КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТімені ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра прикладних інформаційних систем

122 «Комп'ютерні науки»
(шифр і назва спеціальності)

«Прикладне програмування»
(назва освітньої програми)

Кваліфікаційна робота бакалавра

на тему: «Веб-система підбору медикаментів та підтримки лікарів в онлайн режимі»

Виконала _____
(Підпис)
Овчиннікова Анна Сергіївна
(прізвище, ім'я, по батькові)

Керівник Міронова Вікторія Леонідівна
(прізвище, ім'я, по батькові)

_____
(Резолюція «До захисту»)

Попередній захист:

_____

_____
(Висновок: "До захисту в екзаменаційній комісії ")

Завідувач кафедри _____ Плескач В.Л.
(Підпис) (Прізвище, ініціали)

(Дата)

Київ – 2021

Календарний план

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів проекту	Примітка
1	*Вибір теми та наукового керівника кваліфікаційної роботи бакалавра*	26.10.2020	виконано
2	*Видача завдання кваліфікаційної роботи бакалавра*	23.11.2020	виконано
3	*Настановча групова співбесіда з питань кваліфікаційної роботи бакалавра*	01.12.2020	виконано
4	*Затвердження плану кваліфікаційної роботи бакалавра*	18.02.2021	виконано
5	*Підбір та вивчення літературних та інших джерел з теми дослідження*	25.02.2021	виконано
6	*Підготовка і подання науковому керівнику першого варіанту I розділу роботи*	05.03.2021	виконано
7	*Підготовка і подання науковому керівнику першого варіанту II розділу роботи*	09.04.2021	виконано
8	*Підготовка і подання науковому керівнику першого варіанту III розділу роботи*	07.05.2021	виконано
9	*Подання роботи у першому варіанті*	11.05.2021	виконано
10	*Оформлення пояснювальної записки кваліфікаційної роботи бакалавра*	12.05.2021	виконано
11	*Подання кваліфікаційної роботи бакалавра на попередній захист*	**24.05.2021**	виконано
12	*Врахування зауважень керівника і подання роботи в остаточному варіанті (з відповідним висновком про допуск) на кафедру*	28.05.2021	
13	*Затвердження роботи в цілому (підготовка письмового відгуку керівника, письмова рецензія на бакалаврської роботу)*	11.06.2021	
14	*Захист кваліфікаційної роботи бакалавра*	25.06.2021	

Студент _____ _____
(підпис) (ініціали, прізвище)

Керівник проекту _____ _____
(підпис) (ініціали, прізвище)

ВІДОМІСТЬ ДИПЛОМНОГО ПРОЕКТУ

Складові частини дипломного проекту	Обсяг,арк.
Титульний аркуш	1
Завдання до дипломного проекту(календарний план проекту)	2
Відомість дипломного проекту	1
Анотація	1
Анотація(іноземною мовою-англійською)	1
Зміст	2
Перелік умовних позначень, символів, одиниць, скорочень і термінів	1
Вступ	2
1 Загальнотеоретичні питання веб-систем	28
2 Існуючі веб-системи підбору медикаментів в онлайн режимі	10
3 Опис вимог до розробки веб-системи підбору мендикаментів в онлайн режимі	7
4 Представлення веб-системи для підбору медикаментів та підтримки лікарів в онлайн-режимі	10
Висновки	1
Перелік посилань	2
Додатки	5

				ДП ХХХХ 00.000.00		
	ПІБ	Підп.	Дата			
Розробн.	Овчиннікова			Відомість дипломного проекту	Лист	Листів
Керівн.	Міронова				1	67
Н/контр.						
Зав.каф.	Плескач					

Анотація

Бакалаврської дипломної роботи Овчиннікової Анни Сергіївни на тему: «Веб-система підбору медикаментів та підтримки лікарів в онлайн режимі».

Дипломна робота присвячена розробці веб-системи для підбору медикаментів та підтримці лікарів в онлайн режимі, що включає в себе: розробку клієнтськоїчастини веб-системи; розробку серверної частини веб-системи; створення авторизації та реєстрації медичного працівника; створення профілю лікаря, з можливостями зберігати медикаменти та заповнювати журнал своїх пацієнтів.

Тема роботи є актуальною у зв'язку із підвищенням попиту на веб-системи та на пошук інформації онлайн. Тому найпростіше та найшвидше підібрати той чи інший медикамент, безпосередньо згідно інструкцій, можна в пошуковій системі в онлайн-режимі.

Наукова новизна результатів дослідження полягає в створенні веб-системи саме для медичних працівників, де вони можуть коментувати медикаменти і бачити коментарі інших кваліфікованих працівників, що зможе поліпшити вибір медикаментів при призначенні лікування.

Практичне завдання та значущість роботи полягає в розробленні веб-системипідбору медикаментів та підтримки лікарів в онлайн режимі, яка матимевеликі переваги перед своїми конкурентами, адже буде оптимізована та матиме функції орієнтовані на кваліфікованого медичного парцівника.

Загальний обсяг роботи: 80 с., 40 рис., 21 джерел, 2 дод.

Ключові слова: Веб-системи, Підбір медикаментів, MedSearch, Backend, Frontend.

Annotation

Bachelor degree Ovchynnikova Anna Sergiivna on: «Web system for the selectionof medicines and support for doctors online»

Thesis is devoted to the development of a web system for the selection of drugs andsupport for doctors online, which includes: development of the client part of the websystem; development of the server part of the web system; creation of authorizationand registration of a medical worker; creating a doctor's profile, with the ability to store medications and fill out a journal of their patients.

The topic of the work is relevant due to the growing demand for web systems and online information retrieval. Therefore, the easiest and fastest way to find a particular drug, directly according to the instructions, is in the search engine online.

The scientific novelty of the results of the study is to create a web system specificallyfor health professionals, where they can comment on medications and see the comments of other skilled workers, which can improve the choice of medications when prescribing treatment.

The practical task and significance of the work is to develop a web system for drugselection and support of doctors online, which will have great advantages over its competitors, because it will be optimized and will have functions focused on a qualified medical staff.

Total volume of work: 80 pp., 40 figs., 21 sources, 2 appendices. Keywords: Web systems, Drug selection, MedSearch, Backend, Frontend.

ЗМІСТ

Перелік умовних позначень

ПЗ – програмне забезпечення.

ПК – персональний комп’ютер.

ІТ – інформаційні технології.

ПК – персональний комп’ютер;

JS – JavaScript;

ПЗ – програмне забезпечення;

JWT - JSON Web Token.

ОС – операційна система.

Фреймворк – інфраструктура програмних рішень, що полегшує розробку складних систем. Спрощено дану інфраструктуру можна вважати своєрідною комплексною бібліотекою, але при цьому вона має ряд обмежень, що задають правила створення структури проєкту та написання коду.

Фронтенд – (від анг. Front-end) це інтерфейс для взаємодії між користувачем.

ВСТУП

Актуальність дослідження

Сьогодні наше життя піддається великому впливу зі сторони інформаційних технологій. Ми вже не можемо уявити своє життя без нових веб-сервісів.

У сучасному світі, на жаль, людство не застраховано від хвороб, і кожного може настигнути така біда. Медичному працівнику найпростіше та найшвидше підібрати той чи інший медикамент, безпосередньо згідно інструкцій, можна в пошуковій системі в онлайн-режимі, беручи до уваги рейтинг та коментарі продукуту.

Актуальність роботи полягає у полегшенні пошуку медичних препаратів, можливості лікарям вести особистий журнал пацієнтів, залишати свою рецензію, оцінку та коментарі будь-якому продукту.

Мета дослідження

Метою є удосконалення процесу підбору медикаментів та підтримки лікарів в онлайн-режимі.

Завдання

- дослідити теоретичні основи з теми веб-сервісів;
- розглянути існуючі концепції веб-систем для підбору медикаментівта шляхи удосконалення;
- описати вибране програмне забезпечення для створення веб-сервісу;
- розробити веб-систему для підбору медикаментів згідно описанихвимог розробки.

Об’єкт дослідження

Об’єктом дослідження є процес підбору медикаментів в онлайн-режимі.

Предмет дослідження

Предметом є принципи і засоби створення веб-системи підбору медикаментів в онлайн режимі.

Методи дослідження: Теоретичний науковий метод для аналізу існуючих систем пошуку ліків, та сучасних технологій розробки он-лайнсистем.

Епміричний науковий метод для виявлення закономірностей пошуку іаналізу та формування алгоритмів розробленої системи.

Аналіз недоліків у пошукових системах підбору медикаментів.

Практичне значення

Розроблена веб-система заснована на алгоритмах найлегшого пошуку і дозволяє користувачам (медпрацівника) швидко шукати та зберігати медичніпрепарати.

Структура роботи. Дипломна робота складається зі вступу, чотирьох розділів, розподілених на підрозділи, висновки, додатки та списку використаних джерел.

РОЗДІЛ 1
ЗАГАЛЬНОТЕОРЕТИЧНІ ПИТАННЯ ВЕБ-СЕРВІСІВ

1.1 Веб-система як ресурс інформації та послуг

Розробка веб-системи полягає у використанні комп'ютерного коду для створення програмного забезпечення, орієнтованого на користувача, у вигляді веб-сайтів. Розробники веб-систем часто більше займаютьсянедизайнерськими аспектами веб-розробки, хоча це не завжди так. Веб- розробники використовують кодування та розмітку для написання інтерактивних веб-сторінок. Ці сторінки можуть бути простими, як текстові файли в Інтернеті, або такими ж складними, як веб-сайти електронної комерції. Незалежно від кінцевого продукту, якщо користувачі отримують донього доступ у своєму веб-браузері, тоді є велика ймовірність того, що його створив - принаймні частково - веб-розробник.

Безпосередня різниця між веб-системою та просто сайтом, як на мене, полягає в динаміці та функціональності. Для мене сайт - це в першу чергу щось інформаційне та статичне: візитка компанії, сайт рецептів, міський портал або певна вікі. Набір підготовлених заздалегідь HTML-файлів, які лежать на віддаленому сервері та віддаються браузеру за певним запитом.

Сайти містять різну статику, яка як і HTML-файл не генерується під час завантаження веб-сторінки. Найчастіше це картинки, CSS-файли, JS-скрипти, але можуть бути і будь-які інші файли: mp3, mov, csv, pdf.

Блоги, візитки з формою для зв’язку, лендінги с купою ефектів я теж відношу до простих сайтів. Хоча на відміну від зовсім статичних сайтів, вони вже включають в себе якусь бізнес-логіку.

Веб-система – сайт, з частково розробленими сторінками, кінцеве наповнення яких, визначається по запиту користувача. Перша сторінка, на яку потрапляє користувач, називається статичною, її наповнення завжди статичне, незмінне. Більшість решти сторінок, на які користувач потрапляє, натиснувши на певну кнопку, або перейшовши на певну вкладку – динамічні, тому що формуються під певний запит користувача. Почтові клієнти, соціальні мережі, інтернет-магазини – це все веб-системи. Але щоб отримати доступ до веб-

системи, спочатку потрібно мати доступ до її IP-адреси. Ця адреса дозволяє користувачам розрізняти буквальні мільярди підключених пристроїв, що складають Інтернет, а також дозволяє цим пристроям точно обмінюватися цифровою інформацією між собою.

IP-адреса, скорочена від Internet Protocol address, - це ідентифікаційний номер мережевого обладнання, підключеного до мережі. Наявність IP-адреси дозволяє пристрою обмінюватися даними з іншими пристроями через мережу, що базується на IP, наприклад Інтернет. Зазвичай IP-адреса виглядає так: 151.101.65.121, але можна зіткнутися і з таким виглядом: 2001: 4860: 4860 :: 8844.

IP-адреса забезпечує ідентифікацію мережевого пристрою в Інтернеті. Подібно до домашньої або ділової адреси, яка забезпечує конкретне фізичне місце розташування ідентифікованою адресою, пристрої в мережі відрізняються між собою за допомогою IP-адрес.

Якщо ви відправляєте подарунок другові в іншу країну, ви повинні знати точний пункт призначення. Цей самий загальний процес використовується для надсилання даних через Інтернет. Однак, замість фізичної поштової адреси, комп'ютер використовує DNS-сервери для пошуку імені хосту, щоб знайти його IP-адресу.

Рисунок 1.1.1 - Загальний процес надсилання (отримування) даних на Сайті

Наприклад, коли ви вводите URL-адресу веб-сайту, наприклад

www.medsearch.com, у браузері, ваш запит на завантаження цієї сторінки надсилається DNS-серверам, які шукають ім’я хосту medsearch.com, щоб знайти відповідну IP-адресу. Без IP-адреси комп’ютер не здогадується, що саме ви шукаєте.

1.2 Принципи роботи веб-системи

Обробка статичних сторінок. Все починається з простого запиту користувача. Після того, як користувач зробив запит в браузері, веб-сервер обробляє його і у відповідь відсилає заздалегідь створену веб-сторінку. Це сторінка з заданим HTML-кодом. Найпопулярніші веб-сервери – це Microsoft Internet Information Server (IIS) та Apache HTTP Server.

Обробка динамічних сторінок. На відміну від статичних сторінок, динамічні не потрапляють безпосередньо з веб-сервера браузеру. Вони спочатку направляються на сервер додатків. Там зчитується код, підбираються дані і з них формується сторінка. Потім сторінка направляється сервера, а звідти - браузеру.Робота з базою даних. Часто для веб-додатків створюються бази даних (наприклад, особиста інформація про користувачів (ПІБ, вік, стать), їх досягнення і показники). При формуванні динамічних сторінок, сервер додатків відправляє запит до бази. Для цього використовується драйвер бази даних. Він встановлює з'єднання сервера з базою. Для написання запитувикористовується мова SQL. У відповідь база надає необхідні дані, з яких формується динамічна сторінка. Для створення баз даних використовуються сервери Microsoft Access, Microsoft SQL, MySQL, Oracle 9i, MongoDB. Бажано,щоб веб-сервер і база даних знаходилися на одному комп'ютері. Це значно скоротить час обробки і відповідно очікування користувачем завантаження сторінки. Якщо це неможливо, варто встановити високошвидкісне підключення.

1.3 Процес розробки веб-системи

Веб-розробку можна розділити на два окремі напрямки, а саме на стороні клієнта (frontend) та сторону сервера (backend). Клієнтська розробка відповідає за кожен елемент, до якого користувачі можуть безпосередньо отримати доступ

на сторінці, тоді як розробка на стороні сервера підтримує фонові системи, що складають цифрову інфраструктуру веб-сторінки. Клієнтські системи дозволяють користувачам повідомляти веб-сторінці, що вони хочуть зробити, а серверні системи відповідають за виконання цих запитів. Якщо ви знаєте серверну розробку та інтерфейсну розробку, вас називатимуть розробником повного стеку.

Також невід'ємною частиною веб-розробки є дизайн майбутнього застосунку, адже розробник потрібно бачити майбутній продукт, щоб коректно розприділити час, поділити веб-систему на блоки і т.д.

Веб-дизайн - це швидко зростаюча сфера в технічній галузі завдяки цифровій трансформації та збільшенню кількості веб-додатків. Веб-дизайнери шукають зручну платформу для своїх творінь, однією з таких є Figma. Figma - це унікальний інструмент веб-дизайну, який додає цінності веб-дизайну з точки зору доступності, співпраці та унікальних функцій, завдяки своїй величезній потужності для створення макетів та прототипів програмних продуктів та різних додатків; це веб-інструмент для проектування інтерфейсу, який складається з потужних та захоплюючих функцій для веб-дизайну. Цей інструмент забезпечує спільну та економічну платформу для створення веб- дизайнів.

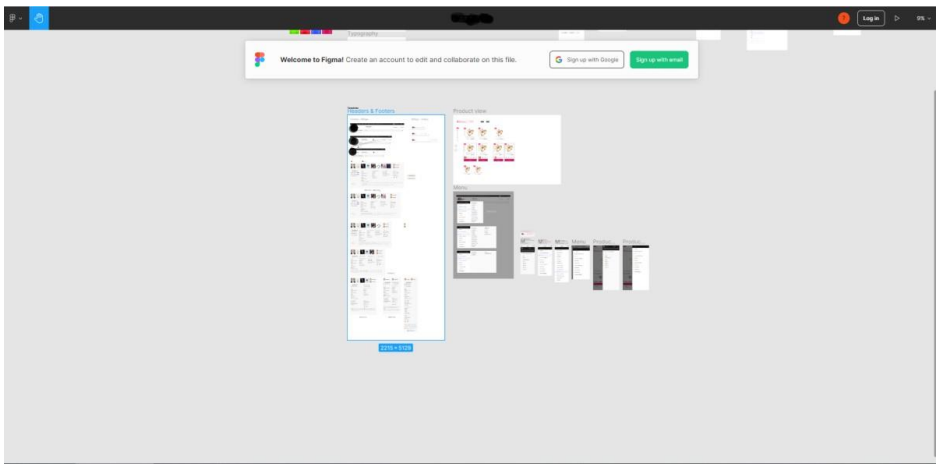


Рисунок 1.3.1 - Інтерфейс Figma

Він складається з потрібних інструментів дизайну, які надають цікавий досвід веб-розробникам. Він може використовуватися для виконання різних завдань, таких як Векторні ілюстрації, дизайн інтерфейсу користувача, дизайн

додатків та створення прототипів.

Файли Figma зберігаються в хмарі. Це означає, що користувачі можуть отримати доступ до цих файлів з будь-якого місця. Якщо користувачі внесуть зміни до цих файлів, вони будуть збережені автоматично. Також є можливість скасувати зміни. Управління проектами полегшено завдяки цьому інструменту, оскільки всі дизайнерські проекти можна зберігати в одному місці.

Figma не тільки зручний інструмент для веб-дизайнера, а й для веб-розробника, адже там одразу можна скопіювати потрібні стилі, побачити відстань між потрібними блоками у пікселях та багато чого іншого.

1.4 Розробка веб-системи на стороні клієнту

Фронтальний інтерфейс (Frontend) побудований з використанням комбінації таких технологій, як мова розмітки гіпертексту (HTML), JavaScript та каскадні таблиці стилів (CSS). Фронтенд розробники створюють елементи інтерфейсу користувача на веб-сторінці або в додатку, включаючи кнопки, меню, сторінки, посилання, графіку тощо. Завдання проекту веб-сайту полягаєу забезпеченні того, щоб користувачі, відкриваючи сайт, бачили інформацію у форматі, який легко читати та відповідати. Це ускладнюється тим, що користувачі зараз використовують велику кількість пристроїв із різними розмірами екрана та роздільною здатністю, що змушує дизайнера враховуватиці аспекти при розробці сайту. Їм потрібно переконатися, що їх сайт правильно відображається в різних браузерах (крос-браузер), різних операційних системах (крос-платформа) та різних пристроях (крос-пристрій), що вимагає ретельного планування з боку розробника. Тобто вміти адаптивно верстати.

Адаптивні сайти змінюють зовнішній вигляд залежно від середовища браузера, в якому їх переглядають (найпоширеніша річ: ширина браузера).

