Вінницький національний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення

Бакалаврська дипломна робота

на тему: «Розробка програмного забезпечення для системи підтримки фізичного та ментального здоров'я».

Виконав: студент IV курсу
групи 1ПІ-20б
спеціальності
121 – Інженерія програмного забезпечення
(шифр і назва напряму підготовки, спеціальності)
Бохонько Д.В.
(прізвище та ініціали)
Керівник: к.т.н., доц. каф. ПЗ Коваленко О.О.
(прізвище та ініціали)
Рецензент: к.т.н., доц. каф. ОТ Крупельницький Л.В.
(прізвище та ініціали)

Допущено до захисту
Зав. кафедри _____
«10» червня 2024 р.

ВНТУ– 2024

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Рівень вищої освіти перший бакалаврський
Галузь знань12 – Інформаційні технології
Спеціальність 121 – Інженерія програмного забезпечення
Освітньо-професійна програма – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
д.т.н., професор
Романюк О. Н._____
«12 » березня 2024 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Бохоньку Дмитру Володимировичу

1. Тема роботи – «Розробка програмного забезпечення для системи підтримки фізичного та ментального здоров'я».

Керівник роботи: Коваленко Олена Олексіївна, к.т.н., доцент кафедри ПЗ, затверджені наказом вищого навчального закладу від 11 березня 2024 року № 80.

2. Строк подання студентом роботи: 3 червня 2024.

3. Вихідні дані до роботи: операційна система – Windows 11; середовище програмування Visual Studio Code 2022, IntelijIDEA; мови програмування – Java Script, HTML, CSS, Java. Розмір вікна додатку не менше 1024x768. Кольоровий режим: TrueColor.

4. Зміст розрахунково–пояснювальної записки: вступ; обґрунтування вибору методу розробки та постановка задачі дослідження; розробка методу, структури, моделі та алгоритмів роботи програмного продукту; програмна реалізація програмного забезпечення; тестування програми; висновки; список використаних джерел; додатки.

5. Перелік графічного матеріалу: аналоги програмного застосунку; структура графічного інтерфейсу програмного забезпечення; модель роботи системи; алгоритми роботи застосунку; результати тестування.

6. Консультанти розділів бакалаврської дипломної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1-4	Коваленко О.О., к.т.н., доцент	13.03.24	03.06.24

7. Дата видачі завдання 13 березня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи	Примітки
1	Аналіз проблеми, обгрунтування актуальності розробки системи та постановка задачі	14.03.2024 – 22.04.2024	Вик
2	Розробка моделі, методів та алгоритмів роботи програмного забезпечення	22.04.2024 – 29.04.2024	Вик
4	Аналіз і вибір середовища програмування	29.05.2024 – 03.05.2024	Вик
5	Розробка модулів програми	03.05.2024 – 20.05.2024	Вик
6	Тестування роботи розробленого програмного забезпечення	20.05.2024 – 26.05.2024	Вик
7	Оформлення матеріалів до захисту БДР	26.05.2024 – 30.05.2024	Вик

Студент (підпис) **Бохонько Д.В.** (прізвище та ініціали)

Керівник бакалаврської дипломної роботи (підпис) **Коваленко О.О** (прізвище та ініціали)

АНОТАЦІЯ

Бакалаврська дипломна робота складається з 56 сторінок формату А4, на яких є 25 рисунків, 6 таблиць, список використаних джерел містить 20 найменувань.

В рамках бакалаврської дипломної роботи було проведено детальне дослідження програмних забезпечень, спрямованих на підтримку фізичного та ментального здоров’я людей. Аналіз об’єкту, предмету, завдань і методів дослідження був включений до роботи. Розробка програмного продукту для підтримки фізичного та психічного здоров’я мала на меті покращити здоров’я населення. Було зроблено висновок, що розробка є актуальною після аналізу аналогів. У рамках роботи були розроблені алгоритми та блок-схеми для основних операцій, а також був створений макет інтерфейсу користувача.

Програмний продукт був реалізований з використанням мов програмування HTML, CSS, JavaScript та Java. В якості середовища розробки було використано Visual Studio Code та IntelijIDEA. Фінальний програмний продукт відповідає поставленим завданням та меті дослідження. В результаті було створено програмне забезпечення для підтримки фізичного та ментального здоров'я, який надає користувачам зручний та ефективний інтерфейс для отримання інформації щодо фізичного та ментального здоров'я.

Ключові слова: програмне забезпечення, JavaScript, Java, фізичне здоров’я, ментальне здоров’я.

ANNOTATION

The bachelor's thesis consists of 56 A4 pages with 25 figures, 6 tables, and a list of references containing 20 titles.

As part of the bachelor's thesis, a detailed study of software aimed at maintaining the physical and mental health of people was conducted. The analysis of the object, subject, tasks and methods of the study was included in the work. The development of a software product to support physical and mental health was aimed at improving public health. It was concluded that the development is relevant after analyzing analogs. As part of the work, algorithms and flowcharts for the main operations were developed, and a user interface mockup was created.

The software product was implemented using the HTML, CSS, JavaScript and Java programming languages. Visual Studio Code and IntelijIDEA were used as the development environment. The final software product meets the objectives and purpose of the study. As a result, software was created to support physical and mental health, which provides users with a convenient and effective interface for obtaining information on physical and mental health.

Keywords: software, JavaScript, Java, physical health, mental health.

ЗМІСТ

ВСТУП 4
1. ОБҐРУНТУВАННЯ ВИБОРУ МЕТОДУ РОЗРОБКИ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ 8
1.1. Аналіз програмних забезпечень для підтримки фізичного та ментального здоров'я 8
1.2. Порівняльний аналіз аналогів 9
1.3 Аналіз методів розв'язання задачі 13
1.4. Постановка задач розробки програмного забезпечення 14
1.5. Висновки 14
2. РОЗРОБКА МЕТОДУ, СТРУКТУРИ, МОДЕЛІ ТА АЛГОРИТМІВ РОБОТИ ПРОГРАМНОГО ПРОДУКТУ 15
2.1 Аналіз даних 15
2.2 Метод збору даних для підтримки фізичного та ментального здоров'я .. 16
2.3 Розробка структури графічного інтерфейсу програмного забезпечення . 16
2.4 Розробка моделі системи програмного забезпечення 20
2.5 Розробка алгоритмів роботи системи 22
2.6 Висновки 24
3. ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 25
3.1 Варіантний аналіз і обґрунтування вибору засобів для реалізації програмного засобу 25
3.2 Вибір середовища розробки 32
3.3 Розробка клієнтської частини програмного продукту для підтримки фізичного та ментального здоров'я 38
3.4 Розробка серверної частини програмного забезпечення для системи підтримки фізичного та ментального здоров'я 44
3.5 Висновки 46
4 ТЕСТУВАННЯ ПРОГРАМИ 47
4.1 Аналіз методів тестування програмного забезпечення 47
4.2 Тестування розробленого програмного продукту 48

4.3 Розробка інструкції користувача 51
4.4 Висновки 55
ВИСНОВКИ 57
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 58
ДОДАТКИ 60
Додаток А – Технічне завдання 61
Додаток Б. Перевірка на плагіат 65
Додаток В – Лістинг програми 66
Додаток Г – Графічна частина 83

ВСТУП

Обґрунтування вибору теми дослідження. В сучасному світі зростання свідомості про важливість збереження фізичного та ментального здоров'я стає все більш актуальним і високопріоритетним завданням. Поглиблене розуміння взаємозв'язку між фізичним та психічним здоров'ям людини відкриває широкі можливості для розробки та впровадження систем підтримки, спрямованих на поліпшення якості життя.

Одним із головних мотивів вибору даної теми є недостатня доступність і вузьке розповсюдження фізичної та ментальної підтримки. Багато людей зіткнулися з труднощами у зверненні до фізичних або ментальних послуг через різні причини, такі як висока вартість, обмежені ресурси, які пов'язані з пошуком допомоги. Розробка програмного забезпечення для системи підтримки фізичного та ментального здоров'я надає можливість зручного та конфіденційного доступу до ресурсів, інформації, фізичних та ментальних інструментів для зміцнення і полегшення стану людини.

У виборі даної теми важливим фактором є широкі можливості сучасних вебтехнологій. Створення інтерактивних і функціональних інструментів, які можуть ефективно впливати на фізичне та ментальне благополуччя користувачів, можливо завдяки розробці програмного забезпечення і інструментів для розробки вебзастосунків. Вебзастосунки можуть виконувати різні завдання, наприклад, вони можуть пропонувати психологічну інформацію, допомогу в реальному часі та інструменти для фізичного здоров'я.

Дослідження, спрямовані на створення програмних забезпечень для фізичного та ментального здоров'я, можуть не тільки покращити доступ до психологічної та фізичної допомоги, але й розширити область досліджень у цій галузі. Розробка нових алгоритмів, інтерфейсів і методик фізичної та ментальної підтримки може сприяти постійному покращенню в цій галузі.

Програмна системи «HealthyLife» дозволяє користувачам легко та швидко знаходити інформаційні матеріали, які відповідають їхнім потребам у

відношенні до їх здоров'я. Даний проєкт є кроком у вдосконаленні програмних застосунків та збільшенні кількості користувачів, які можуть користуватися даною інформацією. Створення програмного забезпечення для системи підтримки фізичного та ментального здоров'я є необхідним для задоволення зростаючих потреб користувачів та слідування тенденціям розвитку інформаційних технологій у сфері охорони здоров'я

Отож, тема «Розробка програмного забезпечення для системи підтримки фізичного та ментального здоров'я» є актуальною та важливою, оскільки вона спрямована на покращення доступності та ефективності фізичної та ментальної допомоги. Сучасні технології розробки дозволяють створювати функціональні інструменти для підтримки фізичного та ментального здоров’я. Крім того, застосування має здатність створювати зручну, конфіденційну та взаємодійну платформу для користувачів.

Зв’язок роботи з науковими програмами, планами, темами. Наукова робота виконувалася відповідно до плану наукових досліджень кафедри програмного забезпечення.

Мета і завдання дослідження. Метою дослідження є поліпшення фізичного та ментального благополуччя людей шляхом розробки програмного забезпечення для системи підтримки фізичного та ментального здоров’я.

Основними задачами дослідження є:

- оцінка поточного стану програмних забезпечень для підтримки фізичного та ментального здоров’я;
- порівняльний аналіз аналогів;
- аналіз методів розв’язання задачі;
- постановка задач для програмного забезпечення для підтримки фізичного та ментального здоров’я;
- розробка структури і алгоритмів програмного продукту;
- розробка дизайну інтерфейсу програмного забезпечення;
- варіантний аналіз і обґрунтування вибору засобів для реалізації програмного продукту;

- розробка програмного забезпечення для підтримки фізичного та ментального здоров’я;
- тестування програми.

Об’єкт дослідження – процес розробки програмного забезпечення для системи підтримки фізичного та ментального здоров’я.

Предмет дослідження – методи та засоби для реалізації програмного забезпечення для системи підтримки фізичного та ментального здоров'я.

Методи дослідження. В роботі було використано методи розробки інтерфейсу програмного забезпечення, методи побудови блок-схем алгоритмів, методи реалізації вебзастосунків мовою JavaScript та Java.

Новизна отриманих результатів. Подальшого розвитку набув метод підбору інформації та, зокрема, алгоритм надання допомоги в фізичному та ментальному плані користувачам застосунку, який, на відміну від існуючих, пропонує актуальні статті, калькулятор BMI, інтерактивний чат-бот для консультацій з фізичних та ментальних питань, а також можливість запису на консультацію, що значно підвищує зручність користування програмним продуктом, забезпечує високий рівень доступності інформації та індивідуальну підтримку фізичного та ментального здоров'я користувачів.

Практична цінність одержаних результатів. Отримані в ході дослідження результати є практично цінними, оскільки вони послужили основою для розробки алгоритмів та програмних засобів для програмного забезпечення, спрямованих на підтримку фізичного та ментального здоров'я, відповідно до теоретичних принципів, викладених у бакалаврській дипломній роботі.

Особистий внесок. Усі наукові висновки та результати були здобуті дослідником індивідуально.

Апробація роботи. Основні положення бакалаврської дипломної роботи доповідалися та обговорювалися на Міжнародна науково-практична інтернет-конференція студентів, аспірантів та молодих науковців "Молодь в науці: дослідження, проблеми, перспективи" (МН-2024).

Публікації. За тематикою дослідження опубліковано 1 наукову працю у збірниках матеріалів конференцій [1].

Структура та обсяг роботи. Бакалаврська дипломна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел, що містять 20 найменувань, 4 додатка. Робота містить 25 ілюстрацій та 6 таблиць.

1. ОБҐРУНТУВАННЯ ВИБОРУ МЕТОДУ РОЗРОБКИ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1. Аналіз програмних забезпечень для підтримки фізичного та ментального здоров’я

У останні роки відбувається зростання популярності програмних забезпечень для систем підтримки фізичного та ментального здоров'я, що дозволяють користувачам отримати доступ до необхідних ресурсів та послуг через Інтернет. Під час аналізу ринку таких платформ можна виділити кілька основних типів, які забезпечують підтримку в цій області.

Перший тип – це платформи, що надають можливість отримання онлайн консультацій з професійними фахівцями. Вони дозволяють користувачам записуватися на індивідуальні консультації через веб-камеру або чат. Основною перевагою цих платформ є можливість отримати кваліфіковану допомогу віддалено та зручно для користувачів з будь-якої локації.

Другий тип – це онлайн курси та тренінги з фізичного та ментального здоров'я, які дозволяють користувачам самостійно вивчати та розвивати свої навички. Ці платформи зазвичай містять відеоуроки, матеріали, завдання та тести для поліпшення здоров'я та самопочуття. Головною перевагою є доступність та можливість навчання у зручний для користувача час.

Третій тип - це мобільні додатки, які надають доступ до інструментів та технік для покращення фізичного та ментального здоров'я. Деякі додатки пропонують медитаційні техніки, вправи дихання та інші корисні інструменти для збереження здоров'я та підтримки добробуту.

Основною складовою успішного програмного забезпечення є його інтерфейс. Вони повинні бути зручними у користуванні та зрозумілими для користувачів. Дані інтерфейси мають включати зрозумілі навігаційні кнопки для пошуку інформації та для можливої взаємодії.

Таким чином, програмні забезпечення для підтримки фізичного та психічного здоров’я потребують додаткового вдосконалення. Дипломна робота

про новий застосунок має на меті надати високоякісну, індивідуальну та безпечну підтримку фізичного та ментального здоров’я користувачів. Використовуючи зручний та ефективний інтерфейс користувача, він спрямований на надання інформації, ресурсів і онлайн-допомоги. Такі програми можуть значно покращити підтримку та доступність для широкого кола людей.

Отже, у даному підрозділі було вивчено та описано проблеми та необхідність розробки програмного застосування.

1.2. Порівняльний аналіз аналогів

Для аналізу стану програмних забезпечень, які надають підтримку, необхідно дослідити їхні основні риси та переваги. Однією з ключових переваг таких платформ є можливість отримати підтримку у будь-який зручний для користувача час. До числа найбільш популярних платформ для отримання інформації про фізичне і ментальне здоров’я можна віднести:

– BetterHelp;
– Calm;
– MyFitnessPal.

BetterHelp (рис. 1.1) — це платформа для онлайн-терапії, яка дозволяє легко та доступно отримати психологічну підтримку в Інтернеті [2]. BetterHelp, заснована в 2013 році, зараз є однією з найбільших і найвідоміших платформ онлайн-терапії у світі.

Користувачі BetterHelp можуть отримати психологічну допомогу з таких проблем, як тривожність, депресія, стрес, міжособистісні відносини або проблеми самопізнання. Коли вони зареєструються на платформі, вони мають можливість вибрати сертифікованих консультантів і психотерапевтів відповідно до їхніх потреб і бажань. Онлайн-терапія за допомогою текстових повідомлень, відеоконференцій і аудіовикликів є частиною послуг BetterHelp. Забезпечуючи психологічну підтримку у зручний для них час і місце, клієнти можуть обирати спосіб спілкування з терапевтом, який вони вважають зручним для себе. Дана платформа гарантує можливість отримати психологічну підтримку в будь-який

час і в зручному місці для клієнта. Особливо це стосується тих, хто має обмежені можливості відвідати психотерапевта особисто.

Основними мінусами даного вебзастосунку є висока вартість, тому що Psychology Today має комерційний аспект, оскільки рекламні матеріали і оголошення відображаються на сайті, та неможливість особистого контакту з спеціалістами.

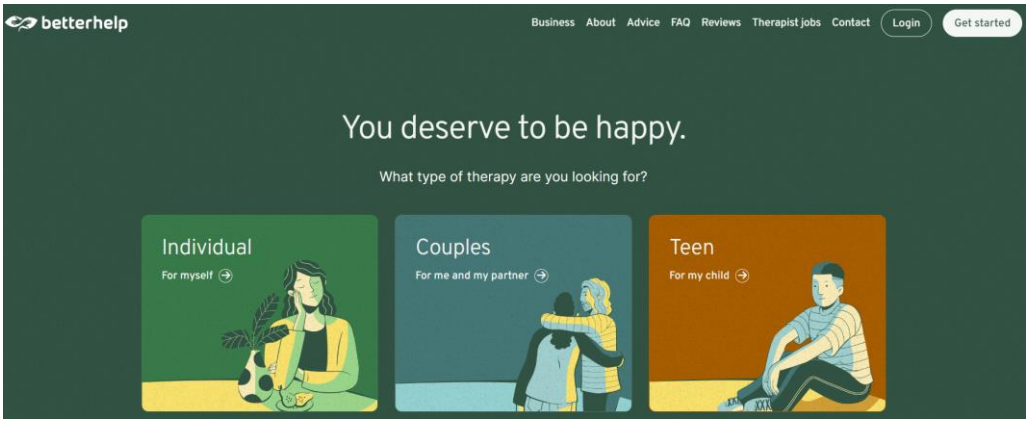


Рисунок 1.1 – Інтерфейс програмного забезпечення BetterHelp

Calm (рис. 1.2) - це вебзастосунок, який надає доступ до різноманітних ресурсів для психічного здоров'я, таких як медитація, релаксація та сон [3]. Вебверсія Calm пропонує користувачам можливість отримувати доступ до медитаційних практик, звукових пейзажів та програм для поліпшення сну прямо через веббраузер. Зручний доступ та багатий вибір контенту є головними плюсами даного програмного забезпечення, тому що він надає доступ до своїх послуг та розслабляючих інстрментів на всіх пристроях. Також він надає широкий вибір музики для медитації, релаксації та поліпшення сну.

Головними недоліками є платна підписка, що зменшує його ефективність для неплатоспроможної аудиторії і даний застосунок може бути менш значущим, ніж очікувалося, і користувачі можуть не відчути змін у своєму ментальному стані.

Рисунок 1.2 – Інтерфейс програмного забезпечення Calm

MyFitnessPal (рис. 1.3) – це інтерактивний інструмент для ведення обліку харчування та фізичної активності [4]. Він дозволяє користувачам вести детальний щоденний журнал споживаних продуктів, калорій та фізичних навантажень, а також створювати персоналізовані плани харчування та тренувань для досягнення фітнес-цілей. Серед головних переваг можна виділити простоту використання, тому що інтерфейс даного застосунку дозволяє зручно вводити та відстежувати харчові та фізичні дані користувача. Також можна виділити широкий вибір як ще одним плюсом, додаток має велику базу продуктів харчування, то користувачам легко знайти необхідні продукти та додати їх до щоденного журналу.

Серед недоліків можна виділити, що дана програма може містити помилки калорій або інших поживних речовин, що може призвести до неправильного розрахунку споживаних калорій. Також доступна преміум-версія, яка надає доступ до більших можливостей, але це не є ефективним інструментом для всіх користувачів.

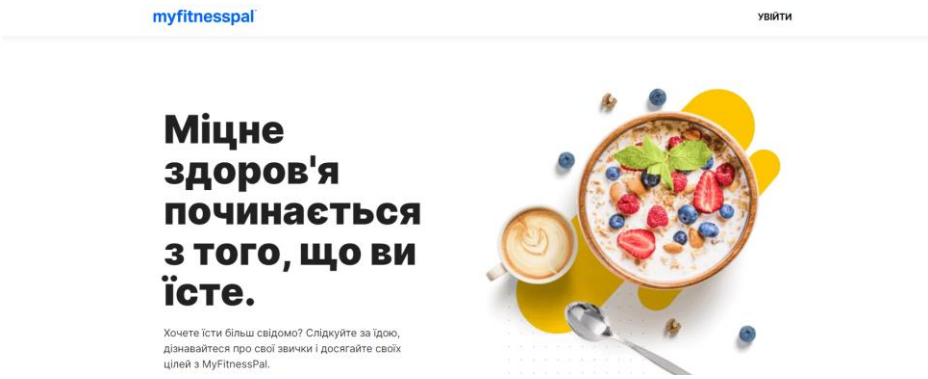


Рисунок 1.3 – Інтерфейс платформи MyFitnessPal

Отже, було здійснено порівняльний аналіз наведених платформ та розробленої конкуретної платформи HealthyLife (табл.1.1).

Таблиця 1.1 – Порівняльний аналіз аналогів

Критерій	BetterHelp	Calm	MyFitnessPal	HealthyLife
Можливість запису на консультацію	+	-	+	+
Наявність фізичної і ментальної допомоги	-	-	-	+
Наявність калькулятора BMI	-	-	+	+
Безкоштовне користування	-	+	-	+
Зручний інтерфейс	-	+	+	+
Сума	1	2	3	5

Таким чином, результати аналізу показують, що розроблений програмний засіб має більшу функціональність, ніж його аналоги.

