роллю.

2.5.3 Безпека зберігання паролів та захист персональних даних

Паролі користувачів зберігаються у базі виключно у вигляді стійких хешів, з використанням сучасних криптографічних алгоритмів (bcrypt, scrypt або подібних)[2]. Це унеможливлює отримання реального пароля навіть у разі компрометації бази даних.

Крім того, Flask-Login забезпечує захист сесій та використання захищених cookie-файлів для авторизації.

Таким чином, впроваджена система авторизації та ролей гарантує як відповідність вимогам безпеки, так і комфортну роботу для всіх типів користувачів веб-застосунку.

2.6. Проєктування інтерфейсу користувача: шаблони, стилі та принципи UI/UX

Однією з основних вимог до сучасних медичних веб-застосунків є наявність зручного, інтуїтивно зрозумілого та доступного для різних категорій користувачів інтерфейсу. Інтерфейс “Виписки електронних рецептів” побудовано із застосуванням шаблонізатора Jinja2, що є стандартом для проєктів на Flask і дозволяє ефективно організовувати динамічні сторінки, розділяти структуру, дизайн і бізнес-логіку[2],[14].

2.6.1 Організація шаблонів

Всі HTML-сторінки у застосунку створюються як Jinja2-шаблони. Головний шаблон (base.html) містить спільні для всіх сторінок компоненти: шапку, меню, підключення CSS та JavaScript, зону для flash-повідомлень.

Дочірні шаблони наслідують базовий, визначають унікальний контент для кожної сторінки (кабінет пацієнта, лікаря, провізора, форми реєстрації/логіну, списки візитів і рецептів, таблиці для аптек)[2].

Для повторюваних елементів використовуються фрагментовані шаблони (include): картки рецептів, елементи списків, блоки повідомлень, кнопки. Це дозволяє досягти максимального повторного використання коду та зручності оновлення дизайну.

2.6.2 Використання Tailwind CSS для стилізації

Дизайн системи реалізовано за допомогою Tailwind CSS — сучасного CSS-фреймворку, який базується на використанні утилітарних класів для кожного елемента розмітки[15]. Такий підхід дозволяє швидко змінювати вигляд елементів, легко підлаштовувати інтерфейс під корпоративні кольори чи вимоги до стилю медичних систем, забезпечувати адаптивність для різних типів пристроїв.

Ключові особливості Tailwind CSS у проекті:

1. Адаптивність**:** Сторінки і форми виглядають коректно як на ПК, так і на планшетах і смартфонах.
2. Єдиний стиль**:** Оформлення кнопок, таблиць, форм, алертів — єдиний підхід, що створює цілісний імідж системи.
3. Гнучкість кастомізації**:** Зміна кольорів, фонів, градієнтів, тіней, заокруглень, шрифтів виконується через короткі класи, не змінюючи основний CSS.
4. Оптимізація**:** Tailwind автоматично видаляє невикористані стилі під час фінальної збірки, забезпечуючи швидке завантаження навіть для користувачів із повільним інтернетом.

2.6.3 Вибір кольорів, шрифтів, градієнтів та адаптація під потреби медицини

Колірна гама, шрифти, розміри кнопок і полів підбиралися так, щоб забезпечити комфорт для всіх вікових груп користувачів і відповідати стандартам доступності. У проєкті використано читабельні беззарубкові шрифти, достатній контраст тексту й фону, чітко виділені інтерактивні елементи, а також графічні індикатори статусів (наприклад, кольори для активних/неактивних рецептів, повідомлень про помилки чи успіх).

Для важливих блоків використано градієнти та м’які відтінки, які не перевантажують зір, але виділяють ключову інформацію (наприклад, статус рецепта, кнопку “Виписати”, блок із порадами лікаря).

2.6.4 Доступність та дружність до користувача (UX)

Всі елементи інтерфейсу організовано так, щоб користувач мінімально витрачав час на пошук потрібної функції. Меню інтуїтивно логічне, поля мають підписи, у всіх формах реалізовані підказки та автоматичне відображення повідомлень про помилки.

Особлива увага приділена мобільній адаптації: усі поля, кнопки та списки мають достатній розмір для роботи на сенсорних пристроях, забезпечується прокручування великих таблиць та форм без втрати читабельності й зручності.

Таким чином, поєднання Jinja2, Tailwind CSS і сучасних принципів UI/UX забезпечує високу зручність роботи, естетичний вигляд і доступність системи для всіх категорій користувачів — лікарів, пацієнтів та провізорів[15].

2.7. Захист інформації, інформаційна безпека та валідація даних

Захист даних у медичних веб-застосунках — це не лише вимога закону, а й ключова умова довіри користувачів та коректної роботи системи. У проєкті “Виписка електронних рецептів” питання інформаційної безпеки вирішуються на декількох рівнях: від захисту облікових записів до комплексної перевірки

даних і запобігання несанкціонованому доступу[2],[6],[14].

2.7.1 Захист паролів та персональних даних

Усі паролі користувачів зберігаються у базі даних лише у вигляді криптографічних хешів із використанням сучасних алгоритмів (bcrypt/scrypt), що унеможливлює відновлення реального пароля навіть при потенційному витоку БД[2]. Жодні критичні персональні дані не зберігаються у відкритому вигляді або у вигляді, що дозволяє ідентифікувати користувача без його авторизації.

CSRF-захист та валідація форм

Для захисту від атак типу Cross-Site Request Forgery (CSRF) всі форми у системі оснащені унікальним токеном, який генерується для кожної сесії користувача і перевіряється під час кожного POST-запиту. Це гарантує, що дії у системі можуть виконуватися лише від імені автентифікованого користувача через легітимний інтерфейс системи[2],[14].

Перевірка і обмеження доступу (RBAC)

Всі маршрути з критичним або приватним функціоналом закриті декораторами @login_required та додатковою перевіркою ролі користувача. Наприклад, лікар не може переглядати “чужих” пацієнтів чи рецепти, а провізор — бачити медичні дані, що не стосуються відпуску в його аптеці. Це відповідає вимогам медичних стандартів і принципам захисту приватності[6],[14].

Захист від SQL-ін’єкцій та XSS

Використання ORM SQLAlchemy дозволяє формувати всі запити до БД через безпечні методи, що унеможливлює ін’єкції шкідливого коду (SQL Injection)[12].

У шаблонах автоматично екрануються всі динамічні дані, що захищає від атак типу XSS (Cross-Site Scripting).

2.7.2 Валідація даних на всіх рівнях

1. На клієнтському рівні: HTML5-атрибути (required, pattern), випадаючі списки, підказки користувачу.

2. На серверному рівні: WTForms-валідатори (DataRequired, Length, EqualTo, власні перевірки), перевірка унікальності логінів, коректності пароля, дати, логіки ролі тощо.

3. У базі даних: обмеження унікальності, зовнішні ключі, транзакції.

Захист сесій та cookie

1. Flask-Login використовує захищені cookie (HttpOnly, Secure), що знижує ризик викрадення сесії, навіть якщо користувач використовує спільний чи публічний пристрій.

Організаційні заходи та резервне копіювання

1. Передбачена можливість регулярного резервного копіювання бази даних та аудиту ключових операцій (наприклад, змінення ролей, виписки рецептів). Це дозволяє швидко відновити систему у разі збоїв чи несанкціонованих змін.

2.8. Можливості масштабування, супроводу та розвитку веб-застосунку

Розробка інформаційних систем у сфері медицини передбачає не лише вирішення поточних задач, а й закладення основ для подальшого розширення, масштабування та адаптації до нових законодавчих і технологічних вимог. Веб-застосунок “Виписка електронних рецептів” спроектований таким чином, щоб забезпечити максимально просту можливість розвитку та супроводу у майбутньому[14].

2.8.1 Архітектурна гнучкість

Використання Flask як мікрофреймворку дозволяє гнучко організовувати код, ділити його на окремі логічні модулі (blueprints), що спрощує додавання нових розділів і компонентів без порушення основної структури проєкту. Завдяки ORM SQLAlchemy додавання нових сутностей (наприклад, нових ролей, розширених записів про препарати чи аптеки)

виконується шляхом оновлення моделей та автоматичної міграції БД без складних ручних змін[12],[14].

2.8.2 Підтримка сучасних стандартів

Архітектура застосунку вже враховує вимоги ДСТУ ISO 17523:2016, що дозволяє безболісно інтегруватися із зовнішніми системами, наприклад, державними реєстрами, медичними інформаційними платформами або сервісами аналітики у майбутньому[6].

Вибір відкритого та добре документованого стека технологій (Python, Flask, PostgreSQL, Tailwind CSS) гарантує легкість пошуку розробників для супроводу системи та швидке оновлення компонентів у разі появи нових вимог чи функціоналу.

2.8.3 Масштабування і резервування

Завдяки використанню PostgreSQL як серверної БД система може масштабуватися горизонтально (через кластери, реплікацію, шардінг) або вертикально (збільшенням потужності серверів).

Архітектурна роздільність фронтенду, бекенду та БД дозволяє розміщувати компоненти на різних серверах чи в хмарних середовищах для підвищення надійності та доступності.

Організація резервного копіювання та аудит змін забезпечують можливість швидкого відновлення системи при збоях або несанкціонованих діях.

Легкість впровадження оновлень

Кодова база підтримує концепцію міграцій (Flask-Migrate), що дає змогу поступово змінювати структуру БД без втрати даних, а також гнучко змінювати інтерфейс, стилі й шаблони під вимоги користувачів.

2.8.4 Перспективи розвитку

Враховуючи сучасний стек та універсальність архітектури, система може бути розширена під потреби державних або приватних медичних установ, інтегруватися із зовнішніми eHealth-платформами, аналітичними сервісами, мобільними додатками.

Також можлива подальша автоматизація — наприклад, використання додаткових фільтрів, електронних підписів, систем нагадувань, багатомовності, розширеного аудиту дій користувачів, підтримки інтеграції з іншими електронними медичними сервісами.

Висновок до розділу 2

У даному розділі було детально розкрито засоби реалізації веб-застосунку “Виписка електронних рецептів” — від вибору мови програмування та серверного фреймворку до проєктування бази даних, моделювання ролей і організації сучасного інтерфейсу з використанням Jinja2 і Tailwind CSS.

