ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка програмного засобу для щоденної психологічної підтримки користувача мовою Python»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Поліна П'ЯВЧУК
(підпис)

Виконала: здобувачка вищої освіти групи ПД-44

Поліна П'ЯВЧУК

Керівник: Вікторія ЖЕБКА
д.т.н., професор

Рецензент:

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
« ____ » _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

П’явчук Поліні Сергіївні

1. Тема кваліфікаційної роботи: «Розробка програмного засобу для щоденної психологічної підтримки користувача мовою Python» керівник кваліфікаційної роботи д.т.н., професор, завідувач кафедри ТЦР Вікторія ЖЕБКА,
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «27» лютого 2024 р. № 36.

2. Строк подання кваліфікаційної роботи «28» травня 2024 р.

3. Вихідні дані до кваліфікаційної роботи: теоретичні відомості про методи психологічної підтримки за допомогою програмних засобів, опис моделей представлення емоцій, технічна документація з описом процесу розробки веб-застосунків, використання мови Python для вирішення задач веб-застосунків.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1.Аналіз предметної галузі та існуючих інструментів самостійного фіксування психологічного стану користувачем та психологічної підтримки.
2. Проектування веб-застосунку для щоденної психологічної підтримки користувача.

3. Програмна реалізація веб-застосунку для щоденної психологічної підтримки користувача.
4. Тестування веб-застосунку.

5. Перелік графічного матеріалу: *презентація*
1. Аналіз аналогів.
2. Вимоги до програмного забезпечення.
3. Алгоритм роботи застосунку.
4. Програмні засоби реалізації.
5. Діаграма варіантів використання.
6. Діаграма класів.
7. Діаграма діяльності.
8. Схема бази даних.
9. Карта сайту.
10. Екранні форми.
11. Апробація результатів дослідження.

6. Дата видачі завдання «28» лютого 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури	28.02.-06.03.2024	
2	Аналіз та дослідження існуючих аналогів	07.03-13.03.2024	
3	Аналіз предметної галузі та існуючих інструментів самостійного фіксування психологічного стану користувачем і психологічної підтримки	14.03-19.03.2024	
4	Проектування веб-застосунку для щоденної психологічної підтримки користувача	20.03-29.03.2024	
5	Програмна реалізація веб-застосунку для щоденної психологічної підтримки користувача	30.03-20.04.2024	
6	Тестування веб-застосунку	21.04-28.04.2024	
7	Оформлення роботи: вступ, висновки, реферат	29.04-05.05.2024	
8	Розробка демонстраційних матеріалів	06.05-12.05.2024	
9	Попередній захист роботи	13.05-31.05.2024	

Здобувачка вищої освіти _____ Поліна П’ЯВЧУК
(підпис)

Керівник
кваліфікаційної роботи _____ Вікторія ЖЕБКА
(підпис)

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 46 стор., 7 табл., 30 рис., 15 джерел.

Мета роботи – спрощення самостійного фіксування користувачем психологічного стану та психологічної підтримки.

Об'єкт дослідження – самостійне фіксування психологічного стану та психологічна підтримка.

Предмет дослідження – програмне забезпечення для самостійного фіксування користувачем психологічного стану та психологічної підтримки.

Короткий зміст роботи: В роботі проаналізовано особливості предметної галузі, застосунки для психологічної психологічної підтримки користувача: Moodtracker.com, eMoods, Better Me: Mental Health. Розроблено діаграму прецедентів, веб-дизайн застосунку, діаграму діяльності веб-застосунку, реляційну базу даних, та програмно реалізовані наступні функціональні можливості фіксування настрою, введення щоденника нотаток та перегляд вже створених нотаток, прослуховування медитацій, перегляд статей на тему психологічної самопідтримки, перегляд статистики настрою за тиждень, місяць рік. Проведено мануальне та Usability тестування веб-застосунку. В роботі використано мову програмування Python з бібліотекою Django для backend частини веб-застосунку та HTML, CSS, JavaScript з бібліотекою React для front-end частини застосунку. База даних в цій роботі реалізована за допомогою MySQL.

Сферою використання застосунку є самостійне фіксування психологічного стану користувачем та психологічна підтримка користувача.

КЛЮЧОВІ СЛОВА: ПСИХОЛОГІЧНА ПІДТИМКА, ТРЕКЕР ЕМОЦІЙ, ВЕБ-ЗАСТОСУНОК, PYTHON.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми та її актуальність: В сучасному світі нових викликів, збільшення робочого навантаження та пришвидшених темпах життя кількість скарг на психологічне здоров’я невпинно зростає. В свою чергу доступ до психологічної підтримки спеціалістом залишається ускладненим, а загальна картина обізнаності та відкритості діалогу про ментальне здоров’я залишається на низькому рівні. Основним ресурсом про психологічне здоров’я стала мережа, стали інтернет ресурси про ментальне здоров’я, згодом веб-застосунки та застосунки для мобільних пристроїв. Враховуючи збільшений попит на запит доступної психологічної підтримки програмні засоби для психологічного здоров’я являються особливо актуальними.

Мета роботи: спрощення самостійного фіксування користувачем психологічного стану та психологічної підтримки.

Об’єкт дослідження: самостійне фіксування психологічного стану та психологічна підтримка.

Предмет дослідження: програмне забезпечення для самостійного фіксування користувачем психологічного стану та психологічної підтримки.

Завдання дослідження:

1. Провести огляд та проаналізувати існуючі методи та технології самостійного відслідковування психологічного стану та психологічної підтримки.
2. Провести аналіз та обрати інструменти, засоби розробки веб-застосунку для реалізації проекту.
3. Сформулювати функціональні, нефункціональні вимоги до веб-застосунку використовуючи проведений огляд та аналіз існуючих аналогів.
4. Розробити прототип та дизайн веб-застосунку для щоденної психологічної підтримки користувача.
5. Провести дослідження зручності розробленого дизайну серед користувачів різного технічного рівня підготовки.
6. Виконати технічне проектування веб-застосунку.

7. Реалізувати веб-застосунок в відповідності до поставлених вимог.
8. Провести мануальне та юзабіліті тестування веб-застосунку.

Наукова новизна – веб-застосунок має на меті задачу не тільки фіксування психологічного стану користувачем, а і допомагає первинно впоратись з емоційним навантаженням. Це має позитивний вплив на користувача та допомагає долати життєві виклики.

Практичне значення даного веб-застосунку полягає в доступності. Дана розробка може бути особливо корисною для користувачів, які не можуть найближчим часом отримати допомогу спеціалістом, але мають можливість полегшити власний емоційний стан за допомогою відслідковування власного настрою та навчання методам психологічної підтримки.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ІСНУЮЧИХ ІНСТРУМЕНТІВ САМОСТІЙНОГО ФІКСУВАННЯ ПСИХОЛОГІЧНОГО СТАНУ КОРИСТУВАЧЕМ ТА ПСИХОЛОГІЧНОЇ ПІДТРИМКИ

1.1 Аналіз предметної галузі

У сучасному світі, де інтенсивне навантаження, великий обсяг інформації та нові виклики стали нормою, турбота про психологічне благополуччя набуває великого значення. Це питання є важливим не лише для окремих користувачів, але й для великих корпоративних організацій, державних установ, таких як університети та органи, що опікуються соціально вразливими групами.

Розглядаючи поширення теми ментального здоров’я в Україні, варто зазначити, що до початку повномасштабної війни близько 30% українців мали розлади психічного здоров’я упродовж свого життя. За даними одного дослідження, понад 80% українців ніколи не зверталися до психолога або психотерапевта, хоча щонайменше третина з них нещодавно відчували дратівливість, проблеми зі сном, поганий настрій, напруженість та тривогу (дослідження охопило 2100 респондентів). Найчастіше вказаними перешкодами у доступі до послуг з охорони психічного здоров'я були вартість та доступність.

Щоб вирішити проблему доступності психологічної допомоги, з кінця 20 століття почали використовувати Інтернет для поширення інформації про психологічне здоров'я. Згодом у мережі з'явилися методи самодопомоги, гарячі лінії допомоги, що використовували електронну пошту. Починаючи з 2010-х років, для підтримки психологічного здоров’я стали застосовувати спеціалізовані мобільні додатки, відомі як додатки для ментального здоров’я. Ці додатки стали додатковим інструментом до традиційних психологічних консультацій і психотерапії, дозволяючи користувачам самостійно оцінювати свій стан,

отримувати інформацію про труднощі, знаходити ресурси підтримки та відстежувати зміни в психічному стані.

Після широкого поширення такого виду психологічної допомоги, Всесвітня організація охорони здоров’я визначила програмні засоби для психологічної підтримки як послуги з охорони психічного здоров'я, "підтримувані мобільними пристроями, такими як мобільні телефони, пристрої моніторингу пацієнта, персональні цифрові асистенти та інші бездротові пристрої". Вони включають додатки для смартфонів, голосові, відео або текстові повідомлення, відстеження в реальному часі та інтервенції на основі веб-технологій.

Оскільки мобільні додатки для психологічного здоров’я з’явилися відносно недавно, більшість досліджень присвячено інтернет-програмам. Веб-додатки для зменшення тривоги та депресії мають значну наукову підтримку щодо їх ефективності у лікуванні цих станів, особливо у поєднанні з роботою спеціаліста.

Найпопулярнішими додатками для підтримки психологічного здоров'я є ті, що відстежують емоційний стан (mood-tracking apps). Дослідження показують, що користувачі відчувають полегшення у самоусвідомленні та оцінці свого попереднього досвіду при використанні таких додатків. Проте, однією з проблем є відсутність рекомендацій або підказок щодо інтерпретації даних чи покращення настрою.

Цільова аудиторія розробок для підтримки психологічного стану охоплює користувачів віком від 18 до 40 років, переважно жінок. За даними одного дослідження, у якому взяли участь 129 осіб (63% з яких – жінки), користувачі діляться на дві вікові групи: "молодь" (18-24 роки) та "дорослі" (24+ років). Невелика перевага належить групі "дорослі" (59% опитаних), тоді як "молодь" складає 41%. Також відзначається, що 88% користувачів мають низький рівень задоволеності життям. Користувачі зазвичай користуються програмами для підтримки ментального здоров'я щодня, прагнучи покращити своє ментальне здоров’я, маючи депресію, тривожність, стрес та інші проблеми з психічним здоров’ям.

1.2 Аналіз існуючих інструментів самостійного фіксування психологічного стану користувачем та психологічної підтримки користувача

1.2.1 Аналіз інструментів та застосунків

Аналіз інструментів, що доступні для користувачів з метою самостійного моніторингу свого психологічного стану, показує наявність кількох основних категорій засобів:

- Ментальна рефлексія.
- Паперові методи.
- Електронні таблиці, як-от Excel, Google Sheets, Apple Numbers.
- Онлайн-ресурси та мобільні додатки.

Ментальна рефлексія є найпростішим, проте менш ефективним методом моніторингу психологічного стану. Через обмежену здатність людської пам'яті важко оцінити загальну ситуацію та визначити необхідні кроки для самодопомоги. По-перше, обмежена здатність людської пам'яті робить складним ретроспективний аналіз. Люди часто не здатні точно запам'ятати всі деталі своїх емоційних переживань за тривалий період, що ускладнює створення повної картини змін у настрої та психологічному стані. По-друге, відсутність записів ускладнює виявлення закономірностей і тригерів, які можуть впливати на емоційний стан. Без чіткої документації важко визначити, які саме події або дії призводять до покращення чи погіршення самопочуття. Крім того, ментальна рефлексія потребує певної дисципліни та самоусвідомлення, що не завжди легко підтримувати на регулярній основі. Без належної структурованості та систематичності цей метод може призвести до поверхневого аналізу та упущення важливих деталей. Отже, хоча ментальна рефлексія може бути корисною як додатковий інструмент для відслідковування психологічного стану, для досягнення більш об'єктивних і детальних результатів варто використовувати інші, більш структуровані, методи моніторингу.

Паперові методи включають як звичайні блокноти, що заповнюються вручну, так і спеціалізовані зошити з готовими шаблонами для заповнення. Використання

звичайних блокнотів дозволяє користувачу вести записи в будь-якій зручній формі, що сприяє вільному вираженню думок і почуттів. Однак такий підхід може бути менш структурованим, що ускладнює систематичний аналіз даних.

Спеціалізовані зошити з готовими шаблонами пропонують більш організований спосіб ведення записів. Вони можуть містити попередньо підготовлені розділи для щоденного фіксування настрою, активності, фізичного стану та інших важливих аспектів. Такі шаблони допомагають користувачу не лише документувати свої емоції та події, але й робити це у стандартизованій формі, що спрощує подальший аналіз. Приклад блокноту з готовими шаблонами для фіксування настрою зображено на рисунку 1.1.

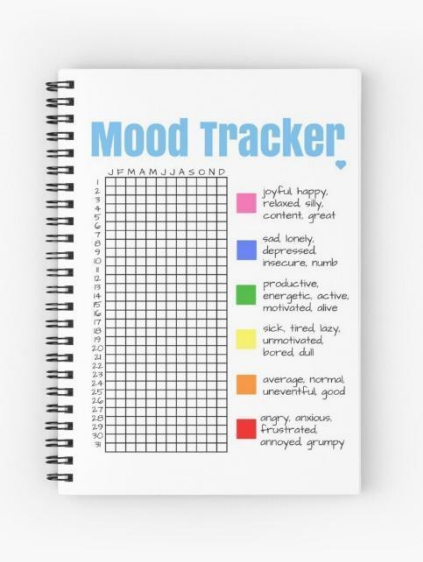


Рис.1.1 Блокнот з готовими шаблонами для фіксування настрою

Проте паперові методи мають і свої недоліки порівняно з програмними розробками. По-перше, вони не забезпечують можливості швидкого пошуку та систематизації інформації, що доступно в електронних системах. По-друге, відсутність автоматичного аналізу та візуалізації даних обмежує можливості для глибокого розуміння емоційних патернів. Нарешті, фізичні записи можуть бути втрачені або пошкоджені, що ставить під загрозу збереження важливої інформації. Таким чином, паперові методи, хоча і мають свої переваги, такі як глибоке

саморефлексування, поступаються програмним рішенням у питанні ефективності обробки та збереження даних.

Використання електронних таблиць, таких як Excel, Google Sheets, Apple Numbers, вирішує більшість проблем, пов'язаних із паперовими методами, зокрема редагування, аналіз, витрати, екологічність і захист інформації. Можна створювати шаблони самостійно або завантажувати готові з інтернету, що значно скорочує час на підготовку та організацію простору для психологічної самопідтримки. Приклад щоденника емоцій у Google Sheets зображено на рисунку 1.2.

Переваги використання електронних таблиць для самостійного фіксування психологічного стану та психологічної підтримки користувача:

– Швидкий та простий процес створення власного шаблону.

– Можливість використовувати розрахунки за допомогою формул в електронній таблиці.

– Автоматичне встановлення часу заповнення інформації.

– Графічне представлення даних для аналізу загальної картини емоційного стану.

– Автоматизовані комірки для більш ефективного та швидкого заповнення інформації.

