

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки, факультет радіoeлектроніки та
телекомунікацій
(повне найменування інституту, факультету)

Кафедра Захисту інформації
(найменування кафедри)

Пояснювальна записка

до магістерської роботи
на тему: Розробка веб месенджера для захищеного обміну повідомленнями

Виконав: студент 2 курсу, групи РТ-719М

Спеціальності 125 Кібербезпека
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Системи технічного захисту інформації,
автоматизація її обробки

Слива О.М.

(прізвище та ініціали)

Керівник Неласа Г.В.

(прізвище та ініціали)

Рецензент Киричук Т.Т.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(організм наймистецтва закладу вищої освіти)

Інститут, факультет: Інститут інформатики та радіоелектроніки, Факультет
радіоелектроніки та телекомунікацій

Кафедра: Захисту інформації

Ступінь вищої освіти: магістр

Спеціальність: 125 Кібербезпека
(код і назва спеціальності)

Освітня програма (спеціалізація): Системи технічного захисту інформації,
автоматизація її обробки

(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри Карпуков Л.М.

Докт. техн. наук, професор

«06» «09» 2020 року

З А В Д А Н Н Я
НА МАГІСТЕРСКУ РОБОТУ СТУДЕНТА

Слива Олександр Миколайович

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи): Розробка веб месенджера для захищеного обміну
повідомленнями

керівник проекту (роботи) Неласа Г.В. к.т.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від «06» листопада 2020 року
№314

2. Строк подання студентом проекту (роботи): 15 грудня 2020 року

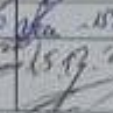
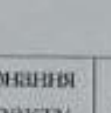
3. Вихідні дані до проекту (роботи) Захист web-сайту від несанкціонованого
доступу. Аналіз існуючих бібліотек та програми для розробки та захисту
web-сайту.

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

1. Аналіз існуючих бібліотек для розробки клієнтської частини веб додатку
2. Отримання з основними поняттями та визначеннями про базу даних
3. Аналіз існуючих бібліотек для розробки серверної частини веб додатку
4. Розробка веб-додатку
5. Розробка бібліотеки шифрування

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень):
18 рисунків; 14 таблиці; 5 додатків; презентація магістерської роботи

6. Консультанти розділів проекту (роботи):

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	Прийняв виконан є завдання
Основні розділи	Неласа Г.В., професор		
Розділ з ОП і НС	Якімцов Ю.В., доцент		
Розділ з економіки	Круглікова В.В., доцент		
Нормоконтроль	Корольков Р.Ю., старший викладач		

7. Дата видачі завдання «01» вересня 2020 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Складання та затвердження ТЗ	06.09 – 11.09	Виконано
2	Підбір літератури	11.09 – 15.09	Виконано
3	Аналіз існуючих бібліотек для розробки клієнтської частини веб додатку	15.09 – 30.09	Виконано
4	Ознайомлення з основними поняттями та визначеннями про базу даних.	01.10 – 10.10	Виконано
5	Аналіз бази даних SQL	10.10 – 16.10	Виконано
6	Аналіз системи управління бази даних postgresql	17.10 – 27.10	Виконано
7	Аналіз існуючих платформ для розробки серверної частини веб додатку	28.10 – 05.11	Виконано
8	Розробка веб-додатку	05.11 – 25.11	Виконано
9	Розробка бібліотеки для шифрування	05.11 – 09.12	Виконано
10	Оформлення пояснювальної записки	10.12 – 12.12	Виконано
11	Оформлення графічної частини	10.12 – 12.12	Виконано
12	Економічна частина	25.11 – 05.12	Виконано
13	Охорона праці	18.11 – 27.11	Виконано

Студент

Керівник проекту (роботи)


(підпис)


(підпис)

Слива О.М.
(прізвище та ініціали)

Неласа Г.В.
(прізвище та ініціали)

РЕФЕРАТ

ПЗ: 112 сторінок, 18 рисунків, 14 таблиць, 16 джерел, 5 додатків.

Тема роботи – Розробка веб месенджера для захищеного обміну повідомленнями

Об'єкт дослідження – процес передачі інформації при взаємодії користувачів веб додатку.

Предмет дослідження – методи шифрування, хешування та організації захищеного каналу зв'язку.

Мета дослідження – розробка веб сайту соціальної мережі контактів за спільними хобі інтересами з організацією захищеного каналу спілкування.

В ході роботи розроблено проект веб застосування у вигляді UML діаграм, розроблено відповідний код клієнт серверного додатку та власну криптографічну бібліотеку для node.js з реалізацією сучасного українського блочного шифру Калина(ДСТУ 7624:2014).

АЛГОРИТМ, БАЗА ДАНИХ, ЗАХИСТ ВІД НСД, СИСТЕМА УПРАВЛІННЯ БАЗАМИ ДАНИХ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ЦІЛІСНІСТЬ ДАНИХ, ШИФРУВАННЯ, ХЕШУВАННЯ.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ.....	7
ВСТУП.....	8
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.....	9
1.1 Постановка задачі	15
1.1.1 Ведення.....	15
1.1.2 Призначення розробки	15
1.1.3 Вимоги до програми	15
1.1.4 Вимоги до надійності	16
1.1.5 Вимоги до інформаційної та програмної сумісності	16
1.1.6 Вимога до транспортування та зберігання	16
1.1.7 Стадії і етапи розробки.....	16
2 ВИБІР ТЕХНОЛОГІЇ.....	17
2.1 Введення в бази даних. Основні поняття і визначення.....	19
2.1.1 Життєвий цикл баз даних	20
2.1.2 Архітектури баз даних.....	20
2.2 Вибір технології для розробки серверної частини	22
3 РЕАЛІЗАЦІЯ ВЕБ ДОДАТКУ	25
3.1 Загальна структура сайту та його основний функціонал.....	25
3.2 Розробка інтерфейсу веб-додатку	26
3.3 Створення веб-додатку	30
3.3.1 Написання серверної частини	34
3.3.2 Створення сторінки реєстрації та авторизації користувача	38
3.3.3 Створення сторінки профіль.	39
3.3.4 Створення сторінки пошуку.....	40
3.3.5 Створення сторінки редагування даних	40
3.3.6 Створення чату.....	41
3.4 Розробка бібліотеки шифрування з використанням алгоритма Калина.....	42
4 ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ ТЕХНІЧНОЇ ЧАСТИНИ	47
4.1 Аналіз маркетингової частини проекту «Розробка веб месенджера для захищеного обміну повідомленнями»	47
4.2 Визначення витрат на розробку веб месенджера для захищеного обміну повідомленнями.....	52

5 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ	
.....	60
5.1 Аналіз потенційних небезпек	60
5.2 Заходи по забезпеченню техніки безпеки.....	61
5.3 Заходи щодо виробничої санітарії та гігієни праці.....	66
5.4 Заходи безпеки у надзвичайних ситуаціях	70
5.4.1 Заходи з пожежної безпеки	70
5.4.2 Заходи по забезпеченню організації навчання працюючого та непрацюючого населення діям у надзвичайних ситуаціях.....	73
ВИСНОВКИ	77
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ	78

ПЕРЕЛІК СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

БД – база даних;

СУБД – система управління базами даних;

ПК – персональний комп'ютер;

ПЗ – програмне забезпечення;

CBC – Cipher Block Chaining (Режим зчеплення блоків шифротекста)

CFB – Cipher Feed Back (Режим зворотного зв'язку по шифротексту)

CTR – Counter Режим лічильника

OFB – Output Feed Back (Режим зворотного зв'язку по виходу)

ECB – Electronic Code Book (Режим електронної кодової книги)

ВСТУП

Інтернет став, найбільш ефективним засобом реклами і просування і є одним з важливих елементів сучасної цивілізації. Інтернет може задовольнити всі потреби сучасної людини: це покупки, висновок ділових відносин, пошук клієнтів і так далі. Але багато заходять в Інтернет читати, розважатися, спілкуватися, дізнаватися щось нове для себе. Тому створення сайтів стало таким популярним.

В даній роботі ми вирішили розробити сайт з зручним та зрозумілим інтерфейсом де користувачі зможуть спілкуватися один з одним та ділитися новинами. Також вони зможуть публікувати фотографії на своїй сторінці, шукати інших людей за допомогою пошуку де можна буде встановлювати параметрами за якими вони хочуть знайти ту чи іншу людину. Наприклад: місце проживання, вік, зріст, хобі та багато інших. Під час спілкування люди часто можуть надсилати свої конфіденційні дані такі як: логін, пароль, дані про картку, номер телефону та інше. Тому вони часто задають питання чи захищені їх данні від зловмисників. Щоб вони не змогли використати їх у своїх цілях.

Для цього часто програмісти використовують бібліотеки в яких реалізовані алгоритми шифрування та хешування. За допомогою яких можна як захистити конфіденційні дані від несанкціонованого доступу. Тому ми вирішили розробити бібліотеку в якій буде реалізований один із сучасних алгоритмів шифрування «Калина».

Актуальність полягає у досягненні максимальної ефективності захисту сайту за рахунок одночасного використання всіх необхідних методів що виключають несанкціонований доступ до інформації.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

В сучасному світі існує дуже багато захищених веб месенджерів такі як: Facebook, Telegram, Chanty, Lark. Але всі веб месенджери зустрічаються з однією проблемою як захистити дані від несанкціонованого доступу. На даний час самим захищеним сайтом вважають Facebook, тому що він включає в себе двухфакторну автентифікацію, наскрізне шифрування, а також має можливість обмежити доступ до інформації. Багато месенджерів для захисту інформації використовують бібліотеки де реалізовано алгоритм шифрування AES.

Безпека веб-додатку – є одним з найбільш важливих питань в сучасній інформаційній безпеці. Зазвичай більшість сайтів, постійно піддаються атакам і мають вразливості різного роду.

Основні аспекти захисту інформації:

- конфіденційність - несанкціонований доступ до даних;
- цілісність - несанкціоноване спотворення або знищення даних;
- доступність - обмеження або блокування доступу до даних.

Основною загрозою для безпеки сайту є хакерська атака.

Хакерська атака — це дія кібер-зловмисника, яка спрямована на виведення з роботи веб-додатку, перехоплення інформації, а також отримання контролю над веб-додатком.

Види атак на веб-додатки.

1) Цільові атаки - це атаки, які направлені на конкретний сайт. Небезпека таких атак полягає в тому що ці атаки в основному провадять люди які вже добре обізнані в галузі захисту веб-додатків. За часту такі атаки роблять для того щоб дістати конфіденційну інформацію для отримання прибутку або ж навпаки для зменшення прибутку конкурентів.

2) Нецільові атаки - це атаки, які спрямовані на різні сайти але за певними критеріями. Наприклад на сайти які використовують одну СУБД.

Такого роду атаки намагаються охопити максимальну кількість сайтів при мінімумі витрат. Якщо ж така атака вдалась на певному сайті то зловмисник намагається отримати контроль над базою даних, внести свій код або ж додати ще одного адміністратора щоб мати повний доступ до веб додатку.

Основні типи атак.

1) Метод ін'єкції – такий тип атаки базується на тому, щоб в робочу систему можна було внести свій сторонній код, який зможе забезпечити доступ до закритої інформації, або вивести з ладу саму систему.

2) DoS - (Denial of Service - Відмова в обслуговуванні) – ця атака спрямована на виведення з ладу сервера щоб він не міг відповідати на запити. Такий тип атаки не призначений для отримання секретної інформації, але може бути допоміжними в ініціалізації інших атак.

3) DDoS - (Distributed Denial of Service) - має таку ж мету що і DoS атака, але за часту їх проводять не з одного комп'ютера, а за допомогою декількох комп'ютерів в мережі. DDoS-атаку використовують для виникнення помилок, що призводять до виведення з ладу сервісу, або спрацьовування захисту, що приводить до блокування роботи, а згодом навіть і до відмови в обслуговуванні. DDoS використовується там, де звичайний DoS є неефективним. Для цього об'єднуються декілька комп'ютерів, які здійснюють DoS атаку на конкретну систему.

4) Перехоплення пакетів. Дані за часту можуть передаватися у незашифрованому вигляді, тому хакерам дуже легко отримати конфіденційну інформацію про користувача за допомогою програмного забезпечення, що називають сніфером. Наприклад: пароль, логін, телефон місце проживання та багато іншого.

5) Мейлбомбінг (розсилка спаму) – це коли на сервери приходить дуже велика кількість листів і з часом він не має змогу їх всіх обробити і просто падає. Таку атаку міг зробити любий недосвідчений хакер. На даний час розробили фільтри які обмежують кількість листів за їх темою, а також за

ідентифікаційним номером відправника (id), але навіть це інколи не допомагає.

6) Мережна розвідка. При цій атаці хакер намагається отримати доступ до сайту щоб дізнатися яке програмне забезпечення використовується на веб-додатку, але не робить не яких змін і не вносить стороннього коду в веб-додаток. Тому ця атака і називається розвідкою.

Загрози від атак.

1) Загроза працездатності сайту. Це може привести до репутаційних і фінансових втрат компанії.

2) Загроза збереження даних користувача. Це може призвести до нанесення як моральних так і матеріальних витрат користувача.

3) Хакери використовують сайт для атак на інші ресурси, для розсилки спаму або проведення DoS атак. В результаті чого пошуковики і браузері блокують сайт, і він згодом втрачає користувачів.

4) Атака на сайт в корпоративному середовищі може бути точкою входу до корпоративної мережі компанії.

5) Атаки на системи електронної комерції можуть бути використані для здійснення шахрайських дій, викрадення клієнтських баз і т.д.

6) Атаки можуть бути націлені на подальше «зараження» користувачів сайту, наприклад за допомогою засобів експлуатації вразливостей браузерів чи їх компонентів

Можна виділити кілька основних способів захистити свій сайт:

- забезпечити захист від DDoS-атак;
- підключити SSL-сертифікат;
- використовувати надійний хостинг ;
- використовувати безпечні бібліотеки;
- використовувати існуючі техніки захисту від SQL-ін'єкцій;
- розробити моніторинг дій користувачів на сайті;
- робити регулярне резервне копіювання;

- використовувати надійні і складні паролі, а також захист від перебору паролів;

- в разі наявності адміністративної панелі, за допомогою якої відбувається управління вмістом веб-сайту, необхідно змінити стандартний адреса входу і забезпечити контроль доступу.

Після розгляду загальної теорії головною цілю нашої роботи є розробка сайту з простим і зрозумілим інтерфейсом, де користувач зможе спілкуватися з іншими користувачами, ділитися новинами, публікувати свої фото, а також знаходити інших людей за допомогою сторінки пошуку, де він зможе вказати стать, зріст, вік, регіон проживання, ім'я та багато інших параметрів. Для того щоб було легше знаходити іншого користувача.

Також нам буде необхідно розробити криптографічну бібліотеку для шифрування даних в якій буде реалізовано алгоритм шифрування Калина. Спочатку необхідно проаналізувати алгоритм шифрування та зрозуміти як він працює. Потім написати програмне забезпечення в якій буде реалізовано шифрування та дешифрування даних користувача.

В криптографії, шифрування це трансформація відкритого тексту в кодований текст або шифротекст. Шифротекст призначений бути нечитабельним для неавторизованих читачів.

Шифрування - це перетворення даних в шифр текст за допомогою алгоритмів шифрування. Самі алгоритми знаходяться в публічному доступі. Для того щоб зашифрувати та розшифрувати дані необхідно встановити ключ, який повинен залишатися таємним, щоб хакери не змогли дістати конфіденційну інформацію про користувача.

Хешування - перетворення вхідного масиву даних довільної довжини у вихідний бітовий рядок фіксованої довжини. Такі перетворення називаються хеш-функціями, результатом яких є - хеш-код. Для порівняння даних використовують хешування, в якому є два результати: якщо хеш-код однаковий, то дані скоріш за все однакові; якщо хеш-код різний, то дані були змінені.

Потім були розглянуті наступні криптографічні бібліотеки: CryptoJS, Pidcrypt, AES-JS.

CryptoJS - це постійно зростаюча колекція стандартних і безпечних криптографічних алгоритмів, реалізованих на JavaScript з використанням передових методів і шаблонів. Вони швидкі, і у них є послідовний і простий інтерфейс.

CryptoJS підтримує наступні режими блочного шифру Rijndael: CBC (за замовчуванням), CFB, CTR, OFB, ECB.

Позитивні якості бібліотеки CryptoJS:

- багатий вибір алгоритмів як шифрування так і хешування;
- послідовний і простий інтерфейс;
- швидке перетворення тексту в хеш або в зашифровану строку.

Негативні сторони бібліотеки CryptoJS:

- необхідно мати великий об'єм пам'яті на жорсткому диску.

Pidcrypt - це копія криптографічної бібліотеки pidCrypt javascript з деякими змінами, щоб зробити її сумісною з node.js і браузером. Всі під модулі Pidcrypt доступні через require і при необхідності автоматично підключаються до будь-яких інших залежностей під модулів.

Позитивні якості бібліотеки Pidcrypt:

- багатий вибір алгоритмів як шифрування так і хешування, але менше ніж у CryptoJS;
- послідовний і простий інтерфейс.

Негативні сторони бібліотеки Pidcrypt:

- необхідно великий об'єм пам'яті;
- має тільки два режими CTR, CBC.

AES-JS – це реалізація алгоритму блочного шифрування AES на чистому JavaScript.

Існує кілька режимів роботи, кожен з яких має різні плюси і мінуси. Однак загалом рекомендуються режими CBC та CTR . Не рекомендується ECB, і включений насамперед для повноти.

Позитивні якості бібліотеки AES-JS:

- має багато додаткових функцій які облегшують роботу;
- необхідно малий об'єм пам'яті;
- швидкий.

Негативні сторони бібліотеки AES-JS:

- тільки один вид шифрування.

Далі розглянемо які алгоритми присутні в кожній із бібліотек (див. таблицю 1.1.)

Таблиця 1.1 – Таблиця алгоритмів які присутні в криптографічних бібліотеках

	CryptoJS	Pidcrypt	AES-JS
SHA-1	+	+	-
SHA-256	+	+	-
SHA-512	+	+	-
SHA-3	+	-	-
RIPEMD-160	+	-	-
AES	+	+	+
DES	+	-	-
Rabbit	+	-	-
RC4	+	-	-
MD-5	+	+	-
RSA	-	+	-

Таким чином було проведено аналіз існуючих методів вирішення проблеми захисту даних від несанкціонованого доступу, а також були проаналізовані бібліотеки які допомагають вирішити дане питання.

1.1 Постановка задачі

1.1.1 Ведення

«Програма для спілкування між користувачами» - це програмне забезпечення, яке дозволяє користувачу знайти іншого користувача за допомогою пошук де він може вказувати стать, зріст, вік та багато інших параметрів, а також програма надає змогу переписуватись з тією людиною яку вибрав користувач або подивитися доступну інформацію про цього користувача.

