ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи

на ступінь вищої освіти бакалавр

на тему: «Розробка SPA- додатку для комунікації користувачів інтернет ресурсу на основі технології Node.js»

Виконав: студент 4 курсу, групи ПД-44

спеціальності

121 Інженерія програмного забезпечення

(шифр і назва спеціальності)

Халецький Віталій Сергійович.

(прізвище та ініціали)

Керівник Золотухіна О.А.

(прізвище та ініціали)

Рецензент _____

(прізвище та ініціали)

Київ – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти – «Бакалавр»
Напрям підготовки – 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
О.В. Негоденко
“ ____ ” _____ 2022 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Халецькому Віталію Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи: «Розробка SPA- додатку для комунікації користувачів інтернет ресурсу на основі технології Node.js»
Керівник роботи _____,
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від «18» лютого 2022 року №__.
2. Строк подання студентом роботи 03.06.2022
3. Вхідні дані до роботи:
   3.1 Положення побудови систем комунікацій між користувачами інтернет-ресурсів
   3.2 Методи побудови побудови систем комунікацій між користувачами інтернет-ресурсів
   3.3 Існуючі інструменти побудови систем комунікацій між користувачами інтернет-ресурсів
   3.4 Науково-технічна література

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
   4.1 Огляд потреб користувачів при роботі з додатками для комунікацій
   4.2 Розробка структури додатку для комунікацій між користувачами інтернет-ресурсів
   4.3 Програмна реалізація додатку
   4.4 Приклади використання та тестування системи
   4.5 Висновки

5. Перелік графічного матеріалу:
   5.1 Мета, об’єкт та предмет дослідження

5.2 Актуальність роботи
5.3 Аналоги
5.4 Порівняння з аналогами
5.5 Технічне завдання
5.6 Програмні засоби реалізації
5.7 Інструменти використані для реалізації
5.8 Архітектура браузерного розширення
5.9 Архітектура плеєра тестових сценаріїв
5.10 Апробація результатів дослідження
5.11 Висновки

6. Дата видачі завдання 19.04.2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	11.04-14.04	Виконано
2	Вивчення та аналіз задачі	15.04-17.04	Виконано
3	Розробка структури додатку	18.04-21.04	Виконано
4	Розробка дизайну та графічних елементів	22.04-25.04	Виконано
5	Програмна реалізація системи	26.04-05.05	Виконано
6	Тестування програми	05.05-07.05	Виконано
7	Оформлення пояснювальної записки	07.05-10.05	Виконано
8	Розробка обов’язкових демонстраційних матеріалів	11.05-15.05	Виконано
9	Попередній захист роботи	16.05-01.06	Виконано
10	Подання роботи в деканат	03.06	Виконано

Студент _____ Халецький В.С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Золотухіна О.А.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи 67 с., 37 рис., 15 джерел.

Мета роботи – підвищення якості процесів комунікації користувачів інтернет ресурсів за рахунок використання програмного забезпечення на основі технології Node.js

Об’єкт дослідження – процеси комунікації користувачів інтернет ресурсів.

Предмет дослідження – програмні засоби для забезпечення комунікації користувачів інтернет ресурсів

Проведено аналіз очікувань користувачів при роботі з додатками для комунікацій. Розглянуто основні види програм, та способи їх використання. Зроблено огляд інструментальних засобів, які можуть бути використані для комунікації в інтернет ресурсах. Серед програмних засобів увага приділялася додаткам, які реалізовані для мобільних пристроїв. Визначено їх основні переваги і недоліки. Проведено огляд засобів програмної реалізації додатку. Описано переваги використання платформи Node JS при розробці кросплатформних додатків та особливості використання платформи.

Розроблено архітектуру та описано основні принципи роботи додатку. Серверна частина додатку реалізована на платформі Node JS. Клієнтська частина реалізована за допомогою бібліотеки Vue, для бази даних обрано нереляційну базу даних MongoDB. Розроблений веб-додаток надає можливість спілкуватись користувачам між собою, знаходити нових знайомих та поширювати враження у форматі фото та відео контенту. Ключовою особливістю додатку є автоматичні рекомендації нових знайомих з відображенням проценту співпадіння по інтересам і ключовим параметрам. Даний додаток може бути використано у сферах маркетингових компаній.

СОЦІАЛЬНІ МЕРЕЖІ, АВТОМАТИЧНІ РЕКОМЕНДАЦІЇ, МЕСЕНДЖЕР, ПОШУК ЗНАЙОМИХ, ПУБЛІКАЦІЇ НОВИН.

ЗМІСТ

1. ОГЛЯД АРХІТЕКТУРИ ІСНУЮЧИХ ДОДАТКІВ ДЛЯ КОМУНІКАЦІЇ В ІНТЕРНЕТ РЕСУРСАХ

1.1. Огляд потреб користувачів при роботі з додатками для комунікацій

Сьогодні додатки для швидкої комунікації в мережі стали частиною нашого життя, і спростили багато процесів у цій сфері.

Користувачі вибираючи додаток для користування, окрім популярності додатку, також звертають увагу на те, як швидко можна обмінюватись повідомленнями у закритих мережах, чи є можливість ділитися враженнями, оцінювати їх та коментувати, або чи легко знаходити нові знайомства по схожим інтересам.

Аналізуючи потреби користувачів при використані додатків для комунікацій можна виділити три з основних можливостей (рис. 1.1).

Рисунок 1.1 – основні потреби користувачів

Сьогодні у світі існує багато застосунків, котрі надають можливість спілкуватися, знайомитися, ділитися враженнями на дуже великих відстанях за

лічені хвилини, але багато з них націлено на вирішення конкретної задачі, або покривають тільки декілька з них.

Чим більше додаток покриває потреб, тим більше варіантів використання додатку користувачем, та тим швидше зростає популярність додатку, прикладами використання додатків можуть бути:

–створення аудиторії бренду/блогеру, для монетизації продуктів;

–розповсюдження своїх навичок, для пошуку клієнтів;

–знайомства з новими людьми зі схожими інтересами для співпраці, дружби, тощо.

Приклад використання додатку для продажі послуг бізнесу (рис. 1.2)

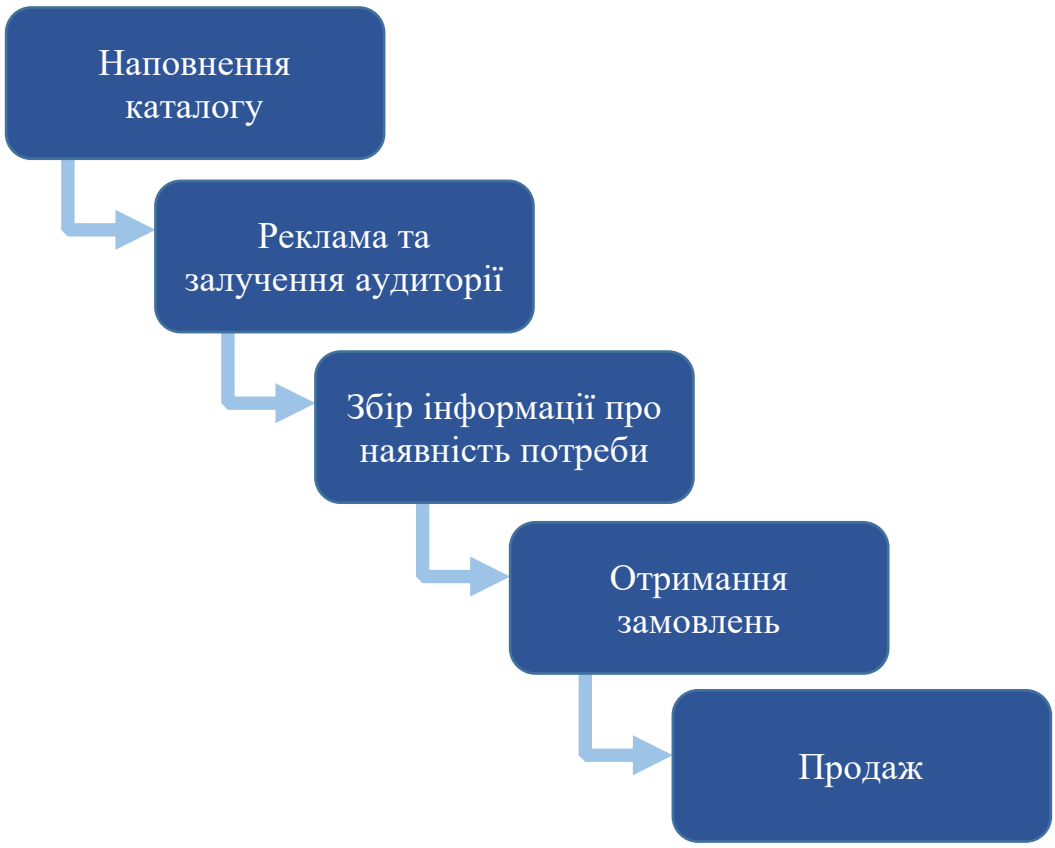


Рисунок 1.2 – Приклад використання додатку для комунікації, як торговий простір

З поширенням популярності додатку кількість інформації яка розміщується в ньому зростає. Для того, щоб мати можливість використовувати її максимально

корисно для користувачів, цю інформацію потрібно організовувати, фільтрувати та акумулювати для забезпечення швидкого доступу для потрібних користувачеві ресурсів по запиту. Для розв'язання цієї задачі підходить інтелектуальний аналіз даних.

Основна ціль інтелектуального аналізу даних, у знаходженні корисної інформації, закономірностей та тенденції які майже неможливо простежити при звичайному перегляду великої кількості даних.

Прикладом задачі, які він допомагає вирішувати у додатках для комунікацій, це рекомендації та пропозиції користувачу на основі його дій в додатку.

1.2. Огляд існуючих додатків для комунікації в інтернет ресурсах

Засоби для комунікації в інтернет ресурсах можуть бути описані наступним переліком:

- месенджери;
- соціальні мережі;
- почтові сервіси;
- контент поширювальні системи (YouTube, TikTok та інше).

Месенджер - це додаток для обміну повідомленнями, який реалізує можливість обмінюватись повідомленнями в режимі реального часу через інтернет.

Соціальні мережі окрім листування домагають знаходити контакти людей, з якими був втрачений зв'язок по різним причинам.

Контент поширювальні системи націлені на можливість ділитися враженнями, та реалізують систему оцінювань для формування розуміння трендів.

Технічні компанії активно створюють та підтримують додатки для комунікації між користувачами, та у своїх розробках використовують інтелектуальний аналіз, постійно шукають найкращі способи взаємодії з великою кількістю даних і способи використання інтелектуального аналізу. Прикладами таких додатків є найпопулярніші сервіси у світі, з величезною аудиторією, котрі ми розглянемо:

Instagram є найпопулярнішим сервісом для поширення фото та відео контенту, належить компанії Meta, котра постійно розвиває та покращує його. Додаток надає можливість користувачам оцінювати та коментувати контент, та взаємодіяти з підписниками за допомогою інтерактивних інструментів. Компанія використовує алгоритми інтелектуального аналізу даних для пошуку зв'язків між знайомими користувача, для формування списку можливих знайомих, які ще не додані у список друзі. Зараз додаток продовжує розвиватись та намагається покрити більше потреб своїх користувачів.

Instagram надає можливість створити бізнес-аккаунт який надає інтернет-магазинам такі можливості, як:

- перегляд аналітичної статистики аудиторії, та результатів продаж;
- зручний зв'язок з клієнтами;
- запуск рекламних компаній у додатку.

Ці можливості є ключовими для розвитку у Instagram. Публікації можна оцінювати та коментувати. Якщо вам сподобалася публікація іншого користувача, ви можете натиснути на серце під публікацією. Кількість натиснутих сердечок впливають на популярність знімоку або відеоролику.

Завдяки інтелектуальному аналізу даних, додаток може рекомендувати вам нових знайомих, та цікаві публікації на основі ваших дій у розділі «Дослідження» (рис. 1.3) (щоб відкрити його, необхідно натиснути на кнопку із зображенням лупи, розташовану в нижній частині екрана), в ньому знаходяться фотографії і відео, які сервіс пропонує, в залежності від ваших уподобань і вашого місця розташування. Тут можна знайти для себе багато цікавого, і можна підписатися на людей, які подобаються.

Одним з основних способів залучення аудиторії та розвиток лояльності до бренду є функція Stories. Вона дає можливість в окремому розділі додатку ділитися фотографіями і короткими відео, прикрашаючи їх фільтрами, наклейками, малюнками або текстом. Є схожість до функціоналу додатку Snapchat.

Для того, щоб відкрити розділ Instagram Stories (рис. 1.4), необхідно натиснути на іконку з плюсом, яка розташована зверху. В окремий розділі можна

викладати фото і короткі відео, які робляться прямо в додатку. Є можливість редагувати фотографії, додаючи до них різні написи, фільтри або наклейки прямо в додатку. Найважливіше правило використання Stories це видалення контенту через добу, в цьому і є головна ідея цієї функції.

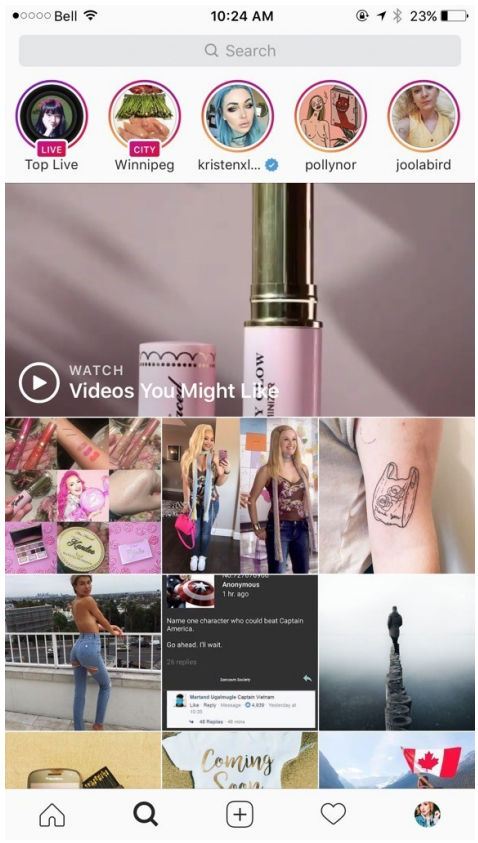


Рисунок 1.3 – Розділ «Дослідження»

Рисунок 1.4 – Процес створення «Stories»

Snapchat - це популярний додаток для комунікації серед підлітків. Тому якщо бізнес орієнтований на молоду аудиторію, то він часто використовується. Зараз це - повноцінна соціальна мережа, де є різноманітний контент, зокрема цілком корисні речі від відомих ЗМІ, виробників одягу, ґаджетів та аксесуарів. Також є фільтри для фото. У Snapchat достатньо простий інтерфейс, але с першого разу незвичний у керуванні, який складається з 5 основних меню (рис. 1.5):

- камера для фотографування з додатку;
- при «свайпі» вгору, відкривається профіль користувача;
- при «свайпі» вниз — відображаються всі попередні «снепи» у розділі memories;
- при «свайпі» вліво відбувається перехід до повідомлень;
- при «свайпі» вправо — відбувається перехід до stories від тих на кого підписаний користувач.

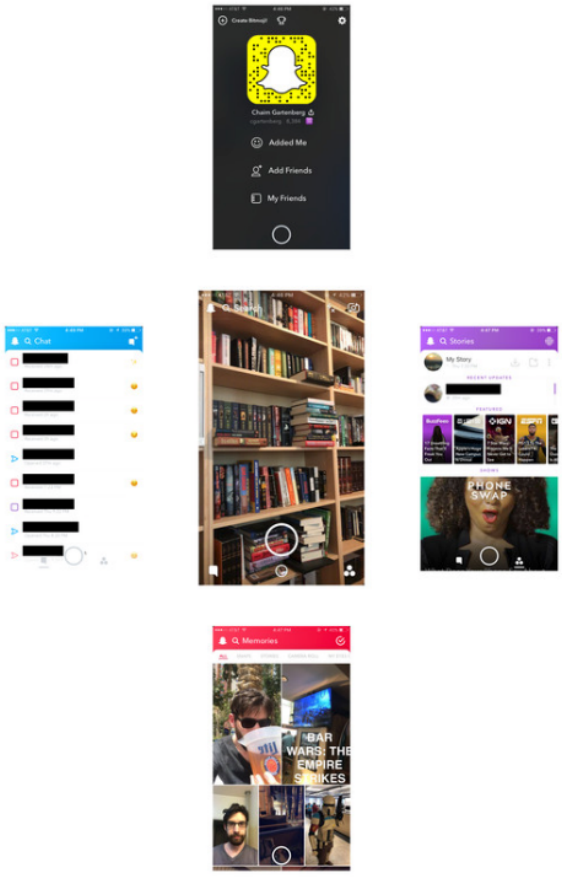


Рисунок 1.5 – Приклад меню Snapchat

TikTok — одна з найновіших платформ соціальних медіа, яка має величезний успіх у різних вікових категоріях. Платформа може бути більше орієнтована на молодшу аудиторію, але батьки та бабусі та дідусі вихідної цільової демографічної групи також перейшли на платформу. Хоча деякі люди думали, що TikTok врешті-решт зникне так само, як раніше Vine і Tumblr, TikTok довів, що він має більшу стійкість, ніж деякі люди віддають йому належне.