1.3 Аналіз методів розв’язання задачі

Розробка програмного забезпечення може відбуватися кількома шляхами, зокрема, через використання готових рішень, індивідуальну розробку або комбіновані методи. Готові рішення зазвичай знаходять своїх прихильників через швидкість впровадження без значних витрат на програмування. Однак, вони можуть бути обмеженими у функціональності та не завжди відповідати

потребам. Далі було розглянуто методи та інструменти, які можуть бути використані для розв'язання даної задачі.

Одним із способів вирішення є вебзастосунок, який допоможе користувачам у пошуку потрібної їм інформації. Такий застосунок може мати такі функції: запис на консультацію, BMI-калькулятор, пошук індивідуальної інформації та чат-бот.

Інші компанії обирають індивідуальну розробку платформи, що дозволяє отримати повний контроль над функціоналом та інтеграцією з іншими системами. Мова йде про мобільний додаток, який забезпечить користувачам зручний та доступний інтерфейс для взаємодії з різними сервісами та функціями. Проте, цей підхід вимагає значних фінансових та часових вкладень.

Після перегляду та аналізу 2 методів було обрано вебзастосунок для вирішення даної проблеми. Вибір віддається даному методу через його простоту використання, особливо для осіб зі специфічним статусом, які не хочуть завантажувати окремий програмний продукт.

Основною метою було розробити програмне забезпечення для доступу до корисної та вірної інформації щодо підтримки фізичного та ментального здоров’я.

1.4. Постановка задач розробки програмного забезпечення

Після аналізу особливостей створення програмного забезпечення для підтримки фізичного та ментального здоров’я людини було встановлено низку завдань для розробки програми:

1. розробити метод і модель системи для створення сучасної онлайн платформи для підтримки здоров’я;
2. розробити оригінальний дизайн і візуальне оформлення програмного забезпечення, яке відповідатиме концепції "HealhyLife" і буде сприяти затишній атмосфері для користувачів;
3. створити користувацький інтерфейс, який буде легким у використанні та дозволить зручно взаємодіяти з системою;

4. здійснити програмну реалізацію системи "HealthyLife" для підтримки фізичного та ментального здоров'я, включаючи розробку необхідних функціональних модулів;
5. здійснити перевірку функціональності та надійності створеного програмного забезпечення шляхом проведення тестування.

Отже, створення програмного модулю для підтримки фізичного та ментального здоров'я включає в себе визначення оптимального підходу до побудови платформи, розробку її структури та архітектури, створення панелі користувача, модулів, а також проведення тестування для забезпечення якості програмного забезпечення.

1.5. Висновки

У першому розділі розглядаються поточні тенденції програмного забезпечення для системи підтримки фізичного та ментального здоров'я. Було проведено аналіз існуючих аналогів, а також їх порівняння з розроблюваним програмним забезпеченням. Даний аналіз дав підставу для вивчення існуючих методів вирішення поставленої задачі та обґрунтував необхідність створення балакалаврської дипломної роботи. Таким чином, було визначено основні завдання, необхідні для розробки програмного продукту.

2. РОЗРОБКА МЕТОДУ, СТРУКТУРИ, МОДЕЛІ ТА АЛГОРИТМІВ РОБОТИ ПРОГРАМНОГО ПРОДУКТУ

2.1 Аналіз даних

Здоров'я, як фізичне, так і ментальне, є найбільшою цінністю для кожної людини. В умовах сучасного світу, програмні забезпечення стають важливим засобом для надання не лише інформації, але й підтримки в цих сферах.

В межах бакалаврської дипломної роботи з підтримки фізичного та ментального здоров'я, основним джерелом інформації є досвідчені фахівці та професіонали з цих галузей. Вони створюють матеріали, які не лише надають корисну інформацію, але й сприяють підтримці та підвищенню обізнаності користувачів.

Графічний дизайн інтерфейсу платформи надає зручний доступ до цієї інформації. Він не лише вказує користувачам на наявність різних матеріалів, але й допомагає зорієнтуватися в них.

Щоб забезпечити максимальний комфорт користувачів, дана платформа також використовує інтерактивні елементи, які сприяють легкості використання і взаємодії з сервісом. Наприклад, було розроблено інтегрований калькулятор BMI, який дозволяє користувачам швидко дізнатися, чи нормальна у них маса тіла відносно до їх росту. Також було зпроєктовано чат-бот, який допомагатиме користувачам самостійно дізнаватись корисну інформацію.

Відсутність можливості для користувачів додавати матеріали на платформу дозволяє забезпечити високий рівень достовірності та якості інформації, що надається, забезпечуючи таким чином високий стандарт сервісу.

2.2 Метод збору даних для підтримки фізичного та ментального здоров'я

У сучасному світі, де стрес, нездорове харчування, недостатній рух та інші фактори негативно впливають на здоров’я людини, важливо мати ефективні інструменти, щоб підтримувати своє фізичне та душевне благополуччя.

HealthyLife використовує цілеспрямований метод збору даних, щоб надати користувачам індивідуальні поради та рекомендації, спрямовані на поліпшення їхнього стану здоров’я.

Дана стратегія збору даних включає використання широкого спектру інструментів, щоб забезпечити широке охоплення різних аспектів здоров’я. Предметом діяльності застосунку є надання користувачам доступу до корисних і навчальних статей про фізичне та ментальне здоров’я. Користувачі можуть краще зрозуміти ідеї здорового способу життя в цих статтях, які також розповідають про різні методи підтримки здоров’я.

Крім того, програмне забезпечення пропонує калькулятор BMI, який дозволяє користувачам визначити свій індекс маси тіла та отримати поради щодо того, як зменшити вагу. Ця програма сприяє впровадженню здорового способу життя та є корисним інструментом для визначення фізичного стану.

Також користувачам надається можливість звертатися до чат-бота програмного продукту, щоб отримати консультації щодо фізичного та ментального здоров’я. Даний бот збирає інформацію про потреби, звички та стан здоров’я користувачів у режимі реального часу. Дані використовуються для надання індивідуальних порад і рекомендацій.

Рекомендації та поради розробляються на основі аналізу даних. На основі цього аналізу спеціалісти вебзастосунку створюють індивідуальні програми здоров’я, враховуючи потреби та цілі кожного користувача.

2.3 Розробка структури графічного інтерфейсу програмного забезпечення

Створення структури програмного забезпечення є важливою частиною створення системи для підтримки фізчиного та ментального здоров’я. Хороша структура сайту дозволяє користувачам легко переміщатися між різними елементами та взаємодіяти з важливим контентом.

Початковим кроком у розробці структури є оцінка потреб користувачів. Цей аналіз допомагає визначити основні функції та можливості програмного застосунку для підтримки фізичного та психічного здоров’я користувачів.

Побудова ієрархії сторінок також важлива. Щоб створити логічну структуру сайту, схожі функції та інформація групуються разом. Сайт добре структурований за допомогою основної сторінки, яка містить інформацію про «Найбільш рейтингові статті», «Про нас», «Чому обирають нас?» та «Наші спеціалісти». Також застосунок має кнопки навігації «Статті», «Калькулятор ВМІ» та «Чат-бот», які допомагають легше організувати сторінки.

Для того, щоб програмне забезпечення було простим у використанні, важлива ефективна навігація. Користувачі можуть швидко знайти потрібну інформацію завдяки використанню елементів, які легко зрозуміти та легко орієнтуватися, таких як кнопки, меню та посилання.

Основні типи структур, які використовуються в вебзастосунках, включають [5]:

- лінійна структура: дозволяє кожній сторінці або елементу мати прямий шлях до іншої сторінки або елемента. Це часто використовується в простих вебсайтах або блогах, на яких відвідувачі переходять з однієї сторінки на іншу в логічному порядку;
- структура дерева: схожа на структуру каталогу файлів комп’ютера. Кожна сторінка складається з численних підсторінок, які утворюють складну ієрархічну структуру. Це корисно для об’єднання великих обсягів даних, як-от електронні бібліотеки або веб-портали;
- мережева структура, також відома як мережева структура: полягає в тому, що кожен елемент, як правило, картинка або текст, розташований у відношенні до інших елементів на сторінці. Ця структура добре працює для візуальної організації даних на головних сторінках новинних сайтів або в інтернет-магазинах;

- структура з кількома рівнями: комбінація різних типів структур, яка використовується для складних веб-застосунків. може включати елементи лінійної або деревовидної форми.

Було створено структуру сторінки програмного забезпечення для системи підтримки фізичного та ментального здоров’я людини (рис. 2.1).

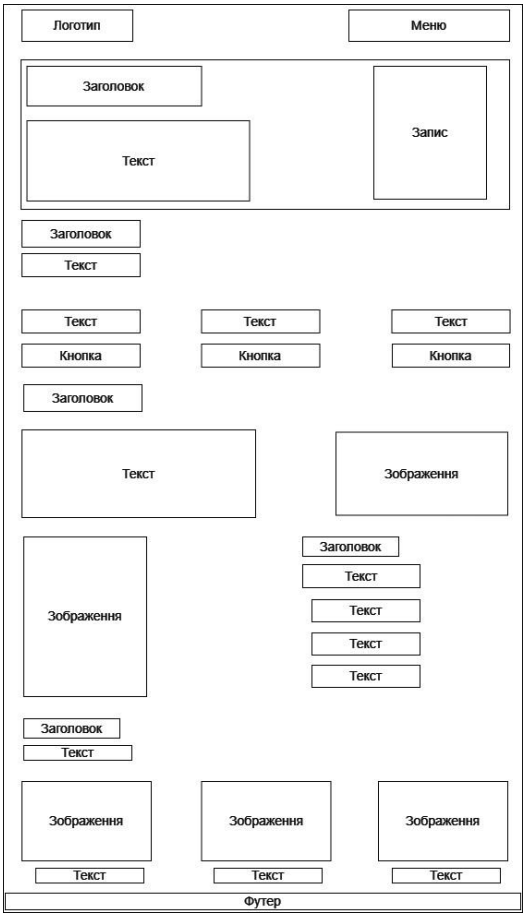


Рисунок 2.1 – Структура сторінки програмного забезпечення

Структура програмного забезпечення для підтримки фізичного та ментального здоров’я дозволяє користувачеві швидко зрозуміти основні функції сайту. Відвідувач першим чим стикається з сайтом, це його дизайн, який дозволяє йому бачити його компоненти та графічний інтерфейс. Люди дивляться на вигляд сайту, щоб визначити його зручність. Візуальне оформлення будь-якого вебзастосунку є творчим процесом. На цьому етапі всі частини дизайну

програмного продукту розробляються відповідно до стилю подачі інформації та загальної концепції. Після цього дизайн перевіряється, вдосконалюється та тестується.

Основною метою інтерфейсів є забезпечення максимального комфорту для користувачів під час перегляду вебсайтів і спрощення їхнього життя. Розробка інтерфейсів повинна базуватися на кількох принципах:

- Ігнорування надто нових ідей в структурі
- Розуміння звичок та цілей корисутвачів
- Утримання від складної логіки
- Дотримування єдиного стилю дизайну

Підбір кольорової гами також є одним з головних критеріїв у створенні дизайну інтерфейсу [6]. Існує кілька методів для підбору кольорової гами для вебсайту або будь-якого іншого проекту. Ось декілька з них:

1. Використання колірної теорії:
    - колірна колісниця: для вибору гармонійних кольорів використовується колірне колесо. Наприклад, близькі кольори на колірному колесі часто виглядають добре разом;
    - тріада відтінків: потрібно обрати три кольори та розташувати їх рівномірно на колірному колесі. Це сприятиме створенню жвавої та збалансованої палітри.
2. Використання зразків та інспірації:
    - природа: натхнення можна отримати з кольорів природи, таких як квіти, ліси та водойми;
    - блоги та портали, присвячені дизайну: звертати увагу на дизайнерські ресурси, щоб отримати оновлення щодо тенденцій кольору та натхнення від інших проєктів.
3. Врахування брендової ідентичності: якщо це дизайн логотипу компанії або бренду, потрібно враховувати стиль і кольори.

    Загальні характеристики кольору включають:

    - червоний: символізує силу, енергію та палкість;

- оранжевий: означає ентузіазм, теплоту та радість;
- жовтий: колір щастя, оптимізму та радості;
- зелений: асоціюється зі спокоєм, природою, відновленням і стабільності;
- фіолетовий: асоціюється з загадковістю, розумом і розширенням свідомості.

Було обрано зелений колір, як головний, тому що саме він найкраще підходить для сфери безпеки і підтримки здоров'я. Приклад вигляду інтерфейсу програмного забезпечення наведено на рисунку 2.2.

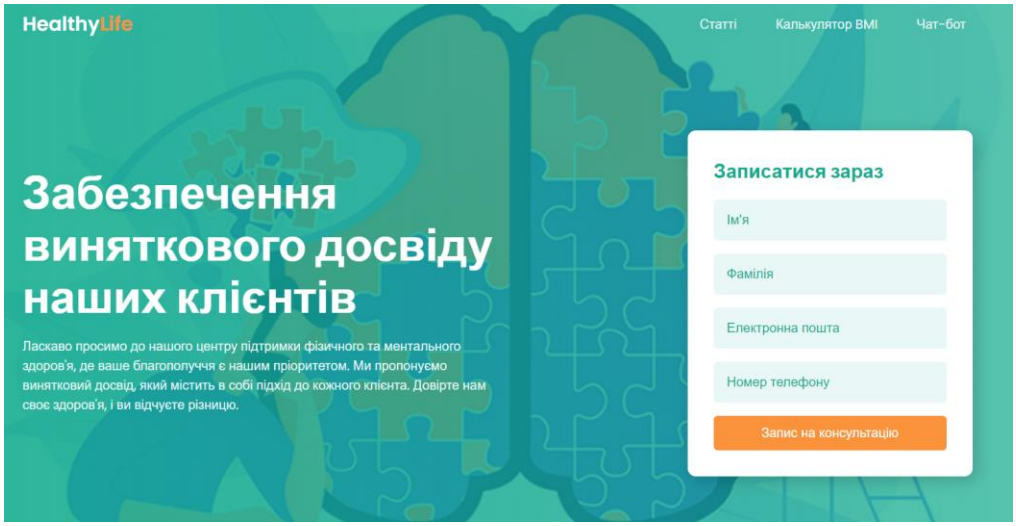


Рисунок 2.2 – Кольорова гамма сайту

Таким чином, була створена структура інтерфейсу сторінки програмного забезпечення для системи підтримки фізичного та ментального здоров’я.

2.4 Розробка моделі системи програмного забезпечення

Для кращого розуміння функціонування програмного забезпечення для системи підтримки фізичного та ментального здоров’я було вирішено створити модель роботи системи за допомогою UML діаграми діяльності (рисунок 2.3). Згідно з цією діаграмою, першим кроком для користувача є запис на консультацію, але це за потреби, якщо потрібна персоналізована допомога.

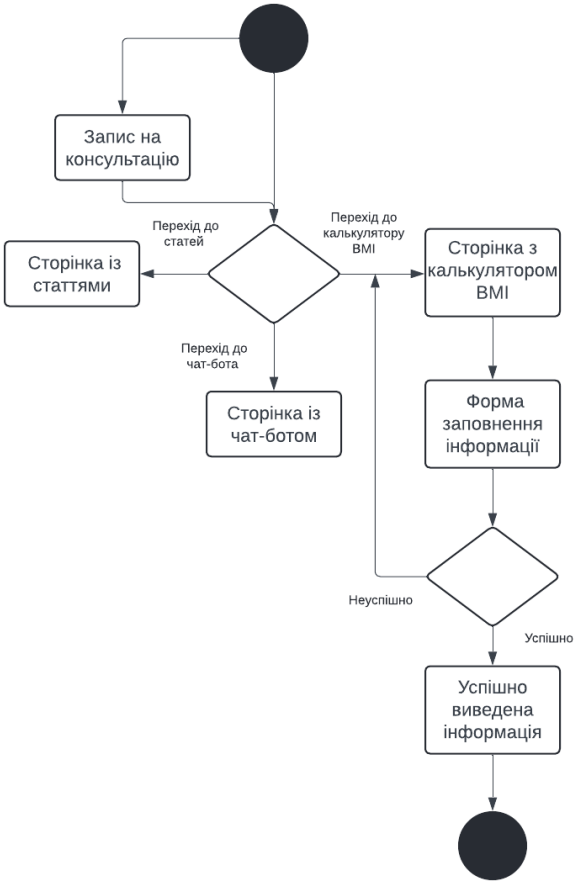


Рисунок 2.3 – Модель системи у вигляді діаграми діяльності

Далі користувач бачить верхню панель меню, де можна обрати перегляд статей, калькулятор BMI або чат-бота. У разі вибору перегляду статей, користувача переносить на нову сторінку, де можна ознайомитися з новітньою інформацією для збереження здоров'я. Якщо обрано калькулятор, користувач переадресовується на сторінку з формою BMI, де можна внести необхідні дані для самостійного ознайомлення зі своїм фізичним станом.

Фінальним етапом є перехід до чат-бота. Даний бот оснащений сучасної інформацією, яка допоможе зберегти людям фізичне та ментальне благополуччя.

Таким чином, була розглянута модель системи програмного забезпечення, яка показує загальну роботу програми для полегшеного розуміння робочих процесів.

2.5 Розробка алгоритмів роботи системи

Розробка алгоритмів, які пояснюють, як працює програмний продукт, покращують розуміння його роботи. Програмне забезпечення складається з двох основних логічних блоків: самостійне опрацювання з наданими інструментами і інформацією та запис користувачів на консультацію.

Алгоритм — це набір чітко визначених процедур або інструкцій, які описують послідовність дій, необхідних для виконання конкретної задачі або вирішення проблеми [7]. Можна представити алгоритм у вигляді етапів, які показують, як обробляти певні вхідні дані, щоб досягти бажаного результату.

Основні характеристики алгоритмів:

- Ефективність: термін, який описує, наскільки швидко або ефективно алгоритм працює з великою кількістю даних або завдань.
- Точність: показник того, наскільки надійно алгоритм виконує завдання або генерує необхідні результати.
- Простота: алгоритми простіші для розуміння та використання. Зазвичай прості алгоритми більш масштабовані та мають менше шансів на помилки.
- Масштабованість: для того, щоб алгоритм міг працювати з різними кількостями даних без значного збільшення часу виконання, важливо, щоб він був легко масштабований.
- Оптимальність: властивість алгоритму, яка дозволяє йому отримати найкращий можливий результат за короткий час або ресурси.

Алгоритм запису на консультацію відбувається по простому алгоритму. Користувач вписує свої персональні дані: ім'я, фамілію, електронну пошту та телефон. У разі неправильного вводу електронної пошти система сповіщає, що потрібно її виправити.

Після внесення всіх коректних даних користувач натискає кнопку запису та отримує сповіщення про успішне надсилання листа. Протягом однієї хвилини клієнт отримує на свою пошту лист з необхідним повідомленням. Схема алгоритму подана на рисунку 2.4.

Рисунок 2.4 – Схема алгоритму запису на консультацію

Алгоритм калькулятору BMI було впроваджено для самостійного опрацювання свого фізичного стану [8]. Користувач вписує свій зріст та вагу, потім натискає на кнопку розрахування BMI і йому виводиться його індекс маси тіла. Також доступна таблиця з інтерпретацією чисел, щоб користувач зрозумів у якому він фізичному стані. Схема алгоритму калькулятору BMI показана на рисунку 2.5.

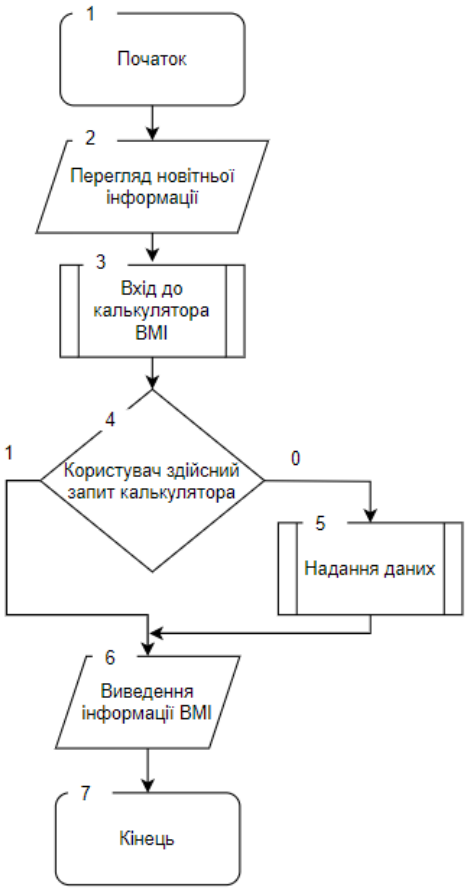


Рисунок 2.5 – Схема алгоритму калькулятору BMI

Таким чином, були розлянуті алгоритми роботи модулів програмного забезпечення для системи підтримки фізичного та ментального здоров'я.

2.6 Висновки

У другому розділі дослідження розглядаються різні способи, якими користувачі можуть взаємодіяти з програмою «HealthyLife», яка спрямована на підтримку фізичного та психічного здоров’я. Для розробки майбутньої графічної інтерфейсної системи обслуговування клієнтів використано прототип типового програмного продукту для аналізу структури. Основна мета полягала в тому, щоб програма була простою та зручною для користувача. Крім того, було розроблено метод збору даних, модель системи та основні алгоритми програмного забезпечення.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Варіантний аналіз і обґрунтування вибору засобів для реалізації програмного засобу

При розробці програмного продукту для системи підтримки фізичного та ментального здоров’я основною точкою відправлення є вибір мови програмування. Це визначає можливості для реалізації функціоналу та використання вебтехнологій та інструментів розробки. Вже на етапі написання коду важливо чітко визначити, які функції необхідно втілити та якими мають бути вимоги до програми.