1.1.2 Призначення розробки

Данна програма призначена для полегшення спілкування між користувачами, а також для захисту їх персональних даних від зловмисника.

1.1.3 Вимоги до програми

Програма складається з двох основних компонентів: клієнтської і серверної частин, між якими має бути налагоджено взаємодія.

Сервер повинен обробляти належним чином запити, які надходять від клієнта. Наприклад, якщо клієнт хоче отримати дані сервер повинен спочатку перевірити чи існує такий клієнт в базі даних, а також потім перевірити чи має він доступ до них. Якщо клієнт має доступ то сервер відправляє йому дані, якщо він не має доступ то сервер відправляє повідомлення де вказано по якій причині він не має доступу. Також сервер повинен вірно обробляти помилки. Наприклад, якщо база даних не працює або невірно записано запит на отримання даних то він також повинен відправити повідомлення з кодом помилки, щоб користувач міг звернутися до адміністрації і повідомити яка помилка відбулась щоб це виправити або надати інформацію чому це відбулося.

Клієнтська частина повинна бути реалізована у вигляді веб-додатку, що запускається в браузері, і представлена у вигляді зручного інтерфейса де користувач має змогу переписуватися з іншими користувачами та обмінюватися з ними інформацією. А також повідомляти про всі помилки та відповіді які надійшли з серверної частини.

1.1.4 Вимоги до надійності

Користувач, який працює з програмою через веб-браузер повинен бути забезпечений безперервний доступ до веб-додатку, розташований за певною адресою. Якщо веб-додаток перервав свою роботу то необхідно повідомити користувача. У разі відмови роботи серверної частини, час відновлення не повинно перевищувати одну робочу добу.

1.1.5 Вимоги до інформаційної та програмної сумісності

Програма повинна бути написана на мовах JS, HTML, CSS. Для спрощення розробки необхідно використовувати React для клієнтської частини та node.js для серверної частини. Для хешування паролів необхідно використовувати бібліотеку crypto, а для шифрування даних необхідно розробити та використати бібліотеку kalyna. Для управління базою даних використати СУБД під назвою postgres.

1.1.6 Вимога до транспортування та зберіганню

Веб-додаток зберігається в репозиторії онлайн-сервісу Git.

1.1.7 Стадії і етапи розробки

Перелік стадій та етапів розробки веб додатку:

- 1) Створення дизайну та навігації
- 2) Підняття проекту та установка необхідних бібліотеки для роботи з ним.
- 3) Створення бази даних та створення таблиць для зберігання даних
- 4) Створення серверної частини та підключення до бази даних
- 5) Написання коду для маніпуляції бази даних через сервер
- 6) Написання end-points для зв'язку між клієнтською та серверною частиною.
- 7) Написання коду для клієнтської частини.

2 ВИБІР ТЕХНОЛОГІЙ

На теперішній час існує дуже багато мов програмування але ми будемо розглядати мови які допомагають розробляти веб додаток, такі як:

- HTML (Hyper Text Markup Language – мова гіпертекстової розмітки) – це стандартизована мова розмітки веб-сторінок у інтернеті [7];

- CSS (Cascading Style Sheets – каскадні таблиці стилів) – це мова яка описує зовнішній вигляд веб-сторінок [7];

- JS (JavaScript) – це мова за допомогою якої ми описуємо послідовність дій які необхідно буде виконати програмі при виконанні деяких умов [7].

Для полегшення розробки веб сайту було розглянуто дві дуже відомі бібліотеки:

- Angular;
- React.

Це JavaScript-бібліотеки з відкритим вихідним кодом, до яких має доступ любий веб-розробник. Вони і зараз дуже швидко розвиваються та пишуть новий функціонал для полегшення розробки веб-додатків. Вони вирішують однакові front-end задачі, але кожна бібліотека підходить до їх вирішення по різному [10, 11].

До основних відмінностей між цими бібліотеками можна віднести:

- 1) Зберігання даних. React необхідно додаткові бібліотеки які допоможуть керувати та зберігати дані під час знаходження користувача на сайті, одна з таких бібліотек є Redux. Angular вже включає в себе роботу з даними тому нам не треба встановлювати додаткових бібліотек.

- 2) Virtual DOM – є головною відмінністю React від Angular. Тобто якщо дані були змінені то React оновить тільки ту частину коду де ці дані використовуються, а Angular буде оновлювати весь код що призводить інколи до довгого оновлення сторінки.

Переваги React:

- за основу беруть прості мови програмування;
- надзвичайна гнучкість програми;
- додаток може витримувати великі навантаження завдяки virtualDom;
- користувачами дуже легко взаємодіяти з нашим веб-додатком;
- дозволяє оновлювати тільки необхідні компоненти;
- має відкриту бібліотеку даних;
- забезпечує просту міграцію між версіями.

Недоліки React:

- велика кількість інструментів що затрудняє вибір який інструмент нам необхідний;
- на вивчення і освоєння всього матеріалу потрібен тривалий час.

Переваги Angular:

- функції взаємозалежні;
- інформацію можна отримувати безпосередньо без використання інших бібліотек;
- мінімальний ризик помилок;
- представлена можливість працювати окремо в одному розділі програми, використовуючи наявні дані.

Недоліки Angular:

- під час міграції між версіями можуть виникати помилки.

Проаналізувавши всі переваги та недоліки бібліотек було вибрано бібліотеку React, так як вона може витримувати великі навантаження і не робить зайвих оновлень інших компонентів, а оновлює тільки потрібну компоненту.

2.1 Введення в бази даних. Основні поняття і визначення.

База даних (БД) - це організована структура, яка представлена у вигляді таблиць і відносинами між ними, де зберігаються дані переважно в дуже великих обсягах [1].

Система управління базами даних (СУБД) - сукупність мовних та програмних засобів, за допомогою яких можливо створювати, обновляти, редагувати та видаляти дані з таблиць.

Основоположними поняттями в концепції баз даних є «дані» і «модель даних».

Дані - це набір конкретних параметрів та значень, які можуть характеризувати як об'єкт, умову, ситуацію та інші чинники.

Модель даних - це деяка абстракція, яка, будучи застосовні до конкретних даних, дозволяє користувачам і розробникам трактувати їх уже як інформацію, тобто відомості, що містять не тільки дані, але і взаємозв'язок між ними.

Види моделей даних:

- об'єктно-орієнтована модель;
- мережева модель;
- ієрархічна модель;
- реляційна модель.

Об'єктно-орієнтована – це модель яку використовують для створення великих БД які мають дані з складною структурою і об'єднують реляційну і мережеву модель.

Мережева – у цієї моделі дані організовані у вигляді графа.

Недоліком такої моделі є:

- жорсткість структури;
- складність організації.

Ієрархічна – представлена у вигляді дерева. Для оперування даним які мають складні логічні зв'язки ця модель підходить найбільше.

Реляційна – це сукупність таблиць та відносин між ними. Така модель дуже часто використовується у сучасному світі. Крім того її дуже легко реалізувати на комп'ютері

Переваги:

- простота;
- гнучкість структури.

2.1.1 Життєвий цикл баз даних

Життєвий цикл БД нерозривно пов'язаний із життєвим циклом інформаційної системи. Він складається з таких етапів:

- планування розробки БД;
- формулювання вимог до системи;
- збір та аналіз вимог користувачів;
- проектування БД;
- розробка прикладних програм;
- створення прототипів (необов'язково);
- реалізація;
- конвертування й завантаження даних;
- перетворення та завантаження даних;
- тестування;
- експлуатація та супровід.

Для маленьких БД, де присутні не велика кількість користувачів життєвий цикл буде не складним, а для великих БД, де присутні декілька десятків а то і сотні тисячі користувачів життєвий цикл буде складним.

2.1.2 Архітектури баз даних

Під архітектурою БД розуміють загальні принципи і способи організації і функціонування СУБД.

В даний час СУБД, прийнято відносити до однієї з архітектури:

а) Локальна архітектура.

І БД і СУБД розміщені на одному комп'ютері. В такій архітектурі працює більшість настільних СУБД середнього типу складності.

б) Архітектура “файл-сервер”

База даних розміщена на сервері, а ПК підключені до нього за допомогою локальної мережі. На цих комп'ютерах функціонують клієнтські СУБД, які звертаються до БД по мережі. Переваги:

- одночасно робить декількох користувачів з однією БД.

Недоліки:

- великий об'єми інформації, яка передаються по мережі;
- клієнтські ПК повинні бути оснащені достатньо потужними програмними і технічними ресурсами;
- не дуже висока надійність.

в) Архітектура “клієнт-сервер”

В архітектурі “клієнт-сервер” на сервері зберігається як БД, так і СУБД, яка обробляє запити користувачів. Це значно зменшує навантаження, оскільки велика частина роботи виконується на сервері. Одночасно зменшуються обсяги даних, що передаються по мережі. Переваги:

- зростає швидкість передачі даних;
- зменшується вимога до технічних і програмних засобів клієнта.

г) Розподілена архітектура

Ця архітектура відрізняється від “клієнт-сервер” тим, що в мережі функціонує декілька серверів. БД зберігається частинами на кожному сервері, при цьому для одних і тих же даних зберігається їх копія на іншому сервері. В такій архітектурі використовуються програми-оптимізатори, які дозволяють оптимізувати обробку великої кількості запитів від користувачів, а також рівномірно розподілити навантаження. Переваги :

- висока надійність;
- підвищена швидкість обробки даних за рахунок паралельної обробки запитів.

Недоліки:

- досить складний і економічно дорогий процес.

д) Інтернет-архітектура

Доступ до БД і СУБД, які розміщені на одному комп'ютері або в мережі, здійснюється із браузера по стандартному протоколу. Переваги:

- мінімальні вимоги до обладнання клієнта;
- системи легко створювати і використовувати, так як всі протоколи і інтерфейси взаємодії стандартизовані.

Недоліки:

- необхідність підключення до Інтернет;
- оплата послуг користування зв'язком.

2.2 Вибір технології для розробки серверної частини

Далі необхідно вибрати мову програмування для серверної частини. Для цього існує велика кількість мов програмування таких як: PHP, node.js та інші.

PHP- це мова програмування яка призначена для створення веб-сторінок на серверній частині, тобто коли ми переходимо на відповідний адрес ми робимо запит на отримання html коду для відображення сторінки [7].

Node.js – це платформа яка призначена для створення серверної частини з використанням мови програмування JavaScript [2].

Проаналізувавши дані дві мови для розробки серверної частини було вирішено використовувати node.js, так як вона використовує мову програмування JavaScript.

Сервер – це програмне забезпечення, де приймаються і обробляються запити які були надіслані від користувача і в залежності від ролі може надати доступ до цих даних чи ні.

Для маніпуляцій над даними нам необхідна СУБД, на даний момент їх існую безліч, наприклад такі як:

- MySQL;
- Postgresql.

MySQL - реляційна система управління базами даних. MySQL на даний час дуже популярна і використовується на багатьох сайтах [16].

Переваги MySQL:

- простота в роботі;
- багатий функціонал;
- велика кількість функцій забезпечують безпеку, які підтримується за замовчуванням;
- легко працює з великими обсягами даних;
- може значно збільшити продуктивність сервера.

Недоліки MySQL:

- відомі обмеження - за задумом в MySQL закладені деякі обмеження функціонала, які іноді необхідні в особливо вимогливих додатках;
- проблеми з надійністю - через деяких способів обробки даних MySQL (зв'язку, транзакції, аудити) іноді поступається іншим СУБД по надійності.
- повільна розробка - хоча MySQL технічно відкрите ПЗ, існують скарги на процес розробки. Варто зауважити, що існують інші досить успішні СУБД створені на базі MySQL, наприклад MariaDB.

PostgreSQL - об'єктно-реляційна система управління базами даних. PostgreSQL на відміну від MySQL не дуже популярна і складна в її вивченні, але вона має дуже великий функціонал [4].

Переваги PostgreSQL:

- безкоштовне ПЗ з відкритим вихідним кодом і яка відповідає стандарту SQL;
- незважаючи на величезну кількість вбудованих функцій, існує дуже багато доповнень, що дозволяють розробляти дані для цієї СУБД і управляти ними;
- має можливість розширення функціоналу за рахунок збереження своїх процедур;
- PostgreSQL це не тільки реляційна СУБД, але також і об'єктно-орієнтована.

Недоліки Postgresql:

- при простих операціях читання Postgresql може значно уповільнити сервер і бути повільніше своїх конкурентів, таких як MySQL;
- немає великої популярності;
- складно знайти хостинг з підтримкою цієї СУБД.

Для того щоб зрозуміти як правильно створювати запити нам необхідно спочатку розібратися з мовою програмування SQL, так як вона використовується в усіх СУБД які існують в сучасному світі.

SQL (Structure Query Language) – це мова програмування структурованих запитів, за допомогою яких можливо пошук потрібних даних, їх редагування та видалення з БД, а також створення нових даних [5].

Загальна структура запитів виглядає наступним чином:

- SELECT (вказуємо стовпці які ми хочемо вибрати якщо ми хочемо вибрати всі стовпці то необхідно написати *);
- FROM (назва таблиці з якої хочемо дістати стовпці);
- WHERE (вказуємо фільтри чи умови за допомогою яких буде вибирати тільки ті дані які нам потрібні);
- GROUP BY (вказуємо назву стовпця по якому ми будемо групувати вибрані данні);
- HAVING (фільтр чи умова на рівні вже згрупованих даних);
- ORDER BY (вказуємо назву стовпця по якому сортируємо данні).

3 РЕАЛІЗАЦІЯ ВЕБ ДОДАТКУ

3.1 Загальна структура сайту та його основний функціонал



Рисунок 3.1 – Структура сайту

Як зображено на рис. 3.1 наш сайт складається з наступних сторінок:

- сторінка авторизації та реєстрації – це сторінка до користувач може зареєструвати себе на нашому веб додатку або увійти уже під існуючим профілем;
- головна сторінка містить список всіх користувачів які зареєстровані в нашому веб додатку а також панель за допомогою якою можливо переходити між іншими сторінками;
- профіль – це сторінка на якій відображена загально доступна інформація така як: ім'я, вік, освіта, релігія та інше;
- змінна даних – це сторінка на якій користувач може редагувати свої дані;
- пошук користувачів – це сторінка на якій можна знайти людей за певними параметрами;
- чат – це сторінка де користувачі можуть спілкуватися між собою.

3.2 Розробка інтерфейсу веб-додатку

Відповідно до структури була розроблена сторінка реєстрації та авторизації (рис. 3.2) за допомогою якої нові користувачі можуть створити свій профіль, а старі користувачі можуть зайти під уже створеним профілем.

The screenshot shows the 'datingchile' website's registration and login interface. The header is blue with the logo and tagline 'Знайдіть свого ідеального партнера'. On the right, there's a login section with fields for 'Логін' (Login) and 'Пароль' (Password), a 'Забули пароль?' (Forgot password?) link, and a 'Увійти' (Login) button. The main area features a large background image of a smiling couple. Overlaid on this is a registration form with fields for 'Логін', 'Пароль', 'Почта' (Email), 'Дата народження' (Date of birth) with a date picker, 'Освіта' (Education), 'Діти' (Children), 'Регіон' (Region) with a dropdown, and 'Вулиця' (Street). A checkbox for 'Приймаю умови використання' (I accept the terms of use) is present. A 'Завершити' (Finish) button is at the bottom right. Below the form, there are three small thumbnail images.

Рисунок 3.2 – Сторінка реєстрації та авторизації

Після того як користувач пройшов реєстрацію та авторизувався його перенаправляє на головну сторінку (рис. 3.3).

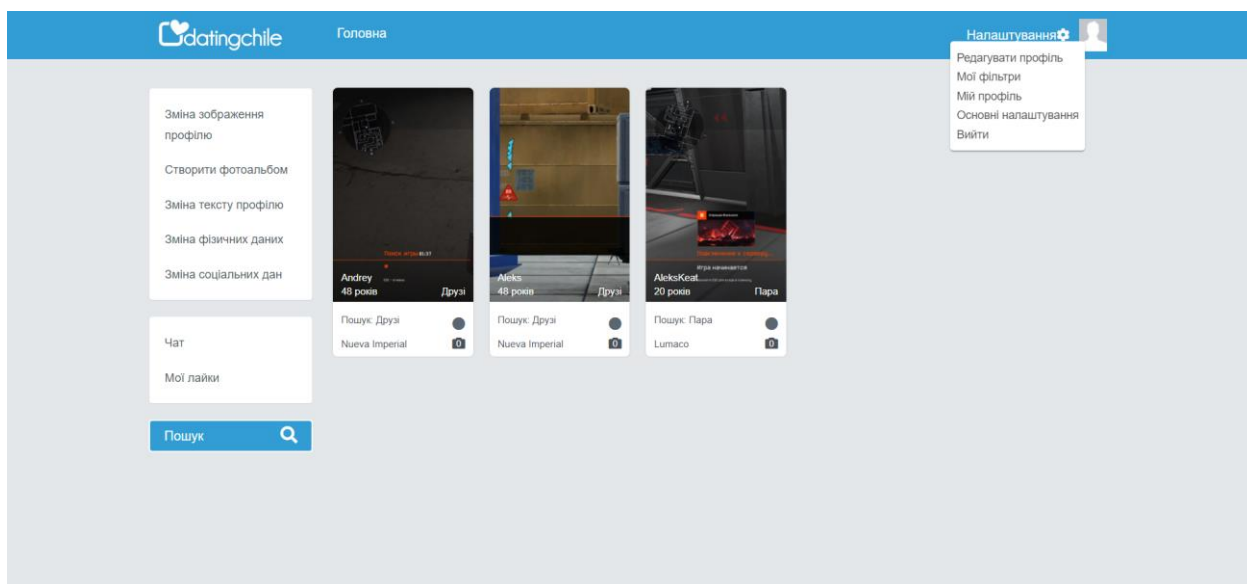


Рисунок 3.3 – Головна сторінка

За допомогою лівого меню ми може перейти на сторінку зміна даних (рис. 3.4).