Візьмемо до прикладу популярну платформу «Розетка» [7] – електронний магазин. У браузері (з шириною екрану 1960 піксерів) веб-система має такий вигляд:

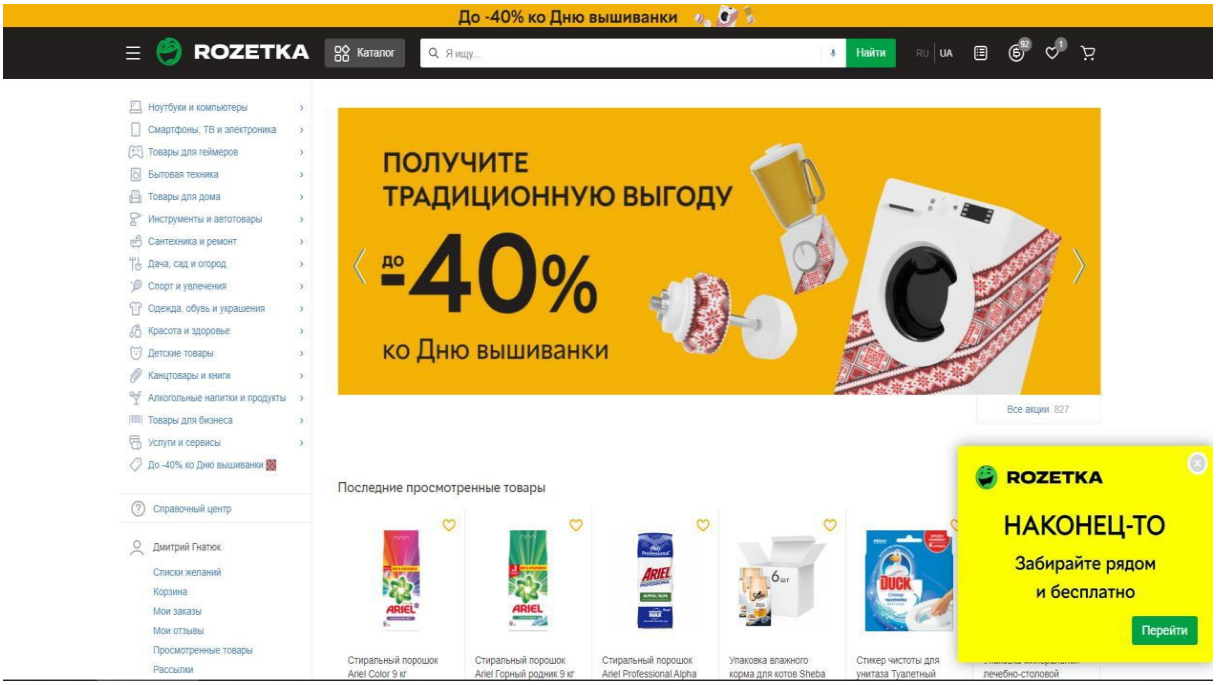


Рисунок 1.4.1 - Інтерфейс «Розетка» з великою шириною екрану

На мобільному екрані (з шириною екрану 360 пікселів) веб-система дещозмінить свій вигляд:

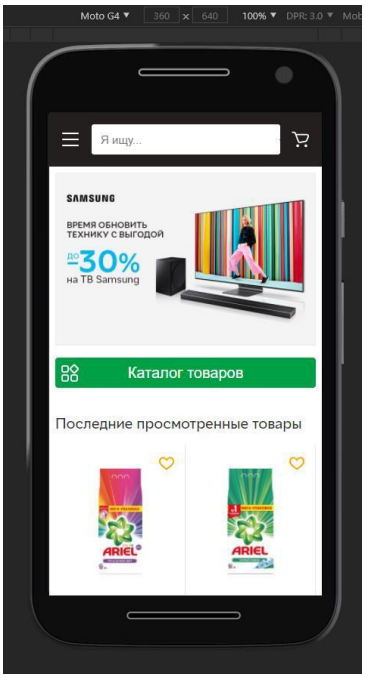


Рисунок 1.4.2 Інтерфейс «Розетка» з великою шириною екрану

Як ми бачимо змінився вигляд блоків для зручнішого користування веб-системою саме на мобільних пристроях, це і є адаптив.

Хоч існує безліч різних типів технологій та стеків, більшість інтерфейсних веб-розробників використовують HTML, CSS та JavaScript.

HTML - це мова розмітки гіпертексту. Він дозволяє користувачеві створювати і структурувати розділи, параграфи, заголовки, посилання і блоки для веб-сторінок і додатків. HTML не є мовою програмування, тобто він не має можливості динамічно створювати функції. Замість цього він дозволяє організовувати і форматувати документи, аналогічно Microsoft Word. При роботі з HTML ми використовуємо прості структури коду (теги і атрибути), щоб розмітити сторінку веб-сайту. HTML-документи – це файли, які закінчуються з розширення .html або .htm. Ці файли можна переглядати за допомогою будь-якої якого веб-браузера (наприклад, Google Chrome, Safari або Mozilla Firefox). Браузер читає HTML-файл і відображає його вміст, щоб користувачі інтернету могли його переглядати. Зазвичай середній сайт включає кілька різних HTML-сторінок. Наприклад: домашня сторінка,контентна сторінка, сторінка контактів мають окремі HTML-документи. Кожна HTML-сторінка складається з набору тегів, які можна назвати будівельними блоками веб-сторінок. Вони створюють ієрархію, яка структурує контент за розділами, параграфами, заголовкам і іншим блокам контенту. HTML - це мова розмітки гіпертексту. Вона дозволяє користувачеві створювати і структурувати розділи, параграфи, заголовки, посилання і блокидля веб-сторінок і додатків.

HTML не є мовою програмування, тобто вона не має можливості створювати динамічні функції. Замість цього вона дозволяє організовувати і форматувати документи, аналогічно Microsoft Word.

При роботі з HTML ми використовуємо прості структури коду (теги і атрибути), щоб розмітити сторінку веб-сайту. Наприклад, ми можемо створити абзац, помістивши додається текст у вихідний тег <p> і закриває </ p>.

В цілому, HTML - це мова розмітки, яка дуже проста в освоєнні навіть для початківців в створенні сайтів.

HTML-документи - це файли, які закінчуються розширенням .html або .htm. Ви можете переглядати його за допомогою будь-якого веб-браузера (наприклад, Google Chrome, Safari або Mozilla Firefox). Браузер читає HTML-файл і відображає його вміст, щоб користувачі інтернету могли його переглядати.

Зазвичай середній веб-сайт включає кілька різних HTML-сторінок (англ). Наприклад: домашні сторінки, звичайні сторінки, сторінки контактів матимуть окремі HTML-документи.

Кожна HTML-сторінка складається з набору тегів (також званих елементами), які ви можете назвати будівельними блоками веб-сторінок. Вони створюють ієрархію, яка структурує контент за розділами, параграфами, заголовкам і іншим блокам контенту.

Більшість елементів HTML мають відкриття і закриття, в яких використовується синтаксис <tag> </ tag>.

А CSS використовують для того, щоб додати цікавості та красоти простим тегам HTML.

Отож, CSS (Cascading Style Sheets) - це код, який ви використовуєте для стилізації вашої веб-сторінки. Основи CSS допоможуть вам зрозуміти, що вам потрібно для початку роботи. CSS вважається однією з трьох основних технологій, що використовуються в Інтернеті. Ми відповімо на такі питання як: Як зробити мій текст чорним або червоним? Як зробити так, щоб контент з'являвся в певному місці на екрані? Як прикрасити мою веб-сторінку за допомогою фонових зображень і кольорів?

Як і HTML, CSS насправді не є мовою програмування. Це не мова розмітки - це мова таблиці стилів. Це означає, що він дозволяє застосовувати стилі вибірково до елементів в документах HTML. Наприклад, щоб вибрати всі елементи абзацу на HTML сторінці і змінити текст всередині них з чорногона червоний, ви повинні написати цей CSS:

Інтернет-браузери розуміють і слухають лише CSS, тому його не можна просто замінити абсолютно новою мовою. Однак він має певні обмеження, які, можливо, не відчуваються суттєвими у невеликих проектах, але все ж беруть своє, коли ви маєте справу з величезними таблицями стилів.

```
p {
  color: red;
}
```

Рисунок 1.4.3 - Приклад CSS коду

Для вирішення цієї проблеми розробники використовують препроцесори CSS. По суті, препроцесори - це програми, що мають свій унікальний синтаксис. Після того,як ви напишете свій код, вони компілюють його до чистого CSS.

Причиною того, що ми використовуємо препроцесори CSS, є додавання додаткових функціональних можливостей, яких CSS інакше не мав би. Наприклад, у вас можуть бути селектори вкладання або успадкування, а також міксини (багаторазові пакети оголошень), змінні. Додаткові зручні функції дозволяють ефективніше виконувати щоденну роботу та забезпечують додаткову масштабованість.

На вибір є багато препроцесорів CSS, але є два дуже популярні препроцесори: Sass та Less.

```
.btn {
  &-primary {}
  &-secondary {}
}
```

Рисунок 1.4.4 - Фрагмент синтаксису SCSS

На даний момент Sass має два окремі синтаксиси: Sass і SCSS. Хоча у перших версіях препроцесора існував лише перший, команда побоювалася, що він може дещо відрізнятися від звичайного CSS. Тому вони представили новий синтаксис під назвою SCSS (Sassy CSS) у третій версії. Файли також можуть мати розширення .sass або .scss.

Основна відмінність між Sass і SCSS полягає в тому, що останній використовує крапки з комою та дужки так само, як це робить CSS. Sass, навпаки, цього не робить - плюс, він використовує знак рівності замість двокрапки для присвоєння.

А якщо порівнювати Less та Sass, ми побачимо, що вони дуже схожі за своїми основними функціональними можливостями. Обидва вони дозволяють вкладати, імпортувати та використовувати змінні. Однак у програмі Less ви також можете піднімати змінні та витягувати певні компоненти з кольору, якого ви не змогли зробити за допомогою Sass - а саме відтінок, насиченість, яскравість та яскравість. З іншого боку, Sass дозволяє використовувати оператори if та інтерполювати їх у селекторах та іменах властивостей. Його синтаксис також містить трикомпонентні оператори та колектори вкладеності. Ще одним невеликим недоліком програми Less є те, що для оголошення змінної використовується символ @ (замість цього Sass використовує знак $).Однак у CSS @ може також використовуватися для ключових кадрів та медіа-запитів. Це може викликати невелику плутанину при читанні коду.

А, JavaScript (JS) - це динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв вебсторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд вебсторінки.

```
let a = "Hello, World!"
console.log(a)
```

Рисунок 1.4.5 - Приклад створення змінної та виведення її у консоль за допомогою JavaScript

JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми

програмування (імперативну та частково функціональну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

Мова JavaScript використовується для:

- написання сценаріїв веб-сторінок для надання їм інтерактивності
- створення односторінкових та прогресивних веб-застосунків(React, AngularJS, Vue.js)
- програмування на боці сервера (Node.js (Express.js))
- стаціонарних застосунків (Electron, NW.js)
- мобільних застосунків (React Native, Cordova)
- сценаріїв в прикладних програмах (наприклад, в програмах зі складу Adobe Creative Suite чи Apache JMeter)
- всередині PDF-документів тощо.

Незважаючи на схожість назв, мови Java та JavaScript є двома різними мовами, що мають відмінну семантику, хоча й мають схожі риси в стандартних бібліотеках та правилах іменування. Синтаксис обох мов отриманий «у спадок» від мови C, але семантика та дизайн JavaScript є результатом впливу мов Self та Scheme.

Коли JavaScript створювався, у нього було інше ім'я - «LiveScript».Однак, мова Java був дуже популярний в той час, і було вирішено, що позиціювання JavaScript як «молодшого брата» Java буде корисно.

Згодом JavaScript став повністю незалежним мовою зі своєю власною специфікацією, яка називається ECMAScript, і зараз не має ніякого відношення до Java.

Сьогодні JavaScript може виконуватися не тільки в браузері, а й на сервері або на будь-якому іншому пристрої, який має спеціальну програму, що називається «движком» JavaScript. У браузера є власний движок, який іноді називають «віртуальна машинаJavaScript».

Різні движки мають різні «кодові імена». наприклад:

- V8 - в Chrome і Opera.
- SpiderMonkey - в Firefox.

Ще є «Trident» і «Chakra» для різних версій IE, «ChakraCore» для Microsoft Edge, «Nitro» і «SquirrelFish» для Safari і т.д.

Ці назви корисно знати, так як вони часто використовуються в статтях для розробників. Ми теж будемо їх використовувати. Наприклад, якщо

«функціональність X підтримується V8», тоді «X», швидше за все, працює в Chrome і Opera.

Сучасний JavaScript - це «безпечний» мову програмування. Він не надає низькорівневий доступ до пам'яті або процесору, тому що спочатку був створений для браузерів, які не потребують цього.

Можливості JavaScript сильно залежать від оточення, в якому він працює. Наприклад, Node.JS підтримує функції читання / запису довільних файлів, виконання мережевих запитів і т.д.

У браузері для JavaScript є все, що пов'язано з маніпулюванням веб-сторінками, взаємодією з користувачем і веб-сервером.

Наприклад, в браузері JavaScript може:

- Додавати новий HTML-код на сторінку, змінювати існуючий вміст, модифікувати стилі.
- Реагувати на дії користувача, клацання миші, перемістити вказівник, натискання клавіш.
- Відправляти мережеві запити на віддалені сервера, завантажувати і завантажувати файли (технології AJAX і COMET).
- Отримувати і встановлювати куки, задавати питання відвідувачеві, показувати повідомлення.
- Запам'ятовувати дані на стороні клієнта («local storage»).

JavaScript - це мова сценаріїв, яка вставляється безпосередньо в HTML сторінки. Це єдина мова програмування такого роду, яку можуть зрозуміти веб-браузери. Браузери можуть читати Javascript, інтерпретувати його, а потім запускати програму, створюючи потужний досвід на стороні клієнта.

Вона досягла цього статусу, оскільки є відкритою, стандартизованою та,

найголовніше, погоджуєтесь чи ні, дуже гарною мовою. Він добре підходить для Інтернету завдяки своїй динамічній природі та тісній інтеграціїз DOM.

Отже, підсумовуючи, можна сказати про таки преваги використання JavaScript:

- Виконання логіки на стороні клієнта забезпечує швидший досвід роботи користувачів. Коли код працює безпосередньо в браузері, необхідність серверних викликів абстрагується, отже, скорочується час завантаження. Навіть при наявності сервера, той факт, що JS є асинхронним, означає, що вінздатний взаємодіяти з сервером у фоновому режимі, не перериваючи взаємодію користувача, що відбувається в інтерфейсі.
- З самого початку JavaScript привносить в Інтернет інтерактивність користувальницького інтерфейсу. Зараз він робить те саме для додатків будь-якого виду, допомагаючи розробляти найбільш захоплюючий UX. Сьогодні такі фреймворки, як Vue.js, виводять переходи та анімацію на новий рівень.
- JavaScript стоїть за будь-яким хорошим адаптивним веб-дизайном. Дедалі більше розробникам потрібно адаптувати свій дизайн у декількох браузерах та пристроях. Поєднуючи HTML5, CSS3 та JavaScript, вони можуть це робити в одній кодовій базі.
- Для розробників JS легко навчатись і швидко починати активний розвиток. Його синтаксис простий і гнучкий для новачків. Це також спрощує розробку складних додатків, дозволяючи розробникам спростити склад програми. Багато фреймворків та пакетів також певною мірою полегшують життя розробникам.

- Якщо у вас цього ще немає, JavaScript користується шаленою популярністю. Якщо популярність не завжди відповідає якості життя в цілому, це принаймні означає одне важливе: ви знайдете рішення будь-якої проблемив громаді. У веб-розробці це не є незначною деталлю. Якщо ви хтось, хто повинен наймати розробників, це теж великий плюс, оскільки пул кандидатів величезний.

JavaScript також сумісний з іншими мовами. Це надзвичайно важливо, оскільки веб-сервери працюють на різних мовах, будь то PHP, Python, Ruby,

Java або .NET. Оскільки JavaScript, що працює в браузері, на 100% відокремлений від того, як генеруються веб-сторінки HTML, користувачі завжди матимуть такий самий багатий досвід, як працює JS, незалежно від мови сервера.

Тривалий час веб-сайти в основному працювали на основі PHP CMS, таких як WordPress. Серверний код обробляв основну частину логіки. Однак ситуація змінюється - ви могли чути, що "статичні" сайти повертаються. Однак вони нічим не схожі на статично створені веб-сайти 90-х.

Сучасні браузери тепер мають можливість зробити ці інтерактивні та повністю динамічні. Особливістю, яку вони поділяють зі своїми предками, є абстракція бекенда. Логіка обробляється на стороні клієнта безпосередньо у браузері завдяки JavaScript.

Деякі найвидатніші веб-програми сьогодні створені за допомогою JS. Подумайте про Facebook, Gmail, Twitter та багато іншого. Якщо ми використовуємо Facebook як приклад, JavaScript дозволяє оновлювати статус та забезпечувати більшу інтерактивність користувачів. Без нього це не мало б привабливості.

Ці технічні гіганти насправді створили власні фреймворки JavaScript, і тепер вони дозволяють тисячам розробників створювати власні веб-програми. Angular за підтримки Google і React за підтримки Facebook. Я також маю тут згадати Vue, який, навіть якщо не підтримується технічною силою, завершує тріаду важливих структур JS.

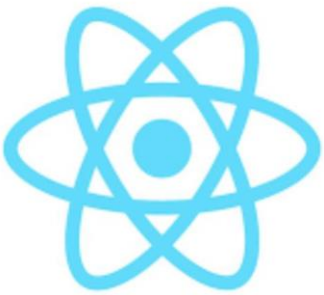


Рисунок 1.4.6 - Логотип фреймворку React

Рисунок 1.4.7 - Логотип фреймворку Angular

Рисунок 1.4.8 - Логотип фреймворку Vue

Окрім зменшення кількості часу та зусиль, необхідних для розробки сайтів та програм на базі JS, ці фреймворки допомогли сформувати новий веб-досвід. Візьмемо, наприклад, односторінкові програми (SPA). Single page application (односторінковий додаток). Більш цікавий варіант, коли використовується і бекенд, і фронтенд. За допомогою їх взаємодії можна створити додаток, який буде працювати без перезавантажень сторінки в браузері. Або в спрощеному варіанті, коли перехід між розділами викликає перезавантаження, але будь-які дії в розділі обходяться без них. На практиці це означає, що користувач бачить в браузері весь основний контент, а під час переміщення або переходах на інші сторінки, замість повного перезавантаження потрібні елементи просто завантажуються. SPA - це веб-

сайт, який взаємодіє з користувачами, динамічно переписуючи сторінку в браузері, а не завантажуючи цілі нові сторінки з сервера, змушуючи їх поводитися більше як настільні програми.