Електронні таблиці підвищують зручність та ефективність моніторингу психологічного стану. Автоматизація процесу заповнення та аналізу зменшує ймовірність помилок і економить час. Однак, незважаючи на всі переваги, існують певні обмеження, пов'язані з використанням електронних таблиць. Недоліки використання електронних таблиць для самостійного фіксування психологічного стану та психологічної підтримки користувача:

– Використання електронних таблиць вимагає базових знань роботи з такими програмами, а створення власних шаблонів потребує спеціальних навичок.

– Кожний новий період заповнення необхідно копіювати та налаштовувати самостійно.

– Великі таблиці створюють значне навантаження на комп’ютер та ускладнюють аналіз.

– Відсутність порад і підтримки користувача в психологічних питаннях.

Онлайн-ресурс для психологічної підтримки Moodtracker.com пропонує широкий спектр корисних функцій, включаючи відмітку настрою, оцінку рівня тривожності, ведення записів про кількість годин сну та особистий щоденник. Однак, дизайн платформи може здатися складним для нових користувачів, що потенційно знижує їхню мотивацію до регулярного використання ресурсу. На рисунку 1.3 зображено сторінку фіксування настрою на онлайн-ресурсі Moodtracker.com.

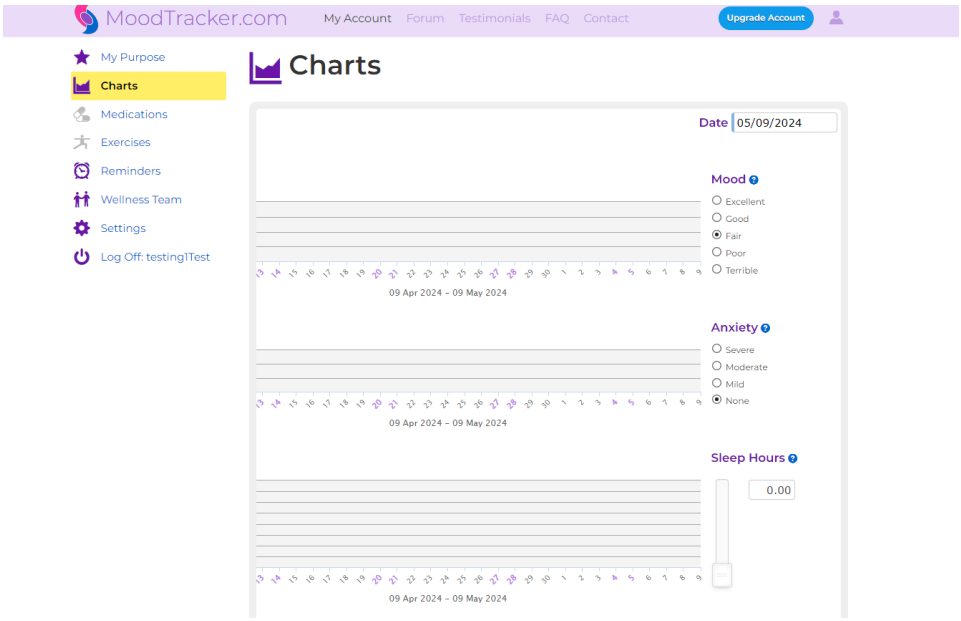


Рис. 1.3 Онлайн-ресурс Moodtracker.com

Moodtracker.com є досить комплексним ресурсом з великою кількістю опцій для самостійного фіксування настрою, що може ускладнити його використання для новачків, які раніше не користувалися інструментами для самостійного моніторингу психологічного стану. Переваги онлайн-ресурсу Moodtracker.com:

– Різноманітність функцій для моніторингу настрою та здоров'я;

– Можливість ведення особистого щоденника;

– Встановлення особистих цілей для підтримання дисципліни.

Недоліки онлайн-ресурсу Moodtracker.com:

– Складний дизайн інтерфейсу, який може бути незручним для новачків;

– Можливість перенасичення інформацією для користувачів, які не мають досвіду користування програмними засобами такого виду.

Таким чином, хоча Moodtracker.com пропонує багатий набір інструментів для психологічного моніторингу, його складний дизайн та відсутність персоналізованої допомоги можуть суттєво знижувати його ефективність і привабливість для користувачів.

Онлайн-ресурс eMoods вирізняється сучасним інтерфейсом. Сервіс пропонує розширені налаштування, що дозволяють вмикати або вимикати окремі елементи для трекінгу. Наприклад, можна активувати або деактивувати трекінг рівня тривожності залежно від потреби. Однак, така гнучкість часто призводить до перевантаження інформацією, що ускладнює використання сервісу для тих, хто не має досвіду в моніторингу психологічного стану. Новачки можуть відчувати труднощі через велику кількість опцій та функцій, що потребують значних зусиль для освоєння. Важливо зазначити, що більшість функцій доступна користувачу з наявною платною підпискою на сервіс. На рисунку 1.4 представлено інтерфейс онлайн-ресурсу eMoods.

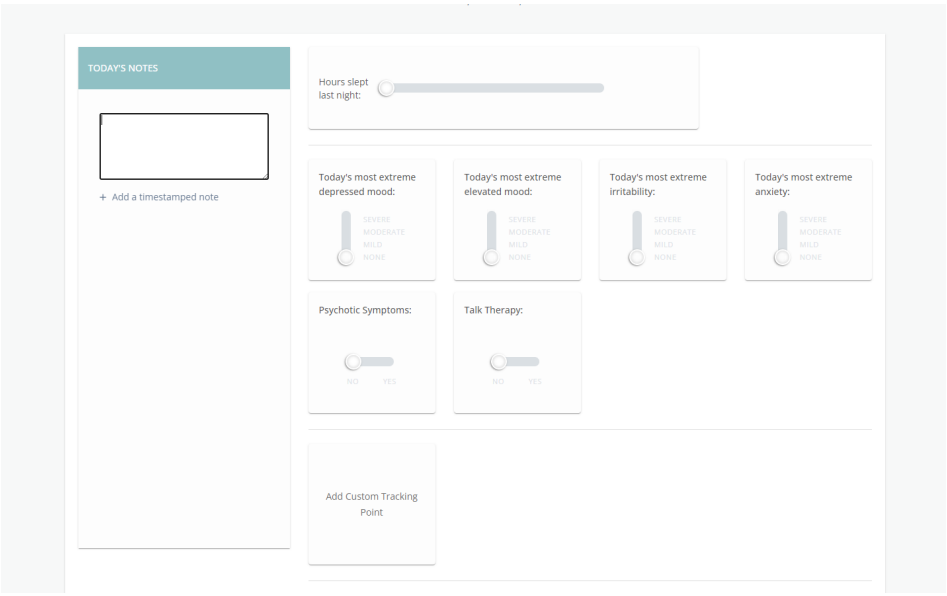


Рис. 1.4 Онлайн-ресурс eMoods.

Переваги онлайн-ресурсу eMoods:

– ведення особистого щоденника для документування емоційного стану;
– розширені налаштування для індивідуалізації трекінгу;
– сучасний дизайн, який робить платформу привабливо.

Недоліки онлайн-ресурсу eMoods:.

– вимога значного часу на налаштування та ознайомлення з усіма функціями;
– підвищений вхідний поріг через багатофункціональність платформи;
– висока складність використання для новачків, які не мають досвіду у веденні детального моніторингу;

Таким чином, eMoods пропонує широкий спектр можливостей для моніторингу психологічного стану, зокрема ведення особистого щоденника та індивідуалізацію трекінгу. Однак його складний інтерфейс та багатофункціональність можуть значно ускладнити використання для багатьох користувачів.

Better Me: Mental Health - це мобільний застосунок, який пропонує інноваційний підхід до надання психологічної підтримки користувачам, зосереджуючись на медитації, дихальних вправах та самопізнанні в контексті турботи про своє психічне здоров'я. Застосунок створений для надання користувачам ефективних інструментів для зняття стресу та покращення загального психологічного благополуччя. Одним з основних розділів додатку є секція, що пропонує звуки, спрямовані на заспокоєння тривожності та поліпшення якості сну. Ці звуки допомагають користувачам розслабитися, зменшити стрес та покращити свій сон. Вкладка «Discovery» надає доступ до різних видів навчальних матеріалів та статей, які спрямовані на освоєння технік, що допомагають покращити психологічний стан користувача. На рисунку 1.5 зображено інтерфейс застосунку.

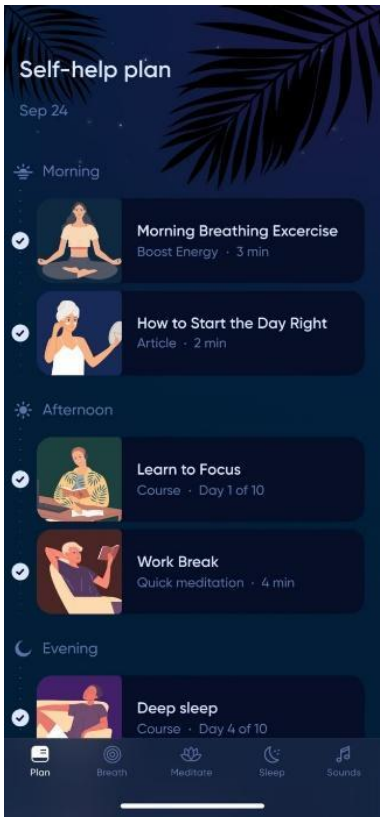


Рис. 1.5 Застосунок Better Me: Mental Health

Переваги додатку Better Me: Mental Health:

- інноваційний підхід до психологічної підтримки через медитацію та дихальні вправи;
- спеціальні звуки для заспокоєння тривожності та поліпшення якості сну;
- розділ «Discovery» з навчальними матеріалами та статтями для покращення психологічного стану.

Недоліки додатку Better Me: Mental Health:

- відсутність функціоналу самостійного фіксування психологічного стану;
- відсутність функціоналу нотатків.

Таким чином, Better Me: Mental Health є прикладом додатку для психологічної підтримки, який містить медитації та дихальні вправи, навчання, як спосіб підтримки користувача. Проте немає функціоналу фіксування

психологічного стану та нотатків, що може негативно впливати на процес психологічної підтримки користувача.

1.2.2 Результати проведення аналізу існуючих інструментів самостійного фіксування психологічного стану користувачем та психологічної підтримки користувача

Результати проведеного аналізу представлені у вигляді зведеної таблиці (табл. 1.1). Показниками обрано ключові функції застосунків.

Таблиця 1.1

Зведена таблиця результату аналізу існуючих інструментів

Показник	**moodtracker.com**	**eMoods**	**Better Me: Mental Health**
Платформи	**Web**	**Web**	**IOS, Android**
Трекер емоцій	+	+	-
Наявність медитацій	-	-	+
Щоденник текстовий	+	+	-
Щоденні поради	-	+	+
Навчання для самопідтримки	-	-	+
Зручність використання	-	-	+
Наявність української локалізації	-	-	+

З огляду на проведений аналіз існуючих аналогів веб-застосунок для щоденної психологічної підтримки користувача має мати наступні особливості:

1. трекер емоцій;

2. наявність медитацій;
3. можливість вести щоденник;
4. містити інформаційні ресурси з навчанням самопідримки;
5. бути зручним в використанні;
6. бути локалізованим української мовою.

1.3 Опис програмних засобів та мов програмування, які можуть бути використані для веб-застосунку з психологічної підтримки

Для розробки ефективного та інтуїтивно зрозумілого веб-застосунку, що надає психологічну підтримку, необхідно використовувати сучасні програмні засоби та мови програмування. Від правильного вибору технологій залежить функціональність, надійність та зручність використання застосунку. У цьому розділі ми розглянемо основні інструменти та технології, які можуть бути застосовані для створення такого веб-застосунку, зокрема мови програмування, фреймворки, бази даних та інструменти для аналізу даних.

1.3.1 Вибір мови Python програмування для back-end частини веб-застосунку

Для розробки back-end частини веб-застосунку з психологічної підтримки було обрано мову програмування Python у поєднанні з бібліотекою Django. Такий вибір обумовлений рядом переваг, які надає Python та його фреймворк Django для створення веб-застосунків. По-перше, Python відрізняється простотою та зрозумілістю коду, що дозволяє швидко розробляти та підтримувати код. Це особливо важливо для забезпечення надійності та масштабованості веб-застосунку. По-друге, Python пропонує багату екосистему бібліотек для різних задач, включаючи обробку даних, машинне навчання, веб-розробку та багато іншого, що значно полегшує інтеграцію різних функціональних можливостей у застосунок. Крім того, Python широко застосовується в індустрії, що підтверджує його

надійність та ефективність, а також існує велика спільнота розробників, яка може надати підтримку та допомогу у вирішенні різних технічних питань.

Django як фреймворк високого рівня дозволяє швидко розпочати розробку веб-застосунку завдяки своєму зручному інтерфейсу та готовим до використання компонентам. Він забезпечує зручну роботу з базами даних, підтримуючи різні системи управління базами даних (PostgreSQL, MySQL, SQLite). Також Django забезпечує можливість масштабування застосунку відповідно до зростання навантаження, що є важливим аспектом для довготривалої підтримки та розвитку проекту. Завдяки поєднанню переваг Python та Django, розробка back-end частини веб-застосунку з психологічної підтримки стає ефективною, надійною та зручною, що дозволяє швидко реалізувати необхідну функціональність та забезпечити високу якість роботи застосунку.

1.3.2 Вибір інструментів для створення front-end частини додатку

Для розробки front-end частини веб-застосунку з психологічної підтримки використовується HTML, CSS, JavaScript та була обрана бібліотека React.js. Ці технології забезпечують створення інтуїтивно зрозумілого та динамічного інтерфейсу користувача, що є критично важливим для досягнення цілей застосунку.

HTML (Hyper Text Markup Language) використовується для створення структури веб-сторінок. Він дозволяє визначати різні елементи інтерфейсу, такі як заголовки, параграфи, кнопки та форми. Завдяки своїй простоті та універсальності, HTML є основою будь-якого веб-додатку. CSS (Cascading Style Sheets) відповідає за стильове оформлення веб-сторінок. Використовуючи CSS, можна задавати кольори, шрифти, розміри та розташування елементів на сторінці. Це дозволяє створювати естетично привабливий та зручний для користувачів інтерфейс, що сприяє кращій взаємодії з застосунком. JavaScript використовується для додавання динамічності та інтерактивності до веб-сторінок. Завдяки JavaScript можна реалізувати різні функції, такі як валідація форм, динамічне оновлення контенту та

взаємодія з сервером без перезавантаження сторінки. Це забезпечує більш зручний та швидкий досвід для користувачів.

React.js, як одна з найпопулярніших бібліотек JavaScript, дозволяє створювати складні та високоефективні користувацькі інтерфейси. Використовуючи компоненти React, можна розробляти багаторазові та добре структуровані частини інтерфейсу, що значно полегшує підтримку та розширення застосунку.