Показано, як за допомогою бібліотек WTForms, Flask-Login, SQLAlchemy реалізовано надійну систему валідації, авторизації, збереження та захисту даних. Обґрунтовано підходи до організації структури проєкту, масштабування, супроводу та розвитку системи відповідно до вимог ДСТУ ISO 17523:2016 та сучасних практик медичної інформатики.

Обраний стек технологій забезпечує гнучкість, надійність і безпеку роботи із чутливою медичною інформацією, а також створює підґрунтя для масштабування і впровадження нових функцій у майбутньому. Реалізований підхід дозволяє формувати зручний, доступний і ефективний інструмент для всіх учасників медичного процесу — лікарів, пацієнтів та провізорів.

РОЗДІЛ 3
ОБҐРУНТУВАННЯ ВИБОРУ ПРОГРАМНОГО СЕРЕДОВИЩА ДЛЯ РЕАЛІЗАЦІЇ ПОСТАВЛЕНОЇ ЗАДАЧІ

У цьому розділі представлено обґрунтований вибір програмного середовища для реалізації веб-застосунку “Виписка електронних рецептів”. Розділ базується на аналізі існуючих рішень та аналогів, що були розглянуті в попередніх розділах, та враховує сучасні галузеві стандарти і вимоги до автоматизації медичних процесів.

Особлива увага приділяється вибору мови програмування, інструментів для побудови серверної та клієнтської логіки, бібліотек для організації бази даних, засобів захисту та валідації інформації, а також відповідності прийнятих рішень до стандартів Міністерства охорони здоров’я України і ДСТУ ISO 17523:2016. Також у розділі обґрунтовуються використані медичні форми, методи зберігання та захисту інформації, пояснюється вибір підходів до моделювання і реалізації функціоналу з огляду на специфіку сфери та практичні потреби системи електронних рецептів.

Виконано аналіз альтернатив, розкрито переваги й недоліки основних програмних середовищ, а також обґрунтовано прийнятий вибір як найбільш відповідний для виконання завдань, поставлених у дипломній роботі.

3.1. Аналіз існуючих рішень та обґрунтування вибору середовища розробки

При обґрунтуванні вибору програмного середовища для реалізації веб-застосунку “Виписка електронних рецептів” було проведено порівняльний аналіз існуючих вітчизняних і зарубіжних рішень у сфері електронного документообігу, медичних інформаційних систем та електронних рецептів. Такий аналіз дозволяє визначити переваги та недоліки різних підходів, а також

сформулювати власні вимоги до майбутньої системи на основі кращих практик[1],[9].

3.1.1 Аналіз аналогів

Серед найвідоміших платформ у цій сфері можна відзначити:

1. DrChrono (США) — комплексна EHR-система із підтримкою електронних рецептів, календарів, billing-сервісів, інтеграцій з мобільними пристроями[1].
2. OpenEMR — відкрите програмне забезпечення для ведення електронної медичної документації, що дозволяє налаштовувати модулі під локальні потреби, має розвинену спільноту підтримки, але часто потребує суттєвої кастомізації для національних стандартів[9].
3. Національна система eHealth (Україна) — державна платформа для управління даними про пацієнтів, лікарів, медичні послуги, включає електронний рецепт, але інтеграція зі сторонніми системами відбувається лише через визначені протоколи та із суворою сертифікацією[13].

Аналіз аналогів показав, що існуючі рішення або занадто громіздкі (DrChrono, OpenEMR), або потребують складних процедур впровадження й інтеграції, а також не завжди повністю відповідають українським державним стандартам (наприклад, ДСТУ ISO 17523:2016)[6]. Окрім того, ліцензійна політика закордонних платформ часто передбачає суттєві фінансові витрати на обслуговування, оновлення та підтримку[4].

3.1.2 Обґрунтування власної розробки

Враховуючи вищезгадані чинники, для реалізації задачі було прийнято рішення розробляти власний веб-застосунок “з нуля”, використовуючи сучасний open source-стек (Python, Flask, PostgreSQL, Tailwind CSS). Такий підхід дозволяє:

1. повністю адаптувати структуру даних, форми, сценарії роботи до національних нормативів МОЗ та ДСТУ;

2. реалізувати лише необхідний функціонал без надлишкових модулів, що спрощує впровадження та використання;
3. забезпечити високу гнучкість, масштабованість, легкість супроводу й майбутньої інтеграції із державними системами або локальними сервісами;
4. уникнути ліцензійних витрат та залежності від сторонніх компаній;
5. впроваджувати зміни та оновлення у найкоротші строки, враховуючи специфіку законодавства та побажання користувачів[4],[7].

Таким чином, обґрунтований вибір середовища розробки базується на прагненні створити ефективний, легкий у підтримці, безпечний та сучасний інструмент для роботи з електронними рецептами, максимально адаптований до українського ринку та реальних потреб медичних працівників і пацієнтів.

3.2. Обґрунтування вибору стеку технологій, мови програмування, бібліотек та засобів зберігання даних

Після аналізу аналогічних систем та визначення потреб майбутнього веб-застосунку, було обґрунтовано вибір стеку технологій, що дозволяє досягти балансу між гнучкістю розробки, відповідністю стандартам, зручністю супроводу та безпекою даних. Опис вибору подано нижче.

3.2.1 Вибір мови програмування та серверного фреймворку

Обрано мову Python як одну з найпопулярніших, універсальних і зручних для швидкої розробки веб-застосунків мов програмування[11]. Головними перевагами Python для цього проєкту є:

1. висока читабельність і зрозумілість коду, що пришвидшує розробку та спрощує командну роботу;

2. потужна бібліотечна екосистема, зокрема для роботи з вебом, базами даних, безпекою, тестуванням та інтеграціями;
3. широка підтримка open source-інструментів;
4. відсутність ліцензійних обмежень[4],[11].

Як серверний фреймворк вибрано Flask — сучасний мікрофреймворк для Python, що відзначається мінімалізмом, гнучкістю та легкістю інтеграції із зовнішніми бібліотеками[2]. Flask дає змогу самостійно визначати архітектуру проєкту, підбирати оптимальні модулі для безпеки, авторизації, обробки форм тощо, що особливо цінно для спеціалізованих галузевих рішень.

3.2.2 Організація бази даних та ORM

Для зберігання структурованих даних обрано реляційну СУБД PostgreSQL, яка має такі переваги:

1. підтримка ACID-транзакцій і багаторівневої цілісності даних;
2. розвинені можливості масштабування, резервування, реплікації;
3. відкритий код, відсутність ліцензійних платежів;
4. широкий досвід використання в медичних системах у світі[10].

Для взаємодії з БД використовується ORM-бібліотека SQLAlchemy[12], яка дозволяє описувати структуру таблиць як Python-класи, автоматично керувати зв’язками, міграціями, уникати SQL-ін’єкцій та підвищувати прозорість коду.

3.2.3 Бібліотеки для форм, безпеки та валідації

1. WTForms/Flask-WTF — для опису, рендерингу й валідації форм із підтримкою CSRF-захисту[2],[14];
2. Flask-Login — для організації авторизації, управління сесіями, контролю доступу за ролями користувачів;

3. bcrypt/scrypt — для надійного хешування паролів користувачів[2].

Засоби організації інтерфейсу

Для побудови сучасного, адаптивного і доступного інтерфейсу вибрано:

1. Jinja2 — шаблонізатор для динамічного HTML;
2. Tailwind CSS — сучасний CSS-фреймворк для швидкої стилізації, що дозволяє реалізувати корпоративний стиль, адаптивність та доступність навіть на мобільних пристроях[15].

3.2.4 Відповідність стандартам і нормативам

Весь функціонал (структура форм, полів, валідація даних, зберігання рецептів і ролей) реалізовано відповідно до ДСТУ ISO 17523:2016 та рекомендацій МОЗ щодо електронних медичних записів і рецептів[6],[8].

Таким чином, обраний стек технологій гарантує відповідність галузевим стандартам, гнучкість розвитку, легкість супроводу та високу безпеку для зберігання та обробки персональних і медичних даних.

3.3. Обґрунтування використання медичних форм МОЗ та відповідності стандартам

Важливою особливістю проєкту “Виписка електронних рецептів” є точна відповідність вимогам Міністерства охорони здоров’я України (МОЗ) та державним стандартам щодо структури електронних рецептів, оформлення медичних записів і збереження медичної інформації. Це є обов’язковою умовою для впровадження подібних систем у практику охорони здоров’я та гарантує юридичну значимість створюваних документів[6],[8].

3.3.1 Обґрунтування форм і полів

Усі форми, які використовуються у веб-застосунку — від реєстрації до виписки рецепта, побудовані на основі затверджених МОЗ медичних форм і

рекомендацій ДСТУ ISO 17523:2016 щодо “електронного рецепта”. Це забезпечує такі переваги:

1. Уніфікація і стандартизація**:** Кожне поле у формі (ПІБ, дата народження, діагноз, призначення, дані про лікаря, аптеку, препарат, термін дії рецепта) відповідає обов’язковим атрибутам державних і міжнародних стандартів електронної медицини[6].
2. Юридична сила документів**:** Електронні рецепти, що створюються через систему, мають усі необхідні реквізити для використання в офіційному документообігу, для відпуску ліків у аптеках та перевірки контролюючими органами.
3. Зручність інтеграції: Оскільки структура форм уніфікована, система може без труднощів інтегруватися із зовнішніми платформами (eHealth, аптеки, аналітичні сервіси), передавати чи приймати дані у стандартному форматі.

Логіка розміщення та використання форм

1. На етапі реєстрації форми відповідають вимогам щодо збору персональних і професійних даних (наприклад, перевірка секретного коду лікаря, обов’язковий вибір аптеки для провізора).
2. Форма виписки рецепта лікарем містить усі ключові атрибути, визначені ДСТУ, — дані пацієнта, лікаря, діагноз, перелік і дозування призначених препаратів, інструкції, інформацію про аптеку, термін дії рецепта, ознаку “лікарняної формули” та додаткові поля для алергій і особливих випадків[6].
3. Форма для провізора дозволяє працювати з рецептом у рамках, встановлених МОЗ — лише змінювати статус рецепта (наприклад, “відпущено”), але не вносити зміни у медичні дані.