Рисунок 3.4 – Сторінка зміни даних

Також за допомогою лівого меню ми можемо перейти на сторінку пошуку (рис. 3.5)

Рисунок 3.5 – Сторінка пошуку

Через меню налаштування можливо перейти на профіль (рис. 3.6)

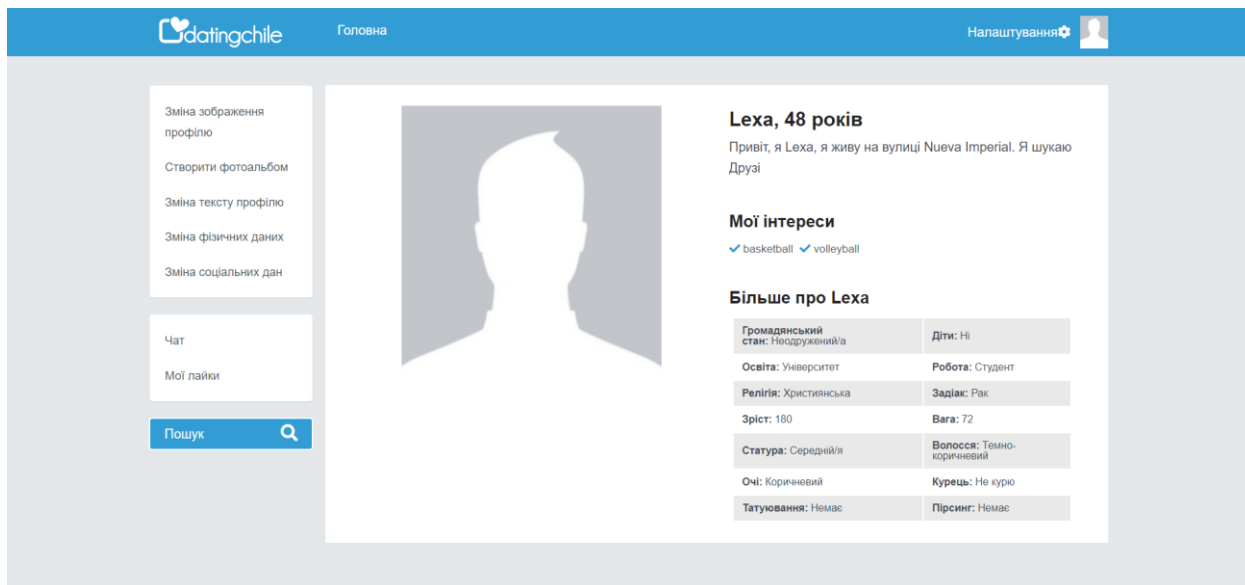


Рисунок 3.6 – Сторінка профілю

За допомогою лівого меню можливо відкрити сторінку чата (рис. 3.7) де користувач може побачити всі переписки які були та прочитати повідомлення які йому надійшли.

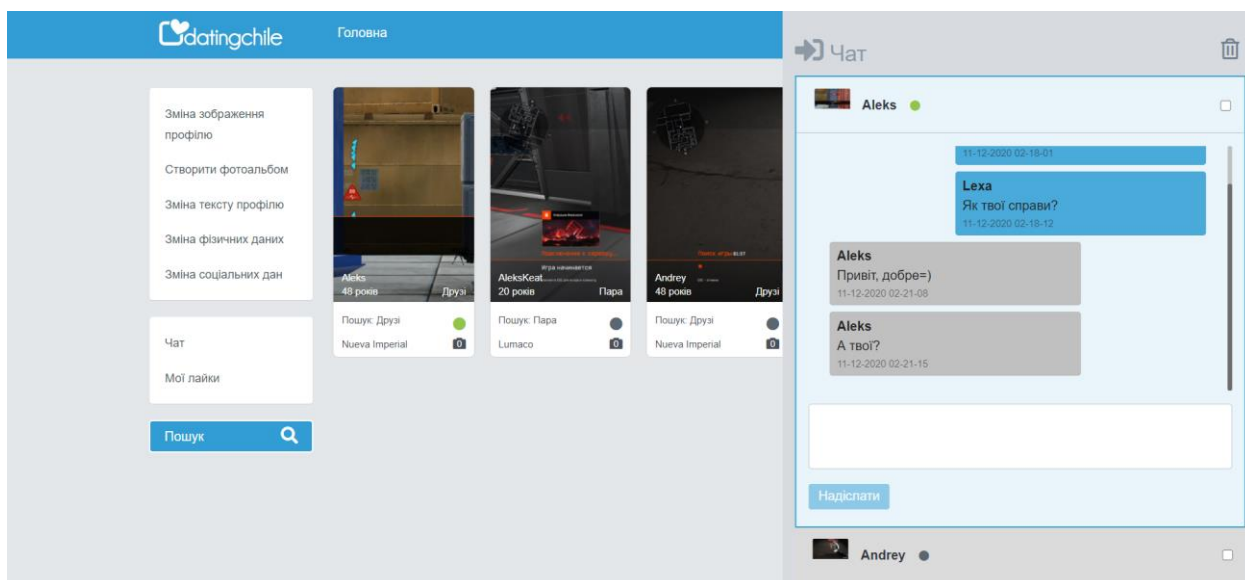


Рисунок 3.7 – Сторінка чату

Також якщо зайти під профілем адміністратора в лівому меню можливо буде перейти на сторінку редагування інтересів (рис. 3.8)

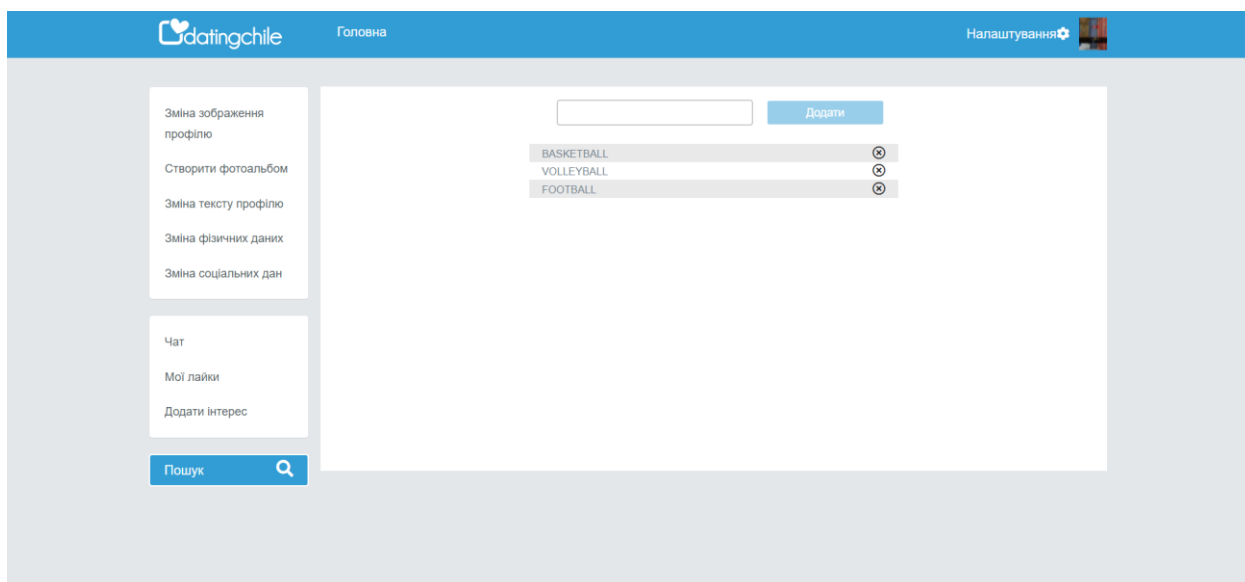


Рисунок 3.8 – Сторінка редагування інтересів

Також коли перейти на одно з користувачів буде добавлена адміністраторська панель (рис. 3.9) за допомогою якою адміністратор має змогу забанити користувача, редагувати його дані а також призначити модератором. Модератор також має таку панель, але без призначення модератора.

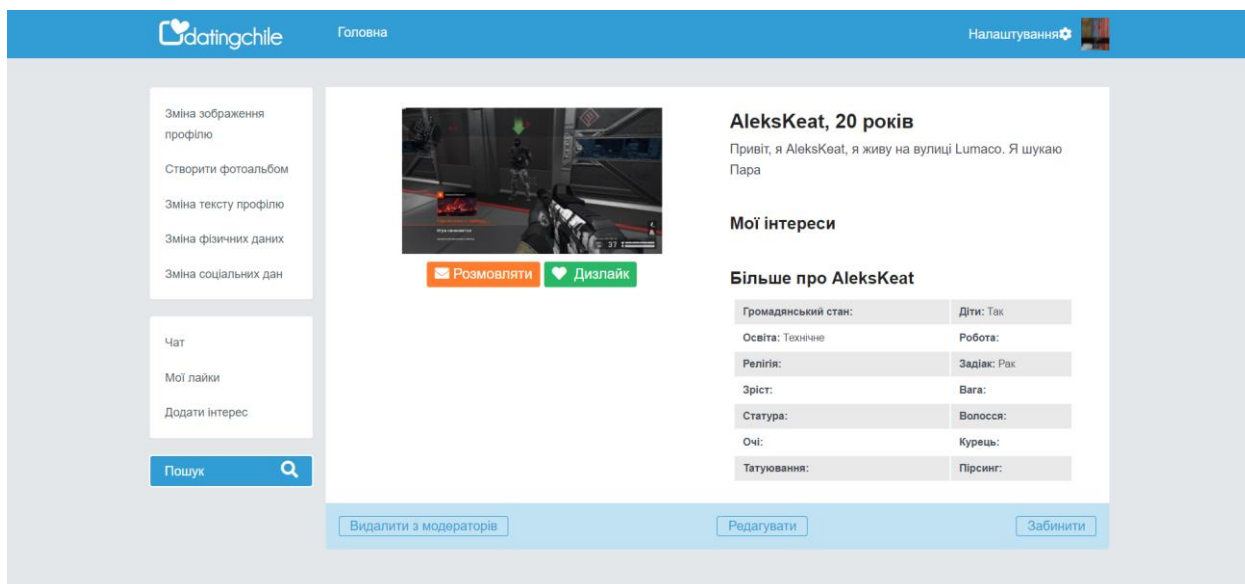
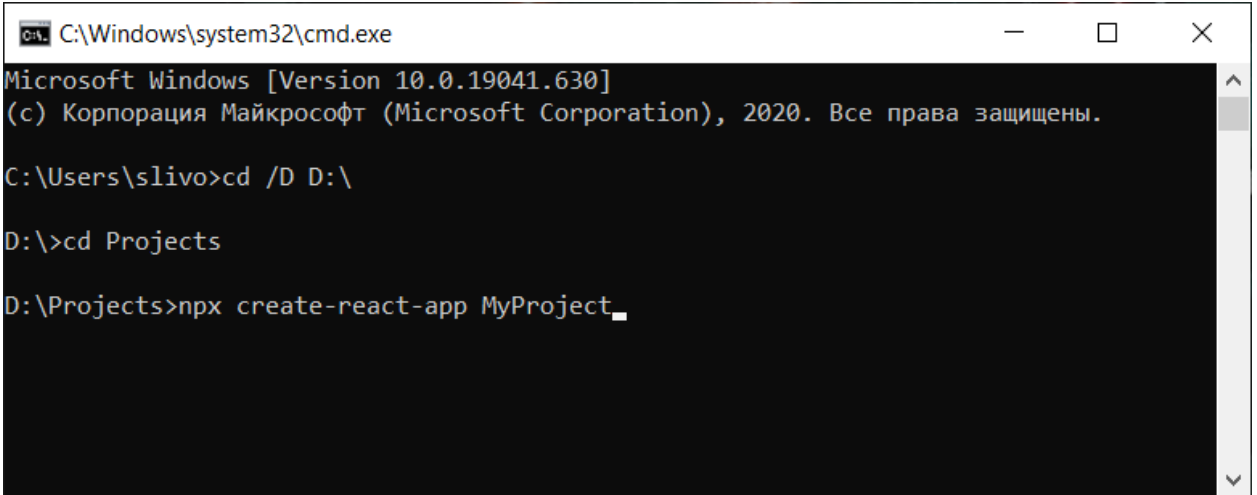


Рисунок 3.9 – Сторінка користувача з адміністративною панелью

3.3 Створення веб-додатку

Для того щоб почати розробку веб-додатка спочатку треба встановити редактор IntelliJ IDEA. Далі необхідно відкрити консоль через яку перейти до папки де будемо зберігати наш веб додаток і прописати наступну команду дивись рис. 3.10.



```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19041.630]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

C:\Users\slivo>cd /D D:\

D:\>cd Projects

D:\Projects>npx create-react-app MyProject_
  
```

Рисунок 3.10 – Команда для створення веб-додатку.

Далі необхідно встановлені залежності (бібліотеки) які вказані в файлі package.json за допомогою пакетного менеджера yarn. Після чого розділити веб-додаток на дві основні частини:

- сторінка реєстрації та авторизації користувача;
- основна сторінка – це сторінка де користувач може редагувати дані свого профіля, шукати та спілкуватися з іншими людьми.

Потім переходимо до папки src і створюємо наступні папки Main і Home. В папці Main ми будемо розробляти основну сторінку, а в папці Home – сторінку реєстрації та авторизації. Далі відкриваємо файл App.js видаляємо все що там знаходиться і вставляємо наступний код (лістинг 3.1).

Лістинг 3.1 – Код створення сторінки авторизації

```

import React, { useEffect } from "react";
import { Route, Switch, withRouter, Redirect } from "react-router-dom";
import { connect } from "react-redux";
import { ToastContainer } from "react-toastify";
import "./App.scss";
import "react-toastify/dist/ReactToastify.css";
  
```

```

import Home from "./Home";
import ResetPassword from "./ResetPassword";
import Main from "./Main";
import { checkAuth } from "./Redux/userReducer";
const App = ({ auth, checkAuth }) => {
  useEffect(checkAuth, []);
  const acceptRoute = auth ? ( <Route path="/profile" component={Main} /> ) : (<Route
path="/home" component={Home} />);
  const acceptLink = auth ? "/profile" : "/home";
  return (
    <div>
      <ToastContainer />
      <Switch>
        {acceptRoute}
        <Route path="/reset/:token" component={ResetPassword} />
        <Redirect to={acceptLink} />
      </Switch>
    </div>
  );
};
const mapStateToProps = ({ userData: { auth } }) => ({ auth });
export default connect(mapStateToProps, { checkAuth })(withRouter(App));

```

В файлі App.js ми імпортуємо компоненти Route і Redirect з бібліотеки react-router-dom. В компоненті Route нам необхідно вказати два параметри: path (маршрут), component (компонента). Створимо дві компоненти Route де маршрут `/profile` буде відображати компоненту Main, а `/home` - Home.

Далі необхідно підключити бібліотеку Redux за допомогою якою ми зможемо зберігати дані протягом сесії. Для цього спочатку нам необхідно створити папку Redux та файл store.js де вставляємо наступний код (лістинг 3.2).

Лістинг 3.2 – Код для створення змінної store.

```

import { combineReducers, createStore, applyMiddleware } from "redux";
import thunk from "redux-thunk";
import { userReducer } from "./userReducer";
const reducers = combineReducers({
  userData: userReducer,
});
let store = createStore(reducers, applyMiddleware(thunk));
export default store;

```

Store – це об’єкт який складається з Reducers в яких зберігається об’єкт з даними та функції з їх маніпуляцією.

Лістинг 3.3 – Код для створення Reducer.

```

import { get, post } from "../API/Requests";
import { toast } from "react-toastify";
import { getHash } from "../registrationReducer";
const SET_USER_DATA = "SET-USER-DATA";
const SET_FILL = "SET-FILL";
let initialState = {
  login: "",
  password: "",
};
export const userReducer = (state = initialState, action) => {
  switch (action.type) {
    case SET_USER_DATA:
      return {
        ...state,
        ...action.data,
      };
    case SET_FILL:
      return {
        ...state,
        [action.fill]: action.value,
      };
    default:
      return state;
  }
};
export const setUserData = (data) => ({ type: SET_USER_DATA, data });
export const setFill = (fill, value) => ({ type: SET_FILL, fill, value });
export const authorize = (data, history) => {
  return (dispatch) => {
    data.password = getHash(data.password)
    post("/login", data).then((result) => {
      if (!result.error) {
        dispatch(setUserData(result));
        history.push("/profile");
      } else {
        toast.error(result.message);
      }
    });
  };
};
export const getUserData = () => {
  return (dispatch) => {
    get("/me").then((result) => dispatch(setUserData(result)));
  };
};
export const checkAuth = () => {
  return (dispatch) => {
    get("/check-auth").then((result) => dispatch(setFill("auth", result.auth)));
  };
};

```

Потім відкриваємо файл `index.js` який знаходиться в папці `src` і обертаємо наш код в компоненту `Provide` яку імпортуємо з бібліотеки `'react-redux'` і передаємо в неї нашу зміну `store` (лістинг 3.4).

Лістинг 3.4 – Код підключення `Redux` до клієнтської частини.

```
import React from "react";
import ReactDOM from "react-dom";
import { BrowserRouter } from "react-router-dom";
import { Provider } from "react-redux";
import App from "./App";
import store from "./Redux/store";
ReactDOM.render(
  <BrowserRouter>
    <Provider store={store}>
      <App />
    </Provider>
  </BrowserRouter>,
  document.getElementById("datingchile")
);
```

Після того як ми створили основні сторінки і підключили `Redux` до нашого проекту нам необхідно створити серверну частину куди ми зможемо надсилати наші запити на отримання даних і проводити маніпуляції над ними. Для цього в кореневій папці необхідно створити папку `server` де буде зберігатися наступний код (лістинг 3.5).