1.3.3 Вибір системи управління бази даними для веб-застосунку

Для управління даними у веб-застосунку з психологічної підтримки було обрано систему управління базами даних MySQL. Цей вибір обґрунтований низкою важливих переваг, які пропонує MySQL. По-перше, вона забезпечує високу продуктивність та здатність обробляти великі обсяги даних, що є критично важливим для застосунку з багатьма користувачами. По-друге, MySQL відома своєю надійністю та безпекою, маючи вбудовані механізми резервного копіювання та відновлення даних, а також підтримуючи різні методи аутентифікації та авторизації, що захищає конфіденційну інформацію користувачів. MySQL є гнучкою та адаптивною системою, яка підтримує різні типи даних і дозволяє створювати складні структури баз даних, що сприяє легкому масштабуванню у майбутньому. Нарешті, велика спільнота розробників і широке розповсюдження MySQL забезпечують доступ до численних ресурсів та підтримки, що полегшує вирішення будь-яких технічних питань.

Django, як потужний фреймворк для веб-розробки, має вбудовану підтримку для роботи з MySQL, що робить інтеграцію цих технологій безпроблемною. Використання Django з MySQL забезпечує зручне та ефективне маніпулювання даними завдяки ORM (Object-Relational Mapping), що дозволяє працювати з базою даних, використовуючи об'єкти Python, спрощуючи розробку та підтримку коду. Крім того, Django надає інструменти для міграції баз даних, що полегшує управління змінами в структурі бази даних. Велика спільнота розробників і широке

розповсюдження MySQL забезпечують доступ до численних ресурсів та підтримки, що полегшує вирішення будь-яких технічних питань.

1.3.4 Програмні засоби для розробки веб-застосунку

Для розробки веб-застосунку з психологічної підтримки було використано ряд програмних засобів, серед яких основними є Visual Studio Code (VSCode) та Figma. Ці інструменти забезпечують ефективне середовище для написання коду, проектування інтерфейсу користувача та колаборації між розробниками та дизайнерами.

Visual Studio Code (VSCode) є одним з найпопулярніших редакторів коду, відомим своєю гнучкістю та розширюваністю. VSCode підтримує широкий спектр мов програмування і пропонує безліч розширень, що дозволяють налаштувати редактор під конкретні потреби проекту. Інструмент також надає зручні функції для дебагінгу, інтеграції з системами контролю версій, такими як Git, і потужні засоби для роботи з терміналом. Завдяки своїй продуктивності та зручності у використанні, VSCode значно спрощує процес написання та тестування коду. На рисунку зображено інтерфейс редактору коду Visual Studio Code.

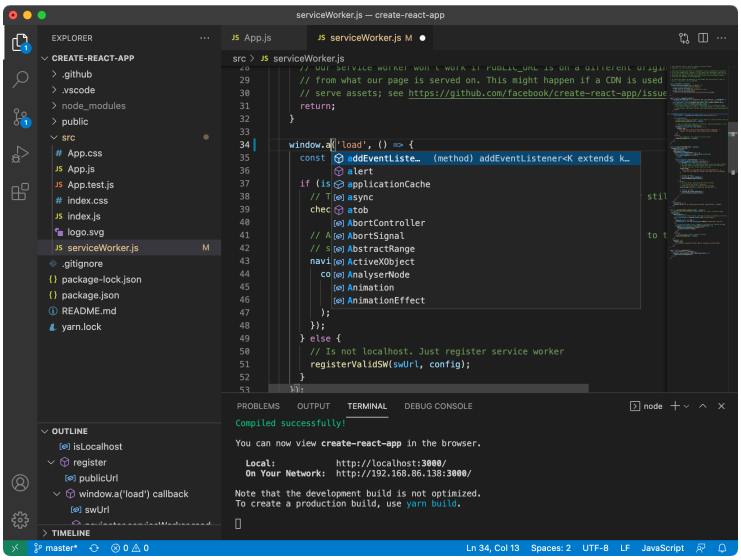


Рис. 1.6. Інтерфейс редактору коду Visual Studio Code

Figma є сучасним інструментом для дизайну, який використовується для створення макетів та прототипів інтерфейсів користувача. Figma надає потужні засоби для створення інтерактивних прототипів, що допомагає краще уявити кінцевий вигляд і функціональність застосунку ще на етапі проектування. Крім того, інструмент підтримує імпорт та експорт різних форматів файлів, що робить його універсальним рішенням для дизайну веб-застосунків. На рисунку 1.7 зображено інтерфейс Figma.

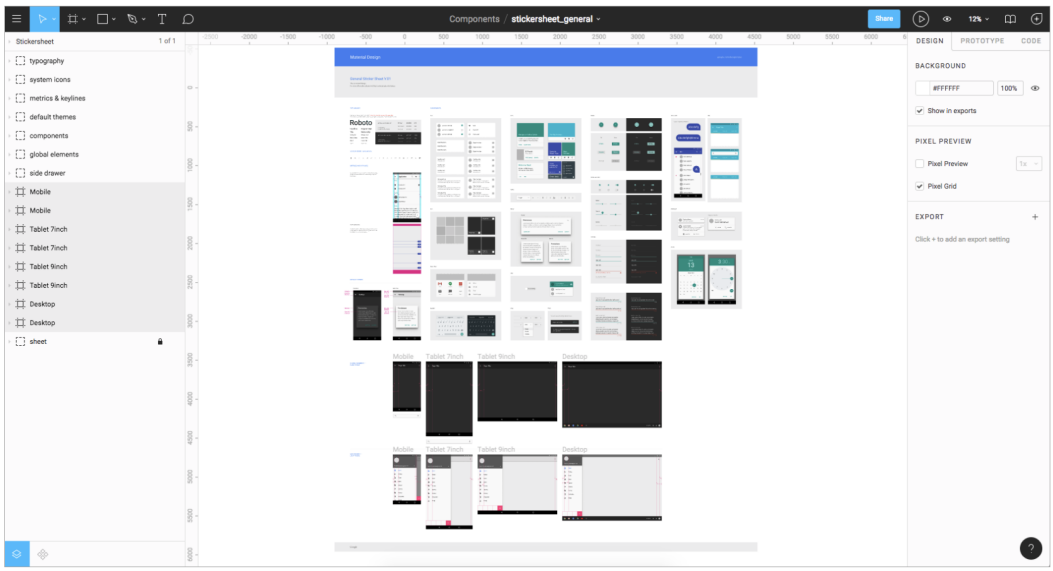


Рис. 1.7 Інтерфейс Figma

Використання Visual Studio Code та Figma дозволило створити продуктивне та зручне середовище для розробки веб-застосунку з щоденної психологічної підтримки користувача.

2 ПРОЕКТУВАННЯ ВЕБ-ЗАСТОСУНКУ ДЛЯ ЩОДЕННОЇ ПСИХОЛОГІЧНОЇ ПІДТРИМКИ КОРИСТУВАЧА

2.1 Постановка функціональних і нефункціональних вимог до веб-застосунку

Функціональні вимоги до веб-застосунку для щоденної психологічної підтримки DailyMindfullness включають такі функціональні вимоги:

1. Реєстрація та авторизація користувача.
2. Запис даних про емоційний стан користувача.
3. Можливість вести щоденник записів.
4. Можливість прослуховування медитацій.
5. Представлення даних, внесених користувачем за тиждень, місяць та рік.
6. Отримання щоденних порад для покращення психологічного самопочуття.
7. Можливість переглянути статті з методами психологічної самопідтримки.

Функціональні вимоги, які відрізняють веб-застосунок DailyMindfulness від аналогів, включають поєднання трекеру емоцій, щоденника, а також можливості покращення психологічного стану за допомогою медитацій, порад і статей з методами психологічної самопідтримки.

Для реалізації роботи системи необхідно розділити ролі користувачів веб-застосунку на користувача та модератора. Користувачі матимуть доступ до всіх основних функцій, включаючи трекер емоцій, щоденник, медитації та навчальні статті. Модератори будуть відповідальні за додавання та перевірку контенту сторінок статті та медитації. На рисунку 2.1 зображено діаграму прецедентів для веб-застосунку щоденної психологічної підтримки DailyMindfulness.

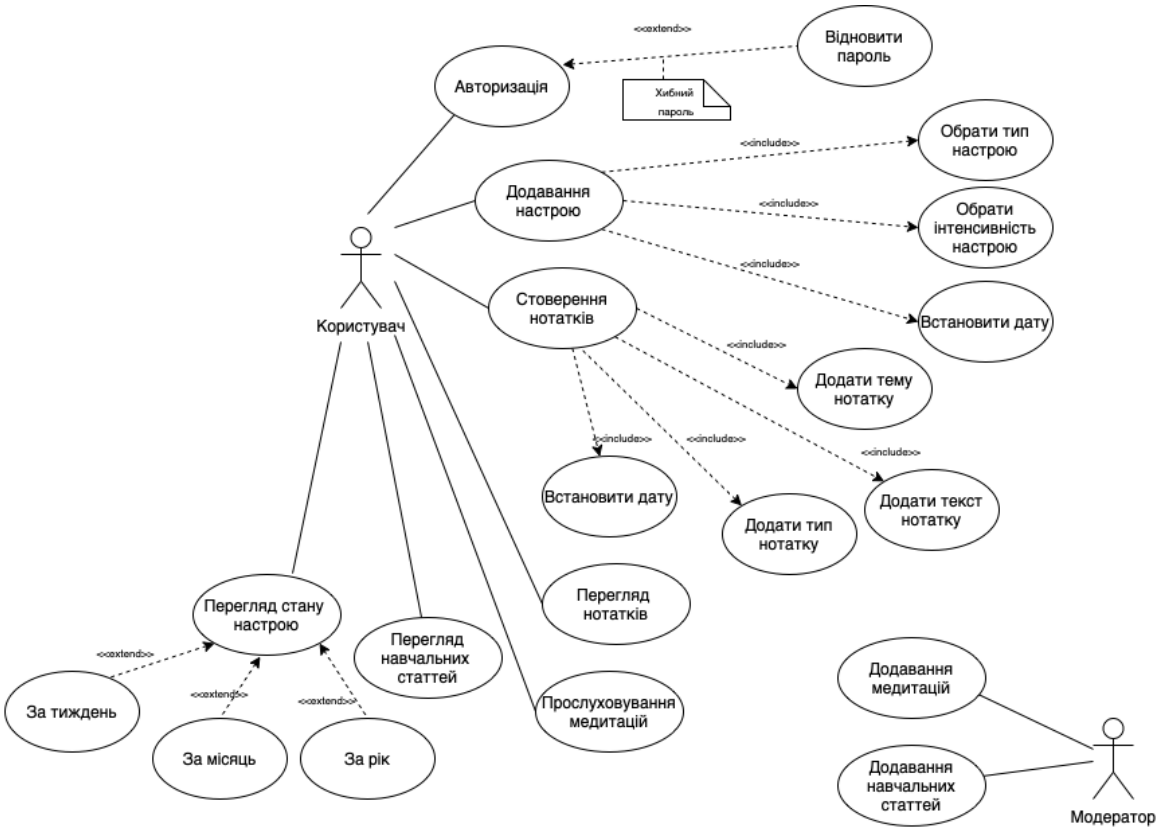


Рис. 2.1 Діаграма діяльності

Нижче описані прецеденти для ролі користувача:

Авторизація: включає реєстрацію та можливість відновлення паролю у випадку його втрати.

Додавання настрою: процес включає вибір одного з п’яти типів настрою, встановлення інтенсивності емоції від одного до п’яти, та зазначення дати заповнення.

Створення нотатки: прецедент включає наступні кроки - додавання теми нотатки, введення тексту, вибір типу нотатки, встановлення дати.

Перегляд нотаток: можливість перегляду вже створених нотаток.

Прослуховування медитацій: користувач може прослуховувати записи медитацій для різних цілей, таких як покращення сну, отримання натхнення, або для ранкової медитації.

Перегляд навчальних статей: користувач має доступ до навчальних статей на тему психологічної самопідтримки.

Перегляд стану настрою: користувач може переглядати стан настрою за допомогою діаграм за обрані періоди (тиждень, місяць, день).

До нефункціональних вимог належать наступні:

1. Інтерфейс інтуїтивно зрозумілий і доступний для користувачів з різним рівнем технічної підготовки.
2. Коректна робота на сучасних веб-браузерах (Google Chrome, Mozilla Firefox, Safari, Microsoft Edge).
3. Використання атласу емоцій Екмана: 5 типів емоцій (задоволення, злість, страх, відраза, смуток) з інтенсивністю від 1 до 5.

Для розробки інтуїтивно зрозумілого і доступного інтерфейсу використовується принципи UI/UX дизайну, а саме проведення дослідження на етапі розробки дизайну сайту та usability testing для етапу тестування готового веб-застосунку.

Коректна робота на сучасних веб-браузерах важлива для забезпечення зручного доступу для користувачів без додаткових вимог з боку користувача.

Використання атласу емоцій Екмана є ключовою вимогою для цього веб-застосунку, що відрізняє його від аналогів. Модель емоцій, розроблена Полом Екманом та Євою Екман, покликана поглибити наше розуміння того, як емоції впливають на нашу поведінку та мову. Для створення атласу емоцій Екмана було проведено опитування серед 149 науковців (фахівців з емоцій, нейробіологів і психологів), які є визнаними експертами у своїх галузях, щоб досягти консенсусу щодо природи емоцій та станів, які вони викликають. 88% дослідників погодилися з тим, що існують універсальні емоції — ті, що є спільними для всіх людей, незалежно від культурного та соціального контексту: гнів, страх, відраза, смуток та задоволення. Для застосунку DailyMindfulness ця модель використовується у спрощеному форматі, де залишаються п’ять основних типів емоцій, визначених Полом Екманом, з додаванням шкали інтенсивності від одного до п’яти. Такий підхід дозволить користувачам, навіть без досвіду використання трекерів емоцій, легко користуватися веб-застосунком DailyMindfulness.

2.2 Розробка дизайну веб-застосунку

Для розробки дизайну веб-застосунку для щоденної психологічної підтримки використовувались принципи User Interface / User Experience дизайну. UI/UX дизайн – це два окремі, але взаємопов'язані поняття в розробці цифрових продуктів. UI (інтерфейс користувача) дизайн стосується візуальних та інтерактивних елементів цифрового продукту, які користувачі бачать і з якими взаємодіють. Він включає в себе розробку макету, типографіки, кольорової схеми, іконок, кнопок та інших візуальних елементів, що складають інтерфейс користувача. Основна мета UI дизайну – створити візуально привабливий, інтуїтивно зрозумілий та зручний у користуванні інтерфейс, який дозволяє легко навігувати та взаємодіяти з продуктом.

З іншого боку, UX (досвід користувача) дизайн охоплює весь шлях користувача при взаємодії з цифровим продуктом. UX дизайн включає розробку загального користувацького досвіду та тестування зручності використання. Основна мета UX дизайну – створити безперебійний, приємний і змістовний користувацький досвід, який задовольняє потреби та очікування користувача.

Для створення дизайну використано програмний застосунок Figma.