Отож, детальніше про фреймворки: вони допомагають нам робити програми більш структурованими. Основний принцип роботи полягає в тому, що нам не потрібно постійно винаходити колесо. Отже, використовуючи фреймворк для розробки програми, ми можемо заощадити багато часу, а також створити стійкий та якісний код. Кожен фреймворк має власну структуру для розробки програми.

Frontend фреймворки - це пакети із заздалегідь написаним стандартизованим кодом у файлах та папках. Вони надають вам основу для створення інтерактивних програм, що забезпечують винятковий досвід користувачів. Ось декілька компонентів, які постачаються з більшістю інтерфейсних фреймворків:

- Сітка для спрощення елементів дизайну веб-сайту
- Попередньо визначений стиль та розмір шрифту для заголовків та абзаців
- Готові веб-елементи, такі як заклики до дії, кнопки та навігаційні панелі

Окрім цього, фронтед-фреймворки мають безліч інших функціональних можливостей, які можна використовувати для проекту. Одним із найпопулярніших фреймворків є Angular.

Angular – це платформа для розробки веб-сайтів та створенняефективних та складних односторінкових додатків. Він розробляється Google. Angular підтримує Typescript та Html для створення веб-систем.

Односторінковий додаток означає, що ви не будете спрямовані на іншу сторінку, натискаючи будь-який параметр, ця сторінка відкриється уже на тій самій сторінці, просто дещо зміниться функціонал, інформація, розмітка не змінюється. Це означає, що в односторінковій програмі ви не рухаєтесь на різних сторінках, ви просто переходите від однієї частини своєї програми до іншої.

Angular пропонує двосторонню прив'язку даних та синхронізацію в режимі реального часу між моделлю та видом. Це означає, що будь-які зміни у поданні відображатимуться на моделі в режимі реального часу і навпаки.

Angular може бути ідеальним вибором, коли ви прагнете створити Інтернет або мобільний додаток, особливо багатосторінковий або PWA. Сьогодні він забезпечує мільйони сайтів та додатків, включаючи декілька відомих імен у цій галузі.

Хоч Angular - не найпростіший фреймворк, до якого можна звикнути - документація полегшує вступ до початківців. Це, безумовно, робить Angular однією з найкращих платформ інтерфейсу користувача зараз.

Будучи одним з найпопулярніших фреймворків, Angular використовується декількома найпопулярнішими веб-сайтами та додатками. Ось декілька найкращих веб-програм, що працюють від Angular:

- PayPal
- Upwork
- Netflix

Отож, плюси використання Angular:

- Вбудована функціональність для оновлення змін, внесених у модель, до подання, а також навпаки
- Зменшує кількість коду, оскільки більшість таких відомих функцій, як двостороннє прив'язка даних, надаються за замовчуванням
- Від’єднує компоненти від залежностей, визначаючи для них зовнішні елементи
- Компоненти багаторазові та прості в управлінні за допомогою введення залежності
- Величезне співтовариство для навчання та підтримки

Мінуси:

- Оскільки Angular є повним динамічним рішенням, існує кілька способів виконати завдання, тому крива навчання є крутішою. Однак велика спільнота

Angular полегшує вивчення нових понять про поняття татехнологіїБуває, що динамічні програми іноді не працюють добре через їх складну структуру та розмір. Однак оптимізація коду та найкращі практикиAngular

Наступним найпростішим фреймворком є React. Розроблений і підтримуваний Facebook, React , безумовно, є одним з найпростіших фреймворків інтерфейсу JavaScript для роботи. Він використовується як основа в односторінкових та мобільних додатках. Структура постійно набирала популярність з моменту запуску в травні 2013 року.

Як свідчать дані опитування, React - найпопулярніший фреймворк, у якому понад 3 мільйони користувачів та гігантська екосистема розробників. Віртуальна об'єктна модель документа (DOM) та одностороння прив'язка даних можуть бути двома основними причинами, що відповідають за її швидке прийняття та широку популярність.

Більше того, інструменти розробника React - це розширення, що використовується для бібліотеки JavaScript з відкритим кодом React, що дозволяє перевіряти ієрархію компонентів React у Chrome DevTools.

Зі постійно зростаючою популярністю React кількість програм, що використовують його, також зросла. Від індивідуальних підприємців до мільярдних підприємств, React, безумовно, є однією з найбільш бажаних платформ для розробки мобільних програм. За допомогою React написаний один із найпопулярніших додатків молоді – Instagram.

ReactJS - це бібліотека JavaScript, яка використовується для створення багаторазових компонентів інтерфейсу користувача. Згідно з офіційною документацією React, наступне визначення -

React.js - це бібліотека JavaScript з відкритим кодом, яка використовується для створення інтерфейсів користувача спеціально для односторінкових програм. Він використовується для обробки шару перегляду для веб- і мобільних додатків. React також дозволяє нам створювати багаторазові компоненти інтерфейсу користувача. Реакцію вперше створив Джордан Уолк, інженер програмного забезпечення, який працює у Facebook. Реакт вперше з'явився на стрічці новин Facebook у 2011 році та на Instagram.com

у 2012 році.

React дозволяє розробникам створювати великі веб-додатки, які можуть змінювати дані, не перезавантажуючи сторінку. Основна мета React - бути швидким, масштабованим та простим. Він працює лише на користувальницьких інтерфейсах у додатку. Це відповідає перегляду в шаблоні MVC. Він може використовуватися з комбінацією інших бібліотек або фреймворків JavaScript, таких як Angular JS в MVC.

У React є рідні бібліотеки, про які було оголошено Facebook у 2015 році, що забезпечує архітектуру реагування на рідні програми, такі як IOS, Androidта UPD.

React-native - це структура мобільних додатків, що використовує лише Javascript. Він використовує той самий дизайн, що і React, дозволяючи вам використовувати/включати багату мобільну бібліотеку інтерфейсу / декларативні компоненти. Він використовує ті самі основні будівельні блоки інтерфейсу, що і звичайні програми для iOS та Android. Найкраща частина використання нативної реакції - це дозволити / прийняти компоненти, написані на Objective-C, Java або Swift.

React не є такою структурою, як Angular, і не надає великого спектру бібліотек, як це робить Angular. Тому React є підходящим середовищем для сучасних додатків веб-розробки, які плануються розширюватися в майбутньому в декількох операційних системах.

Отож, плюси використання React:

- багаторазове використання компонентів полегшує співпрацю та повторне їх використання в інших частинах програми
- постійна та безперебійна робота з використанням віртуального DOM
- найкраща альтернатива написанню компонентів у хуках React, це дозволяє писати компоненти без занять і дозволяє легше вивчити React
- інструменти розробника React є вдосконаленими та надзвичайно корисними

А мінусами є:

- Через багаторазові та постійні оновлення у фреймворці важко

скластиналежну документацію, і це впливає на криву навчання для початківців

- Розробникам важко зрозуміти складність JSX, починаючи з фреймворку
- Він надає лише інтерфейсні рішення

Наступним, не менш популярним фреймворком є Vue. На сьогоднішній день, він є одним з найпопулярніших фронтенд фреймворків. Vue простий і зрозумілий фреймворк. Він добре усуває складності, з якими стикаються розробники Angular. Він менший за розміром і пропонує дві основні переваги

- візуальний DOM та компонентний. Це також двосторонній зв'язок.

Vue універсальний, і він допомагає виконувати кілька завдань. Починаючи від створення веб-додатків та мобільних додатків, до прогресивних веб-систем, він з легкістю може обробляти як прості, так і динамічні процеси.

Хоча він створений для оптимізації продуктивності додатків та вирішення складних ситуацій, він не користується широкою популярністю серед гігантів ринку. Однак, користувачами цього фреймворку є Alibaba, 9gag, Reuters, Xiaomi. Vue продовжує зростати з точки зору усиновлення, незважаючи на меншу кількість учасників з Кремнієвої долини.

Vue має найменші бібліотеки та фреймворки. Тому це ідеальний вибір для розробки легких додатків для веб-розробки та односторінкових додатків. Якщо ви хочете вибрати просту бібліотеку та менший за розміром, то Vue - ідеальний та найкращий вибір для ваших програм веб-розробки.

Як ми знаємо, Vue - це наймолодший фреймворк серед інших із неймовірними можливостями для подолання перешкод, з якими розробники стикаються з Angular та React. Він складається з усіх хороших особливостей React та Angular. Тому, використовуючи віртуальний DOM, Vue забезпечує високу продуктивність без помилок та виділення пам'яті.

Отож, плюсами використання Vue є:

- велика та детальна документація
- вростий синтаксис - програмісти з базою JavaScript можуть легко розпочати роботу з Vuejs
- гнучкість при розробці структури програми

- підтримка машинопису

Але на жаль, і тут є свої мінуси:

- Відсутність стійкості в компонентах
- Порівняно невелика громада
- Мовний бар’єр із плагінами та компонентами (більшість плагінів написані китайською мовою)

1.5 Розробка веб-системи на стороні серверу

З самого початку були невдалі спроби змусити JavaScript працювати на стороні сервера. Багато хто думав, що це просто ніколи не стане стабільною бекенд-мовою до появи Node.js.

Сьогодні це середовище виконання JS є популярним інструментом для живлення веб-серверів. Це означає, що розробники JS можуть використовувати Node.js для написання як коду на стороні клієнта, так і на стороні сервера в JavaScript, не покладаючись на зовнішні веб-сервери. Node JS є основною базовою мовою для eBay та AliExpress.

Простіше кажучи, Node.js - це JavaScript, який працює на сервері.Node.js - це платформа, заснована на середовищі виконання JavaScript Chrome.Node.js - це середовище JavaScript, що керується подіями сервера вводу- виводу. На основі механізму V8 Google, механізм V8 виконує Javascript дужешвидко і має дуже хорошу продуктивність.

Node.js виник, коли оригінальні розробники JavaScript розширили йогоз того, що можна було запускати лише у браузері, на те, що можна запускати на своїй машині як окремий додаток. Тепер можна зробити набагато більше за допомогою JavaScript, ніж просто зробити веб-сайти інтерактивними. JavaScript тепер має можливість робити те, що можуть робити інші мови сценаріїв, такі як Python. Як JavaScript вашого браузера, так і Node.js працюють на механізмі виконання V8 JavaScript. Цей механізм бере ваш код JavaScript і перетворює його в швидший машинний код. Машинний код - це низькорівневий код, який комп’ютер може запустити без необхідності його попередньої інтерпретації.

Node.js - це платформа для зручного створення швидких та масштабованих мережевих додатків. Node.js використовує подію, що неблокує модель вводу / виводу, яка робить її легкою та ефективною,

ідеальною для використання в реальному часі додатків, що працюють на розподілених пристроях.

Node.js - це відкрите міжплатформенне середовище виконання програм для розробки серверних та мережевих додатків. Програми Node.js написані на JavaScript, і їх можна запускати в режимі виконання Node.js на OS X,

Microsoft Windows та Linux.

Node.js також пропонує багату бібліотеку різноманітних модулів

JavaScript, що значно спрощує розробку веб-додатків, що використовуютьNode.js.

Вивчаючи Node.js неможливо не почути про вбудовану підтримку управління пакетами за допомогою npm. Популярна бібліотека пакетів - npm - це найцінніше володіння спільноти Node.js. Він містить мільйони завантажуваних бібліотек відповідно до конкретних вимог. Ці масивні бібліотеки абсолютно безкоштовно користуються своїм реєстром. З кожним днем ці бібліотеки швидко збільшуються, роблячи спільноту Node.js міцнішою.

Чому Netflix обрав NodeJs?

Маючи понад 130 мільйонів підписників та мільярд годин щотижневої передачі даних, Netflix шукав масштабоване рішення, яке могло б дозволити йому обробляти кілька запитів одночасно.

І це вийшло у Node.js! Команда вибрала технологію, щоб вони могли використовувати одну і ту ж мову у бекенді та інтерфейсі. Вони також побудували SPA, що значно скоротило час запуску. Це призвело до загального скорочення часу запуску - з 40 хв до менш ніж 60 секунд

Будь-яка серверна частина не обходиться без бази даних. Серед

найбільш відомих є: Oracle 12c, MySQL, Microsoft SQL Server, PostgreSQL,MongoDB.

- Oracle 12c

Не дивно, що Oracle постійно входить до списку популярних баз даних.

Перша версія цього інструменту управління базами даних була створена наприкінці 70-х років, і існує цілий ряд видань цього інструменту, які відповідають потребам вашої організації.

Найновіша версія Oracle, 12c, розроблена для хмари і може розміщуватися на одному сервері або декількох серверах, і вона дозволяє

керувати базами даних, що містять мільярди записів. Деякі функції останньоїверсії Oracle включають структуру сітки та використання як фізичної, так і логічної структур. Ідеально підходить для великих організацій, які працюютьз величезними базами даних і потребують різноманітних функцій.

Плюси:

- найновіші інновації та функції, що надходять від їх продуктів, оскількиOracle прагне встановити планку для інших інструментів управління

базами даних.

- інструменти керування базами даних Oracle також неймовірно надійні, іможна знайти такий, який може робити майже все, про що можна подумати.

Мінуси:

- вартість Oracle може бути надмірною, особливо для невеликих організацій.

- після встановлення система може вимагати значних ресурсів, тому для впровадження Oracle може знадобитися оновлення обладнання.

- MySQL

MySQL - одна з найпопулярніших баз даних для веб-програм. Це безкоштовна програма, але вона часто поповнюється функціями та

покращеннями безпеки. Є також різноманітні платні видання, призначені для комерційного використання. У безкоштовній версії ви більше зосереджуєтесь на швидкості та надійності, замість того, щоб включати широкий спектр

функцій, які можуть бути хорошими чи поганими залежно від того, що ви

намагаєтесь зробити.

Цей механізм баз даних дозволяє вибирати з безлічі механізмів зберігання, які дозволяють змінювати функціональність інструменту та обробляти дані з різних типів таблиць. Він також має простий у використанні інтерфейс, а пакетні команди дозволяють обробляти величезні обсяги даних. Система також неймовірно надійна і не має тенденції залучати ресурси.

Ідеально підходить для організацій, які потребують надійного інструментууправління базами даних, але мають бюджет.

Плюси:

- він доступний безкоштовно.
- він пропонує багато функціональних можливостей навіть для безкоштовного механізму баз даних.
- існує безліч користувальницьких інтерфейсів, які можна реалізувати.
- його можна змусити працювати з іншими базами даних, включаючи DB2 та Oracle.

Мінуси:

- можна витратити багато часу та сил, щоб змусити MySQL робити те,що інші системи роблять автоматично, наприклад, створювати

додаткові резервні копії.

- немає вбудованої підтримки XML або OLAP.
- підтримка доступна для безкоштовної версії, але доведеться заплатити за неї.

- MongoDB

Ще одна безкоштовна база даних, яка також має комерційну версію, MongoDB розроблена для додатків, які використовують як структуровані, такі неструктуровані дані. Механізм баз даних дуже універсальний, і він працює, підключаючи бази даних до програм через драйвери баз даних MongoDB . Доступний повний вибір драйверів, тому легко знайти драйвер, який буде працювати з мовою програмування, що використовується.

Оскільки MongoDB не був розроблений для обробки реляційнихмоделей даних, хоча це можливо, проблеми з продуктивністю можуть

виникнути, якщо ви спробуєте використовувати його таким чином. Однак механізм баз даних призначений для обробки змінних даних, які не є

реляційними, і він часто може добре працювати там, де інші механізми базданих борються або не справляються.

MongoDB 3.2 є останньою версією та має нові підключаються двигуни зберігання даних. Документи тепер також можна перевірити під час

оновлення та вставки, а функції пошуку тексту вдосконалено. Нова можливість часткового індексу також може дозволити покращити продуктивність за рахунок зменшення розміру індексів.

Плюси:

- він швидкий і простий у використанні.
- механізм підтримує JSON та інші документи NoSQL .
- дані будь-якої структури можна швидко і легко зберігати та отримуватидо них доступ.
- схему можна писати без простоїв.Мінуси:
- SQL не використовується як мова запитів.
- доступні інструменти для перекладу SQL на запити MongoDB, але вонидодають додатковий крок до використання механізму.
- налаштування може бути тривалим процесом.
- налаштування за замовчуванням не захищені.

1.6 Еволюція та актуальність веб-системи у сучасному просторі Інтернету

За останні п’ять років комерційне використання Інтернету та веб-систем різко зросло. За цей час Інтернет перетворився з передусім на те, що є засобом комунікації (електронна пошта, файли, групи новин та чати), і став засобом для розповсюдження інформації на повноцінний ринковий канал для електронної комерції. Веб-сайти, які колись просто відображали інформацію для відвідувачів, стали інтерактивними, високофункціональними системами, які дозволяють багатьом типам бізнесу взаємодіяти з багатьма типами користувачів.

Спільним першим етапом у розвитку інформаційних систем, що базуються на Інтернеті, є інформаційний кіоск, де інформація про тип маркетингу представлена або для підвищення престижу, покращення ідентифікації торгової марки або для стимулювання звичайної торгової діяльності.

Другий етап полягає у відкритті односторонньої системи інформаційних кіосків із використанням, скажімо, веб-форм, щоб браузери могли розміщувати замовлення та робити запити з веб-сторінок, що цікавлять,розглядаючи веб-сайт як електронний каталог замовлення електронної пошти.Третій етап - це переосмислення традиційних меж між замовником та підприємством. Наприклад, клієнт може перейти не лише на перегляд онлайн каталогу та натискання кнопок для вибору продуктів, замість цього переглядти графіки виробництва.

Потім вони можуть вносити індивідуальні зміни в конструкціюпродукції і бачити вплив своїх вимог на графіки виробництва, терміни поставки та внесок у загальні витрати.

Сфера застосування та складність сучасних веб-додатків різниться в широкому діапазоні: від невеликих, нетривалих служб до великих корпоративних додатків, що розповсюджуються через Інтернет та корпоративні інтрамережі та екстрамережі. Веб-програми можна згрупувати за семи категоріями:

• інформаційні, наприклад, Інтернет-газети, каталоги товарів, бюлетені, посібники з обслуговування, Інтернет-оголошення, Інтернет-книги;

• інтерактивні, наприклад, реєстраційні форми, індивідуальна інформація;презентація користувача, онлайн-ігри;

• транзакції, наприклад, електронні покупки, замовлення товарів та послуг, Інтернет-банкінг;

• робочий процес, наприклад, системи онлайн-планування та планування, управління запасами, моніторинг стану;

• робочі середовища для спільної роботи, наприклад, розподілені авторські системи, засоби спільного проектування;

• Інтернет-спільноти, торгові площі, наприклад, чат-групи, системи, що рекомендують, що рекомендують товари чи послуги, Інтернет-торгові площадки, Інтернет-аукціони;

• Веб-портали, наприклад електронні торгові центри, посередники в Інтернеті.