Лістинг 3.5 – Код створення серверної частини.

```
const express = require("express");
const http = require("http");
const publicDir = "public";
const app = express();
const server = http.Server(app);
app.use(express.static(publicDir));
app.use(express.json());
server.listen(80, () => {
  console.log("Server is running: http://localhost")
});
```

Тепер веб-сайт має наступ архітектуру дивись рис. 3.11

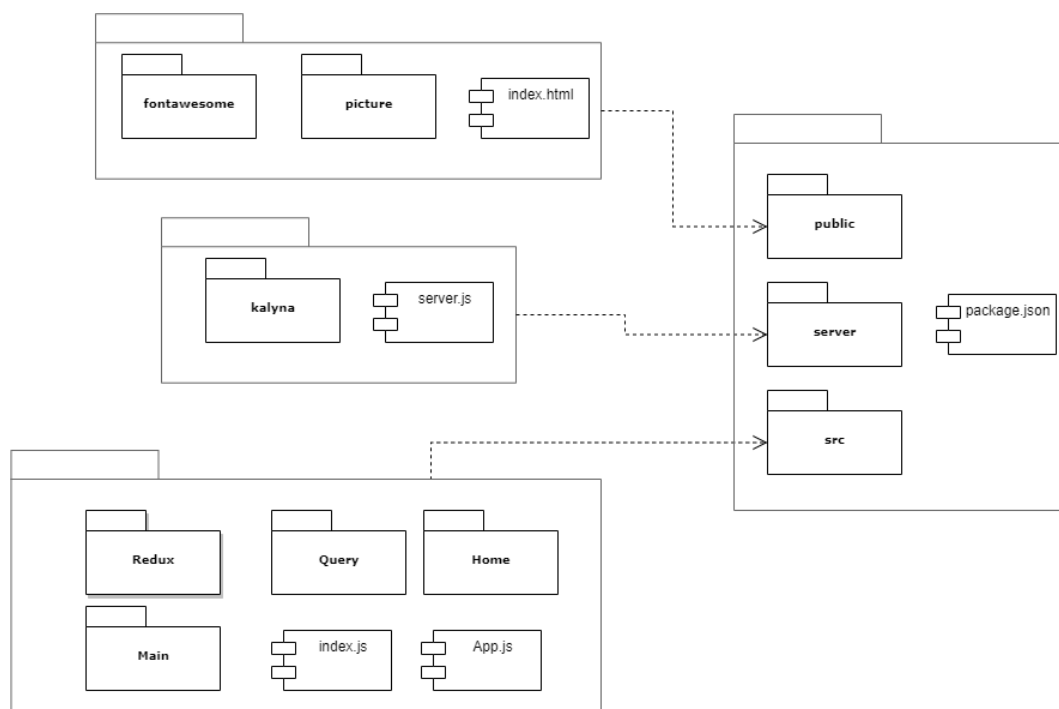


Рисунок 3.11 – Архітектура веб-сайту

Отже в результаті роботи було створено клієнтську частину та серверну частину, а також підключено бібліотеку Redux для збереження та маніпуляції отриманих даних під час сесії.

3.3.1 Написання серверної частини

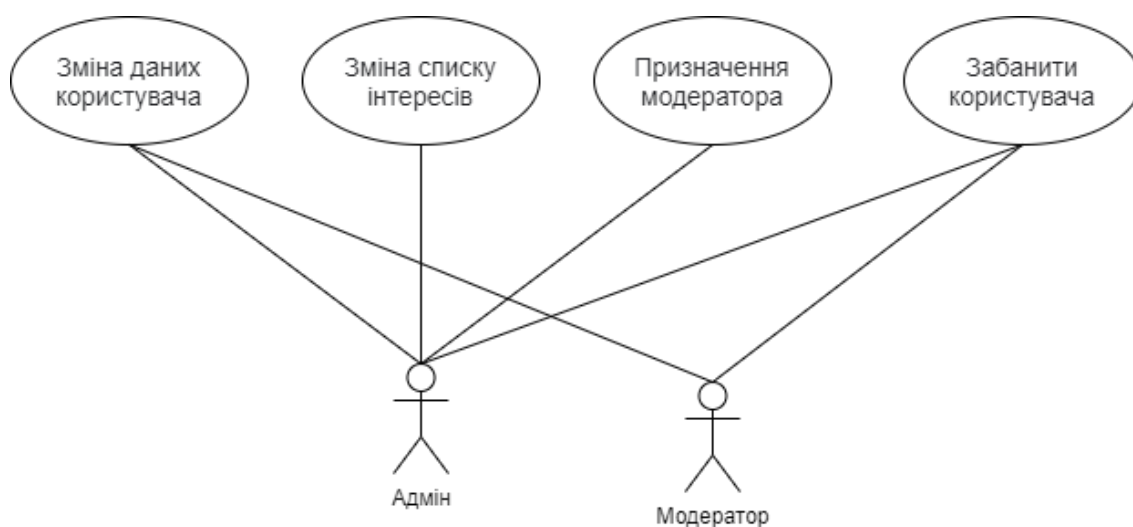


Рисунок 3.12 – Основні функції адміністратора та модератора

Дивлячись на рис. 3.12 можемо зробити висновок що у нас на сайті буде три основні ролі це адміністратор, модератор та користувач. Адміністратор має наступні права:

- змінна даних користувача;
- зміна списку інтересів;
- призначення модератора;
- забанити користувача.

Модератор має наступні права:

- змінна даних користувача;
- забанити користувача.

Користувач буде мати можливість:

- редагувати дані;
- переписуватися з іншими користувачами;
- користуватися пошуком.

Тепер коли ми розібрались які ролі у нас присутні і які права у них є, нам спочатку треба побудувати діаграму класів (рис. 3.13) по якій ми будемо будувати нашу базу даних, а також будемо писати функцію для розмежування прав щоб кожні ролі були доступні тільки свої права.

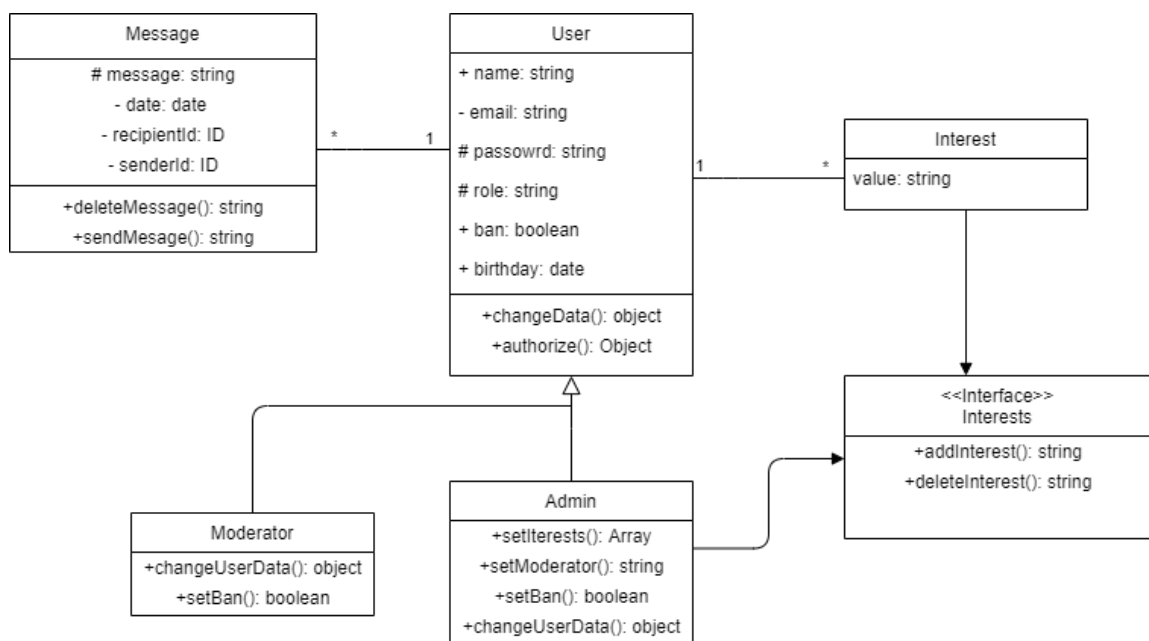


Рисунок 3.13 – Діаграма класів

Тепер нам необхідно створити в базі даних таблиці де ми будемо зберігати наші дані. Створимо запит який при старті нашого сервера буде генерувати цю таблицю якщо її не має в базі даних (лістинг 3.6).

Лістинг 3.6 – Створення таблиць в базі даних.

```
create table public.users
(id          serial not null
 constraint users_pk
 primary key,
 login       varchar not null,
 password    varchar not null,
 email       varchar not null,
 gender      varchar,
 birthday    varchar,
 education   varchar,
 children    varchar,
 region      varchar,
 district    varchar,
 type        varchar,
 category_age varchar,
 profile_photo varchar,
 age         integer,
 role        varchar,
 ban         boolean);
create table public.interests
(id serial not null
 constraint intereses_pk
 primary key,
 value varchar not null);
create table public.messages
(id serial not null
 constraint messages_pk
 primary key,
 sender_id integer not null,
 recipient_id integer not null,
 message varchar not null,
 date timestamp);
create table public.user_interests
(id serial not null,
 user_id integer not null,
 interest_id integer not null);
```

Далі створимо функцію яка буде перевіряти користувача на його роль і буде надавати дозвіл на отримання певних даних або забороняти (лістинг 3.7).

В цьому нам допоможе бібліотека passport.

Лістинг 3.7 – Функція для перевірки прав користувача.

```
const checkRole = (res, next, message, check) =>
```

```

    check ? next() : res.json({ error: true, message });
const moderatorAndAdmin = (adminCheck) => (req, res, next) => {
const isAdmin = req.user.role === "admin";
const isModerator = req.user.role === "moderator";
req.isAuthenticated()
  ? adminCheck
    ? checkRole(res, next, `Ви не маєте доступ!`, isAdmin)
      : checkRole(res, next, `Ви не маєте доступ!`, isModerator || isAdmin)
    : res.json({ auth: req.isAuthenticated() });
};

```

Як можемо бачити функція `moderatorAndAdmin` приймає параметр `adminCheck` якщо цей параметр має значення `true` то він перевіряє чи є користувач адміністратор якщо ж має значення `false` то перевіряє чи є він адміністратором або модератором. Якщо ж користувач не має даної ролі то йому приходить повідомлення що він не має доступ до цих даних.

Далі створимо `end-points` на які ми будемо робити запит з клієнтської частини. Давайте розглянемо один з прикладів і розберемо його (лістинг 3.8).

Лістинг 3.8 – Функція призначення модератора.

```

const setModerator = async (id) => {
  await User.update({ role: moderator, }, { where: { id } });
  return { error: false, message: "Пользователь был назначен модератором!" };
};
app.post("/set-moderator", moderatorAndAdmin(true), async (req, res) => {
  const result = await setModerator(req.body.id);
  await res.json(result);
});

```

Як можемо бачити перша функція `setModerator` дає можливість назначити користувача модератором для цього необхідно передати `id` того користувача якому ми хочемо змінити роль. Наступна функція `app.post` приймає декілька параметрів:

- перший параметр – це маршрут за яким можливо звертатись до цієї функції;
- другий параметр – це функція яка перевіряє чи є користувач адміністратором;
- третій параметр – це функція яка назначає користувача модератором та повертає відповідь що користувач був назначений модератором.

Але якщо функція яка перевіряє чи являється користувач адміністратором верне false то друга функція яка назначає користувача модератором не виконається.

3.3.2 Створення сторінки реєстрації та авторизації користувача

Спочатку розробимо сторінку з зручним і зрозумілим інтерфейсом де користувач зможе заповнити свої дані на сервер для подальшого збереження та зареєструватися (Додаток А).

Тепер напишемо функцію яка буде посилати дані на сервер щоб користувач зміг зареєструватися, а також функцію яка буде хешувати пароль для того щоб злоумисник не зміг його прочитати (лістинг 3.9).

Лістинг 3.9 – Функція хушевання паролю та зберігання користувача в базі даних.

```
export const getHash = text => {
  return crypto
    .createHmac("sha256", "This is secret key!")
    .update(text)
    .digest("hex");
};
export const registration = (data, toast) => {
  return (dispatch) => {
    data.password = getHash(data.password)
    post("/registration", data).then((result) => {
      if (result && !result.error) {
        toast("Registration completed successfully!");
        dispatch(showRegistration(false));
        for (let key in data) dispatch(fillRegistration(key, ""));
      } else toast("You couldn't register!");
    });
  };
};
```

Як може побачити функція getHash приймає текст та робить з нього хеш-текст для цього використовується бібліотека crypto звідки ми дістаємо функцію createHmac яка приймає два параметри:

- алгоритм хешування;
- секретний ключ.

А також дістаємо функцію update в яку передаємо текст який хочемо за хешувати.

Як вже було сказано вище ми використовуємо бібліотеку crypto для хешування. Вона не тільки дозволяє хешувати текст, а й дає можливість генерувати ключ за алгоритмом DiffieHellman, та використовувати функції шифрування та дешифрування (рис. 3.14).

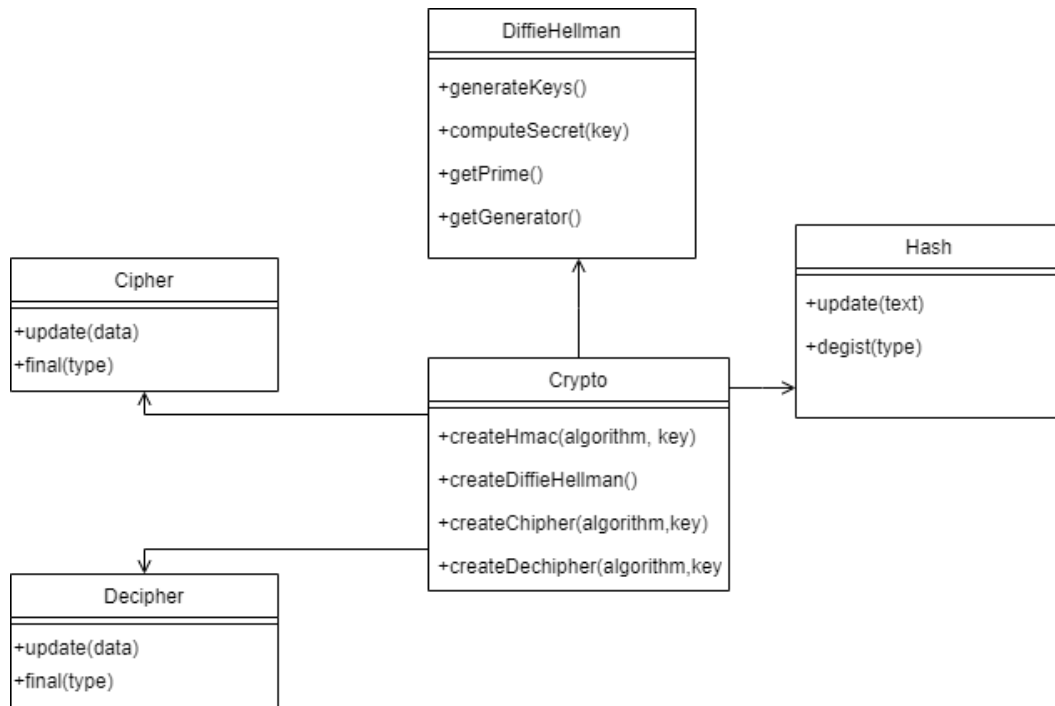


Рисунок 3.14 – Діаграма класів бібліотеки crypto.

3.3.3 Створення сторінки профіль.

На цій сторінці як вже було сказано буде відображатися загально доступна інформація про користувача. Для цього нам спочатку необхідно написати функцію на отримання даних певного користувача (лістинг 3.10).

Лістинг 3.10 – Отримання публічних даних користувача.

```

export const getUser = (id) => {
  return (dispatch) => {
    get(`/user/${id}`).then((result) => {
      dispatch(setData(result));
    });
  };
};

```

Як можемо побачити з даної функції вона приймає `id` по якому буде шукати користувача в нашій базі даних, а потім повертати отримані дані і зберігати в `store`.

Тепер коли ми отримали дані то ми можемо згенерувати сторінку де будемо їх відображати (Додаток Б).

3.3.4 Створення сторінки пошуку

Спочатку нам необхідно створити сторінку пошуку, на якій ми будемо генерувати об'єкт даних, в якому ми вкажемо параметри пошуку інших користувачів (Додаток В).

Далі створимо запит, який буде відправляти наш об'єкт на сервер, за допомогою цього запиту він поверне нам тих користувачів, що будуть відповідати встановленим параметрам (лістинг 3.11).

Лістинг 3.11 – Функція пошуку користувачів в базі даних.

```
export const findPeople = (data, history) => {
  return (dispatch) => {
    post("/user/find", data).then((result) => {
      dispatch(setFoundPeople(result));
    });
    history.push("/profile/filtered");
  };
};
```

3.3.5 Створення сторінки редагування даних

Для того щоб, користувач мав можливість редагувати свої дані, нам необхідно створити сторінку з формою для їх редагування. Ця сторінка буде розділена на декілька частин:

- зовнішній вигляд (зріст, вага, статура, колір волосся, колір глас, татуювання, пірсинг);
- персональна інформація (регіон, вулиця, тип пошуку, назва профілю, короткий опис);
- соціальна інформація (громадський стан, діти, освіта робота, релігія, куріння);
- список інтересів.

Для створення сторінки редагування необхідно написати наступний код (Додаток Г).

Тепер коли ми створили сторінку нам необхідно сформовані данні надіслати на сервер (лістинг 3.12).

Лістинг 3.12 – Функція для оновлення даних користувача.

```
const handleUpdate = () => {
  const newData = {
    userData: data,
    interests: interests.userInterests,
  };
  setExitingData(data);
  setExitingInterests(interests.userInterests);
  post(`/user/${match.params.id}/update-profile`, newData).then(
    ({ error, message }) => error ? toast.error(message) : toast(message));
};
```

3.3.6 Створення чату

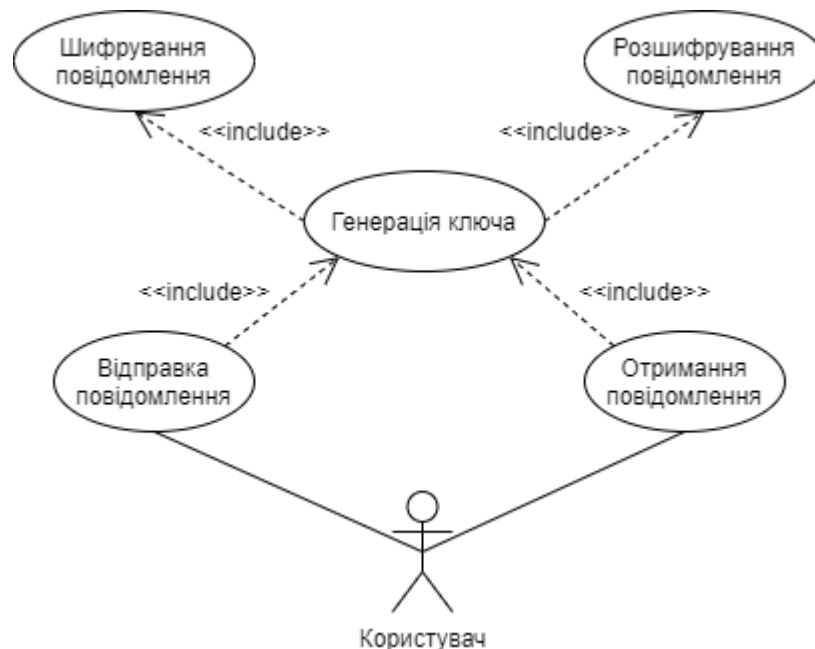


Рисунок 3.15 – Схема відправки та отримання повідомлення

Дивлячись на рис. 3.15 можемо зробити висновок, що спочатку необхідно створити сторінку, для відображення повідомлень та їх написання, а також створити бібліотеку, яка буде шифрувати повідомлення, для того щоб

зловмисник не зміг перехопити та редагувати вміст повідомлення, або отримати інформацію яку би ви не хотіли розголошувати.