TikTok - це ідеальний шторм психологічних трюків, які маркетологи використовують щодня, шанс змінити життя та викликає у людей багато сильних емоцій. На платформі соціальних медіа є багато для перегляду та вивчення, незважаючи на те, що відео зазвичай тривають менше хвилини.

У TikTok міститься різноманітна інформація: від відео милих тварин до активістів, які поширюють інформацію про свої справи. Унікальним аспектом TikTok є величезна різноманітність типів аудиторії, які переглядають і публікують на платформі. Як і інші платформи соціальних медіа, такі як YouTube або Facebook, TikTok знайшов власний спосіб надати платформі привабливу якість. Короткі відео та кадри дофаміну, які TikTok дає своїм глядачам, можуть призвести до того, що вони втрачають години свого дня на платформі.

TikTok також заявив про свою популярність, переконавшись, що він надає тип вмісту, який хочете бачити користувач, за допомогою алгоритму, який дізнається, як користувач використовує платформу. Чим більше користувач використовує TikTok, тим краще програма знає, які відео показувати/рекомендувати. Багато користувачів розглядають TikTok більше, ніж спосіб спілкування в Інтернеті чи збору розваг. Було багато музикантів і артистів, які отримали або збільшили популярність завдяки платформі соціальних мереж. TikTok найбільш відомий усіма тенденціями, які в кращі чи гірші моменти зросли до популярності на сайті. Від відтворення танців або сцен до викрадення раковин зі шкіл — TikTok побачив, що на його платформі охопили низку тенденцій.

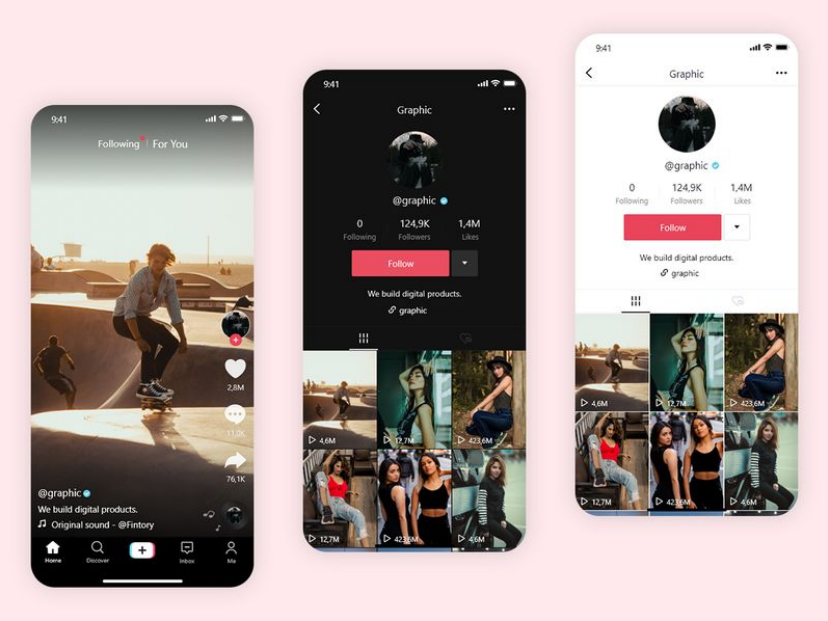


Рисунок 1.6 – Приклад інтерфейсу TikTok

Таблиця 1.1- Зведені результати аналізу характеристик додатків для комунікації користувачів у інтернет ресурсах

Показник платформи	Instagram	Snapchat	TikTok
Можливість обміну повідомленнями	+	+	+
Можливість пошуку нових контактів	+	+	+
Можливість поширювати контент	+	+	+
Можливість оцінювати та коментувати контент	+	+	+
Можливість редагувати фото та відео у додатку	+	+	+
Наявність рекомендацій контактів	+ (на основі аналізу даних про користувача)	+ (тільки контакти з соц. мереж, по телефону)	+ (тільки контакти з соц. мереж, по телефону)
Рекомендація публікацій на основі інтересів	+	+ (частково)	+ (частково)

1.3. Огляд технологій реалізації веб-додатку

Веб-додатки активно замінюють настільні додатки. Ними зручніше користуватися, їх простіше оновлювати розробникам, вони не прив’язані до одного пристрою. І навіть незважаючи на те, що користувачі плавно переходять від веб-додатків на базі браузера до мобільних, попит на складні та вдосконалені програми вже величезний і все ще зростає. Існує два основних шаблони дизайну для веб-програм: багатосторінковий додаток (MPA) і односторінковий додаток (SPA). Обидві моделі мають свої плюси і мінуси.

При виборі кращої моделі дотримуються підходу, що ґрунтується на вмісті, який підкреслює важливість розміщення вмісту вашої програми перед усім іншим. Це тому, що вміст є основною причиною, по якій користувачі будуть або не будуть використовувати програму. Таким чином, є потреба визначити, який контент необхідно представити і який контент буде підходити користувачам найбільше.

Існує багато плюсів і мінусів SPA, а також MPA. Далі будуть наведені відмінності між цими двома шаблонами дизайну.

1.3.1 Одно-сторінковий додаток

Одно-сторінковий додаток – це програма, яка працює всередині браузера і не вимагає перезавантаження сторінки під час використання. Це, наприклад: Gmail, Google Maps, Facebook або GitHub.

SPA – це забезпечення UX, який дозволяє імітувати «природне» середовище у браузері — без перезавантаження сторінки, без додаткового часу очікування. Це лише одна веб-сторінка, яку відвідує користувач, яка потім завантажує весь інший вміст за допомогою JavaScript, від якого вони сильно залежать.

SPA запитує розмітку та дані незалежно і відтворює сторінки прямо в браузері. Це можна зробити завдяки вдосконаленим фреймворкам JavaScript, таким як AngularJS, Ember.js, Meteor.js, Knockout.js.

Одно-сторінкові сайти допомагають утримувати користувача в одному зручному веб-просторі, де вміст представлений користувачеві простим, легким і зручним способом.

Плюси одно-сторінкової програми:

- SPA працює швидко, оскільки більшість ресурсів (HTML+CSS+Scripts) завантажуються лише один раз протягом усього терміну служби програми. Назад і назад передаються тільки дані.
- Розробка спрощена та впорядкована. Немає необхідності писати код для відтворення сторінок на сервері. Набагато простіше розпочати роботу, тому що можнарозпочати розробку з файлу file://URI, взагалі не використовуючи сервер.
- SPA легко налагоджувати за допомогою Chrome, оскільки можна контролювати роботу мережі, досліджувати елементи сторінки та пов’язані з нею дані.
- Створити мобільний додаток простіше, оскільки розробник може повторно використовувати той самий бекенд-код для веб-додатків і нативних мобільних додатків.
- SPA може ефективно кешувати будь-яке локальне сховище. Додаток надсилає лише один запит, зберігає всі дані, потім може використовувати ці дані й працювати навіть офлайн.

Мінуси односторінкової програми наведено нижче.

Зробити SEO-оптимізацію односторінкового додатка дуже складно. Його вміст завантажується за допомогою AJAX (Asynchronous JavaScript and XML) — методу обміну даними та оновлення в програмі без оновлення сторінки. Також, це можна зробити і на стороні сервера. Він повільно завантажується, оскільки для клієнта потрібно завантажити важкі клієнтські фреймворки. Для цього потрібно наявність та ввімкнення JavaScript. Якщо будь-який користувач вимкне JavaScript у своєму браузері, він не зможе правильно представити програму та її дії.

За допомогою ізоморфного візуалізації / рендерингу на стороні сервера можна вже відображати сторінку на сервері. Коли початкова візуалізація знаходиться на сервері і може бути кешована, вимкнення JS не буде проблемою для

отримання відтвореної сторінки. Теоретично, це правильно, але відсутність JS може бути проблемою для інших функцій.

1.3.2 Багатосторінковий додаток

Багатосторінковий веб-сайт містить кілька сторінок і підсторінок у меню. На відміну від односторінкового веб-сайту, єдиний спосіб переходу та перегляду сторінок у багатосторінковому дизайні — це натискання посилань у меню.

Багатосторінковий дизайн добре підходить майже для кожного типу проектів. Приклади багатосторінкового веб-дизайну можна знайти на сайтах електронної комерції (наприклад, Amazon), інформаційних панелях (наприклад, Atlassian) і сайтах електронного навчання (наприклад, Lynda).

Багатосторінковий веб-сайт має три основні переваги перед односторінковим.

По-перше, багатосторінковий сайт пропонує необмежену масштабованість. Можна створювати скільки завгодно сторінок і розширити навігаційну систему за потреби. Наприклад, замінювати верхню панель навігації на спеціальне меню або на рядок пошуку для нескінченних можливостей навігації. Тип дизайну навігації буде залежати від глибини веб-сайту.

По-друге такою навігацією зручно користуватись. Цей тип веб-сайтів існує з 90-х років, а це означає, що більшість користувачів знайомі з ним і часто очікують знайти кілька сторінок на сайтах.

По-третє, сайти з кількома сторінками мають потужні можливості SEO. Багатосторінкові сайти мають більшу кількість вмісту, ніж сайти з однією сторінкою. І хоча потенціал SEO кожного веб-сайту в значній мірі залежить від вашої стратегії цифрового маркетингу, просто наявність потенціалу контенту для оптимізації SEO — це чудовий початок.

Мінуси багатосторінкової програми наведено нижче.

По-перше, багатосторінковий сайт зазвичай має багато вмісту тому його підтримувати та оновлювати, так як кожну сторінку потрібно редагувати окремо

По друге, веб-сайти з великою кількістю вмісту часто повільно завантажуються, відволікають і можуть викликати відмову користувачів – згідно з журналом пошукової системи.

По-третє, багатосторінковий дизайн важче адаптувати до мобільних пристроїв. На відміну від односторінкових сайтів, де для розробки мобільного сайту можна використовувати той самий код, багатосторінковий дизайн потрібно починати з нуля, щоб створити мобільну версію. Це не тільки дорожче та забирає багато часу, але призводить до ризику втратити узгодженості дизайну на веб та мобільних платформах.

1.3.3 Гібридні технології

При розгортанні веб-програми, в першу чергу розглядається її мета. Якщо потрібні кілька категорій (оскільки, наприклад, відбувається керування інтернет-магазином або публікація великої кількості іншого вмісту), використовують багатосторінковий сайт. Якщо сайт підходить лише для односторінкової роботи – використовують SPA. У випадку, коли контент не вміщується на одній сторінці, але використовують SPA, альтернативою є цього гібридний сайт.

Гібридна програма використовує те, що є найкращим в обох підходах, і намагається мінімізувати недоліки. По суті, це односторінкова програма, яка використовує прив’язки URL-адрес як синтетичні сторінки, що дає змогу створювати навігацію в браузері та функціональність налаштувань.

Модель Single Page Application (включаючи гібридну програму) приносить багато переваг. Багато програм на ринку переходять на цю модель. Однак, оскільки деякі проекти просто не вписуються в SPA, модель MPA все ще широко використовується.

1.4. Огляд технології Progressive web application

Прогресивні веб-програми (PWA) — це програми, створені за допомогою веб-технологій, як-от HTML, CSS та JavaScript. Але вони мають відчуття та

функціональність справжнього нативного додатка. PWA створені з такими можливостями, як push-повідомлення та можливість працювати в автономному режимі. Вони також побудовані на основі сучасних API, що полегшує надання покращених можливостей, а також надійність і можливість їх встановлення на будь-якому пристрої.

PWA використовує переваги величезної веб-екосистеми, яка включає плагіни, спільноту та відносну простоту розгортання та підтримки веб-сайту всупереч нативній програмі, яку досить важко розробити. Загальною особливістю цих продуктів є те, що вони можуть працювати в автономному режимі з того місця, де востаннє залишився користувач. Подібно до того, як під час створення нативного мобільного додатка існують певні очікування, які слід задовольнити, щоб створити хороший продукт для споживчого використання, те саме стосується і PWA.

1.4.1 Характеристики PWA

Гнучкість. Різні компанії виробляють гаджети з різними розмірами екрана, і розробник несе відповідальність за те, щоб усі користувачі отримували задоволення від продукту, незалежно від пристрою, який вони використовують. Тому необхыдно переконатися, що додаток можна використовувати на будь-якому розмірі екрана, а його вміст доступний у будь-якому розмірі області перегляду.

Можливість встановлення. Дослідження показали, що користувачі, як правило, частіше користуються встановленими програмами, ніж відвідують офіційны сайти. PWA надає користувачам вигляд, відчуття та залучення до звичайного додатка.

Незалежне підключення. Завдяки тому, що користувач буде залучений до програми, навіть коли він перебуває в автономному режимі, забезпечується більш послідовний досвід, ніж повернення його на офлайн-сторінку за умовчанням. Хорошим прикладом для ілюстрації цього є те, що для музичного додатка користувачі повинні мати доступ до відтворення в автономному режимі та слухати збережену музику навіть без підключення до Інтернету. Іншим хорошим

прикладом є додаток twitter, користувач може повернутися до прочитаних твітів, які вони могли пропустити.

Виявленість. Оскільки більшість PWA є перетвореними веб-сайтами, справедливо робити їх видимими в пошукових системах, це допоможе створити додатковий трафік до програми. Це також є перевагою перед нативними програмами, які неможливо знайти за допомогою пошукових систем.

Зовнішній вигляд. Зовнішній вигляд програми має бути схожим на звичайну програму, тому використовують такі речі, як значок програми, заставка тощо, це допоможе зробити додаток легко впізнаваним.

Кросплатформеність. PWA спочатку розробляються як веб-додатки, а це означає, що вони повинні працювати в усіх браузерах/системах, а не лише в деяких. Користувачі повинні мати можливість використовувати їх у будь-якому браузері, перш ніж вирішити їх встановити.

1.4.2 Відмінності між PWA та нативними програмами

Вартість розробки. PWA дешевше розробляти порівняно з Native Apps. Коли розробляється нативна програма, доводиться вивчити певну мову програмування, а потім створювати версію програми для кожного типу пристроїв, Android та iOS. З іншого боку, можна найняти досвідченого професіонала, який виконає роботу, яка навіть обійдеться дорожче. У подальшому також знадобляться ресурси для підтримки та оновлення програми, а це означає, що потрібно багато грошей і часу. У разі PWA існує єдина кодову базу для різних платформ. Це також економить час, оскільки не потрібно розробляти його з нуля і можна налаштувати оточний веб-сайт відповідно до вимог.

Виявленість. Вбудовані програми не можуть бути проіндексовані пошуковими системами, їх можна просто знайти на веб-сайті App/Play Store. На відміну від нативних програм, PWA працюють як веб-сайти, тому їх можна індексувати пошуковими системами. Це допомагає їм краще ранжуватися в результатах пошуку.

Безпека У наш час для запуску веб-сайту він повинен бути зашифрований сертифікатом SSL, це додає додатковий рівень безпеки. PWA – це сайт, перетворений на додаток, що означає, що вони більш безпечні, оскільки працюють на HTTPS. Це протоколи безпеки, які дозволяють безпечно обмінюватися даними між клієнтом і сервером, щоб вони не були підроблені. Щоб захистити власні програми, потрібно застосувати різні заходи безпеки, як-от багатофакторну автентифікацію тощо.

Установка та завантаження. Нативні програми потрібно завантажити та встановити з магазину додатків. Це вимагає певних зобов'язань від користувача, щоб зробити це від початку до кінця. Користувачі повинні передати та перевірити кілька дозволів перед встановленням програми. PWA не вимагає жодного з цих кроків. У браузері можна додати його в закладки та додати програму на головний екран лише кількома натисканнями. Багато організацій, як приватних, так і державних, переходять на PWA не тільки тому, що їх розробка дешева, але й тому, що вони пропонують залучення більшої аудиторії.