Отож, веб-системи стали важливою складовою бізнесу в сучасномусвіті. Використовуючи веб-додатки, підприємства тепер можуть розвиватися та ставати простішими та набагато швидше досягати своїх цілей. Ці програми можуть допомогти одночасно націлити численних клієнтів та покупців. Організації, компанії швидко використовують цей аспект Інтернету, створюючи веб-додатки за допомогою розробників, щоб задовольнити їхні бізнес-вимоги. Веб-програми важливі з ряду причин.

Підприємства вже не можуть спостерігати зростання частки свого ринку, якщо у них немає належного веб-додатку. Хоч і більші компанії можуть дозволити собі власні команди розробників для таких цілей, менші компанії передають роботу компаніям веб-розробникам, щоб отримати таку ж перевагуза зниженою вартістю. Це допомагає організаціям охопити нових клієнтів та повідомити їх про продукти або послуги, що вона має.

Веб-системи можуть відігравати вирішальну роль у процесі брендування. З їх допомогою легше підтримувати належний канал зв'язку міжпотенційними клієнтами та організацією бізнесу. Поширення знань про товар можна збільшити завдяки створення інформаційного ресурсу для клієнтів в Інтернеті. Збільшуються також можливості продажу послуг або продуктів. Одночасно популярність організації посилюється, а покоління лідів покращується. За допомогою розробників електронної комерції бізнес може отримати абсолютно новий ринок для здійснення продажів, особливо за допомогою веб-сайтів, адже для цього не потрібно нічого встановлювати достатньо мати доступ до Інтернету.

Веб-системи також надають можливості для розширеної підтримки клієнтів. Хороші програми можуть стати першою лінією контакту між потенційними клієнтами та бізнесом. Краса таких додатків полягає в тому, що

до них можна отримати доступ у будь-який час. Навіть розташування вже не є обмеженням з їх допомогою. Звичайно, лише високоякісна компанія з веб-розробки може забезпечити такий механізм у своєму застосуванні.

Нинішній стан бізнесу у світі став настільки сильно конкурентним, тому наявність веб-додатків стає все більш важливою, присвячених справі організації. Ці веб-системи можуть стати важливим інструментом для залучення клієнтів. Веб-застосунок може бути відкритим як на персональному комп’ютері, так і на мобільному пристрої, ось що робить веб-системи такими актуальними.

1.7 Висновки

Проведено огляд та аналіз веб-системи як ресурсу інформації та

послуг. Вони почали набирати темпи розвитку, адже у наш час в основномувся інформація в онлайн доступі, усі послуги та запис на них теж в онлайн режимі.

Було виявлено, що є безліч різних програмних забезпечень для створення веб-системи. Проаналізовано усі популярні фреймворки для створення клієнтської частини та обрала оптимальний варіант для створення майбутньої веб-системи. Також було проагалізовано популярні бази даних для зберігання інфорамації, виявлено основні плюси та мінуси, обрано оптимальну для веб-системи підбору медикаментів.

РОЗДІЛ 2
ІСНУЮЧІ ВЕБ-СИСТЕМИ ПІДБОРУ МЕДИКАМЕНТІВ В ОНЛАЙН РЕЖИМІ

2.1 Огляд існуючих систем підбору медикаментів в онлайн-режимі

Проаналізувавши мережу Інтернет із запитом «підбір медикаментів» було знайдено тільки «Ліки контроль», «Національний перелік лікарських засобів» і офіційний сайт МОЗ (Міністерства охорони здоров'я) і декілька онлайн-аптек.

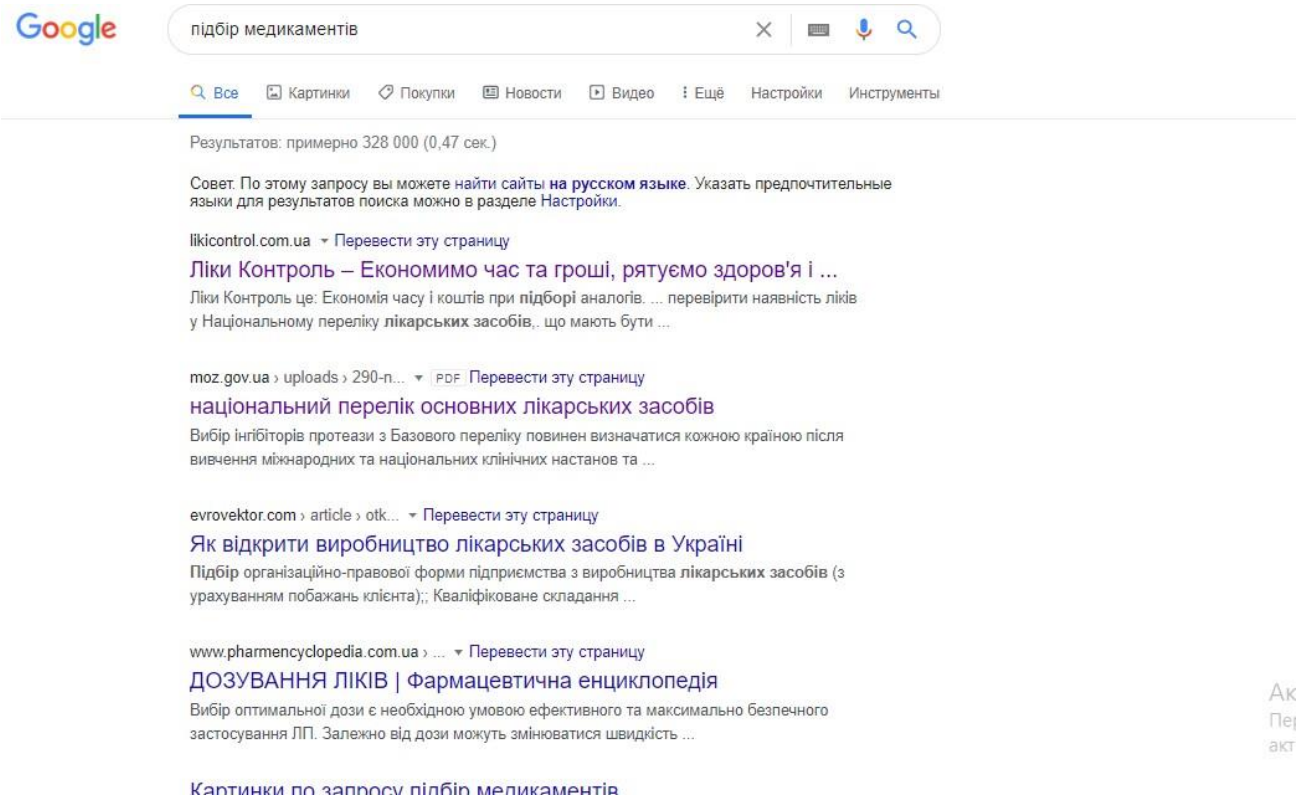


Рисунок 2.1.1 - Пошук в пошуковій системі Google запиту – «підбір медикаментів»

Онлайн-аптеки поширене явище в мережі Інтернет, але в більшості вони орієнтовані на своє ім'я, тобто на те, щоб користувач, знайшов та купив той чи інший медикамент саме у них в аптеці. Тому їхні застосунки орієнтовні на те, щоб знайти найближчу аптеку клієнту.

До прикладу візьмемо Liki24 [8] – веб-система пошуку медикаментів: Головним недоліком є те, що при реєстрації обов'язково потрібно вказати телефон, також немає профілю для лікаря, аби він міг залишатикоментарі під

препаратами. Наступне, що немає вподобаних або переглянутихпрепаратів, є тільки корзина.

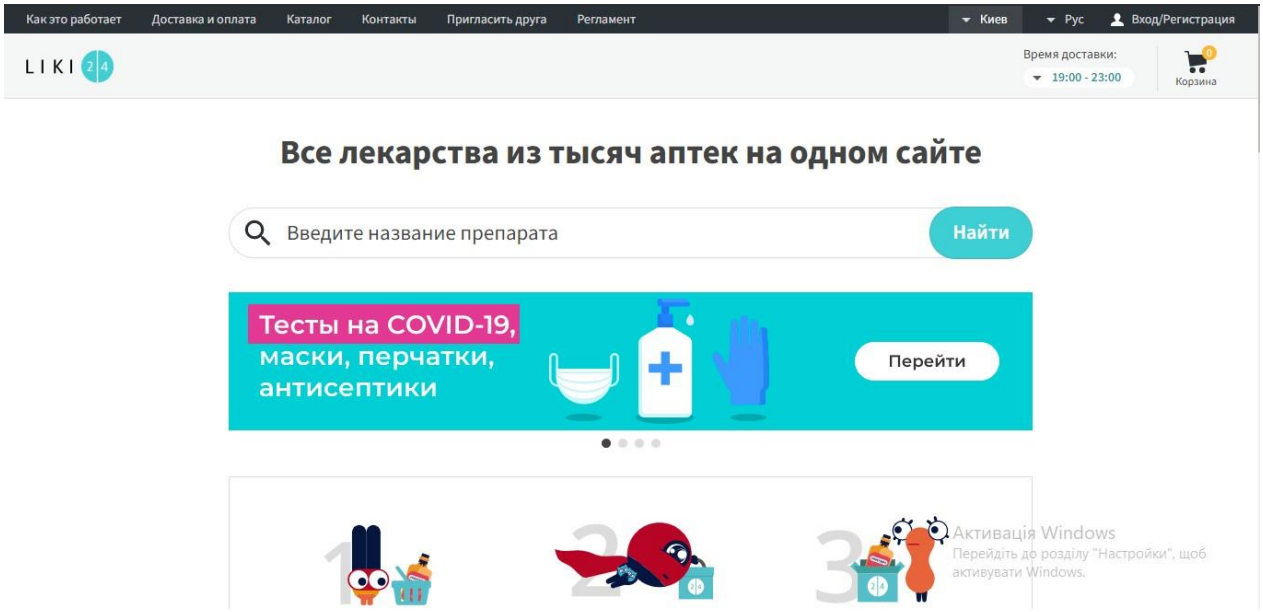


Рисунок 2.1.2 - Головна сторінка ресурсу Liki24

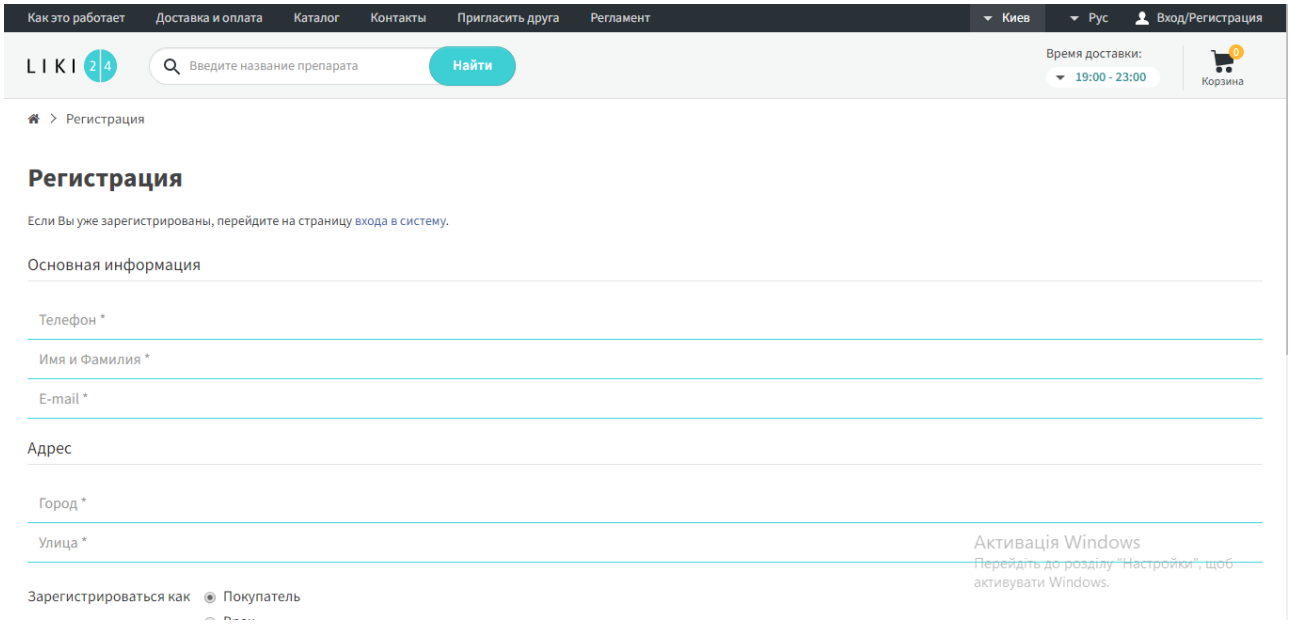


Рисунок 2.1.3 - Сторінка реєстрації ресурсу Liki24

Тобто сайт орієнтовний більше на доставку різних препаратів, а не правильний та комфортний підбір медикаментів в онлайн режимі.

Наступним прикладом візьмемо аптеку "Доброго дня” [20] – веб-система пошуку аптек та медикаментів

Знову ж таки одним із основних недоліків є орієнтація на доставку

товару, відсутність збереження медикаментів та пошуку за симптомами.

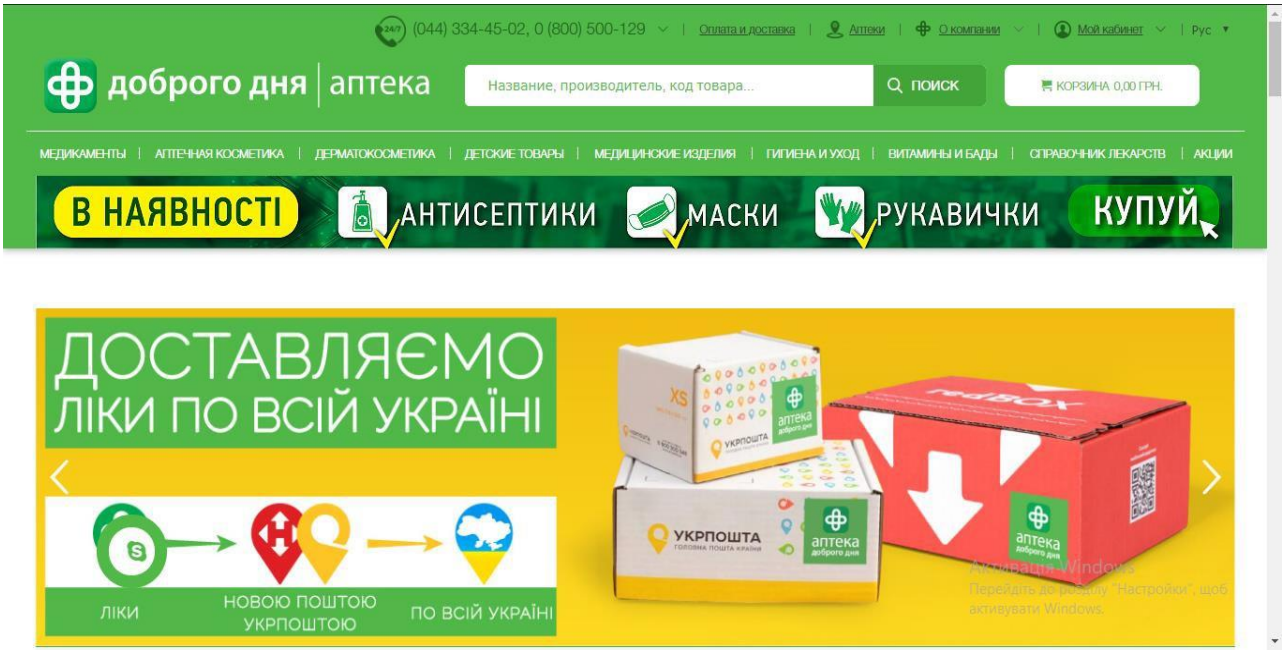


Рисунок 2.1.4 - Головна сторінка ресурсу аптека "Доброго дня"

Також хочу взяти до прикладу Мед-сервіс [3] – веб-система дял підбору медикаментів.

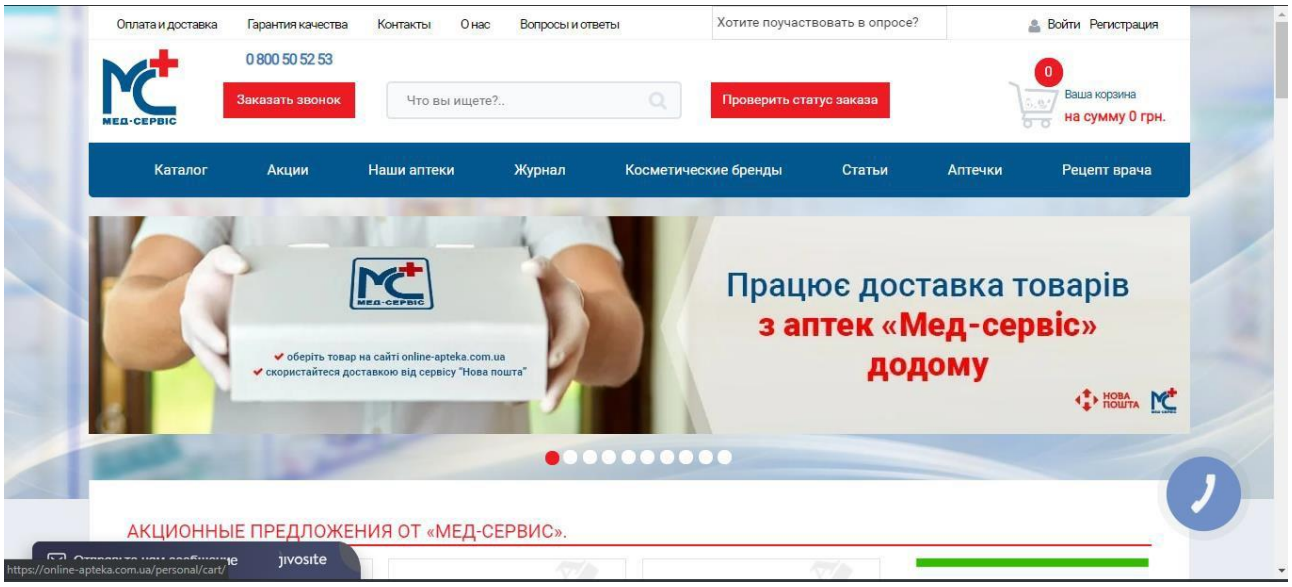


Рисунок 2.1.5 - Головна сторінка Мед-сервісу

Також у реєстраційній формі цього онлайн-сервісу є обов’язкові поля прізвища, пошти та дати народження, що суперечить правилам анонімності для користувача. Також немає профілю медпрацівника.