Отже необхідно спочатку розробити сторінку де користувач зможе відправляти та отримувати повідомлення (Додаток Д).

Далі необхідно створити запит який буде надсилати повідомлення іншому користувачу та зберігати його на сервері (лістинг 3.13).

Лістинг 3.13 – Функція відправки повідомлення.

```
export const send = (message, addMessage) => {
  chat.emit("send message", {
    ...message,
    date: moment().format("DD-MM-YYYY HH:mm:ss"),
  });
  addMessage(message.recipientId, { ...message, date: moment() });
};
```

3.4 Розробка бібліотеки шифрування з використанням алгоритма Калина.

Калина - блочний симетричний шифр описаний у національному стандарті України ДСТУ 7624:2014 «Інформаційні технології. Криптографічний захист інформації. Алгоритм симетричного блокового перетворення»

Таблиця 3.1 – Числові параметри алгоритма Калина.

Довжина блоку, біт	Довжина ключа, біт	Кількість циклових ключів	Кількість стовпців
128	128	10	2
	256	14	
256	256	14	4
	512	18	
512	512	18	8

Алгоритм шифрування:

- додавання циклового ключа за модулем 2^{64} ;
- шар нелінійного бієктивного відображення (SubBytes);

Функція π' виконує заміну кожного елементу $g_{j,i}$, матриці внутрішнього стану $GF(2^8)$, на $\pi_{i \bmod 4}(g_{j,i})$.

- перестановка елементів (ShiftRow);

Функція виконує циклічний зсув вправо. Кількість елементів зсуву залежить від номеру рядку $i \in \{0, 1, 2, 3, 4, 5, 6, 7\}$, розміру блоку $l \in \{128, 256, 512, \dots\}$, та обчислюється за формулою:

$$\delta = \frac{i \cdot l}{512}$$

- лінійне перетворення (MixColumns);

При обчисленні результату функції ψ кожен елемент $g_{j,i}$, матриці внутрішнього стану $G=(g_{j,i})$ розглядається як елемент скінченного поля $GF(2^8)$, що утворене незвідним поліномом $v(x) = x^8 + x^4 + x^3 + x^2 + 1$, або 0x11d у шістнадцятковому поданні.

Кожен елемент результуючої матриці стану $W = w_{j,i}$ отримується як результат множення векторів довжини 8 над скінченним полем $GF(2^8)$ за формулою:

$$w_{j,i} = (v \ggg i) \otimes G_j$$

де $v = (0x01, 0x01, 0x05, 0x01, 0x08, 0x06, 0x07, 0x04)$ – вектор, що утворює циркулянтну матрицю МДР-коду і складається з послідовності байтових констант у шістнадцятковому поданні, які інтерпретуються як елементи поля $GF(2^8)$, при цьому циклічний зсув виконується відносно елементів вектора над скінченним полем;

G_j – j -й стовпець матриці стану $G=(g_{j,i})$.

- функція додавання циклового ключа за модулем 2.

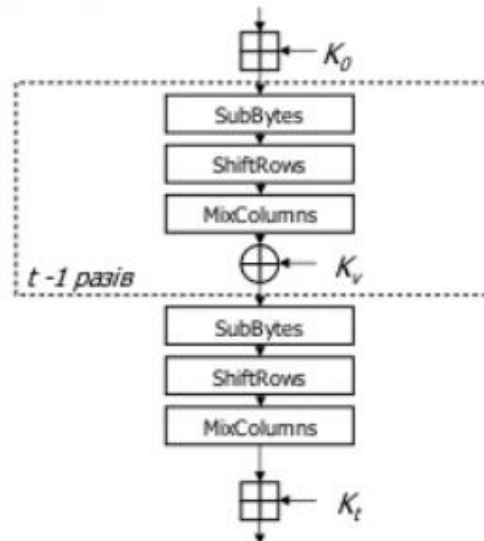


Рисунок 3.16 – Алгоритм шифрування[6]

Алгоритм розшифрування:

- віднімання циклового ключа за модулем 2^{64} ;
- шар оберненого нелінійного бієктивного відображення;
- обернена перестановка елементів;
- обернене лінійне перетворення;

При обчисленні результату функції ψ кожен елемент $g_{j,i}$ матриці внутрішнього стану $G=(g_{j,i})$ розглядається як елемент скінченного поля $GF(2^8)$, що утворене незвідним поліномом $v(x) = x^8 + x^4 + x^3 + x^2 + 1$, або 0x11d у шістнадцятковому поданні.

Кожен елемент результуючої матриці стану $W = (w_{j,i})$ отримується як результат множення векторів довжини 8 над скінченним полем $GF(2^8)$ за формулою:

$$w_{j,i} = (v \ggg i) \otimes G_j$$

де $v = (0xAD, 0x95, 0x76, 0xA8, 0x2F, 0x49, 0xD7, 0xCA)$ – вектор, що утворює циркулянтну матрицю МДР-коду і складається з послідовності байтових констант у шістнадцятковому поданні, які інтерпретуються як елементи поля $GF(2^8)$, при цьому циклічний зсув виконується відносно елементів вектора над скінченним полем;

G_j – j –й стовпець матриці стану $G=(g_{j,i})$.

- функція додавання циклового ключа за модулем 2.

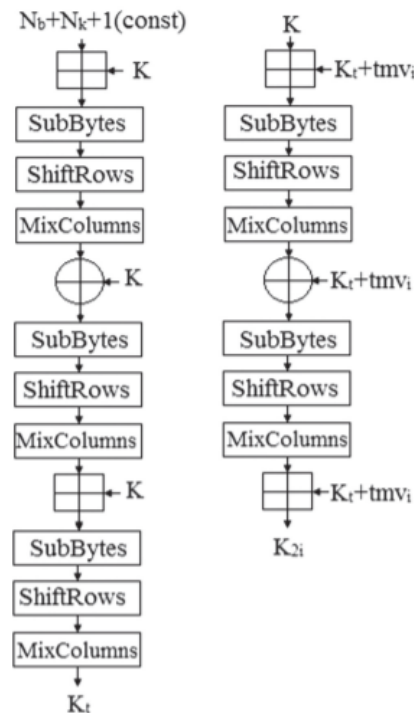


Рисунок 3.17 – Схема отримання циклового ключа[6]

Про аналізувавши алгоритми шифрування та дешифрування були написані наступні функції для кожного з пунктів алгоритму.

- SubBytes (лістинг 3.14);

Лістинг 3.14 – Функція SubBytes.

```
const subBytes = (array, isEncrypt) => {
  let numberSBox = 0
  return _.map(array, (item) => {
    if(isEncrypt) {
      if (numberSBox === 0) return encryptSBoxP0[item]
      if (numberSBox === 1) return encryptSBoxP1[item]
      if (numberSBox === 2) return encryptSBoxP2[item]
      if (numberSBox === 3) return encryptSBoxP3[item]
    } else {
      if (numberSBox === 0) return decryptSBoxP0[item]
      if (numberSBox === 1) return decryptSBoxP1[item]
      if (numberSBox === 2) return decryptSBoxP2[item]
      if (numberSBox === 3) return decryptSBoxP3[item]
    }
    numberSBox = numberSBox === 4 ? 0 : numberSBox++
  })
}
```

- ShiftRows (лістинг 3.15);

Лістинг 3.15 – Функція ShiftRows.

```
const shiftRow = (array) => {
  const preparedArray = separateArray(array, 8)
  const rows = getArrayRowsOrColumns(preparedArray)
  const reversedRows = _.map(rows, (row, index) => {
    index > 3 ? row.reverse() : row
  })
  const columns = getArrayRowsOrColumns(reversedRows)
  return _.concat(...columns)
}
```

- MixColumns (лістинг 3.16).

Лістинг 3.16 – Функція MixColumns

```
const mixColumn = (sentence, matrixState) =>
  _.map(sentence, (array) => _.map(array, (char, idx) => {
    const newArray = _.map(array, (item, index) => {
      return multiplication(item, matrixState[idx][index])
    });
    return newArray.reduce((prev, item) => item ^ prev, 0);
  })))
```

4 ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ ТЕХНІЧНОЇ ЧАСТИНИ

4.1 Аналіз маркетингової частини проекту «Розробка веб месенджеру для захищеного обміну повідомленнями»

В нашому сучасному світі дуже багато людей користуються веб сайтами, веб месенджерами та іншими веб додатками. Тому ця сфера має дуже великий попит у нас в Україні та загалом в цілому світі. За допомогою веб месенджерів люди можуть спілкуватися один з іншим на відстані, ділитись новинами та інше. Під час спілкування люди можуть надсилати конфіденційну інформацію, щоб зловмисники не змогли її перехопити та використати у своїх цілях ми розробляємо веб месенджер для захищеного обміну повідомленнями.

В таблиця 4.1 ми описали зміст нашої ідеї, її напрямки застосування, а також вигоди для споживачів(користувачів).

Таблиця 4.1 - Опис ідеї .

Зміст ідеї	Напрямки застосування	Вигоди для споживачів (користувачів)
Розробка веб месенджеру для захищеного обміну повідомленнями	1. Пошук клієнтів та партнерів	Спілкування на відстані
	2. Розповсюдження реклами	Захист персональних даних від зловмисника
	3. Спілкування між користувачами	Змога знайти людину за допомогою пошуку за певними критеріями такі як стать, вік, зріст, регіон проживання та інше.

В таблиці 4.2 проаналізували ринкові можливості запуску реалізації ідеї.

Таблиця 4.2 - Попередня характеристика потенційного ринку

№	Показники стану ринку (найменування)	Характеристика
1	Головні конкуренти	Компанію які створюють веб додатки та використовуючи більш новітню технологію розробки та захисту веб сайту.
2	Динаміка ринку (якісна оцінка)	Зростає
3	Наявність обмежень для входу (вказати характер обмежень)	Існує декілька великих гравців які диктують умови в даному сегменті, багато невеликих гравців
4	Специфічні вимоги до стандартизації та сертифікації	Нема

Як можна побачити з таблиці 4.2, динаміку ринку зростає, так як дуже багато людей використовують веб месенджери і не можуть без них обійтися. Головними конкурентами являються ті компанії які також розробляють веб месенджери і вже зарекомендували себе в цій сфері. Але ми розробляємо веб месенджер який використовує новий алгоритм «Калина» для захищеного обміну повідомленнями якого ще немає в сучасному ринку. Тому для його реалізації необхідно значні інвестиції.

Далі в таблиці 4.3 ми визначили потенційні групи клієнтів, їх характеристики, та сформулювали орієнтовний перелік вимог до товару (робіт чи послуг) для кожної групи.

Таблиця 4.3 – Попередня характеристика потенційних клієнтів

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів (користувачів)
	Спілкування на відстані	Компаній	Пошук клієнтів.	Надійність.
	Пошук інших клієнтів	Користувачі	Пошук клієнтів сайту з однаковими інтересами для подальшого спілкування з ним	Обґрунтована ціна.
	Реклама	Блогери	Пошук партнерів для співпраці з компанією	Якість.
			Пошук робітників для компанії	Зручність
			Розповсюдження реклами	

Із таблиці 4.3 можна зробити висновок, що основною цільовою аудиторією є споживачі(користувачі), блогери, та інші компанії для яких основні вимоги є надійність, якість, зручність, а також - обґрунтована ціна.

В таблиці 4.4 визначили слабкі та сильні сторони використання алгоритму «Калина» для шифрування повідомлень, для цього був здійснений SWOT-аналіз середовища реалізації інноваційного проекту.

Таблиця 4.4 - SWOT-аналіз

Сильні сторони:	Слабкі сторони:
- Надійність та технічна підтримка - Зручне та просте використання - Потужне сучасне обладнання	- Висока конкуренція - Необхідне значне інвестування для старту - Нехватка кваліфікаційного персоналу
Можливості:	Загрози:
- Вихід на міжнародний ринок - Прогресування - Розвиток - Універсальність	- Зменшення ціни на товар у конкурентів - Важкість виходу на міжнародний ринок - Зменшення попиту

Далі в таблиці 4.5 ми розробили ринкову стратегію першим кроком якої передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів

Таблиця 4.5 - Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Компанії	Готові	Високий попит	Досить велика конкуренція	Легко
2	Користувачі	Готові	Високий попит	Досить велика конкуренція	Легко
3	Блогери	Готові	Високий попит	Досить велика конкуренція	Легко
Які цільові групи обрано: Компанії, користувачі, блогери					

Далі ми ідентифікували стейкхолдери, здійснили їх пріоритизацію та побудували карту стейкхолдерів, яка дозволить візуалізувати картину взаємозв'язків стейкхолдерів. Вона зображена на рисунку 4.1.

Створення карти має відбуватись у 4 етапи:

- ідентифікація – окреслення актуальних груп стейкхолдерів;
- аналіз – дослідження впливу та інтересів стейкхолдерів;
- візуалізація – візуальна ілюстрація взаємодії зі стейкхолдерами;
- пріоритезація – вибір ключових стейкхолдерів, які найкраще відповідали б істотним питанням, визначеним на попередньому етапі.

З метою відображення взаємозв'язків стейкхолдерів на карті має бути виокремлено три концентричних області, в яких розміщено всі зацікавлені особи інноваційного проекту за можливостями впливу на них ініціатора проекту.

Область внутрішніх стейкхолдерів – це область повноважень/відповідальності ініціатора. Внутрішні зацікавлені сторони знаходяться в прямій підлеглості ініціатора, що дозволяє використовувати досить прості методи адміністрування проекту.

Самі ж внутрішні зацікавлені сторони не являються підлеглими, проте мають тісні ділові стосунки і потребують пошук взаємовигідних рішень та компромісів. На периферії ж знаходяться зовнішні зацікавлені сторони. Це область стейкхолдерів опосередкованого впливу на успішність реалізації інноваційного проекту.



Рисунок 4.1 - Карта стейкхолдерів

4.2 Визначення витрат на розробку веб месенджера для захищеного обміну повідомленнями

Спочатку ми розраховували заробітну плату працівників на основі погодинної тарифної ставки виконуючих робіт, додаткову заробітну плату у вигляді премії 15% на 1 місяць, які необхідні на розробку веб месенджера для захищеного обміну повідомленнями. Єдиний соціальний внесок дорівнює 22% від основної та додаткової заробітної плати. Склад, чисельність та фонд заробітної плати виробничих працівників наведено у таблиці 5.6, а для адмінперсоналу у таблиці 5.7.

Таблиця 4.6 - Склад, чисельність та фонд заробітної плати виробничих працівників

Категорії працівників	Наявна чисельність, осіб		Тарифна ставка за розрядом виконуваних робіт, грн / годину	Ефективний фонд робочого часу, годин	Тарифний заробіток, грн.	Преміальний відсоток до тарифного	Розмір премії, грн.	Річний фонд заробітної плати, грн.	ЄСВ, грн.
	за зміну	на добу							
1	2	3	4	5	6	7	8	9	10
Виробничі працівники, в тому числі:	4	4	93.75	8	15 000	15%	2250	207 000	3 795
1.Основні працівники									
2.Допоміжні працівники	1	1	75	8	12 000	15%	1800	165 600	3 036
3.Черговий та ремонтний персонал	1	1	62.5	8	10 000	15%	1500	138 000	2 530
Разом виробничих працівників	6	6	X	X	X	X	5550	510 000	9 361

Таблиця 4.7 - Склад, чисельність та фонд заробітної плати адмінперсоналу

Посада	Кількість осіб	Посадовий оклад, грн	Преміальний відсоток до окладу, %	Сума премії грн	Місячна заробітна плата, грн	Річний фонд оплати праці, грн	ЄСВ, грн
1	2	3	4	5	6	7	8
Директор підприємства	1	9 480	50%	4 740	20 000	410 640	7 528,4
Заступник директора	1	8 534	40%	3 413	18 000	359 364	6 588,34

Продовження таблиці 4.7

Разом управлінсько го персоналу	2	18014	X	8 153	38 000	770 004	15 657, 74
--	---	-------	---	-------	--------	---------	---------------

В таблиці 4.8 ми визначили матеріальні витрати та вартість спожитих послуг, а також розрахували можливі транспортно-заготівельні витрат як 5% від вартості матеріалів.

Таблиця 4.8 – Розрахунок матеріальних витрат

Матеріальні витрати	Норматив у розрахунку на один. продук. (послуг)	Виробнича програма	Обсяг сировини	Ціна	Сума грн
1	2	3	4	5	6
Принтер Canon I-SENSYS MF112	2	-	2	7 000	14 000
Картридж для принтера	2	-	2	350	700
Ручка	8	-	8	10	80
Упаковка паперу формату А4 для принтера	4	-	4	86	344
USB Flash накопичувач об'ємом 32Gb	2	-	2	427	854
Транспортно-заготівельні витрати 5%	-	-	-	-	798,9
Разом	-	-	-	-	16 776,9

Далі ми розраховували вартість спожитих послуг таких як: електропостачання, водопостачання та теплопостачання. Для приміщення площею 70м² які були внесені до таблиці 4.9.

Таблиця 4.9 – Розрахунок вартості спожитих послуг

Вид послуг	Норматив у розрахунку на один. продук. (послуг)	Виробнича програма	Обсяг послуг	Тарифи	Сума грн
1	2	3	4	5	6
Електропостачання	T1	100%	185 кВт	2,68 грн./кВт	495,8
Водопостачання	B1	100%	34м ³	5,676 грн./м ³	192,98
Теплопостачання	TE	100%	70м ²	39,21 грн/м ²	2 744,7
Разом					3 433,48

В таблиці 4.10 ми визначили річну амортизацію відрахування з урахуванням первісної вартості основних засобів та нормативного терміну експлуатації, а також визначили витрати на поточний ремонт у розмірі 2% від балансової вартості основних засобів.

Балансова вартість основного засобу дорівнює різниці між його первісною вартістю об'єкта та сумою накопиченої амортизації та накопичених збитків від зменшення його корисності.

Балансова вартість досягає нуля, коли первісна вартість об'єкта дорівнює сумі накопиченої амортизації та збитків від зменшення його корисності.

Слід розуміти, що досягнення нульової балансової вартості об'єктом основних засобів не є причиною для їх списання.

Балансова вартість: $64\,500 - 12\,900 = 51\,600$ грн.

Витрати на поточний ремонт $= 51\,600 \times 0,02 = 1\,032$ грн.