2.2.1 Створенням прототипу веб-застосунку для щоденної підтримки користувача

Першим етапом виконання дизайну веб-застосунку для щоденної психологічної підтримки користувача являється розробка прототипу дизайну. Прототипування – це процес створення інтерактивної моделі дизайну, що дозволяє тестувати функціональність і зручність використання продукту до його програмної реалізації. Прототипування допомагає виявляти потенційні проблеми зручності використання та покращувати загальний дизайн продукту.

Результат виконання проектування сторінок “Головна сторінка”, “Додати настрій” для веб-застосунку для щоденної психологічної підтримки користувача

зображено на рисунку 2.2. Також було створено прототипи для сторінок: “Навчання”, “Медитації” та “Нотатки”.

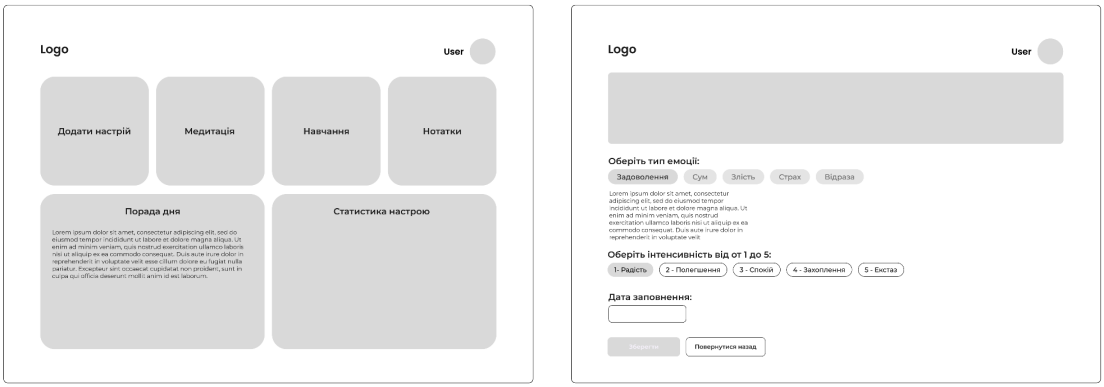
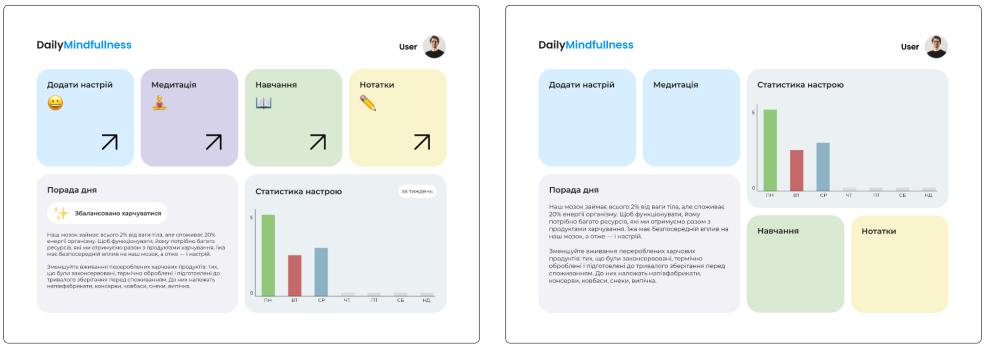


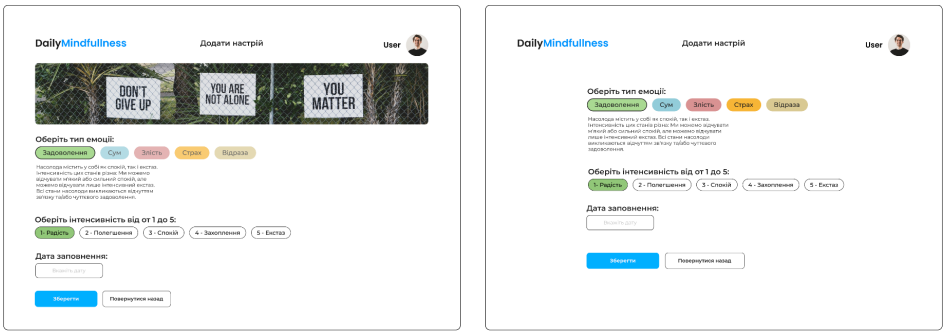
Рис. 2.2 Прототип сторінок “Головна сторінка” та “Додати настрій”

2.2.2 Створення проміжного дизайну веб-застосунку для щоденної підтримки користувача

Після завершення етапу створення прототипу дизайну веб-застосунку, розробляється проміжна версія дизайну сторінок для подальшого користувацького A/B тестування. Це дозволяє визначити найкращий варіант для програмної реалізації. На рисунку 2.3 зображено два варіанти сторінки “Головна сторінка” та на рисунку 2.4 два варіанти для сторінки “Додати настрій”. Також було створено варіанти дизайну для сторінок: “Навчання”, “Медитації” та “Нотатки”. Ці варіанти дозволяють протестувати різні підходи до дизайну та вибрати оптимальний для остаточної реалізації.



2.3 Два варіанти дизайну “Головної сторінки”



2.4 Два варіанти дизайну сторінки “Додати настрій”

2.2.3 A/B тестування дизайну веб-застосунку для щоденної психологічної підтримки користувача

A/B тестування, також відоме як спліт-тестування, є основною методологією, яка використовується в дослідженні користувацького досвіду для порівняння двох версій цифрового продукту з метою визначення, яка з них працює краще. Воно включає створення двох варіантів: A (контрольний) і B (змінений), та випадкове призначення користувачів для взаємодії з кожною з версій. Метою є визначення, який з варіантів дає кращі показники продуктивності, такі як вищі коефіцієнти конверсії, покращена взаємодія користувачів або більша задоволеність користувачів.

Для проведення A/B тестування веб-застосунку з щоденної психологічної допомоги використовувався опитувальник створених в Google Forms. Google Forms

– це безкоштовний онлайн-інструмент від Google, який дозволяє створювати опитування, анкети, вікторини та форми для збору даних.

Тестування проводилось анонімно та окрім A/B тестування включало наступні питання:

1. Оберіть Ваш рівень технічної обізнаності/підготовки (низький, середній, високий)
2. Скільки годин на день Ви проводите в інтернеті? (Більше 6, 3-5 годин, 1-2 години, менше години)
3. Чи користувались ви раніше веб-застосунками/застосунками для психологічної підтримки користувача? (Так або Ні)

В опитуванні взяли участь 10 осіб, серед яких: 40% мають середній рівень технічної обізнаності, 30% - високий рівень, та 30% - дуже низький рівень підготовки. Шість учасників опитування проводять в інтернеті 3-5 годин на день. Двоє учасників обрали варіант "більше 6 годин на день", а по одному учаснику відповіли "менше години" та "1-2 години". Досвід використання застосунків для психологічної підтримки розділився на рівні групи: 50% учасників мають досвід користування такими застосунками, а 50% - не мають.

На рисунку 2.5 зображено результати для тестування сторінки «Додати настрій». Зведена таблиця результатів A/B тестування наведена нижче (табл. 2.1).

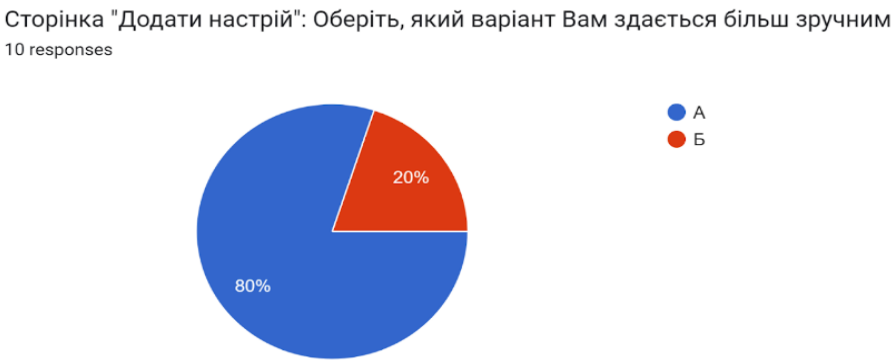


Рис. 2.5 Результат для сторінки «Додати настрій»

Таблиця 2.1

Зведена таблиця результатів A/B тестування

Назва Сторінки	А	Б
Головна Сторінка	70%	30%
Додати настрій	80%	20%
Медитації	70%	30%
Нотатки	80%	20%

2.2.4 Фінальний дизайн сторінок веб-застосунку

На основі результатів проведеного A/B тестування було створено остаточний дизайн макетів сторінок для веб-застосунку, призначеного для щоденної підтримки користувачів. На рисунку 2.6 зображені фінальні макети сторінок веб-застосунку, що відображають оптимізований дизайн, розроблений на підставі результатів тестування.

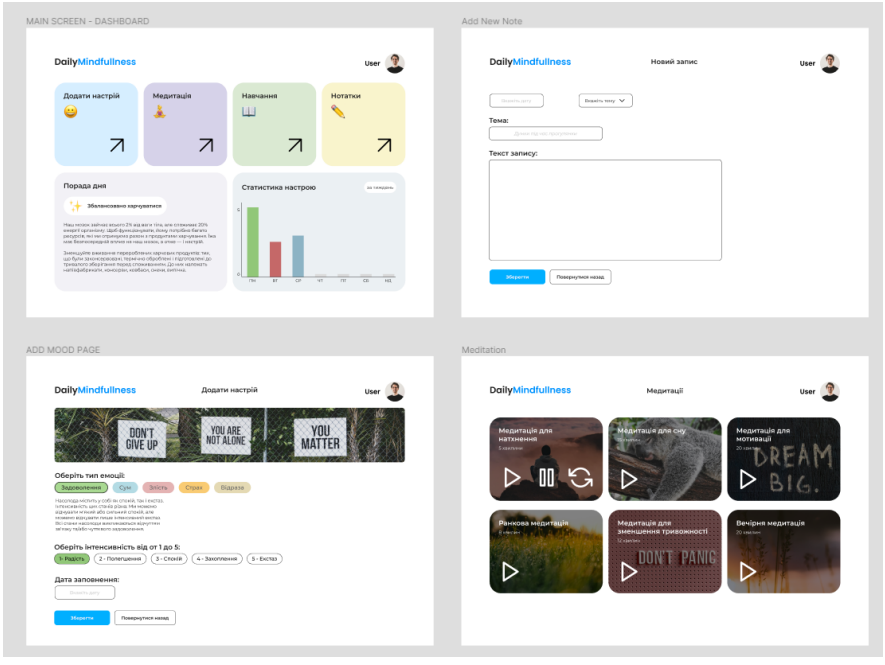


Рис 2.6 Остаточні макети для основних сторінок веб-застосунку

2.3 Проектування діаграми діяльності

Діаграма діяльності використовується в розробці програмного забезпечення для візуалізації послідовності дій і процесів, що виконуються в системі. Вона допомагає розробникам та аналітикам моделювати робочі потоки, показуючи, як завдання переходять від одного етапу до іншого і як взаємодіють різні компоненти системи. Використання діаграм діяльності дозволяє краще зрозуміти функціонування системи, виявити можливі вузькі місця та оптимізувати процеси. Вони також слугують ефективним засобом для комунікації між членами команди, забезпечуючи зрозумілу та наочну форму представлення інформації про процеси і їх послідовність. Крім того, діаграми діяльності є важливим інструментом для планування та контролю проектів, оскільки дозволяють стежити за прогресом виконання завдань. Вони допомагають визначити ключові етапи проекту та забезпечити своєчасне виконання кожного з них. Використання діаграм діяльності сприяє підвищенню прозорості проекту, що полегшує управління ресурсами та координацію роботи над проектом.

Наступним кроком у проектуванні веб-застосунку для психологічної підтримки стало створення діаграми діяльності, яка описує процеси, що відбуваються під час використання цього застосунку. На рисунку 2.7 представлено діаграму діяльності для веб-застосунку.

З діаграми видно, що багато процесів можна виконувати паралельно, наприклад, прослуховувати медитацію та читати статтю одночасно. Така гнучкість паралельного виконання завдань розширює можливості використання застосунку, роблячи його більш зручним і ефективним для користувачів. Крім того, діаграма дозволяє чітко визначити, які саме дії користувач може виконувати на кожному етапі, що сприяє покращенню загального розуміння структури та функціональності системи. Завдяки цьому легше ідентифікувати потенційні проблеми та вузькі місця, а також знайти способи для їх усунення, що в підсумку підвищує якість та надійність веб-застосунку.

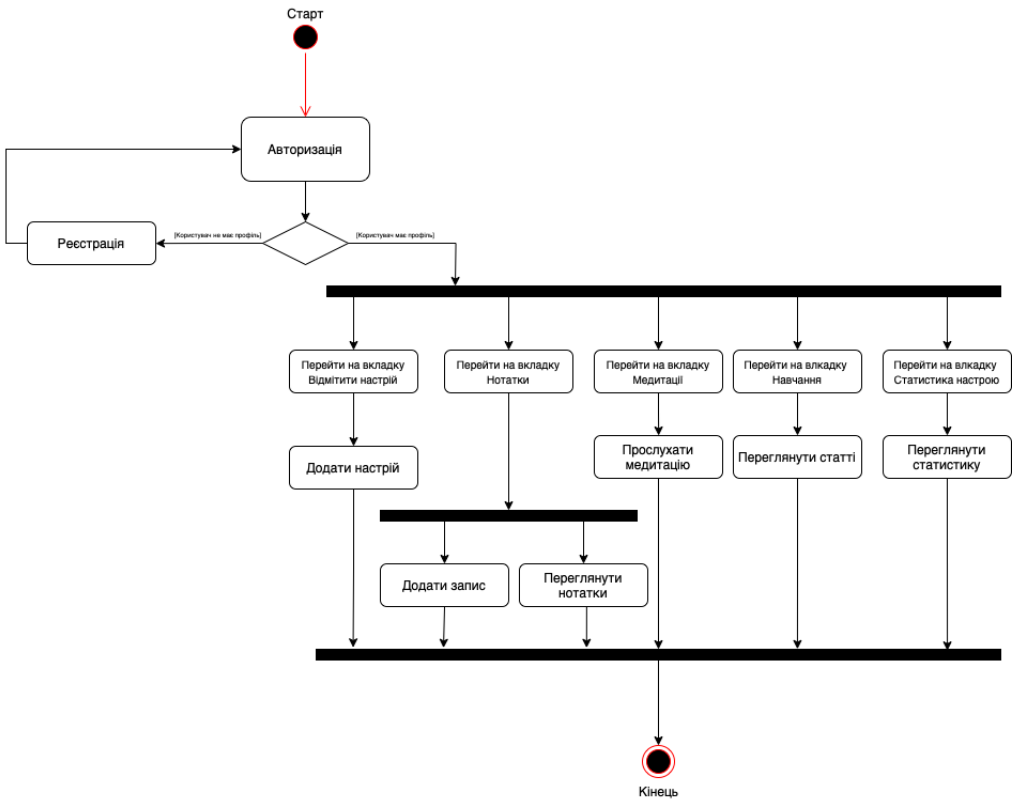


Рис 2.7 Діаграма діяльності веб-застосунку

2.4 Проектування карти сайту

Карти сайту є невід'ємною частиною сучасного веб-дизайну, виконуючи кілька ключових функцій, які значно покращують користувацький досвід та оптимізують роботу веб-ресурсу. Вони забезпечують зручну навігацію, дозволяючи користувачам швидко знаходити потрібну інформацію, надаючи чіткий огляд структури сайту та посилання на всі основні розділи і сторінки. Окрім цього, карти сайту важливі для пошукової оптимізації (SEO), оскільки вони допомагають пошуковим системам ефективніше індексувати сторінки сайту. Адміністратори сайтів також використовують карти для зручного управління контентом, відстежуючи структуру та здійснюючи необхідні оновлення. Карти сайту сприяють виявленню та виправленню структурних недоліків, що може покращити загальну функціональність сайту. Для веб-застосунку, призначеного для щоденної психологічної підтримки користувачів, карта сайту є важливим інструментом, який демонструє структуру сторінок додатку та значно покращує

навігацію. На рисунку 2.8 представлено карту сайту для цього веб-застосунку DailyMindfullness.