Програмне забезпечення розробляється за допомогою 2 частин: клієнтська та серверна. Клієнтські та серверні компоненти вебзастосунку відповідають за різні аспекти його роботи:

1. Клієнтська частина:
   - взаємодія з користувачем: Клієнтська частина надає користувачеві інтерфейс для взаємодії з веб-застосунком [9]. Це охоплює всі частини веб-сторінки, з якими користувач взаємодіє;
   - відображення даних: Відображення даних на вебсторінці контролюється клієнтською частиною відповідно до дій користувача та даних, отриманих від сервера даних.
2. Серверна частина:
   - обробка запитів: серверна частина обробляє запити [10]. Це може включати обробку форм, перевірку доступу, обробку запитів на отримання або збереження даних тощо;
   - збереження та доступ до даних: дані, необхідні для вебзастосунку, зберігаються на сервері, але при необхідності клієнтська частина може отримати до них доступ;
   - виконання бізнес-логіки: виконання бізнес-логіки також належить до серверної частини програмного забезпечення. Це містить всі

необхідні вказівки, щоб програмний продукт правильно поводився в різних ситуаціях.

Для побудови клієнтської частини програмного продукту використовувались мови програмування HTML, CSS та JavaScript.

Вебсторінки та додатки створюються за допомогою мови розмітки HTML. Його можна використовувати для групування вебконтенту в різні форми, такі як зображення, відео, тексти, посилання та інші елементи. Дана мова включає структуру та вміст сторінки користувача, тому веббраузери використовують його для відображення сторінки.

HTML є основою для створення динамічних вебсторінок, які взаємодіють і змінюють вигляд залежно від того, як їх використовує користувач [11]. Для цього зазвичай використовується програмне забезпечення, таке як JavaScript і CSS. Мова розмітки HTML складається з 2 частин: header (інформація про документ) та body (основний зміст).

CSS — це мова стилів, яка визначає, як виглядають і форматуються елементи програмного продукту [12]. Вона дозволяє розробникам програм і вебсторінок керувати тим, як виглядають компоненти, такі як текст, зображення, посилання та інші компоненти.

Селектор, властивості та значення є стандартами стилів CSS. Селекціонер визначає елемент вебсторінки, до якого буде застосовано правило стилю. Незважаючи на те, що властивість визначає, який аспект елементу буде змінюватися, значення визначає, на що зміниться властивість.

Крім того, CSS дозволяє розробникам використовувати складні візуальні ефекти, такі як анімації та переходи, щоб покращити вигляд вебсторінки. Також можна створювати привабливі, зручні та легко використовувані сторінки за допомогою CSS, який є важливою частиною веброзробки.

Крім того, CSS має такі підкатегорії:

- стиль Inline CSS — стиль, який додається безпосередньо до елементу HTML за допомогою атрибуту стилю;

- стиль внутрішнього CSS — це стиль, який додається до документа HTML за допомогою тегу "стиль";
- зовнішній стиль CSS — стиль, який додається до HTML-документу та посилається на зовнішній файл CSS.

JavaScript (JS) — це мова програмування високого рівня, яка використовується для створення елементів, які є інтерактивними на програмних продуктах. Вебдизайнери можуть використовувати динамічні ефекти, анімацію, обчислення та інші можливості, які вона пропонує [13].

Замість компіляції коду перед виконанням JavaScript інтерпретується веббраузером або Node.js. Це дозволяє швидко та просто додати додаткові функції до програмних забезпечень, не компіляючи код. JavaScript використовується для багатьох речей, наприклад:

- створення інтерактивних елементів, таких як кнопки, форми та меню;
- обробка даних і виконання обчислень на стороні клієнта;
- створення динамічних вебсторінок, які змінюються залежно від поведінки користувача;
- створення вебдодатків.

Види JS:

1. Vanilla JS: чистий JavaScript без фреймворків і бібліотек. До особливостей Vanilla JS належить:
   - простий і швидкий, і не потребує додаткового коду, розробники мають повний контроль над кодом і можливість його змінювати за потреби;
   - підходить для проектів малого та середнього розміру;
   - оскільки немає рівня абстракції, розробникам потрібно писати більше коду, щоб досягти того ж результату, що і при використанні бібліотек або фреймворків.

2. jQuery: дозволяє легко працювати з елементами HTML і виконувати різні завдання. Особливості jQuery:
    - бібліотека JavaScript, яка полегшує роботу з DOM та обробку подій, надає простий і простий у використанні API для вибору та маніпулювання елементами HTML;
    - підтримує анімацію, ефекти та запити AJAX;
    - пропонує широкий спектр розширень і плагінів;
    - дана бібліотека підходить як для великих, так і для малих проектів;
    - можна використовувати для швидкої розробки та створення прототипів, оскільки має уніфікований API та усуває багато невідповідностей браузерів.
3. React: бібліотека JS, яка дозволяє створювати інтерактивні користувачеві інтерфейси для творців вебсторінок. До основних React особливостей належить:
    - бібліотека JavaScript, яка використовується як шар перегляду під час створення користувацьких інтерфейсів;
    - для підвищення продуктивності та оптимізації рендерингу використовує віртуальний DOM (полегшене представлення в пам’яті реального DOM);
    - компоненти легко компонувати разом і можна повторно використовувати;
    - завдяки підтримці рендерингу на стороні сервера можна створювати складні програми, керовані даними;
    - він має велику та активну спільноту через багато сторонніх бібліотек і інструментів;
    - вимагає хорошого розуміння JavaScript і функціонального програмування;
    - розробка складних і масштабованих програм є однією з його можливостей.

4. Angular: фреймворк, який дає змогу створювати складні програмні продукти. До особливостей Angular належить:
    - JavaScript-фреймворк, призначений для створення складних додатків з керуванням даними;
    - використовує модульну архітектуру та пропонує широкий спектр функцій для створення додатків;
    - підтримує двостороннє зв’язування даних, тому інтерфейс користувача автоматично оновлюється, коли змінюються дані;
    - має велику та активну спільноту, яка складається з великої кількості інших бібліотек і інструментів;
    - може бути використаний для розробки складних програм.
5. Vue.js: JS фреймворк для створення інтерактивних вебзастосунків. Особливості Vue.js:
    - прогресивний та гручний фреймворк JavaScript, призначений для створення інтерфейсів користувача та додатків;
    - використовує компонентну архітектуру та надає широкий спектр функцій для створення додатків;
    - потужний маршрутизатор, мова шаблонів і ін’єкція залежностей включені;
    - підтримує двостороннє зв’язування даних і надає потужний набір інструментів для створення складних додатків;
    - спільнота постійно розвивається, а в ньому й багато сторонніх бібліотек і інструментів;
    - потребує глибокого розуміння концепцій Vue та синтаксису, а також JavaScript, HTML і CSS;
    - хороший інструмент для розробки складних і масштабованих програм.

Результати порівняння розглянутих версій для написання клієнтської частини за обраними критеріями наведено в таблиці 3.1.

Таблиця 3.1 – Порівняння версій для написання клієнтської частини

Критерій	Angular	React	Vanilla
Вага проєкту	0.5	0.5	1
Простота коду	0	0	1
Встановлення	0,5	0.5	1
Спільнота	0.5	1	0.5
Функціональність	0,5	0,5	0.5
Підсумковий результат	2	2.5	4

Як показано в таблиці 3.1, версія Vanilla JS задовольняє всі вимоги, які можуть виникнути під час розробки програмного забезпечення.

Створення серверної частини програмного забезпечення може відбуватися за допомогою багатьох мов програмування, наприклад Java, JavaScript, Python, PHP багато інших. Мова програмування, яка використовується на серверній частині, залежить від потреб проєкту, кількості доступних ресурсів і специфіки розробки. Проаналізуємо PHP, JavaScript та Java — три основні мови програмування, які використовуються для написання серверної частини вебпродуктів.

PHP — це скриптова мова, призначена для створення сторінок HTML на вебзастосунках. Крім Java,.NET, JavaScript, Python і Ruby, PHP є однією з найпоширеніших мов веброзробки. Більшість провайдерів хостингу дозволяють використовувати PHP [14]. PHP є програмним забезпеченням, яке є відкритим кодом.

Особливості мови програмування PHP:

- використовується для моніторингу сесій, управління базами даних, управління динамічним вмістом і навіть створення повноцінних сайтів електронної комерції. Оскільки PHP має інтерфейс командного рядка, він також може використовуватися для створення автономних настільних додатків;
- код PHP виконується на сервері, який генерує HTML для клієнта. Клієнт отримує результати, які виконуються за допомогою цього сценарію, але він не знає базового коду;

- може збирати дані форми, надсилати та отримувати файли cookie; створювати, відкривати, читати, записувати, видаляти та закривати файли на сервері; створювати, видаляти, змінювати та шифрувати дані в базі даних; контролювати доступ користувачів; і шифрувати дані.

Python - це мова програмування, яка славиться своєю лаконічністю та легкістю розуміння коду. Вона заснована на інтерпретації та відрізняється від компіляції, оскільки програми на ній не перетворюються безпосередньо у машинний код. Серед основних плюсів даної мови програмування можна виділити:

- Python — це набір простих правил, які роблять код зрозумілим навіть для початківців;
- застосовується в найрізноманітніших галузях, включаючи веброзробку, наукові дослідження, штучний інтелект та багато іншого;
- спроможний працювати на різних платформах, включаючи Windows, macOS і різні дистрибутиви Linux, що робить його зручним для розробників;
- має велику та жваву спільноту розробників, яка активно підтримує мову, надає допомогу початківцям та сприяє розвитку різноманітних бібліотек та фреймворків.

Програмне забезпечення Java було розроблено Sun Microsystems (тепер Oracle Corporation) у 1995 році [15]. Вона є багатопотоковою, платформонезалежною та об’єктно-орієнтованою мовою програмування.

Деякі особливості Java:

- підтримує створення об’єктів, які можуть містити методи та властивості;
- дозволяє працювати з кількома потоками одночасно;
- працює на багатьох платформах, таких як Windows, Linux і MacOS;
- розробники можуть створювати складні програми та застосунки за допомогою великої кількості фреймворків і бібліотек Java;

- має сильну типізацію, що означає, що вона перевіряє типи даних під час компіляції, щоб запобігти помилкам.

Результати порівняння розглянутих мов програмування для написання серверної частини за обраними критеріями наведено в таблиці 3.2.

Таблиця 3.2 – Порівняння мов програмування для написання серверної частини

Критерій	PHP	Python	Java
Продуктивність	0.5	0.5	1
Безпека	0.5	0.5	1
Зручне використання	1	0.5	0.5
Масштабованість	0.5	0.5	1
Підтримка спільноти	0.5	0.5	1
Платформна незалежність	0,5	1	1
Підсумковий результат	3,5	3.5	5.5

На основі таблиці 3.2 можна зробити висновок, що мова програмування Java є найбільш підходящою серед розглянутих варіантів для розробки серверної частини програмного забезпечення. Java — це мова, яка добре компілюється, особливо для великих і складних програм. Крім того, корпоративні програми люблять Java через її безпеку. Нарешті, Java є мовою, яку можна розширити, що дозволяє створювати великі та складні програми, які можуть обробляти велику кількість даних і підтримувати велику кількість користувачів.

Отже, проаналізувавши наявні на ринку мови програмування для написання сучасних динамічних програмних забезпечень було обрано HTML, CSS та JavaScript для клієнтської частини та мову програмування Java для серверної.

3.2 Вибір середовища розробки

Сучасний веброзвиток, який надзвичайно динамічний і різноманітний, часто залежить від вибору правильного середовища розробки. Швидкість, продуктивність і якість розробки є важливими для розробки клієнтської частини вебдодатку.

Під час розмови про кожне середовище розробки потрібно визначити його основні функції, переваги та поради щодо використання їх для розробки клієнтської частини програмного забезпечення. Вдалість проєкту, його стійкість і ефективність залежить від правильного середовища розробки. Серед найбільш поширених середовищ для клієнтської частини програмого продукту можна виділити Atom, Sublime Text та Visual Studio Code.

Atom — безкоштовний редактор коду, який знаходиться на GitHub. Він працює з багатьма мовами програмування та технологіями, такими як HTML, CSS, JavaScript, Python та Ruby. Atom прагне зробити середовище для написання коду зручним для розробників.

Його головні особливості:

- розширюваність: завдяки великій кількості пакетів і розширень дане програмне середовище може виконувати більше завдань;
- співпраця з GitHub: чудово працює з платформою GitHub, що полегшує зберігання та спільну розробку проектів;
- командна робота: завдяки вбудованій підтримці Git Atom забезпечує спільну роботу над проєктами, дозволяючи створювати гілки, створювати коміти та відслідковувати зміни прямо з редактора;
- налаштування: надає вам можливість змінювати редактор, як ви хочете. Ви можете змінити тему, шрифт і вибрати плагіни та розширення, які вам потрібні.

Sublime Text широко використовується розробниками та програмістами по всьому світу, оскільки він простий і ефективний редактор коду. Його швидкість роботи, широкий асортимент функцій і здатність адаптуватися до потреб користувача сприяють його популярності.

Основні властивості даного програмного середовища:

- легкість і швидкість: редактор працює швидко та не потребує багато системних ресурсів. Це дозволяє завершувати значні проекти швидко та ефективно;

- множинні види редагування: підтримує кілька видів редагування, включаючи автоматичне додавання закриваючих тегів для HTML, швидке копіювання та вставка рядків і редагування в кількох місцях одночасно;
- підтримка багатьох мов програмування: підтримує широкий спектр мов програмування, для більшості з яких є підсвічування синтаксису. Це полегшує читання та написання коду для різних платформ і технологій;
- швидкі клавіатурні комбінації: редактор пропонує безліч швидких клавіатурних комбінацій, які дозволяють виконувати різні завдання без використання миші. Це дозволяє розробникам працювати швидше та ефективніше;
- підтримка проєктів: об’єднує файли в одній сесії, що полегшує створення та керування проектами. Це робить планування та завершення великих проектів простішим.

Microsoft Visual Studio Code — це потужний, безкоштовний редактор коду. Зазвичай розробники використовують його для роботи з різними мовами програмування та технологіями [16].

Ось деякі особливості VS Code:

- має простий інтерфейс, тому навіть новачки можуть швидко розпочати роботу з редактором;
- для розширення функціональності редактор пропонує багато розширень. Підтримку мов програмування можна покращити, можна додати нові функції та інтегрувати розширення з різними сервісами та API;
- підтримує системи контролю версій, зокрема Git, що дозволяє легко взаємодіяти з репозиторіями, виконувати операції комітів, створювати гілки та виконувати багато інших завдань;

- завдяки таким інструментам розробки, як інтегрована консоль, відладчик і переглядач документів, редактори можуть спростити процес розробки та налагодження програмного забезпечення;
- працює на різних платформах, включаючи Windows, macOS і Linux, що дозволяє розробникам використовувати кілька операційних систем одночасно.

Результати порівняння розглянутих програмних середовищ для написання клієнтської частини за обраними критеріями наведено в таблиці 3.3.

Таблиця 3.3 – Порівняння програмних середовищ для написання клієнтської частини

Критерій	Atom	Sublime Text	VS Code
Легкість використання	0	0	1
Розширюваність	1	1	1
Швидкість	0	1	1
Інтеграція з Git	1	0.5	1
Вбудовані інструменти розробки	1	1	1
Підсвічування синтаксису та автодоповнення	1	1	1
Підсумковий результат	4	4.5	6

Можна зробити висновок, згідно таблиці 3.3, що VS Code найкраще підходить для вирішення нашої проблеми клієнтської частини.

Вибір найкращого середовища розробки сервера також важливий для успішного програмного забезпечення. Обробка запитів клієнтів, взаємодія з базою даних, забезпечення безпеки та ефективне функціонування програмних застосунків – усе це компетенції серверної частини. Але так як у бакалаврській роботі буде застосована мова програмування Java, то буде використано не програмні середовища, а саме потужні інструменти для виконання Java. З найбільш поширених додатків для виконання Java можна виділити Spring Framework, Apache Tomcat, а також хочу приділити увагу до Intelij IDEA.

Незважаючи на те, що Intellij IDEA не завжди розглядається як серверне середовище, це потужне інтегроване середовище розробки (IDE) для Java, яке широко використовується для розробки компонентів програмного забезпечення як для серверів, так і для клієнтів.

IDEA пропонує широкий спектр інструментів, які підтримують різноманітні фреймворки та технології, такі як Spring, Hibernate та Maven, для створення Java-проектів [17]. Крім того, можливості роботи з базами даних, версійного контролю, автоматичного завершення коду, відлагодження та профілювання програм були розширені завдяки IDEA.

З особливостей інтегрованого середовища розробки для Java можна виділити:

- працює з Java та кількома відомими мовами програмування, такими як Kotlin, Groovy, Scala, JavaScript, TypeScript, SQL, HTML і CSS. Це робить його ідеальним для великої кількості проектів;
- пропонує більше можливостей для автоматичного завершення коду, що полегшує роботу з складними або складними конструкціями коду та підвищує продуктивність розробників;
- щоб гарантувати якість і стабільність кодової бази, IDEA надає автоматичні перевірки стилю коду, інспекції коду, виявлення потенційних помилок і оптимізацій, а також інші інструменти аналізу коду;
- підтримує багато фреймворків і технологій, таких як Spring, Hibernate, Java EE, Android, Maven і Gradle, що полегшує розробку різноманітних додатків.

Spring Framework — популярний і потужний фреймворк для розробки програмного забезпечення на мові Java. Від простих вебдодатків до складних бізнес-систем, його гнучка та ефективна платформа дозволяє розробникам створювати різноманітні програми.

Основні властивості Spring Framework:

- пропонує механізм інверсії управління, який дозволяє відрізняти компоненти додатка один від одного та встановлювати зв’язки між ними через контейнер IoC. Це полегшує окреме тестування компонентів і керування залежностями;
- дозволяє реалізувати аспекти, щоб відокремити крос-відомості, такі як транзакції, журналювання, безпека, тощо, від основної бізнес-логіки додатка;
- надає підтримку для розробки різних типів додатків

Apache Tomcat — це контейнер сервлетів, який використовується для розгортання та запуску програмних забезпечень на Java-платформі. Він створює просте середовище для роботи з сайтами, які використовують Java та JavaServer Pages.

Основні властивості Apache Tomcat:

- підтримує стандарти Java Servlet і JavaServer Pages, що дозволяє розробникам розробляти динамічні вебпрограми;
- простий у використанні та конфігурований, і його файл конфігурації та інтерфейс керування роблять конфігурацію та керування сервером більш простими;
- популярний для робочих вебдодатків через свою надійність і продуктивність;
- підтримує багато надбудов і функцій, що дозволяє розробникам налаштовувати свої функції відповідно до своїх потреб;
- проєкт із відкритим кодом, наданий Apache Software Foundation і має активну спільноту користувачів і розробників, які допомагають створювати та підтримувати продукт.

Таблиця 3.3 містить результати порівняння розглянутих додатків для написання клієнтської частини за обраними критеріями.

Таблиця 3.4 – Порівняння додатків Java для серверної частини

Критерій	Spring Framework	Apache Tomcat	Intelij IDEA
Легкість використання	1	1	1
Функціональність	0,5	0,5	1
Підтримка розширень та додатків	0,5	1	1
Швидкодія	1	1	1
Гнучкість	0,5	0,5	1
Інтеграція з іншими інструментами	0.5	0.5	1
Підсумковий результат	4	4.5	6

За результатами з таблиці 3.4 можна зробити висновок, що засіб розробки Intelij IDEA найкраще підійде для вибраної цілі.

Отже, для реалізації програмного забезпечення HealthyLife було вибрано VS Code, як для розробки клієнтської частини та Intelij IDEA для серверної. Дані середовища забезпечують необхідні інструменти для створення, тестування та налагодження програмного продукту, що робить їх найкращими для розробки даного проєкту.

3.3 Розробка клієнтської частини програмного продукту для підтримки фізичного та ментального здоров’я

Розробка програмних застосунків - це захоплюючий та творчий процес, що вимагає дбайливого підходу до кожного етапу. Починаючи з HTML, будується основна структура додатку, створюючи його скелет, на якому буде ґрунтуватися весь функціонал. Використання семантичних тегів не лише полегшує розуміння структури сторінки для розробників, але й покращує SEO, роблячи вміст доступнішим для пошукових систем.

Проте, навіть найкраща структура не зможе зацікавити користувача без привабливого зовнішнього вигляду. CSS стає рятівником у цьому плані, дозволяючи користувачу надати програмному забезпеченню естетичний вигляд та забезпечити її зручність у використанні. Кольори, шрифти, розміри та розташування елементів - все це відбувається завдяки CSS.

Щоправда, щоб додаток був справжньою майстернею взаємодії та динаміки, не можна обійтися без JavaScript. Дана мова програмування дозволяє створювати динамічний контент, реалізовувати різноманітні можливості та надавати користувачам більше взаємодії з вебсторінкою.

В процесі створення програмного продукту варто пам'ятати, що зазвичай розробку розпочинають із верхньої частини, де розташовується заголовок і навігаційне меню. Це як вхідна брама у світ додатку, де логотип у верхньому лівому куті вітає користувача, а панель навігації відкриває двері до його можливостей. На рисунку 3.1 зображено шапку сторінки програмного забезпечення.

Рисунок 3.1 – Шапка програмного забезпечення

Під час створення додатку, особливу увагу слід приділяти його адаптивності – він має бути комфортним для користування на будь-яких пристроях та розмірах екрану. Паралельно із цим, важливо постійно перевіряти та покращувати безпеку, продуктивність та правильність роботи.

В меню програмного забезпечення знаходиться 3 кнопки-посилання, які будуть переміщати на вибрану тему користувачем. Вигляд посилання «Статті» знаходиться на рисунку 3.2. Дана сторінка показує актуальну інформацію для збереження фізичного та ментального здоров’я.