3.3.2 отримання та аудит стандартів

Усі зміни структури форм, полів чи бізнес-логіки здійснюються з урахуванням актуальних рекомендацій МОЗ, ДСТУ та нормативних

документів, що дозволяє гарантувати актуальність і легітимність системи у разі змін законодавства.

Таким чином, застосування медичних форм МОЗ та відповідність ДСТУ ISO 17523:2016 є ключовою запорукою ефективності, правової сили і масштабованості розробленої системи.

3.4. Аналіз і обґрунтування методів зберігання та захисту інформації

Зберігання та захист інформації — одна з найкритичніших складових у сфері електронної охорони здоров'я. Система "Виписка електронних рецептів" побудована з урахуванням кращих світових практик, вимог законодавства України та особливостей реляційних баз даних.

3.4.1 Методи зберігання інформації

Для надійного зберігання даних обрано реляційну систему управління базами даних PostgreSQL, яка широко використовується у медичних інформаційних системах по всьому світу завдяки:

1. Підтримці транзакційності (ACID)**:** це гарантує цілісність і відновлюваність даних навіть у разі збою апаратного чи програмного забезпечення;
2. Гнучким можливостям для резервного копіювання, масштабування і відновлення даних;
3. Розвиненим механізмам контролю доступу до таблиць, полів і навіть окремих записів;
4. Можливості створення зовнішніх ключів, що забезпечують структурну цілісність між сутностями (пацієнт — візит — рецепт — аптека тощо)[10],[12].

Дані організовано відповідно до принципів нормалізації, що мінімізує дублювання і забезпечує логічну цілісність усіх записів.

3.4.2 Захист даних і відповідність вимогам безпеки

У системі реалізовано комплекс заходів для захисту інформації:

1. Хешування паролів: усі паролі користувачів зберігаються лише у вигляді стійких хешів із застосуванням сучасних криптографічних алгоритмів (наприклад, bcrypt), що унеможливлює отримання реального пароля навіть при компрометації бази даних[2].
2. CSRF-захист: кожна форма містить унікальний токен для попередження атак типу Cross-Site Request Forgery, що особливо важливо для дій із високим ризиком (реєстрація, логін, зміна статусу рецепта)[14].
3. Розмежування доступу: для кожного користувача чітко визначено рівень доступу відповідно до ролі (пацієнт, лікар, провізор). Реалізовано перевірку прав на рівні кожної функції, маршруту та навіть шаблону сторінки.
4. Захист сесій: використання захищених cookie-файлів, автоматичне завершення сесії після певного періоду неактивності.
5. Використання ORM: усі запити до бази даних виконуються через ORM SQLAlchemy, що унеможливлює SQL-ін’єкції, оскільки розробник не пише “сирий” SQL-код[12].
6. Валідація даних: застосування WTForms для багаторівневої перевірки введених користувачем даних на клієнтському та серверному рівні.

3.4.3 Резервне копіювання і аудит

Передбачено можливість регулярного резервного копіювання всієї бази даних для запобігання втратам у разі збою, а також аудит ключових дій (наприклад, виписка, відпуск рецепта), що дозволяє відстежувати всі критичні зміни у системі та швидко реагувати на потенційні загрози.

Відповідність галузевим стандартам

Система спроєктована з урахуванням ДСТУ ISO 17523:2016, а також рекомендацій МОЗ щодо захисту персональних даних у медичних інформаційних системах, що гарантує легітимність і безпеку впровадження на практиці[6],[8].

Висновок до розділу 3

У цьому розділі було проведено ґрунтовний аналіз існуючих програмних рішень у сфері електронних медичних записів, зокрема електронних рецептів, а також виконано обґрунтований вибір програмного середовища для реалізації власного веб-застосунку “Виписка електронних рецептів”.

Порівняння з провідними закордонними та вітчизняними аналогами засвідчило доцільність створення власної системи на основі open source-стеку, що дозволяє забезпечити відповідність національним стандартам, високу гнучкість, мінімальні витрати та легкість масштабування[1],[2],[4],[6],[9],[10].

Обґрунтовано вибір таких основних інструментів і бібліотек, як Python, Flask, PostgreSQL, SQLAlchemy, WTForms, Tailwind CSS — з урахуванням їхніх можливостей для ефективної розробки, гнучкості інтеграції, надійності та безпеки. Окремо розглянуто організацію структури даних і форм згідно з вимогами МОЗ та ДСТУ ISO 17523:2016, що гарантує юридичну значимість електронних рецептів і зручність обміну даними із зовнішніми системами[6],[8].

Зроблено акцент на забезпеченні захисту персональних і медичних даних: використання криптографічного хешування, CSRF-захисту, багаторівневої валідації, обмеження доступу на основі ролей, резервного копіювання та аудиту дій користувачів.

Усі прийняті рішення спрямовані на те, щоб розроблений веб-застосунок був не лише функціональним, а й безпечним, надійним, зручним для лікарів, пацієнтів і провізорів, а також готовим до подальшого розвитку й інтеграції з іншими компонентами електронної системи охорони здоров’я України.

РОЗДІЛ 4
ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ. ПРОГРАМНА РЕАЛІЗАЦІЯ ТА МЕТОДИКА РОБОТИ ВЕБ-ЗАСТОСУНКУ ДЛЯ ВИПИСКИ ЕЛЕКТРОННИХ РЕЦЕПТІВ

У цьому розділі наведено детальний опис реалізації веб-застосунку «Виписка електронних рецептів». Розкрито основні елементи інтерфейсу, їхню функціональність та логіку обробки дій користувачів. Представлено архітектуру клієнт-серверної взаємодії, реалізовану за допомогою фреймворку Flask та мови програмування Python.

Розглянуто процеси реєстрації та автентифікації користувачів (лікарів і пацієнтів), захищене збереження даних, створення, редагування та перегляд електронних рецептів. Особливу увагу приділено завантаженню довідкової інформації про лікарські засоби, їх пошуку та відображенню.

Описано реалізацію цифрового підпису, що забезпечує юридичну силу рецептів та захист даних. Наведено характеристику використаних інструментів — Flask, SQLAlchemy, HTML/CSS, Bootstrap/Tailwind, PostgreSQL — та проаналізовано їхні переваги у контексті проєкту.

4.1. Загальна архітектура системи

Розроблений веб-застосунок “Виписка електронних рецептів” реалізовано за класичною багаторівневою архітектурою “клієнт–сервер”, що поєднує гнучкість, масштабованість і простоту супроводу. Такий підхід дозволяє фізично та логічно відокремити відповідальність за збереження, обробку і відображення даних, забезпечує надійність та легкість розширення проєкту в майбутньому.

4.1.1 Компоненти архітектури

Клієнтська частина (Front-end)

Користувацький інтерфейс реалізовано у вигляді адаптивних HTML-сторінок, які формуються із використанням шаблонізатора Jinja2 у зв’язці з бібліотекою стилів Tailwind CSS (або Bootstrap на початкових етапах).

Основні задачі фронтенду:

1. відображення всіх інтерфейсних форм (авторизації, реєстрації, виписки рецепта, запису на прийом, перегляду рецептів, тощо);
2. забезпечення навігації між функціональними розділами (кабінети пацієнта, лікаря, провізора, аптеки);
3. інтерактивна взаємодія з користувачем (валидація, підказки, флеш-повідомлення, динамічне оновлення інтерфейсу);
4. забезпечення доступності для різних пристроїв (десктоп, планшет, смартфон) завдяки адаптивній верстці.

Серверна частина (Back-end)

Серцем серверної логіки є Python-фреймворк Flask.

Його головні ролі:

1. обробка HTTP-запитів від користувачів (реєстрація, логін, створення візиту, виписка рецепта, зміна статусу, пошук лікаря/препарату);
2. динамічне формування HTML-сторінок на основі шаблонів;
3. взаємодія з базою даних через ORM SQLAlchemy;
4. реалізація бізнес-логіки для кожної ролі (пацієнт, лікар, провізор);
5. обробка форм (Flask-WTF, валідація, CSRF-захист);
6. управління сесіями та авторизацією користувачів через Flask-Login;
7. відправлення email (за потреби), генерація QR-кодів, формування PDF.

Структура серверної частини побудована модульно:

1. models.py — містить моделі для всіх сутностей (User, Pharmacy, Medication, Visit, Prescription, Specialty, PrescriptionStatusLog);
2. routes/blueprints — кожен функціонал (реєстрація, логін, кабінет лікаря, пацієнта, аптеки) винесений у свій модуль для легкості масштабування;
3. forms.py — містить визначення всіх форм із валідаторами, що забезпечують захист та коректність даних.

Реляційна база даних (PostgreSQL)

База даних містить усі основні сутності: користувачі (User), аптеки (Pharmacy), препарати (Medication), спеціальності (Specialty), візити (Visit), електронні рецепти (Prescription), історія статусів рецепта (PrescriptionStatusLog).

Всі сутності чітко пов’язані між собою через зовнішні ключі, що забезпечує:

1. збереження зв’язків між пацієнтами, лікарями, візитами, рецептами, аптеками;
2. забезпечення цілісності, унікальності та структурованості даних;
3. простоту аналітики та генерації звітів.

4.1.2 Принципи взаємодії компонентів

1. Користувачі (пацієнт, лікар, провізор) взаємодіють із системою через браузер, отримуючи HTML-сторінки, сформовані на сервері відповідно до їхньої ролі.
2. Кожна дія користувача (наприклад, запис на прийом, створення рецепта, зміна статусу рецепта) породжує HTTP-запит, що надсилається серверу.
3. Серверна логіка аналізує отримані дані, валідує їх, взаємодіє з БД через ORM, після чого формує відповідь (сторінку, повідомлення

про помилку, зміни в базі).

4. Аутентифікація користувачів реалізована на основі Flask-Login із захищеними сесіями, а права доступу визначаються роллю користувача, збереженою в БД.

4.1.3 Захист, масштабованість і легкість супроводу

1. Всі чутливі дані (логіни, паролі, рецепти, персональні дані) зберігаються у зашифрованому вигляді або хешуються.

2. Завдяки ORM SQLAlchemy, структура моделей легко розширюється для додавання нових функцій (наприклад, нових ролей, додаткових атрибутів у рецепті чи візиті).