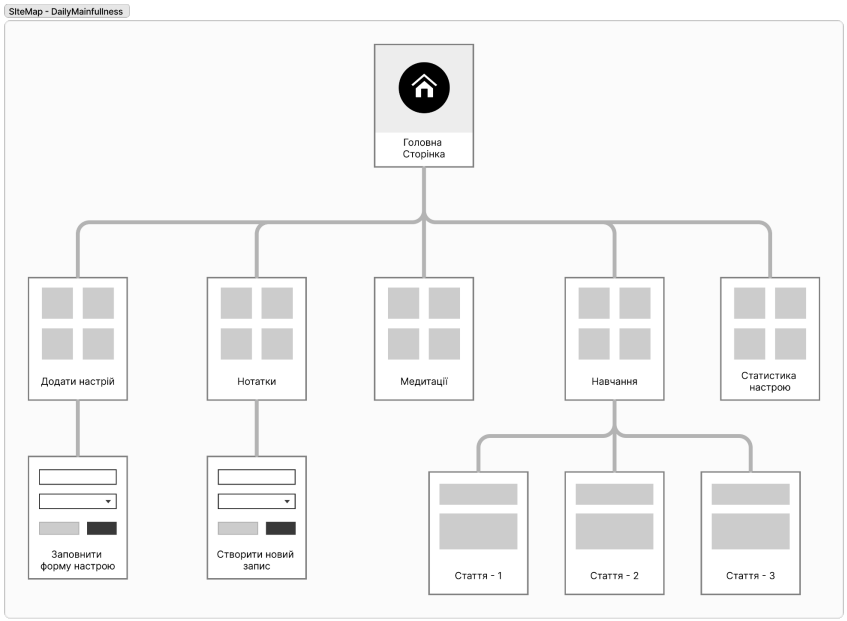


Рис. 2.8 Карта сайта для веб-застосунку DailyMindfullness

2.5 Проектування бази даних

Для веб-застосунку DailyMindfulness було розроблено базу даних за допомогою системи управління базами даних MySQL. Схема містить в собі три таблиці: Emotion, User, Notes. Зображення схеми бази даних для веб-додатку зображена на рисунку 2.9.

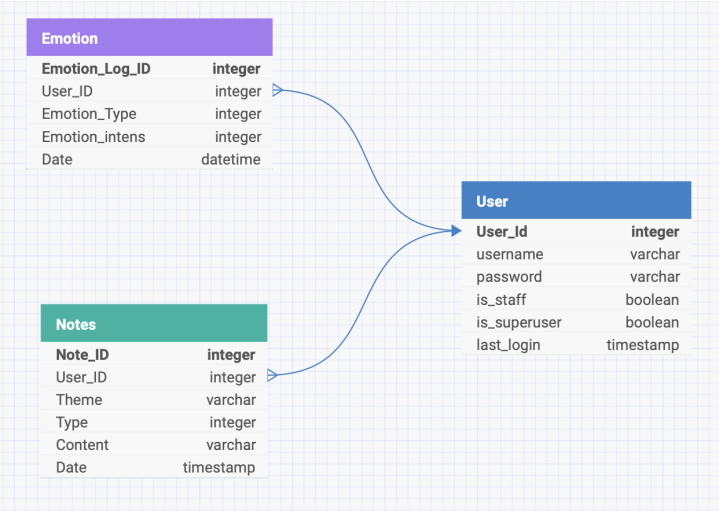


Рис. 2.9 Схема бази даних застосунку DailyMindfullness

Таблиця Emotion використовується для зберігання введених користувачем емоцій та містить 5 рядків для зберігання типу емоції, інтенсивності, дату введення емоції, додатково використовуються рядок для підключення таблиці до таблиці User - User_ID (Foreign Key) і рядок Emotion_Log_ID (Primary Key).

Таблиця Notes використовується для зберігання внесених користувачем нотатків. Містить в собі тему додатку, тип додатку, вміст додатку (контент) та дату. Note_ID - Primary Key, User_ID в свою чергу виконує функцію Foreign Key для з'єднання з таблицею User.

Таблиця User зберігає в собі дані про користувача, його пароль, логін, пароль та його роль в веб-застосунку за допомогою полів is_staff, is_superuser, зберігає дані про останній логін в веб-застосунок для контролювання неактивних профілів на сервісів. Всі з'єднання таблиці User з таблицями Emotion, Notes виконується через рядок Uder_Id (Primary Key) - тип з'єднання один до багатьох.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-ЗАСТОСУНКУ ДЛЯ ЩОДЕННОЇ ПСИХОЛОГІЧНОЇ ПІДТРИМКИ КОРИСТУВАЧА

3.1 Діаграма класів для веб-застосунку

Діаграма класів є важливою складовою об'єктно-орієнтованого моделювання і відіграє ключову роль у процесі розробки веб-застосунків. Вона надає візуальне представлення структури системи, показуючи класи, їхні атрибути, методи та відносини між ними. Використання діаграми класів дозволяє розробникам чітко зрозуміти внутрішню архітектуру програмного забезпечення, а також сприяє ефективному спілкуванню між членами команди.

У контексті розробки веб-застосунку для щоденної психологічної підтримки, діаграма класів допоможе визначити основні компоненти системи, такі як користувачі, їхні профілі, записи емоційних станів та функціональні можливості, що забезпечують роботу застосунку. Це дозволить створити стійку та гнучку архітектуру, яка легко адаптується до змінних вимог і забезпечує масштабованість у майбутньому. На рисунку 3.1 зображена діаграма класів для веб-застосунку щоденної психологічної підтримки користувача.

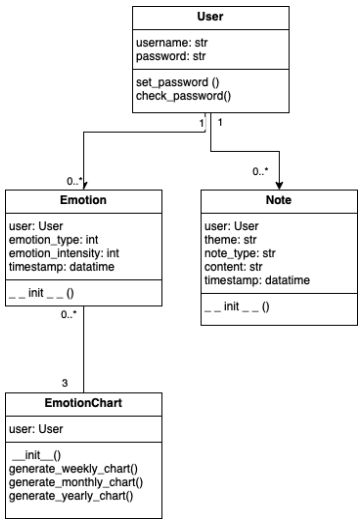


Рис. 3.1 Діаграма класів для веб-застосунку Daily Mindfulness

3.2 Опис функціонування веб-застосунку

3.2.1 Головна сторінка

Після авторизації у веб-застосунку для психологічної підтримки користувач бачить головний екран, який містить такі елементи: кнопки "Додати настрій", "Медитації", "Навчання", "Нотатки", блок "Порада дня" та "Статистика настрою". Кожна з цих кнопок веде на відповідні сторінки. Блок "Порада дня" оновлюється кожні 6 годин, надаючи нову пораду двічі на день. Блок "Статистика настрою" показує тижневу статистику та при натисканні переносить користувача на сторінку з детальною статистикою за тиждень, місяць та рік. Головний екран зображено на рисунку 3.1.

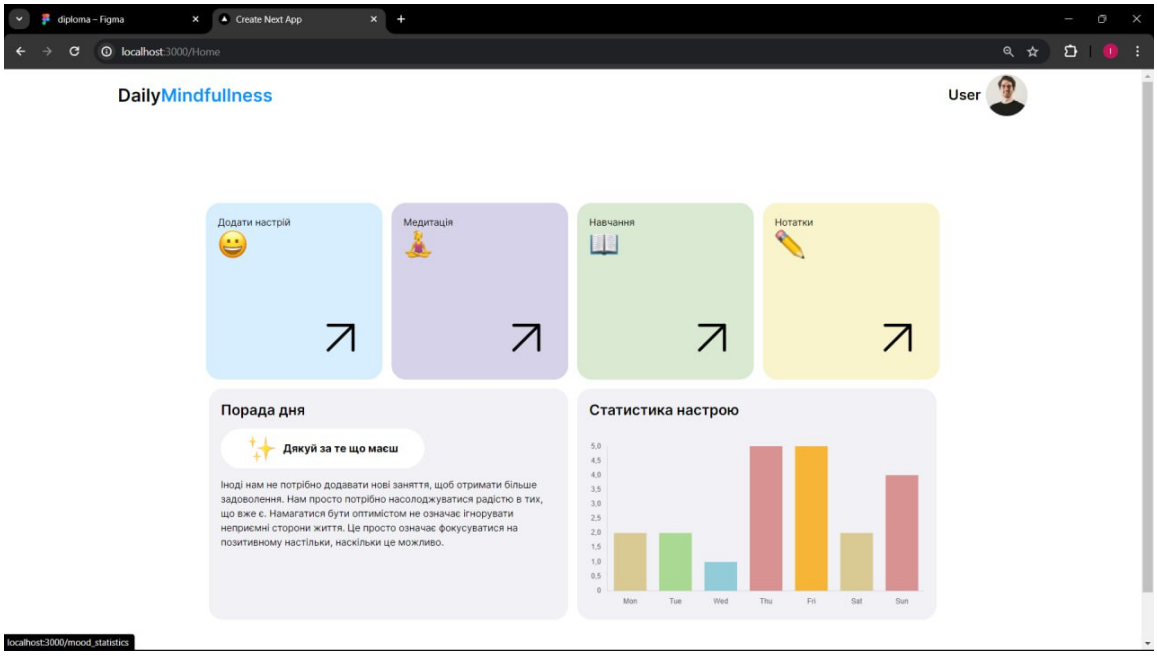


Рис. 3.2 Головний екран веб-застосунку

Код реалізації функціоналу оновлення для блоку «Порада дня» наведено на рисунку 3.3.

```python
class RandomAdviceView(APIView):
    def get(self, request):
        advices = Advice.objects.all()
        if not advices:
            return Response({"detail": "No advices found."}, status=status.HTTP_404_NOT_FOUND)
        random_advice = random.choice(advices)
        serializer = AdviceSerializer(random_advice)
        return Response(serializer.data)


class AddAdviceView(APIView):
    def post(self, request):
        serializer = AdviceSerializer(data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data, status=status.HTTP_201_CREATED)
        return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
```

Рис. 3.3 Код реалізації оновлення «Порада дня»

3.2.2 Сторінка “Додати настрій”

Сторінка "Додати настрій" призначена для щоденного фіксування поточного емоційного стану користувача. Для цього користувач повинен вибрати один з п'яти типів емоцій у блоці "Оберіть тип емоцій". Щоб полегшити вибір, під блоком надається підказка з описом кожної емоції. Після вибору типу емоції, наступний крок – вибір інтенсивності, яка оцінюється за шкалою від 1 до 5, де 5 – це найвища інтенсивність. Кожен рівень інтенсивності має відповідну назву, щоб спростити орієнтування. Завершальним кроком є встановлення дати заповнення. Користувач може вносити дані ретроспективно, наприклад, якщо протягом деякого часу не мав змоги використовувати застосунок і записував свої емоції іншим способом, а тепер бажає перенести їх у цифровий формат. Сторінка "Додати настрій" зображена на рисунку 3.4. Код реалізації для функції запису настрою зображено на рисунку 3.5.

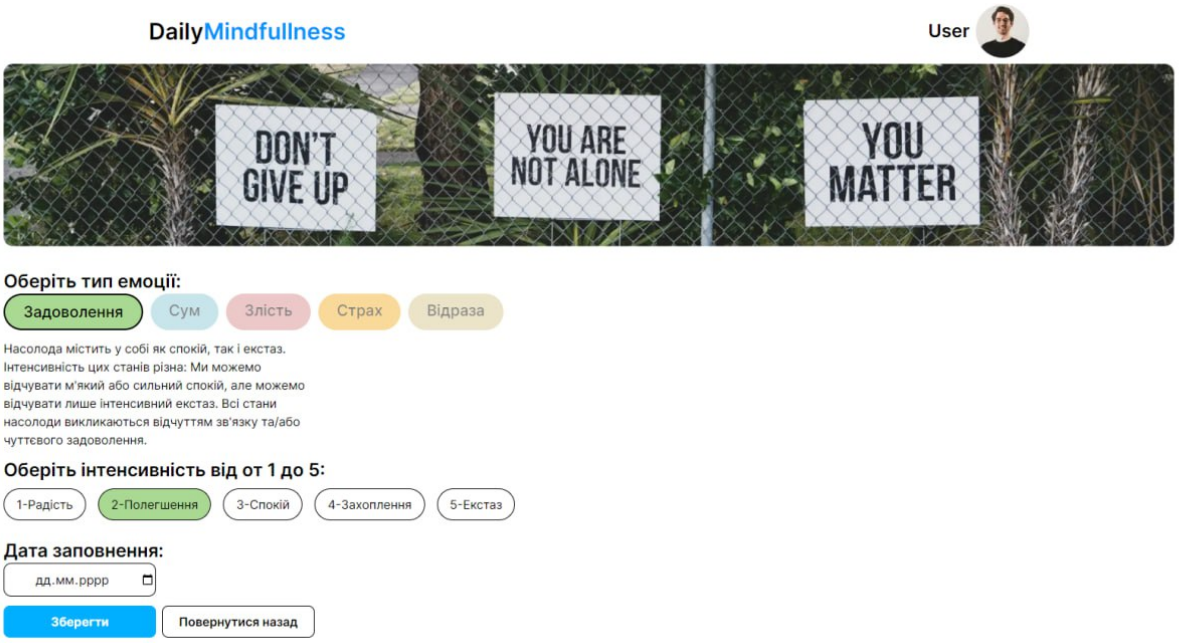


Рис. 3.4 Сторінка “Додати настрій”

```python
operations = [
    migrations.CreateModel(
        name="Mood",
        fields=[
            (
                "id",
                models.BigAutoField(
                    auto_created=True,
                    primary_key=True,
                    serialize=False,
                    verbose_name="ID",
                ),
            ),
            (
                "type_of_mood",
                models.CharField(
                    choices=[
                        ("Задоволення", "Задоволення"),
                        ("Сум", "Сум"),
                        ("Злість", "Злість"),
                        ("Страх", "Страх"),
                        ("Відраза", "Відраза"),
                    ],
                    max_length=255,
                ),
            ),
            (
                "rate",
                models.IntegerField(
                    validators=[
                        django.core.validators.MinValueValidator(1),
                        django.core.validators.MaxValueValidator(5),
                    ]
                ),
            ),
            ("date", models.DateField()),
        ],
    ),
]
```

Рис. 3.5 Реалізація запису настрою

3.2.3 Сторінка “Медитація”

Сторінка "Медитації" призначена для прослуховування попередньо завантажених модератором медитацій. На рисунку 3.6 показана ця сторінка. Вона містить блоки, кожен з яких відповідає окремій медитації. У верхній частині блоку вказано назву медитації, яка відображає її призначення, а також тривалість у хвилинах. У нижній частині блоку розміщені елементи управління: кнопки для початку, паузи та перезапуску запису. Реалізація блоку медитацій зображена на малюнку 3.7.

