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

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

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка веб-застосунку для супроводу процесів донорства крові

Виконав: студент IV курсу, групи СН-42
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

_____ Мартинюк П.Л.
(підпис) (прізвище та ініціали)

Керівник _____ Гладьо Ю.Б.
(підпис) (прізвище та ініціали)

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

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

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль
2024

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

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

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
« » червня 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

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

Студенту Мартинюку Петру Любомировичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка веб-застосунку для супроводу процесів донорства крові

Керівник роботи Гладьо Юрій Богданович, кандидат технічних наук, доцент кафедри ММ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*29*» *квітня* 2024 року № *4/7-470*

2. Термін подання студентом завершеної роботи 25 червня 2024р.

3. Вихідні дані до роботи Літературні та інтернет джерела щодо донорства крові, веб-застосунків для супроводу процесів донорства крові

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1. Постановка задачі та формування вимог до веб-застосунку. 1.1. Аналіз предметної області. 1.2. Формування вимог до веб-застосунку. 1.3. Пошук актантів та варіантів використання веб-застосунку для супроводу процесів донорства крові. 1.3.1. Пошук актантів. 1.3.2. Варіанти використання веб-застосунку. 1.4. Опис ключових варіантів використання веб-застосунку. 1.5. Вибір середовища розробки. 1.5.1. Мова програмування Python. 1.5.2. Фреймворк Django. 1.5.3. Мова розмітки HTML. 1.5.4. Каскадні таблиці стилів CSS. 1.5.5.Системи управління базою даних SQLite. 1.5.6. Текстовий редактор. 1.6. Висновок до першого розділу. 2. Проектування веб-застосунку для супроводу процесів донорства крові. 2.1. Моделювання архітектури веб-застосунку. 2.2. Проектування структури веб-застосунку. 2.3. Проектування бази даних. 2.3.1. Аналіз логічних процесів. 2.3.2. Розробка логічних правил. 2.3.3. Розробка структури таблиць. 2.4. Проектування візуального вигляду веб-застосунку. 2.5. Висновок до другого розділу. 3. Реалізація веб-застосунку для супроводу процесів донорства крові. 4. Безпека життєдіяльності та основи охорони праці. Висновки. Перелік джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Титульний слайд. 2. Мета кваліфікаційної роботи. 3. Формування вимог. 4. Діаграма використання для неавторизованого користувача. 5. Діаграма використання для авторизованого користувача "Донор". 6. Діаграма використання для авторизованого користувача "Пацієнт". 7. Діграма використання для адміністратора. 8. Вибір засобів розробки. 9. Архітектура веб-застосунку. 10. Модель бази даних. 11. Вигляд головної сторінки веб-застосунку. 12. Вигляд сторінки адміністратора. 13. Висновки. 14. Кінцевий слайд.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Сенчишин В.С., доцент кафедри МТ		

7. Дата видачі завдання 29 січня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	30.01.2024	
2.	Підбір джерел про донорство крові, засоби розробки	31.01.2024-03.02.2024	
3.	Опрацювання джерел про веб-застосунок для супро-	04.02.2024-06.02.2024	
	воду процесів донорства крові		
4.	Проектування та розробка веб-застосунку для супро-	07.02.2024-11.02.2024	
	воду процесів донорства крові		
5.	Оформлення розділу «Постановка задачі та формува	03.06.2024-05.06.2024	
	ння вимог до веб-застосунку»		
6.	Оформлення розділу «Проектування веб-застосунку	06.06.2024-08.06.2024	
	для супроводу процесів донорства крові»		
6.	Оформлення розділу «Реалізація веб-застосунку для	09.06.2024-11.06.2024	
	супроводу процесів донорства крові»		
7.	Виконання завдання до підрозділу «Безпека життєдіяльності»	12.06.2024-13.06.2024	
8.	Виконання завдання до підрозділу «Основи охорони	14.06.2024-15.06.2024	
	праці»		
9.	Оформлення кваліфікаційної роботи	16.06.2024-17.06.2024	
10.	Нормоконтроль	18.06.2024-19.06.2024	
11.	Перевірка на плагіат	20.06.2024	
12.	Попередній захист кваліфікаційної роботи	21.06.2024	
13.	Захист кваліфікаційної роботи	26.06.2024	

Студент _____ Мартинюк П.Л.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Гладьо Ю.Б.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка веб-застосунку для супроводу процесів донорства крові // Кваліфікаційна робота освітнього рівня «Бакалавр» // Мартинюк Петро Любомирович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СН-42 // Тернопіль, 2024 // С. 65, рис. – 28 , табл. – 19, слайди – 14, додат. – 2, бібліогр. – 39.

Ключові слова: кров, донор, донація, переливання крові, python, django, html, css, база даних, sqlite, веб-застосунок.

Кваліфікаційна робота присвячена розробці веб-застосунку для супроводу процесів донорства крові

В першому розділі кваліфікаційної роботи проаналізовано предметну область, поставлено та сформовано вимоги до веб-застосунку. Було обрано мову програмування Python, мову розмітки HTML та CSS для стилізації сторінок. Текстовим редактором було обрано PyCharm.

В другому розділі кваліфікаційної роботи проведено проектування веб-застосунку. Змодельовано архітектуру веб-застосунку, спроектовано структури веб-застосунку. Спроектовано та описано базу даних і візуальний вигляд веб-застосунку

В третьому розділі кваліфікаційної роботи описано реалізацію веб-застосунку для супроводу процесів донорства крові. Структуризовано файлу систему проекту, а саме: описано і структуровано каталоги веб-застосунку; описано файлову структуру веб-застосунку. Розроблено та детально описано користувацький інтерфейс та інтерфейс адміністратора.

У розділі безпеки життєдіяльності та основи охорони праці було проаналізовано і описано питання домедичної допомоги при кровотечі та вплив кольору на покращення умов праці та підвищення продуктивності виробництва.

ANNOTATION

Development of a web application to support blood donation processes // Qualification work of the educational level "Bachelor" // Martyniuk Petro Liubomyrovych // Ternopil Ivan Pulyu National Technical University, Computer and Information Systems and Software Engineering Faculty, Computer Sciences Department, group SN-42 // Ternopil, 2024 // P. 65 , fig. - 28, tabl. - 19, slides - 14, annexes. – 2, references - 39.

Keywords: blood, donor, donation, blood transfusion, python, django, html, css, database, sqlite, web application.

The qualification work is devoted to the development of a web application to support blood donation processes

In the first chapter of the qualification work, the subject area is analyzed, requirements for the web application are set and formed. The Python programming language, HTML markup language and CSS for styling pages were chosen. PyCharm was chosen as a text editor.

In the second section of the qualification work, the web application was designed. The architecture of the web application was modeled, the structures of the web application were designed. Designed and described the database and visual appearance of the web application

The third chapter of the qualification work describes the implementation of a web application to support blood donation processes. The project file system is structured, directories and file structure of the web application is described. The interface are developed and described in detail.

In the section on life safety and the basics of labor protection, the issues of pre-hospital care in case of bleeding and the impact of color on improving working conditions and increasing production productivity were analyzed and described.

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ І ТЕРМІНІВ

HTML (Hypertext Markup Language) – це стандартна мова розмітки, яка використовується для створення веб-сторінок.

CSS (Cascading Style Sheets) – каскадні таблиці стилів

Актанти – це підхід до аналізу текстів та комунікації. Вони використовуються для опису взаємодій між різними суб'єктами або об'єктами дії.

SQLite – це популярна, легка та автономна система управління базами даних на мові SQL, яка широко використовується для різних додатків

ЗМІСТ

ВСТУП

Актуальність теми: У сучасному світі питання донорства крові стає все більш важливим і актуальним для забезпечення потреб медичної галузі та підтримки громадського здоров'я. Постійне зростання популяції, розвиток медичних технологій та збільшення кількості травматичних випадків та хвороб потребує постійного поповнення запасів крові. Однак, існують виклики, пов'язані з пошуком донорів та організацією процесу донорства, що часто ускладнюють роботу медичних установ та можуть впливати на якість надання медичних послуг.

Мета і задачі дослідження: Метою цієї дипломної роботи є створення зручного, інтуїтивно зрозумілого та доступного веб-застосунку для донорів та пацієнтів. Цей сайт має спростити та упорядкувати весь процес донорства крові, забезпечуючи ефективну взаємодію між усіма учасниками цього процесу, зокрема медичними установами, донорами та пацієнтами.

Практичне значення одержаних результатів: Розробка цього веб-сайту матиме велике практичне значення для всіх зацікавлених сторін. Для донорів вона забезпечить зручний та простий спосіб реєстрації та управління їхніми даними. Для медичних установ – ефективне управління запасами крові та організаціям процесів здачі та переливання крові. Для громадськості – збільшення доступності донорської допомоги та збереження більшої кількості життів.

РОЗДІЛ 1. ПОСПОСТАНОВКА ЗАДАЧІ ТА ФОРМУВАННЯ ВИМОГ ДО ВЕБ-ЗАСТОСУНКУ

1.1 Аналіз предметної області

Завданням кваліфікаційної роботи бакалавра є розробка веб-застосунку для супроводу процесів донорства крові. Донорство крові є життєво важливою складовою системи охорони здоров'я, оскільки забезпечує необхідні запаси крові для лікування пацієнтів, які потребують термінових переливань, операцій, або лікування важких захворювань. Відповідно до статистичних даних, потреба в крові та її компонентах постійно зростає, що вимагає ефективної організації процесів збору, обробки, зберігання та розподілу крові між медичними закладами.

Сучасні технології можуть значно покращити ефективність цих процесів, зменшити витрати часу та ресурсів, а також підвищити зручність для донорів та медичного персоналу. Веб-застосунки для супровідних процесів донорства крові мають потенціал стати ключовими інструментами в організації та управлінні донорськими кампаніями, відстеженні запасів крові та забезпеченні оперативного зв'язку між донорами та медичними установами.

Ринок веб-застосунків для медичних послуг постійно розширюється, привертаючи увагу як великих IT-компаній, так і спеціалізованих розробників. Існують такі відомі рішення, як Red Cross Blood Donor App, Blood4Life, та інші подібні сервіси, що пропонують різноманітні функції для підтримки донорства крові. Проте, попит на інноваційні та зручні у використанні веб-застосунки залишається високим, особливо в контексті покращення якості медичних послуг та розширення їх доступності для населення.

Веб-застосунок для супроводу процесів донорства крові повинен включати широкий спектр функцій, які б задовольняли потреби всіх учасників процесу: донорів, медичного персоналу, координаторів донорських кампаній та

керівників медичних установ. Зокрема, система має забезпечувати реєстрацію та облік донорів, планування та координацію донорських акцій, управління запасами крові та її компонентів, а також надання інформаційної підтримки та консультацій для донорів. Важливою функцією є також інтеграція з медичними системами та базами даних для забезпечення оперативного доступу до необхідної інформації та ефективної взаємодії між різними підрозділами системи охорони здоров'я.

Таким чином, розробка веб-застосунку для супровідних процесів донорства крові є актуальним та важливим завданням, яке сприятиме покращенню організації донорства, забезпеченню надійного постачання крові та її компонентів, а також підвищенню рівня медичних послуг в цілому.

1.2 Формування вимог до веб-застосунку

Веб-застосунок для супровідних процесів донорства крові повинен мати зручну платформу для користувачів, з можливістю швидкого доступу до необхідної інформації та функціональності. Застосунок повинен забезпечувати наступні можливості:

Для користувачів (донори та пацієнти):

Реєстрація та авторизація:

- Можливість створити обліковий запис як донор або пацієнт.
- Авторизація для доступу до персоналізованої інформації.

Подання заявок:

- Подання заявки на здачу крові (для донорів).
- Подання заявки на отримання крові (для пацієнтів).

Перегляд інформації:

- Доступ до інформації про заплановані донорські акції.
- Перегляд наявного каталогу необхідних компонентів крові.
- Перегляд історії власних заявок та процесів донорства.

Облік кількості крові:

- Інформація про наявність різних груп крові та їх компонентів у реальному часі.

Для адміністратора:

Панель керування:

- Вхід як адміністратор.
- Прийняття або відхилення заявок на здачу та переливання крові.
- Внесення нових записів про донорів та пацієнтів.
- Перегляд та редагування всіх створених акаунтів користувачів.
- Управління базою даних щодо кількості наявної крові та її компонентів.

Історія процесів:

- Перегляд історії всіх процесів донорства та переливання крові.
- Аналітичний дашборд.
- Доступ до аналітичних даних щодо процесів донорства, кількості крові, активності донорів та інших важливих показників.

Структура сайту та бази даних:

База даних SQLite:

- Зберігання інформації про користувачів (донори, пацієнти, адміністратори).
- Зберігання інформації про заявки на здачу та переливання крові.
- Облік кількості наявної крові та її компонентів.
- Історія всіх донорських процесів та акцій.

Функціональні компоненти:

- Вхід та реєстрація як донор і пацієнт.
- Вхід як адміністратор з розширеними можливостями управління.
- Управління заявками на здачу та переливання крові.
- Облік кількості крові та її компонентів у реальному часі.
- Перегляд історії всіх процесів донорства та переливання крові.

Ці вимоги забезпечать ефективну роботу веб-застосунку, задовольняючи потреби як користувачів, так і адміністраторів, а також сприятимуть покращенню організації та управління процесами донорства крові.

1.3 Пошук актантів та варіантів икористання веб-застосунку для супровідних процесів донорства крові

1.3.1 Пошук актантів

Актанти – це підхід до аналізу текстів та комунікації. Вони використовуються для опису взаємодій між різними суб'єктами або об'єктами дії. Основна мета актантів полягає в тому, щоб виділити різні аспекти комунікації та зрозуміти, як вони впливають один на одного.

Діаграма акторів у веб-застосунку для супровідних процесів донорства крові має наступний вигляд [1]:

- Неавторизований користувач.
- Донор.
- Пацієнт.
- Адміністратор.

Ця діаграма допомагає краще зрозуміти, як різні типи користувачів взаємодіють із системою та між собою, а також як вони сприяють досягненню основних цілей веб-застосунку.

На рисунку 1.1 зображено акторів розроблюваного веб-застосунку.

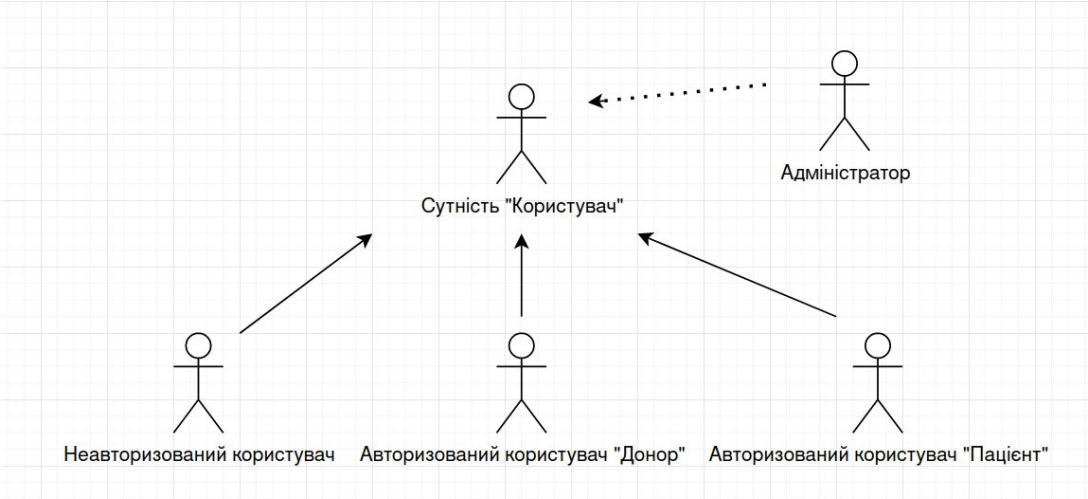

Рисунок 1.1 – Актори розробленого веб-застосунку

1.3.2 Варіанти використання веб-застосунку

Після побудови діаграми акторів слід сформулювати основні варіанти використання веб-застосунку [2]. Опис основних варіантів використання для неавторизованого користувача наведено у таблиці 1.1.