1.4.3 Переваги PWA

Вони чутливі до форм-фактору пристроїв та працюють з різними розмірами екрана. Вони функціонують так само, як і звичайні нативні програми. Оновлення є незалежними, вам не потрібно відвідувати Play Store для оновлення. Вони створені за допомогою поширених веб-технологій. Вони швидкі та легкі. Вони працюють офлайн на відміну від інших сайтів. Їх можна знайти за допомогою пошукової системи. Вони легко встановлюються. Низька вартість обслуговування.

1.5. Архітектура сучасних архітектурних рішень для створення веб-додатків

Усі веб-додатки засновані на архітектурі клієнт-сервер. У цій моделі клієнт - це браузер або програма, яка «просить» сервер зробити можливу певну дію. Сервер визначає, чи можна виконати якусь дію, і надсилає відповідну відповідь браузеру.

Існує кілька способів того, як архітектори можуть організувати систему клієнт-сервер.

1.5.1. Базові види архітектури для веб-додатків

Дворівнева модель — це традиційна структура веб-програми, що складається з двох основних частин:

– сторона клієнта - браузер, який отримує необхідні дані, надані сервером. Він відображає інтерфейс користувача, надсилає запити на сервер і обробляє відповіді з нього.

– сторона сервера - веб-сервер, який надсилає дані на сторону клієнта, що дозволяє клієнту виконати конкретне завдання.

Інтернет забезпечує зв'язок між браузером і веб-сервером за допомогою HTTP-запитів і відповідей. Цей тип архітектури найбільш актуальний для невеликих веб-додатків, розрахованих на менше ніж 100 користувачів. У цій моделі небажана більша кількість користувачів і запитів. Один сервер може не витримати значне навантаження даних. Якщо він аварійно завершує роботу, використовувати весь додаток стає неможливим. (рис. 1.7)

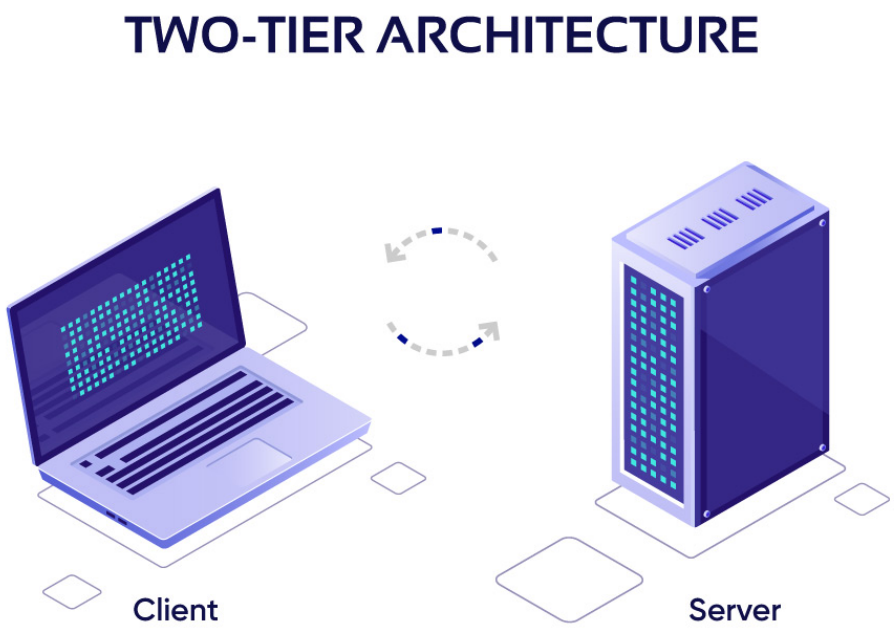


Рисунок 1.7 – Ілюстрація дворівневої архітектури

Найпоширенішим способом створення програми клієнт-сервер є створення трирівневої архітектури веб-додатків. Використовуючи три рівні, можна найбільш зручно розподілити навантаження програми між клієнтом, сервером і базою даних і забезпечити її високу швидкість і безперебійну роботу. Діаграма архітектури високого рівня для веб-програми нижче зображує елементи кожного шару зсередини (рис. 1.8). Його основні частини включають:

– рівень презентації – використовується для відображення даних користувачеві (UI) та введення даних (веб-браузер);

– рівень бізнес-логіки - рівень основних функцій програми та обробки даних (здійснюється завдяки різноманітним серверним фреймворкам для PHP, Python, Java тощо);

– рівень даних - відповідає за зберігання та доступ до даних (база даних).

У порівнянні з традиційною моделлю сервер-клієнт, багаторівнева архітектура характеризується наступними ознаками.

1. Вища масштабованість. Усі функції розподілені між серверами другого та третього рівнів. Дворівнева модель простіша, оскільки пропонує використовувати один сервер. Однак це вимагає більшої продуктивності сервера, оскільки він повинен витримувати значні навантаження.

2. Підвищена конфігураційність. Шари не залежать один від одного, тому замість того, щоб переробляти весь додаток, розробник може просто змінити рівень.

3. Покращена безпека. Багаторівнева структура дає можливість дбайливо забезпечити захист для кожного рівня або послуги.

Монолітний веб-додаток неподільний і виконується в одному процесі. Додати щось нове до великої веб-програми стає складно, оскільки зміна лише однієї її частини часто вимагає зміни всієї структури програми. Переробка системи вимагає багато зусиль команди розробників, а також часу та грошей підприємця. Однак узгодженість простих монолітних додатків робить їх швидкими та високочутливими.

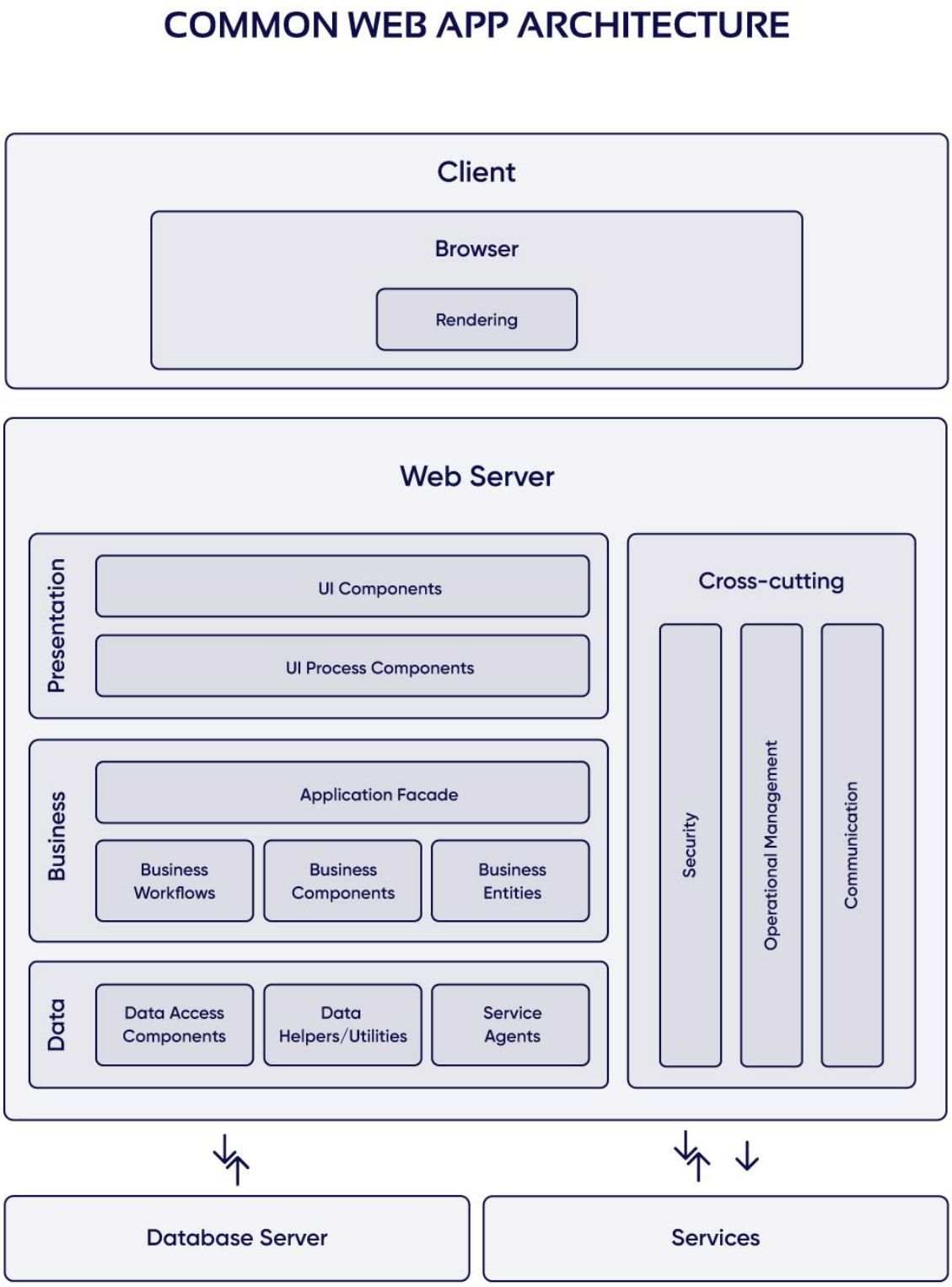


Рисунок 1.8 – Діаграма зв’язків елементів архітектури веб-додатку

У рамках підходу мікросервісів фахівці поділяють складні програми на частини, які називаються веб-сервісами. Кожен блок відповідає за одну функціональність. Якщо фахівцям доводиться щось змінити або додати до

програми, вони можуть змінити один блок, не впливаючи та не перериваючи роботу всієї системи. Додаток стає легше масштабувати, оскільки потреби вашого бізнесу зростають.

У рамках моделі безсерверної архітектури розробник використовує сторонні хмарні рішення для серверної частини веб-додатка. Такі інструменти, як AWS Cloud, Microsoft Azure, Google Cloud та IBM OpenWhisk, обробляють усі аспекти розміщення веб-програм – від обчислень і безпеки до зберігання та розповсюдження даних. Точніше, безсерверна архітектура пропонує такі переваги.

Зосередження на функціях програми. Постачальник хмарних послуг несе повну відповідальність за моніторинг, масштабування та обслуговування сервера. Таким чином, розробники зосереджуються на якості функцій, а не витрачають час на з'ясування ризиків та обслуговування сервера.

Висока продуктивність програми. Хмарні рішення роблять веб-програми надзвичайно гнучкими. Незалежно від зміни кількості користувачів і запитів, які підлягають обробці, вони динамічно масштабують додаток і автоматично балансують навантаження. В результаті веб-додаток може витримувати великі навантаження на трафік, зберігаючи при цьому високу продуктивність.

Швидкий час виходу на ринок. Оскільки розробникам не потрібно розробляти виділений сервер, значна частина роботи делегується хмарному провайдеру. Отже, команди розробників працюють швидше над випуском програми.

Використання безсерверних обчислювальних рішень і баз даних є більш бюджетним, ніж володіння виділеним сервером. Компанії скорочують витрати, оскільки їм не потрібно використовувати будь-яке обладнання та платити за ліцензії на власне програмне забезпечення. Зазвичай вони платять саме за спожиту площу та використані послуги.

Багато людей вважають хмарний хостинг менш безпечним, ніж володіння виділеним сервером. По-перше, розробники довіряють розташування сервера в Інтернеті, який уразливий для хакерських атак. По-друге, постачальник хмарних послуг стає відповідальним за повний контроль над сервером.

Завдяки виділеному серверу розробники контролюють його продуктивність і безпеку. Хоча це може зайняти більше часу, ви отримуєте гарантію, що ваша довірена команда розробників вирішує всі проблеми безпосередньо, а не покладатиметься на третіх сторін.

1.5.2 NodeJS як інструмент для написання сервера на JavaScript

Node.js використовує архітектуру «Single Threaded Event Loop» для одночасної роботи з кількома клієнтами. У багатопоточній моделі запит-відповідь кілька клієнтів надсилають запит, і сервер обробляє кожен з них, перш ніж надіслати відповідь назад. Однак для обробки одночасних викликів використовуються кілька потоків. Ці потоки визначаються в пулі потоків, і щоразу, коли надходить запит, для його обробки призначається окремий потік (рис. 1.9).

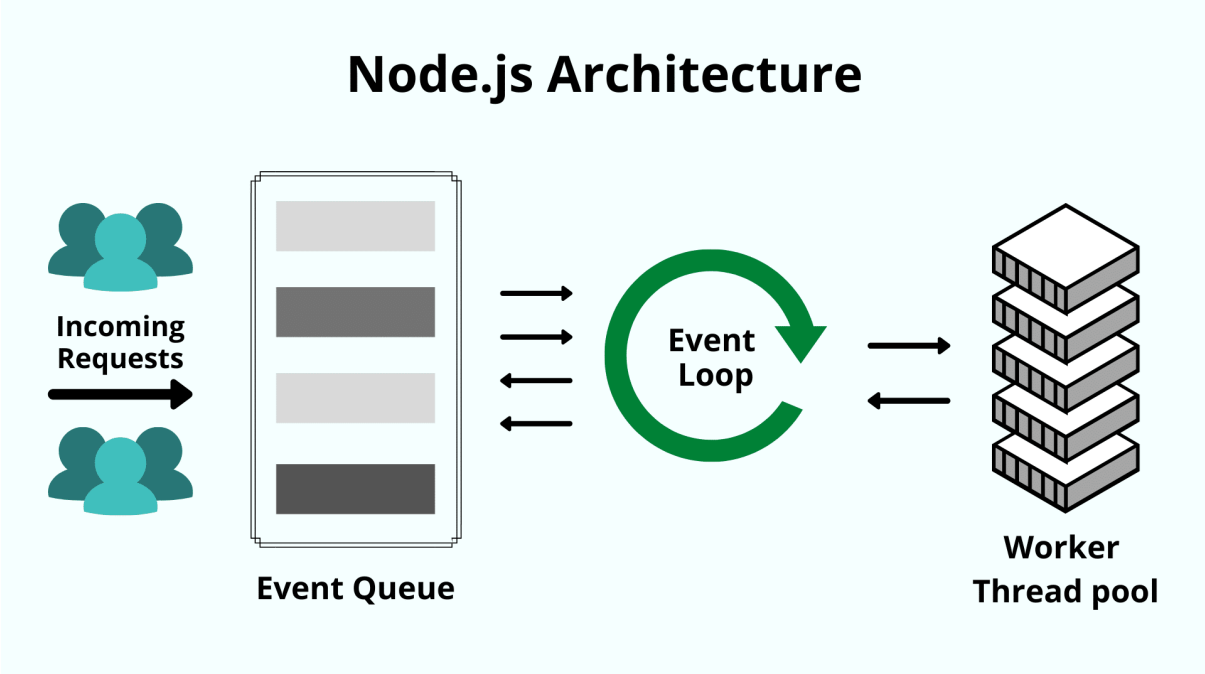


Рисунок 1.9 - Схема обробки node.js вхідних запитів
за допомогою циклу подій

Нижче наведено опис алгоритму обробки.

Node.js підтримує обмежений пул потоків для обслуговування запитів. Щоразу, коли надходить запит, Node.js поміщає його в чергу.

Після цього з’являється однопотоковий «цикл подій» — основний компонент. Цей цикл подій чекає на запити необмежено довго.

Коли надходить запит, цикл забирає його з черги і перевіряє, чи потрібна для нього операція блокування введення/виводу (I/O). Якщо ні, він обробляє запит і надсилає відповідь.

Якщо запит має виконати операцію блокування, цикл подій призначає потік із внутрішнього пулу потоків для обробки запиту. Доступні обмежені внутрішні потоки. Ця група допоміжних потоків називається робочою групою.

Цикл подій відстежує запити блокування та розміщує їх у черзі після обробки завдання блокування. Таким чином він зберігає свою неблокуючу природу.

Оскільки Node.js використовує менше потоків, він використовує менше ресурсів/пам’яті, що призводить до швидшого виконання завдання. Отже, для цілей дипломної роботи ця однопотокова архітектура еквівалентна багатопотоковій архітектурі.