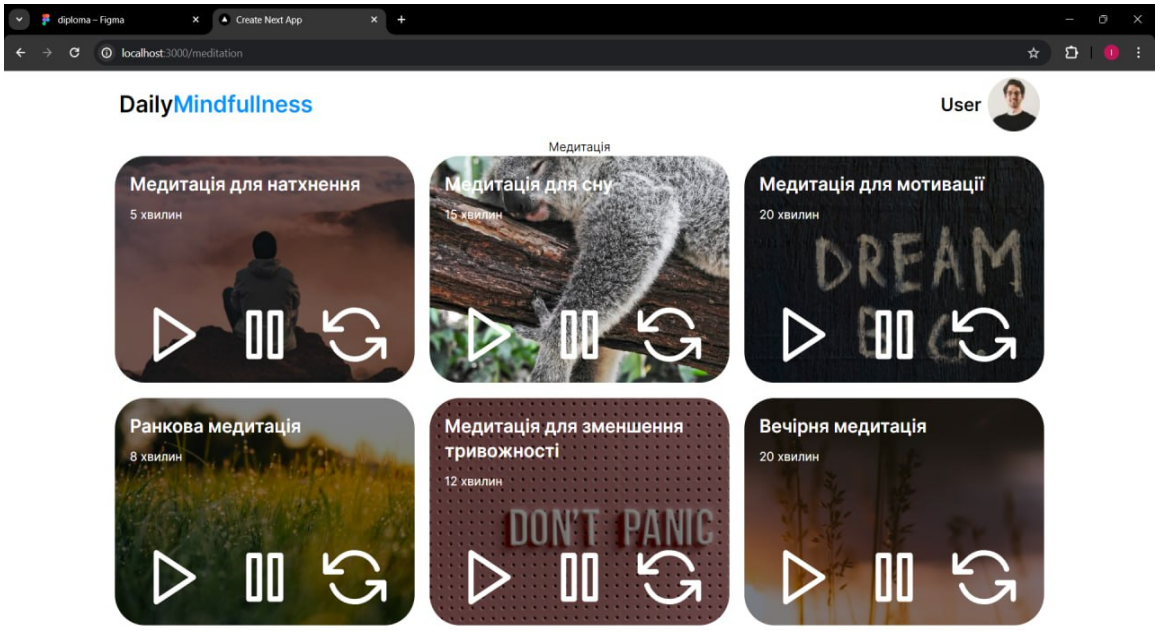


Рис. 3.6 Сторінка “Медитація”

```
const page = () => {
  const [ID, setID] = useState(null);
  const audioList = [
    {
      id: 1,
      title: 'Медитація для натхнення',
      durationtime: '5 хвилин',
      url: new Audio(meditation_inspire),
      bg: boy.src,
    },
    {
      id: 2,
      title: 'Медитація для сну',
      durationtime: '15 хвилин',
      url: new Audio(sleep_meditation),
      bg: coala.src,
    },
    {
      id: 3,
      title: 'Медитація для мотивації',
      durationtime: '20 хвилин',
      url: new Audio(meditation_motivation),
      bg: dreamBig.src,
    },
    {
      id: 4,
      title: 'Ранкова медитація',
      url: new Audio(morning_meditation),
      durationtime: '8 хвилин',
      bg: grass.src,
    },
    {
      id: 5,
      title: 'Медитація для зменшення тривожності',
      url: new Audio(meditation_anxiety),
      durationtime: '12 хвилин',
      bg: dontPanic.src,
    },
    {
      id: 6,
      title: 'Вечірня медитація',
      url: new Audio(meditation_evening),
      durationtime: '20 хвилин',
      bg: evening.src,
    },
  ];
```

Рис. 3.7 Реалізація блоку медитації

3.2.3 Сторінка «Додати нотатки»

Після переходу на сторінку «Нотатки» користувач потрапляє на сторінку зі всіма своїми нотатками. Ця сторінка зображена на рисунку 3.8. Кожна нотатка містить свою тему, яку користувач може вводити самостійно, а також тип нотатки. На зображенні представлені типи нотаток, такі як «мрії», «хобі», «подорожі». В лівому верхньому кутку зображена дата створення запису. При натисканні на блок з нотаткою відкривається повний текст запису, який можна переглянути (рис. 3.9). Натиснувши на блок «Додати новий запис», відкривається вікно для додавання нової нотатки, яке зображене на рисунку 3.10.

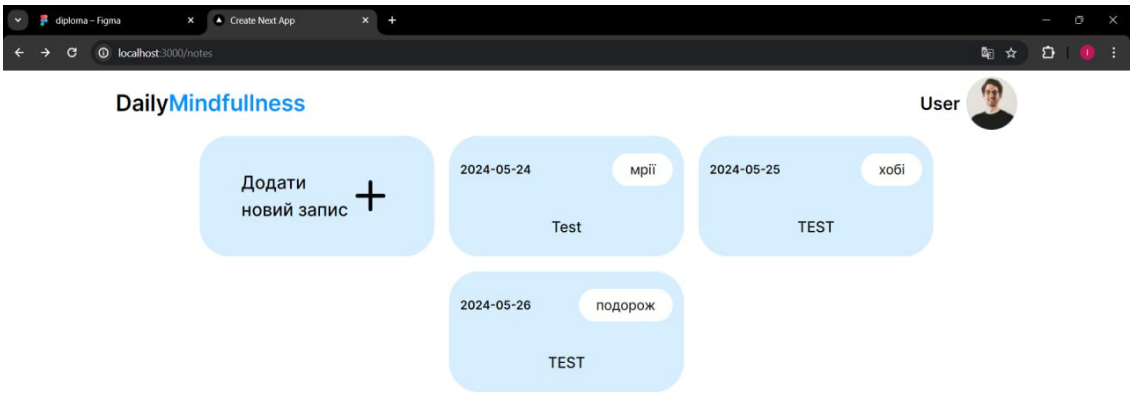


Рис. 3.8 Сторінка «Нотатки»

Сторінка «Додати новий запис» містить наступні елементи: заповнення дати нотатку, категорію запису, тему запису, текст запису. Після заповнення полів користувач натискає «Зберегти» або «Повернутися назад». Після збереження нотатки користувач повертається на сторінку з всіма нотатками. Реалізація функції нотаток зображена на рисунку 3.11.

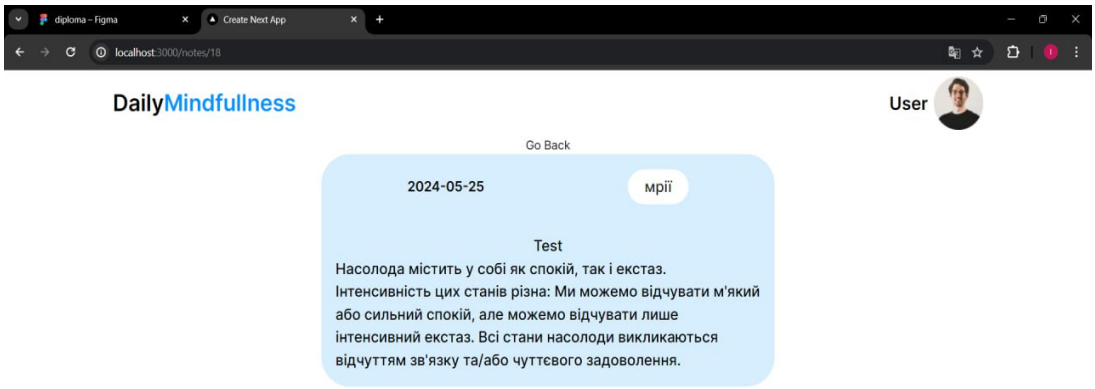


Рис 3.9 Сторінка нотатку

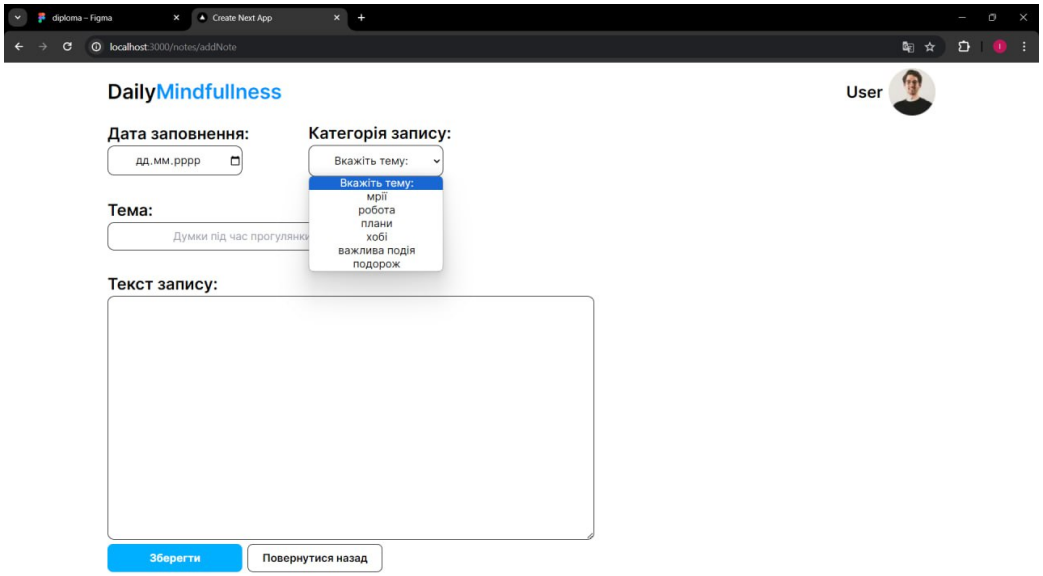


Рис 3.10 Сторінка «Створити новий запис»

```
class AddNoteView(APIView):
    def post(self, request):
        serializer = NoteSerializer(data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data, status=status.HTTP_201_CREATED)
        return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)


class NoteListView(APIView):
    def get(self, request):
        notes = Note.objects.all()
        serializer = NoteListSerializer(notes, many=True)
        return Response(serializer.data)


class NoteDetailView(generics.RetrieveAPIView):
    queryset = Note.objects.all()
    serializer_class = NoteSerializer
    lookup_field = 'id'
```

Рис 3.11 Реалізація функції нотаток

3.2.3 Сторінка статистики настрою

Сторінка статистики відкривається після натискання на блок статистики на головній сторінці. Вона надає можливість переглядати дані про відмічений настрій за тиждень, місяць та рік. Для місячної та річної статистики відображаються середні значення настрою, які зазначав користувач. На рисунку 3.12 представлено інтерфейс сторінки статистики, що демонструє інтуїтивний дизайн для аналізу

особистих даних користувача. На рисунку 3.13 показана реалізація сторінки статистика.

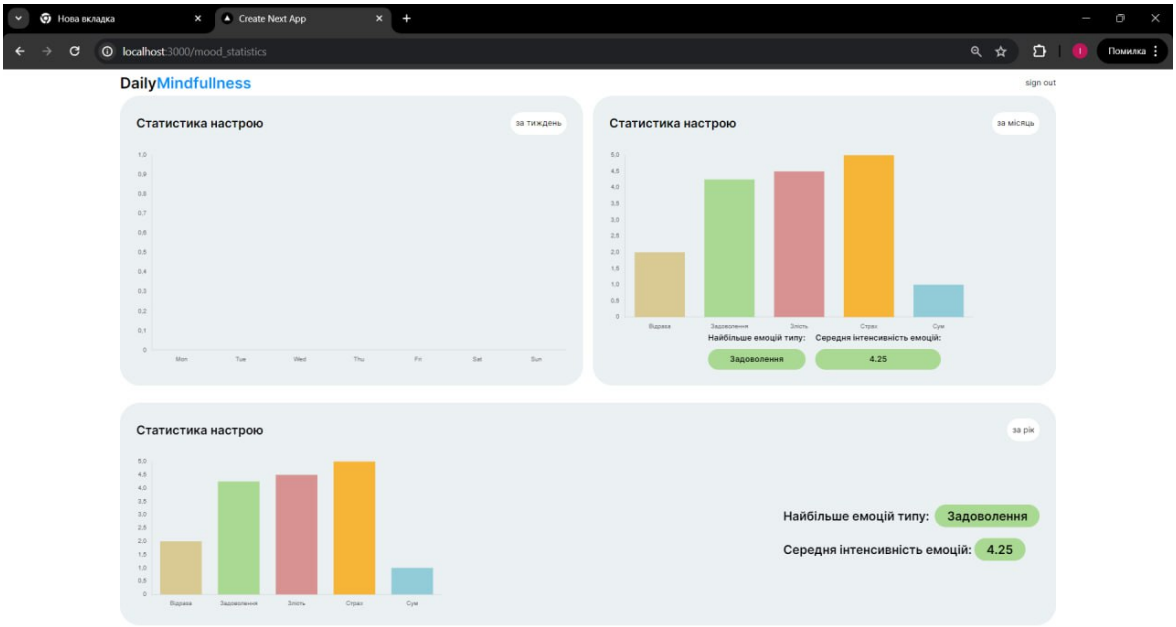


Рис. 3.12 Сторінка статистики настрою

```python
def get_queryset(self):
    queryset = Mood.objects.all()
    start_date_param = self.request.query_params.get('start_date', None)
    end_date_param = self.request.query_params.get('end_date', None)
    if start_date_param and end_date_param:
        start_date = parse_date(start_date_param)
        end_date = parse_date(end_date_param)
        if start_date and end_date:
            queryset = queryset.filter(date__range=[start_date, end_date])
        else:
            queryset = Mood.objects.none()  # return empty queryset if start_date or end_date is invalid
    return queryset

def list(self, request, *args, **kwargs):
    queryset = self.get_queryset()
    mood_stats = queryset.values('type_of_mood').annotate(
        mood_count=Count('id'),
        avg_rate=Avg('rate'),
        color=Max('color')
    )
    response_data = []
    for stats in mood_stats:
        response_data.append({
            "type_of_mood": stats["type_of_mood"],
            "count": stats["mood_count"],
            "avg_rate": round(stats["avg_rate"], 2) if stats["avg_rate"] else None,
            "color": stats["color"]
        })
    return Response(response_data)
```

Рис. 3.13 Реалізація відображення статистики настрою

4 ТЕСТУВАННЯ ВЕБ-ЗАСТОСУНКУ

4.1 Мануальне тестування веб-застосунку

Для перевірки роботи веб-застосунку для щоденної психологічної підтримки користувача було проведено мануальне тестування.