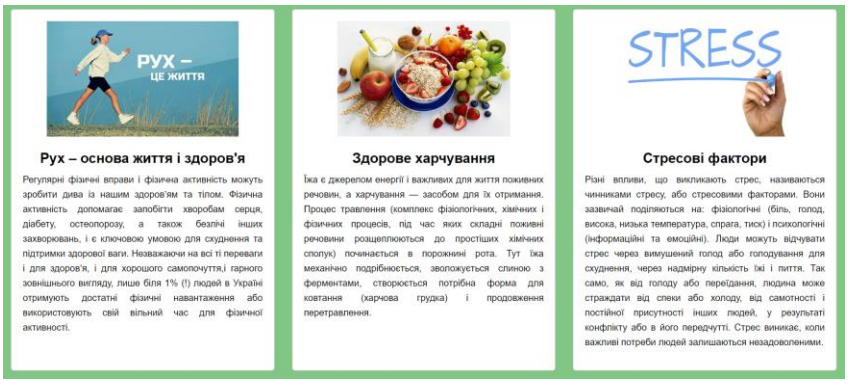
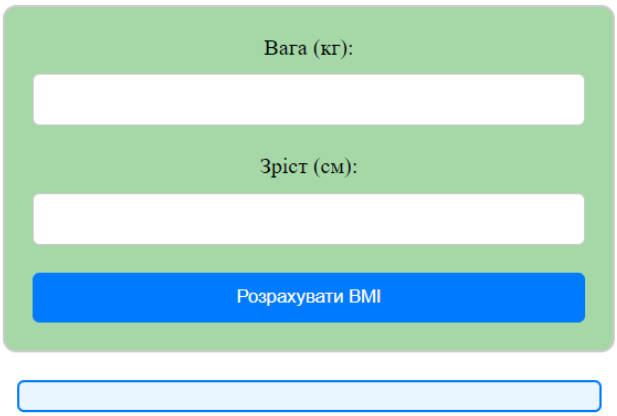


Рисунок 3.2 – Зображені статті головного меню

Наступним посиланням є «Калькулятор BMI». Це корисний інструмент, який допоможе користувачеві самостійно визначити індекс маси тіла (рис. 3.3). За допомогою даного калькулятора, користувач програмного забезпечення може швидко та зручно отримати важливу інформацію про його здоров'я та фізичну форму. В даній формі знаходиться 2 поля, які потрібно заповнити: вага та зріст. Користувач вписує свої дані і отримує важливі показники, які можуть бути незрозумілими, але на сторінці також присутня таблиця з інтерпретацією, яка допоможе зрозуміти рівень маси тіла



BMI	Інтерпретація
<18.5	Недостатня маса тіла
18.5 - <25	Нормальна маса тіла
25 - <30	Надлишкова маса тіла
30+	Ожиріння

Рисунок 3.3 – Калькулятор BMI

Фінальною кнопкою меню програмного забезпечення HealthyLife є «Чат-бот» (рис. 3.4). Це чудовий замінник живого спеціаліста, який здатний допомогти знайти відповіді на багато питань щодо збереження фізичного та ментального здоров’я. Чат-бот розроблений таким чином, щоб надавати корисну та своєчасну інформацію користувачам, забезпечуючи їм підтримку та поради в різних аспектах здорового способу життя.

Чат-бот містить такі пункти: сон, спорт, харчування та психічний стан. Кожен з цих пунктів охоплює різні аспекти здоров'я та благополуччя, надаючи користувачам можливість отримати відповіді на свої запитання та дізнатися більше про те, як покращити своє здоров'я.

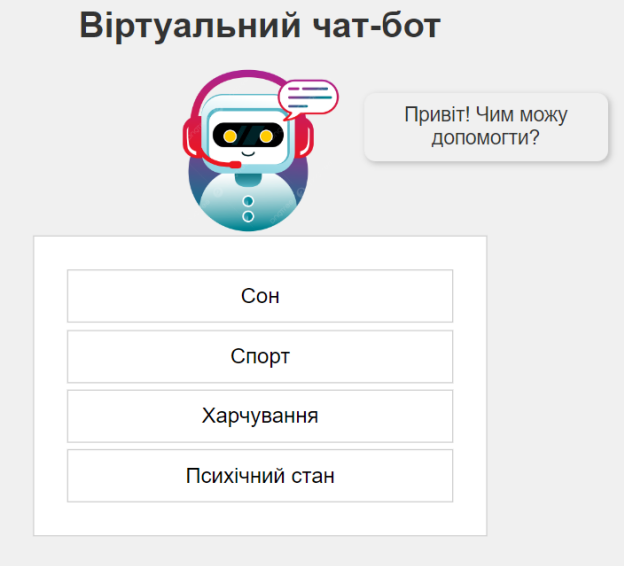


Рисунок 3.4 – Розроблений чат-бот

Наступним кроком розробляється головний блок програмного забезпечення, який містить заголовок з привітнім повідомленням та запису на консультацію (рис. 3.5).

Форма запису на консультацію містить такі поля:

1. Ім’я;
2. Фамілія;
3. Електронна пошта;
4. Номер телефону;
5. Кнопка «Запис на консультацію».

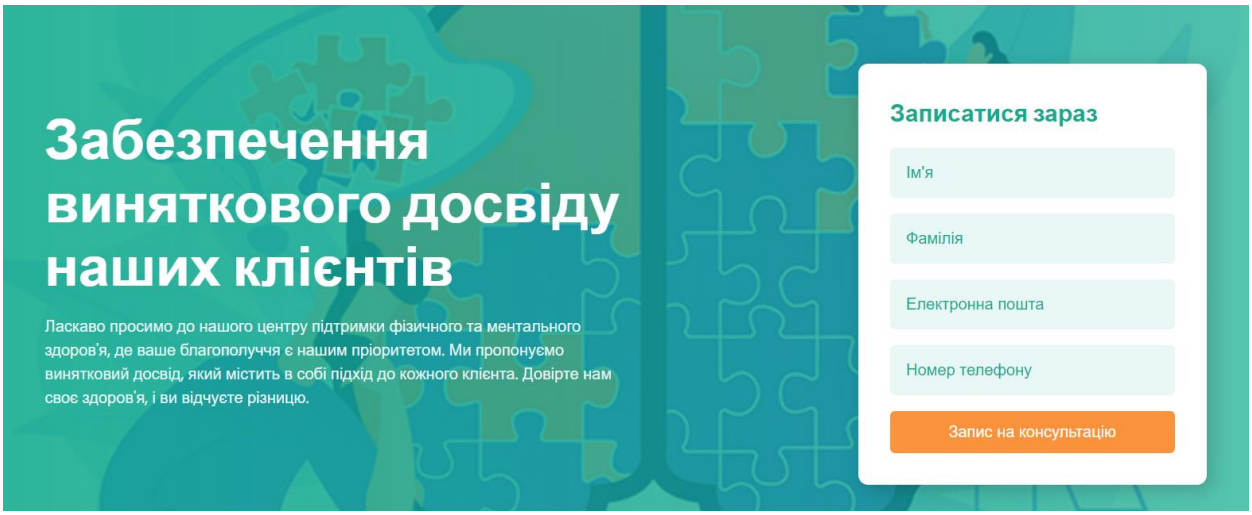


Рисунок 3.5 – Розроблений головний блок

За схожою логікою розроблено наступні декілька блоків програмного забезпечення. Після головного блоку зустрічає блок з найбільш рейтинговими статтями – статті, які найкраще себе проявляють та являються найактуальнішими у наш час. Розроблений блок з найбільш рейтинговими статтями зображений на рисунку 3.6.

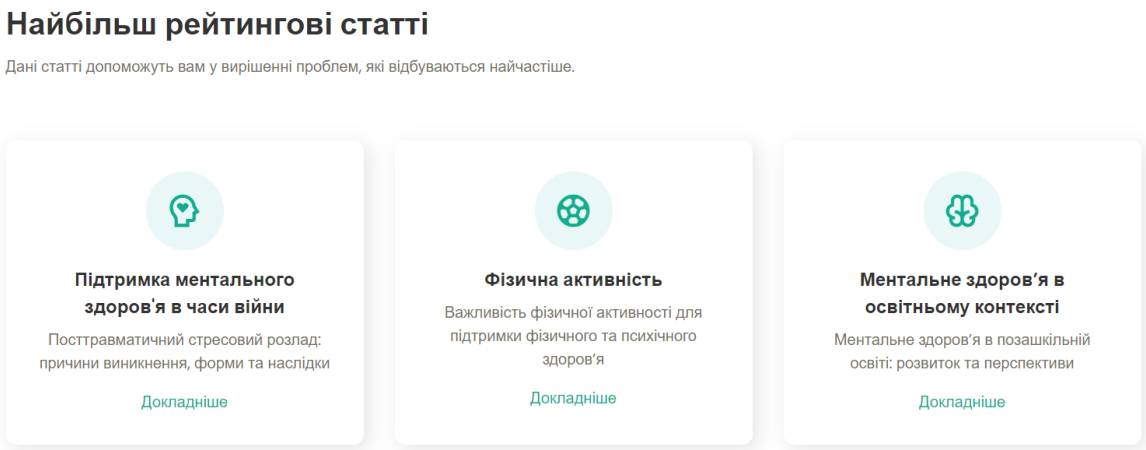


Рисунок 3.6 – Блок з найбільш рейтинговими статтями

На рисунку 3.7 зображено один із найбільш рейтингових статей програмного забезпечення.

Рисунок 3.7 – Частина найбільш рейтингової статті

Після того користувач ознайомився з основною інформацією для підтримки фізичного та ментального здоров’я, він скролить сторінку та ознайомлюється з підтримкою даного вебзастосунку (рис. 3.8).

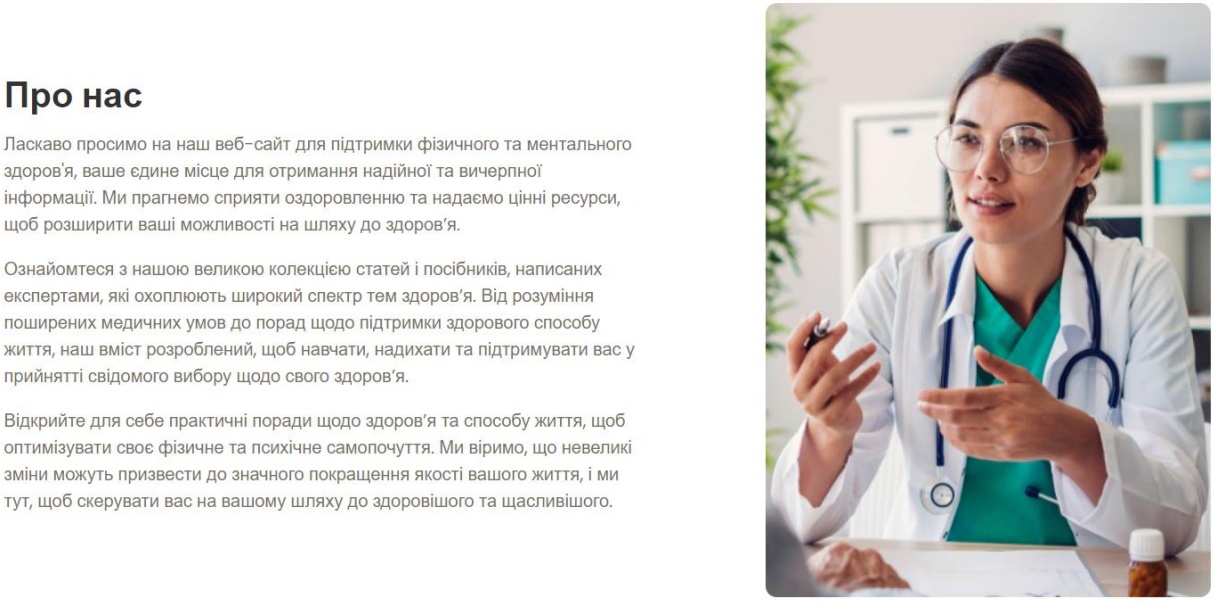


Рисунок 3.8 – Інформаційний блок «Про нас»

Потім по програному продукту буде міститися інформація про плюси програмного забезпечення HealthyLife (рис. 3.9).

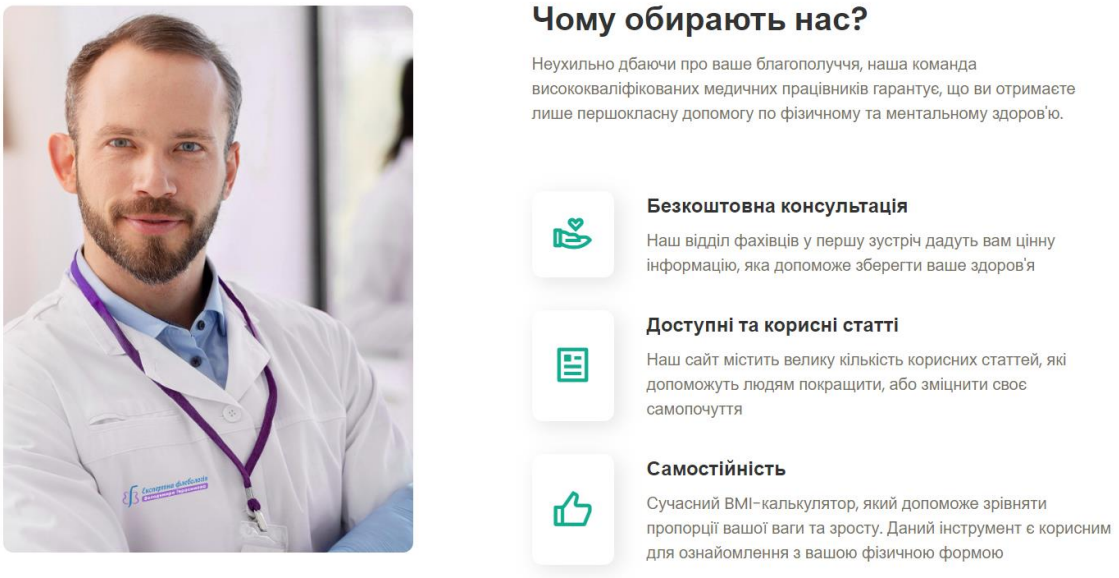


Рисунок 3.9 – Інформація про плюси програмного забезпечення

На фінальному етапі сторінки розміщені фотографії найкращих фахівців даного програмного забезпечення, їхні імена та посади (рис. 3.10).

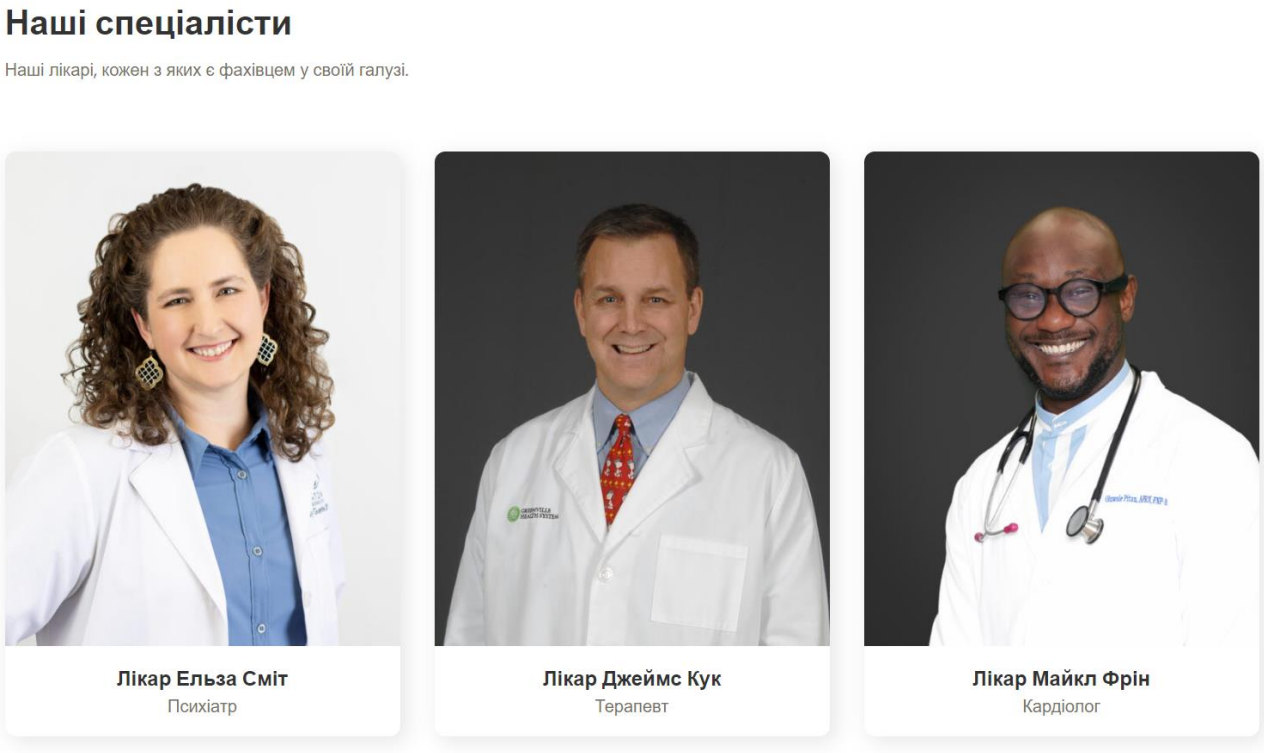


Рисунок 3.10 – Спеціалісти програмного продукту

Отже, було описано розробку клієнтської частини програмного забезпечення для системи підтримки фізичного та ментального здоров’я. Лістинг коду наведений у додатку Г.

3.4 Розробка серверної частини програмного забезпечення для системи підтримки фізичного та ментального здоров’я

Серверна частина є невід'ємною складовою програмного продукту. У даному випадку розробка буде прилягати на запис користувача на консультацію. Без даної функції клієнт не зможе отримати персональну допомогу, якщо він не знайшов інформацію на вебсайті. Тому було вирішено розробити запис на консультацію на мові програмування Java за допомогою фреймворку Spring Boot.

Spring Boot - це фреймворк, який забезпечує розробникам можливість створювати застосунки на Java швидко та ефективно. Він пропонує простий спосіб налаштовування та запуску додатків, зменшуючи необхідність у складних конфігураціях. Spring Boot автоматизує багато задач, що дозволяє розробникам фокусуватися на важливих аспектах проекту, таких як бізнес-логіка та функціональність. Завдяки вбудованим налаштуванням та бібліотекам, Spring Boot дозволяє швидше розпочати роботу над проектом та забезпечує його консистентність. Цей фреймворк також має велику спільноту та багатий екосистем, що робить його популярним вибором серед розробників [18].

Після створення основного класу main створювався простий POJO (Plain Old Java Object), що містить дані для створення електронного повідомлення: ім'я, прізвище, номер телефону та адреса електронної пошти [19].

Наступною розробкою є клас EmailSenderService, який відповідає за відправку електронних повідомлень у програмному забезпеченні. Він містить метод sendEmail, який приймає отримувача, тему та текст повідомлення. Після цього даний метод використовує JavaMailSender для відправлення листа на вказану адресу у записі на консультацію.

Для обробки HTTP-запитів, спрямованих на створення та надсилання електронних листів був підключений контролер – клас EmailController. У даного класа є метод sendEmail, який приймає об’єкт EmailData із даними для створення листа та передає їх сервісу EmailSenderService для подальшої обробки. Таким чином виконується важлива роль у взаємодії з клієнтом, обробляючи його запити та забезпечуючи відповідну відповідь на них.

Фінальним класом є WebConfig, який відповідає за конфігурацію програмного продукту. Зокрема, він налаштовує CORS (Cross-Origin Resource Sharing), що є важливим аспектом для вебзастосунків. Налаштовуючи CORS, користувач дозволяє йти запитам з інших джерел, таких як localhost:5500 та взаємодіяти з даним додатком. Таким чином, WebConfig створює необхідне середовище для безперешкодної комунікації програмного забезпечення з іншими системами чи джерелами даних.

Отже, було описано розробку серверної частини програмного забезпечення для системи підтримки фізичного та ментального здоров'я на прикладі запису на консультацію.

3.5 Висновки

Був проведений аналіз та обґрунтування вибору засобів для реалізації програмного продукту, і зроблено висновок, що для розробки будуть використовуватися наступні інструменти: HTML, CSS, JavaScript для клієнтської частини та мова програмування Java для серверної частини програмного забезпечення. Також було обрано програмні середовища VS Code та Intelij IDEA. Завершено розробку програмного забезпечення для системи підтримки фізичного та ментального здоров’я.

4 ТЕСТУВАННЯ ПРОГРАМИ

4.1 Аналіз методів тестування програмного забезпечення

Тестування стає важливою частиною процесу розробки програмного забезпечення. Його результати прямо впливають на те, наскільки зручним і функціональним буде продукт, а також на те, наскільки він відповідатиме його вимогам. Такий метод також дозволяє оцінити простоту та швидкість взаємодії з системою.

Кожен із трьох типів ящиків може використовуватися для тестування: чорний, білий і сірий. Кожна з цих стратегій має свої переваги та недоліки.

Наприклад, під час тестування в чорному ящику тестувальник не може побачити внутрішню структуру або дизайн програми. Основна мета полягає в перевірці поведінки та ефективності програми.

Ця стратегія має переваги, оскільки не потрібно розуміти внутрішню структуру програми та дозволяє користувачам виявити проблеми.

Недоліки: можуть бути пропущені деякі деталі або помилки в програмі, не дає докладного розуміння внутрішньої структури програми.