Таблиця 1.1 – Варіанти використання для неавторизованого користувача

Актор	**Найменування**	**Формулювання**
Неавторизований користувач	Реєстрація	Можливість для користувача отримати функціонал донора або пацієнта зареєструвавшись
	Авторизація	Можливість для користувача отримати функціонал донора або пацієнта авторизувавшись

Як бачимо варіантів використання у неавторизованого користувача тільки два, це або зареєструватись, або авторизуватись, що достатньо мало в порівняння з авторизованими, основні варіанти використання одного з яких наведенно у таблиці 1.2.

Таблиця 1.2 – Варіанти використання користувача «Донор»

Актор	**Найменування**	**Формулювання**
Авторизований користувач “Донор”	Статистика запитів на донорство	Можливість користувача переглянути кількість запитів на здачу крові
	Донорство	Подача запиту на здачу крові
	Історія донацій	Перегляд всіх запитів на донацію
	Запит на переливання крові	Перегляд всіх зроблених замовлень
	Історія переливання	Перегляд усіх запитів на переливання

Як ми бачимо кількість варіантів використання донора є найбільшою. Донор має доступ як до донацій крові, так і до запитів на периливання, та може дивитися на свою статистику і історію запитів. Варіанти використання пацієнта наведено в таблиці 1.3.

Таблиця 1.3 – Варіанти використання користувача «Пацієнт»

Актор	**Найменування**	**Формулювання**
Авторизований користувач “Пацієнт”	Статистика запитів на переливання	Можливість користувача переглянути кількість запитів на переливання крові
	Запит на переливання крові	Перегляд всіх зроблених замовлень
	Історія переливання	Перегляд усіх запитів на переливання

В свою чергу, варіанти використання пацієнта є меншими. Пацієнт отримує доступ до усіх потрібних функцій, а саме: запит на переливання крові, перелік статистики запитів і історії запитів на переливання. Варіанти використання адміністратора наведено у таблиці 1.4.

Таблиця 1.4 – Варіанти використання для адміністратора

Актор	**Найменування**	**Формулювання**
Адміністратор	Доступ до обліку крові	Редагування кількості крові яка є в наявності
	Взаємодія з користувачами	Дозволяє редагувати/видаляти дані користувачів
	Дашборд	Збір всієї інформації сайту про кількість крові та користувачів
	Опрацювання запитів	Підтвердження або відхилення запитів на донацію або переливання крові

Адміністратор має найбільше прав: він може редагувати кількість крові, що є в наявності, редагувати або видаляти дані користувачів, переглядати аналітичний дашборд, а також підтверджувати або відхиляти запити на донацію та переливання крові.

1.4 Опис ключових варіантів використання веб-застосунку

Опис варіантів використання є важливим інструментом при розробці програмного забезпечення або будь-якого іншого продукту. Він допомагає зрозуміти, як система буде використовуватися реальними користувачами та як вона повинна працювати в різних сценаріях. На рисунку 1.2 зображено діаграму використання для неавторизованого користувача[3].

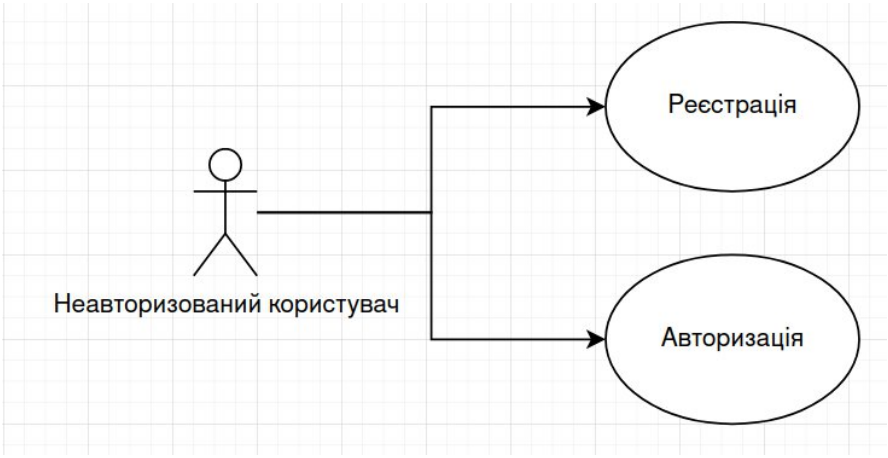

Рисунок 1.2 – Діаграма використання для неавторизованого користувача

Як вже зазначалось, неавторизований користувач немає багатьох привілеїв, а тільки авторизація та реєстрація. Щоб отримати більше привілеїв, йому необхідно авторизуватись. На рисунку 1.3 зображено діаграму використання для авторизованого користувача “Донор”.

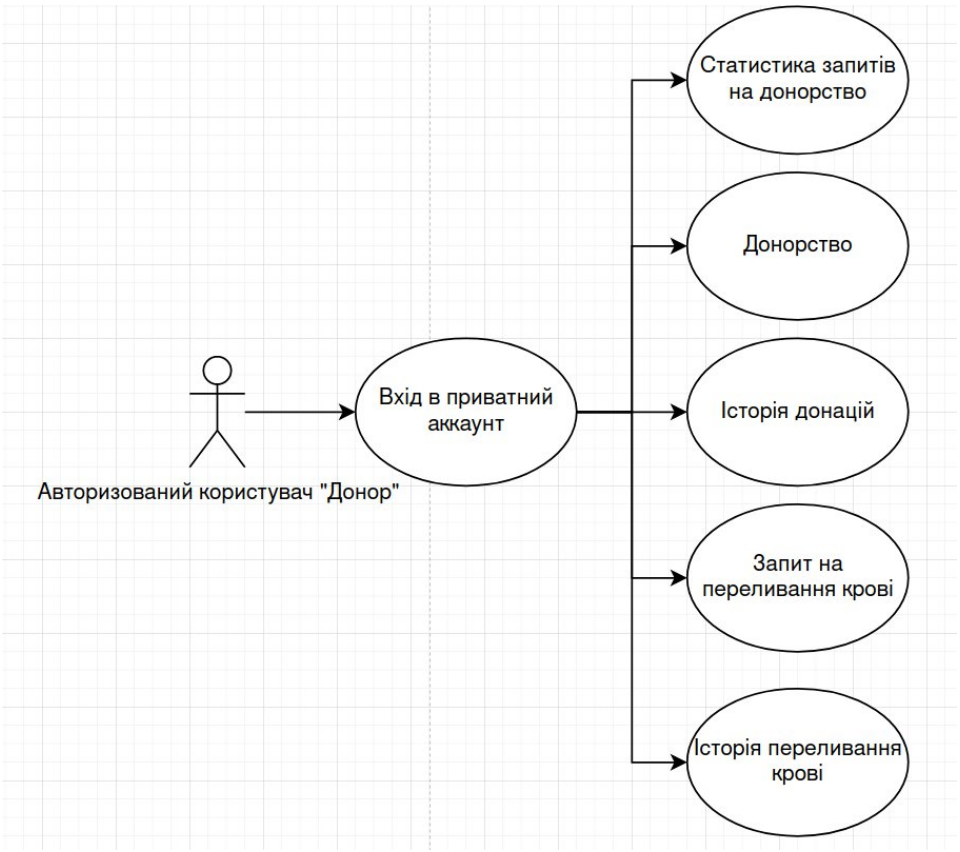

Рисунок 1.3 – Діаграма використання для авторизованого користувача “Донор”

Авторизований користувач “Донор” має значно більше привілеїв у порівнянні із неавторизованим користувачем, оскільки він окрім входу має привілеї дивитися свою статистику запитів, історію донацій і переливання крові та робити запити на донацію та переливання. Діаграму іншого авторизованого користувача, а саме “Пацієнт” зображенно на рисунку 1.4

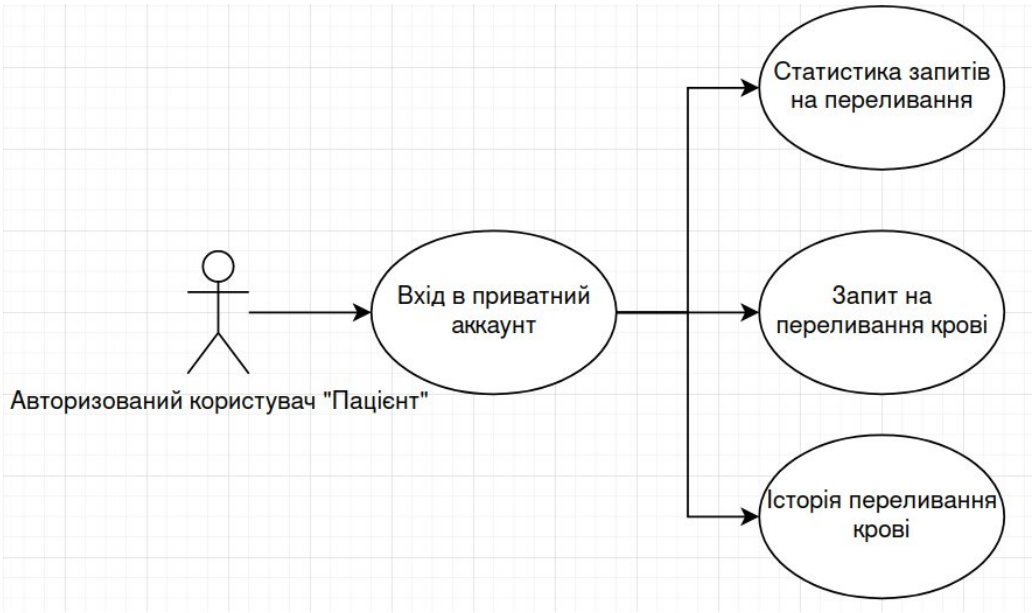

Рисунок 1.4 – Діаграма використання для авторизованого користувача “Пацієнт”

Авторизований користувач “Пацієнт” має менше привілеїв у порівнянні із донором, оскільки він може тільки робити запити на переливання крові.

Найбільше привілеїв має адміністратор. На рисунку 1.5 зображено діаграму використання для адміністратора.

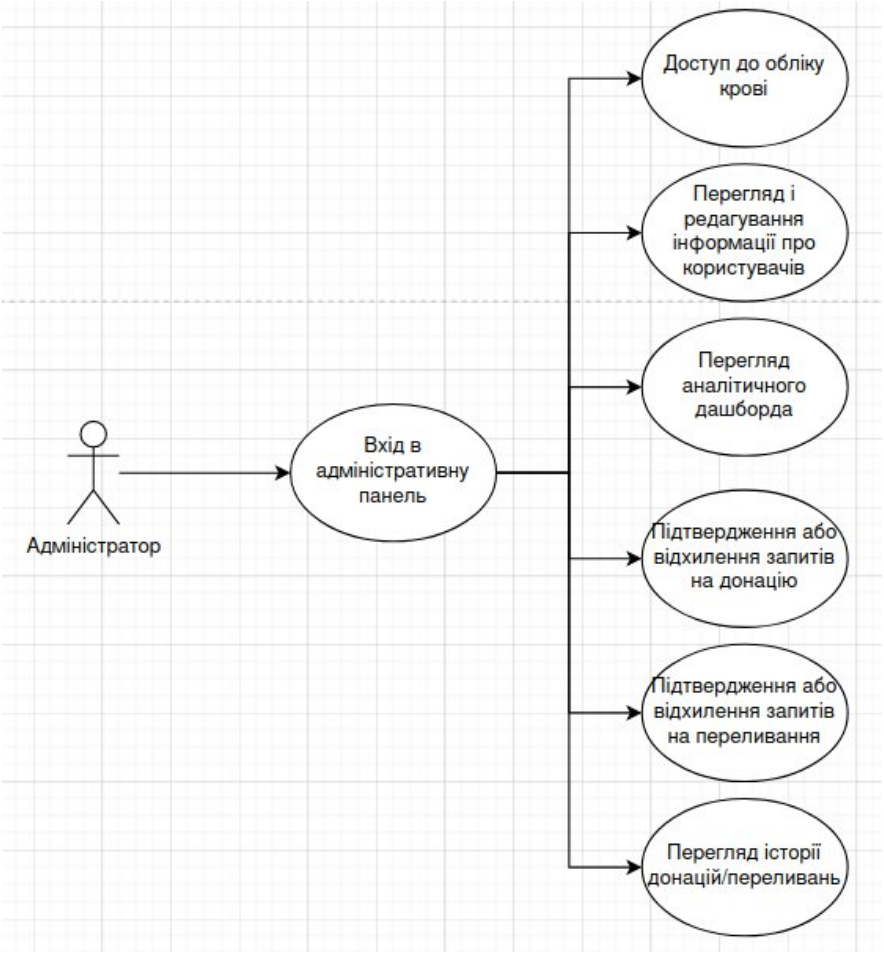

Рисунок 1.5 – Діаграма використання для адміністратора

Адміністратор має найбільше привілеїв, він має доступ до адміністративної панелі та обліку крові, може переглядати і редагувати інформацію про користувачів, приймати і відхиляти запити на донацію і переливання крові, змінювати інформацію про кількість крові, переглядати історію усіх запитів на переливання чи донацію.

1.5 Вибір середовища розробки

1.5.1 Мова програмування Python

Python – це високорівнева, інтерпретована мова динамічного програмування загального призначення, яка фокусується на читабельності коду. Зазвичай нею пишуть невеликі програми, порівняно з Java та C. Python була

заснована в 1991 році розробником Гвідо Ван Россумом і є однією з найпопулярніших мов у світі, що найшвидше розвиваються. Python – це потужна, гнучка та проста у використанні мова. Крім того, спільнота Python дуже активна. Її використовують у багатьох організаціях, оскільки вона підтримує декілька парадигм програмування. Вона також виконує автоматичне керування пам'яттю[4].

Переваги мови програмування Python[5]:

- Багата екосистема сторонніх модулів і бібліотек, які розширюють її функціональність для різних завдань.
- Велика кількість бібліотек підтримки, таких як NumPy для чисельних розрахунків та Pandas для аналізу даних, що робить її придатною для наукових та пов'язаних з даними додатків.
- Відкритий вихідний код і велику та активну спільноту, яка робить свій внесок у її розвиток та надає підтримку.
- Мова відома своєю простотою та читабельністю, що робить її чудовим вибором як для початківців, так і для досвідчених програмістів.
- Інтуїтивно зрозумілі та прості у використанні структури даних, що спрощує маніпуляції з даними та управління ними.
- Мова високого рівня, яка абстрагується від низькорівневих деталей, що робить її більш зручною для користувача.
- Є динамічно типізованою мовою, що означає, що вам не потрібно оголошувати типи даних явно, що робить її гнучкою, але при цьому надійною.
- Підтримує як об'єктно-орієнтоване, так і процедурне програмування, забезпечуючи універсальність стилів кодування.
- Переноситься на різні операційні системи та є інтерактивною, що дозволяє виконувати та тестувати код у реальному часі.
- Лаконічний синтаксис Python дозволяє розробникам швидко створювати прототипи додатків з меншою кількістю коду.
- Чистий дизайн Python забезпечує покращений контроль процесів, а також має чудові можливості обробки тексту, що робить її ефективною для

різних додатків. Більшість завдань можна виконати за допомогою інструментів та функцій, які вже вбудовані в мову, що робить розробку швидшою та зручнішою.