3. Архітектура підтримує масштабування (додавання нових серверів, підключення кешування, горизонтальне масштабування БД).

4. Використання міграцій (Alembic/Flask-Migrate) дозволяє безпечно оновлювати структуру БД без втрати даних.

4.1.4 Загальна схема

В результаті застосунок складається з чітко структурованих шарів:

1. Інтерфейс користувача (браузер, HTML+CSS+JS)

2. Серверна логіка (Flask, Python, маршрути, шаблони, обробка форм, логіка бізнес-процесів)

3. База даних (PostgreSQL, моделі SQLAlchemy)

4. Захисні механізми (Flask-Login, валідація, CSRF, шифрування паролів, контроль доступу)

Такий підхід забезпечує:

1. безпечне, централізоване зберігання медичних та персональних даних;

2. швидку реакцію системи на дії користувачів;

3. простоту підтримки й подальшого розширення

функціоналу;

4. відповідність вимогам ДСТУ ISO 17523:2016 та галузевим стандартам у сфері охорони здоров’я.

4.2. Реалізація інтерфейсу користувача

Користувач потрапляє на сторінку входу (рис. 4.1). після відкриття веб-сайту. Інтерфейс авторизації містить два поля — логін та пароль. Для входу в систему користувач повинен ввести дані, які були задані під час реєстрації. Якщо обліковий запис ще не створено, під формою входу розміщено посилання, яке дозволяє перейти на сторінку реєстрації.

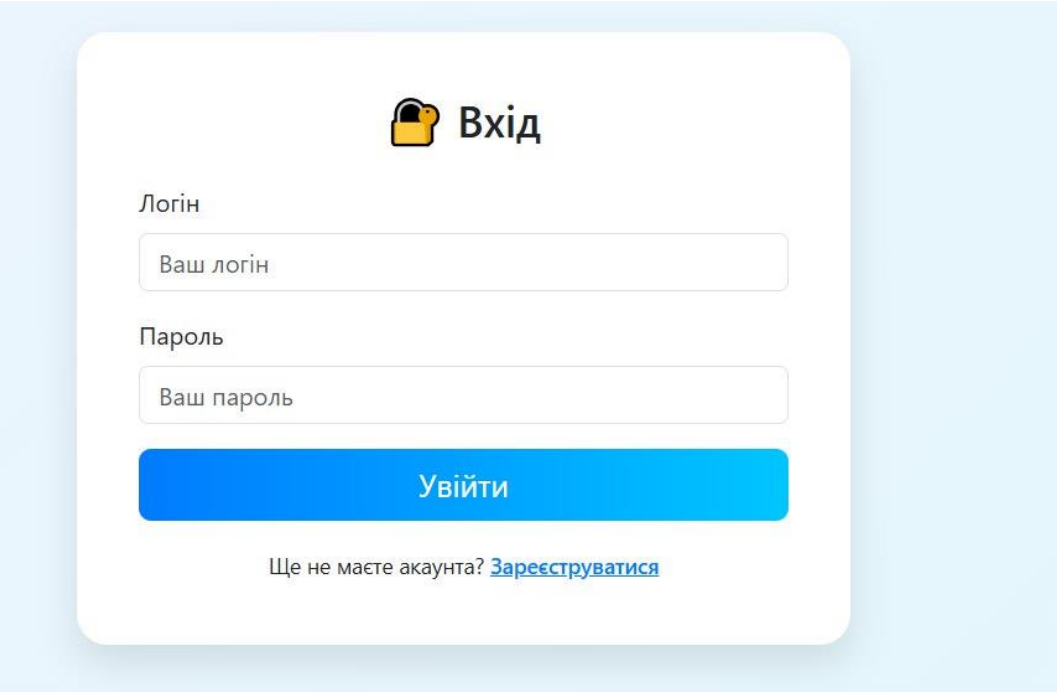

Рисунок 4.1 – UI авторизації

На сторінці реєстрації (рис.4.2) реалізовано функціонал для створення облікового запису двох типів користувачів — лікарів, фармацевтів та пацієнтів. Користувач самостійно обирає роль із випадаючого списку.

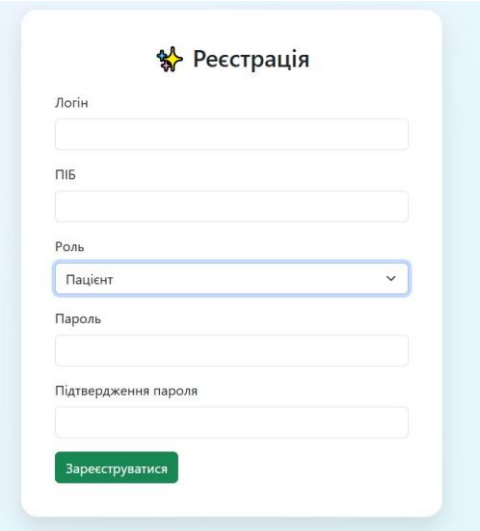

Рисунок 4.2 – UI реєстрації як пацієнта

У разі вибору ролі лікаря чи фармацевта з’являється додаткове поле для введення секретного коду (рис 4.2), який видається медичним закладом. Це дозволяє уникнути несанкціонованої реєстрації користувачів як лікарів і забезпечує базову перевірку достовірності.

Обов’язковими для заповнення є поля логіна, прізвища, імені та по батькові, пароля і підтвердження пароля.

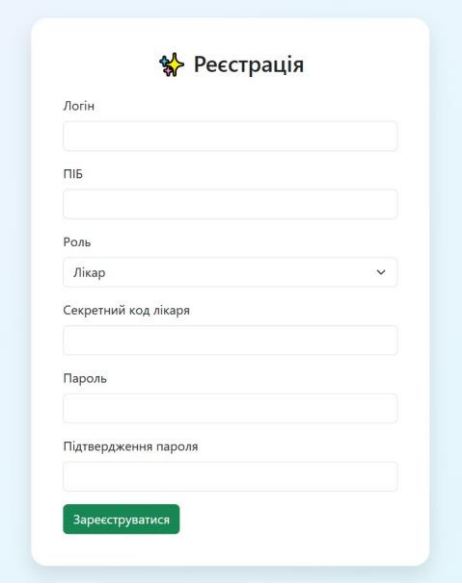

Рисунок 4.3 – UI реєстрації як лікаря

При помилці введення пароля або якщо паролі не співпадають, форма

виводить відповідне повідомлення. Після коректного заповнення форми користувач надсилає дані на сервер, де проводиться перевірка, збереження в базу даних та перенаправлення на форму входу.

Цей механізм дозволяє чітко розмежовувати функціонал для лікарів, фармацевтів та пацієнтів, забезпечуючи базову авторизацію користувачів системи, яка надалі використовується для ідентифікації в межах сесії, контролю доступу до сторінок та безпечної взаємодії з електронною медичною інформацією. Код генерується випадково й не передбачувано, що додає захисту нашій системі. Код Може бути виданий адміністратором системи. Також ця система є легко-масштабованою, у майбутньому можлива додача криптографічних елементів та авторизацію за допомогою файлового ключа цифрого підпису.

На рис. 4.4 представлено інтерфейс сторінки “Активні візити”, що доступна користувачу після авторизації в системі. Ця сторінка відображає перелік візитів пацієнта до лікаря, які наразі перебувають у статусі “активний”.

У випадку, якщо активних візитів у користувача немає, система виводить інформаційне повідомлення у вигляді окремого блока з текстом “Немає активних візитів.” Це дозволяє користувачу легко зорієнтуватися у власному розкладі та оперативно отримати інформацію про заплановані або поточні прийоми.

Оформлення сторінки виконане у фірмових кольорах застосунку з використанням помітного заголовку та делікатного фонового підсвічування для інформаційного блоку. Такий підхід підвищує зручність сприйняття інформації та відповідає принципам дружнього до користувача інтерфейсу.

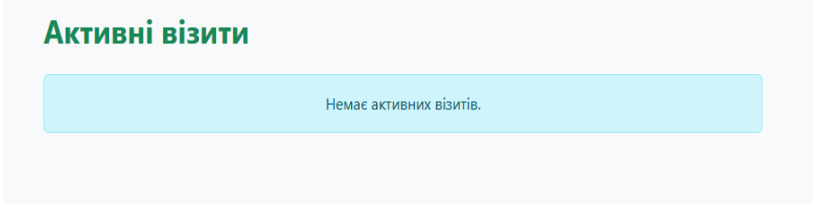

Рисунок 4.4 – Кабінет лікаря, коли відсутні заплановані візити

На наступних рисунках представлено основний інтерфейс кабінету пацієнта, що реалізований у вигляді вкладок для зручності навігації між двома основними функціональними розділами — “Записи на прийом” (рис. 4.5) та “Мої рецепти” (рис.4.6).

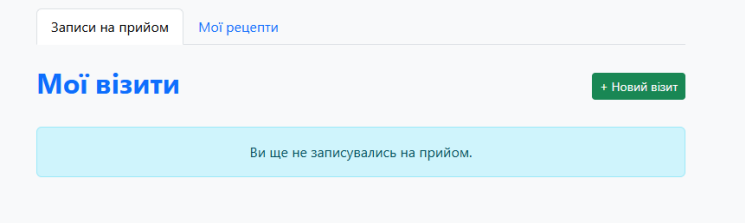

Рисунок 4.5 – Кабінет пацієнта, вкладка створення нового візиту

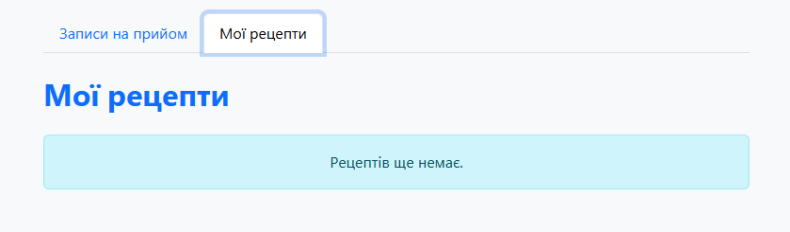

Рисунок 4.5 – Кабінет пацієнта, вкладка зі списком виписаних рецептів

На рис.4.6 зображено інтерфейс сторінки для співробітників аптеки у веб-застосунку “Виписка електронних рецептів”. У верхній частині розташовано фірмовий логотип та яскравий заголовок з назвою конкретної аптеки (“Аптека Довіра”), що підвищує пізнаваність та персоналізує сторінку для співробітників закладу.