Мануальне тестування включає детальну перевірку функціональності веб-застосунку вручну, без використання автоматизованих інструментів тестування. Під час цього процесу ретельно перевіряється робота всіх функціональних компонентів застосунку, таких як форми введення, кнопки, навігаційні елементи та інші інтерактивні елементи, щоб виявити можливі помилки або недоліки у їх роботі. Тестування охоплює перевірку відповідності кожного елемента встановленим вимогам. Всі результати тестування документуються, а виявлені проблеми передаються для подальшого виправлення, забезпечуючи таким чином високу якість кінцевого продукту.

Для проведення мануального тестування було створено три тест кейси:

1. Авторизація в веб-застосунок.
2. Додавання нового нотатку.
3. Додавання відмітки про настрій.

4.1.1 Тест кейс 1: Авторизація в веб-застосунок

Таблиці 4.1 вказує назву тест кейсу, дію, умови виконання та очікуваний результат виконання і результат.

Таблиця 4.1

Тест кейс 1: авторизація у веб-застосунок

Авторизація в Веб-застосунок	
Дія	Пройти авторизацію в застосунок
Умова виконання	Користувач не авторизований в веб-додатку
Дані для введення:	Логін: User. Пароль: User
Очікуваний результат	Користувач авторизований в веб-застосунку
Результат виконання	Авторизація успішна

4.1.2 Тест кейс 2: Додавання нового нотатку

Таблиця 4.2 містить інформацію про другий тест кейс.

Таблиця 4.2

Тест кейс 2: Додавання нового нотатку

Додавання нового нотатку	
Дія	Додати нотаток
Умова виконання	Користувач авторизований
Дані для введення:	Дата нотатку: 24.04.2024 Тип нотатку: Робота Тема нотатку: Важлива зустріч Текст нотатку: Сьогодні на роботі у нас була дуже важлива зустріч, яка принесла багато емоцій. Мене переповнювала суміш хвилювання та ентузіазму, адже ці зміни обіцяли значні покращення в нашій роботі.
Очікувальний результат	Нотаток додано з коректним відображення введених даних
Результат виконання	Нотаток успішне додано

4.1.3 Тест кейс 3: Додавання відмітки про настрій

Таблиця 4.3 містить інформацію про третій тест кейс.

Таблиця 4.3

Тест кейс 3: Додавання відмітки про настрій

Додавання відмітки про настрій	
Дія	Додавання відмітки про настрій
Умова виконання	Користувач авторизований. На дату 15.04.2024 дані про настрій не додавались
Дані для введення	Дата нотатку: 15.04.2024 Тип нотатку: Задоволення Інтенсивність: 3 - Спокій
Очікуваний результат	Відмітка про настрій додана Відображена на діаграмі
Результат виконання	Відмітка про настрій додана успішно

4.2 Юзабіліті тестування веб-застосунку

Юзабіліті тестування – це процес оцінки зручності використання програмного забезпечення або веб-сайту шляхом залучення реальних користувачів. Основна мета такого тестування – виявити проблеми, з якими можуть стикатися користувачі під час взаємодії з продуктом, та знайти шляхи для їх вирішення. Під час тестування користувачам пропонують виконати певні завдання, спостерігаючи за їхніми діями та фіксуючи труднощі або помилки. Юзабіліті тестування дозволяє покращити інтерфейс та загальну взаємодію з продуктом, що підвищує задоволеність користувачів та сприяє досягненню бізнес-цілей.

Для тестування веб-застосунку для щоденної психологічної підтримки DailyMindfulness було проведено юзабіліті тестування за участю п'яти

користувачів. Тестування проводилося анонімно, а користувачі зазначали свій рівень володіння технологіями. Детальна інформація про учасників наведена в таблиці 4.4.

Таблиця 4.4

Фокус група юзабіліті тестування веб-застосунку DailyMindfullness

№	**Досвід користування застосунками для психологічної допомоги**	**Рівень користування технологіями**	**К-сть годин в інтернеті на день**
Користувач 1	Так	Середній - Впевнено користуюсь технологіями / застосунками кожен день	3-5 годин
Користувач 2	Так	Низький - Потребую допомоги під час опанування нових технологій / застосунків	1-2 години
Користувач 3	Ні	Середній - Впевнено користуюсь технологіями / застосунками кожен день	3-5 годин
Користувач 4	Так	Середній - Впевнено користуюсь технологіями / застосунками кожен день	Більше 6
Користувач 5	Так	Середній - Впевнено користуюсь технологіями / застосунками кожен день	3-5 годин

Юзабіліті тестування включало в себе 4 завдання сформовані наступним чином:

1. Пройти авторизацію в веб-застосунку.
2. Створити / додати новий нотаток .
3. Відмітити / додати настрій.
4. Прослухати медитацію.

Результати проведеного юзабіліті тестування показали, що загальний користувацький досвід оцінюється на 4.8 з 5. Користувачі оцінювали кожен аспект тестування окремо, що дозволило отримати детальну зворотний зв'язок про різні елементи веб-застосунку. Оцінки користувачів за кожним параметром наведені на рисунку 4.1. Ці результати вказують на високий рівень задоволеності користувачів та дозволяють ідентифікувати області для подальшого покращення, що є важливим для подальшого розвитку веб-застосунку.

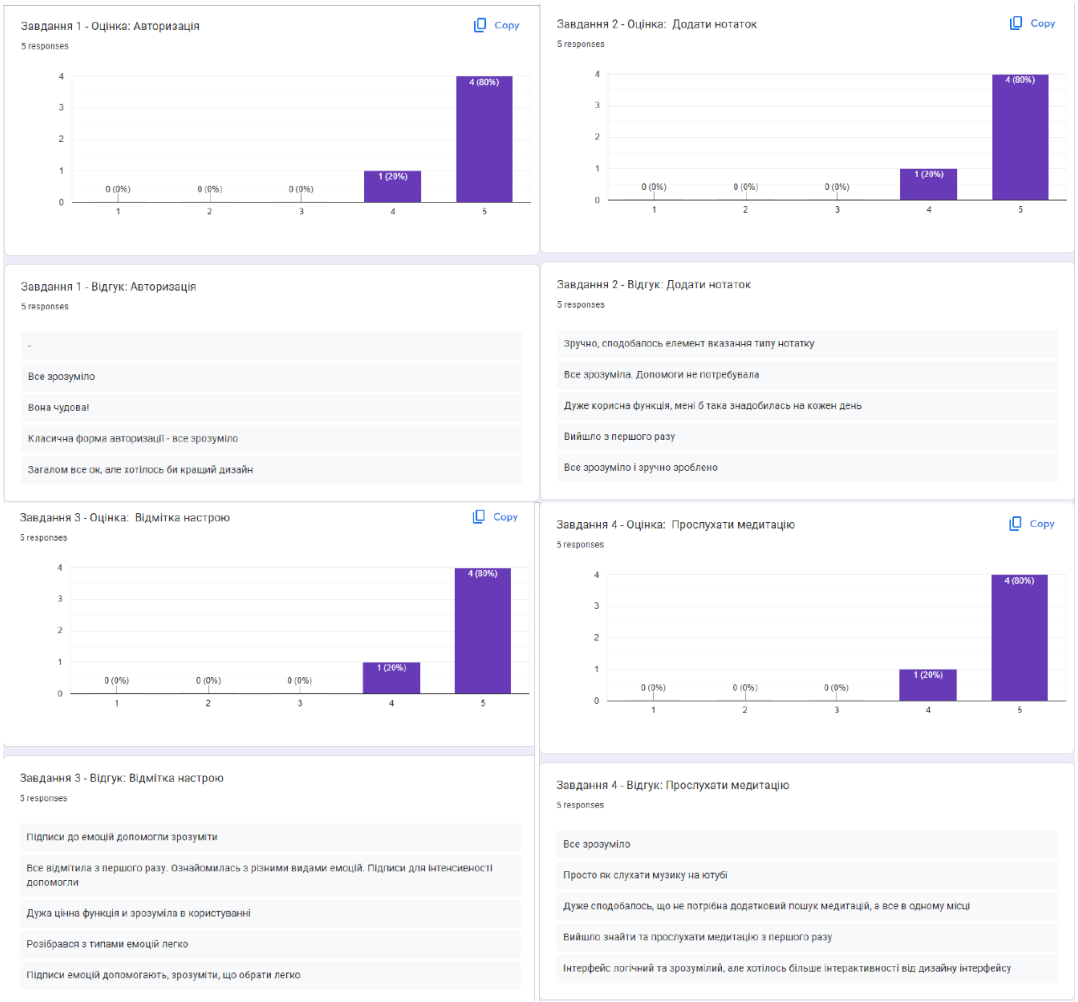


Рис. 4.1 Результати юзабіліті дослідження

ВИСНОВКИ

Під час виконання дипломної роботи було проаналізовано предметну галузь та існуючі інструменти для щоденної психологічної підтримки користувачів, визначено відповідні мови програмування, системи управління базами даних та програмні інструменти для реалізації веб-застосунку для психологічної підтримки.

У другому розділі описано процес проектування веб-застосунку та визначено функціональні і нефункціональні вимоги, які були визначені на основі проведеного аналізу галузі та аналогів. У цьому розділі також представлені діаграма прецедентів, діаграма діяльності, карта сайту, схема бази даних та представлені результати A/B тестування для веб-застосунку.

У третьому розділі детально описано реалізацію веб-застосунку для щоденної психологічної підтримки користувачів, представлено діаграму класів та описано функціоналу веб-застосунку.

У четвертому розділі було проведено мануальне тестування веб-застосунку та тестування зручності використання. Висновки, зроблені на основі тестувань, були враховані та внесені в фінальну версію веб-застосунку.

Виконавши всі завдання, поставлені в рамках даної дипломної роботи, було розроблено веб-застосунок для щоденної психологічної підтримки користувачів на мові Python, який може використовуватись користувачами з різним рівнем технічної підготовки та знань про психічне здоров’я, створюючи доступний інструмент для щоденної психологічної підтримки.

Робота прийшла апробацію на Всеукраїнській науково-технічній конференції «Застосування програмного забезпечення в ІКТ» за темою «Розробка програмного засобу для щоденної психологічної підтримки користувача мовою Python» та на Всеукраїнській науково-практичній конференції «Сучасні інтелектуальні інформаційні технології в науці та освіті» за темою «Використання мови Python для створення веб-додатків» [16,17].

ПЕРЕЛІК ПОСИЛАНЬ

1. The battle for mental well-being in Ukraine: mental health crisis and economic aspects of mental health services in wartime / [V. Seleznova, I. Pinchuk, I. Feldman та ін.]. // International Journal of Mental Health Systems. – 2023. – №17.

2. Dvornyk M. SMARTPHONE APPLICATIONS USAGE IN CONDITIONS OF POPULATION'S PSYCHOTRAUMATIZATON / Maryna Dvornyk. // Information Technologies and Learning Tools. – 2019. – №73. – С. 236–248.

3. Schueller S. Understanding People's Use of and Perspectives on Mood-Tracking Apps: Interview Study [Електронний ресурс] / S. Schueller, M. Neary // JMIR Ment Health. – 2021. – Режим доступу до ресурсу: 10.2196/29368.

4. Şenormancı Ö. Cognitive Behavioral Therapy and Clinical Applications / Ö. Şenormancı, G. Şenormancı. – Rijeka, Croatia, 2018. – 238 с. – (inTech). – Режим доступу до ресурсу: https://books.google.com.ua/books?id=7WaQDwAAQBAJ&lpg=PA201&lr&pg=PR2#v=onepage&q&f=false

5. Who Uses Mhealth? User Archetypes for Physical and Mental Health Apps [Електронний ресурс] // DIGITAL HEALTH. – 2023. – Режим доступу до ресурсу: 10.1177/20552076231152175

6. Patkar U. PYTHON FOR WEB DEVELOPMENT [Електронний ресурс] / Uday Patkar // International Journal of Computer Science and Mobile Computing. – 2022. – Режим доступу до ресурсу: 10.47760/ijcsmc.2022.v11i04.006.

7. Moodtracker.com [Електронний ресурс] – Режим доступу до ресурсу: https://www.moodtracker.com/charts.

8. eMoods [Електронний ресурс] – Режим доступу до ресурсу: https://insights.emoodtracker.com/dashboard.

9. Better Me: Mental Health [Електронний ресурс] – Режим доступу до ресурсу: https://betterme.world/product/meditation.

10. Python for Web Development / [Dr. Uday Patkar] / IInternational Journal

of Computer Science and Mobile Computing, Vol. 11, 2022. Режим доступу: 10.47760/ijcsmc.2022.v11i04.006

11. Django Web Development Framework: Powering the Modern Web / [Chen, S., Ahmmed, S., Lal, K., Deming, C] / American Journal of Trade and Policy, Vol. 7(3), 99-106, USA, 2020. Режим доступу: https://doi.org/10.18034/ajtp.v7i3.675

12. Designing with Data: Improving the User Experience with A/B Testing. 2nd ed. Sebastopol : O'Reilly Media, Inc., 2017. 338 p.

13. Hamidli N. ntroduction to UI/UX Design: Key Concepts and Principles. 2023. P. 5–20.

14. Kirinuki H., Tanno H. Automating End-to-End Web Testing via Manual Testing. Journal of Information Processing. 2022. Vol. 30. P. 294–306. Режим доступу: https://doi.org/10.2197/ipsjjip.30.294.

15. Web App Development. [Електронний ресурс]. – Режим доступу: https://www.trio.dev/blog/web-app-development

16. П’явчук П.С., Жебка В.В. Розробка програмного засобу для щоденної психологічної підтримки користувача мовою Python. Всеукраїнська науково-технічна конференція «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях», 24.04.2024, Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.:ДУІКТ, 2024. С. 419.

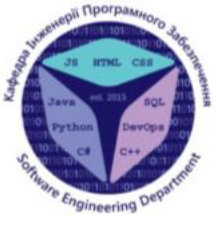
17. П’явчук П.С., Жебка В.В. Використання мови Python для створення веб-додатків. Всеукраїнська науково-практична конференція «Сучасні інтелектуальні інформаційні технології в науці та освіті». 15.05.2024, Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. К.:ДУІКТ, 2024.С. 262.

ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ



Розробка програмного засобу для щоденної психологічної підтримки користувача мовою Python

Виконала студентка 4 курсу
групи ПД-44
П'явчук Поліна Сергіївна
Керівник роботи
д.т.н., професор, зав. кафедри ТЦР, Жебка Вікторія Вікторівна

Київ – 2024

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

- **Мета роботи**: спрощення процесу відслідковування психологічного стану користувача та психологічної підтримки.
- **Об'єкт дослідження:** відслідковування психологічного стану людини та психологічна підтримка.
- **Предмет дослідження:** програмне забезпечення для відслідковування користувачем психологічного стану та психологічної підтримки.