Недоліки мови програмування Python[6]:

- Може бути повільнішою, ніж скомпільовані мови, такі як C або Java. Це може бути проблемою для завдань, що вимагають високої продуктивності. Деякі операції, особливо з великими обсягами даних, можуть бути повільними у порівнянні з іншими мовами.
- Глобальне блокування інтерпретатора є обмеженням Python, яке може ускладнювати багатопоточність і паралельність у деяких додатках. Це може призвести до проблем з продуктивністю та масштабованістю деяких програм.
- Python може споживати багато пам'яті, особливо при роботі з великими наборами даних або виконанні складних алгоритмів. Це може призвести до обмежень на обсяг доступної пам'яті та збільшення витрат на інфраструктуру.
- Python є динамічно типізованою мовою, що може призводити до помилок у час виконання через неочікувані типи даних. Хоча це робить мову більш гнучкою, воно може також збільшувати час виявлення помилок та ускладнювати рефакторинг коду.
- Python має велику кількість пакунків та бібліотек, що може призводити до проблем з версіями та конфліктами пакунків. Керування залежностями та версіями може бути складною задачею, особливо в великих проектах.

Незважаючи на ці недоліки, Python залишається однією з найпопулярніших мов програмування завдяки своїй простоті використання, гнучкості та багатому функціоналу. Її широке застосування в різних галузях, від наукових досліджень до веб-розробки, свідчить про її потужний потенціал і популярність серед розробників.

1.5.2 Фреймворк Django

Для того аби використовувати Python для наших цілей потрібно було обрати фрейморк, який би дозволив писати веб-застосунок з складним функціоналом. Вибір впав на Django[7].

Django – це веб-фреймворк з відкритим вихідним кодом для бекенду веб-додатків на мові Python. Його основна мета – полегшити створення складних веб-сайтів, що керуються базами даних. Він робить акцент на багаторазовому використанні та модульності, що дозволяє швидко розробляти веб-сайти без шкоди для їхньої функціональності та масштабованості. Django надає вбудовані компоненти, такі як об'єктно-реляційне відображення (ORM), інтерфейс адміністратора та маршрутизація URL-адрес, що полегшує розробникам створення та підтримку веб-сайтів[8].

Переваги:

- Філософія дизайну Django забезпечує велику гнучкість і можливість кастомізації.
- Django містить багато вбудованих компонентів, таких як ORM, інтерфейс адміністратора та маршрутизація URL-адрес.
- Django має велику та активну спільноту, яка надає підтримку та ресурси.
- Django включає декілька функцій безпеки, таких як захист від CSRF та захист від SQL-ін'єкцій.
- Вибір дизайну Django сприяє його швидкості та ефективності.

Django підходить для веб-додатків, які потребують складної функціональності, таких як автентифікація користувачів, системи управління контентом або платформи електронної комерції. Він також підходить для проектів, де розробники хочуть зосередитися на функціональності та масштабованості, а не турбуватися про низькорівневі деталі[9].

1.5.3 Мова розмітки HTML

HTML (Hypertext Markup Language) – це стандартна мова розмітки, яка використовується для створення веб-сторінок[10]. Вона забезпечує структуру та вміст, який веб-браузер відображає користувачеві.

Переваги HTML[11]:

- Легка у вивченні.
- Широко підтримується.
- Легка і швидка.
- Гнучкість.
- Незалежна від платформи.
- Легко редагується.
- Добре інтегрується з іншими мовами.
- Швидко завантажується.
- Корисна для початківців.
- Підтримується всіма браузерами.

Недоліки HTML:

- Обмежена підтримка складного дизайну та динамічного контенту.
- Проблеми з безпекою.
- Не підходить для створення динамічних взаємодій.
- Обмежена підтримка мультимедійного контенту.

Ключові особливості HTML:

- Мова розмітки, що використовується для створення веб-сторінок.
- Складається з ряду елементів, представлених тегами.
- Визначає різні частини веб-сторінки, такі як заголовки, абзаци, зображенняпосилання, форми, таблиці тощо.
- Може бути інтегрована з іншими мовами, такими як CSS та JavaScript.
- Підтримується всіма веб-браузерами.

HTML – це основний будівельний блок веб-сторінок і дозволяє створювати структурований та доступний контент для Інтернету.

1.5.4 Каскадні таблиці стилів CSS

CSS (Cascading Style Sheets – каскадні таблиці стилів) – це мова таблиць методів, яка надає веб-дизайнерам контроль над тим, як інтернет-сайт взаємодіє з веб-браузерами, включаючи форматування та відображення їхніх HTML-документів[12]. Основна мета CSS - відокремити вміст документа від його презентації, яка включає в себе елементи стилю, такі як колір, макет і шрифти[13].

Переваги CSS:

- Послідовний стиль на декількох сторінках.
- Легко оновлювати та підтримувати.
- Швидше завантаження.
- Адаптивний дизайн, зручний для пристроїв.
- Спрощує розробку та підтримку веб-сайтів.
- Зменшує розмір переданих файлів.
- Легко налаштовується користувачами.

Недоліки CSS:

- Плутанина між різними версіями CSS (CSS1, CSS2, CSS3).
- Проблеми сумісності з різними браузерами.
- Відсутність безпеки.
- Складність для не-розробників та початківців.
- Кросбраузерні проблеми.
- Кілька рівнів CSS можуть заплутати.

Загалом, CSS має багато переваг, але також має деякі недоліки, які необхідно враховувати при використанні у веб-розробці[14].

1.5.5 Система управління базою даних SQLite

SQLite – це популярна, легка та автономна система управління базами даних на мові SQL, яка широко використовується для різних додатків[15]. Це програмна бібліотека, яка забезпечує систему управління реляційними базами даних (СКБД) і вважається файловою базою даних[16].

Переваги SQLite:

- Простота використання: SQLite проста у використанні і не потребує налаштування чи адміністрування. Це один файл, який можна легко копіювати та переміщувати.
- Переносимість: SQLite є дуже портативною і може використовуватися на будь-якій платформі, включаючи Windows, Linux і Mac.
- Продуктивність: SQLite є швидкою та ефективною, з хорошою продуктивністю для робочих навантажень з великим обсягом читання.
- Надійність: SQLite надійний і має міцну конструкцію, яка забезпечує цілісність даних.
- Економічність: SQLite має відкритий вихідний код і є безкоштовним у використанні, що робить його економічно вигідним рішенням для багатьох додатків.
- Безпека: SQLite має вбудоване шифрування і підтримує різні заходи безпеки, що робить її надійним рішенням для зберігання конфіденційних даних.

SQLite – це потужний і універсальний движок баз даних, який добре підходить для багатьох додатків. Простота використання, портативність, продуктивність, надійність, економічність і безпека роблять його привабливим вибором для багатьох розробників[17].

1.5.6 Текстовий редактор

PyCharm – це популярне інтегроване середовище розробки (IDE), розроблене спеціально для розробників Python[18]. Воно надає широкий спектр

функцій та інструментів, які допомагають розробникам ефективно писати, налагоджувати та тестувати свій код на Python.

Переваги PyCharm:

- Пропонує розширені функції завершення коду, перевірки коду та рефакторингу коду, які допомагають розробникам писати чистий, читабельний та ефективний код.
- PyCharm надає вбудований відладчик і фреймворк для тестування, що дозволяє легко виявляти і виправляти помилки в коді.
- Функції навігації та управління проектами PyCharm дозволяють розробникам легко орієнтуватися та керувати великими проектами.
- Легко інтегрується з популярними системами контролю версій, такими як Git, Mercurial та Subversion.
- Надає вбудовану підтримку фреймворків для веб-розробки, таких як Django, Flask та Pyramid, що робить його ідеальним вибором для створення веб-додатків.
- PyCharm підтримує широкий спектр баз даних, включаючи MySQL, PostgreSQL та SQLite, що дозволяє розробникам легко взаємодіяти зі своїми базами даних.
- Має велику колекцію плагінів, які можна використовувати для розширення його функціональності та підтримки різних завдань розробки.

Недоліки PyCharm:

- Може бути ресурсоємним, особливо для великих проектів, що може призвести до зниження продуктивності.
- PyCharm має багато функцій та інструментів, які можуть бути непосильними для початківців.
- PyCharm є комерційним продуктом, і його професійна версія вимагає підписки або одноразової покупки.
- Хоча PyCharm в першу чергу призначений для розробки на Python, його підтримка інших мов обмежена.

Можливості PyCharm для веб-розробки:

- Забезпечує підсвічування синтаксису, завершення коду та підтримку налагодження для HTML, CSS та JavaScript.
- Підтримує популярні шаблони, такі як Jinja2, Mustache та Handlebars.
- Надає вбудовану підтримку популярних веб-фреймворків, таких як Django, Flask та Pyramid.
- Підтримує широкий спектр баз даних, що дозволяє легко взаємодіяти з базами даних у веб-додатках.

PyCharm – це потужна і багатофункціональна IDE, яка добре підходить для створення веб-додатків. Її переваги, такі як завершення коду, налагодження та тестування, роблять її ідеальним вибором для веб-розробки. Незважаючи на деякі недоліки, такі як ресурсоємність та складність у вивченні, PyCharm є популярним вибором серед розробників Python і широко використовується в індустрії[19].

1.6 Висновок до першого розділу

В першому розділу кваліфікаційної роботи проаналізовано предметну область, сформулювано вимоги до веб-застосунку. Знайдено актанти, а саме неавторизованого користувача, авторизованого користувача “Донор” та “Пацієнт” і адміністратора. Кожен з даних акторів повинен мати різний доступ до веб-застосунку, також наведено діаграми використання для кожного з них.

Окрім цього, було обрано технології для побудови веб-застосунку. Мову програмування Python для серверної логіки з фреймворком Django. HTML,CSS. У якості СУБД було обрано SQLite. Текстовим редактором було обрано PyCharm.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБ-ЗАСТОСУНКУ ДЛЯ СУПРОВОДУ ПРОЦЕСІВ ДОНОРСТВА КРОВІ

2.1 Моделювання архітектури веб-застосунку

У цьому розділі описується архітектура веб-застосунку для супроводу процесів донорства крові. Архітектура такого сайту має велике значення, оскільки ефективна архітектура дозволяє забезпечити швидке завантаження сторінок, оперативну обробку заявок на донорство та ефективну роботу з базою даних. Крім того, добре спроектована архітектура дозволяє створити зручний та привабливий користувацький інтерфейс[20]. Переваги тришарової архітектури включають:

- Масштабованість: Тришарова архітектура дозволяє легко масштабувати кожен шар незалежно один від одного, що сприяє ефективному обслуговуванню великої кількості користувачів.
- Підтримка та обслуговування: Кожен шар архітектури виконує свою специфічну функцію, що спрощує підтримку та оновлення сайту.
- Безпека: Тришарова архітектура забезпечує додаткові рівні захисту, що є важливим для безпеки персональних даних донорів та реципієнтів.
- Переносимість: Внесення змін в один шар не вимагає змін в інших шарах, що підвищує гнучкість та спрощує адаптацію до нових вимог.
- Повторне використання коду: Логіку бізнес-процесів можна використовувати повторно в різних частинах сайту, що знижує час і витрати на розробку.яє легше реагувати на вимоги замовника та зміни у бізнес-процесах.

Тришарова архітектура дозволяє створити структуровану, ефективну та масштабовану систему, що є важливим для успішного функціонування веб-застосунку[21].

В процесі створенні веб-застосунку, буде використовуватись три-шарова клієнт-серверна архітектура, яка складається з:

- Клієнта, який подає запит на отримання інформації.

- Сервера, що містить певну логіку програми, який надає відповідь на запит клієнта.
- База даних, яка містить інформацію.

На рисунку 2.1 зображено архітектуру веб-застосунку.

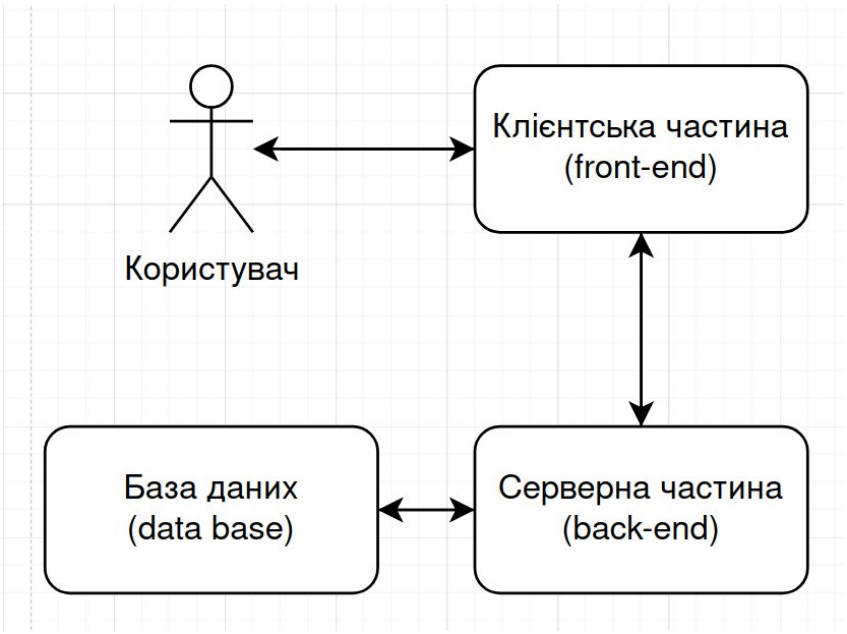

Рисунок 2.1 – Архітектура веб-застосунку

У веб-додатках серверне програмне забезпечення може бути встановлене на фізичних серверах або в хмарних середовищах обчислень. Сервер також може бути на тому ж самому пристрої, що і клієнт, наприклад, у разі локального веб-сервера для розробки.

База даних є системою зберігання та організації даних, яка доступна та змінюється за допомогою певного програмного забезпечення. У клієнт-серверній архітектурі база даних може розташовуватися на сервері та використовуватися для зберігання даних, які обробляються та надсилаються клієнтам.

2.2 Проектування структури веб-застосунку

Проектування структури веб-застосунку передбачає створення організаційної схеми та архітектури, спрямованої на забезпечення ефективної

роботи та відповідність потребам користувачів. Цей процес включає формування детального плану, що описує всі компоненти, їх взаємодію та функціональні можливості[22].