Під заголовком наведено короткий опис функціоналу: “Перелік електронних рецептів, що потрібно відпустити”. Основна частина інтерфейсу складається з табличного представлення даних, де виводиться список електронних рецептів, які очікують на відпуск у даній аптеці.

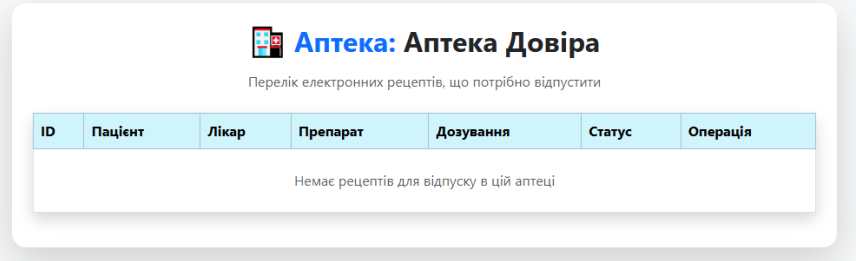

Рисунок 4.6 – Кабінет певної аптеки з таблицею рецептів, що надходять у цю аптеку

На скріншоті (рис.4.7) показано форму для запису на прийом до лікаря, яка доступна пацієнту у відповідному розділі системи.

Інтерфейс містить такі основні поля: Вибір лікаря (випадаючий список) — пацієнт обирає спеціаліста, до якого бажає записатися. Бажана дата — поле для вибору дати прийому, з інтегрованим календарем для зручності. Скарги — текстове поле, в якому пацієнт може детально описати свої скарги або симптоми.

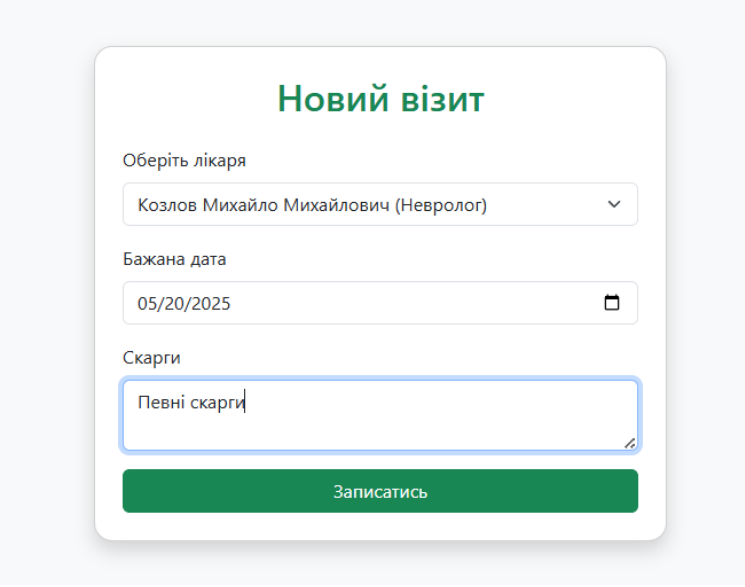

Рисунок 4.7 – Вигляд меню запису до лікаря на прийом

Кнопка “Записатись” підтверджує створення нового запиту на візит. Усі елементи форми логічно згруповані у центрованому блоці з м’якою тінню для покращення сприйняття.

Рис. 4.8 демонструє відображення вже створеного візиту в особистому кабінеті пацієнта. У верхній частині сторінки виводиться повідомлення про успішне створення запиту: “Запит на прийом створено!”

Вкладка “Мої візити” відображає список поточних записів користувача.

Інформаційна картка містить: дату, статус (“Очікує прийому”), спеціальність лікаря, скарги пацієнта, а також поля для діагнозу та рекомендацій (поки порожні, до моменту консультації).

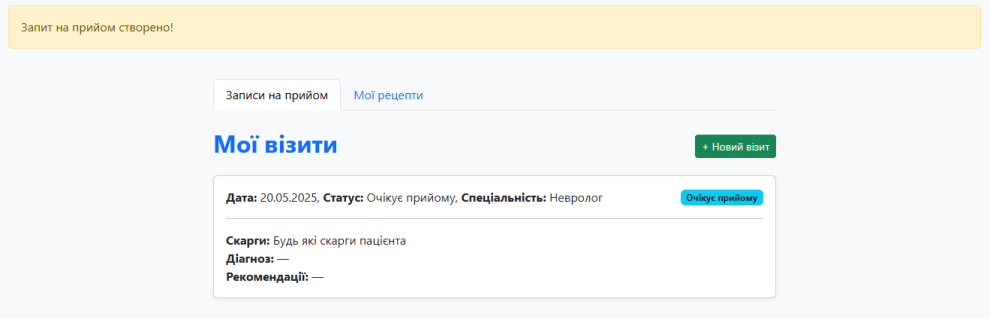

Рисунок 4.8 – Створено запис на прийом

Коли пацієнт створив запис на прийом, він з’являється у особистому кабінеті лікаря з короткою інформацією про скарги, дату та ім’я пацієнта. Також кнопка для початку прийому (рис. 4.9).

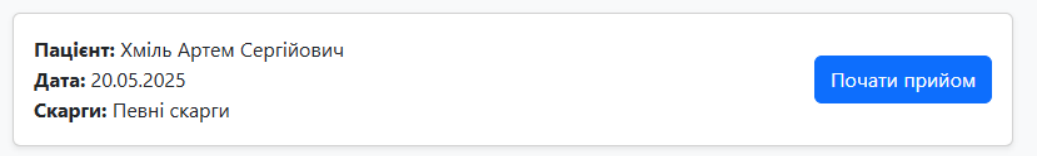

Рисунок 4.9 – Створено запис на прийом

По результатам обслідування лікар може написати діагноз, рекомендації з випискою рецепту або без виписки електронного рецепту.

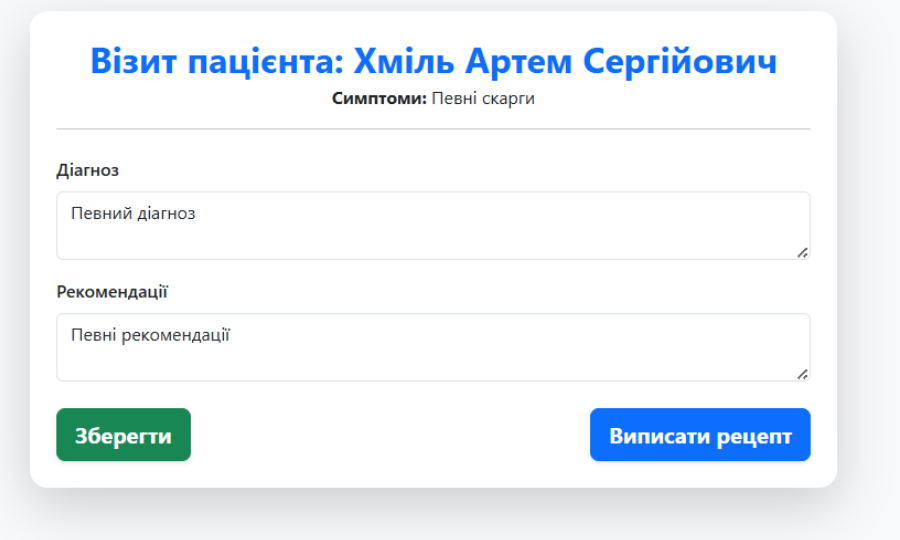

Рисунок 4.10 – Меню прийому з двома полями та двома кнопками

Якщо натиснуто було збереження, відповідно рецепту не буде, якщо виписка рецепту, лікаря буде переведено на сторінку для виписки рецепта.

На даному рис. 4.11 представлено інтерфейс форми створення нового електронного рецепта, яку заповнює лікар під час прийому пацієнта. У верхній частині сторінки чітко вказано ПІБ пацієнта, що мінімізує ймовірність помилки під час виписки рецепта.

Основні поля форми:

1. Препарат — вибір лікарського засобу з випадаючого списку.
2. Дозування — конкретна доза, яка повинна бути призначена пацієнту.
3. Інструкції — поле для додаткових рекомендацій або особливих вказівок до застосування.
4. Вага (кг), Зріст (см) — індивідуальні параметри пацієнта, що дозволяють точніше підібрати дозування, особливо для дітей та пацієнтів з особливими потребами.
5. Алергії — важлива інформація для уникнення небажаних реакцій на лікарські засоби.
6. Діагноз/Показання — підстави для призначення даного препарату згідно з медичними стандартами.

7. Аптека — вибір конкретної аптеки, у якій буде відпускатися рецепт.

8. Дійсний до — дата, до якої рецепт є актуальним.

9. Лікарняна формула — опціональна позначка для спеціальних випадків.

Усі ці поля відповідають вимогам національного стандарту ДСТУ ISO 17523:2016 “Електронний рецепт. Вимоги до функціональних характеристик” та відображають стандартну структуру даних електронного рецепта, забезпечуючи відповідність законодавчим та організаційним вимогам до медичних інформаційних систем в Україні.

Кнопка “Виписати рецепт” розташована у нижній частині форми, що дозволяє лікарю швидко завершити процедуру внесення даних.

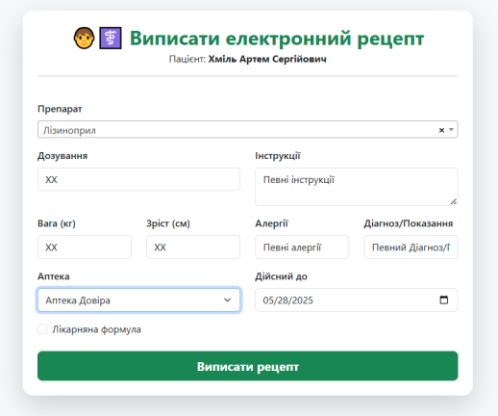

Рисунок 4.11 – Меню виписки електронного рецепту

На даному рис. 4.12 показано, як виписаний електронний рецепт відображається у системі для співробітників аптеки. Сторінка містить перелік електронних рецептів, які потребують відпуску у конкретній аптеці (“Аптека Довіра”).