Node.js завдяки надає широкий перелік функцій та має значні переваги. Node.js забезпечує широку масштабованість для додатків. Будучи однопотоковим, здатний обробляти величезну кількість одночасних з'єднань з високою пропускною здатністю. Виконання потоку без блокування робить Node.js ще швидшим і ефективнішим. Доступний великий набір пакетів Node.js з відкритим кодом, які можуть спростити роботу. Сьогодні в екосистемі NPM є понад мільйон пакетів. Node.js написаний на C і C++, що робить його швидким і додає такі функції, як підтримка мережі. Багатоплатформенна підтримка між платформою дозволяє створювати веб-сайти SaaS, настільні програми і навіть мобільні додатки, використовуючи Node.js. Node.js є простим вибором для розробників, оскільки інтерфейсом і бекендом можна керувати за допомогою JavaScript як однієї мови. Node.js використовується для широкого спектру додатків.

Класи додатків, в яких використання Node.js дозволяє отримати гарні результати наведено нижче.

Чати в режимі реального часу — завдяки однопоточній асинхронній природі Node.js добре підходить для обробки спілкування в режимі реального часу. Він

легко масштабується і часто використовується для створення чат-ботів. Node.js також спрощує створення додаткових функцій чату, таких як чат із кількома особами та push-повідомлення.

Інтернет речей — програми IoT зазвичай містять кілька датчиків, оскільки вони часто надсилають невеликі шматки даних, які можуть накопичуватися у великій кількості запитів. Node.js є хорошим вибором, оскільки він здатний швидко обробляти ці одночасні запити

Передача даних — такі компанії, як Netflix, використовують Node.js для цілей потокової передачі. В основному це пов’язано з тим, що Node.js є легким і швидким, крім того, Node.js надає вбудований API потокової передачі. Ці потоки дозволяють користувачам передавати запити один одному, в результаті чого дані передаються безпосередньо до кінцевого пункту призначення.

Комплексні односторінкові програми (SPA) - у SPA весь додаток завантажується на одній сторінці. Зазвичай це означає, що у фоновому режимі виконується кілька запитів для певних компонентів. Тут на допомогу приходить цикл подій у Node.js, оскільки він обробляє запити неблокуючим способом.

Програми на основі REST API—JavaScript використовується як у інтерфейсі, так і в серверній частині сайтів. Таким чином, сервер може легко спілкуватися з інтерфейсом через REST API за допомогою Node.js. Node.js також надає такі пакети, як Express.js і Koa, які спрощують створення веб-додатків.

1.5.3. Нереляційна база даних MongoDB (NoSQL)

Подібно до того, як реляційні бази даних базуються на таблицях, бази даних, орієнтованих на документи, таких як MongoDB, засновані на колекції документів, причому кожен з цих документів складається з атрибутів ключ/значення. Один документ можна розглядати як еквівалент рядка в таблиці, причому кожен ключ схожий на назву стовпця, а значення кожного ключа подібне значенням відповідного рядка. Основна відмінність полягає в тому, що документ не обмежений певною схемою або стовпцями в таблиці. Два документи можуть мати схожі елементи, як-от поле ідентифікатора, а також мати абсолютно різні елементи.

MongoDB дозволяє динамічно змінювати схему, легко вносити гнучкі зміни, не переробляючи існуючу базу даних для підтримки нових полів. Крім того, ієрархія документів легко зіставляється з ієрархією об’єктів у коді програми, спрощуючи операції створення, читання, оновлення та видалення.

MongoDB не тільки надає всі ці можливості, але і робить це без впливу на продуктивність, високу доступність або масштабованість (рис.1.10).

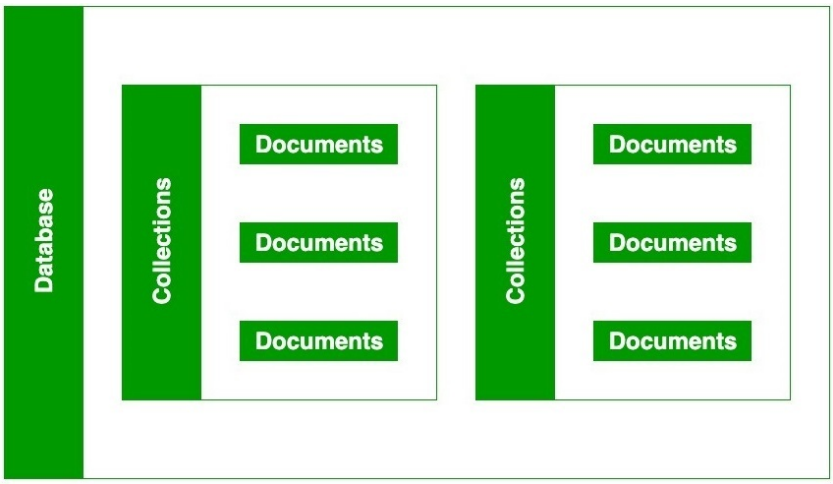


Рисунок 1.10 – Архітектура бази даних

Особливості MongoDB:

- Динамічні схеми – динамічна схема MongoDB забезпечує простий спосіб внесення змін у міру зміни вимог програми. Ці зміни можуть бути внесені в базу даних, не впливаючи на наявні дані або код програми та без простою.

- Операційний інтелект – власна структура карт/зменшень та агрегації MongoDB надає уявлення про програми в режимі реального часу, виходячи за межі можливостей технологій пакетної аналітики, таких як Hadoop та традиційних інструментів BI.

- Гнучкість розгортання – MongoDB була створена для роботи із звичайним обладнанням та хмарними архітектурами. Дані локалізовано для запитів, щоб забезпечити надійну та передбачувану продуктивність незалежно від розміру розгортання.

- Просте масштабування – MongoDB розроблено для масштабування між кластерами серверів. У міру зростання обсягів даних організації можуть просто додати більше вузлів до своїх кластерів, а MongoDB буде плавно і автоматично збалансувати дані у фоновому режимі.

- Розширені запити – MongoDB підтримує повну мову запитів та первинну та вторинну індексацію, а також повнотекстовий пошук із синтаксисом, схожим на Google.

1.6. Постановка завдань дослідження

Мета роботи – підвищення якості процесів комунікації користувачів інтернет ресурсів за рахунок використання програмного забезпечення на основі технології Node.js.

Об’єкт дослідження – процеси комунікації користувачів інтернет ресурсів.

Предмет дослідження – програмні засоби для забезпечення комунікації користувачів інтернет ресурсів.

Під час розробки концепції системи було визначено, що перш ніж проектувати систему, та вибирати конкретні інструменти та засоби реалізації, потрібно дослідити сферу комунікації між користувачами в інтернет-ресурсах, та вивчити системи, які мають подібний функціонал, до нашої майбутньої.

Тому для ефективності побудови додатку, були поставлені завдання на дослідження певних технологій та інструментів, що дозволять реалізувати додаток:

1) дослідити способи комунікації між інтернет користувачами;

2) дослідити інструменти побудови десктопних додатків за допомогою мови програмування JS;

3) дослідити методи передачі інформації між клієнтом та сервером, спроектувати архітектуру майбутньої системи;

4) провести аналіз структур даних майбутньої системи, обрати базу даних, яка задовольнятиме архітектуру та вимоги до системи, та розробити схему бази даних;

5) спроектувати, розробити та протестувати додаток, який забезпечує:

– можливість обмінюватись повідомленнями у режимі реального часу;
– можливість публікувати контент, оцінювати та коментувати його;
– можливість пошуку знайомих по заданому фільтру;
– можливість пошуку цікавих людей за схожими інтересами

2. РОЗРОБКА СТРУКТУРИ ДОДАТКУ ДЛЯ КОМУНІКАЦІЇ КОРИСТУВАЧІВ В ІНТЕРНЕТ-РЕСУРСАХ

2.1. Завдання додатку для комунікації користувачів інтернет-ресурсів

Аналіз програмного забезпечення для комунікації користувачів інтернет-ресурсів показує, що популярні додатки для спілкування не включають можливість пошуку людей по заданим критеріям, наприклад по спільним інтересам, за сферою діяльності.

Створення додатку з функціоналом, що забезпечує можливість швидко знаходити контакт необхідного професіонала для співпраці, а майбутнім підприємцям інвесторів для своїх ідей, а також, для залучення більшої кількості зацікавленої аудиторії, ділитися своїми враженнями, коментувати, і оцінювати контент, дозволить зайняти проміжну нішу між месенджерами, платформами роботи з клієнтами та ресурсів для інтерактивного обміну контентом.

Створення нового додатку можливе на основі аналізу функціоналу вже реалізованих та існуючих соціальних мереж з доповненням, в якому буде можливість знаходити потенційні контакти для співпраці, за допомогою автоматичного аналізу взаємозв'язків між користувачами, які додані у список інтересів або друзів, а також їх сферу діяльності, інтереси та вподобання для формування нового списку потенційних контактів для співпраці. Важливо, зберігати конфіденційність і показувати інформацію тільки у загальному вигляді.

Додаток для ефективної комунікації між користувачами має забезпечувати наступні функції:

- можливість обмінюватись повідомленнями у режимі реального часу;
- можливість публікувати контент, оцінювати та коментувати його;
- можливість пошуку знайомих по заданому фільтру;
- можливість пошуку цікавих людей по схожим інтересам.

2.2. Моделювання об’єкту проектування

2.2.1. Діаграма прецедентів системи

Для моделювання системи найважливішим аспектом є охоплення динамічної поведінки. Динамічна поведінка означає поведінку системи, коли вона працює/працює. Діаграми варіантів використання складаються з акторів, варіантів використання та їх взаємозв'язків. Діаграма використовується для моделювання системи/підсистеми програми. Одна діаграма варіантів використання відображає певну функціональність системи. Тому для моделювання всієї системи використовується ряд діаграм варіантів використання. Метою діаграми варіантів використання є охоплення динамічного аспекту системи. Діаграми варіантів використання використовуються для збору вимог системи, включаючи внутрішні та зовнішні впливи. Ці вимоги здебільшого є вимогами до проектування. Отже, коли система аналізується, щоб зібрати її функціональні можливості, готуються варіанти використання та визначаються дійові особи.

Коли початкове завдання виконано, діаграми варіантів використання моделюються, щоб представити зовнішній вигляд.

Цілі діаграм варіантів використання можна назвати наступними:

- використовуються для збору вимог системи;
- використовуються для огляду системи ззовні;
- визначають зовнішні та внутрішні фактори, що впливають на систему;
- показують взаємодію між вимогами акторів.

Діаграми варіантів використання розглядаються для аналізу вимог високого рівня системи. Коли вимоги системи аналізуються, функціональні можливості фіксуються у варіантах використання. Можна сказати, що варіанти використання - це не що інше, як системні функціональні можливості, написані організовано.

Друга річ, яка має відношення до випадків використання, - це актори. Акторів можна визначити як щось, що взаємодіє з системою. Акторами можуть бути користувачі-люди, деякі внутрішні програми або деякі зовнішні програми.

Метою даного проекту є створення системи, яка буде надавати можливість знаходити друзів, партнерів по схожим інтересам та сферам діяльності. Основними функціями цього додатку також є автоматичні рекомендації нових знайомих, обмін повідомленнями та враженнями.

Користувачами будуть люди у різних сферах діяльності та віку. На рисунку 2.1 показані основні дії у додатку.

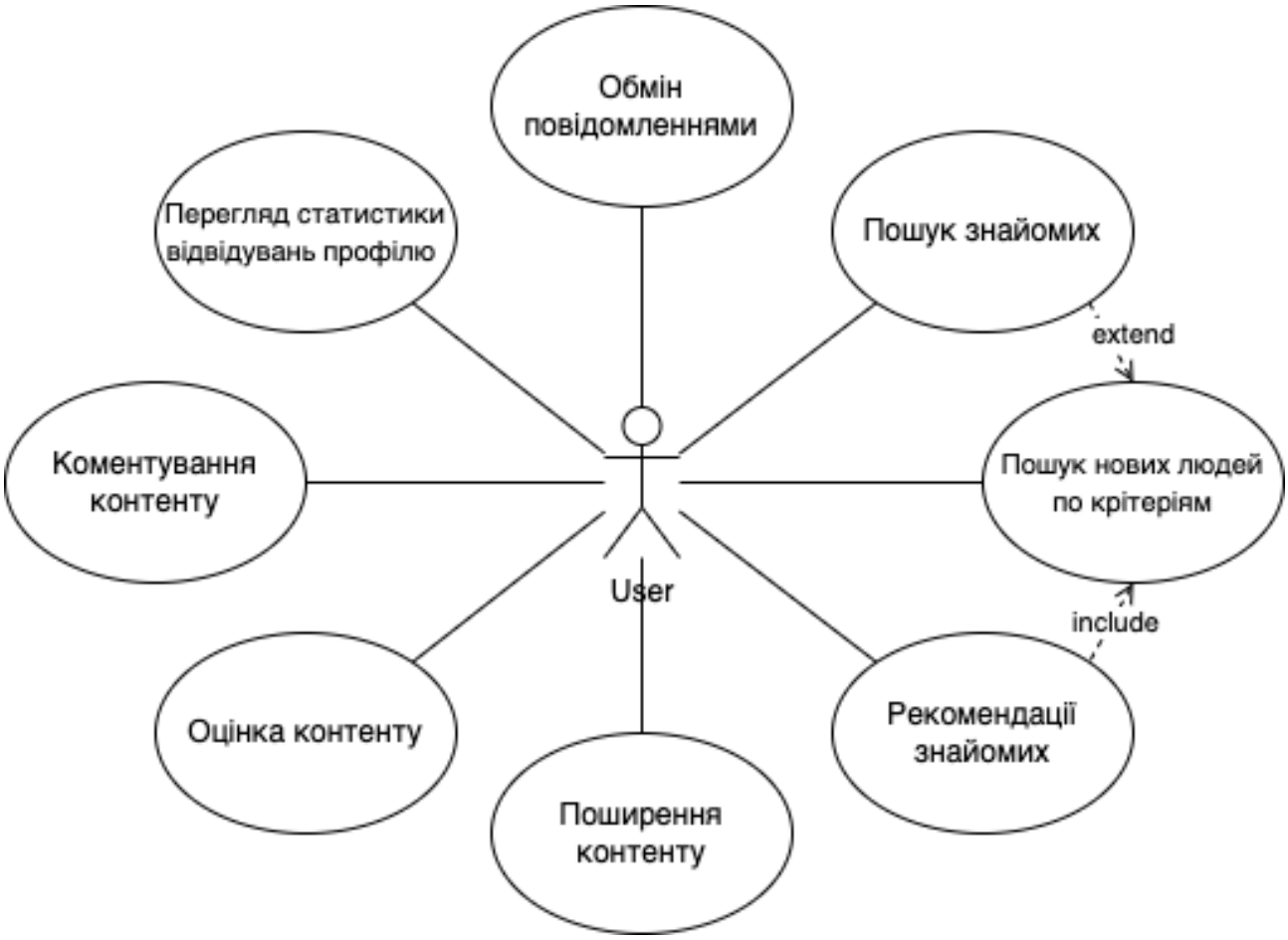


Рисунок 2.1 - UML Діаграма прецедентів системи

Опис дій у системі:

- Обмін повідомленнями – у користувача є можливість написати будь якому користувачу в системі повідомлення.
- Пошук знайомих - у користувача є можливість знаходити нових знайомих за ім’ям.

- Пошук нових людей по критеріям розширює пошук за ім’ям, дозволяючи шукати за інтересами, віком, статтю.
- Рекомендації знайомих - автоматичні рекомендації користувачів на яких користувач ще не підписаний, алгоритм рекомендацій включають в себе механізм ручного пошуку та розширює його аналізом схожості інтересів.
- Поширення контенту – у користувача є можливість публікувати фото та відео контент, з дописами.
- Оцінка контенту – у користувача є можливість ставити вподобання на публікації інших користувачів.
- Коментування контенту - у користувача є можливість коментувати дописи і публікації інших користувачів, коментарі бачать усі користувачі додатку.
- Перегляд статистики відвідувань профілю – користувач має можливість переглядати статистику по своєму профілю за останній тиждень.

2.2.2. Розробка структури бази даних системи

При розробці додатку було використано нереляційну базу даних Mongo DB, основними факторами які вплинули на вибір цієї бази даних є можливості для створення різноманітних індексів та агрегуючі поля, такі як пошук по геолокації, оцінка співпадіння знайденого документа у відсотках. Структуру бази даних наведено на рисунку 2.2.

Складається вона з колекцій, але основними є тільки 5.