Таблиця 4.10 – Розрахунок амортизації

Група основних засобів	Нормативний термін експлуатації	Первісна вартість ОЗ на 01.01	Надійшло ОЗ		Вибуло ОЗ		Річні амортизаційні відрахування грн
			дата	пер. вар.	дата	пер. вар.	
1	2	3	4	5	6	7	8
Сервер	5	41 500	26.10.2020	41 500	-	-	8 300
Комп'ютер	5	23 000	26.10.2020	23 000	-	-	4 600
Разом		64 500		64 500		-	12 900

Брутто-інвестиції або реінвестиції - складаються з вільних грошових коштів (прибутку, амортизації необоротних активів та ін). За своєю економічною природою вони більше тяжіють до активно-пасивним інвестицій, так як спрямовані на придбання нового обладнання, ресурсозберігаючих технологій, створення власними силами нових основних засобів (фондів), заміну застарілого обладнання, модернізацію, а також диверсифікацію капіталу.

Далі в таблиці 4.11 ми визначили собівартість на розробку веб мессенджеру для захищеного обміну повідомленнями.

Таблиця 4.11 – Кошторис витрат

Калькуляційні статті	Витрати	
	у розрахунку на новий сайт, грн.	у розрахунку на базовий варіант, грн.
Сировина та матеріали	8 227,15	16 776,9
Електропостачання	174,2	495,8

Продовження таблиці 4.11

Водопостачання	38,6	192,98
Теплопостачання	1 176,3	2 744,7
Разом	9 616,25	20 210,38
Заробітна плата основних виробничих працівників	17 250	94 300
ЄСВ 22%	3 795	20 746
Амортизація	4 600	12 900
Витрати на утримання та експлуатацію основних засобів поточний ремонт (беремо зарплату основних виробників)	368	1 032
Загальновиробничі витрати	21 767,2	147 537
Виробнича собівартість	35 983	180 647
Адміністративні витрати	-	38 000
Повна собівартість	35 983	218647,38

Протягом місяця підприємство понесло такі постійні загальновиробничі витрати:

- оплата праці загальновиробничого персоналу та нарахований єдиний соціальний внесок — 94 300 грн + 20 746 грн = 115 046грн;
- оренда виробничих приміщень — 25 000 грн;
- оплата охоронних послуг сторонньої організації — 6 000 грн.

Разом постійних загальновиробничих витрат:

$$115046 \text{ грн} + 25\,000 \text{ грн} + 6\,000 \text{ грн} = 146\,046 \text{ грн.}$$

За цей період змінні загальновиробничі витрати становлять:

- поточний ремонт виробничого обладнання — 1 032 грн;
- оплата вартості електроенергії — 495,8 грн.

Разом змінних загальновиробничих витрат:

$$1\,032 \text{ грн} + 495,8 \text{ грн} = 1\,527 \text{ грн.}$$

Усього загальновиробничих витрат:

$$146\,046 \text{ грн} + 1\,527 \text{ грн} = 147\,537 \text{ грн.}$$

Як можемо бачити що собівартість нового сайту який розробляє тільки один програміст необхідно всього 35 983 грн, а базова собівартість на розробку сайту де працюють декілька програмістів необхідно 218 647,38 грн.

$$\text{Економія} = 218\,647,38 \text{ грн} - 35\,983 \text{ грн} = 182\,664,38 \text{ грн.}$$

З даного розрахунку можна зробити висновок, що коли працює тільки один програміст то економія буде складати 182 664,38 грн. Але коли працюють декілька робота буде просуватися на багато швидше та якісніше ніж коли один.

В таблиці 5.12 визначили очікуваний дохід та фінансовий результат при ціні за один сайт у розмірі $2\,500\$ = 2\,500 * 28.13 = 70\,325 \text{ грн}$. Виходячи з цього можна розрахувати ціну за 4 сайта які будуть розробляти декілька програмістів.

$$\text{Ціна} = 4 * 70\,325 \text{ грн} = 281\,300 \text{ грн}$$

$$\begin{aligned} \text{Фінансовий результат} &= \text{Дохід} - \text{Витрати} = 281\,300 \text{ грн} - 218\,647,38 \text{ грн} \\ &= 62\,652,62 \text{ грн} \end{aligned}$$

Таблиця 4.12 - Розрахунок фінансового результату

№ періоду	Обсяги виробництва (послуг)	Обсяги реалізації	Залишок нереалізованої продукції	Ціна	Дохід	Витрати	Фінансовий результат
1	4	4	0	225 040	225 040	218 647,38	62 652,62

Виконавши дану роботу можемо винести наступні ризики:

- дуже велика конкуренція, але даний продукт має великий попит в сучасному ринку. Щоб уникнути даного ризику нам необхідно зробити ціни трохи нижче ніж у конкурентів, але тільки тоді коли ми все одно будемо мати

дохід, якщо ж ні то необхідно себе дуже гарно зарекомендувати, тобто зробити продукт набагато якісніше ніж у конкурентів;

- необхідно великий вклад на початку, для цього краще всього найти інвесторів і показати їм що ваша ідея принесе дохід;

- дуже тяжко знайти кваліфікаційний персонал, для цього по перше запропонувати більш вигідну зарплатню ніж у інших компаніях, по друге запропонувати вигід соціальний пакет, або по третє найняти новачків і обучити їх.

Виходячи з виконаної роботи ми можемо зробити наступний висновок те що навіть при великих витрати ми за місяць можемо отримати значний дохід. За нашими рахунками при витратах в 218 647,38 грн зробивши всього 4 сайта за 1 місяць ми отримаємо дохід в розмірі 62 652,62 грн.

5 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ

5.1 Аналіз потенційних небезпек

Тема дипломної роботи “Розробка веб месенджера для захищеного обміну повідомленнями”, тому розглянемо робоче місце програміста. Робота якого буде проходити в окремому офісному приміщенні. Спочатку нам необхідно провести аналіз потенційних небезпек під час роботи в приміщенні.

Основні потенційні небезпеки при проведенні даної роботи.

а) Загроза ураження електричним струмом, внаслідок недодержання правил електробезпеки або виходу з строю електроприладів.

б) Негативний вплив електромагнітних полів, які створюються офісним обладнанням, на організм працюючого.

в) Порушення функцій кістково-м'язового апарату внаслідок довготривалих статичних навантажень при роботі з ПК.

г) Нервово-психічні перевантаження внаслідок постійного контакту з колегами по роботі, клієнтами, керівництвом при вирішенні робочих питань, які можуть мати конфліктний характер і призвести до емоціонального дискомфорту, морального роздратування, емоційної нестабільності та захворювань нервової системи.

д) Незадовільні ергономічні характеристики робочого місця внаслідок нераціонального планування робочого місця, що може призвести до порушень кістково-м'язового апарату, механічних травм та уражень електричним струмом.

е) Негативний вплив на опорно-руховий апарат працюючого в наслідок недостатньої рухливості протягом робочого часу.

ж) Негативний вплив на продуктивність роботи працюючого та його зір, внаслідок неправильного проектування освітлювальної системи або несправності освітлювальних пристроїв.

з) Негативний вплив незадовільних параметрів повітряного середовища робочої зони на здоров'я працюючого, внаслідок її несправності або неправильного проектування системи вентиляції .

и) Негативний дія підвищеного рівня шуму на психологічний стан працюючого, який пов'язаний з застосуванням застарілої периферійної техніки, копіювальної техніки, кондиціонерів, освітлювальних приладів.

к) Ризик пожежі через несправність електрообладнання, недотримання або порушення правил пожежної безпеки обслуговуючим персоналом може призвести до пожеж.

л) Неправильні дії персоналу у надзвичайних ситуаціях.

5.2 Заходи по забезпеченню техніки безпеки

Офісне приміщення, в яких перебувається розробник веб додатку, відносяться до приміщень без підвищеної небезпеки ураження електричним струмом.

Згідно ПУЕ («Правила улаштування електроустановок») виконані наступні заходів з електробезпеки:

а) Конструктивні заходи.

Забезпечує захист від випадкового контакту з струмоведучими частинами, використовуючи ізоляцію та захисні оболонки.

Струмоведучі частини повинні бути повністю покриті твердою ізоляцією, яка може бути усунена тільки шляхом руйнування. Ізоляція струмоведучих частин електрообладнання повинна відповідати вимогам стандартів, згідно з якими це електрообладнання виконане.

б) Схемно-конструктивні заходи.

Призначений для захисту від ураження електричним струмом при дотику до металевих оболонок, які можуть опинитися під напругою в результаті нещасних випадків.

в) Технічні заходи.

Враховуючи, що електроживлення електрообладнання приміщення з ПК здійснюється від мережі змінного струму напругою 220 В і частотою 50 Гц. Відповідно до вимог «ПУЕ», електрообладнання в приміщенні з ПК, характеризується як електроустановки до 1000 В, тому згідно вимог глави 1.7 «ПУЕ», величина опору контуру захисного заземлення електрообладнання приміщення з ПК у будь-яку пору року не перевищує – 4 Ом.

1) Виробниче освітлення.

При проектуванні або реконструкції виробничих приміщень (відділень) приймаються заходи щодо встановлення окремих видів і систем освітлення (відповідно до ДБН В.2.5-28-2006). Для систем електричного освітлення підбираються типи ламп, джерела живлення, напруга освітлювальної мережі, освітлювачів (вказується використання останніх).

Згідно галузевим нормам передбачається аварійне та ремонтне освітлення (називаються типи освітлювачів, ламп, їх виконання, номінальна напруга).

2) Захист від вібрацій та виробничого шуму.

Існують такі способи боротьби з вібрацією та шумом механічного походження:

- зменшення шуму та вібрації безпосередньо в джерелах їх виникнення, за допомогою обладнання, що не утворює шуму, замінюючи ударні технологічні процеси на без ударні, використовуючи деталі із матеріалів з високим коефіцієнтом внутрішнього тертя (пластмаса, гума, деревина та ін), підшипники ковзання замість кочення, косо зубі та шевронні зубчасті передачі замість прямозубих, проводячи своєчасне обслуговування та ремонт елементів, що створюють шум та ін.;

- зменшення шуму та вібрації на шляхах їх розповсюдження заходами звуко- та віброізоляції, а також вібро- та звукопоглинання;

- зменшення шкідливої дії шуму та вібрації, застосовуючи індивідуальні засоби захисту та запроваджуючи раціональні режими праці та відпочинку

3) Основні заходи щодо нормалізації повітря робочої зони:

- раціональні архітектурно-планувальні рішення виробничих будинків;
- захист від джерел теплового випромінювання (екранування, теплоізоляція);
- застосування вентиляційних систем для видалення шкідливих речовин і тепла;
- кондиціонування та опалення повітря;
- оптимальний режим праці і відпочинку;
- навчання безпечним методам праці;
- медичні огляди та ін;

4) Захист від електромагнітних полів

В залежності від характеристики джерел електромагнітних полів обираються і розроблюються заходи захисту персоналу (захист часом, відстанню, екранами, засобами індивідуального захисту). Виконується оцінка прийнятих заходів і вказуються засоби контролю ЕМВ.

г) Організаційні заходи

Для безпечного проведення робіт необхідно виконувати наступні організаційні заходи.

- 1) Призначити працівників, відповідальних за безпечне проведення робіт.
- 2) Видати наряд або розпорядження.
- 3) Видати дозвіл на підготовку робочих місць.
- 4) Підготувати робочі місця.
- 5) Здійснити допуск до робіт.
- 6) Організувати нагляд при виконанні роботи.
- 7) Організувати перехід на інше робоче місце.
- 8) Оформити перерви в роботі та її закінчення.

Для запобігання статистичного навантаження при користуванні ПК рекомендовано використовувати 10 хвилині перерви в роботі кожні дві

години. Синдром зап'ястного каналу, або тунельний синдром зап'ястя, який може бути наслідком хронічної травми, трапляється у людей внаслідок довготривалої роботи з мишею: постійні напруга і здавлювання приводить до мікротравм, здавлювання нерва прилеглими оточуючими тканинами, через що виникає набряк.

Щоб тунельний синдром вас не турбував, потрібно дотримуватися кількох правил організації робочого місця.

- 1) Оптимальна висота клавіатури від підлоги – 65-75 см.
- 2) Наявність ергономічних і зручних особисто для вас миші і клавіатури.
- 3) Можливість регулювання висоти і нахилу клавіатури (відстань від поверхні стола до середини клавіатури – не більше 30 мм, кут підйому клавіатури – від 2° до 15°).
- 4) Наявність у клавіатури підставки для рук.
- 5) Наявність килимка для миші з захистом від тунельного синдрому (спеціальний виступ забезпечує правильне положення кисті).
- 6) Наявність стільця або крісла з підлокітниками.

Організація робочого місця.

- 1) Правильне розміщення робочого місця у виробничому приміщенні.
- 2) Вибір ергономічно обґрунтованого робочого положення, виробничих меблів з урахуванням антропометричних характеристик людини.
- 3) Раціональне компонування обладнання на робочих місцях.
- 4) Врахування характеру та особливостей трудової діяльності.

Планування розміщення комп'ютеризованих робочих місць у приміщенні проводимо із врахуванням наступних вимог.

- 1) Робочі місця з ВДТ розміщуються на відстані не менше 1м від стіни зі світловими прорізами.
- 2) Відстань між бічними поверхнями ВДТ має бути не менше за 1,2м.

3) Відстань між тильною поверхнею одного ВДТ та екраном іншого не повинна бути меншою за 2,5 м.

4) Прохід між рядами робочих місць має бути не меншим 1м.

Розрахунок захисного заземлення.

1. Вхідні данні

$$U = 330 \text{ В}$$

$L = 8 \text{ м}$ – довжина приміщення;

$B = 4 \text{ м}$ – ширина приміщення;

$R_e = 12 \text{ Ом}$ – опір розтіканню струму природних заземлювачів;

$l_b = 5 \text{ м}$ – довжина вертикального стрижня;

$d = 0,01 \text{ м}$ – діаметр перерізу стрижня;

$b_c = 0,028 \text{ м}$ – ширина смуги;

$R_{\text{вим}} = 94 \text{ Ом} \cdot \text{м}$ – питомий опір ґрунту;

$R_3 = 4 \text{ Ом}$ – опір заземлювального пристрою;

Кліматична зона – IV;

Склад ґрунту – однорідний;

Вологість ґрунту – нормальна;

2. Розрахунок опору штучного заземлювача R_u .

$$R_u = \frac{R_e R_3}{R_e - R_3} = \frac{12 \cdot 4}{12 - 4} = 6 \text{ Ом}$$

3. Розрахунок питомого опору ґрунту, для вертикальних і горизонтальних електродів, за формулою:

$$\rho_{\Gamma(B)} = \rho_{\text{вим}} - \psi_{\Gamma(B)}$$

$$\psi_{\Gamma} = 1.4 \quad \psi_B = 1.1$$

$$\rho_B = 94 \cdot 1.1 = 103.4 \text{ Ом}$$

$$\rho_{\Gamma} = 94 \cdot 1.4 = 131.6 \text{ Ом}$$

4. Обчислити опір розтікання струму одиночного вертикального заземлювача, за формулою:

$$R_B = \frac{\rho_B}{2 \cdot \pi \cdot l_B} \cdot \left(\ln \frac{2 \cdot l_B}{d} + \frac{1}{2} \cdot \ln \frac{4 \cdot t_B + l_B}{5 \cdot t_B + l_B} \right)$$

$$= \frac{103.4}{2 \cdot \pi \cdot 5} \cdot \left(\ln \frac{2 \cdot 5}{0.01} + \frac{1}{2} \cdot \ln \frac{4 \cdot 3.3 + 5}{5 \cdot 3.3 + 5} \right) = 22.5225 \text{ Ом}$$

$$t_B = 0.8 + \frac{1}{2} \cdot l_B = 0.8 + \frac{1}{2} \cdot 5 = 3.3 \text{ м}$$

5. Розрахувати наближену (мінімальну) кількість вертикальних стрижнів.

$$n' = \frac{R_B}{R_u} = \frac{22.5225}{6} = 4 \text{ шт}$$

$$n = \frac{P}{a} = \frac{24}{5} = 4 \text{ шт}$$

$$P = 2 \cdot (L + B) = 2 \cdot (8 + 4) = 24 \text{ м}$$

$$a = k \cdot l_B = 1 \cdot 5 = 5 \text{ м}$$

6. Визначити довжину горизонтальної смуги при конфігурації групового заземлювача – контур:

$$l_r = 1.05 \cdot a \cdot n = 1.05 \cdot 5 \cdot 4 = 21 \text{ м}$$

7. Обчислити опір розтікання струму горизонтальної з'єднуючої смуги, за формулою:

$$R_r = \frac{\rho_r}{2 \cdot \pi \cdot l_r} \cdot \ln \frac{2 \cdot l_r^2}{b_c \cdot t_r} = \frac{131.6}{2 \cdot \pi \cdot 21} \cdot \ln \frac{2 \cdot 21^2}{0.028 \cdot 0.814} = 10.54 \text{ Ом}$$

8. Розрахувати еквівалентний опір розтіканню струму групового заземлювача, за формулою:

$$R_{гр} = \frac{R_B \cdot R_r}{R_B \cdot R_r + R_r \cdot R_B \cdot n} = \frac{22.5225 \cdot 10.54}{22.5225 \cdot 0.45 + 10.54 \cdot 0.69 \cdot 4} = 6 \text{ Ом}$$

5.3 Заходи щодо виробничої санітарії та гігієни праці

Згідно ДБН В.2.5-28-2006 «Інженерне обладнання будинків і споруд. Природне і штучне освітлення» у приміщеннях використовується природне та штучне освітлення. Природне освітлення здійснено через світлові прорізи, які забезпечують коефіцієнт природної освітленості (КПО) не нижче 1,5%. Для

захисту від прямих сонячних променів, які створюють прямі та відбиті відблиски на поверхні екранів і клавіатури, використовуються сонцезахисні пристрої, на вікнах встановлені жалюзі або штори. Штучне освітлення в приміщенні здійснено системою загального рівномірного освітлення. Значення освітленості на поверхні робочого столу в зоні розміщення документів становить 300-500 лк. Як джерела штучного освітлення в приміщенні застосовуються люмінесцентні лампи типу ЛБ.

Показники мікроклімату в офісних приміщеннях відповідають встановленим санітарно-гігієнічним вимогам ДСН 3.3.6-042-99 «Санітарні норми мікроклімату виробничих приміщень», ГОСТ 12.1.005-88 (1991) «ССБТ. Загальні санітарно-гігієнічні вимоги до повітря робочої зони» і ГН 2152-80 «Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень».

При виборі метеорологічних умов враховуються:

- пори року – *холодний період* із середньодобовою температурою зовнішнього повітря нижче $+10^{\circ}\text{C}$; *теплый період* із температурою $+10^{\circ}\text{C}$ і вище;

- категорія фізичного навантаження людини.