У табличному вигляді представлено основні дані по кожному рецепту:

1. ID — унікальний номер рецепта.

2. Пацієнт — прізвище, ім’я та по батькові пацієнта.

3. Лікар — ПІБ лікаря, що виписав рецепт.
4. Препарат — назва призначеного лікарського засобу.
5. Дозування — рекомендована доза.
6. Статус — поточний стан рецепта (у даному прикладі “в обробці”).
7. Операція — випадаючий список дій для співробітника аптеки (наприклад, “В обр” — в обробці, “Відпущено” (рис.4.12), “Скасовано”) та кнопка “Оновити” для зміни статусу рецепта.

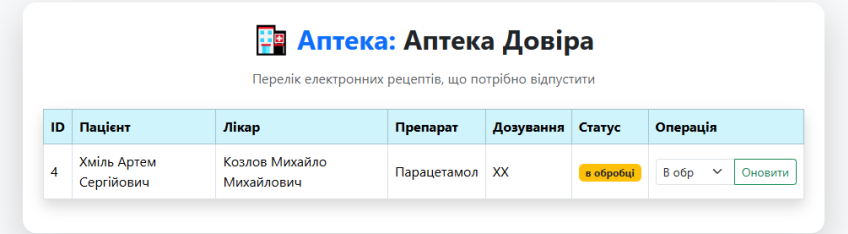

Рисунок 4.12 – Вигляд електронного рецепту який прийшов у аптеку (в обробці)

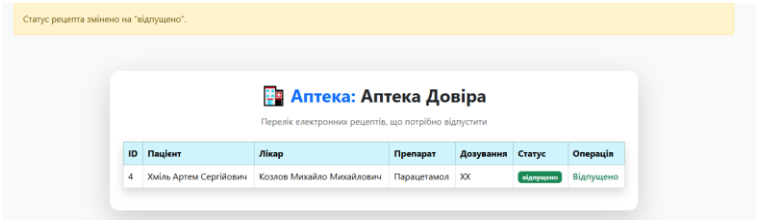

Рисунок 4.13 – Зміна статусу на “Відпущено”

На рис. 4.14 представлено вигляд сторінки “Мої рецепти” в особистому кабінеті пацієнта після успішного відпуску електронного рецепта в аптеці. Відображення рецепта містить такі ключові елементи:

1. Дата виписки — дата створення рецепта лікарем.
2. Лікар — ПІБ лікаря, який виписав рецепт.
3. Препарат — назва призначеного лікарського засобу.
4. Дозування — рекомендована доза.
5. Статус — візуальний індикатор, що відображає стан рецепта

(“відпущено”).

6. Код рецепта — унікальний ідентифікатор (наприклад, fa06089a), який використовується для ідентифікації рецепта в аптеці.

7. Інструкція — додаткові рекомендації лікаря щодо застосування препарату.

8. QR-код — генерується автоматично для кожного рецепта, що дозволяє пацієнту швидко пред’явити рецепт у аптеці з мобільного пристрою або паперового носія.

Додатково вказано інструкцію для пацієнта: “*Покажіть цей QR-код в аптеці для отримання препарату або продиктуйте код рецепта: fa06089a*”. Це підвищує зручність та безпеку отримання ліків, мінімізує ймовірність помилок при відпуску препарату та спрощує процедуру для користувача.

Призначення:

Сторінка дозволяє пацієнту зберігати, переглядати та пред’являти свої електронні рецепти для отримання лікарських засобів, що повністю відповідає вимогам сучасних стандартів електронної медицини та практиці обігу електронних рецептів в Україні.

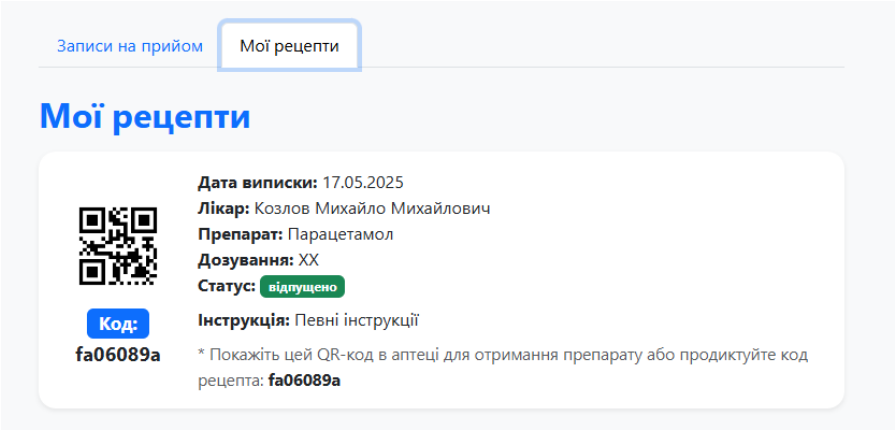

Рисунок 4.14 – Вигляд сторінки “Мої рецепти” з виписаним рецептом у кабінеті пацієнта

Після натискання на QR-код або сам код перевірки, користувач

перенаправляється на окрему сторінку (рис. 4.15), де відкривається візуалізація рецепта у форматі для друку або збереження. На цій сторінці передбачена кнопка для завантаження рецепта у вигляді PDF-файлу, що дозволяє зручно зберігати або пересилати документ.

Рисунок 4.15 – Сторінка для завантаження рецепту

Також у веб-застосунку реалізована окрема сторінка для перевірки дійсності рецепта за унікальним кодом (рис. 4.16). Користувач може ввести код рецепта у відповідне поле, після чого система здійснює пошук у базі даних.

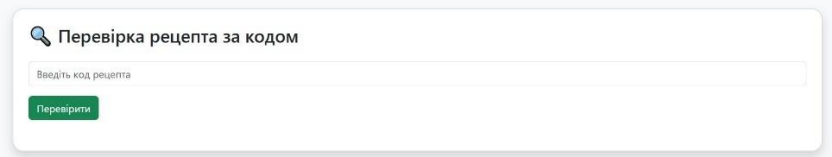

Рисунок 4.16 – Перевірка дійсності рецепта

У разі успішного знаходження, відображається коротка інформація про рецепт (рис. 4.17): ім’я пацієнта, ПІБ лікаря, дата створення, перелік препаратів та статус рецепта (активний, виконаний, анульований). Це дозволяє фармацевтам, пацієнтам та лікарям швидко перевірити справжність рецепта без необхідності входу в особистий кабінет.

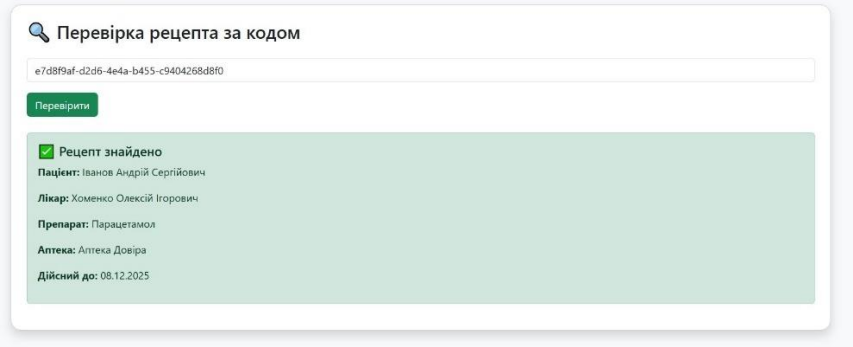

Рисунок 4.17 – Приклад дійсного рецепта

4.3 Робота з базою даних

Веб-застосунок “Виписка електронних рецептів” побудований за принципами багаторівневої архітектури, де окрему важливу роль відіграє взаємодія з базою даних. Усі ключові дані, пов’язані з користувачами, візитами, рецептами, аптеками та їхньою взаємодією, зберігаються у реляційній базі даних (PostgreSQL). Для організації доступу до даних та обробки форм в системі використовується бібліотека Flask-WTF, яка забезпечує зручний механізм роботи з формами, валідації введених даних та їх подальшого збереження у БД.

Основні компоненти роботи з формами:

У програмному коді наведено визначення низки форм для різних сценаріїв взаємодії користувачів із застосунком:

1. LoginForm (Форма авторизації)
   - username — Логін користувача (обов’язкове поле)
   - password — Пароль (обов’язкове поле)
   - submit — Кнопка для входу в систему
2. RegisterForm (Форма реєстрації)
   - username — Логін користувача (обов’язкове поле, 3–25 символів)
   - name — Прізвище, ім’я та по батькові (обов’язкове поле, 2–

50 символів)

- role — Роль користувача (лікар, пацієнт, провізор; обов’язкове поле)
- pharmacy — Аптека (для провізорів, вибір із довідника)
- doctor_secret — Секретний код лікаря (для підтвердження статусу лікаря)
- password — Пароль (обов’язкове поле, не менше 4 символів)
- confirm — Підтвердження пароля (обов’язкове поле, має співпадати з паролем)
- submit — Кнопка реєстрації

3. PrescriptionForm (Форма виписки рецепта)
   - medication — Препарат (вибір із довідника, обов’язкове поле)
   - dosage — Дозування (обов’язкове поле)
   - instructions — Інструкції до застосування
   - weight — Вага пацієнта, кг
   - height — Зріст пацієнта, см
   - allergies — Алергії
   - diagnosis — Діагноз/Показання
   - pharmacy — Аптека (для відпуску, обов’язкове поле)
   - valid_until — Дата, до якої дійсний рецепт (обов’язкове поле)
   - is_hospital_formula — Ознака “лікарняна формула”
   - submit — Кнопка для створення рецепта
4. VisitRequestForm (Форма запиту на візит)
   - doctor — Вибір лікаря (обов’язкове поле)
   - date_requested — Бажана дата прийому (обов’язкове поле)
   - symptoms — Скарги пацієнта (обов’язкове поле)
   - submit — Кнопка для запису на прийом
5. VisitProcessForm (Форма обробки візиту)

- diagnosis — Діагноз
- recommendations — Рекомендації
- create_prescription — Прапорець “Виписати рецепт”
- submit — Кнопка завершення візиту

Всі поля мають перевірку правильності введення (DataRequired, довжина, співпадіння пароля тощо), що забезпечує валідність і захист збережених у базі даних дани Веб-застосунок “Виписка електронних рецептів” побудований за принципами багаторівневої архітектури, де окрему важливу роль відіграє взаємодія з базою даних. Усі ключові дані, пов’язані з користувачами, візитами, рецептами, аптеками та їхньою взаємодією, зберігаються у реляційній базі даних (PostgreSQL). Для організації доступу до даних та обробки форм в системі використовується бібліотека Flask-WTF, яка забезпечує зручний механізм роботи з формами, валідації введених даних та їх подальшого збереження у БД.