Chat – колекція для зберігання інформації про створений чат приклад запису у колекції наведений на рисунку 2.3.

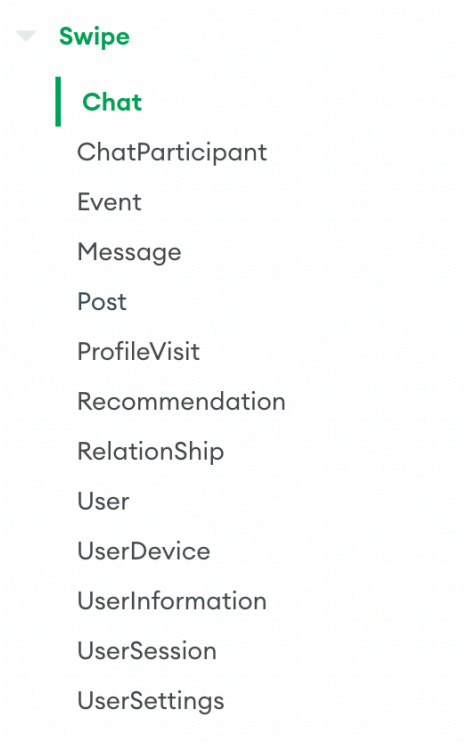


Рисунок 2.2 – Структура бази даних

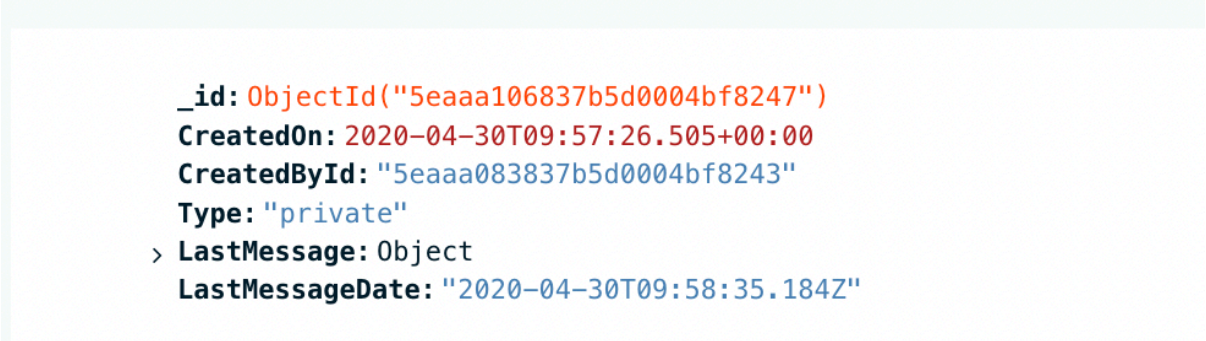


Рисунок 2.3 - Приклад запису у колекції чатів

ChatParticipant – колекція для зберігання інформації про учасників чату приклад запису у колекції наведений на рисунку 2.4.

Рисунок 2.4 - приклад запису у колекції учасників чату

Message – колекція для зберігання інформації про повідомлення у чаті приклад запису у колекції наведений на рисунку 2.5


```
_id: ObjectId("5ecd35ebadf25e0004c86a39")
UserId: "5ecd35cbadf25e0004c86a34"
UserName: "Herdas"
ChatId: "5ecd35e1adf25e0004c86a36"
CreatedById: "5ecd35cbadf25e0004c86a34"
CreatedOn: 2020-05-26T15:29:47.651+00:00
Body: "Привет"
IsNew: true
```


Рисунок 2.5 - Приклад запису у колекції повідомлень

Post - колекція для зберігання інформації про опубліковані новини приклад запису у колекції наведений на рисунку 2.6.


```
_id: ObjectId("5ea5e7c91cd90700043f5d11")
Image: "http://res.cloudinary.com/cdnwebswipe/image/upload/v1587931081/gdhrhgf..."
            "Добро пожаловать в Swipe!
Description:
            С нами найти новых друзей так же легко, как..."
Tags: Array
   0: "Swipe"
UserLikes: Array
   0: "5e9da2b1664f1500043d12f4"
CreatedById: "5e9da2b1664f1500043d12f4"
CreatedOn: 2020-04-26T19:58:01.950+00:00
```


Рисунок 2.6 - Приклад запису у колекції публікацій

User - колекція для зберігання інформації про користувачів системи приклад запису у колекції наведений на рисунку 2.7.


```
_id: ObjectId("5e9da2b1664f1500043d12f4")
UserName: "Swipe"
Password: "
CreatedOn: 2020-04-20T13:25:05.929+00:00
CreatedById: "5e9da2b1664f1500043d12f4"
Image: "http://res.cloudinary.com/cdnwebswipe/image/upload/v1587913372/zi0rbme..."
IsVerified: true
IsOnline: true
LastLogin: 2021-04-02T21:18:51.436+00:00
LastExit: "2021-02-14T21:36:00.845Z"
```


Рисунок 2.7 – Приклад запису у колекції користувачів

Також було створено індекс за допомогою можливостям Mongo DB для оптимізованого пошуку по схожим інтересам (рис. 2.8)

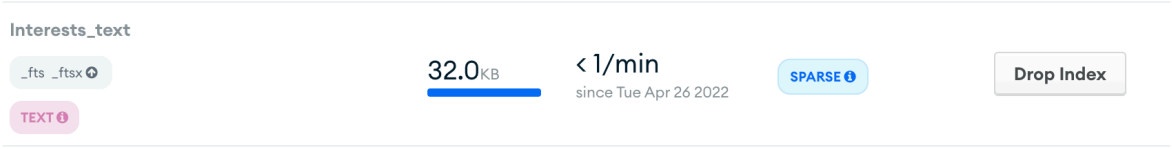


Рисунок 2.8 – Зображення індексу

2.3 Структура системи

2.3.1. Структура клієнтської частини

Клієнтська частина реалізована за допомогою бібліотеки Vue, доповнення до Vue з готових UI-компонентів Vuetify та технологій PWA для встановлення додатку на пристрої користувачів, що робить додаток крос-платформним. Для досягнення відчуття нативного для платформи додатку та реалізації push-повідомлень використовується service worker. Service Worker — це скрипт, який працює у фоновому режимі браузера без взаємодії з користувачем. Крім того, він нагадує проксі-сервер, який працює на стороні користувача. За допомогою цього сценарію можнате відстежувати мережевий трафік сторінки, керувати push-сповіщеннями та розробляти веб-додатки за допомогою Cache API, це надає можливість зберігати у пристрої завантажений контент при першому відкриті, та використовувати його для наступних цей підхід надає можливість користуватись додатком навіть без підключення до мережі.

Життєвий цикл Service Worker складається з 5 етапів:

- встановлення;
- активування;
- простій;
- зупинка;
- отримання даних.

Перехід між етапами зображений на рисунку 2.9.

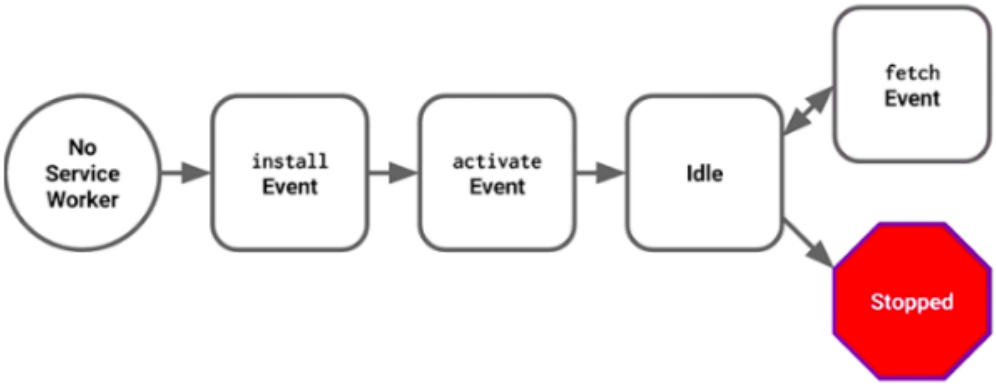


Рисунок 2.9 – Життєвий цикл Service Worker

2.3.2. Структура серверної частини

Серверна частина створена у вигляді набору rest-сервісів. Використовує бібліотеку express.js для надання api для інтеграції з сервером, такий підхід дозволить в майбутньому створювати нові клієнтські додатки не змінюючи бізнес-логіку.

При початку роботи користувач має авторизуватись за допомогою своїх облікових даних, отримати публічний ключ і зберегти його для відправки майбутніх запитів до серверу, закріпивши у заголовку http-запиту. Сервер зберігає приватний ключ, та його відповідність до конкретного користувача у тимчасове сховище у додатку, і встановлює термін дії ключу, після закінчення .

Після авторизації користувачу доступні тільки ті дані, які дозволені правилами додатку, або користувач є автором їх. Для цього на серверу реалізовані алгоритми перевірки публічного ключу у заголовку в запиті, для того, щоб визначити якому користувачу належить публічний ключ переданий у заголовку http-запиту, та чи не вийшов термін дії ключу, якщо, перевірка успішна, користувач може отримати дані.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ ДОДАТКУ

3.1 Оцінка та планування розробки проекту

Один за кращих підходів до розробки продуктів є створення MVP (Minimum Viable Product), цей підхід дозволяє перевірити зацікавленість аудиторії у використанні додатку, сформувати наступні кроки реалізації відштовхуючись від отриманого зворотного зв'язку, і розвивати продукт в залежності від реальних потреб користувачів. Цей підхід включає у себе розробку тільки важливих функцій програми. Тому для аналізу які з функцій є найважливіші на першому етапі, було проведено оцінку за принципом Key Features First (KFF). Основні функціональні вимоги зазначені на рисунку 3.1.

№	Вимога
	Авторизація
	Обмін повідомленнями у режимі реального часу
	Публікація новин
	Стрічка новин, яка відображає усі публікації авторів які цікавлять користувача
	Коментування публікацій
	Оцінка публікацій
	Пошук людей по ФІО
	Пошук людей по схожим інтересам
	Автоматично згенеровані рекомендації людей на основі інтересів

Рисунок 3.1. – Функціональні вимоги системи

Оцінка часу необхідного для розробки продукту була проведена за допомогою Experience Based техніки. Після формування функціональних вимог, і проведення оцінки на розробку кожного з пунктів, було проведено ABC аналіз. За

оцінкою на розробку потрібно 155 годин, але наразі доступно тільки 100 для розробки. ABC аналіз допоміг залишити тільки пріоритетні задачі.

В результаті аналізу було прийнято рішення реалізувати усі пункти з категорії A, та 1 та 4 пункт з категорії B, остатні 10 годин витратити на початок реалізації обміну повідомлень, але з обмеженим функціоналом (рис. 3.2).

№	Вимога	Критерій необхідності (1-10)	Оцінка часу на розробку (год.)	Важливість	Відсоток важливості	Категорія
1	Авторизація	10	10	100	9,4%	B
2	Обмін повідомленнями у режимі реального часу	5	20	100	9,4%	B
3	Публікація новин	7	10	70	6,6%	C
4	Стрічка новин, яка відображає усі публікації авторів які цікавлять користувача	7	20	140	13,1%	B
5	Коментування публікацій	5	10	50	4,7%	C
6	Оцінка публікацій	5	10	50	4,7%	C
7	Пошук людей по ФІО	3	15	45	4,2%	C
8	Пошук людей по схожим інтересам	9	30	270	25,4%	A
9	Автоматично згенеровані рекомендації людей на основі інтересів	8	30	240	22,5%	A
		Сумма	155	1065	100%	

Рисунок 3.2. – Використання ABC аналізу для виявлення пріоритетних вимог

Для відстеження статусу проекту, використано інструмент Jira, усі задачі були описані та добавлені у нього (рис. 3.3).

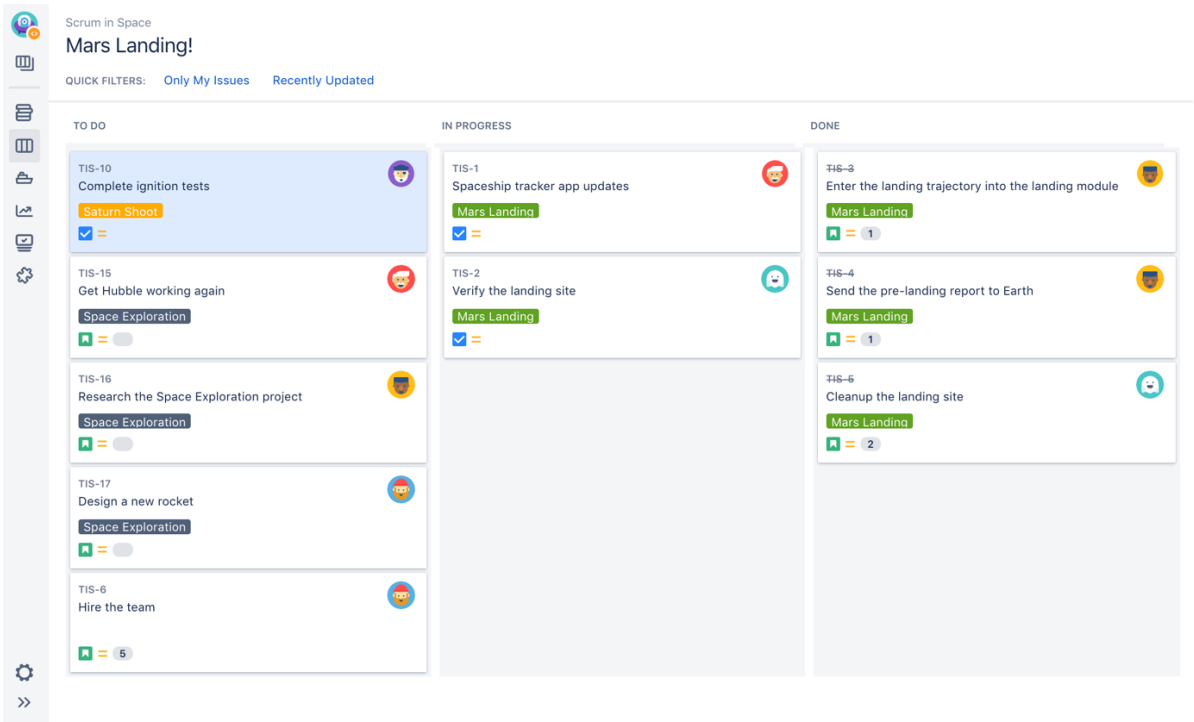


Рисунок 3.3 – Інструмент Jira для відстеження статусу розробки продукту

Кожна вимога була описана в стилі User Story, що дозволяє фіксувати потреби користувачів без деталізації, а на основі цих вимог формувати задачі на розробників, це дозволяє з опису основних задач зрозуміти як буде використовуватись система та її поведінка (рис 3.4).

Вимога
Я як користувач маю мати можливість знаходити нових знайомих за розширеним фільтром
Мені як користувачу необхідно мати можливість переглядати автоматичні рекомендації
Мені як користувачу необхідно розуміти чому саме цей користувач мені автоматично рекомендується
Мені як користувачу необхідно мати можливість публікувати фото-відео контент
Мені як користувачу необхідно мати можливість обмінюватись повідомленнями
Мені як користувачу необхідно мати можливість підписуватися на публікації інших користувачів
Мені як користувачу необхідно мати можливість коментувати і оцінювати публікації

Рисунок 3.4 – Список вимог в форматі опису user story

3.2. Набір інструментів використаних для розробки

Для середовища розробники було обрано Visual Studio Code (рис. 3.5), так як вона більш за все адаптована для написання коду програм з використанням веб технологій та мов JS, HTML, CSS та має багато налаштувань, та готових доповнень, а також в неї:

- продуманий та зручний інтерфейс;
- підтримка більшості мов програмування;
- інтеграція з системою контроля версій Git.