Тестування білого ящика — це метод, який надає тестувальнику повний доступ до внутрішньої структури програми, яка включає код, алгоритми та структури даних. Перевірка логіки програми та її внутрішніх компонентів є основною метою.

Переваги цього методу включають детальне тестування логіки програми, підвищення покриття тестування та виявлення помилок на ранніх стадіях розробки. Однак він може бути дорогим за часом і ресурсами та не гарантує перевірку зовнішньої поведінки програми, оскільки вимагає глибокого розуміння внутрішньої структури програми.

Grey-box testing, також відоме як «тестування сірого ящика», використовує методи чорного та білого ящика для перевірки деталей. Тестувальник використовує цю інформацію для планування тестів і перевірки основних

компонентів програми, навіть якщо йому доступ до внутрішньої структури програми обмежений.

Поєднання переваг двох методів (чорного та білого ящика), які досить незалежні від внутрішньої структури програми, дозволяє зосередитися на важливих частинах програми, є однією з переваг цього методу. Однак є й недоліки — обмежений доступ до програми може пропустити деякі деталі.

Після ретельної оцінки багатьох методів функціонального тестування було визначено, що метод «чорного ящика» є найкращим. Цей вибір був зроблений, щоб пришвидшити процес тестування, оскільки він дозволяє порівняти реальні результати з очікуваними.

4.2 Тестування розробленого програмного продукту

Файл, який містить послідовність дій для проведення певних перевірок програмного забезпечення або його властивостей, називається тест-кейсом. Він описує початковий стан системи, умови, за яких проводиться тестування, послідовність операцій, очікувані результати та стан системи після завершення тестування.

У процесі тестування програмного забезпечення тест-кейси є необхідним інструментом, оскільки вони забезпечують систематичний підхід до перевірки функціональності. Тестування програмного забезпечення полягає в тому, щоб переконатися, що воно працює належним чином, відповідає вимогам і задовольняє потреби користувачів.

Тест-кейси можуть бути категоризовані відповідно до їх призначення:

Позитивні тест-кейси: Ці сценарії перевіряють, як система працює при правильних даних та очікуваних умовах.

Негативні тест-кейси: Такі сценарії спрямовані на виявлення реакції системи на некоректні дані або непередбачувані обставини.

Випадкові тест-кейси: Вони використовують випадкові дані для тестування, що може допомогти виявити непередбачені проблеми.

Екстремальні тест-кейси: Ці сценарії тестують поведінку системи на межі допустимих значень, наприклад, мінімальне та максимальне значення або екстремальні умови.

При перевірці роботи та якості програмного забезпечення для системи підтримки фізичного та ментального здоров’я було протестовано наступний функціонал:

– відкриття програмного продукту;
– перехід між пунктами меню;
– запис на консультацію;
– перехід до перегляду статей
– виконання калькулятору BMI;
– працездатність чат-боту;
– активні посилання рейтингових статтей.

Опис тест-кейсів наведено у таблиці 4.1.

Таблиця 4.1 – Тестові випадки

Іденти-фікатор	Назва	Пріор-итетність	Кроки виконання	Очікуваний результат	Результат
ТК-1	Перевірка, чи відкривається програмне забезпечення	П0	1. Відкрити HealthyLife	1. Відкрита головна сторінка 2. Відображено медіа	Виконано
ТК-2	Перевірка переходу між пунктами меню та чи відчиняється відповідна сторінка	П1	1. Відкрити HealthyLife 2. Натиснути на кожний пункт меню 3. Перевірити, що відповідна сторінка відкрита	1. Відкрита головна сторінка 2. Відбувся перехід на нову сторінку 3. Відображена конкретна сторінка	Виконано
ТК-3	Перевірити, що користувач зможе записатися на консультацію	П2	1. Відкрити HealthyLife 2. Внести персональні дані 3. Натиснути кнопку запису	1. Відкрита головна сторінка 2. Підтведження достовірності персональних даних 3. Отримано повідомлення про успішний запис	Виконано

Продовження таблиці 4.1

ТК-4	Перевірити, що кнопка «Статті» працює вірно і відкриває основні статті програмного продукту	П2	1. Відкрити HealthyLife 2. Відкрити кнопку меню «Статті»	1. Відкрита головна сторінка 2. Користувач зайшов на сторінку із статтями 3. Успішно виведені статті	Виконано
ТК-5	Перевірити, що після натискання кожного пункту чат-бота відкривається відповідний список з інформацією	П1	1. Відкрити HealthyLife 2. Відкрити кнопку меню «Чат-бот» 3. Вибір одного із пунктів	1. Відкрита головна сторінка 2. Користувач зайшов на сторінку із чат-ботом 3. Відкрито правильні підпункти	Виконано
ТК-6	Перевірити, чи зможе користувач дізнатися свій індекс маси тіла	П1	1. Відкрити HealthyLife 2. Відкрити кнопку меню «Калькулятор BMI» 3. Внести свій зріст та вагу. 4. Натиснути кнопку «Розрахувати BMI»	1. Відкрита головна сторінка 2. Користувач зайшов на сторінку з калькулятором 3. Успішно виведена про індекс маси тіла користувача	Виконано
ТК-7	Перевірити, що кнопка «Докладніше» працює на найбльш рейтингових статтях	П2	1. Відкрити HealthyLife 2. Натиснути кнопку «Докладніше»	1. Відкрита головна сторінка 2. Користувач зайшов на сторінку із статтею 3. Успішно виведена стаття	Виконано

Отож, було створено та використано сім тест-кейсів для оцінки роботи графічного інтерфейсу і функціоналу програмного виробу. Це дозволило здійснити аналіз, підтвердити ефективність програмного виробу та переконатися, що його дії відповідають очікуваним результатам.

4.3 Розробка інструкції користувача

Інструкція користувача є інструкцією, яка пояснює, як використовувати програмний продукт. Його основне призначення полягає в тому, щоб відповісти на всі потенційні запитання, які можуть виникнути у людей, які використовують вебзастосунок. Інструкція повинна містити достатньо інформації та вказівок, щоб зробити програмний продукт зручним і ефективним для використання.

При першому запуску програми користувач опиняється на сторінці програмного забезпечення (рис. 4.1).

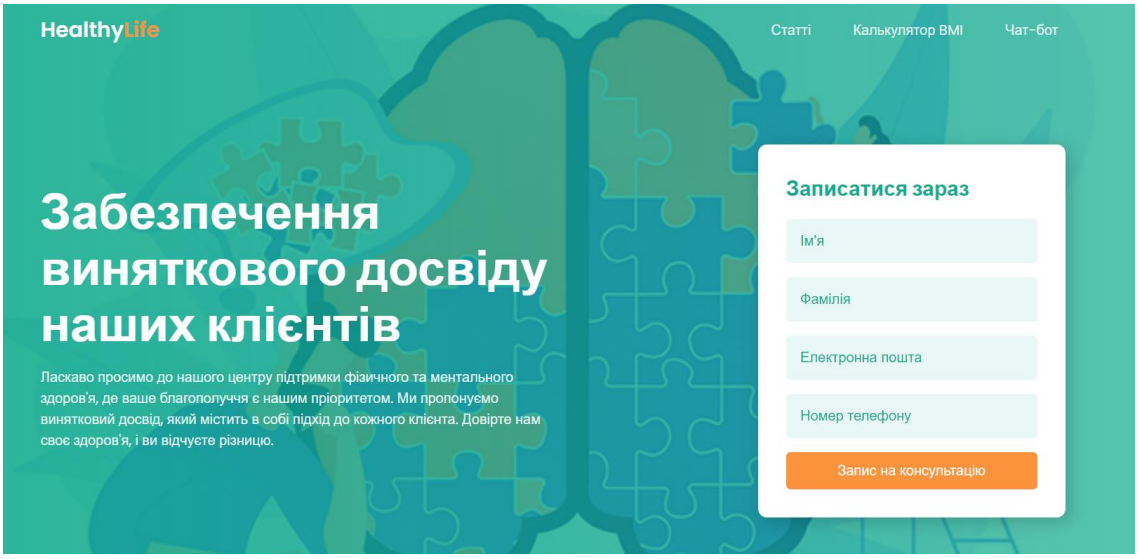


Рисунок 4.1 – Сторінка програмного забезпечення

Сторінка програмного продукту має панель із пунктами меню: Статті, Калькулятор BMI та Чат-бот. Також дана сторінка має вітальне повідомлення з лозунгом.

Для запису на консультацію потрібно ввести свої персональні дані, такі як ім’я, фамілію, електронну пошту та номер телефону. Якщо користувач ввів невірну електронну пошту, то система сповіщає про це. Також всі дані для вводу є обов’язковими для користувача. Після натиснення кнопки «Запис на консультацію» користувачу приходить сповіщення про успішний запис та йому приходить повідомлення на електронну пошту на протязі однієї хвилини.

Вигляд запису на консультацію проілюстрований на рисунку 4.2.

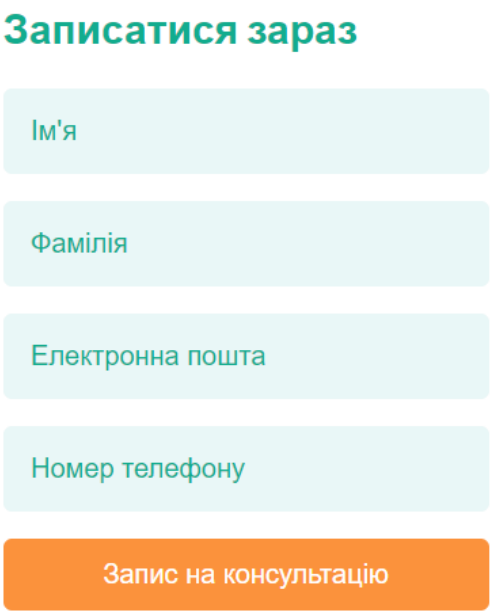


Рисунок 4.2 – Форма авторизації

Далі користувач у пункті меню переходить за посиланням «Статті» та попадає на інформаційні статті, які можуть бути корисними для підтримки системи фізичного та ментального здоров’я (рис. 4.3).

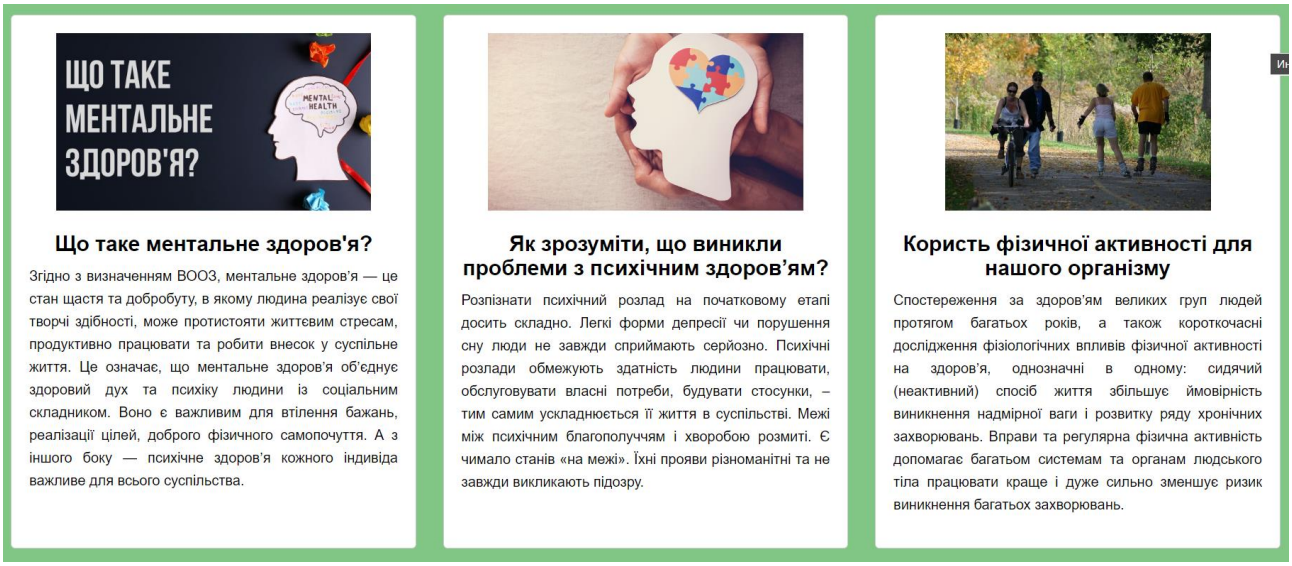


Рисунок 4.3 – Сторінка із статтями

Ще одним пунктом меню є «Калькулятор ВМІ». Використовуючи калькулятор індексу маси тіла (ВМІ), можна визначити співвідношення зросту

до маси тіла. Вимірювання можна отримати, діляючи масу тіла у кілограмах на квадрат висоти у метрах. Простий індикатор маси тіла (BMI) може бути використаний для визначення того, чи ви в нормі, маєте недостатню або зайву масу тіла або маєте ожиріння з точки зору здоров'я.

На вебсторінці користувач бачить перед собою 2 об'єкти: сам калькулятор та його таблиця інтерпретації. Користувач вводить свої дані, такі як вага та зріст та самостійно може обчислити індекс маси тіла. Справа від обчислювального процесу є таблиця, яка допоможе користувачеві визначити на якому рівні маси тіла він знаходиться.

Результат роботи калькулятору BMI зображено на рисунку 4.4.

BMI	Інтерпретація
<18.5	Недостатня маса тіла
18.5 - <25	Нормальна маса тіла
25 - <30	Надлишкова маса тіла
30+	Ожиріння

Рисунок 4.4 – Результат роботи калькулятору BMI

Фінальною кнопкою меню є чат-бот. Відкривши сторінку користувач бачить форму з чат-ботом, де містяться 4 активні кнопки для пошуку інформаці: сон, спорт, харчування, психічний стан. Результат роботи чат-бота наведено нижче (рис. 4.5).

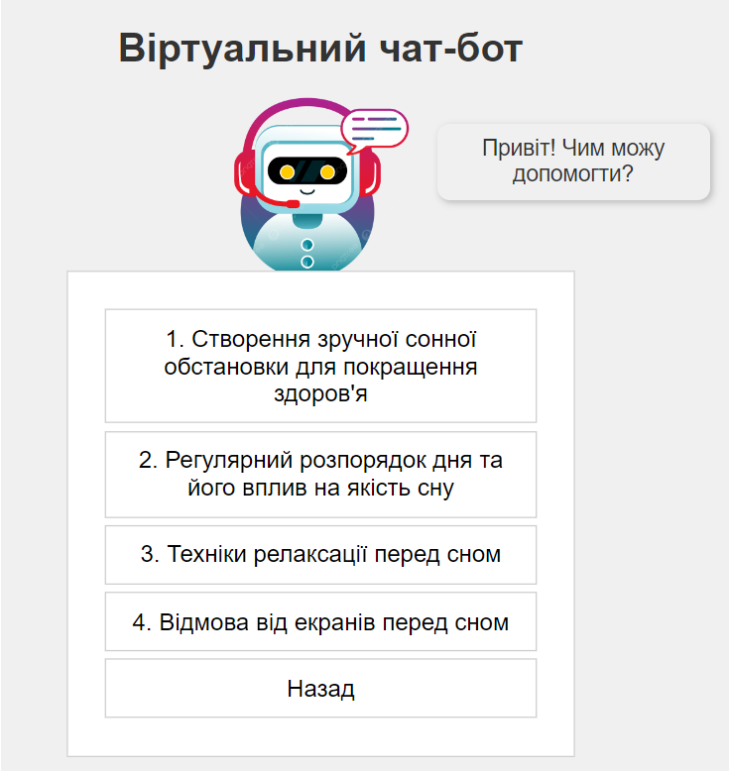


Рисунок 4.5 – Вибір пункту меню

На рисунку 4.6 представлено регулярний розпорядок дня

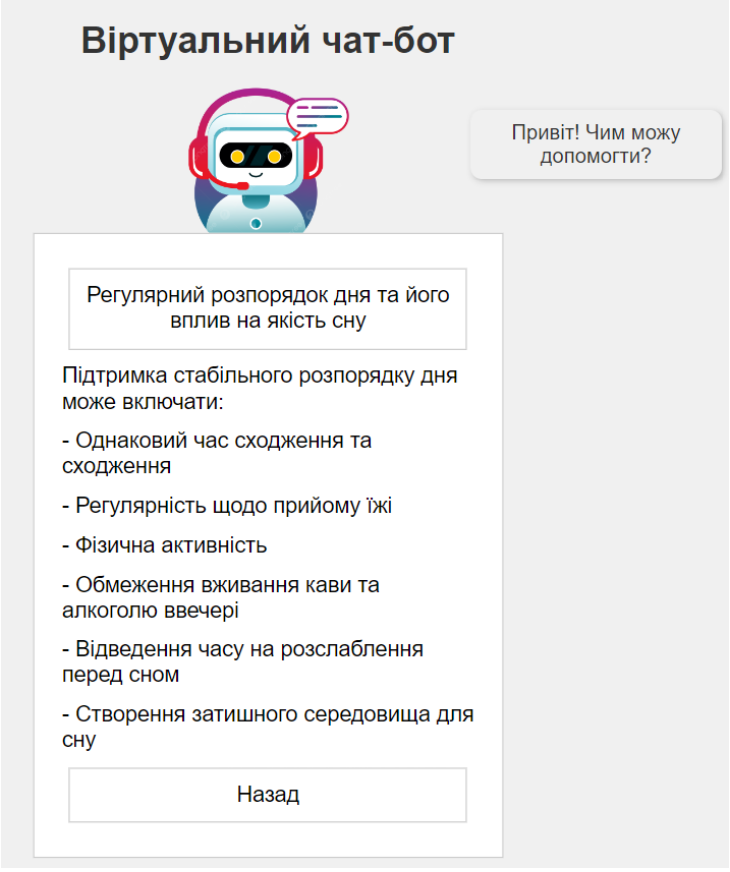


Рисунок 4.6 – Вибір підпункту меню

Блок з найбільш рейтинговими статтями показує статті, які показали себе найкраще та відбиралися реальними користувачами і це свідчить про те, що саме ці статті є актуальними й популярними. На рисунку 4.7 проілюстровано рейтингову статтю про ментальне здоров'я в освітньому контексті.

Ментальне здоров’я в освітньому контексті

Ментальне здоров’я визначає, як ми сприймаємо світ, справляємося із життєвими стресами та викликами, а також як спілкуємося з іншими людьми. Освіта відіграє критичну роль у формуванні цих аспектів ментального здоров’я. Перш за все, вона надає нам знання і розуміння того, як працює наш розум і емоції. Інструкції і навчальні програми розроблені для того, щоб надати нам засоби для кращого розуміння себе і навколишнього світу. Позашкільна освіта доповнює навчання в школах, надаючи вихованцям можливість розвивати навички та

Рисунок 4.7 – Найбільш рейтингова стаття

Отже, було створено інструкцію користувача програмного забезпечення для системи підтримки фізичного та ментального здоров’я.

4.4 Висновки

Після аналізу методів тестування програмного забезпечення було вирішено провести тестування за допомогою методу «Чорного ящика». Застосовано набір тестових випадків для перевірки функціональності розробленого програмного застосунку. Протестовані такі функції, як програмного продукту, перехід між пунктами меню, запис на консультацію, перехід до перегляду статей, виконання калькулятору BMI, працездатність чат-боту та переходи за активними посиланнями рейтингових статей. Крім того, була розроблена інструкція для користувача, яка містить опис деяких частин програмного забезпечення.

ВИСНОВКИ

У бакалаврській дипломній роботі було розроблено програмне забезпечення для систем підтримки фізичного та ментального здоров'я "HealthyLife". Даний програмний продукт надає користувачам змогу самостійно знайти відповіді на свої запитання щодо здоров'я, або звернутися за допомогою до спеціаліста застосунку.

Було розглянуто існуючі аналоги програмних продуктів, такі як BetterHelp, Calm та MyFitnessPal. Також було описано їхні плюси та мінуси, і визначено, що програмне забезпечення "HealthyLife" переважає за своєю функціональністю.

Крім того, було представлено варіативний аналіз і обґрунтування вибору засобів реалізації програмного забезпечення. Було прийнято рішення, що HTML, CSS та JavaScript будуть використані для створення клієнтської частини програмного забезпечення, а Java для створення серверної частини. Також було обрано два програмні середовища: VS Code та Intelij IDEA.

У бакалаврській дипломній роботі було зроблено наступне:

- розроблено модель загального алгоритму роботи
- розроблено основні алгоритми програмного продукту
- розроблено зручний та легкий інтерфейс
- розроблено програмне забезпечення для систем підтримки фізичного та ментального здоров'я
- проведено тестування програмного застосунку

Було розроблено метод збору даних, цей метод включає визначення джерел даних, методів їхньої обробки та зберігання. Також було розроблено інструкцію користувача, що забезпечує легкість освоєння та використання програмного забезпечення.

Бакалаврську дипломну роботу було оформлено відповідно до методичних вказівок [20].