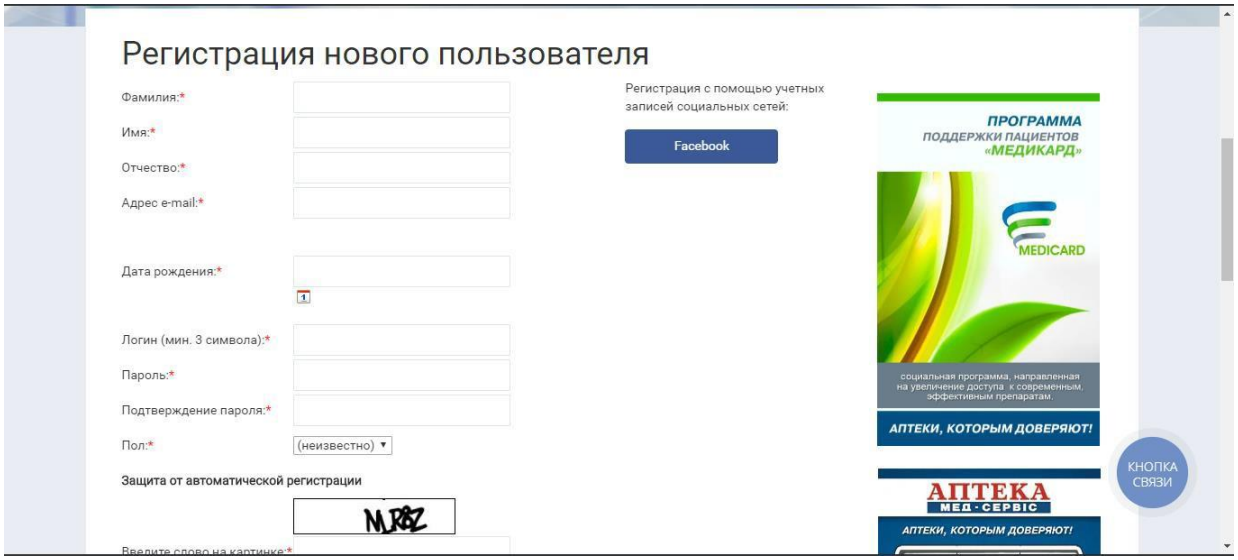


Рисунок 2.1.6 - Реєстраційна форма Мед-сервісу

Ще одним прикладом з схожим функціоналом онлайн-аптеки є Еаптека [13] – веб-система для підбору мендикаментів

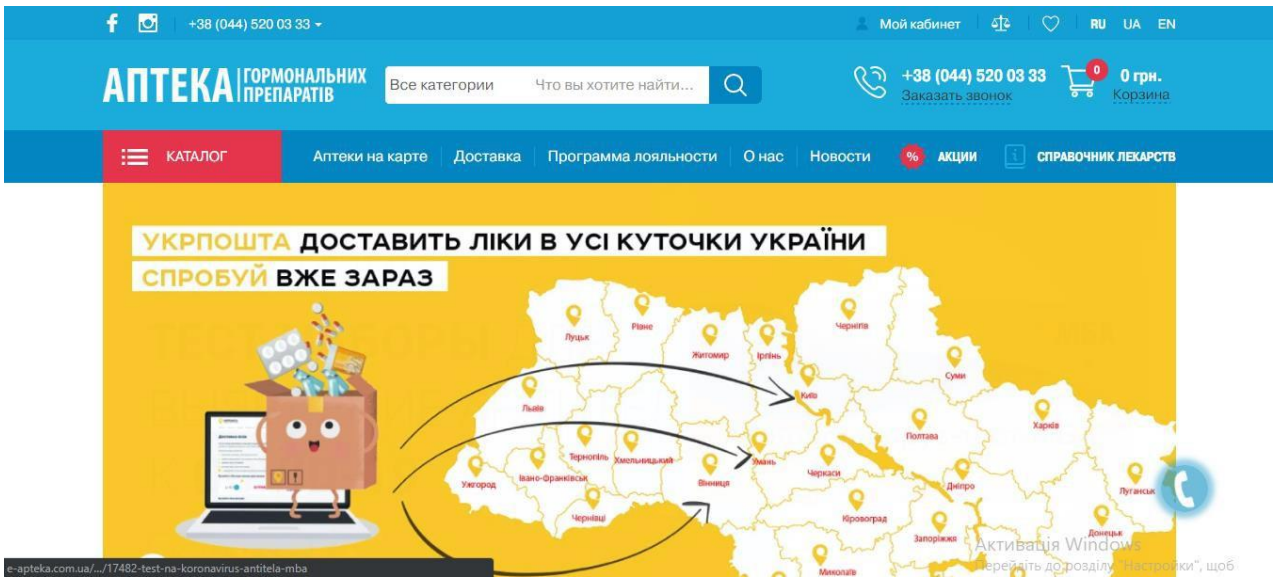


Рисунок 2.1.7 - Головна сторінка веб-сервісу Е-аптека

Першим і найбільшим недоліком цього застосунку є реклама, яка одразу займає більшу частину простору вікна браузера. Наступним недоліком знову ж таки є відсутність будь-яких функцій орієнтованих на лікаря або будь-якого іншого менпрацівника, відсутність коментарів.

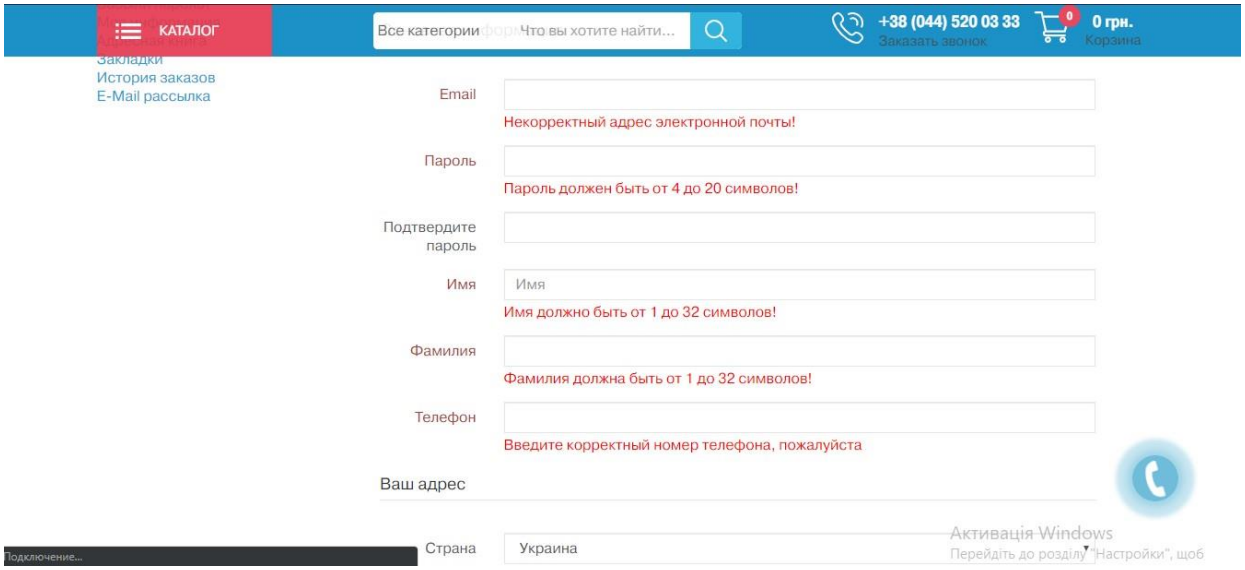


Рисунок 2.1.8 - Сторінка реєстрації веб-сервісу Е-аптека

Переглядавши одні із перших посилань у пошуковій системі Google, можна зробити висновок, що на сьогодні у мережі Інтернет немає доступного безпечного сервісу підбору, зберігання медикаментів та пошуку їх за симптомами, категоріями.Недоліки існуючих систем підбору медикаментів в онлайн-режимі

Проаналізувавши деякі ресурси для пошуку медикаментів, я знайшла декілька недоліків, які упродовж своєї дипломної роботи я виправлю.

Отож, першим недоліком є відсутність збереження вподобаних медикаментів. Раптом користувачу, медичному працівнику потрібно буде зберегти декілька препаратів для зрівняння або для детального ознайомлення в будь-який інший час.

Отож, я вважаю необхідною таку функцію, для комфортного доступу до збережених мед-товарів.

Ще одним мінусом є відсутність пошуку за симптомами. У кожного виникало бажання просто «загуглити» симптому і знайти дієві препарати проти, наприклад, головного болю. На мою думку, найкраще зібрати це в одному веб-застосунку: і пошук медикаментів за назвою,і за категорією, і за симптомами.

Також важливим недоліком усіх ресурів є те, що немає абсолютноніякого сервісу для медичних працівників, де можна залишити коментар під тим чи іншим медикаментом та переглянути інші коментарі кваліфікованих

працівників, аби зробити правильний вибір під час підбору медикаментів для майбутніх пацієнтів; де можна зберігати пацієнтів, ніби як записник, журнал медичного працівника.

Тому у своїй дипломній роботі я створу веб-систему орієнтовану на медичного працівника, в якій він зможе зберігати медикаменіти, коментуватиїх та бачити коментарі інших кваліфіккованих працівників, створювати власний журнал пацієнтів у своєму профілі та слідкувати за їх лікуванням.Шляхи удосконалення та вимоги до розробки систем підбору медикаментів в онлайн-режимі

Найкращим шляхом удосконалення систем підбору медикаментів в онлайн-режимі буде орієнтація на комфортність використання користувачем веб-застосунку, а не на продукт, який потрібно продати. Обов’язковим, вважаю, при розробці, приділення уваги наведеним вище недолікам.

При створенні головної сторінки вважаю за потрібне звернути увагу на мінімізацію реклами, підбір пастельних та приємних людському оку кольорів, Створити сторінку для головних новин сфери медицини та здоров’я. На головній сторінці – важливе повідомлення усім користувачам, що самолікування може бути шкідливе для їхнього здоров’я. Зверху обов’язкове меню для швидкого доступу до всіх можливостей веб-застосунку. В меню має

бути включено:

- логотип веб-застосунку;
- пошукач препаратів і за симптомами, доступний лише медичному працівнику;
- посилання на профіль авторизованого медичного працівника, де буде можливість побачити збережені препарати, заповнити журнал пацієнтів та побачити уже створених, відслідковуючи процес лікування.
- посилання на сторінку головних новин;
- кнопка входу та реєстрації медичного працівника;

Для ілюстрації розташування контенту на головній сторінці, було створено ескіз:

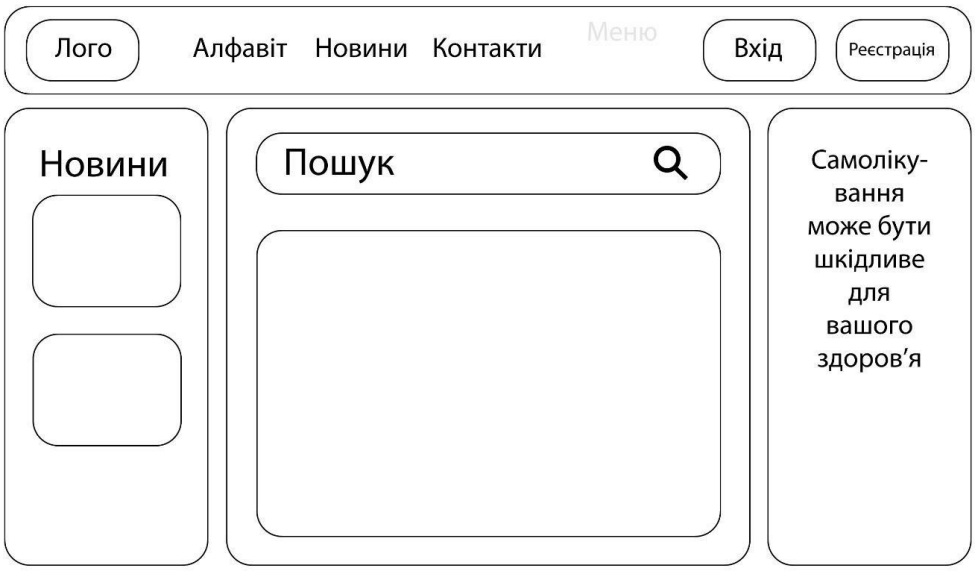


Рисунок 2.3.1 - Ескіз розташування контенту на головні сторінці веб-застосунку

Кольори підібрано максимально пастельні, неяскраві та приємні для людського ока. Так як зі здоров'ям асоціюється зелений колір, тому у веб-застосунку у більшості будуть присутні такі відтінки:

Рисунок 2.3.2 - Відтінки кольорів, які будуть присутні у веб-застосунку

Логотипом застосунку буде серце пастельних тонів і перша буква назви застосунку: Med.search.

Рисунок 2.3.3 - Логотип системи пошуку медикаментів

2.2 Висновки

Було проаналізовано існуючі веб-системи з підбору та пошуку медикаментів. Виявлено основні недоліки з них такі як:

- відсутність збереження вподобаних медикаментів
- надто яскраві кольори та безліч реклами у веб-системах
- відсутність пошуку за симптомами
- відсутність сервісу для медичних працівників

Отже, спираючись на вище наведені недоліки існуючих веб-систем, було виявлено та описано шляхи удосконалення та вимоги до розробки систем підбору медикаментів в онлайн-режимі.

РОЗДІЛ 3
ОПИС ВИМОГ ДО РОЗРОБКИ ВЕБ-СИСТЕМИ ПІДБОРУ МЕНДИКАМЕНТІВ В ОНЛАЙН РЕЖИМІ

3.1 Опис вимог до розробки веб-системи підбору медикаментів в онлайн-режимі

Веб-система має бути створена за допомогою найновішого фреймворку, адже більшість нових фреймворків удосконалені, більш захищені та звертають увагу на пам’ять, тому з використанням таких сторінка завантажується швидше.

Зовнішній вигляд або frontend веб-системи системи підбору медикаментів в онлайн-режимі повинен бути написаний за допомогою JavaScript – фреймворку Nuxt.js.

Nuxt.js - це безкоштовний фреймворк з відкритим кодом, заснований на Node.js, Webpack, Babel.js та Vue.js. Фреймворк Nuxt.js в основному відомий як мета фреймворк для всіх додатків. Більше того, за допомогою Nuxt.js ваш додаток буде дуже оптимізований. Це допоможе вам найкраще створювати програми Vue js. Nuxt поставляється з маршрутизацією та візуалізацією на стороні сервера з коробки. за допомогою Nuxt ви можете створювати односторінкові програми (SPA), статичний сайт та повністю веб-додатки, що надаються на стороні сервера (SSR). Ці програми надзвичайно швидкі та ефективні

Nuxt.js - це, мабуть, найприємніший фреймворк Vue, який можна вибрати. Розробка є надзвичайно плавним процесом, структура файлів красива та організована, а швидкість - це не жарт.

Одне з найбільших позитивних моментів Nuxt.js - це спрощення створення універсальних програм. Для опису коду JavaScript використовується універсальний додаток, який може виконуватися як на клієнтській, так і на серверній стороні.

Багато сучасних фреймворків JavaScript, таких як Vue, спрямовані на

створення односторінкових додатків (SPA). У СПА є багато переваг перед традиційним веб-сайтом. Наприклад, можна створювати дуже швидкі інтерфейси, які швидко оновлюються. Але SPA також мають такі недоліки, як тривалий час завантаження, і Google бореться з ними, оскільки спочатку на сторінці немає вмісту, який можна сканувати для цілей SEO. Весь вміст генерується за допомогою JavaScript.

Універсальний додаток - це наявність SPA, але замість того, щоб мати порожню сторінку index.html, ви попередньо завантажуєте програму на веб-сервер і надсилаєте відтворений HTML як відповідь на запит браузера для кожного маршруту, щоб пришвидшити завантаження разів та покращити SEO, полегшуючи Google сканування сторінки.

Nuxt.js допомагає писати універсальні програми простіше. Створення універсальних додатків може бути нудним, оскільки доводиться багато налаштовувати як на стороні сервера, так і на стороні клієнта.

Цю проблему Nuxt.js має на меті вирішити для програм Vue. Nuxt.js спрощує обмін кодом між клієнтом та сервером, щоб можна було зосередитися на логіці додатку.

Nuxt.js надає доступ до таких властивостей, як isServer та isClient, для ваших компонентів, щоб можна було легко вирішити, відтворювати щось на клієнті чи на сервері. Існують також спеціальні компоненти, такі як компонент no-ssr, який використовується, щоб навмисно запобігти отриманнюкомпонента на стороні сервера.

Нарешті, Nuxt надає доступ до методу asyncData всередині компонентів, які можна використовувати для отримання даних та візуалізації їх на стороні сервера.

Це вершина айсберга того, як Nuxt допомагає створювати універсальні програми.

Найбільше нововведення Nuxt поставляється з командою nuxt generate. Ця команда генерує повністю статичну версію веб-сайту. Він генерує HTML

для кожного з маршрутів і поміщає його у свій власний файл.

Наприклад, якщо у є такі сторінки (термін Nuxt для маршрутів):

```
-| pages/
----| about.vue
----| index.vue
```

Рисунок 3.1.1 – Сторінки на веб-сайті

Nuxt створить ось таку структуру папок:

```
-| dist/
----| about/
------| index.html
----| index.html
```

Рисунок 3.1.2 – Структура маршутів, яку створить Nuxt.js

Вигоди від цього дуже схожі на переваги універсальних додатків. Там розмітка, щоб пришвидшити завантаження сторінки та допомогти веб-сканерам пошукових систем та соціальних мереж сканувати веб-сайт.

Різниця полягає в тому, що більше не потрібен сервер. Все генерується на етапі розробки.

Він потужний, оскільки ми отримуємо переваги універсальної візуалізації без необхідності використання сервера. Ми можемо просто розмістити свій додаток на сторінках GitHub або Amazon S3.

Ось декілька основних каталогів, які для нас створює Nuxt.js:

- components - папка, завдяки якій ви можете впорядкувати свої окремі компоненти Vue.
- layouts - папка, що містить основні макети програм.
- pages - папка, що містить маршрути вашого додатка. Nuxt.js читає

всі файли .vue всередині цього каталогу та створює маршрутизатор додатків.

- store - папка, яка містить усі файли Vuex Store у вашому додатку.

Отож, а серверна частина має бути написана на Node.js. Node.js - це серверна платформа, побудована на JavaScript Chrome Engine (V8 Engine). Node.js був розроблений Райаном Далом у 2009 році, а його остання версія - v0.10.36. Визначення Node.js в його офіційній документації таке:

Node.js - це платформа для зручного створення швидких та масштабованих мережевих додатків. Node.js використовує подію, що не блокує модель вводу / виводу, яка робить її легкою та ефективною, ідеальною для використання в реальному часі додатків, що працюють на розподілених пристроях.

Node.js - це відкрите міжплатформенне середовище виконання програм для розробки серверних та мережевих додатків. Програми Node.js написані на JavaScript, і їх можна запускати в режимі виконання Node.js на OS X, Microsoft Windows та Linux.

Node.js також пропонує багату бібліотеку різноманітних модулів JavaScript, що значно спрощує розробку веб-додатків, що використовують Node.js. На серверній частині у веб-застосунку буде прописана вся логіка системи підбору медикаментів в онлайн-режимі, а саме шифрування паролів, роути для створення користувача і пошук по базі даних.