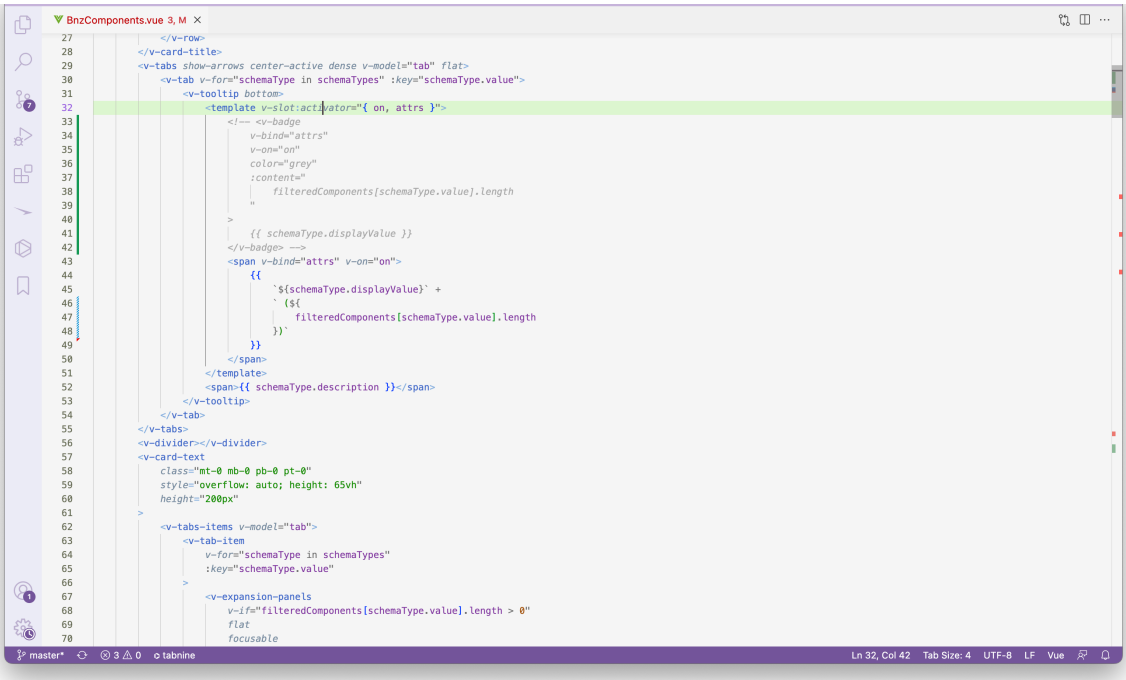


Рисунок 3.5. – Visual Studio Code

Для роботи з базою даних було обрано MongoDB Compass – додаток для підключення до серверу бази даних та адміністрування їх. (рис. 3.6).

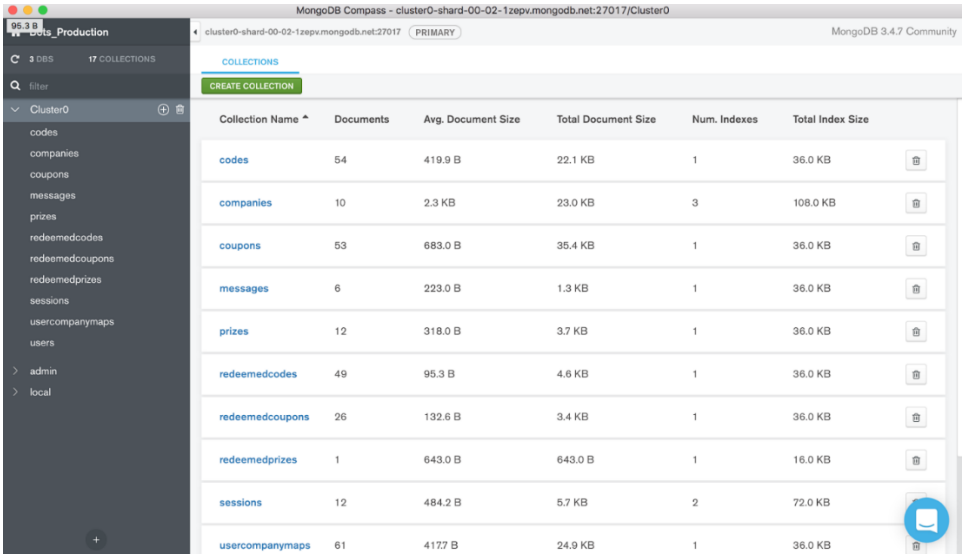


Рисунок 3.6. – Mongo DB Compass

Для швидкого тестування api серверної частини було обрано Postman (рис 3.7.), він дозволяє формувати будь-які HTTP запити, та отримувати результат

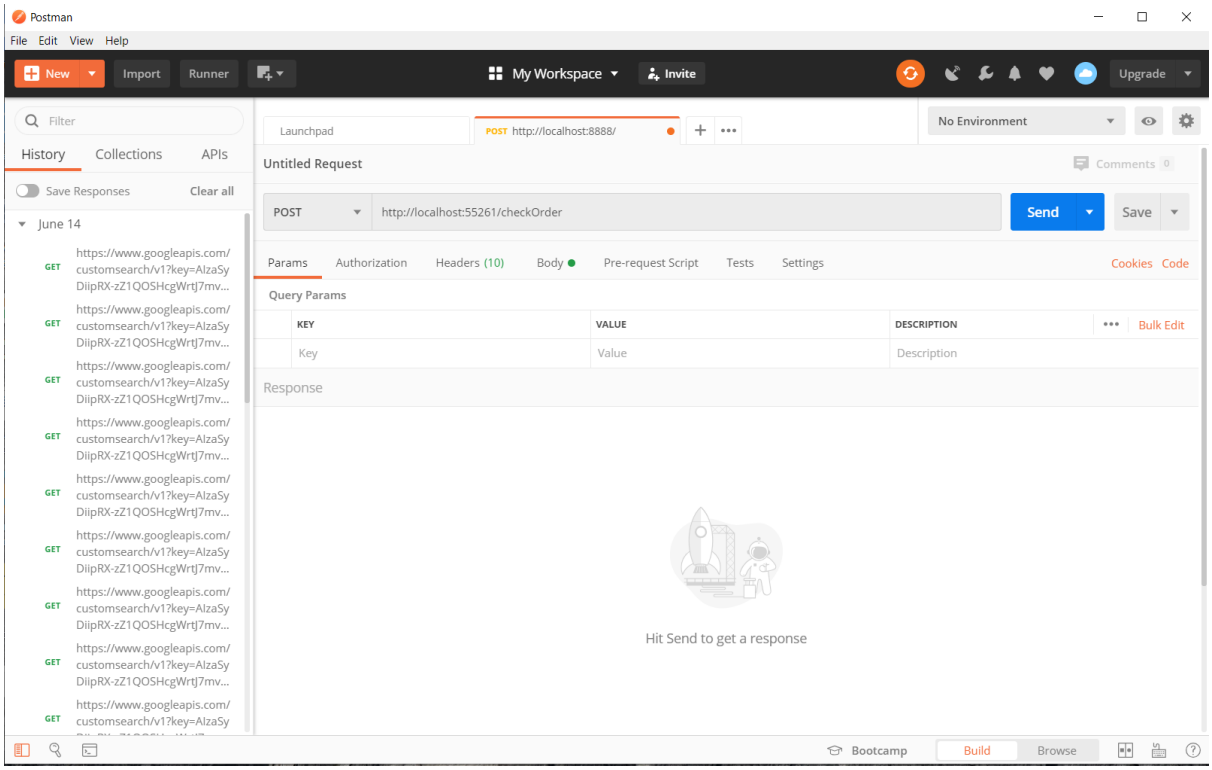


Рисунок 3.7 – Postman

3.3. Функціонал клієнтської частини розширення та його модулі

Клієнтська частина складається з 8 компонентів:

- модуль чату – реалізує логіку обміну повідомленнями у режимі реального часу;
- модуль новин – реалізує перегляд, створення, редагування новин, коментарі та оцінки контенту;
- модуль рекомендацій – реалізує список з відображенням рекомендованих людей користувачу;
- модуль сповіщень – модуль для відображення повідомлень користувачу;
- модуль для навчання при першому запуску – реалізує авторизацію та реєстрацію, підказки при першому запуску;
- модуль «розумного пошуку» - реалізує процес розумного пошуку;
- модуль керування меню – керує навігацією у додатку;
- модуль користувачів – відображення інформації про користувача, редагування, перегляд статистики.

3.4. Функціонал серверу та його модулі

Серверна частина реалізує наступні сервіси:

- сервіс керування користувачами – реалізує механізм авторизації та реєстрації користувачів;
- сервіс керування даними – універсальний сервіс для виконання crud-операцій над колекціями у системі;
- сервіс чату – реалізує логіку обміну повідомленнями та керування чатом;
- сервіс взаємозв’язків – реалізує методи для встановлення взаємозв’язків між користувачами;
- сервіс керування новинами – реалізує методи по управлінню публікаціями користувача;
- сервіс push-повідомлень – реалізує логіку сповіщення;
- сервіс розумного пошуку – реалізує алгоритми пошуку людей для рекомендацій користувачу.

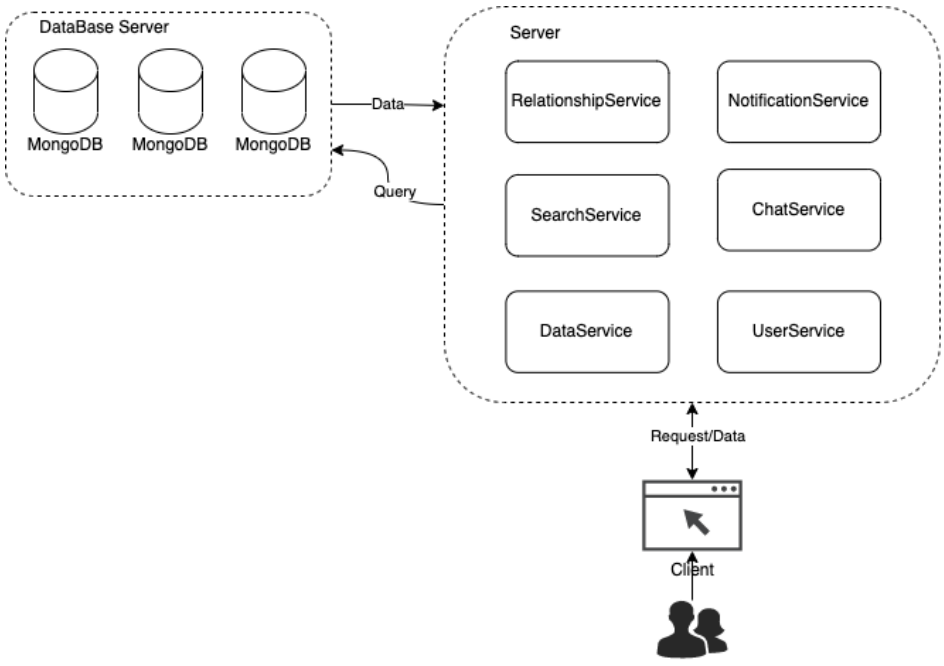


Рисунок 3.8 – Архітектура системи

4. ОПИС ФУНКЦІОНУВАННЯ ТА ТЕСТУВАННЯ СИСТЕМИ

4.1. Опис роботи додатку для комунікації між інтернет користувачами

4.1.1. Процес авторизації у додатку

Для початку роботи у додатку, потрібно авторизуватись або зареєструватись у додатку, реєстрація складається з трьох кроків (рис 4.1):

1) заповнення логіну та паролю;
2) вибір аватару користувача;
3) заповнення інформації про себе.

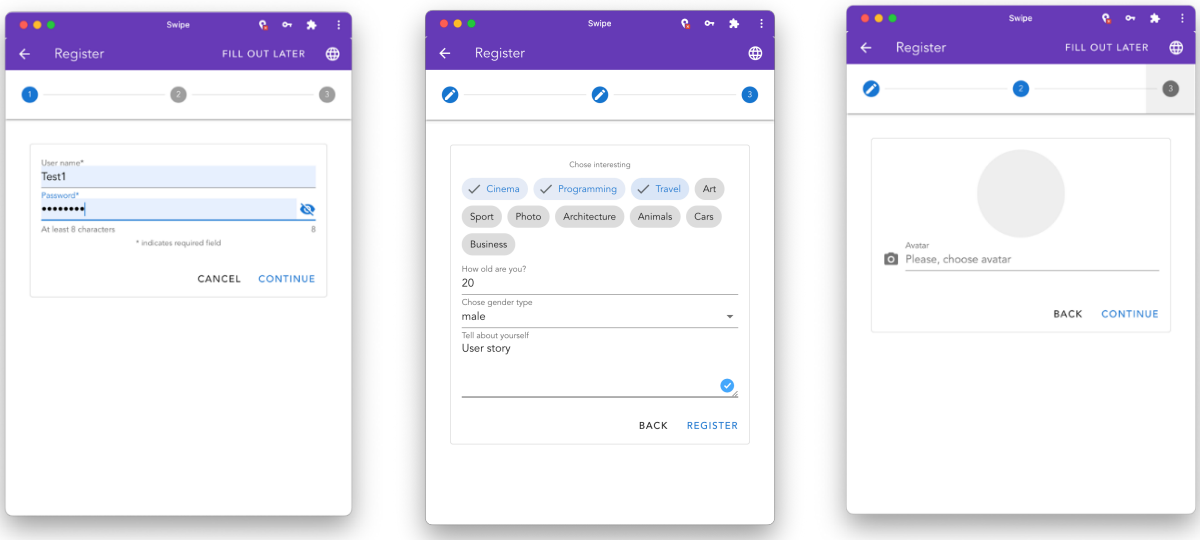


Рисунок 4.1 – Приклад реєстрації

Другий та третій крок є необов’язковим, якщо їх пропустити, то система буде рекомендувати користувачу при наступному логіні до додатку додати інформацію про себе (рис. 4.2).

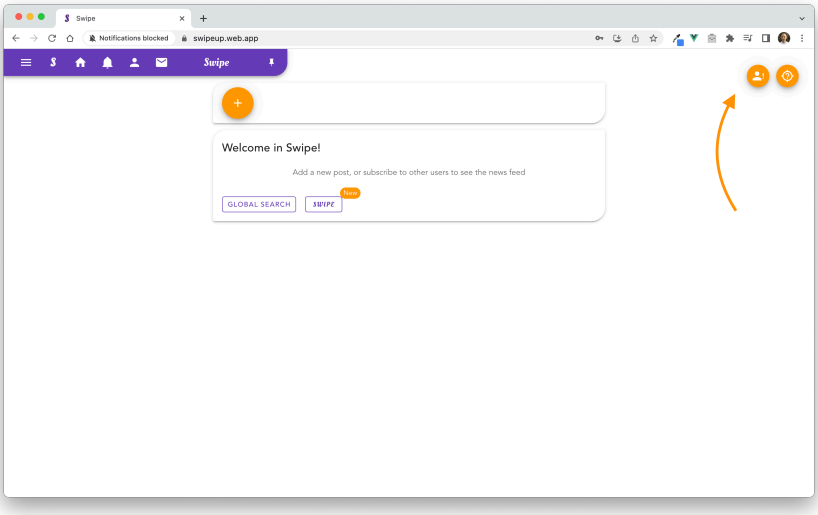


Рисунок 4.2 – Приклад рекомендації у веб-версії додатку

4.1.2. Процес пошуку людей за схожими інтересами

Для того, щоб знайти людей зі схожими інтересами необхідно:

1) відкрити меню;
2) вказати стать для пошуку;
3) вказати приблизний вік;
4) вказати інтереси по яким необхідно здійснювати пошук, за замовченням будуть указані інтереси користувача.

У додатку в спеціальному контейнері на головній сторінці, є блок рекомендацій у якому відображається відсоток схожості користувачів (рис. 4.3).

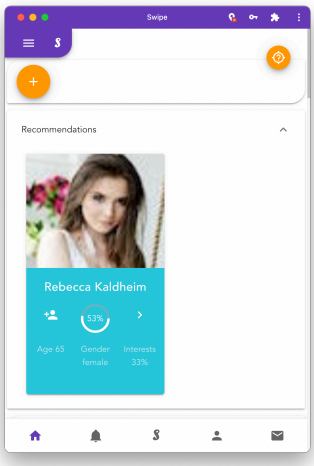


Рисунок 4.4 – автоматичні рекомендації

Після пошуку буде відкрито вікно з результатами пошуку (рис. 4.4).