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бохонько Д.В. Коваленко О.О. Розробка програмного забепезчення для системи підтримки фізичного та ментального здоров’я. / Молодь в науці: дослідження, проблеми, перспективи (МН-2024), Вінниця: ВНТУ, 2024. 2 с.
2. BetterHelp. BetterHelp.: веб-сайт. URL: https://www.betterhelp.com/
3. The #1 App for Meditation and Sleep. Calm.: веб-сайт. URL: https://www.calm.com/
4. MyFitnessPal. MyFitnessPal.: веб-сайт. URL: https://www.myfitnesspal.com
5. Структура сайту: основні види та правила їх розробки. Webtune.: веб-сайт. URL: https://webtune.com.ua/statti/web-rozrobka/struktura-sajtu/
6. Правильні поєднання та підбір кольору для сайту: рекомендації для новачків. Impulse-design.: веб-сайт. URL: https://impulse-design.com.ua/ua/vybor-tsveta-dlya-sajta.html
7. Алгоритм. Wikipedia.: веб-сайт. URL: https://ru.wikipedia.org/wiki/Алгоритм
8. Калькулятор індексу маси тіла. Znaimo.: веб-сайт. URL: https://znaimo.gov.ua/bmi-calculator
9. Клієнт-серверна архітектура. Wikipedia.: веб-сайт. URL: https://uk.wikipedia.org/wiki/Клієнт-серверна_архітектура
10. Що таке Back-end розробка. Wezom.: веб-сайт. URL: https://wezom.com.ua/ua/blog/chto-takoe-back-end-razrabotka
11. Що таке html. Css.in.: веб-сайт. URL: https://css.in.ua/article/shcho-take-css_3
12. CSS. Wikipedia.: веб-сайт. URL: https://uk.wikipedia.org/wiki/CSS
13. JavaScript. Wikipedia.: веб-сайт. URL: https://ru.wikipedia.org/wiki/JavaScript

14. ЩО ТАКЕ PHP. Avada-media.: веб-сайт. URL https://avada-media.ua/ua/what-is-php/
15. Що таке Java і де вона використовується. Goit.global.: веб-сайт. URL: https://goit.global/ua/articles/shcho-take-java-i-de-vona-vykorystovuietsia/
16. Використання розширення Visual Studio Code. Learn Microsoft.: веб-сайт. URL: https://learn.microsoft.com/uk-ua/power-apps/maker/portals/vs-code-extension
17. IntelliJ IDEA. Wikipedia.: веб-сайт. URL: https://uk.wikipedia.org/wiki/IntelliJ_IDEA
18. Як Spring Boot допомагає прискорити розробку додатків. Foxminded.: веб-сайт. URL: https://foxminded.ua/spring-boot/
19. Класи, види вкладених класів із прикладами. Javarush.: веб-сайт. URL: https://javarush.com/ua/groups/posts/uk.588.klasi-vidi-vkladenikh-klasv-z-prikladami
20. Методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів студентів спеціальності 121 "Інженерія програмного забезпечення" / Уклад. О. Н. Романюк, Г. О. Черноволик – Вінниця: ВНТУ, 2022. – 52с.

ДОДАТКИ

Додаток А – Технічне завдання

61

Міністерство освіти і науки України

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
д.т.н., професор
_____ О. Н. Романюк
«12» березня 2024 р.

Технічне завдання
на бакалаврську дипломну роботу
«Розробка програмного забезпечення для системи підтримки фізичного та ментального здоров'я»
за спеціальністю 121 – Інженерія програмного забезпечення

Керівник бакалаврської дипломної роботи:
к.т.н., доцент кафедри ПЗ,
_____ О.О. Коваленко
«12» березня 2024 р.

Виконав:
_____ студент гр. 1ПІ-20б, Д.В. Бохонько
«12» березня 2024 р.

Вінниця ВНТУ 2024

1. Найменування та галузь застосування

Розробка має назву «HealthyLife» і виконується в рамках бакалаврської дипломної роботи.

Галузь застосування – медицина.

2. Підстава для розробки

Підставою для розробки даної бакалаврської дипломної роботи є рішення засідання кафедри протокол №17 від 20 березня 2024 року програмного забезпечення.

3. Мета та призначення розробки

Метою роботи є поліпшення ментального здоров’я населення шляхом розробки веб-застосунку для підтримки психічного здоров’я та тестування якості продукту, що дозволить якісно надавати інформацію користувачам.

Призначення роботи – покращення процесу надання фізичної та ментальної допомоги шляхом залучення до цього програмних технологій.

4. Вихідні дані для проведення НДР

1. Всеукраїнська програма ментального здоров'я "Ти як?" [Електронний ресурс] – Режим доступу до ресурса: http://oblzdrav.mk.gov.ua/index.php/statti/13620-vseukrajinska-programa-mentalnogo-zdorov-ya
2. Розробка програмного забезпечення та ІТ-рішень [Електронний ресурс] – Режим доступу до ресурса: https://armedsoft.com/ua/services/rozrobka-programnogo-zabezpechennya-ta-it-rishen#
3. ПЛАТФОРМА САМОВДОСКОНАЛЕННЯ ОНЛАЙН [Електронний ресурс] – Режим доступу до ресурса: https://sunshineathome.com/uk

5. Технічні вимоги

Для нормальної роботи програми, необхідний персональний комп’ютер з

наступною мінімальною конфігурацією:

- Процесор з частотою не менше 2.0 Гц;
- Оперативна пам’ять ємністю не менше 4 Гб;
- Запам’ятовуючий пристрій ємністю 500 Gb.
- Операційна система Windows

6. Конструктивні вимоги

Конструкція системи має відповідати ергономічним та технічним вимогам. Текстова та графічна документація повинна відповідати стандартам України.

7. Перелік технічної документації, що пред’являється по закінченню робіт:

– пояснювальна записка;
– технічне завдання;
– лістинги програми.

8. Вимоги до рівня уніфікації та стандартизації

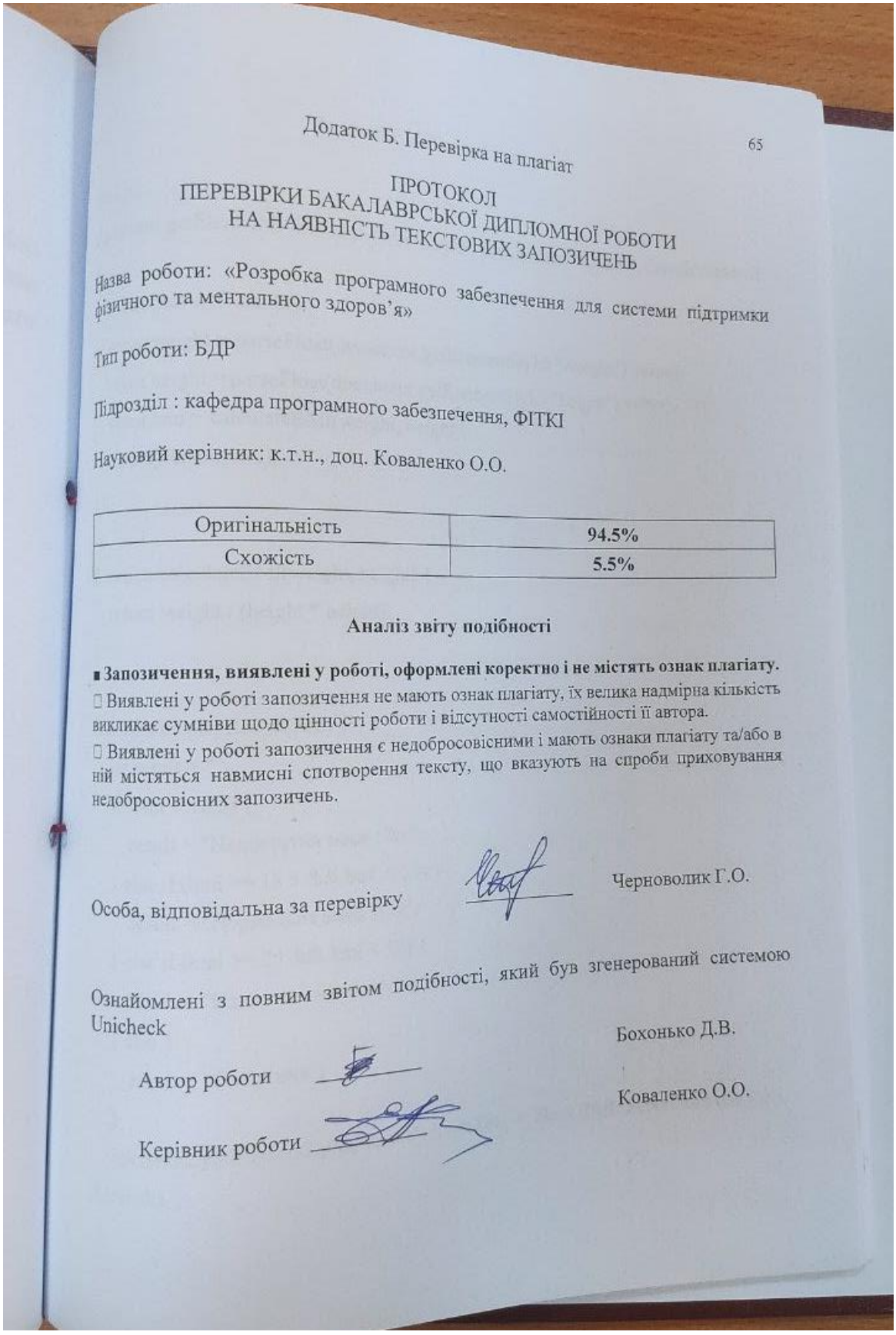
При розробці програмних засобів слід дотримуватися уніфікації і ДСТУ.

9. Стадії та етапи розробки

№ з/п	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз проблеми, обґрунтування актуальності розробки системи та постановка задачі	14.03.2024 – 22.04.2024	
2	Розробка моделі, методів та алгоритмів роботи програмного забезпечення	22.04.2024 – 29.04.2024	
4	Аналіз і вибір середовища програмування	29.05.2024 – 03.05.2024	
5	Розробка модулів програми	03.05.2024 – 20.05.2024	
6	Тестування роботи розробленого програмного забезпечення	20.05.2024 – 26.05.2024	
7	Оформлення матеріалів до захисту БДР	26.05.2024 – 30.05.2024	

10. Порядок контролю та прийняття

Виконання етапів бакалаврської дипломної роботи контролюється керівником відповідно до встановленого графіка. Прийняття бакалаврської дипломної роботи здійснюється ДЕК, яка затверджена завідувачем кафедри згідно з графіком.



Додаток Б. Перевірка на плагіат

ПРОТОКОЛ
ПЕРЕВІРКИ БАКАЛАВРСЬКОЇ ДИПЛОМНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: «Розробка програмного забезпечення для системи підтримки фізичного та ментального здоров'я»

Тип роботи: БДР

Підрозділ : кафедра програмного забезпечення, ФІТКІ

Науковий керівник: к.т.н., доц. Коваленко О.О.

Оригінальність	**94.5%**
Схожість	**5.5%**

Аналіз звіту подібності

■ **Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.**

□ Виявлені у роботі запозичення не мають ознак плагіату, їх велика надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку Черноволик Г.О.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck

Бохонько Д.В.

Автор роботи

Коваленко О.О.

Керівник роботи

Додаток В – Лістинг програми

Bmi.js:

```
document.getElementById("bmi-form").addEventListener("submit", function(event)
{
    event.preventDefault();
    const weight = parseFloat(document.getElementById("weight").value);
    const height = parseFloat(document.getElementById("height").value) / 100;
    const bmi = calculateBMI(weight, height);
    displayResult(bmi);
});

function calculateBMI(weight, height) {
    return weight / (height * height);
}

function displayResult(bmi) {
    let result = "";
    if (bmi < 18.5) {
        result = "Недостатня маса тіла";
    } else if (bmi >= 18.5 && bmi < 25) {
        result = "Нормальна маса тіла";
    } else if (bmi >= 25 && bmi < 30) {
        result = "Надлишкова маса тіла";
    } else {
        result = "Ожиріння";
    }
    document.getElementById("result").innerText = `Ваш BMI: ${bmi.toFixed(1)} -
${result}`;
}
```

ot.js:

```
var currentState = "categories";
var previousCategory = null;

var subCategoriesData = {
   "Сон": {
      subResponses: ["Створення зручної сонної обстановки для покращення здоров'я", "Регулярний розпорядок дня та його вплив на якість сну", "Техніки релаксації перед сном", "Відмова від екранів перед сном"],
      expandedInfo: {
         1: [
            "Кілька порад для зручної сонної обстановки:",
            "- Виберіть правильний матрац та подушку для підтримки тіла.",
            "- Забезпечте темний та тихий середовище для спокійного сну.",
            "- Контролюйте температуру в кімнаті (18°C - 22°C).",
            "- Спробуйте ароматерапію з лавандою для розслаблення.",
            "- Уникайте електроніки перед сном, щоб уникнути збудження.",
            "- Практикуйте ритуали перед сном, такі як тепла ванна або чашка трав'яного чаю, щоб сигналізувати про наближення часу сну."
         ],
         2: [
            "Підтримка стабільного розпорядку дня може включати:",
            "- Однаковий час сходження та сходження",
            "- Регулярність щодо прийому їжі",
            "- Фізична активність",
            "- Обмеження вживання кави та алкоголю ввечері",
            "- Відведення часу на розслаблення перед сном",
            "- Створення затишного середовища для сну"
         ],
```

3: [

"Перед сном використовуйте різні техніки релаксації, які полегшать засинання та покращать сон:",

"- Глибоке дихання: Уповільнюйте темп дихання, щоб заспокоїти нервову систему",

"- Прогресивна м'язова релаксація: Розслабте м'язи, починаючи з ніг і закінчуючи обличчям",

"- Медитація: Спрямовуйте увагу на дихання або спокійний образ",

"- Релаксаційні вправи для рук та обличчя: Розтягуйте м'язи та масажуйте їх",

"- Спокійна музика або звуки природи: Створюйте релаксуючу атмосферу перед сном"

],

4: [

"Відмова від екранів перед сном допомагає покращити якість сну, оскільки світло екранів, особливо синє, може заважати природним циклам сну. Ось декілька порад:",

"- Встановіть час вимкнення екранів: Вимикайте всі електронні пристрої, наприклад, за 1-2 години до сну",

"- Читайте або слухайте аудіокниги: Замініть екранні заняття на читання або прослуховування аудіокниг",

"- Створіть ритуал без екранів: Розробіть ритуал, що не включає використання електроніки, наприклад, тепла ванна або розтяжка",

"- Використовуйте режим 'Нічний режим': Багато пристроїв мають цей режим, який зменшує синє світло",

"- Підготуйте приміщення до сну: Забезпечте темне та тихе середовище у спальні, уникайте яскравого світла та екранів перед сном"

]

}

},

```
"Спорт": {
    subResponses: ["Прості вправи для покращення фізичного здоров'я", "Спорт як засіб боротьби з депресією та стресом", "Групові заняття та їх позитивний вплив на ментальне здоров'я", "Спорт як засіб самовираження та підвищення самооцінки"],
    expandedInfo: {
        1: [
            "Отримуйте користь для фізичного здоров'я з цими простими вправами:",
            "- Прогулянка: Щоденні короткі прогулянки підвищать настрій та кровообіг.",
            "- Вправи на силу та гнучкість: Присідання, віджимання, планка та розтяжки покращують м'язову силу та гнучкість.",
            "- Кардіовправи: Стрибки на місці, скакалка, біг на місці або велосипедна прогулянка покращують серцево-судинне здоров'я.",
            "- Йога або пілатес: Прості вправи з йоги або пілатесу поліпшують координацію та гнучкість.",
            "Вправи на розслаблення: Глибоке дихання, медитація та розтяжки допомагають розслабити м'язи та знизити стрес.",
        ],
        2: [
            "Спорт - ефективний засіб проти депресії та стресу:",
            "- Ендорфіни: Виділення ендорфінів підвищує настрій та зменшує стрес.",
            "- Ментальне здоров'я: Заняття спортом поліпшує концентрацію та самоповагу.",
            "- Соціальна підтримка: Спорт створює можливості для соціальних контактів.",
            "- Відволікання: Заняття спортом допомагає відволіктися від проблем.",
```

```
            "- Якісний сон: Регулярна активність сприяє здоровому сну.",
        ],
        3: [
            "Групові заняття позитивно впливають на ментальне здоров'я, оскільки:",
            "- Соціальна підтримка: Створюють можливість для соціальної взаємодії та підтримки.",
            "- Мотивація: Надихають до досягнення цілей через взаємну підтримку.",
            "- Відчуття спільності: Зменшують відчуття самотності та ізоляції.",
            "- Релаксація: Сприяють відпочинку та релаксації.",
            "- Стресреліф: Допомагають знизити рівень стресу через фізичну активність та позитивне спілкування.",
        ],
        4: [
            "Спорт як засіб самовираження та підвищення самооцінки:",
            "- Досягнення цілей: Підвищує самооцінку через досягнення успіхів.",
            "- Позитивний вплив на емоції: Знижує стрес та підвищує настрій.",
            "- Соціальна взаємодія: Розвиває соціальні навички та сприяє спілкуванню.",
            "- Самодисципліна: Вимагає відповідальності та розвиває самоконтроль.",
        ]
    }
},
"Харчування": {
    subResponses: ["Балансоване харчування для збереження здоров'я", "Сезонні продукти та їхні переваги для організму", "Гідратація та її важливість для фізичного та ментального здоров'я", "Стресостійкі продукти для підтримки психічного здоров'я"],
```

```
expandedInfo: {
    1: [
        "Балансоване харчування для здоров'я:",
        "- Різноманітність: Включайте у раціон різноманітні продукти для отримання всіх необхідних поживних речовин.",
        "- Правильні порції: Контролюйте розміри порцій, щоб уникнути переведення.",
        "- Фрукти та овочі: Їжте достатню кількість фруктів та овочів для вітамінів та мінералів.",
        "- Повільні вуглеводи: Віддавайте перевагу повільним вуглеводам, таким як цільнозернові продукти, для стабільного рівня цукру в крові.",
        "- Гідрація: Пийте достатню кількість води для гідрації тіла та підтримання здоров'я.",
    ],
    2: [
        "Сезонні продукти мають свої переваги для організму:",
        "- Свіжість та смак: Сезонні продукти зазвичай свіжіші та смачніші, оскільки вони збираються у відповідний сезон.",
        "- Більша поживна цінність: Оскільки сезонні продукти збираються в оптимальний час, вони мають більшу кількість вітамінів та мінералів.",
        "- Підтримка місцевого господарства: Вибір сезонних продуктів сприяє підтримці місцевих фермерів та розвитку місцевої економіки.",
        "- Екологічна стійкість: Сезонні продукти зазвичай вирощуються та збираються без використання шкідливих хімічних добрив та пестицидів.",
        "- Вартість: У сезон свої продукти зазвичай дешевші, оскільки їх більше на ринку.",
    ],
    3: [
        "Гідратація є ключовою для фізичного та ментального здоров'я:",
```

```
            "- Фізична активність: Забезпечує енергію та підтримує температуру тіла під час тренувань.",
            "- Когнітивні функції: Покращує пам'ять та концентрацію.",
            "- Очищення організму: Видаляє токсини та забруднення.",
            "- Настрій і емоції: Впливає на настрій та запобігає втомі.",
        ],
        4: [
            "Стресостійкі продукти для підтримки психічного здоров'я:",
            "- Омега-3 жирні кислоти: Знаходяться в рибі, горіхах та насінні. Покращують настрій та зменшують вплив стресу.",
            "- Банани: Багаті вітаміном B6, який допомагає виробляти серотонін, гормон щастя.",
            "- Темний шоколад: Містить флавоноїди, які знижують рівень стресу та покращують настрі",
            "- Зелений чай: Містить легкий стимулятор, L-теанін, який сприяє заспокоєнню та концентрації.",
            "- Фрукти та овочі: Багаті вітамінами та антиоксидантами, які знижують рівень стресу та покращують загальне здоров'я.",
        ]
    }
},
"Психічний стан": {
    subResponses: ["Техніки дихальної гімнастики для зняття стресу та зосередження", "Позитивні афірмації для підтримки психічного здоров'я", "Психологічні переваги прогулянок на свіжому повітрі", "Техніки візуалізації для досягнення цілей та зниження тривоги"],
    expandedInfo: {
        1: [
            "Прості техніки дихальної гімнастики:",
```

"- Глибоке дихання: Повільний вдих на 4 секунди, затримка на 7 секунд, повільний видих на 8 секунд.",

"- 4-7-8 метод: Вдихайте на 4, тримайте подих на 7, видихайте на 8.",

"- Дихання через черево: Покладіть руки на живіт, дихайте так, щоб вони піднімалися вгору при вдиху і опускалися при видиху.",

],

2: [

"Ось кілька позитивних афірмацій для підтримки психічного здоров'я:",

"- Я вірю в себе і в свою силу здати подолати будь-які труднощі.",

"- Кожен день я стаю сильнішим і впевненішим у собі.",

"- Моє життя є цінним і повне можливостей.",

"- Я ціную себе і дозволяю собі бути щасливим.",

"- Кожен день я росту та розвиваюся, перетворюючи виклики на можливості.",

],

3: [

"Прогулянки на свіжому повітрі мають безліч психологічних переваг:",

"- Зменшення стресу: Природа та свіже повітря допомагають знизити рівень стресу та відчути спокій.",

"- Підвищення настрою: Прогулянки на свіжому повітрі стимулюють вироблення ендорфінів, гормонів щастя, що покращує настрій.",

"- Покращення концентрації: Відпочинок в природі допомагає очистити розум і підвищити здатність до концентрації.",

"- Сприяння творчості: Природний ландшафт може надихати і стимулювати творчість та інновації.",

"- Підвищення енергії: Прогулянки на свіжому повітрі допомагають зарядитися енергією та відновити втомлений організм.",