Роботи в офісних приміщеннях, належать до категорії Іб - легка робота, тому встановлені наступні оптимальні значення параметрів мікроклімату:

- у холодний період року: температура $21-23^{\circ}\text{C}$; відносна вологість: 40-60%; швидкість переміщення повітря: 0,1 м/с;

- у теплий період року: температура $22-24^{\circ}\text{C}$; відносна вологість: 40-60%; швидкість переміщення повітря: 0,2 м/с.

Оптимальні рівні позитивних (n^{+}) і негативних (n^{-}) іонів у повітрі приміщення з ПК нормовані згідно ГН 2152-80 «Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень» і становлять: $n^{+} = 1500-30000$ (шт. на 1cm^3); $n^{-} = 3000-5000$ (шт. на 1cm^3). Підтримка оптимального рівня легких позитивних і негативних аероіонів у

повітрі на робочих місцях забезпечено за допомогою біполярних коронних аероіонізаторів.

Дотримання вимог цих документів досягається оснащенням приміщень і транспортних засобів пристроями кондиціонування і вентиляції, дезодорації повітря, опалювання.

Рівні звукового тиску в октавних смугах частот, рівні звуку та еквівалентні рівні звуку на робочих місцях у приміщення нормуються згідно ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» та ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку».

Зниження рівня шуму в приміщенні здійснено за допомогою:

- використання більш сучасного обладнання;
- розташування принтерів та різноманітного устаткування колективного користування на значній відстані від більшості робочих місць працівників;
- переведення жорсткого диска в режим сну (Standby), якщо комп'ютер не працює протягом визначеного часу;
- використання блоків живлення ПК з вентиляторами на гумових підвісках.

Приміщення з ПК повинні мати природне та штучне освітлення. При незадовільному освітленні знижується продуктивність праці користувачів ПК, можлива поява короткозорості, швидка стомлюваність.

Природне освітлення має здійснюватися через вікна, орієнтовані переважно на північ або північний схід і забезпечувати коефіцієнт природної освітленості ϵ не нижче 1,5%. Для захисту від прямих сонячних променів, які створюють прямі та відбиті відблиски з поверхні екранів і клавіатури, повинні бути передбачені сонцезахисні пристрої, вікна повинні мати жалюзі або штори.

Штучне освітлення в приміщеннях з робочими місцями, обладнаними ПК, має здійснюватися системою загального рівномірного освітлення. Як

джерела штучного освітлення мають застосовуватись переважно люмінесцентні лампи.

Захистом від прямих відблисків повинно бути зниження яскравості видимої частини джерел світла шляхом застосування спеціальних розсіювачів, відбивачів та інших світлозахисних пристроїв, а також правильне розміщення робочих місць відносно джерел світла.

Для забезпечення нормованих значень освітленості у приміщеннях з ПК слід чистити шибки і світильники не менше двох разів на рік і вчасно замінювати лампи, що перегоріли.

10 найважливіших гігієнічних вимог при роботі з комп'ютером:

- покладіть комп'ютер або його монітор до вікна боком, щоб світло на нього падало зліва;
- при організації та обладнанні робочого місця купуйте меблі відповідно до зросту користувача комп'ютера;
- щодня перед початком роботи обов'язково прибирайте пил на робочому місці;
- перед початком і після закінчення роботи, а також в обідню перерву проводите аерацію (провітрювання) приміщення, де працює комп'ютер;
- щодня проводьте вологе прибирання в приміщенні, де працює комп'ютер;
- при безперервній роботі з кожним комп'ютером 2 години робіть перерву на 15 хвилин для відпочинку і виконання комплексу фізкультурно-оздоровчих вправ;
- слідкуйте за зовнішнім освітленням монітора комп'ютера і навколишнього простору, який мав би бути менше, ніж 5:1;
- при роботі з комп'ютером відстань від очей користувача до монітора складати 600-700 мм, але не менше 500 мм;
- слідкуйте за поставою: спина повинні бути пряма, руки в ліктях повинні бути зігнуті під прямим кутом;
- регулярно проходите профілактичний лікарський огляд.

5.4 Заходи безпеки у надзвичайних ситуаціях

5.4.1 Заходи з пожежної безпеки

Пожежна безпека – стан захищеності об’єкта, який зменшує ймовірність виникнення та розвитку пожежі, виключає вплив небезпечних факторів на людей і забезпечує захист матеріальних цінностей.

Для підвищення рівня пожежної безпеки на об’єкті проводять пожежну профілактику, яка включає в себе комплекс технічних та організаційних заходів, цілю яких є забезпечення безпеки людей, створення умов для успішного гасіння пожежі, а також обмеження її поширення.

Щоб уникнути пожежі на початковій стадії її розвитку на об’єктах використовуються первинні засоби пожежогасіння. До них відносяться:

- вогнегасники;
- пожежний інвентар (покривала з негорючого теплоізоляційного полотна, ящики з піском, пожежні відра, совкові лопати, ломы, сокири тощо);
- системи автоматичного пожежогасіння.

Первинні засоби пожежогасіння, в залежності від категорії приміщень, можуть розташовуватись як окремо, так і в складі пожежних щитів.

Залежно від агрегатного стану й особливостей горіння різних горючих речовин і матеріалів пожежі за ДБН В.1.1.7-2002 «Пожежна безпека об’єктів будівництва» поділяються на відповідні класи. В офісному приміщенні знаходиться дерев’яна мебел, електронна апаратура, бумажні носії інформації.

Клас пожежі у офісному приміщенні (згідно із ДБН В.1.1.7-2002 «Захист від пожежі. Пожежна безпека об’єктів будівництва») – пожежі твердих речовин, переважно органічного походження, горіння яких супроводжується тлінням (деревина, пластмаси, папір) - визначається як клас А. Категорія приміщення (згідно із НАПБ Б.03.002-2007 «Норми визначення категорій приміщень, будинків та зовнішніх установок за вибух пожежною та пожежною небезпекою») – визначається як категорії Д.

Визначення типу первинних засобів пожежегасіння (згідно із ДБН В.1.1.7-2002 «Захист від пожежі. Пожежна безпека об'єктів будівництва») – для офісного приміщення слід застосовувати два порошкових вогнегасниками типу ВП–5 (НАПБ Б.03.002-2007 «Норми визначення категорій приміщень, будинків та зовнішніх установок за вибух пожежною та пожежною небезпекою»).

ДБН В.1.1.7-2002 «Пожежна безпека об'єктів будівництва». Крім цього офісні приміщення повинні бути обладнані автоматичними пожежними сповіщувачами, що реагують на підвищення температури, дим, полум'я.

В приміщеннях встановити наступний протипожежний режим, яким передбачено:

- порядок паління цигарок. Паління в будинках і приміщеннях заборонено. Для паління на територіях об'єктів обладнані спеціальні місця, які позначені знаками пожежної безпеки або написами - "Місце для паління" та облаштовані урнами для недопалків;

- порядок користування електронагрівальними приладами. Приготування кип'ятку, розігрівання та приготування їжі здійснюється в спеціально обладнаних для цих цілей місцях із застосуванням електрочайників з автоматичними пристроями відключення електронагрівальних елементів;

- порядок роботи з електроприладами. Забороняється залишати без нагляду увімкнені в електромережу електроприлади;

Меблі та обладнання мають розміщатися так, щоб забезпечити евакуаційний прохід до виходу з приміщення (заввишки не менше 1,0 м). Евакуаційні шляхи та виходи слід завжди утримувати вільними, нічим не захащувати.

Електричні мережі та електрообладнання слід експлуатувати тільки у справному стані з урахуванням інструкцій і рекомендацій виробників. У разі пошкодження електричних мереж, вимикачів, розеток та інших електроприладів слід негайно вимкнути їх і прийняти необхідні заходи щодо приведення їх у безпечний стан.

Документи, папір та інші горючі матеріали слід зберігати на відстані від електрощитів, електрокабелів, проводів – не менше 1,0 м, від світильника – 0,5 м від приладів опалення – 0,25 м.

Протипожежне обладнання повинно підтримуватися у справному стані. Всі офісні працівники зобов'язані вміти користуватися наявними вогнегасниками, іншими первинними засобами пожежогасіння та внутрішніми пожежними кранами, знати місця їх розташування. Відстань від найвіддаленішого місця офісу до вогнегасника не повинна перевищувати 20 м.

У загальному випадку розрахунок кількості вуглекислотних вогнегасників повинен ґрунтуватися на простій формулі - один кілограм гасильної речовини на 25 м² площі об'єкта. Наш офіс має приблизно 70 м² тому нам необхідно буде розмістити 3 вуглекислотних вогнегасника в приміщенні.

У службових приміщеннях забороняється [15]:

- улаштовувати тимчасові електромережі, застосовувати саморобні некалібровані плавкі вставки в запобіжниках і саморобні подовжувачі, які не відповідають вимогам правил улаштування електроустановок, прокладати електричні проводи безпосередньо по горючій основі, експлуатувати світильники зі знятими ковпаками (розсіювачами);

- використовувати вимикачі та штепсельні розетки для розвішування на них одягу або інших предметів, обгортати електролампи й світильники папером, заклеювати ділянки електропроводки горючою тканиною, папером;

- захаращувати підходи до засобів пожежогасіння, використовувати пожежні крани, рукава й пожежний інвентар не за призначенням;

- використовувати для зберігання документів, різних матеріалів, предметів та інвентарю шафи (інші) інженерних комунікацій;

5.4.2 Заходи по забезпеченню організації навчання працюючого та непрацюючого населення діям у надзвичайних ситуаціях [8]

а) Навчання населення здійснюється.

- 1) за місцем роботи (працюче населення);
- 2) за місцем навчання (діти дошкільного віку, учні та студенти);
- 3) за місцем проживання (непрацюче населення).

б) Організація навчання населення покладається.

1) Для навчання зайнятого і незайнятого населення покладено на центральний орган виконавчої влади, який забезпечує формування та реалізацію державної політики у сфері цивільного захисту, який готує і затверджує відповідні програми з підготовки населення до таких дій та організаційно-методичні вказівки.

2) Навчання дітей дошкільного віку, студентів та учнів покладено на центральний орган виконавчої влади, який забезпечує формування та реалізацію державної політики в сфері освіти і науки, який розробляє і затверджує навчальні програми з надання першої медичної допомоги та з вивченням заходів безпеки, а також форми впливу небезпек, викликаних надзвичайними ситуаціями

в) Навчання населення.

- 1) Навчання безпосередньо на об'єктах виробничої діяльності.
- 2) Навчання за межами підприємств, установ та організацій керівного складу і фахівців з питань цивільного захисту та пожежної безпеки.
- 3) Практичної підготовки під час проведення спеціальних об'єктових тренувань та навчань з питань цивільного захисту.
- 4) Навчання під час здобуття освітнього рівня у навчальних закладах.
- 5) Самостійного вивчення інформації про дії в умовах надзвичайних ситуацій.

г) Навчання персоналу діям в надзвичайних ситуаціях є обов'язковим і проводиться на робочому місці протягом робочого дня за дорученням роботодавця.

1) За програмами підготовки співробітників до дій у надзвичайних ситуаціях.

2) Проведення інформаційних інструктажів в компанії та організації, а також перевірка знань з цивільного захисту, пожежної безпеки та безпеки людини.

3) Проведення практичних занять з питань цивільного захисту.

4) Програми підготовки працівників до дій у надзвичайних ситуаціях поділяються на:

- загальну підготовку працівників підприємств та організацій;

- спеціальну підготовку працівників, що входять до складу спеціалізованих служб;

- додаткову підготовку з техногенної безпеки працівників які працюють на об'єкті підвищеної небезпеки;

- пожежно-технічну мінімуму для працівників які працюють на об'єктах підвищеної пожежної небезпеки;

- прискорену підготовку працівників до дій в особливий період.

5) Інструктажі та перевірка знань з питань цивільного захисту, пожежної та техногенної безпеки зазвичай проводяться особами, які пройшли навчання за програмою навчальних курсів спеціальної підготовки.

6) До заходів практичної підготовки з питань цивільного захисту відносяться:

- спеціально об'єктні навчання (один раз на три роки);

- спеціальні об'єктні тренування спеціалізованих служб цивільного захисту та населення(не менше одного разу на рік);

д) Навчання учнів, студентів та дітей дошкільного віку проводять під час навчально-виховного процесу, а також шляхом:

- 1) проведення шкільних, районних (міських), обласних та всеукраїнських змагань з безпеки життєдіяльності;
- 2) проведення навчально-тренувальних зборів і польових таборів;
- 3) участі команд - переможниць у заходах міжнародного рівня з цих питань.

е) Навчання непрацюючого населення (населення не зайнятого у сферах виробництва та обслуговування) діям у надзвичайних ситуаціях.

1) Проведення інформаційних заходів, організованих місцевими органами виконавчої влади та органами місцевого самоврядування за місцем проживання, шляхом розміщення постійних заголовків в засобах масової інформації, зокрема в пресі, а також з використанням інформаційних і комунікаційних технологій, аудіовізуальних та інтерактивних засобів масової інформації і соціальної реклами.

2) Самостійного вивчення жителями регіону загальної програми навчання населення діям у надзвичайних ситуаціях, пам'яток та іншого інформаційно-довідкового матеріалу з питань цивільного захисту. З метою задоволення потреби у самостійному опрацюванні загальної програми навчання населення в умовах надзвичайних ситуацій місцеві органи виконавчої влади та органи місцевого самоврядування за методичного супроводження територіальних курсів, навчально-методичних центрів цивільного захисту та безпеки життєдіяльності видають навчальні посібники, брошури, розповсюджують інформаційні матеріали, буклети тощо.

3) Отримання необхідної інформації про поведінку в умовах надзвичайних ситуацій на консультаційних пунктах з питань цивільного захисту при органах місцевого самоврядування за місцем проживання (ЖЕО, ОСББ тощо)що передбачає:

- донесення інформації до населення про методи реагування під час виникнення надзвичайних ситуацій;

- створення в консультаційних пунктах умов для оволодіння громадянами навичками користування найбільш поширеними засобами захисту і надання першої допомоги;

- вивчення правил пожежної безпеки у побуті та громадських місцях;

4) Отримання від органів місцевого самоврядування, через засоби масової інформації інших повідомлень про надзвичайні ситуації, у зоні яких можливо ураження в яких можуть опинитися місця проживання непрацюючих громадян, а також розповісти про способи захисту від впливу небезпечних факторів, викликані надзвичайними ситуаціями.

ВИСНОВКИ

В ході роботи над проектом було розроблено захищений веб месенджер для забезпечення можливості спілкування абонентів по захищеному каналу в процесі роботи з сайтом. Для цього було використано технології такі як: node.js, React, Redux.

Також було порівняно три криптографічних бібліотеки: PidCrypto, Crypto, AES-JS за чотирма критеріями швидкість, простота, розмір на жорсткому диску, функціональність. Зробивши їх порівняльну характеристику та розглянувши які функції шифрування та хешування реалізовані в даних бібліотеках, було обрано бібліотеку Crypto, так як вона має велике різноманіття функцій для шифрування та хешування даних, а також зручний API для інтеграції в node.js. В ході роботи були використані декілька класів з цієї бібліотеки:

- Hash (для хешування паролів);
- DiffieHellman (для генерації ключів).

Оскільки в 2014 році в Україні було прийнято криптографічний стандарт блокового симетричного шифрування ДСТУ 7624:2014 (“Калина”), розроблено власну бібліотеку його реалізації, яку в подальшому можна буде використовувати в інших національних веб проектах.

Розроблене програмне забезпечення пройшло тестування та відповідає всім поставленим вимогам.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бази даних. [Електронний ресурс]. – Режим доступу: <https://siblec.ru/informatika-i-vychislitel'naya-tekhnika/bazy-dannykh> [дата звернення: 15.12.2020];
2. Інструкція по Node.js частина 1: загальні відомості і початок роботи [Електронний ресурс]. – Режим доступу: <https://habr.com/ru/company/ruvds/blog/422893/> [дата звернення: 15.12.2020];
3. Passport.js. [Електронний ресурс]. – Режим доступу: <http://www.passportjs.org/docs/> [дата звернення: 15.12.2020];
4. Чим PostgreSQL краще інших SQL баз даних з відкритим вихідним кодом. Частина 1. [Електронний ресурс]. – Режим доступу: <https://habr.com/ru/post/282764/> [дата звернення: 15.12.2020];
5. SQL за 20 хвилин. [Електронний ресурс]. – Режим доступу: <https://proglib.io/p/sql-for-20-minutes> [дата звернення: 15.12.2020];
6. Моделювання блокового шифру «Калина» [Електронний ресурс]. – Режим доступу: <https://openarchive.nure.ua/bitstream/document/4806/1/201-207.pdf> [дата звернення: 15.12.2020];
7. HTML, CSS, PHP, JavaScript, SQL – что и зачем? [Електронний ресурс]. – Режим доступу: <http://codeharmony.ru/materials/125> [дата звернення: 15.12.2020];
8. Про затвердження Порядку здійснення навчання населення діям у надзвичайних ситуаціях. [Електронний ресурс]. – Режим доступу: <https://zakon.rada.gov.ua/laws/show/444-2013-%D0%BF#Text> [дата звернення: 15.12.2020];
9. Методична розробка. Тема: Гасіння пожеж. [Електронний ресурс]. – Режим доступу: https://nung.edu.ua/files/attachments/2_3-gasinnya-pozhezh-kurylovych-sp.pdf [дата звернення: 15.12.2020];

- 10.React. JavaScript-бібліотека для створення користувацьких інтерфейсів. [Електронний ресурс]. – Режим доступу: <https://ru.reactjs.org/> [дата звернення: 15.12.2020];
- 11.Angular. [Електронний ресурс]. – Режим доступу: <https://angular.io/> [дата звернення: 15.12.2020];
- 12.PostgreSQL: Найдосконаліша у світі реляційна база даних з відкритим кодом. [Електронний ресурс]. - Режим доступу: <https://www.postgresql.org/> [дата звернення: 15.12.2020];
- 13.SQL запити швидко. Частина 1. [Електронний ресурс]. - <https://habr.com/ru/post/480838/> [дата звернення: 15.12.2020];
- 14.Про затвердження порядку здійснення навчання населення діям у надзвичайних ситуаціях. [Електронний ресурс]. - Режим доступу: <https://zakon.rada.gov.ua/laws/show/444-2013-%D0%BF#Text> [дата звернення: 15.12.2020];
- 15.Про заходи пожежної та техногенної безпеки для службових (адміністративних) приміщень. [Електронний ресурс]. - Режим доступу: <http://pnpu.edu.ua/wp-content/uploads/2020/03/instrukciya-%E2%84%962-pro-zahodi-pozhezhno%D1%97-ta-tehnogenno%D1%97-bezpeki-v-administrativnih-primishhennyah-pnpu-imeni-v.-g.-korolenka.pdf> [дата звернення: 15.12.2020];
- 16.MySQL. [Електронний ресурс]. - Режим доступу: <https://www.mysql.com/> [дата звернення: 15.12.2020];