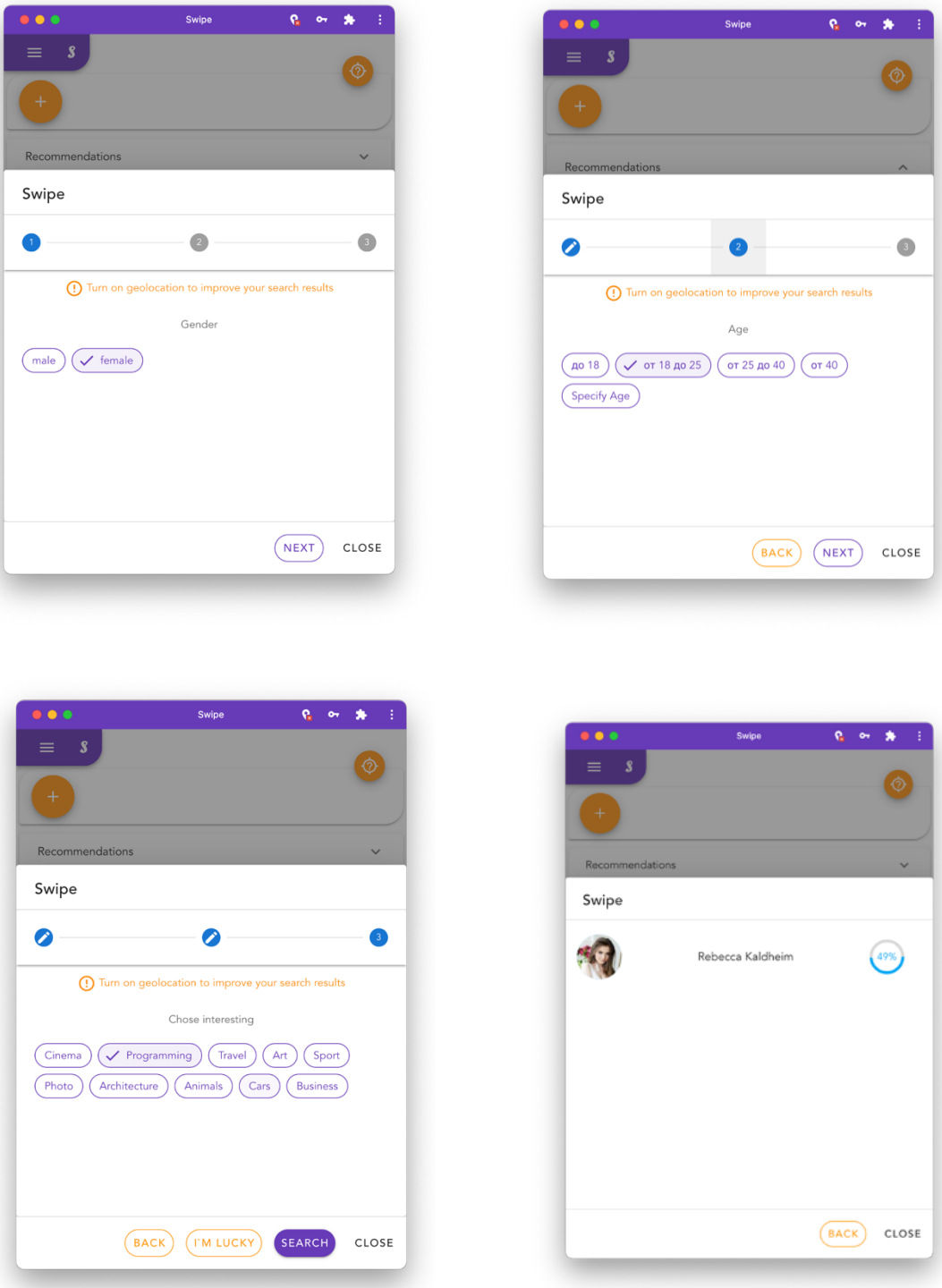


Рисунок 4.4 – Режим пошуку людей

4.1.2. Процес публікації новин

У додатку є можливість поширювати контент, для цього було створено стрічку новин від людей на яких підписаний користувач. Щоб створити новину потрібно натиснути кнопку «додати», та заповнити поля у три кроки (рис.4.5).

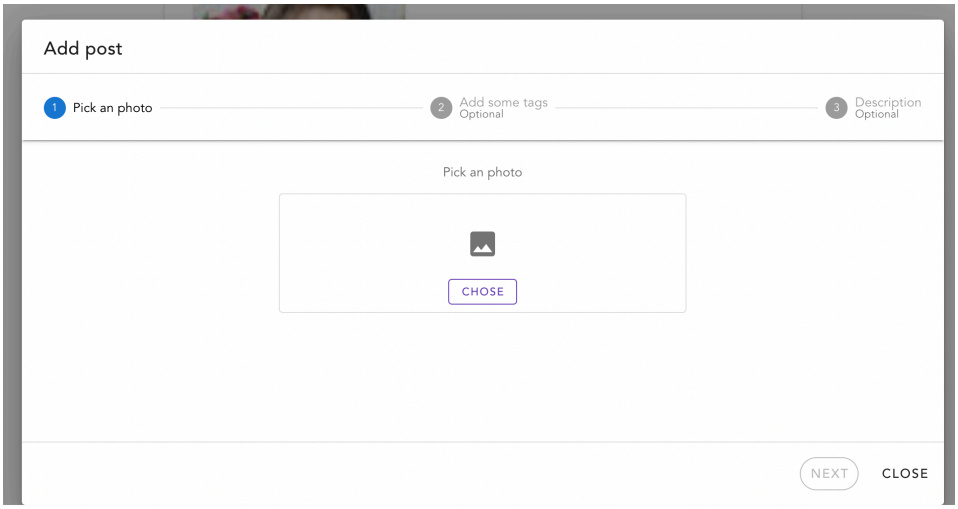


Рисунок 4.5 – Приклад інтерфейсу створення новин

4.1.3. Процес обміну повідомленнями

У додатку є можливість обмінюватись повідомленнями та відправляти реакції, після відправки повідомлення користувачу прийде push-повідомлення.

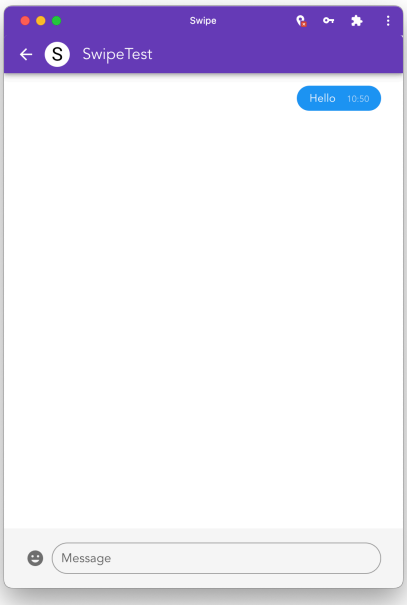


Рисунок 4.6 – Приклад відправки повідомлення

4.2. Набір тестових сценаріїв для забезпечення якості продукту

Для контролю якості продукту необхідно розробити тест кейси програми (рис. 4.7).

Id	Summary	Roles	Precondition	Steps	Expected result	Status
TC-01	[Auth] Login	User	User has opened app	1) Fill credentials 2) Click "Login"	User redirected to tape news with loaded publications from interesting peoples	Pass
TC-02	[Auth] Register	User	User has opened app	1) Click "Create account" 2) Fill credentials 3) Choose avatar 4) Fill information about user 5) Click "Register"	User redirected to welcome page	Pass
TC-02	[Recommendation] Recommendation module in tape news	User	User has opened app, and login	1) Click "Recommendation" container	User can see list of interesting peoples	Pass
TC-03	[Chat] Send message	User	User has opened chat list and chosen the chat member	1) Click "Enter message" 2) Fill message 3) Send message	Chat member receive the message immediately	Fail
TC-04	[News Tape]	User	User has opened app, and login	1) Go to tape news	User can see list of publications from subscriptions	Pass

Рисунок 4.7. – Приклад функціональних тест кейсів для тестування системи

Створено тест–сет функціонального тестування продукту, в якому акцент на негативні та конфліктні сценарії. Для розробки тест-сету використовувались наступні техніки тест дизайну:

- передбачення помилок на основі попереднього досвіду;
- діаграми станів та переходів;
- тестування use-кейсів.

4.3. Результати апробації та подальший розвиток проекту

Апробація даного продукту була проведена на реальних користувачах, завдяки рекламі додатку у інших мережах. В результаті рекламної компанії було залучено користувачів різних країн (рис. 4.8).

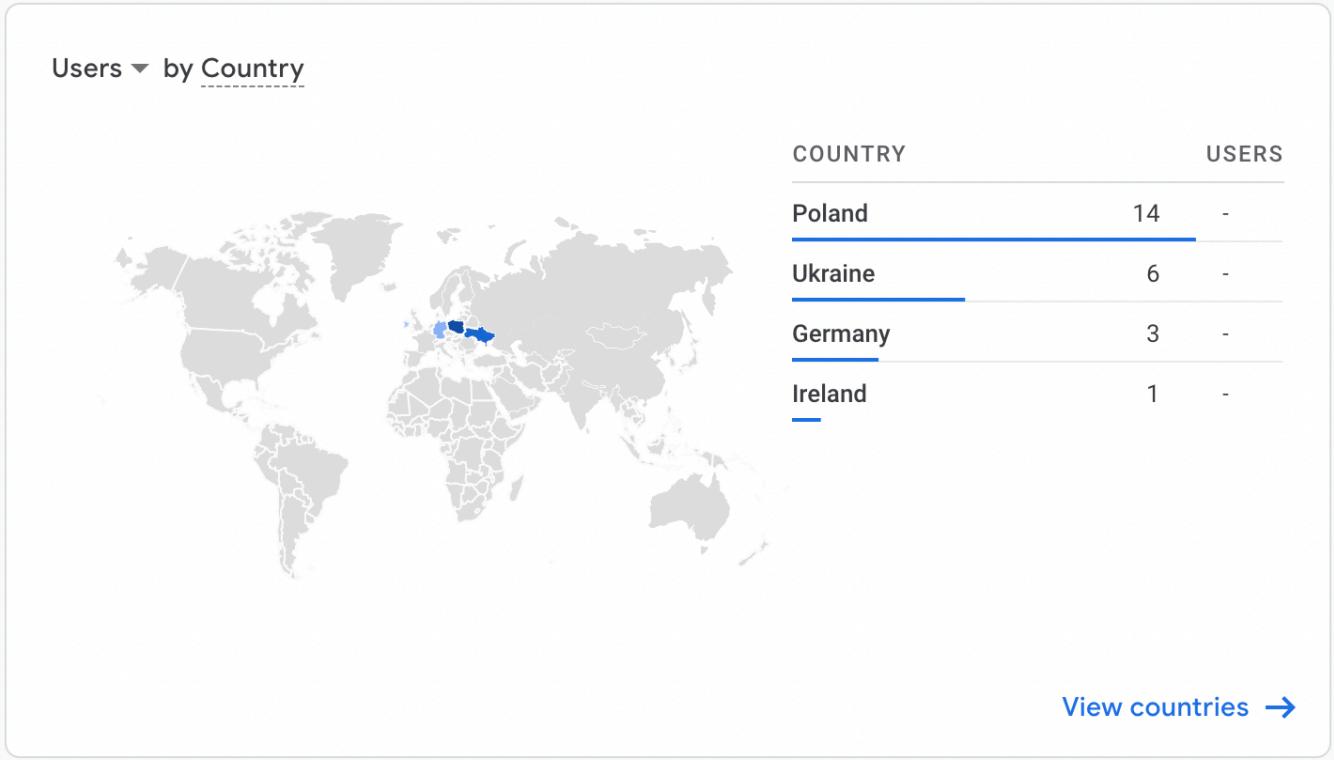


Рисунок 4.8. – Статистика залучення користувачів

Було проаналізовано основні події користувачів у системі (рис. 4.9)

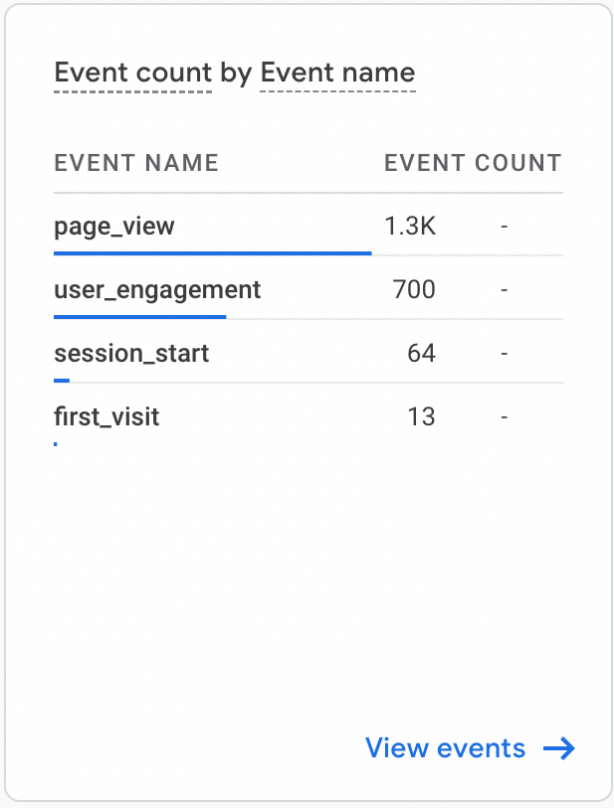


Рисунок 4.9. – Статистика подій

Також було проаналізовано середній час який користувач проводить у додатку (рис 4.10).

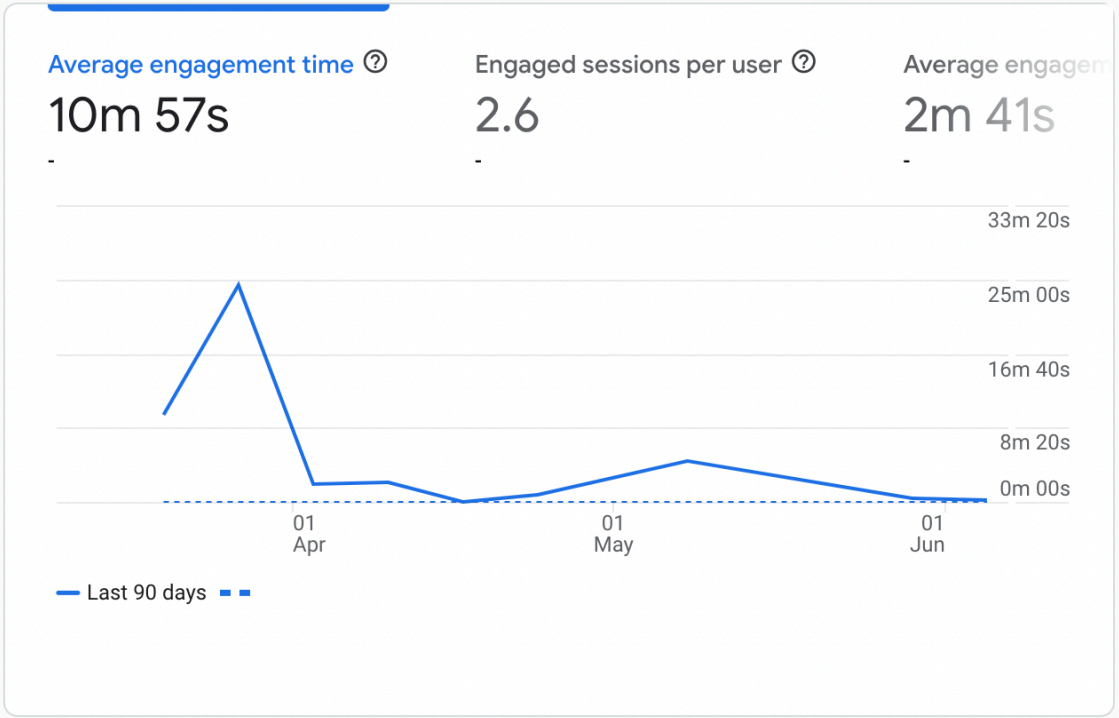


Рисунок 4.10 – Статистика подій

Протягом тестів було виявлено сильні та слабкі сторони програми. Основними перевагами виявились:

- простота використання;
- розумний пошук людей;
- рекомендації користувачу;
- стабільність роботи основних функцій;
- легкий для розуміння інтерфейс;

Декілька зауважень стосовно функцій, яких не вистачило опитаним користувачам:

- неможливо відправити картинку у повідомленні;
- неможливість поширити відео контент;
- неможливість створити групу для обміну повідомленнями.

В результаті апробації продукт показав високу зацікавленість від користувачів, та підтвердив теорію про корисність пошуку людей по схожим

інтересам. Було сформовано план подальшого розвитку продукту:

- можливість поширювати відео-контент;
- можливість відправляти фото та відео-контент у повідомленнях;
- створювати групові чати;
- покращити алгоритми для пошуку людей;
- покращити якість обміну повідомленнями.