4.3.1 Основні компоненти роботи з формами

У програмному коді наведено визначення низки форм для різних сценаріїв взаємодії користувачів із застосунком. Нижче наведена ERD діаграма (рис. 4.18)

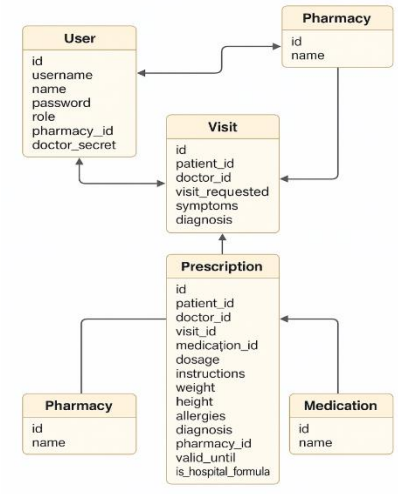

Рисунок 4.18 – ERD-діаграма з сутностями та атрибутами.

4.4 Завантаження та обробка довідкових файлів

На етапі початкової ініціалізації проєкту було реалізовано логіку автоматичного наповнення бази даних ключовими сутностями, необхідними для роботи веб-застосунку. Зокрема, у базу були додані спеціальності лікарів, зареєстровані профілі лікарів і пацієнтів, перелік лікарських засобів та список аптек.

Для кожної з категорій дані були додані вручну або за допомогою коду ініціалізації. Так, для лікарів і пацієнтів було визначено такі атрибути, як ім’я, стать, дата народження, спеціалізація (для лікарів), наявні алергії та протипоказання (для пацієнтів). Було також реалізовано збереження хешованого пароля для кожного користувача з використанням generate_password_hash, що забезпечує базову безпеку при авторизації.

Сутність "ліки" (модель Medication) наповнювалась як вручну, так і шляхом масового імпорту з зовнішнього CSV-файлу — реєстру лікарських засобів. Для зручності реалізовано функцію, яка зчитує CSV-файл, автоматично визначає назву препарату та додає його до бази даних, якщо він відсутній. Таким чином, вдалося швидко імпортувати понад 500 найменувань лікарських засобів.

Імпорт здійснювався з офіційного джерела — Державного реєстру лікарських засобів України , який містить актуальну інформацію про дозволені до застосування препарати в Україні.

Цей підхід дозволив забезпечити повноцінне функціонування веб-застосунку одразу після розгортання, надавши користувачам змогу вибирати з наявного списку лікарів, аптек та ліків при формуванні електронного рецепта.

4.5 Розрахунок економічного ефекту

Метою даного підрозділу є дослідження економічної доцільності впровадження програмного продукту — веб-застосунку для електронної виписки рецептів. Розробка цього рішення дозволяє автоматизувати процес призначення лікарських засобів, підвищити швидкість обслуговування пацієнтів, знизити ймовірність помилок та забезпечити зручну взаємодію між лікарем, пацієнтом і аптекою. Для обґрунтування вибору використано функціонально-вартісний аналіз (ФВА), що дозволяє порівняти різні підходи та визначити найефективніший з економічної та технічної точок зору.

Після ідентифікації ключових функцій програмного продукту була побудована морфологічна карта, що відображає основні можливі комбінації реалізації (рис. 4.19).

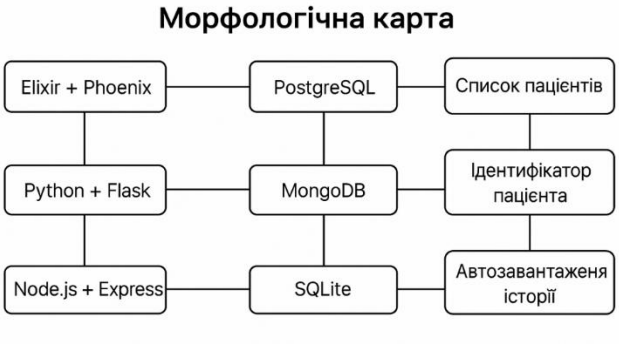

Рисунок 4.19 – Морфологічна карта можливих рішень для розробки веб-застосунку

Було розглянуто три основні компоненти: вибір технологічного стеку (F_1), вибір системи управління базами даних (F_2) та реалізація користувацького інтерфейсу (F_3). Для кожної функції проаналізовано альтернативи (табл. 4.1).

Таблиця 4.1 – Порівняння основних рішень

Функція	Варіант	Переваги	Недоліки
F_1	Elixir + Phoenix	Висока надійність, масштабованість	Складність в освоєнні
	Python + Flask	Простота, велика спільнота	Менш ефективний для великих систем
	Node.js + Express	Асинхронна обробка, багато пакетів	Складність масштабування, оптимізації
F_2	PostgreSQL	Реляційна модель, транзакції	Необхідність у строгих схемах
	MongoDB	Гнучкість, JSON-структури	Проблеми з цілісністю даних
	SQLite	Легкість, швидке прототипування	Не підходить для великих систем
F_3	Список пацієнтів	Зручність для лікаря	Може перевантажуватись при великій кількості
	Ідентифікатор	Швидкий доступ до записів	Необхідно знати ідентифікатор
	Автозавантаження	Автоматизація, зменшення навантаження	Потребує актуальності даних

Для подальшого аналізу було проведено розрахунок техніко-економічних показників кожного з варіантів, включаючи такі параметри:

1. Потенційний об’єм програмного коду,
2. Обсяг оперативної пам’яті (RAM),
3. Обсяг постійної пам’яті (FLASH),
4. Середній час виконання одного циклу програми.

Всі ці характеристики оцінювалися за трибальною шкалою (гірший, середній, кращий), що дозволило сформувати матрицю експертних оцінок і визначити вагомість кожного параметра для цільового проекту. Розрахунки продемонстрували, що оптимальним є поєднання Python + Flask, PostgreSQL та автозавантаження історії пацієнта.

Техніко-економічний розрахунок

У рамках розрахунків було враховано:

1. Витрати на розробку та програмування,
2. Собівартість машинного часу,

3. Зарплатний фонд розробників,
4. Вартість обладнання та електроенергії,
5. Накладні витрати.

Загальна трудомісткість проєкту складає 852 людино-години. Розрахунок заробітної плати та всіх експлуатаційних витрат дозволив встановити собівартість розробки ПЗ у розмірі 411 439 грн. Для визначення ефективності також був обчислений коефіцієнт техніко-економічного рівня, який становить 0,00004958.

Висновок до розділу 4

У ході розробки веб-застосунку “Виписка електронних рецептів” було реалізовано сучасну, надійну та масштабовану структуру бази даних, яка забезпечує зберігання й опрацювання всіх ключових даних про користувачів, медичні візити, електронні рецепти, лікарські засоби та аптеки. Архітектура бази даних відповідає принципам нормалізації, що дозволяє уникати дублювання інформації, зберігати цілісність даних і гарантувати захист від несанкціонованого доступу.

Використання реляційної моделі з чіткими зв’язками між сутностями — пацієнтами, лікарями, провізорами, візитами, рецептами, аптеками й препаратами — дало змогу досягти високої гнучкості у роботі системи та спростити розширення її функціональності в майбутньому. Структура таблиць розроблена таким чином, щоб максимально відповідати вимогам ДСТУ ISO 17523:2016 та специфіці української медичної сфери.

Валідація даних на рівні форм (Flask-WTF) додатково підвищила безпеку й достовірність інформації, що потрапляє до бази даних, а інтеграція ORM SQLAlchemy забезпечила ефективну взаємодію із СУБД та мінімізувала ризики технічних помилок.

Загалом, впроваджена модель роботи з базою даних у системі дозволяє ефективно реалізувати всі основні бізнес-процеси проєкту: реєстрацію та авторизацію користувачів різних ролей, створення і збереження медичних записів та рецептів, обробку запитів пацієнтів, автоматичний облік і контроль електронних рецептів, а також їхній подальший відпуск в аптеках. Це створює надійну основу для подальшого розвитку, масштабування й інтеграції системи у сучасну цифрову інфраструктуру охорони здоров’я.

РОЗДІЛ 5
УЗАГАЛЬНЕННЯ РЕЗУЛЬТАТІВ РОБОТИ

У цьому розділі підсумовано основні результати виконаної роботи, проведено узагальнення та порівняльний аналіз створеного веб-застосунку для виписки електронних рецептів із сучасними аналогами, наведено особливості розробленої системи, визначено її переваги й недоліки. Також розглянуто доцільність вибору архітектурних та програмних рішень, що лягли в основу реалізації застосунку, а також обговорено отримані результати з урахуванням практичних і технологічних аспектів.

5.1 Короткий опис змісту розділу

У даному розділі здійснюється комплексний аналіз функціональності розробленого програмного продукту, порівняння його з існуючими вітчизняними та закордонними аналогами, а також обговорюються одержані результати впровадження. Особливу увагу приділено оцінці унікальних рис веб-застосунку, визначенню основних конкурентних переваг і потенційних напрямів для подальшого вдосконалення системи. Окрім порівняльного аналізу, наведено практичні аспекти експлуатації, адаптації під потреби користувачів та відповідності стандартам інформаційної безпеки.

5.2 Аналіз обраного програмного застосунку

Розроблений веб-застосунок для виписки електронних рецептів був створений із врахуванням сучасних вимог до цифрової медицини, на основі глибокого аналізу аналогічних систем. У процесі роботи детально досліджувалися такі існуючі рішення, як державна система eHealth (Україна), міжнародна платформа OpenEMR, комерційний продукт DrChrono (США) та локальні системи типу MedicsDocAssistant.