],

```
            4: [
                "Ось кілька простих технік візуалізації:",
                "- Позитивна візуалізація цілей: Уявіть, як ви досягаєте свої цілі.",
                "- Візуалізація успіхів минулого: Пригадайте ситуації, коли ви вже досягали успіхів.",
                "- Візуалізація спокою: Замисліться над символом спокою та уявіть, як ви перебуваєте у цьому місці.",
                "- Візуалізація позитивних сценаріїв: Передбачте успішний вихід із ситуацій, що викликають тривогу.",
                "- Візуалізація дихання: Уявіть собі, як ви вдихаєте позитивні емоції та видаляєте негативне.",
            ]
        }
    }
};

function showMainResponses() {
    var chatBox = document.getElementById("chat-box");
    chatBox.innerHTML = "";

    var mainCategories = ["Сон", "Спорт", "Харчування", "Психічний стан"];
    mainCategories.forEach(function(category) {
        var responseDiv = document.createElement("div");
        responseDiv.className = "response";
        responseDiv.textContent = category;
        responseDiv.addEventListener("click", function() {
            showSubResponses(category);
        });
        chatBox.appendChild(responseDiv);
    });
```

```
    currentState = "categories";
}

function showSubResponses(category) {
    var chatBox = document.getElementById("chat-box");
    chatBox.innerHTML = "";

    previousCategory = category;

    var subResponsesData = subCategoriesData[category].subResponses;
    subResponsesData.forEach(function(response, index) {
        var responseDiv = document.createElement("div");
        responseDiv.className = "response";
        responseDiv.textContent = index + 1 + ". " + response;
        responseDiv.addEventListener("click", function() {
            showExpandedResponse(response,
subCategoriesData[category].expandedInfo[index + 1]);
        });
        chatBox.appendChild(responseDiv);
    });

    var backButton = document.createElement("div");
    backButton.className = "response";
    backButton.textContent = "Назад";
    backButton.addEventListener("click", back);
    chatBox.appendChild(backButton);

    currentState = "subResponses";
}
```

```
function back() {
    if (currentState === "subResponses") {
        showMainResponses();
    } else if (currentState === "expandedResponse") {
        showSubResponses(previousCategory);
    }
}

function showExpandedResponse(response, expandedInfo = []) {
    var chatBox = document.getElementById("chat-box");
    chatBox.innerHTML = "";

    var responseDiv = document.createElement("div");
    responseDiv.className = "response";
    responseDiv.textContent = response;
    chatBox.appendChild(responseDiv);

    expandedInfo.forEach(function(info) {
        var infoDiv = document.createElement("div");
        infoDiv.className = "expanded-info";
        infoDiv.textContent = info;
        chatBox.appendChild(infoDiv);
    });

    var backButton = document.createElement("div");
    backButton.className = "response";
    backButton.textContent = "Назад";
    backButton.addEventListener("click", back);
    chatBox.appendChild(backButton);
```

```
    currentState = "expandedResponse";
}

var chatBox = document.getElementById("chat-box");
var greetingDiv = document.createElement("div");
greetingDiv.className = "chat-message";
greetingDiv.textContent = "Привіт! Чи можу допомогти?";
chatBox.appendChild(greetingDiv);
showMainResponses();

function showSpeechBubble(message) {
    const speechBubble = document.querySelector('.speech-bubble');
    const speechText = document.getElementById('speech-text');

    speechText.textContent = "";

    speechBubble.style.display = "block";

    let i = 0;
    let interval = setInterval(() => {
        speechText.textContent += message.charAt(i);
        i++;
        if (i >= message.length) {
            clearInterval(interval);
            setTimeout(() => {
                showSpeechBubble(message);
            }, 3000);
        }
    }, 50);
}
```

```
window.onload = () => {
    showSpeechBubble("Привіт! Чим можу допомогти?");
};
```

Consulatations.js:

```
const FEEDBACK_FORM = document.querySelector('#feedback_form');

function sendFeedBack(feedback) {
    fetch("/api/feedback", {
        method: 'POST',
        headers : {
            "Content-Type": "application/json"
        },
        body: JSON.stringify(feedback),
    }).then((response) =>response.json()).then (data => {
        console.log(data)
        alert('Успіх');
    }).catch((error) => {
        console.error(error);
        alert('Помилка');
    })
}

FEEDBACK_FORM.addEventListener('submit', (e) => {
    e.preventDefault();
    const feedbackFormData = new FormData(e.target);
    console.log('feedbackFormData', feedbackFormData);
    const feedback = Object.fromEntries(feedbackFormData);
    console.log('feedback', feedback);
```

```
    sendFeedBack(feedback);
})
```

DemoApplication.java:

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class DemoApplication {

	public static void main(String[] args) {
		SpringApplication.run(DemoApplication.class, args);
	}

}
```

EmailData.java:

```
package com.example.demo.pojo;

import lombok.Data;

@Data
public class EmailData {

    private String firstName;

    private String lastName;
```

```
    private String phone;

    private String email;

}
```

EmailSenderService.java:

```
package com.example.demo.service;

import lombok.RequiredArgsConstructor;
import org.springframework.mail.SimpleMailMessage;
import org.springframework.mail.javamail.JavaMailSender;
import org.springframework.stereotype.Service;

@Service
@RequiredArgsConstructor
public class EmailSenderService {

    private final JavaMailSender mailSender;

    public void sendEmail(String whom, String subject, String body) {
        SimpleMailMessage mailMessage = new SimpleMailMessage();
        mailMessage.setFrom("doctor2424255@gmail.com");
        mailMessage.setTo(whom);
        mailMessage.setText(body);
        mailMessage.setSubject(subject);

        mailSender.send(mailMessage);
```

```
        System.out.println("Mail sent: " + mailMessage);
    }

}
```

EmailController.java:

```
package com.example.demo.controller;

import com.example.demo.pojo.EmailData;
import com.example.demo.service.EmailSenderService;
import lombok.RequiredArgsConstructor;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequiredArgsConstructor
public class EmailController {

    private final EmailSenderService emailSenderService;

    @PostMapping("/sendEmail")
    public void sendEmail(@RequestBody EmailData emailData) {
        emailSenderService.sendEmail(
                emailData.getEmail(),
                "Запис на консультацію",
                "Привіт " + emailData.getFirstName() + " " + emailData.getLastName() +
                        "\n" +
                        "Ви успішно записалися на консультацію. Ми з Вами зв'яжемось
за номером телефону " + emailData.getPhone()
```

```
        );
    }

}
```

WebConfig.java:

```
package com.example.demo.config;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.CorsRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
public class WebConfig implements WebMvcConfigurer {