Стуктура веб-застосунку може вагомо відрізнятись для неавторизованого, авторизованого користувачів та адмінастратора, тому щоб краще зрозуміти структуру яку потрібно імплементувати було створено наступні графіки

На рисунку 2.2 зображено структуру веб-застосунку для неавторизованого користувача

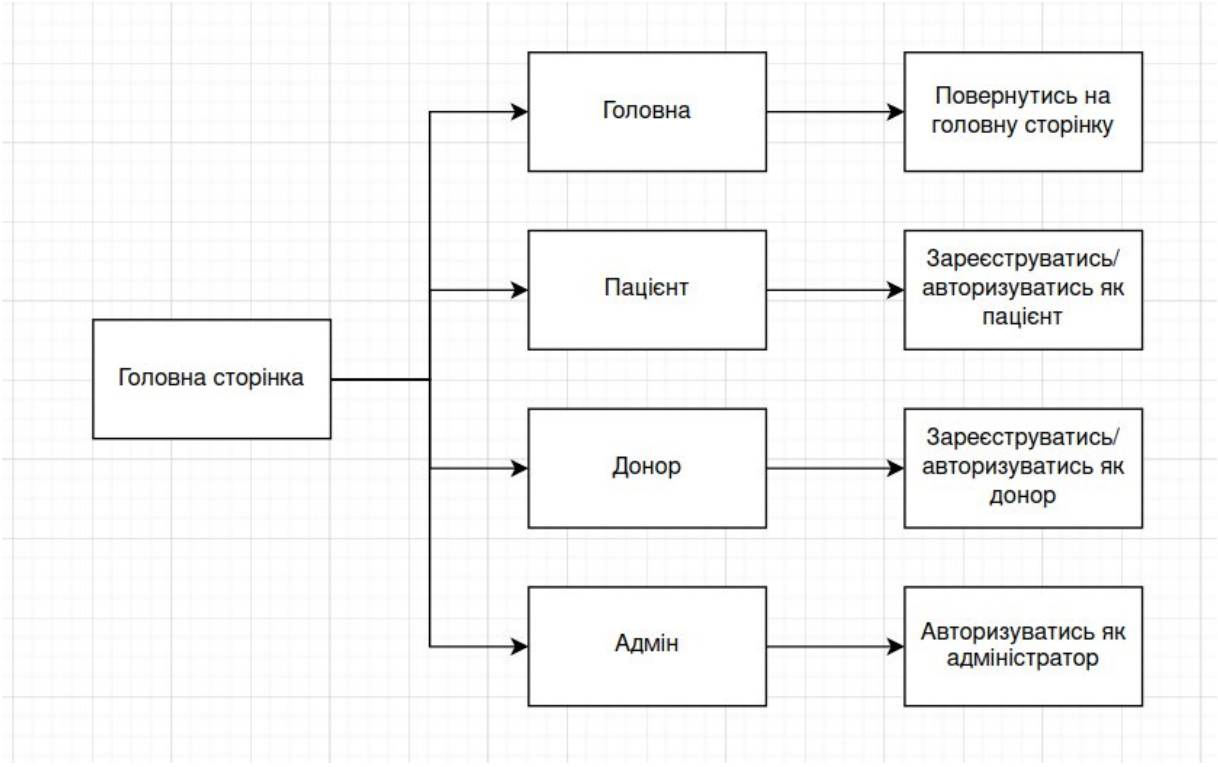

Рисунок 2.2 – Структура веб-застосунку для неавторизованого користувача

Неавторизований користувач має найменше привілеїв та можливостей. Першочергово він попадає на домашню сторінку, звідки уже за допомогою панелі навігації може перейти на сторінку авторизації/реєстрації.

- Головна: Обравши даний варіант користувач повинен повернутись на домашню сторінку веб-застосунку.

- Пацієнт: Обравши даний варіант користувач може зареєструватись або авторизуватись як пацієнт, щоб отримати доступ до запитів на переливання крові.
- Донор: Обравши даний варіант користувачможе зареєструватись або авторизуватись як донор, щоб отримати доступ як до запитів на донацію так і переливання крові.
- Адмін: Обравши цей варіант користувач має змогу авторизуватись як адміністратор, щоб отримати доступ до всіх його функцій.

На рисунку 2.3 зображено структуру веб-застосунку для авторизованого користувача.

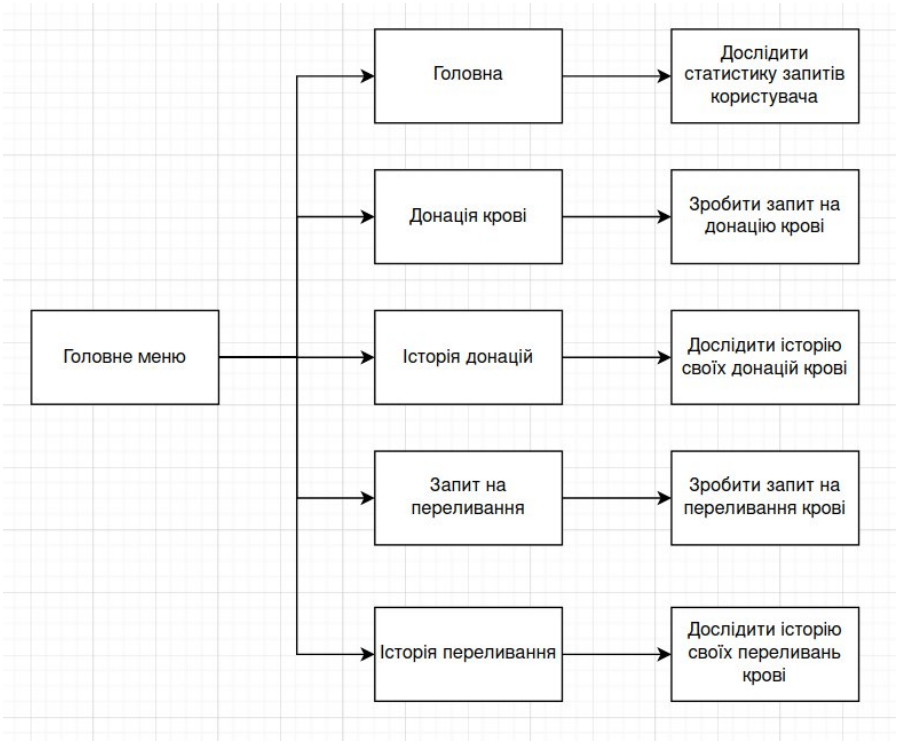

Рисунок 2.3 – Структура веб-застосунку для авторизованого користувача

Після авторизації користувач, незалежно від типу його аккаунту першим ділом потрапляє на сторінку з статисткою запитів

- Головна: Обравши даний варіант користувач повертається на головну сторінку, де він може ознайомитися з власною статистикою запитів на донацію/переливання крові.

- Донація крові: Обравши даний варіант користувач може зробити запит на донацію крові.
- Історія донацій: Обравши даний варіант користувач може ознайомитися з власною історією донацій.
- Запит на переливання: Обравши даний варіант користувач може зробити запит на переливання крові.
- Історія переливання: Обравши даний варіант користувач може ознайомитися з власною історією переливання крові.

Як вже зазначалось вище, адміністратор веб-застосунку повинен мати найбільше функціоналу. На рисунку 2.4 зображено структуру веб-застосунку для адміністратора.

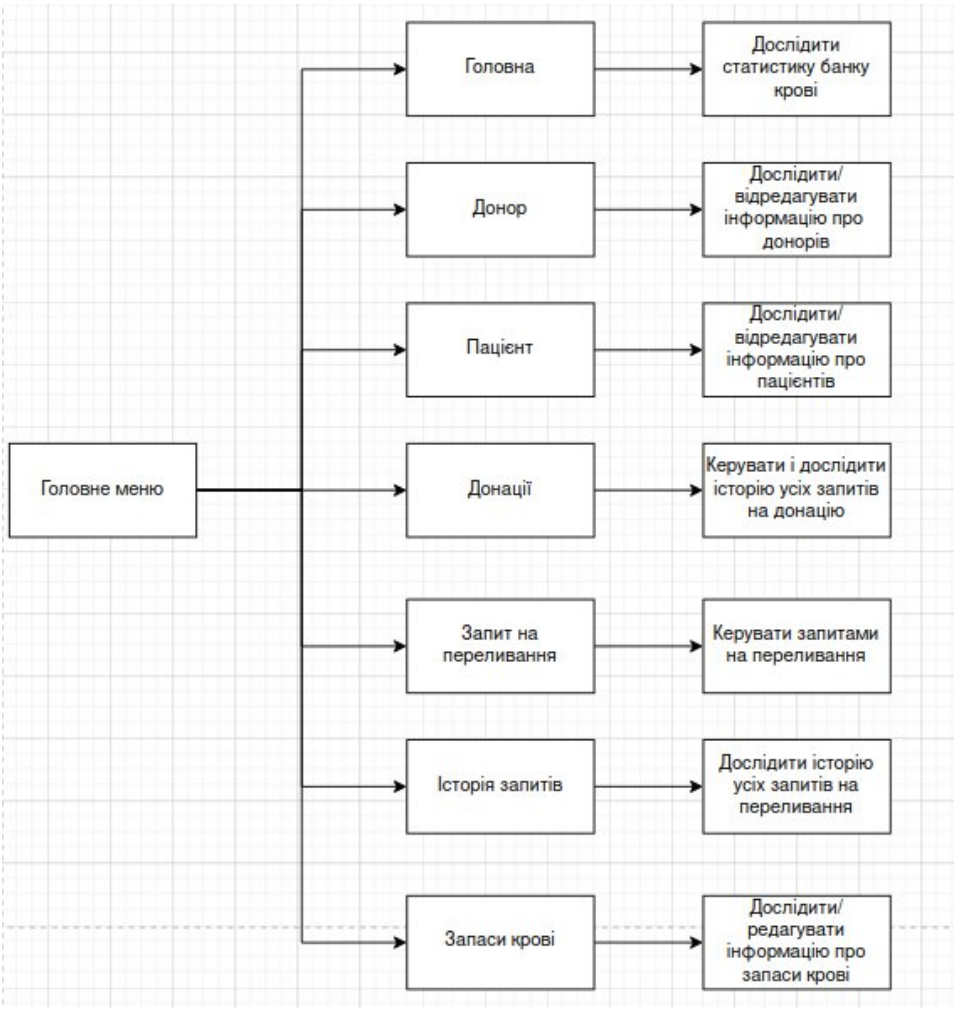

Рисунок 2.4 – Структура веб-застосунку для адміністратора

У головному меню адміністратора є 7 опцій:

- Головна: Обравши даний варіант адміністратор повертається на головну сторінку адмін-панелі, де він може ознайомитися з інформацією про кількість донорів в базі даних, загальною кількістю запитів на переливання, кількість одобрених запитів на переливання, загальною кількістю крові і кількістю крові для кожної окремої групи.
- Донор: Обравши даний варіант адміністратор може ознайомитися з інформацією про усіх користувачів зареєстрованих як донори, та за потреби відредагувати їх інформацію, або видалити їх з бази даних.
- Пацієнт: Обравши даний варіант адміністратор може ознайомитися з інформацією про усіх користувачів зареєстрованих як пацієнти, та за потреби відредагувати їх інформацію, або видалити їх з бази даних.
- Донації: Обравши даний варіант адміністратор може ознайомитися з запитами на донацію, їх історією та одобрити, або відхилити нові запити на донацію.
- Запит на переливання: Обравши даний варіант адміністратор може ознайомитися з запитами на переливання крові, і або одобрити, або відхилити їх.
- Історія переливання: Обравши даний варіант адміністратор може ознайомитися з усією історією запитів на переливання крові.
- Запаси крові: Обравши даний варіант адміністратор може ознайомитися і відредагувати інформацію про запаси крові.

2.3 Проектування бази даних

2.3.1 Аналіз логічних процесів

Для ефективної організації бази даних та оптимізації процесів донорства крові необхідно провести аналіз логічних-процесів, які характеризують взаємодію донорів і медичних закладів[23]. Логічні процеси для першої таблиці auth_user наведені в таблиці 2.1[24].

Таблиця 2.1 – Логічні процеси для таблиці auth_user

Дія	**Логічний процес**
Створення таблиці користувачів (auth_user)	Кожному користувачеві присвоюється унікальний ідентифікатор (id)
	Вводиться пароль користувача (password)
	Фіксується останній вхід користувача (last_login)
	Позначається, чи є користувач суперкористувачем (is_superuser)
	Вводиться ім'я користувача (username)
	Вводиться ім'я користувача (first_name)
	Вводиться електронна пошта користувача (email)
	Позначається, чи є користувач співробітником (is_staff)
	Позначається активність користувача (is_active)
	Фіксується дата реєстрації користувача (date_joined)
	Вводиться прізвище користувача (last_name)

Логічні процеси для таблиці donor_donor наведенні в таблиці 2.2.

Таблиця 2.2 – Логічний процес для таблиці donor_donor

Дія	**Логічний процес**
Створення таблиці донорів (donor_donor)	Кожному донору присвоюється унікальний ідентифікатор (id)
	Додається зображення профілю донора (profile_pic)
	Вводиться група крові донора (bloodgroup)
	Вводиться адреса донора (address)
	Вводиться номер мобільного телефону донора (mobile)
	Пов'язується з користувачем (user_id)

Логічні процеси для таблиці patient_patient наведенні в таблиці 2.3.

Таблиця 2.3 Логічний процес для таблиці patient_patient

Дія	**Логічний процес**
Створення таблиці пацієнтів (patient_patient)	Кожному пацієнту присвоюється унікальний ідентифікатор (id)
	Додається зображення профілю пацієнта (profile_pic)
	Вводиться вік пацієнта (age)
	Вводиться група крові пацієнта (bloodgroup)
	Вводиться хвороба пацієнта (disease)
	Вводиться ім'я лікаря пацієнта (doctorname)
	Вводиться адреса пацієнта (address)
	Вводиться номер мобільного телефону пацієнта (mobile)
	Пов'язується з користувачем (user_id)

Логічні процеси для таблиці donor_blooddonate наведенні в таблиці 2.4.

Таблиця 2.4 – Логічний процес для таблиці donor_blooddonate

Дія	**Логічний процес**
Створення таблиці донорських кровоздач (donor_blooddonate)	Кожній кровоздачі присвоюється унікальний ідентифікатор (id)
	Вводиться хвороба донора (disease)
	Вводиться вік донора (age)
	Вводиться група крові донора (bloodgroup)
	Вводиться кількість одиниць крові (unit)
	Позначається статус кровоздачі (status)
	Вводиться дата кровоздачі (date)
	Пов'язується з донором (donor_id)

Логічні процеси для таблиці blood_bloodrequest наведенні в таблиці 2.5.

Таблиця 2.5 – Логічний процес для таблиці blood_bloodrequest

Дія	Логічний процес
Створення таблиці запитів на кров (blood_bloodrequest)	Кожному запиту присвоюється унікальний ідентифікатор (id)
	Пов'язується з пацієнтом (patient_id)
	Вводиться вік пацієнта (patient_age)
	Вводиться причина запиту (reason)
	Вводиться група крові, яку запитують (bloodgroup)
	Вводиться кількість одиниць крові (unit)
	Позначається статус запиту (status)
	Пов'язується з користувачем, який подав запит (request_by)
	Вводиться дата запиту (date)

Логічні процеси для таблиці blood_stock наведенні в таблиці 2.3.