ВИСНОВКИ

Дана робота була спрямована на проектування та реалізацію інструменту для комунікації між інтернет користувачами.

1. Обґрунтовано актуальність розробленого продукту та його наукову новизну шляхом аналізу інших схожих додатків. Було досліджено типові потреби користувачів інтернет-ресурсів для комунікації та виявлено, що аудиторія потребує інструментів, які можуть спростити пошук людей по схожим інтересам.

2. Розроблено розділену архітектуру додатку з використанням клієнт-серверну архітектуру, оскільки це обумовлено стандартами розробки веб-додатків. Обрано інструменти для побудови продукту, такі як, Node JS, Mongo DB, Vue JS, Vuetify, обґрунтовано перевагу та ефективність над іншими аналогами. Для того щоб розробити успішний продукт, було проаналізовано потреби користувача при роботі з аналогічними додатками та створено набір вимог у вигляді UML діаграми. В ході розробки системи було реалізовано можливість обмінюватись повідомленнями, поширювати контент, підписуватись на цікавих людей, та шукати нових за допомогою розумного пошуку. Було використано Mongo DB для збереження даних.

3. Описано програмні засоби та інструменти, котрі було застосовано для розробки програмного забезпечення. Виявлено що зручним середовищем розробки є VS Code. Підібрано допоміжні інструменти розробки, такі як Postman для тестування розроблених API, та MongoDB Compass для тестування БД. Відстеження статусу виконання проекту було за допомогою інструменту Jira.

4. Відповідність системи визначеним вимогам забезпечує набір тестових сценаріїв, котрий покриває всі функціональні вимоги системи, з урахуванням граничних випадків, та перевірки їх обробки системою.

5. Створений продукт цілеспрямований на допомогу у пошуку людей зі схожими інтересами у соціальній мережі. Визначено, що інструмент може використовуватись для пошуку бізнес-партнерів, обміну повідомленнями, розвитку особистого бренду.

6. Роботу було апробовано серед реальних користувачів. Було зібрано статистику використання, проаналізовано та виявлено ряд переваг та недоліків системи. Також було розроблено план розвитку продукту. Виявлено, що основними перевагами програми є:

- простота використання;
- розумний пошук людей;
- рекомендації користувачу;
- стабільність роботи основних функцій;
- легкий для розуміння інтерфейс.

Результати дослідження бакалаврської роботи апробовані на Всеукраїнських науково-технічних конференціях: «Сучасні інтелектуальні інформаційні технології в науці та освіті» та «Застосування програмного забезпечення в інфокомунікаційних технологіях».

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. developers.google.com [Електронний ресурс] – Режим доступу до ресурсу: https://developers.google.com/ Дата звернення: 01.05.2022

2. Google Chrome API Reference [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://developer.chrome.com/docs/extensions/reference/ Дата звернення: 01.05.2022

3. [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://github.com/nodejs/node/README.md Дата звернення: 02.04.2022.

4. Douglas Crockford. JSON: The Fat-Free Alternative to XML, 2009 – 55 p.

5. Mike Richardson, Ruby Leonard, Sam Amundsen.RESTful Web APIs./-“O'Reilly Media”, 2013 – c. 357 – 363.

6. Eric Elliott. Programming JavaScript Applications: Robust Web Architecture with Node, HTML5, and Moderns JS Libraries. -/ “O’Reilly Media”, 2013 – c. 134– 177.

7. Leavitt Neal.Will NoSQL Databases Live Up to Their Promise./-“IEEE Computer”, 2010 – c. 43 – 45.

8. Офіційний веб-сайт Mongo DB. [Електронний ресурс]: [Веб–сайт]. – електронні дані. – Режим доступу: https://www.mongodb.com/ Дата звернення: 03.05.2022.

9. Mardan A. Practical Node.js: Building Real-World Scalable Web Apps / Azat Mardan., 2018. – 505 с.

10. Brown E. Web Development with Node and Express: Leveraging the JavaScript Stack / Ethan Brown., 2014. – 346 с.

11. Wilson M. Node: Up and Running: Scalable Server-Side Code with JavaScript / M. Wilson, T. Hughes-Croucher., 2012. – 204 с.

12. Hume D. Progressive Web Apps / Dean Alan Hume., 2007. – 200 с.

13. Love C. Progressive Web Application Development by Example / Chris Love., 2018. – 354 с.

14. Халецький В. С. Огляд програмного забезпечення для організації комунікації користувачів інтернет-ресурсів : Матеріали всеукраїнської науково-

технічної конференції: «Застосування програмного забезпечення в інфокомунікаційних технологіях». Збірник тез.\- 20.04.2022, ДУТ, м. Київ – К.: ДУТ, 2022.– С.80-81

15. Халецький В. С. Використання інтелектуального аналізу данних для інтерактивної взаємодії з користувачами у програмному забезпечені : III Міжнародної студентської наукової конференції «Діджиталізація науки як виклик сьогодення ». 03.06.2022, м. Львів, Україна, 2022.– С.181-182.

Додаток А

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

Державний університет телекомунікацій
Навчально-науковий інститут інформаційних технологій
Кафедра інженерії програмного забезпечення

Розробка SPA-додатку для комунікації користувачів інтернет ресурсу на основі технології Node.js

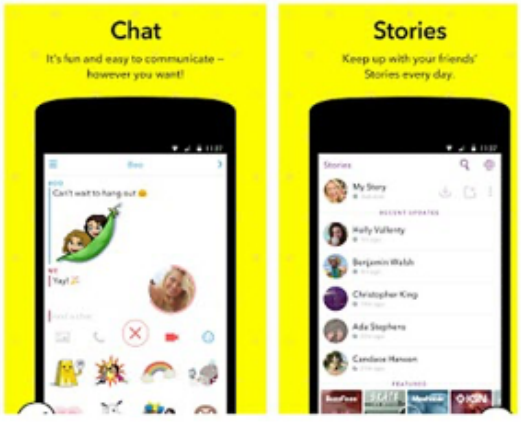
Виконав:
студент 4 курсу групи ПД-44
Халецький Віталій Сергійович

Керівник роботи:
к.т.н., доцент, Золотухіна Оксана Анатоліївна

Київ 2022

АНАЛОГИ

Instagram



SnapChat

АНАЛІЗ АНАЛОГІВ

Назва продукту	Переваги	Недоліки
Instagram	1. Просунуті алгоритми відображення новин 2. Просунуті алгоритми обміну повідомленнями 3. Присутні алгоритми пошуку зв'язків між користувачами	1. Незрозуміло як створюються рекомендації 2. Неможливість управляти рекомендаціями 3. Відсутність можливості переглядати % співпадіння з рекомендованими людьми 4. Відсутність можливості шукати знайомих за розширеним фільтром
SnapChat	1. Пошук знайомих за номером телефону 2. Присутні алгоритми пошуку зв'язків між користувачами 3. Вбудовані інструменти роботи з фото та відео контентом	

3

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи – підвищення якості процесів комунікації користувачів інтернет ресурсів за рахунок використання програмного забезпечення на основі технології Node.js

Об'єкт дослідження – процеси комунікації користувачів інтернет ресурсів.

Предмет дослідження – програмні засоби для забезпечення комунікації користувачів інтернет ресурсів

4

ТЕХНІЧНЕ ЗАВДАННЯ

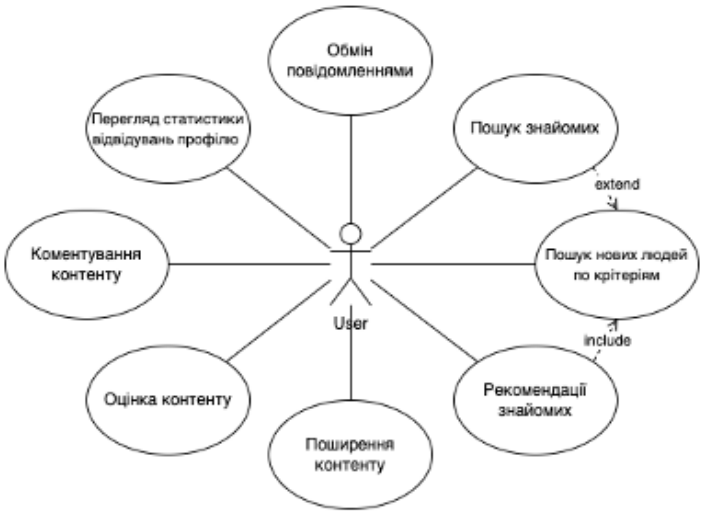
1. Розробити серверну частину для надання доступу до сервісів реалізуючих бізнес-логіку додатку
2. Розробити основні модулі клієнтської частини
3. Реалізувати можливість обмінюватись повідомленнями у режимі реального часу
4. Реалізувати можливість публікувати контент, оцінювати та коментувати його
5. Реалізувати можливість пошуку знайомих по заданому фільтру
6. Реалізувати можливість пошуку цікавих людей по схожим інтересам

5

ЗАСОБИ ТА ІНСТРУМЕНТИ РЕАЛІЗАЦІЇ

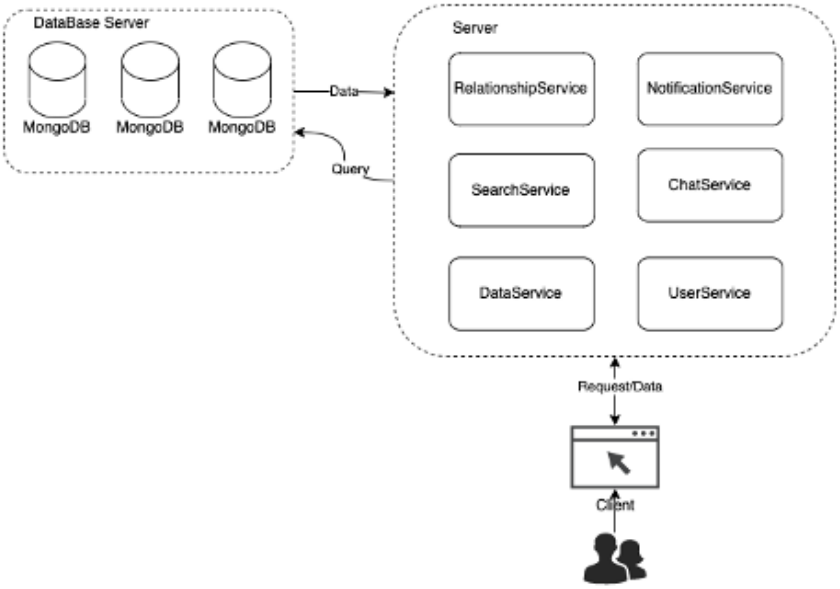
6

ДІАГРАМА ПРЕЦЕДЕНТІВ



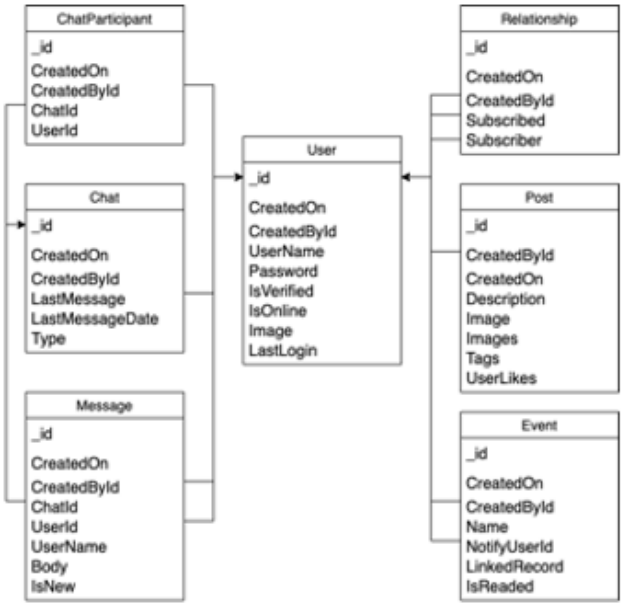
7

СХЕМА АРХІТЕКТУРИ ДОДАТКУ



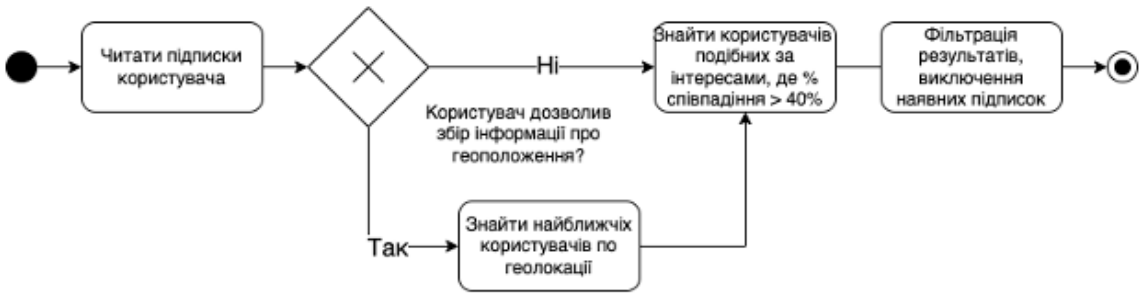
8

СХЕМА БАЗИ ДАНИХ



9

ЗАГАЛЬНИЙ АЛГОРИТМ РЕКОМЕНДАЦІЙ

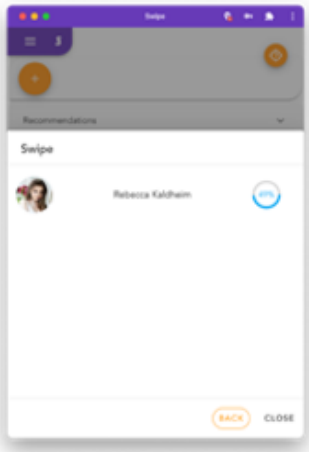


10

ЕКРАННІ ФОРМИ

Реєстрація

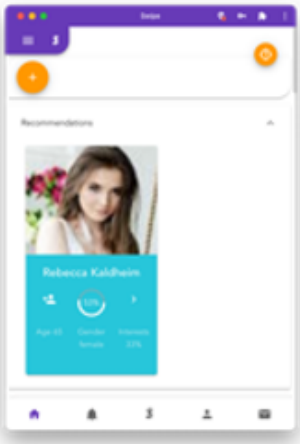
Пошук по розширеним фільтрам



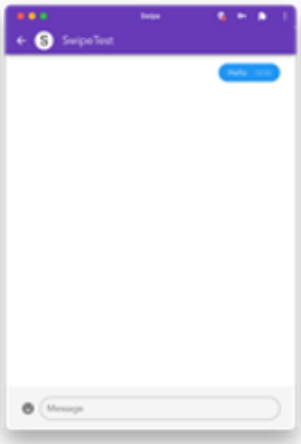
Результати пошуку

11

ЕКРАННІ ФОРМИ



Автоматичні рекомендації



Відправка повідомлень

12

ДЕМОНСТРАЦІЯ РОБОТИ ДОДАТКУ

13

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

1. Халецький В. С. Огляд програмного забезпечення для організації комунікації користувачів інтернет-ресурсів : Матеріали всеукраїнської науково-технічної конференції: «Застосування програмного забезпечення в інфокомунікаційних технологіях». Збірник тез.\- 20.04.2022, ДУТ, м. Київ – К.: ДУТ, 2022.– С.80-81
2. Халецький В. С. Використання інтелектуального аналізу данних для інтерактивної взаємодії з користувачами у програмному забезпечені : III Міжнародної студентської наукової конференції «Діджиталізація науки як виклик сьогодення ». 03.06.2022, м. Львів, Україна, 2022.– С.181-182.

14

ВИСНОВКИ

1. Проведено аналіз очікувань користувачів при роботі з додатками для комунікацій. Розглянуто основні види програм, та способи їх використання.
2. Виконано огляд інструментальних засобів, які можуть бути використані для комунікації в інтернет ресурсах. Серед програмних засобів увага приділялася додаткам, які реалізовані для мобільних пристроїв.
3. Проведено огляд засобів програмної реалізації додатку. Описано переваги використання платформи Node JS при розробці кросплатформних додатків та особливості використання платформи.
4. В ході розробки системи було реалізовано можливість обмінюватись повідомленнями, поширювати контент, підписуватись на цікавих людей, та шукати нових за допомогою розумного пошуку.
5. Реалізовано автоматичні рекомендації людей користувачу
6. Реалізовано механізм сповіщення користувачів

ДЯКУЮ ЗА УВАГУ!