Так як у нашому веб-застосунку у медичного працівника є можливість зареєструватись і потім авторизовуватись за данними введенеми під час реєстрації. Тому нам потрібно десь ці дані зберігати. Також у майбутній веб-системі будуть зберігатися медикаменти та коментірі до них. Для цього має бути обрана база даних – MongoDB.

MongoDB - орієнтована на документи база даних NoSQL, яка використовується для зберігання даних з великим обсягом. Замість використання таблиць та рядків, як у традиційних реляційних базах даних, MongoDB використовує колекції та документи. Документи складаються з пар ключових значень, які є базовою одиницею даних у MongoDB. Колекції містять набори документів та функції, що є еквівалентом таблиць реляційних баз даних.

MongoDB - це база даних, яка з'явилася на світ приблизно в середині2000-х.

Особливості MongoDB

- Кожна база даних містить колекції, які в свою чергу містять документи. Кожен документ може бути різним із різною кількістю полів. Розмір і зміст кожного документа можуть відрізнятися один від одного.
- Структура документа більше відповідає тому, як розробники конструюють свої класи та об’єкти у відповідних мовах програмування. Розробники часто скажуть, що їхні класи не є рядками та стовпцями, але мають чітку структуру з парами ключ-значення.
- У рядках (або документах, як їх називають у MongoDB) не потрібно заздалегідь мати схему. Натомість поля можна створювати на льоту.
- Модель даних, доступна в MongoDB, дозволяє представити ієрархічні відносини, легше зберігати масиви та інші більш складні структури.
- Масштабованість - середовища MongoDB дуже масштабовані. Компанії по всьому світу визначили кластери, де деякі з них працюють 100+ вузлів із мільйонами документів у базі даних

Чому ми маємо використовувати MongoDB? Бо вона документоорієнтована - оскільки MongoDB - це база даних NoSQL, замість того, щоб мати дані у форматі реляційного типу, вона зберігає дані в документах. Це робить MongoDB дуже гнучкою та пристосованою дореального становища та вимог ділового світу.

База даних має спеціальні запити - MongoDB підтримує пошук по полях, діапазонах запитів і регулярних пошукових виразів.

Також можна створити індекси для підвищення ефективності пошуку в MongoDB. Будь-яке поле документа MongoDB може бути індексовано.

MongoDB може забезпечити високу доступність наборів реплік. Набір реплік складається з двох або більше екземплярів DBO DB. Кожен член набору реплік може виконувати роль первинної чи вторинної репліки в будь-який час. Первинна репліка - це основний сервер, який взаємодіє з клієнтом і виконує всі операції читання/запису. Вторинні репліки зберігають копію даних первинного за допомогою вбудованої реплікації. Коли первинна репліка виходить з ладу,

набір реплік автоматично переходить на вторинну, і тоді вонастає основним сервером.

Також MongoDB використовує концепцію різкості для масштабування по горизонталі шляхом розподілу даних на декілька екземплярів MongoDB. MongoDB може працювати на декількох серверах.

3.2 Структура веб-системи

На мою думку, найлегше відобразити та показати архітектуру на структуру майбутньої веб-системи за допомогою UML-діаграм.

Будь-яку складну систему найкраще зрозуміти, зробивши певні схеми чи картинки. Діаграми краще впливають на наше розуміння. Якщо ми подивимось навколо, ми зрозуміємо, що діаграми - це не нова концепція, але вона широко використовується в різних формах у різних галузях.

Ми створимо діаграми UML, щоб краще та простіше зрозуміти систему. Однієї схеми недостатньо, щоб охопити всі аспекти системи. UML визначає різні види діаграм, що охоплюють більшість аспектів системи. Діаграми компонентів представляють уявлення про реалізацію системи. На етапі проектування програмні артефакти (класи, інтерфейси тощо) системи розташовуються в різних групах залежно від їх взаємозв'язку. Зараз ці групи відомі як компоненти. Можна сказати, що схеми компонентіввикористовуються для візуалізації реалізації.

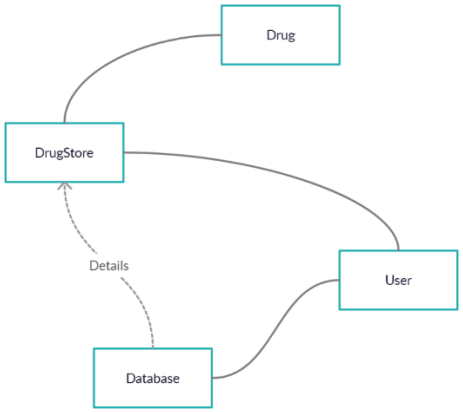


Рисунок 3.2.1 – Діаграма компонентів веб-системи для підбору Медикаментів

3.3 Висновки

Було описано вимоги до розробки веб-системи підбору медикаментів в онлайн-режимі. Клієнтська частина веб-системи має бути створена за допомогою фреймворку Nuxt.js. Серверна частина має бути створена за допомогою Node.js. А для зберігання даних має бути киористано базу даних MongoDB. Також було писано майбутню структуру веб-системи для підбору медикаментів та підтримки лікарів в онлайн режимі за допомогою UML-діаграми.

РОЗДІЛ 4
ПРЕДСТАВЛЕННЯ ЗАСТОСУНКУ З ЕЛЕМЕНТАМИ ЗАХИЩЕНОГО ПРОГРАМНОГО СЕРВЕРУ ТА ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДЛЯ ПІДБОРУМЕДИКАМЕНТІВ В ОНЛАЙН-РЕЖИМІ

4.1 Представлення реалізованого веб-застосунку

Веб-застосунок реалізовано повністю за вимогами програмного забезпечення та повністю відповідає функціоналу описаному у третьому розділі. У додатку А представлені приклади основних файлів серверної частини веб-системи для підбору медикаментів. У додатку Б представлені приклади основних файлів та компонентів клієнтської частини веб-системи для підбору медикаментів Головна сторінка для неавторизованого користувача має вигляд як на рисунку 4.1.1. При натисканні на логотип, ми переходимо на головну сторінку. При натисканні на посилання «Зареєструватись» у верхньому правому кутку перейдемо на сторінку реєстрації. При натисканні на посилання «Увійти» у верхньому правому кутку перейдемо на сторінку авторизації користувача, як на рисунку 4.1.2.

Рисунок 4.1.1 - Головна сторінка веб-системи

Тобто поки користувач не зареєструється або авторизується, йому не доступна функція пошуку, та не доступно перегляд медикаментів.

Як тільки користувач увійде до свого акаунту, його головна сторінка матиме вигляд, як на рисунку 4.1.4.

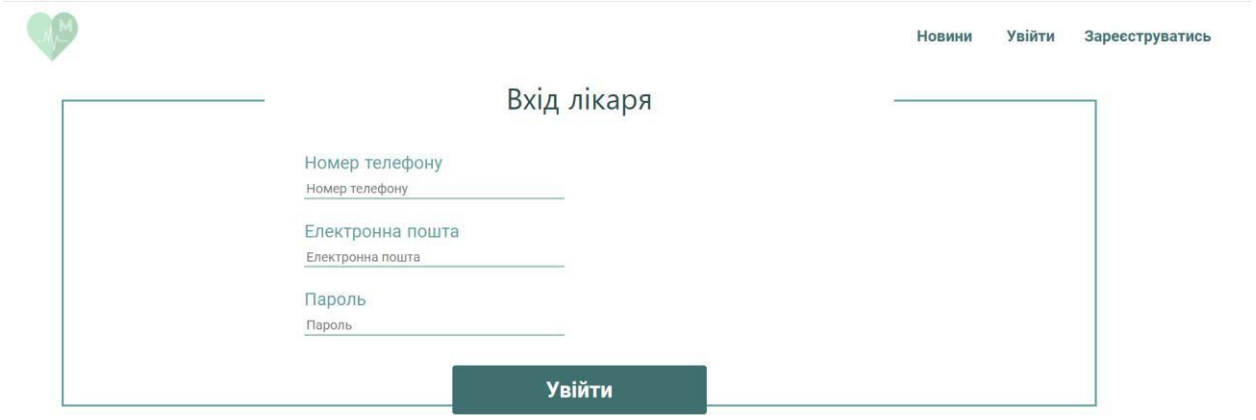


Рисунок 4.1.2 - Сторінка входу веб-системи

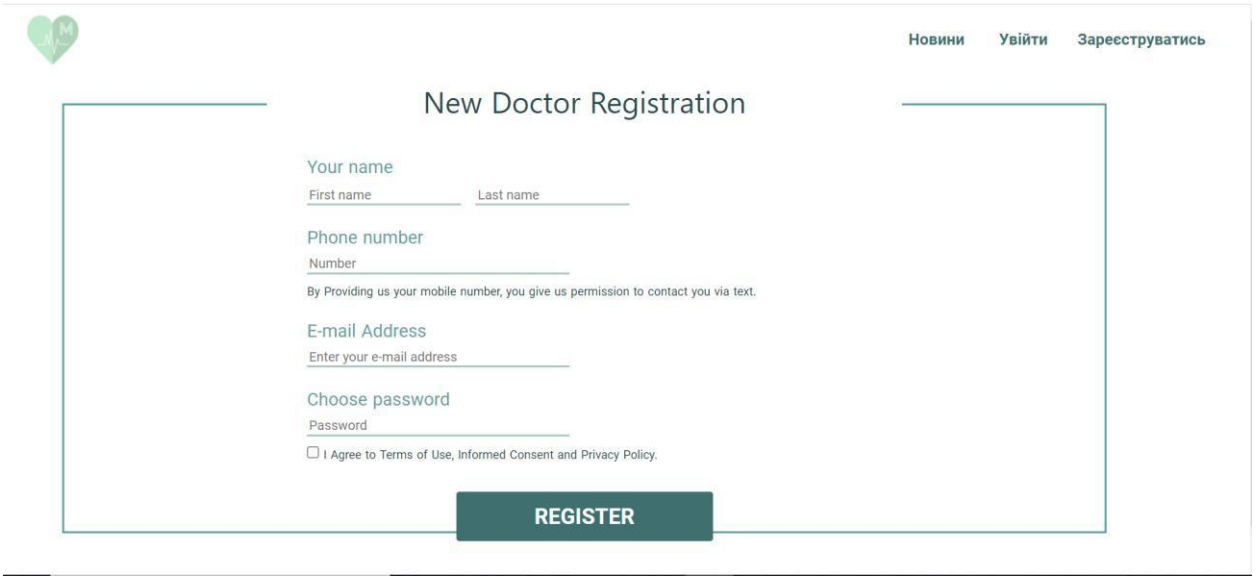


Рисунок 4.1.3 - Сторінка реєстрації веб-системи

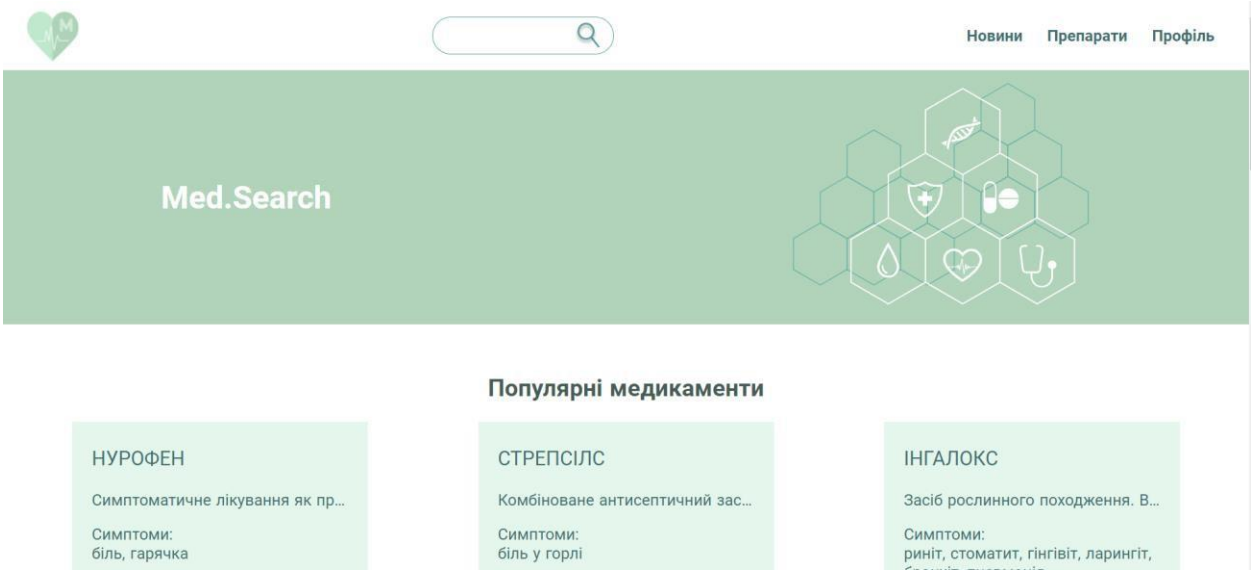


Рисунок 4.1.4 - Головна сторінка веб-системи авторизованого користувача

Прогортавши до низу головну сторінку, видно блок з популярними

медикаментами, де біля кожного знайденого препарату є серце, натиснувшина нього, медичний працівник вподобає медикамент, який буде відображатись у його профілі у таблиці вподобані. Такий функціонл доступний тільки для зареєстрованого кваліфікованого працівника.

Для входу медичному працівнику потрібно телефон, логін та пароль, які він залишав при реєстрації. При неправильному введені даних, з'являється відповідне повідомлення про невдалий вхід. Сторінкаавторизації виглядає так, як показано на рисунку 4.1.2.

Для зручності користувача внизу після кнопки «Увійти» додано запитання: «Чи зареєстровані ви?» та посилання на сторінку реєстрації. При успішному вході користувач перейде на сторінку свого профілю.

Натиснувши на посилання «Реєстрація» медичний працівник перейдена сторінку, яку ми можемо побачити на рисунку 4.1.3.

Для успішної реєстрації потрібні ім'я та прізвище, номер телефону, електронна пошта, пароль, та посада медичного працівнкиа. Обов'язко потрібно поставити галочку на згоду про обробку персональних даних. Натиснувши на кнопку «Зареєструватись», користувач перейде на сторінку профілю, де зможе одразу побачити свій профіль та зберігати своїх пацієнтів. Також для зручності користувача внизу після кнопки «Зареєструвати» додано запитання: «Чи зареєстровані ви уже?» та посилання на сторінку входу.

При успішній реєстрації або авторизації медичний працівник перейде на сторінку свого профілю. Виглядає профіль так, як це показано на рисунку 4.1.5.

При натисканні на кнопку «Додати пацієнта» для медичного працівника відкривається форма, де він зможе ввести ім'я пацієнта, його симптоми, медикаменти, які прописані та процес виздоровлення. Після підтвердження, обраний пацієнт з'являється у профілі лікаря.

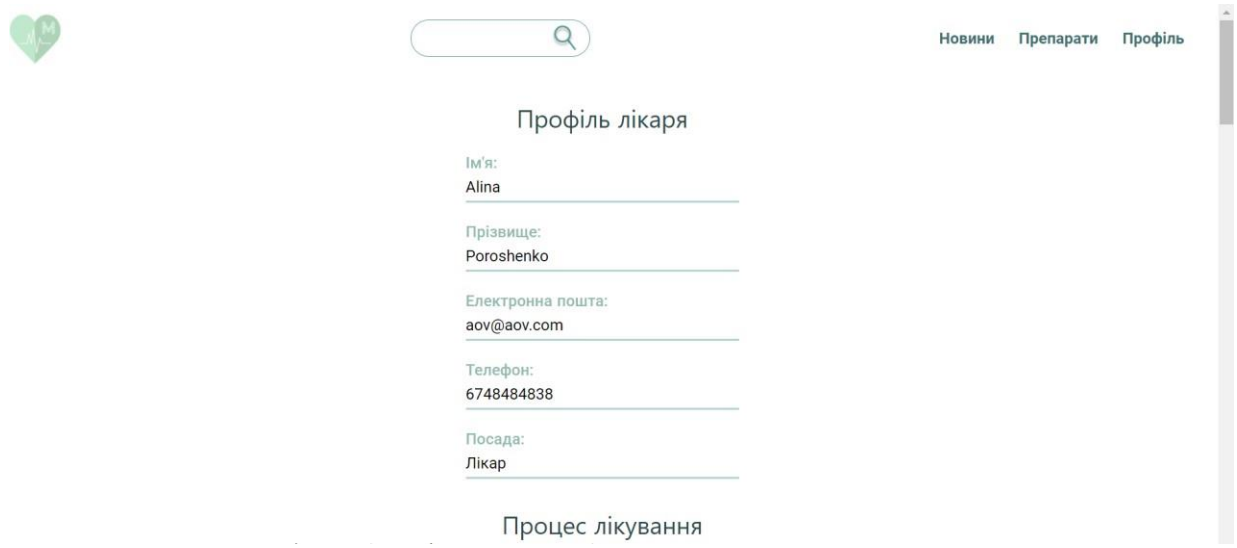


Рисунок 4.1.5 - Сторінка профілю медичного парцівника веб-застосунку

Якщо авторизований працівник перейде на головну сторінку або на сторінку препаратів, новин, то у верхньому кутку будуть посилання «Профіль» так як це показано на рисунку 4.2.3. У той час як у неавторизованого користувача будуть завжди «Вхід» та «Реєстрація».

4.2 Історія користувача

Для того щоб почати шукати медикамент потрібно натиснути на поле вводу і почати вводити симптому, ім’я, категорію, знизу будуть з’являтись варіанти медикаментів в залежності від запита користувача.

Рисунок 4.2.1 - Поле пошуку на головній сторінці веб-застосунку.

Після успішного пошуку, тобто правильного вводу симптом, назв, у користувача знизу у таблиці почнуть з’являтись препарати, які відповідають пошуковому запиту. Але неавторизований користувач не може шукати медикаменти та зберігати їх, тому щоб зареєструватись користувачу потрібно натиснути на кнопку «Реєстрація» у верхньому правому кутку (рисунок 4.2.2).

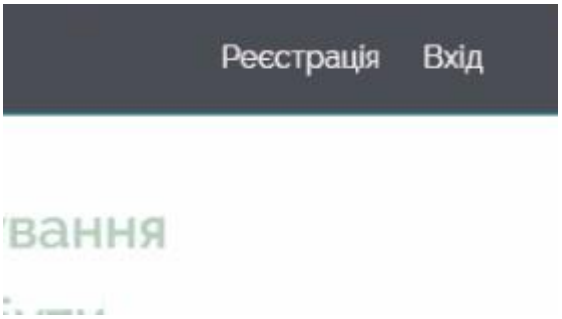


Рисунок 4.2.2 - Фрагмент головної сторінки, де знаходиться кнопка «Реєстрація»

Після переходу на сторінку реєстрація, перед користувачем будуть поля для вводу даних: ім’я, електронна пошта (за бажанням, для простішого відновлення паролю), пароль та повторити пароль. Поле електронна пошта можна залишити пустим, помилки не виникне. Так як наш застосунок поважає анонімність наших користувачів.