Таблиця 2.6 – Логічний процес для таблиці blood_stock

Дія	Логічний процес
Створення таблиці запасу крові (blood_stock)	Кожному запису присвоюється унікальний ідентифікатор (id)
	Вводиться група крові (bloodgroup)
	Вводиться кількість одиниць крові в запасі (unit)

Інші таблиці

- auth_group: Таблиця для груп користувачів.
- auth_group_permissions: Таблиця для зв'язку груп з дозволами.
- auth_permission: Таблиця для дозволів.
- auth_user_groups: Таблиця для зв'язку користувачів з групами.

- auth_user_user_permissions: Таблиця для зв'язку користувачів з дозволами.
- django_session: Таблиця для зберігання сесій.
- django_migrations: Таблиця для відстеження міграцій.
- django_content_type: Таблиця для типів контенту.
- django_admin_log: Таблиця для логів адміністратора.
- sqlite_sequence: Таблиця для збереження послідовностей SQLite.

Отже, в базі даних використовувались Django сесії та міграції.Сесії в Django використовуються для зберігання даних про сеанс користувача між запитами[25]. Це дозволяє веб-додатку запам'ятовувати інформацію про користувача, наприклад, його стан авторизації, елементи в кошику покупок або інші тимчасові дані[26].

Основні характеристики сесій:

- session_key: Унікальний ідентифікатор сесії.
- session_data: Зберігає дані сесії в серіалізованому вигляді.
- expire_date: Дата і час, коли сесія стане недійсною.

Навіщо потрібні сесії:

- Зберігати стан користувача між запитами.
- Відстежувати дії користувача на сайті.
- Підтримувати користувача авторизованим.

Міграції в Django – це механізм для управління змінами в структурі бази даних моделі Django. Вони дозволяють автоматично створювати та застосовувати зміни в схемі бази даних, виходячи зі змін у моделях[27].

Основні характеристики міграцій:

- app: Додаток, до якого відноситься міграція.
- name: Назва міграції.
- applied: Час, коли міграція була застосована.

Навіщо потрібні міграції:

- Зберігати історію змін у структурі бази даних.
- Автоматизувати процес оновлення схеми бази даних.

- Полегшувати розробку, дозволяючи синхронізувати структуру бази даних з моделями в коді.

2.3.2 Розробка логічних правил

В результаті дослідження логічних процесів було сформовано наступні логічні правила:

- Код пацієнта не може бути пустим. Кожен пацієнт повинен мати унікальний код і не має повторюватись.
- Кожен пацієнт повинен мати вказану групу крові.
- Кожна група крові повинна бути точно визначена.
- Кожен пацієнт повинен мати вік, адресу та контактний номер телефону.
- Донор повинен мати унікальний код, групу крові, адресу та номер телефону.
- Кожне донорство крові повинно мати унікальний код, вказівки про стан донора. Кількість крові не може бути від’ємною.
- Кожен запит на кров повинен мати унікальний код та інформацію про пацієнта, для якого потрібна кров.

Ці правила допомагають забезпечити правильне функціонування системи донорства крові, зберігаючи цілісність та точність даних у базі даних.

2.3.3 Розробка структури таблиць

Розробка структури таблиць в базі даних для системи донорства крові є критично важливою, оскільки вона визначає, як дані будуть організовані та зберігатися[28]. Хороша структура може:

- сприяти ефективним запитам та зменшувати дублювання даних;
- дозволяти легко додавати нові функціональності та розширювати можливості системи без значного перероблення бази даних;

- дозволяти масштабувати базу даних;
- спрощувати аналіз.

Таким чином, розробка структури таблиць для системи донорства крові є важливим етапом проектування, який допомагає забезпечити ефективність, цілісність та масштабованість бази даних, а також полегшує адміністрування та аналіз даних.

У таблиці 2.7 представлено структуру таблиці даних про користувачів з переліком імені полів, їх типами, довжиною та описом.

Таблиця 2.7 – Структура таблиці “auth_user”

Ключ	**Ім’я поля**	**Тип даних**	**Опис**
PK	id	int	ID користувача
	password	varchar(128)	Ім’я комікса
	last_login	datetime	Останній вхід
	is_superuser	bool	Чи є адміном
	username	varchar(150)	Нікнейм
	first_name	varchar(30)	Ім’я
	email	varchar(254)	Емайл
	is_staff	Bool	Чи працівник
	is_active	bool	Чи активний
	date_joined	datetime	Дата реєстрації
	last_name	varchar(150)	Прізвище

У таблиці 2.8 представлено структуру таблиці даних про пацієнтів з переліком імені полів, їх типами, довжиною та описом.

Таблиця 2.8 – Структура таблиці “patient_patient”

Ключ	**Ім’я поля**	**Тип даних**	**Опис**
PK	id	int	ID пацієнта
	profile_picture	varchar(100)	Зображення профілю
	age	int	Вік пацієнта
	bloodgroup	varchar(10)	Група крові
	disease	varchar(100)	Захворювання
	doctorname	varchar(50)	Ім’я лікаря пацієнта
	address	varchar(40)	Адреса пацієнта
	mobile	varchar(20)	Номер телефону
FK	user_id	int	ID користувача поєднаний з пацієнтом

У таблиці 2.9 представлено структуру таблиці даних про донора з переліком імені полів, їх типами, довжиною та описом.

Таблиця 2.9 – Структура таблиці “donor_donor”

Ключ	**Ім’я поля**	**Тип даних**	**Опис**
PK	id	int	ID донора
	profile_pic	varchar(100)	Зображення профілю
	bloodgroup	varchar(10)	Група крові
	adress	varchar(40)	Адреса
	mobile	varchar(20)	Номер телефону
FK	user_id	int	ID користувача поєднаний з донором

У таблиці 2.10 представлено структуру таблиці даних про донації крові з переліком імені полів, їх типами, довжиною та описом.

Таблиця 2.10 – Структура таблиці “donor_blooddonate”

Ключ	Ім’я поля	Тип даних	Опис
PK	id	int	ID здачі крові
	disease	varchar(100)	Хвороба
	age	int	Вік
	bloodgroup	varchar(10)	Група крові
	unit	int	Кількість одиниць крові
	status	varchar(20)	Статус кровоздачі
	date	datetime	Дата
FK	donor_id	int	Ідентифікатор донора

У таблиці 2.11 представлено структуру таблиці даних про запити на переливання крові з переліком імені полів, їх типами, довжиною та описом.

Таблиця 2.11 – Структура таблиці “blood_bloodrequest”

Ключ	Ім’я поля	Тип даних	Опис
PK	id	int	ID запиту
	patient_name	varchar(30)	Ім’я пацієнта
	patient_age	int	Вік пацієнта
	reason	varchar(500)	Причина
	bloodgroup	varchar(10)	Група крові
	unit	int	Кількість одиниць крові
	status	varchar(20)	Статус запиту
FK	request_by_donor_id	int	ID донора який подав запит
FK	request_by_patient_id	int	ID пацієнта який подав запит
	date	datetime	Дата

У таблиці 2.12 представлено структуру таблиці даних про запаси крові з переліком імені полів, їх типами, довжиною та описом.

Таблиця 2.12 – Структура таблиці “blood_stock”

Ключ	Ім’я поля	Тип даних	Опис
PK	id	int	ID групи крові
	bloodgroup	varchar(10)	Група крові
	unit	int	Кількість одиниць крові

2.4 Проектування візуального вигляду веб-застосунку

Проектування візуального вигляду веб-застосунку для супроводу процесів донорства крові спрямоване на створення привабливого та зручного для користувачів інтерфейсу, який сприятиме залученню донорів і забезпеченню їх задоволеності[29]. Ось деякі ключові аспекти, які слід врахувати при проектуванні візуального вигляду нашого веб-застосунку:

- Визначення цільової аудиторії;
- Забезпечення легкої та інтуїтивно зрозумілої навігації, щоб користувачі могли швидко знаходити необхідну інформацію та функції;
- Чітка та організована структура меню з категоріями, які полегшують навігацію;
- Короткі, чіткі та інформативні описи всіх доступних послуг та процесів;
- Спрощений дизайн з мінімальною кількістю зайвих елементів для підвищення зосередженості на основній інформації;
- Адаптивний дизайн, що забезпечує зручне відображення на різних пристроях;
- Використання чітких та великих кнопок для легкого доступу до основних функцій, таких як реєстрація донора, запис на здачу крові тощо;
- Перевірка сумісності дизайну з різними браузерами для забезпечення однакового відображення на всіх платформах.

Ефективний дизайн нашого веб-застосунку допоможе зробити його привабливим для користувачів, що сприятиме збільшенню кількості донорів та підвищенню ефективності процесів донорства кровіх[30].

2.5 Висновок до другого розділу

В другому розділі кваліфікаційної роботи було розібрано питання архітектури веб-застосунку, створенно діаграми для кращого розуміння під час розробки. Спроектовано базу даних, проанілазувавши і розписавши логічні процеси, а також було розписано основні таблиці, їх змінні, основні і зовнішні ключі. І обрано ключові аспекти проектування візуального вигляду веб-застосунку.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ ДЛЯ СУПРОВОДУ ПРОЦЕСІВ ДОНОРСТВА КРОВІ

3.1 Структура файлової системи

3.1.1 Структура катологів веб-застосунку

Результатом виконання кваліфікаційної роботи став веб-застосунок для супроводу процесів донорства крові, для полегшення його створення і подальшу підтримку/розробку, було вирішено структурувати файли по каталогах відповідно до їх наповнення і функцій[31].

На рисунку 3.1 зображенно каталог проекту.

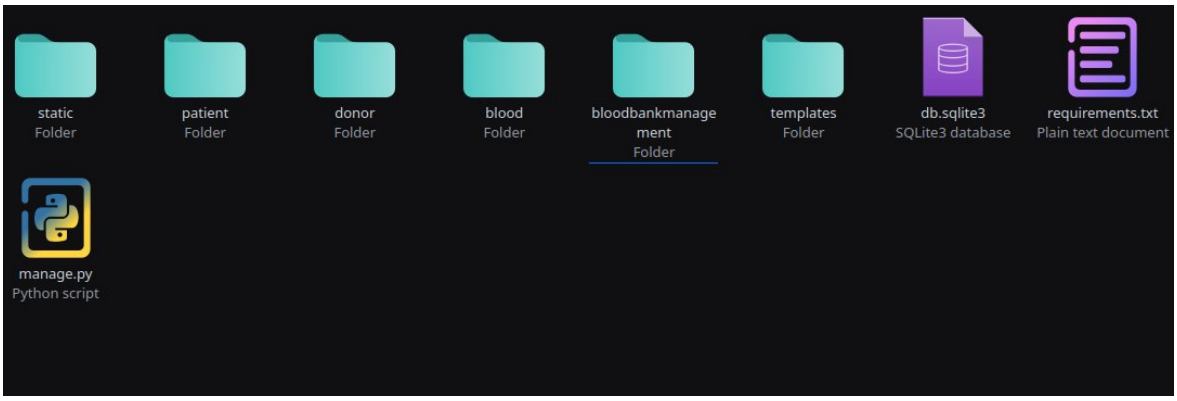

Рисунок 3.1 – Каталог веб-застосунку

Каталог “static” містить у собі усі зображення, включаючи ті, що потрібні для коректного відображення сайту, а також зображення профілів користувачів, на які у базі даних зберігається посилання шляху. Також цей каталог містить у собі файли CSS для правильного відображення дизайну сайту.

Каталог “patient” містить у собі усі файли потрібні для функціонування Django додатку patient, де реалізуються процеси подачі запитів на переливання, запит на структуру панелі користувача і форм для заповнення.

Каталог “donor” так само як і попередній містить усі потрібну інформацію для користувача підтипу “донор”, має усі файли які потрібні для їх функціонування як окремого додатку Django.

Каталог “blood” містить у собі усі файли що потрібні для правильного функціонування усіх запитів і їх історією.

Каталог “bloodmanagment” містить у собі початковий файл для запуску усього веб-застосунку і його підключення до окремих додатків Django, підключення до бази даних і усі потрібні змінні з шляхами, а також файли які потрібні для роботи фреймворка і були згенеровані ним.

Каталог “templates” містить у собі усі HMTL файли які потрібні для веб-застосунку і коректного його відображення.

3.1.2 Файлова структура веб-застосунку

Для правильної роботи веб-застосунку потрібні різні файли, назви файлів Python з основного каталогу і їх описи на веденні в таблиці 3.1[32].

Таблиця 3.1 – Опис файлів основного каталогу веб-застосунку

Назва файлу	**Опис файлу**
manage.py	Основний керуючий скрипт Django. Використовується для запуску сервера розробки, виконання міграцій бази даних, тощо.
db.sqlite3	Файл бази даних.
requirements.txt	Файл з назвами пакетів потрібних для функціонування застосунку і їх версії.

Як бачимо в основному каталозі залишилися тільки три файли, які є основними для функціонування і запуску проекту. Файли з каталогу bloodbankmanagment наведені у таблиці 3.2.

Таблиця 3.2 – Опис файлів з каталогу bloodbankmanagment

Назва файлу	**Опис файлу**
settings.py	Файл конфігурації, де визначені всі налаштування проекту, такі як бази даних, встановлені додатки, середовище тощо.
urls.py	Основний файл маршрутизації URL для проекту. Визначає URL-шляхи та відповідні їм обробники.
wsgi.py	Вхідна точка для WSGI-сумісних веб-серверів для розгортання проекту.
asgi.py	Вхідна точка для ASGI-сумісних веб-серверів для розгортання проекту (якщо проект підтримує ASGI).
init.py	Файл, який позначає каталог як Python-пакет.

Усі файли в цьому каталозі автоматично згенеровані за допомогою фреймворку Django, тут ми бачимо файл конфігурації проекту, який під’єднує усі решта каталоги та базу даних. Враховуючи, що каталоги “blood”, “donor”, “patient” мають однакову структуру назва і опис їх файлів наведено у таблиці 3.3.

Таблиця 3.3 – Опис файлів каталогів “blood”, “donor”, “patient”

Назва файлу	**Опис файлу**
init.py	Файл, який позначає каталог як Python-пакет.
admin.py	Файл для реєстрації моделей у адміністративному інтерфейсі Django.
apps.py	Конфігураційний файл додатка
models.py	Файл для визначення моделей бази даних
views.py	Файл для створення представлень (views), які обробляють запити та повертають відповіді.
urls.py	Файл маршрутизації URL для конкретного додатка.
migrations/	Каталог для управління міграціями бази даних.

3.2 Розробка інтерфейсу і тестування

Перше що бачить користувач заходячи на сайт, це “головна” або “домашня” сторінка сайту, метою якої є подальша його навігація, для авторизації/реєстрації на сайті[33]. На головній сторінці ми бачимо навігаційну панель, яка дає змогу користувачу перейти до форм авторизації. Вигляд навігаційної панелі зображено на рисунку 3.2.