5.3 Основна унікальність (родзинка) розробленої системи

Головною відмінністю створеного застосунку є його орієнтація на максимальну простоту та зручність взаємодії для всіх учасників процесу — лікаря, пацієнта й провізора. Система спеціально розроблена відповідно до вітчизняних стандартів (ДСТУ ISO 17523:2016) та вимог безпеки персональних даних, що дає змогу інтегрувати її із сучасними державними реєстрами та впроваджувати у будь-яких медичних закладах. Окремим фактором конкурентоспроможності є можливість швидкої генерації унікального QR-коду для кожного рецепта, що значно підвищує безпеку, спрощує ідентифікацію в аптеці та зменшує ймовірність людської помилки.

5.4 Порівняльний аналіз із аналогами

Державна система eHealth забезпечує централізований облік рецептів, але є жорстко регламентованою, з обмеженими можливостями кастомізації та залежністю від сторонніх МІС. OpenEMR надає широкий спектр інструментів, але потребує складної інтеграції та технічної підтримки, а інтерфейс системи часто є застарілим для звичайних користувачів. DrChrono демонструє високий рівень автоматизації, але не підтримує українську мову, місцеві стандарти і є недоступним для більшості вітчизняних закладів через вартість та закриту архітектуру. Локальні рішення на зразок MedicsDocAssistant зазвичай орієнтовані лише на стаціонарні мережі, не мають веб-інтерфейсу, обмежують можливість віддаленої роботи та інтеграції з зовнішніми платформами.

Переваги розробленого застосунку:

1. Повна локалізація і відповідність національним нормативам та протоколам захисту персональних даних.
2. Сучасний, інтуїтивно зрозумілий веб-інтерфейс для всіх ролей користувачів.

3. Генерація QR-кодів та унікальних ідентифікаторів для кожного рецепта.
4. Гнучкість у зміні статусу рецепта та контролі його обігу між лікарем, пацієнтом та аптекою.
5. Можливість швидкої перевірки дійсності рецепта без авторизації.
6. Відкритість до розширення функціоналу та інтеграції з державними реєстрами.
7. Адаптивність під різні пристрої (ПК, планшети, смартфони).

Недоліки системи:

1. Відсутність повної інтеграції з національною платформою eHealth (на етапі розробки).

5.5 Обговорення одержаних результатів

Проведене тестування й апробація веб-застосунку показали його практичну цінність у сучасних умовах цифрової медицини. Система забезпечує високу швидкість формування та обробки електронних рецептів, мінімізує ймовірність помилок завдяки автоматичній валідації введених даних, а також спрощує обмін інформацією між лікарями, аптеками та пацієнтами. Генерація QR-кодів та підтримка унікальних ідентифікаторів робить процес видачі ліків більш безпечним і прозорим.

Розроблений програмний продукт не лише відповідає сучасним стандартам захисту даних, а й має потенціал до масштабування й подальшого вдосконалення. Його впровадження може стати основою для підвищення довіри до електронних медичних систем, зниження паперового навантаження на медперсонал, а також підвищення доступності якісної медичної допомоги для пацієнтів.

ЗАГАЛЬНІ ВИСНОВКИ

Результати роботи

У результаті виконання дипломної роботи було розроблено, обґрунтовано та реалізовано веб-застосунок для автоматизованої виписки електронних рецептів, що відповідає сучасним стандартам цифрової медицини та вимогам інформаційної безпеки. В рамках дослідження здійснено аналіз існуючих національних і зарубіжних програмних рішень у сфері електронної охорони здоров'я, виявлено їх переваги й недоліки, що дозволило сформулювати власні вимоги до архітектури, функціоналу та захисту даних майбутнього продукту.

Здійснено обґрунтований вибір середовища розробки, мови програмування та технологічного стеку (Python, Flask, PostgreSQL, Tailwind CSS, WTForms), а також засобів валідації, авторизації та контролю доступу. Розроблено багаторівневу систему ролей користувачів, реалізовано модульну структуру проєкту з розмежуванням доступу для лікарів, пацієнтів та провізорів. Особливу увагу приділено питанням інформаційної безпеки: впроваджено захищене зберігання паролів, механізми CSRF-захисту, резервне копіювання даних, валідацію на всіх рівнях системи.

Результатом стало створення прототипу веб-застосунку, який забезпечує зручний інтерфейс для всіх категорій користувачів, автоматизує ключові етапи виписки та обліку рецептів, дозволяє зберігати і обробляти чутливу інформацію згідно з чинними нормативними документами (зокрема, ДСТУ ISO 17523:2016), а також має потенціал масштабування та інтеграції з державними eHealth-системами.

Висновки

Підсумовуючи виконану роботу, можна зазначити, що поставлені у дипломній роботі цілі та завдання було повністю досягнуто. Науково-

методичні підходи, використані в процесі розробки, дозволили створити надійний, гнучкий і безпечний веб-застосунок, адаптований до потреб вітчизняної медицини та особливостей нормативно-правового поля України. Створена система сприяє підвищенню ефективності взаємодії між лікарями, пацієнтами та аптеками, мінімізує ризики помилок і втрати даних, забезпечує зручність роботи в єдиному цифровому середовищі.

Практична цінність роботи підтверджується можливістю впровадження розробленого продукту у медичних закладах різного рівня для підвищення якості медичного обслуговування, скорочення паперового документообігу, прискорення обігу лікарських засобів та підвищення прозорості всіх процесів, пов’язаних із випискою рецептів. Дипломна робота виконана згідно із затвердженим завданням, відповідає освітньо-професійній програмі спеціальності 122 «Комп’ютерні науки», а отримані результати можуть бути використані як основа для подальших наукових досліджень і практичного впровадження в галузі цифрової охорони здоров’я.

У разі необхідності результати та матеріали роботи можуть бути апробовані на профільних конференціях, опубліковані у фахових виданнях, або представлені у формі доповідей, а також використані для модернізації інформаційних систем у медичних установах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) ДСТУ ISO 17523:2016. Електронний рецепт. Вимоги до функціональних характеристик. – [Електронний ресурс]. – Режим доступу: https://zakon.isu.net.ua/doc/?uid=1137.1545.0

2) Про затвердження порядків надання домедичної допомоги особам при невідкладних станах [Електронний ресурс] // МІНІСТЕРСТВО ОХОРОНИ ЗДОРОВ'Я УКРАЇНИ. – 2022. – Режим доступу: https://zakon.rada.gov.ua/laws/show/z0356-22#n941

3) Національна служба здоров'я України (НСЗУ). Електронна система охорони здоров'я eHealth [Електронний ресурс]. – Режим доступу: https://ehealth.gov.ua/

4) DrChrono - EHR, Practice Management, Medical Billing, RCM, Telehealth [Електронний ресурс]. – Режим доступу: https://www.drchrono.com/

5) OpenEMR – Open Source Electronic Health Records and Medical Practice Management [Електронний ресурс]. – Режим доступу: https://www.open-emr.org/

6) MedicsDocAssistant [Офіційний сайт]. – Режим доступу: https://medicsdocassistant.com/

7) Tailwind CSS Documentation [Електронний ресурс]. – Режим доступу: https://tailwindcss.com/docs/

8) Flask Documentation [Електронний ресурс]. – Режим доступу: https://flask.palletsprojects.com/

9) SQLAlchemy Documentation [Електронний ресурс]. – Режим доступу: https://docs.sqlalchemy.org/

10) PostgreSQL Documentation [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/

11) WTForms Documentation [Електронний ресурс]. – Режим доступу: https://wtforms.readthedocs.io/

12) Flask-Login Documentation [Електронний ресурс]. – Режим доступу: https://flask-login.readthedocs.io/

13) КНП «Електронна система охорони здоров'я» [Електронний ресурс] – Режим доступу: https://ehealth.gov.ua/about/

14) Крамаренко В.В., Атамась І.О. Інформаційні технології в медицині: навчальний посібник. – К.: НУХТ, 2021. – 211 с. – [PDF]: https://naurok.com.ua/posibnik-informaciyni-tehnologii-v-medici-ni-184821.html

15) Security best practices for Flask web applications [Електронний ресурс]. – Режим доступу: https://flask.palletsprojects.com/en/2.2.x/security/

16) ISO/IEC 27001:2013 – Information technology — Security techniques — Information security management systems [Електронний ресурс]. – Режим доступу: https://www.iso.org/standard/54534.html

17) HL7 FHIR – Fast Healthcare Interoperability Resources [Електронний ресурс]. – Режим доступу: https://www.hl7.org/fhir/

18) Державний реєстр лікарських засобів України [Електронний ресурс]. – Режим доступу: https://www.drlz.com.ua/

19) Міністерство охорони здоров'я України. Роз'яснення щодо впровадження електронних рецептів [Електронний ресурс]. – Режим доступу: https://moz.gov.ua/article/ministry-mandates/rozjasnennja-schodo-elektronnogo-recepta

20) User experience (UX) for healthcare applications [Електронний ресурс]. – Режим доступу: https://uxdesign.cc/ux-for-healthcare-1559eae29774

21) Bootstrap Documentation [Електронний ресурс]. – Режим доступу: https://getbootstrap.com/docs/

22) Варламова С.А., Бондаренко І.М. Основи інформаційної безпеки: навч. посібник. — Харків: ХНУРЕ, 2020. — 186 с. – [PDF]: https://repository.kpi.kharkov.ua/bitstream/KhPI-Press/52513/1/posibnik_2020.pdf

23) OAuth 2.0 and OpenID Connect in Healthcare [Електронний ресурс]. – Режим доступу: https://healthcare-secure.com/oauth2-openid-health/

24) Глушков В.М., Прус Ю.І., Чорний В.М. Основи проєктування баз даних. — К.: КНЕУ, 2022. — 246 с. – [PDF]: https://kneu.edu.ua/userfiles/biblioteka/pdf/osnovi_proektuvannja_bd.pdf

25) Міністерство охорони здоров’я України. Перелік затверджених медичних форм [Електронний ресурс]. – Режим доступу: https://moz.gov.ua/documents-perelik