Після успішної реєстрації користувача переносить на головну сторінку,де уже у верхньому кутку з’являються посилання «Профіль».

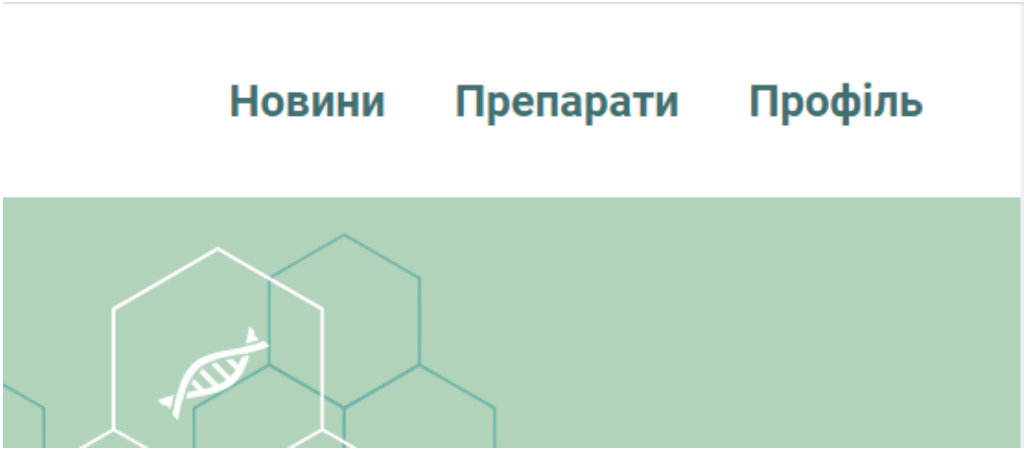


Рисунок 4.2.3 - Фрагмент головної сторінки авторизованого користувача

У авторизованого користувача тепер є можливість зберігати медикаменти.

Щоб вподобати якийсь препарат, потрібно спочатку його знайти за допомогою пошукового поля або перейти на сторінку препаратів і у відображених медикаментів натиснути на серце.

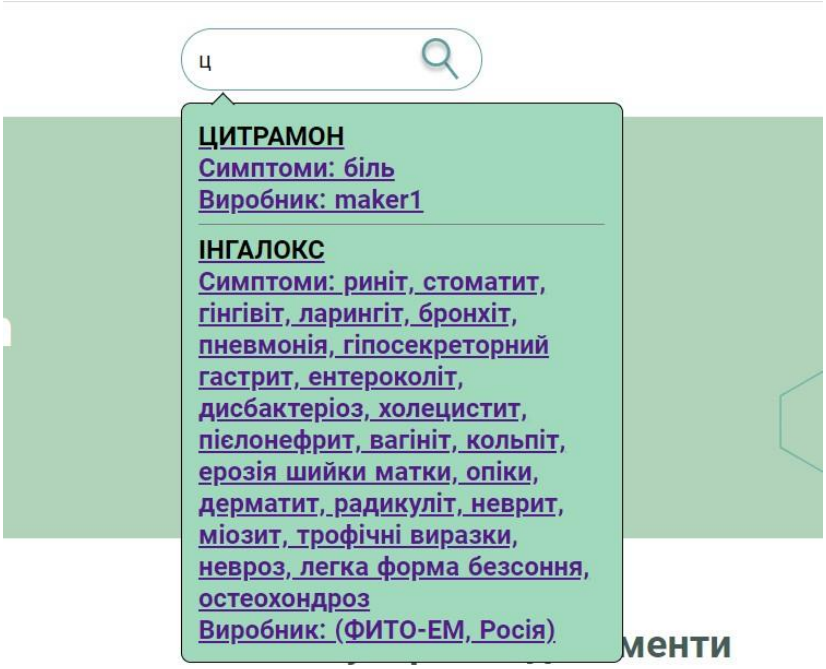


Рисунок 4.2.4 - Фрагмент знайдених препаратів, де у користувача єможливість вподобати медикамент

Аби побачити збережені препарати достатньо натиснути на посилання «Профіль» у верхньому правому кутку, як на рисунку 4.2.3. Після цієї дії користувач побачить вподобані медикаменти.

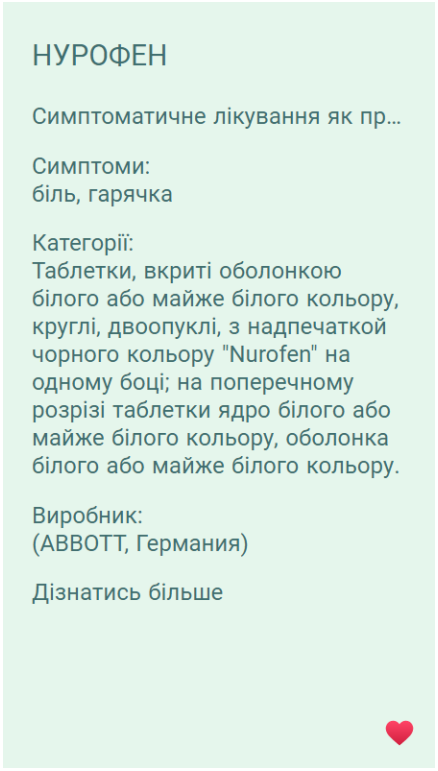


Рисунок 4.2.5 - Препарат, де у лікаря є можливість вподобати медикамент

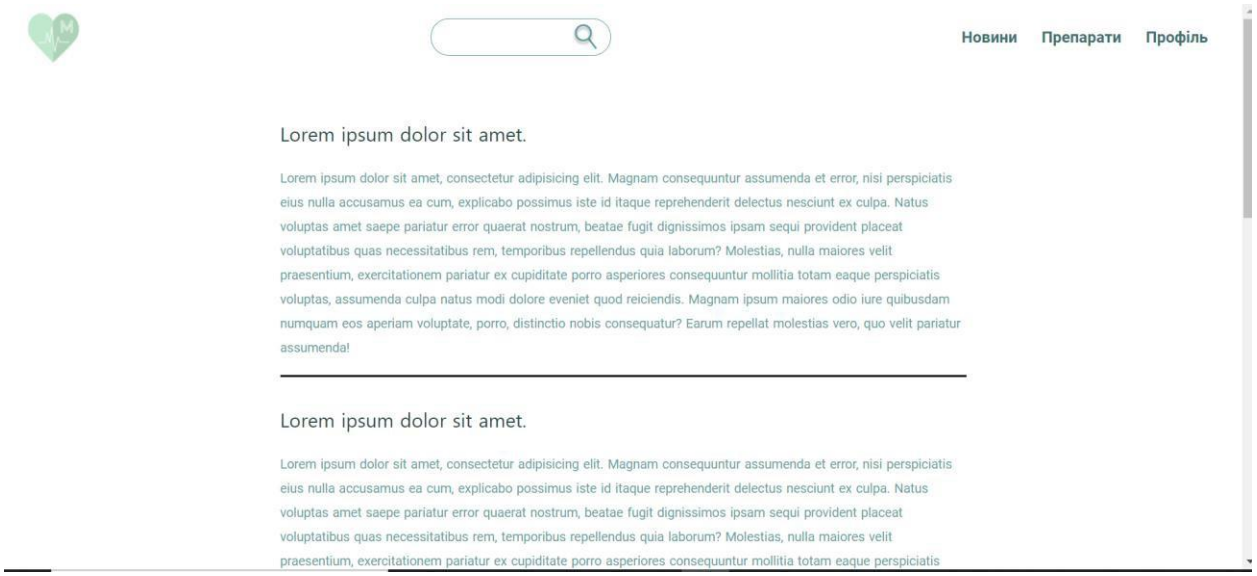


Рисунок 4.2.6. - Сторінка новин

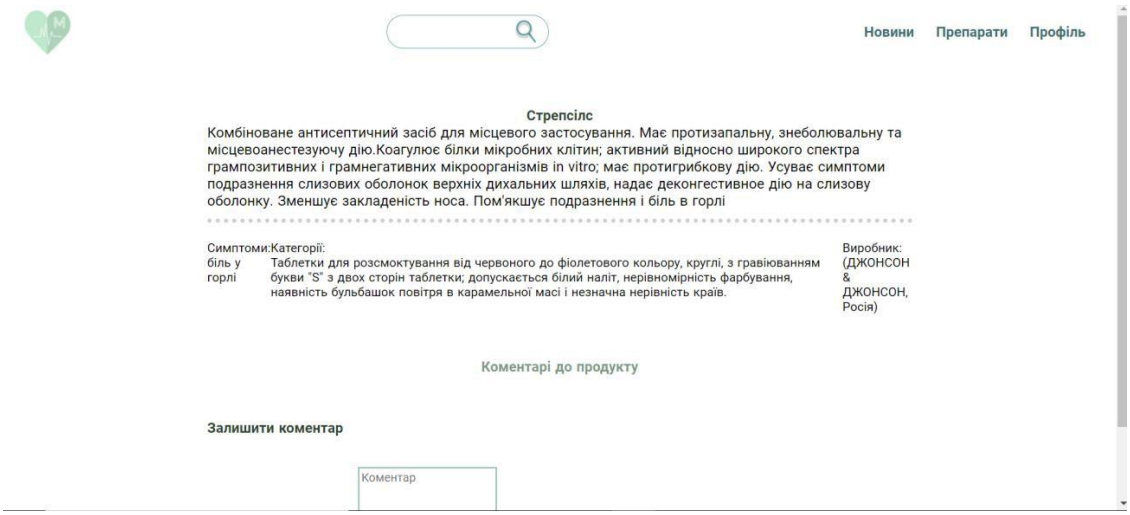


Рисунок 4.2.7 – Сторінка карточки медикаменту

Коже медичний працівник має можливість коментувати медикамент, залишати рицензію на нього (рисунок 4.2.8), аби в майбутньому лікарям було легше обрати медикаменти для лікування пацієнтів.

Рисунок 4.2.8 – Форма для залишання коментарів

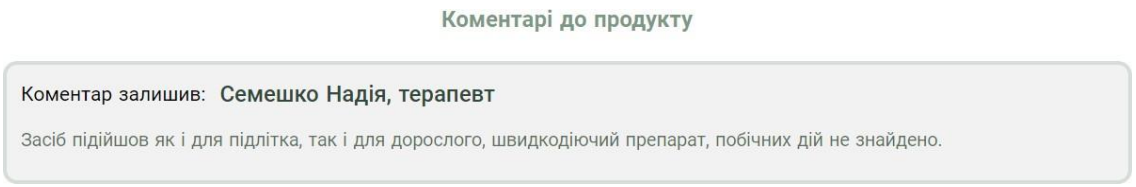


Рисунок 4.2.9 – Приклад залишеного коментаря

Також у авторизованих медичних працівників є можливість скористатись елктронним журналом та записати собі, ніби у нотатки, пацієнтів, що є дуже зручним, адже можна легко знайти медикамент і одразу його записати у електронні нотатки на кожного пацієнта, запис пацієнтів не обмежений.

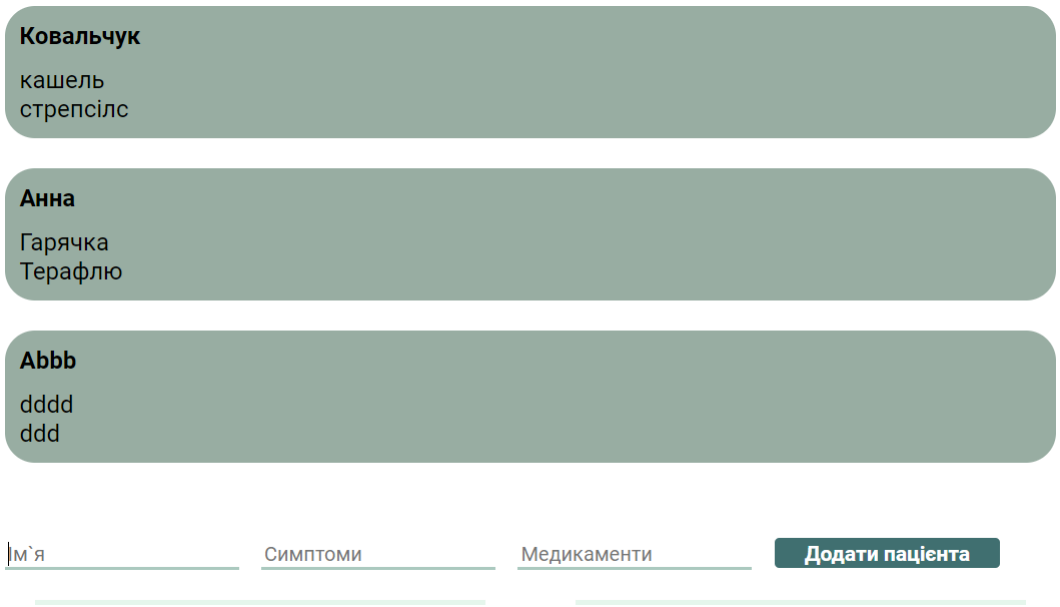


Рисунок 4.2.9 – Приклад додавання пацієнта у журнал та відображення пацієнтів у журналі

Щоб вийти зі свого профілю користувачу потрібно натиснути на кнопку «Вийти» у профілі з самого низу (рисунок 4.2.8).

Рис 4.2.8 – Кнопка виходу з акаунту на сторінці профілю

Висновки

На основі огляду та аналізу декількох систем підбору медикаментів в онлайн-режимі, за допомогою пошукового сервісу Google, було визначено низку недоліків у веб-системах такого формату.

Одним із недоліків є те, що у всіх доступних веб-застосунках немає пошуку за симптомами. Що значно полегшує користувачу підбір медикаментів.

Також основним недоліком є те, що у кваліфікованих працівників немає можливості коментувати медикаменти та спостарігати, аналізувати коментарі інших медикаментів, аби в майбутньому коректно вибрати лікуваннямайбутніх мацієнтів. Також проаналізувавши всі схожі системи, визначила, що ніде немає можливості лікарю зберігати пацієнтів, тобто мати електроннийжурнал.

Ще одним, але не менш важливим, недоліком є дуже багато реклами на сторінках, яка просто закриває весь контент веб-застосунку, і користувач губиться у навігації по сторінці. А також підбір надто яскравих кольорів.

Проаналізувавши усі знайдені недоліки, було описано шляхи удосконалення таких систем підбору медикаментів.

Для усунення усіх вище згаданих недоліків було вирішено спроектувати веб-систему для підбору медикаментів та підтримки лікарів, аби усунути вище згадані недоліки.

Удосконалену систему підбору препаратів реалізовано за допомогою таких технологій: Nuxt.js, Node.js, Express.js, MongoDB.

Розроблений застосунок значно поліпшує та полегшує користувачу пошук та підбір медикаментів, а також дає можливість зареєстрованому кваліфікованому працівнику легко зберегти вподобані медикаменти, аніж додавати кожну сторінку з медикаментом у закладки браузера. Також дає можливість лікарю зберегти пацієнта та бачити їх у профілі, ніби в електронному журналі.

Перелік використаної літератури

1. КалебД. Розробка веб-застосунків за допомогою Node.js.MongoDB та Angular, Бред Дейлі, Брендан Дейлі, Калеб Дейлі., 2020. С. 656.

2. Алекс Я. Node.JS в дії. Алекс Янг, Бредлі Мек, Майк Кантелон.2018. С. 432.

3. Мед-сервіс. Веб-система пошуку аптек та медикаментів [Електронний ресурс]. Режим доступу до ресурсу: https://online-apteka.com.ua/

4. Джон Р. Секрети JavaScript ніндзя. Джон Резіг, Беєр Бібо, ЙосипМарас. 2019. С. 544.

5. Себастіан П. JWT Handbook. Себастіан Пейрот. 2017. С. 94.

6. Деніал Л. Building enterprise JavaScript Applications. Деніал Лі. 2018. С. 710.

7. Rozetka. Електронний магазин [Електронний ресурс]. Режимдоступу до ресурсу: https://rozetka.com.ua/

8. Liki24. Веб-система пошуку медикаментів [Електронний ресурс].Режим доступу до ресурсу: https://liki24.com/uk/

9. Субхашині Ч. MongoDB Recipes: With Data Modeling and QueryBuilding Strategies, Дхаранітхаран Ганесан. 2019. С. 258.

10. Грег Л. Beginning Node.js, Express & MongoDB Development. ГрегЛім., 2019. С. 151.

11. Грищенко В.М. Метод об’єктно-компонентного проектування програмних систем // Проблеми програмування. – 2007. – № 2. – с. 113-125.

12. Герардус Б. MongoDB Transformation A Complete Guide. ГерардусБлокдик., 2019. С. 308.

13. Еаптека. Веб-система пошуку аптек та медикаментів [Електронний ресурс]. Режим доступу до ресурсу: https://e-apteka.com.ua/ua/

14. Фленаган Д. JavaScript. Повне керівництво. 7-е видання / ДевідФленаган., Діалектика 2021. С. 720.

15. Лаврищева Е.М., Грищенко В.М. Сборочное программирование.Основы индустрии программных продуктов. – Второе изд. – К.: Наук. думка,2009. – 371 с.

16. Файн Я., Моїсеєв А. Ф17 Angular и TypeScript. Сайтостроение для профессионалов. — СПб.: Питер, 2018. — 464 с.: ил. — (Серия «Библиотека программиста»).

17. Фентон С. Pro TypeScript: Application-Scale JavaScript Development / Стів Фентон. – Apress, 2014. – 216с.

18. Структура і принципи WEB. WEB-сервери та принципи їх роботи з користувачем [Електронний ресурс]. 2015. Режим доступу до ресурсу: https://dl.sumdu.edu.ua/textbooks/86975/413008/index.html.