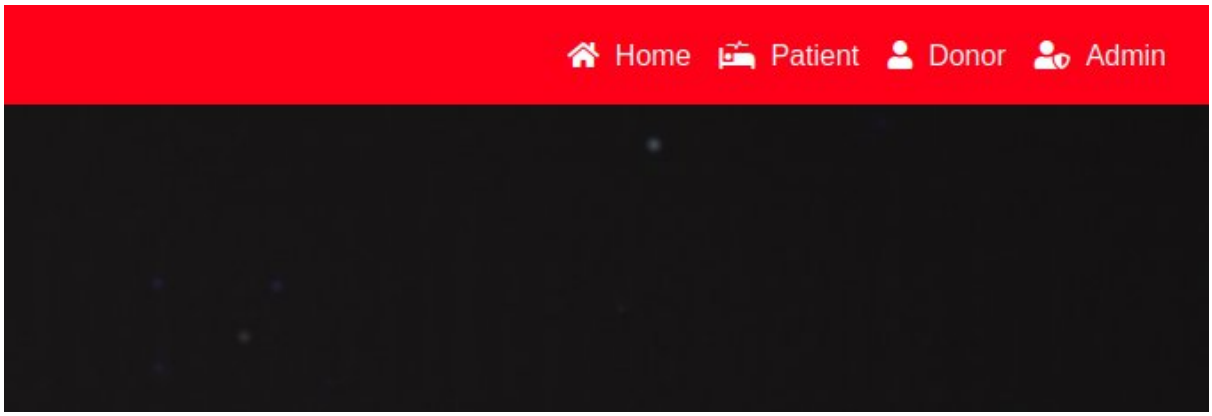

Рисунок 3.2 – Навігаційна панель головної сторінки веб-застосунку

Звідси користувач може перейти до форми авторизації відподно до свого аккаунту. На рисунку 3.3 наведенно форму авторизації для пацієнта.

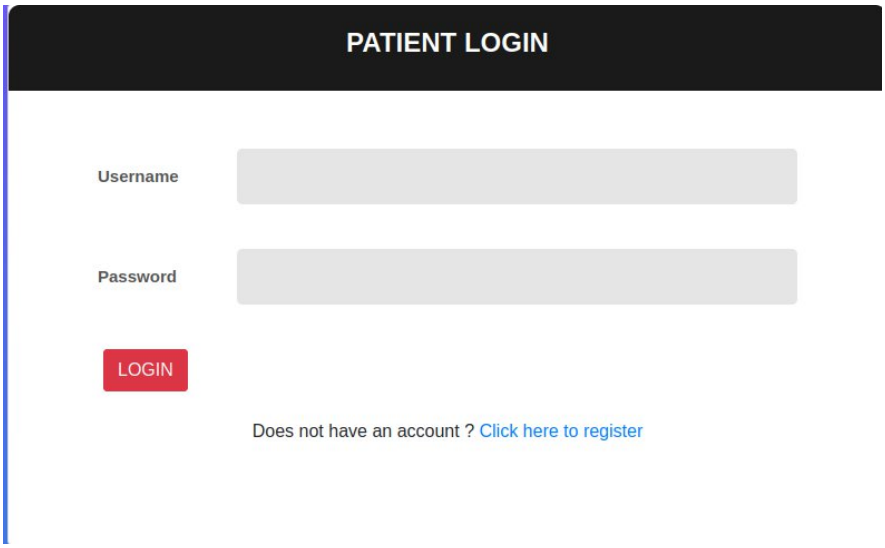

Рисунок 3.3 – Форма авторизації

Як можна побачити форма авторизації доволі проста і не відрізняється для донора, єдина різниця це авторизація для адміна, оскільки його аккаунт ми створюємо перед запуском веб-застосунку командою “python manage.py createsuperuser”. Якщо ж користувач не має аккаунту він може його створити перейшовши за посиланням реєстрації. Поле реєстрації наведено на рисунку 3.4.

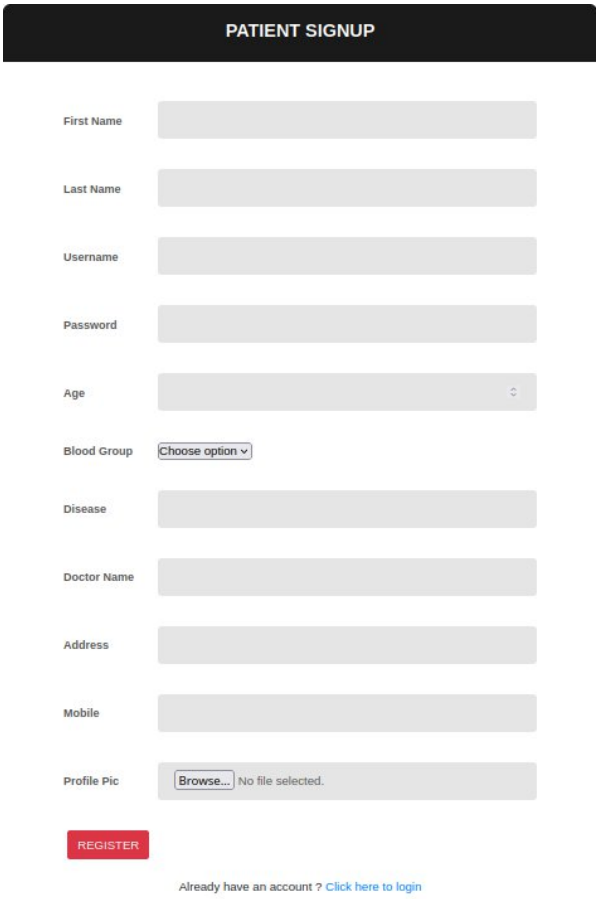

Рисунок 3.4 – Поле реєстрації пацієнта

Як бачимо в полі реєстрації пацієнта є поля для усією потрібної медичним працівникам інформації. Вибір групи крові реалізований через випадаючий список, а також можна завантажити зображення пацієнта, використовуючи стандартний провідник системи. Навідмінно, від авторизації, поля реєстрації для пацієнта і донора відрізняються. Поле реєстрації для донора наведенно на рисунку 3.5.

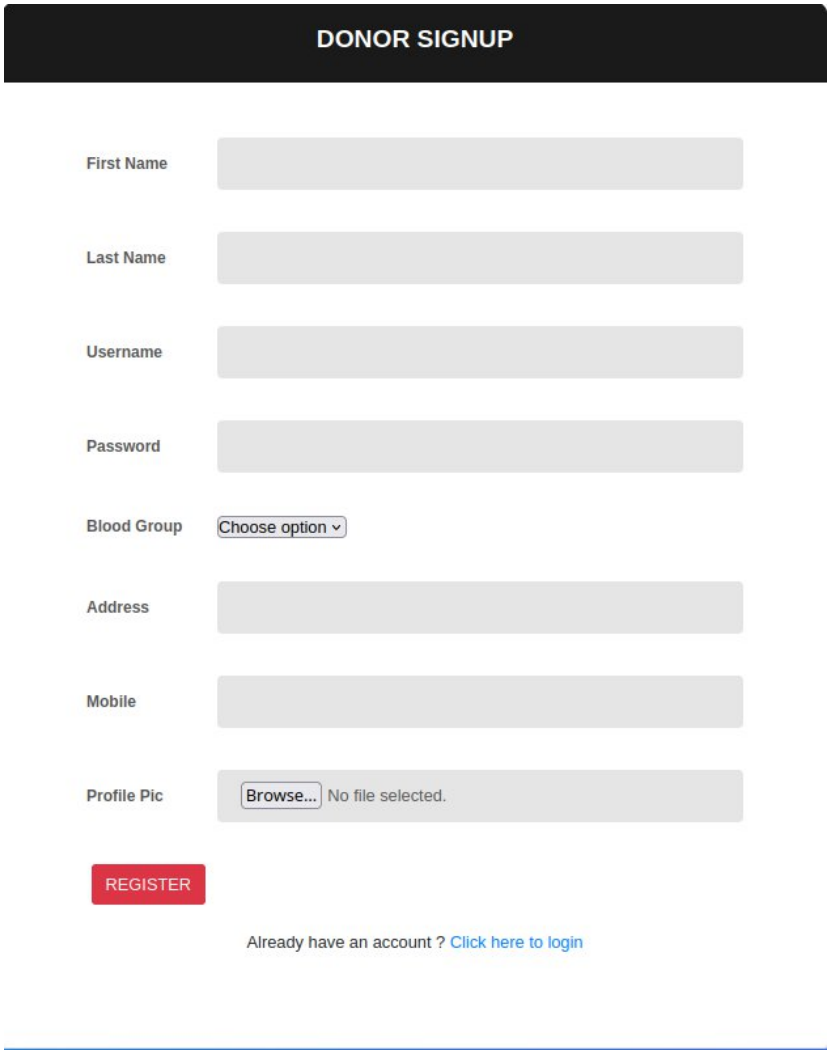

Рисунок 3.5 – Поле реєстрації донора

Як бачимо поле реєстрації донора є меншим, але зберігає усі потрібні поля інформації. Після входу панель донора і пацієнта також відрізняються. На рисунку 3.6 зображено панель пацієнта.

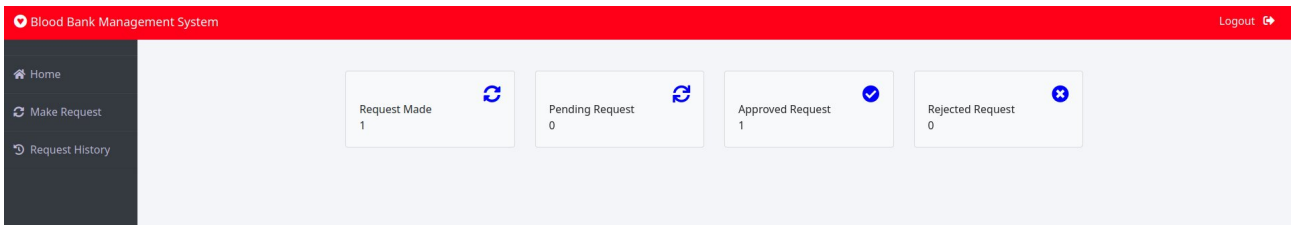

Рисунок 3.6 – Панель пацієнта

Пацієнта зустрічає сторінка з статисткою його запитів на переливання крові, також зліва є меню навігації, з якого можна перейти на форму подачі

заявки на переливання, а також на сторінку з історією усіх запитів користувача. На рисунку 3.7 наведено форму подачі запиту на переливання крові.

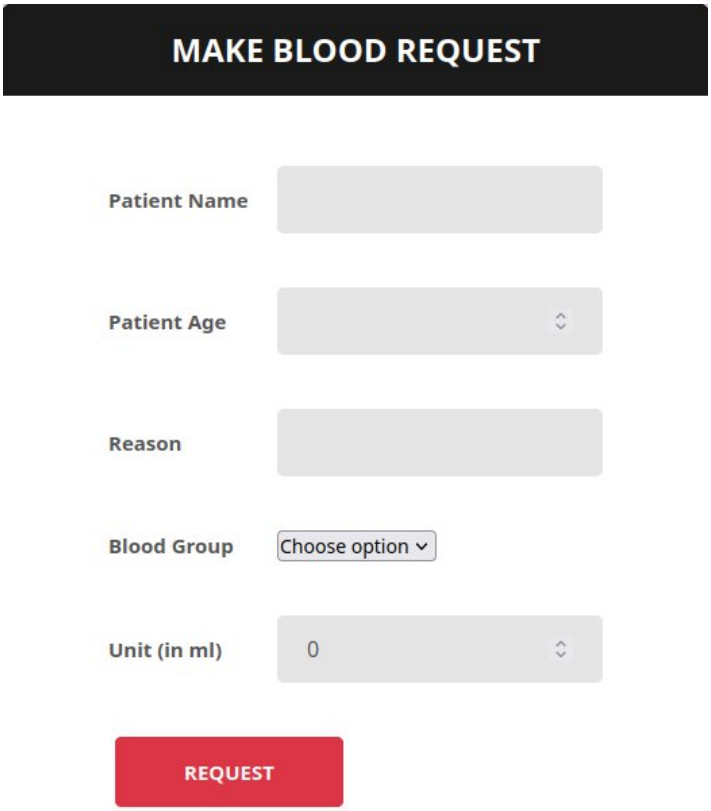

Рисунок 3.7 – Форма подачі запиту на переливання крові

Форма доволі проста і швидка для заповнення, що дозволяє прискорити прицес оформлення запиту. Після подачі запиту він відправляється на опрацювання адміністратором та записується в історію запитів користувача, сторінка якої наведена на рисунку 3.8.

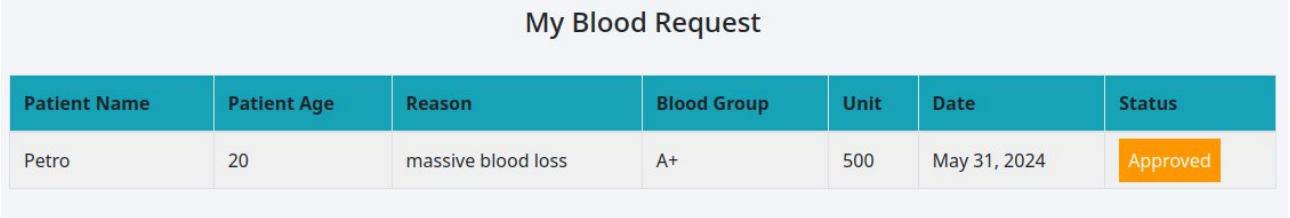

Рисунок 3.8 – Сторінка історії запитів користувача на переливання крові

Як можемо побачити крім тої інформації яку ми вводили, тут ми можемо побачити дату запиту, а також статус запиту, в даному випадку він був одобрений адміністратором, що означає, що переливання крові відбулось. Панель донора має дещо більше функцій, головна сторінка якої наведенна на рисунку 3.9.

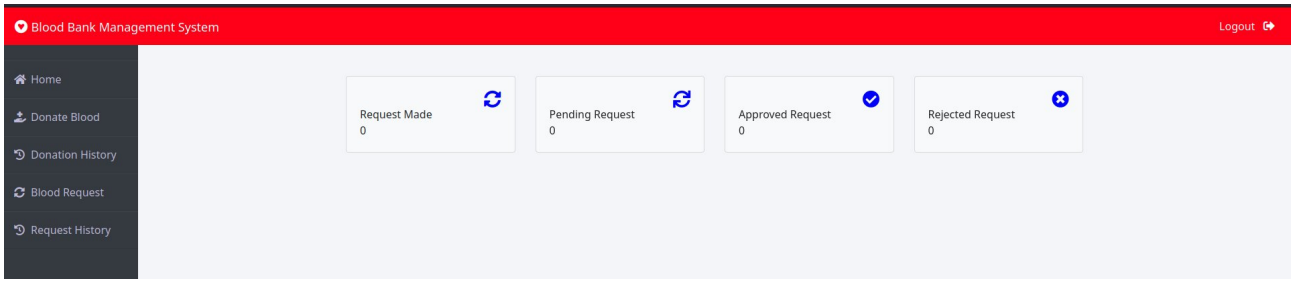

Рисунок 3.9 – Головна сторінка панелі донора

Початкова сторінка не відрізняється від сторінки пацієнта, вона також має статистику запитів на переливання. Проте на панелі навігації добавились посилання на форму донації крові та історію донацій користувача. Форма для запиту на донацію крові зображено на рисунку 3.10.