ДОДАТОК А

КОД ДЛЯ СТВОРЕННЯ СТОРІНКИ РЕЄСТРАЦІЇ ТА АВТОРИЗАЦІЇ

```
import React, { useEffect, useState } from "react";
import { connect } from "react-redux";
import Selector from "../../Selector";
import BirthdaySelector from "../../Selector/BirthdaySelector";
import { fillRegistration, registration } from "../../Redux/registrationReducer";
import { fillDate } from "../../Redux/dateReducer";
import { setModal } from "../../Redux/rulesReducer";
import { fillValidator } from "../../Redux/validatorReducer";
import { toast } from "react-toastify";
const selectorsData = [
  { label: `Дата народження`, type: `date` },
  { label: `Освіта`, type: `education` },
  { label: `Діти`, type: `children` },
  { label: `Регіон`, type: `region` },
  { label: `Вулиця`, type: `district` },
];
const Registration = ({data, show, date, validator, registration, fillRegistration, fillDate,
setModal, fillValidator }) => {
  const [checkedRule, setCheckedRule] = useState(false);
  const [disabled, setDisabled] = useState(false);
  const showRegistrationSecondPart = show ? `registration-second-part` : `hidden`;
  const checkEmail = (name, value) => {
    if (value && /^[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}$/i.test(value)) {
      fillValidator(`${name}Error`, "Invalid email address!");
      setDisabled(true);
    } else {
      fillValidator(`${name}Error`, "");
      setDisabled(false);
    }
  };
  const onChange = (e) => {
    const { id, value } = e.target;
    fillRegistration(id, value);
    if (id === "email") checkEmail(id, value);
  };
  const onClick = () => setModal(true);
  useEffect(() => {
    !data.login || !data.email || !data.password || !checkedRule ? setDisabled(true)
      : setDisabled(false);
  }, [data.login, data.email, data.password, checkedRule]);
  const finishRegistration = () => {
    registration(data, toast);
    setCheckedRule(false);
  };
  const emailError = validator.emailError ? (
    <div className="registration-second-part-body-second-register-error">
      {validator.emailError}
```

```

    </div>): null;
const selectors = selectorsData.map(({ label, type }) => (
  <div className="registration-second-part-body-second-register" key={type}>
    <label>
      {label}
      {type === `date` ? (
        <BirthdaySelector date={date} fill={fillRegistration} fillDate={fillDate}/>
      ) : ( <Selector type={type} value={data[type]} fill={fillRegistration} />)}
    </label>
  </div>));
return (
  <div className={showRegistrationSecondPart}>
    <div className="registration-second-part-body">
      <div className="registration-second-part-body-second-register">
        <label>
          Логін
          <input id="login" value={data.login} className="registration-second-part-body-second-register-input-area" onChange={onChange} />
        </label>
      </div>
      <div className="registration-second-part-body-second-register">
        <label>
          Почта
          <input id="email" value={data.email} className="registration-second-part-body-second-register-input-area" onChange={onChange} />
          {emailError}
        </label>
      </div>
      <div className="registration-second-part-body-second-register">
        <label>
          Пароль
          <input id="password" type="password" value={data.password}
            className="registration-second-part-body-second-register-input-area"
            onChange={onChange} />
        </label>
      </div>
      {selectors}
      <div className="registration-second-part-body-second-register">
        <input type="checkbox" id="condiciones" checked={checkedRule}
          onChange={(e) => setCheckedRule(e.target.checked)} />
        <label htmlFor="condiciones">Приймаю</label>
        <span onClick={onClick}>умови використання</span>
      </div>
      <div className="registration-second-part-body-second-register">
        <button disabled={disabled ? "disabled" : null} className="registration-second-part-body-second-register-submit" onClick={finishRegistration}> Завершити </button>
      </div>
    </div>
  </div>
);
};

```

ДОДАТОК Б

КОД ДЛЯ СТВОРЕННЯ СТОРІНКИ ПРОФІЛЮ

```
import { connect } from "react-redux";
import Album from "../Album";
import ProfileTitle from "../ProfileTitle";
import Interests from "../Interests";
import { getUser } from "../../Redux/dataReducer";
import { getInterests, getUserInterests } from "../../Redux/interestsReducer";
import Addition from "../Addition";
import Admin from "../Admin";
import { getAlbum } from "../../Redux/albumReducer";
import { toast } from "react-toastify";
const userData = [
  { titles: ["Громадянський стан:", "Діти:"], types: ["family", "children"] },
  { titles: ["Освіта:", "Робота:"], types: ["education", "work"] },
  { titles: ["Релігія:", "Задіак:"], types: ["religion", "zodiac"] },
  { titles: ["Зріст:", "Вага:"], types: ["height", "weight"] },
  { titles: ["Статура:", "Волосся:"], types: ["contexture", "hair"] },
  { titles: ["Очі:", "Курець:"], types: ["eyes", "smoker"] },
  { titles: ["Татуювання:", "Пірсинг:"], types: ["tattoo", "piercings"] },
];
const Data = ({ data, match, userId, role, interestsData, album, getAlbum, getUser,
getUserInterests, getInterests }) => {
  const admin = "admin";
  const moderator = "moderator";
  const setData = () => {
    getUser(match.params.id);
    getUserInterests(match.params.id);
    getInterests(toast);
    getAlbum(match.params.id);
  };
  useEffect(setData, [match.params.id]);

  const adminPage =
    userId === data.id ? null : role === admin || role === moderator ? (
      <Admin />
    ) : null;
  const profileTitle = data.id === userId ? null : <ProfileTitle data={data} />;
  const addition =
    data.id === userId ? null : <Addition id={match.params.id} />;
  const pictures = data.id === userId ? null : <Album album={album} />;
  return (
    <div className="profile">
      <div className="profile-body">
        <div className="profile-body-half">
          <div className="profile-body-half-avatar">
            <img src={data.profilePhoto} alt="avatar" />
          </div>
          {addition}
        </div>
      </div>
    </div>
  );
}
```

```

    {pictures}
  </div>
  <div className="profile-body-half">
    <div>
      <h3 className="profile-body-half-name-age">
        {data.login}, {data.age} років
      </h3>
    </div>
    <div>
      <p className="profile-body-half-small-describe">
        Привіт, я {data.login}, я живу на вулиці {data.district}.
        Я шукаю {data.type}
      </p>
    </div>
    <div className="profile-body-half-data">
      <h3>Мої інтереси</h3>
    </div>
    <Interests interestsData={interestsData} />
    {profileTitle}
    <div className="profile-body-half-data">
      <h3>Більше про {data.login}</h3>
      <table className="profile-body-half-data-table">
        <tbody>
          {userData.map(({ types, titles }, index) => (
            <tr key={index}>
              {titles.map((title, index) => (
                <td key={types[index]}>
                  <span>{title}</span>
                  {types[index] ? data[types[index]] : ``}
                </td>
              ))}
            </tr>
          ))}
        </tbody>
      </table>
    </div>
  </div>
  {adminPage}
</div>
);
};

```

```

const mapStateToProps = ({ data, userData: { id, role }, interests, album }) => ({
  data, userId: id, role, interestsData: interests, album,
});
export default connect(mapStateToProps, {
  getAlbum,
  getUser,
  getUserInterests,
  getInterests,
})(Data);

```

ДОДАТОК В

КОД ДЛЯ СТВОРЕННЯ СТОРІНКИ ПОШУКУ

```
import React, { useEffect, useState } from "react";
import { connect } from "react-redux";
import { useHistory } from "react-router-dom";
import Selector from "../Selector";
import Age from "../Slider/Age";
import { findPeople } from "../Redux/filteredReducer";
import {
  fillExtendedSearch,
  getExtendedSearch,
} from "../Redux/extendedSearchReducer";
import Appearance from "../Appearance";
import SocialInfo from "../SocialInfo";
const ExtendedFilter = ({
  fillExtendedSearch,
  extendedSearch,
  getExtendedSearch,
  findPeople,
}) => {
  const [showAppearance, setAppearance] = useState(false);
  const [showSocialInfo, setSocialInfo] = useState(false);
  const history = useHistory();
  useEffect( () => getExtendedSearch(), []);
  const handleChange = (e) => fillExtendedSearch(e.target.id, e.target.checked);
  const onChange = (e) => fillExtendedSearch(e.target.id, e.target.value);
  const handleClick = () => findPeople(extendedSearch, history);
  return (
    <div className="filter">
      <div className="filter-panel">
        <label className="filter-panel-label">ПОШУК</label>
        <div className="filter-panel-settings">
          <div className="filter-panel-settings-slider">
            <label className="filter-panel-settings-slider-label">
              Вікова категорія
            </label>
            <Age
              extendedSearch={true}
              categoryAge={extendedSearch.categoryAge}
              fillSearch={fillExtendedSearch}
            />
          </div>
          <div className="filter-panel-settings-row">
            <div className="filter-panel-settings-row-col">
              <label>
                Регіон
              <Selector
                id="region"
                type="region"
              >
```

```

        value={extendedSearch.region}
        fill={fillExtendedSearch}
      />
    </label>
  </div>
  <div className="filter-panel-settings-row-col">
    <label>
      ВУЛИЦЯ
      <Selector
        id="district"
        type="district"
        value={extendedSearch.district}
        fill={fillExtendedSearch}
      />
    </label>
  </div>
</div>
<div className="filter-panel-settings-row">
  <div className="filter-panel-settings-row-col">
    <label>
      ДІЯЛЬНІСТЬ КОРИСТУВАЧА
      <Selector
        id="userActivity"
        type="userActivity"
        value={extendedSearch.userActivity}
        fill={fillExtendedSearch}
      />
    </label>
  </div>
  <div className="filter-panel-settings-row-col">
    <label>
      ЗАЦІКАВЛЕНИЙ В
      <Selector
        id="type"
        type="type"
        value={extendedSearch.type}
        fill={fillExtendedSearch}
      />
    </label>
  </div>
</div>
<hr />
<div className="filter-panel-settings-row">
  <p>Фото</p>
  <div className="filter-panel-settings-row-checkbox">
    <span>
      <input
        type="checkbox"
        id="withPhoto"
        checked={extendedSearch.withPhoto}
        onChange={handleChange}
      />
    </span>
  </div>

```

```

        <label htmlFor="withPhoto" />
    </span>
</div>
</div>
<hr />
<div className="filter-panel-settings-row">
    <p>Альбом</p>
    <div className="filter-panel-settings-row-checkbox">
        <span>
            <input
                type="checkbox"
                id="withAlbum"
                checked={extendedSearch.withAlbum}
                onChange={handleChange}
            />
            <label htmlFor="withAlbum"/>
        </span>
    </div>
</div>
<hr />
<div className="filter-panel-settings-row">
    <span
        onClick={() => {
            setAppearance(!showAppearance);
        }}
    >
        <h4>
            Зовнішній вигляд <i className="fas fa-chevron-down" />
        </h4>
    </span>
</div>
<div
    className={showAppearance ? "filter-panel-settings-row" : "hidden"}
>
    <Appearance
        data={extendedSearch}
        fillExtendedSearch={fillExtendedSearch}
    />
</div>
<div className="filter-panel-settings-row">
    <span
        onClick={() => {
            setSocialInfo(!showSocialInfo);
        }}
    >
        <h4>
            Соціальна інформація <i className="fas fa-chevron-down" />
        </h4>
    </span>
</div>
<div
    className={showSocialInfo ? "filter-panel-settings-row" : "hidden"}

```

```

    >
    <SocialInfo
      data={extendedSearch}
      fillExtendedSearch={fillExtendedSearch}
    />
  </div>
  <div className="filter-panel-settings-row">
    <input
      id="login"
      type="text"
      placeholder="Ім'я"
      value={extendedSearch.login}
      onChange={onChange}
    />
  </div>
  <div className="filter-panel-settings-row">
    <button
      className="filter-panel-settings-row-btn"
      onClick={handleClick}
    >
      Шукати
    </button>
  </div>
</div>
</div>
</div>
);
};

const mapStateToProps = ({ extendedSearch }) => ({ extendedSearch });
export default connect(mapStateToProps, {
  fillExtendedSearch,
  getExtendedSearch,
  findPeople,
})( ExtendedFilter);

```

ДОДАТОК Г

КОД ДЛЯ СТВОРЕННЯ СТОРІНКИ РЕДАГУВАННЯ

```
import React, { useEffect, useState } from "react";
import { connect } from "react-redux";
import _ from "lodash";
import Photo from "../Photo";
import PersonalInformation from "../PersonalInformation";
import SocialInformation from "../SocialInformation";
import Appearance from "../Appearance";
import Interests from "../Interests";
import { getInterests, getUserInterests, setUserInterests } from
"../../Redux/interestsReducer";
import { post } from "../../API/Requests";
import { toast } from "react-toastify";
import { fillData, getUser } from "../../Redux/dataReducer";
const EditProfile = ({ data, interests, match, getUser, fillData, setUserInterests,
getInterests, getUserInterests }) => {
  const [exitingData, setExitingData] = useState();
  const [exitingInterests, setExitingInterests] = useState();
  const disabled =
    _.isEqual(exitingData, data) &&
    _.isEqual(exitingInterests, interests.userInterests);
  const setData = () => {
    getUser(match.params.id);
    getInterests(toast);
    getUserInterests(match.params.id);
  };
  if (data.id && !exitingData) {
    setExitingData(data);
  } else if (!_.isEmpty(interests.userInterests) && !exitingInterests) {
    setExitingInterests(interests.userInterests);
  }
  useEffect(setData, [match.params.id]);
  const handleUpdate = () => {
    const newData = {
      userData: data,
      interests: interests.userInterests,
    };
    setExitingData(data);
    setExitingInterests(interests.userInterests);
    post(`/user/${match.params.id}/update-profile`, newData).then(
      ({ error, message }) => {
        error ? toast.error(message) : toast(message);
      }
    );
  };
  return (
    <div className="editor">
      <div className="editor-col">
```

```

    <div className="editor-col-panel">
      <span>ФОТО</span>
      <Photo id={match.params.id} />
    </div>
    <div className="editor-col-panel">
      <span>ПЕРСОНАЛЬНА ІНФОРМАЦІЯ</span>
      <PersonalInformation data={data} fillData={fillData} />
    </div>
    <div className="editor-col-panel">
      <span>СОЦІАЛЬНА ІНФОРМАЦІЯ</span>
      <SocialInformation data={data} fillData={fillData} />
    </div>
  </div>
  <div className="editor-col">
    <div className="editor-col-panel">
      <span>ВИД</span>
      <Appearance data={data} fillData={fillData} />
    </div>
    <div className="editor-col-panel">
      <span>ІНТЕРЕСИ</span>
      <Interests
        dataInterests={interests}
        setUserInterests={setUserInterests}
      />
    </div>
    <button
      className="editor-col-accept"
      disabled={disabled}
      onClick={handleUpdate}
    >
      Зберегти
    </button>
  </div>
</div>
);
};
const mapStateToProps = ({ interests, data }) => ({ data, interests });
export default connect(mapStateToProps, { getUser, fillData, setUserInterests,
  getInterests, getUserInterests,})(EditProfile);

```

ДОДАТОК Д

СТВОРЕННЯ СТОРІНКИ ЧАТУ

```
import React, { useEffect, useState } from "react";
import { connect } from "react-redux";
import { fillMessage, deleteChats, clearChat, addMessage, deleteConversation, setUser }
from "../Redux/conversationReducer";
import Messages from "../Messages";
import { online, send, get, checkChat, checkChats } from "../Socket/Socket";

const Chats = ({ conversation, recipientId, setRecipientId, userData, fillMessage,
deleteChats, clearChat, addMessage, deleteConversation, setUser }) => {
  const text = conversation.send.message;
  const [socket, setSocket] = useState(true);
  if (socket) {
    get(addMessage);
    checkChat(clearChat);
    checkChats(deleteConversation);
    setSocket(false);
  }
  useEffect(() => online(userData.id), [userData.id]);
  const handleChange = (e) => {
    const newValue = e.target ? e.target.value : e;
    fillMessage(`message`, newValue);
  };
  const onClick = (id, user) => () => {
    const active = recipientId === id;
    setRecipientId(active ? null : id);
    if (!active) {
      fillMessage("recipientId", id);
      fillMessage("senderId", userData.id);
      setUser(user);
    }
  };
  const sendMessage = () => {
    send(conversation.send, addMessage);
    fillMessage("message", "");
  };
  const handleClick = (id) => () => deleteChats(id);
  const onKeyPress = (e) => e.key === "Enter" && e.preventDefault();
  const onKeyUp = (e) => {
    const cursorPosition = e.target.selectionStart;
    const pressEnter = e.keyCode === 13 || e.keyCode === 10;
    const pressShift = e.shiftKey;
    if (pressEnter && text) {
      if (pressShift) {
        const { value } = e.target;
        const firstPart = value.substring(0, cursorPosition - 1);
        const secondPart = value.substring(cursorPosition - 1, value.length);
        const newValue = `${firstPart}\n${secondPart}`;
      }
    }
  };
}
```

```

        handleChange(newValue);
        e.stopPropagation();
    } else if (!pressShift) {
        sendMessage();
    }
}
};
return conversation.users.map((user) => {
    const showChat = recipientId === user.id;
    const chatStyle = showChat
        ? { background: "#eaf5fb", border: "2px solid #319dd4" }
        : null;
    const messageHeaderStyle = showChat
        ? { borderBottom: "1px solid #b8bfc7" }
        : null;
    const messageBodyStyle = showChat ? null : { display: "none" };
    return (
        <div key={user.login} className="col" style={chatStyle}>
            <span
                className="col-message-header"
                onClick={onClick(user.id, user)}
                style={messageHeaderStyle}
            >
                <div>
                    <span>{user.login}</span>
                    <i className={`fas fa-circle ${user.userActivity}`} />
                </div>
            </span>
            <input type="checkbox" onClick={handleClick(user.id)} />
            <div className="col-image">
                <img src={` ${user.profilePhoto}`} alt={`profile_photo_${user.id}`} />
            </div>
            <div className="col-message-body" style={messageBodyStyle}>
                <Messages conversation={conversation} userData={userData} />
                <textarea
                    value={text}
                    onChange={handleChange}
                    onKeyDown={onKeyPress}
                    onKeyUp={onKeyUp}
                />
                <button disabled={!text} onClick={sendMessage}>
                    Надіслати
                </button>
            </div>
        </div>
    );
});
const mapStateToProps = ({ userData, conversation }) => ({ conversation, userData });
export default connect(mapStateToProps, { fillMessage, deleteChats, clearChat,
addMessage, deleteConversation, setUser })(Chats);

```