19. Веб-безпека. Різновиди веб-систем та їх основні вразливості [Електронний ресурс]. Режим доступу до ресурсу: http://websecurity.com.ua/security/chapter1/

20. Мережа "Аптека Доброго Дня". Про компанію [Електронний ресурс]. Режим доступу до ресурсу: https://www.add.ua/about-company-apteka- dobrogo-dnya/apteka-dobrogo-dnya/

21. Національний перелік основних лікарських засобів [Електронний ресурс]. Режим доступу до ресурсу: https://moz.gov.ua/uploads/0/290-nacionalnij_perelik_osnovnih_likarskih_zasobiv.pdf

Додатки

Додаток А

```
const express = require("express");
const PORT = process.env.PORT || 4000;
const morgan = require("morgan");
var cors = require("cors");
const jwt = require('jsonwebtoken');
const bodyParser = require("body-parser");
const mongoose = require("mongoose");
const config = require("./config/db");
const app = express();
const accessTokenSecret = 'youraccesstokensecret';
mongoose.set("useCreateIndex", true);
mongoose
  .connect(config.database, { useNewUrlParser: true })
  .then(() => {
    console.log("Database is connected");
  })
  .catch(err => {
    console.log({ database_error: err });
  });

app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json());
//configure body-parser ends here
app.use(morgan("dev")); // configire morgan
var corsOptions = {
  origin: '*',
  optionsSuccessStatus: 200 // some legacy browsers (IE11, various SmartTVs) choke on 204
}
app.use(cors(corsOptions));

// define first route
app.get("/", (req, res) => {
  res.json("Hola MEVN devs...Assemble");
});
```

Рисунок А1 – Файл з підключенням бази даних MongoDB

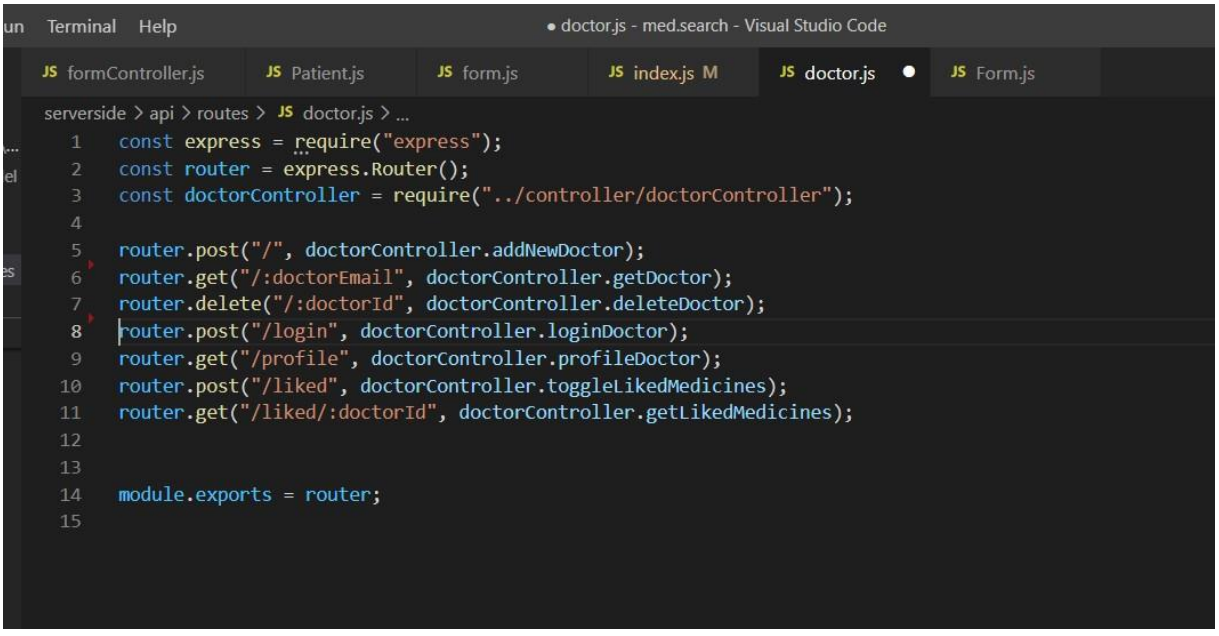


Рисунок А2 – Роути для збереження, видалення, отримання даних зтаблиці лікаря у базі даних

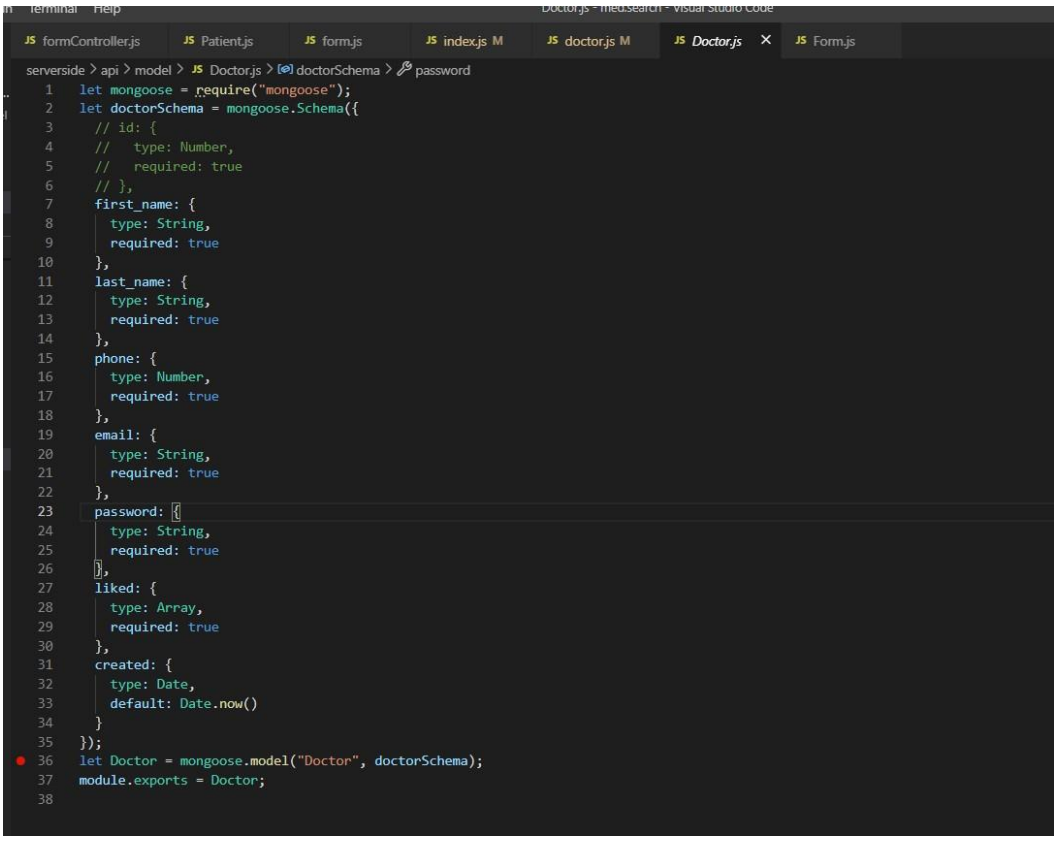


Рисунок А3 – Модель бази даних для лікаря

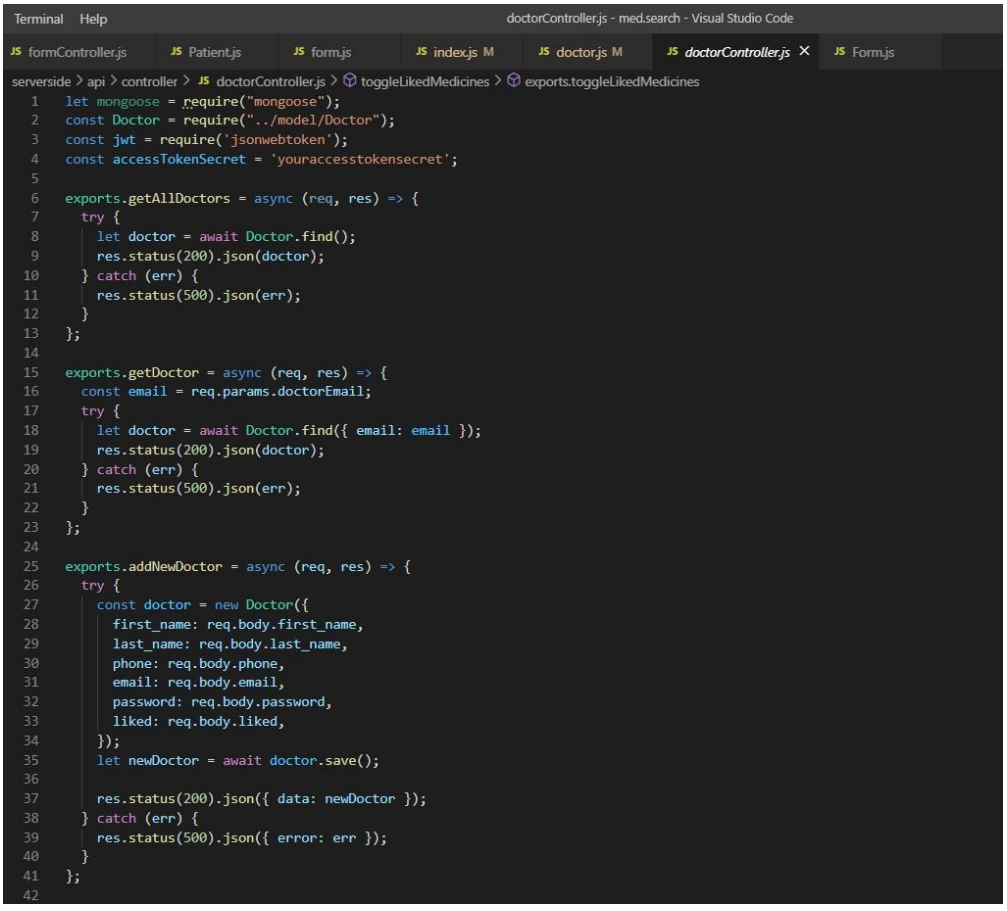
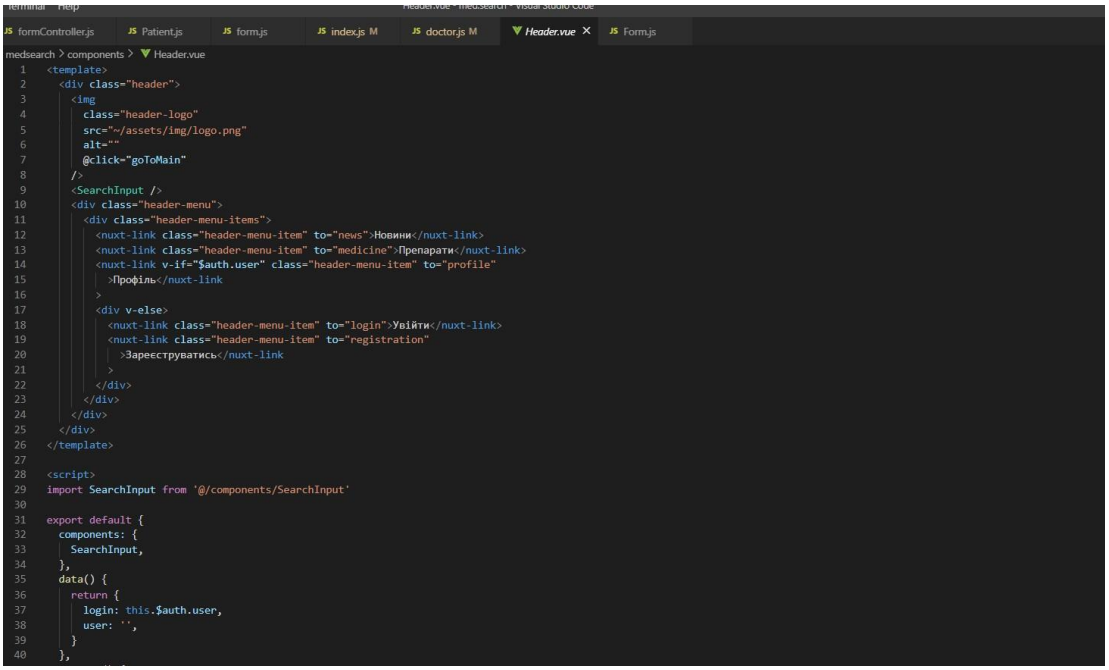


Рисунок А4 – Контроллер з функціями збереження, видалення, отримання даних з таблиці лікаря у базі даних

Додаток Б

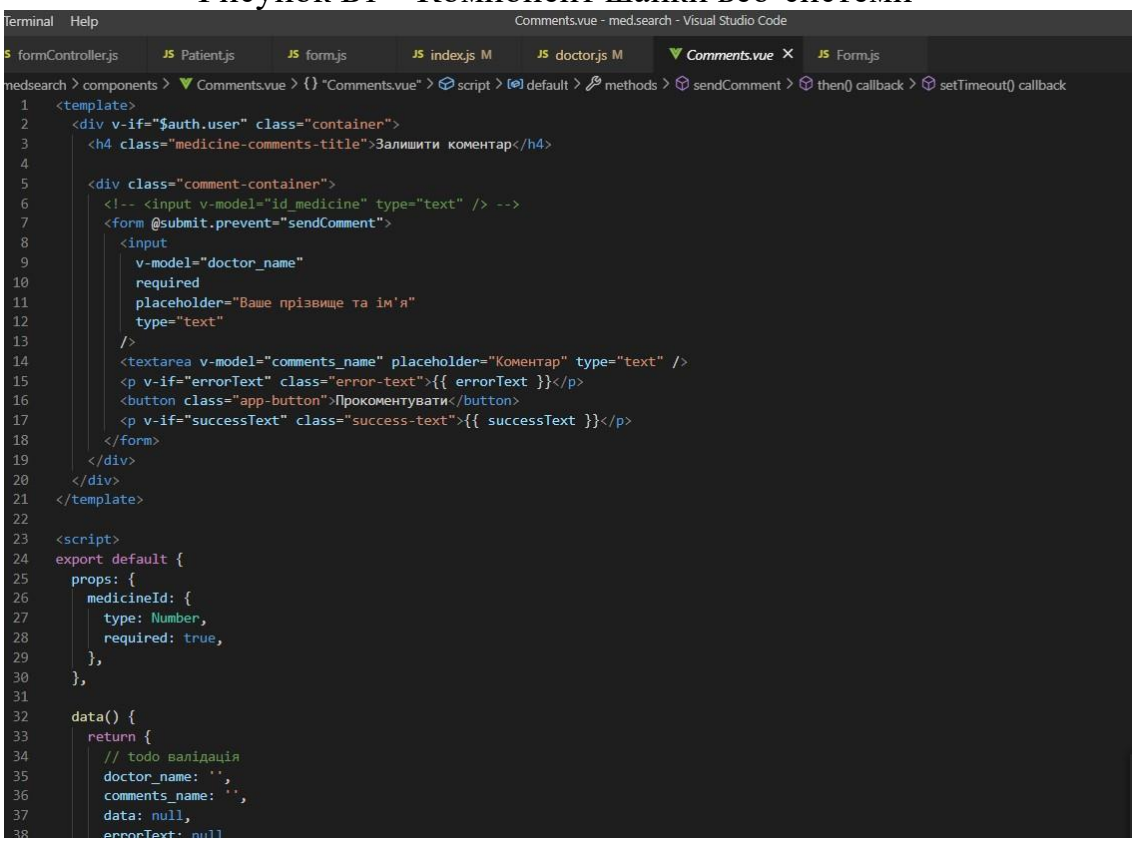


```
<template>
  <div class="header">
    <img
      class="header-logo"
      src="~/assets/img/logo.png"
      alt=""
      @click="goToMain"
    />
    <SearchInput />
    <div class="header-menu">
      <div class="header-menu-items">
        <nuxt-link class="header-menu-item" to="news">Новини</nuxt-link>
        <nuxt-link class="header-menu-item" to="medicine">Препарати</nuxt-link>
        <nuxt-link v-if="$auth.user" class="header-menu-item" to="profile"
          >Профіль</nuxt-link
        >
        <div v-else>
          <nuxt-link class="header-menu-item" to="login">Увійти</nuxt-link>
          <nuxt-link class="header-menu-item" to="registration"
            >Зареєструватись</nuxt-link
          >
        </div>
      </div>
    </div>
  </div>
</template>

<script>
import SearchInput from '@/components/SearchInput'

export default {
  components: {
    SearchInput,
  },
  data() {
    return {
      login: this.$auth.user,
      user: '',
    }
  },
```

Рисунок Б1 – Компонент шапки веб-системи



```
<template>
  <div v-if="$auth.user" class="container">
    <h4 class="medicine-comments-title">Залишити коментар</h4>

    <div class="comment-container">
      <!-- <input v-model="id_medicine" type="text" /> -->
      <form @submit.prevent="sendComment">
        <input
          v-model="doctor_name"
          required
          placeholder="Ваше прізвище та ім'я"
          type="text"
        />
        <textarea v-model="comments_name" placeholder="Коментар" type="text" />
        <p v-if="errorText" class="error-text">{{ errorText }}</p>
        <button class="app-button">Прокоментувати</button>
        <p v-if="successText" class="success-text">{{ successText }}</p>
      </form>
    </div>
  </div>
</template>

<script>
export default {
  props: {
    medicineId: {
      type: Number,
      required: true,
    },
  },

  data() {
    return {
      // todo валідація
      doctor_name: '',
      comments_name: '',
      data: null,
      errorText: null
```

Рисунок Б2 – Компоннт коментарів веб-системи

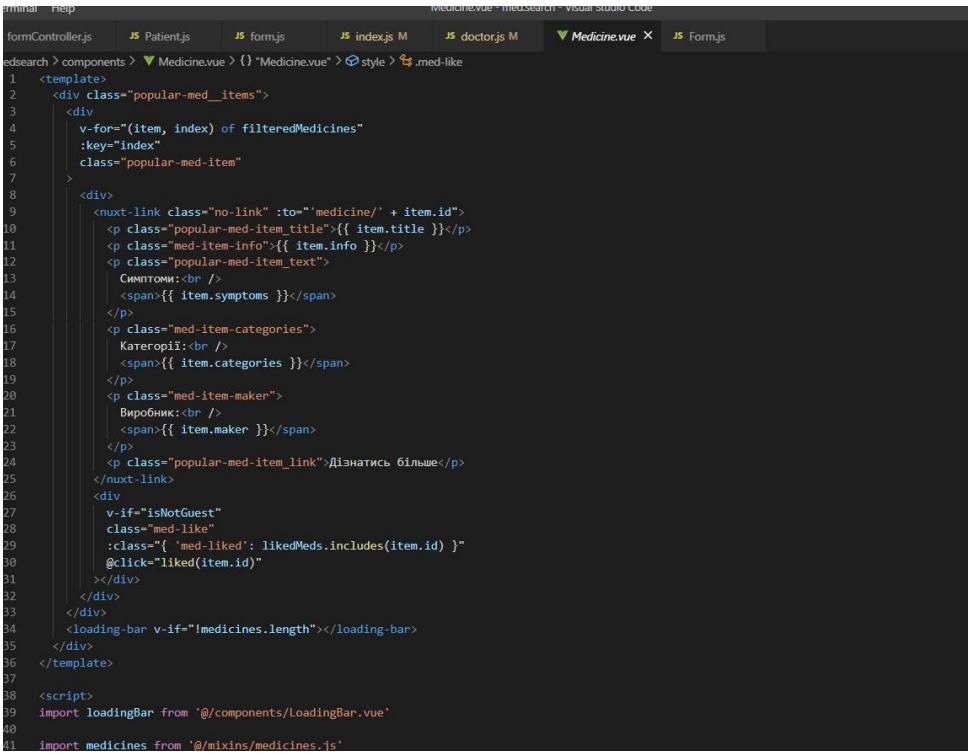


Рисунок Б2 – Компоннт медикаменту веб-системи