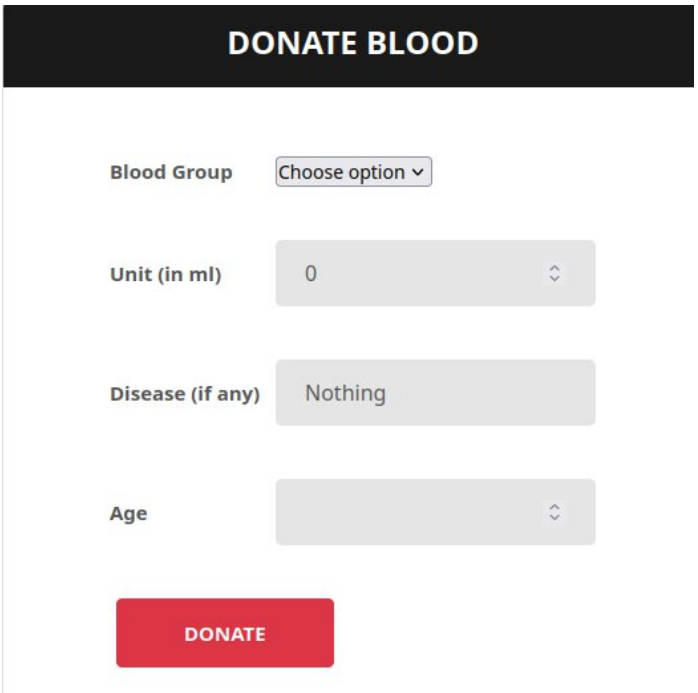

Рисунок 3.10 – Форма подання запиту на донацію крові

Форма подачі запиту на донацію крові є доволі простою і не вимагає введення лишньої інформації. Після того як запит відправлений він буде відображатись в історії донацій, сторінка якої зображена на рисунку 3.11.

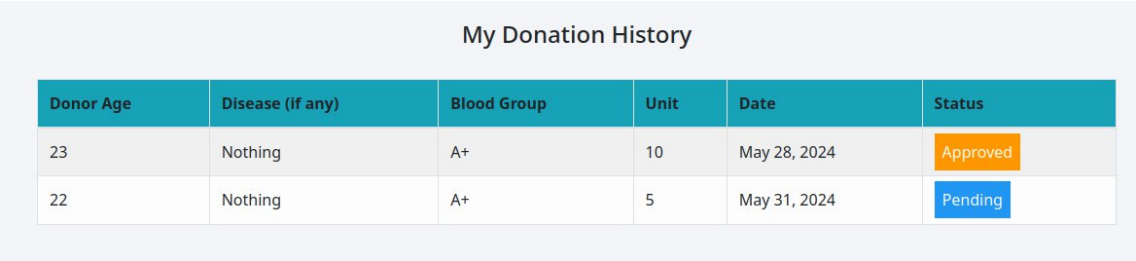

Рисунок 3.11 – Сторінка запитів на донацію крові

Тут користувач може переглянути історію своїх донацій крові, їх статус та дату подачі запиту. На рисунку 3.12 зображено домашню сторінку панелі адміністратора.

Рисунок 3.12 – Домашня сторінка панелі адміністратора

На домашній сторінці панелі адміністратора ми бачимо статистику запасів крові, кількість зареєстрованих донорів, загальну кількість і кількість підтверджених запитів на переливання крові, а також загальну кількість крові. На панелі навігації знаходяться посилання, на сторінки з аккаунтами донорів і пацієнтів, сторінку з запитами на донацію та переливання крові, історію переливань, а також переглянути і редагувати інформацію про запаси крові. На рисунку 3.13 наведено сторінку “Донор”.

Рисунок 3.13 – Сторінка з аккаунтами донорів

На сторінці з аккаунтами донорів адміністратор може переглянути, змінити або зовсім видалити інформацію про певного донора. На рисунку 3.14 наведенно сторінку “Пацієнт”

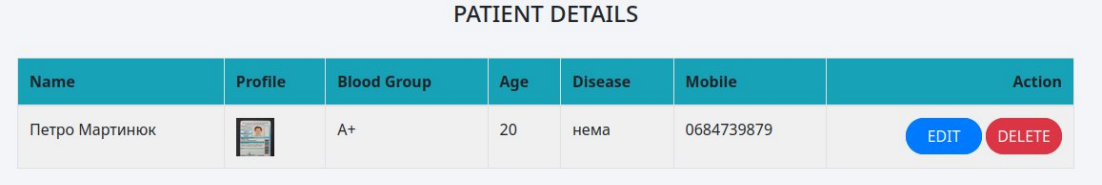

Рисунок 3.14 – Сторінка з аккаунтами пацієнтів

Як і на сторінці з донорами тут адміністратор має можливість переглянути, або за потреби змінити чи зовсім видалити інформацію про пацієнта. На рисунку 3.15 наведено сторінку з запитами на донацію крові.

Рисунок 3.15 – Сторінка з запитами на донацію крові

На цій сторінці адміністратор може переглянути історію запитів на донацію, а також одобрити, або відхилити запит. На рисунку 3.16 зображено сторінку з запитами на переливання крові.

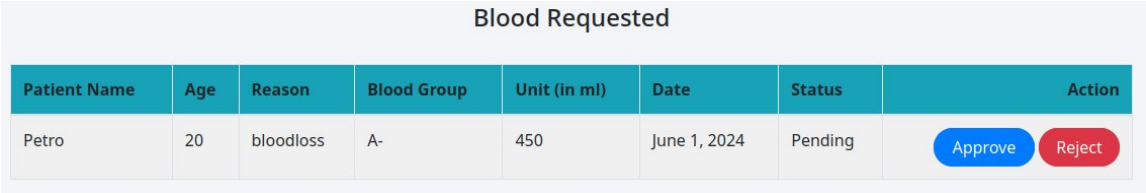

Рисунок 3.16 – Сторінка з запитами на переливання крові

На сторінці з запитами на переливання, адміністратор може переглянути інформацію про пацієнта, кількість потрібної крові, причину і дату подачі запиту, та прийняти рішення чи одобрити чи відхилити даний запит. На рисунку 3.17 подано сторінку історії запитів на переливання крові.

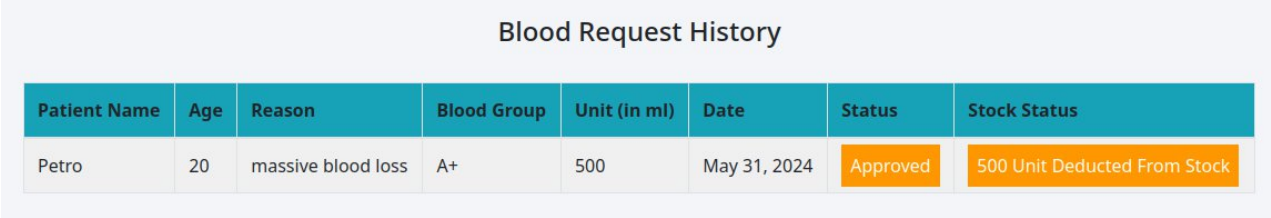

Рисунок 3.17 – Сторінка з історією запитів на переливання крові

На даній сторінці адміністратор може побачити історію усіх прийнятих або відхилених запитів на переливання крові. На рисунку 3.18 зображено сторінку з запасами крові.

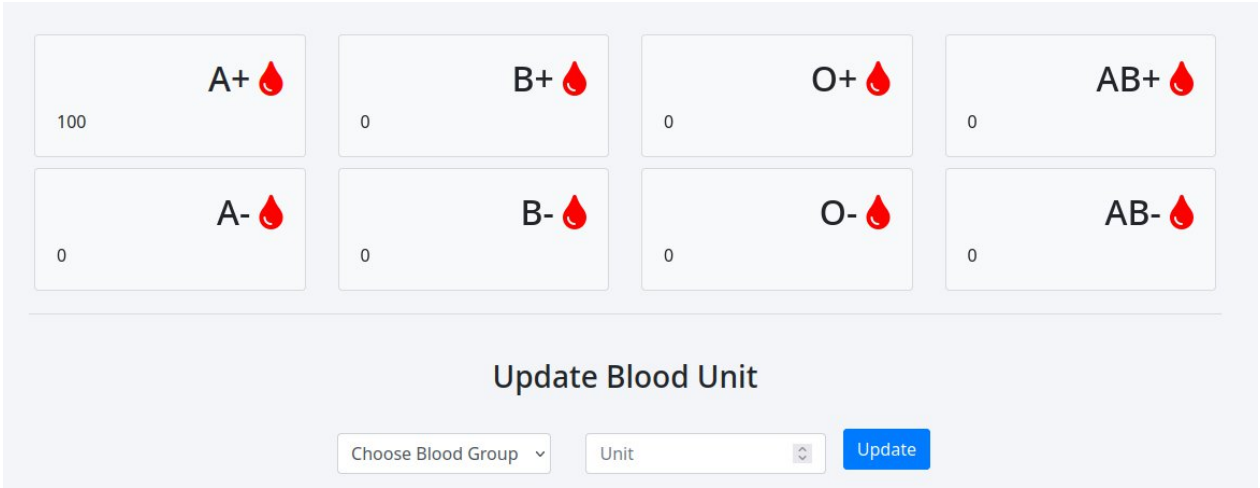

Рисунок 3.18 – Сторінка з запасами крові

На цій сторінці адміністратор може переглянути статистику по кількості крові певної групи, а також оновити цю інформацію.

3.3 Висновок до третього розділу

В третьому розділі кваліфікаційної роботи проаналізована структура файлової системи веб-застосунку, а саме: проведено аналіз каталогів проекту, їх призначення та наповнення; проведено аналіз файлової структури, назви файлів та їх короткий опис. Проаналізовано і описано сторінки веб-застосунку, а саме: головна сторінка сайту; форми авторизації та реєстрації користувачів; панелі навігації пацієнта та донора; форми подачі запитів на донацію та переливання крові, історія запитів; панель адміністрування, опрацювання запитів, статистика, та історія запитів.

РОЗДІЛ 4. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ТА ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Долікарська допомога при кровотечах

Кров у людському організмі циркулює через кровоносні судини, такі як артерії, вени і капіляри.

Кровотеча виникає, коли кров витікає з пошкоджених кровоносних судин внаслідок травми, такої як укол, розріз, удар або розтягнення. Інтенсивність кровотечі залежить від кількості пошкоджених судин, їх діаметра, характеру пошкодження та виду пошкодженої судини (артерія, вена, капіляр). нтенсивність кровотечі також може різнитися за рівнем артеріального тиску, видом кровотечі (зовнішня або внутрішня), віком постраждалої особи та її станом здоров'я. Втрата крові може призвести до гострої недостатності кровопостачання тканин і органів, таких як мозок, легені, серце, що може призвести до смерті[37].

У разі кровотечі рятувальники не повинні торкатися рани руками, промивати її водою або ліками, а також присипати порошком, оскільки існує ризик інфекції. Види зовнішньої кровотечі залежать від характеру ушкодження кровоносних судин (капілярів, вен, артерій) і можуть бути: капілярна кровотеча, венозна кровотеча, артеріальна кровотеча, кровотеча з рота, носа, вух тощо. Капілярна кровотеча виникає при поверхневих ранах або ушкодженнях шкіри. Її можна зупинити, сприяючи згортанню крові, накладаючи тугу стерильну ватно-марлеву пов'язку на рану. Бинт повинен йти знизу вгору від пальців до плечей. Венозна кровотеча виникає з глибоких ран і може бути інтенсивнішою. Кров має темно-червоний колір. Після знезараження шкіри навколо рани слід підняти ушкоджену кінцівку і накласти тугу пов'язку. Артеріальна кровотеча становить безпосередню загрозу життю людини і виникає при глибоких порізах або колотих ранах. Кров має яскраво-червоний колір і викидається струменем в ритмі пульсу (биття фонтану), через високий тиск. Якщо неможливо зупинити кровотечу за допомогою пов'язки, артерію можна притиснути до кістки, ближче

до серця, протягом 10-15 хвилин. Щоб зупинити кровотечу в рані, має з'явитися згусток крові через згортання. Можна накласти турнікет або скрутку (гумову трубку, краватку, рушник) на місце травми, яке знаходиться найближче до серця. Турнікет слід утримувати протягом 1-2 годин, після чого він відпускається на 10-15 хвилин, а потім знову стискається артерія, щоб запобігти некрозу тканин, але це може робити тільки кваліфікований медик. Кровотечу можна тимчасово зупинити, зігнувши кінцівку в колінному та тазостегновому суглобах. У випадку носової кровотечі потерпілий повинен сидіти, злегка нахилити голову та розстебнути комірці. На ніс, лоб і шию можна накласти змочену водою хустку або вставити тампон або марлю, змочену 3% розчином перекису водню, і затиснути ніс пальцями.

При кровотечі з рота потерпілого слід покласти в горизонтальне положення та негайно викликати лікаря. Те саме робиться при кровотечі з вух, оскільки це може бути ознакою порушення внутрішньочерепного тиску внаслідок травми голови. Внутрішня кровотеча є дуже небезпечною. Характерними ознаками є бліде обличчя, прискорений пульс, загальна слабкість, запаморочення, задишка, спрага, а також чорні плями на стегнах і животі у вигляді висипу. Потерпілий повинен перебувати у положенні напівсидячи із зігнутими в колінах ногами. Важливо заборонити потерпілому пити.

Турнікет накладається як найвище на кінцівку. На руку – максимально вище до пахви, на ногу – максимально високо до паху. Якщо ми знаємо де саме знаходиться рана, то накладаємо на 5-10 сантиментрів вище рани. Крутити ручку турнікета на кінцівці потрібно до повної зупинки кровотечі. Перевіряємо пульс на кінцівці нижче місця накладення турнікету. Відсутність пульсу – також контрольний показник вірно накладеного турнікету. Після зупинки кровотечі ручка турнікета заправляється у фіксатор в залежності від типу турнікета він може бути різний. Закручуємо залишки вільного кінця турнікета навколо накладеного турнікета. Фіксуємо всі елементи турнікета стрічкою для написання часу накладання турнікету та пишемо час накладання турнікету маркером. Час накладання турнікету – 2 години[38].

4.2 Вплив кольору на умови праці та продуктивність

Особливу роль в естетичній організації виробничого середовища відіграє колір, оскільки він справляє багатоплановий вплив на людину, а отже, має багатофункціональне призначення. Колір впливає на фізіологічні і психічні процеси, емоційні стани, працездатність і продуктивність праці працівників.

Колірний тон залежить від довжини хвилі, яка вимірюється в мілімікронах. Найдовші хвилі мають червоний та оранжевий кольори. Довжина хвилі фіолетового кольору найменша. Встановлено, що довгохвильові і короткохвильові кольори справляють несприятливий вплив на людину і викликають найбільшу зорову втому. Червоний колір, зокрема, діє як сильний подразник і збуджує нервову систему, а фіолетовий викликає пригнічений настрій. Середньохвильові кольори заспокійливо впливають на нервову систему, сприяють зниженню втоми[39].

Сила впливу різних кольорів на людину залежить від їх насиченості та яскравості. Насичені кольори покращують настрій і стимулюють роботу аналізаторів, ненасичені та малонасичені кольорові відтінки діють заспокійливо, сприяють зосередженню уваги. Світлі кольори, яскраві і насичені також покращують настрій, а темні – викликають песимістичні настрої. Виходячи з цих властивостей кольори використовують як засіб інформації для орієнтування працівників у виробничому середовищі та устаткуванні з метою дотримання ними техніки безпеки. Стандартом визначені такі значення кольорів: червоний – заборона, безпосередня небезпека; жовтий – попередження, можлива небезпека; зелений – безпечно; синій – інформація.

Крім цього, окремі кольори, їх відтінки і поєднання використовуються як додатковий фактор поліпшення освітленості приміщень, для створення необхідного контрасту в полі зору працівника між предметом і фоном, зниження монотонності роботи і втоми, забезпечення психологічного комфорту, підвищення працездатності і продуктивності праці. На рисунку 4.1 наведено вплив кольору на організм людини.