    @Bean
    public WebMvcConfigurer corsConfigurer() {
        return new WebMvcConfigurer() {
            @Override
            public void addCorsMappings(CorsRegistry registry) {
                registry.addMapping("/**")
                        .allowedOrigins("http://127.0.0.1:5500")
                        .allowedMethods("GET", "POST", "PUT", "DELETE", "HEAD")
                        .allowedHeaders("*")
                        .allowCredentials(true);
            }
        };
    }
}
```

Додаток Г – Графічна частина

ГРАФІЧНА ЧАСТИНА

Розробка програмного забезпечення для системи підтримки фізичного та ментального здоров’я

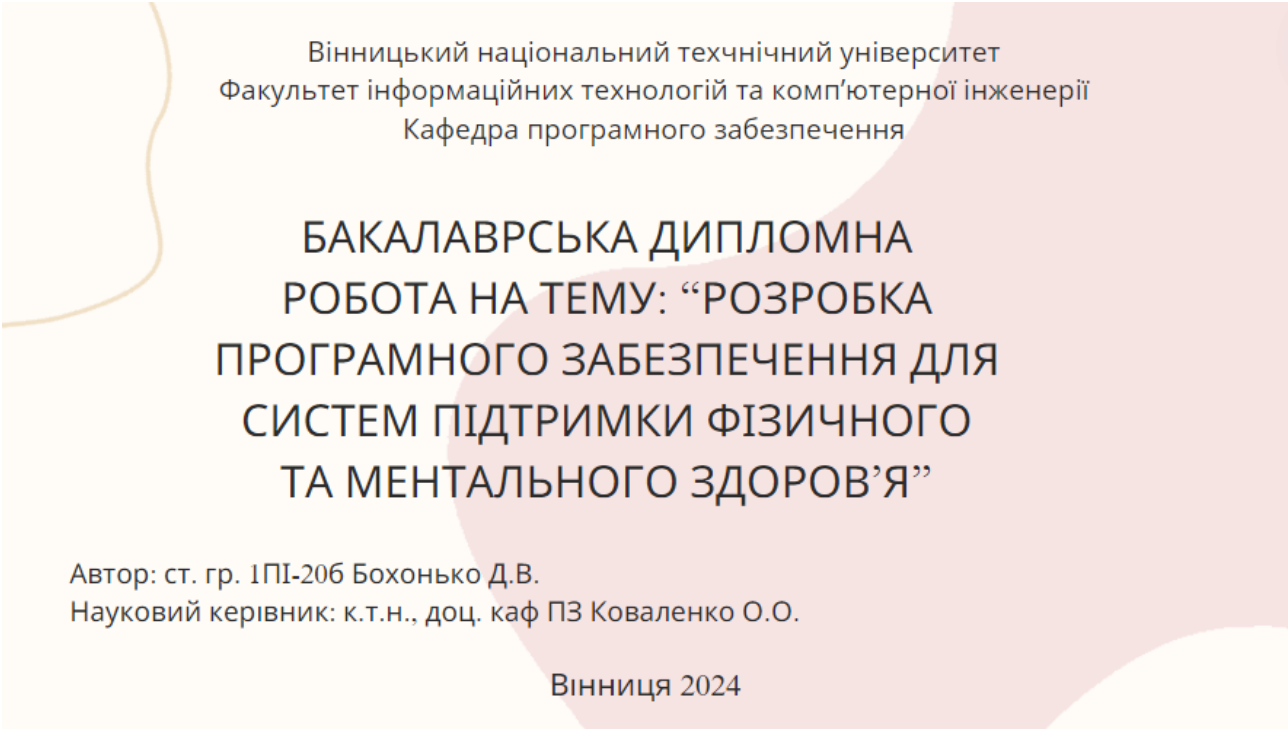


Рисунок Г.1 – Титульний слайд

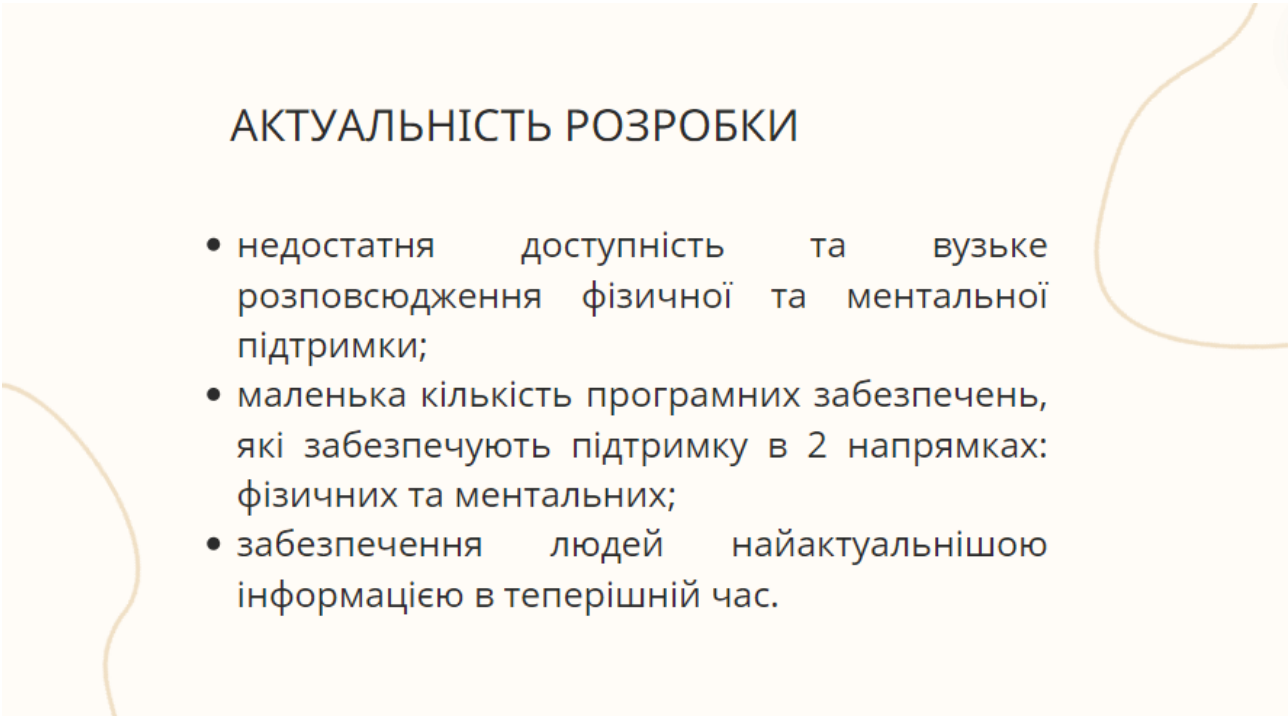


Рисунок Г.2 – Актуальність розробки

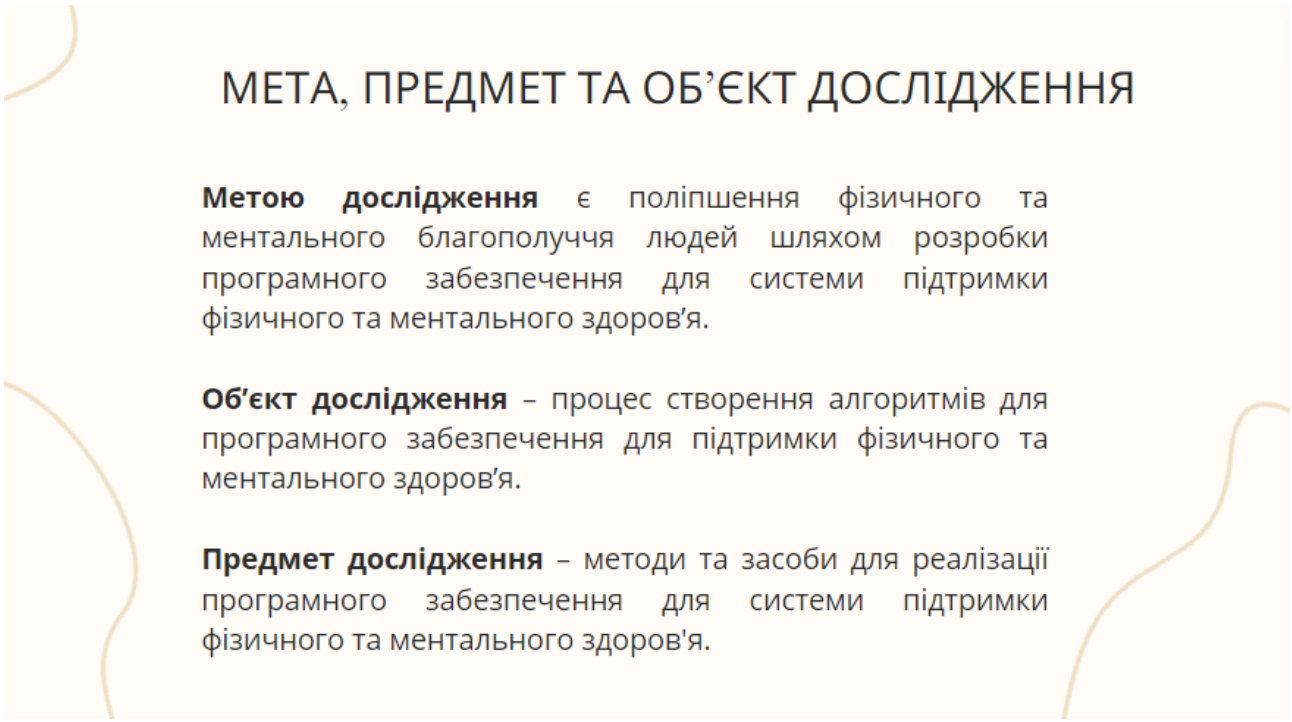


Рисунок Г.3 – Мета, предмет та об'єкт дослідження

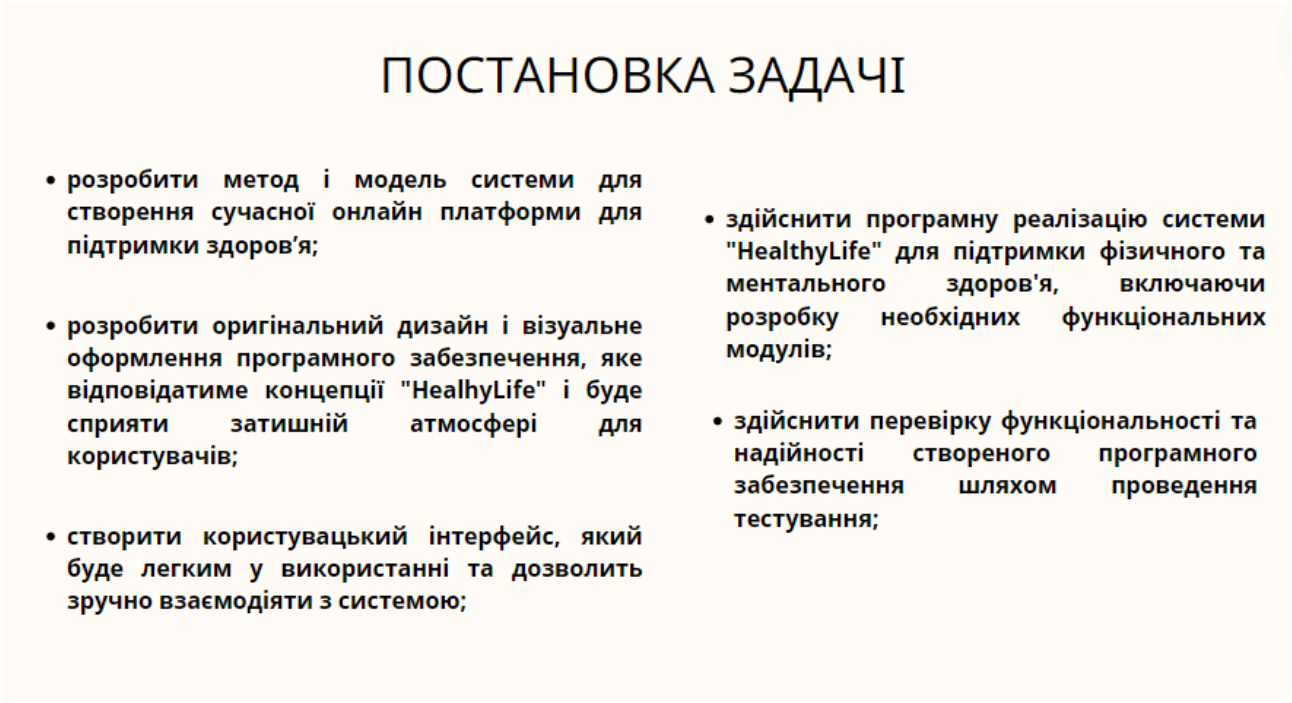


Рисунок Г.4 – Постановка задачі

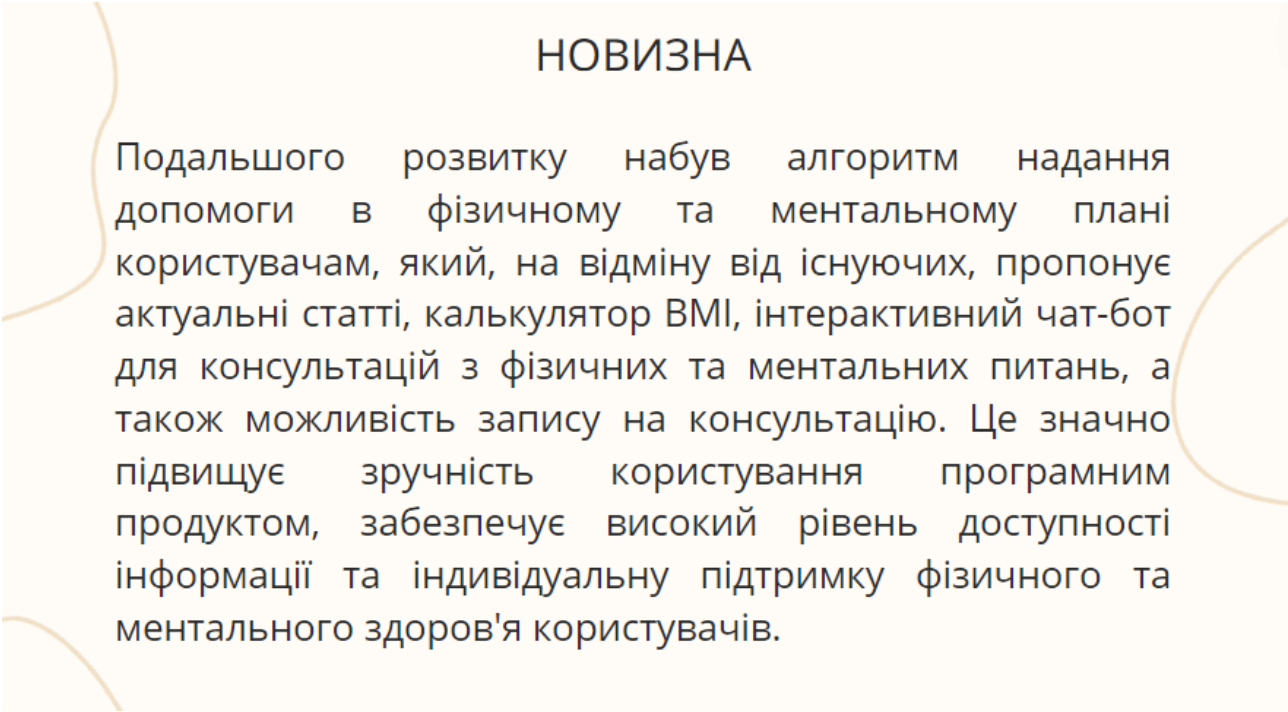


Рисунок Г.5 – Новизна

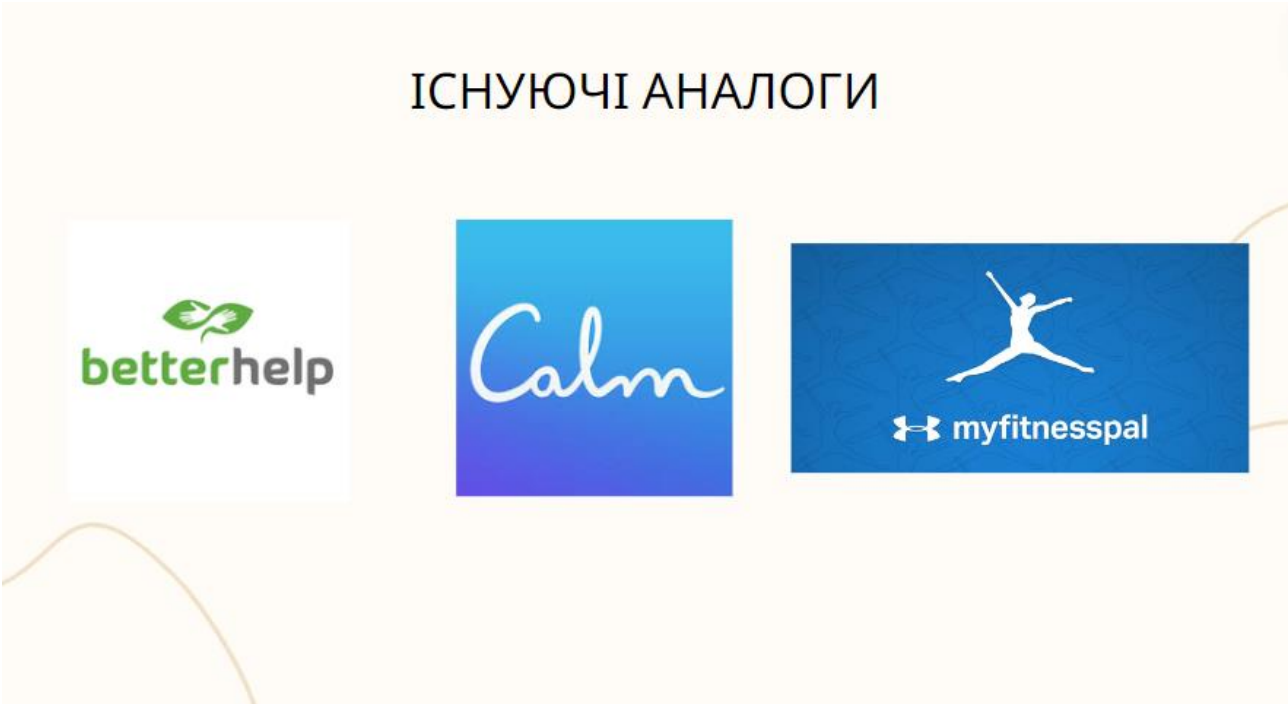


Рисунок Г.6 – Існуючі аналоги

АНАЛІЗ АНАЛОГІВ

Критерій	BetterHelp	Calm	MyFitnessPal	HealthyLife
Можливість запису на консультацію	+	-	+	+
Наявність фізичної і ментальної допомоги	-	-	-	+
Наявність калькулятора BMI	-	-	+	+
Безкоштовне користування	-	+	-	+
Зручний інтерфейс	-	+	+	+
Сума	1	2	3	5

Рисунок Г.7 – Аналіз аналогів

МЕТОД ЗБОРУ ДАНИХ

- **збір інформації з вирахуванням індексу маси тіла**
- **аналіз інформації, яку шукає користувач в ча-боті**
- **надання персоналізованої інформації виходячи з інформаційних пошуків користувача**

Рисунок Г.8 – Метод збору даних

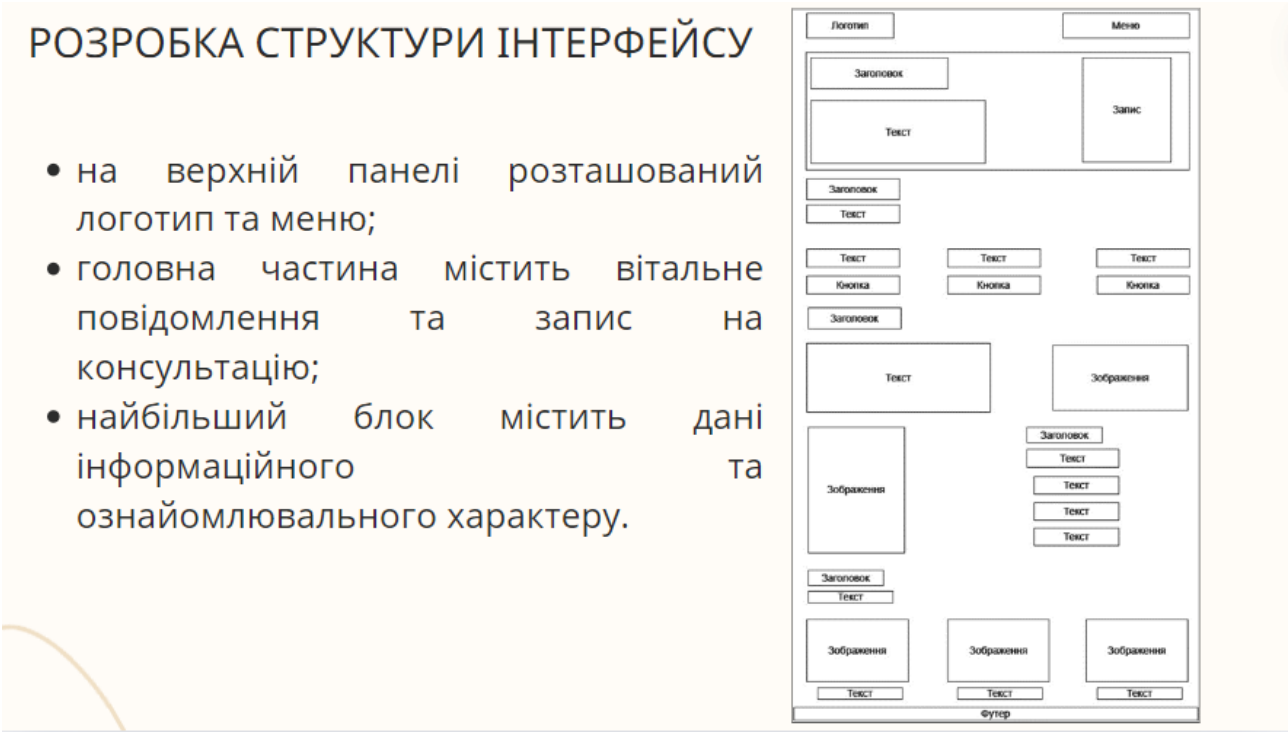


Рисунок Г.9 – Розробка інтерфейсу користувача

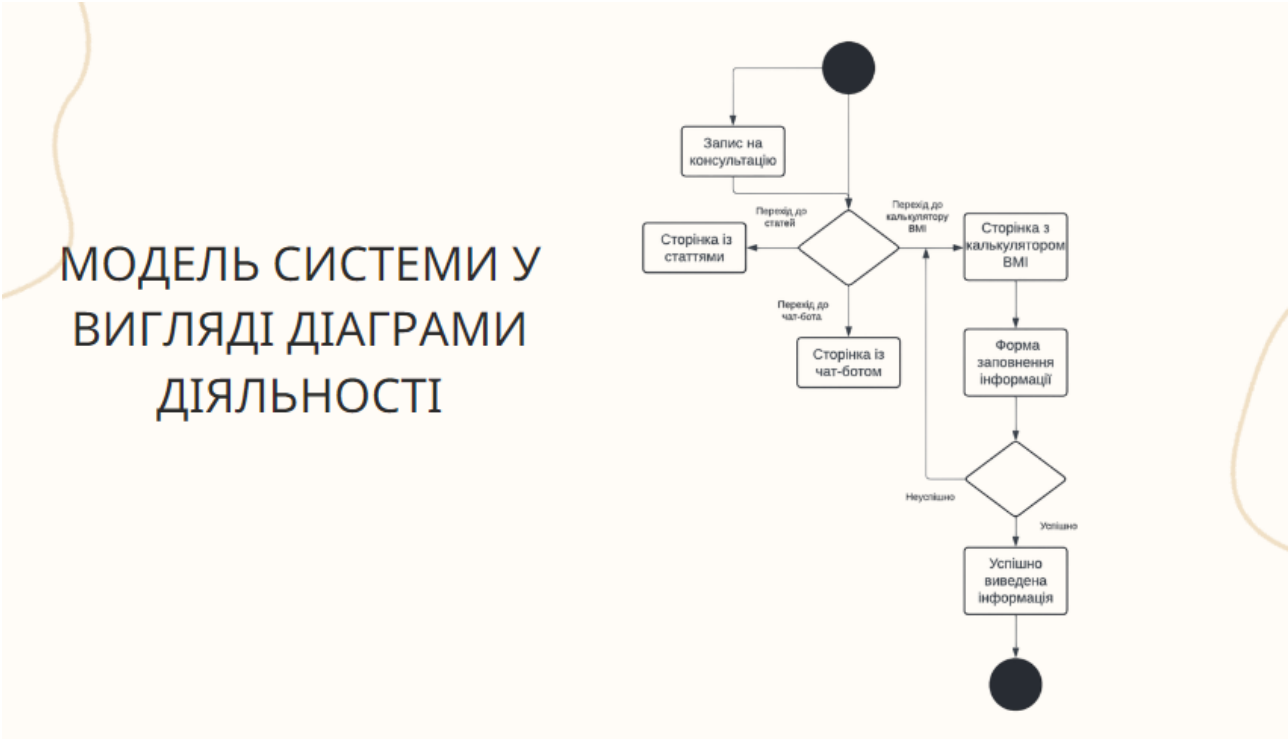


Рисунок Г.10 – Модель системи у вигляді діаграми діяльності

Рисунок Г.11 – Блок-схеми алгоритмів роботи системи

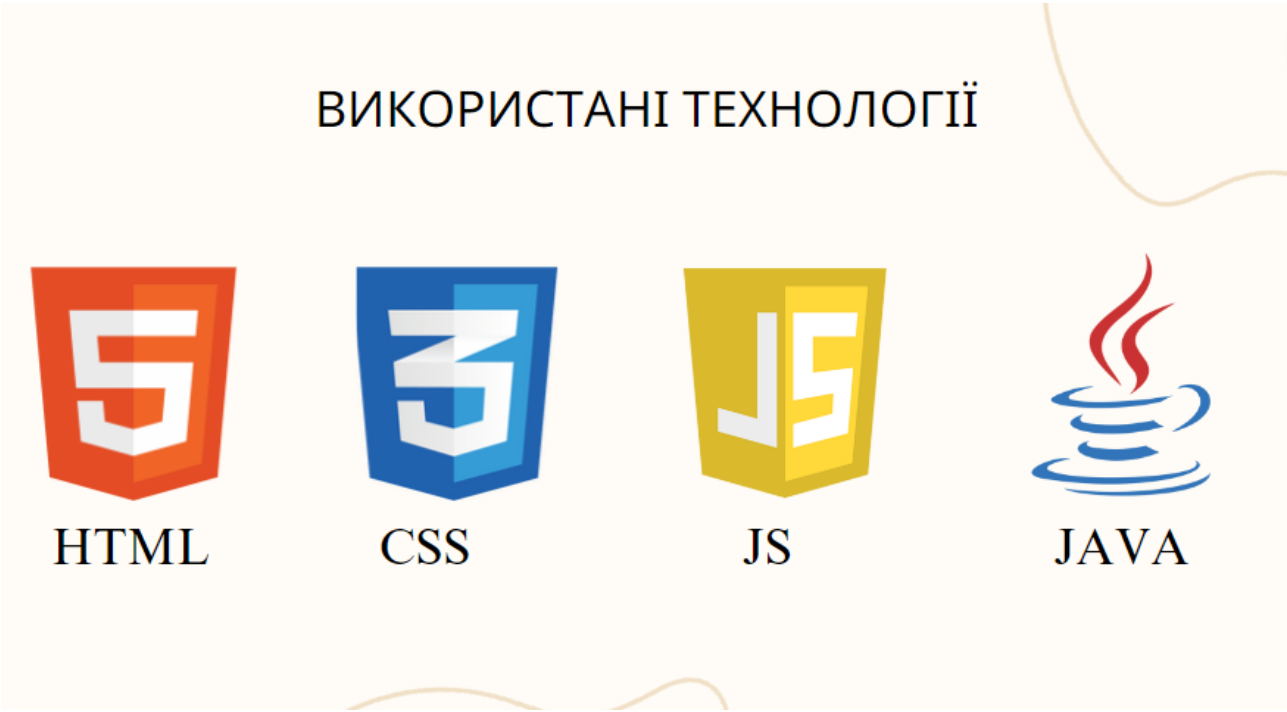


Рисунок Г.12 – Використані технології

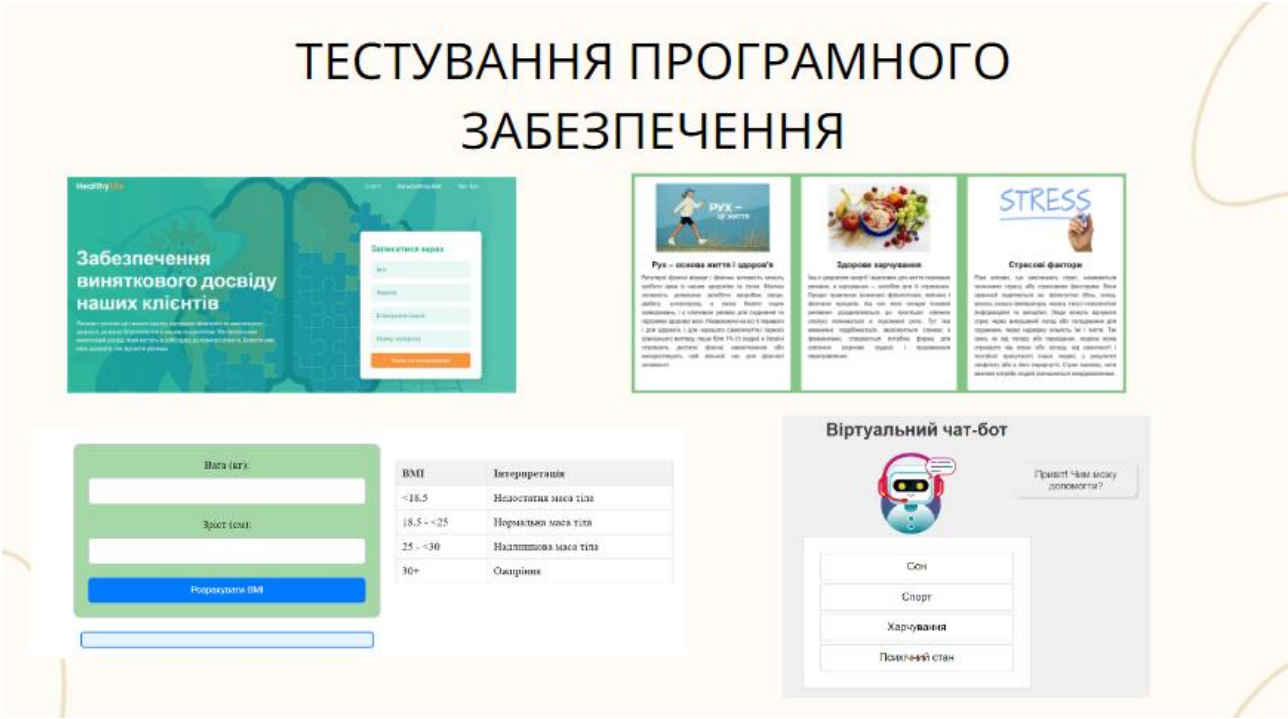


Рисунок Г.13– Тестування програмного забезпечення

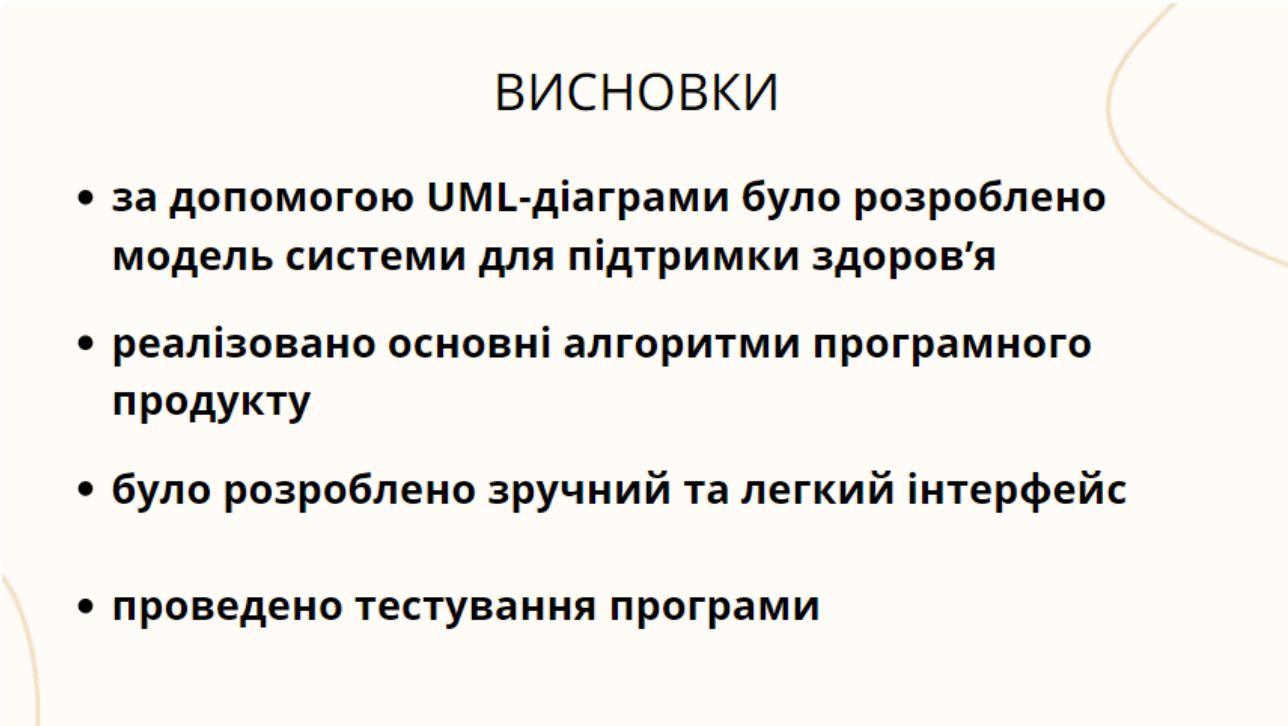


Рисунок Г.14 – Висновки

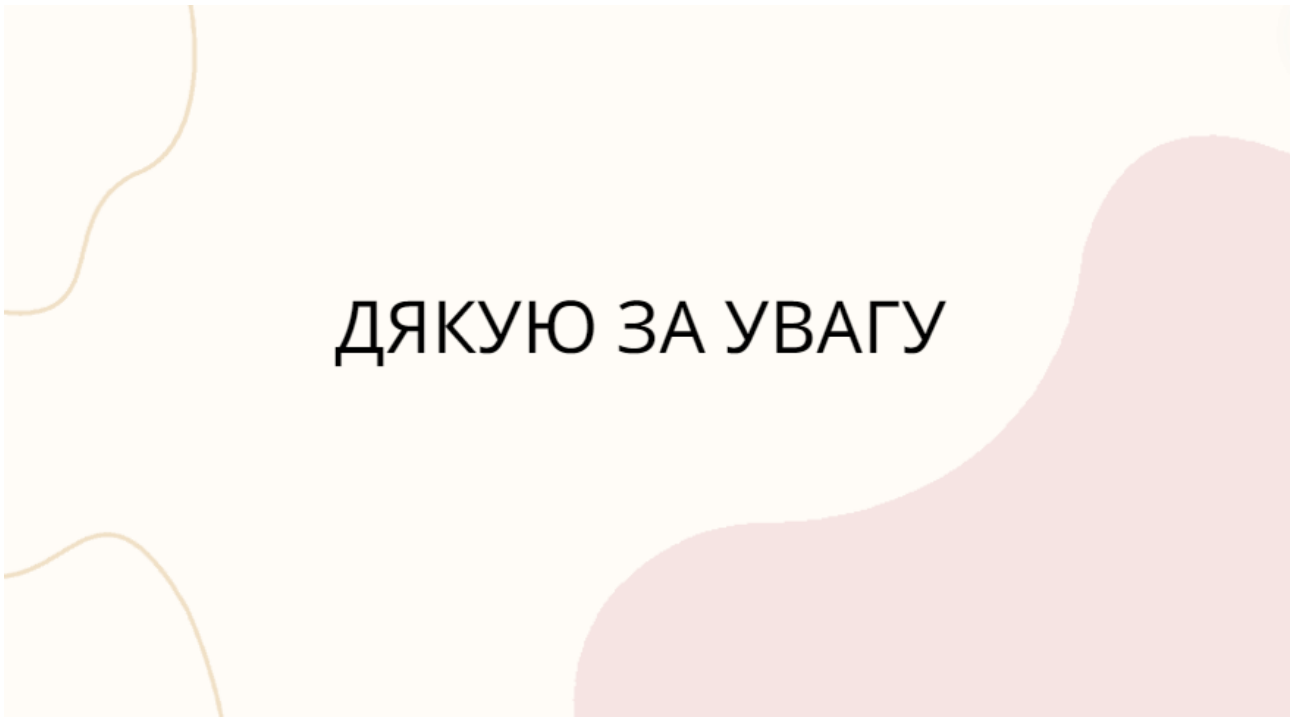


Рисунок Г.15 – Кінцевий слайд