2

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Провести огляд та проаналізувати існуючі методи та технології самостійного відслідковування психологічного стану та психологічної підтримки.
2. Провести аналіз та обрати інструменти, засоби розробки веб-застосунку для реалізації проекту.
3. Сформулювати функціональні, нефункціональні вимоги до веб-застосунку використовуючи проведений огляд та аналіз існуючих аналогів.
4. Розробити прототип та дизайн веб-застосунку для щоденної психологічної підтримки користувача.
5. Провести дослідження зручності розробленого дизайну серед користувачів різного технічного рівня підготовки.
6. Виконати технічне проектування веб-застосунку.
7. Реалізувати веб-застосунок в відповідності до поставлених вимог.
8. Провести мануальне та юзабіліті тестування веб-застосунку.

3

АНАЛІЗ АНАЛОГІВ

Показник	moodtracker.com	eMoods	Better Me: Mental Health	DailyMindfullness
Платформи	Web	Web	IOS, Android	Web
Трекер емоцій	+	+	-	+
Наявність медитацій	-	-	+	+
Щоденик текстовий	+	+	-	+
Щоденні поради	-	+	+	+
Навчання для самопідтримки	-	-	+	+
Зручність використання	-	-	+	+
Наявність української локалізації	-	-	+	+

4

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Функціональні вимоги:

1. Реєстрація та авторизація користувача.
2. Запис даних про емоційний стан користувача.
3. Можливість вести щоденник записів.
4. Можливість прослуховування медитацій.
5. Представлення даних, внесених користувачем за тиждень, місяць та рік.
6. Отримання щоденних порад для покращення психологічного самопочуття.
7. Можливість переглянути статті з методами психологічної самопідтримки.

Нефункціональні вимоги:

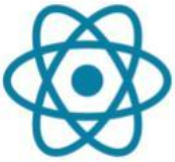
1. Інтерфейс інтуїтивно зрозумілий і доступний для користувачів з різним рівнем технічної підготовки.
2. Використання атласу емоцій Екмана: 5 типів емоцій з інтенсивністю від 1 до 5.

5

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

Python

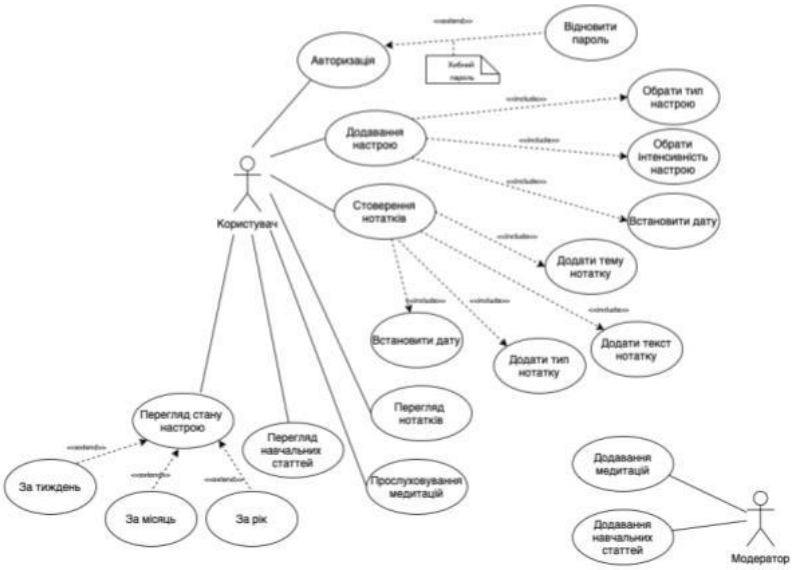
Visual Studio Code



React JS

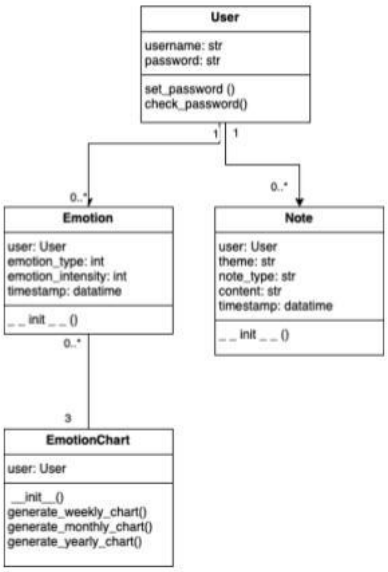
6

Діаграма варіантів використання



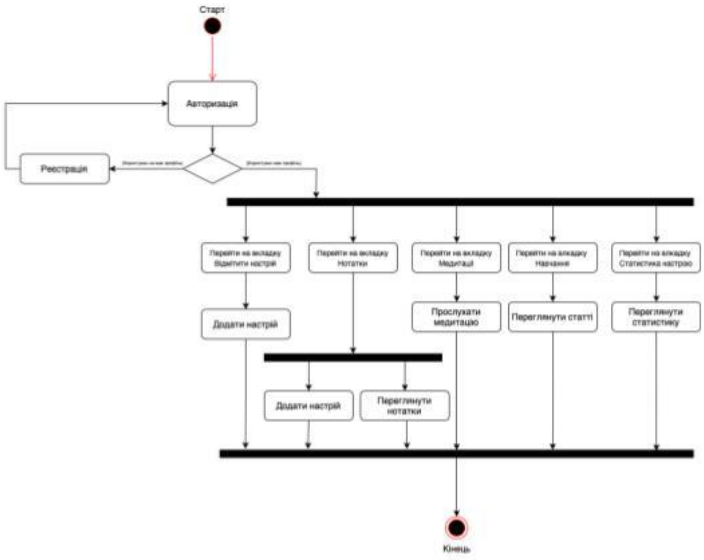
7

Діаграма класів



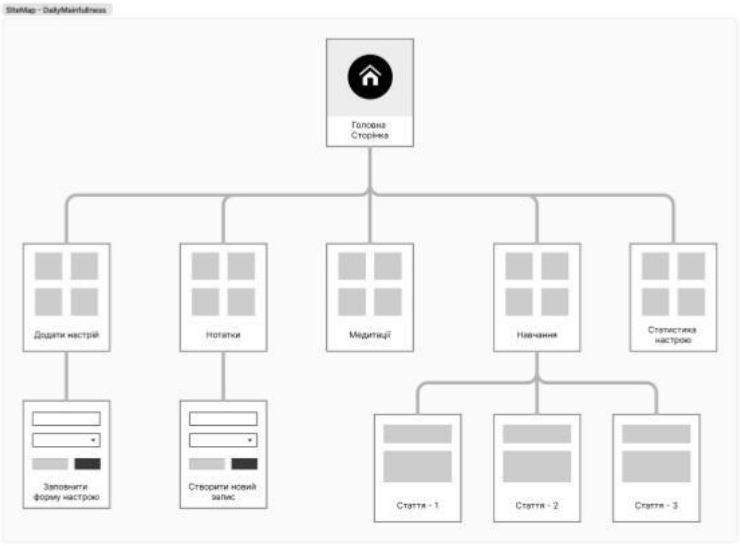
8

Діаграма діяльності



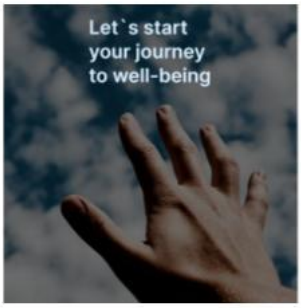
9

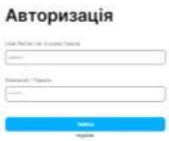
Карта сайту



11

ЕКРАННІ ФОРМИ

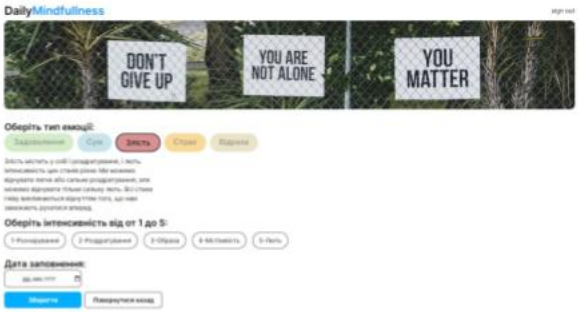


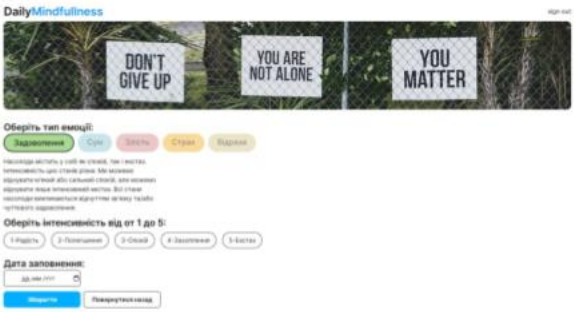


Форма реєстрації

Головний екран

ЕКРАННІ ФОРМИ





Додавання настрою

ЕКРАННІ ФОРМИ

Статистика настрою

Додавання нового нотатку

14

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

1. Жебка В.В.,П’явчук П.С. Розробка програмного засобу для щоденної психологічної підтримки користувача мовою Python: Матеріали Всеукраїнської науково-технічна конференція «Застосування програмного забезпечення в ІКТ». Збірник тез. 24.04.2024, ДУІКТ, м. Київ. С. 419.
2. Жебка В.В., П’явчук П.С. Використання мови Python для створення веб-додатків: Матеріали Всеукраїнська Науково-практична конференція «Сучасні інтелектуальні інформаційні технології в науці та освіті». Збірник тез. 15.05.2024, ДУІКТ, м.Київ. С. 262

15

ВИСНОВКИ

1. Проведено огляд аналогів та проаналізовано актуальність розробки.
2. Проведено аналіз та обрано інструменти, засоби розробки веб-застосунку для реалізації проекту.
3. Розроблено прототип та дизайн веб-застосунку для щоденної психологічної підтримки користувача.
4. Проведено дослідження зручності розробленого дизайну серед користувачів різного технічного рівня підготовки.
5. Виконано технічне проектування веб-застосунку.
6. Реалізовано веб-застосунок в відповідності до вимог.
7. Проведення мануальне та юзабіліті тестування застосунку.

16

ДОДАТОК Б. ЛІСТИНГИ ПРОГРАМНИХ МОДУЛІВ

```
import datetime
import random
from django.utils.dateparse import parse_date
from rest_framework.views import APIView
from rest_framework.response import Response
from rest_framework import status, generics
from .models import Advice, Note, Mood
from .serializers import AdviceSerializer, NoteSerializer,
NoteListSerializer, MoodSerializer
from django.db.models import Avg, Count, Max
from app import models
class RandomAdviceView(APIView):
    def get(self, request):
        advices = Advice.objects.all()
        if not advices:
            return Response({"detail": "No advices found."},
status=status.HTTP_404_NOT_FOUND)
        random_advice = random.choice(advices)
        serializer = AdviceSerializer(random_advice)
        return Response(serializer.data)
class AddAdviceView(APIView):
    def post(self, request):
        serializer = AdviceSerializer(data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data,
status=status.HTTP_201_CREATED)
        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)


class AddNoteView(APIView):
    def post(self, request):
        serializer = NoteSerializer(data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data,
status=status.HTTP_201_CREATED)
        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)
class NoteListView(APIView):
    def get(self, request):
        notes = Note.objects.all()
        serializer = NoteListSerializer(notes, many=True)
        return Response(serializer.data)
class NoteDetailView(generics.RetrieveAPIView):
    queryset = Note.objects.all()
    serializer_class = NoteSerializer
    lookup_field = 'id'
class MoodCreateView(generics.CreateAPIView):
    queryset = Mood.objects.all()
    serializer_class = MoodSerializer
class MoodListView(generics.ListAPIView):
    serializer_class = MoodSerializer
    def get_queryset(self):
        queryset = Mood.objects.all()
        start_date_param =
self.request.query_params.get('start_date', None)
        end_date_param =
self.request.query_params.get('end_date', None)
        if start_date_param and end_date_param:
            start_date = parse_date(start_date_param)
            end_date = parse_date(end_date_param)
            if start_date and end_date:
                queryset =
queryset.filter(date__range=[start_date, end_date])
            else:
                queryset = Mood.objects.none()  # return
empty queryset if start_date or end_date is invalid
        return queryset
    def list(self, request, *args, **kwargs):
        start_date_param =
request.query_params.get('start_date', None)
        end_date_param =
request.query_params.get('end_date', None)
        if start_date_param and end_date_param:
            start_date = parse_date(start_date_param)
            end_date = parse_date(end_date_param)
            if not start_date or not end_date:
```

```
            return Response({"detail": "Invalid start_date
or end_date."},
status=status.HTTP_400_BAD_REQUEST)
        delta = end_date - start_date
        date_range = [start_date +
datetime.timedelta(days=i) for i in range(delta.days + 1)]
        queryset = self.get_queryset()
        moods_by_date = {date: {"type_of_mood": "",
"color": "", "rate": 0} for date in date_range}
        for mood in queryset:
            moods_by_date[mood.date] = {
                "type_of_mood": mood.type_of_mood,
                "color": mood.color,
                "rate": mood.rate}
        response_data = [
            {
                "date": date.strftime("%Y-%m-%d"),
                "type_of_mood": data["type_of_mood"],
                "color": data["color"],
                "rate": data["rate"]   }
            for date, data in moods_by_date.items()]
        return Response(response_data)
    return Response({"detail": "start_date and end_date
parameters are required."},
status=status.HTTP_400_BAD_REQUEST)
class MoodStatsView(generics.ListAPIView):
  serializer_class = MoodSerializer
  def get_queryset(self):
    queryset = Mood.objects.all()
    start_date_param =
self.request.query_params.get('start_date', None)
    end_date_param =
self.request.query_params.get('end_date', None)
    if start_date_param and end_date_param:
      start_date = parse_date(start_date_param)
      end_date = parse_date(end_date_param)
      if start_date and end_date:
        queryset =
queryset.filter(date__range=[start_date, end_date])
      else:
        queryset = Mood.objects.none()  # return
empty queryset if start_date or end_date is invalid
    return queryset
  def list(self, request, *args, **kwargs):
    queryset = self.get_queryset()
    mood_stats =
queryset.values('type_of_mood').annotate(
      mood_count=Count('id'),
      avg_rate=Avg('rate'),
      color=Max('color')  )
    response_data = []
    for stats in mood_stats:
      response_data.append({
        "type_of_mood": stats["type_of_mood"],
        "count": stats["mood_count"],
        "avg_rate": round(stats["avg_rate"], 2) if
stats["avg_rate"] else None,
        "color": stats["color"]})
    return Response(response_data)
class NoteDeleteAPIView(APIView):
  def delete(self, request, note_id):
    try:
      note = Note.objects.get(pk=note_id)
      note.delete()
      return Response({'message': 'Note deleted
successfully.'},
status=status.HTTP_204_NO_CONTENT)
    except Note.DoesNotExist:
      return Response({'error': 'Note does not exist.'},
status=status.HTTP_404_NOT_FOUND)
```