Колір	Артеріальний тиск	Пульс	Частота дихань	Емоційний стан	Час реакції	Працездатність
Червоний	Підвищується	Збільшується	Збільшується	Збуджує, стимулює	Зменшується	Спочатку підвищується ненадовго, а потім зменшується наполовину
Оранжевий	Мало підвищується	Трохи збільшується	Трохи збільшується	Збуджує, бадьорить	Трохи зменшується	Так само, але менш виражено
Жовтий	Не змінюється	Не змінюється	Не змінюється	Урівноважує	Не змінюється	Суттєво не змінюється
Зелений	Не змінюється	Не змінюється	Не змінюється	Урівноважує	Не змінюється	Трохи підвищується
Блакитний	Дещо знижується	Трохи сповільнюється	Трохи сповільнюється	Заспокоює	Суттєво не змінюється	Трохи підвищується
Синій	Знижується	Сповільнюється	Сповільнюється	Заспокоює, трохи загальмовує	Трохи сповільнюється	Підвищується на 3—9 %
Фіолетовий	Знижується	Сповільнюється	Сповільнюється	Пригнічує	Виразно сповільнюється	Стабільно знижується

Рисунок 4.1 – Вплив кольору на організм людини

Психологічно кольори сприймають як теплі і холодні. Теплі кольори (червоний, оранжевий, жовтий) викликають психологічне відчуття тепла, стимулюють діяльність нервової системи, зосередження уваги та короткотривале підвищення продуктивності праці. Холодні кольори (синій, блакитний, зелений) викликають психологічне відчуття холоду, заспокоюють, полегшують напруження очей, сприяють зосередженню уваги.

Вибір колірного оформлення виробничих приміщень залежить від багатьох факторів — м’язових і нервових навантажень, температурного режиму, розмірів та орієнтації приміщення, монотонності роботи. Так, на роботах, які вимагають великих фізичних і нервових навантажень, а також у цехах з високою температурою повітря доцільно використовувати світлі тони голубого, зеленого та інших спокійних холодних кольорів невеликої насиченості. Якщо робота вимагає лише періодичних значних розумових і фізичних навантажень, то вона легше виконується у приміщеннях, пофарбованих у теплі кольори, які підвищують активність організму. Виконання монотонних робіт більш ефективне, якщо приміщення пофарбувати у яскраві кольори, які привертають увагу працівників і розширюють поле коркової активності.

Для раціонального колірного оформлення виробничих приміщень розроблені відповідні стандарти і документи, в яких містяться рекомендовані поєднання основних та допоміжних кольорів, коефіцієнти відображення світла тощо.

4.3 Висновок до четвертого розділу

В четвертому розділі кваліфікаційної роботи описано питання щодо безпеки життєліяльності, та питання з основ охорони праці. З безпеки життєдіяльності було обрано та описано тему “ Долікарська допомога при кровотечах”, а саме: проаналізовано основні види кровотеч, причини їх винекнення, їх інтенсивність взалежності від різних факторів та наслідки кровотеч. Також, правила та порядок дій при надані допомоги людині з кровотечею. З охорони праці було обрано та описано тему “Вплив кольору на покращення умов праці та підвищення продуктивності виробництва.”, а саме проаналізовано вплив певних кольорів на організм людини.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи освітнього рівня “Бакалавр” було розроблено веб-застосунок для супроводу процесів донорства крові. Для розробки було використано мову програмування Python, мову розмітки HTML та CSS для стилізації сторінок. Текстовим редактором було обрано PyCharm.

Веб-застосунок для супроводу донорства крові є зручним як для пацієнтів так і донорів, для подачі запитів на донацію і донорство крові. Має простий та інтуїтивно зрозумілий інтерфейс користувача та адміністратора.

В першому розділі кваліфікаційної роботи:

- Проаналізовано предметну область.
- Сформовано вимоги до веб-застосунку.
- Описано актанти та варіанти використання веб-застосунку.
- Вибрано середовище розробки, а саме мова програмування Python, мову розмітки HTML та CSS для стилізації сторінок. Текстовим редактором було обрано PyCharm.

В другому розділі кваліфікайної роботи:

- Змодельовано архітектуру веб-застосунку.
- Спроектовано структуру веб-застосунку.
- Спроектовано базу-даних.
- Спроектовано візуальний вигляд застосунку.

В третьому розділі кваліфікаційної роботи:

- Структуровано каталоги веб-застосунку.
- Описано файлову структуру веб-застосунку.
- Розроблено і протестовано інтерфейс веб-застосунку.

У розділі безпеки життєдіяльності та основи охорони праці було проаналізовано і описано питання домедичної допомоги при кровотечах та вплив кольору на покращення умов праці та підвищення продуктивності виробництва.

ПЕРЕЛІК ДЖЕРЕЛ

1. Діаграма варіантів використання UML: підручник із прикладом. Guru99. URL: https://www.guru99.com/uk/use-case-diagrams-example.html.

2. Казимир В. В. Об"єктно-орієнтоване програмування : навч. посіб. для студентів вищ. навч. закл. Київ : Слово, 2008. 192 с.

3. UML Use Case Diagram Tutorial. Lucidchart. URL: https://www.lucidchart.com/pages/uml-use-case-diagram.

4. Python tutorial | learn python programming - geeksforgeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/python-programming-language-tutorial/.

5. Advantages of python | disadvantages of python - python geeks. Python Geeks. URL: https://pythongeeks.org/advantages-disadvantages-of-python/.

6. Python: плюси і мінуси мови, які завдання вирішує і чи варто вивчати. Avada Media. URL: https://avada-media.ua/ua/services/python-plyusy-i-minusy-yazyka-kakiye-zadachi-reshayet-i-stoit-li-izuchat/.

7. Django introduction - Learn web development | MDN. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction.

8. Плюси і мінуси Django. Укрнеймс.БЛОГ. URL: https://blog.ukrnames.com/veb-master/plyusi-i-minusi-django.

9. Advantages of django | disadvantages of django - python geeks. Python Geeks. URL: https://pythongeeks.org/advantages-disadvantages-of-django/.

10. HTML tutorial. W3Schools Online Web Tutorials. URL: https://www.w3schools.com/html/.

11. Advantages and Disadvantages of HTML - GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/advantages-and-disadvanatges-of-html/.

12. CSS tutorial. W3Schools Online Web Tutorials. URL: https://www.w3schools.com/Css/.

13. Advantages and Disadvantages of CSS - GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/advantages-and-disadvantages-of-css/.

14. CSS: Cascading Style Sheets | MDN. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/CSS.

15. The SQLite Handbook: A Start-to-Finish Resource for Learning and Using SQLite – SQL Docs. SQL Documentation. URL: https://sqldocs.org/sqlite/introduction/.

16. SQLite advantages and disadvantages - javatpoint. www.javatpoint.com. URL: https://www.javatpoint.com/sqlite-advantages-and-disadvantages.

17. Wong C. Advantages and disadvantages of using sqlite. Medium. URL: https://medium.com/@cw30355/advantages-and-disadvantages-of-using-sqlite-2f490fa467bd.

18. What is PyCharm? – GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/what-is-pycharm/.

19. What is pycharm? Features, advantages & disadvantages. Hackr.io. URL: https://hackr.io/blog/what-is-pycharm.

20. Zmerzlyi I. Архітектура веб додатків. Medium. URL: https://medium.com/@IvanZmerzlyi/архітектура-веб-додатків-ca4c82f75bcf.

21. Учасники проектів Вікімедіа. Тришарова архітектура – Вікіпедія. Вікіпедія. URL: https://uk.wikipedia.org/wiki/Триярусна_архітектура.

22. Що таке структура сайту і як її зробити: види та приклади схем – 107.com.ua. 107.com.ua – Linux Windows Software Tecnologies Seo CMS. URL: https://107.com.ua/blog/sho-take-stryktyra-saity-i-iak-її-zrobiti-vidi-ta-prikladi-shem/.

23. Мікула М., Коцюк Ю., Мікула О. Організація баз даних та знань. Вид-во Нац. ун-ту «Острозька акад.», 2021. URL: https://doi.org/10.25264/978-617-8041-08-3.

24. Що таке бізнес-правила? - Ласкаво просимо IT-професіонал Визначення. Welcome IT professional. URL: https://uk.itpedia.nl/2019/04/17/wat-zijn-business-rules/.

25. How to use sessions | Django documentation. Django Project. URL: https://docs.djangoproject.com/en/5.0/topics/http/sessions/.

26. Khan M. The power of django session with example: a hands-on guide. Medium. URL: https://medium.com/django-unleashed/the-power-of-django-session-with-example-a-hands-on-guide-b46a848e4b27.

27. Django database migrations - javatpoint. www.javatpoint.com. URL: https://www.javatpoint.com/django-database-migrations.

28. Creating Tables and Schema Design. Welcome to Kev's Robots. URL: https://www.kevsrobots.com/learn/sqlite3/03_creating_tables_and_schema.html.

29. Основи web-дизайну. З чого починається web-дизайн?. Захищена сторінка. URL: https://webstudio2u.net/ua/design-web/342-webdesign-start.html.

30. Key Elements Of Successful Web App Design - Jarvee. Jarvee. URL: https://jarvee.com/key-elements-of-successful-web-app-design/.

31. Perez J. C. S. Django project structure: a comprehensive guide. Medium. URL: https://medium.com/django-unleashed/django-project-structure-a-comprehensive-guide-4b2ddbf2b6b8.

32. Django Project Structure and File Structure – TechVidvan. TechVidvan. URL: https://techvidvan.com/tutorials/django-project-structure-layout/.

33. Тестування графічного інтерфейсу користувача – тестові приклади інтерфейсу користувача (приклади). Guru99. URL: https://www.guru99.com/uk/gui-testing.html.

34. Additive mathematical model of gas consumption process / Iaroslav Lytvynenko, Serhii Lupenko, Oleh Nazarevych, Hryhorii Shymchuk, Volodymyr Hotovych // Scientific Journal of TNTU. – Tern. : TNTU, 2021. – Vol 104. – No 4. – P. 87–97.

35. O. Nazarevych, Y. Leshchyshyn, S. Lupenko, V. Hotovych, G. Shymchuk and N. Shabliy, «Method of Gas Consumption Change-point Detection Based on Seasonally Multicomponent Model,» 2020 10th International Conference on Advanced Computer Information Technologies (ACIT), Deggendorf, Germany, 2020, pp. 152-155, doi: 10.1109/ACIT49673.2020.9208924.

36. Y. Leshchyshyn, L. Scherbak, O. Nazarevych, V. Gotovych, P. Tymkiv and G. Shymchuk, «Multicomponent Model of the Heart Rate Variability Change-point,»

2019 IEEE XVth International Conference on the Perspective Technologies and Methods in MEMS Design (MEMSTECH), Polyana, Ukraine, 2019, pp. 110-113, doi: 10.1109/MEMSTECH.2019.8817379.

37. "Безпека життєдіяльності" Желібо Є.П., Зацарний В.В., 2018. 344 с.

38. 6 правил накладання турнікету. Досвід. Надійність. Іновації. URL: https://www.tourniquet.com.ua/6-pravyl-nakladannya-turniketu/.

39. ДБН В.2.5-28 : 2018. „Природне і штучне освітлення”. URL: https://ledeffect.com.ua/images/__branding/dbn2018.pdf

ДОДАТКИ

Додаток А

Конфігураційний файл проекту “settings.py”

```
"""
Django settings for bloodbankmanagement project.
Generated by 'django-admin startproject' using Django 3.0.5.
For more information on this file, see
https://docs.djangoproject.com/en/3.0/topics/settings/
For the full list of settings and their values, see
https://docs.djangoproject.com/en/3.0/ref/settings/
"""
import os
# Build paths inside the project like this:
os.path.join(BASE_DIR, ...)
BASE_DIR =
os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
TEMPLATE_DIR = os.path.join(BASE_DIR,'templates')
STATIC_DIR=os.path.join(BASE_DIR,'static')
MEDIA_ROOT=os.path.join(BASE_DIR,'static')
# Quick-start development settings - unsuitable for production
# See
https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/
# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY =
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True
ALLOWED_HOSTS = []
# Application definition
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
```

```
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'widget_tweaks',
    'blood',
    'donor',
    'patient',
]
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]
CSRF_COOKIE_SECURE=False
ROOT_URLCONF = 'bloodbankmanagement.urls'

TEMPLATES = [
    {
      'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [TEMPLATE_DIR,],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
```

```
            'django.contrib.messages.context_processors.messages',
            ],},},]
WSGI_APPLICATION = 'bloodbankmanagement.wsgi.application'
# Database
# https://docs.djangoproject.com/en/3.0/ref/settings/#databases
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }}
# Password validation
#         https://docs.djangoproject.com/en/3.0/ref/settings/#auth-
password-validators
AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
'django.contrib.auth.password_validation.UserAttributeSimilarityVa
lidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME':
'django.contrib.auth.password_validation.NumericPasswordValidator'
,
```

```
    },
]


# Internationalization
# https://docs.djangoproject.com/en/3.0/topics/i18n/
LANGUAGE_CODE = 'en-us'
TIME_ZONE = 'UTC'
USE_I18N = True
USE_L10N = True
USE_TZ = True
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.0/howto/static-files/
STATIC_URL = '/static/'
STATICFILES_DIRS=[
STATIC_DIR,
 ]
LOGIN_REDIRECT_URL='/afterlogin'
#for contact us give your gmail id and password
EMAIL_BACKEND ='django.core.mail.backends.smtp.EmailBackend'
EMAIL_HOST = 'smtp.gmail.com'
EMAIL_USE_TLS = True
EMAIL_PORT = 587
EMAIL_HOST_USER = 'from@gmail.com' # this email will be used to send
emails
EMAIL_HOST_PASSWORD = 'xyz' # host email password required
# now sign in with your host gmail account in your browser
# open following link and turn it ON
# https://myaccount.google.com/lesssecureapps
# otherwise you will get SMTPAuthenticationError at /contactus
```

```
# this process is required because google blocks apps authentication
by default
EMAIL_RECEIVING_USER = ['to@gmail.com'] # email on which you will
receive messages sent from website
```

Додаток Б

Головна сторінка сайту “index.php”

```
<!DOCTYPE html>
<head>
    <style>
        .xyz{
          margin-bottom: 0px;
        background-image: url('{% static "image/homepage.jpg" %}');
          background-attachment: fixed;
          background-size: 100%,100%;
          background-repeat: no-repeat;
        }
      </style>
    </head>
<body>
  {% include "blood/navbar.html" %}
<br>
<section    id="section-jumbotron"    style="margin-bottom:    3%;"
class="jumbotron   jumbotron-fluid   d-flex   justify-content-center
align-items-center xyz">
    <div class="container text-center">
        <br><br><br><br><br><br><br><br><br><br><br><br><br><br>
      <br><br><br><br>   <br><br><br><br><br><br>


    </div>
  </section>
  <div class="jumbotron" style="margin-top: -50px;margin-bottom: -
50px; height:50%;">
    <p  class="lead  text-center">“Opportunities  knock  the  door
sometimes, so don’t let it go and donate blood.”
```

```
      </p>
        <p class="text-center">- LazyCoder</p>
      </div>
  {% include "blood/footer.html" %}
</body>
</html>
```